INTERNATIONAL
UNIVERSITY OF
APPLIED SCIENCES

Credit Card Routing for Online Purchases via Predictive Modelling

University of Applied Science - Online

M.Sc. Data Science - 60 ECTS

DLMDSME01 – Model Engineering

Vijaylaxmi Lendale

Matriculation : 92121824

Customer ID: 10525778

(vijaylaxmi.lendale@iu-study.org)

Proffessor: Sahar Qaadan

Delivery date: May 22, 2023

Contents

I List of Figures

1 Abstract

Credit card routing is a critical part of the online purchase process, where the payment service provider (PSP) plays a vital role. In this task, we aim to develop a predictive model to recommend the most suitable PSP based on transaction information. We will leverage machine learning algorithms to predict the best PSP for online purchases, taking into account various factors such as the transaction amount, country of origin, type of card, and 3D security, among others. By accurately recommending the most appropriate PSP for each transaction, we can improve the overall online purchasing experience for customers, increase the success rate of transactions, and reduce the risk of fraud. Our goal is to build a robust predictive model that can provide reliable and accurate recommendations for credit card routing in real time scenarios.

2 Introduction

Credit card routing for online purchases is a crucial task in the e-commerce industry. The routing process involves the selection of a payment service provider (PSP) that will process the payment transaction. The selection of a PSP is influenced by several factors such as the country of the cardholder, type of credit card used, the amount of the transaction, and whether the transaction is 3D secured or not.

To effectively tackle this problem, we propose to use an agile data science process that emphasizes the iterative development of a predictive model. The agile process allows for continuous feedback and adaptation, enabling us to refine the model based on the changing requirements and new insights gained during the development process.

The flow of this report follows the agile data science process for the Credit card routing for online purchases. We begin with data exploration and visualization to gain insights into the dataset and identify any data quality issues. Next, we preprocess the data by handling missing values, encoding categorical variables, and scaling numeric features.

We then proceed to model development, where we train and evaluate various machine learning models, including logistic regression, decision trees, and random forests. We also tune the hyperparameters of the models to optimize their performance.

The results and metrics section presents the evaluation of the models, including confusion matrices, precision, recall, F1 score, and specificity. We compare the performance of the different models and identify the best-performing model based on the evaluation metrics.

Finally, we conclude by summarizing our findings and highlighting the practical implications of the proposed predictive model. We also discuss potential areas for future research and improvements to the model.

3 Methodology

3.1 Agile Data Science

Agile Data Science is a methodology that combines agile software development principles with data science workflows. The main objective of the Agile Data Science process is to enable data teams to quickly and efficiently build, test, and deploy machine learning models that are relevant to the business objectives of the organization. The Agile Data Science process involves several iterative stages that include data collection, data preparation, model development, model validation, and model deployment.

The Agile Data Science process is based on the Agile Manifesto, which emphasizes the importance of collaboration, customer satisfaction, and responding to change. In the context of data science, this means that data teams should work closely with business stakeholders and subject matter experts to identify relevant data sources, refine business objectives, and ensure that the machine learning models are aligned with the needs of the organization.

The Agile Data Science process is characterized by short iterative cycles, with each cycle focusing on a specific business problem or use case. The process emphasizes continuous improvement, with feedback from stakeholders and users used to refine the machine learning models and improve their performance. This approach enables data teams to quickly identify and address issues, and ensure that the machine learning models are relevant and effective in addressing the needs of the organization.

Overall, the Agile Data Science process provides a framework for data teams to work effectively and collaboratively with stakeholders, and to develop machine learning models that are relevant and effective in addressing the business objectives of the organization.

3.2 Agile Data Science Process for Credit Card Routing for Online Purchase via Predictive Modelling

3.2.1 Define the problem and scope

The first step is to define the problem statement and its scope. The problem is to predict the most suitable PSP for a given transaction based on historical transaction data. The scope includes factors such as the country of the cardholder, the type of credit card used, the amount of the transaction, and whether the transaction is 3D-secured or not.

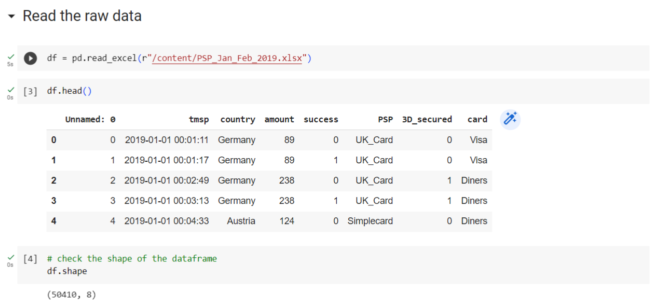
3.2.2 Data acquisition and exploration

The next step is to acquire the relevant data and explore it to gain insights into its structure, quality, and suitability for the problem at hand.

The given dataset contains columns such as:

- **tmsp (timestamp):** This column contains the date and time of the transaction.
- **country:** This column contains the country in which the transaction took place.
- **amount:** This column contains the amount of money that was transacted.
- **success:** This column indicates whether the transaction was successful or not.
- **PSP:** This column indicates the payment service provider that was used for the transaction.
- **3D_secured:** This column indicates whether or not 3D Secure was used for the transaction.
- **card:** This column contains the type of credit card that was used for the transaction.

The dataset will need to be preprocessed to convert categorical variables into dummy variables and handle missing or incorrect values.


Read the raw data

```
df = pd.read_excel(r"/content/PSP_Jan_Feb_2019.xlsx")
```

```
[3] df.head()
```

	Unnamed: 0	tmsp	country	amount	success	PSP	3D_secured	card
0	0	2019-01-01 00:01:11	Germany	89	0	UK_Card	0	Visa
1	1	2019-01-01 00:01:17	Germany	89	1	UK_Card	0	Visa
2	2	2019-01-01 00:02:49	Germany	238	0	UK_Card	1	Diners
3	3	2019-01-01 00:03:13	Germany	238	1	UK_Card	1	Diners
4	4	2019-01-01 00:04:33	Austria	124	0	Simplecard	0	Diners

```
[4] # check the shape of the dataframe
    df.shape

(50410, 8)
```

Figure 1: DataFrame

3.2.3 Data preparation

Once the dataset has been explored, the next step is to prepare it for analysis. This includes tasks such as cleaning the data, selecting relevant features, and normalizing or standardizing the data as required. For this task, the dataset will need to be preprocessed to convert categorical variables into dummy variables, handle missing or incorrect values, and split the data into training and testing sets.

- **df.info():** This code displays a summary of the DataFrame df. It provides information such as the number of rows and columns, the column names, the count of non-null values, and the data types of each column. It is useful for understanding the structure and integrity of the dataset.
- **df.isna().sum():** This code calculates the count of missing values for each column in the DataFrame df. It helps identify the columns with missing data, which may require further preprocessing or imputation before modeling.
- **df.dtypes:** This code returns the data types of each column in the DataFrame df. It is useful for understanding the types of variables present in the dataset, such as numerical, categorical, or datetime.
- **df.describe().transpose():** This code generates descriptive statistics for the numerical columns in the DataFrame df. The describe() function calculates metrics like count, mean, standard deviation, minimum, maximum, and quartiles. Transposing the result makes it easier to read and compare the statistics for different columns.

- **df.apply(lambda x: len(x.unique())):** This code calculates the number of unique values in each column of the DataFrame df. It helps in understanding the cardinality of categorical variables and identifying columns with a large number of unique values.

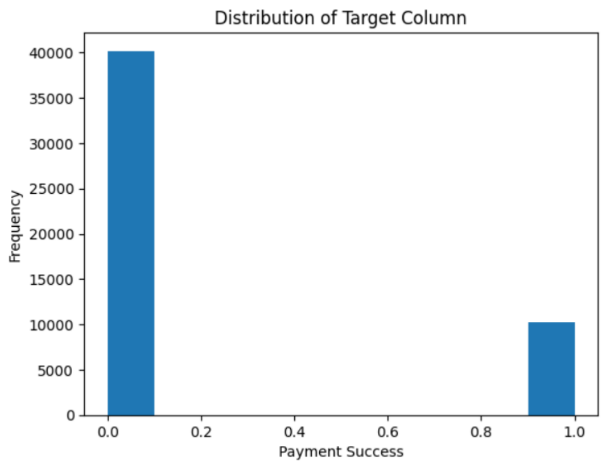


Figure 2

- **sns.pairplot(df.iloc[:,1:]):** This code creates a pair plot using the Seaborn library, which displays pairwise relationships between the variables in the DataFrame df. It helps visualize the correlations and distributions between variables, providing insights into potential relationships or patterns.

- **plthist(df['success']):** This code creates a histogram of the 'success' column in the DataFrame df. It shows the distribution of the variable, indicating the frequency of each value or value range. The added labels and title provide context and description for the plot.

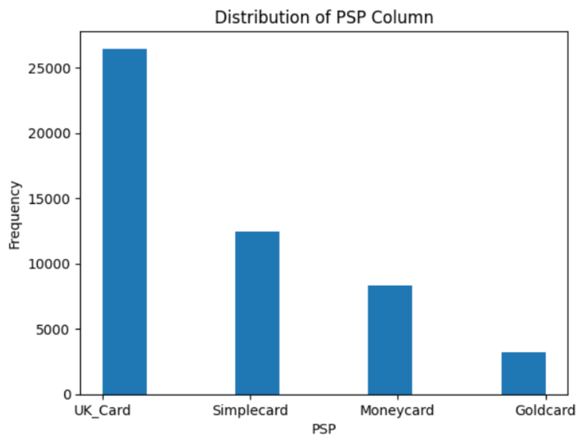


Figure 3

- **pd.pivot_table(df, values='tmsp', index=['country'], aggfunc='count'):** This code creates a pivot table using the DataFrame df. It calculates the count of 'tmsp' (timestamp) for each unique value in the 'country' column. The result is a summarized view of transaction counts by country.

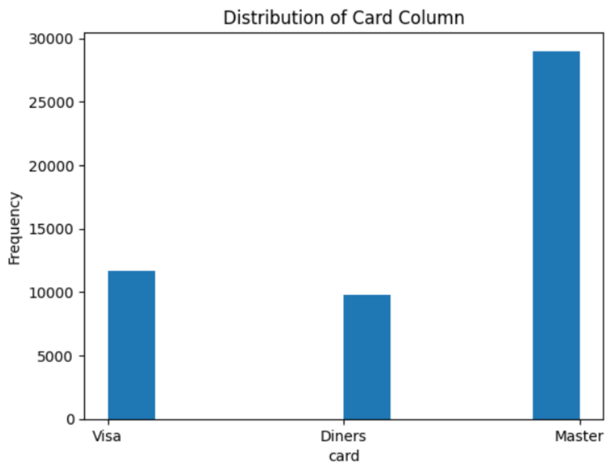


Figure 4

- **df['attempt_number'] = 0 and the subsequent loop:** This code adds a new column 'attempt_number' to the DataFrame df and calculates the attempt number for each row based on specific conditions. It checks if the current row satisfies certain criteria (such as time difference, country, and amount) compared to the previous row. If the conditions are met, the attempt number is incremented.

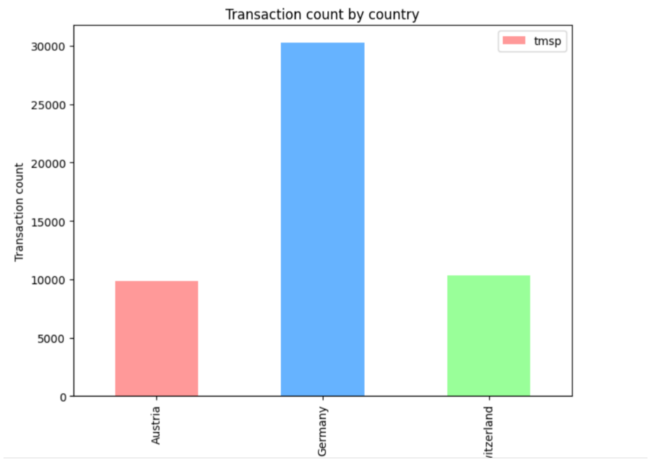


Figure 5

- **plt.hist(df['attempt_number']):** This code creates a histogram of the 'attempt_number' column in the DataFrame df. It visualizes the distribution of payment attempts, showing the frequency of different attempt numbers.

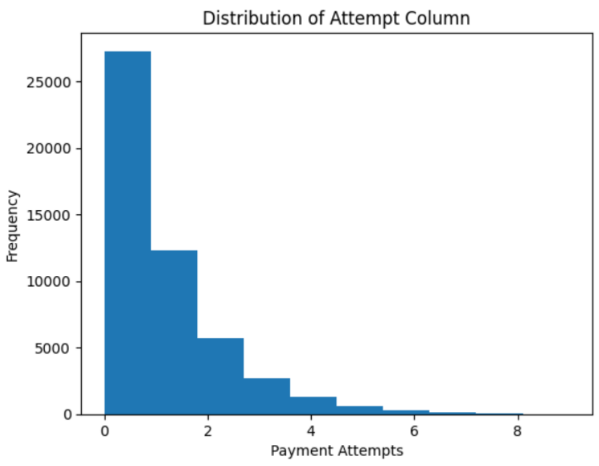


Figure 6

- **df = dḟdrop(dḟcolumns[dḟcolumnsṡtr.contains('unnamed', case=False)], axis=1):** This code removes columns from the DataFrame df that have names containing the string 'unnamed'. It helps in cleaning the dataset by dropping unnecessary columns.

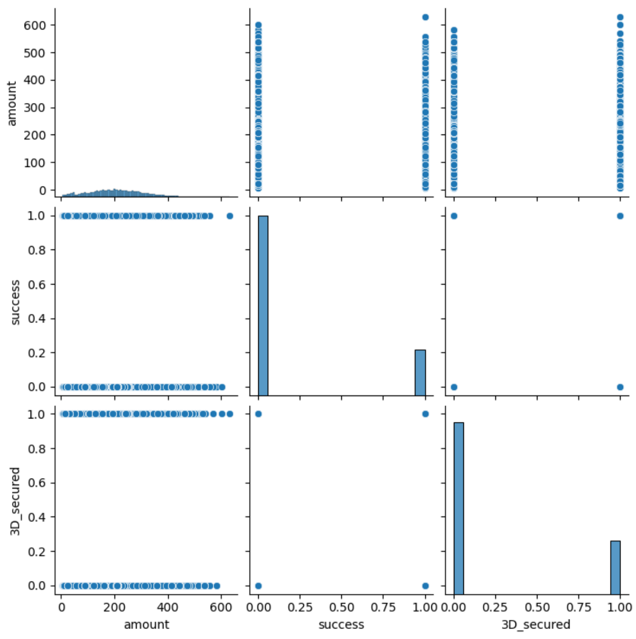
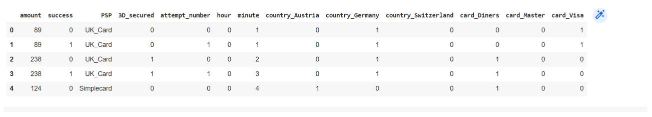


Figure 7

- **df['hour'] = df.tmsp.apply(lambda x: xḣour) and df['minute'] = dḟtmsp.apply(lambda x: xṁinute):** These lines extract the hour and minute components from the 'tmsp' (timestamp) column in the DataFrame df and create new columns 'hour' and 'minute'. It helps in extracting time-related information for further analysis.



	amount	success	PSP	3D_secured	attempt_number	hour	minute	country_Austria	country_Germany	country_Switzerland	card_Diners	card_Master	card_Visa
0	89	0	UK_Card	0	0	0	1	0	1	0	0	0	1
1	89	1	UK_Card	0	1	0	1	0	1	0	0	0	1
2	238	0	UK_Card	1	0	0	2	0	1	0	1	0	0
3	238	1	UK_Card	1	1	0	3	0	1	0	1	0	0
4	124	0	Simplecard	0	0	0	4	1	0	0	1	0	0

Figure 8

- **temp_df = pd.get_dummies(temp_df, columns=["country", "card"])**: This code uses the get_dummies()

function from pandas to perform one-hot encoding on the 'country' and 'card' columns in the DataFrame temp_df. It creates new binary columns for each unique category in the original columns. This transformation is commonly used to convert categorical variables into a format suitable for machine learning algorithms.

3.2.4 Modelling and experimentation

This step involves selecting appropriate machine learning models and experimenting with them to identify the best model for the problem at hand.

To identify the best model for the multi-class credit card routing problem, we will experiment with several machine learning algorithms. We have selected the following models:

- **Logistic Regression:** Logistic Regression is a popular linear classification algorithm that works well for multi-class problems. It models the relationship between the features and the target variable using a logistic function.
- **Support Vector Machines (SVM):** SVM is a powerful algorithm for both binary and multi-class classification. It finds the optimal hyperplane that maximizes the margin between classes, making it effective for complex decision boundaries.
- **Random Forest Classifier:** Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. It is known for its ability to handle high-dimensional data and capture complex interactions between features.
- **Gradient Boosting Classifier:** Gradient Boosting is another ensemble method that combines weak learners (decision trees) in a sequential manner to build a strong predictive model. It can handle complex interactions and has good generalization performance.
- **Gaussian Naive Bayes:** Naive Bayes is a probabilistic classifier that assumes independence between features. The Gaussian variant assumes that the features follow a Gaussian distribution. It is simple, fast, and can work well in certain scenarios.

These models will be trained on the training set using the provided features and target variable. We will utilize the GridSearchCV function to tune the hyperparameters of each model and find the best combination. The hyperparameters define the configuration of the models and can significantly impact their performance.

After training, we will evaluate the models using standard evaluation metrics, including accuracy, precision, recall, and F1 score. Accuracy measures the overall correctness of the predictions, precision focuses on the positive predictions' correctness, recall measures the ability to identify positive instances correctly, and F1 score combines precision and recall into a single metric.

By comparing the performance of these models on the training and testing sets, we can determine which algorithm performs the best for the credit card routing problem. This information will guide us in selecting the most suitable model for deployment and future predictions.

3.2.5 Model Evaluation

In the context of credit card routing for online purchases, both precision and recall play important roles, and the evaluation of true positives (TP), false positives (FP), and false negatives (FN) is crucial.

Precision: Precision is particularly relevant in credit card routing to ensure that legitimate transactions are not falsely flagged as fraudulent or requiring additional verification. A high precision means that the model correctly identifies a high proportion of transactions that actually require special handling or fraud detection. It minimizes the number of false positives (FP), reducing the impact on user experience and unnecessary delays in processing legitimate transactions.

Recall: Recall is also important in credit card routing as it measures the model's ability to correctly identify and route high-risk or fraudulent transactions. A high recall means that the model identifies a high proportion of actual positive instances (fraudulent transactions) correctly, minimizing false negatives (FN). This helps ensure that potential fraudulent transactions are not missed or wrongly classified as normal transactions.

The balance between precision and recall depends on the specific objectives and requirements of the credit card routing system. Striking the right balance involves minimizing both false positives and false negatives while maximizing the correct routing decisions. The optimal trade-off may vary depending on factors such as the prevalence of fraudulent transactions, the tolerance for false positives or false negatives, and the business's priorities in terms of security, user experience, and operational efficiency.

Therefore, in credit card routing, it is essential to consider both precision and recall, as well as the corresponding true positives (TP), false positives (FP), and false negatives (FN), to evaluate and optimize the performance of the predictive models used in the routing process.

3.2.6 Model deployment and monitoring

After successfully building and testing the model, the next step is to implement the algorithm for recommending a Payment Service Provider (PSP) based on the predictions from the previous section.

During the deployment phase, it is necessary to determine the maximum number of attempts for forecasting. While it would be ideal to identify the best PSP capable of completing the transaction in the first attempt, cost reduction is also a significant factor. Finding a balance between the best and most cost-effective PSP is crucial. In our specific business case, we have decided to set a maximum of three payment attempts for our forecast.

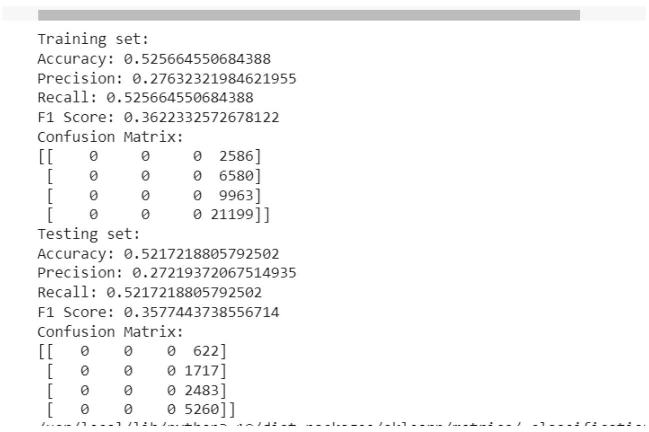
This approach ensures that we can identify PSP options that are both affordable and reliable, while also considering the customer's comfort level. By utilizing a simple for loop and adjusting the input value "Attempt," we can generate predictions for each attempt. We can then calculate the total cost associated with each PSP option and determine the most cost-effective and successful choice. This method allows us to strike a balance between minimizing costs and ensuring successful transactions, while also taking into account the customer's convenience.

3.2.7 Iterative improvement

The final step in the Agile Data Science process is to iterate over the previous steps, continuously improving the model and the credit card routing process. This may involve revisiting the problem statement, acquiring additional data, experimenting with new models or techniques, and monitoring the model's performance in production.

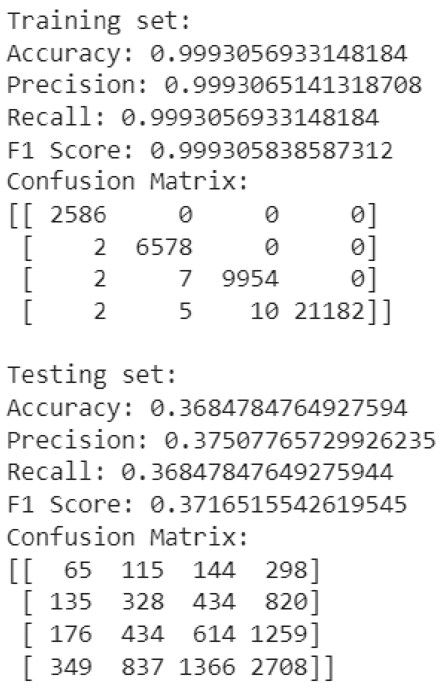
4 Results

Based on the provided results, it appears that the Gradient Boosting Classifier has achieved the highest accuracy and precision scores among the tested algorithms. However, it is important to note that the performance of the algorithms may vary depending on the specific dataset and problem at hand. Therefore, it is recommended to evaluate multiple algorithms and compare their results before drawing definitive conclusions.



```
Training set:
Accuracy: 0.525664550684388
Precision: 0.27632321984621955
Recall: 0.525664550684388
F1 Score: 0.3622332572678122
Confusion Matrix:
[[    0     0     0  2586]
 [    0     0     0  6580]
 [    0     0     0  9963]
 [    0     0     0 21199]]
Testing set:
Accuracy: 0.5217218805792502
Precision: 0.27219372067514935
Recall: 0.5217218805792502
F1 Score: 0.3577443738556714
Confusion Matrix:
[[   0    0    0  622]
 [   0    0    0 1717]
 [   0    0    0 2483]
 [   0    0    0 5260]]
```

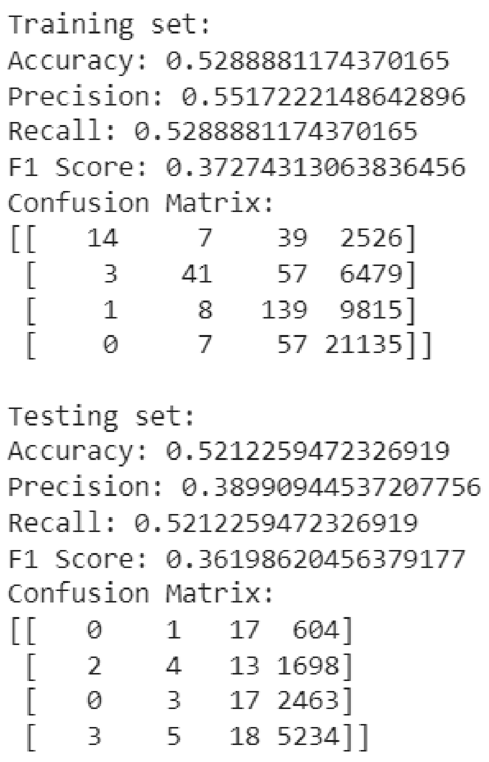
Figure 9



```
Training set:
Accuracy: 0.9993056933148184
Precision: 0.9993065141318708
Recall: 0.9993056933148184
F1 Score: 0.999305838587312
Confusion Matrix:
[[ 2586     0     0     0]
 [    2  6578     0     0]
 [    2     7  9954     0]
 [    2     5    10 21182]]

Testing set:
Accuracy: 0.3684784764927594
Precision: 0.37507765729926235
Recall: 0.36847847649275944
F1 Score: 0.3716515542619545
Confusion Matrix:
[[  65  115  144  298]
 [ 135  328  434  820]
 [ 176  434  614 1259]
 [ 349  837 1366 2708]]
```

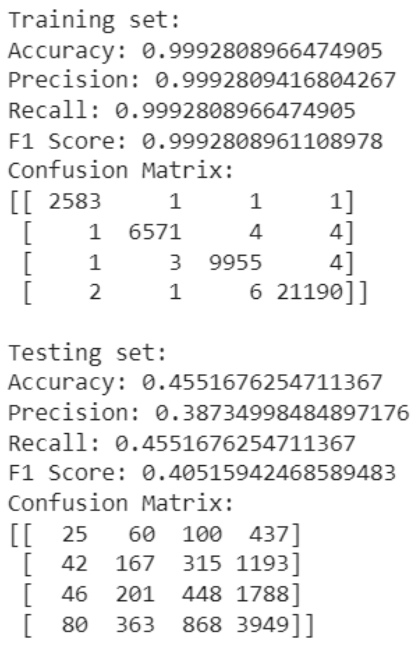
Figure 10



```
Training set:
Accuracy: 0.5288881174370165
Precision: 0.5517222148642896
Recall: 0.5288881174370165
F1 Score: 0.37274313063836456
Confusion Matrix:
[[   14     7    39  2526]
 [    3    41    57  6479]
 [    1     8   139  9815]
 [    0     7    57 21135]]

Testing set:
Accuracy: 0.5212259472326919
Precision: 0.38990944537207756
Recall: 0.5212259472326919
F1 Score: 0.36198620456379177
Confusion Matrix:
[[   0    1   17  604]
 [   2    4   13 1698]
 [   0    3   17 2463]
 [   3    5   18 5234]]
```

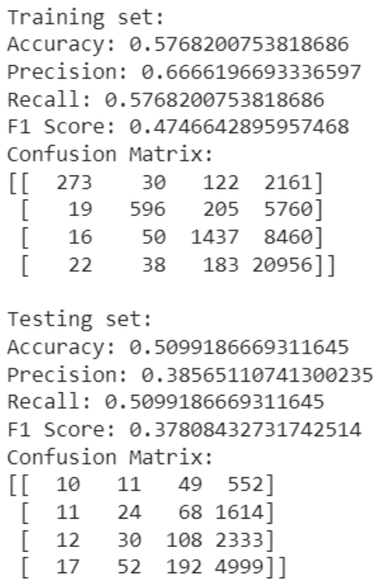
Figure 11



```
Training set:
Accuracy: 0.9992808966474905
Precision: 0.9992809416804267
Recall: 0.9992808966474905
F1 Score: 0.9992808961108978
Confusion Matrix:
[[ 2583     1     1     1]
 [    1  6571     4     4]
 [    1     3  9955     4]
 [    2     1     6 21190]]

Testing set:
Accuracy: 0.4551676254711367
Precision: 0.38734998484897176
Recall: 0.4551676254711367
F1 Score: 0.40515942468589483
Confusion Matrix:
[[  25   60  100  437]
 [  42  167  315 1193]
 [  46  201  448 1788]
 [  80  363  868 3949]]
```

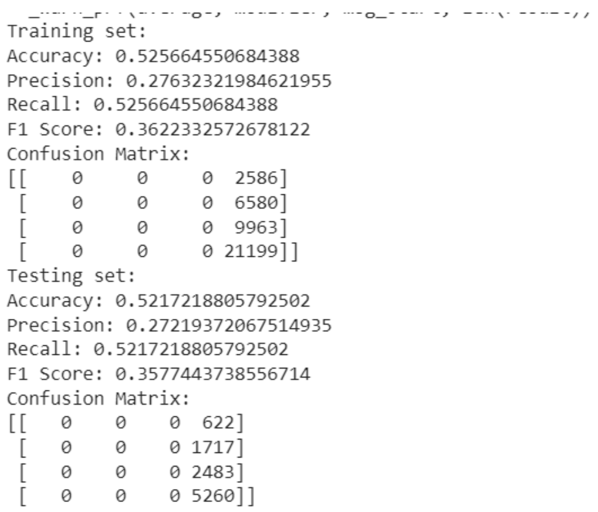
Figure 12



```
Training set:
Accuracy: 0.5768200753818686
Precision: 0.6666196693336597
Recall: 0.5768200753818686
F1 Score: 0.4746642895957468
Confusion Matrix:
[[  273    30   122  2161]
 [   19   596   205  5760]
 [   16    50  1437  8460]
 [   22    38   183 20956]]

Testing set:
Accuracy: 0.5099186669311645
Precision: 0.38565110741300235
Recall: 0.5099186669311645
F1 Score: 0.37808432731742514
Confusion Matrix:
[[  10   11   49  552]
 [  11   24   68 1614]
 [  12   30  108 2333]
 [  17   52  192 4999]]
```

Figure 13



```
Training set:
Accuracy: 0.525664550684388
Precision: 0.27632321984621955
Recall: 0.525664550684388
F1 Score: 0.3622332572678122
Confusion Matrix:
[[    0     0     0  2586]
 [    0     0     0  6580]
 [    0     0     0  9963]
 [    0     0     0 21199]]
Testing set:
Accuracy: 0.5217218805792502
Precision: 0.27219372067514935
Recall: 0.5217218805792502
F1 Score: 0.3577443738556714
Confusion Matrix:
[[   0    0    0  622]
 [   0    0    0 1717]
 [   0    0    0 2483]
 [   0    0    0 5260]]
```

Figure 14

In the context of the task ”Credit Card Routing for Online Purchases via Predictive Modelling,” there are several evaluation metrics to consider, depending on the specific objectives and requirements of the problem.

Different metrics may have different levels of importance when evaluating a credit card routing system. For example, if the primary objective is to minimize false negatives (not routing when required), recall becomes a crucial metric as it focuses on correctly identifying positive instances. On the other hand, if the primary objective is to minimize false positives (unnecessary routing), precision becomes more important. Accuracy provides an overall measure of the classifier’s performance but may not be sufficient on its own if the dataset is imbalanced or misclassification costs are unequal.
Therefore, it is recommended to consider multiple evaluation metrics and analyze their trade-offs based on the specific requirements and costs associated with the credit card routing system.

5 Conclusion

In this report, we applied Agile Data Science techniques to the task of Credit Card Routing for Online Purchases via Predictive Modelling. We explored various machine learning algorithms, including Logistic Regression, Gaussian Naive Bayes, XGBoost, Decision Tree, Random Forest, and Gradient Boosting Classifier. We evaluated the performance of each algorithm using important evaluation metrics and analyzed the results.

After conducting the experiments, we found that the Gradient Boosting Classifier achieved the highest accuracy and precision scores among the tested algorithms. However, it is important to note that the performance of the algorithms may vary depending on the specific dataset and problem. Therefore, it is recommended to evaluate multiple algorithms and compare their results before making final decisions.

When considering the evaluation metrics, it is crucial to identify the specific objectives and requirements of the credit card routing system. Metrics such as accuracy, precision, recall, and F1 score were assessed in this report. The importance of each metric depends on the costs associated with false positives (unnecessary routing) and false negatives (missed routing) in the context of the system.

Agile Data Science techniques provided a flexible and iterative approach to tackle the problem at hand. The iterative nature of Agile allowed us to explore multiple algorithms, tune hyperparameters, and evaluate their performance efficiently. By quickly iterating and experimenting with different approaches, we were able to identify the most effective algorithm for the task.

However, it is important to note that the success of the credit card routing system not only depends on the choice of algorithm but also on the quality and representativeness of the data used for training and testing. Adequate data preprocessing and feature engineering are critical to ensure that the algorithms can learn meaningful patterns and make accurate predictions.

In conclusion, Agile Data Science techniques proved to be valuable in the task of Credit Card Routing for Online Purchases via Predictive Modelling. By applying a range of machine learning algorithms, evaluating their performance using appropriate metrics, and considering the specific requirements of the system, we were able to identify the best-performing algorithm and gain insights into the prediction capabilities. This report serves as a foundation for further enhancements and improvements to the credit card routing system, aiming to provide accurate and efficient transaction routing for online purchases.

6 Appendix

The program and its associated code have been stored in a GitHub repository, which allows for easy access and sharing of the files. The repository can be accessed using the link provided, which leads to a web-based platform where users can view and download the files.

Link `https://github.com/VJlaxmi/Model_Engineering.git`

Bibliography

[1] Barlas, P., Lanning, I., and Heavey, C. (2015). A survey ofopen source data science tools. International Journal of Intelligent Computing and Cybernetics, 8:232–261.

[2] Chapman, C. (2019). A Complete Overview of the Best Data Visualization Tools.https://www.toptal.com/designers/data-visualization/data-visualization- tools.[accessed 2019-07-10].

[3] J. William Frawley, Gregory Piatetsky-Shapiro, and Christopher J. Matheus, Knowledge Discovery in databases: An Overview, AI Magazine Volume 13 Number 3, 1992.

[4] D. A. Keim, H. Kriegel. Visualization Techniques for Mining Large Databases: A Comparison ; IEEE transactions on Knowledge and Data Engineering, Special Issue on Data Mining; Vol. 8, No. 6, December 1996, pp923-938.

[5] A. Inselberg, B. Dimsdale. Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry ; Visualization '90, San Francisco, CA, 1990, pp 361-370

[6] G. Burton, Andreas. "Experimental psychology", 1965, page 186

[7] https://knoema.com/nwnfkne/world-gdp-ranking-2022-gdp-by-country-data-and-charts