Starbucks Capstone Project Report

Udacity Machine Learning Engineer Nanodegree

Lu Liu

August 12, 2020

1 Project Overview

The Starbucks Capstone project is part of the Udacity Machine Learning Engineer Nanodegree. Udacity partnered with Starbucks to provide a real-world business problem and simulated data mimicking their customer behaviour.

Starbucks Corporation is an American coffeehouses company and it is one of the world's largest coffeehouse chain. Once every few days, Starbucks sends out an offer to users of the Starbucks rewards mobile app. The offers contain three types: an advertisement for a drink, an actual offer such as a discount and BOGO (buy one get one free). Some users might not receive any offer during certain weeks. Starbucks wants to utilise technology to provide their customers a better personalised experience and boost their marketing performance. To be specific, they want to target the right customers who are most likely to make a purchase and also know about customers who might not want an offer.

The motivations of this project is to gain more machine learning hands-on practice and have a better understanding about how the technology can make an impact on the real business.

2 Problem Statement

Based on the description above, the problem that needs to be solved in this project is to predict whether an offer will be completed or not based on the given demographics and historical data. This problem can be treated as a supervised learning problem. Specifically, this is a binary classification problem where '1' represents that the offer is successful and '0' represents that the offer is not successful. The metrics used to measure the performance of the model are confusion matrix along with accuracy, precision, recall and F2 score, detailed information will be included in an upcoming section.

3 Datasets and Inputs

The following overview of the datasets are provided by Udacity:

- The program used to create the data simulates how people make purchasing decisions and how those decisions are influenced by promotional offers.

- Each person in the simulation has some hidden traits that influence their purchasing patterns and are associated with their observable traits. People produce various events, including receiving offers, opening offers, and making purchases.
- As a simplification, there are no explicit products to track. Only the amounts of each transaction or offer are recorded.
- There are three types of offers that can be sent: buy-one-get-one (BOGO), discount, and informational. In a BOGO offer, a user needs to spend a certain amount to get a reward equal to that threshold amount. In a discount, a user gains a reward equal to a fraction of the amount spent. In an informational offer, there is no reward, but neither is there a requisite amount that the user is expected to spend. Offers can be delivered via multiple channels.
- The basic task is to use the data to identify which groups of people are most responsive to each type of offer, and how best to present each type of offer.

The provided datasets contain three datasets:
profile.json: Rewards program users (17000 users x 5 fields)

- gender: (categorical) M, F, O, or null
- age: (numeric) missing value encoded as 118
- id: (string/hash)
- became_member_on: (date) format YYYYMMDD
- income: (numeric)

portfolio.json: Offers sent during 30-day test period (10 offers x 6 fields)

- reward: (numeric) money awarded for the amount spent
- channels: (list) web, email, mobile, social
- difficulty: (numeric) money required to be spent to receive reward
- duration: (numeric) time for offer to be open, in days
- offer_type: (string) bogo, discount, informational
- id: (string/hash)

transcript.json: Event log (306648 events x 4 fields)

- person: (string/hash)
- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
- offer id: (string/hash) not associated with any ”transaction”

- amount: (numeric) money spent in "transaction"
- reward: (numeric) money gained from "offer completed"
- time: (numeric) hours after start of test

4 Data Pre-processing

This section includes the process for data cleaning, data transformation (One-Hot encoding).

Figure 1, 2 and 3 show the cleaned profile dataset, portfolio dataset and transcript dataset, respectively.

Profile dataframe

- Remove missing or invalid values for columns 'age', 'income' and 'gender', for a customer, if any of these attributes contains missing data, the other two are missing as well.
- Change 'id' column name to 'customer_id'.
- Change 'became_member_on' column to 'membership_days' which contains how many days since they first signed up for the app and became a member.
- Perform one-hot encoding to column 'gender' and convert it into 3 columns – 'female', 'male' and 'other_gender'.
- Perform one-hot encoding to column 'age' and convert it into 8 age groups – 'age_under25, 'age_25_to_35', 'age_35_to_45', 'age_45_to_55', 'age_55_to_65', 'age_65_to_75', 'age_75_to_85' and 'age_85_to_105'.

customer_id	income	membership_days	female	male	other_gender	age_under25	age_25_to_35	age_35_to_45	age_45_to_55	age_55_to_65	age_65_to_75	age_75_to_85	age_85_to_105
0610b486422d4921ae7d2bf64640c50b	112000.0	1125	1	0	0	0	0	0	0	1	0	0	0
78afa995795e4d85b5d9ceeca43f5fef	100000.0	1192	1	0	0	0	0	0	0	0	1	0	0
e2127556f4f64592b11af22de27a7932	70000.0	840	0	1	0	0	0	0	0	0	1	0	0
389bc3fa690240e798340f5a15918d5c	53000.0	916	0	1	0	0	0	0	0	0	1	0	0
2eeac8d8feae4a8cad5a6af0499a211d	51000.0	1006	0	1	0	0	0	0	0	1	0	0	0

Figure 1: The first five rows in cleaned profile dataset.

Portfolio dataframe

- Change 'id' column name to 'offer_id'.
- Perform one-hot encoding to column 'offer_type' and convert it into 3 columns – 'informational', 'discount' and 'bogo'.
- Perform one-hot encoding to column 'channels' and convert it into 4 columns – 'email', 'mobile', 'social' and 'web'.
- Set column 'offer_id' to the first column.
- Remove 'email' column since the value for this column is always '1'.

	offer_id	reward	difficulty	duration	informational	discount	bogo	mobile	social	web
0	ae264e3637204a6fb9bb56bc8210ddfd	10	10	7	0	0	1	1	1	0
1	4d5c57ea9a6940dd891ad53e9dbe8da0	10	10	5	0	0	1	1	1	1
2	3f207df678b143eea3cee63160fa8bed	0	0	4	1	0	0	1	0	1
3	9b98b8c7a33c4b65b9aebfe6a799e6d9	5	5	7	0	0	1	1	0	1
4	0b1e1539f2cc45b7b9fa7c272da2e1d7	5	20	10	0	1	0	0	0	1
5	2298d6c36e964ae4a3e7e9706d1fb8c2	3	7	7	0	1	0	1	1	1
6	fafdcd668e3743c1bb461111dcafc2a4	2	10	10	0	1	0	1	1	1
7	5a8bc65990b245e5a138643cd4eb9837	0	0	3	1	0	0	1	1	0
8	f19421c1d4aa40978ebb69ca19b0e20d	5	5	5	0	0	1	1	1	1
9	2906b810c7d4411798c6938adc9daaa5	2	10	7	0	1	0	1	0	1

Figure 2: Cleaned portfolio dataset.

Transcript dataframe

- Change 'person' column name to 'customer_id'.
- Change 'time' column from hours to days.
- Create a new column 'offer_id' and extract the value of 'offer_id' from column 'value'.
- Create a new column 'amount' and extract the value of 'amount' from column 'value'.
- Drop the original 'value' column and only keep customer_ids which are in the profile dataframe.
- Drop duplicated rows and set 'offer_id' column to the second column.
- Create a new dataframe called 'offers_df' where contain events except 'transaction'.
- Create a new dataframe called 'transaction_df' where only contains 'transaction' event.

customer_id	event	time	offer_id	amount
24f56b5e1849462093931b164eb803b5	offer completed	29.75	fafdcd668e3743c1bb461111dcafc2a4	NaN
b3a1272bc9904337b331bf348c3e8c17	transaction	29.75	None	1.59
68213b08d99a4ae1b0dcb72aebd9aa35	transaction	29.75	None	9.53
a00058cf10334a308c68e7631c529907	transaction	29.75	None	3.61
76ddbd6576844afe811f1a3c0fbb5bec	transaction	29.75	None	3.53

Figure 3: The final five rows of cleaned transcript dataset.

5 Data Exploration

Profile dataframe

Figure 4, 5, 6 and 7 show us the distributions of the customers' income, age, gender and the distribution of each year that customers became Starbucks members.

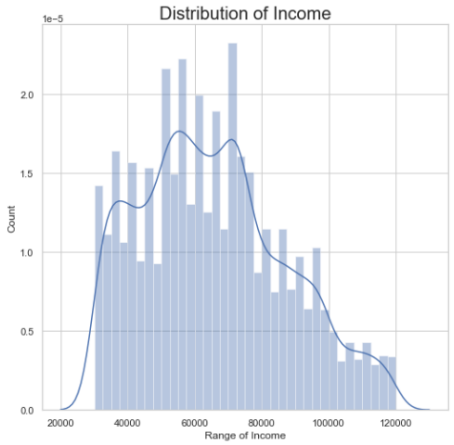

Figure 4: Distribution of income in profile dataset.

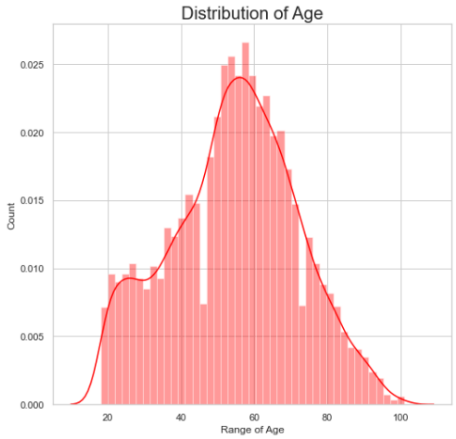

Figure 5: Distribution of age in profile dataset.

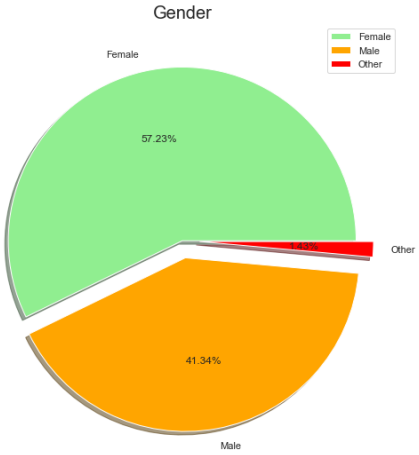

Figure 6: Distribution of gender in profile dataset.

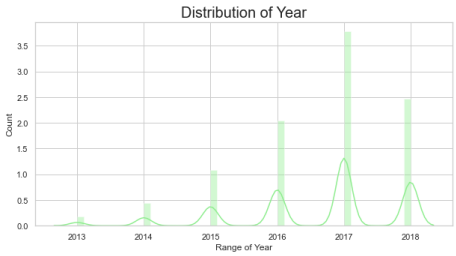

Figure 7: Distribution of year in profile dataset.

- Figure 4 shows that most of the customers' income range from around 50000 to 70000 dollars.
- Figure 5 shows that the youngest customer is 18 years old and the oldest age is 101. Additionally, most of the customers' age range from around 55 to 68. We can see from the given dataset that most customers are from the middle class in America.

- Figure 6 shows there is no huge difference between female and male customers, more than half customers are females.
- From figure 7, we can see there was an increase from 2013 to 2017, whereas there was a drop in 2017, which means the year of 2017 is the year that has the most new customers registered on Starbucks app and became a member between 2013 and 2018.

Portfolio dataframe

- Informational type of offer will never be completed as the difficulty is '0'.
- The values for column 'email' are always '1', which means every promotional offer is sent via email, therefore this column has been removed in the cleaned portfolio dataset.

Transcript dataframe

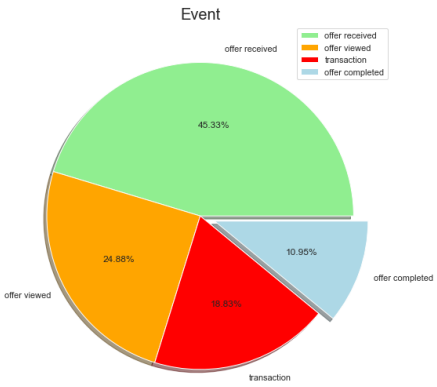

Figure 8: Distribution of events in transcript dataset.

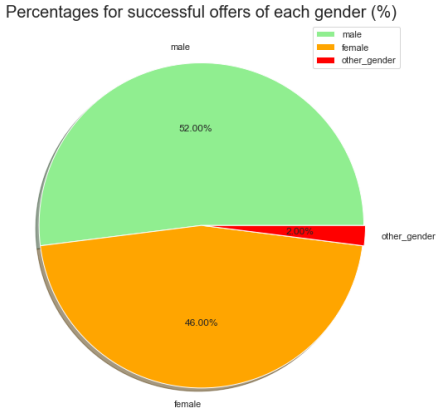

Figure 9: Percentage for successful offers of each gender.

- Figure 8 shows that 10.95% of events are 'offer completed' and 18.83% are 'transactions'.
- A dataframe named 'transaction_df' where 'event' only contain 'transaction' is extracted from transcript dataset, and I also created a new dataframe called 'offers_df' where 'event' contains other three categories.
- From the 'offer_df' dataframe, we know that each customer received an offer more than once, and for the same timestamp a certain customer will only receive any offer once, which means we can merge offer_id and customer_id based on a certain timestamp.
- By identifying expired offers, I merged data extracted from transcript and portfolio, this lead to the creation of the new columns: total_amount and offer_successful, where total_amount is the total amount spent by a customer

from given offers, and offer_successful is marked as '1' if an offer is viewed and completed within the start and end time.

- One interesting finding that I found from figure 9 is that more males have completed the offers than females,whereas in the original datasets, more customers are females as mentioned above. However the difference is fairly small.

6 Data transformation

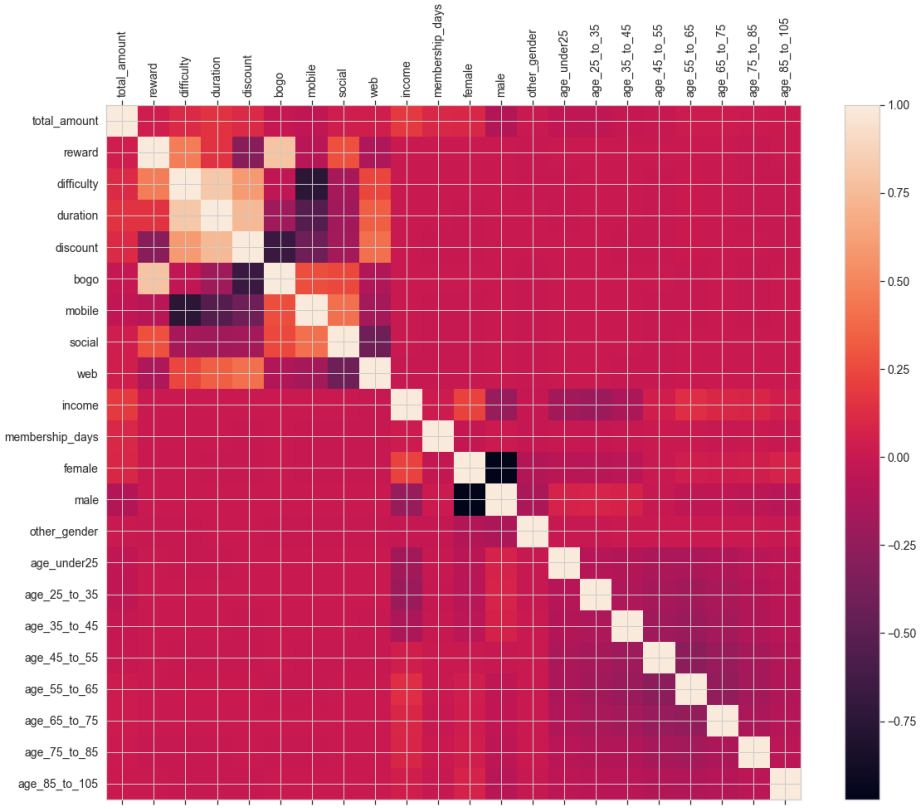

Figure 10: Correlation between each feature.

Before the data can be fed into a model, several steps are still needed to implement.See the details below:

- Drop column 'customer_id' since we don't focus on a specific customer, drop column 'time' since we only used this column to extract the identify the expired offers, and also drop column 'informational' as the difficulty for this type of offer is zero and it will never be completed.

- Remove any duplicated rows and reset index.
- Perform data normalisation on column 'total_amount', 'reward', 'difficulty', 'duration', 'income' and 'membership_days' to avoid features with a large range so that the result will not be effected by the imbalance, the technique adopted is MinMaxScaler.
- The final dataset contains the following 23 columns: 'total_amount', 'reward', 'difficulty', 'duration', 'discount', 'bogo', 'mobile', 'social', 'web', 'income', 'membership_days', 'female', 'male', 'other_gender', 'age_under25', 'age_25_to_35', 'age_35_to_45', 'age_45_to_55', 'age_55_to_65', 'age_65_to_75', 'age_75_to_85', 'age_85_to_105' and 'offer_successful'.
- Visualise the correlation between each two features in Figure 10, we can see that 'mobile' and 'difficulty', 'mobile' and 'duration' have relatively higher negative correlation, 'duration' and 'difficulty' has relatively higher positive correlation.

7 Model training

7.1 Split data

Before the data are fed into the model, it needs to be shuffled and randomly split into training and testing sets with a proportion of 7:3.

7.2 Evaluation metrics

Confusion matrix along with accuracy, precision, recall and F2 score will be considered in the evaluation of the model performance.

The table below is the illustration of Confusion matrix:

		Prediction		
		0	1	Total
Actual	0	$d - TrueNegative$	$c - FalsePositive$	$c + d$
	1	$b - FalseNegative$	$a - TruePositive$	$a + b$
	Total	$b + d$	$a + c$	N

a - True Positive: Send an offer & the customer will be likely to use the offer.

b - False Positive: Send an offer & the customer will not be likely to use the offer or does not want it.

c - False Negative: Do not send an offer & the customer will be likely to use the offer.

d - True Negative: Do not send an offer & the customer will not be likely to use the offer if we sent it.

In this project, we do not want to optimise accuracy only, because in this business case, we want to avoid the situation of sending offers to customers who are not

likely to use them or want them as much as possible (c - False Positive). Precision is often used when the cost of False Positive is high, and we also want to take Recall into consideration. Therefore, we chose F2 score as the final evaluation metrics. The formula for F2 score can be described as below:

$$F_2 = \frac{(1+2^2) * Precision * Recall}{(2^2 * Precision) + Recall} \tag{1}$$

7.3 Model training

7.3.1 Logistic Regression

Logistic regression is a technique for performing binary classification, it is for modelling the probability that an output belongs to one of two possible classes.The formula can be described as below [1]:

$$\phi(x) = \frac{1}{1 + \exp(-x)} \tag{2}$$

It takes in a real number and maps it to a position on the interval from 0 to 1. This is useful when modelling the probability of an output to belong to a particular class, since that probability must always be between zero and one. Figure 11 illustrates that When x is a very large negative number, $exp(x)$ will also be very large. In this case $\phi(x)$ will be very small and tending towards zero. On the other hand, when x is a large positive number, $exp(x)$ will be very small, and $\phi(x)$ will be very close to one.

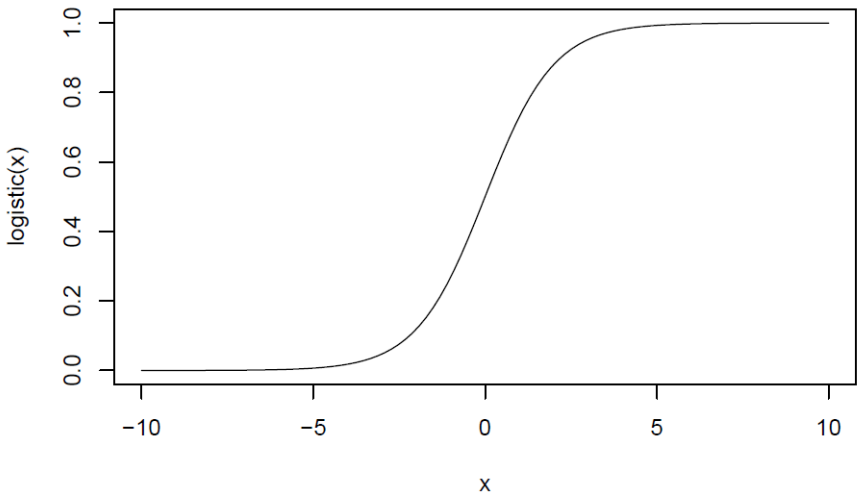

Figure 11: Logistic Regression.

Logistic Regression is chosen as the baseline model in this project since it is relatively straightforward and efficient to implement.

The accuracy is 0.85371, precision is 0.81694, recall is 0.87121, F2 score is 0.85979. The confusion matrix for the Logistic Regression model shows below:

		Prediction	
		0	1
Actual	0	$d/9010 - TrueNegative$	$c/1725 - FalsePositive$
	1	$b/1138 - FalseNegative$	$a/7698 - TruePositive$

7.3.2 XGBoost

XGBoost is one of the most popular and efficient implementations of the Gradient Boosted Trees algorithm, it is created by Tianqi Chen [2]. It is based on function approximation by optimising specific loss functions as well as applying several regularisation techniques. The mathematical theory behind XGBoost is relatively complicated, due to page constraints, here we list two most important features in XGBoost: fast execution speed and good model performance. It has been agreed that XGBoost is the go-to algorithm for competition on the Kaggle competitive data science platform and it normally shows very good performance.

XGBClassifier from package xgboost.sklearn has been adopted in this project. I chose 3 as the maximum tree depth and 100 as the number of estimators.

The accuracy is 0.91319, precision is 0.92964, recall is 0.89424, F2 score is 0.90111. The confusion matrix for the XGBoost model shows below:

		Prediction	
		0	1
Actual	0	$d/9112 - TrueNegative$	$c/663 - FalsePositive$
	1	$b/1036 - FalseNegative$	$a/8760 - TruePositive$

7.3.3 Neural Networks

Neural networks was first introduced by Warren McCulloch and Walter Pitts and it is inspired by biological neural networks that constitute animal brains [3]. Nowadays, neural networks has been widely applied in machine learning tasks. It can learn complex patterns using layers of nodes which mathematically transform the data. The basic architecture of a neural network model can be described in Figure 12. This figure shows the basic architecture of a neural network model, which includes an input layer, multiple hidden layers and an output layer [4].

Assume $x_1, x_2, ..., x_n$ are inputs, $w_1, w_2, ..., w_n$ are the weights for each input and the output y_1 with bias b included can be written as:

$$y_1 = f(b + \sum_{i=1}^{n} w_i \cdot x_i) \tag{3}$$

The main concept of a neural network model is to find the optimal values for each weight w_i, the basic procedure can be described as:

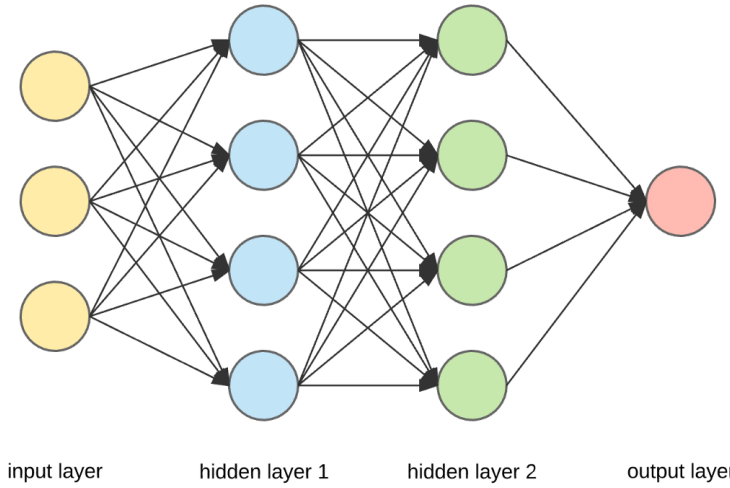

Figure 12: The main architecture of a neural network model [5]

- Initialise the weights for each input.
- Fit NN model and get the value for the output.
- Calculate the error between the output and the actual value.
- Adjust the weights to minimise the error.
- Repeat the third and the fourth steps until the error has been minimised.

Keras with Tensorflow backend is used for building Neural Network model in this project. According to the structure of the features and labels, a stack of fully-connected (Dense) layers with ReLU activation function is adopted in the model. A ReLU (rectified linear unit) is a function meant to zero-out negative values and produce something that can be interpreted as a probability [0 - 1].Here are the details for the hyperparameters:

- Three hidden layers, the first hidden layer has 128 hidden units with ReLU activation function, the second and third hidden layer have 16 hidden units with ReLU activation function.
- The network ends with a Dense layer with 1 unit and a sigmoid activation because the output should be a scalar between 0 and 1, encoding a probability.
- The loss function is binary_crossentropy loss as this is a binary classification problem, crossentropy is usually the best choice when we are dealing with models that output probabilities.
- I first created a validation set by setting apart 9500 samples (around 20% of the training set) from the original training set, then train the model with 36163 samples with 20 epochs and batch size of 128,the batch size defines the number of samples that will be propagated though the network.

From the observation of the accuracy for the validation set, it ranges from around 0.86 to 0.91, which is fairly stable and this result is quite good, this means the model is not overfitting, the reason I check this is because a model that performs better on the training data isn't necessarily a model that will do better on data it has never seen before. For an overfitting model, it will end up learning representations that are specific to the training data and do not generalise to data outside of the training set.

- Retrained the model with the full set of the training set with the same hyperparameters.

The accuracy is 0.90864, precision is 0.91383, recall is 0.89820, F2 score is 0.90128. The confusion matrix for the Neural Network model shows below:

		Prediction	
		0	1
Actual	0	$d/9172 - TrueNegative$	$c/812 - FalsePositive$
	1	$b/976 - FalseNegative$	$a/8611 - TruePositive$

8 Model Interpretation

As suggested in the proposal review, SHAP (SHapley Additive exPlanations) can be adopted to interpret the model output. It is a game theoretic approach to explain the output of any machine learning model.

In the report, we focus on the interpretation of XGBoost model as it is a tree ensemble model and we can use TreeExplainer provided by shap package.

Figure 13 plots the SHAP value of every feature for every sample and shows us an overview of which features are the most important for the model. As we can see, 'total_amount', 'reward' and 'social' are the top three most important features for this XGBoost model.

The SHAP values represent a feature's responsibility for a change in the model output, combined with Figure 14, we can see that each feature contributing to push the model output from the base value, features pushing the prediction higher are shown in red, those pushing the prediction lower are in blue.

Note that 'higher' means the result is more likely to be '0', so in the plots below the 'blue' features are actually helping raise the chance of an successful offer (a value of '1'), while the negative features are lowering the chance.

9 Model Comparison

The table below illustrate the evaluation metrics of each model:

From this table, we can see that XGBoost has the highest accuracy, recall and F2 score, it has the least number of False Negative (663), as mentioned above, in this

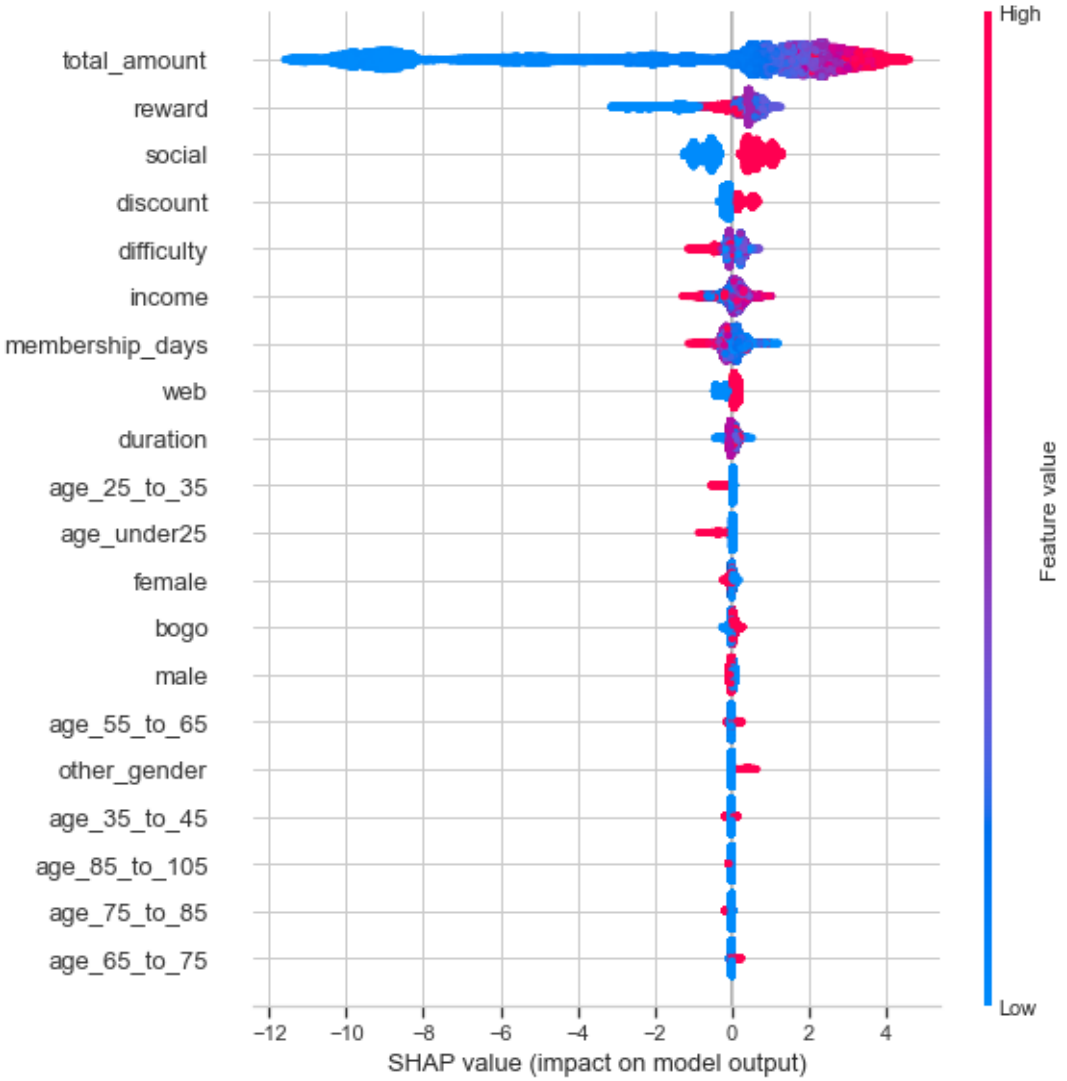

Figure 13: Visualisation of the effects of all the features.

Figure 14: Visualisation of the first prediction's explanation.

Figure 15: Visualisation of the second prediction's explanation.

business case, we want to avoid the situation of sending offers to customers who are not likely to use them as much as possible. Hence, we want the number of False Negative as less as possible.

The baseline Logistic Regression model shows the worst performance with the lowest F2 score and the biggest number for False Positive.

The performance of the Neural Network model is very similar to XGBoost with a similar accuracy and F2 score.

10 Conclusion

In conclusion, the experiments provide some insight into this business case. The performances of XGBoost model is very promising, with the F2 score of 0.90111. Normally with a huge amount of training dataset, the performance of Neural Network model will also be promising, however it is usually considered as a 'black box', which means the interpretation of Neural Network model is relatively difficult than other models like XGBoost, and the performance of Neural Network model is not very stable.

11 Future Work

Future experiments can be conducted by exploring more models such as Support Vector Machines (SVM), or ensemble several models together. Moreover, although the results of our models seem promising, we can still use hyperparameter tuning

Model	Accuracy	Precision	Recall	F2 Score	TP	FP	TN	FN
Logistic Regression	0.85371	0.87121	0.81694	0.82724	7698	1138	9010	1725
XGBoost	0.91319	0.89424	0.92964	0.92234	8760	1036	9112	663
Neural Network	0.90864	0.89820	0.91383	0.91066	8611	976	9172	812

to improve the results and lower the number of False Negative as low as possible and maintain a relatively high F2 score. Due to time constraints, these tasks have not been conducted yet, but this could be a promising direction to continue.

References

[1] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[2] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[3] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[4] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[5] Arden Dertat. Applied deep learning - part 1: Artificial neural networks. `https://towardsdatascience.com`, 2017. [Online; accessed 10-August-2019].