Arielle Capati 2010455
Jarrett Sanchez 20109664

Assignment Two

Semester 1 2022

Student Names:
Student IDs:
PAPER NAME: **Foundations of Data Science**

PAPER CODE: COMP615

Due Date: 5th Jun 2022 (midnight NZ time)

TOTAL MARKS: 100

INSTRUCTIONS:

1. **The following actions may be deemed to constitute a breach of the General Academic Regulations Part 7: Academic Discipline,**
   - Communicating with or collaborating with another person regarding the Assignment
   - Copying from any other student work for your Assignment
   - Copying from any third-party websites unless it is an open book Assignment
   - Uses any other unfair means
2. **Please email DCT.EXAM@AUT.AC.NZ if you have any technical issues with your Assessment/Assignment/Test submission on Canvas immediately**
3. **Attach your code for all the datasets in the appendix section**.

ASSIGNMENT TWO

Group assignment

Semester 1, 2023

Due Date: Midnight Sunday 4th Jun 2023.

Late submissions will incur a 10 marks penalty per day.

Weighting: 25% of the final course mark

Submission: When submitting the assessment, **the names and student IDs must be indicated on the front page of the report**.

Naming of Submitted File: DatasetName_Student1Surname_ID_ Student2Surname_ID
Appendix: Submit the code separately.

Assessment of Contribution

Each team will complete and submit a peer assessment form shown below. This will be used to adjust individual grades when there is an agreement of exceptional or underperformance in a team.

		Names of team members	
PERFORMANCE CRITERIA	Weighting	***Jarrett***	***Arielle***
Effort Contribution to tasks, taking initiative, responsibility, completion of tasks on time	**35%**	17.5%	17.5%
Commitment Reach consensus Cooperate	**20%**	10%	10%
Intellectual input Input during discussions Provides feedback Good listener	**25%**	12.5%	12.5%
Attitude Provides positive reinforcement Builds the self-esteem of the partner Builds team cohesion	**20%**	10%	10%
TOTAL	**100%**	50%	50%
Overall rating[1]		Excellent	Excellent

[1]**Overall Rating:** Please rate the degree to which you and your partner fulfilled the responsibilities in completing the assignment. The possible ratings are as follows:

Excellent: Consistently went above and beyond—carried more than the fair share of the load
Very good: Consistently did what was supposed to be done, very well prepared and cooperative.
Satisfactory: Usually did what was supposed to be done, acceptably prepared and cooperative
Marginal: Sometimes failed to complete tasks, rarely prepared
Unsatisfactory: Consistently failed to complete tasks, unprepared
No show: No participation at all

The rating should reflect each individual's level of participation and effort and sense of responsibility, not his or her academic ability.

Team members' signatures:

Name: Jarrett Sanchez Signature: ____________ Date 6/6/23
Name: Arielle Capati Signature: Arielle Date 6/6/23

Both members must agree and sign the document. This document states each team member's participation in the project. The % participation will be used to determine the assessment mark for the individual group member. **If agreement cannot be achieved, students will need to discuss the matter with the paper leader *before* the assignment is due.**

AIMS

This assignment allows you to solve two real-world problems using the machine learning workbench. The analysis and justifications of your answers carry a high proportion of the marks awarded. Please make sure you read through the entire assignment before you start.

This assignment should be completed in pairs (maximum of two students). You will be given a dataset once you have emailed and notified us of the name of your partner.

Project Report – Basic Structure

Your final report must include the following sections and information.

- ✓ Use provided cover page on Canvas
- ✓ Paper Code, Paper Name, and Semester
- ✓ Students' Names and IDs
- ✓ Table of Contents, List of Figures and Tables
- ✓ Appendix (Append your clean code with comments (no screenshot))

There will be **5 marks** for the presentation of the assignment including spelling and grammar, layout, formatting, and readability of the figures.

COMP615 | Foundations of Data Science | Semester 1, 2023

In-Vehicle Coupon Recommendation Analysis

Arielle Capati | 20104555 and Jarrett Sanchez | 20109664

Table of Contents

1 Introduction

The use of 'discount coupons' is a strategic marketing approach to maintain customer retention and boost sales. There's an estimation that 60% of consumers are likely to try a new 'product' because of a 'discount coupon' (Epstein, 2022). Discount coupons are designed to entice customers into visiting a "brick-and-mortar" stores (Epstein, 2022), however, there is still quite a large percentage of rejections. In this report, we want to examine and address real-world problems using machine learning; we aim to analyse factors which influence the acceptance rate and predictability of the model.

1.1 Dataset Used

The data set is provided by UCI Machine learning Repository consists of features which display the characteristics of users who are given discounted coupons in different driving scenarios. The coupons which were offered to the users in the survey were for a Bar, Coffee Shop, Restaurant and Take-away restaurant. The dataset contains a diverse range of attributes which provides precise and detailed information regarding giving a coupon recommendation. It could help determine whether the coupons being offered by the shops are considered a "beneficial marketing strategy". The dataset includes various types of data which includes general information regarding the coupon or scenario, user-context features and demographics, geographic features.

1.2 Problem

From the 12,684 instances, according to our calculations, only 56.8% of that figure accepted the coupon. The problem with this figure is that we feel that businesses could attain a higher acceptance rate from users, however, there is a lot of different possible factors in which could have influenced the user's acceptance rate of the coupon. This could be from the target audience its given to, the time its given and expiration, etc. We would want to investigate if we can use machine learning to predict what features would influence this rate and to provide insightful business recommendations. We will be classifying the "Y" class, which determines whether the coupon is accepted (1), or not (0).

1.3 Research Question

Therefore, we formulated a research question which we will answer in the report:

"What is the predictive performance comparison between the decision tree classifier and the neural network in estimating the in-vehicle coupon acceptance rate and what features are significant in determining this rate?"

1.4 Methods for Analysis

Our objectives include using python language to create a decision tree classifier: building a model using the decision tree algorithm and providing a confusion matrix. Comparatively, an artificial neural network (ANN) with a 10-fold cross-validation will be created and we will compare the performance of these two models for our dataset and use the following models to support our analysis and give insightful business recommendations.

2 Data Exploration

2.1 Attributes and Datatype

Table 1: Attributes and Datatypes
[Retrieved from UCI Machine Learning Repository: in-vehicle coupon recommendation Data Set]

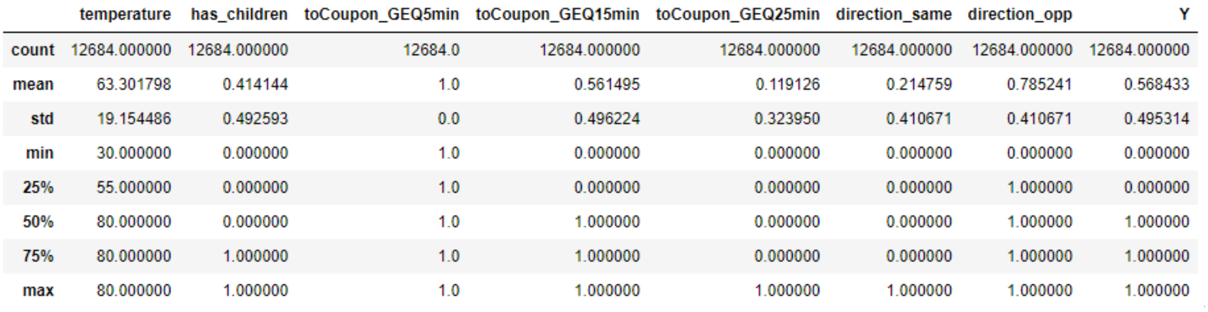
No	Attribute	Datatype	Description	Notes/Values
1	destination	object	-	No urgent Place, Home, Work
2	passenger	object	Passenger in the scenario	Alone, Friend(s), Kid(s), Partner
3	weather	object	Weather condition during the scenario	Sunny, Rainy, Snowy
4	temperature	int64	-	55, 80, 30
5	time	object	-	2PM, 10AM, 6PM, 7AM, 10PM
6	coupon	object	For Restaurants, Bar, Coffee House, Carry out & Takeaway	Restaurant(<$20), Coffee House, Carry out & Take away, Bar, Restaurant($20-$50)
7	expiration	object	The coupon expires in 1 day or in 2 hours	1d, 2h
8	gender	object	-	Female, Male
9	age	object	-	21, 46, 26, 31, 41, 50+, 36, below 21
10	maritalStatus	object	-	Unmarried partner, Single, Married partner, Divorced, Widowed
11	Has_children	int64	If the person has children (where Y=1 is Yes and Y=0 is No)	1, 0
12	education	object	Form of education received	Some college - no degree, Bachelors degree, Associates degree, High School Graduate, Graduate degree (Masters or Doctorate), Some High School
13	occupation	object	-	Unemployed, Architecture & Engineering, Student, Education&Training&Library, Healthcare Support, Healthcare Practitioners & Technical, Sales & Related, Management, Arts Design Entertainment Sports & Media, Computer & Mathematical, Life Physical Social Science, Personal Care & Service, Community & Social Services, Office & Administrative Support, Construction & Extraction, Legal, Retired, Installation Maintenance & Repair, Transportation & Material Moving, Business & Financial, Protective Service, Food Preparation & Serving Related, Production Occupations, Building & Grounds Cleaning & Maintenance, Farming Fishing & Forestry

14	income	object		$37500 - $49999, $62500 - $74999, $12500 - $24999, $75000 - $87499, $50000 - $62499, $25000 - $37499, $100000 or More, $87500 - $99999, Less than $12500
15	car	object	-	-
16	bar	object	feature meaning: how many times do you go to a bar every month?	never, less1, 1~3, gt8, nan, 4~8
17	CoffeeHouse	object	feature meaning: how many times do you go to a coffeehouse every month?	never, less1, 4~8, 1~3, gt8, nan
18	CarryAway	object	feature meaning: how many times do you get take-away food every month?)	n4~8, 1~3, gt8, less1, never
19	RestaurantLessThan20	object	feature meaning: how many times do you go to a restaurant with an average expense per person of less than $20 every month?	4~8, 1~3, less1, gt8, never
20	Restaurant20To50	object	feature meaning: how many times do you go to a restaurant with average expense per person of $20 - $50 every month?)	1~3, less1, never, gt8, 4~8, nan
21	toCoupon_GEQ5min	int64	Driving distance to Restaurant/Bar for using Coupon is greater than 5 minutes	0,1
22	toCoupon_GEQ15min	int64	Driving distance to Restaurant/Bar for using Coupon is greater than 15 minutes	0,1
23	toCoupon_GEQ25min	int64	Driving distance to Restaurant/Bar for using Coupon is greater than 25 minutes	0,1
24	Direction_same	int64	Restaurant/Bar is in the same direction as current direction	1,0
25	Direction_opp	int64	Restaurant/Bar is in the same direction as current direction	1,0
26	Y	int64	Whether the coupon is accepted or not (where Y=1 is Yes and Y=0 is No)	1,0

The dataset contains 26 attributes with 12,684 recorded instances and is considered to have “multivariate characteristics”. Most of these attributes have categorical data types, which also includes numerical data types. The data was collected through a survey on “Amazon Mechanical Turk”. The dataset provides “different driving scenarios including the destination, current time, weather, passenger, etc., and then ask the person whether he will accept the coupon if he is the driver” (Wang et al., 2017).

Note that if the user replies that they would use it eventually “before the coupon expires” or drives there “right away” is labelled by “Y = 1” and on the other hand, situations in which the coupon is rejected will be labelled as “Y = 0”.

We consider five main different categories of businesses which would be categorised by: pubs, takeaway food restaurants, coffee shops, low-cost food restaurants which would be under $20 per person and high-cost restaurants which will vary in the range of $20 to $50 per person. There has also been an indication that there are missing values within the dataset.



	temperature	has_children	toCoupon_GEQ5min	toCoupon_GEQ15min	toCoupon_GEQ25min	direction_same	direction_opp	Y
count	12684.000000	12684.000000	12684.0	12684.000000	12684.000000	12684.000000	12684.000000	12684.000000
mean	63.301798	0.414144	1.0	0.561495	0.119126	0.214759	0.785241	0.568433
std	19.154486	0.492593	0.0	0.496224	0.323950	0.410671	0.410671	0.495314
min	30.000000	0.000000	1.0	0.000000	0.000000	0.000000	0.000000	0.000000
25%	55.000000	0.000000	1.0	0.000000	0.000000	0.000000	1.000000	0.000000
50%	80.000000	0.000000	1.0	1.000000	0.000000	0.000000	1.000000	1.000000
75%	80.000000	1.000000	1.0	1.000000	0.000000	0.000000	1.000000	1.000000
max	80.000000	1.000000	1.0	1.000000	1.000000	1.000000	1.000000	1.000000

Fig. 1: Summary Statistics

Figure 1 show the summary statistics of purely numerical variables (i.e., excludes data on categorical variables as well as answers such as: “less than 1”, “4~8”, “10AM”, “1d”, “2h”, “$37500 - $49999”. Note that aside from temperature, all answers for the other numerical variables are given in the form of 1 (yes) or 0 (no) only.

2.2 Illustration of Features

Graphs that could help determine coupon acceptance rate from the given data features:

- Coupons related visualisations (coupon for what business & expiry)
- How often they go to the places (each business)
- Coupon usage by demographic features (age, education, gender, occupation, income, children, marital status, passenger type)
- Coupon and geographic features (weather, time, temp)
- Coupon and Scenario related features (destination, driving distance < 15 and driving distance < 25, current destination if same)

Coupon-related visualisations (where y=1 and n=0).

Fig. 2: Total Amount of Coupons (by business)

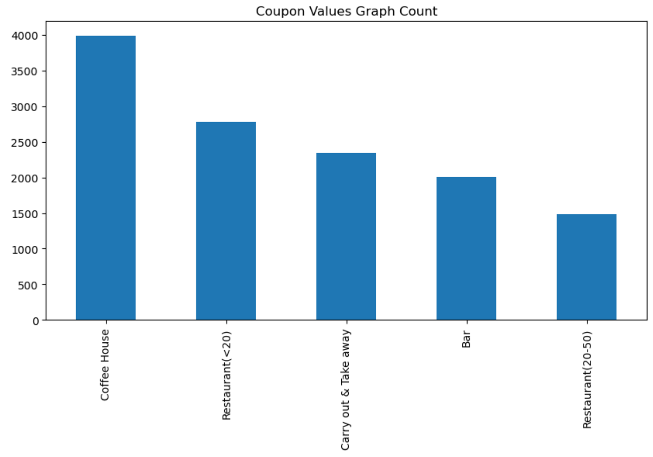


Fig. 3: Amount of coupons used per business

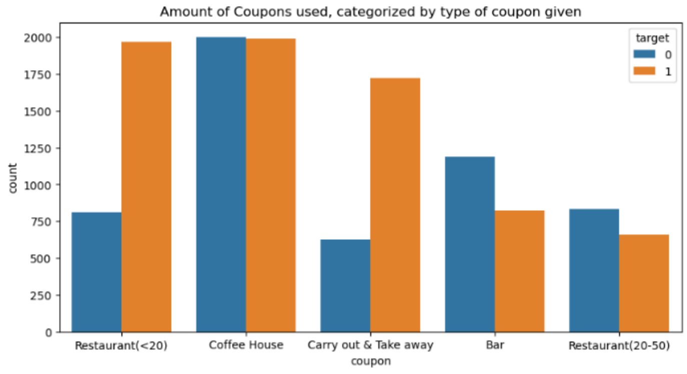


Fig. 4: Coupon Expiration dates

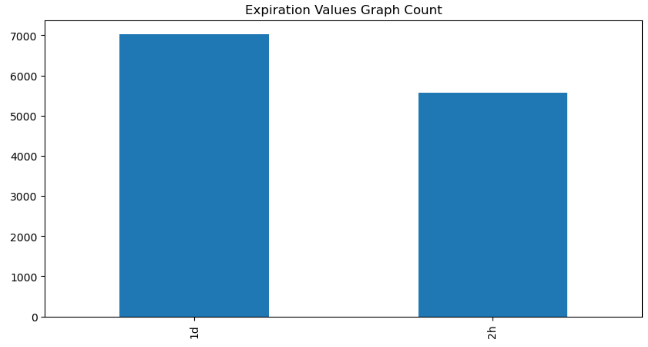
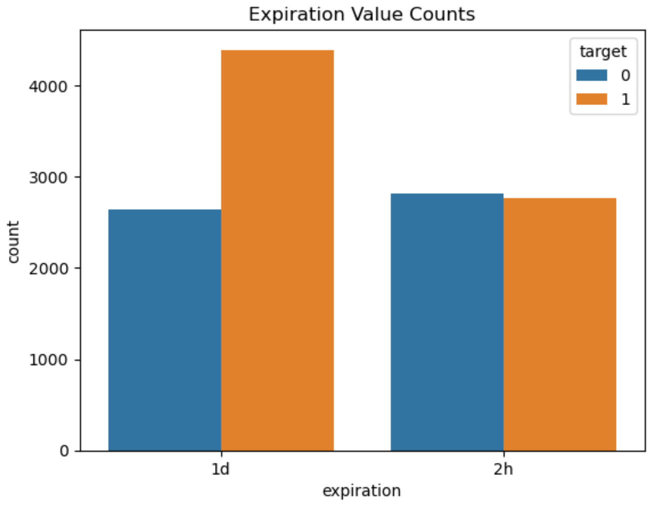


Fig. 5: Coupon used by Expiry



Figures 2-5 allows for “Coupon-related visualizations” where yes is ‘1’ and no is ‘0’. Evidently notable areas of analysis in the following figures would allow us to visualise the total amount of coupons given to each area of business and the difference in those coupons (i.e., regarding expiration date). In figure 2, we see how distributed the coupons are against businesses and in figure 3 it displays the distinct acceptance rates for each area of business. Notably, in restaurants with an average expense per person of less than $20 every month and carry-outs have a high acceptance rate with coffee houses following behind. The acceptance rate appears to be lower for bars and restaurants costing between $20~$50 (per person every month). Figure displays the amount of 1d and 2h coupons given out and in figure 4, we see that the acceptance rate for a coupon that has a 24-hour expiry date almost attracts double the number of potential customers accepting to using the coupon as opposed to a 2 hr one.

Visualising how often the sampled people go to “x” places (where y=1 and n=0).

Fig. 6: Bar

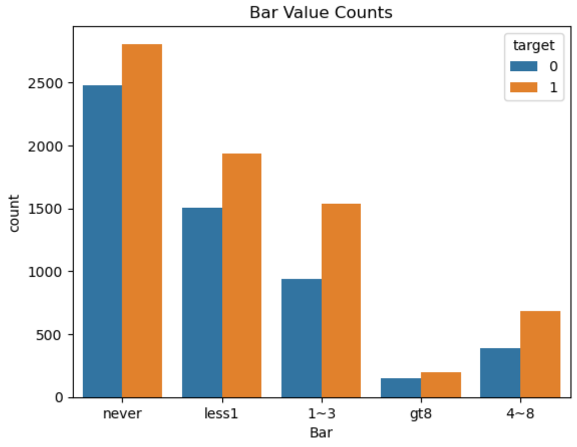


Fig. 7: Coffee House

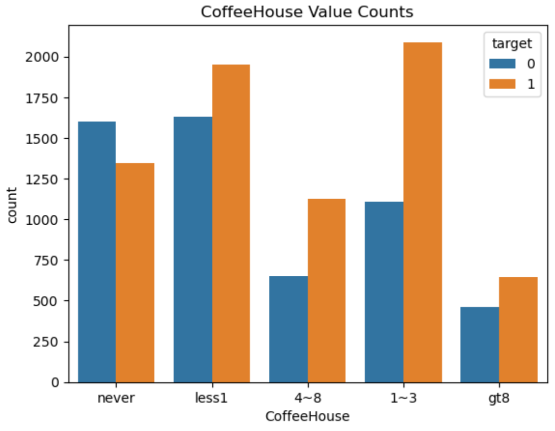


Fig. 8 Carry Away

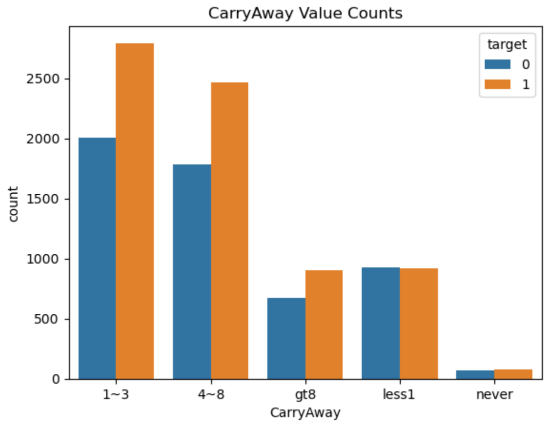


Fig. 9: Restaurant Less than $20

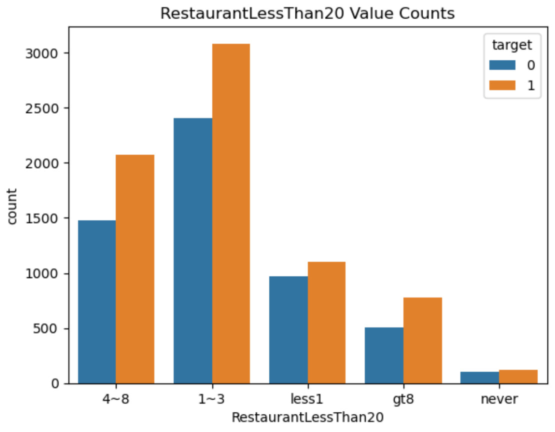
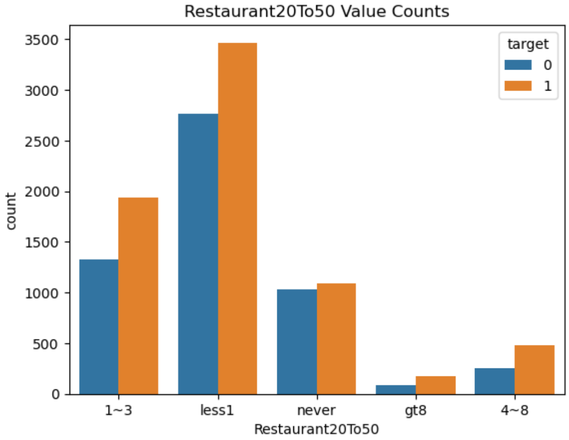


Fig. 10: Restaurants $20 to $50	
	

There appears to be a consistent pattern observed in figures 6-10 indicating that individuals who frequently visit a particular establishment are more likely to accept their discounted coupons. When examining inexpensive restaurants, it becomes evident that there are only a few numbers of individuals who have never visited one but almost the majority would accept the coupon. Analysing coffee-houses, bars, and expensive restaurants, people who have only visited once or have never visited tend to reject their discounted coupons. It appears that acceptance increases as people's monthly frequently of going to these business areas increases. For the carraway coupons, because it appears that, due to the miniscule time amount needed to spend at the restaurant (which may make it easier on the flexibility regarding people's schedule's), it has quite a high acceptance rate.

Demographic (user-context features) visualisation of the sample and coupon usage (where y=1 and n=0).

Fig. 11: Age

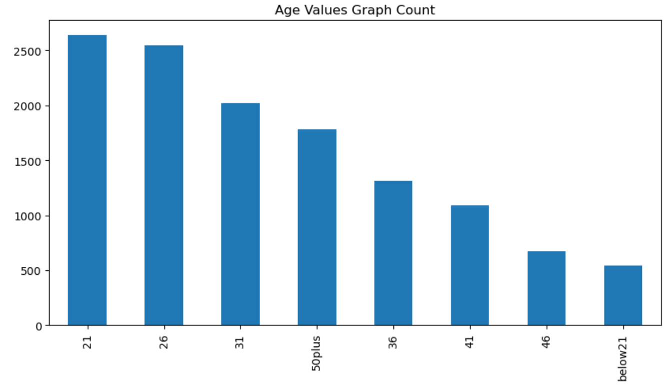


Fig. 12: Coupons used by Age

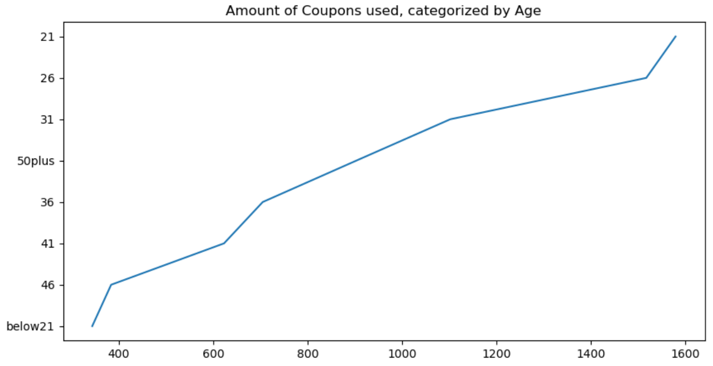


Fig. 13: Education

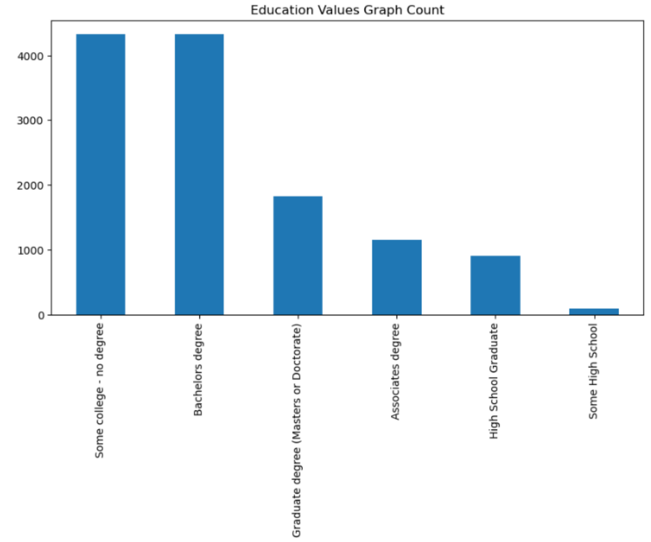


Fig. 14: Coupons used by Education

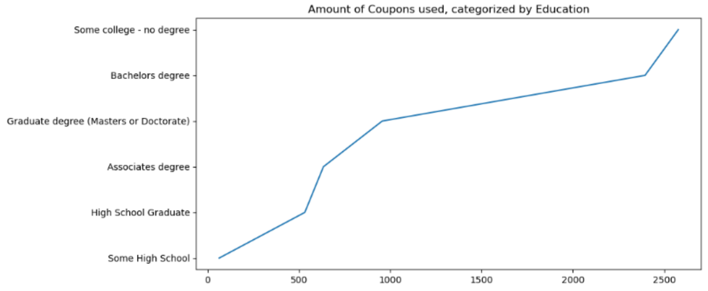


Fig. 15: Gender

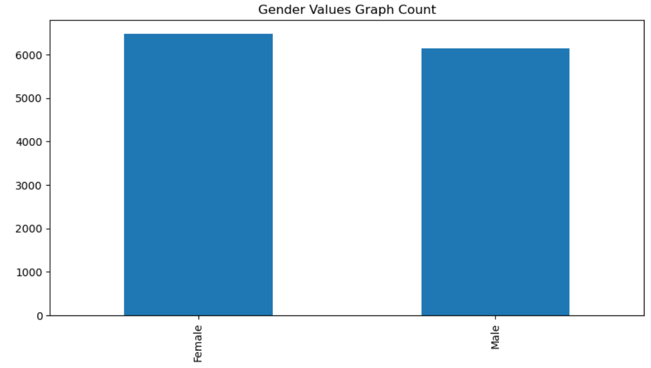


Fig. 16: Coupons used by Gender

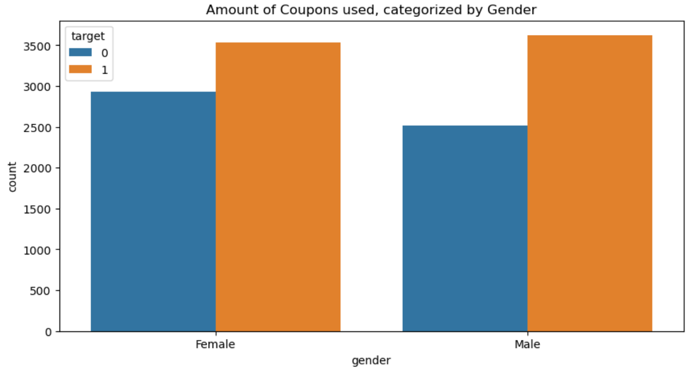


Fig. 17: Occupation

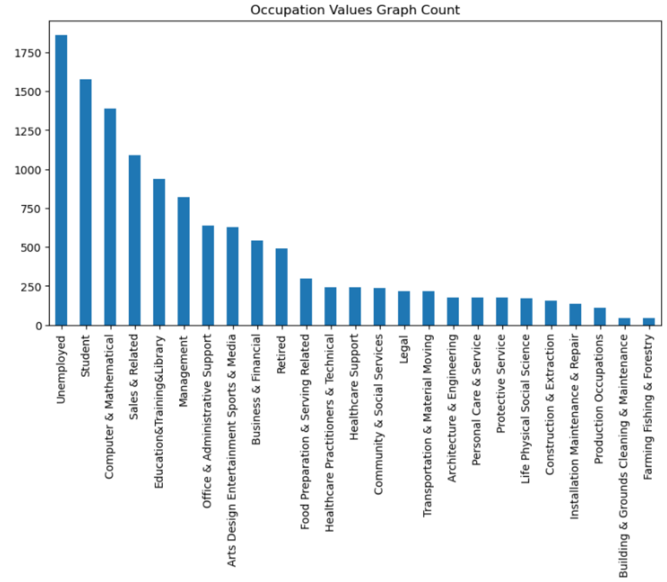


Fig. 18: Coupons used by occupation

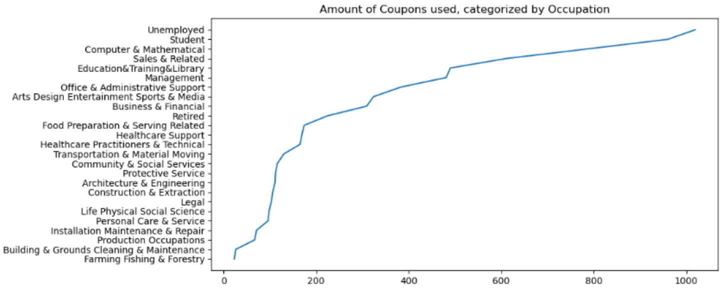


Fig. 19: Income

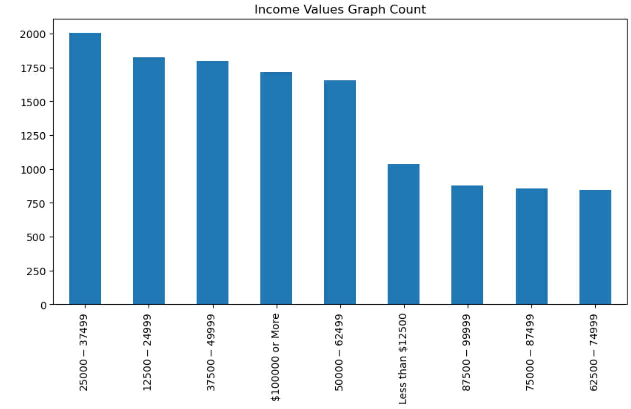


Fig. 20: Coupons used determined by income range

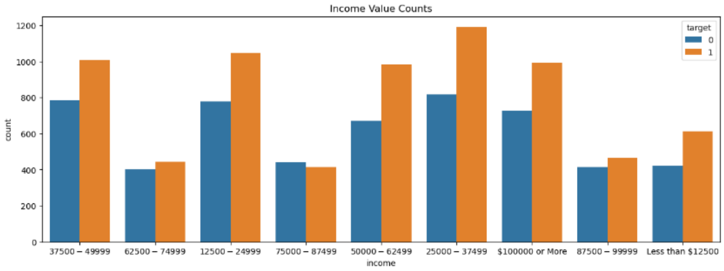


Fig. 21: Marital status

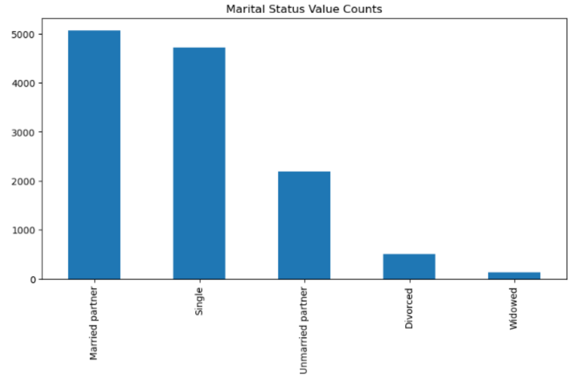


Fig. 22: Coupons used by marital status

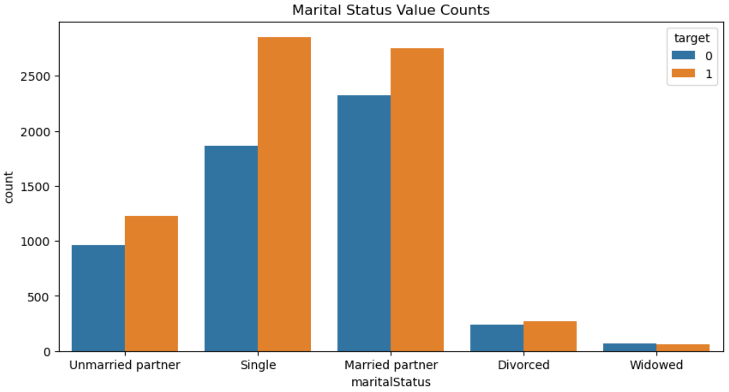


Fig. 23: Passenger Type

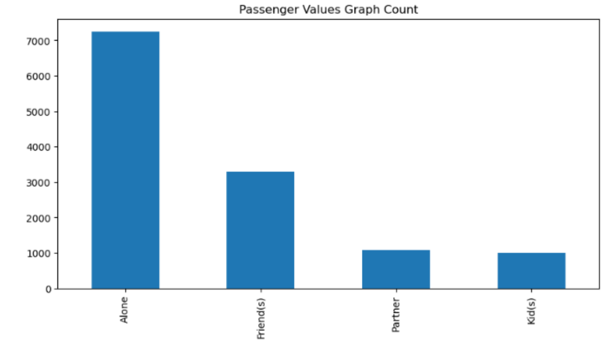


Fig. 24: Used if "x" type of passenger

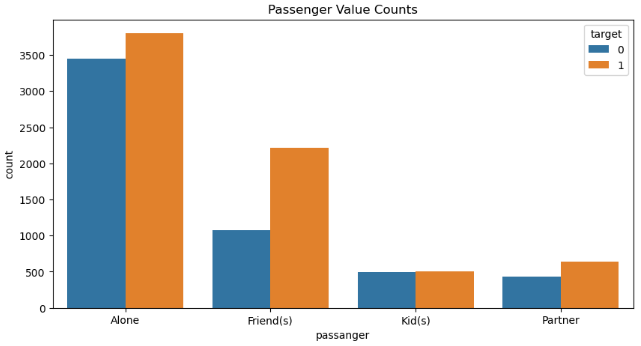


Fig. 25: Person given coupon has children

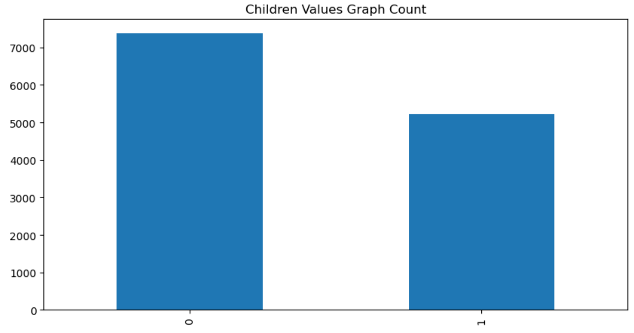
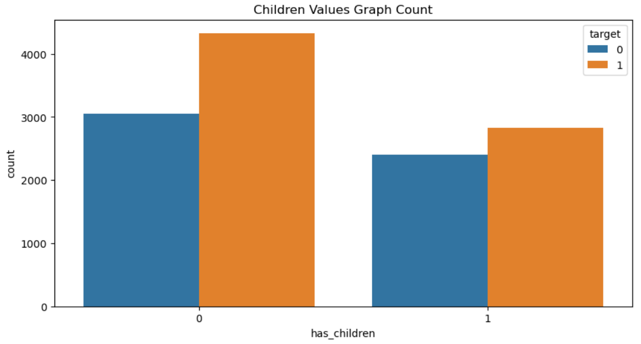


Fig. 26: Coupons used if they have children



The following table above (figures 11-26), provides a graphical representation of the demographic section of the dataset and compares this against the coupon acceptance rate:
Age groups 21 and 26 exhibit the highest frequencies and in observations for education, the highest occurrences are in groups with "no degree" and "bachelor's degree" which are relatively quite similar in count. However, upon examining the coupon acceptance rate (figure 14), it becomes evident that the highest acceptance rate count are from people who do not have a degree. Figure 15 shows that the attribute is roughly split into an even distribution of female and male. Figure 16 evidently shows for males, there is a slightly higher acceptance rate for the coupons than females when comparing the difference between the rejection and acceptance rate for each group respectively. In figure 17 and 18, unmistakably there is a lot of "levels" (i.e., different responses) for the occupation of each individual, with unemployed and students having the highest-level size. In reference to figure 18, there is also quite a high number of coupons being accepted from these two groups. On further analysis, figure 19 illustrates the income brackets of each individual with figure 20 displaying the outcome of the scenario (i.e., if the user accepts the coupon) in relation to their selected range of income. It seems although that the acceptance rate is higher than the rejection rate regardless of one's income, aside from the rare exception of the salary range "$75,000 and $87,499" and "$87,500-$9,999". Lower income ranges such as "Less than $12,500", "$12,500-$24,999", "$25,000-$37,499", "$37,500-$49,999", "$50,000-$62,499" and "$100,000 or more" have significantly higher acceptance rates than their rejection rate. In figure 21 and 22, we analysed that the highest occurrences belong to users who are a "married partner" or is "single" which reflects to the coupons accepted and used with people who are "single" having a marginally high acceptance rate with a person who is a "married partner" following second. For passengers, majority of the passengers are traveling alone looking at figure 23. In figure 24, it seems that the group with the highest acceptance rate, surprisingly is not when the passenger is alone but rather,

when they are with "friends". Lastly, surprisingly, there is many users with children (as depicted by the number 1 in figure 26). When analysing figure 27, although the overall acceptance rate is higher for either group, there is a higher rate of acceptance when the individual has no children.

Geographic features visualisation of the sample and coupon usage (where y=1 and n=0).

Fig. 27: Weather

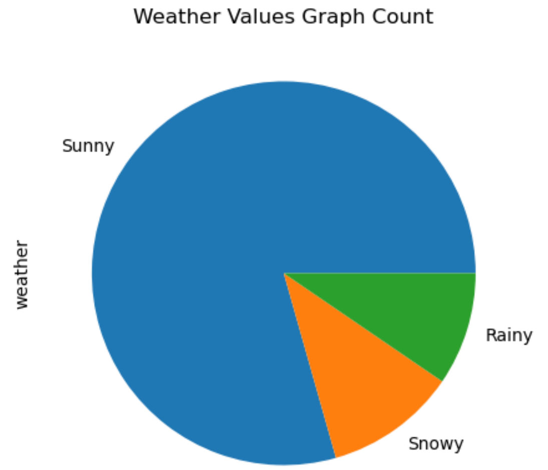


Fig. 28: Weather and Coupon

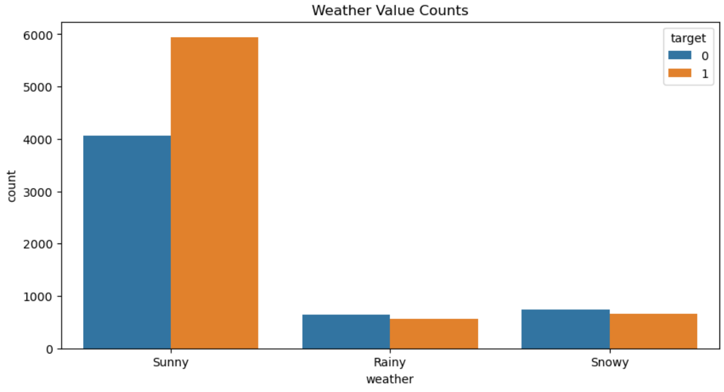


Fig. 29: Time

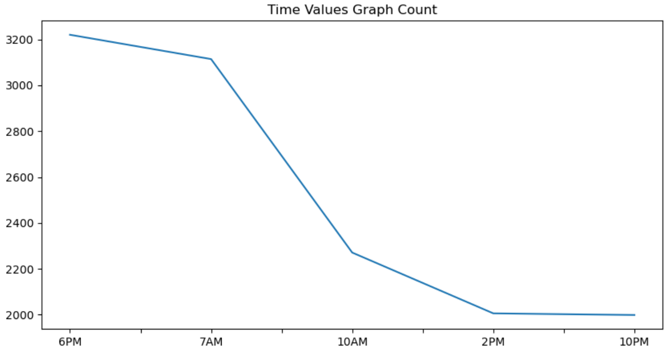


Fig. 30: Time and Coupon

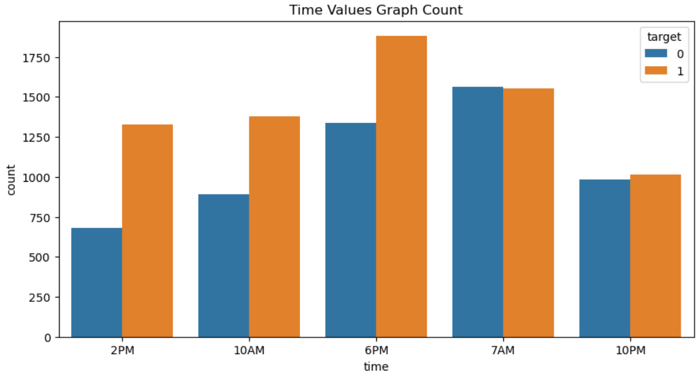


Fig. 31: Temperature

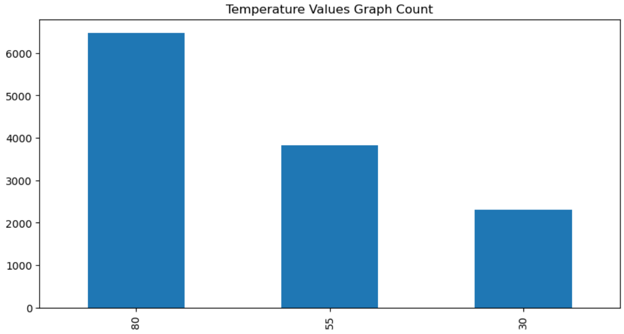


Fig. 32: Temperature and Coupon

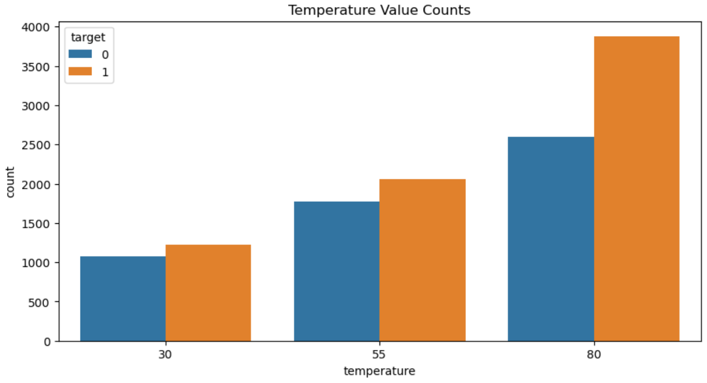


Figure 27 to 32 analyses the geographic features of the scenario per individual. The highest number of observations are obtained when the weather was classified to be “sunny” which is evident in figure 27. For time, in figure 29, most occurrences were at 6pm, following 7am which may be an indicator to “rush hours”. On further analysis, where the weather is sunny, there is a marginally higher acceptance rate as opposed to rainy and snowy weathers where the rejection rate is slightly higher than the acceptance rate. In relation to time and the coupon acceptance rate, 2pm, 10am and 6pm had the highest acceptance rates with 7am and 10pm occurrences being at level with each other. Therefore, this suggests that if the time is too early or late, the probability in which the coupon is accepted is lower – this may impact customer decisions if the business is still yet to open or would close soon too. Examining figure 31 and 32, it appears that in the occurrences the scenario took place, the highest amount recorded was with a temperature of 80. In reference to the acceptance rate, it seems that despite temperature, the coupon’s acceptance rate is higher than the rejection rate, however, looking at a temperature of 80, the acceptance rate is marginally higher.

Scenario-related visualisations of the sample and coupon usage (where y=1 and n=0).

Fig. 33: Destination

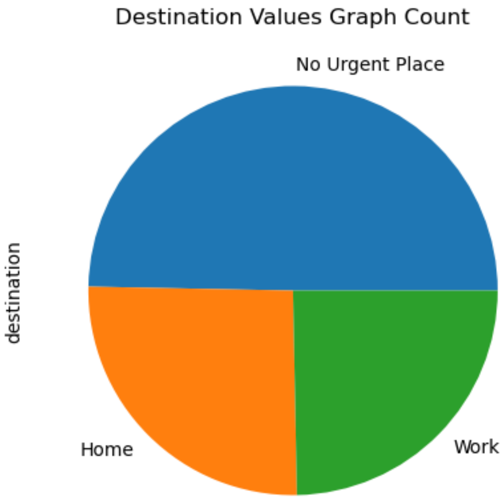


Fig. 34: Coupons used dependent on destination

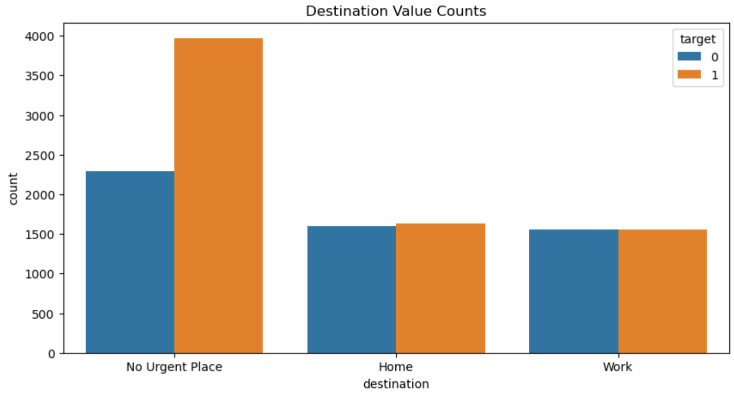


Fig. 35: Where driving distance to restaurant/bar is greater than 15 mins

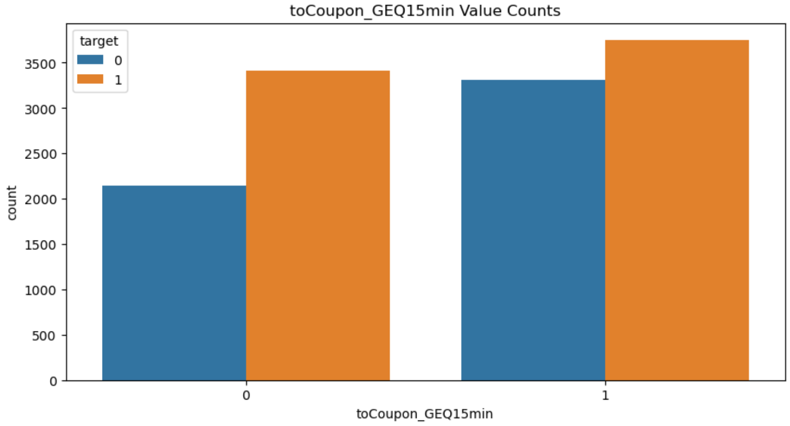


Fig. 36: Where driving distance to restaurant/bar is greater than 25 mins

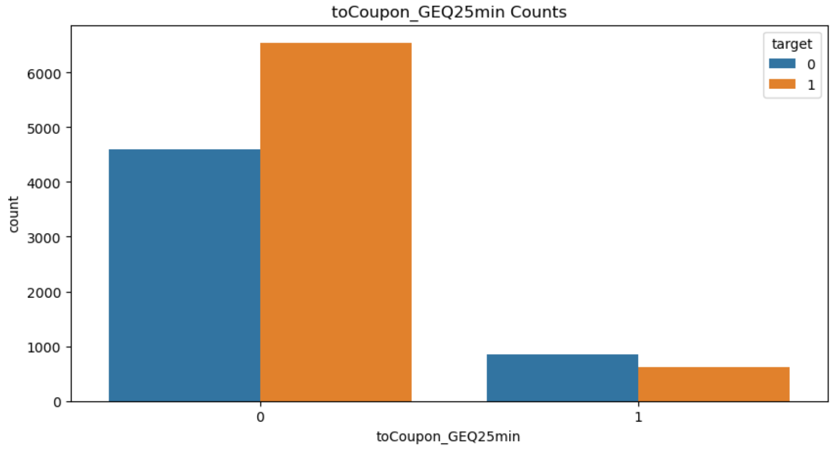


Fig. 37: Where the direction is the same as their current destination

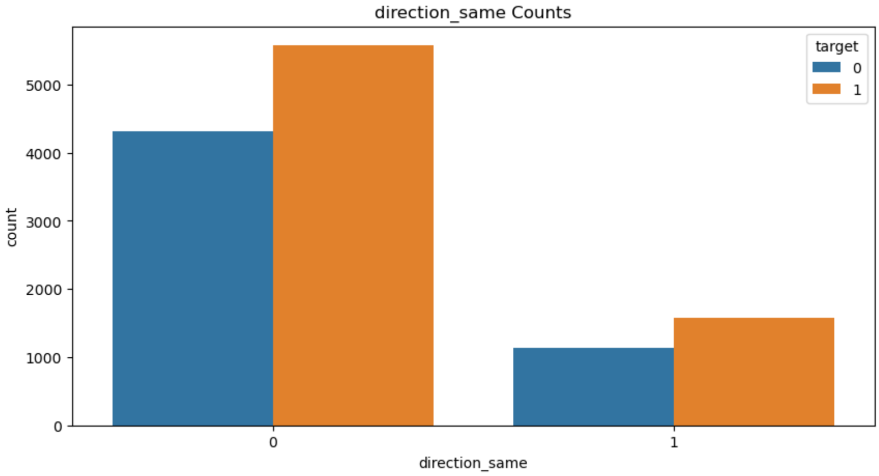
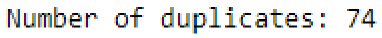
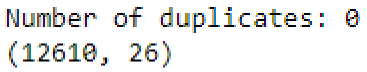
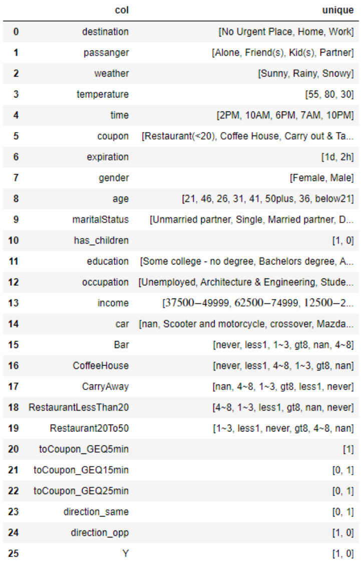


Figure 33 to figure 37 provides visual representation of scenario-based features. It shows that approximately one-third of the samples has no urgent place to be at that time, whilst another one-third was headed to work, and the remaining one-third was headed home. It is important to note that these figures were captured and influence at different times throughout the day. In figure 34, where the user had no urgent place to be, we notice that the acceptance rate was significantly higher as opposed to people who had to get to work or home. Interestingly, figures 35 and 36, an inverse relationship is observed. As "toCoupon_GEQ5min" was dropped, analysing the graphs, we notice that there were more people who accepted the coupon if the driving distance was at a greater than 15 minutes but less than 25. And the ratio for a driving distance of 25 minutes and greater was lower. This implies that if the driving distance increases, then coupon acceptance rate in turn, decreases (hence why there is more rejection seen in figure 36. In figure 37, where the direction is the same (1) and direction is opposite (0), we notice that the acceptance rate is still slightly higher regardless of their destination.

2.3 Data Pre-Processing

Fig. 38: Original Dataset Duplicates Count	**Fig. 39: After dropping Duplicates Count**
 Number of duplicates: 74	 Number of duplicates: 0 (12610, 26)

The first step in data pre-processing is to **handle duplicated values**. To do this, we can use the function "drop_duplicates()" which brings down the total of 74 to 0. Using the ".shape" function, we now see that the instances have dropped from 12,684 to 12,610 meaning that the drop was successful.



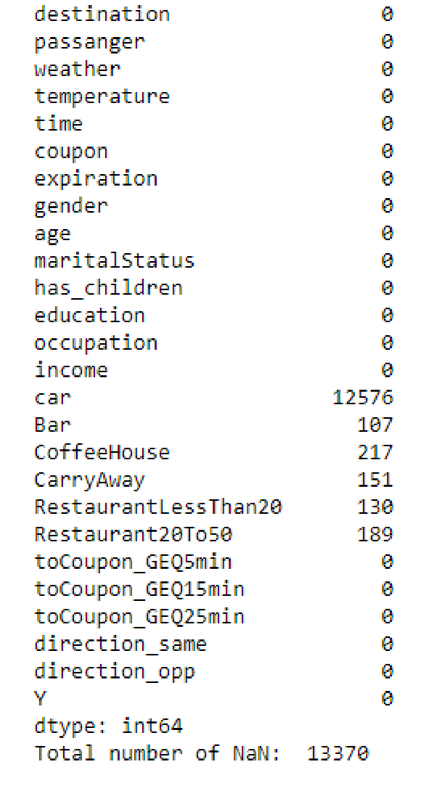
	col	unique
0	destination	[No Urgent Place, Home, Work]
1	passanger	[Alone, Friend(s), Kid(s), Partner]
2	weather	[Sunny, Rainy, Snowy]
3	temperature	[55, 80, 30]
4	time	[2PM, 10AM, 6PM, 7AM, 10PM]
5	coupon	[Restaurant(<20), Coffee House, Carry out & Ta...
6	expiration	[1d, 2h]
7	gender	[Female, Male]
8	age	[21, 46, 26, 31, 41, 50plus, 36, below21]
9	maritalStatus	[Unmarried partner, Single, Married partner, D...
10	has_children	[1, 0]
11	education	[Some college - no degree, Bachelors degree, A...
12	occupation	[Unemployed, Architecture & Engineering, Stude...
13	income	[37500−49999, 62500−74999, 12500−2...
14	car	[nan, Scooter and motorcycle, crossover, Mazda...
15	Bar	[never, less1, 1~3, gt8, nan, 4~8]
16	CoffeeHouse	[never, less1, 4~8, 1~3, gt8, nan]
17	CarryAway	[nan, 4~8, 1~3, gt8, less1, never]
18	RestaurantLessThan20	[4~8, 1~3, less1, gt8, nan, never]
19	Restaurant20To50	[1~3, less1, never, gt8, 4~8, nan]
20	toCoupon_GEQ5min	[1]
21	toCoupon_GEQ15min	[0, 1]
22	toCoupon_GEQ25min	[0, 1]
23	direction_same	[0, 1]
24	direction_opp	[1, 0]
25	Y	[1, 0]

Fig. 40: Checking for Unique Values Count

Secondly, it is important to also **drop variables** which has a significant number of missing values, no variation or features which become redundant as they give the exact same information as another column.

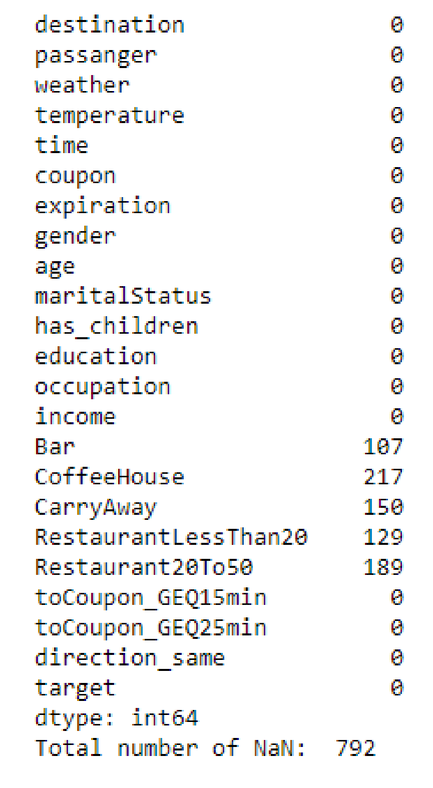
In this case "car" has a lot of missing values with a total of 12,576 as seen in figure 41, therefore it will be ultimately removed. The column "toCoupon_GEQ5min" will also be dropped as there is no variation within their responses – we can see that in figure 40 when we check for unique values, there is only a response of 1 (yes), therefore this will be removed. Lastly, the column "direction_opp" will be removed as this gives the opposite answers to the column "direction_same", thus suggesting perfectly correlated columns (when the other one is 0 then the other one is 1). Therefore, if we have one of the columns mentioned, it would be ideal to remove one of the columns to avoid unneeded repetition.

Fig. 41: Original Dataset Missing Values Count



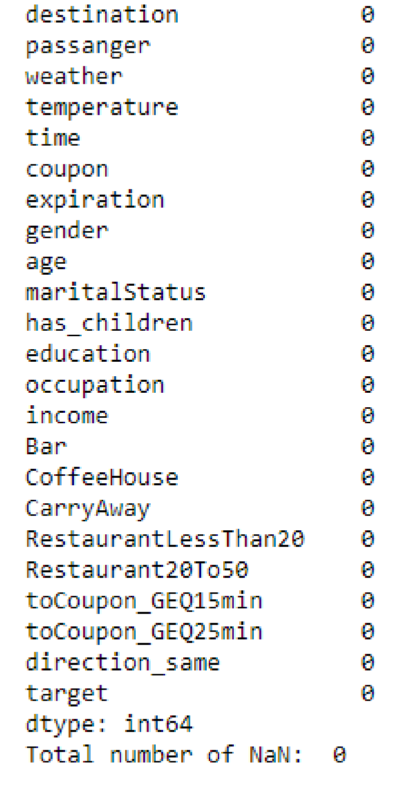
```
destination                  0
passanger                    0
weather                      0
temperature                  0
time                         0
coupon                       0
expiration                   0
gender                       0
age                          0
maritalStatus                0
has_children                 0
education                    0
occupation                   0
income                       0
car                      12576
Bar                        107
CoffeeHouse                217
CarryAway                  151
RestaurantLessThan20       130
Restaurant20To50           189
toCoupon_GEQ5min             0
toCoupon_GEQ15min            0
toCoupon_GEQ25min            0
direction_same               0
direction_opp                0
Y                            0
dtype: int64
Total number of NaN:   13370
```

Fig. 42: After Removing Duplicates Count



```
destination              0
passanger                0
weather                  0
temperature              0
time                     0
coupon                   0
expiration               0
gender                   0
age                      0
maritalStatus            0
has_children             0
education                0
occupation               0
income                   0
Bar                    107
CoffeeHouse            217
CarryAway              150
RestaurantLessThan20   129
Restaurant20To50       189
toCoupon_GEQ15min        0
toCoupon_GEQ25min        0
direction_same           0
target                   0
dtype: int64
Total number of NaN:   792
```

Fig. 43: After Imputation Count



```
destination             0
passanger               0
weather                 0
temperature             0
time                    0
coupon                  0
expiration              0
gender                  0
age                     0
maritalStatus           0
has_children            0
education               0
occupation              0
income                  0
Bar                     0
CoffeeHouse             0
CarryAway               0
RestaurantLessThan20    0
Restaurant20To50        0
toCoupon_GEQ15min       0
toCoupon_GEQ25min       0
direction_same          0
target                  0
dtype: int64
Total number of NaN:  0
```

Thirdly, **missing values** can be influential factors which creates bias, data quality issues and inaccurate analysis. Notably, there are a total of 13,370 missing data values which can be seen in figure 41. For the "car" feature, surprisingly there is approximately 12,576 rows with missing values which is significantly higher than the other features. Comparatively, "Bar" has 107 rows of missing data, "CoffeeHouse" has 217 missing values, "CarryAway" has 151 missing values, "RestaurantLessThan20" has 130 missing values and "Restaurant20to50" has 189 missing values. After removing duplicates, in figure 42, we can see that this number has significantly decreased, and after using imputation in figure 43, we can see that the total number of missing values is now 0. The approach of using imputation helps to handle missing data by imputing it with the "most frequent value" (Makwana, 2021). Additionally, there appears to be **no significant outliers** within the dataset.

Before any modelling, it is required for Machine Learning models to have numerical input and output variables. Since the dataset contains categorical variables, we need to conduct transformations to manoeuvre around this problem. Feature transformations are performed to convert categorical into numerical type. Specifically, we will be using **One-hot encoding and ordinal encoding**. One-hot encoding represents each category as a binary column and suits our data since most of the categories do not have an ordering or any relationship with each other. Categories like "income", "Bar", "CoffeeHouse", "CarryAway", "RestaurantLessThan20", and "Restaurant20to50" however, have ordinality as they are values that are discretised. Since the data for those variables are naturally categorical with discretisation, ordinal encoding will be used to transform these values into numerical form. Through One-hot and ordinal encoding, we enable ourselves to fit and evaluate a model.

We then **split our data into training and testing sets**, where we will use a **70:30** split meaning that 70% of the "knowledge" will belong to the training set and 30% to the testing set. Building the model requires the training set, while the testing set is for validation purposes. To ensure reproducible output across the analysis, scikit-learn's 'train_test_split' will have an integer passed into the 'random_state' parameter.

3 Decision Tree Classifier

We decided to tune the “max_depth” and “max_leaf_nodes” parameters to reduce the size of the tree. The best accuracy scores for both parameters were 0.6803 at max depth of 7, and 0.6872 where max leaf nodes is 24.

3.1 Decision Tree Model Building

Fig. 44: Max depth accuracy scores

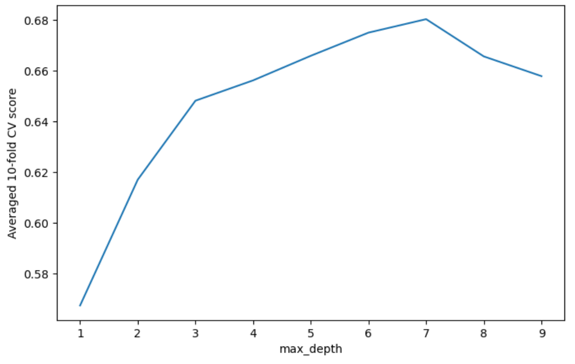
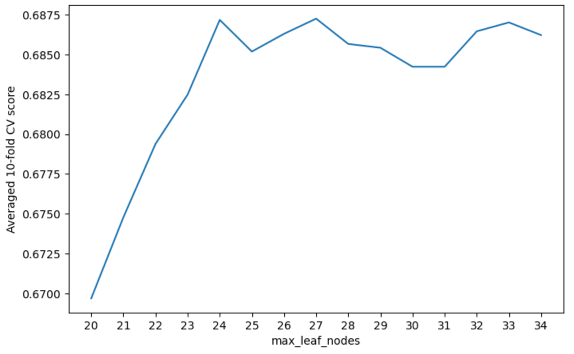


Fig. 45: Max leaf nodes accuracy scores



This provides us with the optimised classification tree below:

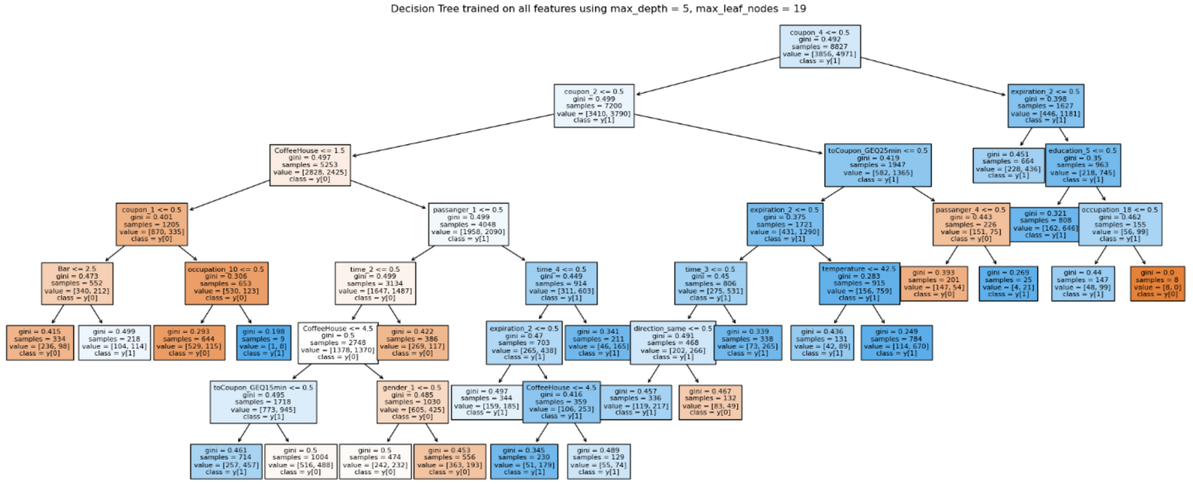

Decision Tree trained on all features using max_depth = 5, max_leaf_nodes = 19
coupon_4 <= 0.5 gini = 0.492 samples = 8827 value = [3856, 4971] class = y[1]
coupon_2 <= 0.5 gini = 0.499 samples = 7200 value = [3410, 3790] class = y[1]
expiration_2 <= 0.5 gini = 0.398 samples = 1627 value = [446, 1181] class = y[1]
CoffeeHouse <= 1.5 gini = 0.497 samples = 5253 value = [2828, 2425] class = y[0]
toCoupon_GEQ25min <= 0.5 gini = 0.419 samples = 1947 value = [582, 1365] class = y[1]
gini = 0.451 samples = 664 value = [228, 436] class = y[1]
education_5 <= 0.5 gini = 0.35 samples = 963 value = [218, 745] class = y[1]
coupon_1 <= 0.5 gini = 0.401 samples = 1205 value = [870, 335] class = y[0]
passanger_1 <= 0.5 gini = 0.499 samples = 4048 value = [1958, 2090] class = y[1]
expiration_2 <= 0.5 gini = 0.375 samples = 1721 value = [431, 1290] class = y[1]
passanger_4 <= 0.5 gini = 0.443 samples = 226 value = [151, 75] class = y[0]
gini = 0.321 samples = 808 value = [162, 646] class = y[1]
occupation_18 <= 0.5 gini = 0.462 samples = 155 value = [56, 99] class = y[1]
Bar <= 2.5 gini = 0.473 samples = 552 value = [340, 212] class = y[0]
occupation_10 <= 0.5 gini = 0.306 samples = 653 value = [530, 123] class = y[0]
time_2 <= 0.5 gini = 0.499 samples = 3134 value = [1647, 1487] class = y[0]
time_4 <= 0.5 gini = 0.449 samples = 914 value = [311, 603] class = y[1]
time_3 <= 0.5 gini = 0.45 samples = 806 value = [275, 531] class = y[1]
temperature <= 42.5 gini = 0.283 samples = 915 value = [156, 759] class = y[1]
gini = 0.393 samples = 201 value = [147, 54] class = y[0]
gini = 0.269 samples = 25 value = [4, 21] class = y[1]
gini = 0.44 samples = 147 value = [48, 99] class = y[1]
gini = 0.0 samples = 8 value = [8, 0] class = y[0]
gini = 0.415 samples = 334 value = [236, 98] class = y[0]
gini = 0.499 samples = 218 value = [104, 114] class = y[1]
gini = 0.293 samples = 644 value = [529, 115] class = y[0]
gini = 0.198 samples = 9 value = [1, 8] class = y[1]
CoffeeHouse <= 4.5 gini = 0.5 samples = 2748 value = [1378, 1370] class = y[0]
gini = 0.422 samples = 386 value = [269, 117] class = y[0]
expiration_2 <= 0.5 gini = 0.47 samples = 703 value = [265, 438] class = y[1]
gini = 0.341 samples = 211 value = [46, 165] class = y[1]
direction_same <= 0.5 gini = 0.491 samples = 468 value = [202, 266] class = y[1]
gini = 0.339 samples = 338 value = [73, 265] class = y[1]
gini = 0.436 samples = 131 value = [42, 89] class = y[1]
gini = 0.249 samples = 784 value = [114, 670] class = y[1]
toCoupon_GEQ15min <= 0.5 gini = 0.495 samples = 1718 value = [773, 945] class = y[1]
gender_1 <= 0.5 gini = 0.485 samples = 1030 value = [605, 425] class = y[0]
gini = 0.497 samples = 344 value = [159, 185] class = y[1]
CoffeeHouse <= 4.5 gini = 0.416 samples = 359 value = [106, 253] class = y[1]
gini = 0.457 samples = 336 value = [119, 217] class = y[1]
gini = 0.467 samples = 132 value = [83, 49] class = y[0]
gini = 0.461 samples = 714 value = [257, 457] class = y[1]
gini = 0.5 samples = 1004 value = [516, 488] class = y[0]
gini = 0.5 samples = 474 value = [242, 232] class = y[0]
gini = 0.453 samples = 556 value = [363, 193] class = y[0]
gini = 0.345 samples = 230 value = [51, 179] class = y[1]
gini = 0.489 samples = 129 value = [55, 74] class = y[1]

Fig. 46: Final optimised classification tree

The classification tree starts off at the root node with an “if” question about the feature “coupon_4” (Keep in mind that One-hot encoding transforms the variables into binary columns, so there will be more features than before under the original features). This node then branches off at the split point to two decision nodes containing more “if” questions about the data’s features, depending on whether the previous criteria was met or not. The tree will keep traversing with this method, eventually reaching the leaf/terminal nodes where the classes are predicted. Looking at Fig. 46 above, the leaf nodes are situated at either less than or equal to level 7 due to the max depth parameter tuning. We observe that there are minimal pure nodes through the varying gini scores as most of them never reach 0. Although the impure nodes can split further, it’s not always a good thing to keep the tree expanding until all the nodes are pure since it can lead to overfitting. This will be discussed further in the next section.

3.2 Parameter Tuning

As aforementioned, the tuning of these parameters assist in reducing the size to improve the accuracy of our model. We managed to reduce the number of nodes from the initial baseline decision tree model with 4711 nodes down to 207 nodes after the “max_depth” tuning. Further reduction to 47 nodes was achieved by tuning the “max_leaf_nodes” parameter, giving us the optimised classification tree found in Fig. 46.

The “max_depth” parameter limits how many levels at most the classification tree can expand to before making a prediction, instead of letting all the nodes expand until all its leaves are pure. As previously stated, the classification tree has a max depth of 7. This tuning is evident in the optimised classification tree in Fig. 46, seeing that there are only seven levels before the expansion is halted.

Like the “max_depth” parameter, “max_leaf_nodes” restricts how many leaf nodes the tree can have at most. Our classification tree has 24 max leaf nodes which can be observed in Fig. 46 by counting how many leaf nodes exist.

Without these limitations, the model ends up learning the detail and noise in the training set which can negatively impact our model’s performance. We can see that in Fig.44, there is a decline in accuracy scores past the max depth of 7 and in Fig. 45, a downwards trend starts between max leaf nodes value of 27 and 31. We can say that past around values, there is a risk of overfitting.

Using the same values from tuning this model will most likely not improve the accuracy for other types of datasets. Firstly, the “max_depth” parameter does not always translate to the actual depth of the tree. For example, a tree that fully expands to six levels at most can still have the parameter set to a max depth of 7. Similarly, a tree that can only expand to less than 24 leaf nodes can have the max leaf nodes parameter set to 24. These parameter values may be unfavourable for other datasets as they may be too big or too small, leading to potential overfitting or underfitting, which causes the model to suffer in performance. We can see this through Fig. 44 and Fig. 45 as not every value for max leaf nodes or max depth provides good accuracy scores.

3.3 Confusion Matrix and Summary Report

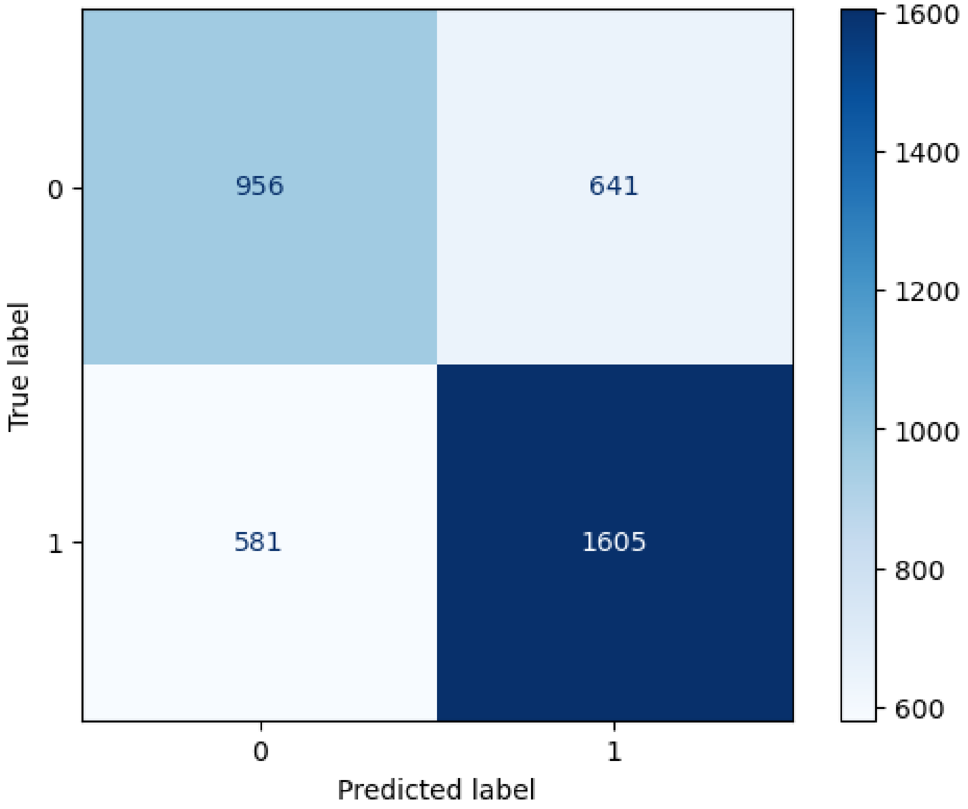
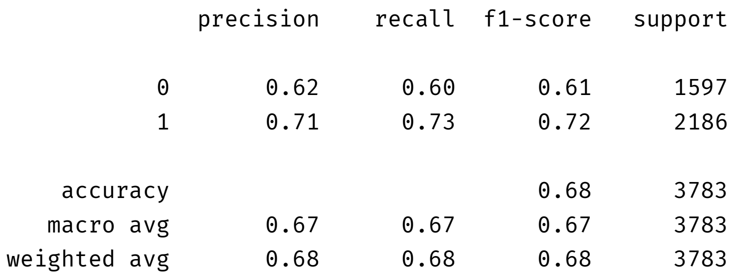


Fig. 47: Confusion Matrix for Decision Tree Model

From the confusion matrix in Fig. 47, we can see that there are 956 true positives, meaning that 956 of true class 0 was predicted as class 0. There are 1605 true negatives, where 1605 of true class 1 was predicted as class 1. 641 of true class 0 was predicted as class 1, which means there are 641 false negatives in the matrix. Lastly, 581 are false positives since 581 of true class 1 was predicted as class 0. We can further evaluate the performance of the current model through the summary report.



```
              precision    recall  f1-score   support

           0       0.62      0.60      0.61      1597
           1       0.71      0.73      0.72      2186

    accuracy                           0.68      3783
   macro avg       0.67      0.67      0.67      3783
weighted avg       0.68      0.68      0.68      3783
```

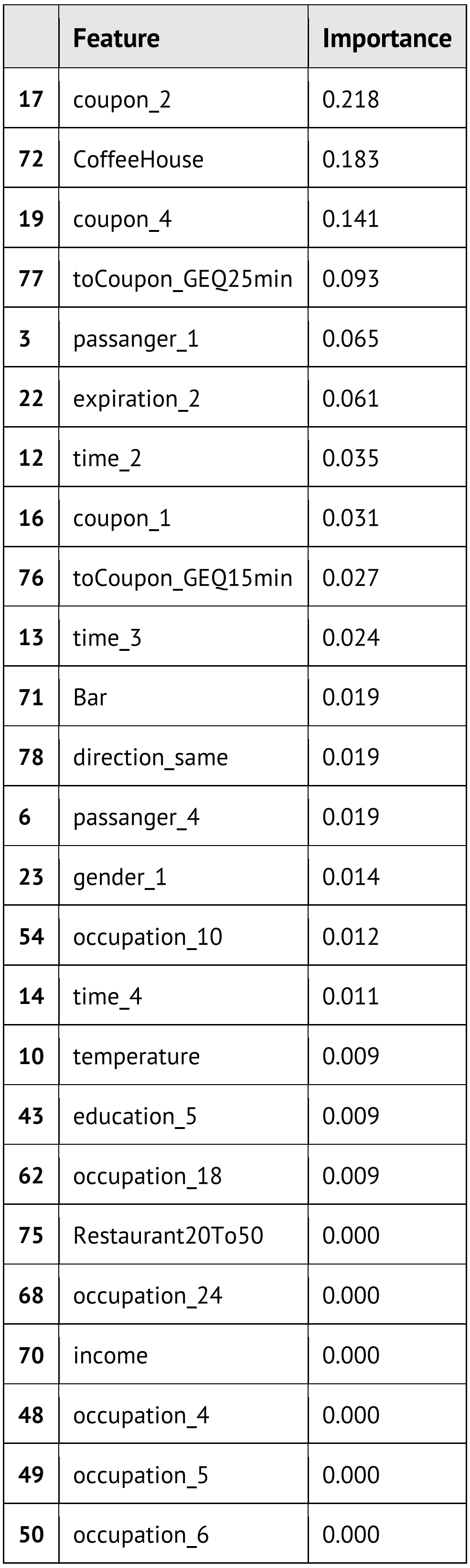
Table 2: Decision Tree Classification Report

From the model summary report in Table 2, the accuracy score is 0.68, meaning that the model has a 68% overall correct prediction.

For class 0, a precision score of 0.62 means that out of all that was predicted as class 0, only 62% were true class 0. Where recall is 0.60 means that out of all data points that should be predicted as class 0, 60% of them were predicted as class 0. Finally, the f1-score of 0.61 reflects the precision and recall abilities of the model, meaning that the current model has poor performance when it comes to classifying class 0.

For class 1, a 0.71 precision score means that out of all that was predicted as class 1, 71% were true class 1. A recall score of 0.73 means that out of all samples of true class 1, a good 73% of them were predicted as class 1. Lastly, an f1-score of 0.72 implies that the current model has good performance when trying to classify class 1.

3.4 Feature Importance



	Feature	Importance
17	coupon_2	0.218
72	CoffeeHouse	0.183
19	coupon_4	0.141
77	toCoupon_GEQ25min	0.093
3	passanger_1	0.065
22	expiration_2	0.061
12	time_2	0.035
16	coupon_1	0.031
76	toCoupon_GEQ15min	0.027
13	time_3	0.024
71	Bar	0.019
78	direction_same	0.019
6	passanger_4	0.019
23	gender_1	0.014
54	occupation_10	0.012
14	time_4	0.011
10	temperature	0.009
43	education_5	0.009
62	occupation_18	0.009
75	Restaurant20To50	0.000
68	occupation_24	0.000
70	income	0.000
48	occupation_4	0.000
49	occupation_5	0.000
50	occupation_6	0.000

Table 3: Feature importance table

The feature importance table allows us to identify which variables have the most impact regarding the model's performance and predictions. In order of importance, we can see that the "coupon_2" variable has a relatively high feature importance value (0.218) indicating that it is the most important feature in predicting the target variable and carries the most weight – the higher the feature importance value, the more significant the feature is for its predictions of the target variable. This is followed by "CoffeeHouse" and "coupon_4" with feature importance values of 0.183 and 0.141 respectively, which is relatively moderate in the level of importance. While it may not be as influential as the variable "coupon_4", it can still be considered to make meaningful contributions to the model's predictions and performance. When we analyse further down the scale, the lowest value, excluding 0, is 0.009. Although it is considerably lower than the other variables mentioned above and considered less significant in capturing any relationships or patterns, it would just have a "lesser role" during the analysis or evaluation process of the dataset. Additionally, there are multiple variables with a feature importance value of 0 which means that those variables have no evident importance or impact in the predictive models thus there would be no significant contribution towards to model's performance and predictability. These could signify that these variables are "irrelevant" or would have "no discriminatory power" in relation to the target variable. Henceforth, these variables could be disregarded when analysing the relationships of this dataset.

4 Artificial Neural Network (ANN)

In this section, we will explore various architectures for building an Artificial Neural Network (ANN) and use the 10-fold cross validation option for testing.

4.1 Feature Selection – Chi-Squared Test

Below, we have decided to use the filter method of a 'Chi-Squared Test'. The reason for using a chi-squared test is because from this dataset, we focus on dealing with categorical target variables. This method "measures the degree of association between to categorical variables" (Goswami, 2020). Additionally, from this, we can identify the **top five most significant features** which are stated on the next page.

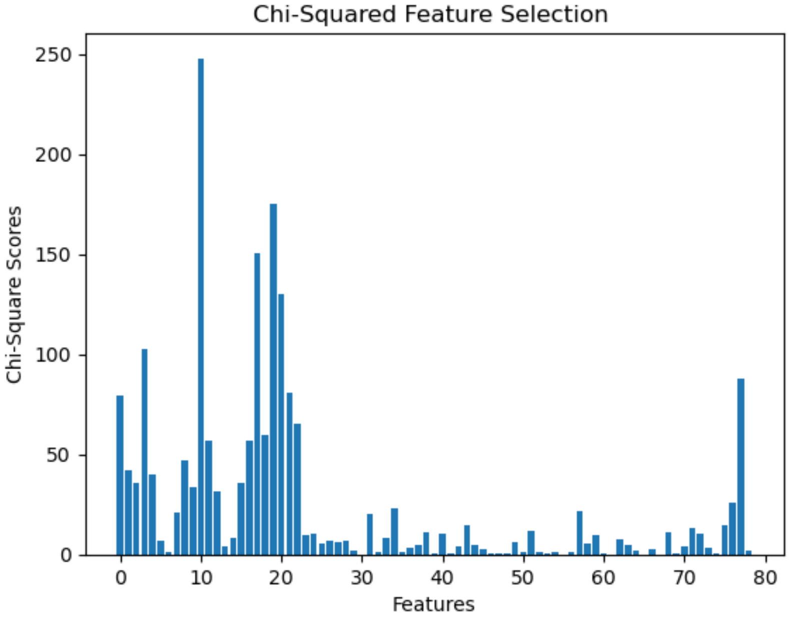
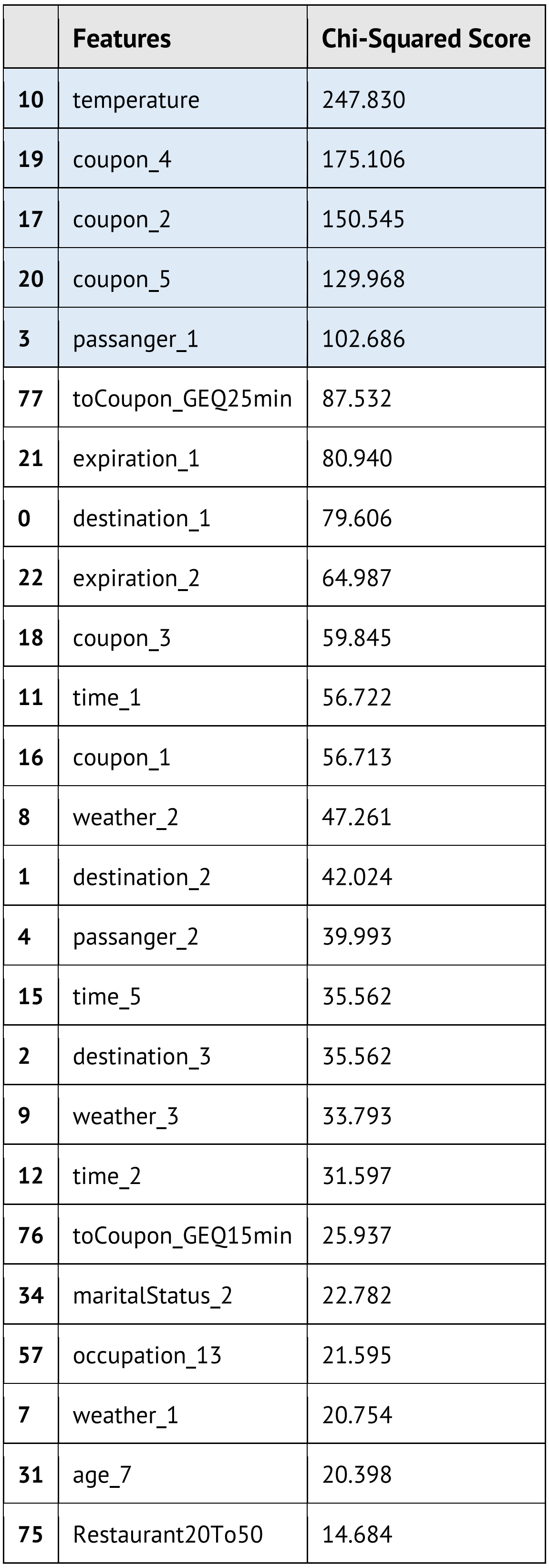


Fig. 48: Chi-Squared Feature Selection Graph

The list of features produced are as follows:

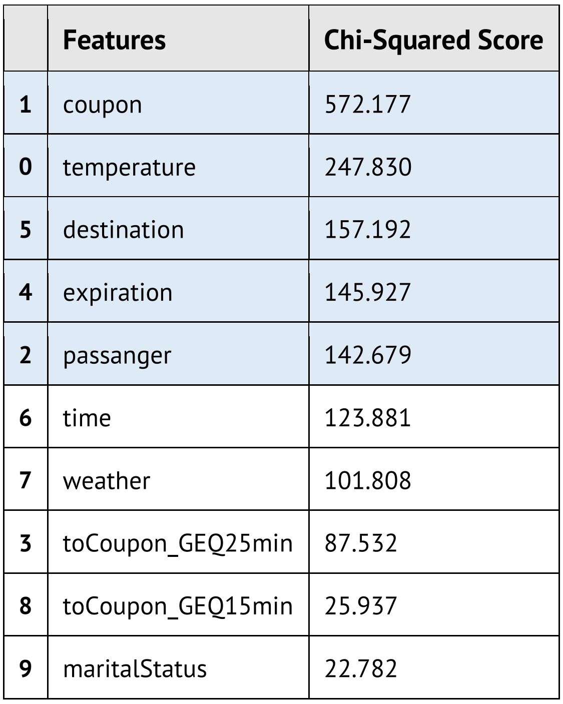


	Features	Chi-Squared Score
10	temperature	247.830
19	coupon_4	175.106
17	coupon_2	150.545
20	coupon_5	129.968
3	passanger_1	102.686
77	toCoupon_GEQ25min	87.532
21	expiration_1	80.940
0	destination_1	79.606
22	expiration_2	64.987
18	coupon_3	59.845
11	time_1	56.722
16	coupon_1	56.713
8	weather_2	47.261
1	destination_2	42.024
4	passanger_2	39.993
15	time_5	35.562
2	destination_3	35.562
9	weather_3	33.793
12	time_2	31.597
76	toCoupon_GEQ15min	25.937
34	maritalStatus_2	22.782
57	occupation_13	21.595
7	weather_1	20.754
31	age_7	20.398
75	Restaurant20To50	14.684

Table 4: Chi-Squared Test scores table

From Table 4, evidently, the five most significant figures are “temperature” with a chi-square score of 247.83, “coupon_4” with 175.106, “coupon_2” with 150.545, “coupon_5” with 129.968, and lastly, “passanger_1” with 102.686. Note that a higher chi-square score would indicate stronger discrepancy between the expected and observed frequencies thus suggesting strong association, of which that the relationship between the variables would have a greater impact.

We could also add the scores from features with the same name to see which of the features are significant prior to the One-hot encoding separation. Strictly from the 25 features listed above, the sum of scores with features of the same name are as follows:



	Features	Chi-Squared Score
1	coupon	572.177
0	temperature	247.830
5	destination	157.192
4	expiration	145.927
2	passanger	142.679
6	time	123.881
7	weather	101.808
3	toCoupon_GEQ25min	87.532
8	toCoupon_GEQ15min	25.937
9	maritalStatus	22.782

Table 5: Sum of Chi-Squared Test scores with the same name

When comparing Table 3’s feature importance table from the Decision Tree model to Table 5, we observe that the “coupon” feature is prominent. More specifically, “coupon_2” and “coupon_4” are in the top five most significant features in both the feature importance and chi-squared scores tables. The feature, “passanger” is also a feature that is in the top five most significant in both tables. The feature “expiration” can be found in similar positions for both tables (6th in Table 3 and 4th in Table 5), giving it some level of significance.

Two distinct differences between the tables are the features: “destination” and “temperature”. The feature “destination” has a feature importance score of 0, but a Chi-Squared test score of 157.192 making it the 3rd most significant feature. The feature “temperature” also had a feature importance value of 0, but surprisingly had a Chi-Squared test score of 247.830 which makes it the 2nd most significant feature behind “coupon”. These two variables had low feature importance values, which could mean that both features were not chosen at an early level in the Decision Tree but proved to be significant features in the Chi-Squared.

4.2 MLP Classifier – Max Iteration Tuning

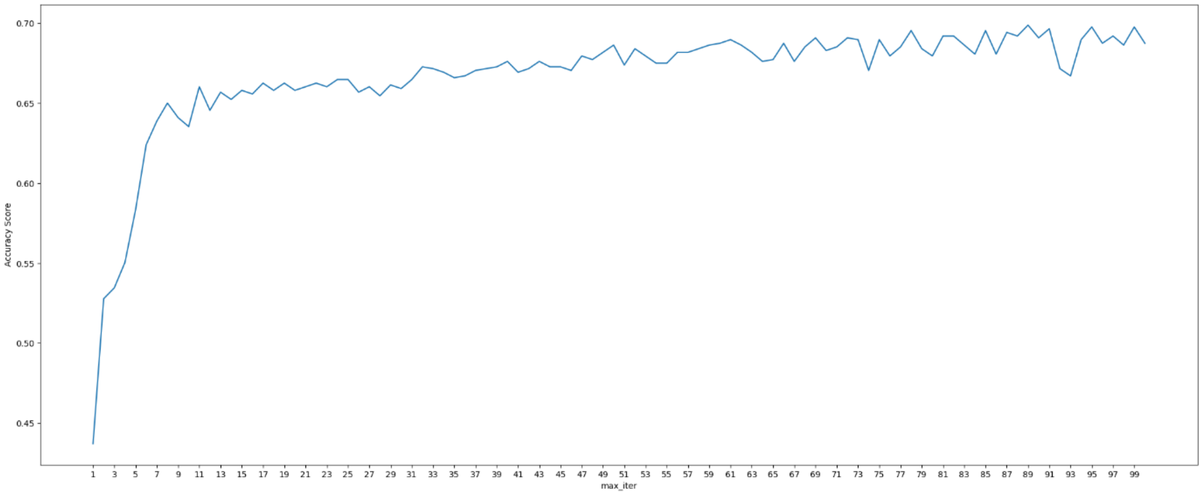
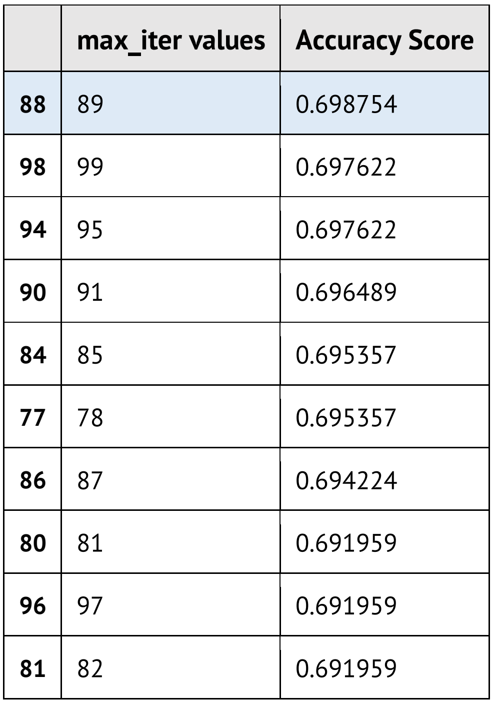


Fig. 49: Accuracy Scores for "max_iter" values between 1 and 99

Figure 49 displays the output after using the MLP Classifier with default parameters and with "hidden_layer_sizes" equal to 25 (as per guidelines given where k neurons <= 25). When analysing the graph, the accuracy score from a max_iter of 1 to 9 appears to have gradually increased before reaching a range which it becomes "stable". After reaching 9, the range of the accuracy score ranges from approximately 0.65 to 0.70. By setting the parameter "early_stopping" to True in the MLP, the training process would be stopped relatively early based on a "particular criterion". Essentially, it allows to 'stop the learning' when there appears to be no improvement in several iterations. Note that as we have set it to true, then it would automatically set aside 10% of training data as validation and then terminate training like mentioned. When also set to true, it helps to prevent "overfitting" which is not wanted in our analysis, i.e., where our model would become overly complex and would rather memorize the training data rather than generalizing effectively to unseen data. This method allows us to determine and report the best number of iterations which would give us the highest accuracy. Note that in our code, although "max_iter" is set to 200, it has stopped early at 99 as it did not seem to detect any significant improvements. Furthermore, employing this technique would allow us to halt the iterations without the need and requirement of human intervention thus providing an automated way to control the training process; it would continuously monitor the validation loss based on a particular criterion and then determine its stopping point which would essentially simplify the training workflow and thus enhance our decision-making processes.



	max_iter values	Accuracy Score
88	89	0.698754
98	99	0.697622
94	95	0.697622
90	91	0.696489
84	85	0.695357
77	78	0.695357
86	87	0.694224
80	81	0.691959
96	97	0.691959
81	82	0.691959

Table 6: Table for top 10 accuracy scores for "max_iter" parameter

The table above lists out the top 10 accuracy scores for the "max_iter" parameter based on figure 49. Note that the total iterations were from 1-99. The best number for iteration we analyse here is 89. This would represent the point at which the model would achieves its highest performance as well. The accuracy score it presented was at 0.698754. Looking at the other "max_iter" values below, it is also quite close and similar in terms of accuracy score, with around only a 0.001 to 0.009 difference. Thus, although the graph appears to have some constant peaks and troughs, analysing it from the table shows that the difference is marginally miniscule.

Note that if there are also "too little iterations", then the model would not have been able to identify and learn any of the underlying patterns or relationships in the data and too much would lead to overfitting.

4.3 Loss Value

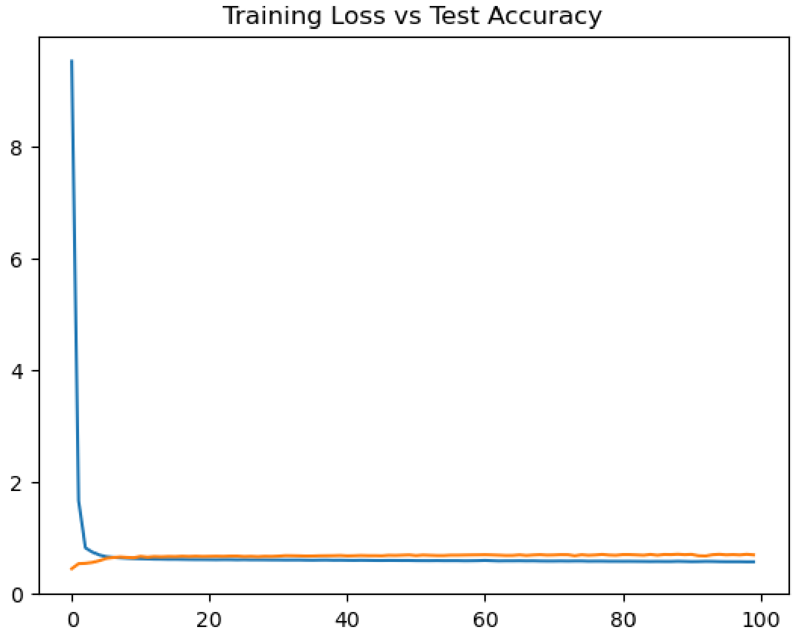
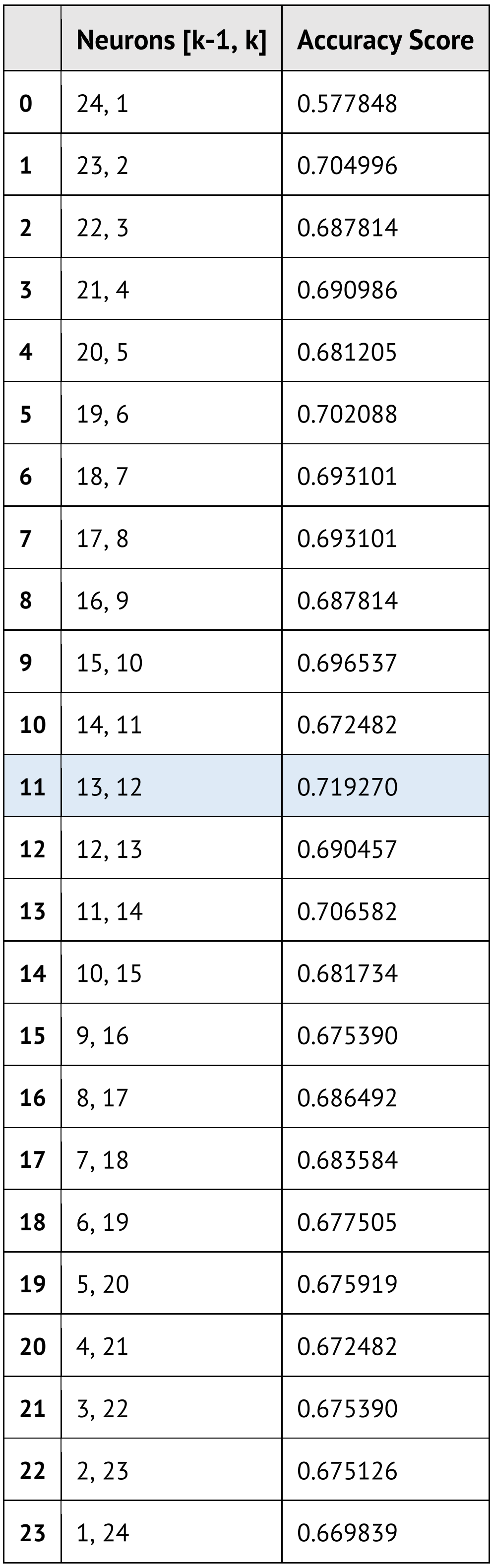


Fig. 50: Loss Values Graph

Figure 50 presents a Loss Value Graph in which is a graphical representation of the summation of errors in our model. It provides insights in how the learning performance changes over several epochs thus helping us to diagnose any problems, particularly with learning that could lead to underfitting or overfitting a model (Cristina, 2022). Thus, essentially the graph above would show the loss of values of the different numbers of iterations. Notably, we observe that from the peak at 0, afterwards, the graph appears to decrease and then “flatten” (there is barely any change after approximately iteration of 5). The training set is set to the blue coloured line and the validation data set is the orange line. Overall, it appears that the data is quite close together, however, it would have been ideal if the training and validation loss line would decrease together to a point of stability but instead it appears to incrementally increase and expand in gap size looking at the two final loss values (it is important to note although, we should expect a minimal gap between the two final loss values) (Brownlee, 2019). Furthermore, when looking at the two final loss values, the slow expansion in the gap between the two lines may suggest a case of overfitting. Also note that the loss of the model should always almost be slightly lower on the training dataset than the validation set. On the other hand, we can see that before approximately the iteration of 5, there is a huge difference in the beginning of the validation line (orange line) – this may be an indicator that at this point from the dataset, it would not provide sufficient information.

4.4 Two Hidden Layers



	Neurons [k-1, k]	Accuracy Score
0	24, 1	0.577848
1	23, 2	0.704996
2	22, 3	0.687814
3	21, 4	0.690986
4	20, 5	0.681205
5	19, 6	0.702088
6	18, 7	0.693101
7	17, 8	0.693101
8	16, 9	0.687814
9	15, 10	0.696537
10	14, 11	0.672482
11	13, 12	0.719270
12	12, 13	0.690457
13	11, 14	0.706582
14	10, 15	0.681734
15	9, 16	0.675390
16	8, 17	0.686492
17	7, 18	0.683584
18	6, 19	0.677505
19	5, 20	0.675919
20	4, 21	0.672482
21	3, 22	0.675390
22	2, 23	0.675126
23	1, 24	0.669839

Table 7: Two Hidden Layers: Neurons Classification Table

In figure 51 we experiment with **two hidden layers** which provides accuracy results in a 24x3 table in which the second column specifies the combination of neurons used and the third column specifying the classification accuracy. The neuron set with the highest accuracy score is [13, 12] with an accuracy score of 0.719270. Note that the table only presents 24 accuracy results as [25, 0] is not within the first iteration i.e., [25, 0] and [0, 25] would be just one layer of 25 neurons. In a single layer, there is k neurons and in a sequence of two hidden layers, we essentially transfer neurons from the first hidden layer to the second, iteratively with a step size of 1. Overall, the accuracy score range is not too far off from the 24 neuron combinations – the highest accuracy score of [13,12] only has a difference of 0.141422 to the neuron combination of [24,1] which has the lowest accuracy score of 0.577848.

4.5 Accuracy Variation

From figure 51, we observe that the accuracy scores do not remain stagnant, and that there is slight variation in accuracy. This could be explained by the proportion of neurons within the hidden layers of the neural network. Specifically, when using too little neurons in a hidden layer, we see the result of underfitting. This occurs since there are too few neurons to sufficiently detect signals in our complicated, non-linear dataset. For example, say that the first hidden layer only had one neuron. We observe a low accuracy score since only one neuron is responsible for adjusting the weighted links in the first hidden layer, leaving another 24 neurons in the second hidden layer. Underfitting also happens when the second hidden layer only has one neuron responsible for adjustment, as there is only one neuron leading to the output layer which is responsible for producing the final prediction. This is evident in our experimentation as the neuron sets of [24, 1] and [1, 24] had the lowest accuracy scores of 0.5778 and 0.6698 respectively.

5 Performance Comparison

Conclusively, after evaluating the accuracy scores of the two classification models, it becomes evident that the ANN classifier outperforms the other model in terms of accuracy.

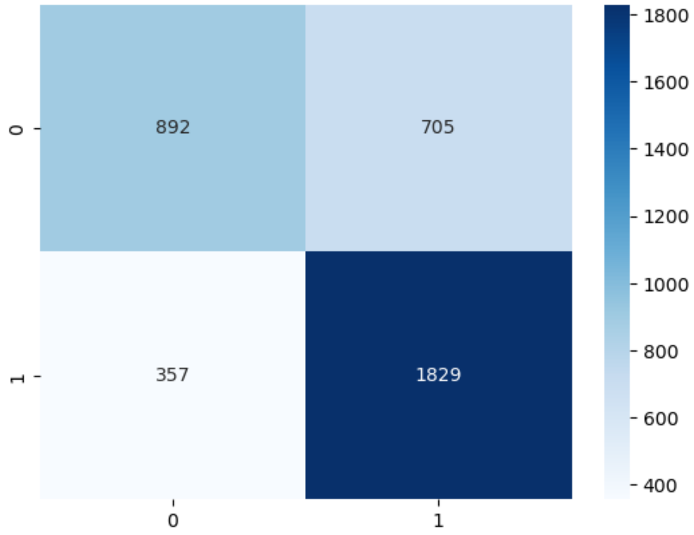


Fig. 51: Confusion Matrix for ANN Model

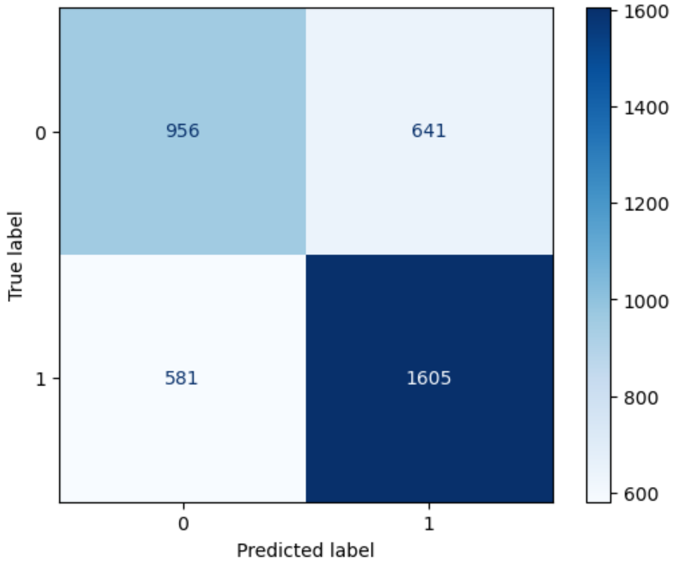


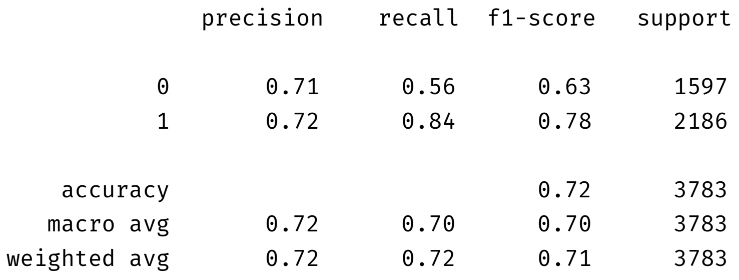
Fig. 47: Confusion Matrix for Decision Tree Model
(Referencing back to the confusion matrix in Section 3.2)

Note that the following models have been adjusted according to suit each model but will have the same true and false positives total and true and false negatives total.

For the confusion matrix of the ANN model, there are 892 true positives which means that 892 of the true class of 0 were predicted as class 0. Furthermore, there are 1829 true negatives in which 1829 of the true class 1 were predicted as class 1. At the top right, 705 of the true class of 0 was predicted as class 1 which means that there are approximately 705 false negatives within the confusion matrix for the ANN model. And, at the bottom left, we have 357 false positives of true class 1 predicted as class 0. Note that a higher score of true positives and true negatives would mean that the model is predicting the instances correctly out of an “x” amount.

When comparing the true positives, we can see that the decision tree performs better in predicting the true positives (956) as opposed to the ANN model with only 892 in class 0. However, we analyse that the ANN model does outperform in correctly predicting the true negatives in class 1 – for the decision tree model, it has a value of 1605, and the ANN model has a value of 1829 which is significantly higher. When comparing the true positives to the false positives and the true negatives to the false negatives, the decision tree classifier outperforms the ANN for the true positive – false positive predictability, however, the ANN outperforms the decision tree classifier in regards true negatives – false positives predictability. However, we could assume that because the difference in true positive – false positives are a smaller margin compared to the true negatives – false positives, then we can assume that ANN would be more suitable from the confusion matrix.

Next, we will perform an analysis on the classification reports:

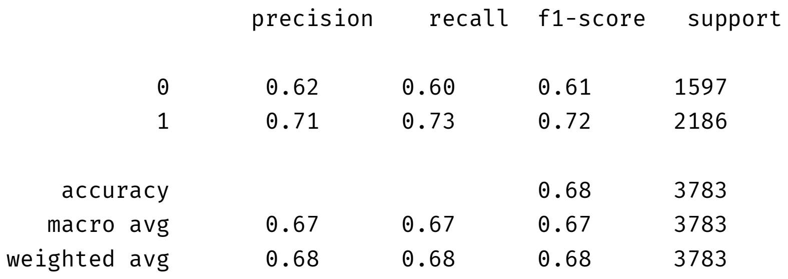


```
              precision    recall  f1-score   support

           0       0.71      0.56      0.63      1597
           1       0.72      0.84      0.78      2186

    accuracy                           0.72      3783
   macro avg       0.72      0.70      0.70      3783
weighted avg       0.72      0.72      0.71      3783
```

Table 8: ANN Model Classification Report



```
              precision    recall  f1-score   support

           0       0.62      0.60      0.61      1597
           1       0.71      0.73      0.72      2186

    accuracy                           0.68      3783
   macro avg       0.67      0.67      0.67      3783
weighted avg       0.68      0.68      0.68      3783
```

Table 2: Decision Tree Classification Report

(Referencing back to the decision tree classification report in Section 3.2)

The ANN model had an accuracy score of 0.72 (Table 8), which is marginally better overall correct prediction compared to the Decision Tree model's 0.68 accuracy score (Table 2). This means that for the ANN model, the model has a 72% overall correct prediction whereas for the Decision Tree model, it has an 68% overall correct prediction.

We can, however, observe that the ANN model has a poor recall performance when it comes to class 0, only having a recall score of 0.56 compared to the Decision Tree model's 0.60. This meant that ANN performed poorly when it came to classifying data points that should be predicted as class 0. Regardless, the f1-score for the ANN model is slightly higher than the Decision Tree's model – 0.63 as opposed to 0.61. Note that the f1-score reflects the precision and recall abilities of the model, therefore we can conclude that both models have relatively similar performance when predicting class 0.

For class 1, we observe an increase in precision, recall, and f1-score when predicting with the ANN Model. The ANN model's recall and f1-score was 0.84 and 0.78 respectively. On the other hand, the values of the recall and f1-score of the decision tree classifier was at 0.73 and 0.72 respectively. The difference between the values of the ANN model to the Decision tree classifier regarding class 1 is evidently larger than those of class 0. For the recall score there was a difference of 0.11 and for the f1-score, there was a difference of 0.06.

Henceforth, we conclude that the ANN model would perform better as opposed to the decision tree classifier.

6 Conclusion

In reference to our research question:

> ***"What is the predictive performance comparison between the decision tree classifier and the neural network in estimating the in-vehicle coupon acceptance rate and what features are significant in determining this rate?"***

Conclusively, from the analysis of the confusion matrixes and tables above, overall, the ANN model would perform better as opposed to the decision tree classifier.

Significant features we found which could help in determining the user acceptance rate included:

- Coupon (specifically coupon_4, coupon_2, coupon_5, which are CoffeeHouse, Bar, and Restaurant $20-$50)
- CoffeeHouse (how many times they go to a coffeehouse per month)
- If they are alone (passangar_1)
- If the Coupon expires in 1 day/urgently (expiration_1)
- Temperature
- If they have no urgent place to be at (destination_1)

From this, we would advise businesses to revise their business and marketing strategies in conjunction with their discounted coupons. We can see how this tie in with the overall market and audience in which the coupons were given to. Additionally, we can also see how these features work together which would help indicate the user acceptance rate such as "expiration" and "destination" – a business recommendation for this particular combination of feature would be because not all users would not have a non-urgent place to be at, it would be ideal to extend the expiration which would account for the variability in each user's time and schedule. This could be seen in reference to figure 5 where the coupons which have a longer validity are more widely and likely to be accepted by customers. Another feature we can focus on is "CoffeeHouse" – in figure 3, we notice that the distribution of the coupon was high for this particular business area, and although there are quite a lot of customers who go to the coffeehouse 1-3 times a month, we could focus on prioritizing to give them a "CoffeeHouse" coupon, and those who have never gone (fig. 7), they could be given other coupons. Although discount coupons do entice and encourage new customers, another form of looking at is keeping regular customers satisfied as not everyone may want to go to a Coffee house.

Appendix

Code to be submitted separately.

References

Brownlee, J. (2019). How to use Learning Curves to Diagnose Machine Learning Model Performance. *MachineLearningMastery.com*. https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/

Cristina, S. (2023). Plotting the Training and Validation Loss Curves for the Transformer Model. *MachineLearningMastery.com*. https://machinelearningmastery.com/plotting-the-training-and-validation-loss-curves-for-the-transformer-model/

Epstein, L. (2022). Advantages and Disadvantages of Using Coupons for Your Business. *Investopedia*. https://www.investopedia.com/articles/personal-finance/051815/pros-cons-using-coupons-your-business.asp

Goswami, S. (2021). Using the Chi-Squared test for feature selection with implementation. *Medium*. https://towardsdatascience.com/using-the-chi-squared-test-for-feature-selection-with-implementation-b15a4dad93f1

Makwana, K. (2021). Frequent Category Imputation (Missing Data Imputation Technique). *Medium*. https://medium.com/geekculture/frequent-category-imputation-missing-data-imputation-technique-4d7e2b33daf7

Wang, T., Rudin, C., Doshi-Velez, F., Liu, Y., Klampfl, E., & MacNeille, P. (2017). A Bayesian framework for learning rule sets for interpretable classification. *Journal of Machine Learning Research 18, no. 1*. https://doi.org/10.5555/3122009.3176814