Capstone Project
Data Science Nanodegree

Bhavesh Jakhar

July 31st , 2020

Starbucks Capstone Project
Project   Definition

Project Overview :

This data set contains simulated data that mimics customer behavior on the Starbucks rewards mobile app. Once every few days, Starbucks sends out an offer to users of the mobile app. An offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users might not receive any offer during certain weeks.

Not all users receive the same offer, and that was the challenge to solve with this data set.

Our task is to combine transaction, demographic and offer data to determine which demographic groups respond best to which offer type. This data set is a simplified version of the real Starbucks app because the underlying simulator only has one product whereas Starbucks actually sells dozens of products.

Every offer has a validity period before the offer expires. As an example, a BOGO offer might be valid for only 5 days. Informational offers also have a validity period even though these ads are merely providing information about a product; for example, if an

informational offer has 7 days of validity, we can assume the customer is feeling the influence of the offer for 7 days after receiving the advertisement.

Data Dictionary :

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)

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

Offer Types :

- 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.

Problem Statement :

Predicting the purchase offer to which a possible higher level of response or user actions like 'offer received', 'offer viewed', 'transaction' and 'offer completed' can be achieved based on the demographic attributes of the customer and other attributes of the companies purchase offers.

Evaluation Metrics :

Accuracy is the quintessential classification **metric**. ... And easily suited for binary as well as a multiclass classification problem.

  Accuracy = (TP+TN)/(TP+FP+FN+TN)

$$accuracy = \frac{true\ positives + true\ negatives}{dataset\ size}$$

Accuracy is the proportion of true results among the total number of cases examined.

- became_member_on: (date) format YYYYMMDD

- income: (numeric)

Data Accessing and Cleaning :

1. Portfolio Data

- create a copy of the original dataframe for further implementation .

- convert the column 'Channels' into 4 different channels on the basis of different types of channel .

- rename the column name from 'ID' to 'offer_id' .

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

2. Profile Data

- create a copy of the original dataframe for further implementation .

- convert the datatype of 'became_member_on' column and sort the date into proper format .

- change the column name from 'ID' to 'customer_id' .

	gender	age	customer_id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	2017-02-12	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	2017-07-15	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	2018-07-12	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	2017-05-09	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43	2017-08-04	NaN
5	M	68	e2127556f4f64592b11af22de27a7932	2018-04-26	70000.0
6	None	118	8ec6ce2a7e7949b1bf142def7d0e0586	2017-09-25	NaN
7	None	118	68617ca6246f4fbc85e91a2a49552598	2017-10-02	NaN
8	M	65	389bc3fa690240e798340f5a15918d5c	2018-02-09	53000.0
9	None	118	8974fc5686fe429db53ddde067b88302	2016-11-22	NaN

3. Transcript Data

- create a copy of the original dataframe for further implementation .

- change the column name from 'person' to 'customer_id' .

- convert the column 'Event' into 4 different columns on the basis of different types of event .

- convert the column 'Values' into 2 different columns .

	customer_id	event	time	offer-completed	offer-received	offer-viewed	transaction	offer_id	amount
0	78afa995795e4d85b5d9ceeca43f5fef	offer-received	0	0	1	0	0	9b98b8c7a33c4b65b9aebfe6a799e6d9	NaN
1	a03223e636434f42ac4c3df47e8bac43	offer-received	0	0	1	0	0	0b1e1539f2cc45b7b9fa7c272da2e1d7	NaN
2	e2127556f4f64592b11af22de27a7932	offer-received	0	0	1	0	0	2906b810c7d4411798c6938adc9daaa5	NaN
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer-received	0	0	1	0	0	fafdcd668e3743c1bb461111dcafc2a4	NaN
4	68617ca6246f4fbc85e91a2a49552598	offer-received	0	0	1	0	0	4d5c57ea9a6940dd891ad53e9dbe8da0	NaN

Data Cleaning :

- Concatenate all the three datasets together .

- Fixed the Offer_ids .

- fixed even_ids

	customer_id	event	time	offer-completed	offer-received	offer-viewed	transaction	offer_id	amount	gender	...	income	reward	difficulty
0	78afa995795e4d85b5d9ceeca43f5fef	offer-received	0	0	1	0	0	0.0	NaN	F	...	100000.0	5.0	5.0
1	78afa995795e4d85b5d9ceeca43f5fef	offer-viewed	6	0	0	1	0	0.0	NaN	F	...	100000.0	5.0	5.0
2	78afa995795e4d85b5d9ceeca43f5fef	transaction	132	0	0	0	1	NaN	19.89	F	...	100000.0	NaN	NaN
3	78afa995795e4d85b5d9ceeca43f5fef	offer-completed	132	1	0	0	0	0.0	NaN	F	...	100000.0	5.0	5.0
4	78afa995795e4d85b5d9ceeca43f5fef	transaction	144	0	0	0	1	NaN	17.78	F	...	100000.0	NaN	NaN

5 rows × 22 columns

Data Analysis

Data Exploration and Data Visualization :

1. Age Group :

Observation :

- Outlier is present Age > 115 is present is a high amount , which does not make sense .

- Average Aged user is middle age ie. around 50-62 years
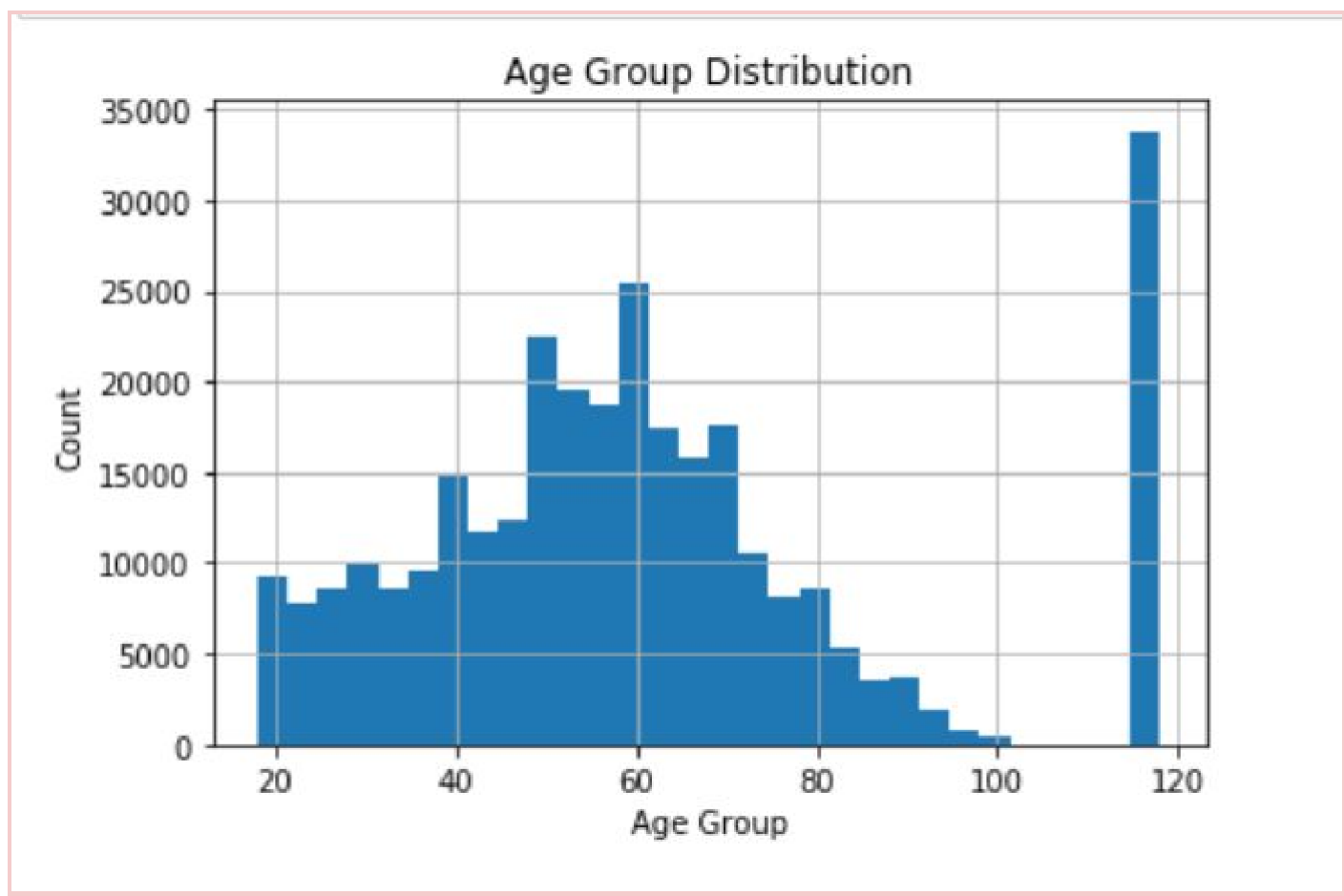
count	306534.000000	mean	60.909367	std	26.032030

min	18.0	25%	43.0	50%	57.0	75%	72.0	max	118.0

Age Group Distribution

2. Income Range :

Observation :

- Average income user is middle income group ie. 65000-70000
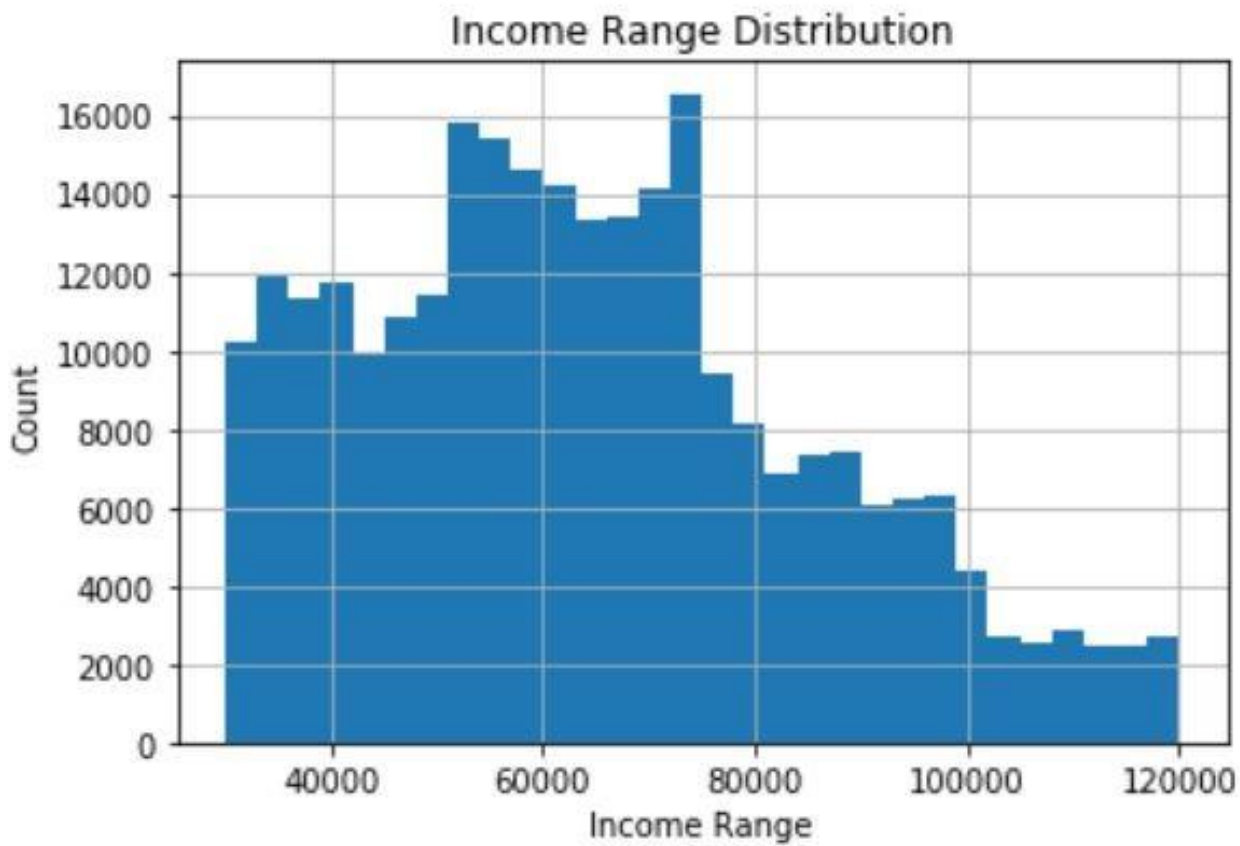
count	272762.00	mean	64337.000755	std	21243.762941

min	30000	25%	48000	50%	62000	75%	78000	max	120000

Income Range Distribution

3. Gender :

Observation :

● males are more than 50 percent of users .

male_proportion = 50.79045065147749,

female_proportion = 36.89672271265177,

Others_proportion = 1.2954517280301696

Gender Distribution
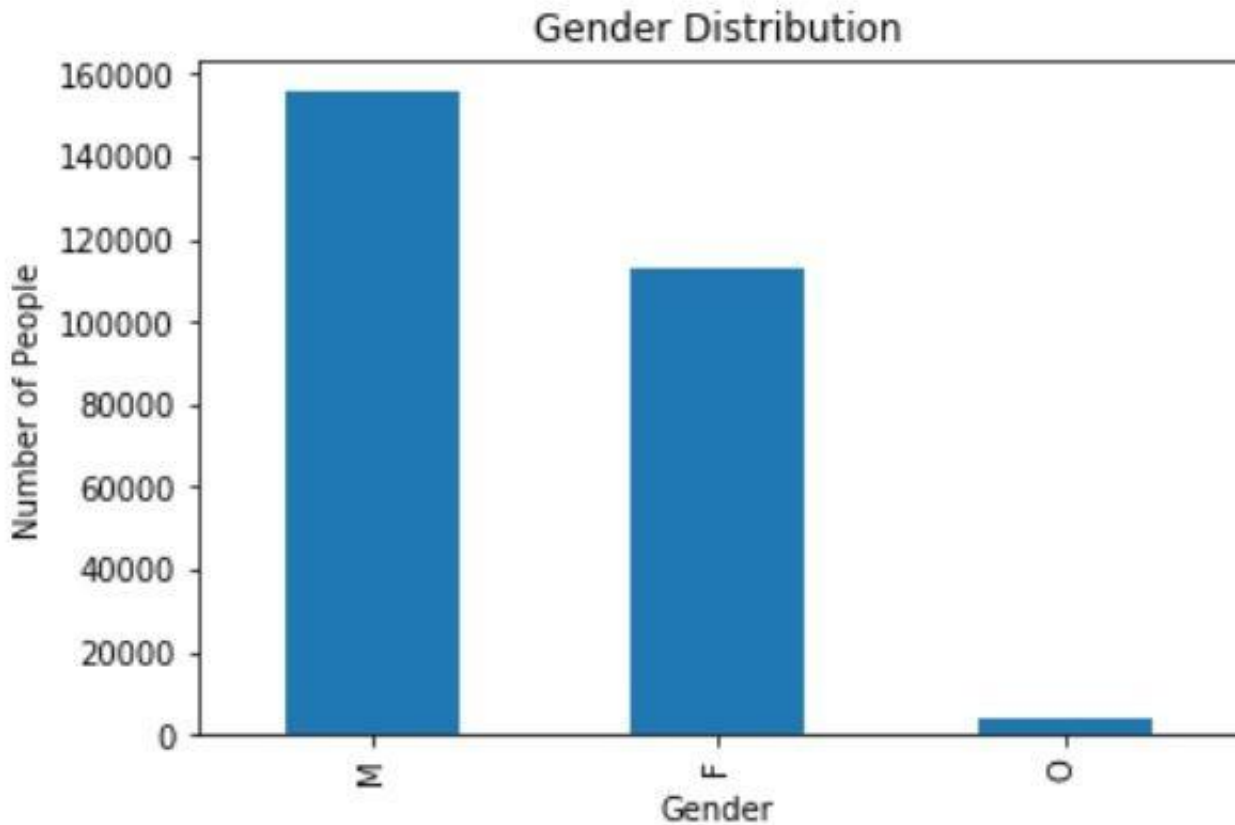
# 4 . No. of Different Type of Offers Received by the Users :

Observation :

- BOGO offers are highly demanding , 30499 users received BOGO offers, 25449 viewed the offer and 15669 completed it.

- The percentage of BOGO Offer viewers is 83 percent .

- The percentage of DISCOUNT Offer viewers is 70 percent .

For BOGO Offer :

Offer _Viewed_proportion = 83.44% ,

Offer_Completed_proportion = 51.37%
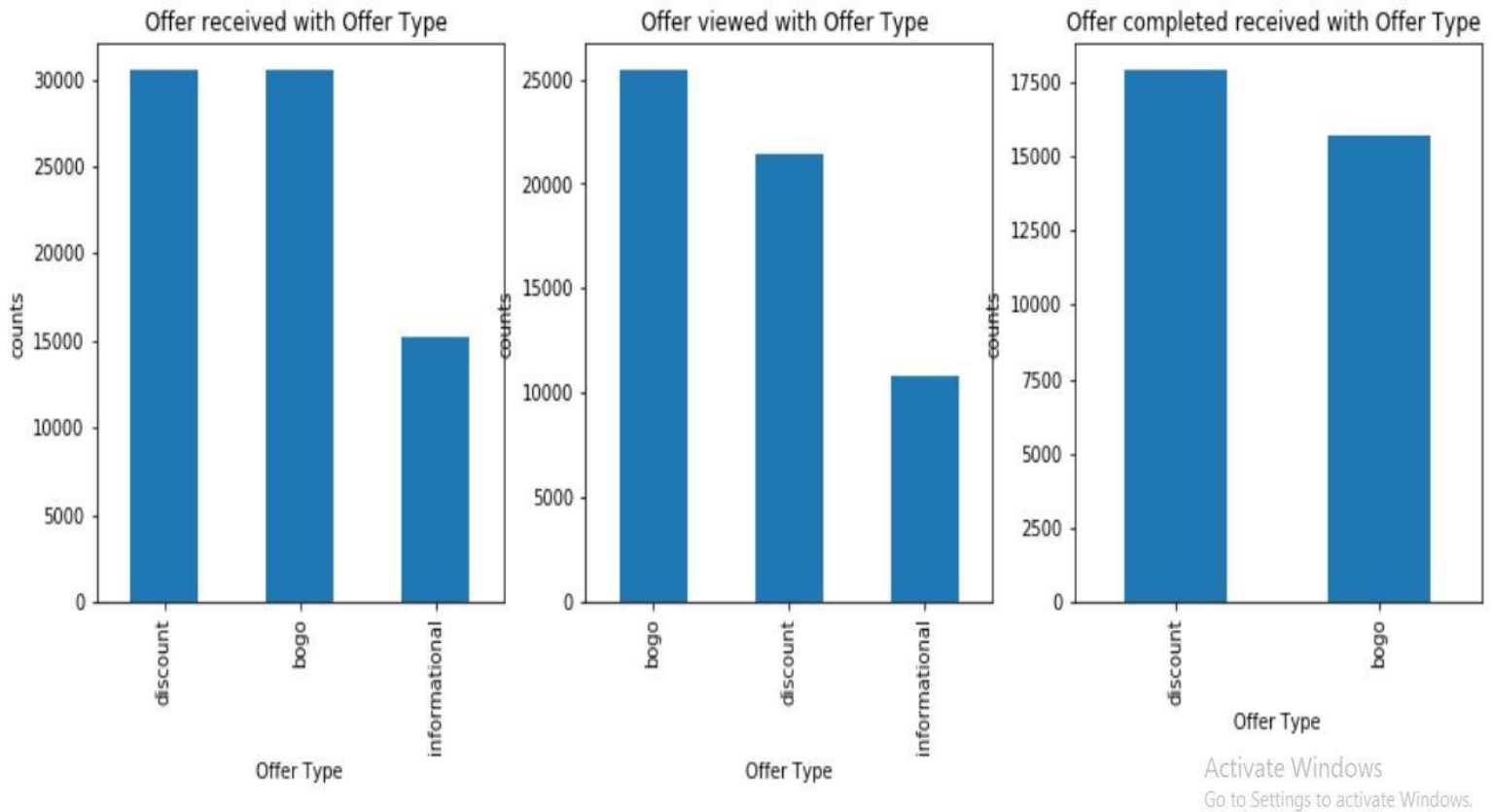
5 . No. of Offers_Id Received Different types of Offers :

Observation :

- evry offer_id received eual offers .
- Viewing ratio decreased for some offer_ids like 0 , 6 , 7 , 5
- Offer completed ration is quite decent .

Offer received with Offer Id — Offer viewed with Offer Id — Offer completed received with Offer Id
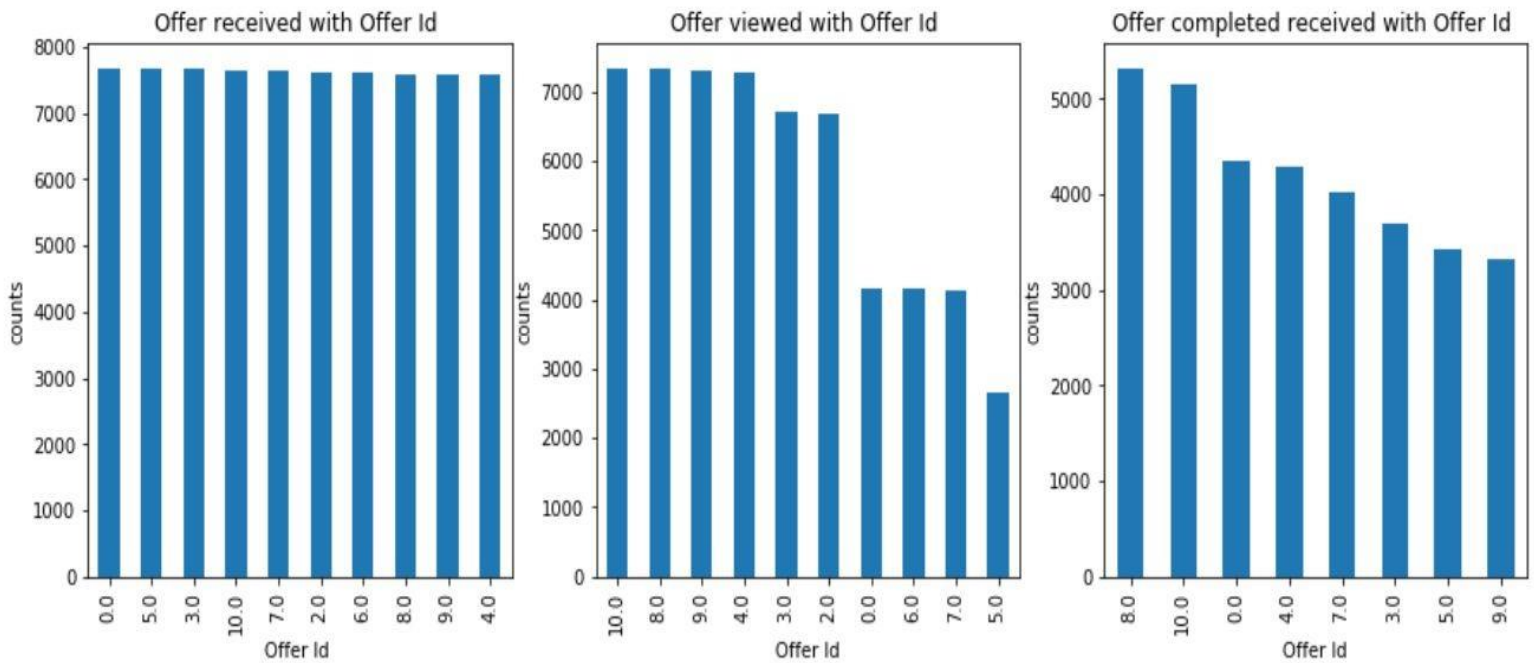
Algorithms and Technique :

The Classifier is **Artificial Neural Network (ANN) .** A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

A Simple Neural Network
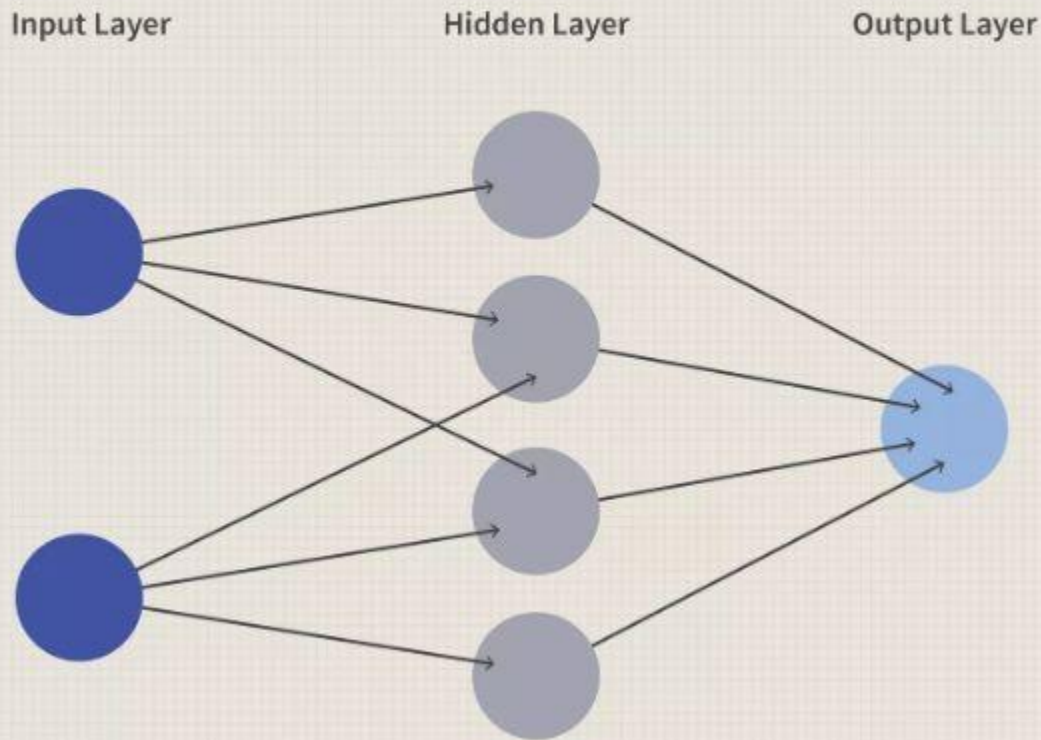
An artificial neural network (ANN) is the piece of a computing system designed to simulate the way the human brain analyzes and processes information. It is the foundation of Artificial Intelligence (AI) and solves problems that would prove impossible or difficult by human or statistical standards. ANNs have self-learning capabilities that enable them to produce better results as more data becomes available.

Inputs

Source data fed into the neural network, with the goal of making a decision or prediction

about the data. Inputs to a neural network are typically a set of real values; each value is fed into one of the neurons in the input layer.

Training Set

A set of inputs for which the correct outputs are known, used to train the neural network.

Outputs

Neural networks generate their predictions in the form of a set of real values or boolean decisions. Each output value is generated by one of the neurons in the output layer.

Neuron/perceptron

The basic unit of the neural network. Accepts an input and generates a prediction.

Each neuron accepts part of the input and passes it through the activation function. Common activation functions are sigmoid, TanH and ReLu. Activation functions help generate output values within an acceptable range, and their non-linear form is crucial for training the network .

Error Function

Defines how far the actual output of the current model is from the correct output. When training the model, the objective is to minimize the error function and bring output as close as possible to the correct value.

Hyperparameters

A hyperparameter is a setting that affects the structure or operation of the neural network. In real deep learning projects, tuning hyperparameters is the primary way to build a network that provides accurate predictions for a certain problem. Common hyperparameters include the number of hidden layers, the activation function, and how many times (epochs) training should be repeated.

Methodology

Data Preprocessing :

One hot encoding :

Encoding for Gender column and Offer_type Column (Pre Model Preparation)

Splitting Dataset :

The train_test_split function is for splitting a single dataset for two different purposes: training and testing. The testing subset is for building your model. The testing subset is for using the model on unknown data to evaluate the performance of the model.

Feature Scaling :

Standardization & Normalization

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

$$X' = \frac{X - \mu}{\sigma}$$

Pandas Dataframe.to_numpy() – Convert dataframe to Numpy array :

Pandas Dataframe is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). This data structure can be converted to NumPy ndarray with the help of **Dataframe.to_numpy()** method.

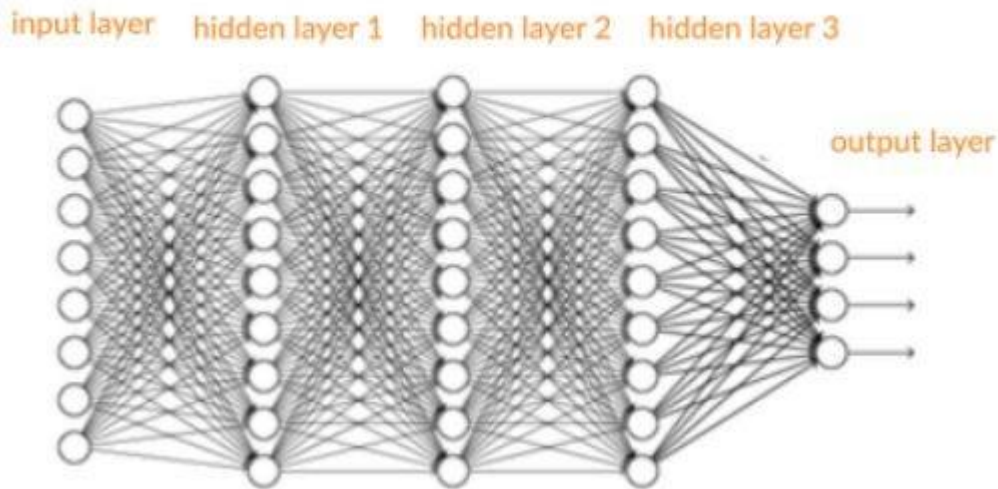# Implementation :

Build a Model :

Model: "sequential_2"

Layer (type)	Output Shape	Param #
dense_8 (Dense)	(None, 6)	90
dense_9 (Dense)	(None, 6)	42
dense_10 (Dense)	(None, 4)	28

Total params: 160

Trainable params: 160

Non-trainable params: 0

Deep neural network

input layer hidden layer 1 hidden layer 2 hidden layer 3

output layer

Activation Function :

Layer 1 and 2 [Input Layer and hidden Layer] :

The ReLu function is highly computationally efficient but is not able to process inputs that approach zero or negative.

Layer 3 [Output Layer] :

Softmax is a special activation function used for output neurons. It normalizes outputs for each class between 0 and 1, and returns the probability that the input belongs to a specific class.
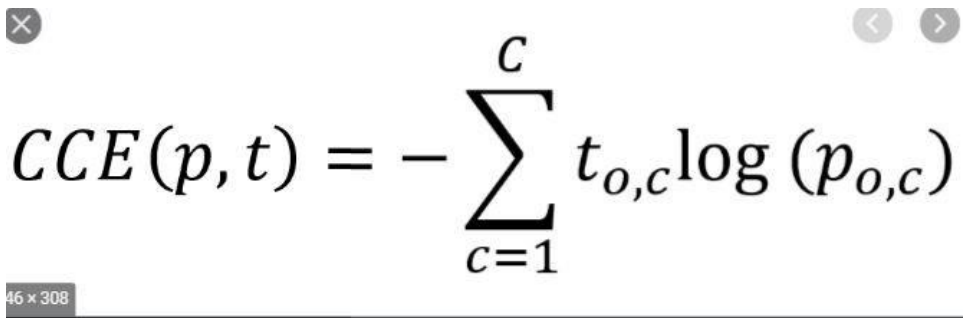
Optimizer :

Adam is a replacement **optimization** algorithm for stochastic gradient descent for training deep learning models. **Adam** combines the best properties of the AdaGrad and RMSProp

algorithms to provide an **optimization** algorithm that **can** handle sparse gradients on noisy problems.

Loss :

sparse categorical cross entropy when your classes are mutually exclusive (e.g. when each sample belongs exactly to one class) and **categorical cross entropy** when one sample can have multiple classes or labels are soft probabilities (like [0.5, 0.3, 0.2]).

$$CCE(p, t) = -\sum_{c=1}^{C} t_{o,c} \log (p_{o,c})$$

46 × 308

Refinement :

Improving Prediction Model :
The model undergoes underfit ie. High Bias .
Bias:It gives us how closeness is our predictive model to training data after averaging predicted values. Generally algorithms have high bias which help them to learn fast and easy to understand but are less flexible. That loses its ability to predict complex problems, so it fails to explain the algorithm bias. This results in underfitting of our model.

How to overcome Underfitting ?

1. Use a bigger network .
    1. more hidden layers .
    2. more hidden units .
2. Train model longer .
3. Advanced optimization algorithm .

```
==========================================================================
dense_4 (Dense)              (None, 32)                       256

dropout_1 (Dropout)          (None, 32)                       0

dense_5 (Dense)              (None, 15)                       495

dropout_2 (Dropout)          (None, 15)                       0

dense_6 (Dense)              (None, 10)                       160

dropout_3 (Dropout)          (None, 10)                       0

dense_7 (Dense)              (None, 6)                        66

dropout_4 (Dropout)          (None, 6)                        0

dense_8 (Dense)              (None, 4)                        28

dense_9 (Dense)              (None, 32)                       160

dense_10 (Dense)             (None, 15)                       495

dense_11 (Dense)             (None, 10)                       160

dense_12 (Dense)             (None, 6)                        66

dense_13 (Dense)             (None, 4)                        28
==========================================================================
Total params: 1,914
Trainable params: 1,914
Non-trainable params: 0
```

Results

Model Evaluation and Validation :

During development , a validation set was used to evaluate the model .

The final architecture and hyperparameters were chosen because they performed better .

```
Train on 214573 samples, validate on 91961 samples
Epoch 1/15
 - 14s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 2/15
 - 13s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 3/15
 - 13s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 4/15
 - 13s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 5/15
 - 17s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 6/15
 - 19s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 7/15
 - 16s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 8/15
 - 18s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 9/15
 - 15s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 10/15
 - 15s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 11/15
 - 15s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 12/15
 - 15s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 13/15
 - 15s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 14/15
 - 16s - loss: nan - accuracy: 0.2481 - val_loss: nan - val_accuracy: 0.2507
Epoch 15/15
```

```
In [128]: ann.evaluate(X_test , y_test)

          91961/91961 [==============================] - 6s 68us/step

Out[128]: [nan, 0.25066059827804565]
```

Justification :

In the Refinement step we performed several Underfitting techniques .

1. **Kernel Initializer : "Normal"** The neural network needs to start with some weights and then iteratively update them to better values. The term kernel_initializer is a fancy term for which statistical distribution or function to use for initialising the weights. In case of statistical distribution, the library will generate numbers from that statistical distribution and use them as starting weights.

    For example in the above code, normal distribution will be used to initialise weights. You can use other functions (constants like 1s or 0s) and distributions (uniform) too.

2. Increased no. of hidden layer .

3. Increased no. of hidden units .

Conclusion

Reflection :

```
In [130]: # PROBABILITY OF EACH SET

          y_prob = ann.predict(x_new)
          y_prob.round(2)                          # RESULT IN 2 DECIMAL PLACE
Out[130]: array([[nan, nan, nan, nan],
                  [nan, nan, nan, nan],
                  [nan, nan, nan, nan]], dtype=float32)

In [131]: # CLASS OF EACH SET

          y_pred = ann.predict_classes(x_new)
          y_pred
Out[131]: array([0, 0, 0], dtype=int64)

In [132]: np.array(class_name)[y_pred]
Out[132]: array(['offer recieved', 'offer recieved', 'offer recieved'], dtype='<U15')
```

- I found this project challenging, mainly due to the structure of the data in the transcript dataset.
- Majority classes are performing well but the minorities are not.Problem of imbalanced dataset
- Most of the events are wrongly predicted as 'offer received'; offer received is the most occuring event or class.
- The main goal I chose was to build something practical the company could use to make their choices more efficient.
- But the results of the model seem not so good . There is no change in rate of accuracy; it remains constant .

Improvement

- As we're using an imbalanced dataset we weren't expecting to get a greater accuracy .

- The accuracy could be improved significantly by using deep neural networks , or using recommendation engines .

- Analysing and building the models was Main challenging part and needed improvement .

References :

- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.replace.html

- https://pandas.pydata.org/pandas-docs/version/0.23.4/generated/pandas.merge.html

- https://stackoverflow.com/questions/37600711/pandas-split-column-into-multiple-columns-by-comma

-https://www.researchgate.net/post/Is_there_a_universal_method_rule_to_choose_the_activation_functi on_for_a_MLP_neural_network#:~:text=For%20binary%20classification%20(i.e.%20problems,entropy %20as%20the%20cost%20function.

-https://stackoverflow.com/questions/55324762/the-added-layer-must-be-an-instance-of-class-layer-foun d-tensorflow-python-ke

-https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standar dization/

- https://www.tensorflow.org/tutorials/keras/classification

Thank You