EXPERIMENT REPORT

Name	Monali Patil
Project Name	Customer-Vehicle-Repurchase-Prediction
Deliverables	Model name: Decision Tree Classifier Notebook name: MachineLearning_DecisionTree_Exp3.ipynb Project Repo: https://github.com/monalippatil/MachineLearning-Customer-Vehicle-Repurchase-Prediction.git

---

1. EXPERIMENT BACKGROUND	
Provide information about the problem/project such as the scope, the overall objective, expectations. Lay down the goal of this experiment and what are the insights, answers you want to gain or level of performance you are expecting to reach.	
1.a. Business Objective	Explain clearly what is the goal of this project for the business. How will the results be used? What will be the impact of accurate or incorrect results? Objective: The aim of this project is to develop a binary classifier model that can predict customers who are likely to purchase a new vehicle. To achieve this, a dataset of car repurchases will be used for this binary classification problem and through data exploration, cleaning, feature scaling, selection of appropriate algorithm, hyperparameters and employing various techniques such as cross-validation, hyperparameter tuning, feature importance etc., to improve the accuracy of the model. Application: The outcomes of this model can be used to identify the customers who are most likely to buy a new vehicle, and this information can be used to focus marketing efforts on those potential customers which can lead to an effective and efficient promotional campaign. Impact: The model's accurate results can help the company to target the right customers who are more likely to buy the vehicle through a marketing campaign which could save marketing costs and the campaign could generate higher sales with fewer marketing expenses. However, incorrect results of the model can result in wasted marketing efforts and resources and thus, the marketing campaign may fail to achieve the desired results and sales figures.

1.b. Hypothesis	Present the hypothesis you want to test, the question you want to answer or the insight you are seeking. Explain the reasons why you think it is worthwhile considering it, Hypothesis: The objective of this experiment is to effectively identify if present customers are likely to buy a new vehicle, and to carry out this experiment, the Decision Tree Classifier algorithm has been employed for conducting and evaluating the performance to determine if this algorithm is suitable for this Binary Classification task. Rationale: Considering the business objective of determining potential customers leads for a marketing campaign, there is no good theory to map and select a suitable algorithm for this binary classification problem, therefore performing different experiments to discover which algorithm and algorithm configuration results in the best performance for this binary classification task.
1.c. Experiment Objective	Detail what will be the expected outcome of the experiment. If possible, estimate the goal you are expecting. List the possible scenarios resulting from this experiment. The expected outcome of this experiment would be the development of a binary classifier model utilizing Decision Tree Classifier algorithm that can accurately predict which customers are more likely to buy a new vehicle. The model would be trained on a dataset of current customer information and their buying behaviour, and it would use Decision Tree Classifieralgorithm to identify patterns and make predictions about future buying behaviour of the customers. The goal of this experiment would be to identify potential customers for a promotional marketing campaign aimed at increasing vehicle sales, employing Decision Tree Classifier algorithm. By focusing on customers who are more likely to buy a new vehicle, the marketing campaign could be more targeted and effective. The possible scenarios resulting from this experiment include following: • The model accurately predicts which customers are likely to buy a new vehicle, allowing the marketing team to focus on these potential customers and increase sales. • The model has a high false positive rate, indicating that the model predicts some customers will purchase a new vehicle when they actually won't purchase. This could result in wasted marketing resources and a lower return on investment. • The model has a high false negative rate, indicating that the model fails to predict some customers will purchase a new vehicle when they actually will. This could result in missed opportunities for sales and revenue. • The model is not accurate enough to be useful for predicting customer behavior, and the experiment is unsuccessful. In this case, the marketing team may need to explore other methods for identifying potential customers for their promotional campaign.

2. EXPERIMENT DETAILS

Elaborate on the approach taken for this experiment. List the different steps/techniques used and explain the rationale for choosing them.

2.a. Data Preparation

Describe the steps taken for preparing the data (if any). Explain the rationale why you had to perform these steps. List also the steps you decided to not execute and the reasoning behind it. Highlight any step that may potentially be important for future experiments.

To prepare for using the binary classification algorithm, the data for current customers was explored, cleaned up and prepared through the subsequent activities.

• Data Understanding

1] Loading Data: To use and create a binary classification model, imported data from a CSV file into the pandas dataframe.

2] Exploring Data: Explored and analysed current customer's data utilizing various following pandas functionalities to understand and identify patterns to further determine potential customers who are more likely to purchase a new vehicle.

* Additionally, to ensure the quality of the data to be utilized by the model analyzed.
- Missing/null values.
- Duplicate records.
- Outliers for numerical features.
- Data distribution for various categorical and numerical features.
- Distinct values of the categorical features.

a) df.head(): Examining initial observations of the dataset.
b) df.shape(): Examining the dimension of the dataset.
c) df.columns: Inspecting features name.
d) df.info(): Inspecting the summary information of the attributes of the dataset.
e) df.describe(): Examining the statistical information of integer variables of the dataset.
f) df.describe(include='all'): Examining statistical summary information for all variables of the dataset across different data types.
g) df.isnull().sum(): Examining whether there are any null values in the dataset.
h) df.duplicated().sum(): Examining whether there are any missing values in the dataset.

• Data Preparation

3] Treating Missing Values

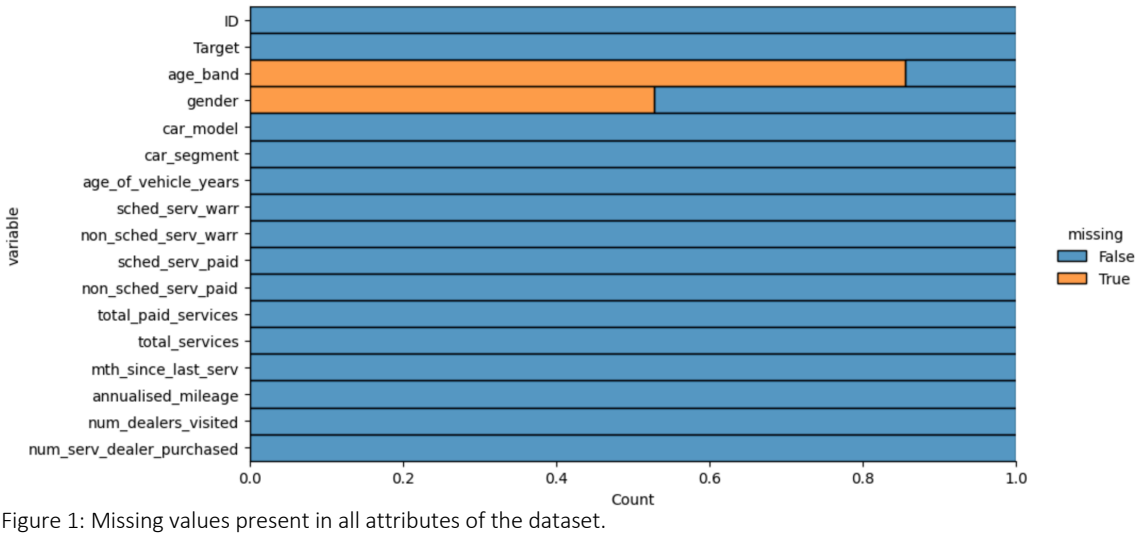

Figure 1: Missing values present in all attributes of the dataset.

* The chart from Figure 1, indicates that only two variables, 'age_band' with around 85% and 'gender' with over 50% contain missing values. However, there are no missing values for the rest of the features.

* As machine learning algorithms are incapable of processing features with missing values, we have two options to address these missing values.

i) Replace missing values with the relevant value: Using the most frequent occurring (mode) value from the feature to replace missing values in categorical variables may not accurately represent the distribution of data in those variables 'age_band' and 'gender', especially when large portion of the data is missing. This approach could also create an imbalance in the data by introducing a high number of imputed mode values, potentially leading to biased results.

ii) Eliminate the columns containing missing values entirely: It would be suitable to remove these two 'age_band' and 'gender' features since replacing any value would significantly distort the data due to the high percentage of missing values in these variables.

* Therefore, removed 'age_band' and 'gender' features from the vehicle repurchase dataset using pandas drop() function and specifying these two feature names.

4] Transforming Categorical Data into Numerical

Attribute Name	Distinct/Unique Values	Count of Distinct Values
car_model	model_1, model_2, model_3, model_5, model_6, model_4, model_7, model_8, model_9, model_10, model_11, model_13, model_12, model_14, model_15, model_16, model_17, model_18, model_19	19
car_segment	LCV, Small/Medium, Large/SUV, Other	4

Table 1: Distinct values present in ‘car_model’ and ‘car_segment’ attributes of the dataset.

* The table 1 illustrates that the features 'car_model' and 'car_segment' contain categorical data, with 19 distinct car models and 4 different vehicle segments respectively. However, this data

cannot be used directly in binary classification algorithms for machine learning because these algorithms operate on mathematical equations that require numerical inputs.

* Additionally, these features 'car_model' and 'car_segment' hold significant information regarding the model and category of vehicles that customers have purchased. This information can be used to predict whether the customer is inclined to purchase a new vehicle.

One-Hot Encoding

* Therefore, it is necessary to convert the categorical data of these features into numerical data. To achieve this, employing one-hot encoding method, which creates binary columns for every distinct category in the feature.

* This method is preferred as there is no inherent order among the category values of the feature, and do not want to create any random relationships between them and prevent model from assuming natural ordering among these distinct categories that may suffer from model bias.

5] Selecting Target Variable & Features

* The 'TARGET' feature is the target variable representing customers who have bought more than one vehicle with class 1 and those who have only purchased one vehicle with class 0. This, along with the predictor features, are separated into 'X' and 'y' variables to build a binary classification model.

6] Splitting Data into Different Sets: Training, Validation and Testing

* Creating a validation set provides adaptability to carry out multiple experiments and allows the comparison of the model's performance on the training set and the validation set to determine if the model is overfitting or underfitting.

* Furthermore, using a validation set, we can adjust model's hyperparameters and retrain the model until it performs well on the validation set. After we have optimized the model's performance on the validation set, we can then apply the model on the test set to assess its ability to generalize to new and unseen data.

* Therefore, utilizing train_test_split() function from sklearn.model_selection module of sklearn library, the car repurchase dataset is split into training (60%), validation (20%), and testing (20%) sets to leverage the flexibility of multiple experimentation.

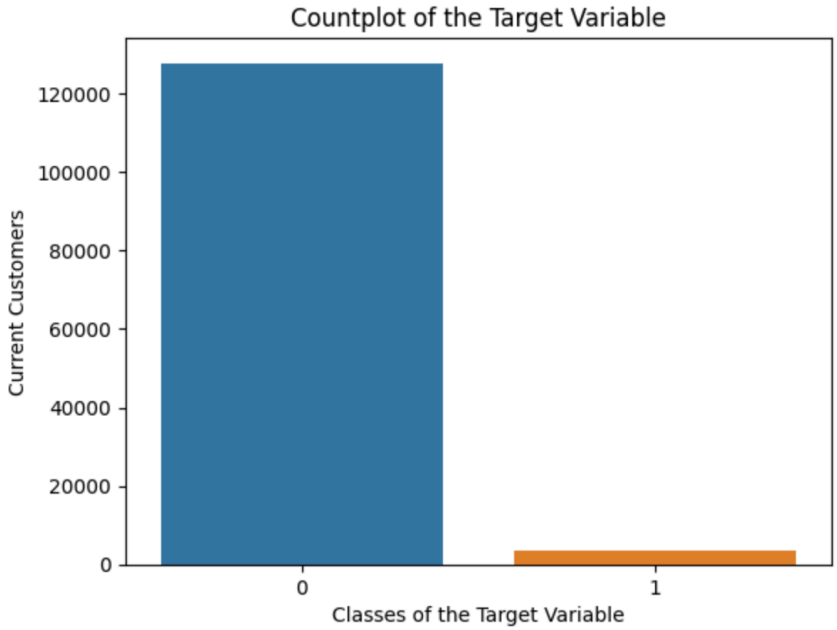

Figure 2: Examining the quantity of each class of the target variable from the actual dataset.

* Additionally, for any (binary) classification problem, it is essential to examine the frequencies/rates of every class within the target variable and ensure that the splitting of data into different sets matches as similar to the actual data spread.

* As indicated in the Figure 2, the car repurchase dataset is highly imbalanced, it is necessary to obtain as similar distribution of distinct classes of the target variable for all of the different sets as original dataset. After splitting process, following are the frequencies of the target variable classes.

Dataset/Frequency Rates	Class 1	Class 0
Actual Data:	0.97	0.026
Training Set:	0.97	0.026
Validation Set:	0.97	0.026
Testing Set:	0.97	0.028

7] Features Scaling

* Machine learning algorithms often use some form of distance calculation to determine the relationship between different features in the dataset. If the features are on different scales, the algorithm may prioritize some high value range features over others which may have more significant information, leading to biased results.

* In addition, the values for the numerical features are divided into ten deciles, ranging from 1 to 10, while categorical data that has been converted into numerical features, consist of either 0 or 1.

* Therefore, performing scaling for all the features so that all the features values are at same level and that the algorithm can use all the information from the dataset features to learn generalised patterns, identify buying behaviour and make accurate predictions.

StandardScaler

* Employing StandardScaler class which is imported from sklearn.preprocessing module to scale and bring all features values at same level. Choosing this method because it does not change the shape of the data distribution and preserves outliers as it scales the data based on the mean of 0 and standard deviation of 1, of the entire dataset, rather than individual datapoints.

The following are some crucial steps that could be important for any of the classification experiments in future.

- It is important to thoroughly examine and handle missing values in order to avoid introducing biases in the model.
- Outliers and duplicate values should be carefully analysed and reviewed from business perspectives and treated accordingly.
- The decision to use LabelEncoder, OrdinalEncoder or OneHotEncoder for converting categorical data into numerical form should depend on various factors, including the nature of the categorical values and whether or not there is any inherent order among them. Careful consideration is necessary to determine the appropriate method.
- Similarly, the selection of feature scaling method - whether to use MinMaxScaler, MaxAbsScaler, or StandardScaler - should be made based on whether or not there is an inherent order in the data, as well as other factors such as the need to maintain the integrity of outliers.
- For classification problems and especially for imbalanced data, is essential to ensure the frequency of each class in the target variable in each set is representative of the actual data. The "stratify=y" parameter could be used during the splitting process if the class distribution is not similar to the actual data. This will help to maintain the integrity of the data and ensure accurate results.
- To prevent the model from overfitting on specific data points and to enable it to learn generalized patterns, it is important to eliminate any identifier attributes.

2.b. Feature Engineering

Describe the steps taken for generating features (if any). Explain the rationale why you had to perform these steps. List also the feature you decided to remove and the reasoning behind it. Highlight any feature that may potentially be important for future experiments

	ID
count	131337.000000
mean	77097.384180
std	44501.636704
min	1.000000
25%	38563.000000
50%	77132.000000
75%	115668.000000
max	154139.000000

Figure 3: Statistical summary of 'ID' attribute.

	* The 'ID' attribute seems to be an identifier and its values ranges from 1 to 154139 as indicated in Figure 3. It does not have any predictive value as it contains a unique value for each observation, which means that its values do not contribute to any patterns or trends in the data. * And including them in the analysis can lead to overfitting straight a way, where model would fit these specific values from the feature rather than the underlying generic patterns in the data. * Therefore, removed 'ID' attribute from the vehicle repurchase dataset using pandas drop() function and specifying this feature name. Note: Removal of 'age_band' and 'gender' features is stated in earlier part 3] Treating Missing Values of 2.a Data Preparation section.
2.c. Modelling	Describe the model(s) trained for this experiment and why you choose them. List the hyperparameter tuned and the values tested and also the rationale why you choose them. List also the models you decided to not train and the reasoning behind it. Highlight any model or hyperparameter that may potentially be important for future experiments The model trained for this experiment is one of the Binary Classification algorithms, Decision Tree Classifier model to accurately determine whether current customers are likely to purchase a new vehicle and evaluating its effectiveness to determine its suitability for this binary classification task which can then be used to target potential customers for a marketing campaign. The target variable consists of either 0 or 1, where 0 represents the purchase of a single vehicle by the customer, while 1 represents the purchase of multiple vehicles. As we aim to predict only these two possible classes outcome, this Binary Classification model is the right choice along with learning objective purpose. Hyperparameters Selected.

Hyperparameters Name	Values Tested		
max_depth:	14	7	8
min_samples_split	8	7	11
class_weight:	balanced		
criterion:	gini		
random_state:	7		

* Initially, utilizing the default hyperparameters of the Decision Tree classifier algorithm and evaluating how well it performs. Then, modifying the hyperparameter values as necessary in exercises to determine optimal performance.

* To address the issue of overfitting, anticipated the below values of hyperparameters and evaluating the efficiency of the Decision Tree classification algorithm.
class_weight='balanced', criterion='gini', max_depth=14, min_samples_split=8, random_state=7

	* Utilising class_weight='balanced' hyperparameter to adjust the weights of the classes based on their proportion in the training set, which can help to address the issue of high imbalance classes and improve model performance. * The criterion='gini' hyperparameter is used to specify the splitting criterion used by the Decision Tree model. * The max_depth hyperparameter is used to specify the maximum depth of the Decision Tree, which is the maximum number of levels allowed in the tree. * The min_samples_split hyperparameter is used to specify the minimum number of samples required to split an internal node. * The random_state parameter is used to ensure that the model can be reproduced exactly in the future which is useful when comparing different models or tuning hyperparameters. <u>Hyperparameter Tuning</u> * Based on the exercises and its evaluation results, it is possible to make informed guesses regarding the approximate range of hyperparameter values for the Decision Tree algorithm. * So, to identify the best hyperparameters values for the Decision Tree algorithm, performing hyperparameter tuning by utilizing cross-validation with StratifiedKFold() and random search with RandomizedSearchCV() functions. <u>Cross-Validation: StratifiedKFold</u> * For the imbalanced dataset, StratifiedKFold method of cross-validation is useful to assess the generalization performance of the model as it splits the dataset into k subsets of roughly equal size ensuring that each subset contains approximately the same proportion of the different classes in the target variable as the original dataset. * Thus, it helps to prevent the model from being biased towards one particular class, which can lead to poor performance on unseen data. <u>Random Search: RandomizedSearchCV</u> * Utilizing Random Search method (RandomizedSearchCV) for hyperparameter tuning, as it introduces randomness by randomly selecting values across the search space which may provide combination of hyperparameter values that result in the lowest possible error. * This may not always be possible with Grid Search, as evenly spaced grid points search may not capture the optimal hyperparameter values. Due to time constraints, it was not possible to thoroughly understand and test additional hyperparameters like criterion='entropy', min_samples_leaf, min_weight_fraction_leaf, max_leaf_nodes and min_impurity_decrease of the Decision Tree classifier algorithm that may impact model performance. However, would like to analyse these hyperparameters in future experiments.

3. EXPERIMENT RESULTS

Analyse in detail the results achieved from this experiment from a technical and business perspective. Not only report performance metrics results but also any interpretation on model features, incorrect results, risks identified.

3.a. Technical Performance

Score of the relevant performance metric(s). Provide analysis on the main underperforming cases/observations and potential root causes.

Performance Metrics

To evaluate the model's performance utilizing the below performance metrics.

- Precision
- Recall
- Weighted F1 Score
- Binary F1 Score
- Confusion Matrix

Baseline Performance (Null Accuracy)

weighted F1 Score: 0.9608

Exercise 1:

Algorithm: Decision Tree Classifier model

Hyperparameters: Default Hyperparameters

Performance Metrics:

Dataset	Precision	Recall	weighted F1 Score	binary F1 Score
Training:	1.0	0.9995	0.9999	0.9997
Validation:	0.7652	0.7801	0.9877	0.7726

	Error Rate		Error Count	
Dataset	False Positive	False Negative	False Positive	False Negative
Training:	0.00%	0.00%	0	1
Validation:	0.64%	0.59%	135	124

* The binary F1 Score of the classification results that belongs to positive class with 0.9997 for training and 0.7726 for validation, indicates that the model is significantly overfitting. The Precision and Recall scores of both the training and validation sets demonstrate similar findings.

* Moreover, the training set has almost zero False Negative and False Positive errors, while the validation set has 124 and 135 errors, respectively. This illustrates that the model not generalised enough and misclassifying potential customers who are likely to purchase a new vehicle on validation set.

* Therefore, to address the issue of overfitting, anticipating the below values of hyperparameters and evaluating the efficiency of the Decision Tree classification algorithm.
class_weight='balanced', criterion='gini', max_depth=14, min_samples_split=8, random_state=7

Exercise 2:

Algorithm: Decision Tree Classifier model

Hyperparameters: class_weight='balanced', criterion='gini', max_depth=14, min_samples_split=8, random_state=7

Performance Metrics:

Dataset	Precision	Recall	weighted F1 Score	binary F1 Score
Training:	0.5360	0.9941	0.9805	0.6965
Validation:	0.4641	0.8599	0.9739	0.6028

	Error Rate		Error Count	
Dataset	False Positive	False Negative	False Positive	False Negative
Training:	2.26%	0.02%	1898	13
Validation:	2.66%	0.38%	560	79

* The performance metrics for all Precision, Recall and binary F1 Score of training and validation set, emphasize that the model is significantly overfitting.

* Although the False Negative errors of 13 for training and 79 for validation are low indicating that the model is incorrectly classifying some current customers who are more likely to buy a new vehicle for validation set.

* However, this problem is not resolved because the misclassification rate for the validation set is still relatively higher at 0.38% compared to the training set's 0.02%.

* Thus, from this evaluation of the model and learning, making the educated selection of hyperparameters values of max_depth=7, min_samples_split=7 to tackle the issue of misclassifying and overfitting.

Exercise 3:

Algorithm: Decision Tree Classifier model

Hyperparameters: class_weight='balanced', criterion='gini', max_depth=7, min_samples_split=7, random_state=7

Performance Metrics:

Dataset	Precision	Recall	weighted F1 Score	binary F1 Score
Training:	0.2668	0.9578	0.9483	0.4174
Validation:	0.2609	0.9379	0.9462	0.4083

	Error Rate		Error Count	
Dataset	False Positive	False Negative	False Positive	False Negative
Training:	6.91%	0.11%	5850	93
Validation:	7.13%	0.17%	1498	35

* The weighted F1 Score and Precision score for both training and validation are almost similar advising that the model is performing uniformly. Additionally, there are lowered but still relative False Negative errors of 93 for training and 35 for validation.

* However, the binary F1 Score of 0.4174 for training and 0.4083 for validation suggests a slightly overfit model, which is also supported by the Recall score.

* Based on the above exercises and evaluation results, it is possible to make informed guesses

regarding the approximate range of hyperparameter values for the Decision Tree algorithm.

* So, to identify the best hyperparameters values for the Decision Tree algorithm, performing hyperparameter tuning by utilizing cross-validation with StratifiedKFold() and random search with RandomizedSearchCV() functions.
Below are the Hyperparameters tuned values of the Decision Tree classifier:
{'class_weight': 'balanced', 'criterion': 'gini', 'max_depth': 8, 'min_samples_split': 11}

Exercise 4:
Algorithm: Decision Tree Classifier model

Hyperparameters: Hyperparameters Tuned Values

Performance Metrics:

Dataset	Precision	Recall	weighted F1 Score	binary F1 Score
Training:	0.3871	0.9537	0.9673	0.5507
Validation:	0.3690	0.9095	0.9647	0.5250

	Error Rate		Error Count	
Dataset	False Positive	False Negative	False Positive	False Negative
Training:	3.96%	0.12%	3331	102
Validation:	4.17%	0.24%	877	51

* The binary F1 Score, which focuses on binary classification, for the classes with positive labels, has 0.5507 score for the training and 0.5250 for the validation set, which suggests that the model is relatively overfitting. The similar nature of overfitting is indicated by Precision and Recall Scores.

* Moreover, the number of False Negative errors has increased in the current 4th model, with 102 for the training and 51 for the validation set, compared to the previous 3rd model with 93 for the training set and 35 for the validation set.

* Therefore, the 3rd Decision Tree classifier model which has hyperparameters max_depth=7, min_samples_split=7 performs relatively better than the 4th Decision Tree classifier model with tuned hyperparameters values obtained from the Random Search.

* So evaluated model's performance on the unseen test data utilizing the 3rd Decision Tree classifier model which had hyperparameters max_depth=7, min_samples_split=7.

Performance Metrics with evaluation on Testing set:

Dataset	Precision	Recall	weighted F1 Score	binary F1 Score
Training:	0.2668	0.9578	0.9483	0.4174
Validation:	0.2609	0.9379	0.9462	0.4083
Testing:	0.2763	0.9307	0.9464	0.4262

	Error Rate		Error Count	
Dataset	False Positive	False Negative	False Positive	False Negative
Training:	6.91%	0.11%	5850	93
Validation:	7.13%	0.17%	1498	35
Testing:	6.97%	0.20%	1830	52

	* The Recall which measures the proportion of actual positive cases that are correctly identified, scores 0.9578 for training, 0.9379 for validation and 0.9307 for testing, indicating that the model is good at identifying customers who are likely to purchase a new vehicle along with slight overfit. * Whereas the binary F1 Score for classes with positive labels, scores 0.4174 for training, 0.4083 for validation and 0.4262 for testing, informing that the model is correctly identifying a large proportion of positive cases (customers who are likely to purchase a new vehicle), but it is also incorrectly classifying many negative cases as positive. * Moreover, the False Negative error of the model is considerably low and by far decent compared to earlier trained model with different algorithm. The False Negative errors are 93, 35 and 52 for training, validation and testing set respectively. * To select a suitable algorithm to identify prospective customers who are interested in buying a new vehicle for the purpose of marketing campaign, comparing the performance of different algorithms, and evaluating their performance based on multiple metrics. Therefore, employing Random Forest algorithm in the next experiment.
3.b. Business Impact	Interpret the results of the experiments related to the business objective set earlier. Estimate the impacts of the incorrect results for the business (some results may have more impact compared to others) Among all the models from the exercises, the Decision Tree classifier binary classification model from the 3rd exercise is justifiable with low False Negative errors are 93, 35 and 52 for training, validation and testing set respectively. This indicated that the model is misclassifying less amount of customers as not interested in purchasing a new vehicle. Additionally, the binary F1 Score resulting in the classifications that belong to the positive class are almost similar across all the sets, with 0.4174 for training, 0.4083 for validation and 0.4262 for testing suggesting that the model is correctly identifying potential customers for buying a new vehicle, while misclassifying some negative instances as positive. Therefore, it is essential to tackle this binary classification problem by considering business objectives and performance targets for the marketing campaign.
3.c. Encountered Issues	List all the issues you faced during the experiments (solved and unsolved). Present solutions or workarounds for overcoming them. Highlight also the issues that may have to be dealt with in future experiments. The criterion hyperparameter of a Decision Tree algorithm determines which metric is used to evaluate the quality of a split. The use of 'gini' or 'entropy' measure could affect the performance of the model. So, I reviewed some canvas and online resources, but was unable to comprehend the mathematical equation and its underlying concept thoroughly. Therefore, would consider testing these measures in the future experiments that could lead to better performance and generalization of the model on unseen data.

4. FUTURE EXPERIMENT

Reflect on the experiment and highlight the key information/insights you gained from it that are valuable for the overall project objectives from a technical and business perspective.

4.a. Key Learning

Reflect on the outcome of the experiment and list the new insights you gained from it. Provide rationale for pursuing more experimentation with the current approach or call out if you think it is a dead end.

Although the Decision Tree classifier model was trained using the hyperparameters tuned values obtained from cross-validation with StratifiedKFold() and random search with RandomizedSearchCV() functions in the 4th exercise, the model was still relatively overfitting. Additionally, the model generated a considerable number of False Negative errors.

Therefore, employing hyperparameters tuned values does not necessarily guarantee optimal performance of the model. Further investigation is required, including testing the model with various hyperparameter combinations of values that could lead to the lowest possible errors, and ensure that the model can generalize better on unseen data.

Additionally, it is essential to take into account the business context and benchmarks while conducting machine learning experiments to assess the model's performance and work out whether it is worthwhile to proceed with additional analysis.

As mentioned in the 2.c Modelling section, would like to test, and assess the model employing additional hyperparameters like criterion='entropy', min_samples_leaf, min_weight_fraction_leaf, max_leaf_nodes and min_impurity_decrease to verify if the generalization performance of the model could be enhanced.

4.b. Suggestions / Recommendations

Given the results achieved and the overall objective of the project, list the potential next steps and experiments. For each of them assess the expected uplift or gains and rank them accordingly. If the experiment achieved the required outcome for the business, recommend the steps to deploy this solution into production.

I would like to perform below task.

* Train the Decision Tree classifier model with additional hyperparameters like criterion='entropy', min_samples_leaf, min_weight_fraction_leaf, max_leaf_nodes and min_impurity_decrease and evaluate if the performance is improving.

To successfully deploy a binary classification algorithm in a production environment, it is advisable to carry out below steps.

- Scale and transform the model to handle large datasets.
- Select an appropriate deployment environment whether cloud-based or on-premises.
- Modify the model to suit production settings while complying with various security, ethical, and privacy guidelines.
- Conduct testing and monitoring of the deployed model.
- Periodically updated the model with new data.
- Provided documentation for usage.
- Review the performance of the model and retrain the model as needed.