

Machine Learning Model

by Sai Jangili

Submission date: 12-Dec-2024 03:00PM (UTC+0000)

Submission ID: 245786664

File name: 18928_Sai_Jangili_Machine_Learning_Model_621176_1582211261.docx (1.77M)

Word count: 3841

Character count: 21541



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Artificial Intelligence

Machine Learning Model

Sai Harsha Jangili

Student ID: 1908692

Contents

Abstract: 3

Introduction: 3

 Overview:..... 3

Problem Statement: 3

Literature Support: 4

 Strengths and weaknesses of Random Forest and Decision Tree. 7

Methodology: 8

Results and Discussion:: 12

Conclusion: 18

Recommendations:..... 18

YouTube link: 19

Abstract:

With the advancement of technology in recent years, Artificial Intelligence (AI) is playing a significant part in most of the industries. These days, AI is being adopted in every industry, making human life easier and more manageable. AI has become an essential tool in machine learning modules like Decision Tree and Random Forest, which provide the system with the most accurate prediction and classification capabilities.

This project involves building a machine learning model using a banking institution's dataset to predict the term subscription probability of the bank's customers.

Introduction:

Overview:

Artificial Intelligence is a rapidly evolving field in computing. Its main aim is to create a model that can replicate human behaviour at its best. One of the main modules in Artificial Intelligence is Machine Learning, which is presently the most popular approach, which is based on principle learning from given data[1] Decision Tree and Random Forest are two main classification algorithms in the Machine learning module which comes under supervised learning.

Problem Statement:

The main aim of this project is to predict the probability of customers subscribing to bank term deposits. These days, banks like to keep track of their customers by observing their deposits, newly opened accounts, customers applying for loans, and more. This is because when they introduce any new policies, they can have an exact judgment of who possibly subscribes to them. This can be achieved by a prediction table, using machine learning modules, and a dataset of the bank's customers. This project aims to develop a Machine Learning model utilizing Decision Tree and Random Forest algorithms to predict the probability of customers subscribing to term deposits and identify the key factors that are influencing customer's behaviours accurately. The traditional approach may not give accurate predictions, so Decision Trees and Random forests are used to read patterns like these to deliver reliable predictions.

Literature Support:

Artificial Intelligence confines technology and science that develops intelligent machines and computer programs capable of performing tasks requiring human intelligence. It is one of the most popular technological innovations that replace manual labour performed by humans in various fields worldwide. Artificial Intelligence was once simply a concept discussed in science fiction and debates about the impact that it could create on the modern world. Its ability to perform tasks that humans cannot lead to various applications that significantly enhance performance and productivity[2].

Artificial intelligence is the study and development of machines and software smart enough to perform human tasks. It has advantages over natural intelligence as it is more permanent, consistent, and can be easily replicated. All functions performed can be easily documented, and they perform tasks faster and better than a human[3].

Machine Learning techniques are being utilized in various fields, such as computer vision, medicine, airport operations, banking, and other medical applications. This discipline involves varying levels of complexity, requiring regular human interactions at higher levels to train models for improved decision-making accuracy[4].

There are two types of characteristics in Machine learning: supervised learning and unsupervised learning, which are widely followed to train the model. These are widely used in software to help the model get trained and enhance the user experience. Implementing Machine Learning in robots to make them smarter by training them in daily life to adapt to the environment is one of the most used practices. So, it can acquire various skills like object placement, language translation, and other day-to-day work with regular human interaction[5].

Supervised and unsupervised learning are the principal algorithms used in machine learning to train the model. In Supervised learning, the model is given data to train it with both input and desired results. With that given data, the model analyses, verifies and gives the output[6]. In Unsupervised learning, the model addresses the problems with little idea and sometimes even no clue about how the results is going to look like[7].

Here's is an example of Supervised learning.

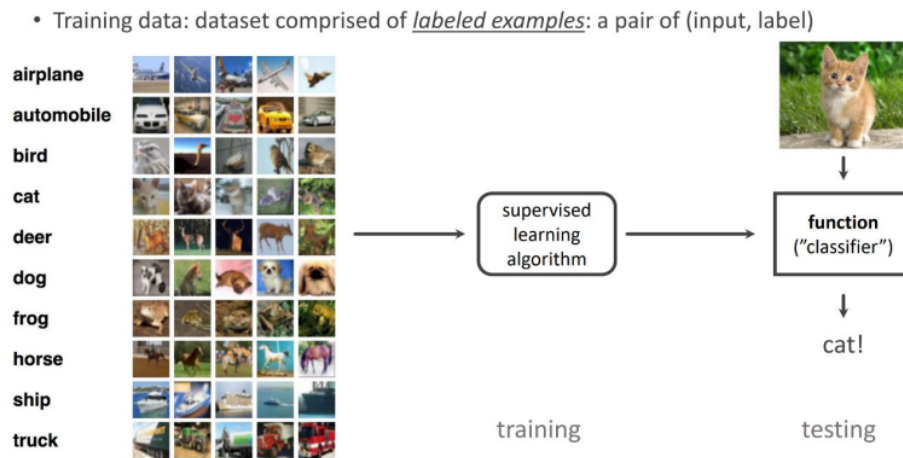


Fig 1. Training the model using Supervised Learning[8].

The above image shows how supervised learning works. A set of data with images from four different variations is provided to train the model. When an image is displayed for classification, the model refers to the training data set provided to verify the features, and classification is done to give an accurate output.

Here's is an example of Unsupervised Learning.

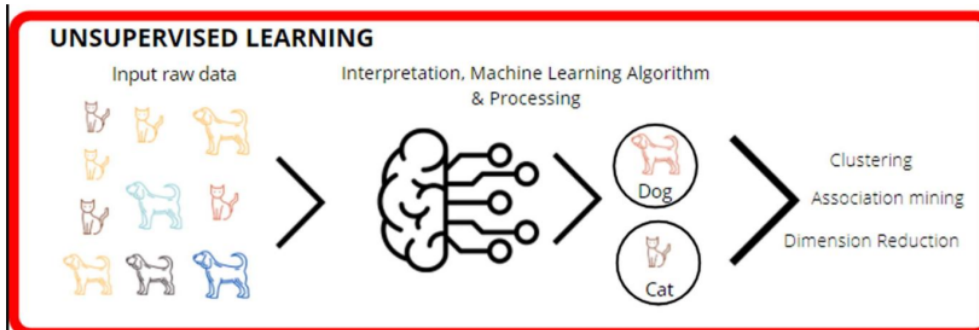


Fig 2. Training the model using Unsupervised Learning[7].

The above image shows how Unsupervised Learning works. The given data is scattered and unstructured. From that input data, the model organizes the clusters. It will train itself by classifying the input data and providing accurate results.

Decision tree is one of the most used methods for classifying data such as statistics, pattern recognition, image classification, text classification, etc [9]. It has various attracting features, such as its ability to handle mixed data, simplicity and ease of understanding, and no parameters. A decision tree is created from a collection of labelled training instances containing tuples of attribute values and their corresponding class labels. Due to the extensive search space, learning a decision tree is typically greedy, proceeding in a top-down manner and using recursion. It starts with the entire training dataset and an empty tree[10]. It is a technique that can produce both classification and Regression trees depending on whether the input variable is categorical or numeric.

A Random Forest is also one of the majorly used models in supervised learning, which has been extremely successful. It associates several random decision trees and then combines their predictions by averaging them. This has shown excellent performance when the number of variables is larger than the number of observations taken, it is adaptable enough to be applied to medium and large-scale problems and can easily be improved to various ad hoc learning tasks and returns measures of mutable importance[13]. Random Forest algorithm estimates the error rate more accurately compared to Decision Tree; it was mathematically proven especially when the number of trees increases[14]. The Random Forest creates a subset of each node of a decision tree, and it then combines the output of individual decision trees and generates the accurate final output. This process of merging the output of multiple individual decision trees is called as Ensemble Learning.

There are studies that show how machine learning models can help predict customer behaviour. Quinlin researched and proposed a Decision Trees, an intuitive model which can work in various domains and can handle both categorical and numerical data. But in complicated datasets, these models frequently needed help with overfitting and sensitivity to noise[15].

But later in 2001, Leo Breiman's development of Random Forests fixed these limitations of decision tree by taking the predictions from multiple decision trees and combining them which resulted in more accurate outputs. Random forests are very effective when larges datasets are given because they generalize the data in a better way and are less liable to overfitting[16].

Here is the table comparing the difference between Decision Tree and Random Forest.

Characteristics	Decision Tree	Random Forest
Nature	Just One Decision Tree	Combination of the results from multiple Decision Trees
Accuracy	Less accurate	High Accuracy
Time required to train	Generally fast as it has only one decision tree	Slow, due to multiple trees
Reliability	Less reliable	More reliable as it takes results from multiple trees
Usability	Perfect for simple tasks	Appropriate for complex tasks and has high dimensional data
Robustness	Sensitive	Robust in nature
Tuning Requirements	Basic tuning required	Demands Hyperparameter tuning
Scalability	Struggles when large dataset is provided	Handles large sets of data very well.
Complexity level	Simple and easy to understand	Complex model and hard to understand
Ability of parallel processing	Less parallel processing as there is only single tree	Does parallel tree construction for increasing efficacy

Strengths and weaknesses of Random Forest and Decision Tree.

Decision Trees:

Strengths:

- Easy to read and understand as it is only a single tree
- Minimum data preparation and processing required
- Easy to validate data by performing various tests
- Less time required and perfect for small programs
- Minimum amount of computer power required

Weakness:

- Can't handle large data sets
- Inclined to overfitting of data
- Few concepts can be hard to understand because the model can't express it clearly
- Might have lower accuracy when large data set is provided

These are strengths and weaknesses of the Decision Tree model[17].

Random Forests:

Strengths:

- Very accurate results provided
- It gives helpful internal information of errors and variable importance
- It can parallelly process multiple trees for quick responses
- Can easily handle large sets of data for complex programmes
- Very reliable

Weakness:

- Very complex so, it is sometimes difficult to understand
- It takes time to process large sets of data hence the output is a bit delayed
- Utilizes more memory
- High power systems are required to process the data
- Parameter tuning required for optimum performance

These are strengths and weaknesses of the Random Forest model[18].

Methodology:

Description:

This section describes the methodology undertaken for developing and implementing Machine Learning models for predicting term deposit subscriptions. The dataset used in this project is taken from the marketing campaign conducted by a Portuguese banking institution. It contains 41188 records and 20 different features, including both numerical and categorical data. It includes two variables, y , and n , representing yes or no, which are used to describe the characteristics of the customer.

Data Exploration and Preprocessing:

This project starts with an in-depth exploration of the dataset. The steps involved in this process are understanding the structure, characteristics, and distribution of the data. Before the data processing the data will be cleaned and pre-processed and if there are any values that are missing in the dataset, they need to be replaced so that there is no information loss.

Some of the key preprocessing steps are:

- Data splitting is used to split the data into training and testing sets in an 80-20 ratio to thoroughly assess model performance on unseen data. Stratified sampling is used to ensure the class distribution is maintained for a balanced and reliable evaluation.
- Data cleaning is the process of appropriately handling missing values to ensure data quality. Imputation techniques are used where necessary, and any rows with substantial missing data are systematically excluded, ensuring that the sample size remains robust.
- Categorical variable encoding is used for effectively encoding using techniques like one-hot encoding or label encoding, chosen strategically based on the nature of the variable and its significance for model interpretability and performance [19].

Data cleaning is not used in this project because there is no missing data in the data set provided.

Exploratory Data Analysis (EDA):

The purpose of EDA is to uncover the patterns and insights in the data. This is one of the essential steps in the analysis. It is the first step, followed by data collection and preprocessing. The primary aim of EDA is to examine the data for distribution. EDA provides tools for hypothesis generation by visualizing and understanding the data through graphical representation. The data is effectively visualized, plotted, and manipulated without assumptions, enabling a comprehensive evaluation of data quality and the development of effective models[19].

Model Selection: Decision Tree and Random Forest Classifiers:

Decision Tree and Random Forest Classifiers are used because they are very effective and interpret well while handling mixed data types and has a high classification capability.

The Decision Tree is an excellent choice due to its simplicity and clarity. It offers straightforward visual representation of decision paths and efficiently focusses critical features in the decision-making process. While it does have limitations, particularly with large datasets where it may tend to overfit, its instinctive nature and ease of understanding make it a powerful tool in many situations[9].

The Random Forest is an excellent choice because it is an ensemble in nature. It effectively handles the overfitting problem that often occurs in Decision Trees by combining the outcomes of multiple trees. This approach significantly enhances its precision and robustness, making it a powerful tool in predictive[14].

Hyperparameter Tuning:

Hyperparameter Tuning is done to enhance the performance of both Decision Trees and Random Forest models.

In Decision Trees, the Hyperparameter tuning is used for various reasons, a few of them are:

- To minimize the number of samples in the leaf node
- Gini index and entropy were tested to effectively optimize the gain of information during the splits
- The depth of the tree is controlled to make sure that there is no overfitting
- Determined the minimum number of samples necessary for effectively splitting an internal node.

```
param_grid = {  
    'criterion': ['gini', 'entropy'],  
    'max_depth': [None, 10, 20, 30],  
    'min_samples_split': [2, 5, 10],  
    'min_samples_leaf': [1, 2, 4]  
}
```

In Random Forests as well, the Hyperparameter tuning is used for various reasons, a few of them are:

- Enabled or disabled bootstrapping of the samples for building trees

- Used for reproducibility of results
- Same as Decision Tree Gini index and entropy were tested to effectively optimize information
- Adjusted the number of trees in the Random Forest to optimize performance while effectively managing computational complexity.
- Specified the number of features considering the splitting at each node.

```
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2']
}
```

Grid Search and Random Search techniques were implemented for effective hyperparameter tuning, ensuring a systematic approach. Cross validation was implemented to assure the robust performance metrics across different data splits.

Development and Training:

Both Decision tree and random forest models were trained using the pre-processed training dataset.

The process of training involved:

- In Decision trees, the first single tree was constructed by continuously splitting the data at every tree node. Based on the feature of the node, it either maximized information gain or minimized impurity.
- In Random Forest an ensemble of Decision Trees was effectively trained on multiple subsets of data and features, with the final prediction confidently derived from the aggregated votes of individual trees.

Interpretation and Comparison

- The significant features were visualized using the results of bar plots, acknowledging their relative importance
- Discussed the impact of these features on model decisions and their real-world implications

- Analysed feature importance scores from two models to identify key factors influencing term deposit subscriptions

Performance metrics

- In-depth analysis of confusion matrices to understand Distribution of predictions across classes
- Observation of model behaviour: Decision Trees showed a tendency to overfit whereas Random Forests demonstrated more reliable performance
- Both the models were compared using metrics like accuracy, complexity, recall, F1 score.

This methodology provided a robust framework for understanding, developing, and comparing machine learning models aimed at predicting term deposit subscriptions. The analysis delivered significant insights into the dataset and model performance, empowering data-driven decision-making.

Results and Discussion:

The image below shows the required libraries imported for developing our model.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#Data Cleaning
from sklearn.preprocessing import StandardScaler, MinMaxScaler, LabelEncoder, OneHotEncoder

#Classification Algorithms
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV, RandomizedSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, roc_auc_score, roc_curve, precision_score, recall_score, f1_score
```

Fig 3. Importing libraries.

The primary step in coding is to import all the libraries including pandas which is used for analysing, cleaning, exploring, and enabling data manipulation. NumPy is used for mainly dealing with arrays, but NumPy also has few extra data types like integers. Matplotlib and seaborn are used for presenting the dataset.

```
# Reading CSV file
df = pd.read_csv('bank_final.csv', sep=";")
df.head(5)
```

Fig 4. Reading the Dataset file.

At this step the dataset file's path is attached, loaded and read using pandas, with command (pd.read_csv).

```
# Descriptive Statistics of the Dataset
df.describe()
```

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
count	41188.00000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000
mean	40.02406	258.285010	2.567593	962.475454	0.172963	0.081886	93.575664	-40.502600	3.621291	5167.035911
std	10.42125	259.279249	2.770014	186.910907	0.494901	1.570960	0.578840	4.628198	1.734447	72.251528
min	17.00000	0.000000	1.000000	0.000000	0.000000	-3.400000	92.201000	-50.800000	0.634000	4963.600000
25%	32.00000	102.000000	1.000000	999.000000	0.000000	-1.800000	93.075000	-42.700000	1.344000	5099.100000
50%	38.00000	180.000000	2.000000	999.000000	0.000000	1.100000	93.749000	-41.800000	4.857000	5191.000000
75%	47.00000	319.000000	3.000000	999.000000	0.000000	1.400000	93.994000	-36.400000	4.961000	5228.100000
max	98.00000	4918.000000	56.000000	999.000000	7.000000	1.400000	94.767000	-26.900000	5.045000	5228.100000

Fig.5 Preprocessing

The above image shows the preprocessing of the data provided. It shows a portion of few rows from the Dataset to give a quick view of structure of data.

```
#Univariate Analysis for Categorical Features

categorical_features = df.select_dtypes(include=['object']).columns

# Calculate the number of rows and columns for the subplot grid
num_features = len(categorical_features)
num_cols = 3 # Number of columns in the grid
num_rows = int(np.ceil(num_features / num_cols)) # Calculate rows needed

plt.figure(figsize=(15, 5 * num_rows)) # Adjust figure height based on rows
for i, feature in enumerate(categorical_features):
    plt.subplot(num_rows, num_cols, i + 1) # Use calculated rows and columns
    sns.countplot(x=feature, data=df)
    plt.title(feature)
    plt.xticks(rotation=45, ha='right') # Rotate x-axis labels if needed
plt.tight_layout()

plt.show()
```

Fig.6 Exploratory Data Analysis

The above image shows the initialization of Exploratory Data Analysis (EDA). This process is done for distribution of each categorical feature in the dataset. It is a single dominant category, so it is imbalanced. The pattern here informs feature engineering or model preparation steps.

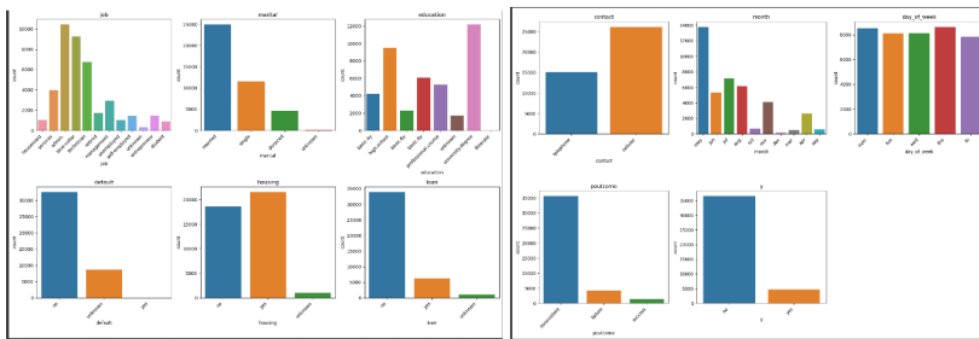


Fig 7 & 8 Results for EDA.

The above two images show the results of Exploratory Data Analysis. It is a set of bar graphs that displays the results of various categories of people who are customers in the bank.

```
#Univariate Analysis for Categorical Features

categorical_features = df.select_dtypes(include=['object']).columns

# Calculate the number of rows and columns for the subplot grid
num_features = len(categorical_features)
num_cols = 3 # Number of columns in the grid
num_rows = int(np.ceil(num_features / num_cols)) # Calculate rows needed

plt.figure(figsize=(15, 5 * num_rows)) # Adjust figure height based on rows
for i, feature in enumerate(categorical_features):
    plt.subplot(num_rows, num_cols, i + 1) # Use calculated rows and columns
    sns.countplot(x=feature, data=df)
    plt.title(feature)
    plt.xticks(rotation=45, ha='right') # Rotate x-axis labels if needed
    plt.tight_layout()

plt.show()
```

Fig 9. Analysis of Numerical Features.

The above image shows the code used to perform Univariate Analysis of the numerical features in a dataset by generating histograms with Kernel Density Estimations arcs to envision the division of each numerical features.

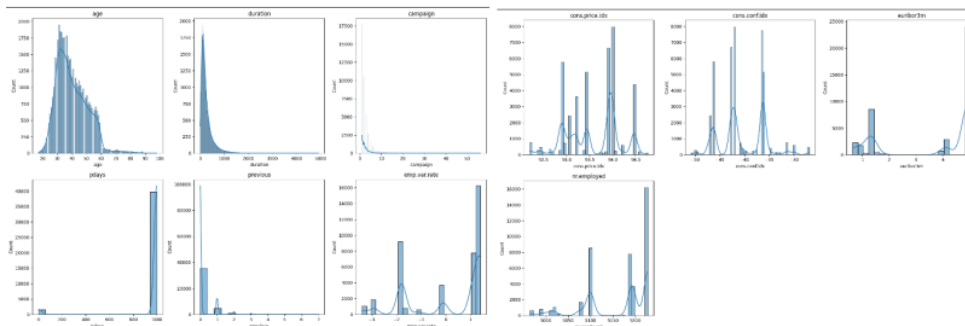


Fig 10 and 11. Histograms of the Numerical Features.

These both images show the Histograms of the numerical features to show the frequency distribution of values.

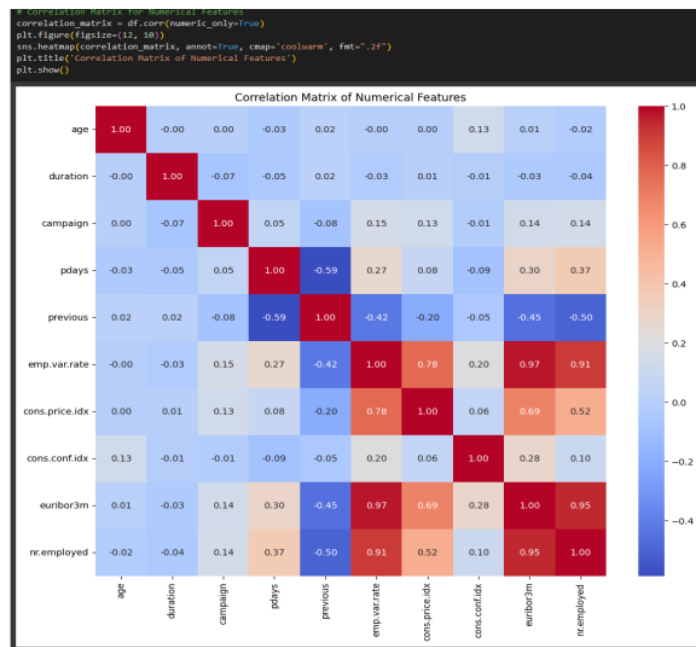


Fig 12. Correlation Heatmap

The above image shows the correlation heatmap that signifies the relationship between the variables in the dataset using the colour coding matrix. It uses colours to show the strength and direction of correlations, with annotations displaying the exact values.

```

# Initialize Decision Tree Classifier
dt_classifier = DecisionTreeClassifier(random_state=42)

# Define hyperparameter grid for tuning
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

# Perform GridSearchCV for hyperparameter tuning
grid_search = GridSearchCV(estimator=dt_classifier, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Get the best hyperparameters and the best model
best_params = grid_search.best_params_
best_dt_classifier = grid_search.best_estimator_

# Train the best model on the entire training set
best_dt_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = best_dt_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
print(f"Best Hyperparameters: {best_params}")

# Further evaluation metrics (optional)
print(classification_report(y_test, y_pred))

```

Fig 13. Implementation of Decision Tree classification

The above image shows the implementation and hyperparameter tuning for decision tree classifier using **GridSearchCV** to identify the best hyperparameters. Model performance is assessed using accuracy and a classification report, which includes precision, recall, and F1-score for each class. This approach ensures that the Decision Tree is optimized for performance while avoiding overfitting.

```

Accuracy: 0.9082078678970374
Best Hyperparameters: {'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 5}

```

	precision	recall	f1-score	support
0	0.94	0.96	0.95	7265
1	0.63	0.53	0.58	971
accuracy			0.91	8236
macro avg	0.79	0.74	0.76	8236
weighted avg	0.90	0.91	0.90	8236

Fig 14. The output of decision tree classification.

The above image shows the results achieved by the Decision tree classifier. It has achieved an accuracy of **90.82%**. The best hyperparameters were:

- **Criterion:** entropy
- **Max Depth:** 10
- **Min Samples Leaf:** 2
- **Min Samples Split:** 5

Considering class wise performance, the class 0 has achieved precision of 94% and recall of 96%. Class 1 has achieved a precision of 63% and low recall of 53% which indicates the difficulty in identifying the minority class. In conclusion the model has performed well for the majority class but struggles handling the minority class due to the imbalances.

```

# Get Feature Importances from the best decision tree model
Feature_Importance = best_01.best_estimator_.feature_importances_

# Create a dictionary to display Feature_Importance
Feature_Importance_dict = dict(zip(X_train.columns, Feature_Importance))

# Sort the dictionary by importance in descending order
Feature_Importance_df = Feature_Importance_dict.items()

# Print or display the Feature_Importance
Feature_Importance_df

```

Feature	Importance
duration	0.449610
nr_employed	0.248507
comm_cost_sib	0.071980
customer	0.057024
age	0.050370
comm_cost_ny	0.033181
postcode_success	0.015086
campaign	0.014778
month_oct	0.012960
polya	0.009541
education_university_degree	0.007671
month_mar	0.007414
day_of_week_mon	0.006949
day_of_week_thu	0.005124
status_married	0.004530
contact_telephone	0.003212
stay_of_week_har	0.002783
education_bachelor	0.002741
previous	0.002704
education_graduate_school	0.002609
job_technician	0.002608
marital_single	0.002326
postcode_notre_dame	0.002298
tax_yes	0.002286
job_research	0.002241
housing_yes	0.002113
job_management	0.001987
day_of_week_wed	0.001740
education_high_school	0.001633

Fig 15. Feature importance of the trained Decision Tree.

The above image shows the feature importances from the trained Decision Tree model to understand which features contribute most to the model's predictions. The `feature_importances_` attribute of the trained `best_dt_classifier` provides a numerical value representing the importance of each feature. The features are sorted in descending order of importance, making it easier to identify the most impactful features. In the output the most important features are `duration` (0.4496) which indicates the length of the last contact with the customer strongly influences whether they subscribe to a term deposit. `nr.employed` (0.2485) indicates the number of employees in the economy and plays a major role in predicting the target.

`cons.conf.idx` (0.0720) and `euribor3m` (0.0570) are moderately important features that indicates the consumer confidence and the three-month Euribor which moderately influences predictions.

Least important features are `education_illiterate`, `month_jun`, and `job_student` have negligible or zero importance, implying they contribute little to the model's decision-making.

```
# Initialize Random Forest Classifier
rf_classifier = RandomForestClassifier(random_state=42)

# Define hyperparameter grid for tuning
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2']
}

# Perform RandomizedSearchCV for hyperparameter tuning
random_search = RandomizedSearchCV(estimator=rf_classifier, param_distributions=param_grid, n_iter=10, cv=5, scoring='accuracy', random_state=42, n_jobs=-1)
random_search.fit(X_train, y_train)

# Get the best hyperparameters and the best model
best_params = random_search.best_params_
best_rf_classifier = random_search.best_estimator_

# Train the best model on the entire training set
best_rf_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = best_rf_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
print(f"Best Hyperparameters: {best_params}")

# Further evaluation metrics (optional)
print(classification_report(y_test, y_pred))
```

Accuracy: 0.9186362311881846
Best Hyperparameters: {'n_estimators': 50, 'min_samples_split': 10, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': None}

	precision	recall	f1-score	support
0	0.93	0.97	0.95	7265
1	0.69	0.44	0.54	971
accuracy			0.91	8236
macro avg	0.81	0.71	0.74	8236
weighted avg	0.90	0.91	0.90	8236

Fig 16. Random Forest Initialisation.

The above image shows the implementation of Random Forest classifier, and hyper parameter tuning. A grid of hyperparameter values is defined by Number

trees, maximum depth of each tree, minimum number of data points required to for splitting the node.

The performance of the model after the parameter tuning has achieved an accuracy of 91.06% which indicates the correct prediction of the target variable whether the customer will subscribe to the term deposit which is approximately 91% of the time.

Conclusion:

Using Decision Tree and Random Forest classifiers the customer's data have been analysed to check whether customer will subscribe to the term deposit or not. Both Decision Tree and Random Forest have identified the probability of the customer subscribing to the policy.

Few of the key findings are:

- Both the classifiers provided the probability of the subscription, highlighting the specific features like duration, deposits, outcomes, etc.
- The Random Forest has provided more accurate assessment as it is ensemble in nature, and reduces bias nature, and collect the data from various trees unlike decision tree.
- The Decision Tree was simple and easy to understand, it was prone to overfitting sometimes while handling the training data.
- The Random Forest overtook Decision Tree in categories like precision recall, accuracy which shows its prediction ability.

Recommendations:

Considering the outcomes using both the models the banks should target marketing with the customers who has higher chances to subscribe to the term deposit. By sending them the custom messages regarding the subscriptions.

Use the outcomes of Random Forest classifier as it was more accurate in terms of providing the probability of the subscription. And banks should also make sure that their data is being regularly updated. Regular fine tuning is necessary to constantly get the accurate prediction values.

YouTube link:

<https://youtu.be/zeU9b-rYVHo?si=52NRhKUNpXHnzBbx>

References

- [1] N. Razavian, F. Knoll and K. J. Geras, "Artificial intelligence explained for nonexperts," *Seminars in musculoskeletal radiology*, 2020.
- [2] F. A. PK, "What is Artificial Intelligence?" *Success is no accident. It is hard work, perseverance, learning, studying, sacrifice and most of all, love of what you are doing or learning to do*, vol. 65, 1984.
- [3] A. Pannu, "Artificial intelligence and its application in different areas," *Artif.Intell.*, vol. 4, no. 10, pp.79–84, 2015.
- [4] I. El Naqa and M. J. Murphy, *What is machine learning?* Springer, 2015.
- [5] J. Bell, "What is machine learning?" *Machine learning and the city: applications in architecture and urban design*, pp.207–216, 2022.
- [6] C. Donalek, "Supervised and unsupervised learning," *Astronomy Colloquia. USA*, 2011.
- [7] I. Kumar, S. P. Singh and Shivam, "Chapter 26 - Machine learning in bioinformatics," in *Bioinformatics*, D. B. Singh and Rajesh Kumar Pathak, Ed., Academic Press, 2022, pp. 443–456. doi: 10.1016/B978-0-323-89775-4.00020-1Available: <https://www.sciencedirect.com/science/article/pii/B9780323897754000201>
- [8] T. M. Mitchell and T. M. Mitchell, *Machine learning*, McGraw-hill New York, 1997.
- [9] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp.20–28, 2021.
- [10] J. Su and H. Zhang, "A fast decision tree learning algorithm," *Aaai*, 2006.

- [11] K. K. Rana, "A survey on decision tree algorithm for classification," *International journal of Engineering development and research*, vol. 2, no. 1, pp.1–5, 2014.
- [12] Y. Freund and L. Mason, "The alternating decision tree learning algorithm," *icml*, 1999.
- [13] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, pp.197–227, 2016.
- [14] M. Schonlau and R. Y. Zou, "The random forest algorithm for statistical learning," *The Stata Journal*, vol. 20, no. 1, pp.3–29, 2020.
- [15] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning.*, vol. 1, no. 1, pp.81–106, 1986.
- [16] Leo Breiman, "Random Forests," *Machine Learning.*, vol. 45, no. 1, pp.5–32, 2001.
- [17] D. C. Asogwa, et al., "Review of Some Machine Learning Techniques for Big Data, Their Uses, Strengths and Weaknesses," 2019.
- [18] LEO BREIMAN, "Machine Learning. Random Forests," vol. 45, no. 1, pp.5–32, 2001.
- [19] M. C. Data, et al., "Exploratory data analysis," *Secondary analysis of electronic health records*, pp.185–203, 2016.

Machine Learning Model

GRADEMARK REPORT

FINAL GRADE

65/100

GENERAL COMMENTS

The code is well documented and the implementation of the code meets the requirements. However, the report is not well presented. It lacks the proper understanding of the models used (Decision Tree and RF). There is no evidence of the theoretical explanation of the models. The report lacks a visual representation of the results and a critical analysis of the results. The results should be presented in figures and tables format rather than using screen shots. The video link provided is not the updated version as it does not display any relevant information related to the given assignment.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

GRADING FORM: ACCA7026 GRADING FORM JAN 25

SAI JANGILI

65

ENGAGEMENT WITH LIT.SKILL [15]

*Ability to critically review and integrate relevant and recent literature [6]
Demonstrating understanding of key concepts and theories from
literature [6] Providing appropriate references and citations [3]*



Limited engagement with literature, mostly descriptive with minimal critical review. It is important to provide the significant information required rather than providing general information. For instance a description of working of supervised models are provided while there is no adequate information provided for Decision Tree and RF.

Good understanding but lacks depth in some areas, especially the theoretical aspects of the models.

10

Mostly accurate referencing with minor inconsistencies.

KNOWLEDGE & UNDERSTANDING [10]

Depth of understanding of Decision Tree and Random Forest algorithms and their application in predictive modelling (5 Marks) Demonstrating the role of hyperparameters in machine learning models and their impact on model performance (3 Marks) Knowledge of evaluation metrics (e.g., accuracy, precision, recall, F1 Score) for classification models (2 Marks)



Basic understanding of the model with significant gaps or unclear explanations.
Provides some information about hyperparameters but lacks depth or detailed explanation.
Mentions metrics but with incomplete explanation and interpretation interpretations.

6

COGNITIVE & INTELLECTUAL [25]

Critical analysis of the performance of the developed models and interpretation of results (8 Marks) Ability to identify and discuss factors influencing customer subscription decisions based on model insights (10 Marks) Evaluation of strengths and limitations of Decision Tree and Random Forest models in the context of term deposit prediction (7 Marks)



Analyzes model performance but lacks thorough comparison or interpretation.
Discusses hyperparameters but lacks depth in explaining their specific effects.
Mentions hyperparameters.

16

Discusses strengths and limitations but lacks depth or supporting examples.

PRACTICAL APPLICATION [40]

*Effective preprocessing of data and explorative data analysis (Eg. handling missing values, encoding categorical variables) (10 Marks)
Application of Decision Tree and Random Forest algorithms to the specific banking dataset (20 Marks) Demonstration of proficiency in model implementation, hyperparameter tuning, and cross-validation (10 Marks)*



Applies appropriate preprocessing techniques, but it lacks proper interpretation and its impacts. Implements both models correctly, applies them effectively to the dataset, but there are some inconsistencies in the results presented. The significant features for the both models are same in the figure provided however the output is different. Uses hyperparameter tuning and cross-validation but lacks optimal implementation.

26

TRANSFERABLE SKILLS [10]

Well organised report (5 Marks) o Clarity and effectiveness in communicating complex technical concepts. o Depth of analysis and understanding demonstrated in exploring predictive modelling techniques and interpreting result. o Alignment of the report content with the initial objectives set for predicting term deposit subscriptions. Demonstrates proficiency in implementing HTML code with well-commented sections and integrating visual presentations effectively (5 Marks)



Report is mostly well-organized but lacks clarity in some sections.

Code is well documented, but the developed models are not displayed in the given code.

7