Enhancing Customer Insights and Personalization Through Classification Algorithms in Retail

Nguyen T. Nhan
IS252.O11.TMCL
University of Information Technology
19521940@gm.uit.edu.vn

Pham H. Hung
IS252.O11.TMCL
University of Information Technology
20521372@gm.uit.edu.vn

Duong N. Minh
IS252.O11.TMCL
University of Information Technology
20520242@gm.uit.edu.vn

Nguyen H. N. Hanh
IS252.O11.TMCL
University of Information Technology
20521287@gm.uit.edu.vn

***Abstract* —** In the fast-paced world of retail, understanding what customers want is key to growing a business and offering them a personalized shopping experience. Today, with so many ways for customers to shop and choose products, businesses need smarter tools to keep up. The paper introduces new computer techniques to help with this challenge. While older methods might not capture everything about a customer, this paper suggests new approaches, like Decision Trees and Random Forest, that can provide deeper insights. By analyzing detailed data, such as age or shopping history, these techniques can reveal more about customer preferences. The main idea is that using these advanced methods will allow businesses to better understand their customers, tailor their shopping experience, and adapt to future trends more effectively.

Keywords: Personalized shopping experience, Data analysis, Customer preferences, Business growth, Future trends.

I. INTRODUCTION

Retail is a dynamic industry that hinges on understanding customer behavior. As the lines between in-store and online shopping blur, businesses are leveraging vast data to enhance customer experiences. Unlike the past, when success meant great products and brand image, retailers now aim to deepen customer connections beyond simple transactions. Today, it's about offering tailored experiences, and that's where classification algorithms enter the conversation.

Why is this so crucial? Because in today's competitive market, understanding isn't just about selling more products. It's about building lasting relationships with customers. It's about anticipating their needs before they even realize them. In the following sections, we'll break down how these tools work, why they matter, and how businesses can use them to shape a brighter, more connected retail future. Through this exploration, we aim to offer a clear roadmap for businesses looking to elevate their game, ensuring they not only survive but thrive in today's challenging retail environment.

This paper dives into how businesses can better navigate this new retail landscape. We'll be looking at classification algorithms, particularly focusing on techniques like Decision Trees, Random Forest, LightGBM, Support Vector Machine. These aren't just fancy tech terms. They represent powerful tools that can help businesses sift through data, find patterns, and make smarter decisions. By using these algorithms, retailers can get a deeper understanding of who their customers are and what they truly want.

To facilitate this exploration, we will utilize a dataset comprising essential customer information, including unique identifiers, birth years, education levels, marital status, household incomes, household composition (number of children and teenagers), enrollment dates, recency of purchases, and specific spending behaviors such as amounts spent on wine (MntWines). This dataset will serve as the foundation for our analysis and discussions in the subsequent sections, enabling a more concrete understanding of how classification algorithms can revolutionize customer engagement strategies and drive success in today's dynamic retail environment.

II. RELATED WORK

In the realm of machine learning and data analytics, several pivotal studies have made significant contributions.

The article "Applying decision tree models to SMEs: A statistics-based model for customer relationship management" *****[1] Ayad Hendalianpour, Jafar Razmi, and Arefe Rameshi Sarvestani delve into the development of a decision tree-based model tailored for Small and Medium-sized Enterprises (SMEs) to enhance customer relationship management. The model stands out for its focus on rule extraction from monotonic data, a critical aspect in the domains of machine learning and decision analysis. The study extensively evaluates various measures like Gini, chi-square, and Gain ratio for decision tree construction, concluding that Shannon's entropy, despite its effectiveness in many tasks, struggles to reflect the ordinal structure in monotonic classification. The overall accuracy depends almost exclusively on the amount of noise and residual variation in the data and on the degree of pruning, not on the type of measure used. With these domains, it ranges from 7% to 37% error rates. Pruning yields increases in

accuracy of between 19% and 25%, except for theDigit data, for which there was no improvement.

Support vector machine (SVM) is on the basis of statistical learning theory by Vapnik et al proposed a new learning method, which is built on the basis of a limited number of samples in the information contained in the existing training text to get the best classification results Initially designed for class classification problems, SVM has evolved to address multi-class classification, propelled by advancements in computer, network, and database technologies. The study emphasizes SVM's proficiency in dealing with small samples, nonlinear issues, high dimensionality, and local minimal, marking it as a versatile tool in machine learning

In the paper "Application of Random Forest Algorithm on Feature Subset Selection and Classification and Regression" by Jitendra Kumar Jaiswal and Rita Samikannu [2] applies the random forest algorithm to feature subset selection, classification, and regression. The study emphasizes the algorithm's efficiency in handling datasets with a large number of variables. It underscores the importance of feature subset selection in improving prediction performance and presents random forest as a robust solution for managing missing data, classification, regression, and noisy data challenges.

In order to improve the accuracy and efficiency of airborne LiDAR point cloud data classification algorithm, a classification algorithm of point cloud based on LightGBM was proposed, and the classification effect of the algorithm on urban point cloud data was tested. In the article by Xiaosong Shi, Yinglei Cheng and Doudou Xue on "Classification Algorithm of Urban Point Cloud Data based on LightGBM"*[3], The algorithm employs a two-step process using LightGBM classifiers, effectively handling the classification of complex urban point cloud data. The study demonstrates the algorithm's superior performance in terms of accuracy and efficiency compared to other methods, marking a significant advancement in the field of LiDAR data analysis. Accuracy relies on data noise, residual variation, and pruning; error rates range from 7% to 37% across domains. Pruning generally boosts accuracy by 19% to 25%, except for Digit data, where no improvement is seen.

III. MODELING/AI

A. *Decision tree*

A decision tree is a supervised learning algorithm that uses a tree-like model to illustrate decision processes through branches that lead to outcomes or predictions. It's used for classification and regression tasks in machine learning, employing simple decision rules inferred from data features.

[4] The structure consists of a root node (the starting point), internal nodes (where decisions are made), branches (representing decision outcomes), and leaf nodes (depicting final choices or predictions). Decision trees offer an interpretable model that traces a clear path from features to conclusion.

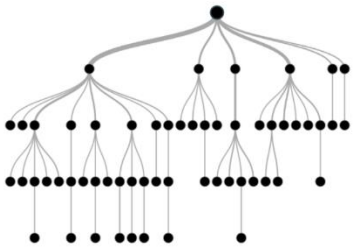


Figure 1. Decision Tree

Key Terms in Decision Trees:

- Root Node: This is where the decision tree starts, splitting the dataset based on features.
- Decision Nodes: These nodes reflect conditions that lead to further branches in the tree.
- Leaf Nodes: Endpoints of the tree indicating outcomes or final decisions, also called terminal nodes.
- Sub-Tree: A smaller portion of the decision tree, representing a path from a node to leaf nodes.
- Pruning: The act of trimming the tree to reduce complexity and prevent overfitting.
- Parent and Child Nodes: A node that splits into other nodes (parent) and the resulting nodes (children), forming the decision pathways.

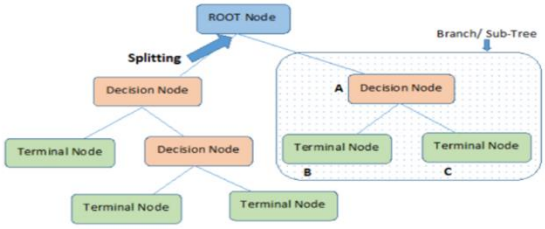


Figure 2. Decision Tree demonstration

Entropy:

Entropy measures disorder in dataset divisions, reflecting data randomness within a node. Pure splits yield homogeneous classes, while impure nodes need further division until achieving 100% purity as leaf nodes. The Entropy formula assesses this impurity, with higher values indicating less purity and a more balanced mix of classes in a node. w

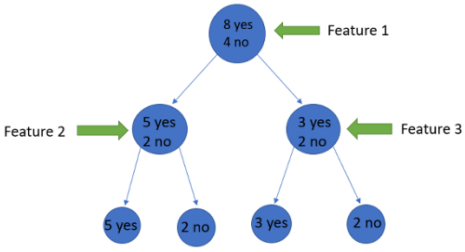


Figure 3. Decision Tree Entropy

Each step of the decision tree will be turned off as follows:

1. Choose the initial dataset with the feature and target attributes defined.

2. Calculate the Information gain and Entropy for each attribute.
3. Pick the attribute with the highest information gain, and make it the decision root node.
4. Calculate the information gain for the remaining attributes.
5. Create recurring child nodes by starting splitting at the decision node (i.e for various values of the decision node, create separate child nodes).
6. Repeat this process until all the attributes are covered.
7. Prune the Tree to prevent overfitting.

B. Random forest

"Random forest" is an algorithm developed by Breiman and Cutler in 2001. It runs by constructing multiple decision trees while training and outputing the class that is the mode of the classes output by individual trees. It has improved performance over single decision trees, and it is much more efficient than traditional machine learning techniques, e. g. artificial neural networks, especially when the dataset is large.

A random forest is an ensemble of unpruned decision trees. Random forests are often used when we have very large training datasets and a very large number of input variables (hundreds or even thousands of input variables). A random forest model is a classifier that consists of many decision trees and outputs the class that is the mode of the class output by individual trees (Breiman 2001).

Random forest is a supervised learning algorithm that can solve both regression and classification problems. In it, I will build many decision trees using the Decision Tree algorithm, but each decision tree will be different (with a random element). Then the prediction results are compiled from the decision trees.

Idea: The model is trained based on a combination of association rules (ensembling) and iterative sampling process (boostrapping).

Each tree is constructed using the following algorithm:

1. Randomly select n data from the data set with Bootstrapping technique.
2. After sampling n data from step 1, I randomly select k attributes (k < n). Now I have a new dataset consisting of n data and each data has k attributes.
3. Use the Decision Tree algorithm to build a decision tree with the data set in step 2.
4. Perform voting or averaging among decision trees to make forecasts.

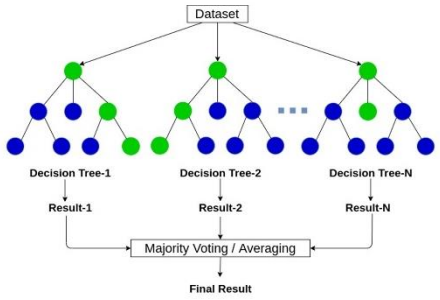


Figure 4. Boostrapping in Random Forest

C. Light GBM

The light gbm algorithm is a gradient boosting framework based on decision trees that aims to increase the efficiency and scalability of machine learning models. It uses two novel techniques: Gradient-based One Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). These techniques reduce the computational cost and memory usage of the algorithm, while maintaining a high level of accuracy and performance [5]

Figure 5. LightGBM

GOSS (Gradient-based One-Side Sampling) is a technique in LightGBM that prioritizes training on data with the largest gradients, which typically need more learning, while randomly reducing less informative data with smaller gradients.

EFB (Exclusive Feature Bundling) decreases the dataset's dimensionality by combining features that are rarely active at the same time, simplifying the model, particularly for sparse data like one-hot encoded features.

LightGBM builds trees by expanding one leaf at a time based on the best gain (leaf-wise growth) rather than level-wise like XGBoost. This approach can lead to overfitting on small datasets, but limiting tree depth can help. Finally, LightGBM's histogram-based strategy bins data for efficiency, which is also well-suited for sparse datasets.

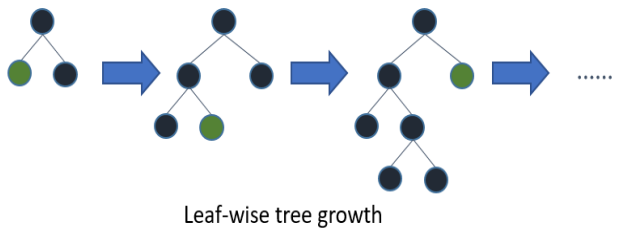


Figure 6. Leaf-wise instead of Level-wise tree growth in LightGBM

The LightGBM algorithm is versatile and can be applied to a range of machine learning tasks, including ranking, classification, and regression. It often surpasses similar gradient boosting frameworks like XGBoost in efficiency and performance due to its speed and lighter memory usage.

Our implementation of the GOSS (Gradient-based One-Side Sampling) technique works as follows: we sort the training instances by the absolute values of their gradients, descending. We retain the top 'a' percent of instances with the largest gradients as subset A. Then we randomly sample 'b' percent of the remaining instances (which have smaller gradients) toThe LightGBM algorithm is versatile, used for ranking, classification, and regression tasks, often outpacing alternatives like XGBoost in speed, memory use, and performance. GOSS, a method within LightGBM, works by first ranking instances by the magnitudes of their gradients. The top a% with the largest gradients are retained as subset A. From the remaining instances, a random sample of size b % is taken as subset B. The tree splitting is then carried out based on the variance gain calculated on the combined subset A∪B, allowing LightGBM to focus learning on the most significant instances while maintaining a representative sample distribution.

$$\tilde{V}_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i \in A_l} g_i + \frac{1-a}{b}\sum_{x_i \in B_l} g_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i \in A_r} g_i + \frac{1-a}{b}\sum_{x_i \in B_r} g_i\right)^2}{n_r^j(d)}\right),$$

Where:

- $\tilde{V}_j(d)$: is the variance gain for feature j when we consider a split on feature value d in a subset O of the training data.
- n: is the total number of instances in the subset O.
- g_i: is the gradient for instance i with respect to the loss function.
- $n_l^j(d)$ and $n_r^j(d)$: are the number of instances in the left and right subsets that result from splitting the set O at the threshold d on feature j.
- A_l and A_r are the subsets of data instances with the largest gradients.
- B_l and B_r are the subsets of data instances that are randomly selected from the remaining data instances.
- The coefficient b1−a is used to normalize the sum of gradients.

The flow of LightGBM can be summarized as follows:

- Data Handling: Preprocess and encode data as needed.
- Initialization: Start with an initial guess, like a mean value or log-odds.
- Iterative Learning: For each iteration, LightGBM performs the following steps:

  1. Selecting a subset of data using GOSS.

  2. Applying EFB to create compact feature sets.

  3. Building a new decision tree leaf-wise based on the gradient of the loss function to find the best fit for residuals/errors.

  4. Computing the leaf values that will minimize the loss when added to the current model.

  5. Updating the model by adding the new tree into the ensemble with a learning rate to scale the step towards the optimal solution.

  6. Repeating the process for a specified number of iterations or until a stopping criteria is met (like a maximum number of trees or early stopping if the validation score does not improve).

- Model Output: Produces an ensemble of decision trees for predictions or testing.
- Model Evaluation: Assess model accuracy, precision, recall, F1-score, Confusion Matrix.
- Hyperparameter Tuning: Fine-tune boosting, tree growth, and regularization settings for improved performance..

D. Support Vector Machine (SVM)

Support Vector Machines (SVM) is a supervised machine learning algorithm primarily used for classification tasks. In SVM, we seek a hyperplane in the feature space that maximizes the margin, i.e., the distance between the hyperplane and the nearest data points from each class, known as support vectors.

Key Concepts of SVM:

Hyperplane: SVM looks for a hyperplane to divide the feature space into two separate parts.

Support Vectors: The data points closest to the hyperplane are called "support vectors." These points are crucial for determining the hyperplane.

Margin: The margin is the distance from the hyperplane to the closest support vectors. SVM aims to maximize the margin to ensure the model's robustness.

In the context of Support Vector Machines (SVM) in a two-dimensional space, the formula for the hyperplane can be expressed as follows:

$$w_1.x_1 + w_2.x_2 + b = 0$$

Where:

- w_1 and w_2 are the weight coefficients of the features.
- x_1 and x_2 are the values of the features.
- b is the bias term.[6]

The flow of the Support Vector Machine (SVM) algorithm, particularly in a 2-dimensional space, can be summarized as follows:

- Data handling: SVM starts with input data, handling features appropriately. Preprocessing, such as scaling, may be applied.

- Initialization: The algorithm initializes the model by finding the hyperplane that best separates different classes in the feature space.

- Iterative Learning:

1. Selecting Support Vectors: SVM selects a subset of data points, known as support vectors, that are crucial for defining the hyperplane.

2. Optimizing Hyperplane: SVM aims to find the hyperplane with the maximum margin, which is the distance between the hyperplane and the nearest support vectors.

3. Kernel Trick (Optional): an choose one of the kernel types: linear, polynomial, radial basis function (RBF), sigmoid. However, in a 2-dimensional space, the linear kernel is often the best choice and yields the optimal results.

4. Training the Model: SVM builds the hyperplane by iteratively adjusting its parameters to minimize the classification error.

5. Updating Model: At each iteration, SVM updates the model by adjusting the position of the hyperplane based on the gradient of the loss function.

6. Convergence Check: The process is repeated for a specified number of iterations or until a convergence criteria is met.

- Model Output: At the end of the training process, SVM outputs a model represented by the hyperplane and support vectors.

- Model Evaluation: The performance of the SVM model can be evaluated using metrics such as accuracy, precision, recall, and F1-score for classification problems.

- Hyperparameter Tuning: SVM allows for hyperparameter tuning. Parameters like the regularization parameter (C) and kernel parameters can be adjusted to optimize the model's performance for a specific dataset and task.

IV. METHOD

In order to perform classification in machine learning, it is essential to ensure that the input data is properly formatted and preprocessed.

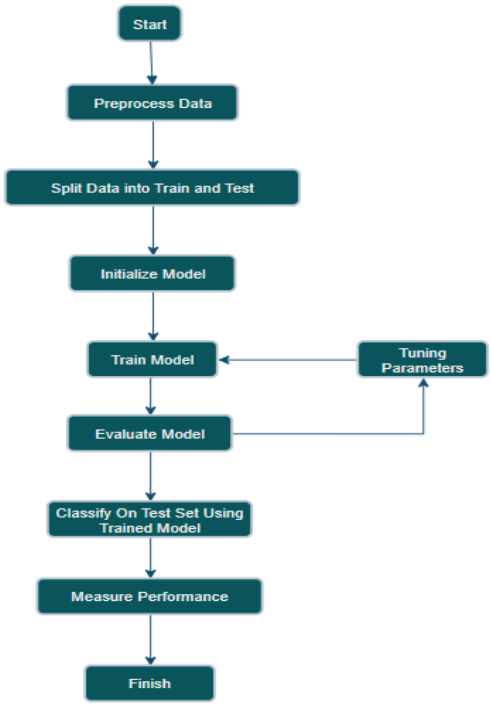


Figure 7. Method used to perform classification

1. Data Preprocessing: Clean the data by handling missing values, removing outliers, and discarding irrelevant features.
2. Data Splitting: Divide the dataset into a training set for model training and a test set for evaluation.
3. Model Initialization: Set up a binary classification algorithm like logistic regression or decision tree.
4. Model Training: Train the model on the training set to learn the relationship between features and the target.
5. Parameter Tuning: Optimize the model's settings through repeated adjustments and training.
6. Model Evaluation: Test the trained model on the test set to measure its generalization capabilities.
7. Classification: Use the model to classify test instances into binary categories.
8. Performance Measurement: Calculate accuracy, precision, recall, and F1 score to determine the model's effectiveness.

This streamlined process focuses on training and tuning a model for binary classification, evaluating its performance, and preparing it for practical application.

V. IMPLEMENTATION

A. Dataset

1) Description

The dataset “Customer Personality Analysis” is a detailed analysis of a company’s ideal customers. It helps a business to better understand its customers and makes it easier for them to

modify products according to the specific needs, behaviors and concerns of different types of customers.

Source of the dataset: [7]

2) Descriptive Statistical

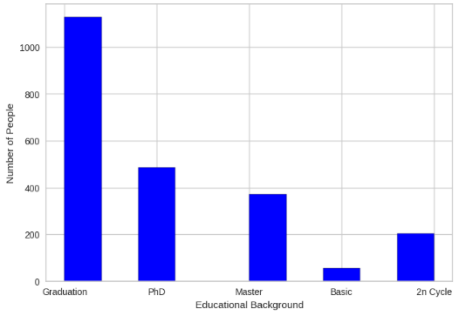
	Min	Max	Mean	Median	Count	Range	Variance	Standard Deviation	Coefficent of variation	Skewness	Kurtosis
Year_Birth	1893	1996	1968.806	1970	2240	103	143.554	11.981	0.006	-0.350	0.717
Income	1730	666666	52247.251	51381.5	2216	664936	633397830.187	25167.396	0.482	6.759	159.637
Kidhome	0	2	0.444	0	2240	2	0.290	0.538	1.212	0.635	-0.780
Teenhome	0	2	0.506	0	2240	2	0.296	0.544	1.075	0.407	-0.986
Recency	0	99	49.109	49	2240	99	838.449	28.956	0.590	-0.002	-1.202
MntWines	0	1493	303.936	173.5	2240	1493	113247.225	336.522	1.107	1.175	0.599
MntFruits	0	199	26.302	8	2240	199	1581.220	39.765	1.512	2.101	4.051
MntMeatProducts	0	1725	166.950	67	2240	1725	50924.685	225.665	1.352	2.082	5.517
MntFishProducts	0	259	37.525	12	2240	259	2982.993	54.617	1.455	1.918	3.096
MntSweetProducts	0	263	27.063	8	2240	263	1703.319	41.271	1.525	2.135	4.377
MntGoldProds	0	362	44.022	24	2240	362	2720.227	52.156	1.185	1.885	3.552
NumDealsPurchases	0	15	2.325	2	2240	15	3.732	1.932	0.831	2.417	8.937
NumWebPurchases	0	27	4.085	4	2240	27	7.718	2.778	0.680	1.382	5.703
NumCatalogPurchases	0	28	2.662	2	2240	28	8.541	2.922	1.098	1.880	8.047
NumStorePurchases	0	13	5.790	5	2240	13	10.564	3.250	0.561	0.702	-0.622
NumWebVisitsMonth	0	20	5.317	6	2240	20	5.886	2.426	0.456	0.208	1.822
AcceptedCmp3	0	1	0.073	0	2240	1	0.067	0.260	3.570	3.289	8.843
AcceptedCmp4	0	1	0.075	0	2240	1	0.069	0.263	3.523	3.239	8.515
AcceptedCmp5	0	1	0.073	0	2240	1	0.067	0.260	3.570	3.289	8.843
AcceptedCmp1	0	1	0.064	0	2240	1	0.060	0.245	3.815	3.553	10.651
AcceptedCmp2	0	1	0.013	0	2240	1	0.013	0.115	8.583	8.466	69.839
Complain	0	1	0.009	0	2240	1	0.009	0.096	10.279	10.182	101.906
Z_CostContact	3	3	3.000	3	2240	0	0.000	0.000	0.000		
Z_Revenue	11	11	11.000	11	2240	0	0.000	0.000	0.000		
Response	0	1	0.149	0	2240	1	0.127	0.356	2.389	1.970	1.889

Figure 8 The original dataset

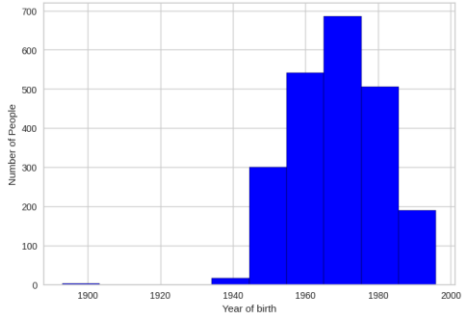
3) Visualization

Here are some visualizations:

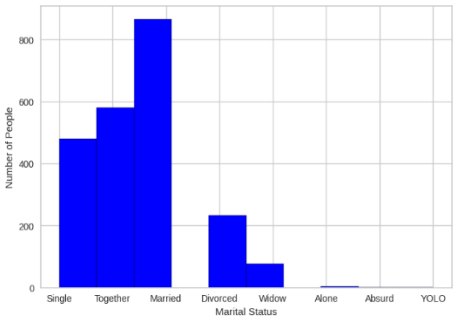
- Education:



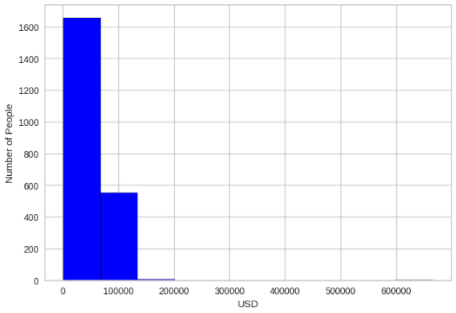
- Year_Birth:



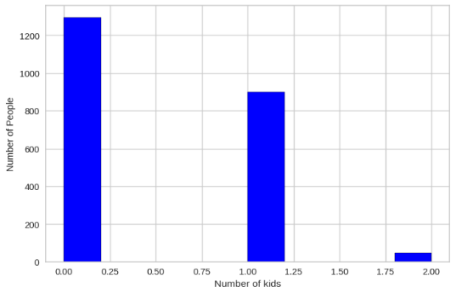
- Martial_Status:



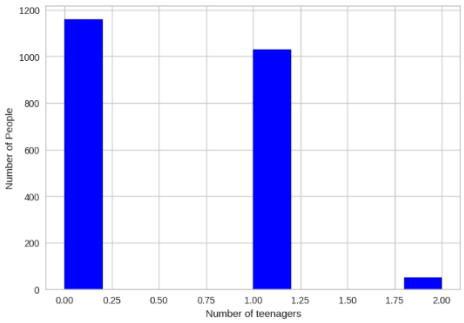
- Income:



- Kidhome:



- Teenhome:



The preprocessing steps are crucial for preparing the dataset for analysis. Here are reasons for preprocessing steps:

- Handling Missing Data:

Delete rows with N/A values to avoid issues with missing data.

- Date Transformation:

Convert "Dt_Customer" to datetime to calculate the time difference from the latest date.

- Age Calculation:

Replace "Year_Birth" with "Age" and handle outliers (remove ages >= 90).

- Total Spending:

Sum spending attributes to create a new attribute, "Spent."

- Marital Status Simplification:

Simplify "Marital_Status" to "Partner" and "Alone," creating "Living_With."

- Children Information:

Sum "Kidhome" and "Teenhome" to create "Children."

- Family Size:

Create "Family_Size" by adding "Children" and "Living_With."

- Parental Status:

Create "Is_Parent" based on whether "Children" is greater than 0.

- Education Grouping:

Simplify "Education" by grouping similar levels.

- Attribute Selection:

Delete unnecessary attributes, keeping only relevant ones.

- Categorical Encoding:

Encode categorical attributes with numerical labels.

- Total Purchases:

Sum purchase attributes to create "Total_Purchases."
Campaign Acceptance:
Create "Total_AcceptedCmp" by summing AcceptCmp attributes.

- Normalization:

Normalize "Recency," "Total_Spending," "Total_Purchases," and "Total_AcceptedCmp."

- Loyalty Score:

Calculate "Loyalty_Score" by subtracting "Complain" from a weighted sum.
Binarize the score using an average threshold.
These steps ensure a clean and transformed dataset, ready for further analysis.

B. *Data Preprocessing*

This procession will be preprocessing the data to perform

```
#Get list of categorical variables
s = (data.dtypes = = 'object')
object_cols = list(s[s].index)

print ("Categorical variables in the dataset:", object_cols)
```

clustering operations.

Steps of preprocess the data:

- Label encoding the categorical features
- Scaling the features using the standard scalerCreating a subset dataframe for dimensionality reduction

Categorical variables in the dataset: ['Education', 'Living_With']

```
#Label Encoding the object dtypes.
LE = LabelEncoder()
for i in object_cols:
    data[i] = data[[i]].apply(LE.fit_transform)

print ("All features are now numerical")
```

All features are now numerical.

```
#Creating a copy of data
ds = data.copy()
# creating a subset of dataframe by dropping the features on deals accepted and promotions
cols_del = ['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1','AcceptedCmp2', 'Complain', 'Response']
ds = ds.drop(cols_del, axis = 1)
#Scaling
scaler = StandardScaler()
scaler.fit(ds)
scaled_ds = pd.DataFrame(scaler.transform(ds),columns = ds.columns )
print ("All features are now scaled")
```

All features are now scaled

```
#Scaled data to be used for reducing the dimensionality
print("Dataframe to be used for further modelling:" )
scaled_ds.head()
```

C. *Technology*

In this study, we will leverage the power of Google Colab for our data analysis tasks. Google Colab is a complimentary cloud-based platform that provides robust hardware capabilities. Its seamless integration with Google Drive and the ability to collaborate through shared notebooks make it an accessible and convenient choice for users.

We will be utilizing a range of libraries for this project, including but not limited to, Pandas for data manipulation and analysis, Matplotlib for data visualization, Scikit-learn (sklearn) for machine learning, and Statsmodels for statistical modeling.

In addition to Google Colab, we will also be using a suite of other tools to enhance our workflow. Neptune AI will be used for experiment tracking and model management, Zotero for reference management, and Optuna for hyperparameter optimization. These tools collectively contribute to a comprehensive, efficient, and effective data analysis pipeline.

D. *Spliting data*

We have successfully divided the dataset into two parts: one used for training the model and another used for testing the model's performance. The 8-2 split method (80% training, 20% testing) and the 7-3 split method (70% training, 30% testing) were applied to ensure the representativeness of both portions, enabling the model to generalize well to new data.

- Set 7-2-1 includes 23 attributes (after processing) and is divided into 442 for the test set, 1549 rows for the training set and 221 for the validation.

- Set 8-1-1 includes 23 attributes (after processing) and is divided into 222 for the test set, 1770 rows for the training set and 221 for the validation.

E. Evaluation

Classification metrics are used to evaluate the performance of a machine learning model that predicts discrete categories. Some of the most common metrics are:

Accuracy: The proportion of correct predictions among the total number of predictions. It is computed by dividing the number of correct predictions by the total number of predictions. Its formula:

$$Accuracy = \frac{TP + TN}{Total\ number\ of\ predictions}$$

Precision: The proportion of correct positive predictions among all positive predictions. It is computed by dividing the number of true positives by the number of positive predictions. The formula is:

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

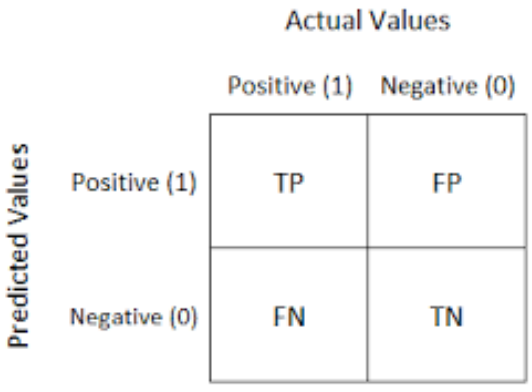
Recall: The proportion of correct positive predictions among all actual positive instances. It is computed by dividing the number of true positives by the number of positive instances. The formula is:

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

F1 score: The harmonic mean of precision and recall. It balances the trade-off between precision and recall. It is computed by multiplying the precision and recall by 2 and dividing by their sum. It is computed as follows:

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Confusion matrix: A table that summarizes the performance of a classifier by showing the number of true positives, false positives, true negatives, and false negatives. The rows represent the actual labels, and the columns represent the predicted labels.



F. Result

1) Decision Tree

- 7-2-1:

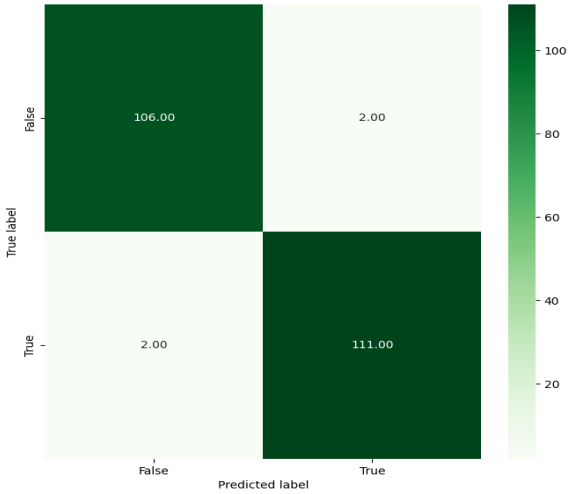


Figure 8. Decision Tree 8-1-1 Test Set

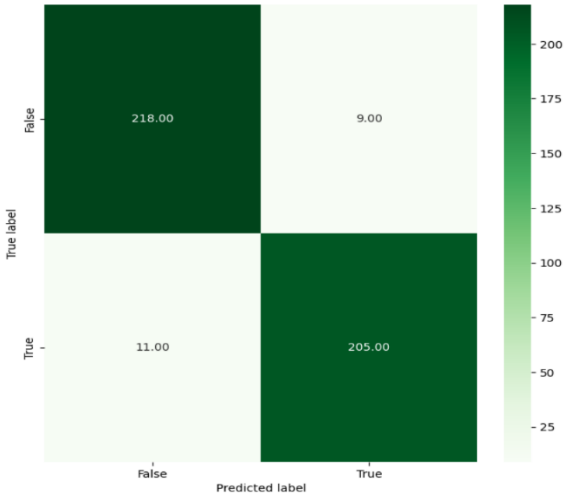


Figure 9.. Decision Tree 8-1-1 Validation Set

- 8.1.1:

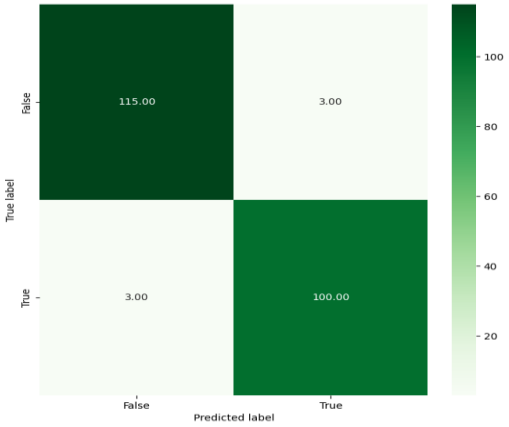


Figure 9. Decision Tree 7-2-1 Test Set

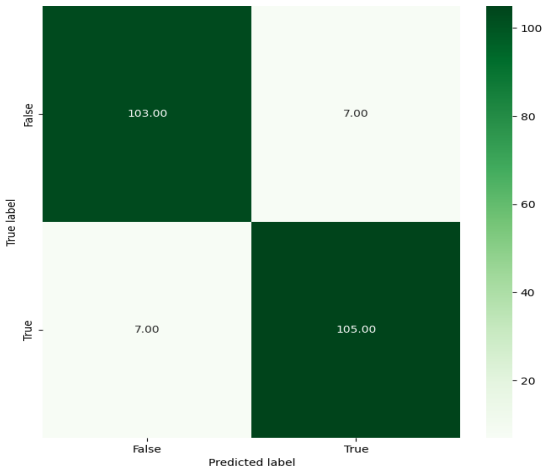


Figure 10. Decision Tree 7-2-1 Validation Set

2) Random forest

- 7-2-1:

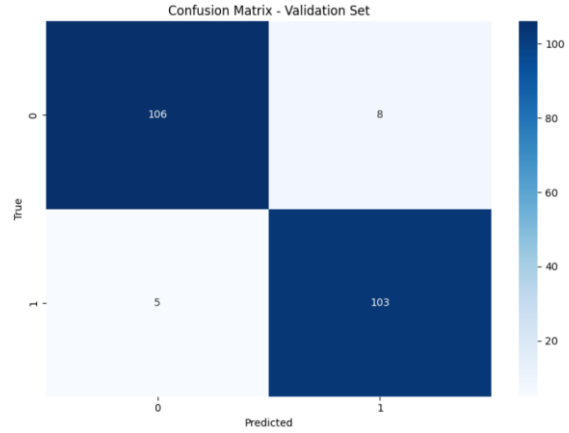


Figure 11. Random Forest 7-2-1 Validation Set

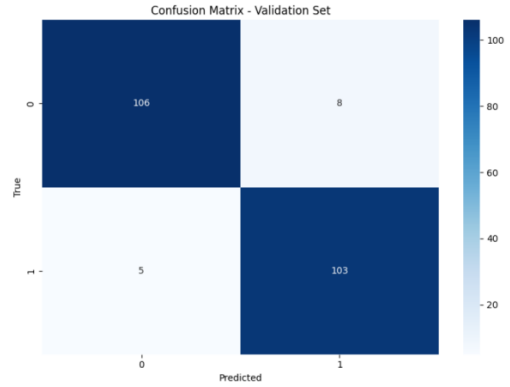


Figure 12. Random Forest 7-2-1 Test Set

- 8-1-1:

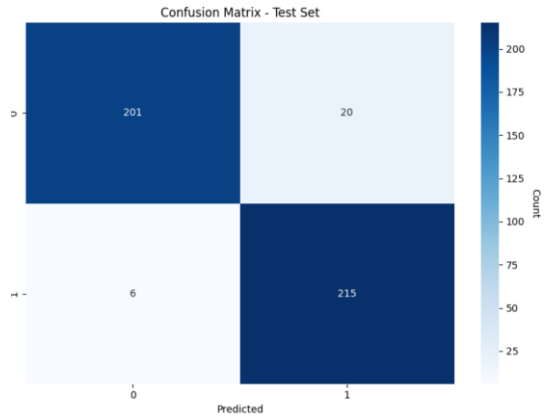


Figure 13. Random Forest 8-1-1 Test Set

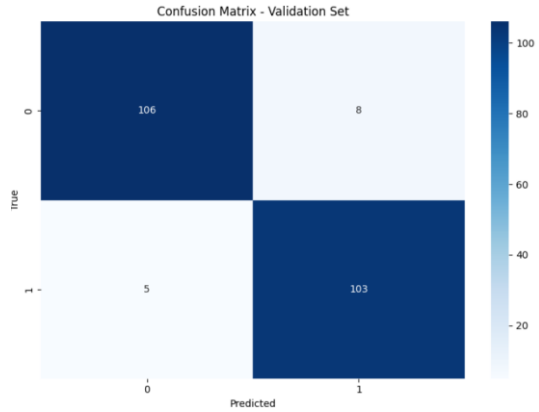


Figure 14. Random Forest 8-1-1 Validation Set

3) Light GBM

- **7-2-1:**

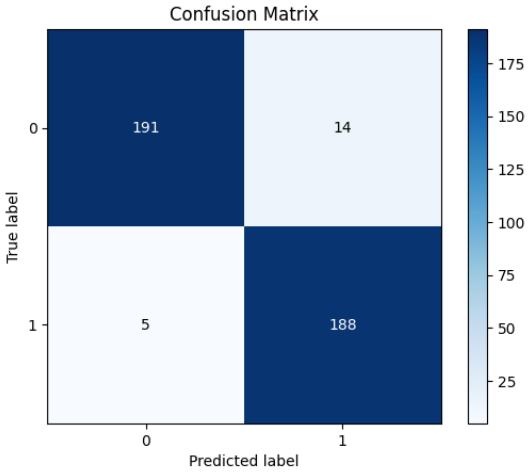


Figure 15 Light GBM 7-2-1 Test Set

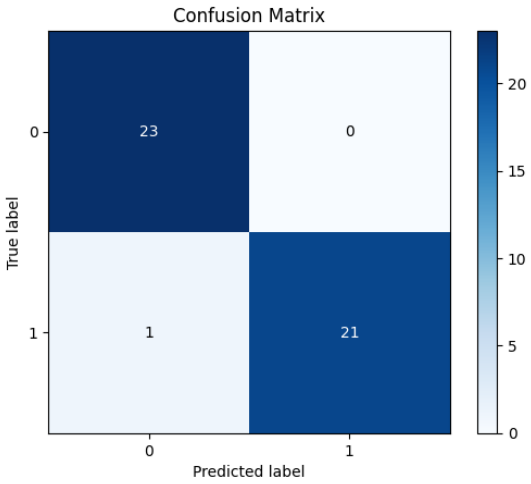


Figure 16 Light GBM 7-2-1 Test Set

- **8-1-1**

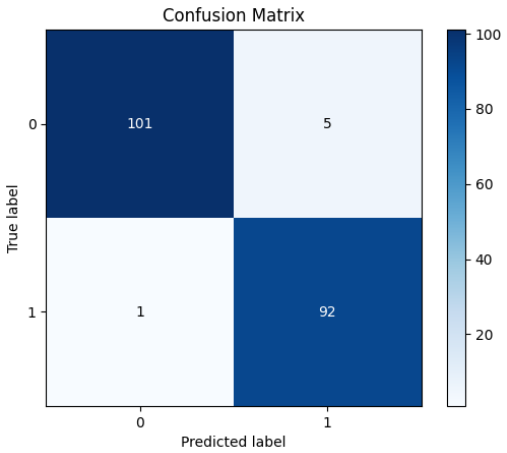


Figure 17 Light GBM 8-1-1 Test Set

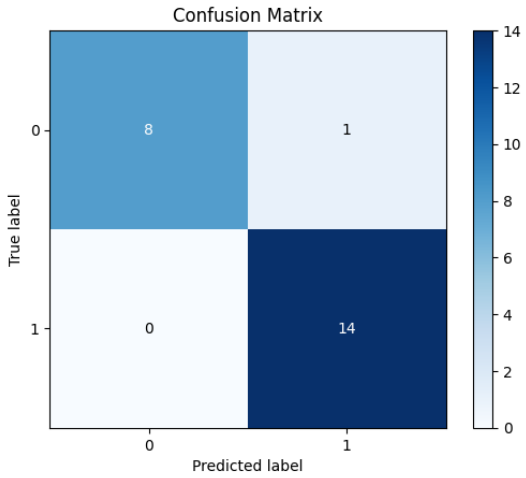


Figure 18 Light GBM 8-1-1 Validation Set

4) Support Vector Machine

- 7-2-1:

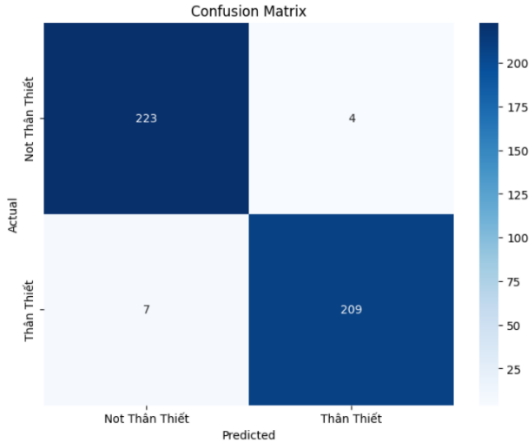


Figure 19 Support Vector Machine 7-2-1 Test Set

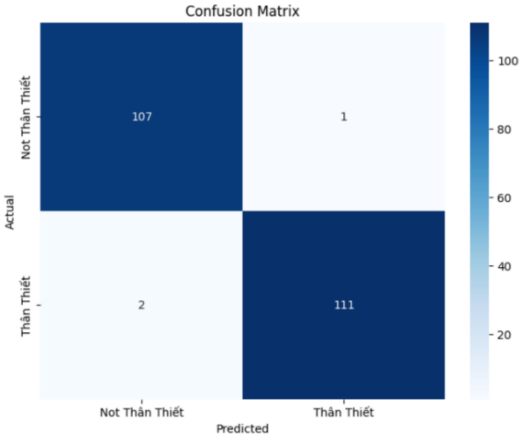


Figure 20 Support Vector Machine7-2-1 Validation Set

- 8-1-1:

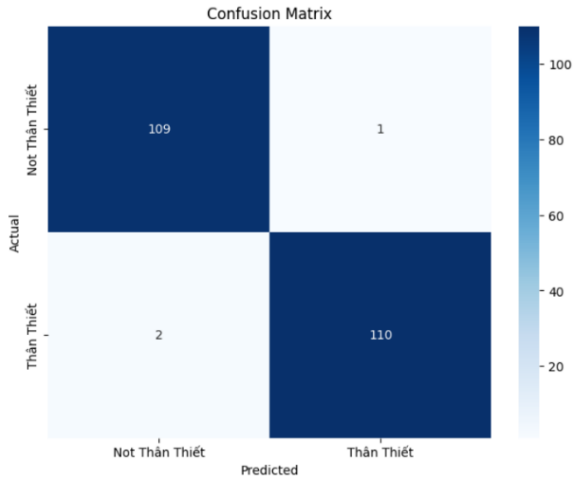


Figure 21 Support Vector Machine 8-1-1 Test Set

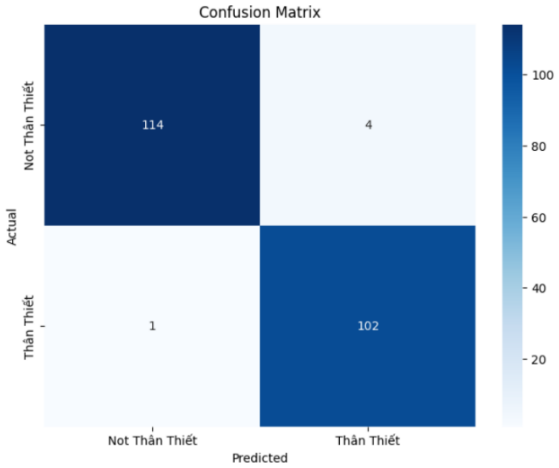


Figure 22 Support Vector Machine 8-1-1 Validation Set

Table report of 7-2-1:

- Test set:

Model	Accuracy	Precision	Recall	F1
Decision Tree	0.95	0.96	0.96	0.96
Random Forest	0.94	0.94	0.94	0.94
Light GBM	0.96	0.96	0.96	0.96
Support Vector Machines	0.97	0.98	0.98	0.98

Table 1 Table report of 7-2-1: Test set

- Validation set:

Model	Accuracy	Precision	Recall	F1
Decision Tree	0.98	0.98	0.98	0.98
Random Forest	0.94	0.94	0.94	0.94
Light GBM	0.98	0.98	0.98	0.98
Support Vector Machines	0.98	0.99	0.99	0.99

Table 2 Table report of 7-2-1: Validate set

Table report of 8-1-1:

- Test set:

Model	Accuracy	Precision	Recall	F1
Decision Tree	0.97	0.97	0.97	0.97
Random Forest	0.95	0.95	0.95	0.95
Light GBM	0.97	0.97	0.97	0.97
Support Vector Machines	0.98	0.99	0.99	0.99

Table 3 Table report of 8-1-1: Test set

- Validation set:

Model	Accuracy	Precision	Recall	F1
Decision Tree	0.94	0.94	0.94	0.94
Random Forest	0.93	0.93	0.93	0.93
Light GBM	0.96	0.96	0.96	0.96
Support Vector Machines	0.97	0.98	0.98	0.98

Table 4 Table report of 8-1-1: Validate set

VI. CONCLUSION

As an academic student delving into the retail analytics project, our aim is to expand customer knowledge and tailor shopping experiences with a focus on technological integration. Building on the promising initial work with LightGBM and SVM, we're looking to enrich our dataset with qualitative insights, drawing from social media and direct customer feedback to capture sentiments and behaviors. The evolution of our analytical tools involves refining the models for both *accuracy and interpretability, making complex algorithms* understandable to stakeholders. These efforts will support the creation of a loyalty program that genuinely resonates with customer preferences, leading to personalized rewards and fostering loyalty. We're committed to monitoring key performance indicators to assess and refine our strategies, ensuring our academic research translates into valuable business applications. The end goal is clear: to develop a more personalized, data-driven retail model that nurtures lasting customer relationships and creates an individualized shopping experience.

VII. ACKNOWLEDGEMENTS

We extend our profound gratitude to PhD. Cao Thi Nhan and Ms. Nguyen Thi Viet Huong, whose guidance and expertise have been integral to the success of our project, “Enhancing Customer Insights and Personalization Through Classification Algorithms in Retail.”

Ms. Nguyen Thi Viet Huong has been a beacon of knowledge and an extraordinary mentor throughout our research venture. With her seasoned expertise and discerning insights, she has not only guided the direction of our study but also significantly enriched our methodological approach and analytical precision. Her unwavering support and encouragement have been key motivators, empowering us to navigate and surmount the complexities encountered during this journey.

In parallel, PhD. Cao Thi Nhan’s supervision and mentorship have been nothing short of inspirational. Her profound acumen and innovative approach to retail analytics have not only deepened our comprehension of the field but also kept us abreast of the industry’s cutting-edge trends. The constructive critiques and valuable recommendations she has proffered have undeniably elevated the caliber and applicability of our research findings.

Their combined contributions have been the cornerstone of our academic endeavor, and it is with eager anticipation that we envisage future collaborations.

It is our sincere hope that the outcomes of our project make a lasting contribution to the enrichment of retail analytics knowledge and customer personalization practices. We are optimistic that our empirical findings will spark further exploration and innovative research in this dynamic and crucial domain of retail industry.

VIII. REFERENCES

[1] J. Mingers, “An empirical comparison of selection measures for decision-tree induction,” *Mach. Learn.*, vol. 3, no. 4, pp. 319–342, Mar. 1989, doi: 10.1007/BF00116837.

[2] “BF00116837.pdf.” Accessed: Dec. 25, 2023. [Online]. Available: https://link.springer.com/content/pdf/10.1007/BF00116837.pdf?pdf=preview

[3] X. Shi, Y. Cheng, and D. Xue, “Classification Algorithm of Urban Point Cloud Data based on LightGBM,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 631, no. 5, p. 052041, Oct. 2019, doi: 10.1088/1757-899X/631/5/052041.

[4] A. Saini, “Decision Tree Algorithm - A Complete Guide,” Analytics Vidhya. Accessed: Dec. 25, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/

[5] “LightGBM (Light Gradient Boosting Machine),” GeeksforGeeks. Accessed: Dec. 25, 2023. [Online]. Available: https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/

[6] “SVM Classifier Tutorial.” Accessed: Dec. 25, 2023. [Online]. Available: https://kaggle.com/code/prashant111/svm-classifier-tutorial

[7] “Customer Segmentation: Clustering 🛍️🛒🛒.” Accessed: Dec. 25, 2023. [Online]. Available: https://kaggle.com/code/karnikakapoor/customer-segmentation-clustering