Job-Candidate Classifying and Ranking System with Machine Learning Method

Miss Thapanee Boonchob

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2022

ระบบจำแนกและจัดอันดับผู้สมัครงานและงานที่สมัครด้วยการเรียนรู้ของเครื่อง

น.ส.ฐาปณีย์ บุญชอบ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2565

Thesis Title	Job-Candidate Classifying and Ranking System with Machine Learning Method
By	Miss Thapanee Boonchob
Field of Study	Computer Science
Thesis Advisor	NUENGWONG TUAYCHAROEN, Ph.D.

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Science

.... Dean of the FACULTY OF ENGINEERING

(Professor SUPOT TEACHAVORASINSKUN, D.Eng.)

THESIS COMMITTEE

.... Chairman

(PITTIPOL KANTAVAT, Ph.D.)

.... Thesis Advisor

(NUENGWONG TUAYCHAROEN, Ph.D.)

.... External Examiner

(Ratthaslip Ranokphanuwat, D.Eng.)

ฐาปณีย์ บุญชอบ : ระบบจำแนกและจัดอันดับผู้สมัครงานและงานที่สมัครด้วยการเรียนรู้ของเครื่อง. (Job-Candidate Classifying and Ranking System with Machine Learning Method) อ.ที่ปรึกษาหลัก : ดร.เนื่องวงศ์ ทวยเจริญ

การคัดเลือกผู้สมัครงานที่เหมาะสมสำหรับตำแหน่งงานที่บริษัทเปิดรับอาจเป็นงานที่ซ้ำซากและใช้เวลานาน โดยเฉพาะอย่างยิ่งจากงานที่มีผู้สมัครจำนวนมาก นอกจากนี้ งานนี้อาจทำให้การคัดกรองและการคัดเลือกอย่างยุติธรรมเป็นเรื่องที่น่าเบื่อหน่าย การสูญเสียโอกาสในการจ้างผู้สมัครงานที่มีความสามารถระดับสูงเนื่องจากกระบวนการคัดกรองที่ช้าหรือการเลือกผิดโดยความผิดพลาดของมนุษย์เป็นสิ่งที่ยอมรับไม่ได้ เอกสารนี้นำเสนอวิธีการสำหรับฝ่ายทรัพยากรบุคคลในการจำแนกและคัดเลือกผู้สมัครงานที่มีความสามารถอันดับต้นๆ สำหรับงานที่เปิดรับสมัคร ระบบที่นำเสนอจะจำแนกผู้สมัครงานจากการเรียนรู้ของเครื่องออกเป็นกลุ่ม i) เหมาะสม และ ii) ไม่เหมาะสม โดยวิธีการประมวลผลข้อมูลที่มีประสิทธิผลของงานที่เกี่ยวข้องจะถูกนำมาประยุกต์ใช้ในงานนี้ด้วย 8 แบบจำลองจะถูกนำมาเปรียบเทียบเพื่อหารูปแบบการจำแนกที่เหมาะสมที่สุด ดังนี้ ต้นไม่ตัดสินใจ (Decision Tree) ซัพพอร์ตเวกเตอร์แมทชีน (Support Vector Machine) การจำแนกแบบเบย์ด้วยการแจกแจงแบบปกติ (Gaussian Naive Bayes) การสุ่มป่าไม้ (Random Forest) การหาเพื่อนบ้านใกล้ที่สุด (K Nearest Neighbor) แคทบูสท์ (CatBoost) เอกซ์จีบูสท์ (Extreme Gradient Boosting) และโครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolution Neural Network) จากนั้นระบบจะจัดอันดับผู้สมัครในกลุ่มเหมาะสมจากมากไปหาน้อย ระบบที่นำเสนอมีค่าความถูกต้อง 83.5% ค่ามัชฌิมฮาร์มอร์นิกถ่วงน้ำหนัก 86% และค่าการจำได้ 79% จากแบบจำลองซัพพอร์ตเวกเตอร์แมทชีน ระบบที่นำเสนอนี้จะช่วยให้ธุรกิจสามารถระบุผู้สมัครงานที่เหมาะสมสำหรับตำแหน่งใดตำแหน่งหนึ่ง และตัดสินใจได้จากการวิเคราะห์ข้อมูลว่าควรคัดเลือกใครเข้ารับการสัมภาษณ์งาน

สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์	ลายมือชื่อนิสิต ....
ปีการศึกษา	2565	ลายมือชื่อ อ.ที่ปรึกษาหลัก ....

6470167221 : MAJOR COMPUTER SCIENCE

KEYWORD: candidate classifying, candidate ranking, cosine similarity, machine learning

Thapanee Boonchob : Job-Candidate Classifying and Ranking System with Machine Learning Method. Advisor: NUENGWONG TUAYCHAROEN, Ph.D.

Finding suitable candidates for an open job position could be a repetitive and time-consuming task, especially from a large pool of candidates. Besides, this task could truly make fair screening and shortlisting tedious. Losing the opportunity to hire top talent candidates due to the slow screening process or the wrong selection by human error is unacceptable. This paper presented a method for human resources to categorize and select the top candidates for job opening they applied for. The proposed system directed to alter a machine learning algorithm to classify the candidate into groups i) shortlist and ii) not-suitable. The productive preprocessing data approaches of many works were applied. The Decision Tree, Support Vector Machine, Gaussian Naive Bayes, Random Forest, k-Nearest Neighbour, CatBoost, Extreme Gradient Boosting, and Convolution Neural Network were compared to find the most suitable classification model. Then, the system ranked the candidates in a shortlist group in descending order. The proposed system operates an accuracy of 83.5%, weighted f1-score of 86%, and recall of 79% from the Support Vector Machine classifier. This enables the business to identify suitable candidates for a certain position and make more informed decisions about who to invite for an interview.

Field of Study:	Computer Science	Student's Signature ....
Academic Year:	2022	Advisor's Signature ....

ACKNOWLEDGEMENTS

We would like to express our appreciation to the Faculty of Engineering for approving this work. Additionally, we would like to extend their gratitude to the STelligence Company Limited for their assistance with data preparation. Last but not least, we would like to thank everyone who was involved in this work, whether directly or indirectly.

Thapanee Boonchob

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Early on in the pandemic coronavirus disease 2019 (COVID-19), there is a significant turnover rate. In April 2020 [1], The workforce noticed an increase in the number of unemployed people with an unemployment rate of more than 15%. This percentage has dropped to 5.2 percent as of August 2021, indicating that most employees are no longer at risk of losing their jobs, but recruitment and personnel retention are now pressing concerns for many businesses.

The voluntary quit rate was increasing as the work-from-home lengthens due to the COVID-19 pandemic. In April 2021, 4 million employees voluntarily quit their employment. Another 3.9 million workers voluntarily left the company in June 2021, bringing the total to 3.9 million. Due to the large number of open roles created by this massive migration, business executives are scrambling to retain their finest and most difficult-to-replace employees.

According to job seeker jobsDB statistics from January to March 2020 [2], the website was seen more than 3.6 million times, with more than 1.6 million unique visitors, over 31.4 million page views, and an average of more than 20 minutes per person of website traffic.

From the preceding paragraph, this can be observed in the increased number of employment applications. The process of selecting persons to interview becomes more time-consuming. It also poses the risk that good people will leave because of the longer processes and the loss of skilled people leading to the cost of working chances rising. This is since other businesses are also competing for talent.

According to the Interview Success Formula [3], 80% of job applicants who submit resumes will not be invited for an interview. As can be seen, Human Resources (HR) must carefully analyze the selection of candidates to be invited for an interview out of the entire number of job applicants. This requires a significant amount of time and consideration. There are many factors HR must consider, the Jobvite 2021 Recruiter Nation Report [4] explores what HR considers about a resume, including technical skills, experience, salary background, education history or degrees, and grade point average (GPA). In addition, HR may consider additional certificates, special training courses, languages, computer, and technology skills, etc.

In its 2018 Eye-Tracking Study [5], Ladders Inc. revealed that recruiters now scan resumes for an average of 7 seconds, compared to just 6 seconds in 2012. Recruiters today just skim at resumes for an average of 7.4 seconds. However, is it too rapid to choose people for a job interview from a resume in 7.4 seconds? It would have been possible if they had taken 7.4 seconds, but only for the first qualifying round. If the number of applicants who pass the first round of selection outnumbers the number of calls for interviews. Of course, there will be a second round of qualifying that is more comprehensive.

Experienced human resources professionals were questioned by Mark Slack, a certified professional resume writer, regarding resume screening, how long they typically spend looking at a resume and their opinions of the "6-second" rule. The responses are listed below: [6]

1. "…Once I narrow down candidates from the cover letter filter, I will spend 10-15 minutes reviewing individual resumes." by Kim Kaupe, Co-Founder, ZinePak / The SuperFan Company.
2. "The 6-second rule? It varies from company to company. Here's what I'll say. Recruiters will spend less time reading a résumé for an entry or junior-level role. Positions that are more senior will be reviewed quite carefully by HR before they pass them on to the hiring manager." by Glen Loveland, HR Manager, CCTV.
3. "Initially, an average resume takes 2-3 minutes for me to scan." by Heather Neisen, HR Manager, Technology Advice.

It may be concluded that, depending on the company and its HR, considering inviting people to interview takes more or less time. However, it is a repetitious task that is subject to human error or bias. If the procedure takes a lengthy period, it will also have an impact on other processes of recruiting certainly. Statistics from Glassdoor indicate that each company's job offer attracts 250 resumes [7]. Four to six of those candidates will be contacted for interviews, but only one will be chosen for the position. This is another example of how all resumes might take a long time to be considered for one position, and this may make the process of recruiting people to work more time-consuming. The Jobvite 2021 Recruiter Nation Report found that only 16% of recruiters are filling jobs in less than 14 days, but more than 54% and 21% of recruiters are filling jobs in 14-30 days and 31-60 days respectively.

Consequently, we will explore a system to classify candidates who apply for a job into 2 groups: shortlist items and not-suitable items and rank the score of candidates in the shortlist group for an interview to work for HR. Before making a final choice, the system is separated into two parts, with the results of both parts being used to rank the candidates who should be invited for an interview in descending order. The first part will be to provide demographic data into the supervised learning model, such as gender, age, education, and so on, for the model to learn whether to invite for an interview or not. This research will not use personally identifiable information, such as name, surname, and identification card number. The second part is to input skills and experiences data into the similarity function, which will rank candidates who have similar skills and experiences to the job description for the position the company wants to fill. The first part's findings will be used to categorize candidates, while the second part's results will be used to rank scores for each candidate invited to an interview.

This research aims to create an automated job-candidate classifying and ranking system that will help to shorten the amount of time it takes to classify the list of candidates to interview and select the right candidates for the desired position based on both skills, experiences, and demographics.

1.2 Objective

The primary goals of this research are

1. Develop an automated job-candidate classifying and ranking system using supervised learning techniques.
2. Find the most relevant resume with job posting-based similarity functions.
3. Study supervised learning techniques for structure data.
4. Study neural network techniques for structure data.

1.3 Scope of Work

This research focuses on developing an automated job-candidate classifying and ranking system as a case study on 3 aspects, supervised learning, neural network, and similarity function.

For the supervised learning and neural network, focus on the demographic of candidates to predict who should be called for a job interview. The results of these aspects classified applicants into two classes: those who were called for a job interview and those who were not. The expected model accuracy rate is 90%.

For the Similarity function, focus on skills and experiences that are most related to the job description. The result of this aspect was to rank candidates with similar skills and experiences as the job description for the position the company was looking for.

STelligence Company Limited's data was extracted for the research, with a total of 2027 job applicants and 36 job openings. There were 1040 applicants who filled out the experience information and 987 applicants who did not fill out the experience information. The following are the examples of information from the candidate's side, such as previous job title, skills, experience, year of experience, education, major, university, gender, age, certificate, language, and expected salary. The following are the examples of information from the company such as job title, job description, qualifications, and work location. The programming language in this research is python 3.8.

1.4 Contributions

1. A proposed system for determining which candidates should be contacted for an interview.
2. Job-candidate classifying and ranking algorithm by using a supervised model, convolutional neural network, and a cosine similarity algorithm.

1.5 Research Methodology

1. Study relevant materials and existing research.
2. Data understanding by exploring the data.
3. Study the supervised learning process to create a model for classifying candidates by using candidates' demographic.
4. Study the neural network process to create a model for classifying candidates by using candidates' demographic.
5. Study the similarity function to rank candidates using candidates' skills and experiences.
6. Design a preliminary process including measurements to evaluate the ability and efficiency of the training models.
7. Develop the job-candidates classifying and ranking system.
8. Evaluate the system from the test set.
9. Conclusion of research results.
10. Compile and prepare academic articles.

1.6 Thesis Organization

The rest of the research is organized as follows. Chapter 2 presents the theoretical background. Chapter 3 describes the literature review, followed by a proposed method in Chapter 4. Chapter 5 contains the result and discussion. Section 6 is about the conclusion and future work.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Machine Learning

Machine learning is the science and art of teaching computers to recognize patterns in data. Here is a definition that might be more common: “Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.” by Arthur Samuel, 1959 [8], and a more engineering-oriented one: “A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.” by Tom Mitchell, 1997 [7].

Machine learning systems come in a wide variety, which is helpful to categorize target data into categories. They can be divided into four types depending on whether they receive training under human supervision.

2.1.1 Supervised Learning

In supervised learning, the algorithm gets a training set that contains labels for the desired target. Classification is a classic supervised learning task. For example, if we want to educate kindergarten children on how to classify cats and dogs, children should be taught by showing them pictures of a cat and dog and telling them what picture is a cat and what picture is a dog. Continue to teach in this manner until the children can accurately respond, as shown in Figure 1.

Regression is a common activity that involves predicting a desired numerical number, like the cost of a car, using a set of factors like miles, age, brand, etc. You must provide the model with several examples of cars, together with their labels and features, to train it. Here are a few of the well-known supervised learning algorithms:

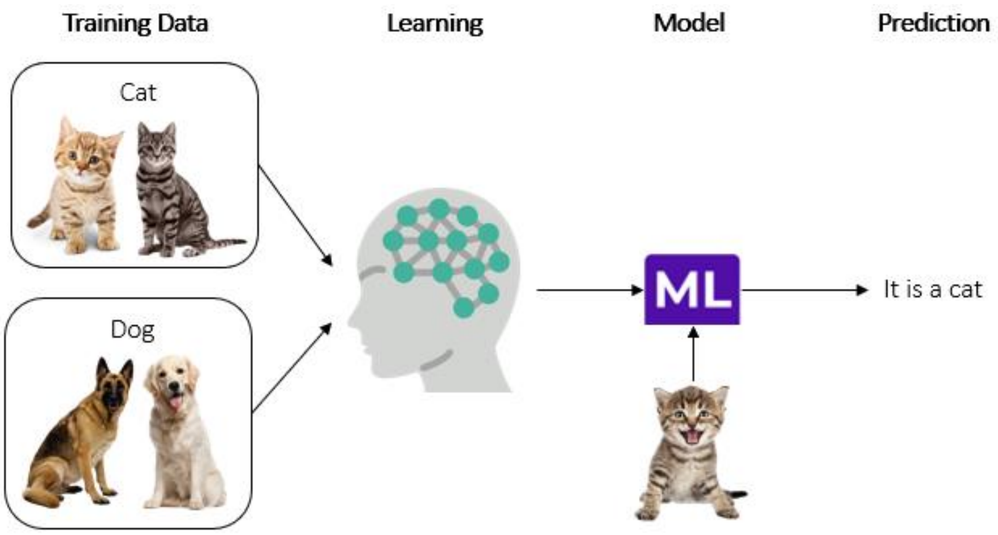


Figure 1 - Example of classification in supervised learning

2.1.1.1 Decision Tree (DT)

A non-parametric supervised learning technique for classification and regression is the decision tree (DT). The purpose is to create a model that predicts the value of a target variable using basic decision rules discovered from the features of the data. The decision tree has a tree structure similar to a flow diagram, where each internal node is a trial on the remaining features, each branch is an experiment result, and each leaf node is a class label. The root node is selected from the highest information gain feature [9], which is a method to calculate how much information a feature provides about classes. The equation of information gain is shown in Equation 1.

$$Information\ Gain = E_{parent} - E_{children} \quad (1)$$

Where E in Equation 1 means entropy, a measure to estimate how impurity of observations is. It controls the way a decision tree chooses how to split data. The equation of entropy is shown in Equation 2 where p_i means the probability of selecting an example in class i.

$$Entropy = \sum_{i=1}^{N} p_i log_2 p_i \quad (2)$$

2.1.1.2 Support Vector Machine (SVM)

SVM uses hyperplane to classify multidimensional data with the support of kernel functions. There are different types of kernels in SVM such as polynomial, radial, and linear kernels. Another data tuning parameter is called regularization (C), which allows for the decision of how strongly to penalize misclassified points. If we assign C to a large value that has the same meaning as a low-margin hyperplane, this means that we want the least amount of misclassification in the training set which could lead to overfitting [8]. On the other hand, If we assign C to a lower value that has the same meaning as a high-margin hyperplane, this means that we accept the amount of misclassification in the training set that could lead to underfitting. Therefore, for the model to be effective, we should define the C properly. The kernel function's width is specified by a gamma parameter. When the gamma value is low, the decision boundary is quite broad so far-away data points are also considered. Alternatively, when the gamma value is high, the decision boarder is quite narrow so only nearby data points are considered. Figure 2 shows a sample illustration to understand SVM in its entirety [10].

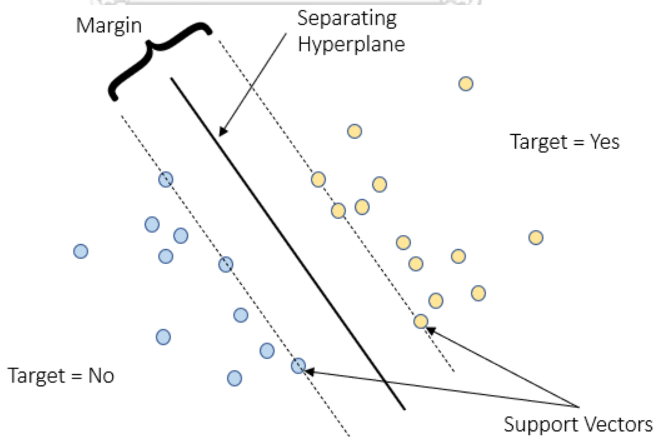


Figure 2 - Hyperplane, support vectors, and margin

2.1.1.3 Gaussian Naive Bayes (GNB)

The Gaussian normal distribution is supported by and followed by the Naive Bayes variant GNB. This enables the conversion of each z-score distance into a p-value. The GNB predicts the likelihood that a specific data item will fall under a certain category for each class in the dataset. The most likely class of data points is the one with the highest membership probability [11]. Figure 3 shows how the GNB classifier works [12].

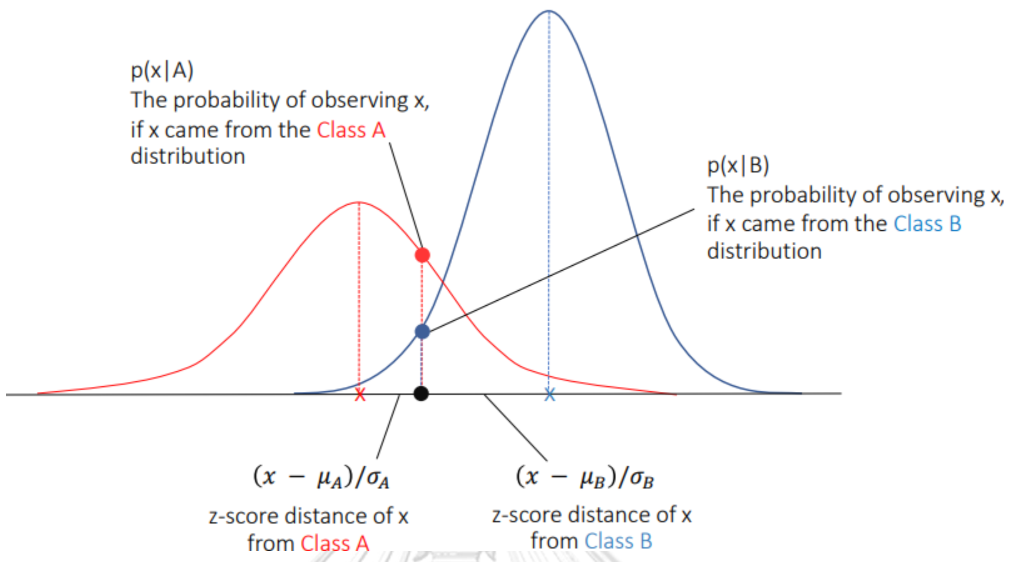

p(x|A)
The probability of observing x, if x came from the Class A distribution
p(x|B)
The probability of observing x, if x came from the Class B distribution
$(x - \mu_A)/\sigma_A$
z-score distance of x from Class A
$(x - \mu_B)/\sigma_B$
z-score distance of x from Class B

Figure 3 - How Gaussian Naive Bayes classifier works

Where μ and σ are the mean and variance of predictor distribution while $P(x|y)$ can be calculated from Equation 3.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma^2}\right) \tag{3}$$

2.1.1.4 Random Forest (RF)

Random Forest is a compilation or assemblage of classification and regression trees produced via resampling at random from the training set. By sampling with replacement to construct unique trees, it employs bootstrap aggregating, often referred to as bagging, to produce subsets of the training data [13] as shown in Figure 4 [14]. To categorize incoming input data, each tree predicts one class, and the forest chooses the class with the most votes [15] as shown in Figure 5.

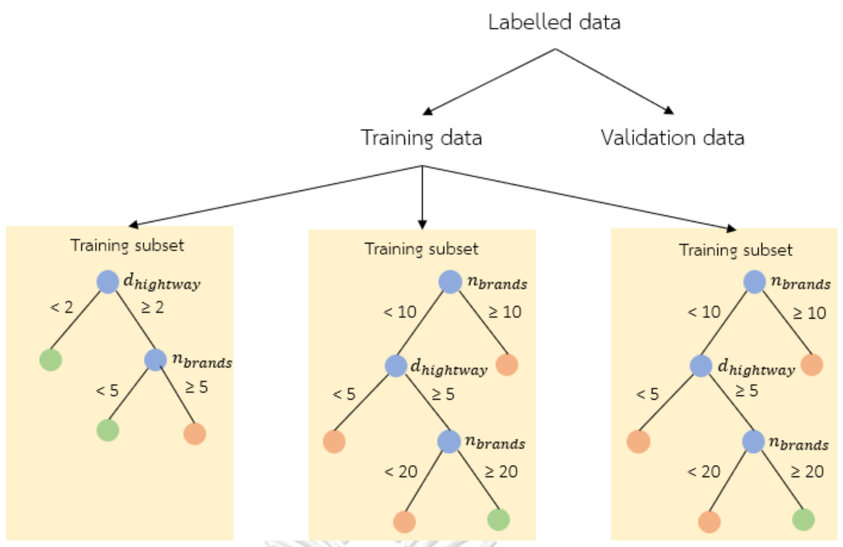


Figure 4 - The example of random forest classification

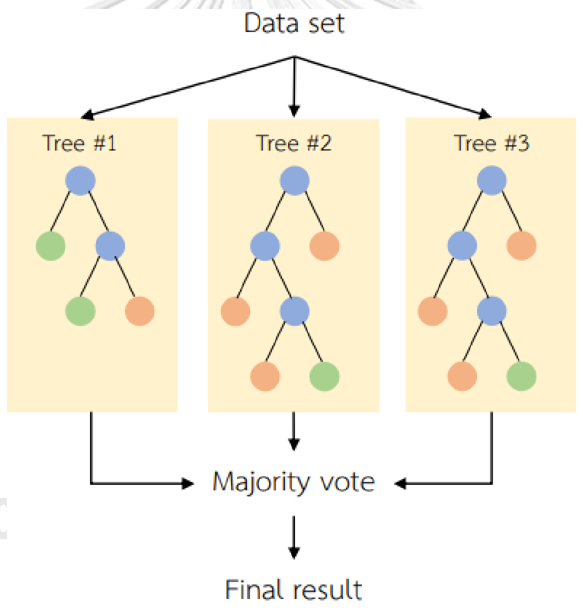


Figure 5 - Majority vote of random forest classification

2.1.1.5 k-Nearest Neighbor (k-NN)

k-NN classifies new data by its neighbor's majority vote. The neighbors were selected from the top k nearest distance between that new data and its neighbors. A distance function that uses the Minkowski distance method, Manhattan distance, or Euclidean distance measures the distance. The number of neighbors is represented by the k value that will be used as references in the majority vote to classify data points. The results will be less stable if the k value is very low. On the other hand, the error can be increased by increasing the k value, but stable results will be obtained. Figure 6 shows an example of the k-NN algorithm [9]. If we choose k = 1, the new data point will be predicted to be class 1 because its nearest neighbor is in class 1. On the other hand, if we choose k = 3, class 2 will be predicted for the new data point because its top nearest neighbor is in class 1, but the next nearest 2 neighbors are in class 2 so this majority vote is class 2.

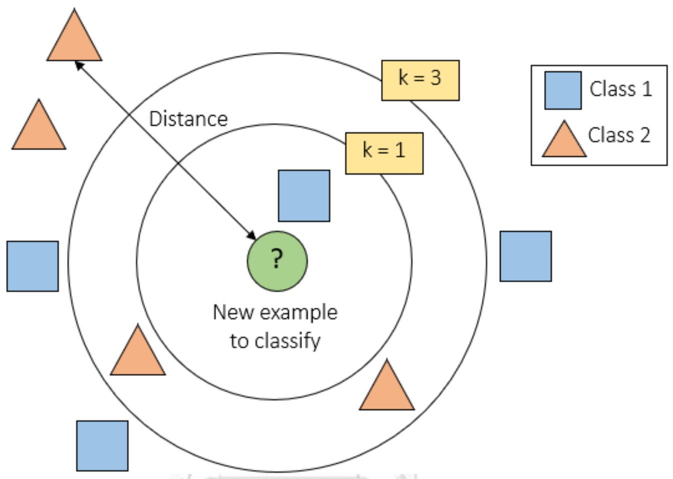


Figure 6 - The example of k-Nearest Neighbor classification

2.1.1.6 Catboost

Another machine learning method that is effective at classifying category features is CatBoost [16]. The name "CatBoost" is a combination of the phrases "Category" and "Boosting". CatBoost works well with many other data “categories”, such as audio, historical data, image, and text data. And “boosting” came from an implementation of gradient boosting which employs base predictors that are binary decision trees [17]. This learning task's purpose is to promote a function that discovers the minimal expected loss. The obvious difference between CatBoost and other algorithms is the data do not need to be converted to any specific formats to use CatBoost.

2.1.1.7 Extreme Gradient Boosting (XGBoost)

XGBoost is a boosted tree algorithm that follows the gradient boosting principle through parallel processing, tree-pruning, handling missing values, and regularization to prevent overfitting and bias [16]. In XGBoost, decision trees are generated sequentially. Before being fed into the decision tree that forecasts the result, each independent variable is given a weight. In addition, before being placed into the second decision tree, variables that the tree mistakenly anticipated are given extra weight. These unique classifiers are then combined to produce an accurate and robust model, so XGBoost is one of the efficient gradient-boosting algorithms.

2.1.2 Unsupervised Learning

The training data in unsupervised learning is not labeled. Without a teacher, the system tries to teach itself as shown in Figure 7.

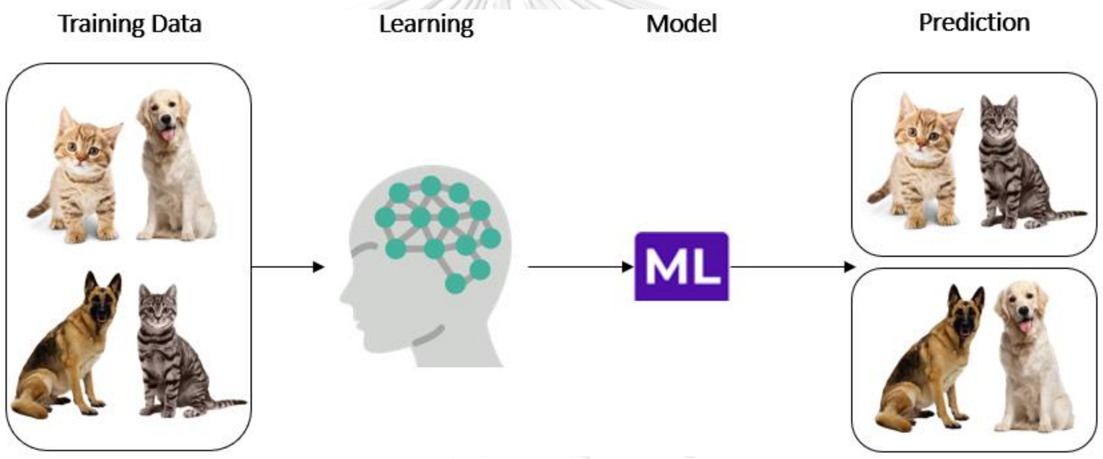


Figure 7 - Example of clustering in unsupervised learning

The most significant unsupervised learning algorithms are listed here:

2.1.2.1 Clustering

Dividing the population or set of data points into various groups is the goal of clustering. Each group's data points are more similar to one another and distinct from those in the other groups. Clustering clusters objects based on how similar and dissimilar they are to one another. Data analysis, consumer segmentation, recommender systems, search engines, image segmentation, and other applications benefit greatly from clustering [18]. Two popular clustering algorithms are k-Means and Density-Based Spatial Clustering of Applications with Noise (DBSCAN).

2.1.2.2 Anomaly Detection

Finding data samples that are notably different from the majority of data instances is the process of anomaly detection. Other names for it include novelty discovery or outlier detection. Finding abnormalities in the data may be helpful immediately or as a starting point for new knowledge discovery. In many applications, anomaly detection is essential including applications for security, critical infrastructure, and health [19]. Figure 8 shows an example of an anomaly detection application for health that detects diseases or anomalies in chest radiographs.

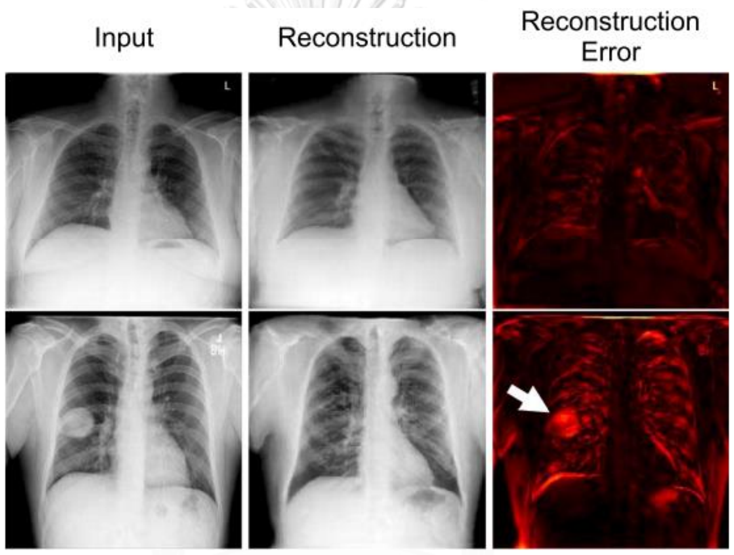


Figure 8 - Chest radiographs anomaly detection [20]

2.1.2.3 Association Rule Learning

Association rule learning is one of the very important concepts of machine learning. It is a machine-learning technique that uses rules to find important connections among variables. It is not only for retail and supermarkets but includes the field of web analytics such as tracking, learning, and predicting user behavior on websites [18]. Examples of association rule algorithms are Apriori and Equivalence Class Clustering and bottom-up Lattice Traversal (ECLAT).

2.1.3 Semi-supervised Learning

There will be a data set that has numerous unlabeled occurrences and few labeled ones because data labeling is often time-consuming and costly. Data that is only partially labeled can be handled by some algorithms, and semi-supervised learning is the term for this type of learning. We can apply semi-supervised learning with many use cases such as image and speech analysis because audio and image files typically lack labels. Labeling is a laborious task that is expensive as well. You can label a limited set of data using human skills. Once the data has been trained, we may use semi-supervised learning to label the remaining audio and image files, which will enhance the speech and image analytic models.

2.1.4 Reinforcement Learning

Reinforcement learning is referred to as an agent that can study its surroundings, choose and carry out activities, and receive rewards or punishments in the form of unfavorable rewards in return. It must determine for itself the appropriate course of action, or policy, for maximizing reward. A protocol defines the plan of action the agent should take [18]. Examples of reinforcement learning applications in real life such as self-driving cars, AlphaGo, and robot motion control.

2.2 Deep Learning

Deep Learning (DL) is an execution of Artificial Intelligence (AI). The main goal of AI is to act and think as humans do in all aspects. It enhances computers' ability to perform tasks automatically by leaning on previous cases of experience. Deep Learning (DL) consists of three groups of layers. The first group is the input layer, which is responsible for receiving data to be processed. The second group is the hidden layer. A noticeable difference between deep learning and machine learning about the hidden layer is deep learning has more than one hidden layer for more complex processing, while machine learning has only one hidden layer. The last group is the output layer, which is causing the showing of the result after being processed. Deep learning is often applied with the development of software that can gather data and use that data for learning [21].

2.2.1 Convolution Neural Network (CNN)

CNN is a type of deep learning model. This mathematical construction typically consists of three different kinds of layers: convolution, pooling, and fully connected layer. Layers of convolution and pooling extract features that can increasingly and hierarchically become more complex. The output of one layer is fed onto the next layer, whereas the retrieved features are mapped into the output using a fully connected layer as shown in Figure 9. The core of the CNN architecture is the convolution layer, which updates learnable parameters by backpropagation with a gradient descent optimization technique after feature extraction using forward propagation on a training dataset, such as kernels and weights. The dimensionality of the feature is decreased via a pooling layer to extract only important parts of the data, optimize processing faster, and lower the number of subsequently learnable parameters. The last convolution or pooling layer's output is often flattened, or changed into a one-dimensional array of integers, and connected to a fully connected layer, also known as a dense layer. After that, the final fully connected layer is mapped to output such as the likelihoods of each class in challenges involving classification [22].

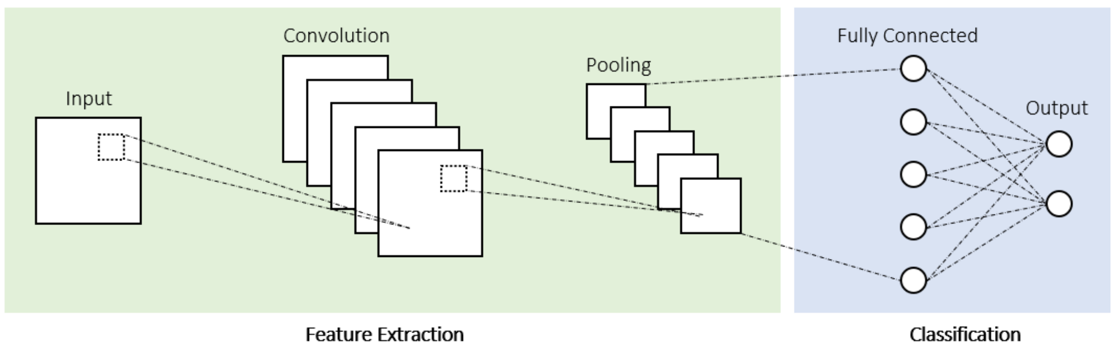
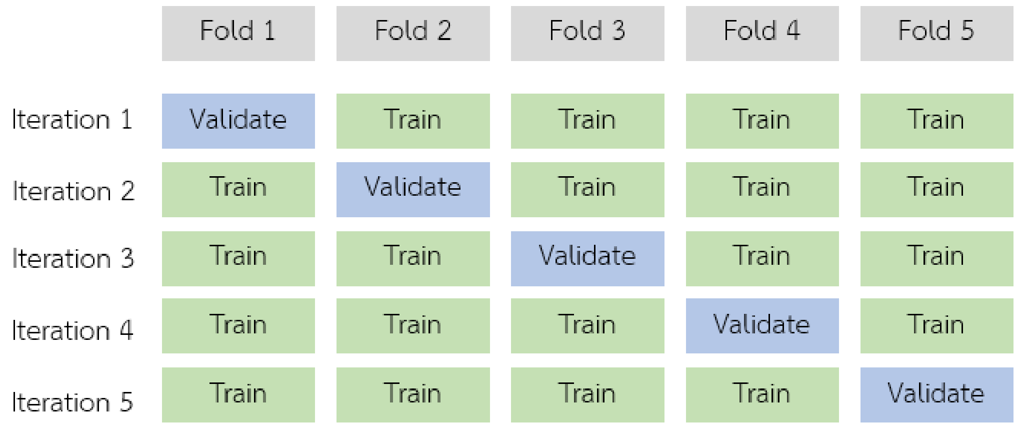


Figure 9 - Convolution neural network layers

2.3 Cross-validation

A resampling approach called cross-validation is used to assess machine learning models on a collection of data. The k parameter of the process determines how many groups should be formed from a specific sample of data. Consequently, the process is sometimes referred to as “k-fold cross-validation”. When a k value is established, it can be used in place of k in the reference of the model, such as when k=5 denotes 5-fold cross-validation.

The two main processes in cross-validation are folding the data and cycling between training and evaluation on each fold as shown in Figure 10. Each fold is roughly the same size, allowing for stratification, which means each fold has the same proportion of observations with a given label as shown in Figure 11. On the other hand, you can split the data by random selection. All folds are used to train the model, except for one that will be rotated as a validation fold, so that each fold has only ever behaved as a validation fold once. The accuracy of each fold will be calculated as the average accuracy for the entire data.



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Iteration 1	Validate	Train	Train	Train	Train
Iteration 2	Train	Validate	Train	Train	Train
Iteration 3	Train	Train	Validate	Train	Train
Iteration 4	Train	Train	Train	Validate	Train
Iteration 5	Train	Train	Train	Train	Validate

Figure 10 - 5-fold cross-validation

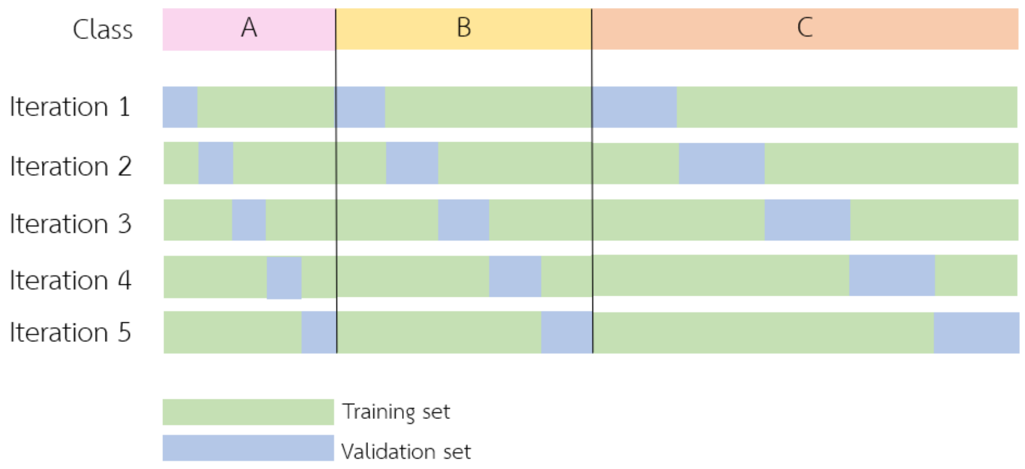


Figure 11 - Stratified 5-fold cross-validation

2.4 Classification Model Evaluation

2.4.1 Confusion Matrix

When we solve classification problems, both for binary classification and multiclass classification, a confusion matrix is a common measurement to describe the performance of a model with testing data as shown in Figure 12.

		Predicted Class	
		Negative	Positive
Actual Class	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Figure 12 - Confusion matrix

The counts between actual and expected values are displayed in the confusion matrix. The outcome “TN” stands for the true negative and displays the quantity of accurately predicted negative class cases. Like this, how many positive classes were predicted with accuracy cases is shown by the abbreviation “TP”, which stands for true positive. The number of genuine negative cases for which erroneous positive predictions were made is indicated by the letter “FP”, which appears as a false positive, while “FN” implies a false negative, which means the number of real positive cases that were mistakenly predicted as negative [23].

2.4.2 Classification Report

A classification report is a performance evaluation metric report in machine learning. Precision, recall, F1-score, and support in each class are the main 4 metrics that are presented.

Its correctness can be evaluated by its precision. It is described as the proportion of true positives to all true and false positives for each class. The other way refers to the percentage of all samples that were predicted to be positive. The equation of precision is shown in Equation 4.

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

A recall gauges a classifier's ability to correctly predict each positive sample. It is defined as the ratio of true positives to the total of true positives and false negatives for each class. The other way means what percentage of all samples that were positive were accurately predicted. The equation of precision is shown in Equation 5.

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

The F1-score, with 1.0 representing the best result and 0.0 the worst, is a weighted harmonic mean of recall and precision. The equation of precision is shown in Equation 6.

$$F1\ score = \frac{2 * Precision * Recall}{Precision+Recall} \tag{6}$$

Support in a dataset is the proportion of real samples in each class. In the training data, the necessity for stratified sampling or rebalancing may be suggested by imbalanced support which may point to structural problems in the model [24].

CHAPTER 3

LITERATURE REVIEW

Related literature for this research is focused on resume-job matching to locate the best candidates for the open position. Each relevant study employs a different approach to analyze and identify potential candidates. In any case, they significantly shorten the time spent selecting a candidate for HR.

3.1 Matching Applicants with Positions for Better Allocation of Employees in the Job Market

The main processes in this system architecture are depicted in Figure 13, divided into five sections. First, data collection, data about resumes was gathered from various sources and job postings were collected from indeed.com. Second, all important information was taken from both resumes and job postings by using the SpaCy NLP library. Third, all letters were converted to lowercase, unnecessary words and spaces were removed, certain special characters were mapped, and others were removed. Fourth, by matching and scoring by using the SpaCy similarity function, it was determined how similar each resume category was to its corresponding one in the job offer. The last one is ranking and reporting, the results are used to put the candidates who are best qualified at the top and the candidates who are least qualified at the bottom. System accuracy was 83 percent overall [25].

We then use the same method for collecting job postings and job applicant data from this paper, but we choose to retrieve from different job sites, using jobsdb.co.th instead of indeed.com.

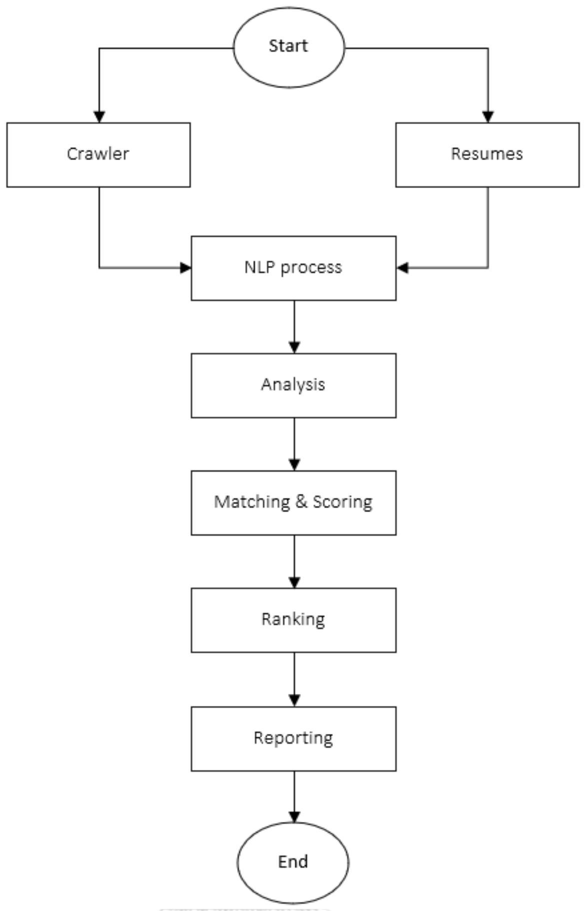


Figure 13 - Recommendation system

3.2 Feature Selection for Job Matching Application using Profile Matching Model

This research aims to choose the key features required for job matching. Using the candidate profile and company profile factors results in the development of match-related people and positions. The system works by gathering data, which are age, gender, military status, highest education level, experience, and years of experience for the candidate data. On the other hand, the researchers collected job titles, the minimum highest education level, age, gender, military status, experience, and years of experience for job posting data. Processing integration involved information extraction, which included cleaning and integrating all obtained data to create a profile-matching model and data categorization. To determine whether a candidate's performance is appropriate for a certain group, the company's information and the candidate's information are combined and grouped into several clusters. For analysis, the proposed approach calculated scores of similarity between the two profiles. The resulting similarity scores are then adjusted to have more accurate values that account for the weighting given to each attribute. In decision-making, whether the two profiles being compared are the same is what is returned by computing the weighted similarity scores. According to the ranking of the selected features, age and gender are ranked lower than a job title, work experience, highest education level, and military status, which each have an attribute value of 58.33. Which age and gender got only 41.66. Only openings that are present in large clusters will be eligible for matching by the matching algorithm, which will be able to identify these clusters. The recommendation of candidates to the company is executed when the matching is completed [26], as shown in Figure 14.

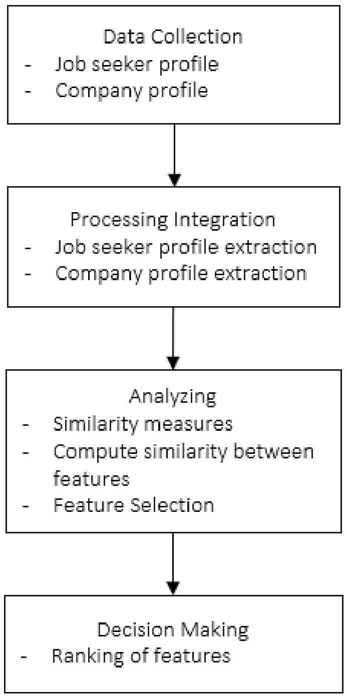


Figure 14 - Profile matching framework

We will choose to select high-attribute-value features from this paper for the data preparation process of this research. This includes job title, work experience, highest education level, age, and gender information, as well as key features that HR considers about a resume from the Jobvite 2021 Recruiter Nation Report.

3.3 Design and Development of Machine Learning based Resume Ranking System

Finding the topmost qualified candidates for a certain job position is the purpose of this research based on two factors: (1) the candidate's abilities will be used to evaluate the test, and (2) The candidate's resume's experience must align with the company's criteria.

This research proposed a system, shown in Figure 15, that comprises two major sections. The first part is the candidate screening part, used to evaluate the candidate's qualifications using the multiple-choice question (MCQ) test. Candidates can submit resumes once they reach a minimal score. But, if they fall short of a minimal score, they will not be permitted to submit a resume.

Second, in the resume screening and ranking part, white spaces, digits, and stop words, e.g., and, etc. are removed from resumes. The words in the resumes are then converted to vectors using term frequency-inverse document frequency (TF-IDF) vectorization. TF-IDF vectorizer is also used to transform the text in the job description (JD) into vectors. The next step is to find resumes that closely match the JD provided by the recruiters using the k-nearest neighbors (k-NN) technique after the cosine distance is used to gauge how similar the resume and the JD offered are to one another.

To begin, this system used “Gensim”, an open-source library, to summarize the JD and resumes. Within the word limit, the information provided was summarized by this library. Before calculating the cosine similarity between the JD and resumes, the model combined the JD and the cleaned resume data into a single data set. Ranking resumes according to the job description following the similarity score achieved. The cosine similarity value of k-NN was utilized to get resumes that intimately matched the specified JD. Recruiters are given recommendations for the top-n ranked resumes based on assigned ranks. The system's scoring accuracy averages 92 percent and parsing accuracy is 85 percent [27].

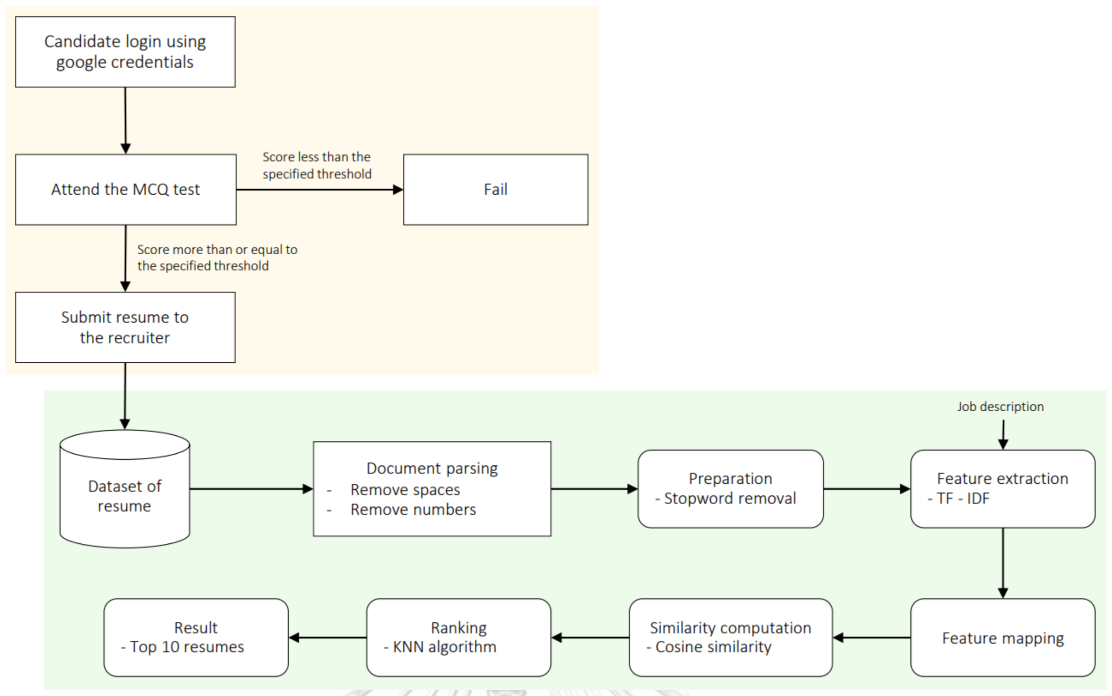


Figure 15 - Machine learning-based resume ranking system

We will take some steps from this research, which is to remove space, numbers, and stop words in the document parsing and preparation process respectively, and to find similarities between JD and resume using cosine similarity.

3.4 A Machine Learning Approach for Automation of Resume Recommendation System

The goal of this research is to choose from a broad pool of resumes the finest candidates. The proposed model primarily operated in two steps: (1) prepare and (2) deploy and inference. It can be seen in Figure 16.

In the preprocessing part, The inputted resumes will be cleaned to get rid of any special or unnecessary characters. Additionally, during cleaning, all special characters, numerals, and words that consist of only a single letter are eliminated. For the next step, stop words like and, the, was, etc. that commonly appear in the text but are useless for prediction are consequently removed. Stemming, which distills word morphology to its basic forms, and lemmatization, dropping the s, es, or ing at the end of a word to confirm that the word has nothing to do with it, are the next steps. The final step is feature extraction, which the researchers accomplished with the TF-IDF. As a result, the texts are converted to the desired vector length using the scikit-learn library to construct a TF-IDF vector, they calculate TF-IDF for each term in their dataset.

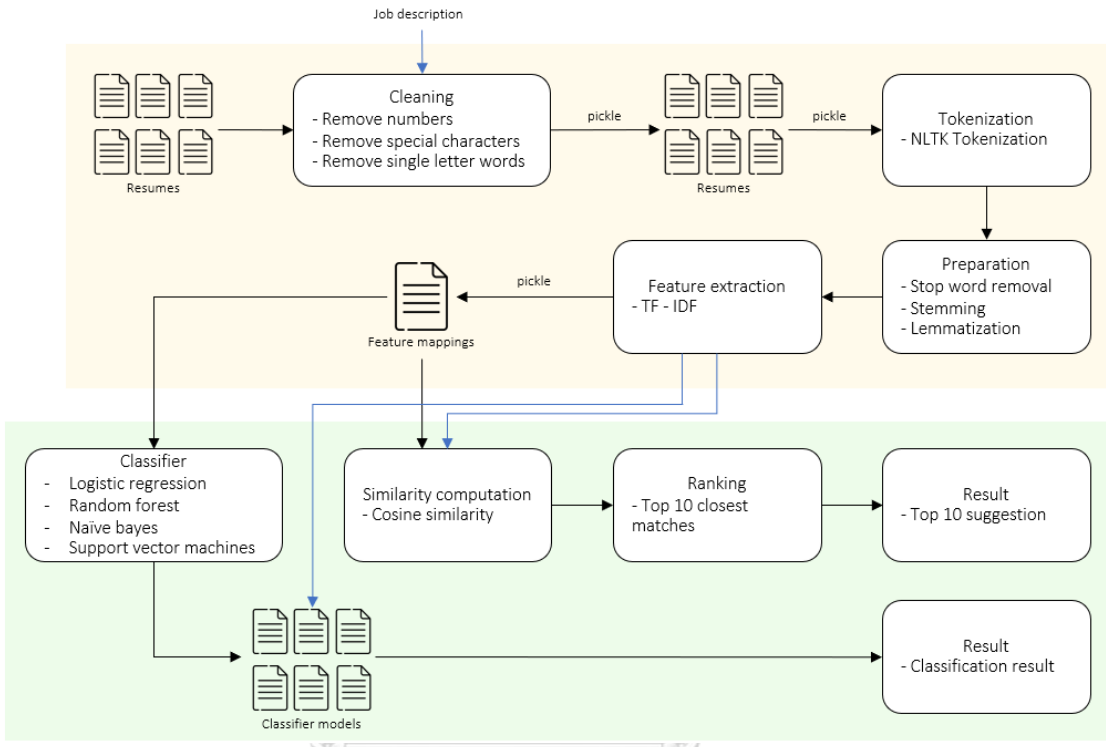


Figure 16 - Recommendation system

In the deploy and inference part, The model would provide resumes that were relevant to the JD after comparing the tokenized resumes data with the JD. On the cleansed data, two models have been developed: (1) classification, the model was created to place the resume into the relevant category. (2) recommendation, based on the JD supplied by HR and the resume's comparability.

For the classification model, A selection of the most relevant resumes was produced by the model, which included defined a summary of the CV and JD. Using 10-fold cross-validation, the average accuracy score of the linear support vector classifier was calculated to be 78.53 percent, which is higher than that of random forest, multinomial naive bayes, and logistic regression.

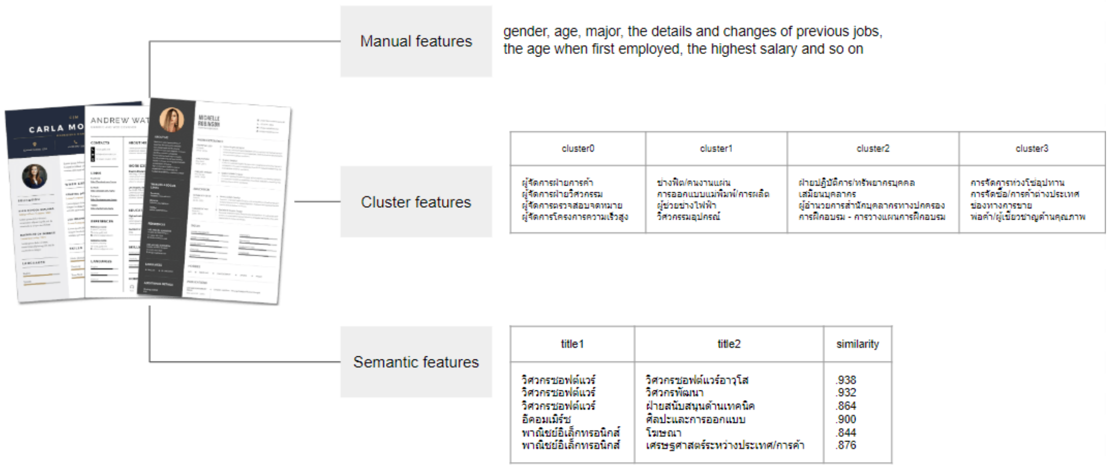
A job description and resumes were input into the recommendation algorithm, which then created a list of resumes that most closely matched the job description.

Two methods are used to do this: (1) Use of cosine similarity for content-based recommendations: the model combined the JD and cleansed resume data into a single data set before calculating the cosine similarity between them. (2) k-NN: the provided text was summarized by the “Gensim” library within the word limit, and then the purpose system used k-NN to find closely matched resumes with the provided JD [28].

The cleaning, tokenization, and preparation in preprocessing part will be applied in this proposed research. This research includes using cosine similarity to calculate a similarity between JD and resumes in the deploy and inference part. The classification model is another step that will be used in this research as in this paper, but with different purposes. In this paper, the classification model is used to divide resumes into working categories (multiclass classification), such as accounting groups, computer groups, engineering groups, etc., while this research will build a classification model to predict whether the candidates are suitable for being called in for a job interview or not (binary classification).

3.5 Machine Learned Resume-Job Matching Solution

The information extraction from the resume is the first step in this paper which can be categorized into 3 types of features. First, manual features, which are gender, age, major, the details and alterations of prior employment, the age at which one was hired, the highest salary, and so on. Numerical keys were used during training instead of the character feature values that were entered into a dictionary. The authors used the entire text of the résumé to train a Word2Vec model for cluster features. Additionally, a phrase's semantic meaning can now be represented via a phrase's typical word embeddings. Using the K-mean algorithm, sentences were divided into 64 and 128 clusters. The writers turned a resume's previous work experience into an ordered list of phrases for the final feature category, the semantic feature. Each phrase would be presented by a vector of 10 dimensions and then calculated with the similarity of vectors. An example of all features is shown in Figure 17.


Manual features
gender, age, major, the details and changes of previous jobs, the age when first employed, the highest salary and so on
Cluster features
Semantic features

cluster0	cluster1	cluster2	cluster3
ผู้จัดการฝ่ายการค้า	ช่างฟิต/คนงานแผ่น	ฝ่ายปฏิบัติการ/ทรัพยากรบุคคล	การจัดการห่วงโซ่อุปทาน
ผู้จัดการฝ่ายวิศวกรรม	การออกแบบแม่พิมพ์/การผลิต	เสมียนบุคลากร	การจัดซื้อ/การค้าต่างประเทศ
ผู้จัดการตรวจสอบจดหมาย	ผู้ช่วยช่างไฟฟ้า	ผู้อำนวยการสำนักบุคลากรทางปกครอง	ช่องทางการขาย
ผู้จัดการโครงการความเร็วสูง	วิศวกรรมอุปกรณ์	การฝึกอบรม - การวางแผนการฝึกอบรม	พ่อค้า/ผู้เชี่ยวชาญด้านคุณภาพ

title1	title2	similarity
วิศวกรซอฟต์แวร์	วิศวกรซอฟต์แวร์อาวุโส	.938
วิศวกรซอฟต์แวร์	วิศวกรพัฒนา	.932
วิศวกรซอฟต์แวร์	ฝ่ายสนับสนุนด้านเทคนิค	.864
อีคอมเมิร์ซ	ศิลปะและการออกแบบ	.900
พาณิชย์อิเล็กทรอนิกส์	โฆษณา	.844
พาณิชย์อิเล็กทรอนิกส์	เศรษฐศาสตร์ระหว่างประเทศ/การค้า	.876

Figure 17 - Example of the manual, cluster, and semantic features

This research using the IBagging method improved from the Bagging method by voting based on the overall probability of each option [29].

We will take some features in the manual and semantic features from this paper to prepare for the research such as gender, age, major, and so on. In addition, we will look at three models that the authors used to train data for this paper: XGBoost, Random Forest, and Convolutional Neural Network and apply them to train the data for this research.

3.6 Embedding-based Recommender System for Job to Candidate Matching on Scale

This research proposed a candidate matching system which is a two-stage recommendation process made up of two main parts. In the first stage retrieval component, hundreds of candidates were selected from a pool of millions using a two-tower embedding structure. The second stage reranked components based on various contextual factors. The applicants were whittled down to a few dozen at this point. The authors proposed a fused embedding technique to learn representations from raw text, parsed text, and geolocation for both applicants and jobs in the first component, as shown in Figure 18.

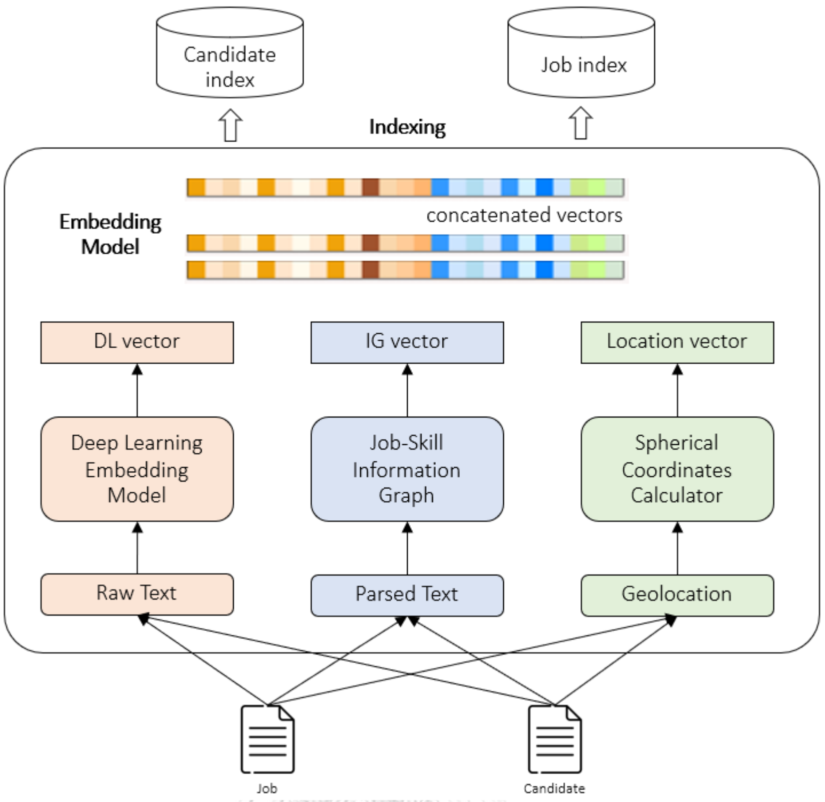


Figure 18 - Embedding-based recommender system

For the deep-learning embedding model, the word2vec model is used to express words in vectors from job-candidate text pairs, which are subsequently sent to the convolutional neural network with the attention layer. The outcome is a context vector, which is then used to generate an embedding vector through the dropout layer, fully connected layer, and RELU activation. Job-skill information graph is used to learn the depiction of job title and skill, respectively. They converted the latitude and longitude indicated in spherical coordinates to cartesian coordinates in the spherical coordinate calculator part [30].

In this research, the method for training a convolutional neural network and calculating distance will be applied.

3.7 Comparing BERT against traditional machine learning text classification

This research compares classic machine learning methods that train machine learning algorithms in features derived from the data by the TF-IDF algorithm with bidirectional encoder representations from transformers (BERT) from 4 datasets on text classification, both binary and multiclass. For TF-IDF, the authors used TfidfVectorizer from the sklearn library to preprocess the text, then predicted the data using Predictor from the auto_ml module and H2OAutoML from the h2o module respectively to find the best model for that data. For BERT, they used the pre-trained BERT model from the pre-trained module. The conclusions drawn from the comparison in this research found that BERT achieves higher accuracy than traditional methods in every dataset [31].

For the proposed method, BERT will be applied to the vectorization process for similarity computation and ranking suggestions to see if BERT can do better on this dataset than TF-IDF.

According to the research, the majority of solutions have their limitations or can only be used in certain situations to complete a particular goal. In this thesis, we aim to demonstrate the combination of those methodologies so that the proposed system can be generalized and more effective.

The proposed method will extract data from a website, JobsDB, then clean the data before processing. Auto-filling is one step in the processing stage to handle missing data. After that, we will convert the experience text data to vectors using the bidirectional encoder representations from transformers (BERT) pre-trained model and calculate the similarity between the experience and the responsibilities of the job that the candidates applied for. On the other hand, we will classify the target group using a supervised learning model into 2 groups which are not-suitable items and shortlist items. The similarity score will be used to rank the candidates from the shortlist group in descending order. Finally, we will explore features that have an impact on target prediction. Table 1 summarizes the similarities and differences between the methods used by the authors in previous papers including the proposed method.

Table 1 - Method comparisons

Step	[25]	[26]	[27]	[28]	[29]	[30]	[31]	Proposed method
Crawler data	✓	✓						✓
Clean data	✓	✓	✓	✓	✓	✓		✓
Vectorization			✓	✓		✓	✓	✓
Similarity computation	✓	✓	✓	✓	✓			✓
Work type classification				✓				
Target group classification								✓
Candidates ranking	✓		✓	✓	✓	✓		✓
Features ranking		✓						✓

CHAPTER 4

PROPOSED METHOD

This proposed method aims to develop an automated job-candidate classifying and ranking system using supervised learning techniques. The method, as shown in Figure 19, starts with scraping candidate and job data from the JobsDB website, then proceeds to preprocess using multiple techniques, and the result from the preprocessing stage will be sent to the prediction stage.

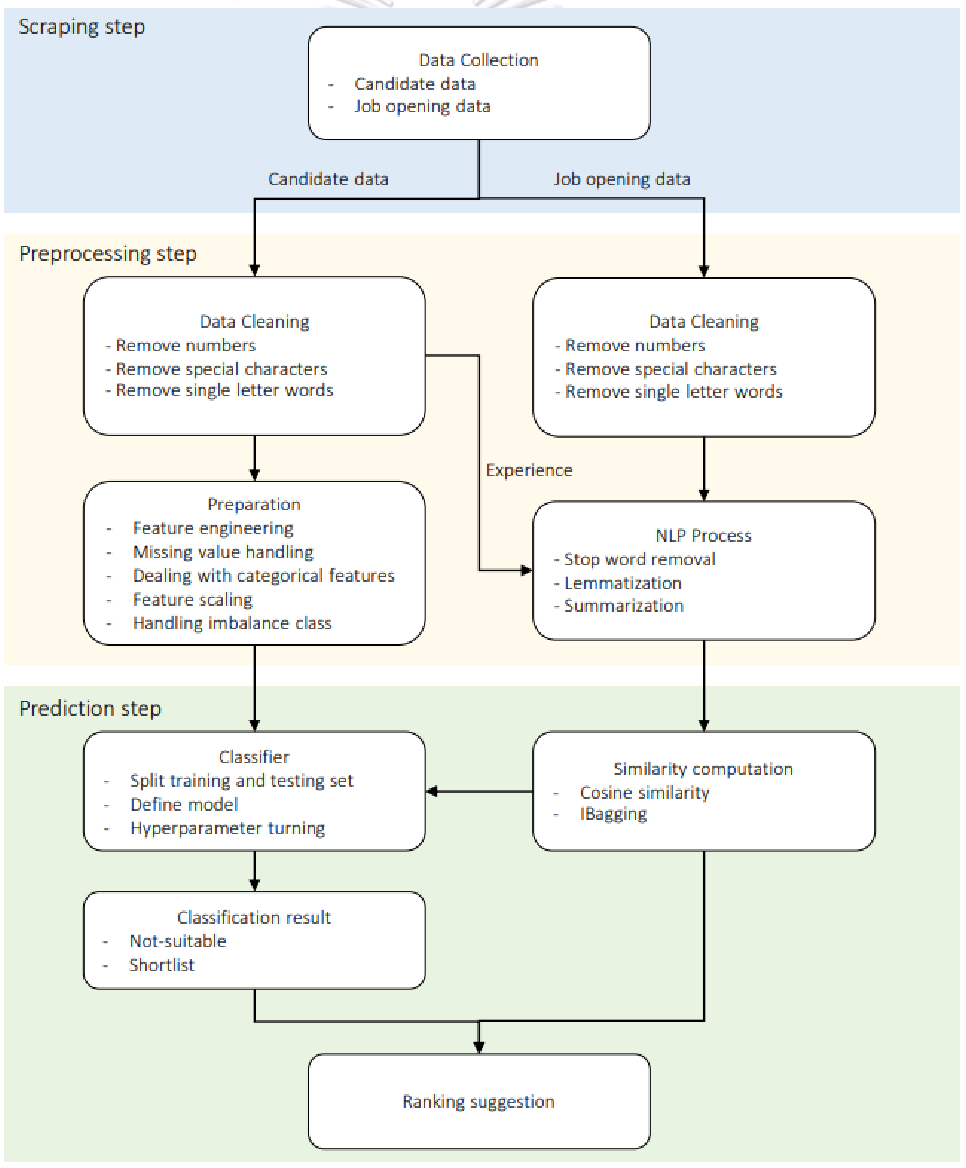
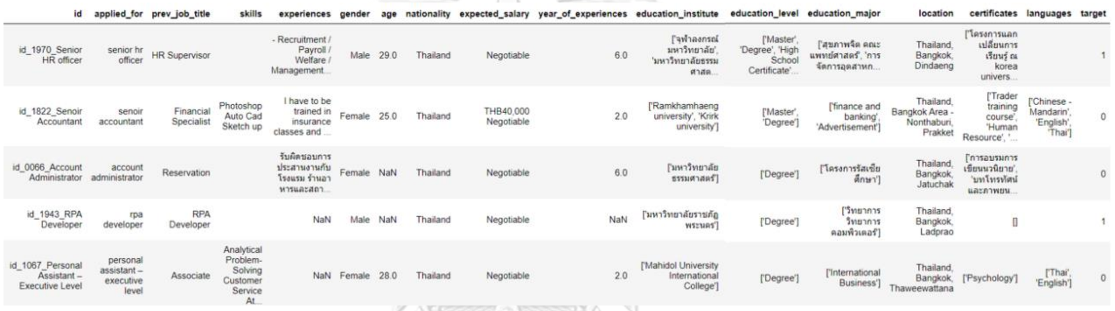


Figure 19 – Overview of the proposed system

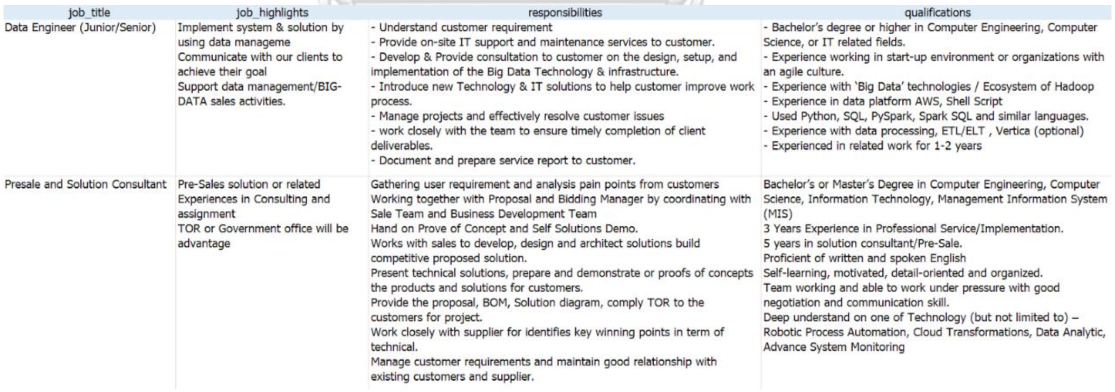
4.1 Data Scraping

Data was collected from JobsDB.com excluding personally identifiable data, such as name, surname, and identification card number. The data is divided into two parts containing 2027 job applicants who applied for 36 job postings at STelligence Company Limited. There were 1040 applicants who filled out the experience information and 987 applicants who did not. The data is presented in both Thai and English, so we need to convert all Thai text into English before using these data in the following step. Figure 20 and Figure 21 showed the example of candidate and job posting data respectively.



id	applied_for	prev_job_title	skills	experiences	gender	age	nationality	expected_salary	year_of_experiences	education_institute	education_level	education_major	location	certificates	languages	target
id_1970_Senior HR officer	senior hr officer	HR Supervisor		- Recruitment / Payroll / Welfare / Management...	Male	29.0	Thailand	Negotiable	6.0	['จุฬาลงกรณ์มหาวิทยาลัย', 'มหาวิทยาลัยธรรมศาสต...	['Master', 'Degree', 'High School Certificate'...	['สุขภาพจิต คณะแพทยศาสตร์', 'การจัดการอุตสาหก...	Thailand, Bangkok, Dindaeng	['โครงการแลกเปลี่ยนการเรียนรู้ ณ korea univers...		1
id_1822_Senoir Accountant	senoir accountant	Financial Specialist	Photoshop Auto Cad Sketch up	I have to be trained in insurance classes and ...	Female	25.0	Thailand	THB40,000 Negotiable	2.0	['Ramkhamhaeng university', 'Krirk university']	['Master', 'Degree']	['finance and banking', 'Advertisement']	Thailand, Bangkok Area - Nonthaburi, Prakket	['Trader training course', 'Human Resource', '...	['Chinese - Mandarin', 'English', 'Thai']	0
id_0066_Account Administrator	account administrator	Reservation		รับผิดชอบการประสานงานกับโรงแรม ร้านอาหารและสถา...	Female	NaN	Thailand	Negotiable	6.0	['มหาวิทยาลัยธรรมศาสตร์']	['Degree']	['โครงการรัสเซียศึกษา']	Thailand, Bangkok, Jatuchak	['การอบรมการเขียนนวนิยาย', 'บทโทรทัศน์และภาพยน...		0
id_1943_RPA Developer	rpa developer	RPA Developer		NaN	Male	NaN	Thailand	Negotiable	NaN	['มหาวิทยาลัยราชภัฏพระนคร']	['Degree']	['วิทยาการ วิทยาการคอมพิวเตอร์']	Thailand, Bangkok, Ladprao	[]		1
id_1067_Personal Assistant – Executive Level	personal assistant – executive level	Associate	Analytical Problem-Solving Customer Service At...	NaN	Female	28.0	Thailand	Negotiable	2.0	['Mahidol University International College']	['Degree']	['International Business']	Thailand, Bangkok, Thaweewattana	['Psychology']	['Thai', 'English']	0

Figure 20 - Example of candidate data



job_title	job_highlights	responsibilities	qualifications
Data Engineer (Junior/Senior)	Implement system & solution by using data manageme Communicate with our clients to achieve their goal Support data management/BIG-DATA sales activities.	- Understand customer requirement - Provide on-site IT support and maintenance services to customer. - Develop & Provide consultation to customer on the design, setup, and implementation of the Big Data Technology & infrastructure. - Introduce new Technology & IT solutions to help customer improve work process. - Manage projects and effectively resolve customer issues - work closely with the team to ensure timely completion of client deliverables. - Document and prepare service report to customer.	- Bachelor's degree or higher in Computer Engineering, Computer Science, or IT related fields. - Experience working in start-up environment or organizations with an agile culture. - Experience with 'Big Data' technologies / Ecosystem of Hadoop - Experience in data platform AWS, Shell Script - Used Python, SQL, PySpark, Spark SQL and similar languages. - Experience with data processing, ETL/ELT , Vertica (optional) - Experienced in related work for 1-2 years
Presale and Solution Consultant	Pre-Sales solution or related Experiences in Consulting and assignment TOR or Government office will be advantage	Gathering user requirement and analysis pain points from customers Working together with Proposal and Bidding Manager by coordinating with Sale Team and Business Development Team Hand on Prove of Concept and Self Solutions Demo. Works with sales to develop, design and architect solutions build competitive proposed solution. Present technical solutions, prepare and demonstrate or proofs of concepts the products and solutions for customers. Provide the proposal, BOM, Solution diagram, comply TOR to the customers for project. Work closely with supplier for identifies key winning points in term of technical. Manage customer requirements and maintain good relationship with existing customers and supplier.	Bachelor's or Master's Degree in Computer Engineering, Computer Science, Information Technology, Management Information System (MIS) 3 Years Experience in Professional Service/Implementation. 5 years in solution consultant/Pre-Sale. Proficient of written and spoken English Self-learning, motivated, detail-oriented and organized. Team working and able to work under pressure with good negotiation and communication skill. Deep understand on one of Technology (but not limited to) – Robotic Process Automation, Cloud Transformations, Data Analytic, Advance System Monitoring

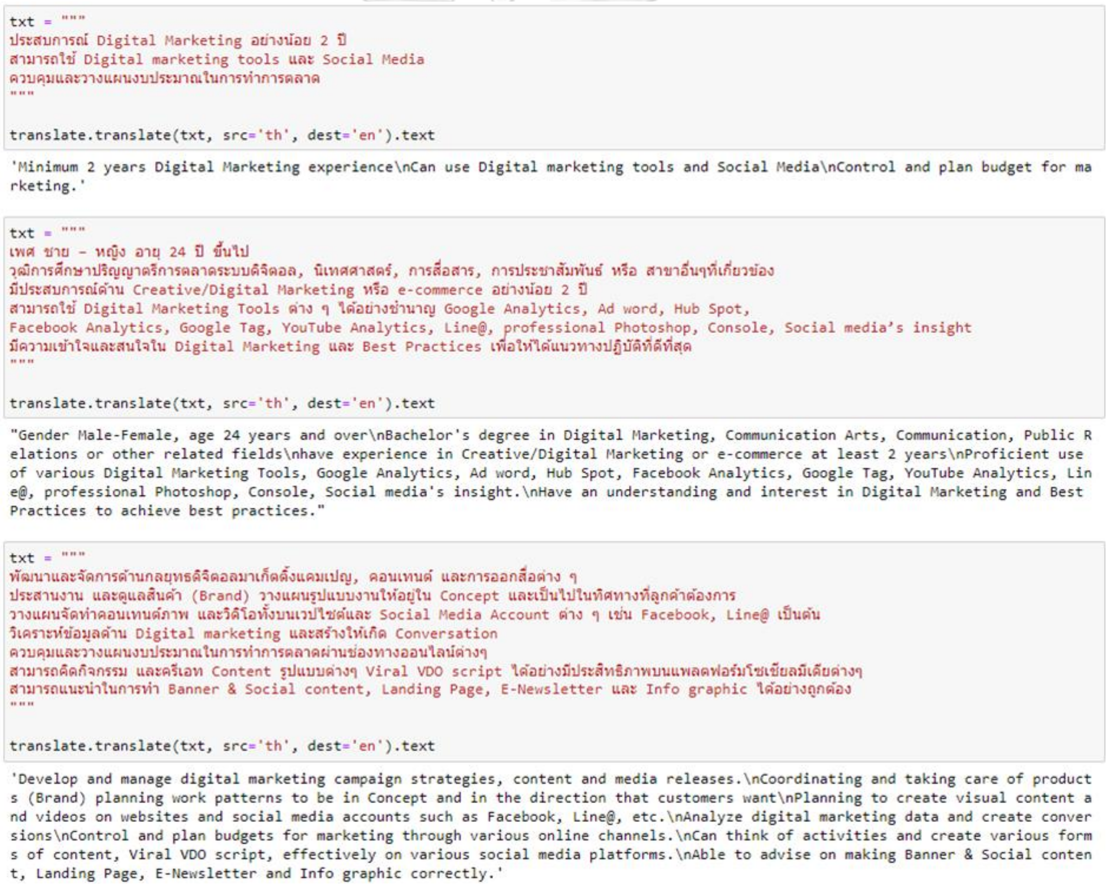
Figure 21 - Example of job opening data

4.2 Data Preprocessing

The most time-consuming process is data preprocessing or data cleaning. When high-quality data is entered into the model, the model's output is also of high quality. However, if the data fed to the model is of bad quality, the model's output will be poor as well. As quoted in “Garbage in, garbage out”, by Nick Harkaway, The Gone-Away World.

4.2.1 Natural Language Processing (NLP) Process

Before stepping into the NLP process, we need to translate Thai to English text for the data that candidates fill in the form in the Thai language first by using The Google Translate Application Program Interface (API) in Python. This API is well-known and trustworthy whether it is used for translating short or long text messages. The examples of translation from Thai to English text are shown in Figure 22.



```
txt = """
ประสบการณ์ Digital Marketing อย่างน้อย 2 ปี
สามารถใช้ Digital marketing tools และ Social Media
ควบคุมและวางแผนงบประมาณในการทำการตลาด
"""

translate.translate(txt, src='th', dest='en').text
```

```
'Minimum 2 years Digital Marketing experience\nCan use Digital marketing tools and Social Media\nControl and plan budget for ma
rketing.'
```

```
txt = """
เพศ ชาย - หญิง อายุ 24 ปี ขึ้นไป
วุฒิการศึกษาปริญญาตรีการตลาดระบบดิจิตอล, นิเทศศาสตร์, การสื่อสาร, การประชาสัมพันธ์ หรือ สาขาอื่นๆที่เกี่ยวข้อง
มีประสบการณ์ด้าน Creative/Digital Marketing หรือ e-commerce อย่างน้อย 2 ปี
สามารถใช้ Digital Marketing Tools ต่าง ๆ ได้อย่างชำนาญ Google Analytics, Ad word, Hub Spot,
Facebook Analytics, Google Tag, YouTube Analytics, Line@, professional Photoshop, Console, Social media’s insight
มีความเข้าใจและสนใจใน Digital Marketing และ Best Practices เพื่อให้ได้แนวทางปฏิบัติที่ดีที่สุด
"""

translate.translate(txt, src='th', dest='en').text
```

```
"Gender Male-Female, age 24 years and over\nBachelor's degree in Digital Marketing, Communication Arts, Communication, Public R
elations or other related fields\nhave experience in Creative/Digital Marketing or e-commerce at least 2 years\nProficient use
of various Digital Marketing Tools, Google Analytics, Ad word, Hub Spot, Facebook Analytics, Google Tag, YouTube Analytics, Lin
e@, professional Photoshop, Console, Social media's insight.\nHave an understanding and interest in Digital Marketing and Best
Practices to achieve best practices."
```

```
txt = """
พัฒนาและจัดการด้านกลยุทธดิจิตอลมาเก็ตติ้งแคมเปญ, คอนเทนต์ และการออกสื่อต่าง ๆ
ประสานงาน และดูแลสินค้า (Brand) วางแผนรูปแบบงานให้อยู่ใน Concept และเป็นไปในทิศทางที่ลูกค้าต้องการ
วางแผนจัดทำคอนเทนต์ภาพ และวิดีโอทั้งบนเวปไซต์และ Social Media Account ต่าง ๆ เช่น Facebook, Line@ เป็นต้น
วิเคราะห์ข้อมูลด้าน Digital marketing และสร้างให้เกิด Conversation
ควบคุมและวางแผนงบประมาณในการทำการตลาดผ่านช่องทางออนไลน์ต่างๆ
สามารถคิดกิจกรรม และครีเอท Content รูปแบบต่างๆ Viral VDO script ได้อย่างมีประสิทธิภาพบนแพลตฟอร์มโซเชียลมีเดียต่างๆ
สามารถแนะนำในการทำ Banner & Social content, Landing Page, E-Newsletter และ Info graphic ได้อย่างถูกต้อง
"""

translate.translate(txt, src='th', dest='en').text
```

```
'Develop and manage digital marketing campaign strategies, content and media releases.\nCoordinating and taking care of product
s (Brand) planning work patterns to be in Concept and in the direction that customers want\nPlanning to create visual content a
nd videos on websites and social media accounts such as Facebook, Line@, etc.\nAnalyze digital marketing data and create conver
sions\nControl and plan budgets for marketing through various online channels.\nCan think of activities and create various form
s of content, Viral VDO script, effectively on various social media platforms.\nAble to advise on making Banner & Social conten
t, Landing Page, E-Newsletter and Info graphic correctly.'
```

Figure 22 - The examples of translation using the Google Translate API

The next step is summarization, we will summarize the candidate's experience and the job opening description using the “Gensim” library. After that, we will go through the NLP process. This process contains 2 steps: stop word removal and lemmatization.

In the first step, stop word removal, we will delete the word that does not have significant meaning such as a, an, the, etc. For the second step, lemmatization, we are going to transform the word to its root word by cutting its postfix or changing its form. For example, the word “go”, “went”, “gone”, and “going” will be transformed to “go”.

4.2.2 Feature Engineering

From already existing features, we can create new ones using business knowledge and data understanding. For example, the distance between home and workplace can be calculated. Long-distance may affect a candidate's journey, making it take longer and cost more. This may affect the decision to hire employees.

Another undeniable factor that most Thai organizations consider is the university ranking from where job applicants graduated. So, we will create a new feature by calculating the score from the university that the candidate graduated from by using Quacquarelli Symonds (QS) world university rankings. The candidate will obtain a higher score if the university he or she graduated from is ranked in the university ranking. The higher the university they graduated from was ranked, the higher the candidate's score. On the other hand, if the candidate graduates from a university that has a low ranking, they will receive a lower score. They will get a zero if the university where they studied is not included in the rankings.

4.2.3 Missing Value Handling

There are a lot of techniques for dealing with missing values ranging from simple to complex. Figure 23 summarizes the techniques, including deletion and imputation. We can delete or impute missing data depending on how many missing values and what effect deleting data has on predictions. The first technique is data deletion which can be divided into 3 more ways: pairwise deletion, listwise deletion, and dropping entire columns. Pairwise deletion is the way to delete only missing values. Listwise deletion is a technique for deleting the row containing the missing value, and dropping entire columns means deleting the column containing the missing value. However, pairwise and listwise techniques are not suitable for our data because some columns have missing values of more than 50% so we will use the dropping entire columns technique.

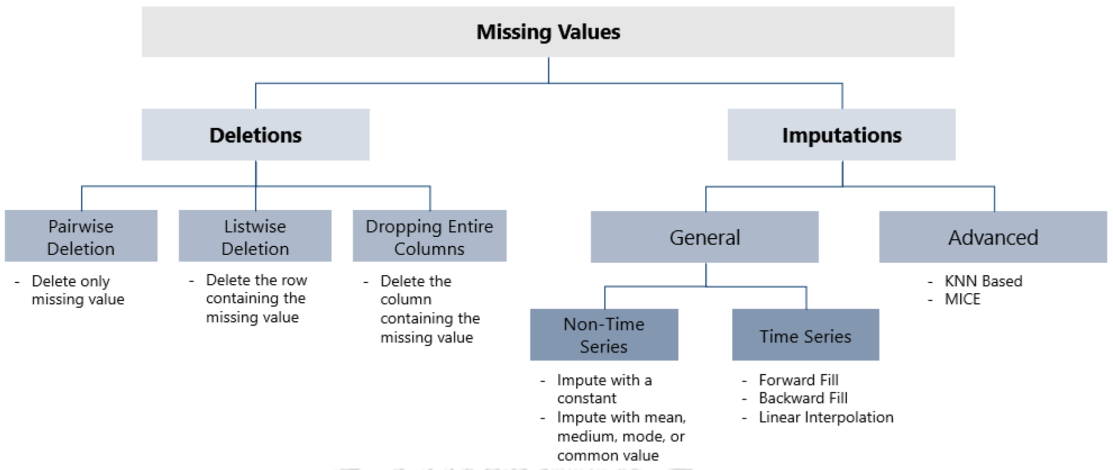


Figure 23 - Missing value handling techniques

The other way to handle missing value is imputations. According to the complexity of the technique, it can be divided into 2 groups: general and advanced. For general techniques, if the data is not a time series, missing data can be filled with any value such as mean, median, mode, etc. If the data is a time series, the appropriate method is forward fill, backward fill, or linear interpolation. For advanced techniques, k-NN Based and Multivariate Imputation by Chained Equations (MICE) are some examples.

We design to use both general and advanced techniques. We will drop the entire column. If the columns are categorical and have a lot of unique values, missing values are imputed with mode value for that column such as nationality feature because 90% of all values are Thailand, so imputation by mode is the appropriate way. A k-NN-based method is used to handle the remaining columns with missing values.

4.2.4 Dealing with Categorical Features

For the nominal feature, a feature that consists of a finite number of discrete values with no relationship between them, we will use the one-hot-encoder technique even though there is some natively supported categorical feature model. The caution of this method is it could lead to a massive amount of dimensionality.

For the ordinal feature, a feature indicating a variable has a finite set of discrete values that can be ranked, we will use the label encoder technique. The example of nominal and ordinal features is shown in Figure 24.

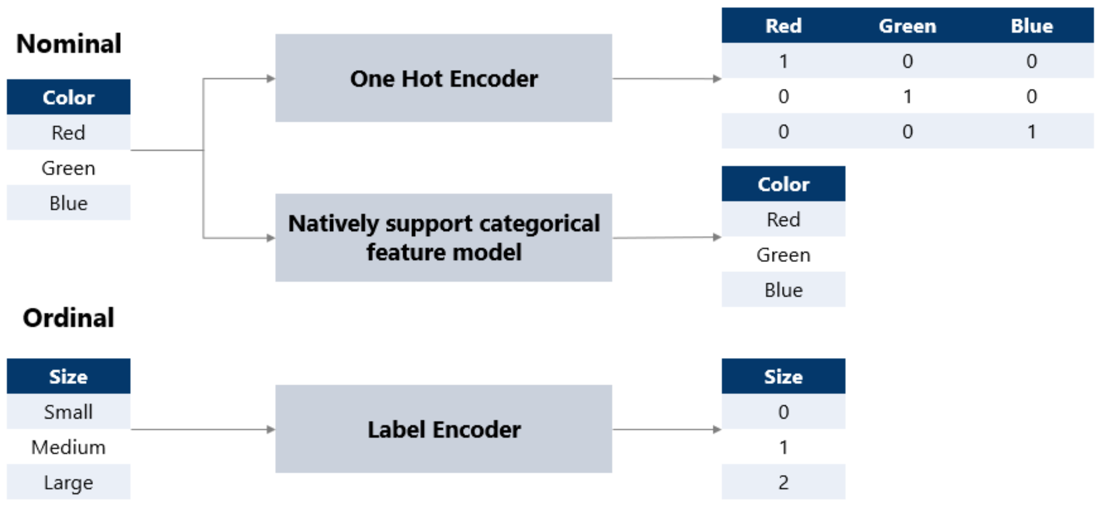


Figure 24 - Example of nominal and ordinal feature

4.2.5 Feature Scaling

We want to rescale the value of all columns. There are 2 ways to do this: normalization and standardization. Normalization is the technique to rescale the value into a range of 0 to 1. This method is useful when all parameters need to have the same positive scale. The equation of normalization is shown in Equation 7.

$$x = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{7}$$

On the other hand, standardization is a technique to rescale the data to have a mean of 0 and a deviation of 1. The distribution produced by setting the feature's mean to zero has a value of standard deviation equal to one. The values are not limited to a specific range. The equation of standardization is shown in Equation 8.

$$x = \frac{x - \mu}{\sigma} \tag{8}$$

To get the best results, we will fit the model to raw, normalized, and standardized data, then compare the results to determine the best ones.

4.2.6 Handling Imbalance Class

An imbalanced classification problem is one where samples are distributed unevenly throughout the identified classes. The distribution can vary from a bias to a major imbalance. This problem causes a challenge for predictive modeling because most machine learning methods for classification assume that each class has an equal number of examples. Subsequently, models can have higher prediction accuracy in the train set, particularly for the majority class. But, lower prediction accuracy in the test set. In a nutshell, we need to deal with this problem by ensuring that all target classes have the same number of rows.

To get the best results, we will fit the model to raw, over-sampling, and under-sampling data, then compare the results to determine the best ones. For over-sampling, we decide to use the Synthetic Minority Oversampling Technique (SMOTE). It is the less time-consuming step of oversampling methods because It generates oversampled datasets that enable supervised classifier training to provide classification outcomes that are statistically comparable to those from other approaches, but in much less runtime [32]. For over-sampling, we select a random-under-sampling technique.

4.3 Data Prediction

In the data prediction part, the data will be divided into training and testing sets. The training data is used to train and evaluate the models using the cross-validation technique whereas the testing data is used to apply the most appropriate model to make predictions and finally evaluate unseen data.

4.3.1 Split Training and Testing Set

We will segment the data into two groups based on their experience data: a group of job applicants who fill their experience and a group who don't. After that, we will divide the data in each group into 90% of the training data and 10% of the testing data as shown in Figure 25 using the same label ratio in both training and testing data. Model learning and model validation will be done using data from the training set using 5-fold cross-validation, while data from the testing set will be utilized to test the model's results with unseen data and display the proposed system's output.

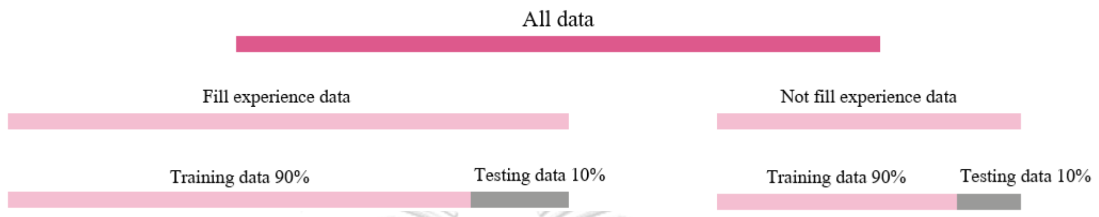


Figure 25 - Split training and testing set

4.3.2 Define Models and Hyperparameter Tuning

We will train a variety of classification models to see which one gives the best performance with this dataset. Moreover, we decide to find the appropriate value of each variable that will be used with models using the GridSearch technique. The following are the models that will be experimented with:

- Decision Tree (DT)
- Suport Vector Machine (SVM)
- Gaussian Naive Bayes (GNB)
- Random Forest (RF)
- k-Nearest Neighbour (k-NN)
- CatBoost
- Extreme Gradient Boosting (XGBoost)
- Convolution Neural Network (CNN)

The next section provides a description of these techniques' specifics as well as the parameter values.

4.3.2.1 Decision Tree (DT)

The decision Tree is the first supervised learning technique that we will experiment with classification to find the most appropriate value for 4 parameters. The detail of each parameter that we will experiment with is shown in Table 2 [33].

Table 2 - Parameters and values that will be tested in the DT model

parameter	description	experiment value
criterion	The function for evaluating a split efficiency	entropy, gini, log_loss
splitter	The method for choosing the split	best, random
max_depth	The tree's maximum depth	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
min_samples_ split	The minimum number of samples required to split an internal node	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

4.3.2.2 Suport Vector Machine (SVM)

A Support Vector Machine (SVM) is a powerful and flexible machine learning model that can perform linear or nonlinear classification, regression, and outlier detection.

There are 3 parameters that we will experimentally adjust values for the Support Vector Machine model which are C, kernel, and gamma as shown in Table 3 [34].

Table 3 - Parameters and values that will be tested in the SVM model

parameter	description	experiment value
C	Regularization parameter	0.05, 0.03, 0.01, 0.5, 1, 3, 5
kernel	The type of kernel to employ in the algorithm	linear, precomputed, poly, rbf, sigmoid
gamma	Kernel coefficient	auto, scale

4.3.2.3 Gaussian Naive Bayes (GNB)

The Naive Bayes variant known as Gaussian Naive Bayes supports continuous data and conforms to the Gaussian normal distribution. We will demonstrate and change the value of just one parameter, var_smoothing, as shown in Table 4 [35].

Table 4 - Parameters and values that will be tested in the GNB model

parameter	description	experiment value
var_smoothing	A fraction of all features' biggest deviations that are contributed to variances for computation stability	10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}

4.3.2.4 Random Forest (RF)

In Random Forest, a collection or ensemble of classification, there are 4 tuning parameters, 3 of these are as same as the decision tree model parameters which are criterion, max_depth, and min_samples_split, with only one difference parameter being n_estimators as shown in Table 5 [36].

Table 5 - Parameters and values that will be tested in the RF model

parameter	description	experiment value
n_estimators	How many trees there are in the forest	50, 60, 70, 80, 90, 100
criterion	The function for evaluating a split efficiency	entropy, gini, log_loss
max_depth	The maximum depth of the tree	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
min_samples_split	The minimum number of samples required to split an internal node	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

4.3.2.5 k-Nearest Neighbour (k-NN)

The k-NN model is one of many models to be turned for the proper parameter value by changing the value of n_neighbors, weights, algorithm, and p to optimize the model's performance as shown in Table 6 [37].

Table 6 - Parameters and values that will be tested in the k-NN model

parameter	description	experiment value
n_neighbors	How many neighbors to utilize	1, 3, 5, 7, 9, 11, 13, 15
weights	What weight function is needed	distance, uniform
algorithm	The algorithm for calculating the nearest neighbors	auto, ball_tree, brute, kd_tree
p	Power parameter for the Minkowski metric	1, 2, 3

4.3.2.6 CatBoost

Another machine learning method that is successful at forecasting categorical features is the CatBoost classifier, 4 parameters which are iterations, learning rate, depth, and l2-leaf-reg will have their values experimentally changed as shown in Table 7 [38].

Table 7 - Parameters and values that will be tested in the CatBoost model

parameter	description	experiment value
iterations	The maximum number of trees that can be constructed	100, 200, 300, 400, 500, 1000, 1500, 2000
Learning_rate	The method for choosing the split at each node	0.001, 0.01, 0.02, 0.03, 0.04, 0.5, 0.1
depth	The trees' highest point in depth	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
l2_leaf_reg	Cost function coefficient at the L2 regularization term	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

4.3.2.7 Extreme Gradient Boosting (XGBoost)

To determine the optimal value of parameters for the XGBoost model, one of the gradient-boosting boosted tree methods that uses parallel processing, tree-pruning, handling missing values, and regularization to prevent bias and overfitting, we will experimentally adjust 5 parameters which are booster, eta, max_depth, lambda, and alpha as shown in Table 8 [39].

Table 8 - Parameters and values that will be tested in the XGBoost model

parameter	description	experiment value
booster	The booster to use in prediction	dart, gblinear, gbtree
eta	Minimum loss reduction is needed to create a new division on a tree leaf node	0, 0.1, 0.2, 0.3, 0.4, 0.5, 1
max_depth	The tree's maximum height	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
lambda	L2 regularization term on weights	0, 0.05, 0.03, 0.01, 0.5, 1, 3, 5
alpha	L1 regularization term on weights	0, 0.05, 0.03, 0.01, 0.5, 1, 3, 5

4.3.2.8 Convolution Neural Network (CNN)

The structure of a CNN in this research composes of the convolution layer that has 32 filters with a 3x3 kernel size. Max-pooling of size 2x2 is then used to reduce the dimensions of the output and flatten the output to 1 dimension. Next to a flattened layer is 2 dense layers, 100 units with the relu activation function and 10 units with the relu activation function, alternate with a dropout layer. The output layer will contain 2 nodes of classification with a sigmoid activation function and categorical cross-entropy will be used as the loss function [22].

The optimizer and dropout parameter will be tuned for the neural network model while batch_size and epoch value will be adjusted in the model training process as shown in Table 9.

Table 9 - Parameters and values that will be tested in the CNN model

parameter	description	experiment value
batch_size	The number of training examples utilized in one iteration	16, 32, 64
dropout	The ratio that nullifies the contribution of some neurons towards the next layer	0.1, 0.2, 0.3, 0.4, 0.5
epoch	The number of times the algorithm has iterated through the training dataset	100, 200, 300, 400, 500, 1000
optimizer	The methods used to minimize an error function	Adam, SGD

4.3.3 Evaluate Metrics for Classification Model

We compare the performance of each model with 4 measures to get the most suitable model in terms of accuracy and productivity.

4.3.3.1 Accuracy

The overall performance of this proposed method will be measured by accuracy. We are going to calculate the average accuracy from all iterations of cross-validation for the training set. The equation of accuracy is shown in Equation 9.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{9}$$

4.3.3.2 Weighted F1-score

We want a model that balances precision and recall because we want to minimize total errors, which means low bias and low variance. Low variance algorithms (high bias) will be less complex and consistent, but inaccurate on average which can lead to underfitting. On the other hand, low bias algorithms (high variance) will be more complex and accurate on average, but inconsistent which can lead to overfitting. The equation of weighted F1-score is shown in Equation 10 when W represents the weight of each class, and the equation of F1-score is shown in Equation 6.

$$F1_{class1}W_1 + F1_{class2}W_2 + \cdots + F1_{classN}W_N \tag{10}$$

4.3.3.3 Recall

We want the FN to be as low as possible for this classification problem. This means that we do not want any potential candidates predicted as not-suitable, leading to the shortlisted candidate being dismissed from the interview call. That is because we do not want to lose someone who could be effective for a job interview and join the company. On the other hand, we want the TP to be as high as possible as well. This means that we want the model to be able to predict the candidates to be shortlisted as accurately as possible for the effectiveness of the system results. Consequently, to be able to evaluate both FN and TP simultaneously, recall value will be used to evaluate system performance. The higher the recall value, the better. The equation of recall is shown in Equation 5.

4.3.3.4 Computation Time

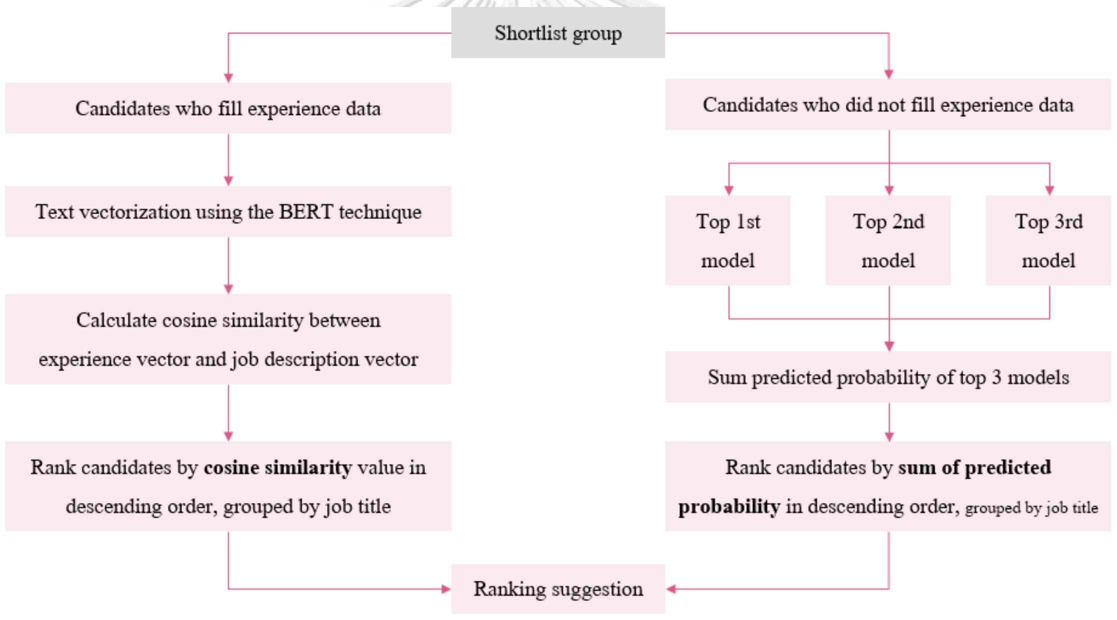
Another factor to consider is the computation time. Since time is valuable in practice, the proposed method should reduce the amount of time spent on human work. The computer used in this research is a 64-bit operating system with an Intel(R) Core (TM) i7-10510U CPU @ 1.80GHz 2.30GHz and 16.0 gigabytes (GB) of random-access memory (RAM).

These 4 measures will give the proper model with high reliability and require less computation time. After we get the most suitable model and parameters. We will retrain the model with the training set through the 5-fold cross-validation technique to find the optimal threshold for each iteration, then take those thresholds to calculate the median to determine the final optimal threshold for the testing set.

4.3.4 Similarity Computation and Ranking Suggestion

After the model can predict who should be in the shortlist group, we will rank the candidates in this group. There are two ways to rank depending on whether the candidate fills in his or her experience information as shown in Figure 25.

Applicants who fill out their experience data will have their experience data converted to vectors of numbers, as well as job description data. The two vectors are then calculated for the cosine similarity value. Finally, the computed value by the sum of cosine similarity and predict probability will be used to display the candidates ranking in descending order that grouping them according to the job title that they apply for.



Shortlist group
Candidates who fill experience data
Text vectorization using the BERT technique
Calculate cosine similarity between experience vector and job description vector
Rank candidates by **cosine similarity** value in descending order, grouped by job title
Candidates who did not fill experience data
Top 1st model
Top 2nd model
Top 3rd model
Sum predicted probability of top 3 models
Rank candidates by **sum of predicted probability** in descending order, grouped by job title
Ranking suggestion

Figure 26 - The candidates ranking approach

On the other hand, candidates who do not fill out their experience are ranked by combining the probability of predicting as a shortlist group from the 3 best models and sorted by the sum in descending order that grouping them according to the job title that they apply for. The final ranking suggestion will order candidates who fill in experience data first, then those who do not.

CHAPTER 5

RESULT AND DISCUSSION

This research compares the classification of data with 8 methods to select the most appropriate one. The following part will provide an explanation of the procedure's details.

5.1 Experimental Dataset Preparation

After data was collected from JobsDB.com, the data was put through many data preparation processes, such as converting Thai to English text, creating new columns from existing columns, missing value handling, categorical features handling, feature scaling, and handling imbalance class. The data after being transformed through the preparation step will be put into the models. The feature details are shown in Table 10.

Table 10 – The feature details

Feature name	Description
age	Candidate's age
can_speak_en	Can the candidate communicate in English?
cosine_sim	The similarity between the candidate's experience data and the job description of the position applied for
count_certificates	The number of certificates that the candidate has
count_extract_skills	The number of candidate’s skills that match the skills of the position applied for
count_lang	The number of languages the candidate can communicate
count_same_keyword	The number of important keywords extracted from candidates' work experience data that match the

	important keywords extracted from the job opening data
fill_expr_in_en	Does the candidate fill in the information in JobsDB in English?
filter_age	Does the candidate's age meet the qualification of the job the candidate applied for?
filter_minimum_degree	Does the candidate's minimum education level meet the qualification of the job the candidate applied for?
filter_minimum_exp	Does the candidate's year of experience meet the qualification of the job the candidate applied for?
filter_national_only	Does the candidate's nationality meet the qualification of the job the candidate applied for?
gender_Female	Is the applicant a female?
gender_Male	Is the applicant a male?
haversine_distance	The distance between the candidate's residence and the workplace
highest_edu	The highest education level of the candidate
ins_in_topu_score	The total score of the university where the candidate has graduated
nice_to_have_major_scores	The total score of the major where the candidate has graduated
similar_job_title	The similarity between the candidate's recent job title and the position applied for
thai_nationality	Is the applicant a Thai national?
year_of_experiences	Total years of work experience
target	The classification target (1 = shortlist, 0 = not suitable)

The difference between the data that we took into the model for those who filled in experience data and those who did not is the cosine_sim column. The candidates who filled in the experience data will have this column in the model, while those who did not fill in the experience data will not have it. The 2027 candidates consist of 1218 who fill in experience data and 809 who did not. Figures 27 and 28 show the percentage of candidates in different groups for candidates who filled in experience and did not respectively.

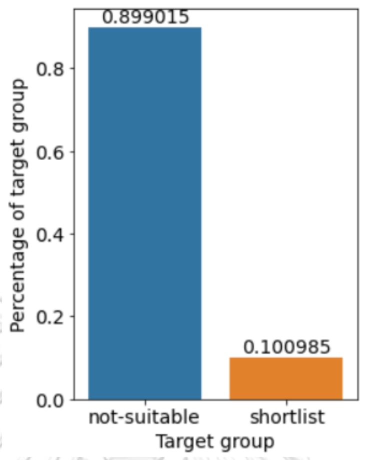


Figure 27 - The percentage of the target group for candidates who filled in the experience

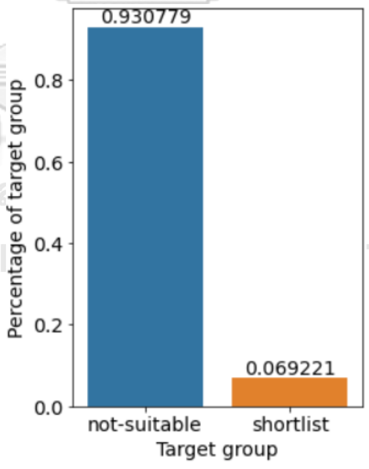


Figure 28 - The percentage of the target group for candidates who did not fill in the experience

5.2 Modeling

We did hyperparameter tuning with 5 StratifiedKFold using the GridsearchCV technique by separating the data between those who filled in the experience and those who did not. Moreover, we experimented with normalization and imbalance class handling. For normalization, we experimented in 3 ways, which are without normalization, min-max scaler, and standard scaler. For imbalance class handling, we experimented in 3 ways, which are without imbalance class handling, oversampling, and undersampling.

Table 11 shows the most proper parameter value for the Decision Tree model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 11 - The most proper parameter value for the DT model

Experience data	Normalization	Imbalance class handling	Parameters			
			criterion	splitter	max_ depth	min_samples_ split
Fill	-	-	gini	best	10	2
		oversampling	entropy	random	7	9
		undersampling	entropy	random	4	6
	min-max scaler	-	entropy	best	10	3
		oversampling	entropy	best	5	4
		undersampling	gini	random	2	6
	standard scaler	-	entropy	best	10	2
		oversampling	gini	random	9	6
		undersampling	gini	random	2	9
Not fill	-	-	gini	best	10	2
		oversampling	entropy	best	10	7
		undersampling	entropy	random	9	1
	min-max scaler	-	entropy	best	10	2
		oversampling	gini	best	10	10
		undersampling	entropy	random	2	5
	standard scaler	-	gini	best	10	2
		oversampling	gini	best	10	10
		undersampling	gini	random	8	8

Table 12 shows the most proper parameter value for the Support Vector Machine model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 12 - The most proper parameter value for the SVM model

Experience data	Normalization	Imbalance class handling	Parameters		
			C	gamma	kernel
Fill	-	-	0.01	scale	linear
		oversampling	1	scale	rbf
		undersampling	0.01	scale	linear
	min-max scaler	-	3	scale	rbf
		oversampling	3	scale	rbf
		undersampling	1	auto	linear
	standard scaler	-	0.5	auto	poly
		oversampling	5	scale	rbf
		undersampling	0.5	scale	rbf
Not fill	-	-	0.01	scale	linear
		oversampling	3	scale	poly
		undersampling	0.03	auto	linear
	min-max scaler	-	0.5	auto	poly
		oversampling	5	scale	poly
		undersampling	1	scale	rbf
	standard scaler	-	3	scale	poly
		oversampling	5	scale	rbf
		undersampling	0.05	scale	rbf

Table 13 shows the most proper parameter value for the Gaussian Naïve Bay model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 13 - The most proper parameter value for the GNB model

Experience data	Normalization	Imbalance class handling	Parameters
			var_smoothing
Fill	-	-	1e-06
		oversampling	1e-10
		undersampling	1e-09
	min-max scaler	-	1e-05
		oversampling	1e-05
		undersampling	1e-05
	standard scaler	-	1e-05
		oversampling	1e-05
		undersampling	1e-05
Not fill	-	-	1e-06
		oversampling	1e-10
		undersampling	1e-09
	min-max scaler	-	1e-05
		oversampling	1e-05
		undersampling	1e-06
	standard scaler	-	1e-05
		oversampling	1e-05
		undersampling	1e-05

Table 14 shows the most proper parameter value for the Random Forest model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 14 - The most proper parameter value for the RF model

Experience data	Normalization	Imbalance class handling	Parameters			
			criterion	max_ depth	min_samples _split	n_estimators
Fill	-	-	entropy	3	1	100
		oversampling	gini	6	1	100
		undersampling	gini	1	1	80
	min-max scaler	-	entropy	2	1	100
		oversampling	entropy	5	1	80
		undersampling	gini	8	1	60
	standard scaler	-	entropy	4	1	100
		oversampling	entropy	1	1	50
		undersampling	entropy	8	1	80
Not fill	-	-	entropy	1	1	60
		oversampling	entropy	5	1	90
		undersampling	entropy	9	1	80
	min-max scaler	-	entropy	2	1	90
		oversampling	entropy	2	1	90
		undersampling	entropy	2	1	60
	standard scaler	-	entropy	5	1	60
		oversampling	entropy	2	1	80
		undersampling	entropy	2	1	70

Table 15 shows the most proper parameter value for the k-Nearest Neighbors model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 15 - The most proper parameter value for the k-NN model

Experience data	Normalization	Imbalance class handling	Parameters			
			algorithm	n_neighbors	p	weights
Fill	-	-	auto	15	1	uniform
		oversampling	auto	1	1	distance
		undersampling	auto	1	1	distance
	min-max scaler	-	auto	13	1	distance
		oversampling	auto	1	1	distance
		undersampling	auto	11	3	distance
	standard scaler	-	auto	1	2	distance
		oversampling	auto	1	1	distance
		undersampling	auto	13	3	distance
Not fill	-	-	auto	5	2	uniform
		oversampling	auto	1	1	distance
		undersampling	auto	1	1	distance
	min-max scaler	-	auto	9	1	uniform
		oversampling	auto	1	1	distance
		undersampling	auto	3	1	uniform
	standard scaler	-	entropy	5	1	60
		oversampling	entropy	2	1	80
		undersampling	entropy	2	1	70

Table 16 shows the most proper parameter value for the CatBoost model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 16 - The most proper parameter value for the CatBoost model

Experience data	Normalization	Imbalance class handling	Parameters			
			iterations	learning_rate	depth	l2_leaf_reg
Fill	-	-	100	0.1	3	6
		oversampling	2000	0.1	4	2
		undersampling	400	0.03	1	10
	min-max scaler	-	100	0.1	3	6
		oversampling	2000	0.03	4	5
		undersampling	400	0.03	1	10
	standard scaler	-	100	0.1	3	6
		oversampling	1500	0.1	3	3
		undersampling	400	0.03	1	10
Not fill	-	-	1500	0.1	1	3
		oversampling	2000	0.05	7	10
		undersampling	500	0.01	2	8
	min-max scaler	-	2000	0.03	7	3
		oversampling	1500	0.1	5	1
		undersampling	2000	0.1	3	8
	standard scaler	-	1500	0.1	1	3
		oversampling	2000	0.04	5	1
		undersampling	2000	0.1	3	8

Table 17 shows the most proper parameter value for the XGBoost model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling.

Table 17 - The most proper parameter value for the XGB model

Experience data	Normalization	Imbalance class handling	Parameters				
			booster	eta	max_depth	lambda	alpha
Fill	-	-	dart	0.1	6	0	5
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0
	min-max scaler	-	dart	0.1	6	0	5
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0
	standard scaler	-	dart	0.1	6	0	5
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0
Not fill	-	-	dart	1	1	0.05	0.05
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0
	min-max scaler	-	dart	1	1	0.05	0.05
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0
	standard scaler	-	dart	1	1	0.05	0.05
		oversampling	dart	0	1	0	0
		undersampling	dart	0	1	0	0

Table 18 shows the most proper parameter value for the Convolutional Neural Network model using the GridsearchCV technique for the data with experimental normalization and imbalance class handling. The learning rate we use is initialed from 0.001 then multiple that learning rate with 0.1 when running over 80% epoch.

Table 18 - The most proper parameter value for the CNN model

Experience data	Normalization	Imbalance class handling	Parameters			
			batch_size	dropout	epoch	optimizer
Fill	-	-	16	0.4	1000	Adam
		oversampling	16	0.5	1000	SGD
		undersampling	32	0.3	1000	Adam
	min-max scaler	-	32	0.4	1000	Adam
		oversampling	32	0.2	1000	Adam
		undersampling	32	0.1	300	Adam
	standard scaler	-	64	0.2	1000	Adam
		oversampling	16	0.4	1000	SGD
		undersampling	64	0.3	1000	SGD
Not fill	-	-	16	0.5	100	Adam
		oversampling	16	0.5	100	Adam
		undersampling	16	0.1	100	Adam
	min-max scaler	-	16	0.5	100	Adam
		oversampling	16	0.2	100	Adam
		undersampling	16	0.1	100	Adam
	standard scaler	-	16	0.5	100	Adam
		oversampling	16	0.1	100	Adam
		undersampling	16	0.5	100	Adam

5.3 Model Performance Comparison

After we knew the best parameter value of each model, we trained the models with the 5-StratifiedKFold cross-validation technique again to calculate average accuracy, average weighted f1-score, average recall, and average computation time. Where the higher the values of average accuracy, average weighted f1-score, and average recall, the better. However, the lower the value of the average computation time, the better. The value is highlighted in bold, denoting the best value, whereas underlined indicates the second-best value.

For the Decision Tree model, the data that the candidates filled the experience without normalization and imbalance class handling gained the best average accuracy but obtained low recall also. The data with normalization using a min-max scaler but without imbalance class handling gets the 2nd highest average accuracy. While oversampling gets the 2nd highest average weighted f1-score. But these 2 ways acquired a low recall which is less than 0.5. The data with imbalance class handling using undersampling get a high recall. By not normalizing, the highest average accuracy and recall are obtained. So, the data without normalization but undersampling gave the best performance for the Decision Tree model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.01076 milliseconds per person.

On the other hand, the data that the candidates did not fill the experience with min-max scaler normalization and imbalance class handling using oversampling technique gained the best average accuracy and average weighted f1-score. But it also obtained a less average recall. While the data without normalization but with imbalance class handling using undersampling acquired the highest recall. But its average accuracy is less than 0.5. The data with normalization using a min-max scaler and imbalance class handling using undersampling is the 3rd highest average recall, while average accuracy and average weighted f1-score were higher than 0.6. So, the data with min-max scaler and undersampling gave the best performance for the Decision Tree model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.09072 milliseconds per person as shown in Table 19.

For the Support Vector Machine model, the data that the candidates filled the experience with normalization using a standard scaler and oversampling imbalance class handling gained the best average accuracy and average weight f1-score. But its average recall is less than 0.5. While the data with min-max scaler and oversampling obtained the second-best average accuracy and average weight f1-score. But its average recall is also not too high. The data with min-max scaler normalization but without imbalance class handling acquired average accuracy and weighted f1-score slightly lower, but the average recall is higher, making all 3 values close. So, the data with min-max scaler normalization but without imbalance class handling gave the best performance for the Support Vector Machine model. Although it is not the least time-consuming, the time it took is considered less. It took an average of 0.10262 milliseconds per person.

Table 19 - The DT evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	**0.86314**	**0.87060**	0.47787	0.05253
		oversampling	0.83577	0.85291	0.52332	0.12475
		undersampling	0.77829	0.81760	0.76601	0.01180
	min-max scaler	-	0.86040	0.86509	0.39684	0.13660
		oversampling	0.85858	0.86760	0.46917	0.07019
		undersampling	0.77280	0.81201	0.74822	**0.01038**
	standard scaler	-	0.85675	0.86170	0.38775	0.04764
		oversampling	0.82299	0.84431	0.55059	0.01699
		undersampling	0.76275	0.80549	**0.77549**	0.03088
Not fill	-	-	0.85711	0.87058	0.24000	**0.02474**
		oversampling	0.85299	0.87194	0.34000	0.07340
		undersampling	0.46965	0.53360	**0.66000**	0.07129
	min-max scaler	-	0.85444	0.87037	0.28000	0.05377
		oversampling	**0.85989**	**0.87294**	0.24000	0.05207
		undersampling	0.69991	0.75352	0.60000	0.06605
	standard scaler	-	0.85436	0.86692	0.20000	0.09499
		oversampling	0.84342	0.86336	0.26000	0.08645
		undersampling	0.59347	0.69074	0.56000	0.06579

On the other hand, the data that the candidates did not fill the experience with min-max scaler normalization but without imbalance class handling gained the best average accuracy and average weighted f1-score. But it also obtained the least average recall. While the data without normalization but with imbalance class handling using undersampling acquired the highest recall. But its average accuracy is between 0.5 and 0.6. The data with normalization using a standard scaler and imbalance class handling using undersampling is the 4th highest average recall, while average accuracy and average weighted f1-score were higher than 0.72. So, the data with standard scaler and undersampling gave the best performance for the Support Vector Machine model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.05655 milliseconds per person as shown in Table 20.

Table 20 - The SVM evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	0.73995	0.7889	0.77549	0.65038
		oversampling	0.75201	0.75846	0.70853	days
		undersampling	0.71167	0.76761	**0.81107**	0.37538
	min-max scaler	-	0.80474	0.83563	0.71186	0.12500
		oversampling	0.82846	0.85086	0.62134	0.25400
		undersampling	0.78010	0.81884	0.76640	0.01793
	standard scaler	-	0.82484	0.84659	0.65810	0.06298
		oversampling	**0.84307**	**0.85728**	0.49526	0.13091
		undersampling	0.79380	0.82777	0.70316	**0.01667**
Not fill	-	-	0.49991	0.60857	0.62000	0.28426
		oversampling	0.53657	0.57680	0.58000	days
		undersampling	0.51923	0.62613	**0.64000**	**0.03435**
	min-max scaler	-	**0.93132**	**0.89820**	0.00000	0.19317
		oversampling	0.74596	0.80323	0.50000	0.20869
		undersampling	0.62768	0.71876	0.58000	0.05233
	standard scaler	-	0.77332	0.81958	0.36000	0.08343
		oversampling	0.78574	0.82658	0.28000	0.18506
		undersampling	0.72378	0.78954	0.52000	0.04575

For the Gaussian Naïve Bay model, the data that the candidates filled the experience without normalization and imbalance class handling gained the best average accuracy and average weighted f1-score, but it also obtained the worst average recall. While the data without normalization but with imbalance class handling using oversampling acquired the second-best average accuracy and average weighted f1-score. Its recall was not the highest but not too less. So, the data without normalization but oversampling gave the best performance for the Gaussian Naïve Bay model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.02228 milliseconds per person.

On the other hand, the data that the candidates did not fill the experience without normalization but imbalance class handling using oversampling gained the best average accuracy and average weighted f1-score. While its average recall was the worst. The data with a min-max scaler but without imbalance class handling obtained the best average recall. Its average accuracy and weighted f1-score were not too bad. While the data with standard scaler normalization and imbalance class handling using oversampling acquired better average accuracy and average weighted f1-score. Its recall is almost the same. So, the data with standard scaler normalization and oversampling imbalance class handling gave the best performance for the Gaussian Naïve Bay model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.07557 milliseconds per person as shown in Table 21.

For the Random Forest model, the data that the candidates filled the experience with min-max scaler normalization but without imbalance class handling gained the best average recall. Which is equal to the data with standard scaler and handling imbalance class handling using the undersampling technique. But they obtained low average accuracy and weighted f1-score. The 2nd highest average recall gave the highest average accuracy and weighted f1-score for the data with min-max scaler and imbalance class handling using the oversampling technique. So, the data with min-max scaler normalization and oversampling gave the best performance for the Random Forest model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.01491 milliseconds per person.

Table 21 - The GNB evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	**0.89873**	**0.85079**	0.00000	**0.01120**
		oversampling	0.77645	0.81347	0.63913	0.02442
		undersampling	0.69255	0.75270	0.75692	0.02060
	min-max scaler	-	0.34764	0.41461	0.92846	0.02028
		oversampling	0.41517	0.49623	0.84743	0.02428
		undersampling	0.35672	0.42076	0.89249	0.01737
	standard scaler	-	0.2345	0.25533	**0.94625**	0.01248
		oversampling	0.35493	0.42422	0.89209	0.01798
		undersampling	0.30017	0.34681	0.91028	0.01147
Not fill	-	-	**0.93132**	**0.89820**	0.00000	**0.02638**
		oversampling	0.59347	0.69168	0.58000	0.04907
		undersampling	0.49991	0.61004	0.70000	0.05591
	min-max scaler	-	0.51368	0.62295	**0.72000**	0.06667
		oversampling	0.52192	0.63038	0.70000	0.07816
		undersampling	0.53933	0.61958	0.58000	0.03805
	standard scaler	-	0.52037	0.62406	**0.72000**	0.05022
		oversampling	0.53275	0.63621	0.70000	0.05502
		undersampling	0.53522	0.61683	0.60000	0.07189

On the other hand, the data that the candidates did not fill the experience with min-max scaler normalization but without imbalance class handling gained the best average accuracy and average weighted f1-score, it also gained the worst average recall. While the data with standard scaler normalization and imbalance class handling using the undersampling technique obtained the best average recall, it also obtained less average accuracy and average weighted f1-score. The data without normalization but imbalance class handling using the undersampling technique acquired the 2nd highest average recall. Its average accuracy and average weighted f1-score were not bad. So, the data without normalization but with imbalance class handling using the undersampling technique gave the best performance for the Random Forest model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.44833 milliseconds per person as shown in Table 22.

Table 22 - The RF evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	0.73891	0.68406	0.20000	0.23730
		oversampling	0.57909	0.51734	0.40000	0.16046
		undersampling	0.74054	0.68554	0.20000	0.08486
	min-max scaler	-	0.42091	0.35208	**0.60000**	0.25862
		oversampling	0.58073	0.51881	0.40000	0.16350
		undersampling	0.57909	0.51734	0.40000	0.22978
	standard scaler	-	**0.89873**	**0.85079**	0.20000	0.34378
		oversampling	0.57909	0.51734	0.40000	**0.13513**
		undersampling	0.41927	0.41927	**0.60000**	0.32541
Not fill	-	-	**0.75890**	**0.72042**	0.20000	**0.31989**
		oversampling	0.58611	0.58611	0.40000	0.55462
		undersampling	0.41370	0.36454	0.60000	0.32639
	min-max scaler	-	**0.75890**	**0.72042**	0.20000	0.64001
		oversampling	0.58630	0.54248	0.40000	0.55260
		undersampling	0.41370	0.36454	0.60000	0.36580
	standard scaler	-	0.41370	0.36454	0.60000	0.34394
		oversampling	0.58630	0.54248	0.40000	0.53638
		undersampling	0.24128	0.18677	**0.80000**	0.41088

For the k-Nearest Neighbors model, the data that the candidates filled the experience with standard scaler normalization and undersampling gained the best average recall. While its average accuracy and weighted f1-score were high even though there was not the highest. So, the data with standard scaler normalization and undersampling gave the best performance for the k-Nearest Neighbors model.

Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.02496 milliseconds per person.

On the other hand, the data that the candidates did not fill the experience without normalization and imbalance class handling gained the best average accuracy and average weighted f1-score, it also obtained a less average recall. While the data with standard normalization and imbalance class handling using the undersampling technique acquired the best average recall. Its average accuracy and average weighted f1-score were not bad. So, the data with standard scaler normalization and imbalance class handling using the undersampling technique gave the best performance for the k-Nearest Neighbors model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.07359 milliseconds per person as shown in Table 23.

For the CatBoost model, the data that the candidates filled the experience without normalization but imbalance class handling using the oversampling technique gained the best average recall. Its average accuracy and average weighted f1-score were in the range of 0.79 to 0.82. So, the data without normalization but with imbalance class handling using the oversampling technique gave the best performance for the CatBoost. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.05275 milliseconds per person.

On the other hand, the data that the candidates did not fill the experience without normalization and imbalance class handling gained the best average accuracy and average weighted f1-score, it also obtained a less average recall. Which are the same result as the data with standard scaler but without imbalance class handling us. While The data without normalization but with imbalance class handling using the undersampling technique acquired the best average recall. Its average accuracy and average weighted f1-score were not bad. So, without normalization but imbalance class handling using the undersampling technique gave the best performance for the CatBoost model. Although it is not the least time-consuming, the time it took is considered less. It took an average of 0.09793 milliseconds per person as shown in Table 24.

Table 23 - The k-NN evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	**0.90200**	0.85562	0.00000	**0.01484**
		oversampling	0.84123	0.85047	0.36957	0.16422
		undersampling	0.60125	0.67848	0.66759	0.09547
	min-max scaler	-	0.90147	**0.88106**	0.20711	0.09533
		oversampling	0.86586	0.86809	0.37787	0.14983
		undersampling	0.41370	0.36454	0.60000	0.32639
	standard scaler	-	0.86769	0.86666	0.33281	0.02655
		oversampling	0.86315	0.86380	0.34190	0.03524
		undersampling	0.74269	0.79041	**0.72134**	0.02736
Not fill	-	-	**0.93133**	**0.90066**	0.02000	**0.02599**
		oversampling	0.84065	0.85924	0.02000	0.05389
		undersampling	0.52751	0.63440	**0.58000**	0.06408
	min-max scaler	-	0.93132	0.89820	0.00000	0.06847
		oversampling	0.87778	0.87779	0.12000	0.05166
		undersampling	0.57267	0.67551	0.54000	0.07512
	standard scaler	-	0.89557	0.88855	0.10000	0.06358
		oversampling	0.87636	0.87877	0.14000	0.05168
		undersampling	0.59889	0.69656	**0.58000**	0.05358

For the Extreme Gradient Boosting model, the data that the candidates filled the experience without imbalance class handling gained the same best average recall no matter what technique of normalization. The score of average accuracy, average weighted f1-score, and average recall are the same for every normalization technique without imbalance class handling. The only difference is the average computation time, which the min-max scaler obtained the least. So, the data with min-max scaler normalization but without imbalance class handling gave the best performance for the Extreme Gradient Boosting model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.33688 milliseconds per person.

Table 24 - The CatBoost evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	**0.91333**	0.89081	0.21581	**0.04466**
		oversampling	0.90148	**0.89928**	0.46917	1.36063
		undersampling	0.78013	0.81631	**0.77470**	0.05782
	min-max scaler	-	**0.91333**	0.89081	0.21581	0.04681
		oversampling	0.89873	0.89397	0.40593	1.67562
		undersampling	0.79013	0.82631	**0.77470**	0.06921
	standard scaler	-	**0.91333**	0.89081	0.21581	0.04966
		oversampling	0.89873	0.89531	0.43241	0.97947
		undersampling	0.79013	0.82631	**0.77470**	0.06757
Not fill	-	-	**0.93404**	**0.91710**	0.18000	0.33639
		oversampling	0.91620	0.91031	0.26000	2.33214
		undersampling	0.67718	0.74643	**0.74000**	**0.07130**
	min-max scaler	-	0.91893	0.90632	0.16000	1.49103
		oversampling	0.88874	0.89459	0.32000	1.06017
		undersampling	0.54948	0.65344	**0.74000**	0.34659
	standard scaler	-	**0.93404**	**0.91710**	0.18000	0.33493
		oversampling	0.88186	0.88970	0.32000	1.54922
		undersampling	0.55085	0.65460	**0.74000**	0.35240

On the other hand, the data that the candidates who did not fill the experience without imbalance class handling gained the same best average recall no matter what technique of normalization. The score of average accuracy, average weighted f1-score, and average recall are the same for every normalization technique without imbalance class handling. The only difference is the average computation time, which those without normalization acquired the least. So, the data without normalization and imbalance class handling gave the best performance for the Extreme Gradient Boosting model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 0.33635 milliseconds per person as shown in Table 25.

Table 25 - The XGB evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	0.86499	0.87355	**0.55798**	0.60949
		oversampling	**0.89873**	0.85079	0.00000	0.44505
		undersampling	**0.89873**	0.85079	0.00000	0.24371
	min-max scaler	-	0.86499	0.87355	**0.55798**	0.41033
		oversampling	**0.89873**	0.85079	0.00000	0.65710
		undersampling	**0.89873**	0.85079	0.00000	**0.20964**
	standard scaler	-	0.86499	0.87355	**0.55798**	0.45088
		oversampling	**0.89873**	0.85079	0.00000	0.44562
		undersampling	**0.89873**	**0.89873**	0.00000	0.63732
Not fill	-	-	**0.93405**	**0.91693**	**0.18000**	0.27211
		oversampling	0.93132	0.89820	0.00000	0.29285
		undersampling	0.93132	0.89820	0.00000	**0.23296**
	min-max scaler	-	**0.93405**	**0.91693**	**0.18000**	0.36774
		oversampling	0.93132	0.89820	0.00000	0.41885
		undersampling	0.93132	0.89820	0.00000	0.34206
	standard scaler	-	**0.93405**	**0.91693**	**0.18000**	0.37007
		oversampling	0.93132	0.89820	0.00000	0.92105
		undersampling	0.93132	0.89820	0.00000	0.31987

For the Convolutional Neural Network model, the data that the candidates filled the experience without normalization and imbalance class handling gained the best average accuracy, average weighted f1-score, and average recall. So, the data without normalization and imbalance class handling using the oversampling technique gave the best performance for the Extreme Gradient Boosting model. Although it was not the least time-consuming, the time it took is considered less. It took an average of 121.83199 milliseconds per person.

On the other hand, the data with all experiments gained the same average accuracy, average weighted f1-score, and average recall. The difference is computation time. The data with standard scaler normalization but without imbalance class handling took the least computation time. So, the data with standard scaler normalization but without imbalance class handling gave the best performance for the Convolutional Neural Network model as shown in Table 26.

Table 26 - The CNN evaluation of the training set

Experience data	Normalization	Imbalance class handling	Evaluation (average) of the training set			
			Accuracy	Weighted f1-score	Recall	Computation time (sec)
Fill	-	-	**0.90056**	**0.86121**	**0.05415**	148.5132
		oversampling	0.89237	0.85246	**0.05415**	142.9685
		undersampling	0.89599	0.85888	**0.05415**	86.70492
	min-max scaler	-	0.89508	0.85235	0.01818	90.75551
		oversampling	0.89416	0.85797	**0.05415**	103.4803
		undersampling	0.89963	0.85294	0.00870	**25.77450**
	standard scaler	-	0.89873	0.85079	0.01779	63.46495
		oversampling	0.89416	0.85503	0.03597	156.5284
		undersampling	0.89873	0.85405	0.01818	57.02587
Not fill	-	-	0.93132	0.89820	0.00000	11.55358
		oversampling	0.93132	0.89820	0.00000	9.28251
		undersampling	0.93132	0.89820	0.00000	11.12407
	min-max scaler	-	0.93132	0.89820	0.00000	11.32896
		oversampling	0.93132	0.89820	0.00000	10.81690
		undersampling	0.93132	0.89820	0.00000	11.32815
	standard scaler	-	0.93132	0.89820	0.00000	**8.42467**
		oversampling	0.93132	0.89820	0.00000	10.97375
		undersampling	0.93132	0.89820	0.00000	8.69763

There are several reasons why CNN takes much longer time to train than other models. The appropriate epoch value was determined via hyperparameter turning to 1000 for the data that the candidate filled in experience and 100 for the candidate who did not, which required CNN time to finish the epoch training. Another reason is

CNN is more complex in terms of Big O notation because they share their parameters within the network. This implies that even if they have fewer parameters, they are still used frequently.

5.4 Summary of Model Comparison

Table 27 shows the best data preparation way for each model for the data that candidates filled in experience. Each model is suitable for different data preprocessing ways. According to average accuracy, weighted f1-score, recall, and computation time, the most suitable model for our data is the SVM. A suitable data preparation way for the SVM is min-max scaler normalization without imbalance class handling.

Table 27 - Summary of model comparison for candidates who filled in the experience

Model	Normalization		Imbalance class handling		Evaluation (average)			
	Standard scaler	Min-max scaler	Over sampling	Under sampling	Accuracy	Weighted f1-score	Recall	Computation time (sec)
DT				✓	0.77829	0.81760	0.76601	0.01180
SVM		✓			0.80474	0.83563	0.71186	0.12500
GNB			✓		0.77645	0.81347	0.63913	0.02442
RF		✓	✓		0.58073	0.51881	0.40000	0.16350
k-NN	✓			✓	0.74269	0.79041	0.72134	0.02736
CatBoost				✓	0.78013	0.81631	0.77470	0.05782
XGBoost		✓			0.86499	0.87355	0.55798	0.41033
CNN					0.90056	0.86121	0.05415	148.5132

Table 28 shows the best data preparation for each model for the data that candidates did not fill experience. 4 of 8 models show that the standard scaler normalization technique is the best way compared to without normalization, standard scaler normalization, and min-max scaler normalization. While 5 of 8 models show that the undersampling technique is the best way, compared without imbalance class handling, oversampling, and undersampling. According to average accuracy, weighted f1-score, recall, and computation time, the most suitable model is the DT. The 2nd and 3rd best models are the CatBoost and SVM respectively.

Table 28 - Summary of model comparison for candidates who did not fill in the experience

Model	Normalization		Imbalance class handling		Evaluation (average)			
	Standard scaler	Min-max scaler	Over sampling	Under sampling	Accuracy	Weighted f1-score	Recall	Computation time (sec)
DT		✓		✓	0.69991	0.75352	0.60000	0.06605
SVM	✓			✓	0.72378	0.78954	0.52000	0.04575
GNB	✓		✓		0.53275	0.63621	0.70000	0.05502
RF				✓	0.41370	0.36454	0.60000	0.32639
k-NN	✓			✓	0.59889	0.69656	0.58000	0.05358
CatBoost				✓	0.67718	0.74643	0.74000	0.07130
XGBoost					0.93405	0.91693	0.18000	0.27211
CNN	✓				0.93132	0.89820	0.00000	8.42467

5.5 The Most Suitable Model Training

After we gained the most suitable model on the data that candidates filled in experience, SVM, we trained the models again to find the best threshold used for the testing set. The best threshold will then be compared between 0.5, an average of threshold values from a 5-fold cross-validation, and a median of threshold values from a 5-fold cross-validation. Table 29 showed a threshold value in each fold for the data that candidates filled in experience. Figure 29 shows the receiver operating characteristic (ROC) curve in each fold with the best threshold value in the training set. The average area under the ROC curve (AUC) is 0.82764. The best threshold is 0.13229 as shown in Table 30.

Table 29 - A threshold value in each fold of the SVM model for candidates who filled in the experience

Model	A threshold value in each fold				
	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
SVM	0.10058	0.17399	0.16123	0.17196	0.05369

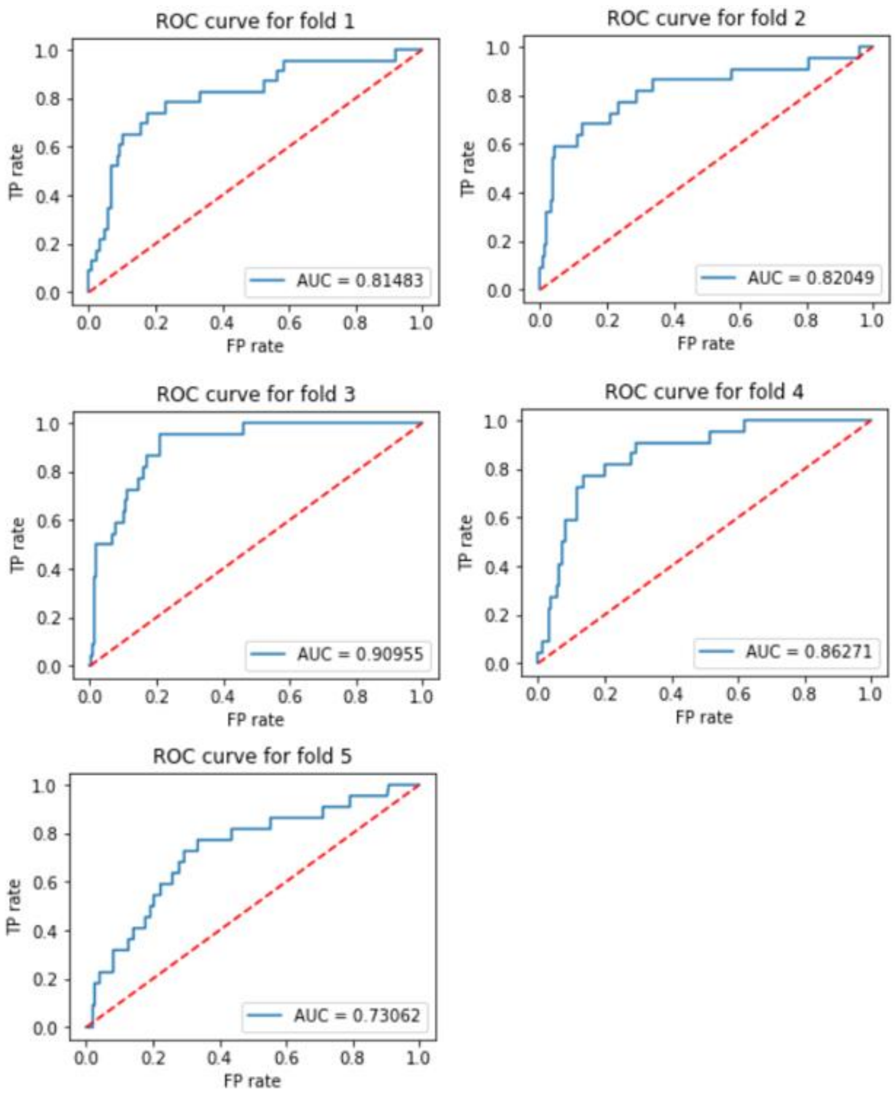


Figure 29 – The ROC curve in each fold of the SVM model for candidates who filled in the experience

Table 30 - The best threshold value of the SVM model for candidates who filled in the experience

Threshold		Evaluation (average)		
Type	Value	Accuracy	Weighted f1-score	Recall
Default	0.50000	0.89689	0.85456	0.02687
Mean	0.13229	0.80655	0.83696	0.71185
Median	0.16123	0.82207	0.84806	0.69407

On the other hand, after we acquired the most suitable model on the data that candidates did not fill in experience, DT, we trained the models again to find the best threshold used for the testing set. The best threshold will then be compared between 0.5, an average of threshold values from a 5-fold cross-validation, and a median of threshold values from a 5-fold cross-validation. Table 31 showed a threshold value in each fold for the data that candidates filled in experience.

Table 31 - A threshold value in each fold of the DT model for candidates who did not fill in the experience

Model	A threshold value in each fold				
	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
DT	0.48837	0.53125	0.90476	0.50000	0.47500

Table 32 - The best threshold value of the DT model for candidates who did not fill in the experience

Threshold		Evaluation (average)		
Type	Value	Accuracy	Weighted f1-score	Recall
Default	0.50000	0.67613	0.73194	0.46000
Mean	0.57987	0.81586	0.84165	0.24000
Median	0.50000	0.67613	0.73194	0.46000

Figure 30 shows the ROC curve in each fold with the best threshold value in the training set. The average AUC is 0.61363. The best threshold is 0.5 as shown in Table 32. The DT, CatBoost, and SVM will be used to predict the target probability. Each model's probability will be then used for candidate ranking by sum as well.

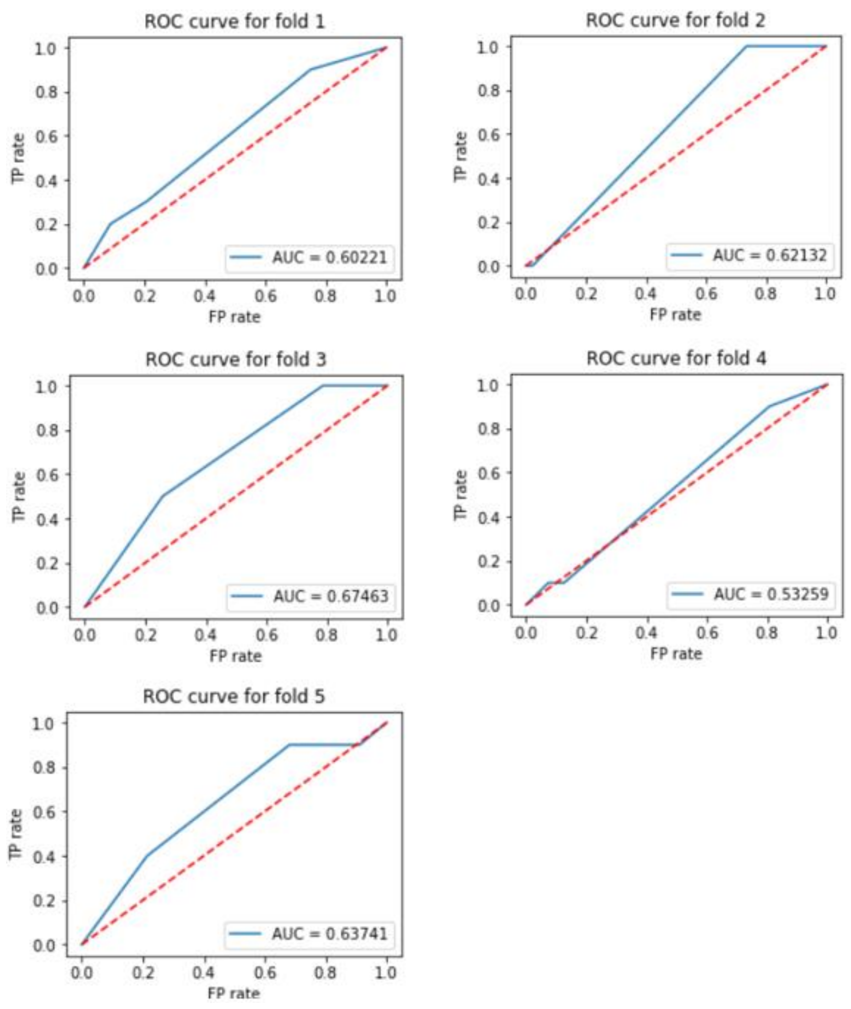
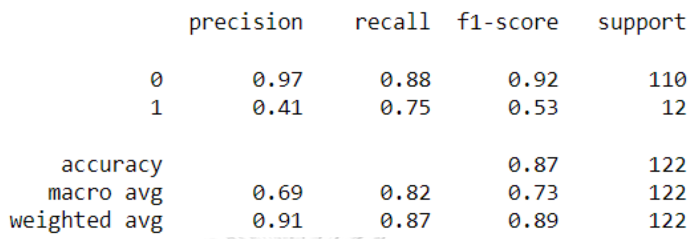


Figure 30 - The ROC curve in each fold of the DT model for candidates who did not fill in the experience

5.6 The Most Suitable Model Testing

For the data that the candidate filled in the experience, the RBF SVM model with min-max scaler normalization but without imbalance class handling was used to classify the testing set into 2 groups. The parameter value used consists of C equal to 3, scale gamma, and RBF kernel. Figure 30 shows the classification report of the SVM model for candidates who filled in the experience. The “0” means the not-suitable class and “1” means the shortlist class. From all 122 testing data, we obtained 97 TN, 13 FP, 3 FN, and 9 TP as shown in Figure 31. The recall is 75% while the weighted f1-score is 88%.



```
              precision    recall  f1-score   support

           0       0.97      0.88      0.92       110
           1       0.41      0.75      0.53        12

    accuracy                           0.87       122
   macro avg       0.69      0.82      0.73       122
weighted avg       0.91      0.87      0.89       122
```

Figure 31 - Classification report of the RBF SVM model for candidates who filled in the experience

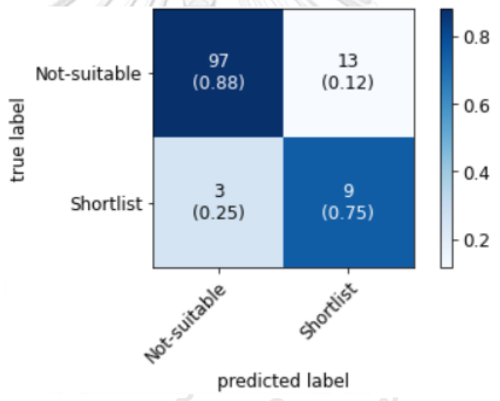


Figure 32 - Confusion matrix of the RBF SVM model for candidates who filled in the experience

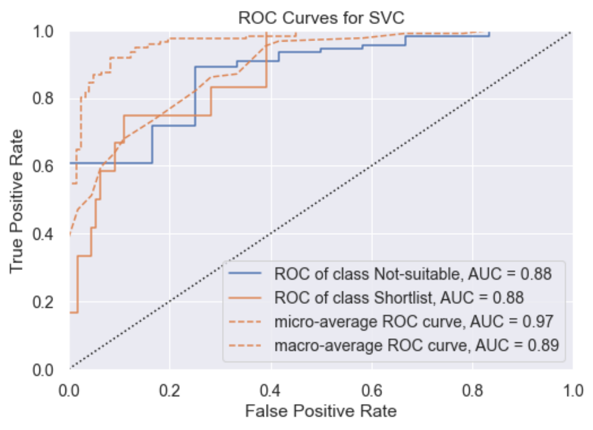
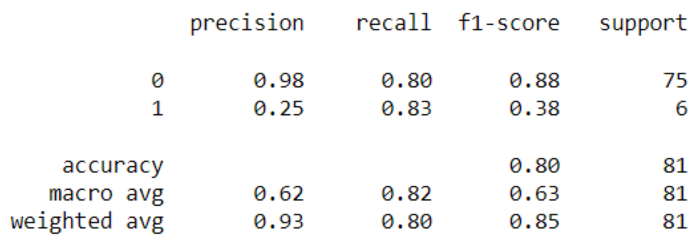


Figure 33 - ROC curve of the RBF SVM model for candidates who filled in the experience

The Receiver Operating Characteristic (ROC) curve for the RBF SVM is shown in Figure 33 which is more similar to the ideal clinical discriminator curve than the no-predictive curve (45-degree line). The Area Under the curve (AUC) of not-suitable and shortlist classes are equal, 88%. The AUC of micro average ROC is 97% while the AUC of macro average ROC is 89%. In the False Positive Rate (FPR) range of 0 to 0.2, the True Positive Rate (TPR) of the shortlist class increases significantly. Thereafter, it slows to 1 as the FPR approaches 0.4. The ROC curve for the not-suitable class is similar, but on the other side, the FPR value rises sharply when the TPR is between 0.6 and 0.9, after which it rises steadily.

For the data that the candidate did not fill in the experience, the DT model with min-max scaler normalization and undersampling imbalance class handling, the CatBoost model without normalization but undersampling imbalance class handling, and the SVM model with standard scaler normalization and undersampling imbalance class handling is used to classify the testing set into 2 groups. For the DT, the parameter value used consists of entropy criterion, splitter by random, max_depth equal to 2, and 5 min_samples_split. For the CatBoost, the parameter value used consists of iterations equal to 500, 0.01 learning rate, depth is 2, and 8 l2_leaf_reg. For the RBF SVM, the parameter value used consists of C equal to 0.05, scale gamma, and RBF kernel. Figure 34 shows the classification report of the DT model for candidates who did not fill in the experience. The DT acquired an accuracy of 80% and a weighted f1-score of 84%. Figure 35 shows the confusion matrix of the DT model for candidates who did not fill in the experience. From all 122 testing data, We acquired 60 TN, 15 FP, 1 FN, and 5 TP from the DT model. The DT's recall is 83%. We obtained 26 TN, 49 FP, 3 FN, and 3 TP from the SVM model. The SVM's recall is 50%. For the Catboost, we gained 46 TN, 29 FP, 1 FN, and 5 TP. The CatBoost's recall is 83%. The ROC curve for the DT model is shown in Figure 36. The AUC of not-suitable and shortlist classes are equal, 61%. The AUC of micro average ROC is 69% while the AUC of macro average ROC is 61%.



```
              precision    recall  f1-score   support

           0       0.98      0.80      0.88        75
           1       0.25      0.83      0.38         6

    accuracy                           0.80        81
   macro avg       0.62      0.82      0.63        81
weighted avg       0.93      0.80      0.85        81
```

Figure 34 - Classification report of the DT model for candidates who did not fill in the experience

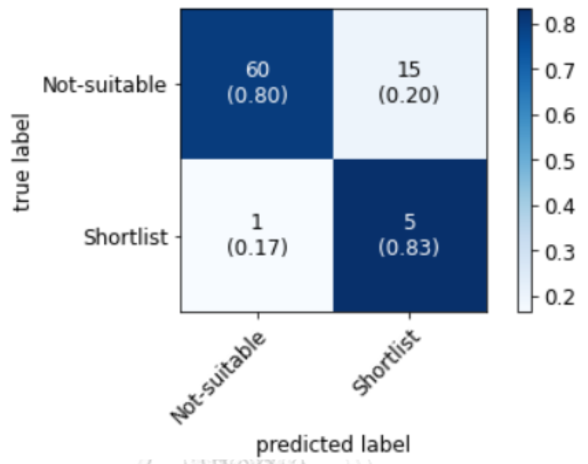


Figure 35 - Confusion matrix of the DT model for candidates who did not fill in the experience

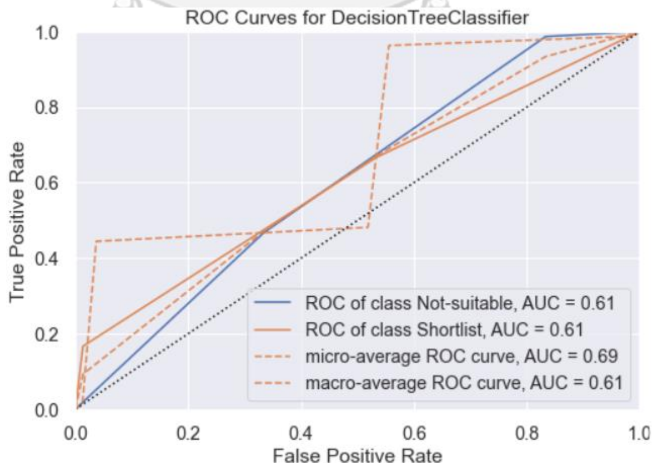
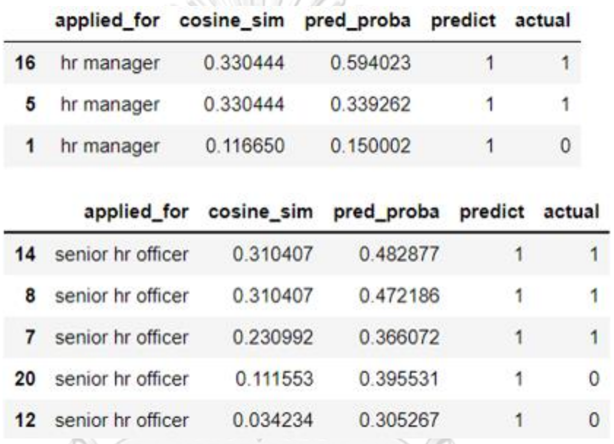


Figure 36 - ROC curve of the DT model for candidates who did not fill in the experience

5.7 Ranking Suggestion

From the previous section, we already classify the candidates into 2 groups: the shortlist group and the not-suitable group. The candidates who fill experience will be ranked in descending order based on the similarity between their experience and the job description they applied for. Figure 37 shows the example ranking of the HR manager and senior HR officer positions that candidates applied for. If anyone has the same cosine similarity value, the system will sort from predicted probability instead. The system can sort candidates in the shortlist group in descending order satisfactorily.

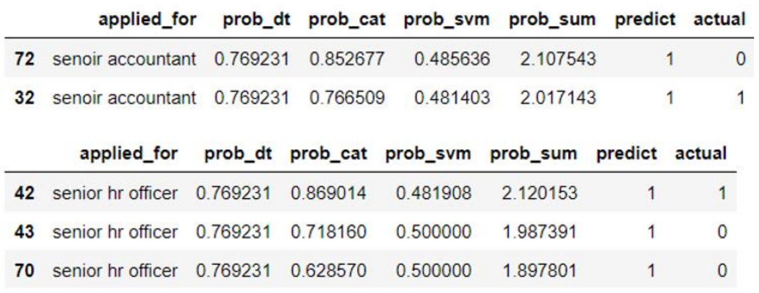


	applied_for	cosine_sim	pred_proba	predict	actual
16	hr manager	0.330444	0.594023	1	1
5	hr manager	0.330444	0.339262	1	1
1	hr manager	0.116650	0.150002	1	0

	applied_for	cosine_sim	pred_proba	predict	actual
14	senior hr officer	0.310407	0.482877	1	1
8	senior hr officer	0.310407	0.472186	1	1
7	senior hr officer	0.230992	0.366072	1	1
20	senior hr officer	0.111553	0.395531	1	0
12	senior hr officer	0.034234	0.305267	1	0

Figure 37 - Ranking of the HR manager and senior HR officer

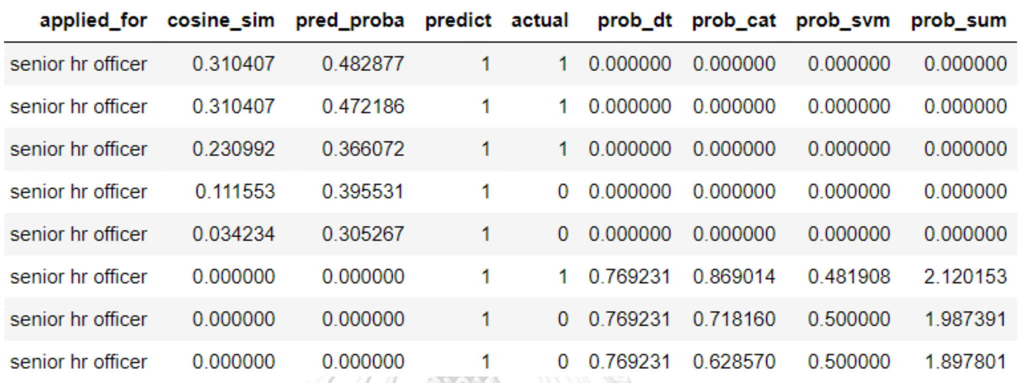
The candidates who did not fill experience will be ranked in descending order based on a summarization of the predicted probability of the best 3 models, which are the DT, CatBoost, and SVM. The ranking suggestion will be ranked from the candidates who fill experience first. Because this group of candidates shows their intention to apply for a job more than those who did not fill their experience. The ranking will be displayed separately according to the job title the candidates applied for. Figure 38 shows the example ranking of the HR manager and senior HR officer positions that candidates applied for. The predict column is the result of the DT classification. The system can sort candidates in the shortlist group in descending order satisfactorily even though some rows are not perfectly correct.



	applied_for	prob_dt	prob_cat	prob_svm	prob_sum	predict	actual
72	senoir accountant	0.769231	0.852677	0.485636	2.107543	1	0
32	senoir accountant	0.769231	0.766509	0.481403	2.017143	1	1

	applied_for	prob_dt	prob_cat	prob_svm	prob_sum	predict	actual
42	senior hr officer	0.769231	0.869014	0.481908	2.120153	1	1
43	senior hr officer	0.769231	0.718160	0.500000	1.987391	1	0
70	senior hr officer	0.769231	0.628570	0.500000	1.897801	1	0

Figure 38 - Ranking of the senior accountant and senior HR officer



applied_for	cosine_sim	pred_proba	predict	actual	prob_dt	prob_cat	prob_svm	prob_sum
senior hr officer	0.310407	0.482877	1	1	0.000000	0.000000	0.000000	0.000000
senior hr officer	0.310407	0.472186	1	1	0.000000	0.000000	0.000000	0.000000
senior hr officer	0.230992	0.366072	1	1	0.000000	0.000000	0.000000	0.000000
senior hr officer	0.111553	0.395531	1	0	0.000000	0.000000	0.000000	0.000000
senior hr officer	0.034234	0.305267	1	0	0.000000	0.000000	0.000000	0.000000
senior hr officer	0.000000	0.000000	1	1	0.769231	0.869014	0.481908	2.120153
senior hr officer	0.000000	0.000000	1	0	0.769231	0.718160	0.500000	1.987391
senior hr officer	0.000000	0.000000	1	0	0.769231	0.628570	0.500000	1.897801

Figure 39 - The final ranking of the senior HR officer

Figure 39 previews the results of the ranking suggestion of candidates for the senior HR officer position. Candidates 1st through 5th have filled their experience on JobsDB.com, while candidates 6th and lower are those who did not. The ranking after classification will help the efficiency of the system in recruiting people for job interviews better.

5.8 Discussion

A direct comparison of any existing approaches presented in various publications may not be fair due to the lack of a sizable dataset, the limitation of some tasks, or the difference in the effectiveness of methods. Our work focuses on considering which candidates to invite for an interview for a certain job opening while taking into consideration both general information and experience. Techniques are used in the proposed method at each step deriving from the fusion of interesting and effective techniques from related works. The results compared with other methods are based on the overall accuracy mentioned therein.

P. K. Roy et al. (2020) [28] used linear SVM to classify candidates into job function groups which is a different objective from our work. They serve the same purpose in recommending top candidates and obtained an accuracy of 78%. Z. Elgammal et al. (2021) [25] made use of the SpaCy library to calculate the similarity between jobs and candidates and then rank candidates to arrive at an accuracy of 83%. The method described by Tejaswini et al. (2021) [27] gives an accuracy of 92% using a smaller dataset and the accuracy significantly decreased as the number of candidates increased. The accuracy decreases by an average of 7.5% when there are 5 more candidates.

The proposed method got an average accuracy of 83.5%, a weighted F1 score of 86%, and a recall of 79%. Compared to other existing works by accuracy score, our work has higher accuracy than some and less than some. The cause of our lower accuracy due to the imbalance of data for the two classes. As mentioned in the previous section, we got the shortlist group data in just 10%, compared to up to 90% of the not-suitable group data. Further, our approach focuses on precisely classifying the shortlist group because we do not want to miss the chance to schedule interviews with qualified candidates with which we have this class data for ML to learn less.

The similarity between the experience of the candidates and the job description of the position they are applying for is one of the most important features used in considering candidates for interviews. This is because companies will always require employees who are knowledgeable about the types of positions that the company opens to be immediately workable with little training needed. Other features should always be taken into consideration, such as years of experience, the similarity between a previous job title and a job title they applied for, the distance between residence and workplace (in case of work from the office), and the highest education level, etc.

CHAPTER 6

SUMMARY AND FUTURE WORK

6.1 Summary

The procedure of screening candidates for job interviews from among all job applicants is a repetitive task that takes a lot of time and resources. This could be caused by human error also. To address this issue, we have proposed a job-candidate classifying and ranking system-based machine learning method that recommends suitable candidates to human resources (HR) based on the job description and candidate profile. The NLP approach which consists of summarization, stop word removal, and lemmatization is used in our work for preprocessing the job opening and experience data. The Decision Tree (DT), Support Vector Machine (SVM), Gaussian Naïve Bay (GNB), Random Forest (RF), k-Nearest Neighbors (k-NN), CatBoost, Extreme Gradient Boosting (XGB), and Convolutional Neural Network (CNN) were compared. As the result, the RBF SVM classifier is the most suitable for our objective and for the data that candidates fill in experience. This part yielded an accuracy of 87%. On the other hand, the DT classifier is the most suitable for our objective and for the data that candidates did not fill in experience and acquire an accuracy of 80%. Resulting in an overall accuracy of 83.5%. Involving domain experts, and HR professionals, assist in developing a more accurate model, and their feedback helps to enhance the model repeatedly.

6.2 Recommendation for Future Work

For future work, the work can (a) add more training data from the candidate and company, (b) compare with another classification model, and (c) perform an end-to-end classification and ranking software.

REFERENCES

[1] E. Paulsen, "Retaining Talent in a New World of Work," ed: Quantum Workplace, 2021.

[2] "ข้อมูลสถิติของผู้หางาน JobsDB ประเทศไทย เดือนมกราคม - มีนาคม 2565." [Online]. Available: https://th.jobsdb.com/th-th/cms/employer/home/candidate-fact-sheet/

[3] I. S. Formula, "The Job Search Today," ISF-JobSearchToday972, Ed., ed.

[4] "2021 Recruiter Nation Report." [Online]. Available: https://www.jobvite.com/wp-content/uploads/2021/09/Jobvite-RecruiterNation-Report-WEB-2.pdf

[5] "Eye-Tracking Study." [Online]. Available: https://www.theladders.com/static/images/basicSite/pdfs/TheLadders-EyeTracking-StudyC2.pdf

[6] M. Slack, "5 Problems with The Ladders' 6 Second Resume Study," ed: Resume Genius, 2014.

[7] "50 HR & Recruiting Stats That Make You Think," ed: Glassdoor, 2015.

[8] R. K. Mariette Awad, *Efficient Learning Machines*. Apress Berkeley, CA, 2015, pp. XIX, 268.

[9] M. P. S. Shovan Chowdhury, "Research Paper Classification using Supervised Machine Learning Techniques," presented at the 2020 Intermountain Engineering, Technology and Computing (IETC), Orem, UT, USA, 2020.

[10] J. K. A. Intisar Shadeed Al-Mejibli, Dhafar Hamed Abd, "The effect of gamma value on support vector machine performance with different kernels," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 10, pp. 5497-5506, 2021, doi: 10.11591/ijece.v10i5.pp5497-5506.

[11] B. J. Shikha Agarwal, Tisu Kumar, Manish Kumar, Prabhat Ranjan, "Hybrid of Naive Bayes and Gaussian Naive Bayes for Classification: A Map Reduce Approach," *International Journal of Innovative Technology and Exploring Engineering (IJITEE),* vol. 8, no. 6S3, 2019.

[12] Y. S. L. Rajeev Raizada, "Smoothness without Smoothing: Why Gaussian Naive Bayes Is Not Naive for Multi-Subject Searchlight Studies," *PloS one,* vol. 8, 2013, doi: 10.1371/journal.pone.0069566.

[13] R. K. Jehad Ali, Nasir Ahmad, Imran Maqsood, "Random Forests and Decision Trees," *IJCSI International Journal of Computer Science Issues (IJCSI),* vol. 9, no. 5, 2012.

[14] O. A. Eric Guérin, Ali Mahdavi-Amiri, "Artificial Intelligence," in *Manual of Digital Earth*, 2019, ch. 10.

[15] C. A. Sarica Alessia, Quattrone Aldo, "Random Forest Algorithm for the Classification of Neuroimaging Data in Alzheimer's Disease: A Systematic Review," *Frontiers in Aging Neuroscience,* vol. 9, 2017, doi: 10.3389/fnagi.2017.00329.

[16] R. L. R. Abdullahi A. Ibrahim, Muhammed M. Muhammed, Rabiat O. Abdulaziz, Ganiyu A. Saheed, "Comparison of the CatBoost Classifier with other Machine Learning Methods," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 11, no. 11, 2020, doi: 10.14569/IJACSA.2020.0111190.

[17] L. Ostroumova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush and Andrey Gulin, "CatBoost: unbiased boosting with categorical features," in *32nd Conference on Neural Information Processing Systems (NeurIPS)*, Montréal, Canada., vol. 31: Curran Associates, Inc.

[18] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc., 2019.

[19] F. X. Patrick Schneider, *Anomaly Detection and Complex Event Processing over IoT Data Streams*. Elsevier Inc., 2022.

[20] S. H. Takahiro Nakao, Yukihiro Nomura, Masaki Murata, Tomomi Takenaga, Soichiro Miki, Takeyuki Watadani, Takeharu Yoshikawa, Naoto Hayashi, Osamu Abe, "Unsupervised Deep Anomaly Detection in Chest Radiographs," *Journal of Digital Imaging,* vol. 34, pp. 418–427, 2021, doi: 10.1007/s10278-020-00413-2.

[21] S. Sunitha, "An Overview of Deep Learning," *International Journal of Engineering Research & Technology (IJERT),* vol. 9, no. 5, 2021.

[22] M. N. Rikiya Yamashita, Kaori Togashi, Rikiya Yamashita, Richard Kinh Gian Do, Mizuho Nishio, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging,* vol. 9, pp. 611–629, 2018, doi: 10.1007/s13244-018-0639-9.

[23] D. C. Ajay Kulkarni, Feras A. Batarseh, "Foundations of data imbalance and solutions for a data democracy," in *Data Democracy*: Elsevier Inc., 2020, ch. 5, pp. 83-106.

[24] Yellowbrick. "Classification Report." https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html (accessed.

[25] A. B. Z. Elgammal, H. Hassan, K. Elgammal, T. Özyer and R. Alhajj, "Matching Applicants with Positions for Better Allocation of Employees in the Job Market," presented at the 22nd International Arab Conference on Information Technology (ACIT), Muscat, Oman, 2021.

[26] E. P. C. Leah G. Rodriguez, "Feature Selection for Job Matching Application using Profile Matching Model," presented at the 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), Singapore, 2019.

[27] U. V. Tejaswini K, Shashank M Kadiwal, Sanjay Revanna, "Design and Development of Machine Learning based Resume Ranking System," in *Global Transitions Proceedings*, 2021: ScienceDirect, doi: 10.1016/j.gltp.2021.10.002.

[28] S. S. C. Pradeep Kumar Roy, Rocky Bhatia, "A Machine Learning approach for automation of Resume Recommendation system," *Procedia Computer Science,* vol. 167, pp. 2318-2327, 2020, doi: 10.1016/j.procs.2020.03.284.

[29] H. L. Yiou Lin, Prince Clement Addo, and Xiaoyu Li, "Machine Learned Resume-Job Matching Solution," 2016.

[30] J. W. Jing Zhao, Madhav Sigdel, Bopeng Zhang, Phuong Hoang, Mengshu Liu, Mohammed Korayem, "Embedding-based Recommender System for Job to Candidate Matching on Scale," 2021, doi: 10.48550/ARXIV.2107.00221.

[31] S. González-Carvajal, Garrido-Merchán, Eduardo, "Comparing BERT against traditional machine learning text classification," 2020. [Online]. Available: https://arxiv.org/pdf/2005.13012.pdf.

[32] D. K. Sotiris Kotsiantis, Panayiotis Pintelas, "Handling imbalanced datasets: A review," *GESTS International Transactions on Computer Science and Engineering,* vol. 30, pp. 25-36, 2006.

[33] "sklearn.tree.DecisionTreeClassifier." scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html (accessed.

[34] "sklearn.svm.SVC." scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed.

[35] "sklearn.naive_bayes.GaussianNB." scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html (accessed.

[36] "sklearn.ensemble.RandomForestClassifier." scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed.

[37] "sklearn.neighbors.KNeighborsClassifier." scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (accessed.

[38] "Training Parameters Overview." CatBoost. https://catboost.ai/en/docs/references/training-parameters/ (accessed.

[39] "XGBoost Parameters." dmlc XGBoost. https://xgboost.readthedocs.io/en/stable/parameter.html (accessed.

VITA

NAME	Thapanee Boonchob
DATE OF BIRTH	28 August 1996
PLACE OF BIRTH	Samutprakan
INSTITUTIONS ATTENDED	Bachelor of Engineering Program in Computer Engineer, King Mongkut’s Institute of Technology Ladkrabang
HOME ADDRESS	1561 Moo.4 Soi.Sriboonrueang, Theparak Road, Samutprakan 10270