Starbucks customer offer success predictor

Nikhil Tank | University of Florida

Udacity Nanodegree – Machine learning Engineer

I. Definition

1. Overview

Starbucks is an American coffeehouse, started as a single store in Seattle's Pike Place Market in 1971. From the start, the company only celebrated coffee and the rich tradition and brought the feeling of connection. Starbucks is ranked 227 among the Fortune 500 companies in 2019. The company has more than 31,000 stores across the world and this figure has almost got doubled in the last decade. They have an app as a convenient way to pay in-store or skip the line and order ahead. Offering rewards and stars earning free drinks and food with every purchase. The promotion offer can be an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free).

The purpose of an offer is to attract customers and wanted to spend more on what they like. Giving the right offer to the right customer effects a lot in building customer satisfaction and loyalty of the brand, resulting in improving sales. The goal of the project is to predict the influence of rewards and offer on customers based on their response to the previously sent offer. The project uses the data which 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. In the dataset three types of files are included:

- profile.json - demographic data for each customer like age, gender, and income.
- portfolio.json - containing offer ids and metadata about each offer, such as the duration and the amount a customer's spending required to qualify for an offer. These offers can be delivered via multiple channels and specific time validity of an offer.
- transcript.json - records for transactions, offers received, offers viewed, and offers completed.

The files were cleaned, missing nan data was replaced by the median of the feature. The values were normalized between the Min-Max. Further, the 3 datasets were merged into 1 dataset, such that it contained the demographic representation of customers, details of the order amount, and whether the offer was completed. The final dataset size was 76277 X 20, with features like age, gender, reward, difficulty, duration, offer type, channel, etc.
The below are head of the table with the column (feature) name and the data value.

	is_complete	age	became_member_on	income	gender_F	gender_M	gender_O	gender_nan	reward	difficulty	duration	offer_type_bogo	offer_
0	1	0.180723	0.747120	0.466667	0	1	0	0	0.0	0.0	0.0	0	
1	0	0.265060	0.891388	0.300000	0	0	1	0	0.0	0.0	0.0	0	
2	1	0.493976	0.520570	0.666667	1	0	0	0	0.0	0.0	0.0	0	
3	1	0.072289	0.658804	0.333333	1	0	0	0	0.0	0.0	0.0	0	
4	1	0.096386	0.780581	0.477778	1	0	0	0	0.0	0.0	0.0	0	

r_type_bogo	offer_type_discount	offer_type_informational	channel_mobile	channel_web	channel_social	channel_email	difficulty_duration	reward_difficulty
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0

2. Problem Statement

I choose to build a machine learning model that determines which demographic groups will respond best to which offer type and duration; whether or a customer is going to complete the assigned offer. This is a binary classification problem as the customer after getting the notification of offer, sees the offer will he finish the offer before offer expiration, i.e. Yes/No, this is binary classification:

My approach to the solution is as follows:

1. Pre-process the data for missing information and normalizing the data so that they can have equally weighted.
2. Training the XGBoost Model as the benchmark
3. Training PyTorch Neural Network model and tuning it hyperparameter to get better results.

For binary classification, we have options like Random Forest Classifier, Decision Tree Classifier, neural networks, support vector machines. In this project, I will be using the XGBoost model and NN using PyTorch as a solution model.

3. Metrics

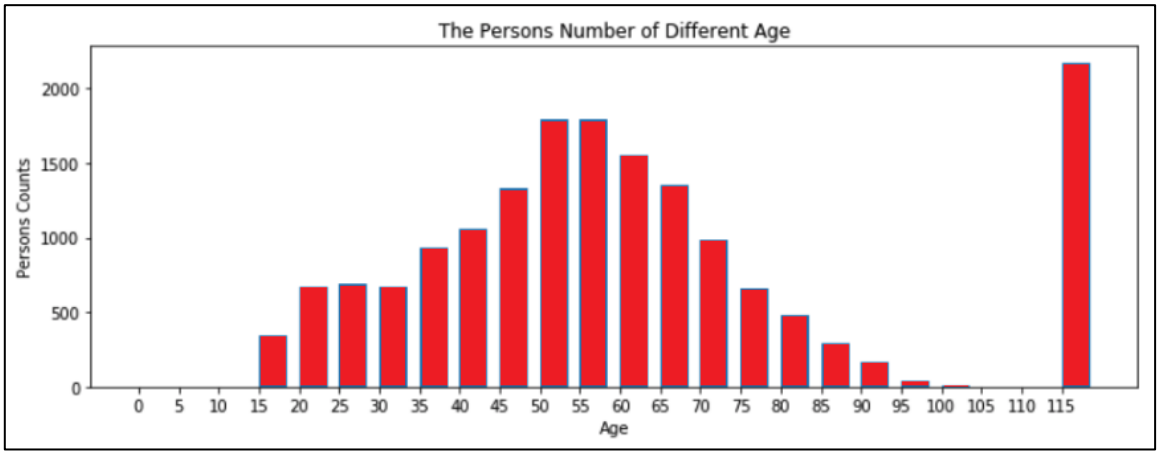
Receiver operation characteristic ROC-AUC will be used for model evaluation. The probabilistic interpretation of the ROC-AUC score is that if we randomly choose a positive case and a negative case, the probability that the positive case outranks the negative case according to the classifier is given by the AUC. ROC-AUC score is independent of the threshold set for classification because it only considers the rank of each prediction and not its absolute value. The same is not true for the F1 score which needs a threshold value in case of probabilities output. ROC-AUC is used because we don't want to tune the threshold in our model.

II. Analysis

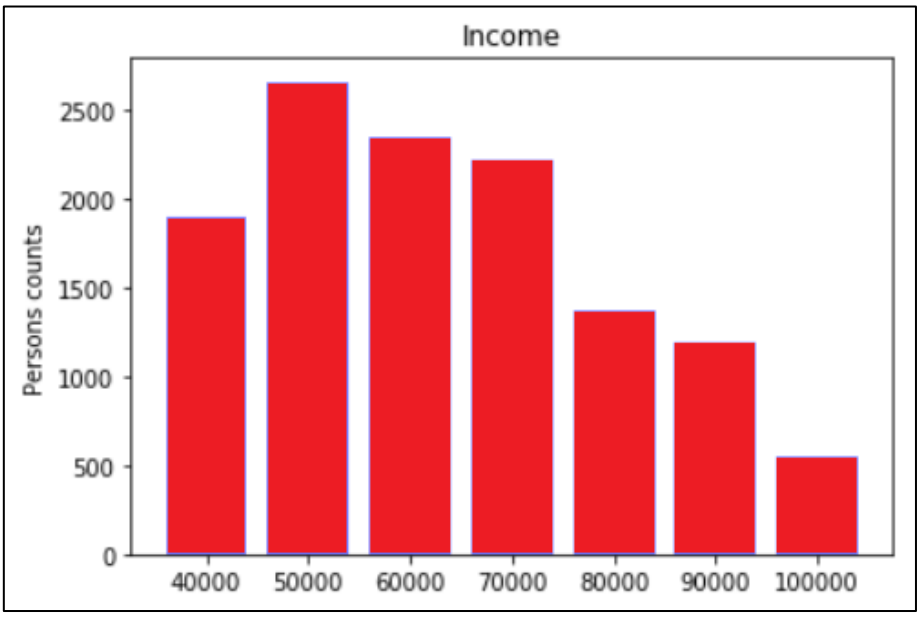
1. Data Visualization & Exploration

Profile Data

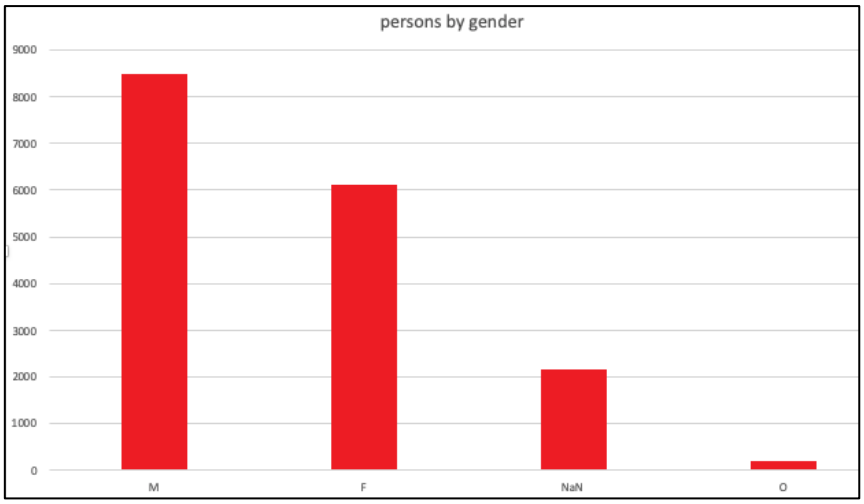
This data contains the information of the customer age, when they registered as member date, and income, gender. There are about 2000 nan for age data, to take care that the values are replaced with the age 118. Most of the age group lies in the 50 to 60 years of range. There are total 17000 user in this data set.



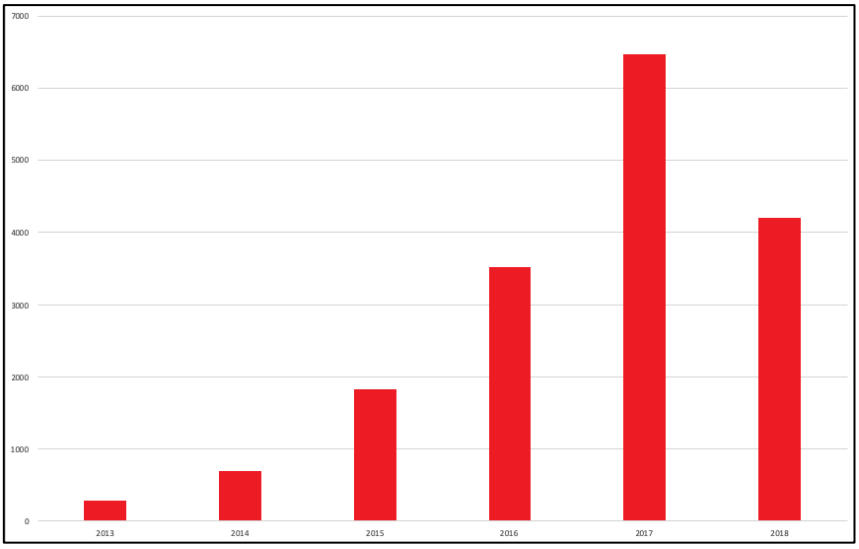
The graph below shows that the income range for the people is between 40,000 to 70,000.



The graph below shows that the majority of the customers are male.



The graph show that the no. of customers has joined the reward system in recent years, and the highest was 2017.



Portfolio

This dataset tells about the offer and the way they are distributed to them. There are three types of offers that can be sent:

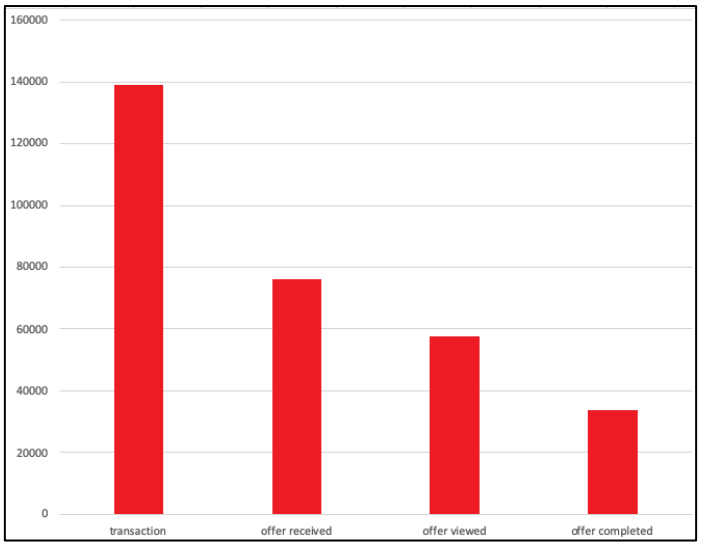
- Buy-one-get-one (BOGO): a user needs to spend a certain amount to get a reward equal to that threshold amount.
- Discount: a user gains a reward equal to a fraction of the amount spent.
- Informational: there is no reward, but neither is there a requisite amount that the user is expected to spend.

There are total 10 offers, 4 are BOGO, 4 are discount and 2 are information. All offers are distributed via email only 6 offers are in social media channel.

Transcript

For each transection there is event in the dataset. There are 76277 events without processing, and 306648 events after processing the transcript data.

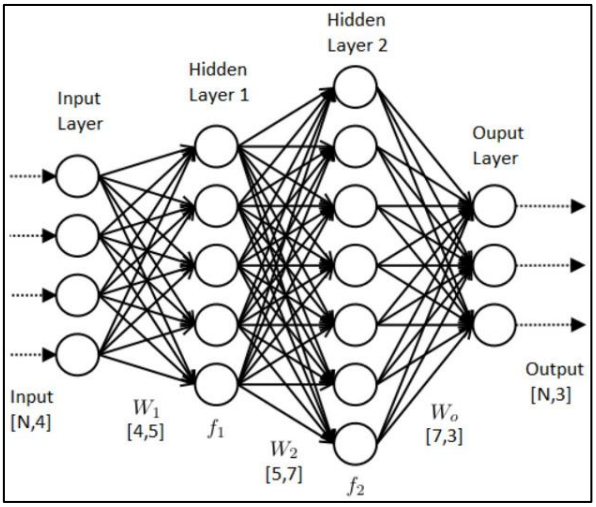
The are total 4 type of events: Transaction, offer received, offer viewed, offer completed. It is clearly visible that the no. decrease from received to completed. Only ½ of the offer received were completed and availed.



Algorithms and Techniques

I'm using a PyTorch neural network classifier model as a binary classifier for our solution. Neural networks are a series of algorithms that can recognize relationships in a dataset through a process that replicated the way the human neurons work. The neural network has a beautiful strength when come to learning from changing inputs. They "learn" to perform tasks by considering or supervised by the example or rewards system, generally without being programmed with any task-specific rules.

An example of a Neural network with 2 hidden layers is depicted below with 4 inputs and 3 output. For our case, I'm trying to change the hidden layer also know as a hidden dimension; and the output connected to sigma function to scale the output.



To avoid over-fitting, we are introducing a dropout function. With a certain probability few of the data point are going to dropout from our prediction. I'm using Adam Optimizer to optimize the result based on our resource constraints like time, computing power, memory, etc. Binary Cross Entropy is used to measure the loss.

2. Benchmark

I will be using XGBoost model as me benchmark. I have calculated the Accuracy: 0.7213, F1: 0.6820, ROC-AUC: 0.7160 with the following parameters:

- max-depth = 5
- eta = 0.2
- gamma = 4
- min_child_weight = 6
- objective = 'binary:logitics'
- subsample= 0.8

III. Methodology

1. Data Pre-processing

To make the dataset more consistent we need to process the dataset for better and sane results. The pre-processing is done in the *Starbucks_Capstone_notebook.ipynb* and script folder.

Profile Dataset

In this data set, there is the information of customer gender, this information can be encoded into different columns with Male/Female flag i.e. 1 or 0. Further, there are multiple NAN present in the dataset which makes it hard to interpret. The is missing customer income and age, we have to populate the data in a way that it doesn't much affect the distribution of the feature. For income, a good option is to use the median of the distribution. The NAN age can be replaced with a significantly large value of age (example age: 118), by this it will come as a noise for the data. The following processing has to be done for them: Encoding for gender in gender_M, gender_F

1) Replace the NAN value of income with the median income.
2) Replace the NAN value of age with age = 118 years.
3) Convert timestamp into Unix timestamp.
4) Normalize the income, age, became_member_on based on Min-max value of feature.

	age	id	became_member_on	income	gender_F	gender_M	gender_O	gender_nan
0	0.445783	68be06ca386d4c31939f3a4f0e3dd783	0.709819	0.377778	0	0	0	1
1	0.445783	0610b486422d4921ae7d2bf64640c50b	0.793747	0.911111	1	0	0	0
2	0.445783	38fe809add3b4fcf9315a9694bb96ff5	0.992320	0.377778	0	0	0	1
3	0.686747	78afa995795e4d85b5d9ceeca43f5fef	0.756994	0.777778	1	0	0	0
4	0.445783	a03223e636434f42ac4c3df47e8bac43	0.804717	0.377778	0	0	0	1

Portfolio Dataset

In this dataset there are 2 columns with composite attributes, so it's better idea to split it into Its different components. These 2 columns are offer_type and channels. The following processing has to be done for them:

1) Make different columns for different offer type
2) Make different columns for different channel like: mobile, web, social, email.
3) Create new two ratio features for *difficulty_duration* and *reward_difficulty.*
4) Normalize the values based on the min-max of the column values for rewards, difficulty, and duration.

The reward vs difficulty ratio shows how much customer gains by completing the offer, the more the better for customer. Ans the difficulty vs duration ratio shows how much the customer pays to avail an offer, the less the beneficial for them.

	reward	difficulty	duration	id	offer_type_bogo	offer_type_discount	offer_type_informational	channel_mobile	channel
0	1.0	0.50	0.571429	ae264e3637204a6fb9bb56bc8210ddfd	1	0	0	1	
1	1.0	0.50	0.285714	4d5c57ea9a6940dd891ad53e9dbe8da0	1	0	0	1	
2	0.0	0.00	0.142857	3f207df678b143eea3cee63160fa8bed	0	0	1	1	
3	0.5	0.25	0.571429	9b98b8c7a33c4b65b9aebfe6a799e6d9	1	0	0	1	
4	0.5	1.00	1.000000	0b1e1539f2cc45b7b9fa7c272da2e1d7	0	1	0	0	

r_type_bogo	offer_type_discount	offer_type_informational	channel_mobile	channel_web	channel_social	channel_email	difficulty_duration	reward_difficulty
1	0	0	1	0	1	1	1.428571	1.00
1	0	0	1	1	1	1	2.000000	1.00
0	0	1	1	1	0	1	0.000000	0.00
1	0	0	1	1	0	1	0.714286	1.00
0	1	0	0	1	0	1	2.000000	0.25

Transcript Dataset

Transcript dataset provide label for our experiment or model. The is dictionary data structure for column name value, and this can be broken down into two columns offer_id and amount.

1) To get information whether the offer was completed or not we need information of difficulty, duration and offer type we need to join the portfolio and transcript dataset.
2) Next, we have to group by person and offer and calculate amount spent, received time, view time, expected complete time and complete time.
3) The final column is label column, this will return the label 1 for complete while 0 for incomplete offer.

	person	offer_id	offer_type	difficulty	amount	receive_time	view_time	complete_time	expecte
0	0009655768c64bdeb2e877511632db8f	5a8bc65990b245e5a138643cd4eb9837	informational	0.0	22.16	168	192.0	NaN	
1	0009655768c64bdeb2e877511632db8f	3f207df678b143eea3cee63160fa8bed	informational	0.0	8.57	336	372.0	NaN	
2	0009655768c64bdeb2e877511632db8f	f19421c1d4aa40978ebb69ca19b0e20d	bogo	5.0	8.57	408	456.0	414.0	
3	0009655768c64bdeb2e877511632db8f	fafdcd668e3743c1bb461111dcafc2a4	discount	10.0	14.11	504	540.0	528.0	
4	0009655768c64bdeb2e877511632db8f	2906b810c7d4411798c6938adc9daaa5	discount	10.0	10.27	576	NaN	576.0	

_time	view_time	complete_time	expected_complete_time	is_in_expected_complete_time	is_enough_amount	is_view_event	is_complete_event	is_complete
168	192.0	NaN	240.0	False	True	True	False	True
336	372.0	NaN	432.0	False	True	True	False	True
408	456.0	414.0	528.0	True	True	True	True	True
504	540.0	528.0	744.0	True	True	True	True	True
576	NaN	576.0	744.0	True	True	False	True	False

Merging the Datasets

The final data processing step is to join all 3 data sets and removing unnecessary columns like person_id, offer_id in transcript. Finally, we have 76277 data points with 19 features and 1 label.

	is_complete	age	became_member_on	income	gender_F	gender_M	gender_O	gender_nan	reward	difficulty	duration	offer_type_bogo	offer_
0	1	0.180723	0.747120	0.466667	0	1	0	0	0.0	0.0	0.0	0	
1	0	0.265060	0.891388	0.300000	0	0	1	0	0.0	0.0	0.0	0	
2	1	0.493976	0.520570	0.666667	1	0	0	0	0.0	0.0	0.0	0	
3	1	0.072289	0.658804	0.333333	1	0	0	0	0.0	0.0	0.0	0	
4	1	0.096386	0.780581	0.477778	1	0	0	0	0.0	0.0	0.0	0	

r_type_bogo	offer_type_discount	offer_type_informational	channel_mobile	channel_web	channel_social	channel_email	difficulty_duration	reward_difficulty
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0
0	0	1	1	0	1	1	0.0	0.0

Split into the train and test Datasets

I wrote a function to split the data into train and test dataset, for that a variable decimal fraction is defined, current we choose to divide the data into 70% train data and 30 % test data.

- train_x.shape, train_y.shape : (53393, 19) (53393,)
- test_x.shape, test_y.shape : (22884, 19) (22884,)

2. Implementation

The step for the implementation of the project is as follows:

1) Loading the data into the Notebook and to S3: For deploying and testing the model, the training and testing data has to be uploaded to the amazon storage service S3.
2) Training the XGBoost model in SageMaker as a Benchmark
3) Training Deep Learning Model on SageMaker
    a. Define PyTorch Model: We are using PyTorch forward method with two-layer network and the output layer is sigmoid function which produce one-dimension label.
    b. Train.py: Since this a customized model a train program has to be written. In this we load training data, parse hyperparameter, and initialize the model, train the model and save the prediction
    c. Model.py: This acts as the entry point for the SageMaker estimator
    d. Define and Train the estimator with the train data from S3. In this step we provide hyperparameter, hidden dimension
    e. Predict.py: This acts as entry point and deploys the trained model
4) Evaluating train model using the Sklearn methods: Accuracy [6], F1[7], ROC-AUC[8] metric for different hyperparameter configuration
5) The different Hyperparameter can be tuned for the PyTorch model for improving the performance with respect to the benchmark.
6) Deleting the endpoints and cleaning the resources.

3. Refinement

For refining the model I'm tuning the hyperparameters like hidden dim, epochs, drop out. Changing the hidden dimension resulted in different metric values for Accuracy, F1, and ROC-AUC. By increasing the hidden dimension, it is evident that the loss value is reducing with increasing hidden dimensions as it might be covering more dimensions of the data, which are not clearly visible to naked eyes. Epochs is also an important parameter. It controls how many iterations go in to improve loss function. Higher epochs are leading a better score. We should note that drop out is affecting overfitting so without dropout the results seem better but that is the quite overfitted model.

Hidden dim	*Epochs*	*Drop out*	*Loss*	*Accuracy*	*F1*	*ROC-AUC*
30	100	0.25	0.5623	0.7213	0.6708	0.6766
15	100	0.25	0.5691	0.6709	0.6795	0.6783
15	**200**	**0.25**	**0.5690**	**0.6748**	**0.6808**	**0.6814**
100	100	0.25	0.5539	0.6580	0.6716	0.6665
180	200	0.15	0.5500	0.5921	0.6479	0.6102
100	100	0	0.5539	0.6580	0.6716	0.6665

IV. Results

1. Model Evaluation and Validation

From the refinement step it is clear that the we are getting almost similar results with respect to our benchmark. From XGBoost model we have recorded **Accuracy: 0.7213, F1: 0.6820, ROC-AUC: 0.7160** for the final model we have taken the hyperparameter as follows: Hidden dim: 15, Epochs: 200, Drop out: 0.25

From this I'm getting good **Accuracy: 0.6748** and **ROC-AUC: 0.6814**. The loss value is also good overall.

2. Justification

The final model is giving similar results when tuned as compared to our benchmark XGBoost model. The improvement is not much but the results quite good and with less loss and ang good ROC-AUC score. The model is ready to go for predicting future test conditions.

V. References

[1] https://pytorch.org/

[2] https://aws.amazon.com/sagemaker/

[3] https://pytorch.org/docs/stable/optim.html

[4] https://pytorch.org/docs/master/generated/torch.nn.Dropout.html

[5] https://pytorch.org/docs/master/generated/torch.nn.BCELoss.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

[7] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[8] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html

[9] https://towardsdatascience.com/hyperparameter-tuning-c5619e7e6624

[10] https://seaborn.pydata.org/