**A COUPON RECOMMENDATION SYSTEM USING USER BEHAVIOR DATA**

**BY**

**TIFFANY OLUCHI UGWUNEBO**

**Intern, HubbleMind**

**HSR Layout, Bangalore, Karnataka, India**

[**info@hubblemind.com**](mailto:info@hubblemind.com)

**29th December, 2024**

**TABLE OF CONTENTS**

[1. INTRODUCTION](#_4d34og8)

[2. DATA UNDERSTANDING AND CLEANING](#_2s8eyo1)

[2.1 DATA EXPLORATION](#_17dp8vu)

[2.2 HANDLING INCONSISTENCIES AND MISSING VALUES.](#_3rdcrjn)

[2.3 ENCODING AND SCALING](#_26in1rg)

[3. EXPLORATORY DATA ANALYSIS](#_lnxbz9)

[3.1 DATA VISUALIZATION](#_35nkun2)

[3.2 TREND ANALYSIS](#_1ksv4uv)

[3.3 CORRELATION ANALYSIS](#_44sinio)

[3.4 DISTRIBUTION VISUALIZATION](#_2jxsxqh)

[4. DATA MODELING](#_z337ya)

[4.1 CREATING TRAIN AND TEST DATASET](#_3j2qqm3)

[4.2 IMPLEMENTATION AND EVALUATION OF VARIOUS MACHINE LEARNING MODELS](#_1y810tw)

[4.3 HYPER PARAMETER TUNING.](#_4i7ojhp)

[5. CONCLUSION AND RECOMMENDATIONS](#_2xcytpi)

[6. GITHUB REPOSITORY](#_1ci93xb)

# INTRODUCTION

The primary objective of this project is to develop a machine learning model capable of predicting coupon acceptance based on a variety of features/conditions/patterns. Coupon acceptance is a crucial factor for businesses seeking to understand consumer behavior and improve their marketing strategies. By predicting whether a customer will use a coupon in certain circumstances, businesses can better target their promotions and increase the probability of coupon redemption.

The dataset provided contains both categorical and numerical features, and effective analysis of these features is essential for model development. This project is divided into several key phases: data understanding and cleaning, exploratory data analysis (EDA), model building, evaluation, and model fine-tuning.

After preparing the data, multiple machine learning algorithms will be implemented to build models for predicting coupon acceptance.

# DATA UNDERSTANDING AND CLEANING

## 2.1 DATA EXPLORATION

The dataset, ‘in-vehicle-coupon-recommendation.csv’ was loaded as a dataframe for analysis using the Pandas library. Pandas will be the core framework used for analysis for this project as it is built on top of NumPy, making it perfect for data analysis.

The columns of the dataframe were set to the max to provide a full overview to the data, enabling exploration. The dataframe was of the shape, 12684 data points(rows) x 26 features(columns). The initial data types revealed 8 columns of datatype int64(integers) and 18 of datatype object(strings). This implied the dataset comprised both numerical and categorical features. The details of each columns are as listed below:

* Destination: where is the user going?.
* Passenger: who are the passengers in the car?
* Weather: weather conditions when the user is driving.
* Temperature: temperature (in Fahrenheit) when the user is driving.
* Time: time of the day when coupon was presented to the user.
* Coupon: type of coupon offered by the company.
* Expiration: time before coupon expires.
* Gender: gender(male/female) of the user.
* Age: age of user.
* Marital Status: marital status of user.
* Has Children: does the user have children?
* Education: user’s level of education.
* Occupation: occupation of user.
* Income: income of user.
* Car: description of vehicle used by user.
* Bar: user’s frequency of bar visitations.
* CoffeeHouse: user’s frequency of coffeehouse visitations.
* CarryAway: frequency of user ordering take-away.
* RestaurantLessThan20: frequency at which a user visits a restaurant with an average expense per person of less than 20$ per month.
* Restaurant20To50: frequency at which the user visits a restaurant with an average expense per person of 20$-50$ per month.
* toCoupon\_GEQ5min: is the driving distance to the restaurant/cafe/bar for using the coupon greater than 5 minutes?
* toCoupon\_GEQ15min: is the driving distance to the restaurant/cafe/bar for using the coupon greater than 15 minutes?
* toCoupon\_GEQ25min: is the driving distance to the restaurant/cafe/bar for using the coupon greater than 25 minutes?
* Direction\_same: is the restaurant/cafe/bar in the same direction as the user’s destination?
* Direction\_opp: is the restaurant/cafe/bar in the opposite direction as the user’s destination?
* Y: was the coupon accepted or rejected? (target attribute).

After exploring the full dataframe, it was noticed that a few features were labeled with the wrong data types. The following columns: Gender, Has\_Children, toCoupon\_GEQ15min, toCoupon\_GEQ25min, Direction\_same, Direction\_opp and Y were labeled as numerical columns. However, their values only had two variations: 1 and 0( M or F for the case of Gender). These variations represented responses ‘Yes’ and ‘No’ respectively. Therefore, rather than numerical, these columns were classified as ‘binary categorical’ columns.

Though the ‘Expiration’ feature has only two unique values(1 day and 2 hours), rather than binary categorical, it is preferable to convert this feature to numerical as the duration before a coupon expires could influence the user's decision to accept the coupon.

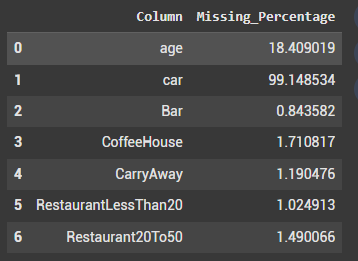
The Income column is designated as an object datatype due to the form its values takes i.e a range. However, for modelling, it is best to treat these values as continuous variables rather than strings. This will help the model learn patterns based on the magnitude of the income of a user. Thus, a midpoint converting algorithm was applied to the income columns to better capture numeric relationships.

The Age column was labelled as an object type as it contained values like ‘below20’ and ‘above50’. For modeling, these two values were converted to whole numbers for ease in scaling.The final mislabeled column was the ‘toCoupon\_GEQ5min’ column which was given an int64 data type. This stems from the fact it only had one value for every datapoint: 1. Though the column won’t impact modelling and visualization, it was treated as a categorical column. Below are the proper classification of our dataset’s features:

* Categorical Features: Destination, Passenger, Weather, Time, Coupon, Gender, Marital Status, Has Children, Education, Occupation, Car, Bar, CoffeeHouse, CarryAway, RestaurantLessThan20, Restaurant20To50, toCoupon\_GEQ5min, toCoupon\_GEQ15min, toCoupon\_GEQ25min, Direction\_same. Direction\_opp , Y.
* Numerical Features:Expiration, Temperature, Age, Income

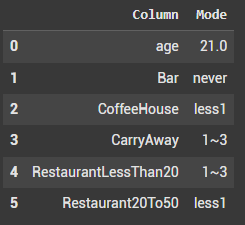
## 2.2 HANDLING INCONSISTENCIES AND MISSING VALUES.

Before handling missing values, the columns containing those missing values have to be identified. The features containing null values were as follows: Age, Car, Bar, CoffeHouse, CarryAway, RestaurantLessThan20 and Restaurant20To50. The percentage of missing values in each column were calculated and tabulated for assessment as the image below shows:



From this table, it can be inferred that obtaining users' car descriptions is the most challenging data for the company to collect.This is likely due to the fact that the knowledge of car variations is not universally known. A recommendation for the company to improve data acquisition in this area is to implement a process to allow users submit pictures of their vehicles. These images can then be processed to extract accurate information, ensuring a more reliable dataset.

As the percentage of missing values for the ‘Car’ feature exceeded 50%, it would make little impact on modeling and prediction. Thus, the column was dropped. For the rest of the columns with missing values, the missing values were replaced with the mode. Here is an overview of the most frequent values for each of the columns with missing values:



From this table, it can be inferred that many users do not visit the bar. A recommendation for the company to increase the likelihood of users accepting coupons would be to offer more coupons that aren't for the bar.

The first step in fixing inconsistencies in a dataframe is viewing the unique values belonging to every feature. Using Pandas’s replace method, the relevant values were mapped and corrected accordingly.The ‘occupation’ column had too many unique values and was therefore mapped down to general occupations. Misspelt columns were also corrected to avoid unnecessary spelling errors during analysis and visualization.

## 2.3 ENCODING AND SCALING

One-hot encoding is the technique used in this analysis. It converts the categorical data into a format that can be used in machine learning algorithms thus making better predictions. Rather than object data types, the selected features are converted to binary dtypes(1 or 0). One-hot encoding is necessary for linear regression, decision trees and neural networks as these algorithms require numerical input. This mode of encoding also ensures that there is no unintended ordinal(sequential) relationship between categories.

One-hot encoding is used on categorical features with nominal values(discrete values with no inherent relationship). The features in our dataset that fit the requirements are: 'destination', 'passenger', 'weather', 'coupon' and 'maritalStatus', ‘gender’, ‘education’ and ‘occupation’.

For scaling features, StandardScaler preprocessing technique was applied. This is a technique that normalizes the features of your dataset. Data is transformed by eliminating the mean and scaling to unit variance. As a result, each feature has a standard deviation of 1 and a mean of 0.

The formula:

Where:

Z = Z score(standardized value)

X = input value

σ= standard deviation

Standard scaling is important as many machine learning algorithms are sensitive to numerical data. An extreme imbalance in scales could lead to inefficient model performance. Standardization also helps gradient-based algorithms like neural networks converge faster.

The features selected for standard scaling are numerical columns with varied units of measurements or scales. There are four features in our dataset that fit this requirement namely; Temperature(F), Age, Expiration(hrs) and Income($), Time(24-Hour clock).

‘Bar’, ‘CoffeeHouse’, and ‘CarryAway’ are features with unique values of less than 1, greater than 1 etc which indicate frequency. Rather than one-hot encoding, these sort of columns were converted to numerical data that represent a nominal order between values to maintain the hierarchal relationship between frequencies. ]

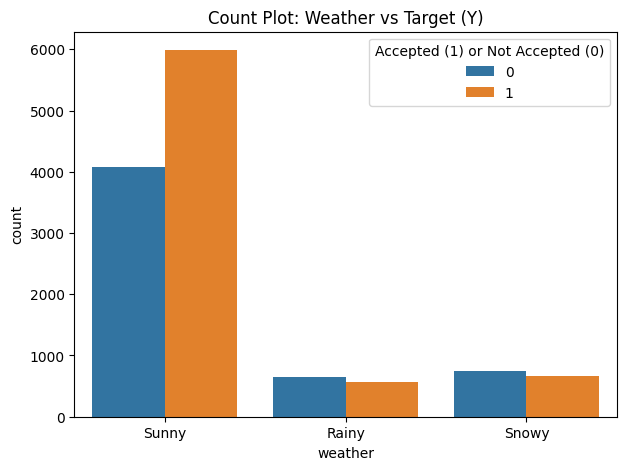
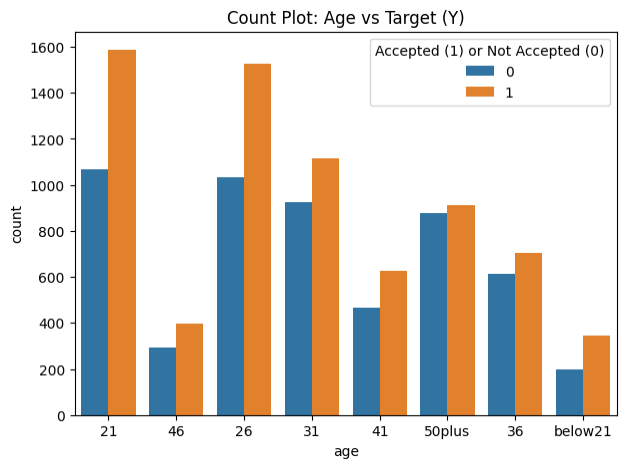
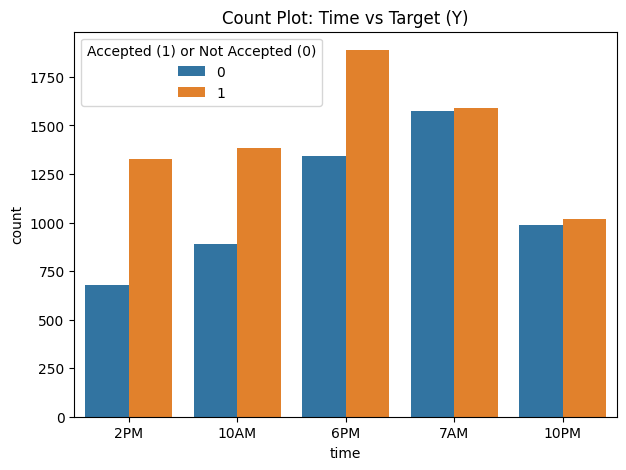
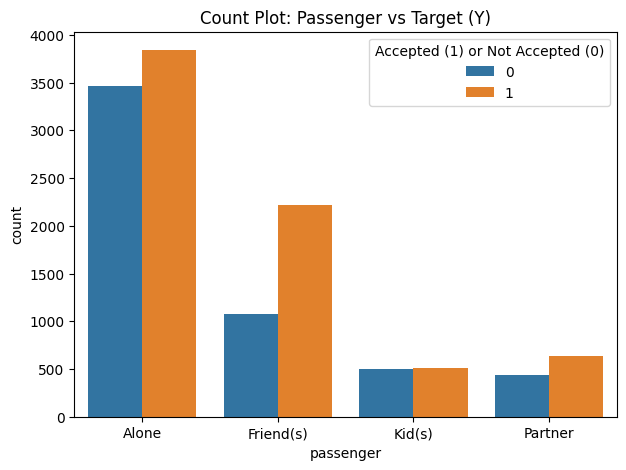
These features were scaled down to a range between 1 and 0.

After encoding and scaling, the shape of the altered dataframe was 12684 data points × 46 features

# 3. EXPLORATORY DATA ANALYSIS

## 3.1 DATA VISUALIZATION

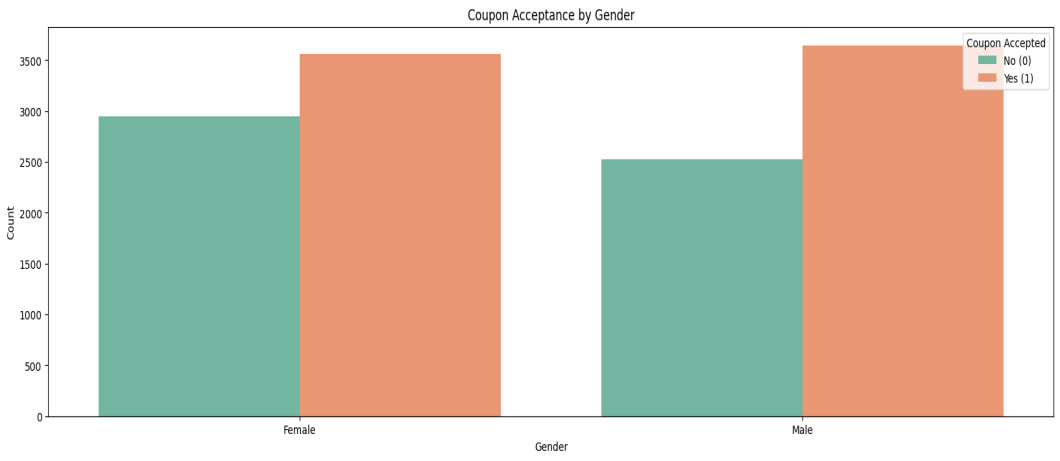
For data visualization, box plots for the target Y variable against features like weather, passengers, age and time were plotted. These were the results:

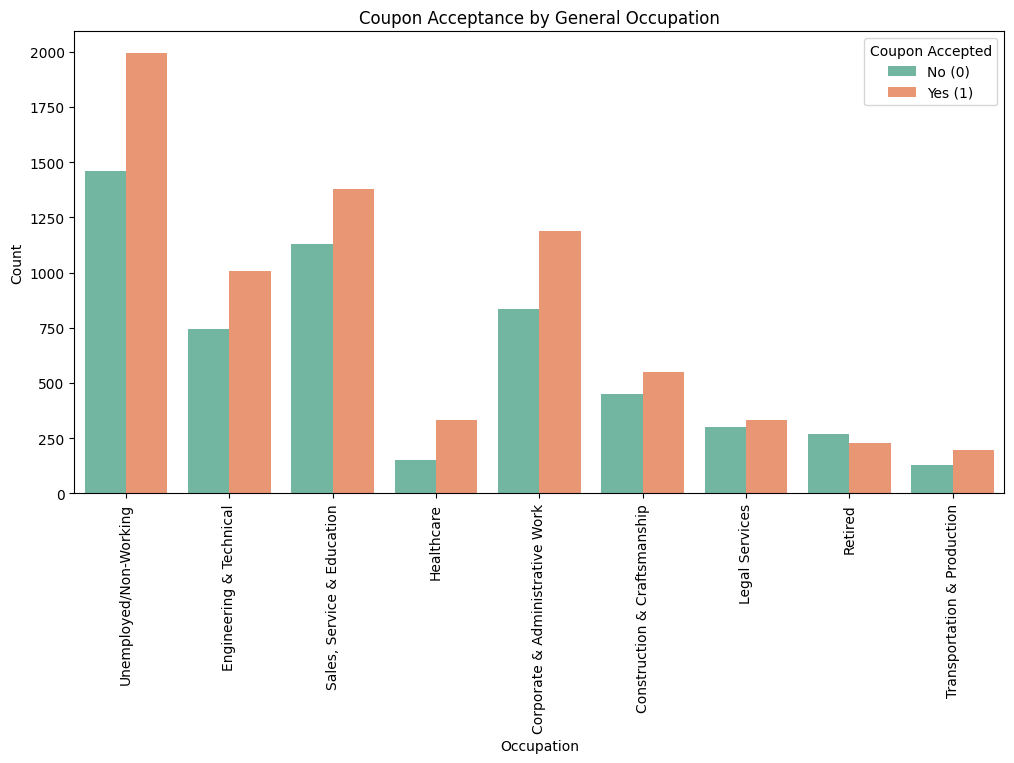
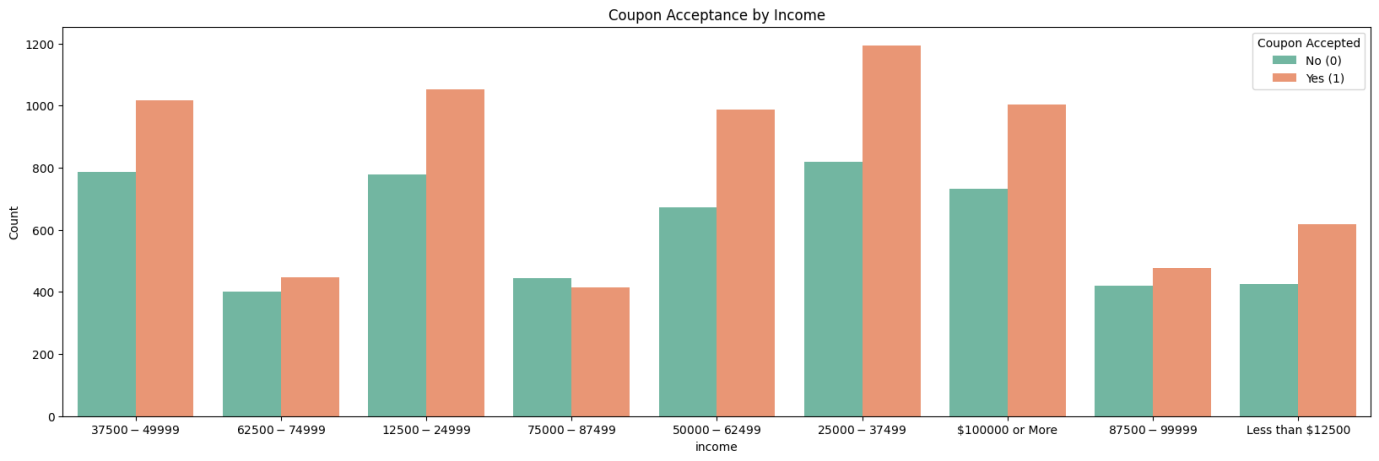
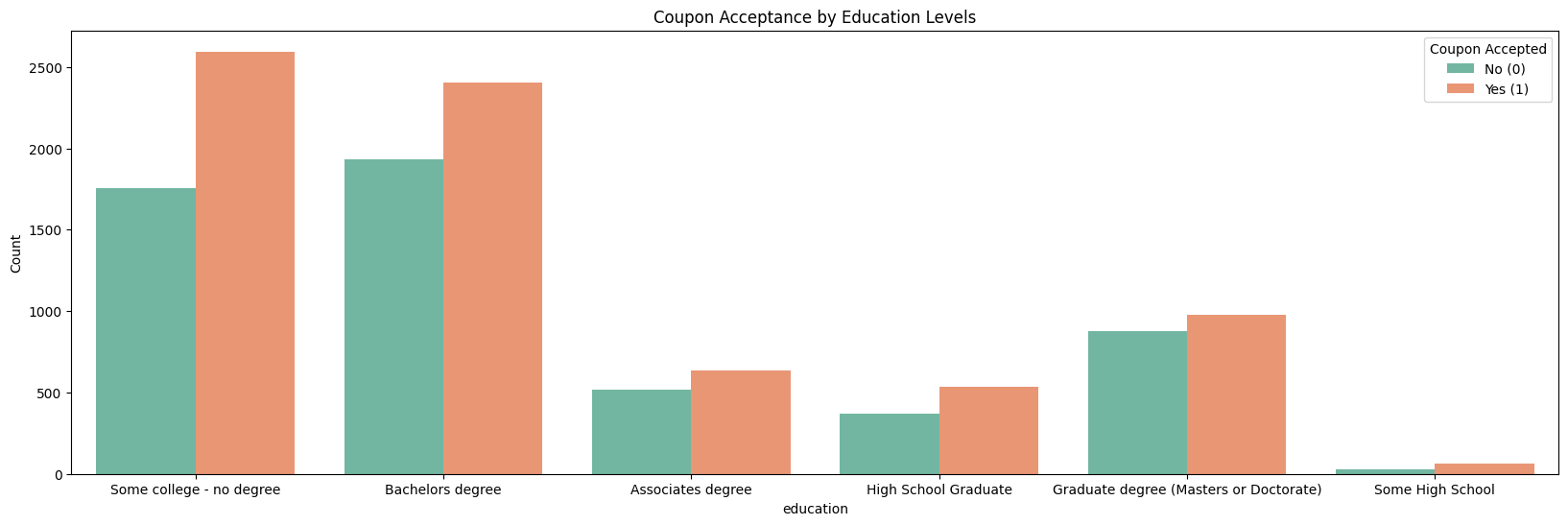
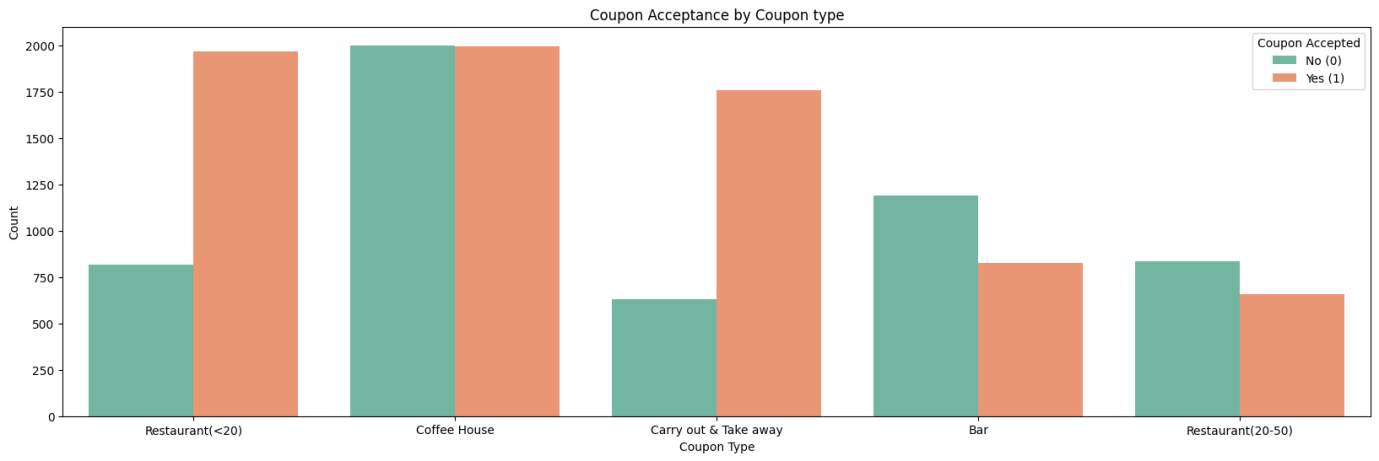


The above count plots reveal that users show a higher likelihood of accepting coupons at 6PM. At 7AM, the probability of users rejecting or accepting coupons are fairly equal. Older users are less likely to accept coupons compared to their younger counterparts and even possess a near equal chance of accepting or declining coupons. Finally, there is a higher probability of a user accepting coupons when the weather condition is sunny.

## 3.2 TREND ANALYSIS

Various trends for categorical features against coupon acceptance were analyzed using count plots. Below are the results:

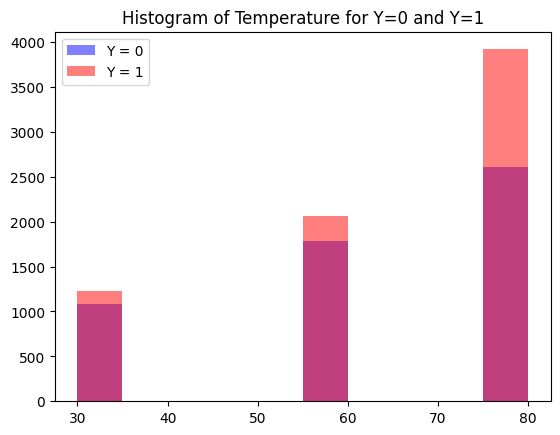
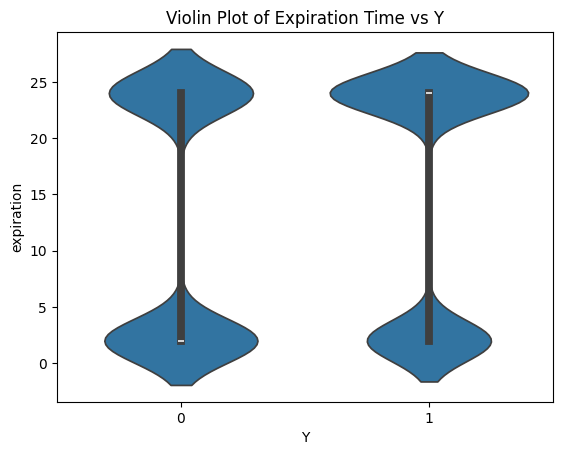


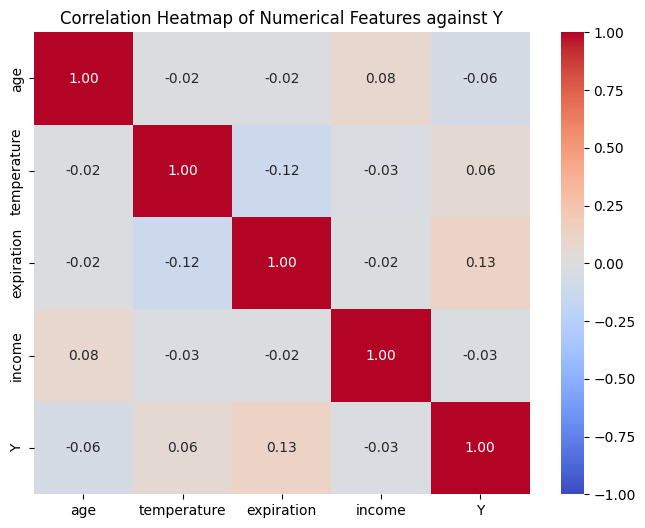


From the stacked bar plots above, it can be inferred that:

* Gender does not necessarily impact coupon acceptance.
* Users are least likely to accept Bar coupons. More favour is directed towards carryout/takeaway coupons and coupons for restaurants that have an average expense per person of less than 20$.
* Users without a degree have a higher probability of accepting coupons compared to users with a masters or doctorate degree.
* Users with an income range of 75k to 87k declined coupons the most while users with an income range of 25k to 37k accepted coupons the most.
* The unemployed and students have a higher rate of accepting coupons. Users who have retired from their job have the highest decline rate.

## 3.3 CORRELATION ANALYSIS

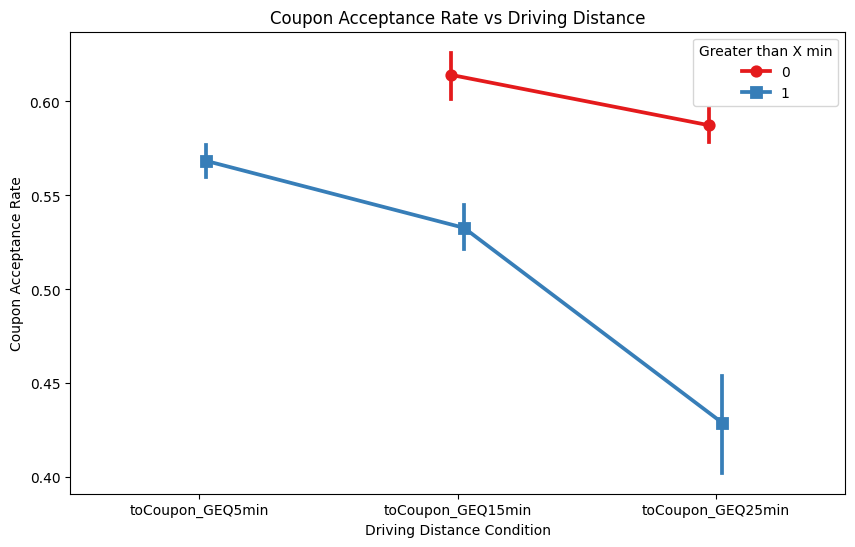
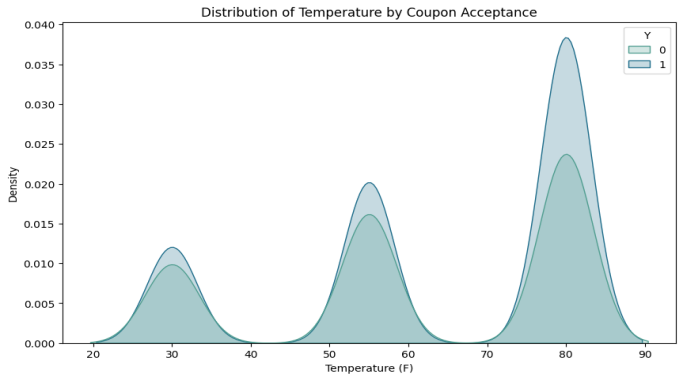




* The violin plot of Expiration time against the target variable Y indicates a higher density of users accepting coupons that expire in 24 hours while a higher density of users declining coupons that expire in 2 hours. This proposes that a longer time before coupon expiration increases its acceptability.
* The steady increase in height of bars as the temperature rises in the histogram of Temperature(F) for target variable Y shows that coupon acceptance rate increases with temperature.
* There is no correlation between all numerical features in the dataset as displayed by the combined heatmap.

## 3.4 DISTRIBUTION VISUALIZATION

## 



* The KDE(Kernel Density Estimation) plot reveals coupon acceptance rate being highest at temperatures between 70-90F(room temperature).
* The point plot of Coupon Acceptance against Driving Distance shows a steady decline in acceptance the farther away the driving distance to the coupon usage location. It was also noted that there were no declined coupons for distances greater than or equal to 5 minutes.

# 4. DATA MODELING

## 4.1 CREATING TRAIN AND TEST DATASET

The preprocessed dataset was randomly split into two sets: 80% for training and 20% for testing. All features except the target feature was set as the X(independent) variable while the target feature(0 or 1) was set as the Y(dependent) variable

After splitting, the distribution of target variables were checked to ensure they were not skewed. The distribution is as shown below:

| Set | Y= 0 (Not accepted) | Y = 1(Accepted) |
| --- | --- | --- |
| Train | 42.8% | 57.2% |
| Test | 44.5% | 55.5% |

A relatively fair split has been achieved, thus, model training can be commenced.

## 4.2 IMPLEMENTATION AND EVALUATION OF VARIOUS MACHINE LEARNING MODELS

The models used for coupon acceptance prediction and evaluation were as follows:

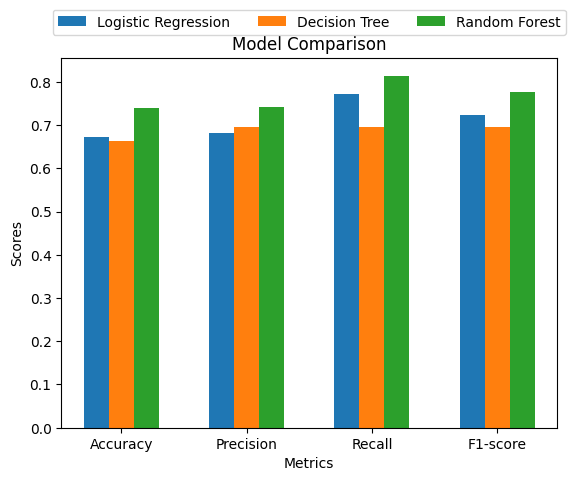
* Logistic Regression:A basic linear model used for binary classification tasks. In this case, predicting coupon acceptance (Y = 0 or Y =1)
* Decision Tree Classifier: This classifier works by spliting the dataset into branches based on various feature values to make predictions.
* Random Forest Classifier: Random Forest operates by building multiple decision trees and taking the mean of their predictions. This reduces overfitting while improving accuracy.

Each model was fit to the train set and the following metrics were used to evaluate each model’s performance:

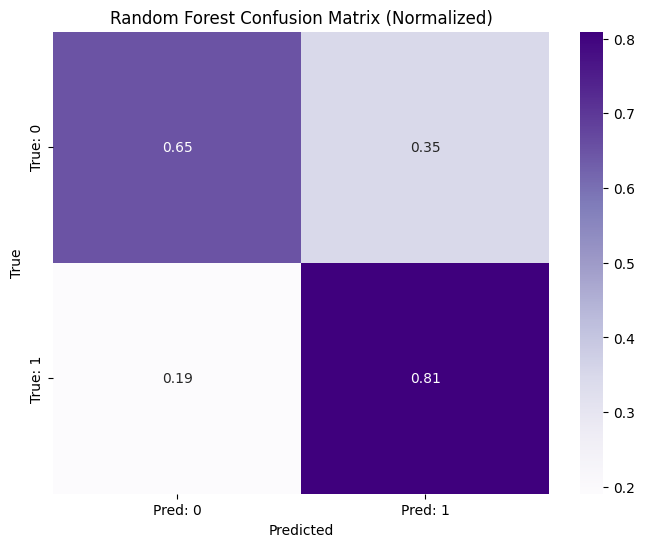
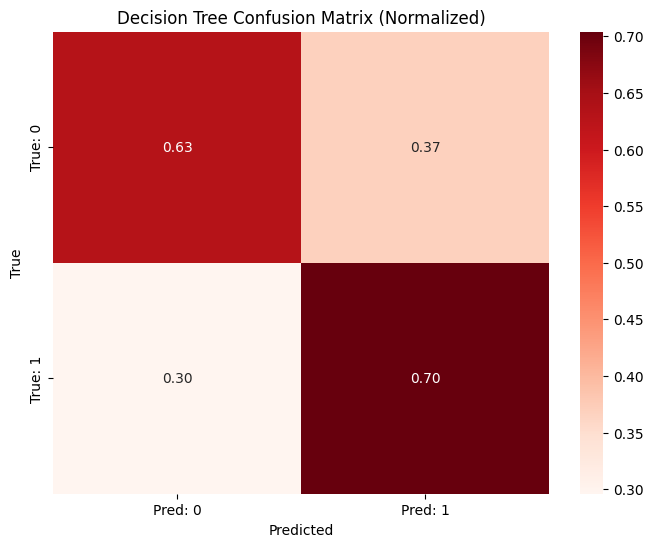
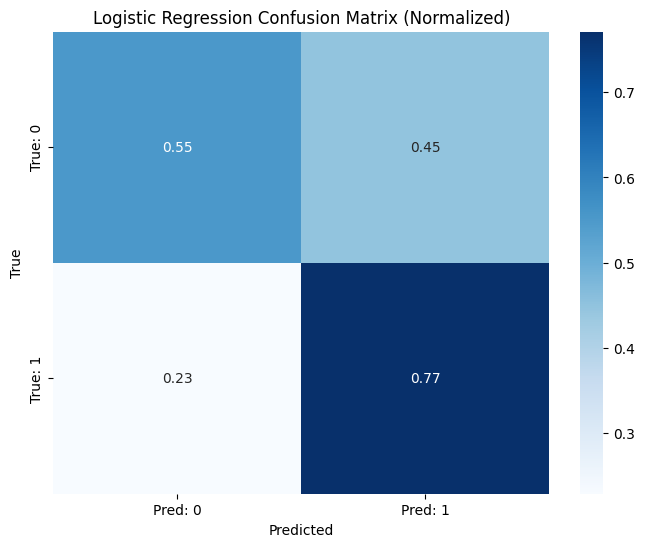
* Accuracy:This represents the rate at which the model makes correct predictions. Out of all the predictions, how many were true?
* Precision: This represents the proportion of true positives to the amount of total positives that the model predicts.Out of all the positive predictions(predicted accepted coupons), how many were actually true(actually accepted)?
* Recall: Also referred to as ‘true positive rate’, Recall represents the rate the model correctly predicts. Out of all the Y labels that are true(coupons that were actually accepted), how many did the model correctly predict as true?
* F1-score: As there is a trade-off between precision and recall, F1-score is used to measure the effectiveness of each model’s tradeoff.

Below is a table comparing the initial accuracy, precision, recall, and F1-score of each model.

| Model | Accuracy | Precision | Recall | F1-score |
| --- | --- | --- | --- | --- |
| Logistic Regression | 67% | 68% | 77% | 72% |
| Decision Tree | 67% | 70% | 70% | 70% |
| Random Forest | 74% | 74% | 81% | 77% |

A bar plot comparing each model’s performance was visualized below:

A confusion matrix was also applied on each model to gather further inference on its performance. The diagrams below propose the Random Forest model as the best at predicting true positives while the Decision Tree model produced the most false negatives.

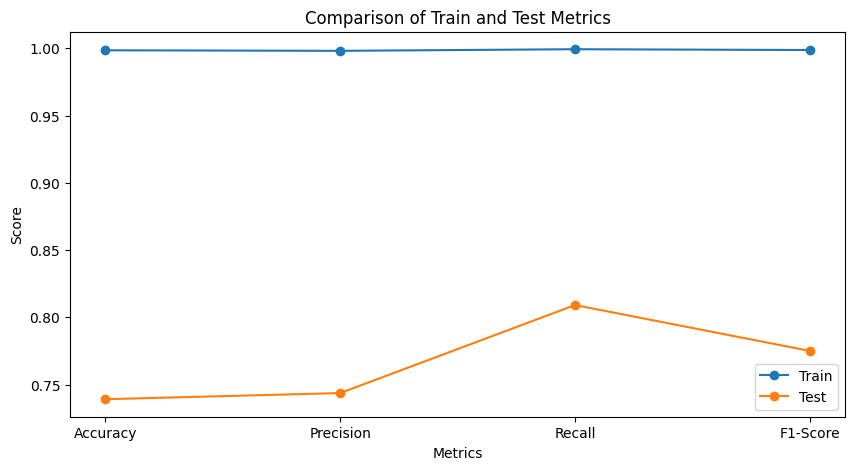


As the Random Forest algorithm performed best out of all three models, it was selected as the classifier for the coupon recommendation system. However, despite the fact the Random Forest classifier performed best, as a standalone model, its metrics are not very impressive.

The low accuracy and precision indicates the random forest model is making a significant number of incorrect predictions, specifically in the form of predicting a lot of false positives.

The model's higher recall however suggests the model is fairly good at detecting actual positive instances. The top two reasons behind this phenomenon are as explained below:

* Class imbalance: The ‘in-vehicle-coupon-recommendation.csv’ dataset has a significant class imbalance as the ratio of Y(1) to Y(0) is 6:4. Therefore, the model prioritizes detecting the majority class(positive instances).
* Overfitting: The model might be overfitting to the training data, resulting in poor generalization to new, unseen data. A plot comparing the Random Forest test and train metrics is shown below:



The graph shows the model delivered a near 100% performance across all metrics with train data but performed poorly with test data. This indicated overfitting as the model fit a bit too well with the train data to the point of *memorization*. The solution to this is reducing the model’s complexity through *hyperparameter tuning*.

## 4.3 HYPER PARAMETER TUNING.

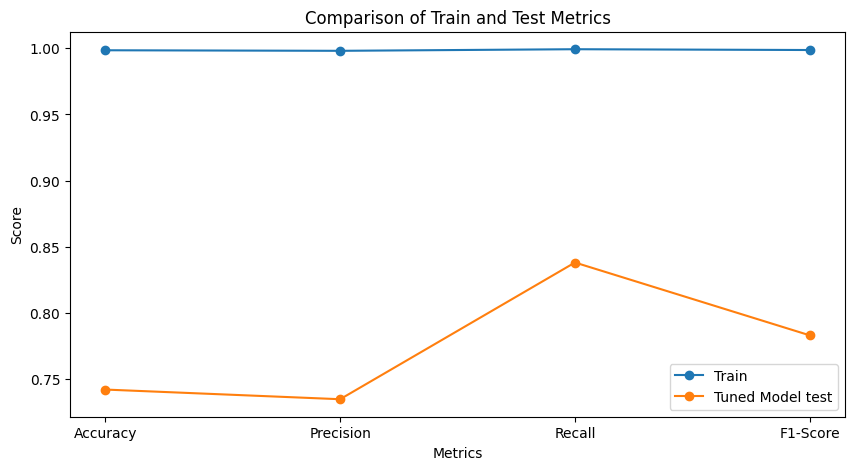
For this section, the GridSearchCV algorithm will be implemented. In this method, cross-validation (k-fold CV) is used to search for the best combination of hyper parameters based on a predefined range of parameters and scoring metric(usually f1-score). This reduces the risk of overfitting/underfitting and simplifies the rigorous task of hyperparameter tuning by automating the search. The hyperparameters available for manipulation in this algorithm are:

* n\_estimators:This defines the number of trees in the forest. Variance increases with the tree count.
* max\_depth: This parameter limits tree growth. The deeper the tree, the more complex its patterns which may lead to overfitting. In return, shallow trees could cause underfitting as its structure is too simple.
* min\_samples\_split: The minimum number of samples required to split an internal node. This value is inversely proportional to model complexity. The higher the values, the simpler the model which reduces overfitting.
* min\_samples\_leaf: The minimum number of samples required to be at a leaf node. Increasing leaf sizes also prevents overfitting as the model does not learn overly specific patterns.
* Bootstrap: This determines whether bootstrap samples (sampling with replacement) are used to build each tree. Setting this parameter to *True* helps reduce overfitting since each tree views a random data subset.
* Scoring: This is an evaluation metric that specifies what metric of model performance to optimize during hyperparameter tuning.

After several combinations, it was noted that the precision and accuracy always remained the same or decreased. This proposes that the upper limit for precision and accuracy for the Random Forest model applied on the coupon recommendation system is at the late 70 percentile when using *GridSearvhCV*. This is also due to the nature of the dataset as discussed before.

However, recall and f1-score showed some improvement with a 3% increase in the former and 1 % increase in the latter. The final combination of hyperparameters were as follows:

* n\_estimators: 200
* max\_depth: 25
* min\_samples\_split: 5
* min\_samples\_leaf: 1
* Bootstrap: True

A new graph comparing train and test metrics applied on the fine tuned model is shown below:

Finally, here is a table comparing all four model performances:

| Model | Accuracy | Precision | Recall | F1-score |
| --- | --- | --- | --- | --- |
| Logistic Regression | 67% | 68% | 77% | 72% |
| Decision Tree | 67% | 70% | 70% | 70% |
| Random Forest | 74% | 74% | 81% | 77% |
| Tuned Random  Forest | 74% | 73% | 84% | 78% |

# 5. CONCLUSION AND RECOMMENDATIONS

This project possessed the sole aim of predicting coupon acceptance based on a variety of features such as age, coupon type, weather using machine learning algorithms.

The process encompassed data understanding and cleaning down to model building and fine tuning. After the evaluation of Logistic Regression, Decision Tree, Random Forest, and fine-tuned Random Forest models, it was found that the fine tuned Random Forest model provided the best results. The model produced the highest F1-score which indicates a balanced trade off between precision and recall. This improvement from the original Random Forest was a direct result of optimizing hyperparameters such as the number of trees, max depth, and minimum samples per split using *GridSearchCV*.

The Random Forest model, with its ensemble nature, demonstrated the ability to handle complex relationships in the data more effectively than simpler models like Logistic Regression and Decision Tree. The fine-tuning process further enhanced its predictive accuracy by minimizing overfitting and improving generalization on the test data.

Further improvements should focus on the issue of class imbalance in the dataset, using solutions like *BalancedRandomForestClassifier*, *Stratified Cross-Validation* and reassembling techniques (e.g *SMOTE*) to help the model train and predict better.

Another recommendation is using *RandomizedSearch* instead of *GridSearch* for hyperparameter tuning. The reasoning behind this is not only does this search provide good results with limited computational resources, it is also widely known to be more efficient as it can handle complex relationships between hyperparameters than *GridSearch*. It also balances exploration (searching new areas of hyperparameter space) and exploitation (refining existing good solutions).

# 6. GITHUB REPOSITORY

The GitHub link to view the code and cleaned dataset for this project is available here:

https://github.com/Ahny678/COUPON-RECOMMENDATION-SYSTEM