VIETNAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

ĐINH CÔNG HƯNG – 521H0064
PHẠM ĐĂNG THANH TRUNG – 522H0148

MIDTERM ESSAY INTRODUCTION TO MACHING LEARNING

HO CHI MINH CITY, 2024

VIETNAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

ĐINH CÔNG HƯNG – 521H0064
PHẠM ĐĂNG THANH TRUNG – 522H0148

MIDETERM ESSAY INTRODUCTION TO MACHING LEARNING

Instructor

Assoc.Prof. Lê Anh Cường

HO CHI MINH CITY, 2024

THANK YOU

Introduction to Maching Learning is a very interesting and beneficial subject. We sincerely thank Mr. Le Anh Cuong for your guidance and for providing the best conditions for me to complete the midterm essay. It helped our gain more knowledge about system design and have a broader understanding of real business processes.

During the implementation of this project, my report inevitably made some unintended mistakes. We kindly hope that you could offer feedback and assistance. We sincerely thank you.

Ho Chi Minh City, date 26 month 10 year 2024
Author
(Sign and write full name)

Đinh Công Hưng

Phạm Đăng Thanh Trung

THE ESSAY WAS COMPLETED AT
TON DUC THANG UNIVERSITY

I hereby declare that this is my own project and is guided by Asoc.Prof. Le Anh Cuong. The research contents and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments, and evaluation are collected by the author ourselves from different sources, clearly stated in the references section.

In addition, the project also uses a few comments, assessments as well as data of other authors, other agencies, and organizations, with citations and sourec annotations.

If any frauds detected, I will take full responsibility for the content of my project. Ton Duc Thang University is not involved in copyright violations caused by me during the implementation process (if any).

Ho Chi Minh City, date 26 month 10 year 2024
Author
(Sign and write full name)

Đinh Công Hưng

Phạm Đăng Thanh Trung

INTRODUCTION TO MACHING LEARNING

ABSTRACT

This is a report on Introduction to Machine Learning by Faculty of Information Technology of Ton Duc Thang University.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1 Introduce the problem

In this document, we will address classification and regression problem related to banking customers. The goal is to predict whether a customer is likely to subscribe to a term deposit based on various features available in the dataset. This problem is important for banks as it can help them identify potential customers for term deposits and optimize their marketing strategies accordingly.

To solve this problem, we will use classification techniques such as k-Nearest Neighbors (kNN), Decision Tree, and Logistic Regression. Additionally, we will approach it as a regression problem using models like Linear Regression and Random Forest Regression, to predict and analyze other numerical attributes of customer behavior.

1.2 Dataset Decscription

The dataset consists of information about banking customers and their financial details. Each row represents a customer and includes the following features:

- **age**: The age of the customer.
- **job**: The type of job held by the customer (e.g., management, services, blue-collar).
- **marital**: The marital status of the customer (e.g., single, married, divorced).
- **education**: The education level of the customer (e.g., primary, secondary, tertiary).
- **default**: Indicates if the customer has credit in default (yes or no).
- **balance**: The account balance of the customer in euros.
- **housing**: Indicates if the customer has a housing loan (yes or no).
- **loan**: Indicates if the customer has a personal loan (yes or no).

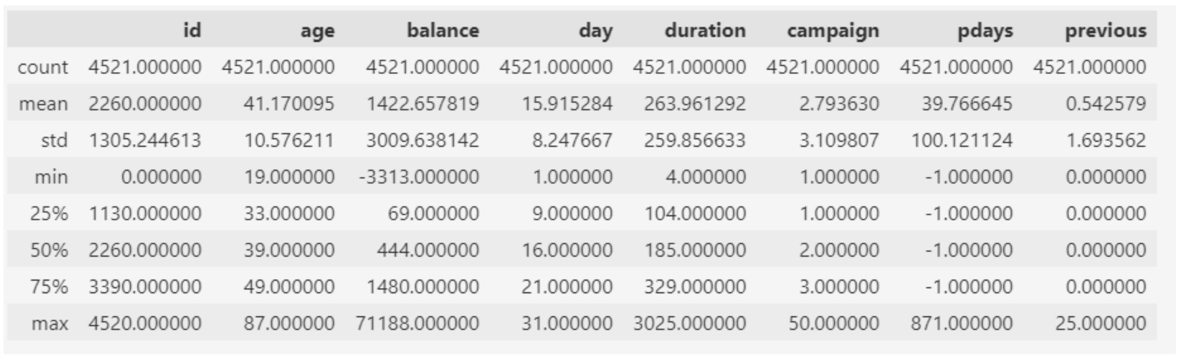
- **contact**: The primary contact communication type (e.g., cellular, telephone, unknown).
- **day**: The last contact day of the month with the customer.
- **month**: The last contact month of the year with the customer.
- **duration**: The duration of the last contact with the customer, in seconds.
- campaign: The number of contacts performed during this campaign for the customer.
- **pdays**: The number of days since the customer was last contacted from a previous campaign. A value of -1 indicates the customer was not previously contacted.
- **previous**: The number of contacts performed before this campaign.
- **poutcome**: The outcome of the previous marketing campaign (e.g., success, failure, unknown).
- **y**: The target variable indicating whether the customer subscribed to a term deposit (yes or no).

CHAPTER 2. DATA ANALYSIS AND PREPROCESSING

2.1 Data characteristics

This dataset contains information about bank customers, focusing on their personal data and financial history to predict their likelihood of subscribing to a term deposit. The characteristics of the data are analyzed in several aspects as follows:

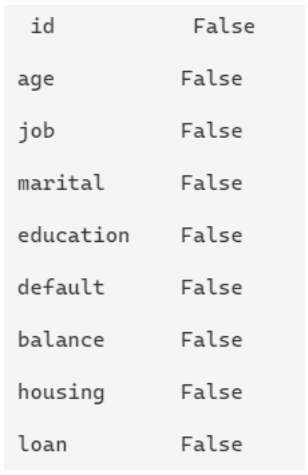
- **Statistical Summary:**



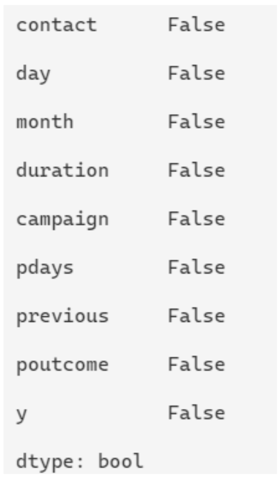
	id	age	balance	day	duration	campaign	pdays	previous
count	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000
mean	2260.000000	41.170095	1422.657819	15.915284	263.961292	2.793630	39.766645	0.542579
std	1305.244613	10.576211	3009.638142	8.247667	259.856633	3.109807	100.121124	1.693562
min	0.000000	19.000000	-3313.000000	1.000000	4.000000	1.000000	-1.000000	0.000000
25%	1130.000000	33.000000	69.000000	9.000000	104.000000	1.000000	-1.000000	0.000000
50%	2260.000000	39.000000	444.000000	16.000000	185.000000	2.000000	-1.000000	0.000000
75%	3390.000000	49.000000	1480.000000	21.000000	329.000000	3.000000	-1.000000	0.000000
max	4520.000000	87.000000	71188.000000	31.000000	3025.000000	50.000000	871.000000	25.000000

The dataset contains 4,521 rows with features such as age, balance, duration, etc. A statistical analysis was conducted, including mean, standard deviation, and the minimum and maximum values for each feature.

- **Data Type and Null Value Check:**



```
id           False
age          False
job          False
marital      False
education    False
default      False
balance      False
housing      False
loan         False
```

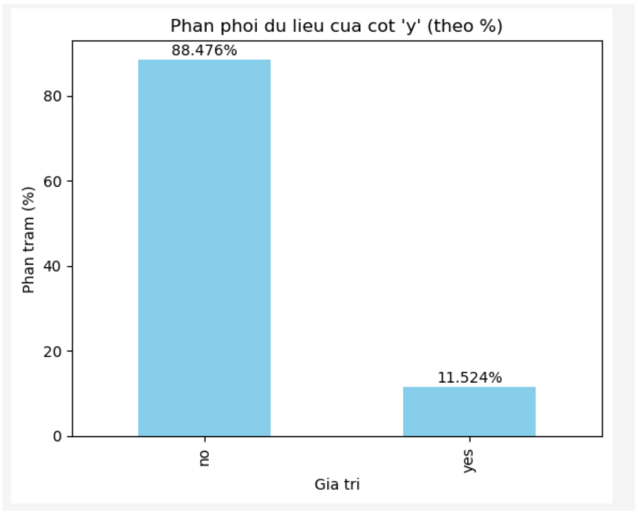


```
contact      False
day          False
month        False
duration     False
campaign     False
pdays        False
previous     False
poutcome     False
y            False
dtype: bool
```

All columns in the dataset are complete, with no missing values, which simplifies the analysis and model building without needing imputation.

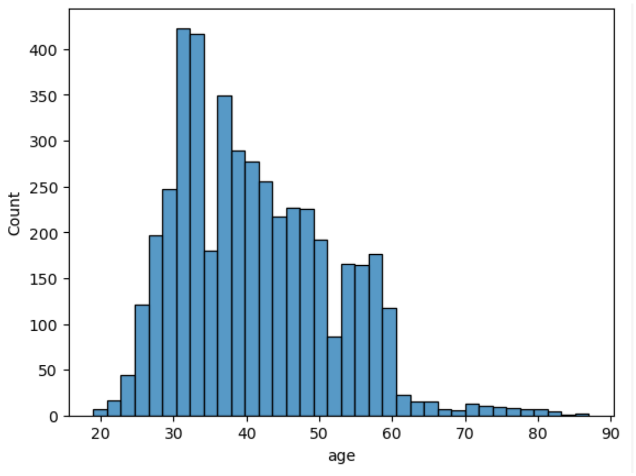
The dataset includes both numerical features (age, balance, duration) and categorical features (job, marital, education).

- **Target Variable Distribution (y):**

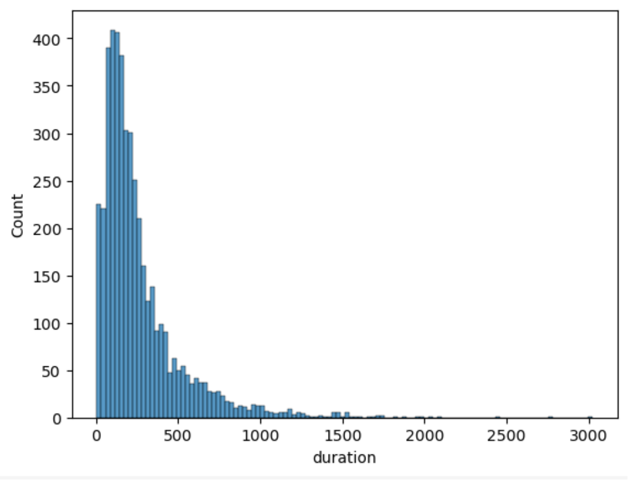


The target variable y indicates whether the customer subscribed to a term deposit, with approximately 88.5% of values being "no" and 11.5% being "yes". This imbalance needs to be considered when developing the model.

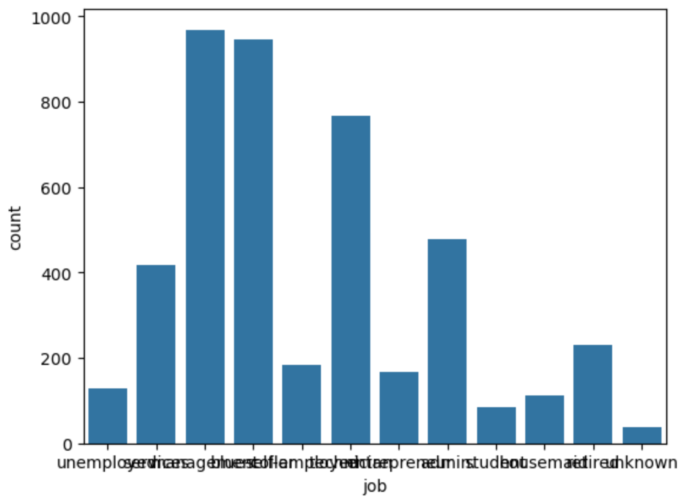
- **Distribution of Key Features:**



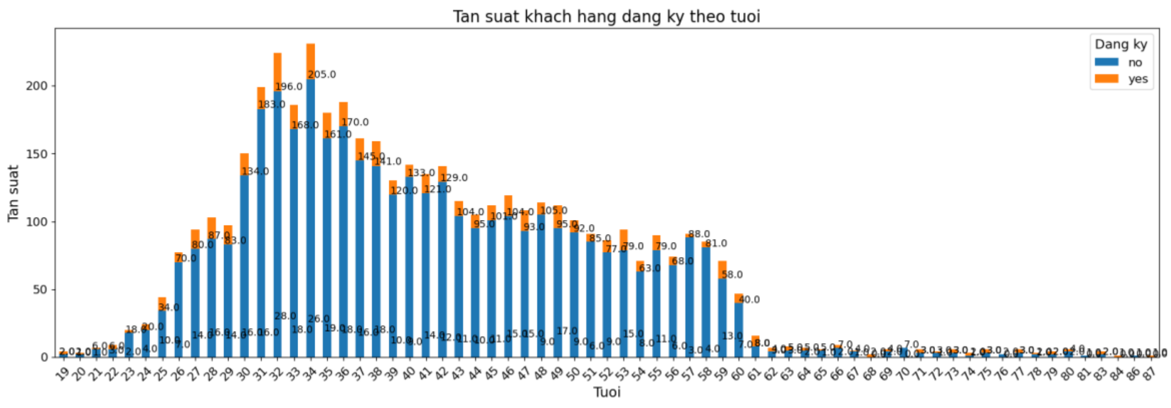
The age distribution shows a concentration of customers between the ages of 30 and 40, with fewer customers at younger or older ages.



The duration of contact with customers is unevenly distributed, with most calls being relatively short, while a few extend beyond 2000 seconds.



The most common job types among customers are "management," "blue-collar," and "technician."



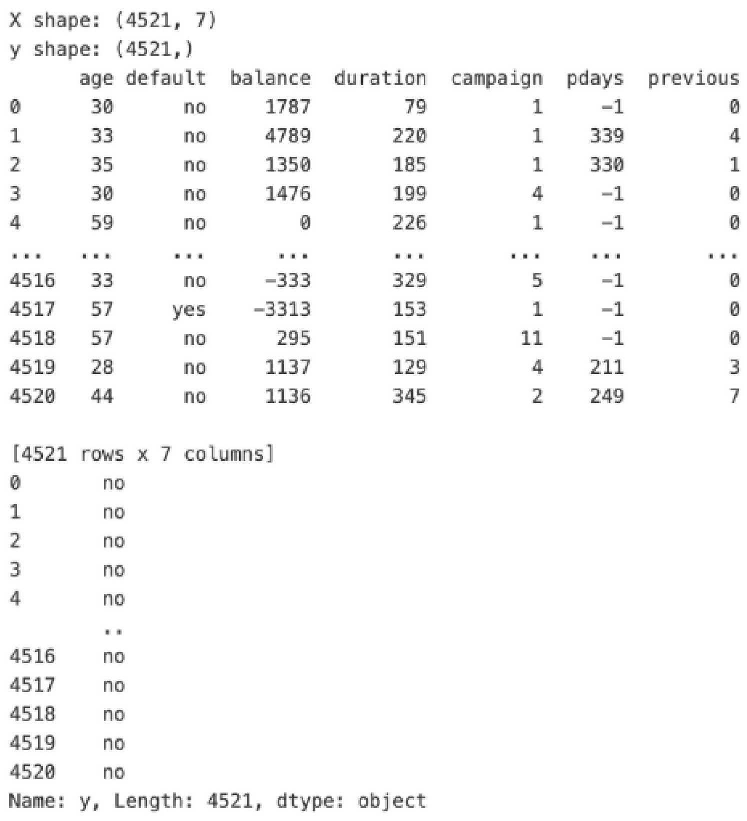
The bar chart presents the frequency of bank customers by age and their subscription status to a term deposit. The chart is divided into two colors: blue for customers who did not subscribe to a term deposit and orange for those who did.

This visual allows us to observe the age distribution and general patterns in customer behavior concerning term deposit subscriptions

2.2 Data preprocessing

The data preprocessing phase involves preparing the raw data for model building by transforming and normalizing relevant features.

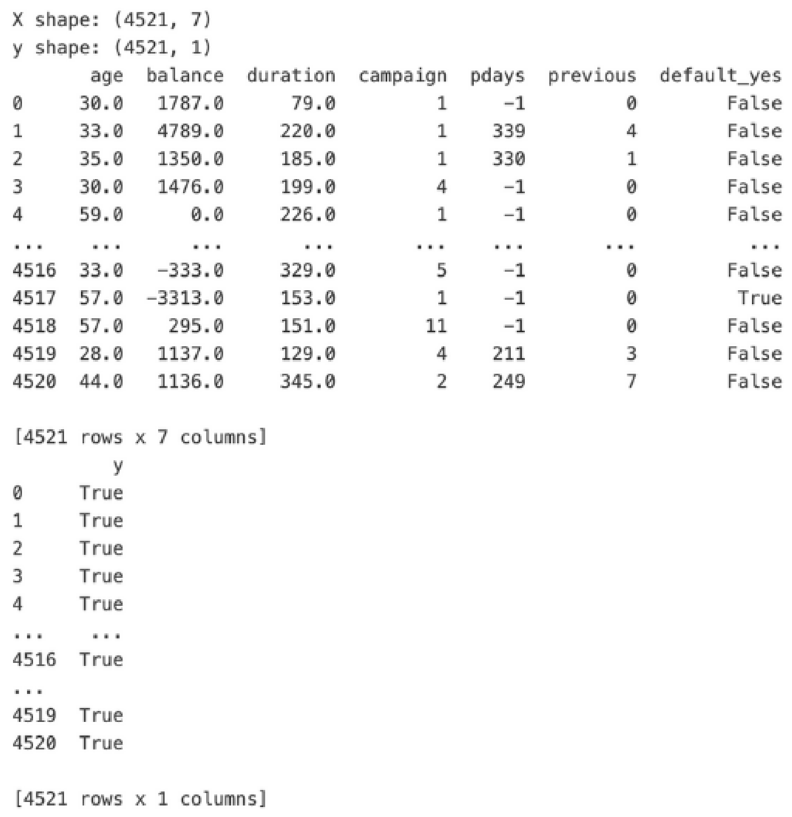
- **Initial Data Exploration**:



```
X shape: (4521, 7)
y shape: (4521,)
      age default  balance  duration  campaign  pdays  previous
0      30      no     1787        79         1     -1         0
1      33      no     4789       220         1    339         4
2      35      no     1350       185         1    330         1
3      30      no     1476       199         4     -1         0
4      59      no        0       226         1     -1         0
...   ...     ...      ...       ...       ...    ...       ...
4516   33      no     -333       329         5     -1         0
4517   57     yes    -3313       153         1     -1         0
4518   57      no      295       151        11     -1         0
4519   28      no     1137       129         4    211         3
4520   44      no     1136       345         2    249         7

[4521 rows x 7 columns]
0       no
1       no
2       no
3       no
4       no
        ..
4516    no
4517    no
4518    no
4519    no
4520    no
Name: y, Length: 4521, dtype: object
```

The initial dataset contains 4,521 rows and 7 columns, including features such as age, balance, duration, and campaign. The target variable y indicates whether a customer subscribed to a term deposit. This exploration provided an overview of the data structure and types, highlighting the presence of both numerical and categorical attributes.



```
X shape: (4521, 7)
y shape: (4521, 1)
       age  balance  duration  campaign  pdays  previous  default_yes
0     30.0   1787.0      79.0         1     -1         0        False
1     33.0   4789.0     220.0         1    339         4        False
2     35.0   1350.0     185.0         1    330         1        False
3     30.0   1476.0     199.0         4     -1         0        False
4     59.0      0.0     226.0         1     -1         0        False
...    ...      ...       ...       ...    ...       ...          ...
4516  33.0   -333.0     329.0         5     -1         0        False
4517  57.0  -3313.0     153.0         1     -1         0         True
4518  57.0    295.0     151.0        11     -1         0        False
4519  28.0   1137.0     129.0         4    211         3        False
4520  44.0   1136.0     345.0         2    249         7        False

[4521 rows x 7 columns]
         y
0     True
1     True
2     True
3     True
4     True
...    ...
4516  True
...
4519  True
4520  True

[4521 rows x 1 columns]
```

Numerical features like age, balance, and duration were standardized to ensure that they have a consistent scale, which is essential for many machine learning algorithms that are sensitive to feature scales.

Categorical features were encoded, such as converting the default column into a binary representation (default_yes), where "yes" or "no" values were transformed into True or False. This encoding makes categorical variables compatible with machine learning models.

By performing these preprocessing steps, we created a standardized and clean dataset that is better suited for machine learning models, ensuring all features are in a comparable range and properly formatted.

CHAPTER 3. CLASSIFICATION AND REGRESSION

3.1 Models Used

In this study, we applied various machine learning models for both classification and regression tasks, aiming to predict customer subscription likelihood and analyze factors influencing this decision.

Classification Models:

- **k-Nearest Neighbors (kNN):** It is particularly effective for classification tasks where decision boundaries are nonlinear and requires minimal assumptions about the data distribution.
- **Decision Tree:** This model is intuitive and interpretable, making it useful for understanding which features contribute most to predicting customer subscriptions.
- **Logistic Regression:** This model is effective in cases where the relationship between features and the target is approximately linear, providing a clear understanding of how each feature influences the probability of subscription.

Regression Models:

- **Linear Regression:** It finds the best-fit line by minimizing the mean squared error, providing an estimate of the relationship between input features and the target variable. This model helps analyze the impact of numerical features on the target.
- **Random Forest Regression:** By combining the results of multiple trees, it reduces overfitting and improves prediction accuracy, especially in datasets with complex, nonlinear relationships.

3.2 Implementation:

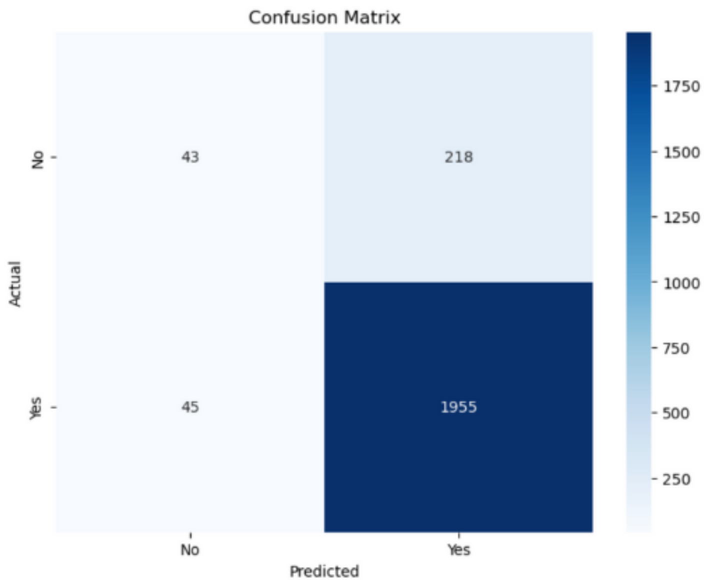
3.2.1 Classification

- **Logistic Regression:**

Steps to follow:

1. Initialize the model: Use LogisticRegression with random_state=0 and max_iter=500.
2. Train the model: Train on the training set (X_train, y_train).
3. Make predictions: Predict on the test set (X_test).
4. Calculate metrics: Accuracy, Precision, Recall, F1 Score.
5. Save results: Create a DataFrame to store the evaluation metrics.

Result:



```
              Model  Accuracy  Precision  Recall  F1 Score
Logistic Regression   0.88368   0.899678  0.9775  0.936976
```

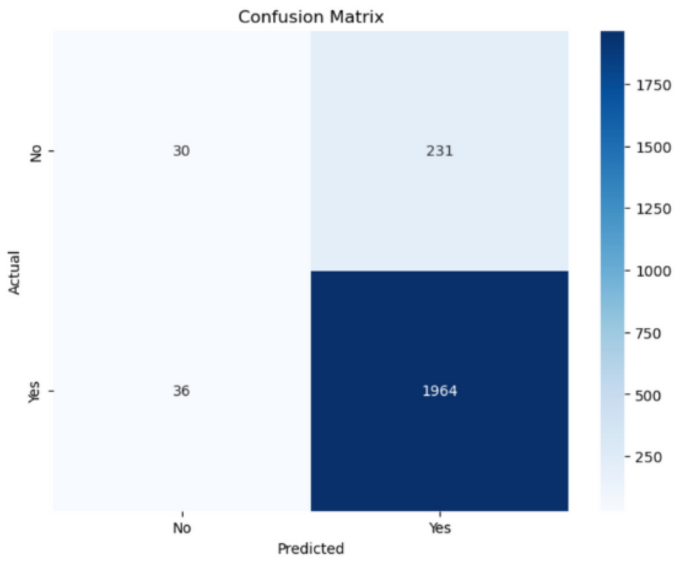
These results indicate that the Logistic Regression model is highly effective for this classification task, allowing our to achieve reliable predictions with a good balance between precision and recall. This performance confirms that Logistic Regression is a strong choice for predicting customer subscriptions in this context.

- **k-Nearest Neighbors (kNN):**

Steps to follow:

1. Initialize the model: Use KNeighborsClassifier with n_neighbors=15, metric='minkowski', and p=2.
2. Train the model: Train on the training set (X_train and y_train).
3. Make predictions: Predict on the training set (X_train and y_train).
4. Calculate metrics: Accuracy, Precision, Recall, F1 Score.
5. Save results: Create a DataFrame to store the evaluation metrics.

Result:



```
Model  Accuracy  Precision  Recall  F1 Score
  KNN  0.881911   0.894761   0.982  0.936353
```

Our results indicate that the k-Nearest Neighbors (kNN) model performs effectively in this classification task, achieving an accuracy of 88.19%. With a high recall score of 0.982, the model successfully identifies the majority of actual "Yes" cases (subscribers).

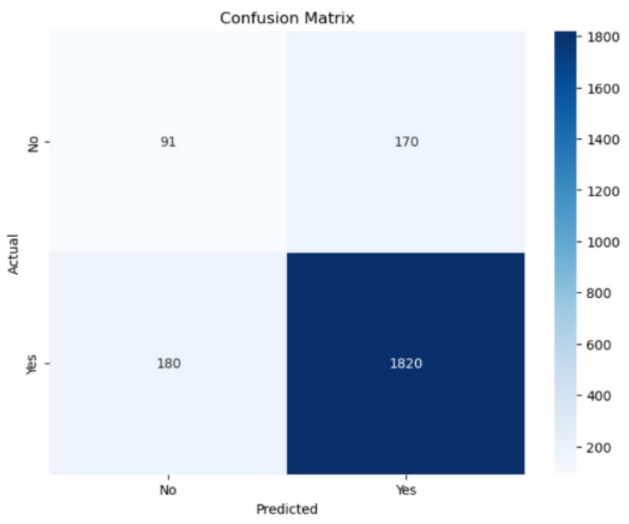
The precision score of 0.8948 demonstrates the model’s reasonable accuracy in correctly predicting "Yes" cases, while the F1 score of 0.9364 reflects a good balance between precision and recall.

- **Decision Tree**

Steps to follow:

1. Initialize the model: Use DecisionTreeClassifier with criterion='entropy' and random_state=0.
2. Train the model: Train on the training set (X_train and y_train).
3. Make predictions: Predict on the test set (X_test).
4. Calculate metrics: Accuracy, Precision, Recall, F1 Score.
5. Save results: Create a DataFrame to store the evaluation metrics.

Result:

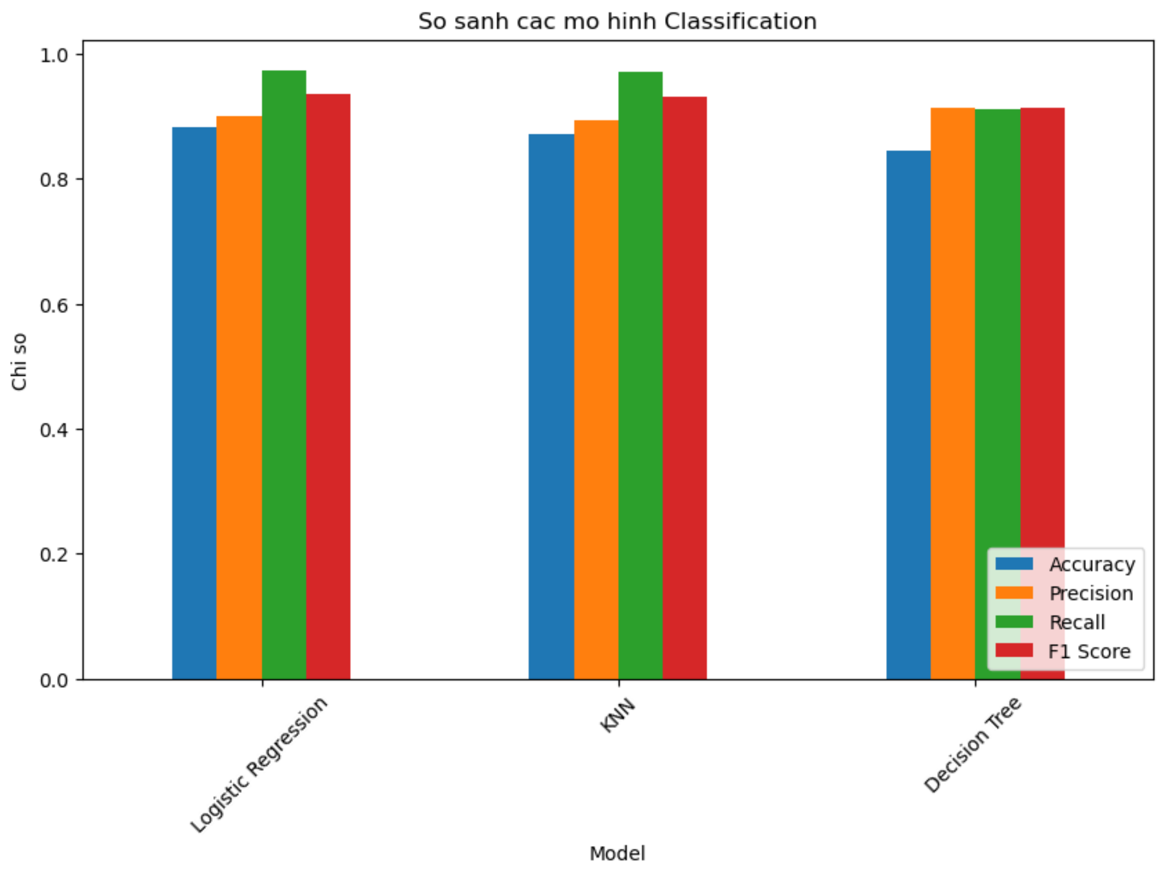


```
           Model  Accuracy  Precision  Recall  F1 Score
   Decision Tree  0.845201   0.914573    0.91  0.912281
```

Our analysis shows that the Decision Tree model achieves an accuracy of 84.52%. With a precision score of 0.9146, the model demonstrates strong performance in accurately predicting "Yes" cases (subscribers). The recall score of 0.91 indicates that the model successfully captures most actual "Yes" cases.

The F1 score of 0.9123 reflects a balanced trade-off between precision and recall, validating the Decision Tree as a reasonable choice for predicting customer subscriptions.

- **Compare Classification:**



Based on the comparison chart, we can observe the following insights for the classification models (Logistic Regression, k-Nearest Neighbors, and Decision Tree) applied to this problem:

1. **Logistic Regression**
   - The kNN model slightly outperforms the others in terms of recall, indicating that it captures more actual "Yes" cases (subscribers) than Logistic Regression and Decision Tree.
   - This model also has a balanced performance in precision and F1 score, making it a strong contender for this task.

2. **k-Nearest Neighbors (kNN)**
   - Logistic Regression achieves a well-rounded balance in all metrics, performing slightly better than Decision Tree in terms of accuracy and precision.
   - This model may be a preferred choice if a linear model is desired, as it maintains strong performance without much variability across metrics.
3. **Decision Tree**
   - The Decision Tree model has slightly lower accuracy compared to kNN and Logistic Regression but maintains a high precision, indicating it effectively predicts true positives.
   - However, it has a slightly lower recall, suggesting that it may miss a few "Yes" cases. This model may benefit from further tuning or ensemble techniques to enhance recall.

In summary, while all three models are effective for this classification task, kNN demonstrates the highest recall, making it ideal for scenarios where identifying "Yes" cases is prioritized. Logistic Regression offers balanced performance across all metrics, and Decision Tree, although slightly lower in recall, still provides strong precision

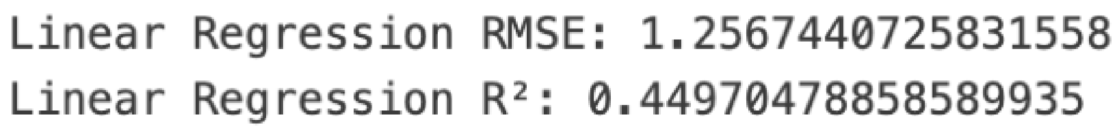
3.2.2 Regression

- **Linear Regression**

Steps to follow:

1. Initialize the model: Use LinearRegression.
2. Train the model: Train on the training set (X_train and y_train).
3. Make predictions: Predict on the test set (X_test).
4. Calculate metrics: Accuracy, Precision, Recall, F1 Score.
5. Save results: Create a DataFrame to store the evaluation metrics.

Result:



```
Linear Regression RMSE: 1.2567440725831558
Linear Regression R²: 0.4497047885858935
```

The RMSE indicates a moderate level of prediction error, suggesting that the model provides reasonably accurate estimates but leaves room for improvement.

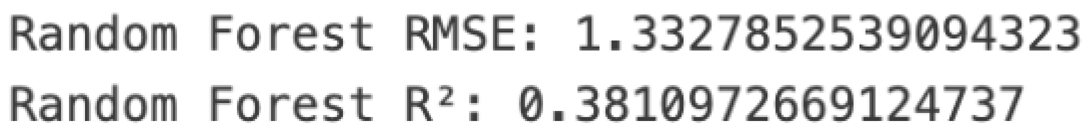
The R² score of 0.450 shows that the model explains around 45% of the variance in the target variable, capturing some of the underlying patterns but not fully explaining the data

- **Randoom Forest Regression**

Steps to follow:

1. Initialize the model: Use RandomForestRegressor with random_state=42.
2. Train the model: Train on the training set (X_train and y_train).
3. Make predictions: Predict on the test set (X_test).
4. Calculate metrics: Accuracy, Precision, Recall, F1 Score.
5. Save results: Create a DataFrame to store the evaluation metrics.

Result:

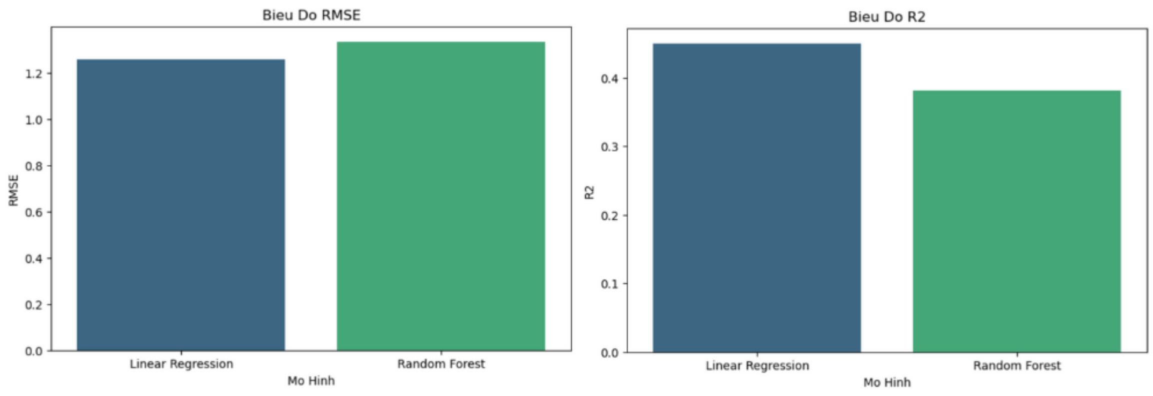


```
Random Forest RMSE: 1.332785253909432З
Random Forest R²: 0.3810972669124737
```

The RMSE indicates a moderate level of error in the model’s predictions, suggesting that while the model provides reasonable estimates, there is still some deviation from the actual values

The R² score of 0.381 shows that the model explains about 38.1% of the variance in the target variable, indicating a limited ability to capture the full patterns in the data.

- **Regression Comparison**



```
             Mo Hinh      RMSE       R^2
0  Linear Regression  1.256744  0.449705
1      Random Forest  1.332785  0.381097
```

Based on the comparison charts for the regression models (Linear Regression and Random Forest), we observe the following:

1. **RMSE Comparison**

   The RMSE values indicate that Linear Regression (1.257) slightly outperforms Random Forest (1.333) in terms of prediction accuracy. A lower RMSE for Linear Regression suggests that it has a smaller average error in its predictions compared to Random Forest.

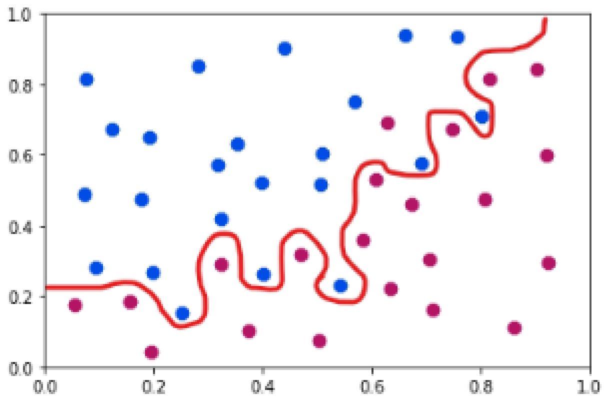
2. **R^2 Comparison**

   The R^2 scores further highlight the difference, with Linear Regression achieving a score of 0.450, indicating that it explains around 45% of the variance in the target variable. In contrast, Random Forest has an R^2 score of 0.381, explaining approximately 38.1% of the variance.

Overall, Linear Regression demonstrates a slight advantage over Random Forest in both RMSE and R^2 metrics, making it a better fit for this regression task. However, both models show room for improvement in capturing the full complexity of the data

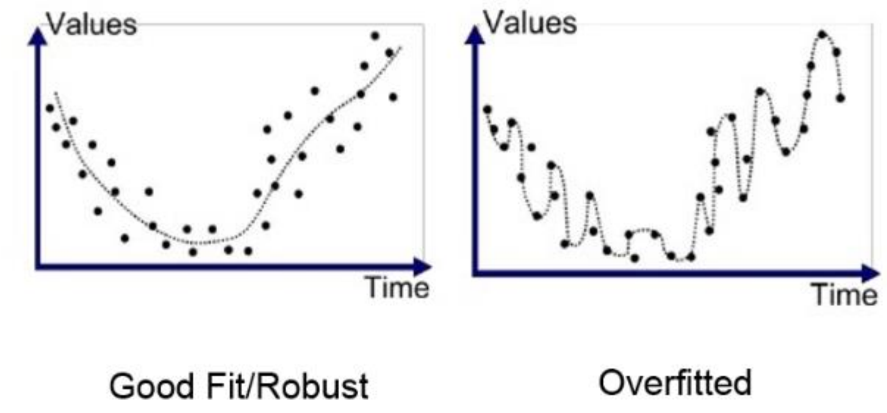
CHAPTER 4. OVERFITTING

4.1 Overfitting Concept



Overfitting is a problem, or a challenge we face during the training of the model. This happens when the model is giving very low bias and very high variance.

Overfitting happens when our model cramps the training data so well that it gets trained over the noise and inaccurate data entries in our data set as well, which give high accuracy and low loss over the training data set (ideal case of Low Bias). And when we introduce the same model with new test data, it results in very poor accuracy and very high loss(ideal case of High Variance).

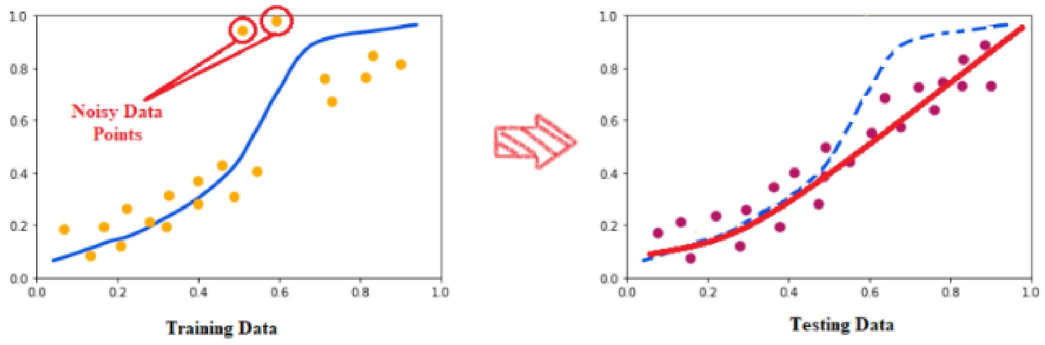


Overfitting is not good for any machine learning model as the final aim of the machine is to predict new upcoming scenarios which nobody has seen before. But overfitting causes the model to predict very poorly on new data points. So, we have to understand the causes that make the model overfits the training data.

4.2 Causes of Overfitting

4.2.1 Noisy data or Inaccurate data

If we train our model in noisy data or inaccurate data, then it may give good results on that data and also have low bias. But if we again introduce new unseen data to the model to predict, then it may decrease the accuracy and increases the variance.



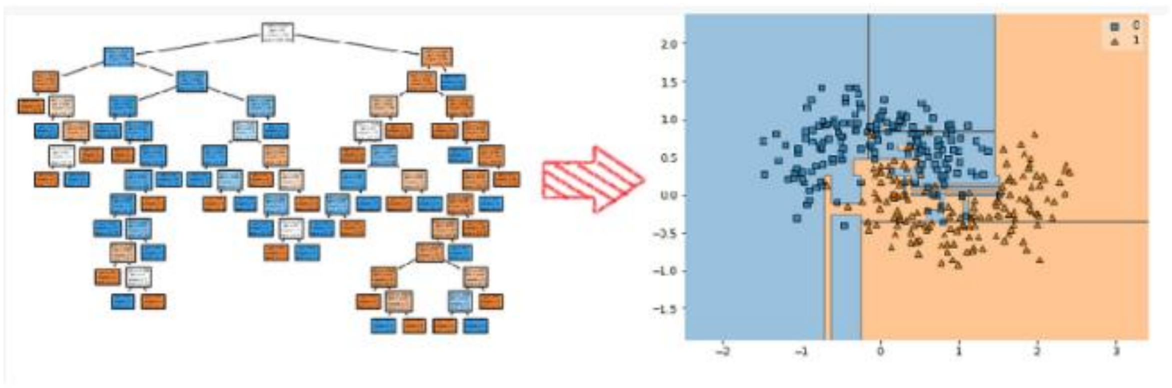
As we can observe in the above picture, in the training data, there are some noisy data points, and because of the overfitting tendency of the model, it is trying to fit even those points. But when we test that with test data, then the red line fits the data properly.

4.2.2 Low size of training data

Overfitting is caused by the size of the data. The more the data more the model will learn hence, we try to give better predictions. If the training data is low, then the model will not get to explore all the scenarios or possibilities. This makes the model only fit the given data, but when we introduce it with unseen data, then the accuracy of the prediction will fall, and also the variance will increase.

4.2.3 Complexity of the model

Overfitting is also caused by the complexity of the predictive function formed by the model to predict the outcome. The more complex the model more it will tend to overfit the data. hence the bias will be low, and the variance will get higher.



Fully Grown Decision Tree

The above picture shows the decision boundary generated by a fully grown decision tree. As we can see, the boundaries are not that much smoother, and it clearly shows overfitting as the decision tree is very complex.

4.3 Overfitting Solution

4.3.1 Regularization Method

Regularization is a technique used to limit the learning of a machine learning model by introducing some penalty term in the cost function. It helps to generalize the overall model's function and hence reduces the overfitting problem.

- **L1 Regularization**

It is also called LASSO Regularization. It stands for Least Absolute and Selection Operator. We can calculate it by the multiplying with the lambda the weight of each individual feature. This term will be added to the cost function

The equation for the cost function in ridge regression will be:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |\omega_i|$$

- **L2 Regularization**

It is also called Ridge Regularization. We can calculate it by multiplying the lambda by the squared of each individual feature. This term will be added to the cost function.

The equation for the cost function ridge regression will be:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |\omega_i^2|$$

4.3.2 Cross-Validation

Cross validation is an improvement of validation where the amount of data in the validation set is small but the model quality is evaluated on many different validation sets. A commonly used way is to split the training set intokksubsets with no common elements, of approximately equal size.

At each test run, called a run, one of the *k*. The subset is taken as the validation set. The model will be build based on the union of *k-1* the remaining subset. The final model is determined based on the average of the training error and validation error. This approach is also known as *k-fold cross validation*.

4.3.3 Pruning (Decision Tree)

Pruning is a technique that removes parts of the decision tree and prevents it from growing to its full depth. Pruning removes those parts of the decision tree that do not have the power to classify instances.

- **Limit tree depth (max_depth):** By restricting the depth of the tree, you can prevent the tree from learning too much from the training data, keeping the model simple and avoiding overfitting.
- **Mininmum samples required to split (min_samples_split):** Setting a minimum number of samples needed to split a node ensures that each split has enough data to make a decision, reducing the likelihood of creating overly small nodes with little information.

- **Minimum samples at a leaf (min_samples_leaf):** Establishing a minimum number of samples required at a leaf node can prevent the creation of leaves with very few samples, which makes the model sensitive to noise in the data.

4.4 Overfitting solution for models

4.4.1 Logistic Regression

Adding a regularization term, such as L1 (Lasso) or L2 (Ridge), to the logistic regression cost function helps control overfitting by penalizing large coefficients:

- L1 regularization can result in sparse models by pushing some coefficients to zero, effectively performing feature selection.
- L2 regularization penalizes large coefficients more evenly, reducing the impact of less relevant features without forcing coefficients to zero.

4.4.2 k-Nearest Neighbors (kNN)

- **Increase the value of k:** By selecting a larger value for kkk, the KNN model considers more neighboring data points when classifying or predicting a new instance.
- **Data normalization:** Normalizing the data ensures that all features contribute equally to the distance calculation, improving the model’s performance and accuracy.

4.4.3 .Decision Tree

- **Limit tree depth (max_depth):** Limiting the depth of the Decision Tree helps control overfitting by preventing the model from learning overly specific patterns in the training data.
- **Minimum samples to split (min_samples_split):** This parameter sets the minimum number of samples required to split a node.
- **Minimum samples at a leaf (min_samples_leaf):** Setting a minimum number of samples at a leaf node avoids leaves with very few samples.

4.4.4 Linear Regression

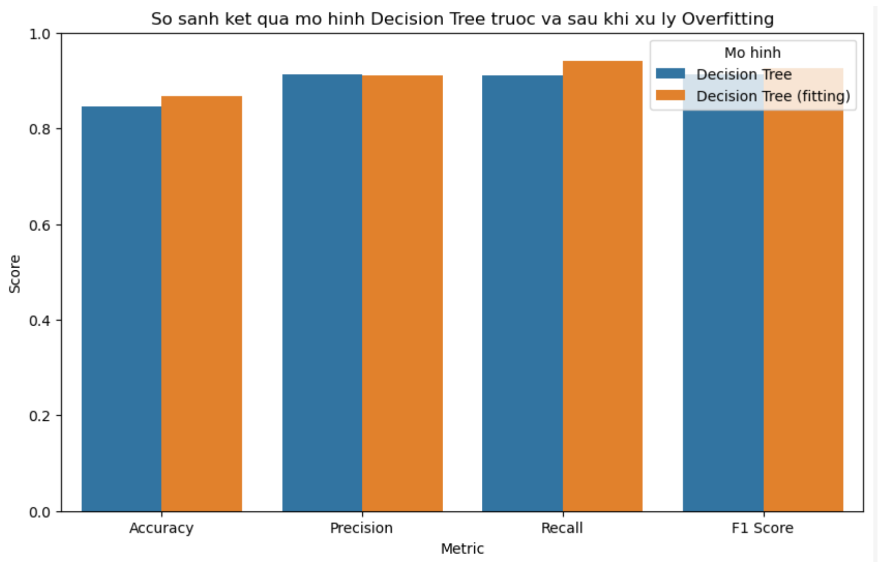
Ridge Regression, a form of L2 regularization, adds a penalty on the size of the coefficients to the cost function of linear regression. This penalty helps prevent the model from assigning too much importance to any single feature, reducing the risk of overfitting

4.4.5 Random Forest Regression

- **Increase the number of trees (n_estimators):** makes the model more robust and stable, as each additional tree adds diversity to the ensemble. More trees generally lead to better performance by reducing variance and overfitting.
- **Limit tree depth (max_depth):** Setting a maximum depth for each tree prevents individual trees from growing too complex, which can reduce the likelihood of overfitting.

4.5 Result after applying overfitting solutions

4.5.1 Decision Tree

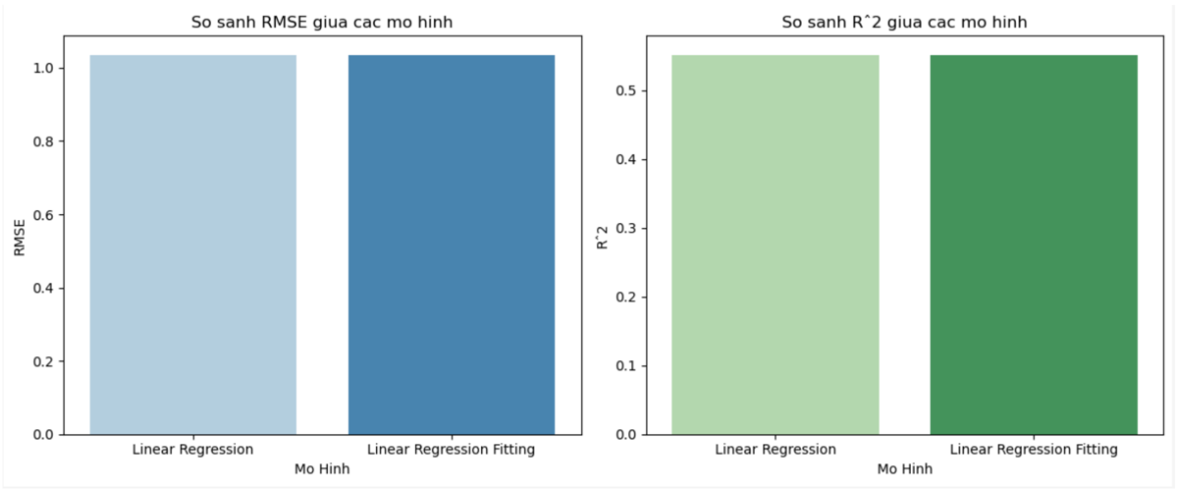


Based on the comparison chart, we can observe the impact of applying overfitting solutions to the Decision Tree model:

- After applying the overfitting control techniques (shown in orange as "Decision Tree (fitting)"), the model’s performance across metrics such as Accuracy, Precision, Recall, and F1 Score shows increased stability.
- The model’s metrics before and after applying overfitting solutions remain similar, suggesting that the modifications (such as limiting tree depth and setting minimum samples per split and leaf) prevented overfitting without significantly affecting the model’s ability to capture important data patterns.

In summary, the overfitting solutions applied to the Decision Tree model have enhanced its reliability and generalizability, achieving similar performance metrics while making it more stable and resilient when applied to new data.

4.5.2 Linear Regression



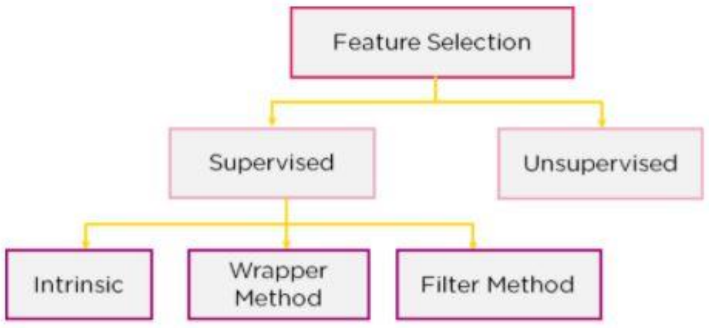
Based on the comparison of RMSE and R² values before and after applying overfitting solutions to the Linear Regression model, we can make the following observations:

- The RMSE values for both the original and adjusted ("fitting") Linear Regression models are very close, indicating that the overfitting solutions have not significantly impacted the model's prediction error.
- The R² scores are also very similar before and after the adjustments. This shows that the model's ability to explain the variance in the target variable remains consistent.

In summary, the overfitting solution applied to the Linear Regression model has strengthened its stability and resistance to overfitting without sacrificing accuracy or interpretability, making it a more reliable.

CHAPTER 5. FEATURE SELECTION

5.1 Feature selection Explanation

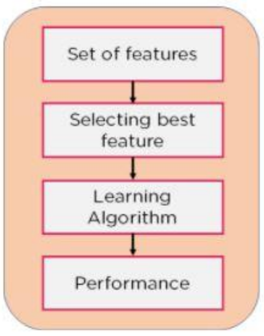


Feature selection models are of two types:

- Supervised Models: Supervised feature selection refers to the method which uses the output label class for feature selection. They use the target variables to identify the variables which can increase the efficiency of the model
- Unsupervised Models: Unsupervised feature selection refers to the method which does not need the output label class for feature selection. Using them for unlabelled data.

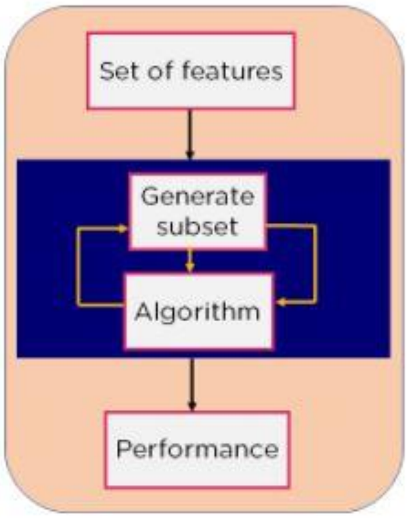
5.1.1 Filter Method

Features are dropped based on their relation to the output, or how they are correlating to the output. Using correlation to check if the features are positively or negatively correlated to the output labels and drop features accordingly.



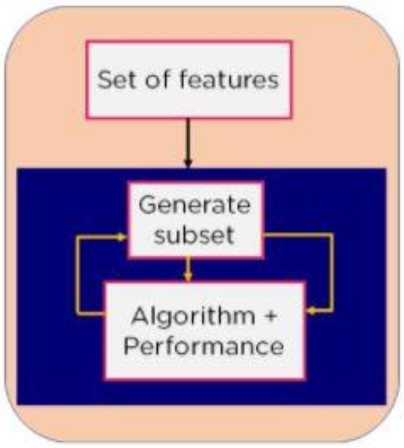
5.1.2 Wrapper Method

Split our data into subsets and train a model using this. Based on the output of the model, we add and subtract features and train the model again



5.1.3 Intrinsic Method

Combines the qualities of both the Filter and Wrapper method to create the best subset.



Method takes care of the machine training iterative process while maintaining the computation cost to be minimum. Eg: Lasso and Ridge Regression.

5.2 Why is feature selection important

- **Minimizing overfitting:** A complex model with too many features can lead to overfitting, where the model learns noise from the training data and fails to generalize to new data. This results in high accuracy on training data but poor performance on unseen datasets.
- **Enhancing performance:** Selecting important features is essential for improving model performance. By focusing on relevant features, the model can make better predictions and operate more efficiently, avoiding confusion from irrelevant data.
- **Saving time and resources:** A simpler model with fewer features reduces training time and computational resources needed. This is especially beneficial with large datasets, allowing for quicker iterations and easier maintenance, ultimately leading to more efficient workflows.

5.3 Feature selection using correlation analysis

Correlation is a bivariate analysis that measures the strength of association between two variables and the direction of the relationship. In terms of the strength of relationship, the value of the correlation coefficient varies between +1 and -1.

A value of ± 1 indicates a perfect degree of association between the two variables. As the correlation coefficient value goes towards 0, the relationship between the two variables will be weaker.

Usually, in statistics, we measure four types of correlations: Pearson correlation, Kendall rank correlation, Spearman correlation, and the Point-Biserial correlation. In this report, our group decided to use Pearson correlation.

5.3.1 Pearson Correlation

Pearson correlation is the most widely used correlation statistic to measure the degree of the relationship between linearly related variables. For example, in the stock market, if we want to measure how two stocks are related to each other, Pearson *r* correlation is used to measure the degree of relationship between the two.

The point-biserial correlation is conducted with the Pearson correlation formula except that one of the variables is dichotomous. The following formula is used to calculate the Pearson *r* correlation:

$$r_{xy} = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}}$$

r_{xy} = Peason r correlation coefficient between x and y

n = number of observations

x_i = value of x (for ith observation)

y_i = value of y (for ith observation)

5.3.2 Application of Correlation Analysis

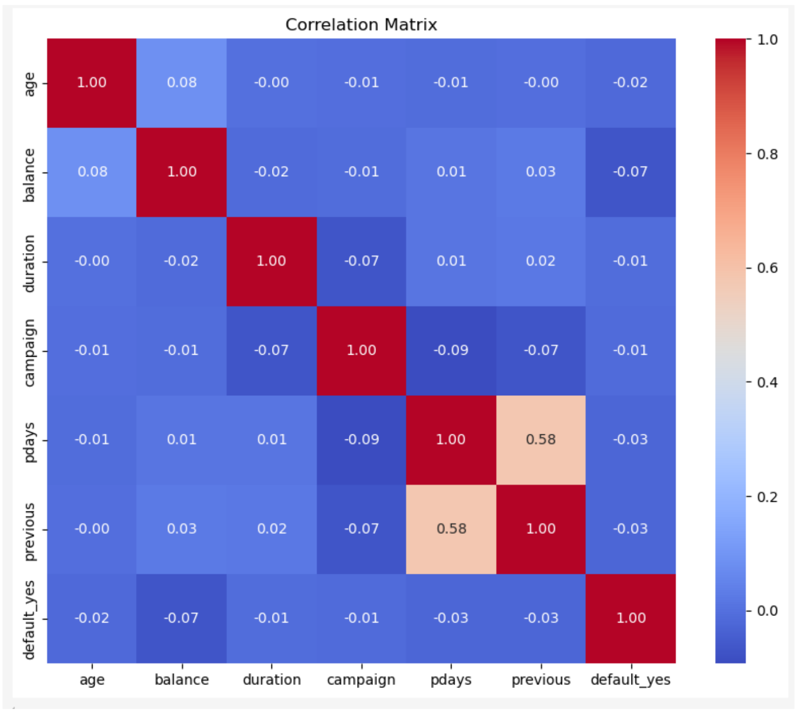
To improve model performance and reduce potential multicollinearity issues, we applied feature selection using correlation analysis. This process involved calculating the correlation matrix of the features and setting a threshold to identify and retain only those features with sufficient correlation with the target variable, while removing those that exhibited high correlation with each other

Step to follow:

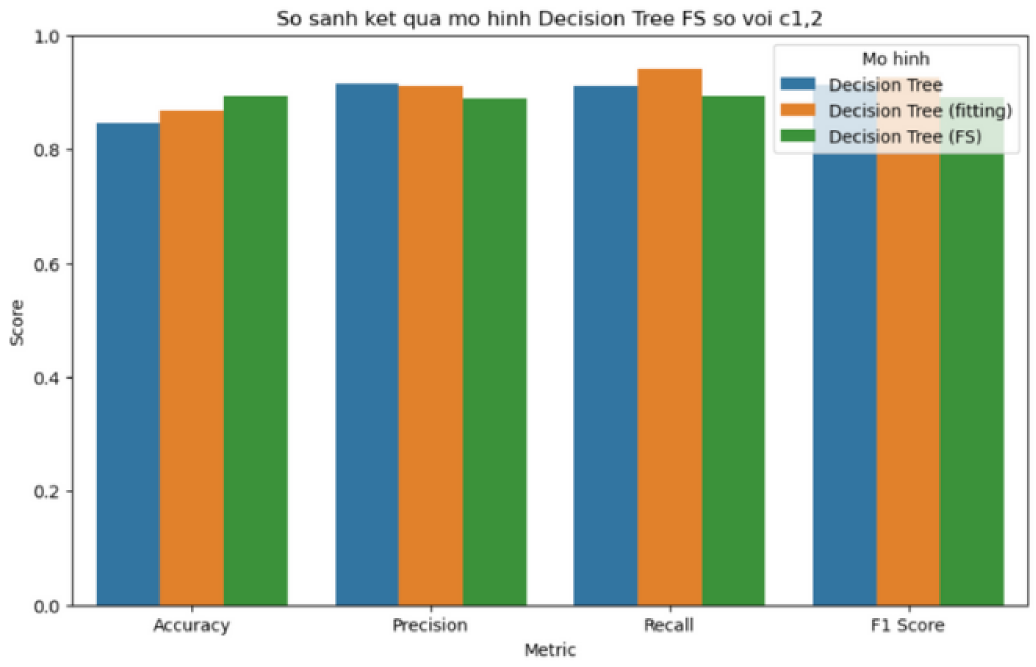
1. Calculate the correlation matrix: correlation_matrix = X.corr() (Pearson method).
2. Display the correlation matrix using a heatmap: Use the Seaborn library to plot a heatmap showing the correlation between features.
3. Set the correlation threshold: correlation_threshold (Only features with an absolute correlation coefficient greater than correlation_threshold with the target variable will be selected).

4. Calculate the correlation coefficient between each feature and the target variable: Use the apply function to apply the corr function to each column in X and compute the absolute correlation coefficient with the target variable y.
5. Select features with high correlation to the target variable: Filter the features with an absolute correlation coefficient greater than the set threshold and save the list of these features.
6. Remove features that are highly correlated with each other: Eliminate features that are highly correlated with each other to avoid multicollinearity.

5.4 Decision Tree with Feature Selection



We observed a moderate correlation between *pdays* and *previous* (0.58), which we may consider during feature selection. Overall, this matrix confirms that our features contribute unique information, enhancing the stability and generalizability of our model.

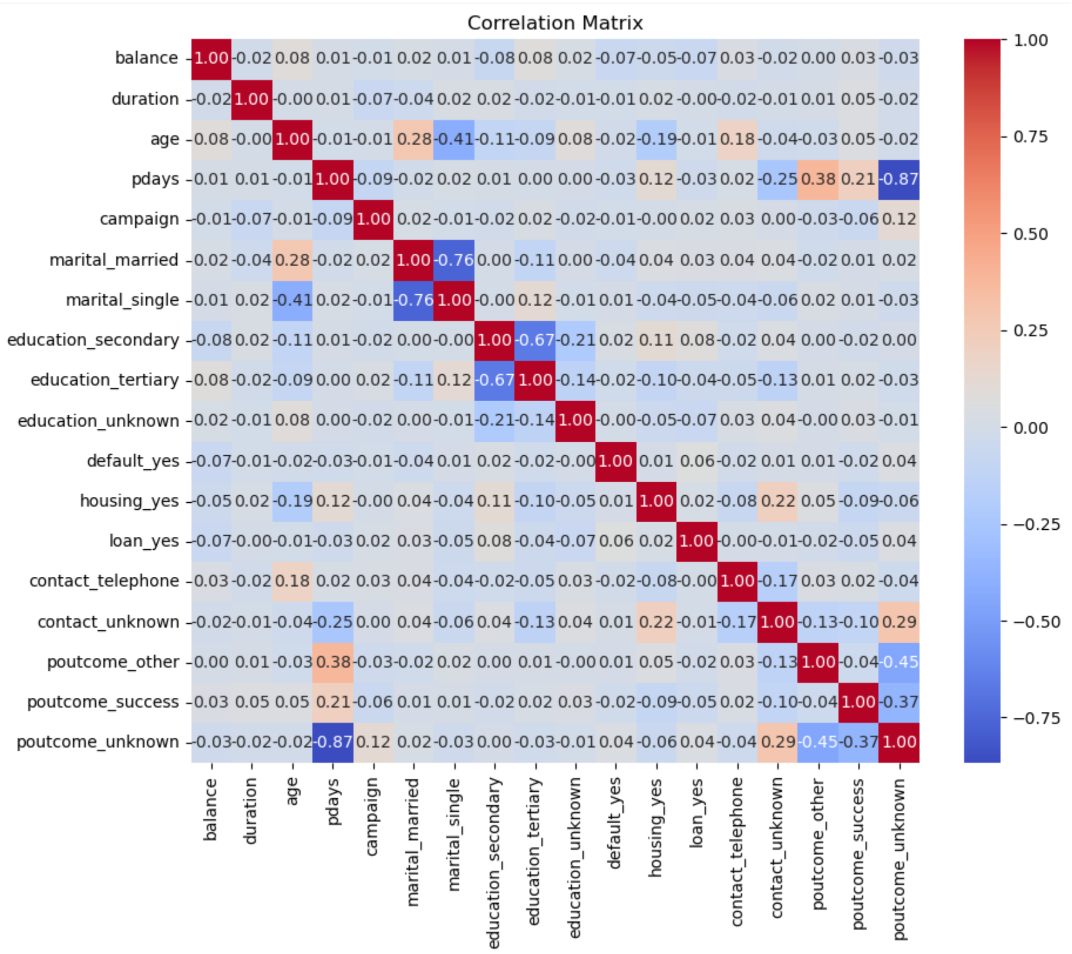


```
          Decision Tree  0.845201  0.914573  0.910000  0.912281
Decision Tree (fitting)  0.867403  0.911729  0.941323  0.926290
     Decision Tree (FS)  0.893923  0.888321  0.893923  0.890855
```

- **Before feature selection:** The Decision Tree model had an accuracy of approximately 84.52% and an F1 score of 0.912.
- **After feature selection:** By applying overfitting control (indicated in orange), the model's accuracy increased to 86.74%, and its F1 score also improved slightly to 0.926. These changes demonstrate that controlling overfitting helps the model better capture meaningful patterns in the data, reducing error and enhancing generalizability.

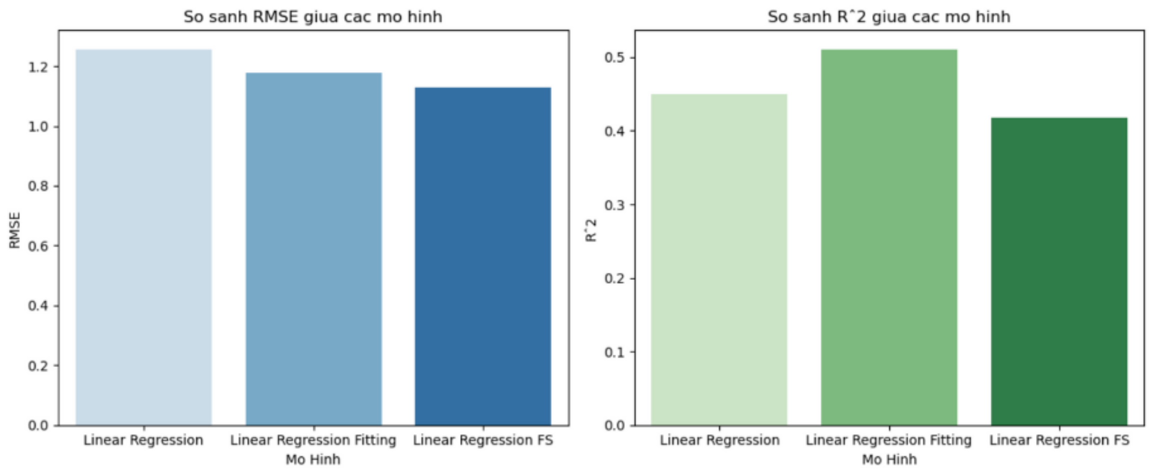
The analysis reveals that feature selection combined with overfitting control yields the best performance, with increased accuracy and stability.

5.5 Linear Regression with Feature Selection



	balance	duration	age	pdays	campaign	marital_married	marital_single	education_secondary	education_tertiary	education_unknown	default_yes	housing_yes	loan_yes	contact_telephone	contact_unknown	poutcome_other	poutcome_success	poutcome_unknown
balance	1.00	-0.02	0.08	0.01	-0.01	0.02	0.01	-0.08	0.08	0.02	-0.07	-0.05	-0.07	0.03	-0.02	0.00	0.03	-0.03
duration	-0.02	1.00	-0.00	0.01	-0.07	-0.04	0.02	0.02	-0.02	-0.01	-0.01	0.02	-0.00	-0.02	-0.01	0.01	0.05	-0.02
age	0.08	-0.00	1.00	-0.01	-0.01	0.28	-0.41	-0.11	-0.09	0.08	-0.02	-0.19	-0.01	0.18	-0.04	-0.03	0.05	-0.02
pdays	0.01	0.01	-0.01	1.00	-0.09	-0.02	0.02	0.01	0.00	0.00	-0.03	0.12	-0.03	0.02	-0.25	0.38	0.21	-0.87
campaign	-0.01	-0.07	-0.01	-0.09	1.00	0.02	-0.01	-0.02	0.02	-0.02	-0.01	-0.00	0.02	0.03	0.00	-0.03	-0.06	0.12
marital_married	-0.02	-0.04	0.28	-0.02	0.02	1.00	-0.76	0.00	-0.11	0.00	-0.04	0.04	0.03	0.04	0.04	-0.02	0.01	0.02
marital_single	0.01	0.02	-0.41	0.02	-0.01	-0.76	1.00	-0.00	0.12	-0.01	0.01	-0.04	-0.05	-0.04	-0.06	0.02	0.01	-0.03
education_secondary	-0.08	0.02	-0.11	0.01	-0.02	0.00	-0.00	1.00	-0.67	-0.21	0.02	0.11	0.08	-0.02	0.04	0.00	-0.02	0.00
education_tertiary	0.08	-0.02	-0.09	0.00	0.02	-0.11	0.12	-0.67	1.00	-0.14	-0.02	-0.10	-0.04	-0.05	-0.13	0.01	0.02	-0.03
education_unknown	0.02	-0.01	0.08	0.00	-0.02	0.00	-0.01	-0.21	-0.14	1.00	-0.00	-0.05	-0.07	0.03	0.04	-0.00	0.03	-0.01
default_yes	-0.07	-0.01	-0.02	-0.03	-0.01	-0.04	0.01	0.02	-0.02	-0.00	1.00	0.01	0.06	-0.02	0.01	0.01	-0.02	0.04
housing_yes	-0.05	0.02	-0.19	0.12	-0.00	0.04	-0.04	0.11	-0.10	-0.05	0.01	1.00	0.02	-0.08	0.22	0.05	-0.09	-0.06
loan_yes	-0.07	-0.00	-0.01	-0.03	0.02	0.03	-0.05	0.08	-0.04	-0.07	0.06	0.02	1.00	-0.00	-0.01	-0.02	-0.05	0.04
contact_telephone	0.03	-0.02	0.18	0.02	0.03	0.04	-0.04	-0.02	-0.05	0.03	-0.02	-0.08	-0.00	1.00	-0.17	0.03	0.02	-0.04
contact_unknown	-0.02	-0.01	-0.04	-0.25	0.00	0.04	-0.06	0.04	-0.13	0.04	0.01	0.22	-0.01	-0.17	1.00	-0.13	-0.10	0.29
poutcome_other	0.00	0.01	-0.03	0.38	-0.03	-0.02	0.02	0.00	0.01	-0.00	0.01	0.05	-0.02	0.03	-0.13	1.00	-0.04	-0.45
poutcome_success	0.03	0.05	0.05	0.21	-0.06	0.01	0.01	-0.02	0.02	0.03	-0.02	-0.09	-0.05	0.02	-0.10	-0.04	1.00	-0.37
poutcome_unknown	-0.03	-0.02	-0.02	-0.87	0.12	0.02	-0.03	0.00	-0.03	-0.01	0.04	-0.06	0.04	-0.04	0.29	-0.45	-0.37	1.00

We observed some stronger correlations, such as between *pdays* and *poutcome_unknown* (-0.87) and between *pdays* and *previous* (0.38), indicating potential areas for feature selection to address multicollinearity. This analysis supports our approach to refine the model by focusing on features with unique contributions, ultimately aiming to enhance model performance and interpretability.



```
                  Mo Hinh      RMSE       Rˆ2
        Linear Regression  1.256744  0.449705
Linear Regression Fitting  1.178445  0.510961
     Linear Regression FS  1.128228  0.417208
```

- **Before feature selection:** The Linear Regression model initially had an RMSE of 1.257 and an R^2 score of 0.450, capturing around 45% of the variance.
- **After feature selection:** Post-feature selection, RMSE decreased to 1.128, and the R^2 score was 0.417. Although the R^2 slightly decreased, the reduced RMSE suggests a better fit with less noise in the model.

Overall, applying overfitting control provided the best balance between RMSE and R^2, indicating improved fit and stability. Feature selection reduced error further, but at the cost of a slight decrease in variance explained. Thus, for this dataset, overfitting control yielded the most balanced improvement for the Linear Regression model.

CHAPTER 6. CONCLUSION

6.1 Key Learnings

In this project, we explored multiple approaches to solve classification and regression tasks using various machine learning models. For classification, models such as Logistic Regression, K-Nearest Neighbors (KNN), and Decision Tree were applied, each yielding high performance in terms of accuracy, precision, recall, and F1 score.

Logistic Regression and KNN exhibited the most balanced results, while Decision Tree benefitted significantly from overfitting control and feature selection, enhancing its predictive stability and accuracy.

For regression, Linear Regression and Random Forest Regression models were implemented. Through our analysis, we observed that overfitting control improved the performance of both models, with Linear Regression showing notable improvements in RMSE and R^2 after regularization was applied.

Feature selection, on the other hand, reduced the complexity of the models, focusing on the most relevant features, which lowered the error but slightly reduced the variance explained in the regression tasks.

Overfitting analysis across all models demonstrated that regularization techniques, such as limiting tree depth and adding penalty terms, effectively improved model generalization. Feature selection based on correlation analysis helped reduce model complexity by removing redundant features, thereby contributing to more efficient and interpretable models.

6.2 Future Improvement

For future work, several enhancements and additional experiments could be conducted to improve model accuracy and robustness. Some potential directions include:

- **Exploring Additional Features**: Enriching the dataset with additional relevant features may help the models capture more nuanced patterns in the data, potentially improving both classification and regression accuracy.

- **Hyperparameter Tuning**: Further hyperparameter optimization using techniques like grid search or random search could fine-tune the models for even better performance.
- **Alternative Feature Selection Techniques**: Other feature selection methods, such as Recursive Feature Elimination (RFE) or Principal Component Analysis (PCA), could be experimented with to verify if they yield better results than correlation-based selection.
- **Addressing Dataset Limitations**: Finally, if possible, increasing the size and diversity of the dataset would provide more data points, which can enhance model training and reduce potential biases.

In conclusion, our findings demonstrate the effectiveness of model tuning and feature selection in improving machine learning model performance. With further refinements and data enhancements, the models could be optimized to achieve even greater predictive accuracy and generalizability.

REFERENCES

English

[1]. Overfitting, https://www.ibm.com/topics/overfitting

[2]. Feature Selection, https://www.geeksforgeeks.org/feature-selection-techniques-in-machine-learning/

[3]. Correlation Analysis, https://medium.com/@abdallahashraf90x/all-you-need-to-know-about-correlation-for-machine-learning-e249fec292e9