Kavya Shah SUID: 330126500 PIN: 68

Introduction to ML HW-08

Problem Statement:

The problem statement aims to apply an AdaBoost for classification on citation data spanning from cit_2017 to cit_2022.
The dataset is split into two parts: 80% for training and 20% for testing.

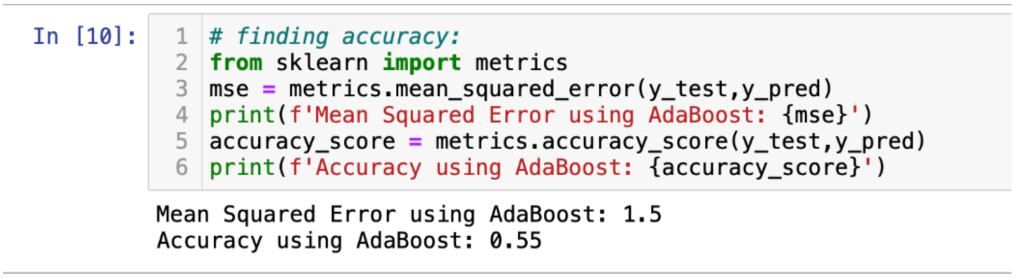
Approach:

1. Imported the important and useful libraries.
2. Loaded the data set into the python notebook using pandas.
3. Explored and analysed the data.
4. Splitted the dataset into target and features in 80 - 20 ratio.
5. Normalised the dataset using MinMaxScaler.
6. Applied AdaBoost for classification using AdaBoostClassifier library from sklearn.
7. I used test data to predict the correct category of test dataset.
8. Evaluation of Results:

- I evaluated the model's performance using the remaining 20% of the test data.
    1. Mean Square Error: To find the mean error between the predicted and the actual values.
    2. Accuracy: To find how correct model predictions are.
- The evaluation metrics provided insights into how well the model has learned to classify individuals based on the data set and citation ratios.

Result:

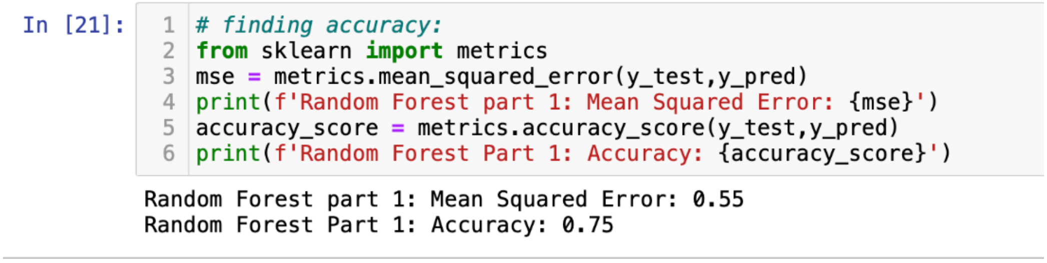
Hw-08 AdaBoost o/p:



```
In [10]:
# finding accuracy:
from sklearn import metrics
mse = metrics.mean_squared_error(y_test,y_pred)
print(f'Mean Squared Error using AdaBoost: {mse}')
accuracy_score = metrics.accuracy_score(y_test,y_pred)
print(f'Accuracy using AdaBoost: {accuracy_score}')
```

```
Mean Squared Error using AdaBoost: 1.5
Accuracy using AdaBoost: 0.55
```

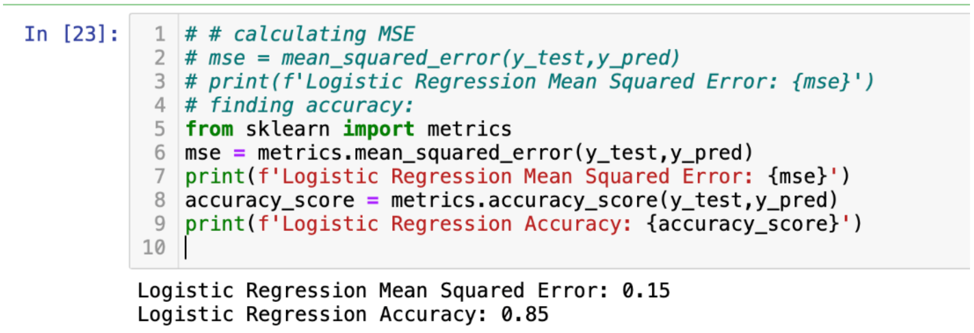
Hw-07 Random Forest Part 1 o/p:



```
In [21]:
# finding accuracy:
from sklearn import metrics
mse = metrics.mean_squared_error(y_test,y_pred)
print(f'Random Forest part 1: Mean Squared Error: {mse}')
accuracy_score = metrics.accuracy_score(y_test,y_pred)
print(f'Random Forest Part 1: Accuracy: {accuracy_score}')
```

```
Random Forest part 1: Mean Squared Error: 0.55
Random Forest Part 1: Accuracy: 0.75
```

Hw-06 Logistic Regression o/p:

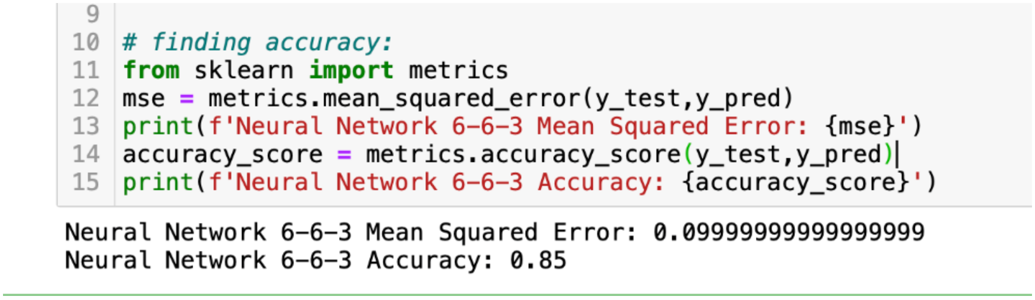


```
In [23]:
# # calculating MSE
# mse = mean_squared_error(y_test,y_pred)
# print(f'Logistic Regression Mean Squared Error: {mse}')
# finding accuracy:
from sklearn import metrics
mse = metrics.mean_squared_error(y_test,y_pred)
print(f'Logistic Regression Mean Squared Error: {mse}')
accuracy_score = metrics.accuracy_score(y_test,y_pred)
print(f'Logistic Regression Accuracy: {accuracy_score}')

```

```
Logistic Regression Mean Squared Error: 0.15
Logistic Regression Accuracy: 0.85
```

Hw-05 Neural Network 6-6-5 o/p:



```

# finding accuracy:
from sklearn import metrics
mse = metrics.mean_squared_error(y_test,y_pred)
print(f'Neural Network 6-6-3 Mean Squared Error: {mse}')
accuracy_score = metrics.accuracy_score(y_test,y_pred)
print(f'Neural Network 6-6-3 Accuracy: {accuracy_score}')
```

```
Neural Network 6-6-3 Mean Squared Error: 0.09999999999999999
Neural Network 6-6-3 Accuracy: 0.85
```

Conclusion:

Comparison:

	MSE	**Accuracy**
HW 8 Adaboost	1.5	0.55
HW 7 Random Forest Part 1	0.55	0.75
HW 6 Logistic Regression	0.15	0.85
HW 5 Neural Network 6-6-3	0.09	0.85

The MSE indicates, on average, the squared difference between the predicted and actual values.

Comments:

Adaboost HW-08: *MSE 1.5 and Accuracy 0.55*
In my code, it seems that Adaboost is struggling to fit the data well. The high MSE and low accuracy suggest that the model is not making accurate predictions, possibly due to the complexity of the data or the inadequacy of the weak learners.

Random Forest HW-07: *MSE 0.55 and Accuracy 0.75*
The lower MSE and higher accuracy in Random Forest Approach suggest that it is capturing more patterns in the data comparatively. The randomness introduced by multiple trees are contributing to better generalisation.

Logistic Regression HW-06: *MSE 0.15 and Accuracy 0.85*

The Lower MSE and Higher Accuracy indicate that the logistic regression model is fitting the data relatively well.

Neural Network HW-05: *MSE 0.09 and Accuracy 0.85*
The Low MSE and Higher Accuracy indicates that the neural network is performing well on the classification task. The network's ability to learn intricate patterns in the data is high.

Thus, According to my code I conclude that,
Accuracy of models for classification from worst to best is:
AdaBoost (worst) < Random Forest < Logistic Regression <= Neural Network

In summary, Adaboost performance depends on the stumps, the characteristics of the data and the potential presence of outliers or noise.
Again, performance is subject to the dataset and the parameters set in it.