**Institute of Information and Communication Technologies – BAS**

**Sofia, BULGARIA**

Department: “Information Processes and Decision Support Systems”

**EMILIANO MANKOLLI**

# OPTIMIZATION METHODS FOR MACHINE LEARNING APPLICATIONS

# DISSERTATION THESIS

**Sofia**

**2023**

**Institute of Information and Communication Technologies – BAS**

**Sofia, BULGARIA**

Department: “Information Processes and Decision Support Systems”

**EMILIANO MANKOLLI**

# OPTIMIZATION METHODS FOR MACHINE LEARNING APPLICATIONS

# DISSERTATION THESIS

for acquiring Educational and Scientific Degree
“DOCTOR”
in Professional field 4.6. “Informatics and Computer Science”,
in Doctoral Program: “Informatics”

**Scientific Advisor:**
Prof. Dr. Vassil Georgiev Guliashki

**Sofia**
**September, 2023**

# Acknowledgments

This thesis is a significant achievement in my career, and I owe it to my family, colleagues, and friends.

I would like to express my appreciation and gratitude to my supervisor, Prof. Dr. Vassil Georgiev Guliashki. He has been a great mentor, providing me with guidance and advice that has helped me grow professionally and apply my industry experience to academic research.

I also want to acknowledge my high-school mathematics teacher, Professor Gjergji Çikopana, who inspired me to pursue mathematics and guided me in the first steps of this journey.

I am grateful to my colleagues for their constant support during these four challenging years, especially to Besmira for her assistance and for being the best reviewer of all my publications.

Last, but not least, I want to thank my family. My wife Enela and our children Eilan, Elynn, and Eloy are the main source of motivation that drives me to give my best every day. I appreciate your unwavering support during my busy career years.

I dedicate this accomplishment to my parents as a way of expressing my gratitude for all their efforts and sacrifices they have made to support me from the beginning.

# Table of Contents

# Dissertation Structure

Mankolli, E. **Optimization Methods For Machine Learning Applications**. Scientific Advisor: Prof. Dr. Vassil Georgiev Guliashki. Department Information Processes and Decision Support Systems. Doctoral Program: “Informatics”. Sofia, 2023. The dissertation, with 128 pages, consists of an introduction, 3 chapters, a conclusion - a summary of the obtained results, 6 contributions, a list of publications on the dissertation, declaration of originality of the results, and a bibliography.

The main goal of the dissertation is:

> To develop and refine innovative machine learning (ML) and natural language processing (NLP) algorithms to revolutionize the recruitment industry by focusing on enhancing the efficiency and effectiveness of the recruitment process. This goal includes the development of new algorithms for optimized candidate screening, advanced job-candidate matching, and streamlined initial filtration processes, while maintaining computational efficiency and algorithmic simplicity. The aim is to provide a cutting-edge tool for recruitment companies to transform industry practices by leveraging semantic analysis, information retrieval, and recommendation systems.

**Keywords**

machine learning, natural language processing, algorithms, embedding, optimization, recruitment, job title similarity, job succes prediction

# List of Abbreviations

| **Abbreviation** | **Description** |
|---|---|
| **AI** | Artificial Intelligence |
| **BERT** | Bidirectional Encoder Representations from Transformers |
| **BoW** | Bag-of-Words |
| **CBOW** | Common Bag Of Words |
| **CNN** | Convolutional Neural Networks |
| **CRF** | Conditional Random Fields |
| **DL** | Deep Learning |
| **DT** | Decision Trees |
| **GPUs** | Graphic Processing Units |
| **GRUs** | Gated Recurrent Units |
| **HMM** | Hidden Markov Models |
| **HR** | Human Resources |
| **k-NN** | k-Nearest Neighbors |
| **LDA** | Latent Dirichlet Allocation |
| **LSTM** | Long Short-Term Memory |
| **LP** | Linguistic Principles |
| **MaxEnt** | Maximum Entropy |
| **MCMC** | Markov Chain Monte Carlo |
| **ML** | Machine Learning |
| **NC** | Name-Class |
| **NER** | Named Entity Recognition |
| **NN** | Neural Networks |
| **NLP** | Natural Language Processing |
| **PPO** | Proximal Policy Optimization |
| **POS** | Part-of-Speech |

| **Abbreviation** | **Description** |
|---|---|
| **RL** | Reinforcement Learning |
| **RNN** | Recurrent Neural Networks |
| **SSL** | Semi-Supervised Learning |
| **SVMs** | Support Vector Machines |
| **TF-IDF** | Term Frequency-Inverse Document Frequency |

# Introduction

In the dynamic landscape of the recruitment industry, the integration of Machine Learning (ML) and Natural Language Processing (NLP) is pivotal. This thesis delves into the applications of ML and NLP, focusing on enhancing and streamlining the hiring process. An exhaustive review of the existing literature has been conducted to pinpoint the current strengths, limitations, and emerging gaps in the field. It is proposed to utilize ML and NLP techniques to elevate the quality of candidate selection, aiming for efficiency and reduced intervention, thereby reducing the time and costs associated with recruitment.

Central to this thesis are novel algorithms and models that not only refine the recruitment process, particularly in candidate screening and job candidate matching, but also emphasize computational efficiency and reduced algorithmic complexity. This aspect is vital, as it addresses one of the major challenges in applying ML and NLP in large-scale recruitment processes: the often prohibitive computational costs and extended execution periods.

The initial filtration process in candidate screening was enhanced using ML and NLP techniques. Language models are deployed to go through vast numbers of candidate profiles and resumes to extract salient information. These were then processed by downstream ML algorithms to identify the most suitable candidates. Furthermore, job matching was examined by using ML and NLP to align job requirements with candidate capabilities more accurately. Semantic analysis, information retrieval, and recommendation systems constitute the backbone of the proposed framework.

This thesis reveals the transformative potential of ML and NLP in the recruitment sector. Recent advancements in machine learning can expedite crucial hiring stages, including candidate screening and job matching, enabling organizations to optimize operations, diminish biases, and enhance recruitment outcomes. The findings contribute significant methodological advancements and offer practical insights for HR professionals and industry practitioners, laying a robust foundation for future research on recruitment technology optimization.

This study is set against the backdrop of artificial intelligence (AI) profound impact across various disciplines, with ML emerging as a powerful tool in creating intelligent systems that mimic human intelligence. The research initially focused on a comprehensive review of the existing literature in the field. The optimization of recruitment processes through ML has shown remarkable efficacy. Recruitment, a vital component of human resource management, has traditionally been labor-intensive and prone to bias. The advent of ML and NLP heralds a new era of automation and optimization in various recruitment phases.

The realization of AI and ML in the future of recruitment motivated this study. The application of ML algorithms in recruitment is proposed to revolutionize the industry and enhance efficiency and impartiality. The hiring process has been inefficient in the past because resumes had to be screened by hand and decisions were made based on bias. ML and NLP are changing this by promising more streamlined processes, fewer biases, and better candidate selection.

NLP, which is integral to ML, plays a pivotal role in this transformation. NLP algorithms are capable of parsing job descriptions, applications, and candidate profiles, thereby allowing for automated resume screening and skill-based job matching. In addition, NLP facilitates the creation of intelligent chatbots and virtual assistants, enhancing candidate engagement and satisfaction.

Moreover, with ML's expanding applications across domains, a key challenge lies in optimizing processes and algorithms to be more time- and resource-efficient. This research contributes to this endeavor in the recruitment industry by leveraging data analysis and automation to accelerate the recruitment cycle and enable more effective resource allocation by HR professionals.

In conclusion, this thesis acknowledges the sweeping changes in the labor market influenced by globalization, the COVID-19 pandemic, and the advent of new professions. The increased volume of job applicants necessitates methodologies that can handle large data volumes, a domain in which ML and NLP excel. Numerous companies are now adopting AI to streamline their recruitment processes, marking a significant shift in the industry.

Around 24% of businesses have adopted artificial intelligence as a method of talent acquisition, according to a survey by The Sage Group in 2020. According to a recent study, the majority of managers (56%) expressed their intention to adopt automated technologies in the upcoming year [1]. Furthermore, the projected value of the AI platform market is expected to reach $29 billion by the year 2019, with a subsequent increase of $52 billion by 2024 [2].

While there is a significant prevalence of AI utilization in the field of recruiting, candidates often have a considerable degree of skepticism regarding the efficacy of their evaluation using intelligent systems. In a survey published in 2019, the focus was on the influence of AI in the recruitment sector. The following responses that the candidates gave when asked whether they preferred to have a human interviewer or a computer interview introduced a novel method of hiring.

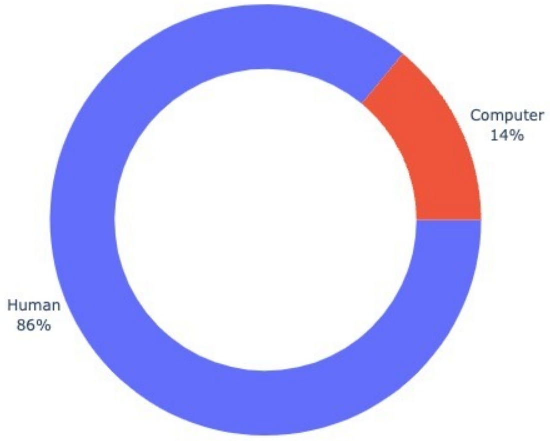


Figure 1: Preference percentage for the interviewer

This suggests that the level of skepticism among the applicants regarding the effectiveness of these algorithms is significant. Subsequently, the candidates were asked, "*Have you ever submitted an application for employment and not received any response?*" The graph below shows the responses provided by the candidates.

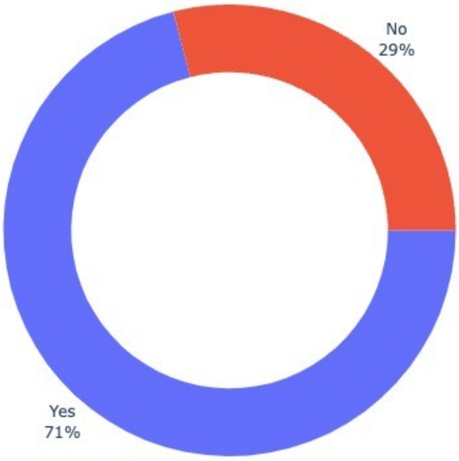


Figure 2: Percentage of companies that respond to applicants

The observed outcome can be attributed to the challenges faced by recruiters or human resources (HR) firms in effectively managing the data of all potential candidates. Based on the aforementioned findings, it can be inferred that the utilization of AI in the field of recruitment has transitioned from optional to imperative.

The candidate selection process involves evaluating the alignment between a candidate's CV elements, such as job title, years of experience, skills, and job criteria. This thesis examined an algorithm that utilizes machine learning techniques to identify similarities between two job titles. The primary objective is to ascertain job titles that can be deemed viable replacements for vacant positions.

In pursuit of these objectives, this dissertation consists of four interconnected studies that employ ML and NLP techniques to optimize particular phases of the recruitment process [3] [4] [5] [6].

These studies aimed to demonstrate the transformative potential of machine learning and natural language processing techniques for recruitment. By utilizing these technologies, we aim to provide HR professionals with invaluable insights and advance the recruitment industry. The idea that ML and AI can improve people's lives and encourage fair competition in the recruitment industry is what drives our research. A future is envisioned where recruitment processes are streamlined, biases are minimized, and candidates are accurately matched with opportunities that align with their talents and aspirations. This thesis endeavors to illuminate the convergence of machine learning, natural language processing, and recruitment. Its goal is to optimize recruitment processes, diminish biases, and enhance overall outcomes through the application of these technologies. The research is directed towards providing HR professionals with essential tools and knowledge for making informed decisions, thereby fostering a more equitable and effective recruitment ecosystem. Ultimately, by harnessing the capabilities of ML and NLP, this thesis aims to refine recruitment practices and facilitate meaningful connections between candidates and organizations.

# Chapter 1

# Survey of the State of the Art: Machine Learning, Natural Language Processing, and Optimization

This chapter provides a comprehensive survey of the latest advancements in ML and NLP. These fields are at the forefront of technological evolution, especially within the domain of recruitment. The aim is to provide a detailed overview of the current state-of-the-art techniques, highlighting the most innovative and impactful developments in these areas.

The focus is on an in-depth examination of contemporary ML and NLP methods, which are considered benchmarks of excellence. These techniques represent the apex of current research and development efforts and are pivotal for understanding the capabilities and future prospects of AI in various applications, including the optimization of recruitment processes.

Beyond merely presenting these state-of-the-art methods, this chapter also proposes optimization approaches for their functions. This involves a critical evaluation of the efficiency, accuracy, and practicality of the existing models and algorithms, with suggestions for enhancing their performance. Such optimization proposals aim to contribute to the ongoing advancement of these technologies.

By offering a clear and comprehensive view of the current landscape of ML and NLP, this chapter sets the foundation for understanding how these technologies can be further refined. This establishes the groundwork for subsequent sections of the thesis, where these insights are applied to tackle specific challenges in the recruitment industry. This chapter serves as a crucial component of the thesis, providing essential knowledge and inspiring future research and application pathways, particularly in the realm of HR and recruitment technologies.

Machine learning, in general, and natural language processing, in particular, are very expansive fields that have inspired a large number of researchers to develop and enhance algorithms and methods.

The emphasis in Section 1.1 is on NLP methods and applications. A discussion of various text preprocessing techniques, such as tokenization, stemming, and stop-word elimination, follows an introduction to the significance and challenges of NLP. In addition, we investigated prevalent text representation and feature extraction techniques in NLP, such as Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and word embeddings. Important NLP tasks, including Named Entity Recognition (NER), sentiment analysis, and text classification, have been examined in depth. In addition, we emphasize the applications of NLP in the field of recruitment, demonstrating how it improves the various phases of the recruitment procedure.

In Section 1.2, an overview of ML techniques is provided. The fundamental concepts of machine learning are introduced, classifying algorithms as supervised, unsupervised, reinforcement, semi-supervised, and active learning. In addition, we examined the principles of deep learning and neural networks, as well as the recently popular transfer learning and ensemble methods.

In Section 1.3, emphasis is shifted to significant ML models. A variety of prevalent ML models, including linear regression, logistic regression, decision trees, random forests, support vector machines, naive Bayes classifiers, k-nearest neighbors, neural networks, and gradient boosting models, have been presented. Each model's strengths, weaknesses, and appropriate use cases were analyzed to provide a thorough comprehension of their properties.

Section 1.4 concludes with an examination of optimization problems in ML. Examine the fundamentals of optimization, including loss function optimization and hyperparameter optimization. Additionally, techniques for feature selection and dimensionality reduction, model selection, and model complexity optimization are discussed. In addition, we discuss optimization tradeoffs, such as bias variance and the difficulties of overfitting and regularization. This section illuminates the critical role of optimization in machine learning and the strategies used to address its challenges. By looking at the present state of machine learning, natural language processing, and optimization, this chapter gives a good understanding of the methods and techniques needed to meet the research goals and make the hiring process more efficient. It provides a thorough overview of the discipline and lays the groundwork for subsequent chapters of this thesis.

## 1.1 NLP Methods and Applications

### 1.1.1 Introduction to NLP

Natural Language Processing, or computational linguistics, is one of the most essential technologies of the twenty-first century. It is fundamentally interdisciplinary and founded on linguistics, computer science, and artificial intelligence. Machine learning is related to natural language processing in that it must "understand" natural language and perform complex tasks, such as language translation and question answering. Because of the following characteristics, the accurate comprehension of human language messages is exceedingly challenging:

- Human language is a complex, discrete, symbolic signaling system with high signal reliability.

- Defined symbols in a particular language can be encoded as communication signals in a variety of forms, including writing, sounds, gestures, and images. Words are the primary, inseparable components of a language.
- Some terms in the language have more than one meaning (polysemy). Frequently, their meanings are diametrically opposed (auto-antonyms). Synonyms are groups of words with the same meaning. In addition, some words function differently as nouns and verbs [7]. Additionally, idiomatic word combinations are characterized by a particular meaning.
- Specific grammatical principles differentiate the structure of various human languages.

In general, texts may contain lexical (multiple meanings of a single word), syntactic (related to morphology ) [8], semantic (sentences with multiple meanings) [9], and anaphoric ambiguity (phrases or words previously mentioned, but with a distinct meaning). Because of this, there is a "high level of complexity in presenting, learning, and using
language/situational/contextual/word/visual knowledge" based on human language [3]. On the other hand, "computers interact with humans through programming languages that are unambiguous, precise, and frequently structured." [7].

As noted in [7] and [10], the most important tasks and applications requiring NLP are:

- Spell checking, keyword search, and synonyms and antonyms
- Information extraction from documents and websites: web content categorization and spam filtering
- Classifying the reading level of academic texts and the positive or negative tone of longer documents
- Automatic Translation.
- Spoken Conversation Systems (Speech Recognition)
- Answering difficult questions.

Additionally, it is essential to mention recently developed applications such as:

- Text summarization
- Anomaly Detection

As previously mentioned, the presentation of knowledge through natural language is often vague or imprecise. To elucidate or reduce ambiguity in the verbal or written linguistic representations of knowledge, the aforementioned NLP tasks were performed. To achieve this objective, potent analysis instruments such as statistical techniques and machine learning are employed. Machine learning employs optimization techniques that significantly influence the development of this field. The main purpose of [8] is to examine the machine learning techniques used in natural language processing, categorized according to the models employed, and then to provide a concise summary of the most important approaches and optimization techniques employed in this scientific field. [8] provides an overview of contemporary machine learning models for NLP. Optimization methods from the perspective of machine learning were reviewed in [11].

### 1.1.2 NLP Techniques for Text Preprocessing

The foundation lies in effective text preprocessing, a phase crucial for structuring and cleaning raw text data.

Tokenization: An initial step in preprocessing, tokenization dissects text into smaller components such as words, phrases, or symbols. This ensured that the text was in digestible pieces for further processing.

Stemming: Simplifying words to their base or root forms and stemming aids in reducing the dataset size. For instance, “running,” “runner,” and “ran” might all be reduced to “run.”

Lemmatization: A sophisticated version of stemming, lemmatization, uses lexical knowledge bases to associate word forms with canonical bases. Unlike stemming, this ensures that the root word has a valid dictionary representation.

Stopword Removal: In data analytics, not all words carry equal weight. Stopword removal eliminates common words like ‘and,’ ‘the,’ or ‘in,’ which might clutter the analysis without adding substantial meaning.

Normalization: Ensuring text consistency is vital. Normalization processes, such as converting text to lowercase or standardizing data formats, create a uniform dataset.

Noise Removal: This involves purging data of irrelevant elements such as special characters, numbers, or even misprints that do not contribute to the primary analysis.

### 1.1.3 Text Representation and Feature Extraction in NLP

Machine comprehension of text requires effective representation and feature extraction.

BoW: A rudimentary technique, Bag-of-Words, enumerates the occurrence of words in a document, ignoring the context or sequence in which they appear.

TF-IDF: Term Frequency-Inverse Document Frequency offers a more nuanced representation, calculating word importance relative to its frequency across multiple documents.

Word Embeddings: Pioneering approaches such as Word2Vec and GloVe convert words into vectors. In this multidimensional space, linguistic similarities translate into spatial proximity.

n-grams: This technique captures context by considering sequences of n adjacent words or tokens, providing a richer representation.

Part-of-speech (POS) tagging: By labeling words as nouns, verbs, adjectives, etc., POS tagging sheds light on grammatical structure and potential relationships within the text.

### 1.1.4 Named Entity Recognition and Entity Linking

Gleaning specific information from vast textual data involves the recognition and linking of entities. Named Entity Recognition: A crucial step in data extraction, NER pinpoints and categorizes entities within the text, whether they are the names of individuals, organizations, dates, or other specifics.

Entity Disambiguation: Context is king. This technique ensures the correct interpretation of recognized entities, differentiating, for instance, between ‘Apple’ the company and ‘apple’ the fruit.

Entity Linking: By connecting the identified entities to external knowledge bases or databases, this process provides depth and context to the extracted information, facilitating a comprehensive understanding.

### 1.1.5 Sentiment Analysis and Opinion Mining

In the age of online reviews and social media, extracting sentiments and opinions from texts is invaluable.

Polarity Detection: A fundamental aspect of sentiment analysis, polarity detection, discerns whether the expressed sentiment leans positive, negative, or remains neutral.

Emotion Detection: Beyond mere polarity, emotion detection strives to identify specific feelings such as joy, anger, or sadness expressed in the text.

Aspect-based Sentiment Analysis: Focusing on specific features or components of a topic, this technique determines the sentiments tied to individual aspects.

Opinion Extraction: Structuring opinions from a corpus aids in comprehensive sentiment analytics and provides a clear overview of public perception.

### 1.1.6 Text Classification and Document Clustering in NLP

Organizing and categorizing vast volumes of text is instrumental for effective data analysis.

Supervised Classification: With foundation in labeled data, this method trains models to sort and classify new, unseen data into predefined categories.

Topic Modeling: Algorithms such as Latent Dirichlet Allocation (LDA) are employed to uncover dominant themes or topics in extensive text corpora, offering insights into the primary subjects discussed.

### 1.1.7 NLP Applications in Recruitment

Modern recruitment strategies harness the power of NLP for efficiency and precision.

Automated resume screening: One is the number of days of manual resume sorting. NLP algorithms swiftly parse and rank resumes, thereby highlighting the most suitable candidates.

Chatbots for Initial Interactions: Digital assistants powered by NLP engage candidates in preliminary discussions, gather essential data, and answer queries.

Sentiment Analysis: Candidate feedback, whether from interviews or online platforms, undergoes sentiment analysis, ensuring that the recruitment process aligns with candidate expectations and improving employer branding.

## 1.2 ML Techniques

### 1.2.1 Introduction to ML

Machine Learning is a subfield of artificial intelligence concerned with the development of algorithms and models that enable unprogrammed computers to learn patterns and make predictions or decisions based on data. ML algorithms have been widely implemented in numerous domains, including Natural Language Processing, where they have substantially contributed to the field's advancement. ML techniques in NLP enable the automatic extraction of linguistic features, comprehension of semantic relationships, and generation of human-like languages.

There are two primary approaches to natural language processing: 1) language processing based on linguistic principles (LP) and 2) machine learning (ML), in which statistical machine learning algorithms are used [9, 12]. There are two main categories of statistical methods for NLP: clustering techniques and machine-learning algorithms. ML attempts to solve the problem of knowledge extraction through a classification procedure, whereas clustering is based on a measure of similarity. Not only are ML algorithms widely used in the NLP domain but also in many other domains. In the LP approach, the developer employs a linguistic engine with knowledge of the syntax, semantics, and morphology of a language [12] and then adds program rules that search for key semantic concepts that determine the meaning of a particular verbal expression. Manually constructed by linguists or grammarians to execute a variety of NLP-specific tasks.

By using neural networks and cognitive modules that can learn from historical data, machine learning employs data-driven machine learning. In ML, language processing principles are derived from the training data. Statistical machine learning algorithms synthesize textual characteristics using historical data. These characteristics are then employed for classification or prediction. In contemporary research in this discipline, machine-learning models are used. Real specific weight values are assigned to the introduced characteristics in the models, resulting in the generation of probabilistic solutions. The models have the advantage of being able to represent "the relationship quality in different dimensions" [9]. Machine learning rule extraction can be accomplished by solving an optimization problem. [13] describes and contrasts two GA-based strategies for discovering non-dominated rule sets. The extraction of linguistic rules is expressed as a three-objective combinatorial optimization problem [13]. The NLP tasks were executed using statistical and probabilistic techniques. ML techniques are widely used to execute primary NLP.

In general, ML techniques fall into six general categories:

- supervised machine learning
- unsupervised machine learning
- reinforcement learning algorithms
- semi-supervised machine learning
- deep learning, neural networks
- transfer learning and ensemble methods

The most common method for performing NLP tasks is supervised machine learning, which is based on the automatic extraction of principles from the training data [4]. The classifications for supervised ML are (i) sequential and (ii) non-sequential. Deep Learning (DL), Conditional Random Fields (CRF), k-Nearest Neighbors (k-NN), Hidden Markov Models (HMM), and Maximum Entropy (MaxEnt) are sequentially supervised ML techniques. Naive Bayes, Decision Trees (DT), and Support Vector Machines (SVM) are non-sequential supervised ML techniques. Semi-supervised machine learning employs a small amount of supervision. This method is illustrated using bootstrapping. Unsupervised machine learning employs models that have not yet been trained. The ML task is solved by searching for intra-similarities and inter-similarities between objects (see [9]). Clustering and vector quantization are the most common approaches in this ML category (refer to [14]).

To comprehend the utility of ML techniques in NLP applications, it is necessary to comprehend the underlying concepts and methodologies. The following sections provide an overview of machine learning techniques and their applications in natural language processing.

### 1.2.2 Supervised Learning Algorithms

In supervised learning, a model is trained on labeled data, where each data instance is associated with a known output or label. Supervised learning algorithms construct a generalizable model from input-output pairings. Various tasks, including text classification, sentiment analysis, named entity recognition, and machine translation, have utilized supervised learning algorithms in the context of NLP.

In text classification, supervised learning algorithms can be trained on labeled data to determine the category or class of a given text. Popular supervised learning algorithms utilized in NLP tasks include Support Vector Machines [15], decision trees, and neural networks [16]. SVMs are famous for being able to handle large amounts of data and relationships that do not follow a straight line. They have been used a lot in text classification tasks to get state-of-the-art results. By contrast, decision trees provide interpretable models that can be readily comprehended and visualized. Neural networks, specifically deep learning architectures, have demonstrated exceptional performance in various NLP tasks, including sentiment analysis and text generation.

### 1.2.3 Unsupervised Learning Algorithms

When data is unlabeled, unsupervised learning algorithms are used to uncover latent patterns or structures within the data. Unsupervised learning techniques such as clustering and topic modeling are extensively used in natural language processing. Clustering algorithms group similar documents or words based on their semantic similarity, which is beneficial for document organization and recommendation systems. Topic modeling algorithms, like Latent Dirichlet Allocation [17], help with tasks like summarizing documents and finding information by revealing hidden topics in a group of documents.

Clustering algorithms, such as k-means clustering and hierarchical clustering, divide data into groups or clusters based on similarity measures. These techniques are valuable for organizing massive quantities of unlabeled text data, such as news articles and customer reviews. Clustering algorithms can assist with duties such as document recommendation and content organization by grouping similar documents together.

Algorithms for topic modeling, such as LDA, unearth latent themes or topics within a collection of documents. By giving words and documents probabilistic distributions, these algorithms make it possible to find topics and how common they are in each document [18]. This can be useful for applications such as content recommendation, sentiment analysis, and the identification of trends in large document collections.

### 1.2.4 Reinforcement Learning Algorithms

Reinforcement learning (RL) is a learning paradigm in which an agent interacts with its environment and learns to maximize a cumulative reward signal by taking action [19]. RL has been applied to tasks such as dialogue systems, text generation, and machine translation in the context of NLP. RL techniques have been used to teach conversational agents to generate cogent and context-appropriate responses. By rewarding the agent to generate the desired responses, RL can optimize the agent's behavior over time.

In dialogue systems, RL algorithms have been used to teach agents how to have natural, coherent conversations with users. These algorithms use rewards and punishments to guide the agent's responses and optimize the quality of the dialogue [20]. Text-generation tasks, such as machine translation and summarization, have also benefited from RL techniques. By adjusting reward signals based on translation quality or summary coherence, RL algorithms can make translations and summaries that are more accurate and flow better.

### 1.2.5 Semi-Supervised and Active Learning Techniques

Semi-supervised learning techniques enhance the performance of NLP models by utilizing both labeled and unlabeled data. These methods are particularly useful in situations where labeled data is limited. Semi-supervised learning algorithms seek to improve the accuracy and generalization of the model by combining large amounts of unlabeled data with limited labeled data.

Semi-supervised learning has been applied to several tasks in NLP, including text classification, named entity recognition, and sentiment analysis. For instance, in text classification, a small set of labeled data and a larger set of unlabeled data can be combined to train a more accurate classifier [21]. The model can learn stronger representations from a lot of unlabeled data with this combination, which makes it better at the target task.

Active learning techniques, on the other hand, focus on picking the most useful examples from the big collection of unlabeled data to label, which makes labeling easier [22]. These methods ask a human annotator on the fly to label the cases that are the most uncertain or hard to label. This improves model performance while lowering the cost of labeling.

### 1.2.6 Deep Learning and Neural Networks

Deep Learning, powered by neural networks, has revolutionized NLP by attaining cutting-edge performance on a variety of tasks. Neural networks (NN) consist of interconnected neurons that mimic the structure and functions of the human brain. Deep learning models, such as Convolutional Neural Networks (CNN) and Recurent Neural Networks (RNN), have been effectively applied to NLP tasks, such as text classification, sentiment analysis, machine translation, and text generation.

CNNs have demonstrated outstanding efficacy in text classification and sentiment analysis tasks [23]. These models use convolutional layers to pull out local patterns and features from text. This lets them find important features at different levels of detail. By using convolutional filters on input text, CNNs can learn hierarchical representations that hold useful data for classification tasks.

With their recurrent connections, RNNs are ideally suited for sequential data tasks, such as language modeling, machine translation, and text generation. These models can capture the context and dependencies between words or characters in a sequence, enabling them to generate text that is coherent and appropriate. Some well-known types of RNN are Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs). They help with the vanishing gradient problem and make it possible to accurately model long-range dependencies [24].

### 1.2.7 Transfer Learning and Ensemble Methods

Transmission learning permits the transmission of knowledge from one task or domain to another related task or domain. A lot of the time, in NLP, models that have already been trained are used to set up new models for later tasks [25]. Examples of these models are Word2Vec, GloVe, and BERT. Models can benefit from large-scale pre-training and generalize well even when there is not a lot of labeled data in the target task by using the representations they have learned.

By contrast, ensemble methods combine multiple models to make predictions or decisions. Ensemble methods can be used to enhance the performance and robustness of NLP models in natural language processing. Ensemble methods can reduce the risk of overfitting and enhance generalization by combining multiple models trained on distinct subsets of data or by utilizing distinct architectures [26].

## 1.3 Important Machine Learning Models

### 1.3.1 Supervised ML models

**A. Sequential models**

- *Deep Learning*

  Deep learning is a subfield of machine learning that employs artificial neural networks. It is becoming increasingly popular and can be used to achieve multiple levels of abstraction. Neural networks were first introduced in 1958 [27], but their popularity has only increased since 2012 [28] with the use of large labeled data sets and graphic processing units (GPUs) [29]. "Deep learning" means that a neural network, which is a layered model of inputs, uses more than one layer [30] to learn more than one level of representation or of increasing complexity (abstraction) [31]. Each succeeding layer utilizes the output of the preceding layer and passes its output to the succeeding layer.

  DL networks are utilized for processing natural language [32] and speech [33]. In addition, they can perform sentiment analysis, parsing, name entity recognition, and other NLP duties. Using a DL approach, [34] performs tasks involving Chinese word segmentation and part of speech. [35] proposes a DL neural network architecture for word- and character-level representations as well as POS labeling. In [36], word segmentation is performed using a perceptron-based learning algorithm and a character-based classification system. [37] proposes a deep neural network system for information extraction tasks. Arabic NER tasks are conducted in three phases

[38]. The results demonstrated that neural networks outperform decision trees.

- *Conditional random field model*

  In [39], a conditional random field was introduced. This statistical model applies structured prediction to pattern recognition and machine learning. CRFs are the most adaptable models for many tasks in the NLP domain [40]. Instead of a joint distribution, the CRF model is founded on a conditional distribution [41].

  CRF is defined in [39] for $X$-random variables over labeled data sequences and $Y$-random variables over label sequences. All components $Y_i$ of $Y$ are presumed to fall within the finite label alphabet. Random variables $X$ and $Y$ were jointly distributed. A conditional model, $p(Y|X)$, was constructed from paired observations and label sequences. The marginal $p(X)$ is not explicitly modeled.

  Let $G = (V, E)$ be a graph such that $Y = (Y_v)$ for $v \in V$, where $Y$ is indexed to $G$'s vertices. Then $(X, Y)$ is a conditional random field if the random variables $Y_v$, conditioned on $X$, satisfy the following Markov property with respect to the graph:

  $$p(Y_v|X, Y_w, w \neq v) = p(Y_v|X, Y_w, w \sim v), \tag{1.1}$$

  where $w \sim v$ indicates that $w$ and $v$ are neighbors in $G$. The CRF is regarded as a global random field constrained by observation $X$, and graph $G$ is assumed to be fixed. In the CRF model, the objective is to maximize a log-likelihood objective function that maximizes the conditional likelihood $p(y|x)$ for a sequence $x$. Every HMM can be written in this form, as can be seen simply by setting $\lambda_{i,j} = \log p(y_0 = i|y = j)$ and so on. Each feature function has the form $f_k(y_t, y_{t-1}, x_t)$. Because it is not required the parameters to be log probabilities, it is no longer guaranteed that the distribution sums to 1, unless it is explicitly enforced by using a normalization constant $Z$.

  CRF models are typically feature-based and can operate on binary or real-valued features/variables. According to the fundamental theorem of random fields [42], the joint distribution over label sequence $Y$ given $X$ is given as:

$$p(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{t=1}^{T} \lambda_k f_k(y_t, y_{t-1}, x, t)\right) \tag{1.2}$$

The conditional random field method for NER is described in [43]. A K-nearest neighbor classifier that has been mixed with a standard linear CRF model is suggested as a way to do NER. A CRF-based bootstrapping technique for Arabic-named entity recognition was discussed in [44]. The proposed method was applied to an Arabic-named entity recognition problem. The NER problem in Chinese was looked at in [45] in the context of a CRF model that used an active learning algorithm based on pools. [46] [47] [48] proposed models for language processing based on the CRF model for POS labeling.

- *Hidden Markov Model*

The Hidden Markov Model is part of a probabilistic model. It has both seen events, like words in the input, and hidden events, like part-of-speech marks, that are thought to be causal factors [49]. HMM also performs a classification or labeling of sequences. In the case of a sequence classifier, it identifies a class label for each token and then allocates that label to each token or word of the input sequence. HMMs are highly effective in textual classification and POS labeling tasks, such as name recognition [50]. The objective is to recognize person names, place names, brand names, designations, dates, times, abbreviations, numbers, etc., and classify them into predefined categories [52].

In this assignment, labels are assigned to the words in context (one label per word). The states of the HMM model are divided into regions, one for each desired class and one for Not-A-Name. In addition, there are two special states: Start-Of-Sentence and End-Of-Sentence. Using a probabilistic Bigram language model, the probability of a sequence of words occurring within each region (name-class (NC)) is computed. The model employs a Markov chain in which each word's probability depends on the probability of the preceding word. Each word in the

Bigram paradigm has a state symbol, si, to represent it. There is a probability associated with the transition from the present to the following. Two assumptions are made in the HMM [51]:

1) Markov assumption:

$$P(s_i|s_1 \dots s_{i-1}) = P(s_i|s_{i-1}) \tag{1.3}$$

and

2) Output-independent assumption:

$$P(o_i|s_1, \dots, s_T, o_1, \dots, o_i, \dots, o_T) = P(o_i|s_i) \tag{1.4}$$

where the probability of the resultant observation $o_i$ is calculated based solely on the probability of the state generating the observation and is independent of the probabilities of the surrounding states.

A vector of initial probabilities, a matrix of transition probabilities, and a matrix of observation probabilities are utilized in the HMM model. This model determines the probability of a sequence of words $w_1$ through $w_n$, as follows:

$$P = \prod_{i=1}^{n} p(w_i|w_{i-1}). \tag{1.5}$$

Using a unique "begin" word, the probability of $w_1$ is calculated. According to the statistical bigram model, the number of states in each name-class region is equal to the vocabulary size $|V|$.

Because it generates a sequence of words and labels, HMM is a generative model. In the task of name recognition, the most probable sequence of name-classes $NC$ is determined given a sequence of words $W$:

$$\max P(NC|W) = \frac{P(W, NC)}{P(W)}, \tag{1.6}$$

based on the Bayes' rule. The denominator in (1.6) represents the unconditional probability of the word sequence and is constant for any given sentence. Consequently, only maximization of the numerator is required.

HMM was proposed for NER tasks in [53]. It operates on sentence-by-sentence data and assigns each word to a corresponding NE identifier. [54] examines the NER for Hindi, Marathi, and Urdu using HMM with high levels of precision. For POS identification, [55] employs a rule-based method in conjunction with HMM. The objectives of sentence boundary detection (SBD) are discussed in [56] [57]. The problem of word segmentation was resolved using HMM in [58] [59].

- *Maximum entropy model*

The maximum-entropy model is a general-purpose model for drawing inferences or predictions from incomplete data. It is used to classify sequential data

This model accomplishes its mission in three steps: 1) extracting pertinent features from a given input sequence; 2) conducting a linear combination of the extracted features; and 3) calculating the exponent of the resulting sum [60]. Given the observation $o$, the probability distribution of a certain class $x$ is given by:

$$P(x|o) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} w_i f_i\right), \tag{1.7}$$

where $Z$ is a normalization function, and exp = $e^x$.

In [61], a NER system based on MaxEnt was proposed. In [62], a POS labeling system based on the MaxEnt model is developed.

- *$k$-Nearest Neighbors*

$k$-Nearest Neighbors are simple instance-based or idle supervised machine-learning algorithms [63], [64], [65], [63], [67]. It can be used for regression analysis. The

$k$-NN is a non-probabilistic and non-parametric model [65] that does not require explicit training [66]. This model is ideally suited for classifying data in the absence of prior knowledge of their distribution. This classification method relies on a distance-based metric. The fundamental concept of the $k$-NN algorithm is to select the $k$-Nearest Neighbors for each test sample and then use them to predict the test sample. Complexity increases as the number of dimensions increases. Therefore, dimensionality reduction techniques [68] must be implemented prior to utilizing $k$-NN.

### B. Non-sequential models

- *Naive Bayes*

The Naive Bayes classifier is a supervised machine learning algorithm (a labeled dataset) based on the well-known Bayes theorem of probability. The likelihood of a given specimen belonging to a particular category depends on Bayes' hypothesis. Based on the strong presumption that all input features are independent of one another and have no correlation, the term "naive" is applied. This classifier is utilized for binary and multiclass statistical classification problems, particularly document classification [69] and [70]. The Naive Bayes classifier requires a small amount of set preparation to determine the classification parameters [71].

- *Decision trees*

Decision trees are used for classification and prediction in machine learning. The DT model is a tree in which each node represents a decision and each leaf represents an output class. A node may have more than two offspring, but the vast majority of algorithms employ only binary trees. DT was first applied to language modeling in [72] to estimate the probability of spoken syllables. A single node serves as the starting point, after which binary questions are posed to arbitrarily divide the space of the histories. As the space is partitioned, "leaves" are formed, and training data are utilized to calculate the conditional probability of $P(w|h)$ for the following element. Using information-theoretic metrics, questioning becomes increasingly informative as the traversal proceeds. Among these metrics are Kolmogorov complexity, entropy, and relative entropy [73]. In [74], a tree-pruning method that uses the development set to delete overfitted model nodes and a result caching technique were proposed. This means that the new algorithm works one to three times faster than a simple one. It also works correctly on datasets for letter-to-sound (LTS), syllabification, part-of-speech tagging, text classification, and tokenization.

- *Support vector machines*

SVM is primarily utilized for binary classification of both linear and nonlinear data. Utilizing a small subset of data (feature vectors), this model divides data across a decision boundary (hyperplane). In the field of natural language processing, SVMs are applied to a variety of tasks, such as content arrangement, POS, NER, and segmentation. The classification rule for the separating hyperplane is as follows:

$$f(x, w, b) = W.X + b = 0, \tag{1.8}$$

where $W = w_1, w_2, w_3, \ldots, w_n$ is a weight vector, $n$ is the number of attributes, $x$ is the instance to be classified, and $b$ is a scalar, also known as a bias. Equation (1.8) can be written as:

$$w_0 + w_1 x_1 + \ldots + w_n x_n = 0, \tag{1.9}$$

where $b$ is represented as an additional weight, $w_0$, and $x_1$ and $x_2$ are the values of $X$'s attributes $A_1$ and $A_2$, respectively.

Based on [75], a combination of SVM and CRF was used for Bengali-named entity recognition. In [76], an SVM-based POS tagger for the Malayalam language was developed.

### 1.3.2 Unsupervised ML models

- *Clustering*

A clustering model [77] and [78] were used to discover a structure or pattern in an unlabeled dataset collection. A clustering algorithm divides a specified set of data into $K$ clusters. The data points within each cluster were comparable to one another, whereas those between clusters were dissimilar. Similar to the $k$-NN algorithm, a similarity metric or distance metric is employed, such as Euclidean, Mahalanobis, cosine, or Minkowski. The Euclidean distance metric is widely used; however, [79] demonstrated that this metric cannot guarantee clustering quality. $K$-means clustering is one of the most straightforward clustering algorithms. It divides the data points into $K$ equal variance groups while minimizing the within-cluster sum of squares, or inertia.

- *Vector quantization*

The vector quantization model [80], [81], and [82] organizes data into vectors and represents them through their centroids. The quantizer is typically trained using a $K$-means clustering algorithm. The centroids comprise code words. The Codebook stores all codewords. Vector quantization is a lossy compression technique. It has been utilized in numerous coding applications. The data compression errors were inversely proportional to the density. Vector quantization has been utilized in speech coding [81] and [84], audio compression [83], and numerous other applications.

### 1.3.3 Reinforcement learning

- *Proximal Policy Optimization*

The popular reinforcement learning algorithm proximal policy optimization (PPO) integrates deep learning with policy optimization. The objective of the PPO is to train agents to make sequential decisions in a given environment by maximizing cumulative rewards. It approximates the policy function using a policy network that is often based on deep neural networks [85]. PPO has been used successfully for many NLP tasks, such as training dialogue agents and text generation models. It lets agents try different strategies until they find the best ones.

### 1.3.4 Semi-supervised ML models

- *Bootstrapping*

  In [86], the DIPRE system is described, which uses bootstrapping to derive a relationship between books (author, title) from the Internet. This system represents seed occurrences as three-string contexts: words preceding the first entity (BEF), words between the two entities (BET), and words following the second entity (AFT). DIPRE creates extraction patterns by putting contexts into groups based on string matching and controlling semantic drift by limiting the number of instances a pattern can pull out.

  The Snowball system is talked about in [87]. It is based on the DIPRE method of collecting three contexts for each occurrence and computing a TF-IDF representation for each one. A single-pass algorithm was used to group contexts together. It was based on the cosine similarity between contexts and vector multiplication between the sentence-score vectors [88].

### 1.3.5 Deep Learning and Neural Networks

- *Transformer Model BERT*

  The Transformer models, especially Bidirectional Encoder Representations from Transformers (BERT), have changed the way NLP tasks are done by giving us cutting-edge results. BERT is a deep learning model based on a self-attention mechanism that can detect contextual relationships between words in a text. It has been pretrained on massive datasets and can be fine-tuned for a variety of NLP tasks, including language comprehension, sentiment analysis, and named entity recognition [25] BERT's ability to capture context in both directions significantly improved the NLP model's performance.

### 1.3.6 Transfer Learning and Ensemble Methods

- *Stacked Ensemble Model*

  The stacked ensemble model is a potent ensemble technique that generates a final prediction by combining the predictions from multiple base models. It capitalizes on the advantages of the individual models and mitigates the risk of overfitting. In the field of natural language processing, the stacked ensemble method has been used to make models better at tasks like text classification and sentiment analysis [89]. By combining the predictions of multiple models, the stacked ensemble model provides more accurate and reliable results.

## 1.4 Optimization Problems in Machine Learning

In the machine learning process, an optimization problem is formulated, and the extremum (maximum or minimum) of an objective function is sought. The objective function is the objective or criterion that the model seeks to optimize. During the first step of any machine learning method, a good model for the job and a reasonable objective function that matches the goal are chosen.

Supervised learning, which includes classification and regression tasks, formulates optimization problems with the goal of minimizing classification errors or maximizing prediction accuracy. For example, the goal of sentence classification [90] and [91] might be to correctly put text into different groups, while the goal of regression problems [92] is to guess continuous variables from the features that are given.

Unsupervised learning concentrates on clustering and dimensionality reduction-related optimization problems. Clustering strives to group data points with similar characteristics, whereas dimensionality reduction attempts to identify the most important features while reducing the number of input dimensions. Document clustering and customer segmentation are examples of these methods [93].

The following sections delve into these machine learning classifications and discuss specific optimization problems and techniques for each. It investigates the optimization techniques used in supervised learning, unsupervised learning, and other areas of machine learning, shedding light on how they contribute to improving the model's performance and attaining the desired results.

### 1.4.1 Supervised learning optimization problems

The goal of supervised learning is to create an ideal mapping function, $f(x)$, that accurately predicts the labels given input features. The optimization problem consists of minimizing the loss function of the training samples, which quantifies the difference between the predicted and actual labels.

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y^i, f(x^i, \theta)), \tag{1.10}$$

where $N$ is the number of training samples, $x^i$ is the feature vector of the $i$-th sample, $y^i$ is the corresponding label, $\theta$ is the corresponding parameter of the mapping function, and $L$ is the loss function.

Typically, the loss function is defined as the error or dissimilarity between the predicted and ground-truth labels. Different loss functions, like the square of the Euclidean distance, contrast loss, hinge loss, cross-entropy, and information gain, can be used depending on the problem. Functions capture various facets of the prediction error and direct the optimization process toward locating the optimal model parameters.

Regularization terms are typically included in the objective function to prevent overfitting and enhance generalization. These regularization terms encourage simplified models by penalizing complicated parameter configurations. The L2 norm is a common regularization term that encourages smaller parameter values and helps regulate the complexity of the model.

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y^i, f(x^i, \theta)) + \lambda ||\theta||_2^2 \tag{1.11}$$

A regularization parameter, $\lambda$, can be determined by means of cross-validation. One can pick the value using methods like cross-validation, which tests the model's performance on validation data for different values and picks the one that fits the training data the best while also avoiding overfitting.

It seeks to attain a balance between model complexity and generalization performance by formulating an optimization problem in supervised learning.

### 1.4.2 Semi-supervised learning optimization problems

The semi-supervised learning (SSL) bridges the gap between supervised and unsupervised learning. It utilizes both labeled and unlabeled data during the training process, making it useful in situations in which acquiring labeled data is costly or time-consuming. SSL can be used in a variety of applications, including classification, clustering, regression, and dimensionality reduction.

In classification tasks [94], SSL uses labeled data to learn a model that can generalize to unlabeled data, thereby improving classification accuracy. By using unlabeled data, which often has more information than labeled data, SSL can effectively take advantage of the way data is distributed and find more accurate representations.

SSL algorithms can utilize both labeled and unlabeled data to identify meaningful clusters for clustering tasks [95]. This allows the model to uncover concealed data structures and enhance clustering performance.

SSL combines labeled and unlabeled data in regression assignments [96] to improve the accuracy of the regression models. SSL can develop more robust regression models by incorporating unlabeled data, which provides additional information about the data distribution.

Dimensionality reduction tasks [97] consist of reducing the number of input features while preserving essential data. When SSL methods are used to reduce the number of dimensions, they use both labeled and unlabeled data to find a low-dimensional representation that shows how the data is really structured.

There are various approaches in the semi-supervised learning domain. GANs and generative probabilistic models are two types of generative models that look for the underlying distribution of data and add labeled synthetic data to a dataset that already has labels.

Support vector machines (SVMs) [98] are a type of traditional support vector machines that can work with unlabeled data to make decision and classification lines more accurate. Self-training methods iteratively train models on initially labeled data and then use the trained models to generate pseudo-labels for unlabeled data. This process persists until convergence occurs and the model is iteratively refined.

Labeled and unlabeled data points are connected to one another based on their similarities in graph-based methodologies. These methods propagate label information throughout the graph, enabling the inference of labels for previously unlabeled data points.

Multilearning methods seek to improve the learning process by utilizing multiple related tasks. By training multiple tasks simultaneously, knowledge can be shared across tasks, leading to enhanced performance for each task.

These are only a few examples of a wide variety of semi-supervised learning techniques. Each method has its own advantages and disadvantages, and the choice of method depends on the nature of the problem and the available resources. By using both labeled and unlabeled data, semi-supervised learning methods may be able to take advantage of the best parts of both supervised and unsupervised learning, leading to better performance and better use of data.

This example explains the SSSVM as a learning model capable of handling binary classification issues. Only a portion of the training set was labeled in this instance.

Let $D^l$ be labeled data:

$$D^l = \{\{x^1, y^1\}, \{x^2, y^2\}, \dots, \{x^l, y^l\}\},$$

and $D^u$ be unlabeled data:

$$D^u = \{x^{l+1}, x^{l+2}, \dots, x^N\}, \quad \text{with } N = l + u.$$

The original goal of the SVM with slack variable $i$ is changed to include an extra constraint on the unlabeled data so that it can be used. It defines $e_j$ as the misclassification error of the unlabeled instance if its true label is positive, and $z_j$ as its misclassification error if its true label is negative.

The SSSVM problem can be formulated as follows:

$$\min \|\omega\| + \gamma \left[ \sum_{i=1}^{l} \beta^i + \sum_{j=l+1}^{N} \min(\alpha^i, z^j) \right], \tag{1.12}$$

subject to

$$\begin{aligned} y^i(\mathbf{w} \cdot x^i + b) + \beta^i &\geq 1, & \beta^i &\geq 0, & i &= 1, \dots, l, \\ \mathbf{w} \cdot x^j + b + \alpha^j &\geq 1, & \alpha^j &\geq 0, & j &= l+1, \dots, N, \\ -(\mathbf{w} \cdot x^j + b) + z^j &\geq 1, & z^j &\geq 0; & & \end{aligned}$$

where $\lambda$ is a coefficient of penalty. The optimization problem (1.12) is a mixed-integer problem that is NP-hard [99]. There are numerous methods [100] for resolving this issue, including convex relaxation [101] and branch-and-bound methods [102].

### 1.4.3 Unsupervised learning optimization problems

In unsupervised learning, optimization problems arise in various tasks, such as clustering and dimensionality reduction. Let us explore these optimization problems in more detail.

Clustering algorithms [103] aim to group samples into multiple clusters based on their similarities. The objective is to minimize the differences between samples within the same cluster while maximizing the dissimilarities between samples in different clusters. The optimization problem for the popular k-means clustering algorithm involves minimizing the loss function.

$$\min_{S} \sum_{k=1}^{K} \sum \|x - c_k\|_2^2 \tag{1.13}$$

Here, $K$ represents the number of clusters, $x$ denotes the feature vector of the samples, $c_k$ represents the center of cluster $k$, and $S_k$ represents the sample set of cluster $k$. The objective is to minimize the sum of variances within each cluster, ensuring that samples within the same cluster are as similar as possible. By optimizing this objective function, the algorithm identifies the optimal cluster centers and assigns samples to their respective clusters.

Dimensionality reduction algorithms aim to project high-dimensional data into a lower-dimensional space while preserving essential information. One widely used technique for dimensionality reduction is Principal Component Analysis (PCA) [104]. The objective of PCA is to minimize the reconstruction error, which quantifies the loss of information during the dimensionality reduction process. The error is calculated as follows:

$$\min \sum_{i=1}^{N} \|\bar{x}^i - x^i\|_2^2, \quad \text{where } \bar{x}^i = \sum_{j=1}^{d'} z_j^i e_j, \quad d \gg d', \tag{1.14}$$

where $N$ represents the number of samples, $x^i$ is a $d$-dimensional vector, $\bar{x}^i$ is the reconstruction of $x^i$. The projection of $x^i$ in $d'$-dimensional space is $z^i = \{z_1^i, \ldots, z_{d'}^i\}$, and $e_i$ is the standard orthogonal basis in the $d'$-dimensional coordinates.

The optimization problem in PCA involves finding a set of orthogonal vectors, known as principal components, that capture the maximum variance in the data. These principal components are derived from the covariance matrix of the input data. The goal of PCA is to minimize the reconstruction error and keep as much of the original information as possible in the lower-dimensional representation. This is done by choosing a subset of the principal components that explain a large portion of the data variance.

By solving the optimization problem in unsupervised learning, this aims to discover meaningful patterns and structures in the data. In clustering, the objective is to group similar samples together, while in dimensionality reduction, the goal is to find a compact representation that preserves the underlying information. Unsupervised learning algorithms use these optimization problems to help us understand the data without using labeled data. These techniques also assist in uncovering hidden patterns and reducing the number of dimensions, enabling a more comprehensive analysis or visualization of the data.

In probabilistic models, the optimization problem frequently entails locating the maximal value of a probability density function (PDF) $p(x)$. This is the logarithmic likelihood function of the training samples, where the goal is to maximize the likelihood of the observed data given the model parameters.

The optimization problem can be stated as follows:

$$\max \sum_{i=1}^{N} \log p(x^i, \theta) \tag{1.15}$$

where $x$ represents the observed data, and $\theta$ represents the model parameters. The objective is to determine the values that maximize the probability of the data. This can be accomplished by determining the optimal parameter configuration that produces the highest probability of the observed data.

In Bayesian methods, prior distributions are often used to make sure that the model parameters do not fit too well and that the results are more general. Before observing the data, these prior distributions encode our prior knowledge or beliefs about parameter values. Using prior distributions, the optimization problem tries to find the parameter values that both maximize the chance and match what it is already known.

One can use Bayesian optimization methods, like Markov Chain Monte Carlo (MCMC) or variational inference, to look into the parameter space and get an idea of the posterior distribution over the model parameters. These methods provide a principled method for balancing the likelihood and the prior while accounting for uncertainty and generating more robust and calibrated models.

A detailed overview of the numerous optimization methods in machine learning, such as gradient-based methods, evolutionary algorithms, and Bayesian optimization, is available in [5]. This survey explores various optimization techniques, their strengths, limitations, and applications in diverse ML domains.

Optimization methods are used in probabilistic models to identify the best parameter settings that increase the likelihood of the observed data while incorporating prior knowledge.

This makes the models more accurate and reliable for capturing the patterns and distributions in the data.

### 1.4.4 Trends of ML and NLP

In recent years, machine learning and natural language processing have made significant strides, spurred by the creation of novel pretrained language models and the incorporation of linguistic knowledge. Owing to these developments, NLP applications have become more accessible, efficient, and cost-effective.

Language models that have already been trained, such as ULMFiT, CoVe, ELMo, OpenAI GPT, BERT, OpenAI GPT-4, XLNet, RoBERTa, and ALBERT, have revolutionized the NLP landscape. These models let huge amounts of text data be used to train large-scale language representations, which can then be fine-tuned for specific NLP tasks. This method substantially reduces the need for extensive task-specific labeled datasets and enables adaptation to new tasks to occur more quickly. These pretrained models have shown remarkable performance in a variety of NLP tasks, including text classification, named entity recognition, question answering, and machine translation [105].

Additionally, the incorporation of linguistic knowledge and human expertise has the potential to improve the performance of NLP models and systems. Linguistics can shed light on the structural and semantic aspects of a language, thereby enhancing the interpretability and explicability of data-driven approaches. Researchers can enhance the robustness, accuracy, and generalization capabilities of NLP models by incorporating linguistic principles.

Novel approaches to unsupervised and semi-supervised learning have emerged in the field of machine translation, stretching the boundaries between translation quality and efficiency. These methods use large monolingual corpora and unsupervised learning techniques to discover cross-linguistic representations. These methods facilitate text translation without the need for parallel corpora, thereby reducing reliance on costly and time-consuming human translations [105].

In addition to the progress made in supervised learning, there is growing interest in unsupervised machine learning methods across a variety of domains. Specifically, the Local Aggregation approach for object detection and recognition has shown promise in the field of computer vision. This method of learning without being watched tries to get local image statistics and learn how to tell the difference between things without using clear object labels. This approach offers a scalable and cost-effective solution for object detection and recognition [106] by leveraging large quantities of unlabeled data.

In addition, unsupervised learning has led to significant advancements in the field of 3D shape analysis. In [107], unsupervised learning techniques were used to compute the correspondence between 3D geometries. These methods use the structure and geometry of the shapes to make it possible to align and match objects without having to manually add notes or label data. This creates new opportunities for shape comprehension, retrieval, and synthesis.

These recent developments illustrate the ongoing progress and emergent trends in ML and NLP. Using language knowledge, pre-trained language models, and unsupervised learning methods together reveals new ways to do difficult natural language processing tasks, improve the quality of translations, and explore new areas with little labeled data.

Further innovations and applications are anticipated in the fields of ML and NLP, which are expected to stretch the limits of what is currently possible as these domains continue to evolve.

### Chapter Summary and Conclusion

This chapter provides a detailed survey of the latest advancements in ML and NLP, identifying the top performing methods relevant to the recruitment industry. In addition, the optimization of their objective functions has been proposed, enhancing their effectiveness and applicability. The selected ML and NLP techniques were rigorously evaluated for efficacy and relevance. This selection is foundational for the development of advanced algorithms in subsequent chapters, ensuring that the proposed solutions are built on the most efficient and applicable methods.

Furthermore, the proposed optimizations were tailored to improve the accuracy, efficiency, and adaptability of these methods in recruitment scenarios.

In conclusion, this chapter establishes a solid foundation for current ML and NLP technologies, setting the stage for their practical application in transforming recruitment processes. The insights gained here will guide the development of innovative algorithmic solutions in the following chapters, aiming to make a significant impact on the field of recruitment optimization.

## ❖ The Goal of the Thesis is:

> To develop and refine innovative machine learning and natural language processing algorithms to revolutionize the recruitment industry by focusing on enhancing the efficiency and effectiveness of the recruitment process. This goal includes the development of new algorithms for optimized candidate screening, advanced job-candidate matching, and streamlined initial filtration processes, while maintaining computational efficiency and algorithmic simplicity. The aim is to provide a cutting-edge tool for recruitment companies to transform industry practices by leveraging semantic analysis, information retrieval, and recommendation systems.

The research presented in this thesis aims to improve the candidate recruitment process using algorithms that integrate different ML and NLP models. One of the objectives is to reduce the execution time and memory consumption of the candidate filtering process. This is done through two algorithms that we created specifically to reduce the pool of candidates. Another objective is to select the best candidate for a job by predicting the chances of success. Within this, the importance of including text data as a feature in the prediction model is also shown. An objective that is also considered important is the presentation of the main techniques of ML and NLP, and the presentation of some approaches to the formulation of optimization functions within them.

The primary research objectives addressed in this thesis can be divided into two categories:

- The first main goal is to give an overview of the most up-to-date machine learning and natural language processing techniques, as well as different ways to make them work better.
- The second and most important objective is the presentation of algorithms that integrate ML and NLP techniques to optimize the recruitment time and resources.

Of course, a challenging process, such as recruitment, offers many spaces where you can intervene for improvement. This study contributes to some specific nodes of the recruitment process that are often considered bottle necks.

The tasks for which this research provides solutions can be formulated as follows.

**Task 1:** to present state-of-the-art of ML and NLP methods with focus in application of them in recruitment industry.

**Task 2:** to present optimization algorithm for reducing the pool of candidates based on job title similarity and industry relevance by using hybrid ML techniques.

**Task 3:** to present advanced optimization algorithm for reducing the pool of candidates based on job title similarity and industry relevance by using cutting edge ML methods.

**Task 4:** to identify most important features for predicting job success for candidates.

**Task 5:** to create a prediction model for job success that integrates both quantitative and textual data.

**Task 6:** to address problems and challenges related to the application of AI in recruitment and offering an advanced solution for facing these challenges.

Tasks 1 aims to introduce some of the most important ML and NLP techniques. The main focus is on the possible applications and formulation of the optimization problem. Task 2 and 3 aim to create algorithms to reduce the pool of candidates. These aims to exclude unsuitable candidates in the early application phase, facilitating work and effort in other steps of the process. Task 4 aims to extract features which better define potential successful candidates for a job position. Task 5 aims to build a prediction model that is efficient and accurate for predicting the job success of a candidate. It speeds up decision making for the candidate who will be the new employee. Task 6 aimed to shed more light on the recruitment industry in the age of AI. The presentation of trends, challenges, and the future of the industry is the focus.

# Chapter 2

# Streamlining the Candidate Pool: Optimization of Accuracy and Efficiency for Job Title Similarity Problem

In this chapter, the focus is on revolutionizing the candidate selection process in recruitment through the innovative application of ML and NLP. This chapter introduces a novel hybrid method that synergistically combines the k-nearest neighbor algorithm with support vector machines to refine the candidate pool. This approach is centered on leveraging job title similarity and industry relevance to categorize candidates efficiently and accurately. By exploring this hybrid method, this chapter aims to demonstrate how combining these distinct ML techniques can optimize the recruitment process, enhance the precision of candidate job matching, and significantly streamline the pool of candidates based on relevant job titles and industry contexts. This approach promises to be a significant step in making recruitment processes more efficient and effective, offering a detailed exploration of the implementation of the method and its potential impact on the recruitment industry.

- *Problems with Varying Job Titles*

  The multitude of job title variants complicates and diversifies the recruitment process in today's dynamic labor market. Conventional keyword-based approaches have difficulty capturing the subtle nuances and contextual relationships inherent in job titles. Consequently, candidates with the necessary qualifications may go unnoticed owing to variations in the wording of job titles. To address this critical issue, our research employs advanced NLP techniques to incorporate job title descriptions, which enables us to discover semantic similarities and contextual information.

- *Word2Vec, BERT, and KNN: Embracing NLP's Power*

  This research encompasses Word2Vec, BERT, and $k$-Nearest Neighbours, which are all transformative NLP techniques. The potent Word2Vec technique is employed. It captures word embeddings and enables the quantification of semantic relationships between job titles. However, this research goes beyond a simple keyword analysis. Recognizing that job titles within the same industry share similar skill requirements and responsibilities, it introduces industry embeddings that encapsulate the essential characteristics of each industry. These industry embeddings, coupled with job title embeddings from Word2Vec, produced a comprehensive label that captured the industry similarity between job titles. In this context, $k$-NN is used to define the label if two job titles are similar based on the industry context, laying the groundwork for the SVM classification model that will be used later.

  BERT's contextual embeddings enable us to delve deeper into the intricate semantic relationships among job titles. With BERT's exceptional ability to comprehend context, it can identifies job title substitutes—titles that may not be identical but serve as viable alternatives to specific job positions. This acquired knowledge optimizes the recruitment process by focusing on targeted and relevant candidate searches.

- *Increasing Recruitment Effectiveness*

  This research aims to maximize recruitment efficacy by precisely matching job titles to candidates' skill sets and experiences. Utilizing industry embeddings, Word2Vec, BERT, and KNN strategically narrowed the candidate search space to include only those with pertinent qualifications. This approach ensures that candidates are accurately matched to job requirements, saving valuable time and resources for recruiters and candidates. Moreover, the SVM and XGBoost classification models contributed to the optimization of efficiency by delivering rapid and precise results. Armed with data-driven insights, recruiters can make informed decisions in a timely manner, thereby elevating the recruitment process.

- *Summary*

  Using advanced NLP techniques, industry embeddings, and classification models, this chapter describes a ground-breaking plan to change the way candidates are chosen and how quickly they are hired. Using Word2Vec, BERT, and KNN, it is determined the semantic relationships between job titles and optimized candidate selections based on industry similarity. Our research revolutionizes the recruitment process, ushering in an era of data-decision-making, and ensuring a perfect match between job titles and candidate abilities. With this comprehensive strategy, it intends to streamline the candidate pool, improve the quality of candidate evaluation, and create a streamlined and effective recruitment process that connects the most qualified candidates with ideal job openings.

## 2.1 Dataset Description and Relevance

In this section, it is provided a comprehensive overview of the datasets used in our research and explain their significance in narrowing the candidate pool through job title similarity and efficiency optimization. Our datasets include the names of employment titles and industries, as well as their corresponding brief descriptions obtained through web scraping. These datasets serve as the basis for our research, allowing us to extract valuable insights using the sophisticated NLP techniques Word2Vec and BERT for effective candidate selection.

**Job Title Dataset:**

This dataset comprised approximately 80,000 unique job titles. There is a brief description of each job title that offers additional context and context-appropriate information about the roles and responsibilities associated with that particular job. These job titles were obtained through the extensive web scraping of reputable job portals, corporate websites, and career platforms. The descriptions have been carefully checked for veracity and relevance.

**Industry Dataset:**

In addition to the dataset of job titles, here is compiled information on approximately 700 industries. There is a brief description of each industry entry that offers important details about the sector's characteristics and the kinds of employment opportunities it provides. Web crawling was used to compile industry-related data from a variety of reputable sources, including industry-specific websites, market research reports, and official publications.

**Relevance of datasets:**

The datasets are of utmost significance to our research because they provide the foundation for identifying job title similarities and optimizing the candidate pool. The primary characteristics that highlight the significance of our datasets are as follows:

1. Semantic context and skill correlations

   Brief descriptions affixed to each job title provide an essential semantic context by expanding the roles and responsibilities associated with specific positions. This context enabled us to identify intricate skill correlations inherent in job titles. As a result, our research goes beyond conventional keyword-based matching and delves into the essence of job requirements, resulting in more precise candidate selection.

2. Industry-Specific Insights

   The industry descriptions provide valuable insights into the characteristics of various sectors and the employment opportunities they offer. By incorporating industry data, it enriches the semantic relationships between job titles, considering industry-specific skill requirements and contextual nuances. This allows us to optimize the selection of candidates for industry-specific positions.

3. Word2Vec Embeddings

   The Word2Vec embeddings derived from our datasets served as the basis for identifying semantic similarities between job titles. By transforming job title descriptions into high-dimensional word embeddings, it measures the semantic relationships and identify job titles that share characteristics beyond ordinary keyword matches. This inventive method improves the accuracy of candidate selection by matching candidates with job titles that are a seamless fit for their skill sets and experiences.

4. BERT Embeddings

   In conjunction with word2vec embeddings, our datasets were instrumental in the generation of BERT embeddings, an innovative NLP technique that captures contextual relationships between words. By incorporating BERT embeddings, it investigates the concept of job title substitutes, titles that may not be identical but are context-appropriate for particular job positions. This profound insight revolutionizes the effectiveness of candidate search by ensuring that recruiters consider viable alternatives, thereby conserving valuable time and resources.

5. Data-driven Decision-Making

   Our datasets facilitate data-driven decision making during the recruitment process. Based on our research on vast and pertinent datasets, it ensures that candidate selection is no longer solely based on subjective judgment but rather on data-supported insights. This data-driven methodology promotes objectivity, productivity, and precision in recruitment.

In conclusion, the job title and industry datasets serve as the foundation of our research, allowing us to realize the full potential of NLP techniques and sophisticated ML models. The deep semantic context, skill correlations, industry-specific insights, and word embeddings make it easier to choose the best candidates, speed up the hiring process, and encourage making decisions based on data. With these datasets as our compass, it embarks on a transformational voyage to redefine candidate selection by matching the right talent with ideal job openings.

## 2.2 Candidate Pool Streamlining Algorithm: Harnessing Job Title Similarity for Efficient Recruitment

In recent years, the employment market has become more globalized, resulting in substantial expansion of the recruitment industry. In addition to its numerous benefits, globalization has introduced obstacles, most notably in the processing of massive amounts of data to identify the best candidates for specific occupations. Traditional candidate selection methods, which rely on the manual analysis of characteristics such as industry, job title, experience, and talent, have proven to be inefficient and expensive. One-third of talent acquisition professionals spend more than 20 hours sourcing candidates for a single position [108]. Talent acquisition professionals spend more than 30 percent of their workweeks sourcing candidates for a single position. In addition, the average American business spends $4,000 on engaging a new employee [109] and [110]. These statistics emphasize the need for more effective and economical recruitment procedures.

The traditional recruitment process has a negative impact on a number of economic factors for both candidates and employing companies. There are bottlenecks when working with hiring managers, with 50 percent of recruiters encountering problems in moving candidates through the recruiting process and forty-four percent citing hiring managers' resume reviews as reasons for slowing the process [111]. 67% of the recruiters [112] cite the scarcity of competent and highly qualified candidates as their greatest hiring obstacle. The adoption of machine learning methods and natural language processing algorithms has gradually replaced traditional processes with more efficient AI-powered alternatives. In 2019, 64% of talent acquisition professionals were budgeted for AI-powered recruitment tools in 2019 [108].

Artificial intelligence and machine learning enable rapid analysis of enormous amounts of data, allowing data-driven decisions and forecasts. As a result, recruiters embrace AI to characterize job postings as “ideal fit”, interact with prescreened talent, and accelerate the hiring process. AI makes recruiters more productive and expedites the recruitment process by eliminating manual tasks [108].

The recruitment industry’s application of machine-learning techniques is diverse and multifaceted. AI-driven methods are revolutionizing talent acquisition on several fronts, including ensuring the best match for positions, enhancing online applications, and targeting candidates more precisely. In addition, AI-driven interviewing techniques and automation of complex data analytics are gaining ground in contemporary recruitment practices. Numerous studies have investigated the potential of various machine learning techniques, confirming the industry’s commitment to leveraging AI and machine learning to optimize talent acquisition [113], [114], [115], [116], [117], [118] and [119].

### 2.2.1 $k$-NN and SVM methods

Machine learning techniques are widely utilized to accomplish the primary tasks of Natural Language Processing. In general, ML techniques can be categorized into three groups [115]: (1) supervised machine learning, (2) semi-supervised machine learning, and (3) unsupervised machine learning [117]. The most common method for performing NLP tasks is supervised machine learning, where models automatically extract rules from training data [115]. The classifications for supervised ML are (i) sequential and (ii) nonsequential. Support Vector Machines are non-sequential supervised ML techniques, whereas k-nearest neighbors are sequential supervised ML techniques. According to [113], k-NN and SVM are the most popular classifiers for solving the feature selection problem, accounting for 42% and 29% of all the available classifiers, respectively.

**A. $k$-Nearest Neighbors**

The $k$-Nearest Neighbor algorithm is a simple yet effective instance-based supervised learning method frequently used for predictive analysis. The $k$-NN algorithm considers the nearest neighbor instances based on a predefined distance metric when classifying an instance. The majority class of the instance's $k$ adjacent neighbors determines its class. To prevent ties, it is crucial to choose an appropriate value for $k$ to avoid multiples of the number of classes. Larger $k$ values can reduce the effect of noise on classification; however, they can also obscure class boundaries [117].

The efficacy of $k$-NN is highly dependent on the distance measure used to calculate the distances between instances [115]. $k$-NN is a non-parametric technique, which means that the algorithm can classify without making assumptions about the function $Y = f(x_1, x_2, \ldots, x_n)$, that links class $Y$ to attributes $x_j$, $(1 \leq j \leq n)$.

The weaknesses of the algorithm are as follows [114]:

- The $k$-NN algorithm is sluggish because it examines each instance individually.
- The algorithm is susceptible to dimensionality plague.
- This algorithm considers irrelevant and correlated attributes.
- Poor selection of the distance or $k$ value degrades efficacy.

Numerous researchers have attempted to address the aforementioned shortcomings of the $k$-NN algorithm [120]. For example, weights are added to the traditional $k$-NN algorithm, and the W$k$-NN algorithm is described in [121]. The weights are computed using the $K$-kernel function. Such functions are intended to weigh observations relative to a reference point, so that the closer an instance is to the reference point, the more weight it will receive. In text classification, a novel $k$-NN model enhanced by clustering was implemented [122]. The $k$-means algorithm is used to cluster the training instances of each class. The resultant cluster centers were used to create a new training dataset.

Each training instance was assigned a weight value proportional to its significance. In [123], an idea for a real-time way to find deluge attacks using genetically weighted $k$-NN for anomaly detection was put forward. An unsupervised clustering algorithm is used to cut down on the number of instances in the sampling dataset, speed up training and runtime, and make the system more accurate overall.

### B. Support Vector Machines

AT&T Bell Laboratories created Support Vector Machines, a classification technique, for practical applications. SVMs are frequently employed for binary classification, both linear and nonlinear, with known class labels. This method sorts data into groups by drawing a decision boundary (hyperplane) using support vectors, which are a small group of feature vectors. SVM is capable of supporting vector classification, support vector regression, and the detection of outliers. Regression and classification can be used for both linear and nonlinear data. SVM can be either 1) simple SVM, which is typically used for linear regression and classification problems, or 2) kernel SVM, which is typically applied to nonlinear data.

Typically, an SVM is used to identify intricate relationships between datasets. It is typically applied to a few datasets with a large number of features. The ability of the SVM to manage small, complex datasets allows it to produce more accurate results than other algorithms. Several advantages of the SVM should be highlighted (see [118]).

Efficacy of high-dimensional datasets containing numerous features.

The algorithm is effective when the number of features exceeds the number of training examples in the data sample.

Support vectors are a subset of the training examples used by an SVM in its decision function. This characteristic makes the SVM a memory-efficient algorithm. Outliers are less influential. A variety of standard or customized kernel functions may be specified for the decision function. The SVM is the optimal method when classes are distinct.

Several disadvantages of SVM [118] must be mentioned.

- SVMs cannot explicitly estimate the probabilities.
- SVM performed best on small datasets because it required extensive training time for larger datasets.
- Selecting an appropriate kernel function is challenging.
- When overlapping classes are present, the SVM performs poorly.

The linear kernel function is the most widely used kernel function.

$$f(x, w, b) = W.X + b = 0$$

$W = [w_1, w_2, w_3, \ldots, w_n]$ is a weight vector, $n$ is the number of attributes, $X$ is the sample to be classified, and $b$ is a scalar, also known as bias. The preceding equation can be expressed as $w_0 + w_1 x_1 + \ldots + w_n x_n = 0$, where weights $w_0$ and $X = [x_1, x_2, \ldots, x_n]^T$ represent $b$.

Text-classification problems are typically linearly separable; therefore, linear kernel functions are ideally suited for text classification. Text classification problems benefit from linear kernel functions when there are numerous features, as in the case when there are many features. The most essential factor in this instance is that linear kernel functions are faster than other kernel functions and that there are fewer optimization parameters.

Other common kernel functions include the following:

$$f(X_1, X_2) = (a + X_1 T.X_2)b,$$

where $X_1$ and $X_2$ represent the data sample and polynomial kernel functions, respectively.

$f(X_1, X_2) = \exp(-\gamma \|X_1 - X_2\|^2)$, where $\gamma = 1/(2\sigma^2)$ and $\sigma$ is the variance. $\|X_1 - X_2\|^2$ is the squared Euclidean distance between the two feature vectors.

Sigmoid kernel function: $f(X, y) = \tanh(\alpha X^T.y + C)$, where $C$ is an offset to accommodate data misclassification and $\alpha$ is a weight vector.

Polynomial kernels are infrequently used because they are less computationally efficient than the other kernels. Moreover, these predictions were not particularly accurate. RBF is typically applied to nonlinear data. SVM can utilize numerous alternative kernels. Typically, this is due to error constraints, fine-tuning of the parameters, or acceleration of the training time. Numerous NLP tasks such as content arrangement, Part Of Speech, Name Entity Recognition, and segmentation employ SVMs. For Bengali named entity recognition, [72] used a combination of Support Vector Machines and Conditional Random Fields. In [116], an SVM-based POS tagger for the Malayalam language was developed.

### 2.2.2 Optimization Method I: Maximizing the precision of job similarity

This section describes a proposal for the hybrid application of the SVM and k-NN methods to locate similar job titles. Before explaining the hybrid application of the two methods from a technical standpoint, let's examine some job-title-related characteristics that can be viewed as points of similarity.

The primary characteristic of a job title is its definition, which broadly describes what it entails. There are many additional job-related details that are unclear from the definition. This indicates that comparing job titles based solely on their definitions is not the most effective method. Accuracy issues begin to appear when the definition is not explicitly stated. In addition, given that this procedure is applied each time a candidate is needed to fill a vacancy, the computational complexity increases swiftly.

A specific job title is also associated with a variety of duties and responsibilities that the candidate must perform in the course of his daily activities. Some job titles clearly differentiate duties and responsibilities, making it simple to determine whether they are interchangeable. This method considers the fact that many job titles are comparable because of their shared duties and responsibilities. This similarity renders the "area" of responsibilities and duties a "gray area." This implies that a substantial portion of the duties and responsibilities of the two distinct job titles can be the same. In other words, they are interchangeable job titles.

Obviously, more factors than just the job title affect a candidate's profile. There are numerous other variables, such as years of experience, specialized talent, particular projects, and companies. Standard recruitment procedures dictate that the best candidate

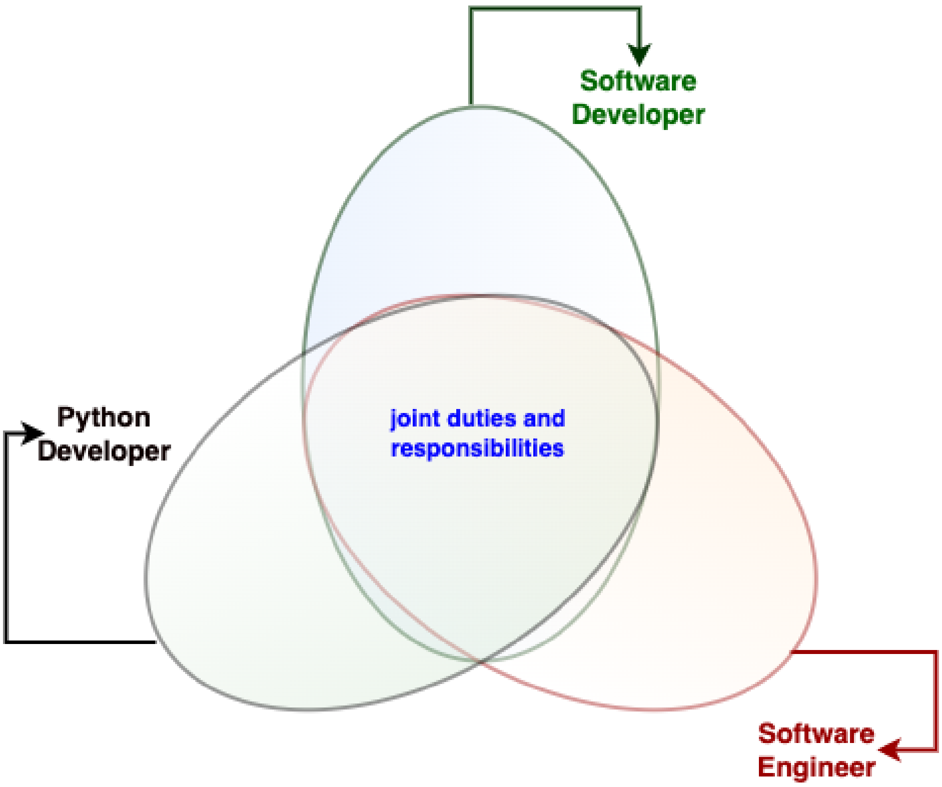


Figure 3: Example of similar job titles

for a vacant position is the one whose profile most closely matches the aforementioned factors. In this instance, are considered the candidate to be a good fit for the position despite their similar job titles.

The hybrid method seeks to improve the process of locating job titles similar to a given title. This is accomplished by integrating two crucial factors: optimizing execution time and memory allocation and enhancing the precision of similarity measurements.

The hybrid application combines two machine learning techniques: Support Vector Machines and $k$-Nearest Neighbors.

Optimization of Execution Time and Memory Allocation: During the computation of job title similarity, the hybrid method optimizes the execution time and memory allocation. It can take a lot of computing power to process large datasets and feature vectors. The hybrid strategy uses useful data structures and algorithms to make the processing easier. The hybrid method attempts to establish a balance between computational efficiency and accuracy by combining the strengths of SVM and $k$-NN.

Improving Similarity Accuracy: Accurate measurement of job title similarity is essential for the success of the hybrid method. SVM and $k$-NN have strengths in that they complement one another in capturing various aspects of similarity. An SVM is ideally suited for handling high-dimensional feature spaces because of its ability to identify complex relationships between data. The instance-based learning method of $k$-NN, on the other hand, does a great job of representing local data and dealing with nonlinear patterns.

The hybrid method incorporates the advantages of SVM and $k$-NN to provide a more exhaustive and accurate representation of job-title similarity. The hybrid approach is better at matching job titles because it takes the best parts of both methods and puts them together. This is especially true when SVM or $k$-NN alone might not work.

Overall, the hybrid method provides a well-balanced solution that capitalizes on the strengths of SVM and $k$-NN, while also resolving their respective limitations. This synergy between the two techniques enables a more efficient and accurate computation of job title similarity, which is essential for a variety of applications, including talent acquisition, job recommendation systems, and workforce planning.

The data utilized in the application's initial phase consisted of job titles and industry descriptions.

These data were preprocessed to generate a final dataset, which was represented by an embedding vector containing 100 entries for each occupation and industry. Word2Vec is a method used for creating this type of embedding. Word2vec, a technique for natural language processing, was proposed in 2013. This algorithm learns word associations from a massive corpus of text by using a neural network model. Once trained, such a model can identify synonyms and suggest additional terms for sentence fragments. As its name suggests, Word2vec represents each distinct word with a vector, which is a specific list of integers. [124]

Two methods are available for obtaining embedding: Skip Gram and Common Bag Of Words (CBOW). Skip-gram Word2Vec is a computing architecture for word embedding.

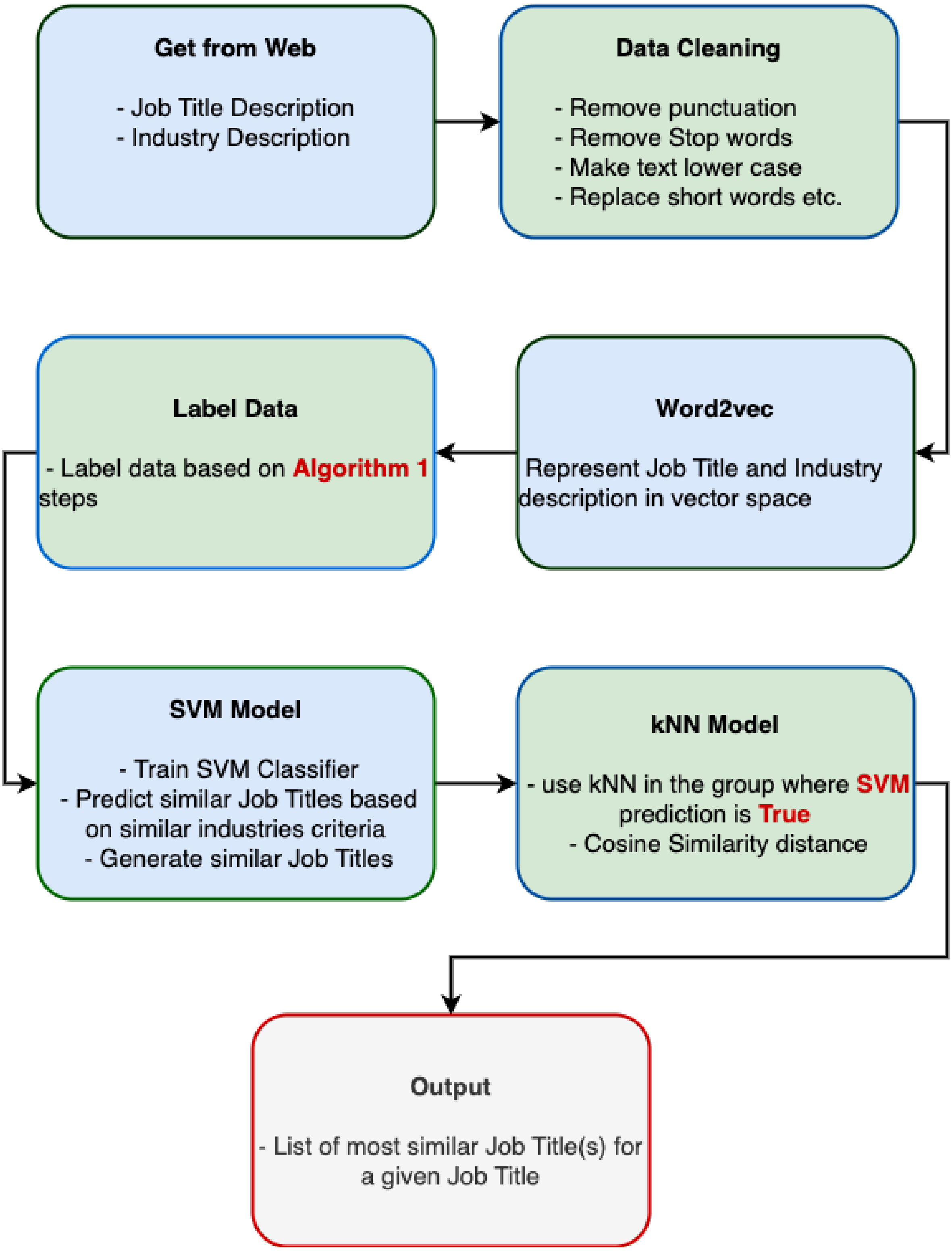

Get from Web
- Job Title Description
- Industry Description
Data Cleaning
- Remove punctuation
- Remove Stop words
- Make text lower case
- Replace short words etc.
Label Data
- Label data based on Algorithm 1 steps
Word2vec
Represent Job Title and Industry description in vector space
SVM Model
- Train SVM Classifier
- Predict similar Job Titles based on similar industries criteria
- Generate similar Job Titles
kNN Model
- use kNN in the group where SVM prediction is True
- Cosine Similarity distance
Output
- List of most similar Job Title(s) for a given Job Title

Figure 4: Optimization Method I: Based on Hybrid Approach I

Instead of using surrounding words to predict the center word, as CBow Word2Vec does, skip-gram uses only the center word. Word2Vec predicts the surrounding syllables based on a central word.

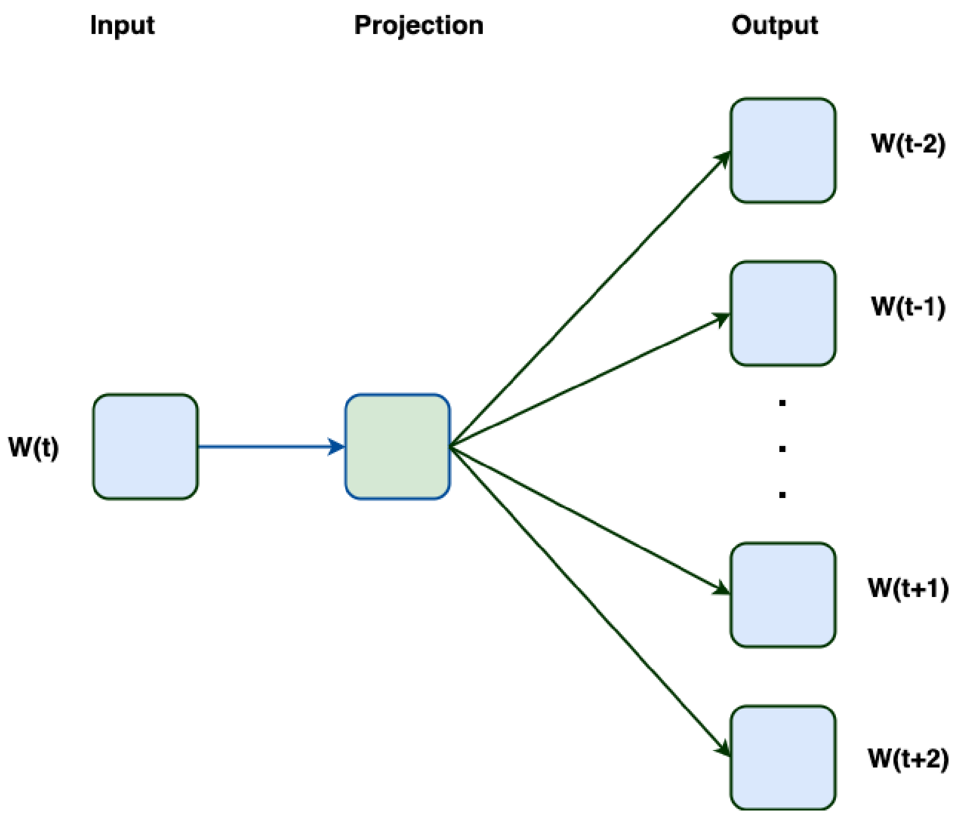


Figure 5: Skip-gram flow

The Skip-gram objective function produces the following objective function [124] by adding the log probabilities of $n$ words to the left and right of the target word $w_t$.

$$J_\theta = \frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-n\leq i\leq n\\ i\neq 0}} \log P(w_{j+1}|w_t), \tag{2.1}$$

where $\theta$ denotes the vector representation of each word.

The vectors are carefully chosen so that a simple mathematical function (the cosine similarity between vectors) can show how semantically similar the words that these vectors represent are. In document query situations, a document can be represented as a word vector whose dimensions correspond to the document's available words [119]. The value of a dimension is the number of times a word appears in a document. As a vector form, a document can be described as follows:

$$\vec{d} = (w_{d_0}, w_{d_1}, \ldots, w_{d_k}).$$

Similar to the document, the query of a word can be described as a vector form as follows:

$$\vec{q} = (w_{q_0}, w_{q_1}, \ldots, w_{q_k})$$

where $w_{d_i}$ and $w_{q_i}$, $(0 \leq i \leq k)$ are float numbers representing the frequency of each word in a document, and each vector dimension corresponds to a word in the document. Using vector similarity, it defines the similarity between two vectors as [125]:

$$\mathrm{Sim}(\vec{d}, \vec{q}) = \frac{\vec{d}\vec{q}}{|\vec{d}||\vec{q}|} = \frac{\sum_{k=1}^{t} w_{q_k} w_{d_k}}{\sqrt{\sum_{k=1}^{t}(w_{q_k})^2}\sqrt{\sum_{k=1}^{t}(w_{d_k})^2}}.$$

The most straightforward approach for identifying job titles similar to a given query involves calculating the cosine similarity between their corresponding vectors. However, this method presents two primary challenges that must be addressed for effective implementation.

- *Quadratic Execution Time*

  One of the challenges is the execution time required for this approach. When dealing with a large number of job titles, the process of calculating cosine similarity between all pairs of vectors becomes computationally demanding. As the number of job titles increased, the execution time increased quadratically, resulting in longer processing times. This is particularly problematic in scenarios where real-time or near-real-time results are necessary.

- *Memory Usage Complexity*

  Memory usage complexity is another significant issue, in addition to execution time. Storing and managing a substantial number of vectors for all job titles requires a significant amount of memory. This can strain the computational resources of the system, potentially leading to memory bottlenecks and slower performance.

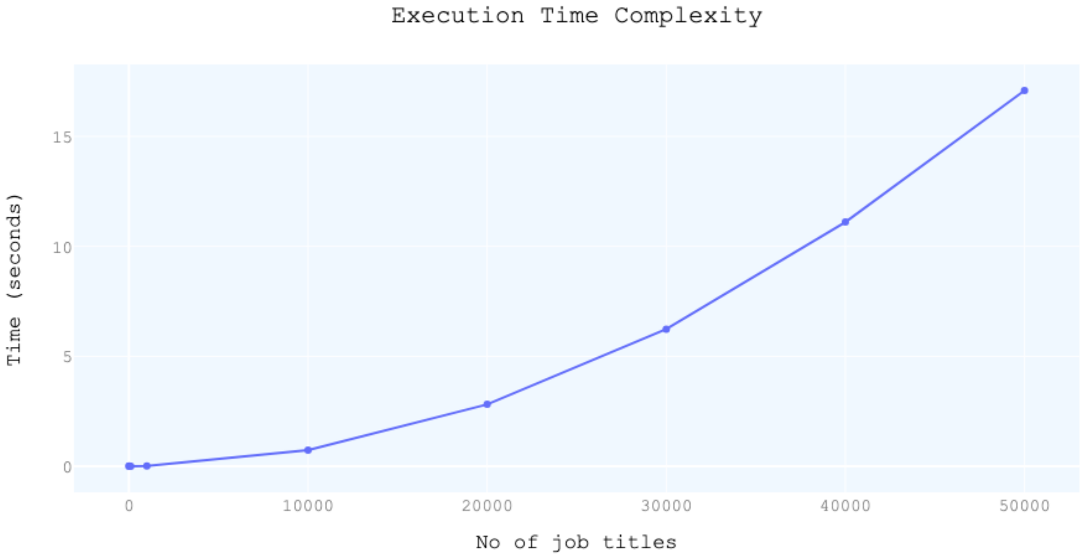


Figure 6: Execution time complexity

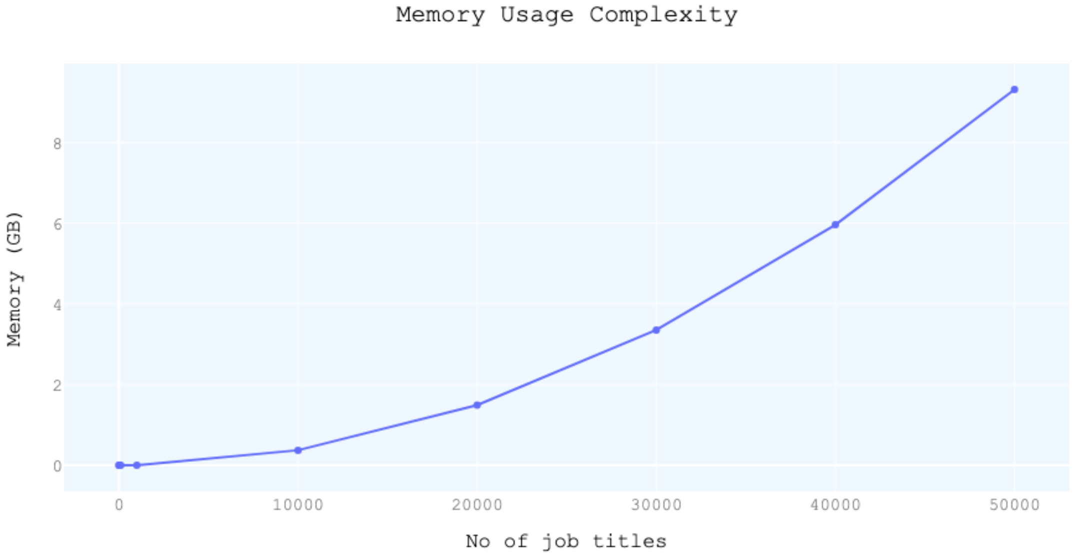


Figure 7: Memory Usage Complexity

*When measuring the cosine similarity (specifically, using the* ***Intel Core i9-10850K*** *processor), the computation time varied. It takes about* ***0.001 seconds*** *to calculate* *the similarity between a single pair of job titles, while this increases to* ***12.3 seconds*** *when processing 50,000 pairs of job titles.*

*With regard to memory utilization, the outcomes are equally noteworthy. Allocating approximately* ***120 B*** *of memory was required to determine the similarity between individual pairs of job titles. However, this memory allocation escalated significantly to approximately* ***19GB*** *when processing 50,000 pairs of job titles simultaneously.*

As depicted in the graphs, it becomes evident that as the number of job titles increases, the complexity of both execution time and memory usage also increases significantly. This observation underscores the suboptimal nature of this simplistic approach. Our solution, which entails the hybrid implementation of the SVM and $k$-NN methods, is designed to not only enhance speed but also to markedly alleviate memory usage intricacies.

With the vector embeddings for job titles and industries established via Word2vec, the subsequent steps involved employing the ensuing algorithm to construct a labeled dataset. This dataset serves as the foundation for training the SVM model.

**Algorithm 1**

```
Step 1: Calculate the similarity between Job Title and Industry using Cosine Similarity.
J = [Job title embedding]
I = [Industry embedding]
similarity = Cosine(J,I)
Step 2: Select $n$ most similar industries for a job title.
Top_n (J)
Step 3: Find the joint industries of every pair of job titles by taking the intersection of the most similar industries for any pair of job titles.
joint_industry = $x \cap y$, where $x \in$ Top_n$(J_x)$, $y \in$ Top_n$(J_y)$
Step 4: Classify job titles industry similarity as 0 or 1 based on the threshold, where we compare $\ell$, which indicates the number of joint industries between two job titles, with $t$, the criterion that determines how many joint industries must have two job titles to be called similar.
industry_similarity = $\begin{cases} 1, & \text{if } \ell \geq t \\ 0, & \text{otherwise} \end{cases}$
where $\ell = \text{length}(x \cap y)$ and $t :=$ threshold.
```

Typically, a user has the flexibility to determine the threshold value t based on specific requirements. An informed recommendation guided by heuristic models suggests that this criterion could encompass approximately 1-2% of the total number of industries. This would result in the inclusion of approximately 5 to 7 shared industries from the upper echelons of the industry list corresponding to each job title.

Algorithm 1 is instrumental in creating a labeled dataset, wherein the embedding vectors for each job title constitute the features, whereas the classes generated in Step 4 serve as labels. Subsequently, Algorithm 2 was implemented to select the k most similar job titles for a given query job title.

**Algorithm 2**

Consider the job title $J_1$.

**Step 1:** Apply SVM on the embedding vector of $J_1$ and the embedding vectors of the remaining job titles of the dataset.

**Step 2:** Create a reduced dataset with the $n$ job titles where SVM outputs 1 for the classification of industry_similarity.

**Step 3:** Apply k-NN method to the reduced dataset to find the $k$ job titles most similar to $J_1$. The distance we use will be the cosine similarity.

**Step 4:** Show $k$ job titles most similar to $J_1$.

### 2.2.3 Example of Results of Hybrid Approach I

To illustrate the efficacy of the innovative hybrid machine-learning approach, consider a scenario in which a recruiter aims to hire a data scientist while keeping the search broadened beyond this particular query. The application involves models trained on a dataset comprising *500 job titles* and *700 industries*. This dataset was acquired from the Internet and constitutes the initial phase of the methodology. After the extraction, the data underwent a series of preprocessing procedures. Consequently, each job title was transformed into an embedding vector with *100 dimensions*. At this juncture, the data stands were prepared for engagement with the machine-learning models.

As described in the first step of Algorithm 1, our procedure begins by calculating the degree of similarity between each job title and industry. The primary goal of this stage was to compile a list of relevant fields for each occupation. Step 2 of Algorithm 1 takes this list of related industries for each job title and uses it to determine the *10 industries* that are most similar. It is worth noting that the selection of the illustrative example's top industry count is the result of testing and heuristic methods. Diverse uses should account for no more than 2% of the total number of industries.

As a result, it is a document in which every occupation is associated with a group of 10 relevant fields of work. Following this step, is determined the intersection of the industry lists for each pair of job titles, as described in Step 3 of Algorithm 1. The result of this effort was a symmetrical matrix with numeric values between 0 and 10.

$$\begin{bmatrix} 10 & 1 & 1 & \dots & 1 & 1 & 0 \\ 1 & 10 & 4 & \dots & 0 & 0 & 1 \\ 1 & 4 & 10 & \dots & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & \dots & 10 & 1 & 1 \\ 1 & 0 & 0 & \dots & 1 & 10 & 1 \\ 0 & 1 & 1 & \dots & 1 & 1 & 10 \end{bmatrix}$$

Moving on to Step 4 of Algorithm 1, it initially establishes the threshold as $t = 2$. It is crucial to emphasize that this threshold value, as exemplified in this demonstrative scenario, is an outcome of heuristic methodologies. Naturally, users retain the flexibility to employ distinct values based on the significance they assign to an industry in the context of candidate selection.

This threshold delineates a binary classification scheme, attributing values of either 0 or 1 to each pair of job titles. The outcome of this process results in a dataset that is suitably labeled. With the labeled dataset prepared, the SVM model was trained using the training data and then subjected to testing with the designated testing segment of the data. The evaluation of the performance of the model is presented in the following sections.

Table 1: Model Classification Report (SVM)

The Evaluation of the Model

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.92 | 0.87 | 0.89 | 1243 |
| 1 | 0.87 | 0.92 | 0.89 | 1208 |
| accuracy | | | 0.89 | 2451 |
| macro avg | 0.89 | 0.89 | 0.89 | 2451 |
| weighted avg | 0.89 | 0.89 | 0.89 | 2451 |

The derived confusion matrix determined that the number of false positives and false negatives was relatively small.

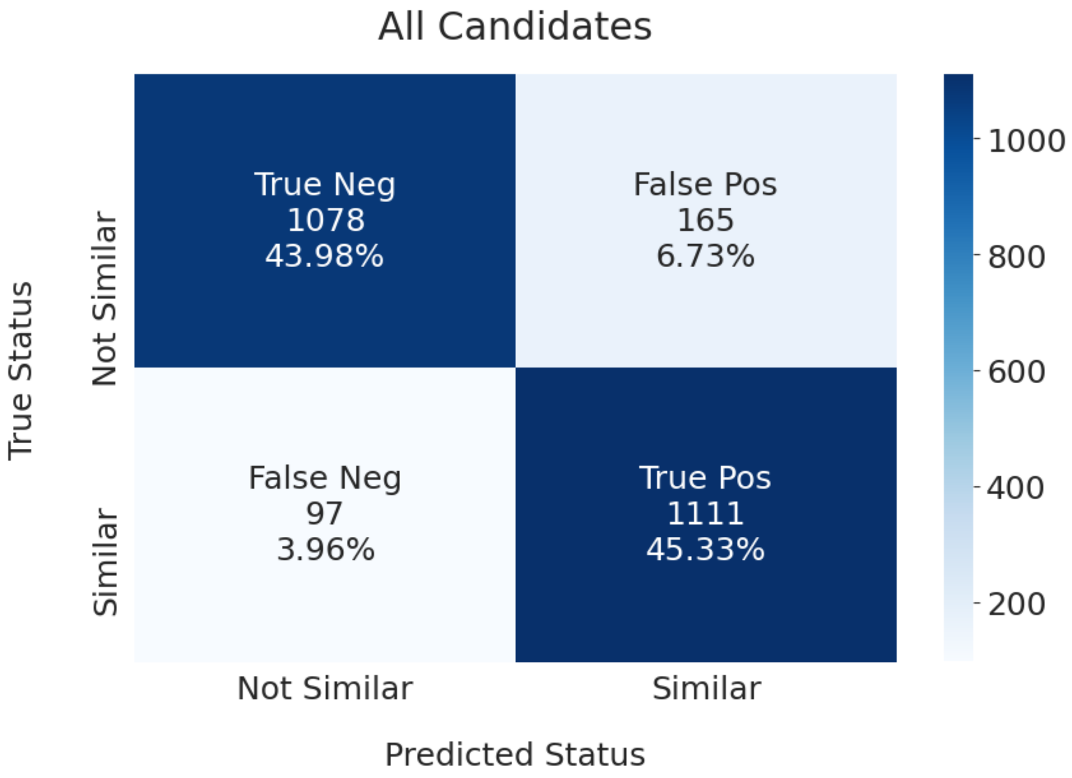


Figure 8: Confusion Matrix for Job Title similarity prediction - SVM model

The accuracy of the model was computed to be 89.3%. As the culmination of this process, Algorithm 2 is then applied, yielding a listing of the 10 most similar job titles achieved through the hybrid methodology. For instance, considering the query job title “data scientist,” Table 2 presents the results. If the objective is to recruit a candidate with the specified job title “data scientist,” the algorithm would not only propose candidates explicitly holding this job title but also those with closely related job titles, as displayed in the second column of Table 2. The similarity score, which ranges from 0 to 1, quantifies how closely aligned the alternative job title is in relation to the query title.

### 2.2.4 Conclusion and upcoming work

The outcomes illustrated in the table encapsulate the conclusive results obtained through the implementation of this hybrid methodology. In the context of a job vacancy inquiry, the algorithm identifies potential alternative job titles that can serve as substitutes. When

Table 2: Results of substitute Job Titles for “Data Scientist”

DATA SCIENTIST MOST SIMILAR JOB TITLES

| Query | Most Similar Job Titles | Similarity Score |
|---|---|---|
| data_scientist | data_scientist | 1 |
| | data_science_consultant | 0.96577 |
| | senior_data_scientist | 0.95804 |
| | lead_data_scientist | 0.95755 |
| | data_science_fellow | 0.95331 |
| | data_science_instructor | 0.95038 |
| | data_science_lead | 0.94927 |
| | data_science_mentor | 0.93409 |
| | machine_learning_engineer | 0.93386 |
| | statistical_modeler | 0.92812 |

this dynamic feature is added to a recruitment platform, it greatly improves the accuracy of choosing candidates by using the basic idea of job-title similarity. A noteworthy facet of this approach lies in its translation of similarity into a quantifiable numerical value, providing users with the autonomy to establish the most suitable similarity threshold.

This novel hybrid application, which merges the SVM and k-NN methods, enhances the efficiency of the execution time and memory usage in locating similar job titles for a given query. As depicted in the workflow of the application, the initial dataset for the subsequent phase is derived from the classification of the relevance of job titles based on their similarities to industries. This decision significantly reduces both the execution time and memory usage by a factor of approximately 80. Subsequently, k-NN was employed on this refined dataset to generate a list of the most similar job titles to the query job title.

Looking ahead, further sections explore other high-performance methods for candidate selection. Beyond considering basic job title information, this process aims to incorporate additional features, such as specific job skills and the candidate’s work history with previous companies.

## 2.3 Optimization Method II: For finding job title(s) best similar to a given job title

### 2.3.1 Description of Machine Learning and Natural Language Processing Models

**BERT**

BERT, which stands for Bidirectional Encoder Representations from Transformers, is a groundbreaking NLP model that has redefined the landscape of language understanding tasks. Developed by Google AI researchers [25], BERT leverages a transformer architecture, a deep neural network framework specifically designed to process sequential data, making it highly effective for tasks involving language understanding and generation.

BERT is pre-trained on an extensive corpus of text, including Wikipedia (~2.5 billion words) and the Book Corpus (~800 million words). This pre-training process is unsupervised and aimed at learning the statistical properties of language. During this process, BERT learns to represent words, phrases, or sentences in arrays of length 768.

Traditional NLP models often struggle with capturing contextual nuances due to their unidirectional nature. BERT addresses this limitation by introducing bidirectionality. Unlike previous models that process text linearly, BERT employs a bidirectional training approach. This innovative strategy allows BERT to consider both preceding and succeeding words when understanding the context of a given word. As a result, BERT can better grasp intricate relationships between words and phrases, significantly enhancing its ability to comprehend the subtleties of language.

The transformer architecture is what holds BERT together. It has two main parts: the self-attention mechanism and feedforward neural networks. The self-attention mechanism mathematically calculates contextual relationships through equations like:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where $Q$, $K$, and $V$ are matrices of query, key, and value vectors, respectively, representing words, and $d_k$ is the dimension of key vectors. Additionally, the multi-head attention mechanism combines multiple of these attentions, resulting in enriched representations:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_0$$

BERT's success rests upon its sophisticated pre-training and fine-tuning methodology.

1. *Pre-Training Loss:* In the pre-training phase, BERT learns by predicting masked words in sentences. The loss function, often cross-entropy loss, is utilized:

   $$\mathcal{L}_{\text{pretrain}} = \sum_i \sum_j y_{i,j} \log(p_{i,j})$$

   where $y_{i,j}$ is the true label (1 if the word is masked, 0 otherwise), and $p_{i,j}$ is the predicted probability of correctness.

2. *Fine-Tuning Loss:* For specific tasks, BERT's pre-trained weights are fine-tuned using task-specific data. The loss function varies based on the task. For instance, in binary classification,

   $$\mathcal{L}_{\text{fine-tune}} = -\sum_i (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

   where $y_i$ is the true label (0 or 1), and $p_i$ is the predicted probability of the positive class.

BERT's cutting-edge technical sophistication, which combines creative architectural design with exact mathematical precision, has helped it achieve top-level performance in a wide range of natural language processing tasks. Its amazing ability to understand and interpret language in different situations comes from the way its bidirectional training, transformer architecture, and methodical pre-training and fine-tuning paradigms work together.

### XGBoost

XGBoost, standing for Extreme Gradient Boosting, is a distributed gradient boosting framework introduced with the primary intent of enhancing computational speed and model performance. At its core, the gradient boosting algorithm builds a group of decision trees one after the other, with each tree trying to fix the mistakes made by the ones that came before it [126].

Let us start with a basic representation of a decision tree, which serves as the foundational element of the boosting mechanism in XGBoost.

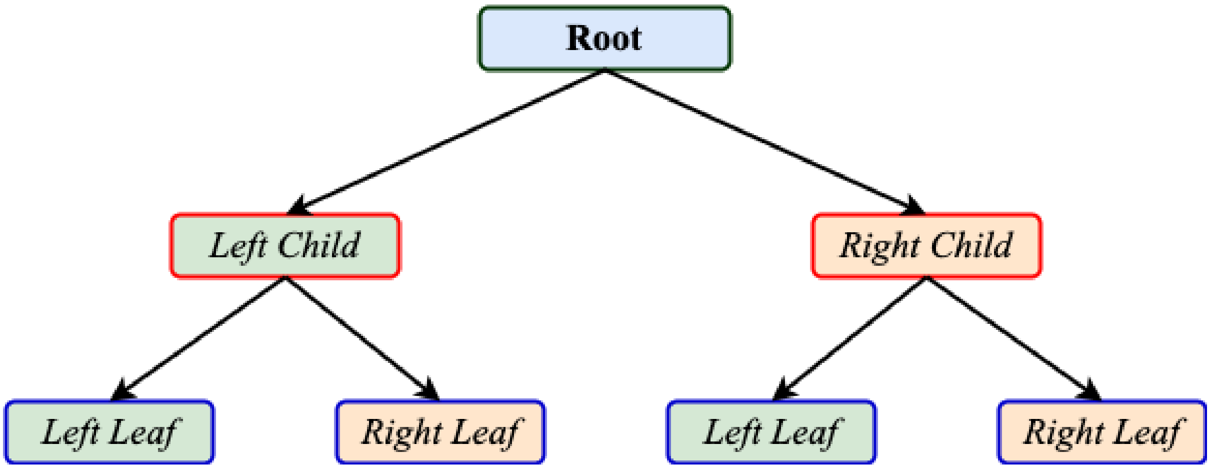


Figure 9: A basic decision tree structure illustrating the foundation of XGBoost algorithm

In this simple tree diagram,

- "Root" represents the starting node, where the initial data split occurs.
- "Left Child" and "Right Child" are the first set of branches, representing the primary binary decisions made from the root node.
- "Left Leaf" and "Right Leaf" are the secondary decisions branching from the first set.

Such trees play a pivotal role in the XGBoost algorithm. Each node in the tree represents a decision based on a certain feature's value, which helps in recursively segmenting the dataset into subsets. XGBoost's gradient-boosting mechanism incrementally builds an ensemble of these trees. It aims to correct the errors of preceding trees by focusing more on instances that were previously mispredicted. Conceptually, it sequentially adds trees:

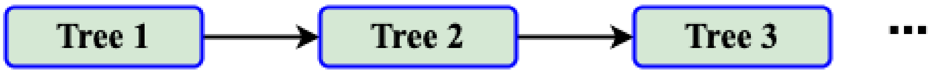
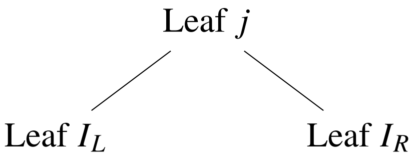


Figure 10: Sequential construction of decision trees in the XGBoost algorithm

Apart from its fundamental tree-construction approach, XGBoost integrates a variety of advanced optimization methods. What sets XGBoost apart from classic gradient boosting frameworks is its utilization of both L1 (Lasso) and L2 (Ridge) regularization in its target function. Faced with sparse data or missing values, XGBoost uses a method that sets a predetermined direction for every tree node. If a specific feature in a node lacks data, XGBoost guides it in this predetermined direction, selected based on maximizing the target function's gain.

The "gain" from a particular split is central to this decision.



$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

where

- $G_L$ and $G_R$: These are the gradient statistics for the left ($I_L$) and right ($I_R$) child nodes after a split. In XGBoost, the gradient statistics are basically the sum of the first-order gradients (which come from the loss function) for each instance that fits into a child node.

- $H_L$ and $H_R$: These are the hessian statistics (sum of second-order gradients) for the left and right child nodes. The Hessian provides a measure of the "curvature" or "second derivative" of the loss function. In a sense, it gives information about how much the predicted scores should change in response to errors. In the XGBoost framework, this can be thought of as a measure of how confident the model is about the predictions for the instances in the respective child nodes.

- $\lambda$ and $\gamma$: These are regularization parameters. $\lambda$ is the L2 regularization term on the leaf weights, which prevents the model from becoming overly reliant on any single leaf and helps in generalization. $\gamma$ is the regularization term for tree depth; increasing it makes the algorithm more conservative and prevents overfitting by effectively imposing a penalty for adding additional complexity to the tree.

The gain formula itself can be broken down into three main components:

1. The first term, $\frac{G_L^2}{H_L+\lambda}$, represents the "score" or "quality" of the left child node after the split.
2. The second term, $\frac{G_R^2}{H_R+\lambda}$, represents the "score" or "quality" of the right child node after the split.
3. The third term, $\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$, represents the score of the original leaf (before splitting).

When calculating the gain of the split, the third term is subtracted from the sum of the first two terms. This difference essentially tells us how much better our model became due to the split. If the gain is positive, it indicates that splitting the leaf (node) improved the model; otherwise, it did not.

Finally, $\gamma$ is subtracted, which can be interpreted as a penalty for adding an additional leaf (i.e., making the model more complex). This ensures that splits which do not provide a sufficient improvement in the objective will not be preferred.

A unique feature of XGBoost is its approach to tree pruning. Rather than the conventional breadth-first method, it employs a depth-first methodology. Trees are grown to a designated maximum depth and then pruned using the "max_depth" parameter. This depth-first strategy differs from the customary breadth-first approach that regular GBMs use.

On the computational front, the parallel processing capabilities of XGBoost are notable. While it doesn't build trees in parallel, it does parallelize column-related tasks, particularly during histogram calculations. This strategy is essential for evaluating the effectiveness of potential data splits. In distributed computational settings, maintaining uniform split quality is crucial. XGBoost tackles this with its weighted quantile sketch, efficiently combining feature histograms across multiple distributed data nodes.

### 2.3.2 Description of Algorithms and Hybrid Approach II

The following diagram shows the flow of the proposed low-complexity hybrid method. In each step, the transformation from the inputs to the final output, which is the finding of a job title similar to a given job title, is described.

The data that this hybrid method receives as input is a finite list of 80,000 job titles and 480 industries. Similar to the method explained in the previous section, the embedding of job title and industry was first performed using the BERT model. As a result, each job title and industry is represented as an array of length 768.

The “Labels” step shows how to define whether a pair of job titles are similar or not. This is performed using Algorithm 1 and Algorithm 2, as discussed in the previous section. The threshold used as the similarity criterion is 2, which means that two job titles are defined as similar if they have two common industries.

It is reiterated that industry serves as a decisive criterion for determining job title similarity. Among various factors, a job title relates to the domain knowledge of the industry, its requirements, and applications. To summarize, after this step, there exists a list of similar job titles for each given job title, based on the commonality of industries.

After the similarity label was defined, an XGBoost model was trained in the next step. This model can predict each new pair of job titles with a similarity class of 0 or 1.These job titles are similar based on common industries and are similar among them. After this step, for a given job title, our hybrid method generates a list of the top similar job titles that can serve as substitutes.

The figure below shows the Classification Report for the XGBoost model used for

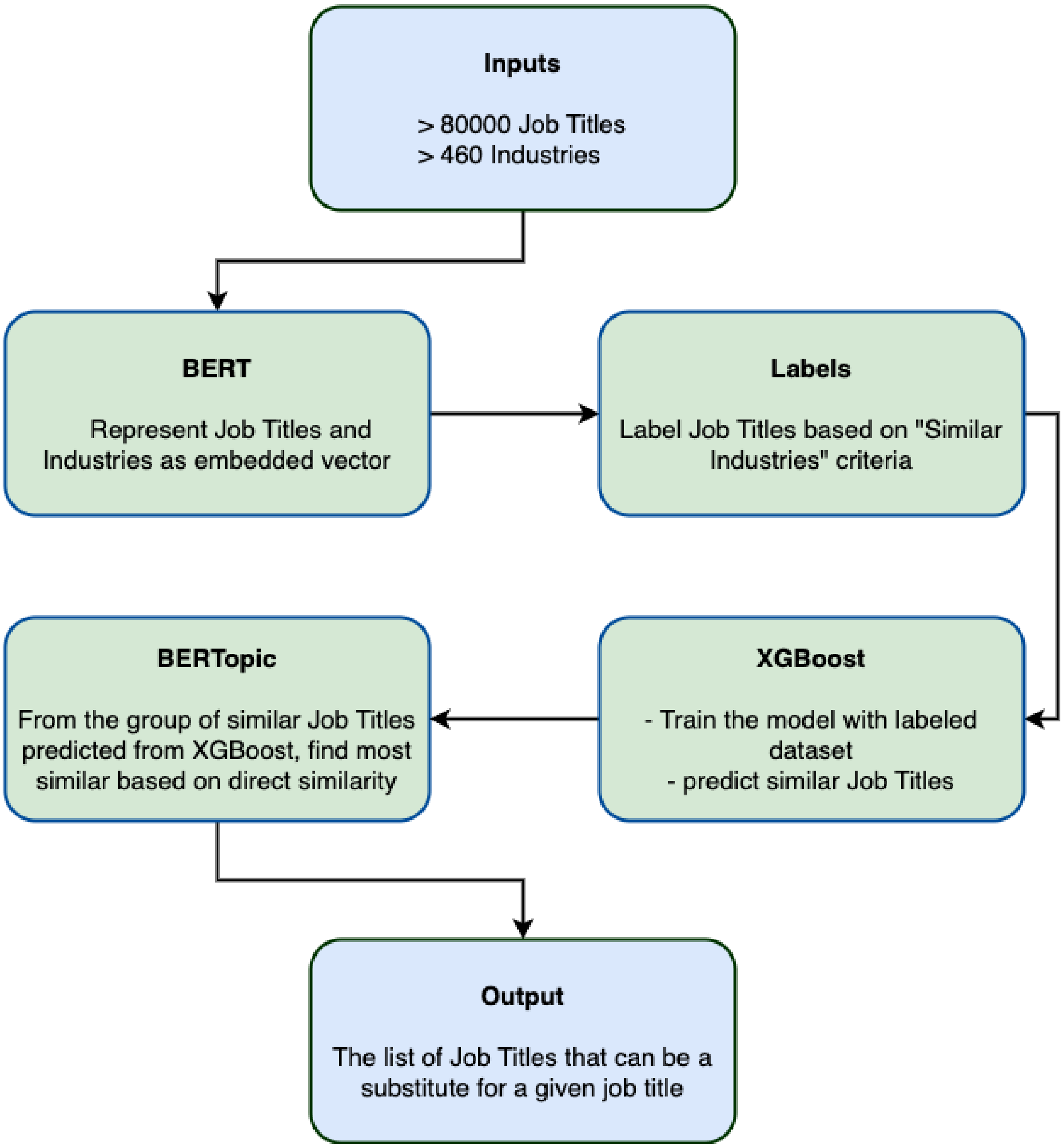

Inputs
> 80000 Job Titles
> 460 Industries
BERT
Represent Job Titles and Industries as embedded vector
Labels
Label Job Titles based on "Similar Industries" criteria
BERTopic
From the group of similar Job Titles predicted from XGBoost, find most similar based on direct similarity
XGBoost
- Train the model with labeled dataset
- predict similar Job Titles
Output
The list of Job Titles that can be a substitute for a given job title

Figure 11: Optimization Method II: Based on Hybrid Approach II

prediction. Compared to the SVM model used in the method presented in the previous section, the XGBoost model used in this hybrid method improved the classification accuracy by **2%**. There was also an improvement and balance in the prediction accuracy between classes.

The confusion matrix of the XGBoost model confirmed the improvement in the prediction of the job title similarity class.

Table 3: Model Classification Report (XGBoost)

THE EVALUATION OF THE MODEL

|  | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.91 | 0.90 | 0.90 | 14514 |
| 1 | 0.90 | 0.91 | 0.91 | 14638 |
| accuracy |  |  | 0.91 | 29152 |
| macro avg | 0.91 | 0.91 | 0.91 | 29152 |
| weighted avg | 0.91 | 0.91 | 0.91 | 29152 |

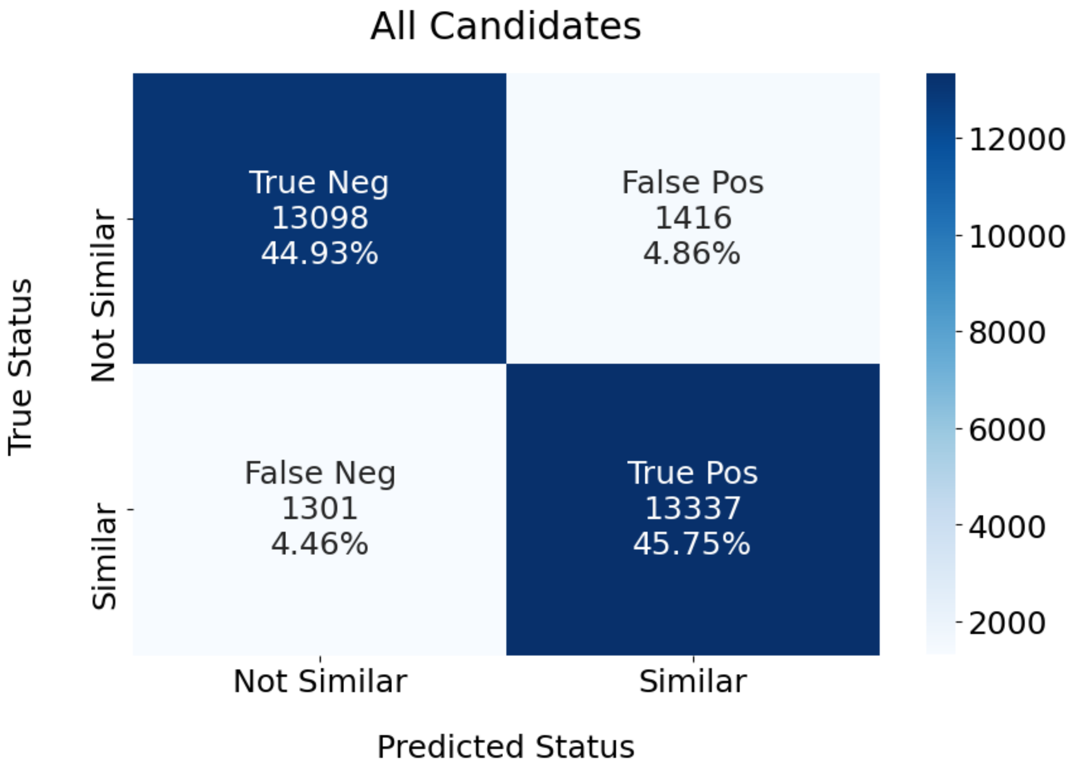


Figure 12: Confusion Matrix for XGBoost model

The above results demonstrate that the XGBoost model predicts the job title similarity class with high accuracy. However, this is not enough, and for this reason, another step is included: BERTopic. This step aims to tighten the criteria for determining similarity. In a list with predicted similar job titles, the similarity between them is calculated, and a reduced list is reached.

After this step, the result will be what is called the final product of this hybrid method, which is a list of the top most similar job titles for a given job title. The following table shows the most similar job titles for “machine learning engineer”.

Table 4: Second hybrid approach results of substitute Job Titles for “Machine Learning Engineer”

MACHINE LEARNING ENGINEER MOST SIMILAR JOB TITLES

| Query | Most Similar Job Titles | Similarity Score |
|---|---|---|
| machine_learning_engineer | machine_learning_engineer | 1 |
| | software_engineer_machine_learning | 0.95 |
| | senior_machine_learning_engineer | 0.91 |
| | machine_learning_engineer_intern | 0.91 |
| | machine_learning_scientist | 0.91 |
| | machine_learning_researcher | 0.90 |
| | machine_learning_research_assistant | 0.89 |
| | artificial_intelligence_engineer | 0.88 |
| | machine_learning_intern | 0.86 |
| | machine_learning_research_intern | 0.85 |

# Chapter conclusions

To summarize the findings of this section, the notable changes observed when using BERT as a transformer and XGBoost for predicting the similarity class are highlighted. It's important to recall that in the initial method, which shared a preprocessing step similar to the hybrid method discussed in this section, Word2vec was utilized as the transformer, and SVM was employed to predict the similarity class.

Some of the conclusions drawn from the hybrid approach are as follows.

- The use of machine learning techniques that are considered "state of the art" significantly improves the results of our hybrid method.
- 2% improvement in the accuracy of job title similarity class prediction
- 11.6 times reduction in execution time and memory consumption, which makes this hybrid approach highly efficient for recruitment platforms that process big data.

For clarity, it emphasizes that the mentioned values are an average calculation. Job title similarity cannot be determined in the same way for all cases. For some job titles, a large number of other job titles can be found that can serve as substitutes, and for some other job titles, it is difficult to find substitutes.

# Chapter 3

# Predictive Models for Job Success: Optimization of candidate selection strategy

In this chapter, an advanced ML method is introduced for predicting job success, surpassing the efficiency of traditional manual approaches. This method is distinct in its utilization of BERT for embedding text and XGBoost for predicting job success. The integration of BERT enables a sophisticated understanding of textual data, thereby capturing the nuances of soft skills and other text-based candidate information. XGBoost complements this by offering a powerful and efficient predictive model known for its high performance in classification tasks. This chapter explores how the combination of these advanced techniques enhances a model's ability to accurately predict job success by incorporating a diverse array of data, including speech features. The synergistic use of BERT and XGBoost in this context illustrates a pioneering approach in the recruitment field, showing how cutting-edge ML techniques can be leveraged to assess candidate potentials more comprehensively and accurately. Through this innovative methodology, this chapter contributes significantly to the field, demonstrating the potential of ML in transforming recruitment processes and improving the accuracy of job success predictions.

## 3.1 Introduction

Employment stands as one of the most crucial factors that directly impact the quality of our lives. Not only does it provide individuals with financial stability, but it also shapes their sense of purpose, identity, and overall well-being. On the other side, employees represent the most valuable asset for companies, contributing to productivity, innovation, and the achievement of organizational goals. With such significance placed on both employment and employees, organizations are compelled to invest in efficient and innovative methods to improve the quality of their recruitment processes.

In the past decade, there has been a notable shift in the utilization of machine learning techniques for recruitment purposes. The immense processing power and capabilities that these technologies offer are what are primarily driving this shift. Machine learning algorithms can swiftly analyze vast amounts of data, providing organizations with valuable insights to make informed decisions. Compared to traditional recruitment methods, AI-powered tools offer several advantages, including reduced processing time, enhanced accuracy, and the ability to handle large volumes of candidate data.

One advantage of AI-powered recruitment tools compared to traditional recruitment methods is the reduced processing time. This efficiency stems from the ability of machine learning algorithms to rapidly process and analyze large datasets, enabling organizations to swiftly move through the recruitment process. By reducing the time from application to employment, organizations can gain a competitive edge, secure top talent, and expedite subsequent processes such as onboarding and training.

Although modern recruitment tools bring several advantages, their trustworthiness remains under scrutiny. A study by Atkinson and Wright reveals the anxieties of job seekers. An overwhelming 90% of applicants chose a human interviewer over a machine when given the choice. Furthermore, the same group stated that they did not receive any feedback from prospective employers more than 70% of the time. These findings underline the conundrums in the recruitment sector. While traditional methods grapple with efficiently handling large volumes of applicants, concerns linger regarding the capability of cutting-edge systems to deal effectively with a vast array of candidate information and applications.

The landscape of work has rapidly transformed, with remote work becoming more prevalent due to the global outbreak of the COVID-19 pandemic. This shift has significantly impacted recruitment standards. Geographic location is no longer a barrier for most occupations, as remote work allows organizations to consider candidates from locations different from their own. As a result, the number of candidates applying for each vacancy has increased, requiring the adoption of more advanced methods, such as machine learning techniques, to efficiently process and analyze massive volumes of candidate data.

### 3.1.1 Recruitment Evaluation Strategies

At the interview stage, recruiters carefully analyze candidate data to assess whether they possess the necessary skills, qualifications, and attributes to be a good fit for the job position. Recruiters typically consider various criteria to evaluate candidates effectively. Some of these criteria include:

1. **Quality of hiring:** It is essential for recruiters to select candidates who meet the specific job requirements and align with the broader goals of the company. Making the right hiring decisions contributes to overall organizational success.

2. **Recruitment duration:** The speed at which the recruitment process is completed can be advantageous for both candidates and organizations. Swift recruitment processes enable candidates to secure employment promptly, while organizations can fill vacancies expediently and minimize the impact of prolonged recruitment cycles.

3. **Average employment cost:**

$$\text{Average Employment Cost} = \frac{\text{recruitment cost}}{\text{number of hires}}$$

The average employment cost is a critical factor for organizations. As they aim to decrease the time and resources allocated to the recruitment process, methods utilizing Big Data processing are becoming more attractive. Techniques such as machine learning are being increasingly adopted, primarily because of their speed and cost-efficiency.

4. **Retention Period/Job Turnover:** These terms refer to the duration an employee stays within a company and the rate at which a company loses its workforce, respectively. Ensuring employees' long-term engagement is a priority for organizations. By selecting candidates who are not only likely to remain but also prosper in their roles, companies can lower turnover rates and the associated costs. The task for recruiters is to locate candidates who demonstrate the appropriate skills, align with the company culture, and possess the potential for growth within the organization.

In addition to these primary criteria, factors like employee satisfaction and recruiter satisfaction also significantly influence recruitment success. All these elements work collectively towards the ultimate goal of finding the best candidate for the position. One could define a successful hiring process as one where the selected candidates hold their positions for at least six months, indicating a positive fit between the candidate and the job.

### 3.1.2 Proposed solution

Recruiters heavily rely on candidate data, including job titles, experience, hard skills, soft skills, and job requirements, to assess the likelihood of a candidate's success in a specific role. However, a candidate's success depends on a variety of factors, and recruiters typically lack the resources to manually analyze a large number of variables. Furthermore, human bias can inadvertently impact the selection process, leading to the exclusion of potentially excellent candidates or the acceptance of less qualified ones. Machine learning algorithms offer a solution to mitigate these challenges.

In light of these considerations, the utilization of machine learning techniques to predict the likelihood of candidate success, is proposed. Unlike classical recruitment methods, the incorporation of machine learning expands the number of parameters considered for each candidate, resulting in more accurate predictions. In addition to experience and skills, it is proposed including other textual and speech data analyzed using natural language processing algorithms. Parameters such as the candidate's profile description, the job description, and other relevant text and speech data can significantly improve the accuracy of the predictive model. By adding these extra parameters, our proposed model does better than other machine learning models that were used in similar situations and is more accurate than traditional recruitment methods.

An in-depth study was undertaken on predictive models focused on improving job success, aiming to combine skills and candidate information to enhance recruitment outcomes. Advanced analysis methods that consider a wide range of factors are employed to significantly impact future recruitment practices. Ultimately, our focus is on ensuring the best possible job fit, enhancing retention rates, and fostering the conditions for organizational success through effective use of data analysis techniques.

## 3.2 Theoretical Background and Empirical Findings

Theoretical frameworks play a crucial role in shaping the understanding of predictive models for job success. In this section, it is explored the key theories and concepts that underpin the integration of skills and candidate data in predicting job success.

### 3.2.1 Human Capital Theory

A person's knowledge, experience, and skills are significant assets that increase their productivity and performance in the job market, according to the human capital theory that Gary Becker developed [127]. This idea holds that investing in the development of human capital via education, training, and experience improves people's performance and employability. When applying this theory to predictive models for job success, the focus is on identifying and assessing the relevant human capital attributes of candidates, such as educational qualifications, certifications, and specialized skills.

Human capital theory provides a valuable theoretical framework for understanding the role of skills and candidate data in predicting job success. Organizations can find people who have the required skills and experience by taking into account the specific human capital traits that are pertinent to a particular employment role. To increase the likelihood of success, this approach emphasizes the need to make investments in the development of human capital and match candidates' skills with job requirements.

### 3.2.2 Social Cognitive Theory

Albert Bandura [128] developed the social cognitive theory, which emphasizes the interaction between people's cognition, behavior, and social environment. It asserts that individuals learn and develop their skills, self-efficacy, and motivation through observation, modeling, and social interaction. In the context of predictive models for job success, social cognitive theory suggests that assessing candidates' self-efficacy, motivation, and social skills can provide valuable insights into their potential job performance and success.

Through the lens of Social Cognitive Theory, it can better comprehend how an individual's beliefs, motivations, and social interactions can influence their job success. Evaluating a candidate's self-efficacy, which is their belief in their capacity to carry out specific job tasks, can provide a glimpse into their potential success. Moreover, by assessing their motivation and social skills, organizations can further ascertain their adaptability to the work environment and their capacity to collaborate effectively with others. Ingraining Social Cognitive Theory into predictive models augments their capacity to encompass the multifaceted dimensions of job success.

### 3.2.3 Person-Environment Fit Theory

The person-environment fit theory underscores the relationship between an individual's attributes, the expectations of the job, and the broader organizational context. It argues that when individuals' characteristics, such as traits, values, and skills, significantly align with job requirements and organizational culture, there's a higher probability of job satisfaction, performance, and success [129]. In the context of predictive models for job success, this theory necessitates an evaluation of how well a candidate's skills, personality traits, and values match with the job needs and organizational culture to forecast their adaptability and prospective success.

This theory provides a valuable framework for understanding the interaction between candidates' attributes and the work environment. By assessing the alignment between candidates' skills, traits, and values with the specific job requirements and the organizational culture, organizations can identify candidates who are likely to thrive in the given context. This theory recognizes that job success is not solely determined by individual attributes but also by the fit between individuals and their work environment, emphasizing the importance of considering both candidate data and contextual factors in predictive models.

### 3.2.4 Machine Learning Techniques in Predictive Modeling

Machine learning techniques have gained considerable attention in predictive modeling for job success. These algorithms have the ability to analyze large volumes of candidate data, identify patterns, and make accurate predictions. Various machine learning algorithms, such as decision trees, random forests, support vector machines, and neural networks, have been employed in the context of recruitment to predict job success based on skills and candidate data. By leveraging machine learning algorithms, organizations can gain valuable insights into the factors that contribute to job success and make more informed decisions in the candidate selection process.

An important aspect of predictive models for job success is the initial filtering of candidates. Previous research [4] and [5], has demonstrated that machine learning algorithms can effectively reduce the candidate pool based on similar job titles and sectors. By filtering out candidates who do not have the same or very similar job titles as required for the open position, subsequent stages of the recruitment process can focus on candidates with a higher likelihood of success. This approach reduces processing time and resources and eliminates irrelevant applications from consideration in subsequent stages.

In the context of a larger study, E.T. Albert's research [130] provides significant insights into the integration of AI in recruitment and selection processes. Through a combination of extensive literature review and primary research involving interviews HR managers, consultants, and academics, Albert identifies 11 potential areas where AI can be incorporated in recruitment and selection, which include contacting candidates, selecting them, finding the most suitable jobs, training to be successful, etc.

Several empirical studies have explored the integration of skills and candidate data in predictive models for job success. For example, the study conducted by M.L. Demircan and K. Aksaç [131] in the private banking sector in Turkey, analyzed a sample pool of 597 individuals. Their research utilized various machine learning models and achieved an accuracy rate of over 73% in predicting candidate success before employment commences. This study highlights the potential of machine learning techniques for predicting job success based on skills and candidate data.

A lot of empirical research has been done on predictive models for job performance. This research helps us understand why it is important to include skills and candidate-specific data. Such research underscores the capability of machine learning algorithms to scrutinize a variety of factors and predict job performance with precision. Using this empirical data as a foundation, companies can build their trust in implementing predictive models within their hiring process, leading to data-informed decision-making.

## 3.3 The proposed Approach

The innovation in our methodology involves integrating additional data, including text and speech, to enhance the candidate's profile and achieve greater precision in predicting job success. BERT transforms the text data into a format that the machine learning model picked for the prediction stage can use. This is done during the preprocessing phase as well as the standard steps. After the finish the preprocessing and feature engineering steps, is performed the train of an XGBoost model that tells us whether the candidate will be successful or not.

### 3.3.1 Data Description

In the context of this study, it is analyzed a dataset comprised of 648 individual entries. Each entry provides comprehensive details about an individual, encapsulating their job designation, associated industry, cumulative professional experience, predominant skills, descriptions of the most recent three workplaces or projects they've been involved with, and the average duration they've remained at a position. A particularly interesting component of our dataset are the nuanced insights sourced from their CV or video CV, which highlight elements of motivation, enthusiasm, and communication skills. Throughout this thesis, it is refered to this compilation of information as the "candidate profile".

Each candidate entry in this dataset comes with a "project profile" for the specific project to which they applied. This profile provides details based on the criteria set at the time of application, capturing the requisite job title, associated industry, an outline of the project (often detailed with specific roles and responsibilities), the desired experience level (typically highlighting a minimum year threshold), and essential skills a candidate must possess. Additionally, the dataset features a "job success class," a binary metric signaling whether a candidate successfully navigated the project's probationary phase. While probationary durations can vary across projects, a consistent measure of success is a duration of 6 months. For the purposes of our modeling, the "job success class" is the specific target variable that the XGBoost Classifier model is trained to predict.

The figures above are a graphical representation of the distribution of samples by industry, job title, and success class. These data are used to train and evaluate the model. The

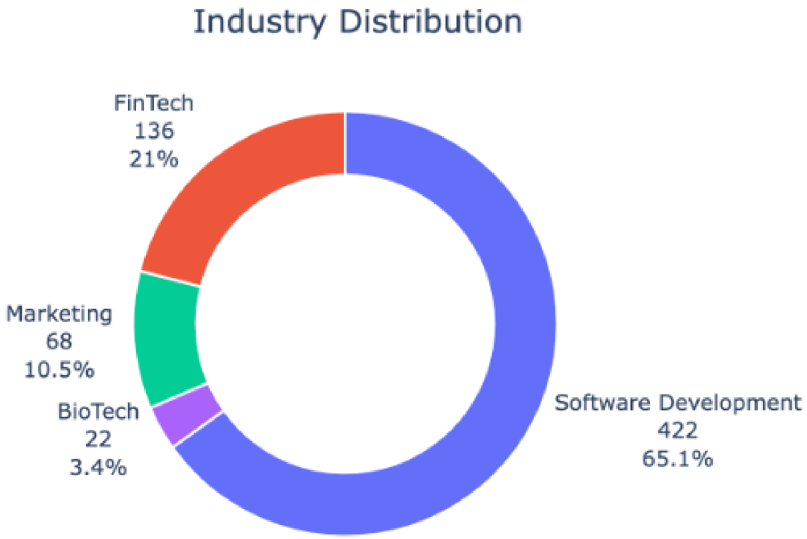


Figure 13: Distribution of applicants by Industry

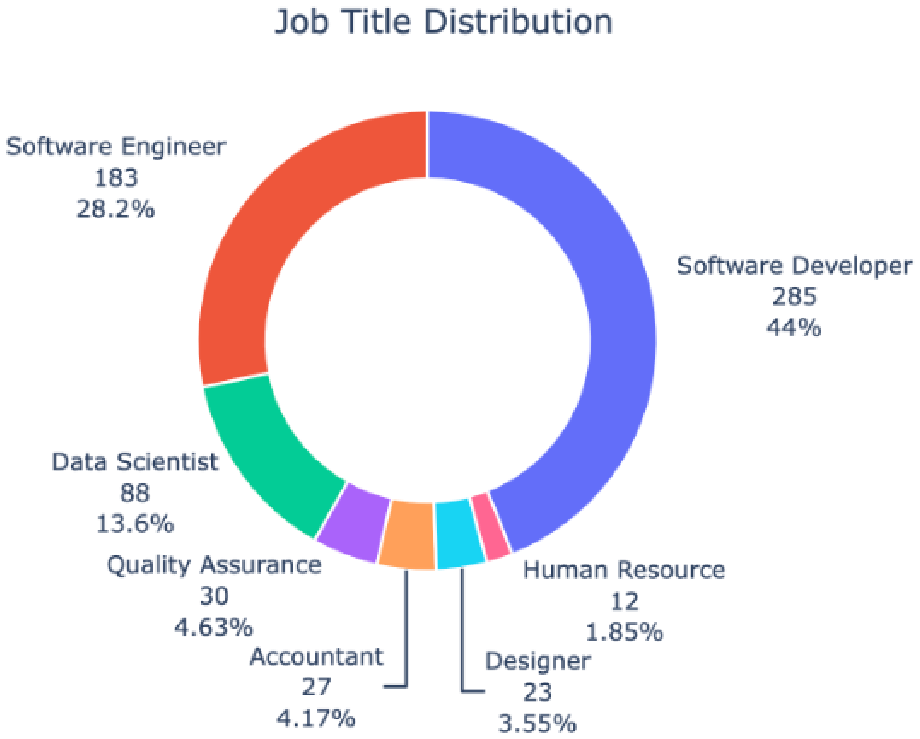


Figure 14: Distribution of applicants by Job Title

candidates were selected from four different industries: Software Development, FinTech, BioTech, and Marketing. The Software Development industry dominates more than 65% of the sample.

Regarding distribution by job title, the candidates in this population are Software De-

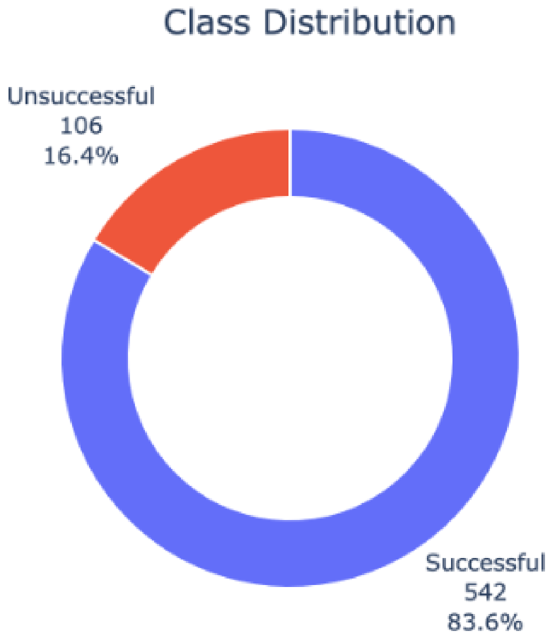


Figure 15: Distribution of applicants by Success class

velopers, Software Engineers, Designers, Data Scientists, Human resources, Quality assurance, and Accountants. Most of the candidates are in roles related to software development. The third graph, Class Distribution, shows the distribution of the label, where it is noted that **83.6%** belongs to the successful class and the rest to candidates who have failed to stay at work for more than the probationary period.

### 3.3.2 Pre-processing and Feature Engineering phase

In this section, the steps through which each input is prepared for use as a feature in the prediction model are described.

As explained above, in the candidate and project profiles, the data inputs are of different types. In the feature-engineering phase, different types of inputs follow different processing paths. The job title, industry, and project description provided as textual data go through the same pre-processing procedure. The first step was embedding using the BERT model. The embedding process represented each data text, job title, industry, and project description in an array with a length of 768.

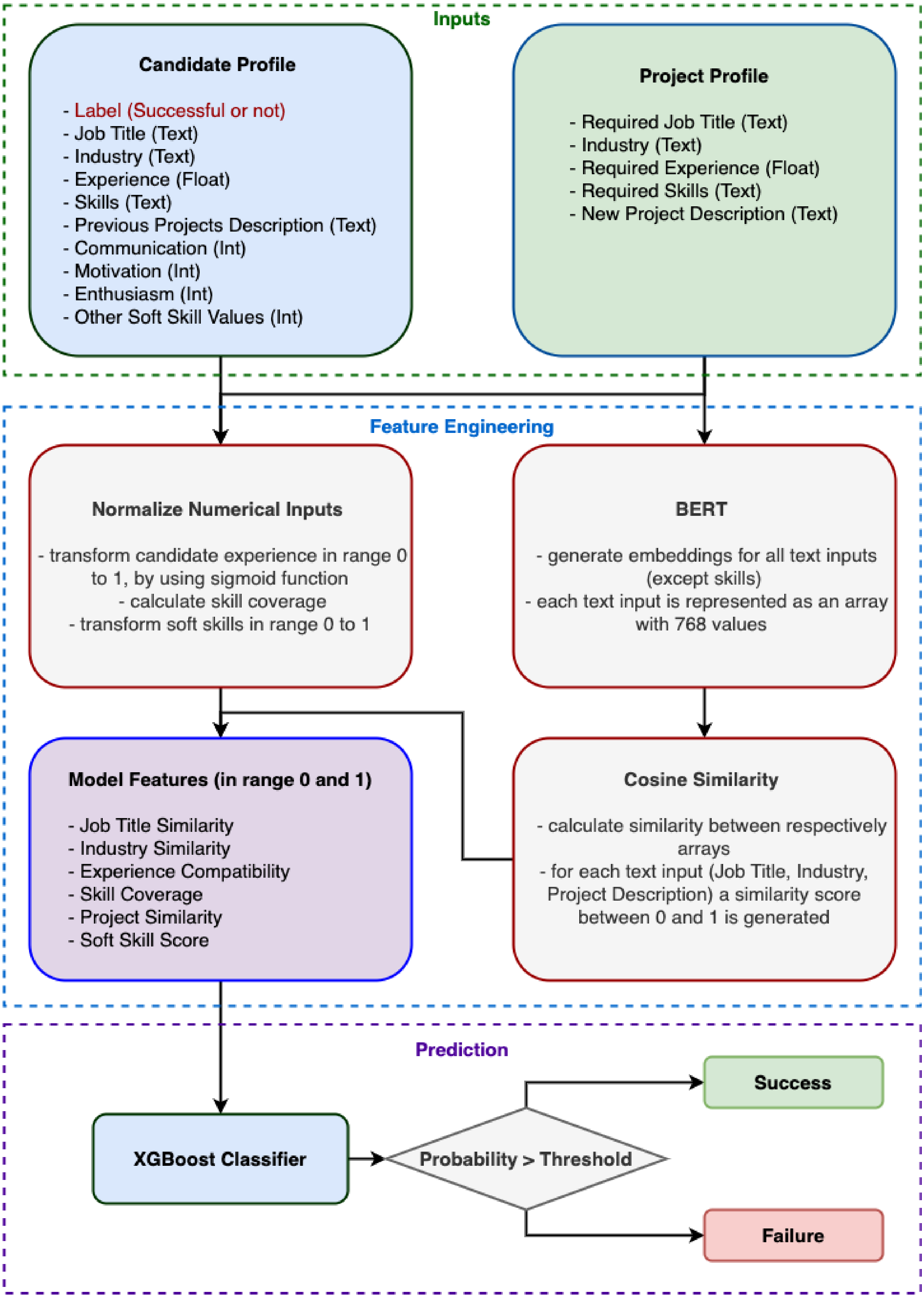

Inputs
Candidate Profile
- Label (Successful or not)
- Job Title (Text)
- Industry (Text)
- Experience (Float)
- Skills (Text)
- Previous Projects Description (Text)
- Communication (Int)
- Motivation (Int)
- Enthusiasm (Int)
- Other Soft Skill Values (Int)
Project Profile
- Required Job Title (Text)
- Industry (Text)
- Required Experience (Float)
- Required Skills (Text)
- New Project Description (Text)
Feature Engineering
Normalize Numerical Inputs
- transform candidate experience in range 0 to 1, by using sigmoid function
- calculate skill coverage
- transform soft skills in range 0 to 1
BERT
- generate embeddings for all text inputs (except skills)
- each text input is represented as an array with 768 values
Model Features (in range 0 and 1)
- Job Title Similarity
- Industry Similarity
- Experience Compatibility
- Skill Coverage
- Project Similarity
- Soft Skill Score
Cosine Similarity
- calculate similarity between respectively arrays
- for each text input (Job Title, Industry, Project Description) a similarity score between 0 and 1 is generated
Prediction
XGBoost Classifier
Probability > Threshold
Success
Failure

Figure 16: Job Success Prediction Model Flow

Subsequently, the cosine distance was calculated for each pair of embedding arrays. This distance assigns each pair a similarity coefficient ranging from 0 to 1. Recall that the first element of the pair is related to the job title, industry, and description of the candidate, and the second refers to the job title, industry, and description of the project. After this step, three similarity features are generated to show how similar the job title, industry, and description are between the candidates and the project they applied for.

The following equation shows the formula for calculating the cosine distance:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}|| \, ||\mathbf{B}||} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}}$$

Another piece of text information provided in both the candidate profile and project profile is the skills. Usually, in a job vacancy, some skills that the candidate must have are given. For example, in a job vacancy for a data scientist role, the candidate must have knowledge of Python, GraphSQL, and Cloud Infrastructure. This is a very important feature because the candidate cannot work on a project that requires skills that the candidate does not know. In the feature engineering phase, skills are transformed into a numerical value from 0 to 1, which describes the percentage of skills required in the project profile possessed by the candidate. Therefore, this coefficient shows the intersection between the set of candidate skills and the set of skills required by the project, as shown in percentage terms.

The candidate's experience is one of the most important criteria that greatly affects the success or failure of a job position. However, depending on the role, complexity, and requirements of the project, years of experience vary over a wide range. It is not advisable to include in the prediction model a feature that can take a variety of values with a large standard deviation. Therefore, in the feature engineering phase, a sigmoid function is created to transform the years of experience into a score between 0 and 1. This experience score takes small values (close to 0) if the candidate's experience (CE) is much less than required (RE) and high values (up to 1) if the candidate's experience is closer to what is required. The following formula shows how the Experience Score feature is calculated:

$$\text{Experience score} = \begin{cases} \dfrac{CE}{CE + e^{-c(CE-RE)}}, & \text{if } CE \leq RE \\ 1, & \text{otherwise} \end{cases} \tag{3.1}$$

where $c$ is a constant that represents aggressivity; in this case, $c = 0.5$.

The other inputs are collected through a structured procedure, and they are represented by integer numbers in the range from 1 to 10. In the feature engineering phase, these inputs are transformed by dividing them by their maximum, 10 in this case, resulting in features with values in the range 0 to 1.

## 3.4 Advantages of this Approach

The advantages of this approach are evident in at least two primary directions:

1. Efficiency
2. Accuracy

Machine learning approaches are known to be more efficient, specifically for time-consuming tasks such as the recruitment process. The proposed machine-learning approach makes a significant difference in reducing the time needed to process applicants, reducing costs, and many other benefits that arise in relation to classical recruitment methodologies.

The article on employment statistics published by R. Thakkar stated that the average time to recruit an applicant is 36 days, and more than 70% of recruiters believe that automation would significantly reduce this time.

It is interesting that, in addition to being more efficient, a well-designed and implemented machine learning solution is also more accurate than manual or classical recruitment methods.

In the treated experiment, the recruiters correctly predicted the success of the candidates in **84%** of the cases, while, after the interview stage, only **542** of the **648** selected candidates successfully passed the testing stage. However, incorporating text and speech data into a machine learning model results in more accurate candidate predictions than manual recruiting.

After implementing the steps explained in the previous section, a dataset is generated suitable for the XGBoost prediction model. This dataset contains features that take continuous values between 0 and 1 and a label indicating whether a candidate is successful (1) or unsuccessful (0).

To maximize the size of the data used for validation, a cross-validation method, is implemented. The data with 648 samples is divided into ten folds, of which nine folds are used for training and one for validation, to ensure that the model does not overfit. The procedure is repeated ten times, each time with a different validation fold. Once the model is trained, it is ready to predict the success of a candidate. It is worth noting that the entire training process, including all preprocessing steps and feature engineering, is performed on a computer with an *Intel Core i7-9750H CPU* with *6 cores* and *16 GB* of RAM and takes approximately *7 hours*. However, once the model is trained, the prediction time is almost instantaneous.

A prediction is generated for each candidate in the original dataset, which improves the accuracy of the model's validation and ensures that the model creates predictions for samples that it didn't "see" during the training phase.

A key aim of this research is to highlight the importance of features extracted from text and speech and measure their impact on prediction accuracy. To quantify this, it is used the same model, namely XGBClasiffier, with the aforementioned hyper-parameters, to predict job success using different sets of features. In the first scenario, the model is trained using only the“classic features” such as experience, skills, job title, etc. In the second scenario, the model is trained with all available features, including those generated from text and speech data. The results are demonstrated in the evaluation metrics table in the section below. It can be observed that prediction accuracy increases as the candidate’s profile is enriched with features from text and speech data.

## 3.5 Model evaluation and result analysis

1. **Inclusion of text and speech data**

   The proposed machine learning approach aimed to determine whether using text and speech data could improve the ability to select the correct candidates for a job.

   ❖ *Comparison of the Old Model and New Model*

   Table 5 presents the main evaluation metrics used to compare the two models used in this experiment.

   *Old Model:* This model considered only what is called “hard skills.” These are measurable skills, such as a person’s qualifications, years of experience, and technical knowledge.

   *New model:* This newer approach not only considers hard skills but also integrates text and speech data. This would help assess “soft skills” such as communication skills, adaptability, and teamwork.

Table 5: Evaluation metrics of Old Model and New Model

| **Model** | **Evaluation Metrics** | | | | |
|---|---|---|---|---|---|
| | **Accuracy** | **Precision** | | **Recall** | |
| | | Class (1) | Class (0) | Class (1) | Class (0) |
| Old Model | 0.88 | 0.97 | 0.59 | 0.88 | 0.87 |
| New Model | 0.93 | 0.98 | 0.72 | 0.93 | 0.91 |

❖ *Findings:* By including Text and Speech Data

- The overall model's accuracy improved by 5%. This implies that the new model can predict suitable candidates 5% better than the old model.
- The model's precision, which is a measure of how many selected candidates are actually suitable, increased by 1%.
- The model's recall, indicating how many of the truly suitable candidates it could correctly identify, was improved by 5%.

**2. Machine learning vs. manual recruiting**

Figure 17 shows the confusion matrix of the trained model with all features, which is referred to above as the "New Model".

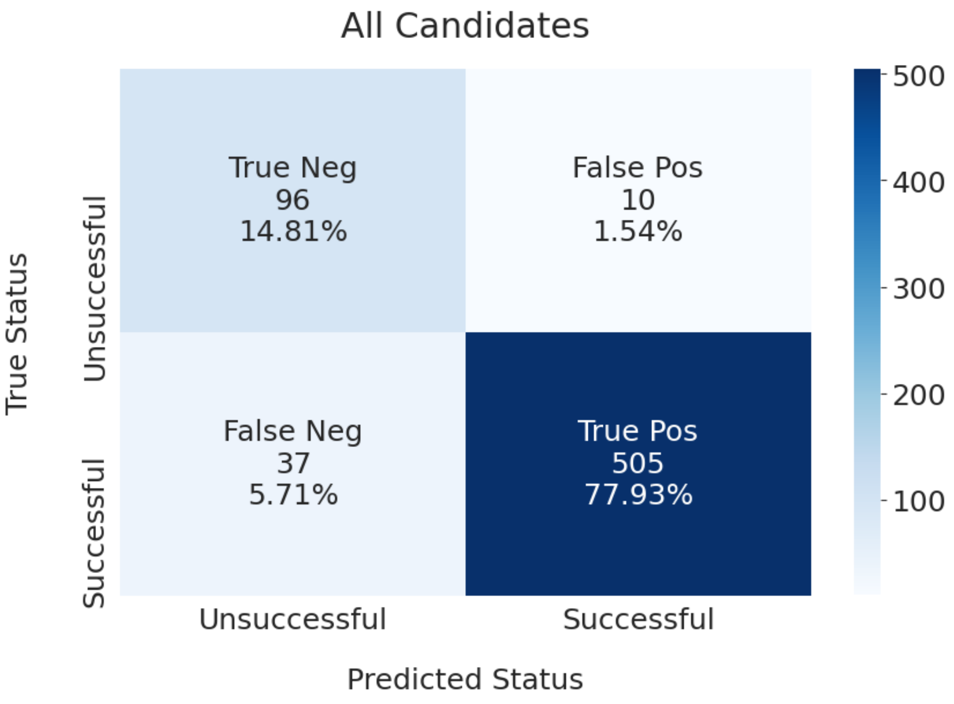


Figure 17: Confusion Matrix of Job Success prediction - XGBoost model

This experiment compared the effectiveness of a fully trained machine learning model with traditional manual recruitment methods.

❖ *Findings:*

- The human recruiters had an accuracy rate of 84%. This means that they could correctly identify a candidate's potential success in a role 84% of the time.

- The machine learning model, on the other hand, boasted an accuracy of 92.7%, which is significantly higher.

- A key metric for businesses is the "False Positive" rate, which refers to candidates who are expected to do well but do not. The machine learning model had a lower rate of such candidates, making it more cost-effective in the long term.

**3. Efficiency and speed**

The effectiveness of machine learning was a major theme in this study.

❖ *Findings:*

- The machine learning model can process and predict candidate success in seconds.

- In contrast, manual recruiting procedures can take days, which is frequently a limitation due to the availability of human recruiters.

As a general conclusion, this case shows that with the integration of machine learning and natural language processing into the recruitment process, companies can better identify suitable candidates with higher accuracy, but they can also do so much faster than traditional manual methods.

Future paradigm shifts brought about by technology might redefine and enhance recruitment practices.

### 3.5.1 Challenges

- *Challenges in Data Collection:* Obtaining comprehensive data for a candidate often involves pulling from multiple sources. This multi-source gathering demands rigorous data processing because inaccuracies or inconsistencies can skew predictions and lead to suboptimal hiring decisions.
- *Machine Learning's Perceived Credibility:* While there's increasing interest in harnessing machine learning for recruitment, its trustworthiness is still debated. Many organizations use it as a supplementary tool, with human recruiters retaining the final say in candidate selection.
- *The Elusiveness of Defining Success:* Pinpointing what constitutes "job success" is complex, varying from one organization to the next. It is challenging to distinguish why a candidate is deemed unsuccessful. For instance, was it a shortfall in their skills or did they simply receive another job offer elsewhere? Both scenarios lead to the same label, "Unsuccessful," yet the underlying reasons are vastly different.

## Chapter Summary and Conclusion

This chapter presented a comprehensive analysis of a machine learning model trained for predicting job success, comparing two distinct versions: the 'old model' using classical features and the 'new model' incorporating advanced text and speech features. A comparative study revealed that the inclusion of these nuanced features in the new model significantly enhanced its predictive capabilities.

These improvements, especially in terms of accuracy and recall, are particularly significant in the context of recruitment, where efficiently identifying the right candidate can have a profound impact on organizational success. Notably, both models, with their respective feature sets, outperformed manual approaches, underscoring the value of machine learning in enhancing the recruitment processes.

The chapter concludes by affirming the substantial benefits of incorporating advanced data types such as text and speech into predictive models for job success. These findings underscore the potential of machine learning to revolutionize recruitment and offer more precise, efficient, and effective candidate assessment methods.

# Thesis Conclusions

The success of an organization has always been intricately linked to its recruitment strategies. Identifying the right talent not only ensures that tasks are executed efficiently but also shapes the growth trajectory of a company. In this research, the primary objective is to assess the applicability and benefits of Machine Learning and Natural Language Processing in the recruitment process, by providing a holistic overview of existing methods, pushing the boundaries of these techniques with novel optimization strategies.

The first approach was a combination of Word2vec and the SVM method, with the aim of creating a system that could systematically analyze a pool of candidates and match their qualifications with job titles and industry-specific details. While Word2Vec expertly turned textual parts like resumes and job descriptions into a numerical format that computers could read, SVM separated and matched candidates based on job requirements. The integration of these two algorithms brought a noticeable improvement in candidate-job matching, speeding up the process by 80% compared to traditional recruiting. This approach showed that even simple technological integration could yield substantial benefits.

Another method combination explored is BERT and XGBoost, with the goal of further elevating precision in identifying job title similarities. With its advanced capabilities, BERT delves deep into textual data, uncovering patterns and nuances that simpler models might overlook, making it an ideal choice for understanding diverse job descriptions and varied resumes, while XGBoost enhances the decision-making process. This integration managed to enhance the system's accuracy by an additional 2% while operating more efficiently and requiring less memory, thereby establishing it as a cost-effective solution for large-scale recruitment operations.

Beyond the initial sorting based on job titles, the essence of recruitment lies in identifying candidates' potential for success. Thus, the research introduced a comprehensive evaluation system. This new approach not only assessed candidates on their technical or specific skills but also evaluated their general qualities, such as teamwork, adaptability, and problem-solving. The holistic evaluation led to a 5% improvement in accuracy. Even more impressively, when compared to recruitment by humans, the system showed an accuracy rate of 92.7%, showing that the method is reliable and has a lot of potential.

Putting together all of these research findings and observations, it is clear that using machine learning and natural language processing in hiring is not just a concept for the future, but a real, useful way to solve problems in the present. This research not only presents an innovative perspective on recruitment but also offers actionable tools and methodologies that can streamline and enhance the talent acquisition process for organizations of all sizes.

# Thesis Contributions

1. Analysis of the application of Machine Learning (ML) and Natural Language Processing (NLP) techniques in the recruitment industry
2. Suggested approach: Combination of Word2vec and the SVM method, with the aim of creating a system that could systematically analyze a pool of candidates and match their qualifications with job titles and industry-specific details.
3. Suggested approach: Combination of BERT and XGBoost method, with the goal of further elevating precision in identifying job title similarities.
4. Suggested holistic evaluation approach in recruitment for identifying candidates' potential for success
5. Formulated Job success prediction model
6. Created new perspective on how the recruitment industry should evolve to address the shifts in the globalization of the labor market across many professions, especially in the era of AI

# List of publications in connection with the dissertation thesis

1. Mankolli E. M., Guliashki V. G. (2020), “Machine Learning and Natural Language Processing: Review of Models and Optimization Problems”, Proceedings of 12th ICT Innovations Conference 2020, held on 24-26 September 2020 in Skopje, Republic of North Macedonia, “Machine Learning and Applications”, Vesna Dimitrova, Ivica Dimitrovski (editors), **Springer**, Volume 1316 of the Communications in Computer and Information Science series (CCIS), **ISBN: 978-3-030-62097-4, ISSN: 1865-0937, SJR (0.188)**, Computer Science - **Quartile Q3**, pp. 71-86

   https://link.springer.com/chapter/10.1007/978-3-030-62098-1_7

2. Mankolli E., Guliashki V. (2021) “A Hybrid Machine Learning Method for Text Analysis to Determine Job Titles”, TELSIKS 2021, *Proceedings of papers of the “15th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services”*, October 20-22, 2021, Niš, Serbia, IEEE Catalog Number: CFP21488-USB, ISBN: 978-1-6654-2912-2 (IEEE), pp. 380-385

   doi: 10.1109/TELSIKS52058.2021.9606341.

   https://ieeexplore.ieee.org/document/9606341

3. Mankolli E., Reducing the complexity of candidate selection using Natural Language Processing, In: *Proceedings of 29-th IEEE International Conference on Systems, Signals and Image Processing “IWSSIP 2022”*, June 01 - 03, 2022, Sofia, Bulgaria, pp. 1-4,

   doi: 10.1109/IWSSIP55020.2022.9854488.

   https://ieeexplore.ieee.org/document/9854488

4. Mankolli E., S. Bushati, Candidate Engagement Success Prediction Using Machine Learning and Natural Language Processing Techniques, In: *Proceedings of 24th Conference on Control Systems and Computer Science* (CSCS), May 24-26, 2023, Bucharest, Romania, pp. 431-435,

   doi: 10.1109/CSCS59211.2023.00074.

   https://ieeexplore.ieee.org/document/10214773

5. Guliashki V., E. Mankolli and S. Bushati, (2023), “A machine learning approach improving university campus security”, *IEEE International Workshop on Technologies for Defense and Security TechDefense 2023*, November 20-22, 2023, Rome, Italy, pp.341-345.

   `https://www.techdefense.org/files/IEEETechDefense2023_FinalProgram.pdf`

# Declaration of originality of the results

I declare that

1. This dissertation contains original research results obtained by me with the support and assistance of my supervisor. The presented dissertation work has been prepared by me.

2. Results that have been obtained, described and/or published by other scientists are duly and extensively cited in the bibliography. In the dissertation, no foreign texts are used directly or indirectly or, if parts of them are used, they are referred to / quoted and no part of my dissertation infringes the copyright of institution or person.

3. No part of my dissertation work has been presented in this form in the same or in another university, educational and/or scientific institution for the award of educational or scientific degree.

In the event of a discrepancy with the circumstances declared by me according to points 1, 2 and 3 of this declaration, I bear responsibility in accordance with the law and normative documents of IICT.

Signature:

# List of Figures

# List of Tables

# Bibliography

[1] Stacey McIntosh, "The changing face of HR: A research report for HR and people leaders", 2020.

[2] GlobalData. (n.d.). Retrieved March 30, 2022, https://www.globaldata.com/themes/artificial-intelligence/

[3] E. Mankolli, V. Guliashki, "Machine Learning and Natural Language Processing: Review of Models and Optimization Problems", October 30, 2020. https://link.springer.com/chapter/10.1007/978-3-030-62098-1_7

[4] E. Mankolli, V. Guliashki, "A Hybrid Machine Learning Method for Text Analysis to Determine Job Titles Similarity", October 20-22, 2021. https://ieeexplore.ieee.org/document/9606341

[5] E. Mankolli, "Reducing the complexity of candidate selection using Natural Language Processing", June 1-3 2022. https://ieeexplore.ieee.org/document/9854488

[6] E. Mankolli, S. Bushati, "Candidate Engagement Success Prediction Using Machine Learning and Natural Language Processing Techniques", May 24-26, 2023. https://ieeexplore.ieee.org/document/10214773

[7] Agarwal S., "Word to Vectors — Natural Language Processing", 2017. https://towardsdatascience.com/word-to-vectors-natural-language-processing-b253dd0b0817

[8] Daud A., Khan W. & Che D., "Urdu language processing: a survey.", Artificial Intelligence Review, 1-33, DOI 10.1007/s10462-016-9482-x, 2016.

[9] Khan W., Daud A., Nasir J. A., Amjad T, "A survey on the state-of-the-art machine learning models in the context of NLP", Kuwait Journal of Science 43 (4), pp. 95-113, 2016.

[10] Le J., "The 7 NLP Techniques That Will Change How You Communicate in the Future(Part I)". https://heartbeat.fritz.ai/the-7-nlp-techniques-that-will-change-how-you-communicate-in-the-future-part-i-f0114b2f0497

[11] Sun A., Cao Z., Zhu H., and Zhao J., "A Survey of Optimization Methods from a Machine Learning Perspective", in IEEE Transactions on Cybernetics, pp. 1-14, 2019.

[12] Goebel T., "Machine Learning or Linguistic Rules: Two Approaches to Building a Chatbot", 2017. https://www.cmswire.com/digital-experience/machine-learning-or-linguisticrules-two-approaches-to-building-a-chatbot/

[13] Ishibuchi H., Nakashima T., Murata T., "Multiobjective Optimization in Linguistic RuleExtraction from Numerical Data", E. Zitzler et al. (Eds.), pp. 588-602, Springer-Verlag Berlin Heidelberg, 2001.

[14] Shanthamallu, U.S.; Spanias, A.; Tepedelenlioglu, C.; Stanley M., "A brief survey of machine learning methods and their sensor and IoT applications", In Proceedings of the 2017 8th International Conference on Information, Intelligence, Systems and Applications, IISA, Larnaca, Cyprus, 2017.

[15] Joachims, T., "Text Categorization with Support Vector Machines: Learning with Many Relevant Features", Proceedings of the 10th European Conference on Machine Learning (ECML), 137-142, 1998.

[16] Kim, Y., "Convolutional Neural Networks for Sentence Classification", Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746-1751, 2014.

[17] Blei, D. M., Ng, A. Y., & Jordan, M. I., "Latent Dirichlet Allocation", Journal of Machine Learning Research, 3, 993-1022, 2003.

[18] Manning, C. D., Raghavan, P., & Schütze, H., Introduction to Information Retrieval. Cambridge University Press, 2008.

[19] Li, J., Monroe, W., & Jurafsky, D., “A Simple, Fast Diverse Decoding Algorithm for Neural Generation”, Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), 885-895, 2016.

[20] Young, S., Gašić, M., Thomson, B., & Williams, J. D., “POMDP-Based Statistical Spoken Dialogue Systems: A Review”, Proceedings of the IEEE, 101(5), 1160-1179, 2013.

[21] Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Schölkopf, B., “Learning with Local and Global Consistency”, Advances in Neural Information Processing Systems, 321-328, 2005.

[22] Settles, B., “Active Learning”, Synthesis Lectures on Artificial Intelligence and Machine Learning, 6(1), 1-114, 2012.

[23] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I., “Attention Is All You Need”, Advances in Neural Information Processing Systems, 5998-6008, 2017.

[24] Hochreiter, S., & Schmidhuber, J., “Long Short-Term Memory”, Neural Computation, 9(8), 1735-1780, 1997.

[25] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), 4171-4186, 2019.

[26] Zhou, Z. H., Ensemble Methods: Foundations and Algorithms. CRC Press, 2012.

[27] Rosenblatt, “The perceptron: A probabilistic model for information storage in the brain”, Psychological review, Vol. 65, pp. 386-408, 1958.

[28] Krizhevsky et al., “ImageNet classification with deep convolutional NN”, Adv. Neural Info. Process. Sys., Vol. 25, pp. 1090-1098, 2012.

[29] Sattigeri P., J. J. Thiagarajan, K. N. Ramamurthy and A. Spanias, “Implementation of a fast image coding and retrieval system using a GPU”, 2012 IEEEESPA, Las Vegas, NV, pp. 5-8, 2012.

[30] Schmidhuber, J., “Deep learning in neural networks: An overview”, Neural networks Vol. 61, pp. 85-117, 2015.

[31] Deng, L., & Yu, D., “Deep learning”, Signal Processing, Vol. 7, pp. 3-4, 2014.

[32] Mikolov T., Sutskever I., Chen K., Corrado G.S., Dean J., “Distributed Representations of Words and Phrases and their Compositionality”, Advances in NIPS, October 2013.

[33] Dong Y. and Deng L., Automatic speech recognition: A deep learning approach., Springer, 2014.

[34] Zheng, X., Chen, H. & Xu, T., “Deep learning for Chinese word segmentation and pos tagging”, Proceedings of the Conference on EMNLP-ACL-2013, pp. 647-657, 2013.

[35] Santos, C.D. & Zadrozny, B., “Learning character-level representations for part-of-speech tagging”, Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp.1818-1826, 2014.

[36] Li, Y., Miao, C., Bontcheva, K. & Cunningham, H., “Perceptron learning for Chinese word segmentation”, Proceedings of Fourth Sighan Workshop on Chinese Language Processing (Sighan-05), pp. 154-157, 2005.

[37] Qi, Y., Das, S.G., Collobert, R. & Weston, J., “Deep learning for character-based information extraction”, Proceedings of European Conference on Information Retrieval, pp. 668-674, 2014.

[38] Mohammed, N.F. & Omar, N., “Arabic named entity recognition using artificial neural network”, Journal of Computer Science, Vol. 8(8), pp.1285-1293, 2012.

[39] Lafferty, J., McCallum, A. & Pereira, F.C., "Conditional random fields: Probabilistic models for segmenting and labeling sequence data", Proceedings of the Eighteenth International Conference on Machine Learning, (ICML), pp. 282-289, 2001.

[40] Shanthamallu, U.S.; Spanias, A.; Tepedelenlioglu, C.; Stanley M., "A brief survey of machine learning methods and their sensor and IoT applications", In Proceedings of the 2017 8th International Conference on Information, Intelligence, Systems and Applications, IISA, Larnaca, Cyprus, 2017.

[41] Yang E., Ravikumar P., Allen G. I., Liu Z., "Conditional Random Fields via Univariate Exponential Families", Proceedings of Advances in Neural Information Processing Systems 26 (NIPS), 2013.

[42] Hammersley, J. M. & Clifford, P., "Markov fields on finite graphs and lattices", Computer Science, 1971.

[43] Liu, X., Wei, F., Zhang, S. & Zhou, M., "Named entity recognition for tweets", ACM Transactions on Intelligent Systems and Technology (TIST), 4(1):1524-1534, 2013.

[44] Abdel Rahman, S., Elarnaoty, M., Magdy, M. & Fahmy, A., "Integrated machine learning techniques for Arabic named entity recognition", International Journal of Computer Science Issues (IJCSI), 7(4):27-36, 2010.

[45] Yao, L., Sun, C., Li, S., Wang, X. & Wang, X., "CRF-based active learning for Chinese named entity recognition", Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics, pp. 1557-1561, 2009.

[46] Ammar, W., Dyer, C. & Smith, N.A., "Conditional random field auto encoders for unsupervised structured prediction", Proceedings of the Advances in Neural Information Processing Systems 26 (NIPS-2014), pp. 1-9, 2014.

[47] Pandian, S.L. & Geetha, T., "CRF models for tamil part of speech tagging and chunking", Proceedings of International Conference on Computer Processing of

Oriental Languages, pp. 11-22, 2009.

[48] Patel, C. & Gali, K., “Part-of-speech tagging for Gujarati using conditional random fields”, Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages, pp. 117-122, 2008.

[49] Jurafsky, D., and Martin, J. H., “Speech and Language Processing (3rd ed. draft)”, `https://web.stanford.edu/~jurafsky/slp3/`, 2019.

[50] Bikel, D. M., Schwartz, R. L., & Weischedel, R. M., “An algorithm that learns what’s in a name”, Machine Learning, 34: 211–231, 1999.

[51] Jurafsky, D. & James, H., “Speech and language processing an introduction to natural language processing, computational linguistics, and speech”, Publisher: Prentice Hall, United States of America, 2000.

[52] Singh, U., Goyal, V. & Lehal, G.S., “Named entity recognition system for Urdu”, Proceedings of COLING 2012 Technical Papers, pp. 2507-2518 2012.

[53] Morwal, S. & Chopra, D., “NERHMM: A tool for named entity recognition based on hidden Markov model”, International Journal on Natural Language Computing (IJNLC), 2:43-49, 2013.

[54] Morwal, S. & Jahan, N., “Named entity recognition using hidden Markov model (HMM): An Experimental Result on Hindi, Urdu and Marathi Languages”, International Journal of Advanced Research in Computer Science and Software Engineering, 3(4):671-675, 2013.

[55] Youzhi, Z., “Research and implementation of part-of-speech tagging based on hidden Markov model”, Proceedings of Asia-Pacific Conference on Computational Intelligence and Industrial Applications (PACIIA), pp. 26-29, 2009.

[56] Kolar, J. & Liu, Y., “Automatic sentence boundary detection in conversational speech: Across-lingual evaluation on English and Czech”, Proceedings of International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 5258-5261, 2010.

[57] Rehman, Z. & Anwar, W., “A hybrid approach for Urdu sentence boundary disambiguation”, International Arab Journal of Information Technology (IAJIT), 9(3):250-255, 2012.

[58] Gouda, A.M. & Rashwan, M., “Segmentation of connected Arabic characters using hidden Markov models”, Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA.), pp. 115-119, 2004.

[59] Wenchao, M., Lianchen, L. & Anyan, C., “A comparative study on Chinese word segmentation using statistical models”, Proceedings of IEEE International Conference on Software Engineering and Service Sciences (ICSESS), pp. 482 - 486, 2010.

[60] Jurafsky, D. & James, H., “Speech and language processing an introduction to natural language processing, computational linguistics, and speech”, Publisher: Prentice Hall, United States of America, 2000.

[61] Saha, S.K., Sarkar, S. & Mitra, P., “A hybrid feature set based maximum entropy Hindi named entity recognition”, Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages, pp. 343-349, 2008.

[62] Ekbal, A., Haque, R., Das, A., Poka, V., & Bandyopadhyay, S., “Language independent named entity recognition in Indian languages”, Proceeding of International Joint Conference on Natural Language Processing (IJCNLP), pp. 1-7, 2008.

[63] Cover, T. M., & Hart, P. E., “Nearest neighbour pattern classification”, IEEE Transactions on Information Theory, vol. 13(1), pp. 21-27, 1967.

[64] Peterson, L., “K-nearest neighbor”, Scholarpedia, vol. 4, p. 1883, 2009.

[65] Lifshits, Y., “Nearest neighbor search”, SIGSPATIAL, v. 2, p. 12, 2010.

[66] Qin, Z., Wang, A.T., Zhang, C., & Zhang, S., “Cost-sensitive classification with k-nearest neighbors”, In: Wang, M. (ed.) KSEM 2013. LNCS, vol. 8041, pp. 112–131. Springer, Heidelberg, 2013.

[67] Agrawal V., et al., “Application of K-NN regression for predicting coal mill related variables”, 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), India, pp. 1-9, 2016.

[68] Hinton, G., & Salakhutdinov, R., “Reducing the dimensionality of data with neural networks”, Science 313 № 5786, pp. 504-507, 2006.

[69] Han, J., Kamber, M. & Pei, J., “Data mining: concepts and Techniques”, Publisher: Elsevier, Amsterdam, Netherlands, 2006.

[70] Vedala, R. et al., “An application of Naive Bayes classification for credit scoring in elending platform”, ICDSE, pp. 81-84, 2012.

[71] Sunny, S., David Peter, S. & Jacob, K.P., “Combined feature extraction techniques and Naive Bayes classifier for speech recognition”, Computer Science & Information Technology (CS & IT), pp. 155-163, 2013.

[72] Bahl, L. R., de Souza, P. V., Gopalakrishnan, P. S., Nahamoo, D., & Picheny, M. A., “Context Dependent Modeling of Phones in Continuous Speech Using Decision Trees”, Proc. DARPA Speech and Natural Language Processing Workshop, pp 264–270, Pacific Grove, Calif., 1991.

[73] Kotsiantis, S.B., “Decision Trees: a recent overview”, Artif. Intell. Rev., 39, pp. 261–283, 2013.

[74] Boros, T., Dimitrescu, S. D., & Pipa, S., “Fast and Accurate Decision Trees for Natural Language Processing Tasks”, Proceedings of Recent Advances in Natural Language Processing, pp. 103-110, Varna, Bulgaria, 2017.

[75] Antony, P., Mohan, S.P., & Soman, K., “SVM based part of speech tagger for Malayalam”, Proceedings of IEEE International Conference on Recent Trends in Information, Telecommunication and Computing (ITC), pp. 339-341, 2010.

[76] Ekbal, A., & Bandyopadhyay, S., “Named entity recognition in Bengali: A multi-engine approach”, Proceeding of the Northern European Journal of Language Technology, pp. 26-58, 2009.

[77] Bindal, A., & Pathak, A., “A survey on k-means clustering and web-text mining”, IJSR, vol. 5, No 4, pp. 1049-1052, 2016.

[78] Sun, J., “Clustering Algorithms research”, J. Software, V. 19, 2008.

[79] Bouhmala, N., “How Good is the Euclidean Distance Metric for the Clustering Problem”, IIAI-AAI, Kummamoto, pp. 312-315, 2016.

[80] Gersho, A., & Gray, R. M., “Vector quantization and signal compression”, 6th ed. Boston, MA, United States: Kluwer Academic Publishers, 1991.

[81] Makhoul, J. et al., “Vector quantization in speech coding”, in Proceedings of the IEEE, vol. 73, no. 11, pp. 1551-1588, Nov. 1985.

[82] Linde, Y., Buzo, A., & Gray, R. M., “An Algorithm for Vector Quantization”, IEEE COM-28, No. 1, pp. 84-95, 1980.

[83] Spanias, A., Painter, T., & Atti, V., “Audio Signal Processing and Coding”, Wiley, March 2007.

[84] Spanias, A.S., “Speech Coding: A Tutorial Review”, Proceedings of the IEEE, Vol. 82, No. 10, pp. 1441-1582, 1994.

[85] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O., “Proximal Policy Optimization Algorithms”, arXiv preprint arXiv:1707.06347, 2017.

[86] Brin, S., “Extracting Patterns and Relations from the World Wide Web”, In Proceedings World Wide Web and Databases International Workshop, Number 1590 in LNCS, pp. 172-183. Springer, 1998.

[87] Agichtein, E., & Gravano, L., “Snowball: extracting relations from large plain-text collections”, Proceedings of the fifth ACM conference on Digital libraries, pp.85-94, 2000.

[88] Batista, D. S, Martins, B., & Silva, M. J., “Semi-Supervised Bootstrapping of Relationship Extractors with Distributional Semantics”, In Empirical Methods in Natural Language Processing. ACL, 2015.

[89] Wolpert, D. H., “Stacked Generalization”, Neural Networks, 5(2), 241-259, 1992.

[90] Kim, Y., “Convolutional neural networks for sentence classification”, in Conference on Empirical Methods in Natural Language Processing, pp. 1746-1751, 2014.

[91] Yin, W. & Schiitze, H., “Multichannel variable-size convolution for sentence classification”, in Conference on Computational Language Learning, pp. 204-214, 2015.

[92] Burukin, S., “NLP-based Data Preprocessing Method to Improve Prediction Model Accuracy”, `https://towardsdatascience.com/nlp-based-data-preprocessing-method-toimprove-prediction-model-accuracy-30b408a1865f`, 2019.

[93] Ding, X., He, Zha H., & Simon, H. D., “Adaptive dimension reduction for clustering high dimensional data”, in IEEE International Conference on Data Mining, pp. 147-154, 2002.

[94] Guillaumin, M. & Verbeek, J., “Multimodal semi-supervised learning for image classification”, in Computer Vision and Pattern Recognition, pp. 902-909, 2010.

[95] Kulis, B. & Basu, S., “Semi-supervised graph clustering: a kernel approach”, Machine Learning, vol. 74, pp. 1-22, 2009.

[96] Zhou, Z. H. & Li, M., “Semi-supervised regression with co-training”, in International Joint Conferences on Artificial Intelligence, pp. 908-913, 2005.

[97] Chen, P. & Jiao, L., “Semi-supervised double sparse graphs based discriminant analysis for dimensionality reduction”, Pattern Recognition, vol. 61, pp. 361-378, 2017.

[98] Bennett, K. P., & Demiriz, A., “Semi-supervised support vector machines”, in Advances in Neural Information processing systems, pp. 368-374, 1999.

[99] Cheung, E., “Optimization Methods for Semi-Supervised Learning”, University of Waterloo, 2018.

[100] Chapelle, O., Sindhwani, V., & Keerthi, S. S., “Optimization techniques for semi-supervised support vector machines”, Journal of Machine Learning Research, vol. 9, pp. 203-233, 2008.

[101] Li, Y. F. & Tsang, I. W., “Convex and scalable weakly labeled svms”, Journal of Machine Learning Research, vol. 14, pp. 2151-2188, 2013.

[102] Chapelle, O., Sindhwani, V., & Keerthi, S. S., “Branch and bound for semi-supervised support vector machines”, in Advances in Neural Information Processing Systems, pp. 217-224, 2007.

[103] Castro, V. & Yang, J., “A fast and robust general purpose clustering algorithm”, in Knowledge Discovery in Databases and Data Mining, pp. 208-218, 2000.

[104] Jolliffe, I., “Principal component analysis”, in International Encyclopedia of Statistical Science, pp. 1094-1096, 2011.

[105] Yao, M., “What are Important AI & Machine Learning Trends for 2020?”, https://www.forbes.com/sites/mariyayao/2020/01/22/what-are--important-ai--machinelearning-trends-for-2020/#3f46f07b2323, 2020.

[106] Zhuang, C., Zhai, A. L., & Yamins, D., “Local Aggregation for Unsupervised Learning of Visual Embeddings”, [CS, CV], https://arxiv.org/abs/1903.12355, 2019.

[107] Roufosse, J.-M., Sharma, A., & Ovsjanikov, M., “Unsupervised Deep Learning for Structured Shape Matching”, pp. 1617-1627, 2019.

[108] Entelo, “2018 Recruiting Trends Report”, https://cdn2.hubspot.net/hubfs/202646/Entelo%27s%202018%20Recruiting%20Trends%20Report.pdf?t=1530708036795, 2019.

[109] Glassdoor, “HR and Recruiting Stats for 2019”, 2019.

[110] Glassdoor, “Mission & Culture Survey 2019”, https://www.glassdoor.com/aboutus/app/uploads/sites/2/2019/07/Mission-Culture-Survey-Supplement.pdf, 2019.

[111] Jobvite, “Recruiter Nation Study 2018”, https://www.jobvite.com/wpcontent/uploads/2018/11/2018-Recruiter-Nation-Study.pdf, 2018.

[112] Piwiec, K., “75+ Human Resources Statistics for 2019 - DevSkiller”, https://devskiller.com/human-resources-statistics-2019, 2019.

[113] Agrawal, P., Abutarboush, H. F., Ganesh, T., & Mohamed, A. W., “Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019)”, in IEEE Access, vol. 9, pp. 26766-26791, doi: 10.1109/ACCESS.2021.3056407, 2021.

[114] Cherif, W., “Optimization of K-NN algorithm by clustering and reliability coefficients: application to breast-cancer diagnosis”, Procedia computer Science, 127, pp. 293-299, 2018.

[115] Goldberger, J., Hinton, G. E., Roweis, S. T., & Salakhutdinov, R. R., “Neighborhood component analysis”, In Advances in neural information processing systems, pp. 513-520, 2005.

[116] Kotsiantis, S.B., “Decision Trees: a recent overview”, Artif. Intell. Rev., 39, pp. 261–283, 2013.

[117] Mathieu-Dupas, E., “Weighted k-NN Algorithm and application in diagnosis. Algorithme des k plus proches voisins ponder´ es e application ´ en diagnostic”, In 42nd Days of Statistics, 2010.

[118] McGregor, M., “SVM Machine Learning Tutorial – What is the Support Vector Machine Algorithm, Explained with Code Examples”, https://www.freecodecamp.org/news/svm-machine-learning-tutorialwhat-is-the-support-vector-machine-algorithm-explained-with-codeexamples/, 2020.

[119] Salton, G. and Buckley, G., “Term-weighting Approaches in Automatic Text Retrieval”, Information Processing and Management, vol.24, no.5, pp.513-523, 1988.

[120] Bhatia, N., “Survey of nearest neighbor techniques”, arXiv preprint https://arxiv.org/abs/1007.0085, 2010.

[121] Bailey, T. & Jain , A. K., “A note on distance-weighted k-nearest neighbor rules”, IEEE Transactions on Systems, Man, and Cybernetics, (4), pp. 311-313, 1978.

[122] Yong, Z., Youwen, L. & Shixiong, X., “An improved k-NN text classification algorithm based on clustering”, Journal of computers, 4(3), pp. 230-237, 2009.

[123] Su, M. Y., “Using clustering to improve the k-NN-based classifiers for online anomaly network traffic identification”, Journal of Network and Computer Applications, 34(2), pp. 722-730, 2011.

[124] Mikolov,T., Chen,K., Corrado,G., & Dean,J., “Efficient Estimation of Word Representations in Vector Space”, Computation and Language, https://arxiv.org/abs/1301.3781v3, 2013.

[125] Rahutomo, F., Faisal & Kitasuka, Teruaki & Aritsugi, Masayoshi, “Semantic Cosine Similarity”, 2012.

[126] Chen, Tianqi & Guestrin, Carlos., “XGBoost: A Scalable Tree Boosting System”, pp. 785-794, https://doi.org/10.1145/2939672.2939785, 2016.

[127] Becker, S.G., “Human capital: A theoretical and empirical analysis”, Journal of Political Economy, 1975.

[128] Bandura, A., “Social foundations of thought and action”, Prentice Hall, Englewoods Cliffs, NJ, 1986.

[129] Van Vianen, Annelies., “Person–Environment Fit: A Review of Its Basic Tenets”, Annual Review of Organizational Psychology and Organizational Behavior, 5, pp. 75-101, https://doi.org/10.1145/2939672.2939785, 2018.

[130] Albert, E.T., “AI in talent acquisition: a review of AI-applications used in recruitment and selection”, Strategic HR Review, Vol.18 No. 5, pp. 215-221, https://doi.org/10.1108/SHR-04-2019-0024, 2019.

[131] M. L. Demircan, K. Aksac., “Prediction of the Future Success of Candidates Before Recruitment with Machine Learning: A Case Study in the Banking Sector”, https://link.springer.com/chapter/10.1007/978-3-031-09176-6_3, 2022.