Assignment-2
Prediction of High Risk Cancer Patients

Megha Mathur (MT19104) , Abhishek Vyas (MT19086) , Chirag Chawla (MT19089)

Problem Statement : We are given the dataset consisting of "ID", 318 features along with the labels. Our task is to classify the cancer patients into high risk and low risk using deep learning algorithms.

Features Selection Techniques used :-

1. **Variance Threshold :-** It is a filter feature selection technique, which basically removes particular below threshold variance features from the feature vector. We have taken the threshold value as 1.

2. **L1 Regularization :** L1 regularization is the embedded feature selection method SVM is trained with an argument penalty as l1.

Validation Set:-

We have splitted our dataset into train & test in the ratio of 80:20. So we have used the sklearn "train_test_split" method which splits the dataset with some random state value.

K-Fold Validation:-

- We have used KFold using sklearn.model_selection for validating our result. For CNN implementation we have used 5 folds.
- We have used the cross_val_score of sklearn.model_selection for a multilayer perceptron model.

Evaluation Metric Used:-

Matthews correlation coefficient (MCC) is used as an evaluation metric for this assignment.

Deep Learning Techniques Used :

Various deep learning algorithms used to classify the data sample into high risk and low risk are as follows:-

- **Convolution Neural Network(CNN)**
    1. Initially train data is read. Then splitted into x and y.
    2. Then padding of 0's is done to make data suitable for CNN model (new shape of data is 242,324)
    3. Then data is reshaped using np.reshape function that is updated shape is (242,18,18,1).
    4. Then using train_test_split, data is splitted into 80% train and 20% validation data.

5. Then keras.sequential model is called and 3 layers are incorporated, i.e. Conv2d, flatten and Dense layers.
6. Conv2D layer takes filter size as 250 and kernel size as (3,3) with input_shape as 18*18*1.
7. Dense layer has activation function as sigmoid and output dimension as 1.
8. Then the model is compiled with optimizer='adam', loss='binary_crossentropy', metrics=[mcc]. Here mcc metric is taken from tensorflow addons.
9. Then the model is fitted using training data and predicts validation data labels and test data labels.
10. Then the submission file is created in the defined format.

- **Artificial Neural Network_without padding**
    1. Initially train data is read. Then splitted into x and y.
    2. Then data is converted into a numpy array.
    3. Then using train_test_split, data is splitted into 80% train and 20% validation data.
    4. Then keras.sequential model is called and 1 layer i.e. keras.layers.Dense(1, input_shape=(x_train_up.shape[1],), activation='sigmoid')
    5. Then the model is compiled with optimizer='adam', loss='binary_crossentropy', metrics=[mcc]. Here mcc metric is taken from tensorflow addons.
    6. Then the model is fitted using training data and predicts validation data labels and test data labels.
    7. Then the submission file is created in the defined format.

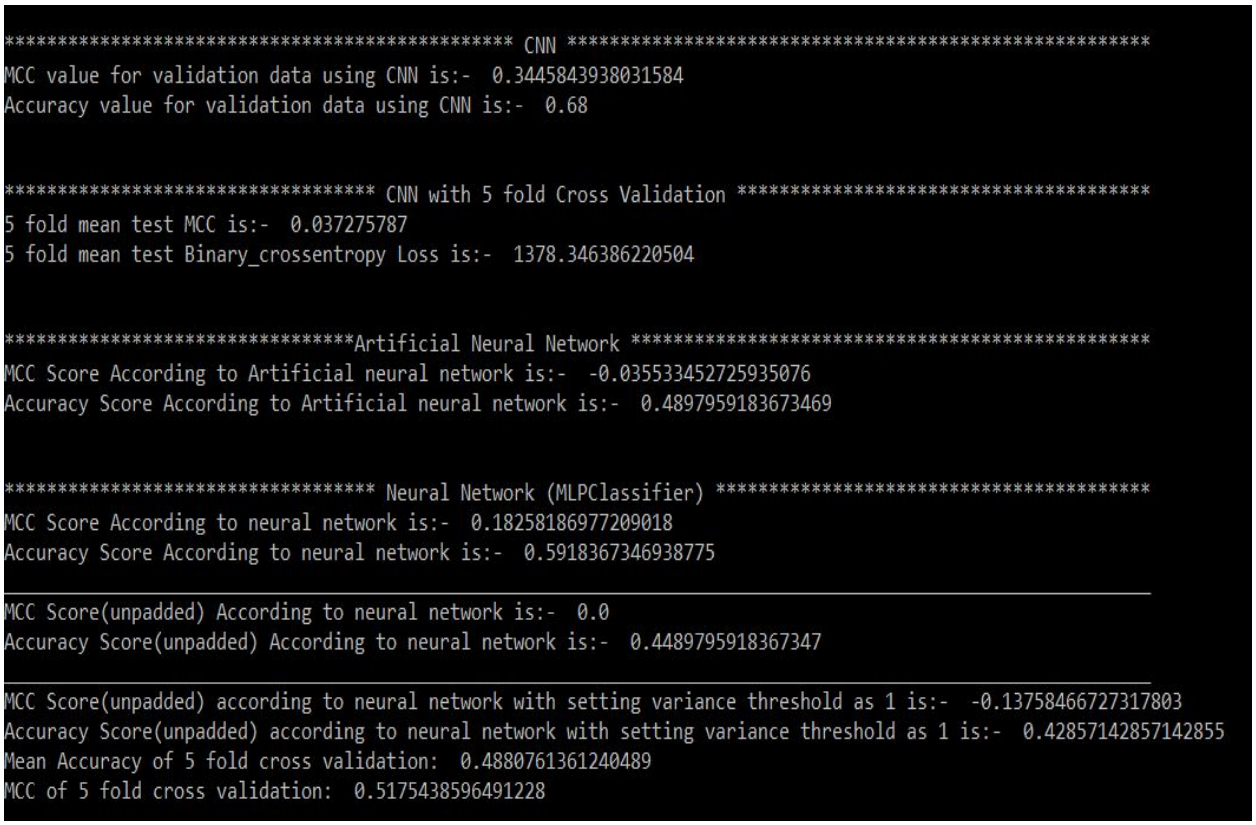
- **Multilayer Perceptron Model with padding**
    1. Initially train data is read. Then splitted into x and y.
    2. Then padding of 0's is done to make data a 4 dimensional. New shape of data is 242,324)
    3. Then data is reshaped using np.reshape function that is updated shape is (242,18,18,1).
    4. Then using train_test_split, data is splitted into 80% train and 20% validation data.
    5. Then sklearn MLPClassifier is called with default hyperparameters.
    6. Then the model is fitted using training data and predicts the validation data labels and test data labels.
    7. Then the submission file is created in the defined format.

- **Multilayer Perceptron Model without padding**
    1. Initially train data is read. Then splitted into x and y.
    2. Then data is converted into a numpy array.
    3. Then using train_test_split, data is splitted into 80% train and 20% validation data.
    4. Then sklearn MLPClassifier is called with (alpha=0.1, hidden_layer_sizes=10, max_iter=15, random_state= 9, solver='lbfgs') hyperparameters.

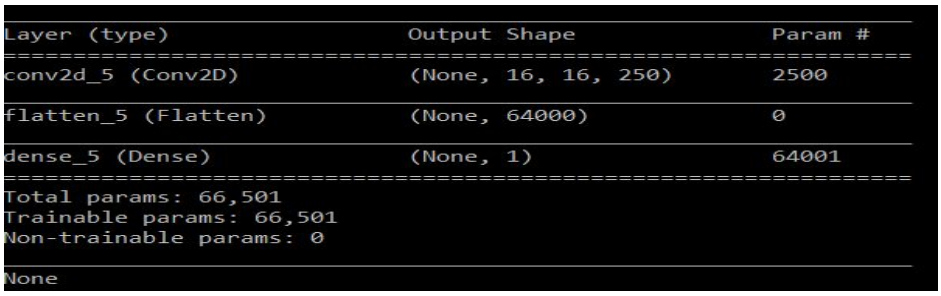
5. Then the model is fitted using training data and predicts the validation data labels and test data labels.
6. Then the submission file is created in the defined format.

- **Multilayer Perceptron Model without padding+feature_selection**
    1. Initially train data is read. Then splitted into x and y.
    2. Then data is converted into a numpy array.
    3. Then using train_test_split, data is splitted into 80% train and 20% validation data.
    4. Then using VarianceThreshold as feature selection technique with threshold value is 1, features are selected for both train and test.
    5. Then the sklearn MLPClassifier is called with (random_state=1, max_iter=300) hyperparameters.
    6. Then the model is fitted using training data and predicts the validation data labels and test data labels.
    7. Then the submission file is created in the defined format.

Output



Best Model CNN Summary (Our best 3 models are from cnn only)



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 16, 16, 250)       2500
_________________________________________________________________
flatten_5 (Flatten)          (None, 64000)             0
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 64001
=================================================================
Total params: 66,501
Trainable params: 66,501
Non-trainable params: 0
_________________________________________________________________
None
```

HOW TO RUN THE CODE:-

- Unzip the folder provided.
- Run the following command after running cmd on folder location to run the code:-
  python group.py -tr (kaggle_train.csv file path) -te (kaggle_test.csv file path) -o (output file path)
  For example :-

```
>python group30.py -tr "D:\MLBA_Assignments\hrpred\kaggle_train.csv" -te "D:\MLBA_Assignments\hrpred\kaggle_test.csv" -o "D:\MLBA_Assignments\hrpred\output.csv"
```

- Our Best model output file will be generated with the same name as provided by you and the rest of the models output files are generated with numbers appended to it.
  For example :-
  output.csv => cnn model (Best Model)
  output1.csv =>Artificial neural network
  output2.csv =>Neural network (Multilayer perceptron) padded model
  output3.csv =>Neural network (Multilayer perceptron) unpadded model
  output4.csv =>Neural network (Multilayer perceptron) unpadded model (variance threshold as 1)

Best 3 Kaggle submission Files

- A2_submission_7.csv (Kaggle Score=0.30675)
- A2_submission_8.csv (Kaggle Score=0.35180)
- A2_submission_11.csv (Kaggle Score=0.29960)