Nam Nguyen – npn190000

Link:
https://colab.research.google.com/drive/1ik4DYMFV7EnfoOz74CKtfKWHPUX6uoFG#scrollTo=OEN55SDcN3-J

Exercise 1

Construct the following models on the same dataset:

Bagging

Random Forest

Adaboost

Compare their performance and write a short paragraph on which one is the best. You are free to change the hyperparameters.

Answer:

```
# Create Bagging Classifier
clf = BaggingClassifier( n_estimators=10, random_state=0)
```

```
              precision    recall  f1-score   support

     chinese       0.76      0.71      0.73       245
      indian       0.85      0.90      0.88       239
    japanese       0.78      0.75      0.76       245
      korean       0.81      0.76      0.78       233
        thai       0.75      0.83      0.79       237

    accuracy                           0.79      1199
   macro avg       0.79      0.79      0.79      1199
weighted avg       0.79      0.79      0.79      1199
```

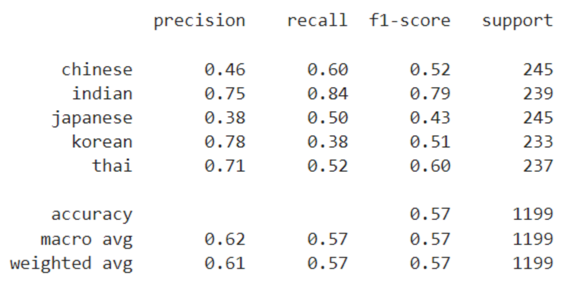
```
Accuracy:  0.7881567973311092
```

```
# Create RandomForest Classifier
clf = RandomForestClassifier(max_depth=3, random_state=0, n_estimators = 1
0)
```

```
              precision    recall  f1-score   support

     chinese       0.65      0.25      0.36       245
      indian       0.78      0.74      0.76       239
    japanese       0.46      0.81      0.59       245
      korean       0.57      0.72      0.64       233
        thai       0.81      0.54      0.65       237

    accuracy                           0.61      1199
   macro avg       0.65      0.61      0.60      1199
weighted avg       0.65      0.61      0.60      1199
```

```
Accuracy:  0.6105087572977481
```

```
# Create Adaboost Classifier
clf = AdaBoostClassifier(n_estimators=10, random_state=0)
```



```
              precision    recall  f1-score   support

     chinese       0.46      0.60      0.52       245
      indian       0.75      0.84      0.79       239
    japanese       0.38      0.50      0.43       245
      korean       0.78      0.38      0.51       233
        thai       0.71      0.52      0.60       237

    accuracy                           0.57      1199
   macro avg       0.62      0.57      0.57      1199
weighted avg       0.61      0.57      0.57      1199
```

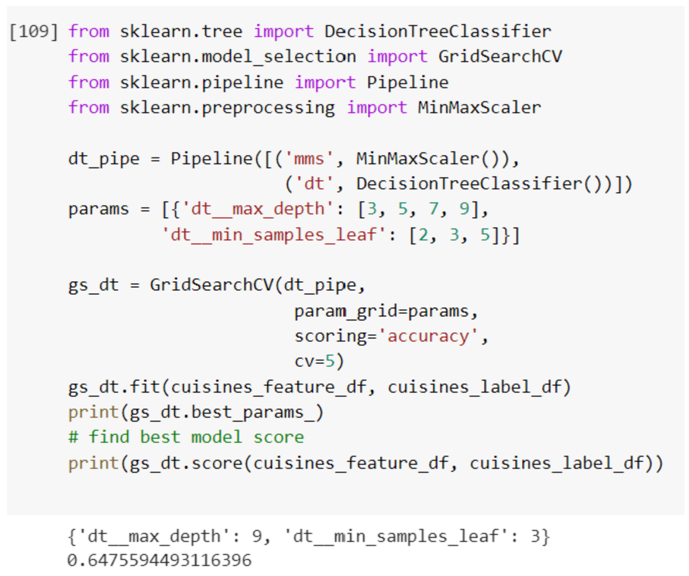
```
Accuracy:  0.5696413678065054
```

After training and testing the models, the Bagging model outperforms both the Random Forest and Ada boost models. However, It also depends how to optimal the hyperparameters to create a best version for each algorithm. The bagging model is a simple ensemble technique that relies on multiple models to make predictions. The bagging model is less likely to overfit the data than a single model. The random Forest model and The Ada boost model are more sophisticated ensemble technique that relies on multiple models to make predictions

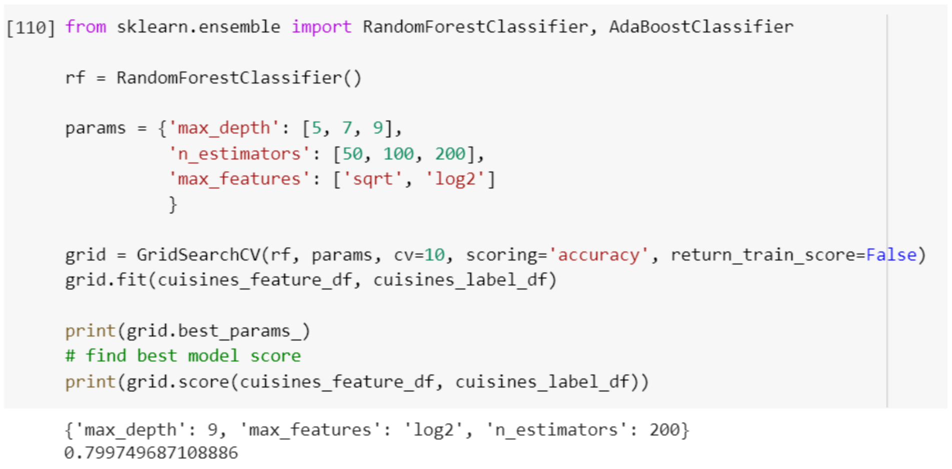
Exercise 2

The accuracy for this dataset is quite low. Can you try any other method that increases the accuracy. You can try either Random Forest or Adaboost. What do you notice?

Answer:

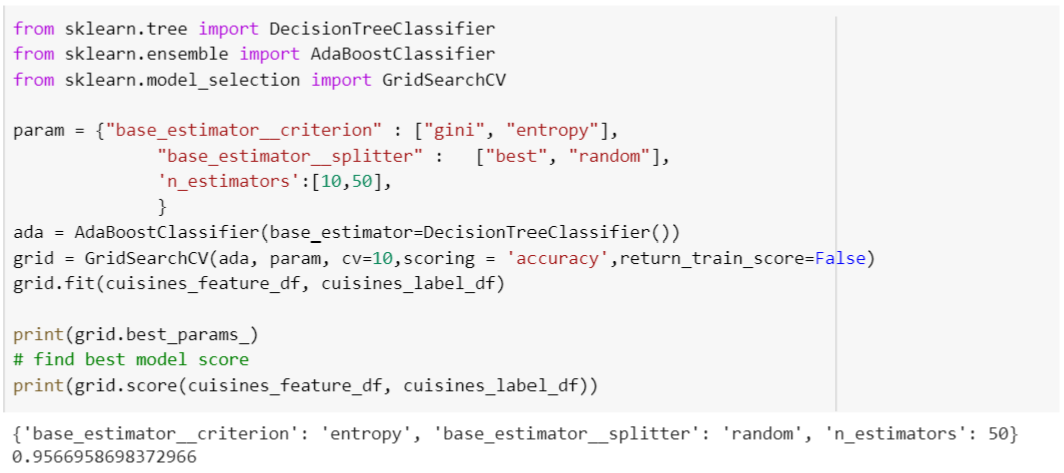


```
[109] from sklearn.tree import DecisionTreeClassifier
      from sklearn.model_selection import GridSearchCV
      from sklearn.pipeline import Pipeline
      from sklearn.preprocessing import MinMaxScaler

      dt_pipe = Pipeline([('mms', MinMaxScaler()),
                          ('dt', DecisionTreeClassifier())])
      params = [{'dt__max_depth': [3, 5, 7, 9],
               'dt__min_samples_leaf': [2, 3, 5]}]

      gs_dt = GridSearchCV(dt_pipe,
                           param_grid=params,
                           scoring='accuracy',
                           cv=5)
      gs_dt.fit(cuisines_feature_df, cuisines_label_df)
      print(gs_dt.best_params_)
      # find best model score
      print(gs_dt.score(cuisines_feature_df, cuisines_label_df))

      {'dt__max_depth': 9, 'dt__min_samples_leaf': 3}
      0.6475594493116396
```



```
[110] from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier

      rf = RandomForestClassifier()

      params = {'max_depth': [5, 7, 9],
                'n_estimators': [50, 100, 200],
                'max_features': ['sqrt', 'log2']
                }

      grid = GridSearchCV(rf, params, cv=10, scoring='accuracy', return_train_score=False)
      grid.fit(cuisines_feature_df, cuisines_label_df)

      print(grid.best_params_)
      # find best model score
      print(grid.score(cuisines_feature_df, cuisines_label_df))
```

```
      {'max_depth': 9, 'max_features': 'log2', 'n_estimators': 200}
      0.799749687108886
```



```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.model_selection import GridSearchCV

param = {"base_estimator__criterion" : ["gini", "entropy"],
              "base_estimator__splitter" :   ["best", "random"],
              'n_estimators':[10,50],
              }
ada = AdaBoostClassifier(base_estimator=DecisionTreeClassifier())
grid = GridSearchCV(ada, param, cv=10,scoring = 'accuracy',return_train_score=False)
grid.fit(cuisines_feature_df, cuisines_label_df)

print(grid.best_params_)
# find best model score
print(grid.score(cuisines_feature_df, cuisines_label_df))
```

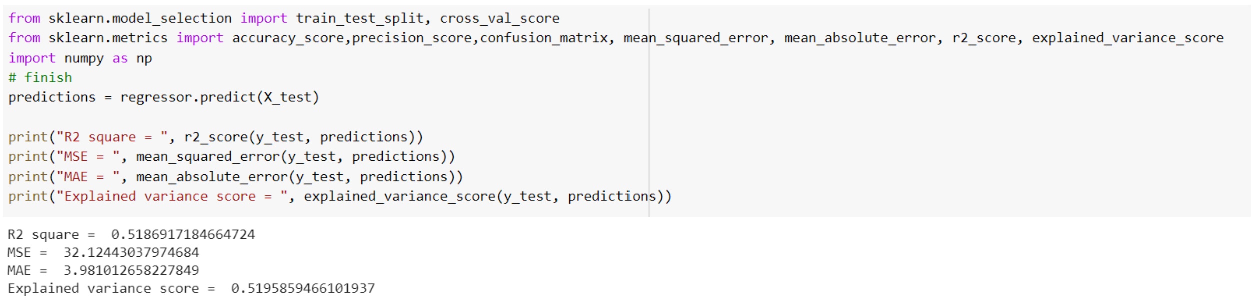
```
{'base_estimator__criterion': 'entropy', 'base_estimator__splitter': 'random', 'n_estimators': 50}
0.9566958698372966
```

Ada Boost is an algorithm that can be used with many different types of classifiers, such as decision trees. Ada boost work by weights instances in the training data set according to their importance. The weighting of instances is done according to weighting function. Both Ada boost and Random Forest achieved high accuracy on this dataset. Ada boost slightly outperformed Random Forest, but this is not significant.

Exercise 3

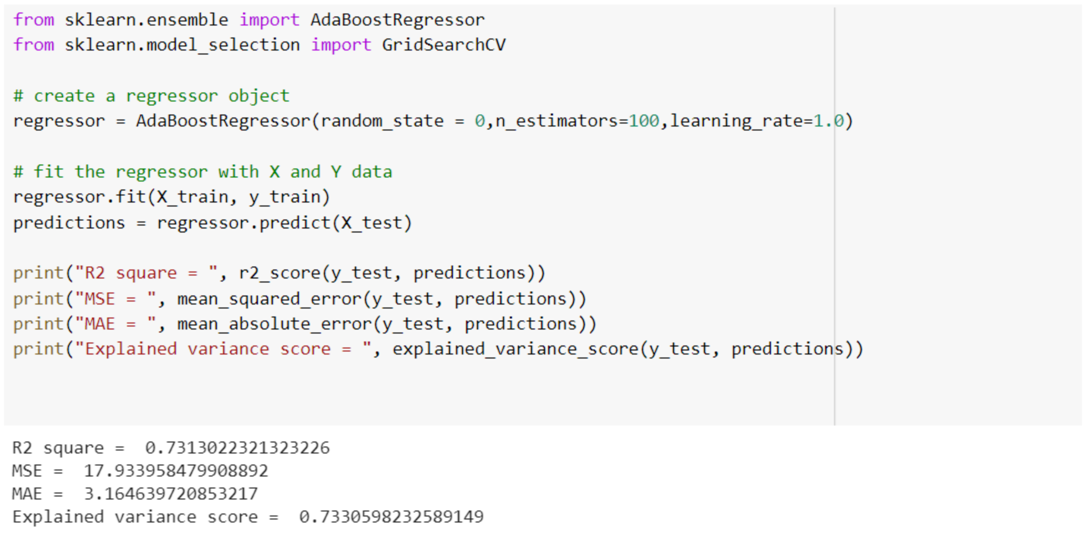
Try other combination of hyperparameters for Random Forest and AdaBoost models and check how good of an accuracy you can obtain.

Answer:



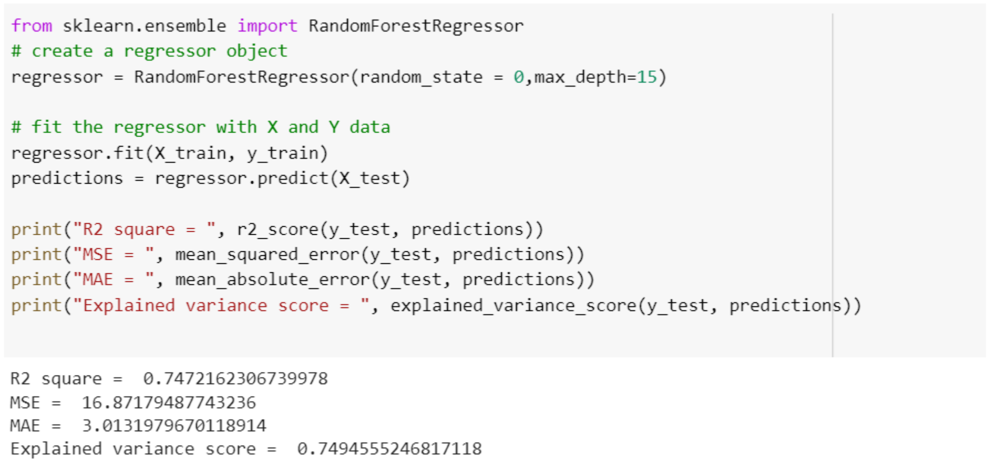
```
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score,precision_score,confusion_matrix, mean_squared_error, mean_absolute_error, r2_score, explained_variance_score
import numpy as np
# finish
predictions = regressor.predict(X_test)

print("R2 square = ", r2_score(y_test, predictions))
print("MSE = ", mean_squared_error(y_test, predictions))
print("MAE = ", mean_absolute_error(y_test, predictions))
print("Explained variance score = ", explained_variance_score(y_test, predictions))
```

```
R2 square =  0.5186917184664724
MSE =  32.12443037974684
MAE =  3.981012658227849
Explained variance score =  0.5195859466101937
```



```
from sklearn.ensemble import AdaBoostRegressor
from sklearn.model_selection import GridSearchCV

# create a regressor object
regressor = AdaBoostRegressor(random_state = 0,n_estimators=100,learning_rate=1.0)

# fit the regressor with X and Y data
regressor.fit(X_train, y_train)
predictions = regressor.predict(X_test)

print("R2 square = ", r2_score(y_test, predictions))
print("MSE = ", mean_squared_error(y_test, predictions))
print("MAE = ", mean_absolute_error(y_test, predictions))
print("Explained variance score = ", explained_variance_score(y_test, predictions))
```

```
R2 square =  0.7313022321323226
MSE =  17.933958479908892
MAE =  3.164639720853217
Explained variance score =  0.7330598232589149
```



```
from sklearn.ensemble import RandomForestRegressor
# create a regressor object
regressor = RandomForestRegressor(random_state = 0,max_depth=15)

# fit the regressor with X and Y data
regressor.fit(X_train, y_train)
predictions = regressor.predict(X_test)

print("R2 square = ", r2_score(y_test, predictions))
print("MSE = ", mean_squared_error(y_test, predictions))
print("MAE = ", mean_absolute_error(y_test, predictions))
print("Explained variance score = ", explained_variance_score(y_test, predictions))
```

```
R2 square =  0.7472162306739978
MSE =  16.87179487743236
MAE =  3.0131979670118914
Explained variance score =  0.7494555246817118
```

Both Ada boost and Random Forest achieved high accuracy on this dataset. Ada boost slightly outperformed Random Forest, but the difference is not significant. However, It also depends how to optimal the hyperparameters to create a best version for each algorithm. Boosting is combining a set of "weak" classifiers to produce a strong classifier. Start with equal weights given to all training instances. Create a final hypothesis by taking a weighted average of these k hypotheses. Random Forest the main hyperparameters I tuned were the number of trees in the forest (n_estimators) and the maximum depth of each tree (max_depth). Ada boost main hyperparameters I tuned were the number of trees in the forest (n_estimators) and the learning rate of each tree (Learning_rate). We can create a better version by adjust the number to have a optimal version for this data set.