Pradip Goban S S
5 April 2020

Starbucks Capstone Project Report

Machine Learning NanoDegree

Project Overview

Starbucks Corporation is an American coffee company. Starbucks was founded in Seattle, Washington, in 1971. As of early 2019, the company operates over 30,000 locations worldwide.

Its quite interesting to see how different people respond to or how they don't respond to an offer, the factors that come into play in making that decisions, giving an insight into the complex working for the human mind, thinking about even the mundane and everyday tasks such as buying coffee.

Every offer has a validity period before the offer expires. As an example, a BOGO offer might be valid for only 5 days. You'll see in the data set that informational offers 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, you can assume the customer is feeling the influence of the offer for 7 days after receiving the advertisement.

If the company sends more offers or irrelevant offers to a person then there is no use. That's why they should send only relevant offers to a person who is likely going to use this offer which benefits the company. We need to predict who are more likely going to use a certain offer and send that offer only to them to make company beneficial.

Problem Statement

The problem can be formulated in to a binary classification problem, where there can be two possible outcome cases:

1. A person will respond to an offer encoded as (1)
2. A person will not respond to an offer encoded as (0)

The metrics that will be used in solving this problem is Accuracy. Based on this metric we will calculate how likely a person is going to respond to this offer making it as 0 or 1.

Steps:

1. **Feature Engineering :** Removing the redundant features, Analysing the relation between features.

2. **Preprocessing :** One-hot-encoding the data, handling "null" values, dimensionality reduction.

3. **Splitting the data :** Splitting the data into training and test dataset.

4. **Model :** Building a Deep Learning Model with PyTorch.

5. **Training :** Training the model with train dataset.

6. **Testing :** Testing the previously trained model with test data.

7. **Tuning :** Tuning the hyperparameters of the model to get better results.

Metrics

I have used F_1 score to select the benchmark, **F1-score** . F1-score = 2 * (precision * recall) / (precision + recall)

Finally used Accuracy to validate the selected model.

II. Analysis

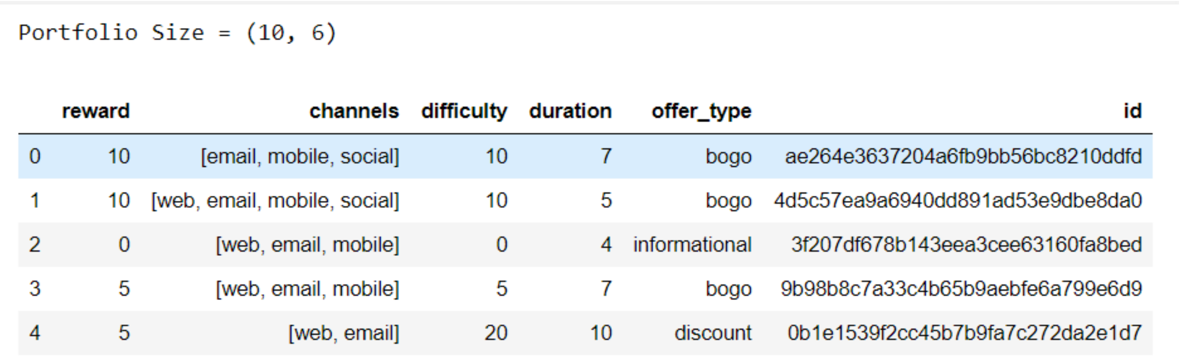
Data Exploration

The data is contained in three files:

1. portfolio.json - containing offer ids and meta data about each offer (duration, type, etc.) 2. profile.json - demographic data for each customer
3. transcript.json - records for transactions, offers received, offers viewed, and offers

completed
Here is the schema and explanation of each variable in the files: **1. portfolio.json**

id (string) - offer id
offer_type (string) - type of offer i.e, BOGO, discount, informational difficulty (int) - minimum required spend to complete an offer reward (int) - reward given for completing an offer
duration (int) - time for offer to be open, in days
channels (list of strings)

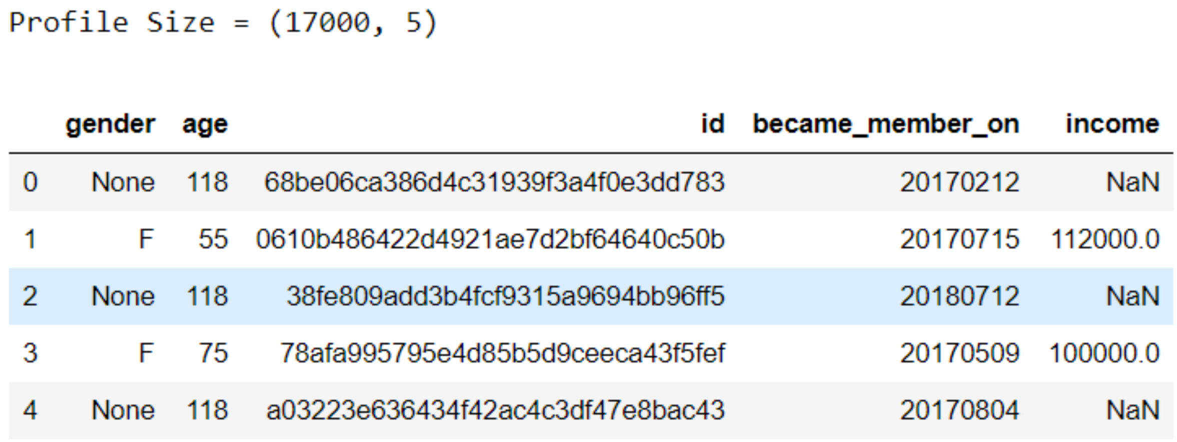


```
Portfolio Size = (10, 6)
```

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7

2. profile.json

age (int) - age of the customer
became_member_on (int) - date when customer created an app account
gender (str) - gender of the customer (note some entries contain 'O' for other rather than M or F)
id (str) - customer id
income (float) - customer's income

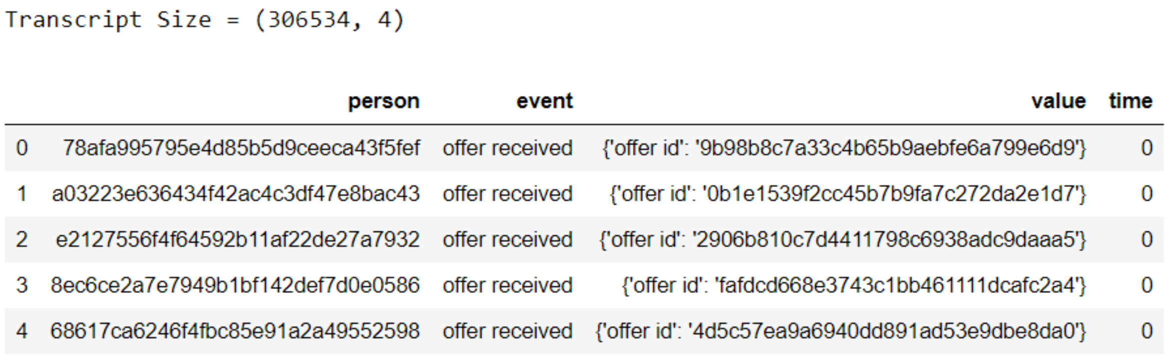


```
Profile Size = (17000, 5)
```

	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43	20170804	NaN

3. transcript.json

event (str) - record description (ie transaction, offer received, offer viewed, etc.)
person (str) - customer id
time (int) - time in hours since start of test. The data begins at time t=0
value - (dict of strings) - either an offer id or transaction amount depending on the record



```
Transcript Size = (306534, 4)
```

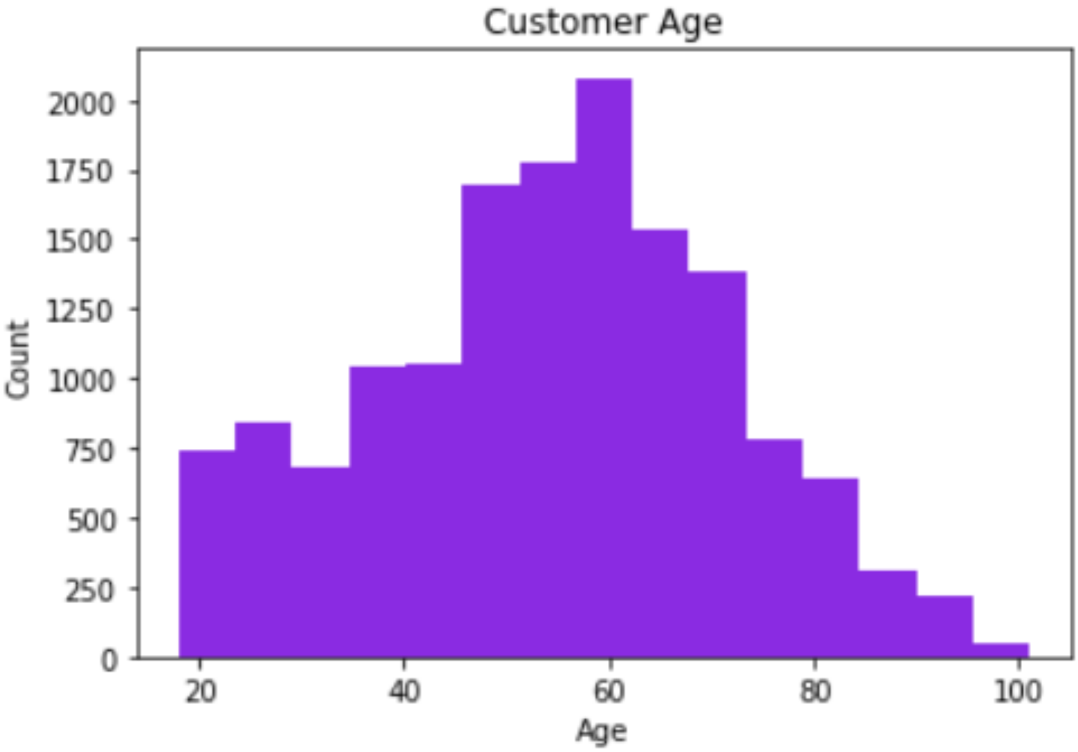
	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0
2	e2127556f4f64592b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	{'offer id': 'fafdcd668e3743c1bb461111dcafc2a4'}	0
4	68617ca6246f4fbc85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0

Here we can see some data is in number form and some data is in characters. Since a Machine Learning Model doesn't take input in characters, we need to convert the field which are not numbers to numbers by using **one-hot-encoding** method. Some of the values are high in number, it will be computationally expensive task, so we need to reduce these values such as field ***income*** in **profile.json** using **PCA** method.

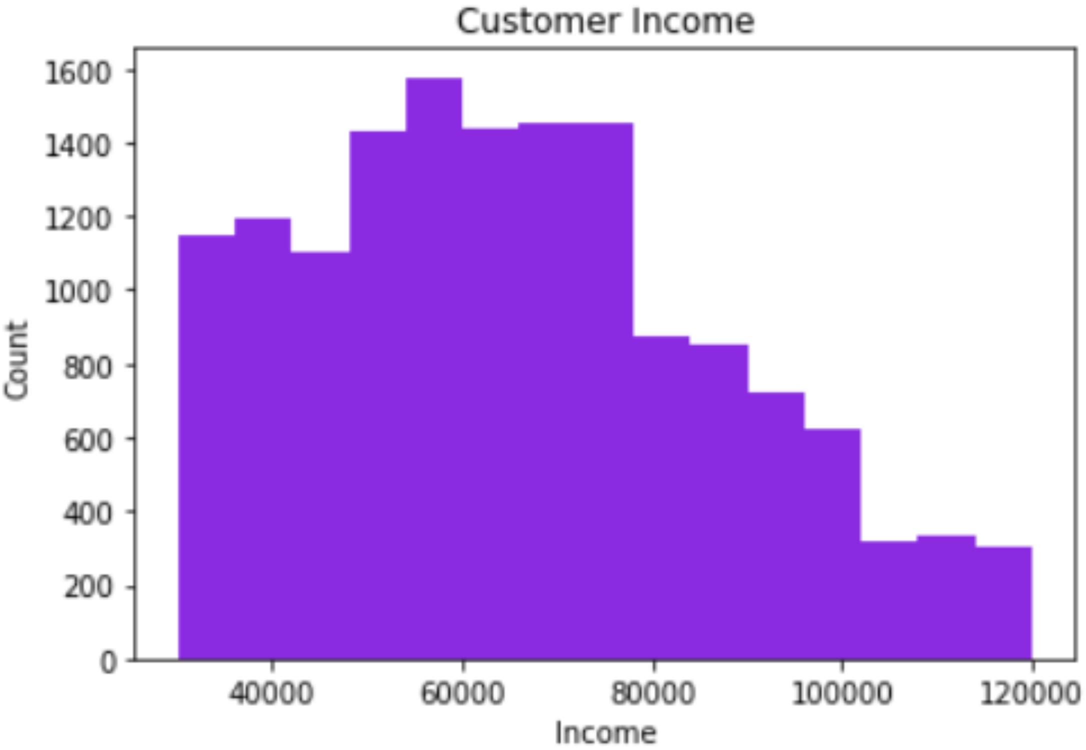
\

Exploratory Visualisation

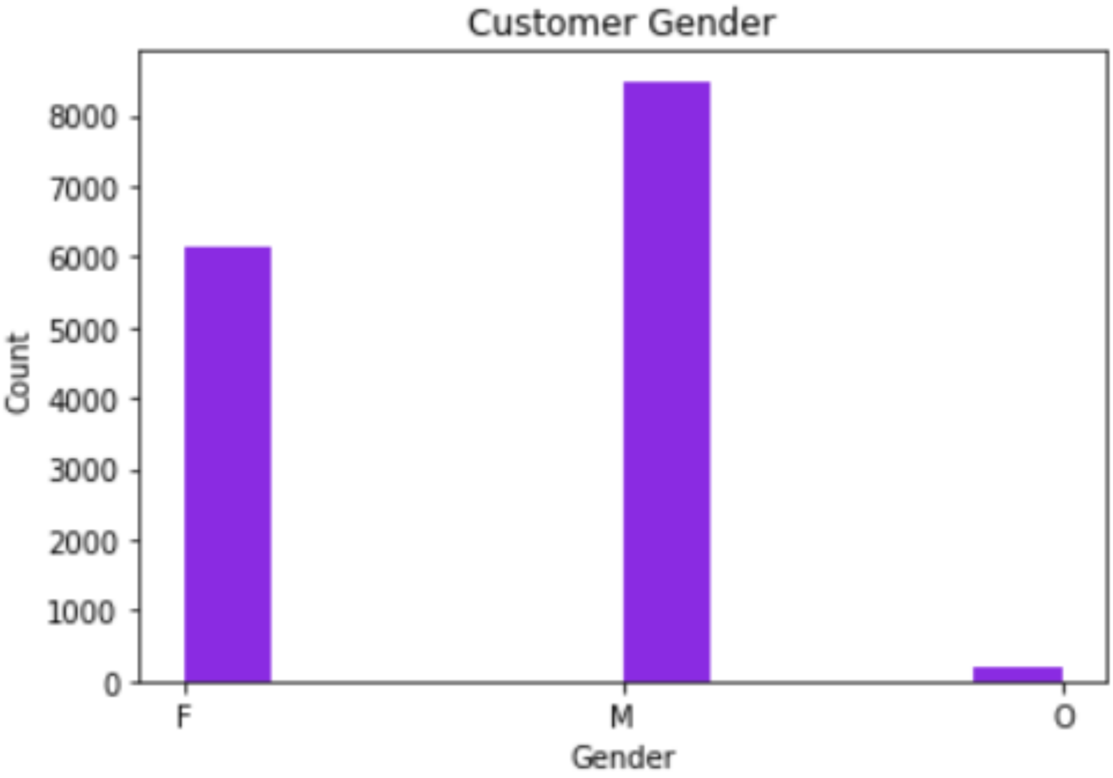
The Histogram is a normal distribution between customer age and count.



The Histogram is a right skewed distribution between customer income and count.



The chart shows between customer gender and count.



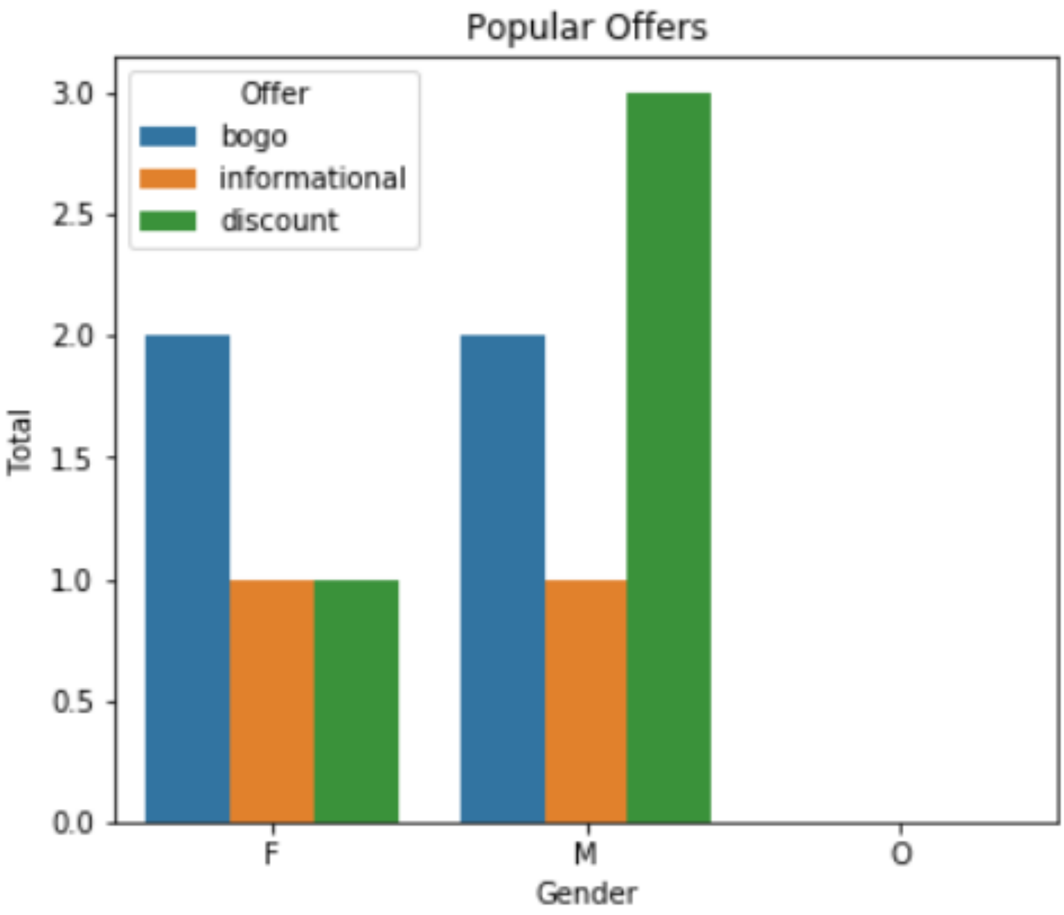
The chart shows between popular offer types.

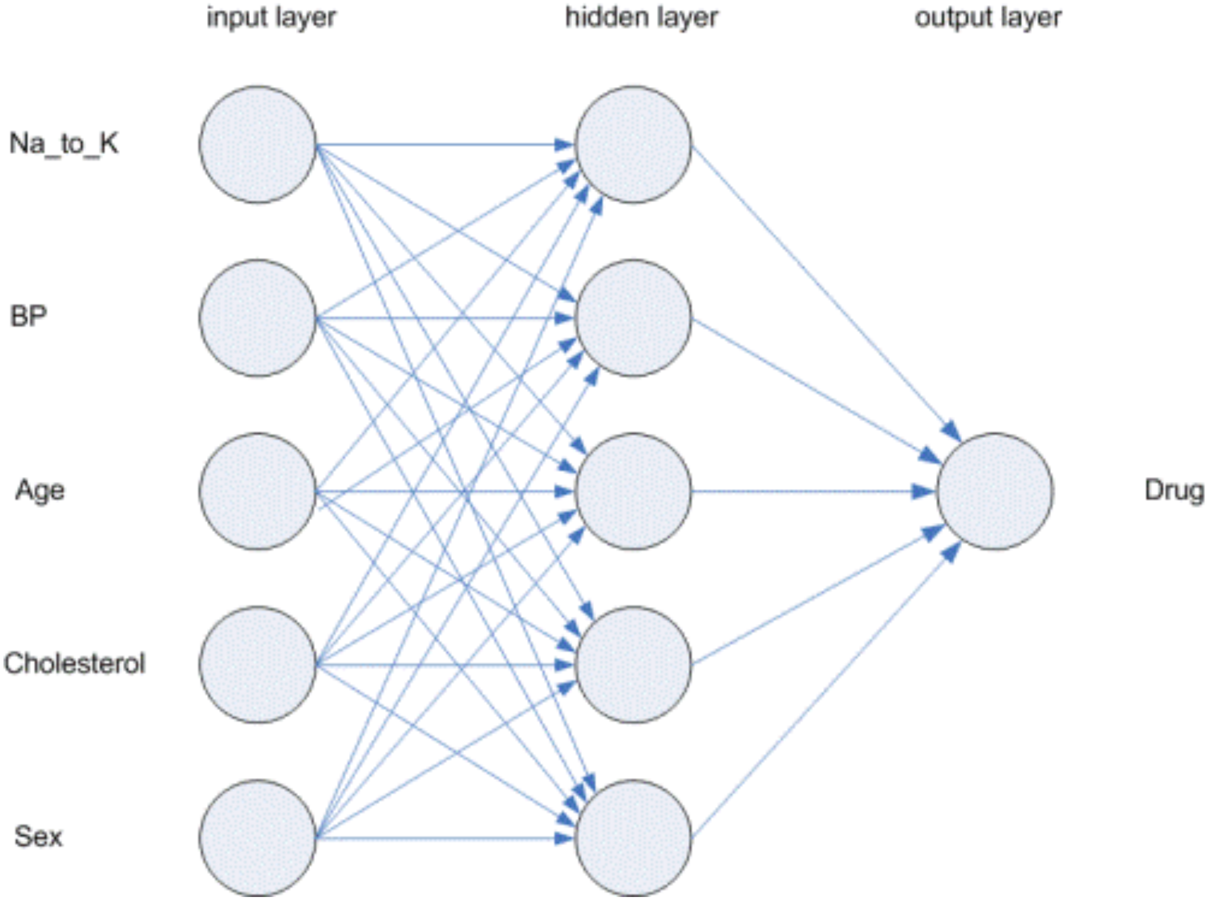
Algorithms and Techniques

As we are predicting that a person will/not respond to a given offer, this will be binary classification problem. So I am using here are a Binary Classifier and a Neural Network as Neural Network can be used for almost any task.

Neural Network Model :

Neural networks are simple models of the way the nervous system operates. The basic units are neurons, which are typically organised into layers, as shown in the following figure.


Popular Offers
Offer
bogo
informational
discount
Total
3.0
2.5
2.0
1.5
1.0
0.5
0.0
F
M
O
Gender


input layer
hidden layer
output layer
Na_to_K
BP
Age
Cholesterol
Sex
Drug

A neural network is a simplified model of the way the human brain processes information. It works by simulating a large number of interconnected processing units that resemble abstract versions of neurons.

The processing units are arranged in layers. There are typically three parts in a neural network: an input layer, with units representing the input fields; one or more hidden layers; and an output layer, with a unit or units representing the target field(s). The units are connected with varying connection strengths (or weights). Input data are presented to the first layer, and values are propagated from each neuron to every neuron in the next layer. Eventually, a result is delivered from the output layer.

The network learns by examining individual records, generating a prediction for each record, and making adjustments to the weights whenever it makes an incorrect prediction. This process is repeated many times, and the network continues to improve its predictions until one or more of the stopping criteria have been met.

Initially, all weights are random, and the answers that come out of the net are probably nonsensical. The network learns through training. Examples for which the output is known are repeatedly presented to the network, and the answers it gives are compared to the known outcomes. Information from this comparison is passed back through the network, gradually changing the weights. As training progresses, the network becomes increasingly accurate in replicating the known outcomes. Once trained, the network can be applied to future cases where the outcome is unknown.

K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition already in the beginning of 1970’s as a non-parametric technique.

A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor. It should also be noted that all three distance measures are only valid for continuous variables. In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when there is a mixture of numerical and categorical variables in the dataset. Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise but there is no guarantee. Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. Historically, the optimal K for most datasets has been between 3-10. That produces much better results than 1NN.

GaussianNB is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.
Since we are using the Gaussain one it predicts that the data used follows a normal distribution.
For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.
Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even
highly sophisticated classification methods.
Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

$$\underset{\text{Posterior Probability}}{P(c \mid x)} = \frac{\overset{\text{Likelihood}}{P(x \mid c)}\,\overset{\text{Class Prior Probability}}{P(c)}}{\underset{\text{Predictor Prior Probability}}{P(x)}}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

After I use both of the algorithms, I have used **F1-score** metric to choose the best model out of the benchmark models.

Benchmark

As the benchmark models I have used KNeighborsClassifier and GaussianNB to compare against a Neural Network model using the metric **F1-score**.

III. Methodology

Data Preprocessing **portfolio.json**

1. One-hot-encoding to be done to the categorial fields - "offer_type", "channels"

profile.json

1. Some of the rows in "gender" and "income" are Nan. we have dropped those rows.

2. "became_member_on" field has the date and need to be separated as "year" and
"month" columns.

transcript.json

1. One-hot-encoding to be done to the categorial fields - "offer_type", "channels".

2. One-hot-encoding to be done to the categorial fields - "event".

3. In "value" field inside json object there are 2 keys i.e., "offer id" and "offer_id". These key
names have been changed to "offer_id".

4. Fill with '0' in "amount" and "reward".

Merging "transcript" and "profile" with field "person" and "id" = transcript_profile
Merging "transcript_profile" and "portfolio" with field "offer_id" = transcript_profile_portfolio

Implementation

Neural Network Model has been used to train the data here. **PyTorch**

Model :

1. Linear Layer = 24 -> 10 2. Dropout = 0.3
3. Linear Layer = 10 -> 1 4. Sigmoid()

Optimiser = SGD **Criterion** = BCELoss **Learning Rate** = 0.001 **Momentum** = 0.5 **Epochs** = 10

Refinement

I have used Adam optimiser initially but the model didn't perform well. It was giving me usual outputs other than 0 to 1 even though I have used Sigmoid activation function. Later I changed it to SDG and it performed well.

IV. Results

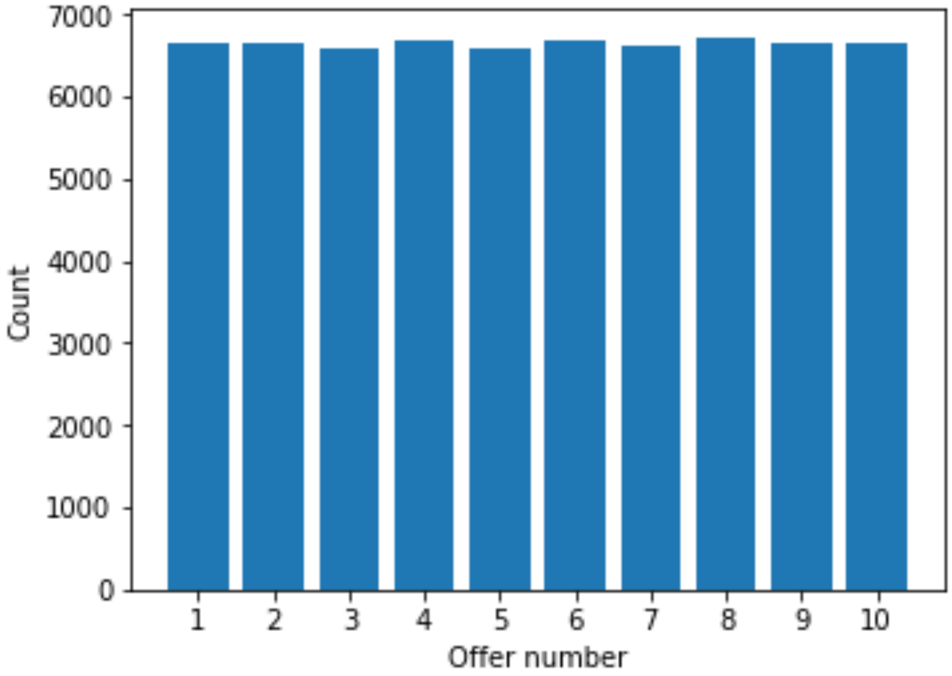
Model Evaluation and Validation

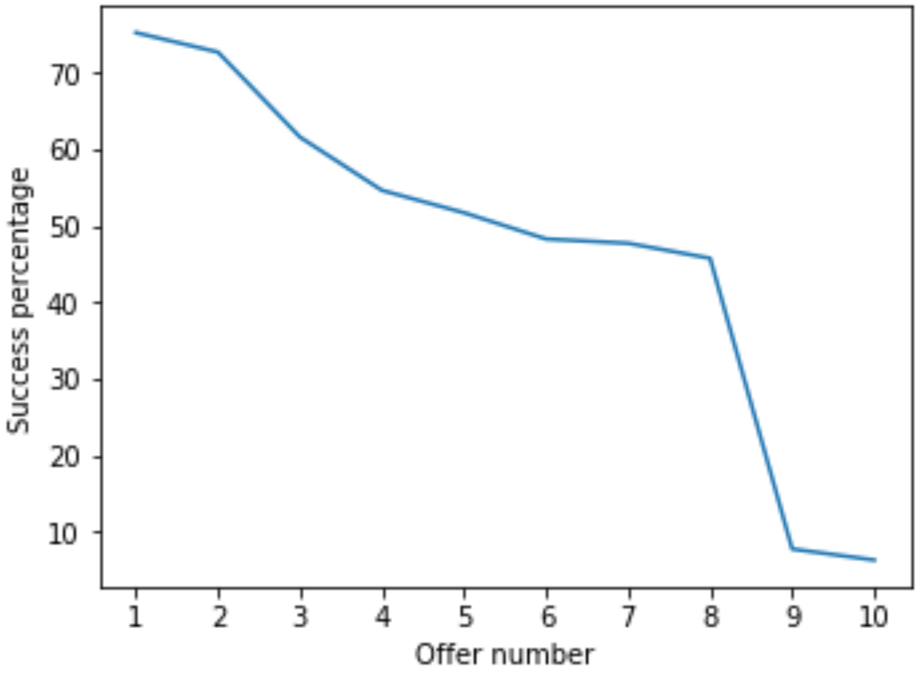
I have tested the model with 30% of the data which has not been used for training the model and got 99% accuracy .

Justification

As the model has 99% accuracy , if the model has been given the demographics of a person and an offer type it will give the correct output of whether a person will respond to an offer or not.

V. Conclusion Free-Form Visualisation





Reflection

Summary :

1) Analysing the given three data files and performing preprocessing steps required to form the data.
2) Splitting the data into training and test dataset.
2) Constructing the Machine Learning Model.

3) Training the model with the training data.
4) Testing the previously trained model with test data.

The difficult part me was to reconstruct the given data into a meaningful data which can be used to train the model.

Improvement

The one aspect that can be improved is we can be able to predict the time in which a user will respond to a particular offer.