Binary Recommendation System with Neural Networks

Soorya Suresh

Stevens Institute of Technology

CS 583

Dataset Descriptions

Real Dataset: This dataset came attached to the project where it shows user interaction data. Meaning, each row in the CSV file represents a user and each column denotes a specific feature. The cells are either 1 or 0. 1 indicating the user has interacted with that feature and 0 indicating the user has not interacted with that feature. The dataset has 610 rows with 70 features (71 columns) from feature_0 to feature_70. The goal of the model is to predict a new user's feature_0 value. For training and testing purposes, the dataset was divided into training and testing subsets using an 80/20 split with the target variable being feature_0.

```
(610, 70)
(610,)
Training set shape: (488, 70)
Testing set shape: (488,)
   Unnamed: 0  feature_0  feature_1  feature_2  feature_3  feature_4  \
0           0          1        0.0        0.0        1.0        0.0
1           1          0        0.0        0.0        1.0        0.0
2           2          1        0.0        0.0        1.0        0.0
3           3          1        0.0        0.0        1.0        0.0
4           4          0        0.0        0.0        1.0        0.0

   feature_5  feature_6  feature_7  feature_8  ...  feature_61  feature_62
\
0        0.0        1.0        0.0        1.0  ...         1.0         0.0
1        0.0        1.0        0.0        1.0  ...         1.0         0.0
2        0.0        1.0        0.0        1.0  ...         1.0         0.0
3        0.0        1.0        0.0        1.0  ...         1.0         0.0
4        0.0        1.0        0.0        1.0  ...         1.0         0.0

   feature_63  feature_64  feature_65  feature_66  feature_67  feature_68
\
0         1.0         0.0         0.0         0.0         0.0         0.0
1         1.0         0.0         0.0         0.0         0.0         0.0
2         1.0         0.0         0.0         0.0         0.0         0.0
3         1.0         0.0         0.0         0.0         0.0         0.0
4         1.0         0.0         0.0         0.0         0.0         0.0

   feature_69  feature_70
0         0.0         1.0
1         0.0         1.0
2         0.0         1.0
3         0.0         1.0
4         0.0         1.0

[5 rows x 72 columns]
```

Randomly Generated Dataset: This dataset reflected the same format as the real dataset but values were created randomly. It was done by utilizing NumPy where it first generates a random dataset of 0 or 1's with the dimensions of 610 rows with 71 columns. However, an index column or 'Unamed: 0' column was not created in the random data. For training and testing purposes, the dataset was divided into training and testing subsets using an 80/20 split with the target variable being column '0'.

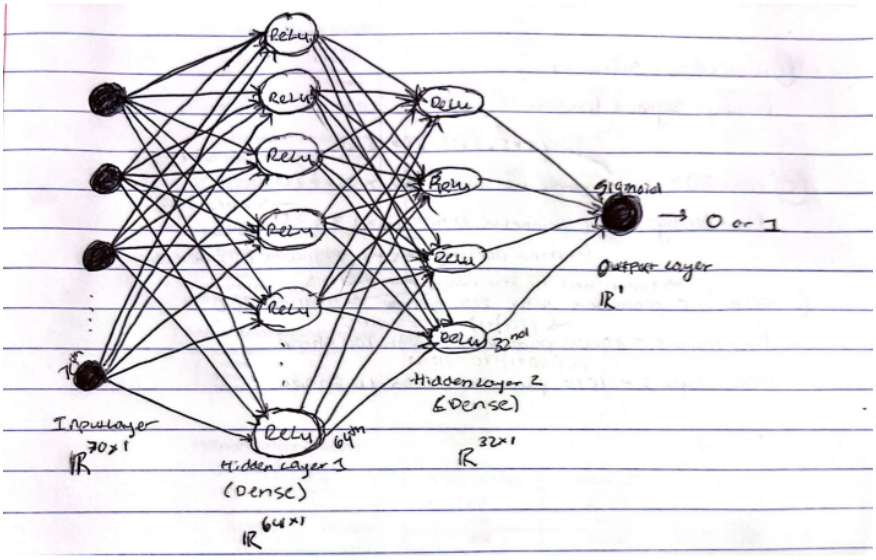
```
(610, 70)
(610,)
(610, 71)
Training set shape: (488, 70)
Testing set shape: (488,)
   0   1   2   3   4   5   6   7   8   9   ...  61  62  63  64  65  66  67
\
0   1   0   0   1   1   0   1   0   0   0 ...   0   0   0   0   1   0   1
1   1   0   1   1   1   1   0   1   0   1 ...   1   0   0   1   1   1   1
2   0   0   0   0   1   0   1   1   1   0 ...   0   0   0   1   0   1   1
3   0   0   0   0   0   1   0   1   0   1 ...   1   0   1   0   1   1   1
4   0   0   1   1   0   0   1   1   1   1 ...   1   1   1   1   1   0   1

   68  69  70
0   1   0   0
1   0   0   0
2   0   0   0
3   1   0   1
4   1   0   1

[5 rows x 71 columns]
```

Neural Network Model Architecture

The architecture of the neural network model used here follows a shallow MLP format.



The typical neural network model structure includes an input layer, hidden layer and output layer. In this case the input layer is defined by the number of features in the training data (X_train.shape[1]=70). The first hidden layer or Dense layer with activation Relu has 64 neurons. The second hidden layer or dense layer with activation Relu has 32 neurons. The output layer uses sigmoid activation due to the binary classification properties of the dataset. Therefore, this neural network follows an MLP architecture.

Layer (type)	Output Shape	Param #
dense_1524 (Dense)	(None, 64)	4544
dense_1525 (Dense)	(None, 32)	2080
dense_1526 (Dense)	(None, 1)	33

Total params: 6,657
Trainable params: 6,657
Non-trainable params: 0

Training Methodology and Hyperparameters

The hyperparameters in the model architecture were decided based on a GridSearchCV which was built with the help from DigitalSreeni's video on 'Tuning deep learning hyperparameters using GridSearchCV'. This GridSearchCV runs through different learning rates and neurons per layer value and returns the learning rate and neurons per layer value with the best score. It went through neurons per layer of 32, 64 and 128 and learning rate values of 0.0001, 0.001 and 0.01. As a result, the optimal neurons for first layer was 64 and learning rate of 0.01.

```
Best: 0.536822 using {'learning_rate': 0.01, 'neurons_per_layer': 64}
```

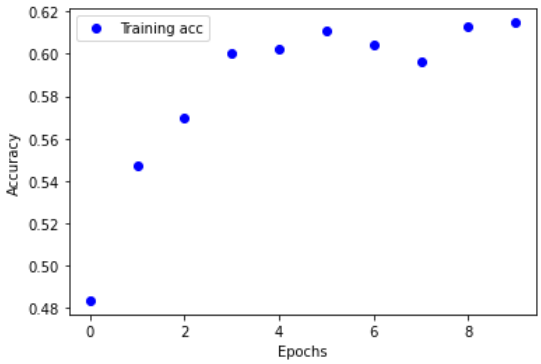
The hyperparameters in model architecture were discussed in the neural network architecture above. The input layer represents the input data hence the shape being the number of features in the training data. The first hidden layer contains 64 neurons and second hidden layer contains 32 neurons - both with ReLu activation. These hidden layers contain neurons that are in between input and output layers to process the input data by applying non linear functions (ex: ReLu activation) and generates an output for the next

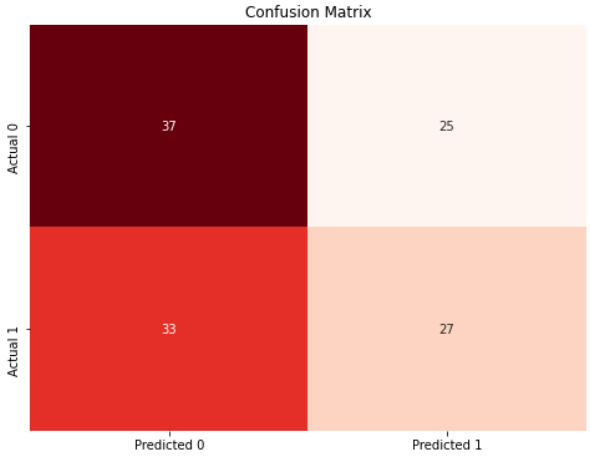
layer. The output layer has a single neuron and utilizes a sigmoid function for binary classification, thus this layer produces the final results which can be a 0 or 1.
The hyper parameters in model compilation consist of learning rate, optimizer and loss function. The learning rate is set to 0.01 as it specifies the size of the gradient descent steps. The Adam optimizer determines how the models weights, which are applied (visulized via lines in between layers) from one layer to the next, are updated during training. Finally, the loss function is used to compute the difference between predicted values and actual values to get accuracy score. Since, this is a binary classification the loss function was set to binary cross entropy.

Evaluation metrics and their interpretation

Real Dataset: After training the model and running through the test, the accuracy of this model on this dataset yielded a 52.46% with the below precision, recall, F1-scores, training accuracy progression and confusion matrix.

	precision	recall	f1-score	support
0	0.53	0.60	0.56	62
1	0.52	0.45	0.48	60
accuracy			0.52	122
macro avg	0.52	0.52	0.52	122
weighted avg	0.52	0.52	0.52	122

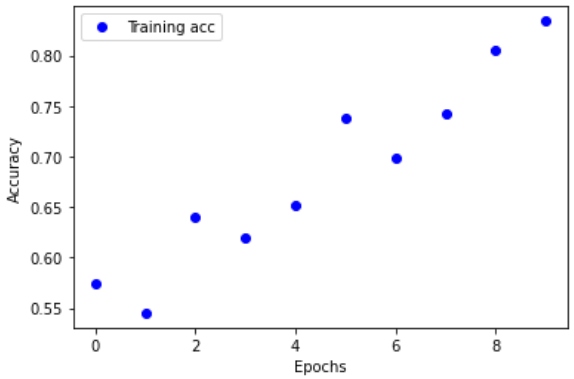


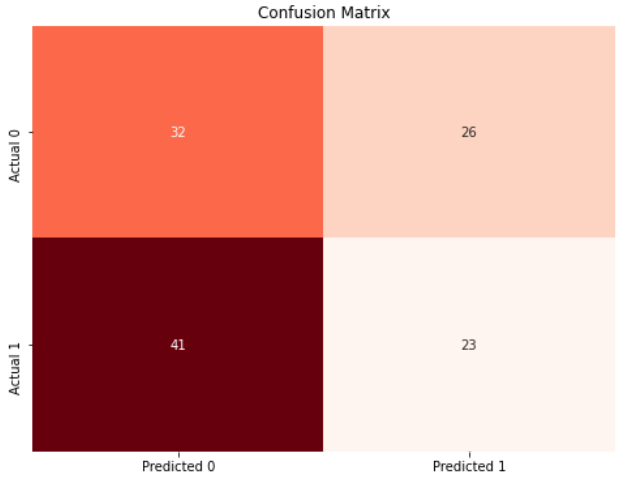


The precision values measures that out of all instances predicted as ‘0’ or ‘1’, the percentage that are actually ‘0’ or ‘1’, respectively. The precision for both 0 and 1 are above 50% with precision for 0 a bit higher. The recall value measure that out of all actual ‘0’ or ‘1’ instances, the percentage that were accurately identified as ‘0’ or ‘1’, respectively. The recall for 0 is much higher than 1. Finally, F1 score represents the balance between precision and recall by taking into both values into its formula. In this case, the precision, recall and F1 scores are larger for value ‘0’ in comparison to ‘1’ which may indicate a class imbalance or bias towards 0.

Randomly Generated Dataset:After training the model and running through the test, the accuracy of this model on this dataset yielded a 45.08% with the below precision, recall and F1-scores.

	precision	recall	f1-score	support
0	0.44	0.55	0.49	58
1	0.47	0.36	0.41	64
accuracy			0.45	122
macro avg	0.45	0.46	0.45	122
weighted avg	0.45	0.45	0.45	122





The precision values measures that out of all instances predicted as ‘0’ or ‘1’, the percentage that are actually ‘0’ or ‘1’, respectively. The precision for 0 is a bit higher. The recall value measure that out of all actual ‘0’ or ‘1’ instances, the percentage that were accurately identified as ‘0’ or ‘1’, respectively. The recall for 0 is much higher than 1. Finally, F1 score represents the balance between precision and recall by taking into both values into its formula. In this case, the precision, recall and F1 scores are larger for value ‘0’ in comparison to ‘1’ which may indicate a class imbalance or bias towards 0.

Comparison and Analysis of the model’s performance on Real Data vs. Randomly Generated Data

Both models shows performance with different accuracy ratings and had class ‘0’ with higher recall, precision and F1 scores. However, the model trained on real data outperforms the model trained on random data by approximately 7 percentage points, indicating that it has learned more from the real data in comparison to the random data. Additionally, precision, recall, and F1-score for both classes are mainly higher in the model trained on real data compared to random data. This showcases the model's capability is more maxed out on the real data than the random data.

The contrast in performance between the real data vs randomly generated data shows the importance of learning from real data for a machine learning algorithm. The only difference between the real data vs. randomly generated data is that the real data encompasses user interface that must have meaningful patterns while, the random data does not. The model architecture, compilation and execution were the same for both models. Yet the neural network trained on the real dataset returned higher accuracy rates than that of randomly generated data. This is because the neural network, just like any other machine learning algorithm, has the ability to generalize and identify patterns of the dataset through its hyperparameters and architecture to return more accurate predictions. Thus, the lack of meaningful patterns make it hard for a neural network to learn the dataset and return accurate predictions.