Fundamentals of Data Science Project

Decoding Programming Language Trends in Indonesia:
Insights from Clustering and Predictive Analysis

BY:

Ella Raputri (2702298154)
Ellis Raputri (2702298116)

Class L3AC
Computer Science Program
School of Computing and Creative Arts

Bina Nusantara International University
Jakarta
2024

A. PROBLEM ANALYSIS

Nowadays, technology is indispensable in our lives. Various applications and websites have played major roles in our era, enabling the automation of numerous activities requiring significant time and accuracy. From within the infrastructure of those technologies, artificial language exists to describe the underlying process that the computer has to follow. The said artificial language will be translated into machine language and executed by computers [1]. This artificial language is often referred to as the term “programming language”. Programming languages serve as a foundation for a program or software, playing a crucial role in technology growth. They enable developers to create mobile to desktop applications and enforce complex algorithms for data analysis and AI engineering.

For these past decades, programming languages have been evolving rapidly thanks to the involvement of technology growth and innovations. The progression of programming languages can not be separated from the evolution of programming language features throughout the year, where old programming languages acquire new features and new programming languages for specific problems are born. As a result, there exists an abundant amount of programming languages out there, with different features and characteristics. Due to the dynamic and fast-paced nature of the technology industry, programming languages have also experienced major popularity adjustments over time. Some general-purpose programming languages may not address problems as profoundly as the new domain-specific languages [2]. Hence, many of them are eliminated and replaced by new emerging languages.

In Indonesia, the significance of programming languages is also indisputable as in recent years, the Ministry of Industry has devised a roadmap policy to achieve digital transformation in Indonesia 4.0 [3]. In other words, Indonesia has seen potential influences of digital technology development in various sectors, from economy, fintech, and politics, to culture and education. Sectors like fintech, commerce, and education have been the leading sector in instigating demands for skilled programmers. The government initiatives of “Making Indonesia 4.0”, accompanied by the digital growth of diverse sectors in Indonesia have further pushed the need for programming skills in areas, such as data analysis, artificial intelligence, big data, and other enterprise-level systems [4]. As the demand for digital solutions grows, maintaining the ability to adapt popular programming languages to Indonesia's system has become more crucial as it can affect Indonesia's competitiveness in the world's digital landscape.

Globally, the popularity of programming languages can be tracked using the TIOBE Index. TIOBE Index is a rating of programming languages based on the number of skilled developers, courses, etc. worldwide. It is calculated at several popular sites, like Google, Bing, and Amazon. TIOBE Index can be used to determine the ranking of the programming language globally, thereby providing insights to the global and local developer about the current global trend of programming languages, helping them to make impactful decisions about which languages are suitable for their future projects.

However, global trends are not necessarily the same as local trends as challenges exist in adopting a particular programming language in Indonesia. One example is C++ which is popular globally but unfavored in Indonesia [5]. Another example is Javascript, which is one of the most famous languages in Indonesia due to the local trend of using website applications for promotion and other tasks. Moreover, different sectors may have unique challenges and issues, leading to distinct preferences for programming languages and diverse effects that the programming language has on that sector.

The domain of analyzing the effect of global popularity ratings of programming languages on the job market and search trends in Indonesia is crucial for tech industries and education institutions. With this research, we wish to predict the factor most affected by the global ratings of programming languages using machine learning. The result can give insights to developers in tech and educational companies to further determine their decisions in making certain projects or curricula for students. New Indonesian programmers who want to pursue a career in Computer Science will also be more aware of the most advantageous language to learn based on their needs. Understanding the dynamics between the trends can help us explore how the global trends of programming language can shape the sector technology landscape in Indonesia.

In conclusion, our research seeks to analyze the effect of global programming language trends on the job market and search trends in Indonesia and determine the factors affected by language ratings with machine learning. By doing this research, we also aim to conclude which machine learning technique is the most precise in predicting the correlation between the global trends of programming language and local components in Indonesia. By doing so, we can further analyze their interrelationship and use it to generate valuable insights into the tech outlook in Indonesia.

B. RELATED WORKS

In this section, we aim to discuss similar works related to the topic to gather new insights and observations that help us formulate decisions. We will gather data about programming languages, especially their connections with fields such as workplace, education, community, and usability.

Programming languages are artificial languages made by humans to communicate with computers. Programming languages enable a human to command the computer with a given flow and algorithm [6]. This concept of programming arose back in 1822 when Charles Babbage invented the computer. From then on, programs, or the set of instructions to communicate with the computer, have been prominent until now [6].

With the increasing usage of programming language, many studies have been conducted to analyze the effects of programming language popularity on human society. One of the studies is the study [7] that aims to delve into the popularity, interoperability, and impact of programming languages, especially in GitHub open sources. They tried to answer the question about programming language popularity by assessing 100,000 real-world software projects and determining the correlation between programming language, issue reports, project success, and development team size [7].

Another study tried to analyze the performance of programming languages in terms of the decision-making process [8]. The authors analyzed and compared Golang (Go) to Java and Python by their calculation time, RAM usage, and CPU usage to assess the effectiveness of these three languages in commercial remote systems associated with target customer preferences. They implemented the decision tree algorithm to help them compute and analyze the data. In the end, they found that Golang's performance does not differ significantly from Python and Java, thus making it suitable for development despite its popularity being lower than Python and Java.

Another popular study tried to examine the effect of programming languages on academic achievement and programming anxiety levels [9]. The author used the random sampling method to determine the participants in the study. After that, the author implemented a series of performance evaluations of the participants by pre-test and post-test. In the end, the author concluded that the integration of programming language for theory and practice in a course could increase academic achievement and in-class performance while reducing programming anxiety.

Another similar study aimed to study the correlation of programming language to development productivity [10]. They studied the code quality, development speed, and maintenance effort of programmers by analyzing various development environments. The result showed that high-level programming languages work better in enhancing development productivity, while low-level programming languages perform better for projects that emphasize the use of time or space. This study aims to provide insights to developers to choose the right programming language before starting a project so they can have more optimized productivity and results.

Lastly, the study [11] used the Stack Overflow annual survey as their dataset to analyze the correlation between the popularity of programming languages and demographic factors and job satisfaction. The study found that different gender and age have significant differences in determining the programming language they use. The job satisfaction factor also shows a substantial connection with programming languages, with some programming languages having lower job satisfaction than others.

However, despite the in-depth studies shown above, there are not many studies about the effects of global popularity rankings on the job market and search trends. The studies also tend to focus on a global scale, which is remarkably different from our study target which focuses only on Indonesia.

The most similar study that we found is a study [12] that determined to index the programming languages in Indonesia based on their popularity to serve as a guideline for the local information technology field. While this research is similar to ours in terms of the location scale, our main objectives are different. Study [12] aims to rank popular programming languages, while our goal is to determine the effects of popular programming languages in several sectors, such as job and community. The methods also differ, where they used the multi-attributive border approximation area comparison (MABAC) method, while we used machine learning techniques.

In conclusion, through this study, we aim to use machine learning techniques to determine the correlation between programming languages' global popularity ratings and local aspects, thereby generating valuable insights for developers and decision-makers in Indonesia.

C. DATASET AND PREPROCESSING

Since we aim to predict the impacts of programming language global ratings on job and search trends, we need datasets that align with them. We also need the global popularity ratings datasets to determine the rankings of each programming language. We decided to use the TIOBE index as our benchmark to determine the global popularity ratings. For the other datasets, because we need local (Indonesian) datasets, we mainly scrap the web to acquire them. Below is the list of datasets we use:

- Programming language list: Kaggle repository
- Programming language rating: TIOBE Index
- Job datasets: total job listings, salary, and location from Indeed and Jobstreet, number of people who state the programming language in their skills in LinkedIn.
- Search trends datasets: Wikipedia Pageviews Analysis, Google Trends, Stack Overflow Developer Survey, GitHub users search

Below is each dataset's details and preprocessing method:

a. Programming Language Name List

The initial step that we do is finding a dataset of all the programming languages in this world. We first found a Kaggle dataset by Sujay Kapadnis[1] which contains names of programming languages and their description, type, and so on. However, after further inspection, the repository is sourced from a more complete programming language repository by the same author[2]. The dataset contains about 4000 programming language names and 135000 facts about them (its description, features, etc). Upon further investigation, the datasets are obtained from pldb.io[3]. Pldb.io is a public-domain scroll set and website that serves as a database for programming languages. Its sources can be seen on their acknowledgment page[4].

After acquiring the programming language dataset, we dropped all the invalid records. For example, the OS Android is also inside the dataset. So, we dropped the records that have the invalid type (not a programming language). At first, we wanted to use the data (GitHub repos, Stack Overflow, etc.) inside the dataset. However, because the data is not local, we decided not to use it. Finally, we convert the list of the names of the programming languages into a CSV file (nama.csv).

b. Job Dataset

To acquire the job dataset, we did web scraping from id.indeed and jobstreet Indonesia which are both popular job-searching sites. The raw scrapped CSV files from id.indeed have six attributes, i.e. job title, company name, location, salary, job type, and job description. Meanwhile, raw CSV files from jobstreet contain seven attributes, i.e. job title, company name, location, salary, job type, job description, and date. Then, we decided to drop the date attribute from the jobstreet CSV files since it has many NaN values and we need to uniform the acquired CSV files' attributes.

For the preprocessing method, we fill the NaN values in the salary attributes using the mean strategy in the SimpleImputer class. Then, we

[1] See more at https://www.kaggle.com/datasets/sujaykapadnis/programming-languages

[2] See more at https://www.kaggle.com/datasets/sujaykapadnis/programming-language-database

[3] See more at https://github.com/breck7/pldb

[4] See more at https://pldb.io/pages/acknowledgements.html

concatenate all files and remove the duplicates. After that, we clean the job title and extract the programming language from the job description. Lastly, we remove the possible outliers from the datasets. Thus, we get a list of jobs in job_cleaned.csv.

Next, we aim to transform the CSV file before to a new CSV file with the programming language name as the key attribute. So, we calculate the job amount and average salary based on each programming language. We also extract the job location into the new CSV file. The final result is then stored in language_list.csv.

c. Wikipedia Monthly Page Views Dataset

To determine the trend of a specific programming language, we decided to check for their monthly Wikipedia page views in Wikipedia Pageviews Analysis[5]. However, not all programming languages have their own Indonesian Wikipedia page, so we only record those who have it. We combined all the data of individual programming languages into a single dataset that contains the monthly page view from July 2015 to September 2024. After that, we calculate the average monthly views for each programming language, starting from the first month when the view is not 0 (when the page is first created). The result is stored in avg_wiki_monthly.csv.

d. Stack Overflow Annual Developer Survey Dataset

Besides Wikipedia Page monthly views, we also search for languages that Stack Overflow users (developers) utilize or desire to learn. We obtain the data from Stack Overflow Insights[6] and further clean it into a single dataset by dropping all rows with country attributes not equal to ‘Indonesia’ and dropping unrelated columns. Then, for users who do not fill in the information about the languages that they have used or desired, their records are also dropped. After that, we count the occurrences of each language in the column. The result is in count_stack_overflow.csv.

[5] See more at https://pageviews.wmcloud.org/

[6] See more at https://survey.stackoverflow.co/

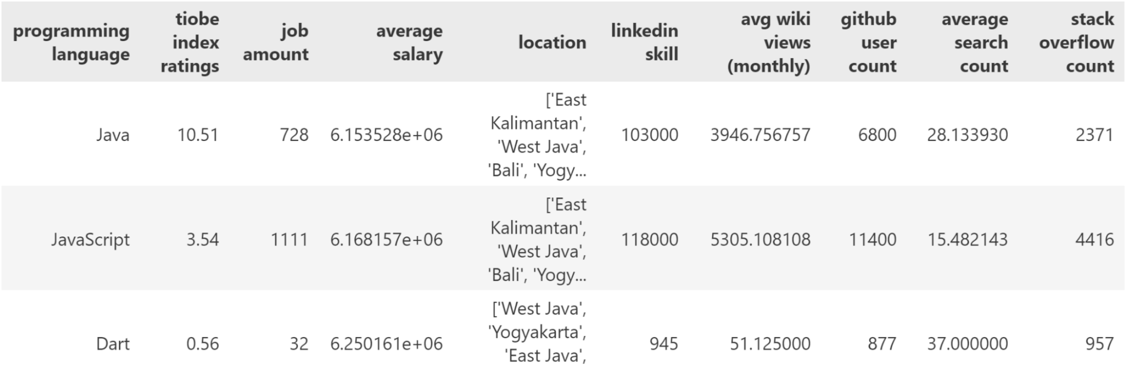
e. Other Datasets

For the other datasets, we use the sweat equity strategy to gather the data. These datasets include the GitHub users, Google Trends, TIOBE index rating, and LinkedIn skill datasets. In other words, we gather the data manually for these datasets. We search and record the search results in the CSV files.

After acquiring all datasets, we merge them into a CSV file called result.csv. The total language we got is 175, which each consists of ten attributes as below:

Attribute Name	Description	Data Type
programming language	The name of the programming language	object (string)
tiobe index ratings	The rating of the programming language based on the TIOBE index	float
job amount	The number of jobs based on the programming language	int
average salary	The average salary of jobs in a specific programming language	float
location	The location (province) of programming language jobs	object (list)
linkedin skill	The number of people who put the programming language as their skill on LinkedIn	int
avg wiki views (monthly)	The average of Wikipedia views on the programming language page monthly	float
github user count	The number of GitHub users who used the programming language in their repository	int
average search count	The average number of searches done by people monthly	float
stack overflow count	The number of Stack Overflow users that desire or have worked with the specific programming language	int

To make it clearer, below is the dataset snippet:



programming language	tiobe index ratings	job amount	average salary	location	linkedin skill	avg wiki views (monthly)	github user count	average search count	stack overflow count
Java	10.51	728	6.153528e+06	['East Kalimantan', 'West Java', 'Bali', 'Yogy...	103000	3946.756757	6800	28.133930	2371
JavaScript	3.54	1111	6.168157e+06	['East Kalimantan', 'West Java', 'Bali', 'Yogy...	118000	5305.108108	11400	15.482143	4416
Dart	0.56	32	6.250161e+06	['West Java', 'Yogyakarta', 'East Java',	945	51.125000	877	37.000000	957

We also got the time series data for the TIOBE index, Wikipedia, Google Trends, and Stack Overflow. The details of each time series dataset can be seen below.

Data	Time Period	Programming Language Count*
TIOBE index	June 2001 to October 2024 (Monthly)	20
Google search count (Google Trends)	July 2015 to October 2024 (Monthly)	55
Wikipedia search count	July 2015 to October 2024 (Monthly)	49
Stack Overflow user count	2014 to 2024 (Yearly)	48

*the programming language count only specify the programming languages that has at least a value that is greater than zero in its time series data

D. MODEL AND TECHNIQUES

In this project, we used both supervised and unsupervised machine learning. We used supervised machine learning to do time-series forecasting aiming to predict the trends of global popularity ratings and search trends of programming languages in Indonesia. Since we only have a single value for each forecasting series, we used models that perform univariate time series forecasting. We used four models which are listed below:

1. **Autoregressive Integrated Moving Average (ARIMA)**

   ARIMA model is a time series forecasting model that aims to predict future value based on past values and error rates with the addition of nonstationary models introduced by Box and Jenkins [13].

2. **Simple Exponential Smoothing (SES)**

   The SES model is mainly used for short-range forecasting which assumes a mean in the data with no trend. It uses a weighted moving average, emphasizing the weight value of more recent observations [14].

3. **Holt-Winters**

   Holt-Winters model is a forecasting model that analyzes time series data based on its pattern. It combines the exponential smoothing method to provide better forecasting for seasonal data that often fluctuates throughout a period [15].

4. **Prophet**

   Prophet is a forecasting model developed by Facebook to handle complicated time series forecasting, especially for seasonal data. It can analyze and evaluate the trends of data with various seasonal patterns, from daily, weekly, monthly, or annually [16].

Since not all of our data is time series data, thus we can only perform the forecasting on global popularity rankings (TIOBE index), Google search count, Wikipedia search count, and Stack Overflow user count. We also only performed the forecasting for programming languages that have a value greater than 0 for the data specified.

To better understand our data, we use unsupervised machine learning methods to determine the clusters of each programming language based on all of the features. The goal of partitioning the programming languages into several clusters is to understand how the data in one cluster differs from the others and detect the distribution or density of the data in the space. Before performing the algorithm, we drop the location column and change it to location_count which represents how many job locations a particular programming language has. After that, we standardize all the data using the StandardScaler from sklearn to enable more accurate data analysis. Below are the unsupervised clustering models that we used:

1. **K-Means Clustering Algorithm**

   The k-means algorithm is one of the most well-known unsupervised machine learning algorithms. It initializes random cluster centers, assigns each data point to a cluster with the nearest mean (the least squared Euclidean distance), and then alternates between updating the best-estimated cluster centroids and each data point assignment to the clusters [17]. The number of clusters (k) is fixed and has to be known before starting the algorithm.

2. **DBSCAN**

   DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that determines the cluster structure of the data points based on their underlying density. The main idea of DBSCAN is to assign points to the same cluster if they are 'density-reachable' from each other [18]. In other words, the new point is within a particular distance from the old point.

3. **Gaussian Mixture Model (GMM)**

   GMM is one of the popular unsupervised clustering machine learning algorithms that determines the probability of each data point assigned to a cluster. The algorithm assumes that each cluster adheres to the Gaussian distribution and group observations based on the estimation of the cluster characteristics [19].

4. **Agglomerative Clustering**

   Agglomerative clustering is a hierarchical clustering, where it forms clusters of data points based on their similarity. Agglomerative clustering uses a bottom-up approach, where each item is initially a single cluster. Then, some points are merged based on their similarity into a large cluster [20].

5. **SOM (Self-Organizing Maps)**

   The SOM method is a machine learning algorithm that groups observations based on input values, where each grouping process is based on the features of the data. In other words, data points can only be considered in the same cluster if they share similar characteristics. Otherwise, they will be treated as separate clusters [21].

6. **OPTICS (Ordering Points To Identify the Clustering Structure)**

   OPTICS is a relatively new clustering algorithm that can be seen as an extension of the DBSCAN algorithm because it also assigns clusters based on density. However, OPTICS does not directly produce the clusters of the dataset but instead creates an ordering of the database based on its density-based structure [22]. It is different from DBSCAN because it can detect clusters with varying densities.

7. **BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)**

   BIRCH is a clustering algorithm that creates clusters of the dataset by generating a compact summary of the large dataset while preserving the large original dataset detail as much as possible. BIRCH can only be used for number attributes because it compresses the dataset into a Clustering Features (CF) Tree which consists of many Clustering Features entries [23]. Each entry is represented in a tuple of N (data points amount), LS (linear sum of N data points), and SS (squared sum of N data points).

For the visualization, we also use PCA (Principal Component Analysis) to reduce the dimension of the data (because it has many features) to only 2 dimensions while retaining the relevant information of the original data. PCA does this by finding a new variable that can maximize the variance of our data [24]. Therefore, we can visualize the data points and their cluster into a scatter plot, increasing our interpretability of the clustering result.

Besides unsupervised machine learning, we also experimented with classification-typed supervised machine learning algorithms for our time series data. Before we fed our machine with the data, we also standardized the data since they had different measurements. We standardized them into a value between 0 to 1, to make it easier to evaluate later. Below are the algorithms that we utilized:

1. **Decision Tree**

   Decision tree is one of the most widely used classification machine learning algorithms. A decision tree predicts the result by comparing the values of a feature to a certain threshold and classifying them into different scenarios [25]. The different scenarios of a feature can be seen as a branch of the root, where each node of the branch can also generate branches and be a parent of other child nodes.

2. **Random Forest**

   Random forest is a refinement of the decision tree classification machine learning algorithm. A single decision tree model is usually ineffective for analyzing complex data with many features. However, a group of trees that have different features or training sets will perform better as they are biased in their way, thereby when the result is aggregated, it will return better predictions [26].

3. **XGBoost (Extreme Gradient Boosting)**

   XGBoost is a model based on the idea of boosting tree models. XGBoost performs a second-order Taylor expansion on the loss function and other variety of methods to reduce the chance of overfitting the data [27].

4. **LightGBM (Light Gradient Boosting Machine)**

   LightGBM is a relatively new implementation of the Gradient Boosting Decision Tree that utilizes two techniques which are GOSS (Gradient-based One Side Sampling) and EFB (Exclusive Feature Bundling) [28]. GOSS focuses on higher error samples and randomizes the rest, while EFB groups multiple mutual features into a single feature. Therefore, LightGBM will be able to accelerate the training process.

5. **SVC (Support Vector Classification)**

   SVM (Support Vector Machine) is a supervised classification machine learning algorithm that classifies the data by finding the optimal hyperplane that maximizes the distance between two categories. SVC is a two-class model that learns to maximize the interval between two categories and then transforms the function into a convex quadratic programming solution [29].

6. **KNN (K-Nearest Neighbors)**

   KNN is regarded as one of the simplest algorithms in machine learning. The idea of KNN is to find k nearest neighbors of a data point and then predict the points based on the classification of its k nearest neighbors [30]. Therefore, we have to choose an initial k before executing the algorithm.

7. **Neural Networks MLP (Multilayer Perceptron)**

   MLP is one of the commonly used neural network algorithms for supervised learning. The objective of this algorithm is to find the optimized function that maps inputs to the desired output. MLP utilizes the backpropagation algorithm by changing the weight between connections if there is a contrast between the desired and the actual output of the algorithm [31].

8. **Naive Bayes**

   Naive Bayes is an efficient machine learning algorithm that assumes all attributes are independent, even though it violates many real-world cases [32]. It is constructed based on Bayes' Theorem. The algorithm assumes that each feature has an independent contribution to the probability of a data point.

9. **Logistic Regression**

   Logistic regression is a supervised machine learning algorithm that predicts the probability or categorical outcome of a data point based on one or more features. Its concept centers around the natural logarithm of an odd ratio [33]. It uses the logistic function to predict the probability of each value and uses a certain threshold to decide the category of the value.

Since our data value is not much and quite imbalanced, we decided to use SMOTE (Synthetic Minority Oversampling Technique). SMOTE is a preprocessing technique that is significantly used to counter the issue of imbalance class in supervised machine learning [34].

We also did hyperparameter tuning using the GridSearchCV from Python. GridSearchCV is a tuning method that aims to search for the best hyperparameters by scanning a number of user-selected hyperparameters [35]. With this method, our goal is to optimize the model as much as possible, so it can predict more accurately.

When training the models, we utilized the Python libraries. ARIMA, SES, and Holt-Winters models are imported from the statsmodel library, while Prophet is imported from its own library. Meanwhile, for unsupervised learning, we use K-Means, DBSCAN, GMM, Agglomerative Clustering, OPTICS, and BIRCH from the sklearn library. SOM is imported from the MiniSom library. Lastly, for classification-typed supervised machine learning, we imported the models from sklearn library, except for XGBoost from xgboost library and LightGBM from lightgbm library. Besides that, we also used matplotlib for plotting, sklearn for evaluating, SMOTE, and PCA, NumPy for calculation purposes, and pandas to handle data frames.

E. EVALUATION METHOD

For time series forecasting, we performed k-fold cross-validation with k=10, except for Stack Overflow data which used k=5 because of data scarcity. In each process, we calculated each programming language's mean absolute error, mean squared error, and root mean squared error, which will be averaged in the end. Below is the explanation of each evaluation metric that we used:

1. Mean Absolute Error (MAE)

MAE is a widely popular metric to evaluate model performance. It measures the average error magnitude between predicted and actual values. MAE gives the same weight to all errors, which means it uses absolute value to assess the model. Thus, changes in MAE are usually monotonic linear, and intuitive [36].

2. Mean Squared Error (MSE)

MSE is a metric that measures the average squared difference between the predicted value and the actual value that is being fed into a model. MSE is suited for assessing model performance, especially data with normal distribution. MSE can calculate all the training results into a single value that represents whether the model is good or bad. The lower the MSE, then the better the model is [37].

3. **Root Mean Squared Error (RMSE)**

   RMSE is the root of Mean Squared Error (MSE) which also can assess model performance. It measures the average difference between the predicted and observed values and provides insights into how good the model is. Same as MSE, it also excels in calculating data with normal distribution. RMSE is also sensitive, making it more suitable to check the gradient value of data [36]. However, because of its sensitivity, it also tends to be affected more by outliers, thus we need to ensure our data is free from outliers so that the RMSE value will be more accurate [36].

For unsupervised machine learning, we use silhouette score. The silhouette score is used to assess the quality of data clustering. It ranges from -1 to +1 with values close to 1 means that the data points are well clustered and values close to -1 means that there is a possibility of assigning points to the wrong clusters [38]. It is calculated based on the average distance between observations inside and between clusters.

Lastly, for classification-typed supervised machine learning, we will use three evaluation metrics:

1. **Accuracy**

   Accuracy is the sum of all correct predictions divided by the total number of the dataset [39]. All correct predictions include True Positive (TP) and True Negative (TN) values. The best accuracy is 1.0, while the worst is 0.0.

2. **ROC and AUC**

   The ROC (Receiver Operating Characteristic) curve is a graph that shows the distribution between the True Positive Rate (TPR) and False Positive Rate (FPR) [39]. It calculates the TPR and FPR for each threshold to visualize how each threshold affects the rates. From the ROC curve, the AUC (Area Under Curve) score can also be calculated. AUC value determines how much an area is below the ROC curve [39]. The higher the AUC value, the better the model is.

F. RESULTS AND DISCUSSION

Exploratory Data Analysis (EDA)

We plot the histogram for numerical columns in the merge.csv (the merged dataset) as shown in Figure 1. The numerical columns in the merge.csv consist of TIOBE Index ratings, job amount, average salary, number of people who display the programming language as a LinkedIn skill, GitHub user count, Wikipedia total views, search count (Google Trends), and number of stack overflow users.

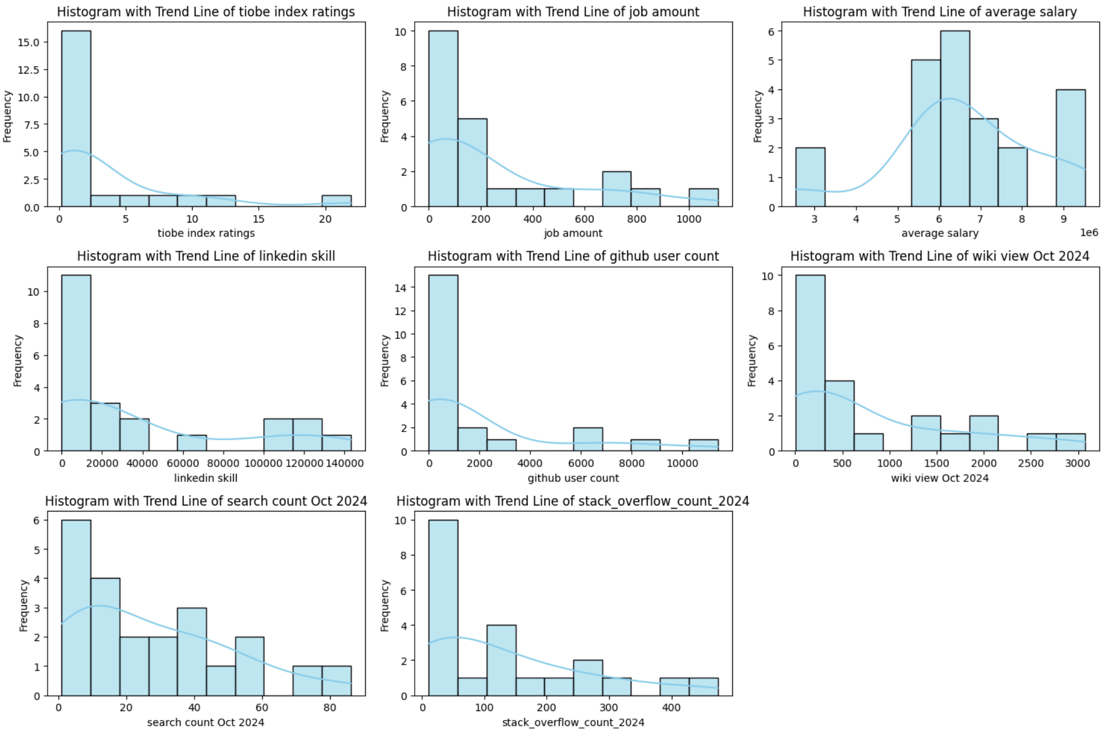


Figure 1: Histogram for Numerical Values in Merged Dataset

All of the histograms in Figure 1 are right-skewed, except for the average salary histogram which is left-skewed, but almost symmetric. We consider all isolated bars to be not outliers, but valid observations because super-popular programming languages like Python often have higher ratings, job amounts, user counts, etc. For example, the isolated bar in the TIOBE Index rating histogram is Python which has a rating above 20.

The histograms proved a big discrepancy between popular and unpopular programming languages regarding all factors, except the average salary. All the popular programming languages tend to become isolated bars or seem to be outliers due to their high numerical values. Meanwhile, unpopular programming languages dominate the lower numerical values.

The average salary histogram is the one that is distinct from the others due to its distribution shape. The reason is that Indonesian job salaries are not that different from those of mastering a certain programming language and another programming language. An unpopular programming language can also have a high average salary because if there is only one job that uses that language, then only the salary of that job will be the average salary of the unpopular programming language. One certain thing is that the average salary for programming or IT jobs in Indonesia is at about 5 to 7 million rupiah.

Next, we will also analyze the correlation of features in the merged programming language dataset by plotting the correlation matrix to observe the Pearson coefficient between each feature. We exclude the languages that do not have any LinkedIn skills, GitHub user count, etc. The result is shown below in Figure 2.

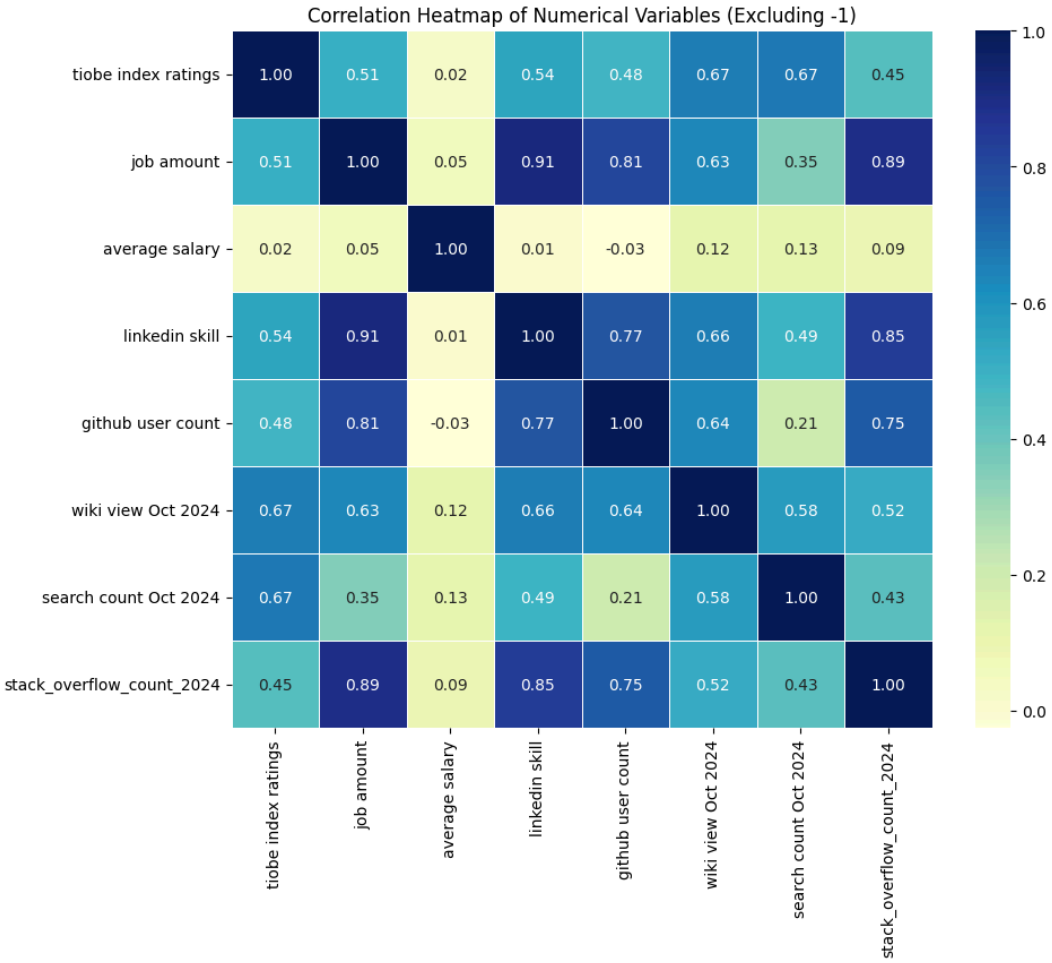


Figure 2: Pearson Correlation Matrix for Numerical Features in Merged Dataset

We define a coefficient greater than 0.7 and lesser than -0.7 as correlated. From all the features in the dataset, there are 6 relationships between features that have Pearson coefficients greater than 0.7, which are

- LinkedIn skill and job amount (0.91)
- GitHub user count and job amount (0.81)
- GitHub user count and LinkedIn skill (0.77)
- Stack Overflow count and job amount (0.89)
- Stack Overflow count and LinkedIn skill (0.85)
- Stack Overflow count and GitHub user count (0.75)

The heatmap shows that the previous 6 relationships have a significant linear correlation between the tested features, with the highest correlation being the relationship between LinkedIn skill and job amount. In other words, the more people that put the programming language as a skill in their LinkedIn, the more job requires that programming language and vice versa.

Upon further inspection, we can see that the relationship that has high correlation coefficient is the permutation of two elements between these features: LinkedIn skill, job amount, GitHub user count, and Stack Overflow user count. All of the relationships between two features from those four features have a high correlation coefficient. This observation implies that the number of people who display the programming language as their skill in LinkedIn, job amount, GitHub user count, and Stack Overflow user count are all associated with each other. If one of them increases, then the others also increase. Conversely, the decrease in one of the four factors will highly likely make the other three values also decrease.

To better examine those 6 relations with the highest Pearson coefficient, we plot a scatter plot for each of the relations as shown below in Figure 3.

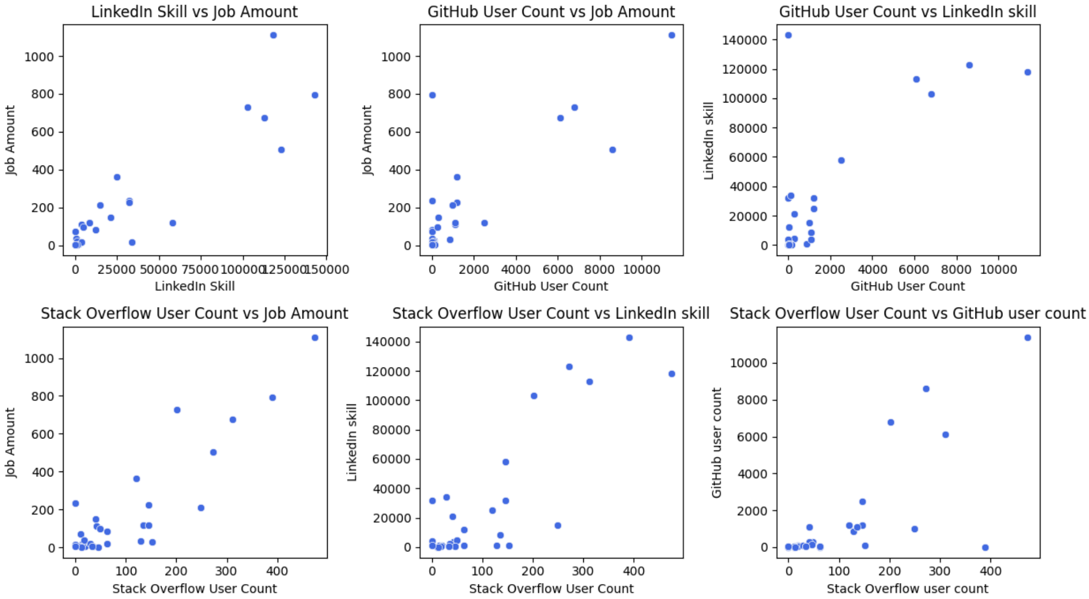


Figure 3: Scatter Plots for Relationship with Pearson Coefficient Greater than 0.7

Based on the scatter plots in Figure 3, all of the relations are not that linear as there are some data points that are far from the other data points that are similar for one of the features. For example, in the GitHub user count versus job amount plot, we can see that even though there is a language that has low GitHub user count, the job amount of that language is still considerably high.

Another similarity that we observed from the scatter plots is that the majority of the data points are in the lower value area. This observation further proved the discrepancy between unpopular and popular programming language as discussed before in the histogram for numerical values.

Furthermore, the LinkedIn skill versus job amount scatter plot is unique as there are no middle value data points, so there is a vast gap between the lower value data points with the higher value ones. Therefore, we can infer that based on the LinkedIn skill user amount and job amount, the data points can be categorized into the low and high value clusters. High value clusters will be the popular programming languages, while the low value clusters will consist of unpopular programming languages.

Scatter plots that plot the relation between GitHub user count with other factors (GitHub user count versus job amount, GitHub user count versus LinkedIn skill, and GitHub user count versus Stack Overflow user count) have outliers that are significantly different from the other data point. In GitHub user count versus job amount, there is a data point that has low GitHub user count but high job amount. In GitHub user count versus LinkedIn skill, there is a data point that has low GitHub user count, but high number of people displaying that language in LinkedIn. In GitHub user count versus Stack Overflow user count, there are several data points that have low GitHub user count, but high Stack Overflow user count. These anomalies may be the reason why the 3 relationships between GitHub user count with other factors are the lowest among the 6 relationships that have a Pearson coefficient greater than 0.7.

Meanwhile, for Stack Overflow user count versus job amount and Stack Overflow user count versus LinkedIn skill scatter plots, the linear relations are more visible. There exists low, middle, and high value data points. However, same as other scatter plots, the lower value dominates the amount of data points in the scatter plots.

Next, we also gathered insights regarding the programming languages' job market in Indonesia. Our research found that the most popular programming language in Indonesia's job market is JavaScript followed by SQL and Java. We also investigate each programming language's job type distribution as seen in Figure 4. As we can see from Figure 4, most programming languages have various job types distributions with the developer type of job remaining as the most widely sought job. Since there are many types of developers of each programming language, thus we classify them under the label of "other developer".

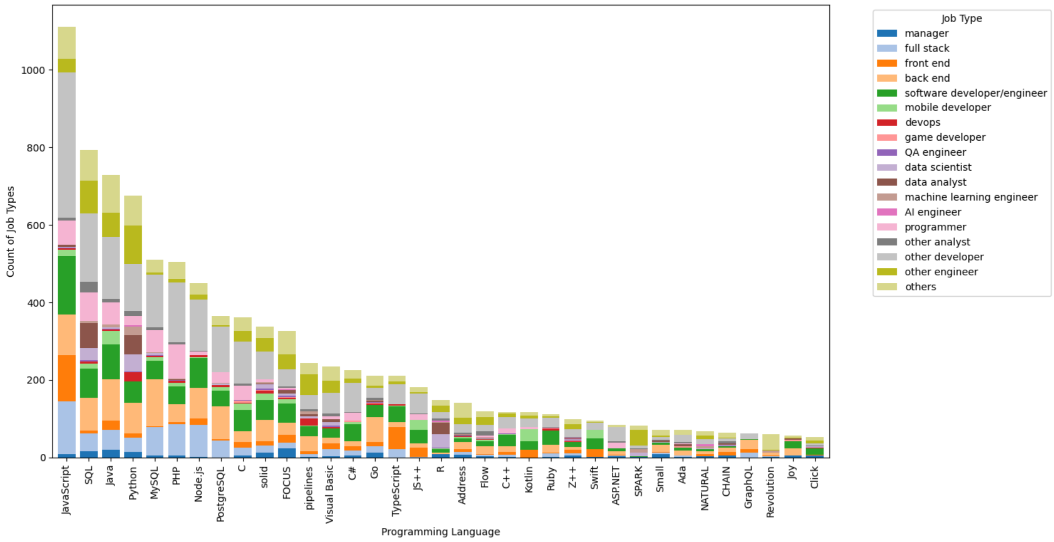


Figure 4: Job Type by Programming Language

We also investigated the job location of programming languages in Indonesia. As shown in Figure 5, most programming language job locations are in Jakarta followed by Banten and West Java. This result indicates that the programming job is more popular and sought on Java Island than on other big islands, such as Sumatra, Kalimantan, Sulawesi, or Papua, which can be interpreted that finding a programming job in Java, especially in Jakarta, is easier than the other places.

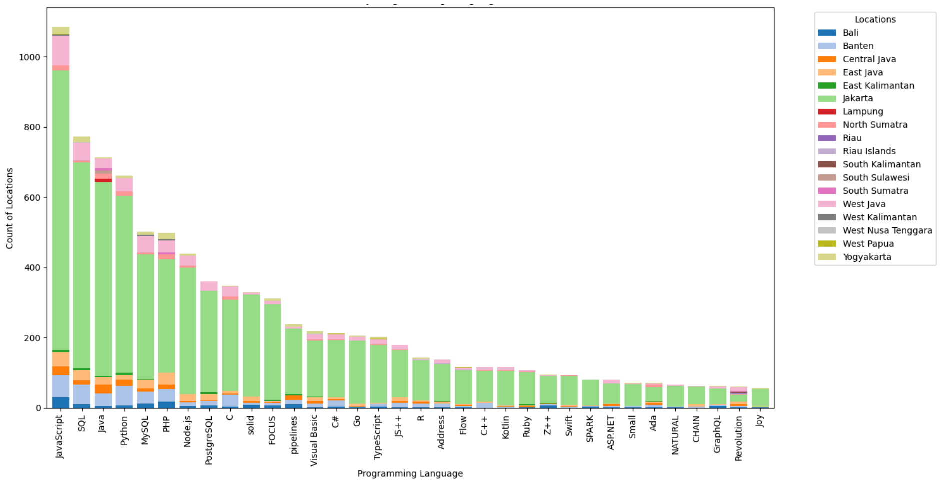


Figure 5: Location Count by Programming Language

Unsupervised Machine Learning (Clustering)

After conducting our exploratory data analysis, we will perform the unsupervised clustering algorithm in the merge dataset to determine the clusters of the languages. Based on our exploratory data analysis, we can see that there is a huge difference between unpopular and popular programming languages, so we will use the number of clusters as 2 for algorithms that require us to input the number of clusters first. With unsupervised clustering, we aimed to learn about the spread of unpopular and popular languages in Indonesia and determine which languages are considered the most popular ones based on the features in the merged dataset.

We performed 7 different clustering algorithms for our merged dataset and tested their accuracies with silhouette score. The silhouette score for each algorithm can be seen below.

Table 1: Silhouette Score for Unsupervised Algorithms

Algorithm Name	Silhouette Score
K-Means	0.754
Agglomerative Clustering	0.712
DBSCAN	0.773
GMM	0.307
SOM	0.191
OPTICS	0.207
BIRCH	0.789

We define a great silhouette score as a score greater than 0.7. Algorithms that have lower scores will not be used for our further analysis. From the table above, the four algorithms that have high scores are K-Means, BIRCH, DBSCAN, and Agglomerative clustering. All of them have similar patterns. All of them group the programming language into two clusters, the majority (unpopular) and minority (popular) clusters. Popular cluster data points have exceptionally high job amount, user count, etc. For visualization purposes, we develop the graph for each of the 7 clustering algorithms in Figure 6.

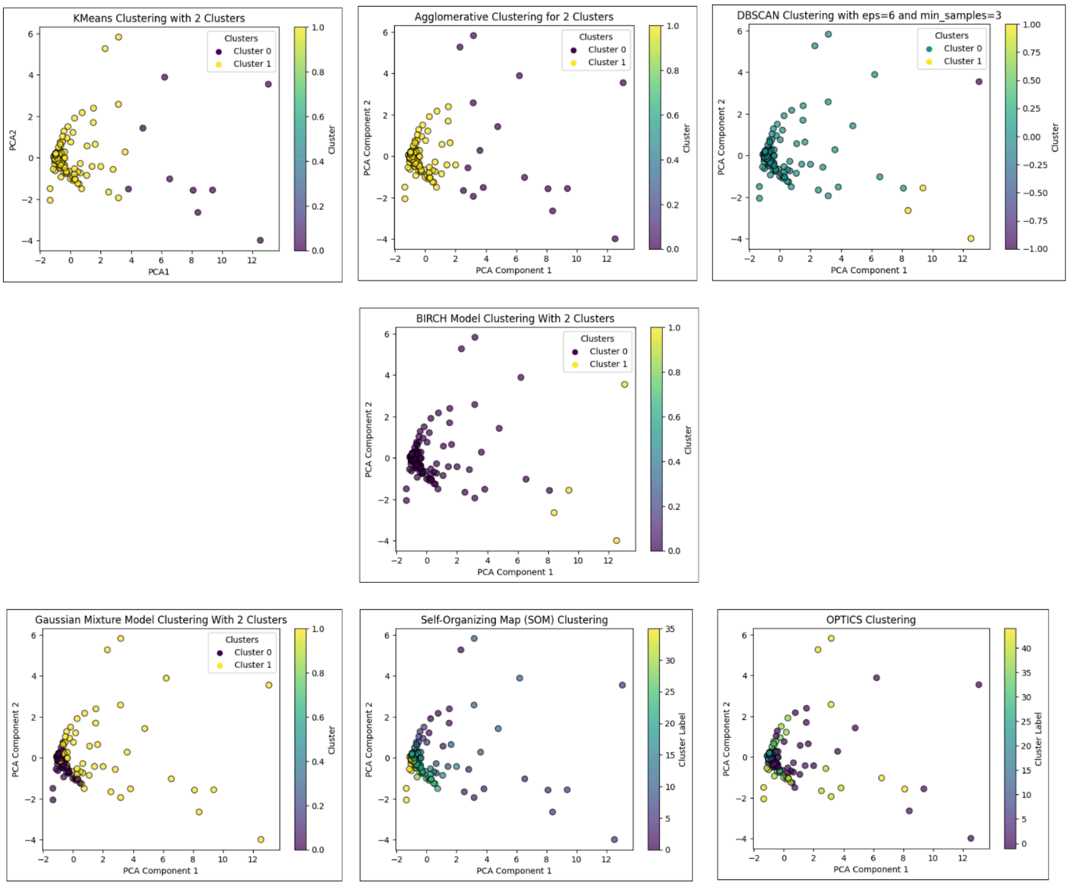


Figure 6: Visualization of Clusters from Each Unsupervised Algorithm

We can see that the first 4 algorithms are similar to each other, where they group the majority of the data points that are located on the left to cluster 0, while the points on the far right are categorized as cluster 1. Cluster 0 is unpopular programming languages, while cluster 1 is popular programming languages. The other three algorithms' performance is not that great. Although GMM partitions the data points into 2 clusters, the distance between cluster 0 and cluster 1 is not that visible. In other words, GMM failed to categorize them into meaningful clusters. Meanwhile, SOM and OPTICS failed to cluster the data points as they overfit with the points, so they created too many clusters for the data.

Next, we will list the names of programming languages that are 'popular'. We define a language as popular if there is more than one algorithm from the 4 best algorithms that categorizes it in the popular (minority) cluster. The result reveals that 4 algorithms categorize Java, Javascript, and PHP as popular, 3 algorithms (K-Means, Agglomerative Clustering, and BIRCH) categorize Python as popular, and 2 algorithms (K-Means and Agglomerative Clustering) categorize

C, C++, and SQL as popular. This result demonstrates that Java, Javascript, and PHP are the most popular programming languages in Indonesia as of October 2024. Meanwhile, other programming languages, such as Python, C, C++, and SQL are considered to be highly likely popular in Indonesia as of October 2024.

One interesting observation is for Python in the DBSCAN clustering, where Python is considered as outlier or cluster -1 based on the algorithm. This means that Python data point is far from the high-density data points, so the algorithm can not assign Python to any other clusters. This result demonstrates that Python might be considered as a popular language, just like Java, Javascript, and PHP, but its score is too high, so is considered as a noise point instead of in cluster 1 (popular cluster).

The reason why Java is popular in Indonesia is likely rooted in its global acclaim, primarily due to its "Write Once, Run Anywhere" capability. Many legacy systems in Indonesia continue to rely on Java despite the emergence of new languages. Besides that, Java has been a dominant language for mobile development with its "Write Once, Run Anywhere" slogan [40]. As one of the largest markets for mobile devices, Indonesia developers tend to utilize Java to meet the demand of mobile application development.

On the other hand, Javascript and PHP are popular because of the rapid growth of website applications in Indonesia. Over the past few years, websites, especially e-commerce sites have accelerated growth in Indonesia [41]. The rise in website applications enables people to prioritize learning languages that are beneficial for frontend and backend website development for better job opportunities. Javascript's ability in building dynamic web interfaces and PHP's capability for server-side development make these languages popular as they are indispensable for creating websites.

Meanwhile, Python is popular most likely because of its simplicity, high readability syntaxes which resembles the English language, and its vast library [42]. Python has also been utilized not only in software development, but in other IT jobs, like data science, machine learning, and Artificial Intelligence. Hence, it is reasonable that it has a high popularity among IT people due to its versatility.

SQL's prominence in Indonesia most likely due to its role as the universal core language for relational databases (PostgresQL, MySQL, etc.). Databases have been a cornerstone for data architecture, playing a big role in data-driven decision making [43]. SQL is in high demand due to its integration with other technology to provide the database feature for a certain program. Besides developers, jobs like Business Intelligence and Data Engineer also use SQL for their work, highlighting the popularity and indispensability of SQL across job sectors.

Finally, we also have the classic languages, which are C and C++. The popularity of C and C++ can be traced to its foundational role. C is considered the mother of all programming languages, so many people are learning it as their first programming language. C++ is also highly similar to C, so many people also started learning programming with C++. Based on [44], many universities have enforced C as a compulsory language for their students to learn programming. Besides their role as a basis in programming education, C and C++ are also widely used in IoT manufacturing, game development, and operating system maintenance, where efficiency is crucial. Additionally, many legacy systems still utilize C and C++ because of their efficacy. Competitive programmers also favor C++ as their main language due to its high performance. Therefore, C and C++ still remain as popular and important languages in Indonesia.

Time Series Analysis

We also used machine learning to predict the global popularity rankings of programming languages, Wikipedia search results, Google search, and Stack Overflow results. We only included the seven popular programming languages that are obtained from our unsupervised machine learning result. The result of the time series model training for Python TIOBE ratings for each model is shown in Figure 7, Figure 8, Figure 9, and Figure 10.

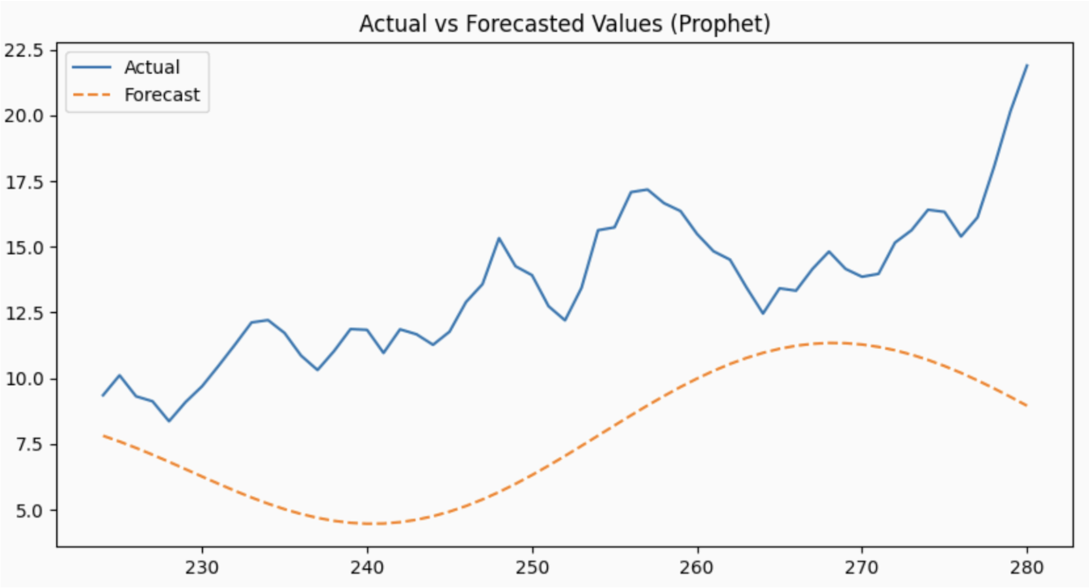


Figure 7: Prophet Model Result

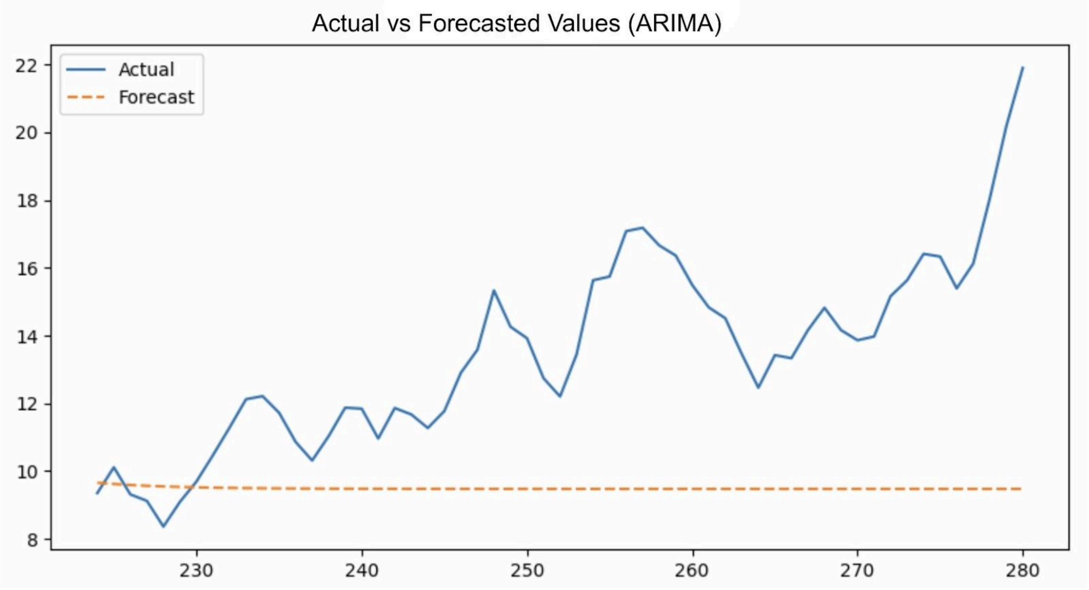


Figure 8: ARIMA Model Result

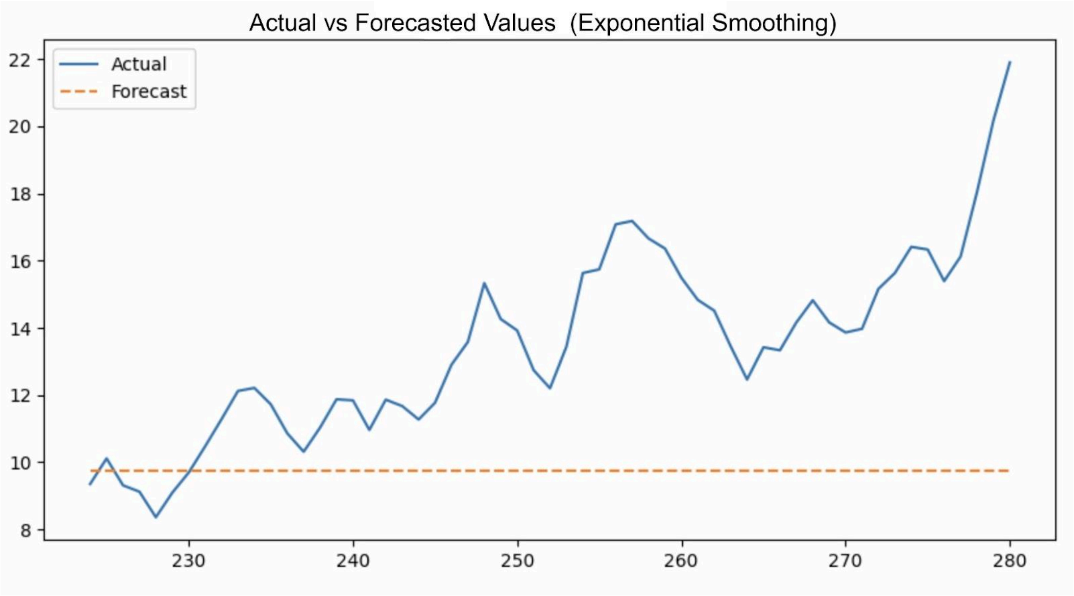


Figure 9: Exponential Smoothing Model Result

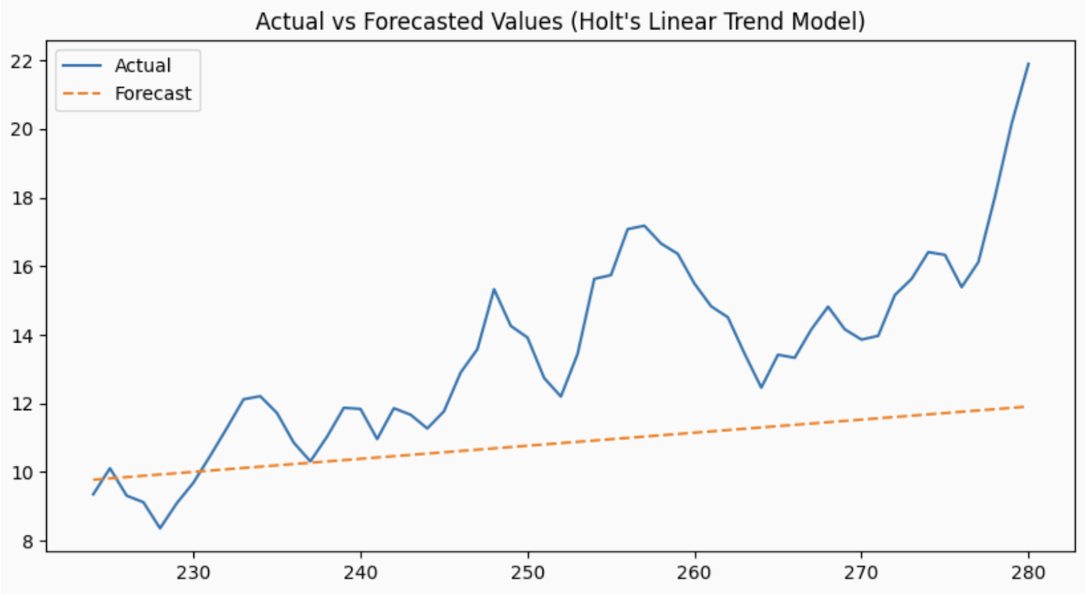


Figure 10: Holt's Linear Trend Model Results

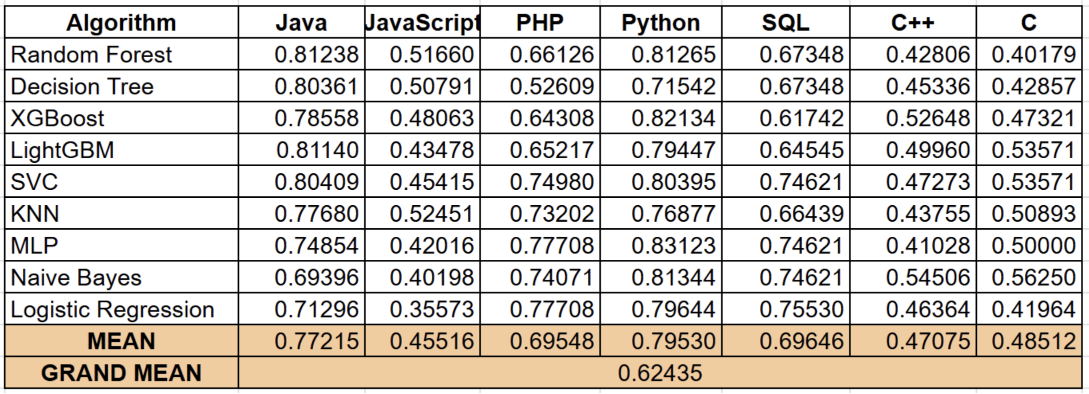
From Figure 7, Figure 8, Figure 9, and Figure 10, we can see that all models are underfitting for the data, which indicates inaccuracy in all models. This result further shows that time-series analysis is not possible for our data, which could happen because the data is not sufficient. Thus, we decided not to progress further with the time series analysis.

Supervised Machine Learning (Classification)

Since our attempt to predict with time series forecasting did not work, we decided to predict using the classification algorithm instead. We used the Google Trends, Wikipedia Search, and TIOBE index data to perform the algorithm. We classified the numbers into two classes (high and low) before the training started.

Our main topic is to predict the trends of programming language, thus we first use the TIOBE index as our label, and Wikipedia search and Google Trends as our features. Table 2 shows all model performances across the programming languages.

Table 2: Classification Model Results Using TIOBE Index Label



Algorithm	Java	JavaScript	PHP	Python	SQL	C++	C
Random Forest	0.81238	0.51660	0.66126	0.81265	0.67348	0.42806	0.40179
Decision Tree	0.80361	0.50791	0.52609	0.71542	0.67348	0.45336	0.42857
XGBoost	0.78558	0.48063	0.64308	0.82134	0.61742	0.52648	0.47321
LightGBM	0.81140	0.43478	0.65217	0.79447	0.64545	0.49960	0.53571
SVC	0.80409	0.45415	0.74980	0.80395	0.74621	0.47273	0.53571
KNN	0.77680	0.52451	0.73202	0.76877	0.66439	0.43755	0.50893
MLP	0.74854	0.42016	0.77708	0.83123	0.74621	0.41028	0.50000
Naive Bayes	0.69396	0.40198	0.74071	0.81344	0.74621	0.54506	0.56250
Logistic Regression	0.71296	0.35573	0.77708	0.79644	0.75530	0.46364	0.41964
MEAN	0.77215	0.45516	0.69548	0.79530	0.69646	0.47075	0.48512
GRAND MEAN	0.62435						

From Table 2, we can see that only Java and Python have good accuracy. On the other hand, PHP and SQL have only decent accuracy, and JavaScript, C++, and C have poor accuracy. This shows that only Java's and Python's data have a pattern that can be predicted by the machine.

Since JavaScript, C++, and C model accuracy is poor, we will not explore them further. For the other four programming languages, we plotted their ROC curve to ensure the good performance of the model. The ROC curve and their AUC values can be seen in Figure 11, Figure 12, Figure 13, and Figure 14. As we can see, their ROC curve is quite good, with quite high AUC values as well. This indicates that the model is good enough, thus we can explore the previous accuracy data more.

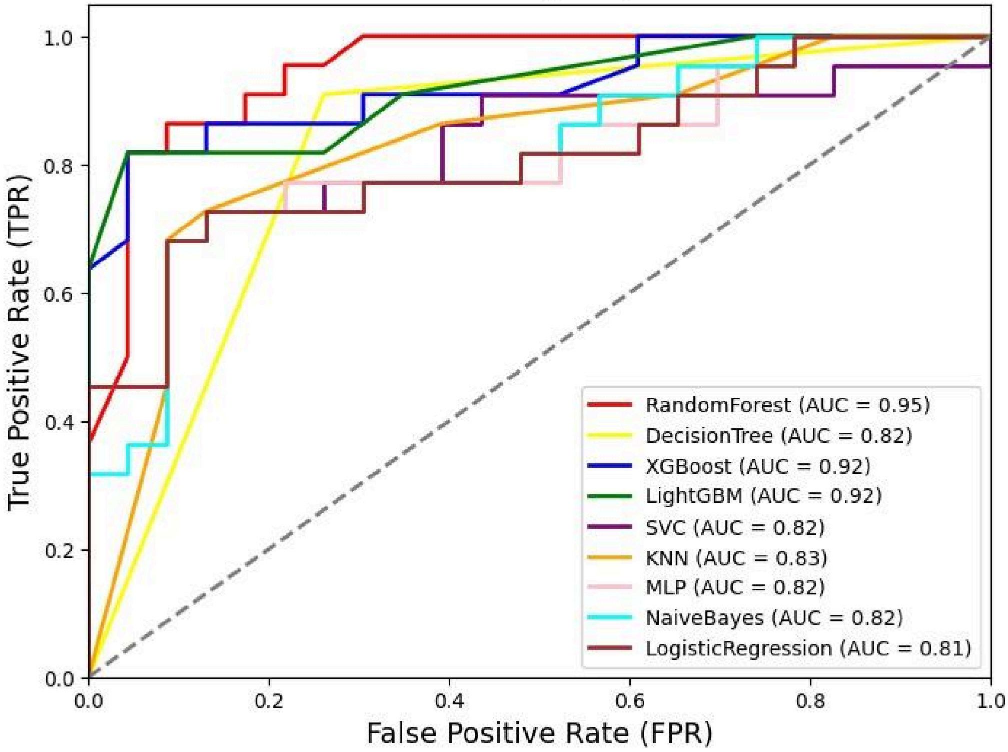


Figure 11: Java ROC Curve with TIOBE Index Label

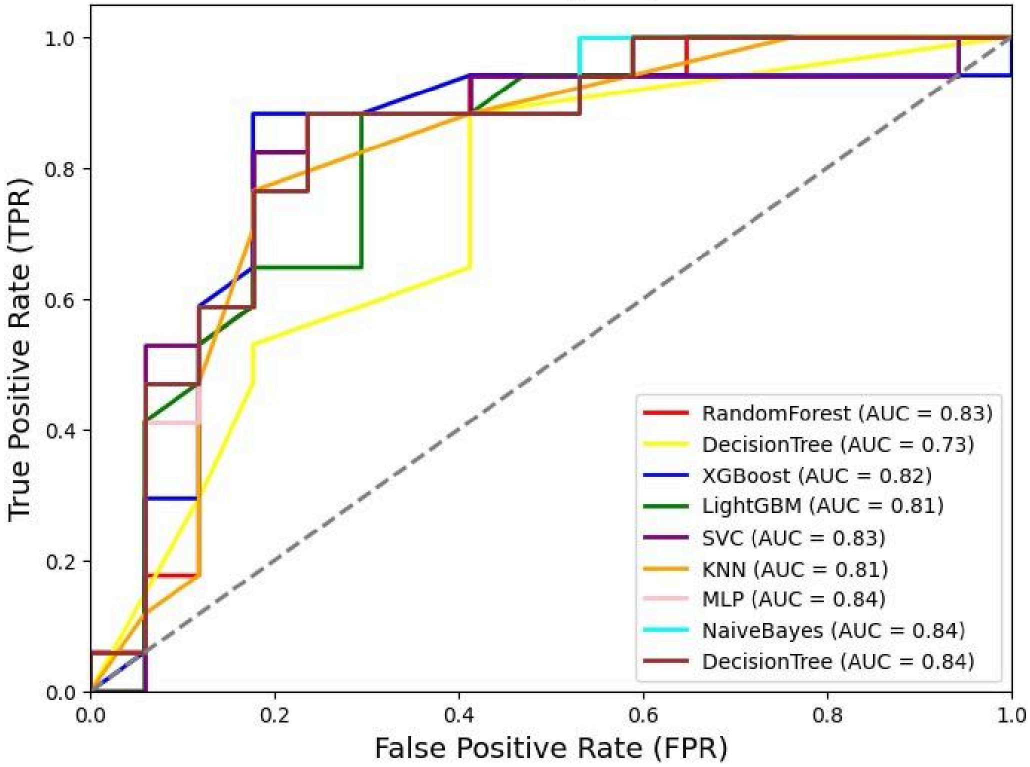


Figure 12: PHP ROC Curve with TIOBE Index Label

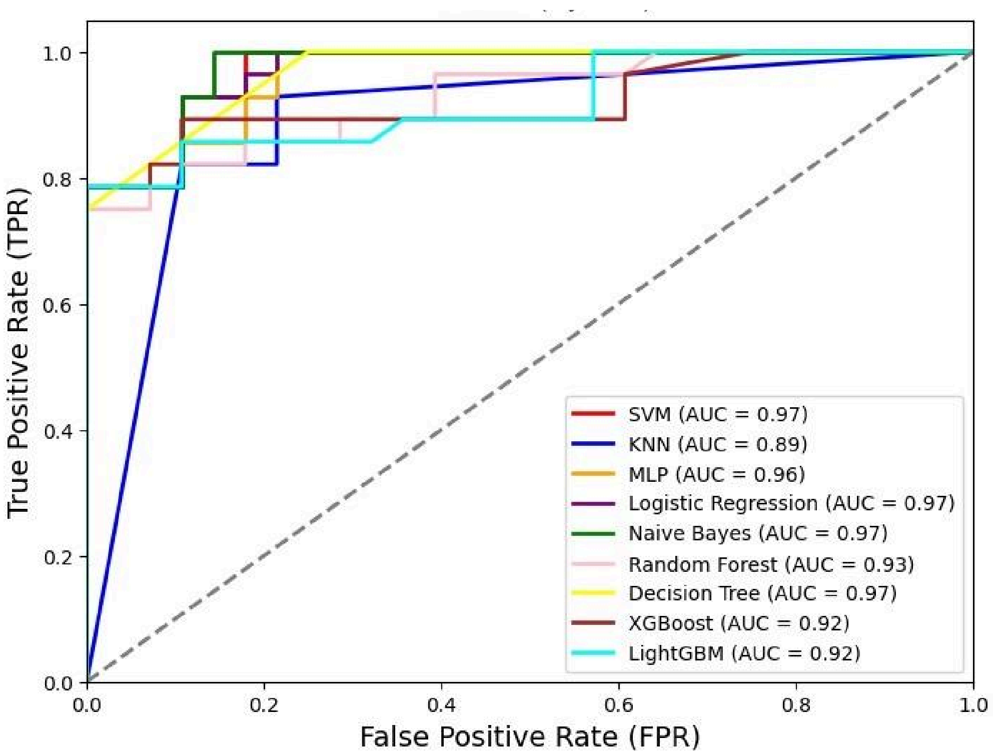


Figure 13: Python ROC Curve with TIOBE Index Label

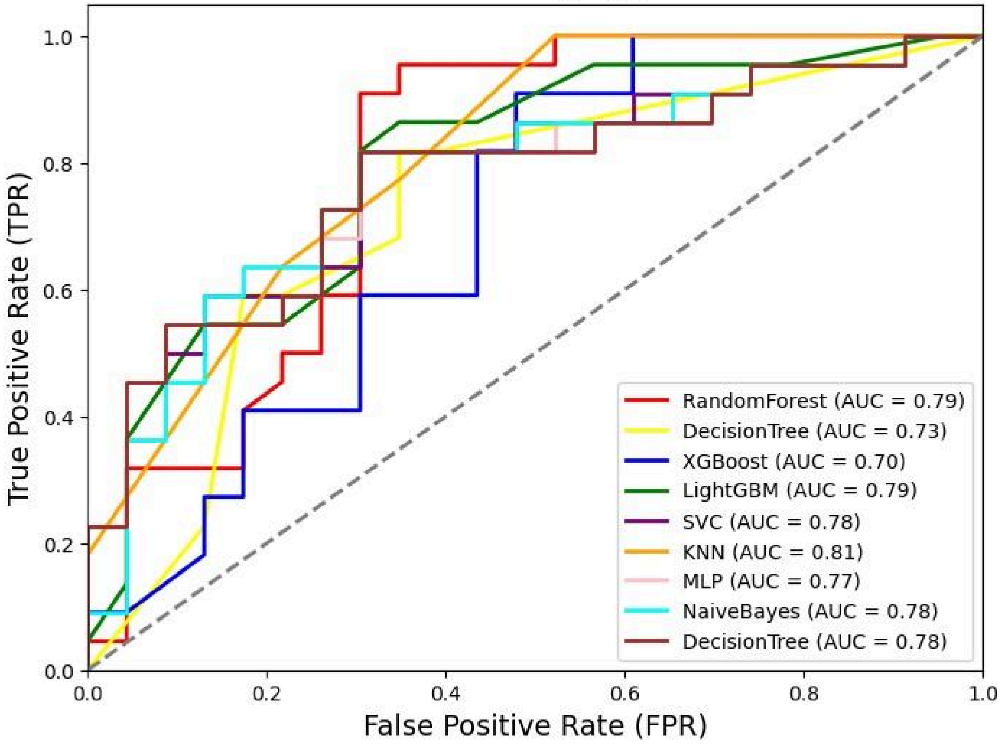
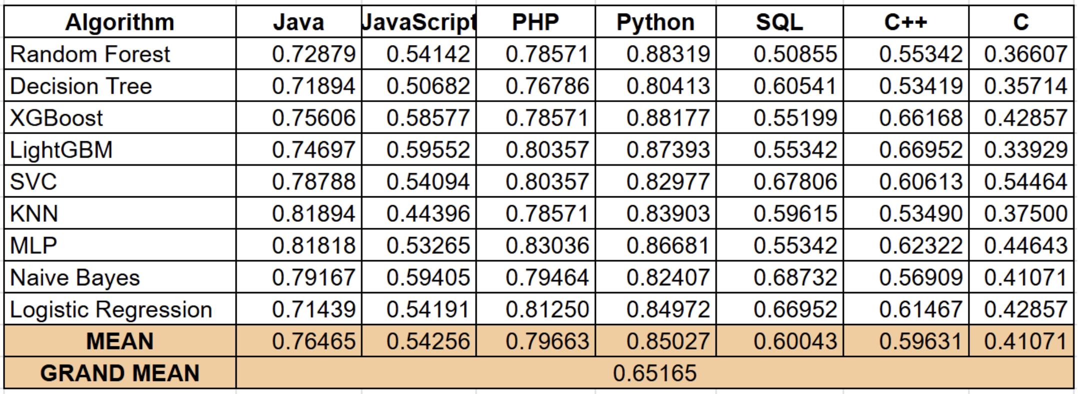


Figure 14: SQL ROC Curve with TIOBE Index Label

We will first discuss Java models' results. Java accuracy values are second highest after Python, with Random Forest being the most accurate model and Naive Bayes being the least. This indicates that Java has complex data patterns that were able to be captured well by the Random Forest model. We then move to PHP, which has MLP as the most accurate model. MLP's high performance may be due to its ability to learn complex and nonlinear patterns. This result shows that PHP has intricate data patterns, thus preferring a more complex model. Next, we have Python, which has the same models as PHP for its most accurate model. This result indicates that Python also has complex data patterns, which are more predictable by complex models such as MLP. Lastly, SQL data patterns are more unique since they favor the simple Logistic Regression model.

Another interesting fact we found is that Google Trends serves as a better label than the TIOBE index. The details of the Google Trends' result is shown in Table 3.

Table 3: Classification Model Results Using Google Trends Label



Algorithm	Java	JavaScript	PHP	Python	SQL	C++	C
Random Forest	0.72879	0.54142	0.78571	0.88319	0.50855	0.55342	0.36607
Decision Tree	0.71894	0.50682	0.76786	0.80413	0.60541	0.53419	0.35714
XGBoost	0.75606	0.58577	0.78571	0.88177	0.55199	0.66168	0.42857
LightGBM	0.74697	0.59552	0.80357	0.87393	0.55342	0.66952	0.33929
SVC	0.78788	0.54094	0.80357	0.82977	0.67806	0.60613	0.54464
KNN	0.81894	0.44396	0.78571	0.83903	0.59615	0.53490	0.37500
MLP	0.81818	0.53265	0.83036	0.86681	0.55342	0.62322	0.44643
Naive Bayes	0.79167	0.59405	0.79464	0.82407	0.68732	0.56909	0.41071
Logistic Regression	0.71439	0.54191	0.81250	0.84972	0.66952	0.61467	0.42857
MEAN	0.76465	0.54256	0.79663	0.85027	0.60043	0.59631	0.41071
GRAND MEAN	0.65165						

As we can see in Table 2 and Table 3, the grand mean when using Google Trends label is higher than when using the TIOBE Index label. This indicates that Google Trends data is more predictable using Wikipedia and the TIOBE index. However, although the grand mean accuracy increases, the models' accuracy in some programming languages decreases, such as in Java, SQL, and C. This is possible since these programming languages' data are more compatible with predicting the TIOBE Index. In other words, the Google Trends data for these languages may not represent as much trend value as the TIOBE index.

Next, we will explore the languages that have an average accuracy score of more than 0.75 when using the Google Trends label. We can see that only Java, PHP, and Python fulfill this condition, with PHP showing the most profound changes, from 0.695 in the TIOBE index label to 0.797 in this label.

We also plotted the ROC curve of Java, Python, and PHP for the Google Trends label. The result can be seen in Figure 15, Figure 16, and Figure 17. Since the ROC curve and AUC values are quite high, we can say that the model is good enough, thus we can explore the model more.

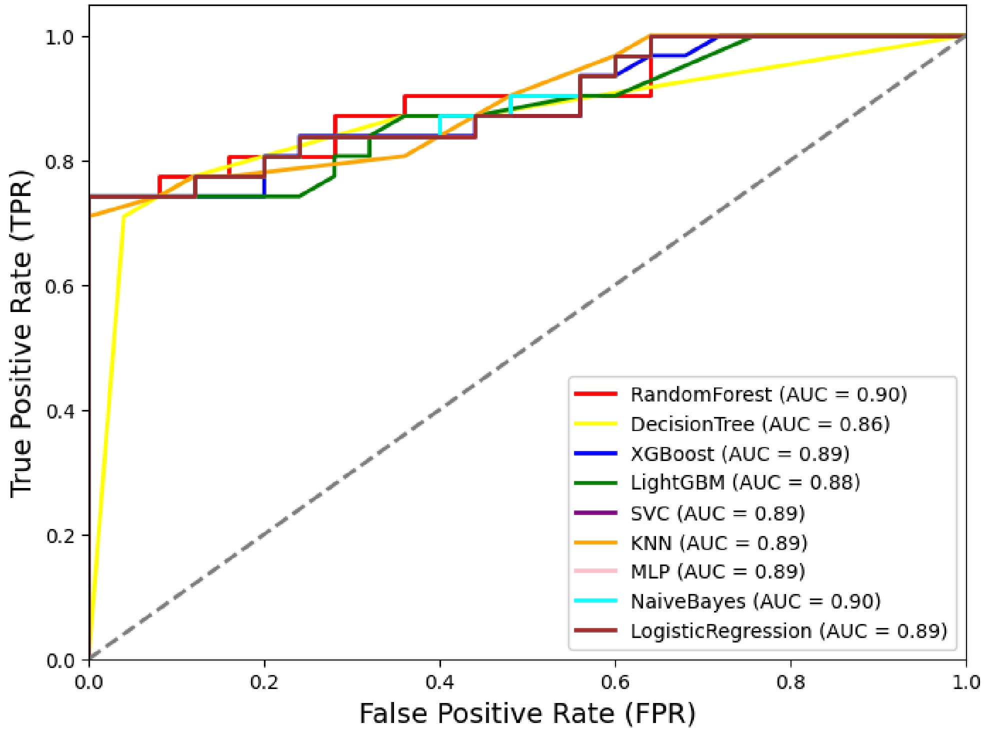


Figure 15: Java ROC Curve with Google Trends Label

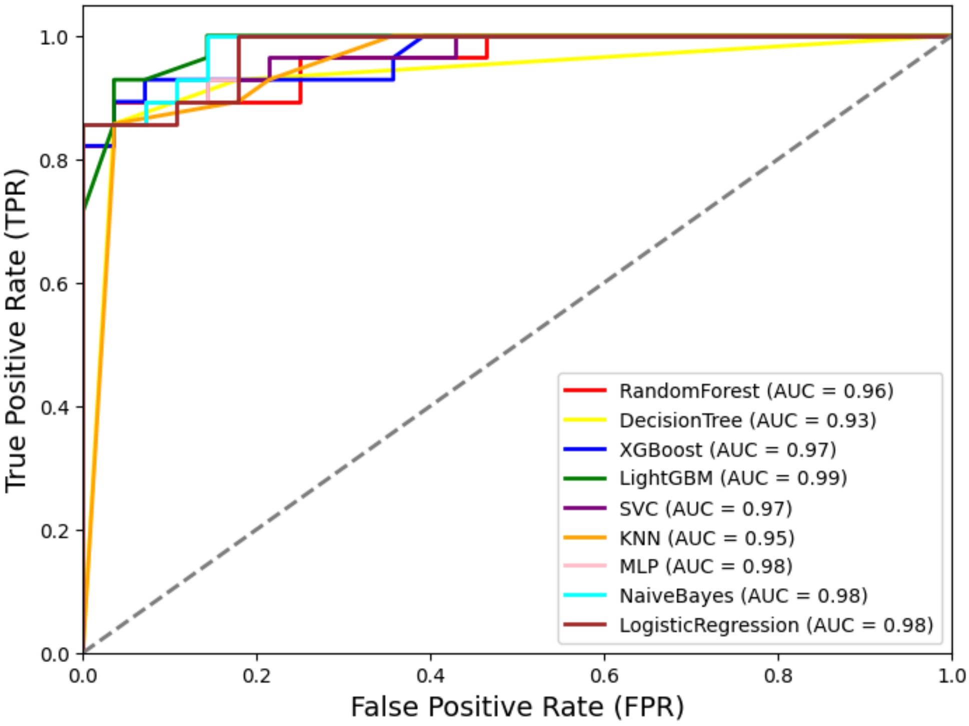


Figure 16: Python ROC Curve with Google Trends Label

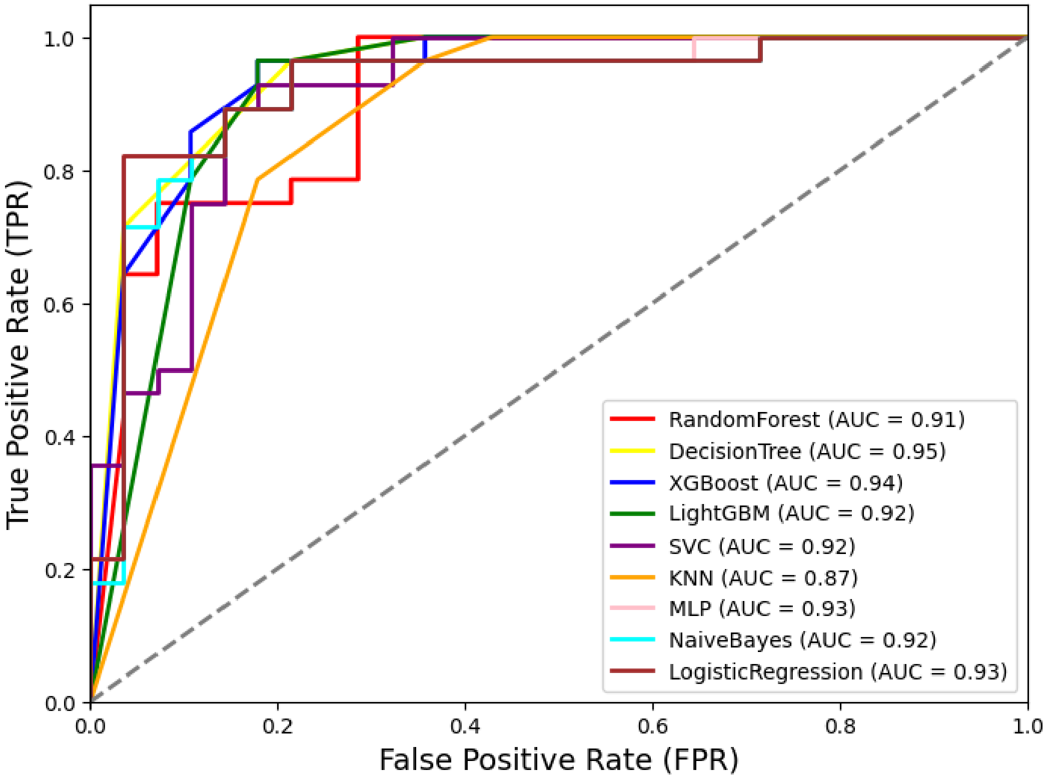


Figure 17: PHP ROC Curve with Google Trends Label

In Java, the best model is KNN, while the least performing is Logistic Regression. This result shows that the Java Google Trends has local clusters that enable the KNN algorithm to search for localized patterns effectively. On the other hand, it may lack linearity, which hinders the Logistic Regression model since this model assumes linearity of all independent variables. Meanwhile, PHP and Python have the same best and least-performing models, which are MLP as the best and Decision Tree as the least. This result is the same when we use the TIOBE index as a label, indicating that Python's and PHP's TIOBE index and Google Trends data have intricate patterns that prefer complex models like MLP.

G. CONCLUSION AND RECOMMENDATION

To conclude, this study analyzes the interrelationship between programming language features in Indonesia to predict the trends and popularity of programming languages. We utilized unsupervised machine learning to cluster the programming languages into popular and unpopular ones. After using unsupervised machine learning, we use classification-type supervised machine learning to predict whether the programming language popularity will be low or high based on Wikipedia views and Google Trends scores. Before the machine learning process, we analyzed the correlation heatmap between several features in the dataset. The result reveals that several aspects are highly correlated.

After reviewing the statistics, we perform a clustering analysis. The result reveals seven programming languages were listed as popular in Indonesia. These languages include Java, JavaScript, PHP, Python, SQL, C++, and C. This result demonstrates that Java, Javascript, and PHP are the most popular programming languages in Indonesia as of October 2024.

After the clustering analysis, we trained each popular programming language with nine models to predict the popularity of the programming language. The results show that only Java and Python models have good accuracy. Meanwhile, the others have poor model accuracies, indicating that the features are not sufficiently correlated with the label, which hinders the models from predicting the correct test set.

Another interesting fact is that the Google Trends score serves as a better label than the TIOBE index. This result further shows that the TIOBE index and Wikipedia views can predict the Google Trends class more effectively than when we use Google Trends and Wikipedia views to predict the TIOBE index. However, only Java, PHP, and Python have a good accuracy score.

Looking ahead, future studies should gather more data and information to increase these predictive model accuracies. Furthermore, fine-tuning hyperparameters and incorporating additional features may also uncover unique characteristics of programming languages in Indonesia. If the data are sufficient, integrating time-series forecasting could also be a nice approach to enhance the predictive power of supervised models.

H. LINK TO SOURCE CODE

https://github.com/Ella-Raputri/FoDS-FinalProject

I. QUESTION AND ANSWER

There is no question from the lecturer. We are only asked to summarize our result to be shorter in the paper (by Miss Nurul) and to explain the step by step of the Chi-test in the Linear Algebra final report (by Sir Raymond).

REFERENCES

[1] D. Tošić, “Role of programming languages in digitalization,” in *Rev. NCD*, 2024, pp. 28–37.

[2] M. Shaw, “Myths and mythconceptions: What does it mean to be a programming language, anyhow?,” in *Proc. ACM Program. Lang.*, vol. 4, no. HOPL, pp. 1–44, Jun. 2020, doi: 10.1145/3480947.

[3] S. Pramana, “Peningkatan literasi data menuju Indonesia 4.0,” in *Empower. Comm.*, 2020.

[4] U. V. Wardina, N. Jalinus, and L. Asnur, “Kurikulum pendidikan vokasi pada era revolusi industri 4.0,” (in Indonesian), *J. Pend.*, vol. 20, no. 1, pp. 82–90, Mar. 2019, doi: 10.33830/jp.v20i1.240.2019.

[5] T. Indriyani, I. Arfyanti, M. Farkhan, I. N. A. Arsana, Joosten, and Sepriano, “Pengantar Bahasa Pemrograman Populer,” in *Bahasa Pemrograman Populer*, Jambi: Sonpedia, 2024.

[6] Dr. M. Raghavender Sharma, “A short communication on computer programming languages in modern era,” *Int. J. Comput. Sci. Mob. Comput.*, vol. 9, no. 9, pp. 50–60, Sep. 2020, doi: 10.47760/IJCSMC.2020.v09i09.006.

[7] T. F. Bissyande, F. Thung, D. Lo, L. Jiang, and L. Reveillere, “Popularity, interoperability, and impact of programming languages in 100,000 open source projects,” in *IEEE COMPSAC 2013*, IEEE, Jul. 2013, pp. 303–312. doi: 10.1109/COMPSAC.2013.55.

[8] P. Dymora and A. Paszkiewicz, “Performance analysis of selected programming languages in the context of supporting decision-making processes for industry 4.0,” *Appl. Sci.*, vol. 10, no. 23, p. 8521, Nov. 2020, doi: 10.3390/app10238521.

[9] F. Demir, “The effect of different usage of the educational programming language in programming education on the programming anxiety and achievement,” *Educ. Inf. Technol.*, vol. 27, no. 3, pp. 4171–4194, Apr. 2022, doi: 10.1007/s10639-021-10750-6.

[10] G. S. Laxminarayana, G. S. Satyanarana, S. A. Shaikh, and A. M. S. Ansari, “The impact of programming language on development productivity: An empirical study,” *IRJMETS*, Aug. 2024, doi: 10.56726/IRJMETS60132.

[11] A. Peslak and M. Conforti, “Computer programming languages in 2020: What we use, who uses them, and how do they impact job satisfaction,” *IIS*, 2020, doi: 10.48009/2_iis_2020_259-269.

[12] E. Widodo, R. Prathivi, and S. Hadi, “Evaluating the popularity of programming languages in Indonesia using the MABAC method,” *J. Transform.*, vol. 21, no. 1, Aug. 2023, doi: 10.26623/transformatika.v21i2.7001.

[13] R. H. Shumway and D. S. Stoffer, “ARIMA Models,” 2017, pp. 75–163. doi: 10.1007/978-3-319-52452-8_3.

[14] E. Ostertagová and O. Ostertag, “Forecasting using simple exponential smoothing method,” *Acta electrotech.*, vol. 12, no. 3, Jan. 2012, doi: 10.2478/v10198-012-0034-2.

[15] E. P. Hendri and S. Fadhlia, “Times series data analysis: The Holt-Winters model for rainfall prediction in West Java,” *App. Sci. Def.*, vol. 2, no. 1, pp. 1–8, Mar. 2024, doi: 10.58524/app.sci.def.v2i1.325.

[16] H. Gnanasekaran, D. P., and U. Köse, “Time-series forecasting of web traffic using Prophet machine learning model,” *FTSCL*, vol. 1, no. 3, pp. 161-177, 2023.

[17] Y. P. Raykov, A. Boukouvalas, F. Baig, and M. A. Little, “What to do when K-Means clustering fails: A simple yet principled alternative algorithm,” *PLOS ONE*, vol. 11, no. 9, p. e0162259, Sep. 2016, doi: 10.1371/journal.pone.0162259.

[18] M. Hahsler, M. Piekenbrock, and D. Doran, “DBSCAN: Fast density-based clustering with R,” *JSS*, vol. 91, no. 1, 2019, doi: 10.18637/jss.v091.i01.

[19] J. Liu, D. Cai, and X. He, “Gaussian mixture model with local consistency,” *AAAI*, vol. 24, no. 1, pp. 512–517, Jul. 2010, doi: 10.1609/aaai.v24i1.7659.

[20] S. S. Lodhi, N. Kumar, and P. K. Pandey, “Autonomous vehicular overtaking maneuver: A survey and taxonomy,” *Veh. Commun.*, vol. 42, p. 100623, Aug. 2023, doi: 10.1016/j.vehcom.2023.100623.

[21] L. R. Iyohu, I. Djakaria, and L. O. Nashar, “Perbandingan metode K-Means clustering dengan self-organizing maps (SOM) untuk pengelompokan provinsi di Indonesia berdasarkan data potensi desa,” (in Indonesian), *J. Stat. App.*, vol. 7, no. 2, Dec. 2023.

[22] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “OPTICS,” in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, New York, NY, USA: ACM, Jun. 1999, pp. 49–60. doi: 10.1145/304182.304187.

[23] B. Munnuru, T. Aditya, and T. A. Srinivas, “From roots to leaves: Understanding birch clustering in ml,” vol. 7, pp. 1–6, 12 2023.

[24] I. T. Jolliffe and J. Cadima, “Principal component analysis: A review and recent developments,” *Philos. Transact. Math. Phys. Eng. Sci.*, vol. 374, no. 2065, p. 20150202, Apr. 2016, doi: 10.1098/rsta.2015.0202.

[25] B. Charbuty and A. Abdulazeez, “Classification based on decision tree algorithm for machine learning,” *JASTT*, vol. 2, no. 01, pp. 20–28, Mar. 2021, doi: 10.38094/jastt20165.

[26] S. J. Rigatti, “Random forest,” *J. Insur. Med.*, vol. 47, no. 1, pp. 31–39, Jan. 2017, doi: 10.17849/insm-47-01-31-39.1.

[27] W. Li, Y. Yin, X. Quan, and H. Zhang, “Gene expression value prediction based on XGBoost algorithm,” *Front. Genet.*, vol. 10, Nov. 2019, doi: 10.3389/fgene.2019.01077.

[28] G. Ke *et al.*, “LightGBM: A highly efficient gradient boosting decision tree,” in *31st Conf. Neural Inf. Process. Syst. (NIPS 2017)*, Long Beach, CA, USA, 2017.

[29] T. Liu, L. Jin, C. Zhong, and F. Xue, “Study of thermal sensation prediction model based on support vector classification (SVC) algorithm with data preprocessing,” *J. Build. Eng.*, vol. 48, p. 103919, May 2022, doi: 10.1016/j.jobe.2021.103919.

[30] S. Zhang, D. Cheng, Z. Deng, M. Zong, and X. Deng, “A novel kNN algorithm with data-driven k parameter computation,” *Pattern Recognit. Lett.*, vol. 109, pp. 44–54, Jul. 2018, doi: 10.1016/j.patrec.2017.09.036.

[31] J. Naskath, G. Sivakamasundari, and A. A. S. Begum, “A study on different deep learning algorithms used in deep neural nets: MLP, SOM, and DBN,” *Wirel. Pers. Commun.*, vol. 128, no. 4, pp. 2913–2936, Feb. 2023, doi: 10.1007/s11277-022-10079-4.

[32] S. Chen, G. I. Webb, L. Liu, and X. Ma, “A novel selective naïve Bayes algorithm,” *KBS*, vol. 192, p. 105361, Mar. 2020, doi: 10.1016/j.knosys.2019.105361.

[33] C.-Y. J. Peng, K. L. Lee, and G. M. Ingersoll, “An introduction to logistic regression analysis and reporting,” *J. Educ. Res.*, vol. 96, no. 1, pp. 3–14, Sep. 2002, doi: 10.1080/00220670209598786.

[34] A. Fernandez, S. Garcia, F. Herrera, and N. v. Chawla, “SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary,” *JAIR*, vol. 61, pp. 863–905, Apr. 2018, doi: 10.1613/jair.1.11192.

[35] I. M. M. Matin, “Hyperparameter tuning menggunakan GridsearchCV pada random forest untuk deteksi malware,” (in Indonesian), *Multinetics,* vol. 9, no. 1, May 2023.

[36] T. Chai and R. R. Draxler, “Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature,” *GMD*, vol. 7, no. 3, pp. 1247–1250, Jun. 2014, doi: 10.5194/gmd-7-1247-2014.

[37] T. O. Hodson, T. M. Over, and S. S. Foks, “Mean squared error, deconstructed,” *J. Adv. Model. Earth Syst.*, vol. 13, no. 12, Dec. 2021, doi: 10.1029/2021MS002681.

[38] H. Belyadi and A. Haghighat, “Unsupervised machine learning: Clustering algorithms,” in *Machine Learning Guide for Oil and Gas Using Python*, Elsevier, 2021, pp. 125–168. doi: 10.1016/B978-0-12-821929-4.00002-0.

[39] Ž. Đ. Vujovic, “Classification model evaluation metrics,” *IJACSA*, vol. 12, no. 6, 2021, doi: 10.14569/IJACSA.2021.0120670.

[40] S. Blom, M. Book, V. Gruhn, R. Hrushchak and A. Köhler, “Write once, run anywhere: A survey of mobile runtime environments,” *2008 The 3rd Int. Conf. Grid Pervasive Comput. - Workshops*, Kunming, China, 2008, pp. 132-137, doi: 10.1109/GPC.WORKSHOPS.2008.19.

[41] A. Setiawan, A. N. Muna, E. R. Arumi, and P. Sukmasetya, “The growth of electronic commerce technology and user interface in Indonesia,” *Test Eng. Manage.*, vol. 83, pp. 16819, 2020.

[42] A. S. Saabith, M. M. M. Fareez, and T. Vinothraj, “Python current trend applications-an overview,” *Int. J. Adv. Eng. Res. Develop.*, vol. 6, no. 10, pp. 6-7, Oct. 2019.

[43] J. Yani, “The role of SQL and NoSQL databases in modern data architectures,” *Int. J. Core Eng. & Manage.*, vol. 6, no. 12, pp. 61–67, 2021.

[44] J. M. R. Corral, “Multimedia system for self-learning C/C++ programming language,” *Innov. Inf. Syst. Technol. Supp. Learn. Res.: Proc. of EMENA-ISTL 2019*, vol. 7, pp. 55, 2019.