Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Course No : CSE4108

Course Title : Artificial Intelligence Lab

Assignment No 5

Date of Submission : 14.08.23

Submitted To : Dr. S.M.A. Al-Mamun
&
Mr. Raihan Tanvir

Submitted By

Group : B2

Name : MD Fardin Jaman Aranyak

Id 190204093

Section : B2

Data Set Used: WhiteWineQuality.csv
Associated Tasks: Linear Regression and Random Forest RegressionCharacteristics:
Number of Instances: 4898

Number of attributes: 12

Attribute Information:

Input variables

1. fixed acidity
2. volatile acidity

3. citric acid
4. residual sugar
5. chlorides
6. free sulfur dioxide
7. total sulfur dioxide
8. density
9. pH
10. sulfates
11. alcohol

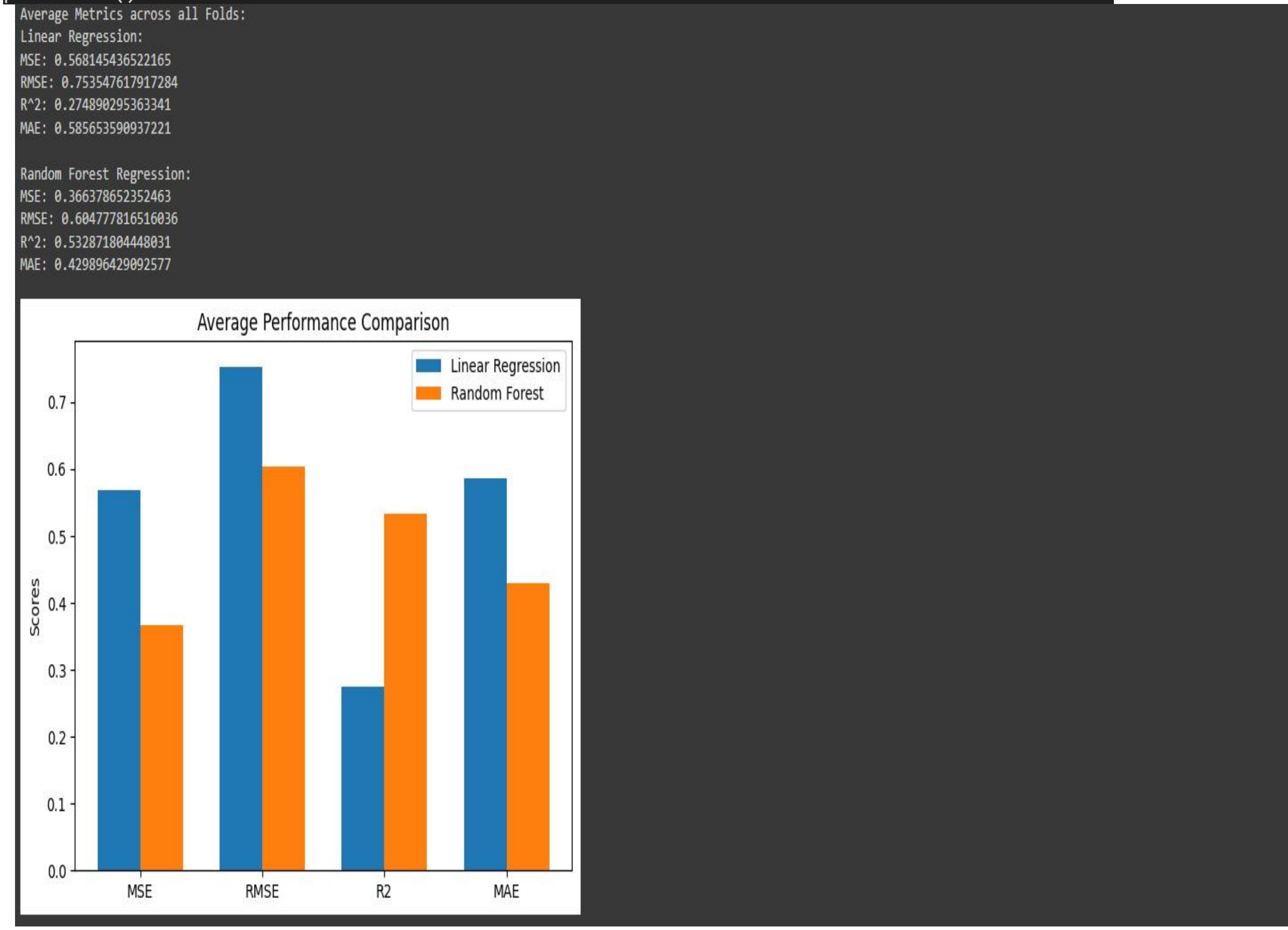
Output variable

12. quality (score between 0 and 10)

```
# -*- coding: utf-8 -*-
"""190204093_ASSIGNMET5.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/14vx7DHmKV4kMQiPFzSrH-SC0lsaM4QTe
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import KFold, train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Load the dataset
df = pd.read_csv('/content/drive/MyDrive/ASSIGNEMET5/Copy of
WhiteWineQuality.csv')

# Extract features and target variable
X = df.drop(columns=['quality'])
y = df['quality']

# Number of cross-validation folds
num_folds = 5

# Initialize KFold cross-validator
kf = KFold(n_splits=num_folds, shuffle=True, random_state=42)

# Initialize lists to store performance metrics for each fold
lr_mse_scores = []
lr_mae_scores = []
lr_r2_scores = []

rf_mse_scores = []
rf_mae_scores = []
rf_r2_scores = []

# Loop through each fold
for fold, (train_index, test_index) in enumerate(kf.split(X), start=1):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Standardize features
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
```

```python
    # Linear Regression
    lr = LinearRegression()
    lr.fit(X_train_scaled, y_train)
    lr_pred = lr.predict(X_test_scaled)

    lr_mse_scores.append(mean_squared_error(y_test, lr_pred))
    lr_mae_scores.append(mean_absolute_error(y_test, lr_pred))
    lr_r2_scores.append(r2_score(y_test, lr_pred))

    # Random Forest Regression
    rf = RandomForestRegressor(n_estimators=100, random_state=42)
    rf.fit(X_train_scaled, y_train)
    rf_pred = rf.predict(X_test_scaled)

    rf_mse_scores.append(mean_squared_error(y_test, rf_pred))
    rf_mae_scores.append(mean_absolute_error(y_test, rf_pred))
    rf_r2_scores.append(r2_score(y_test, rf_pred))

    # Print metrics for each fold
    print(f"Fold {fold}:")
    print("Linear Regression:")
    print(f"MSE: {lr_mse_scores[-1]:.15f}")
    print(f"RMSE: {np.sqrt(lr_mse_scores[-1]):.15f}")
    print(f"R^2: {lr_r2_scores[-1]:.15f}")
    print(f"MAE: {lr_mae_scores[-1]:.15f}")
    print()

    print("Random Forest Regression:")
    print(f"MSE: {rf_mse_scores[-1]:.15f}")
    print(f"RMSE: {np.sqrt(rf_mse_scores[-1]):.15f}")
    print(f"R^2: {rf_r2_scores[-1]:.15f}")
    print(f"MAE: {rf_mae_scores[-1]:.15f}")
    print()

    # Visualization for each fold
    metrics = ['MSE', 'MAE', 'R2']
    lr_scores = [lr_mse_scores[-1], lr_mae_scores[-1], lr_r2_scores[-1]]
    rf_scores = [rf_mse_scores[-1], rf_mae_scores[-1], rf_r2_scores[-1]]

    x = np.arange(len(metrics))
    width = 0.35

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, lr_scores, width, label='Linear Regression')
    rects2 = ax.bar(x + width/2, rf_scores, width, label='Random Forest')

    ax.set_ylabel('Scores')
    ax.set_title(f'Performance Comparison - Fold {fold}')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)
    ax.legend()

    fig.tight_layout()
    plt.show()

# Calculate average performance metrics across all folds
lr_avg_mse = np.mean(lr_mse_scores)
lr_avg_rmse = np.mean(np.sqrt(lr_mse_scores))
lr_avg_r2 = np.mean(lr_r2_scores)
lr_avg_mae = np.mean(lr_mae_scores)

rf_avg_mse = np.mean(rf_mse_scores)
rf_avg_rmse = np.mean(np.sqrt(rf_mse_scores))
rf_avg_r2 = np.mean(rf_r2_scores)
rf_avg_mae = np.mean(rf_mae_scores)

# Print average metrics
print("Average Metrics across all Folds:")
print("Linear Regression:")
print(f"MSE: {lr_avg_mse:.15f}")
print(f"RMSE: {lr_avg_rmse:.15f}")
print(f"R^2: {lr_avg_r2:.15f}")
print(f"MAE: {lr_avg_mae:.15f}")
print()

print("Random Forest Regression:")
print(f"MSE: {rf_avg_mse:.15f}")
print(f"RMSE: {rf_avg_rmse:.15f}")
print(f"R^2: {rf_avg_r2:.15f}")
print(f"MAE: {rf_avg_mae:.15f}")
print()

# Visualization
metrics = ['MSE', 'RMSE', 'R2', 'MAE']
lr_avg_scores = [lr_avg_mse, lr_avg_rmse, lr_avg_r2, lr_avg_mae]
rf_avg_scores = [rf_avg_mse, rf_avg_rmse, rf_avg_r2, rf_avg_mae]

x = np.arange(len(metrics))
width = 0.35

fig, ax = plt.subplots()
rects1 = ax.bar(x - width/2, lr_avg_scores, width, label='Linear Regression')
rects2 = ax.bar(x + width/2, rf_avg_scores, width, label='Random Forest')

ax.set_ylabel('Scores')
ax.set_title('Average Performance Comparison')
ax.set_xticks(x)
ax.set_xticklabels(metrics)
ax.legend()
```

```
fig.tight_layout()
plt.show()
```



```
Average Metrics across all Folds:
Linear Regression:
MSE: 0.568145436522165
RMSE: 0.753547617917284
R^2: 0.274890295363341
MAE: 0.585653590937221

Random Forest Regression:
MSE: 0.366378652352463
RMSE: 0.604777816516036
R^2: 0.532871804448031
MAE: 0.429896429092577
```