Random Forest Regressor

1934012
AML lab exercise 4

Feature scaling:

```
from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
df = pd.DataFrame(scaler.fit_transform(df), columns=df.columns,
index=df.index)
```

The dataset has been scaled with Standard Scaler from sklearn library.

Performance:

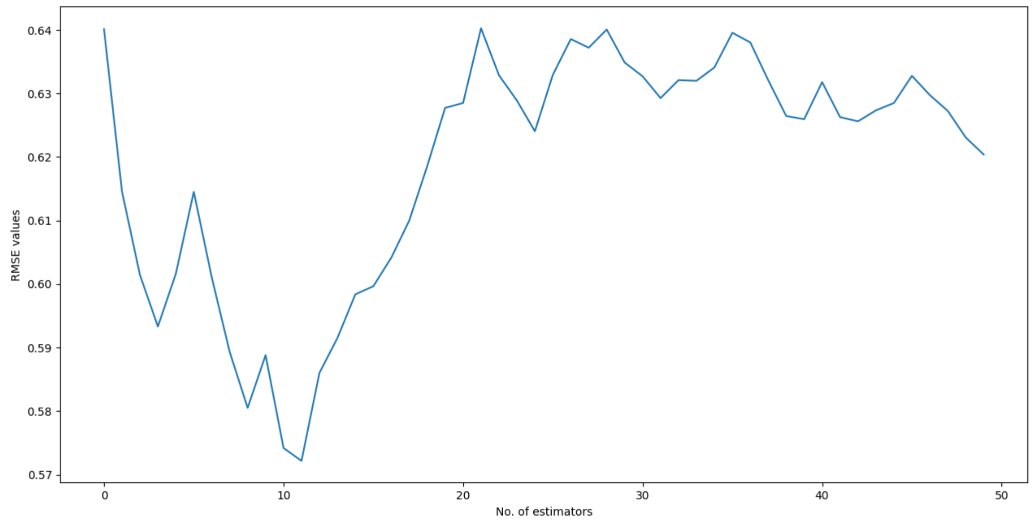
```
   Test Model       MAE       MSE      RMSE  R2 Square
Estimators 20  0.443725  0.329697  0.574192   0.711478

  Train Model       MAE       MSE      RMSE  R2 Square
Estimators 20  0.211683  0.112338  0.335169   0.879397

   Test Model       MAE       MSE      RMSE  R2 Square
Estimators 50  0.502278  0.399173  0.631801   0.650679

  Train Model       MAE       MSE      RMSE  R2 Square
Estimators 50  0.189437  0.087887  0.296458   0.905647
```

For both Regressor models with 20 and 50 estimators the training error is less than the testing error; it shows that the decision trees tend to overfit the training data.When we compare the test error of random forest with 20 and 50 estimators it is obvious that the error of 20 estimators is lesser than the error with 50 estimators.From the below image we can see that the error value increases with increase in no. of estimators.from this we can conclude that the error rate is not proportional to the no. of estimators.In this example the model with 20 estimators performs better.



CODE:

```
import pandas as pd
import numpy as np


url = "D:/g drive/3rd year/5th sem/AML lab/Ex4 - Random Forest/petrol_consumption.csv"
df = pd.read_csv(url)
print(df.head(2))

from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
df = pd.DataFrame(scaler.fit_transform(df), columns=df.columns, index=df.index)

from sklearn.model_selection import train_test_split
X = df.drop(['Petrol_Consumption'],axis=1)
y = df['Petrol_Consumption']

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=3)

from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators=20,random_state=3)
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)

from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error
mae = mean_absolute_error(y_test,y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
```

```
r2 = r2_score(y_test, y_pred)
results_df = pd.DataFrame(data=[["Estimators 20", mae, mse, rmse, r2]],columns=['Test Model',
'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(results_df)
print()

y_pred = rf.predict(X_train)
mae = mean_absolute_error(y_train,y_pred)
mse = mean_squared_error(y_train, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_train, y_pred)
results_df = pd.DataFrame(data=[["Estimators 20", mae, mse, rmse, r2]],columns=['Train
Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(results_df)
print()

rf = RandomForestRegressor(n_estimators=50,random_state=3)
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)

mae = mean_absolute_error(y_test,y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)
results_df = pd.DataFrame(data=[["Estimators 50", mae, mse, rmse, r2]],columns=['Test Model',
'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(results_df)
print()


y_pred = rf.predict(X_train)
mae = mean_absolute_error(y_train,y_pred)
mse = mean_squared_error(y_train, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_train, y_pred)
results_df = pd.DataFrame(data=[["Estimators 50", mae, mse, rmse, r2]],columns=['Train
Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(results_df)
print()


import matplotlib.pyplot as plt
from sklearn import tree
import seaborn as sns
```

```
corr = df.corr()
sns.heatmap(corr, xticklabels=corr.columns.values,yticklabels=corr.columns.values)
#plt.show()

fn= ['Petrol_tax','Average_income','Paved_Highways','Population_Driver_licence(%)']
cn= ['Petrol_Consumption']
fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (4,4), dpi=800)
tree.plot_tree(rf.estimators_[0],
               feature_names = fn,
               class_names=cn,
               filled = True);
#fig.savefig('rf_individualtree.png')

list =[]
for i in range(10,60):
    rfr=RandomForestRegressor(n_estimators=i,random_state=3)
    rfr.fit(X_train,y_train)
    y_pred=rfr.predict(X_test)
    rmse=mean_squared_error(y_test,y_pred,squared=False)
    list.append(rmse)

plt.figure(figsize=(8,8))
plt.plot(list)
plt.xlabel("No. of estimators")
plt.ylabel("RMSE values")
#plt.show()
```