211112262
Vaibhav Patel

ASSIGMNET -1

machine learning lab

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import os

train_dirs=[]
test_dirs=[]
for dir in os.listdir("./"):
if(dir.find("5-fold")!=-1):
train_dirs.append("./"+dir+"/train/")
test_dirs.append("./"+dir+"/test/")
```

```
def cal_header_val(file_path):
with open(file_path, "r") as file:
lines=file.readlines()
return lines.index('@data\n')+1
```

```
headers=[]
for dir in train_dirs:
file_path=dir+os.listdir (dir)[0]
headers.append(cal_header_val(file_path))
```

```
headers=np.array(headers)
```

```
def LinearRegression(train_file, test_file, header):
train_df = pd.read_csv(train_file, header=header, delimiter=",")
test_df = pd.read_csv(test_file, header=header, delimiter=",")
```

```
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values
```

```
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values
```

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
```

```
regressor.fit(X_train, y_train)
```

```
y_pred = regressor.predict(X_test)
```

```
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2score = r2_score(y_test, y_pred)
return np.array([mse, mae, r2score])
```

```
def PolynomialRegression(train_file, test_file, header, degree):
train_df = pd.read_csv(train_file, header=header, delimiter=",")
test_df = pd.read_csv(test_file, header=header, delimiter=",")
```

```
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values
```

```
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values
```

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
```

```
poly_reg=PolynomialFeatures(degree=degree)
X_poly=poly_reg.fit_transform(X_train)
```

```
regressor = LinearRegression()
regressor.fit(X_poly, y_train)
```

```
y_pred = regressor.predict(poly_reg.transform(X_test))
```

```
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
```

```
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2score = r2_score(y_test, y_pred)
return np.array([mse, mae, r2score])
```

```
def Regularization(train_file, test_file, header):
train_df = pd.read_csv(train_file, header=header, delimiter=",")
test_df = pd.read_csv(test_file, header=header, delimiter=",")
```

```
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values
```

```
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values
```

```
from sklearn.linear_model import Ridge
alphas = np.array([2**i for i in range(-18, 51, 2)])
```

```
best_mse, best_mse_alpha=float('inf'), None
best_mae, best_mae_alpha=float('inf'), None
best_r2, best_r2_alpha=float('-inf'), None
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
for alpha in alphas:
regressor=Ridge(alpha=alpha)
regressor.fit(X_train, y_train)
y_pred=regressor.predict(X_test)
mse=mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2score = r2_score(y_test, y_pred)
```

```
if mse<best_mse:
best_mse, best_mse_alpha=mse, alpha
if mae<best_mae:
best_mae, best_mae_alpha = mae, alpha
if r2score>best_r2:
best_r2, best_r2_alpha = r2score, alpha
return np.array([best_mse, best_mse_alpha, best_mae, best_mae_alpha,
best_r2, best_r2_alpha])
```

```
def PolynomialRidge(train_file, test_file, header, degree):
train_df = pd.read_csv(train_file, header=header, delimiter=",")
test_df = pd.read_csv(test_file, header=header, delimiter=",")
```

```
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values
```

```
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values
```

```
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
```

```
poly_reg=PolynomialFeatures(degree=degree)
X_poly=poly_reg.fit_transform(X_train)
```

```
alphas=np.array([2**i for i in range(-18, 30)])
```

```
best_mse, best_mse_alpha=float('inf'), None
best_mae, best_mae_alpha=float('inf'), None
best_r2, best_r2_alpha=float('-inf'), None
```

```
from sklearn.metrics import mean_squared_error, mean_absolute_error,
r2_score
for alpha in alphas:
regressor=Ridge(alpha=alpha)
regressor.fit(X_train, y_train)
y_pred=regressor.predict(X_test)
mse=mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2score = r2_score(y_test, y_pred)
```

```
if mse<best_mse:
best_mse, best_mse_alpha=mse, alpha
if mae<best_mae:
best_mae, best_mae_alpha = mae, alpha
if r2score>best_r2:
best_r2, best_r2_alpha = r2score, alpha
return np.array([best_mse, best_mse_alpha, best_mae, best_mae_alpha,
best_r2, best_r2_alpha])
```

```
# for train_dir, test_dir, header in zip(train_dirs, test_dirs,
headers):
# train_files=os.listdir(train_dir)
# test_files=os.listdir(test_dir)
# val=np.zeros(3)
# for train, test in zip(train_files, test_files):
# print(train, test)
# val+=LinearRegression(train_dir+train, test_dir+test, header)
# # val += PolynomialRegression(txrain_dir + train, test_dir + test,
header, 2)
# # val += PolynomialRegression(train_dir + train, test_dir + test,
header, 3)
# val/=len(train_files)
# val[0]=math.sqrt(val[0])
# val=pd.DataFrame(val, index=["RMSE", "MSE", "R2"],
columns=["Values"])
# print(val)
# print("-----------------------------\n")
```

```
for deg in range(1, 4):
print("Degree: ", deg)
for train_dir, test_dir, header in zip(train_dirs, test_dirs,
headers):
train_files = os.listdir(train_dir)
test_files = os.listdir(test_dir)
```

```
val = np.zeros(6)
```

```
for train, test in zip(train_files, test_files):
val += PolynomialRidge(train_dir + train, test_dir + test, header,
deg)
print(train_dir)
val /= len(train_files)
val = pd.DataFrame(val, index=["Best RMSE", "Best RMSE Alpha", "Best
MAE", "Best MAE Alpha", "Best R2 Score", "Best R2 Alpha"],
columns=["Values"])
print(val)
print("-----------------------------\n")
```

output

```
Degree:  1
./plastic-5-fold/train/
                    Values
Best RMSE          2.341392
Best RMSE Alpha   25.600003
Best MAE           1.231404
Best MAE Alpha   104.000000
Best R2 Score      0.798782
Best R2 Alpha     25.600003
----------------------------

./mortgage-5-fold/train/
                    Values
Best RMSE         0.014254
Best RMSE Alpha   0.300002
Best MAE          0.082841
Best MAE Alpha    0.150002
Best R2 Score     0.998478
Best R2 Alpha     0.300002
----------------------------

./diabetes-5-fold/train/
                      Values
Best RMSE         2.822110e-01
Best RMSE Alpha   4.352000e+02
Best MAE          4.203617e-01
Best MAE Alpha    1.073742e+08
Best R2 Score     3.136618e-01
Best R2 Alpha     4.352000e+02
----------------------------

./stock-5-fold/train/
                    Values
Best RMSE          5.444541
Best RMSE Alpha  409.600002
Best MAE           1.815886
Best MAE Alpha   435.200001
Best R2 Score      0.872061
Best R2 Alpha    409.600002
----------------------------
```