PRACTICAL TECHNICAL ASSESMENT

SHARON P. A

Activity

1. Load the dataset and apply necessary preprocessing steps.

2. Perform exploratory data analysis (EDA) to understand the dataset.

3. Implement classification models and evaluate them using a confusion matrix and

cross-validation.

4. Implement regression models and evaluate them using R-squared, MSE, and cross□validation.

5. Visualize the confusion matrix for at least one classification model.

6. Report and interpret the results of each model.

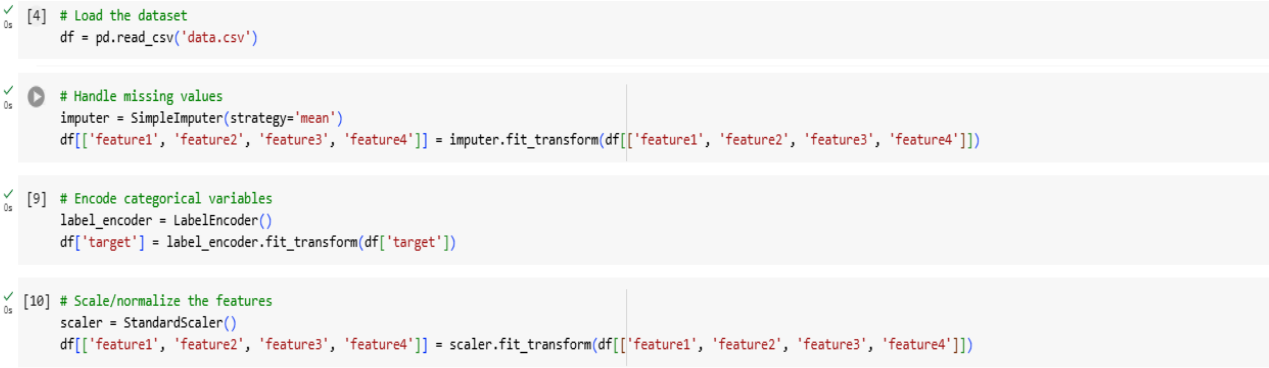
Requirements

- Personal computer/laptop
- Google Collab
- Dataset (data.csv)

Procedure

Data Preprocessing

- Load the dataset using pd.read_csv('data.csv')
- Handle missing values.
- Encode categorical variables.
- Scale/normalize the features.



```
# Load the dataset
df = pd.read_csv('data.csv')
```

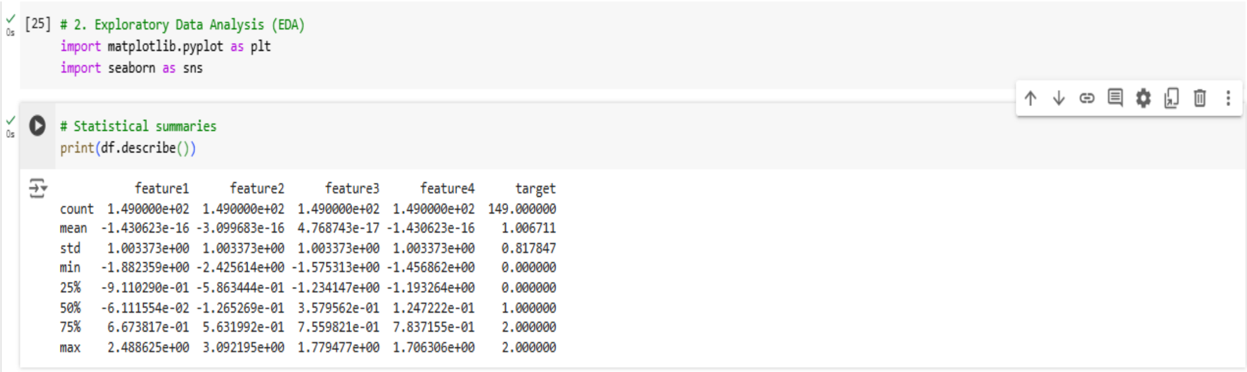
```
# Handle missing values
imputer = SimpleImputer(strategy='mean')
df[['feature1', 'feature2', 'feature3', 'feature4']] = imputer.fit_transform(df[['feature1', 'feature2', 'feature3', 'feature4']])
```

```
# Encode categorical variables
label_encoder = LabelEncoder()
df['target'] = label_encoder.fit_transform(df['target'])
```

```
# Scale/normalize the features
scaler = StandardScaler()
df[['feature1', 'feature2', 'feature3', 'feature4']] = scaler.fit_transform(df[['feature1', 'feature2', 'feature3', 'feature4']])
```

Exploratory Data Analysis (EDA)

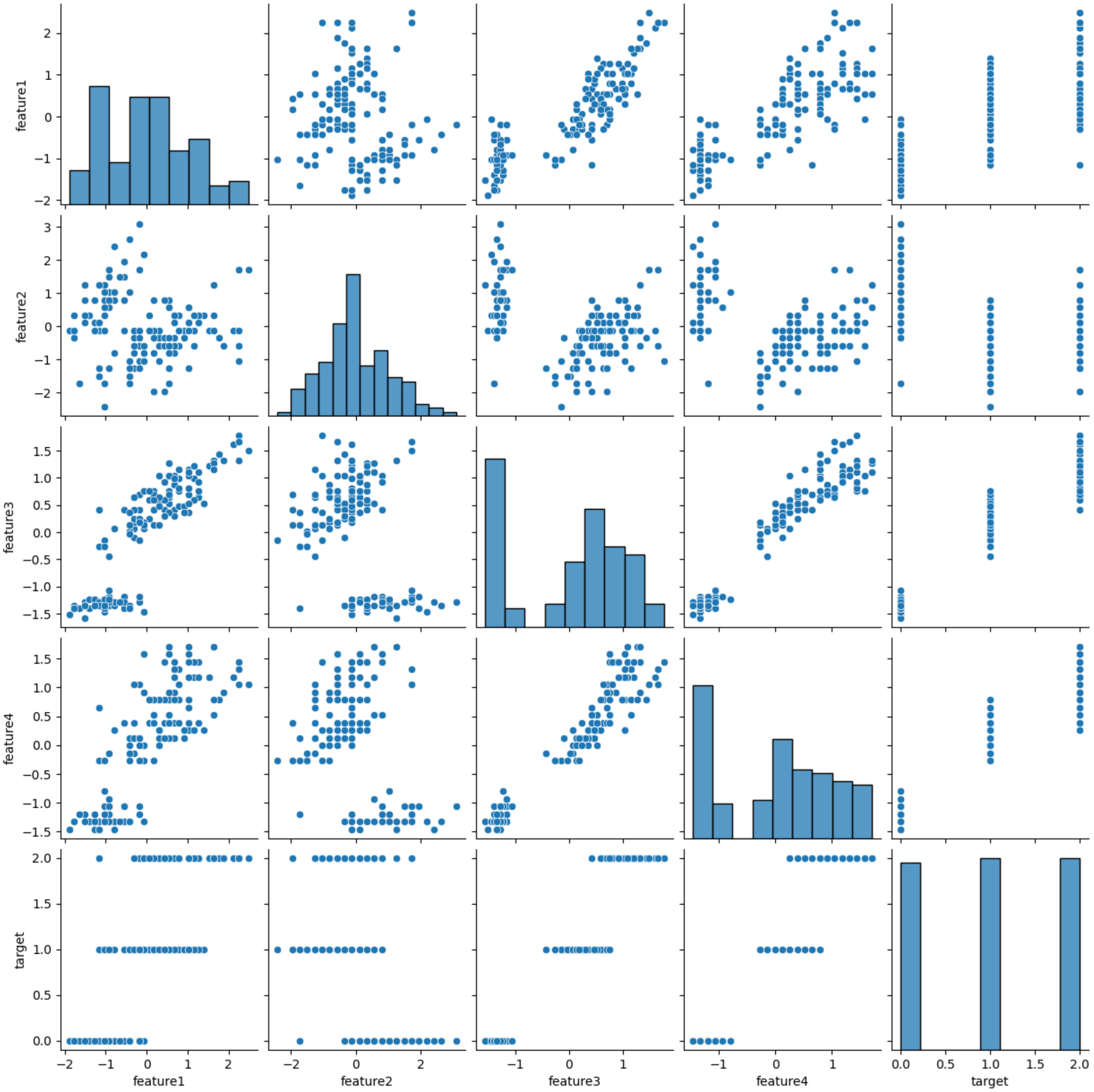
- Provide statistical summaries of the dataset.
- Visualize the data distribution and relationships between features using plots.



```
# 2. Exploratory Data Analysis (EDA)
import matplotlib.pyplot as plt
import seaborn as sns
```

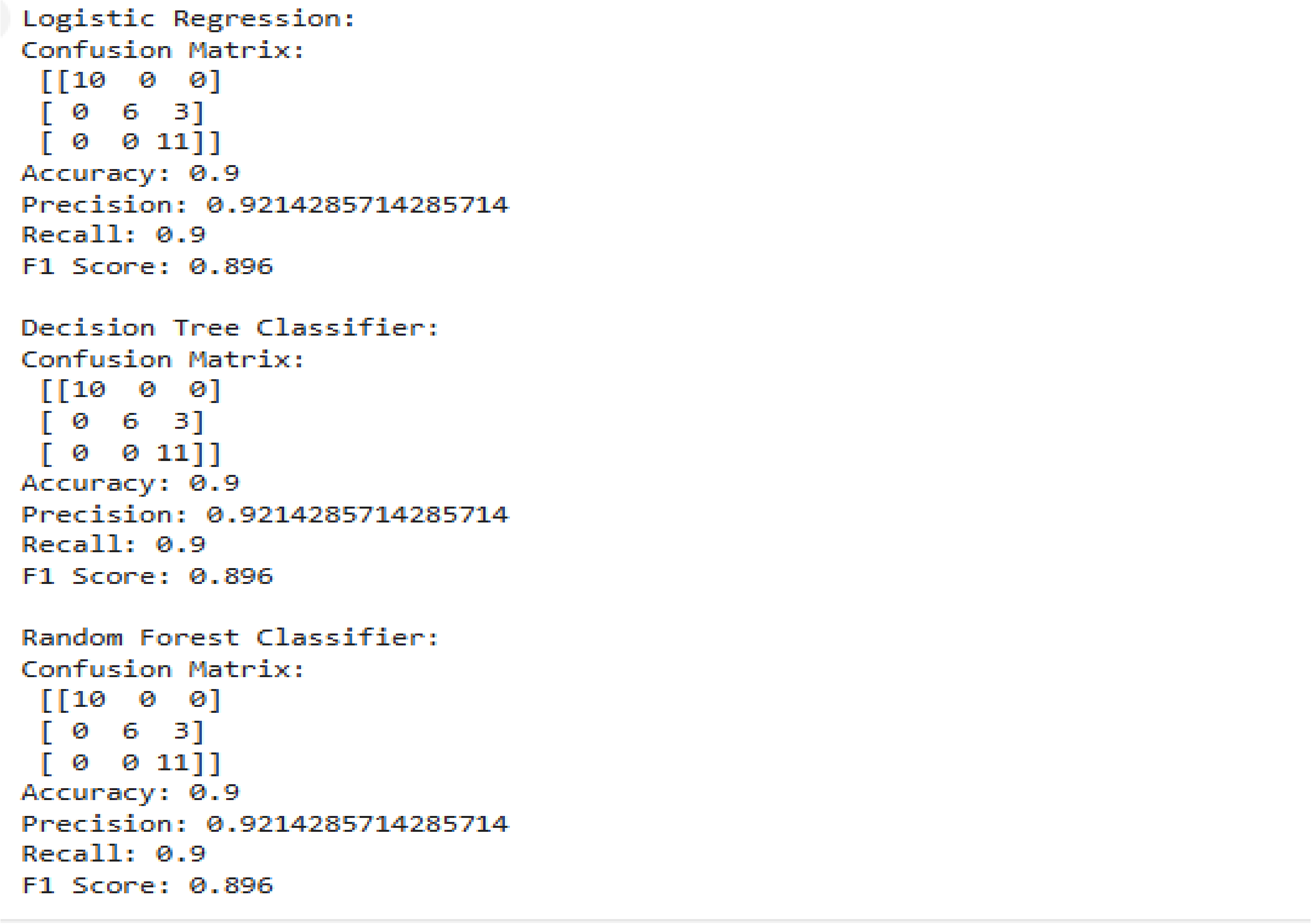
```
# Statistical summaries
print(df.describe())
```

```
           feature1      feature2      feature3      feature4      target
count  1.490000e+02  1.490000e+02  1.490000e+02  1.490000e+02  149.000000
mean  -1.430623e-16 -3.099683e-16  4.768743e-17 -1.430623e-16    1.006711
std    1.003373e+00  1.003373e+00  1.003373e+00  1.003373e+00    0.817847
min   -1.882359e+00 -2.425614e+00 -1.575313e+00 -1.456862e+00    0.000000
25%   -9.110290e-01 -5.863444e-01 -1.234147e+00 -1.193264e+00    0.000000
50%   -6.111554e-02 -1.265269e-01  3.579562e-01  1.247222e-01    1.000000
75%    6.673817e-01  5.631992e-01  7.559821e-01  7.837155e-01    2.000000
max    2.488625e+00  3.092195e+00  1.779477e+00  1.706306e+00    2.000000
```


feature1
2
1
0
−1
−2
feature2
3
2
1
0
−1
−2
feature3
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
feature4
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
target
2.0
1.5
1.0
0.5
0.0
−2
−1
0
1
2
feature1
−2
0
2
feature2
−1
0
1
feature3
−1
0
1
feature4
0.0
0.5
1.0
1.5
2.0
target

Classification

- Apply Logistic Regression, Decision Tree, and Random Forest classifiers.
- Use a confusion matrix to evaluate the performance of each classifier.
- Perform cross-validation to assess the model stability.



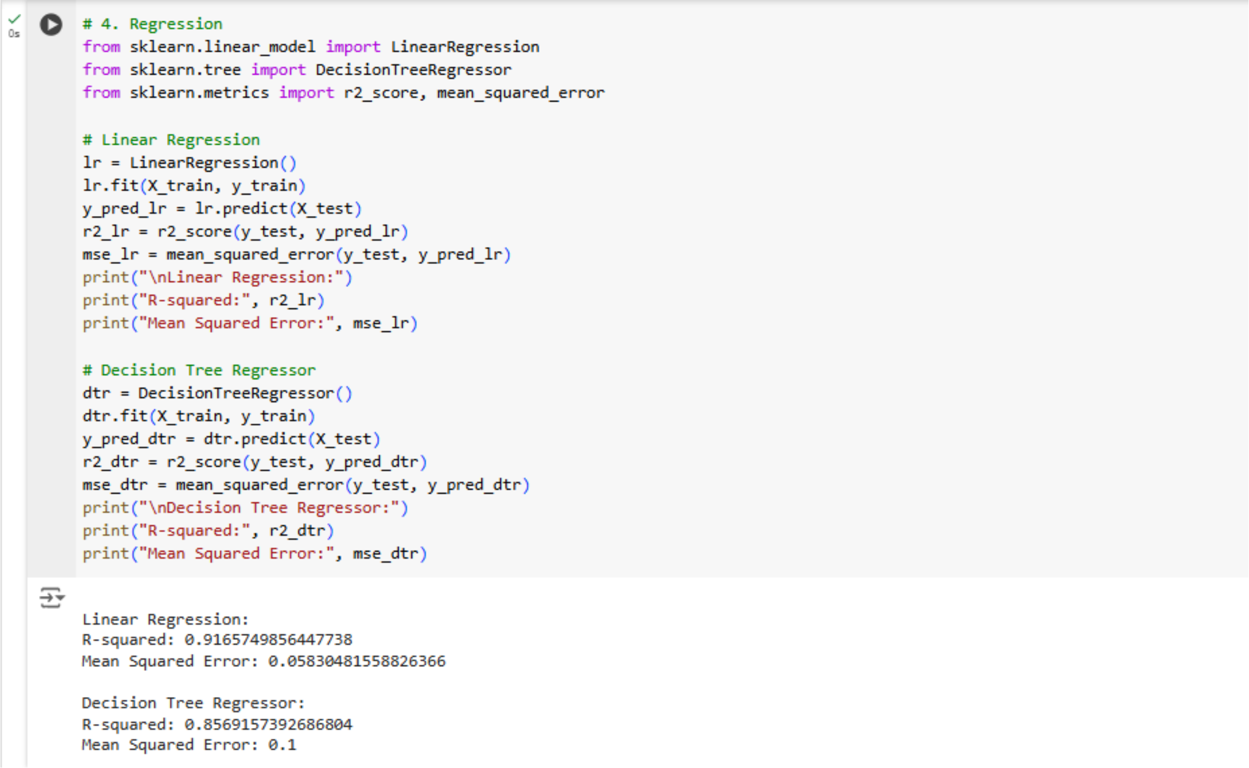
```
Logistic Regression:
Confusion Matrix:
 [[10  0  0]
 [ 0  6  3]
 [ 0  0 11]]
Accuracy: 0.9
Precision: 0.9214285714285714
Recall: 0.9
F1 Score: 0.896

Decision Tree Classifier:
Confusion Matrix:
 [[10  0  0]
 [ 0  6  3]
 [ 0  0 11]]
Accuracy: 0.9
Precision: 0.9214285714285714
Recall: 0.9
F1 Score: 0.896

Random Forest Classifier:
Confusion Matrix:
 [[10  0  0]
 [ 0  6  3]
 [ 0  0 11]]
Accuracy: 0.9
Precision: 0.9214285714285714
Recall: 0.9
F1 Score: 0.896
```

Regression

- Apply Linear Regression and Decision Tree Regressor.
- Evaluate the models using R-squared and Mean Squared Error (MSE).
- Perform cross-validation to assess the model stability.



```
# 4. Regression
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import r2_score, mean_squared_error

# Linear Regression
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
r2_lr = r2_score(y_test, y_pred_lr)
mse_lr = mean_squared_error(y_test, y_pred_lr)
print("\nLinear Regression:")
print("R-squared:", r2_lr)
print("Mean Squared Error:", mse_lr)

# Decision Tree Regressor
dtr = DecisionTreeRegressor()
dtr.fit(X_train, y_train)
y_pred_dtr = dtr.predict(X_test)
r2_dtr = r2_score(y_test, y_pred_dtr)
mse_dtr = mean_squared_error(y_test, y_pred_dtr)
print("\nDecision Tree Regressor:")
print("R-squared:", r2_dtr)
print("Mean Squared Error:", mse_dtr)
```

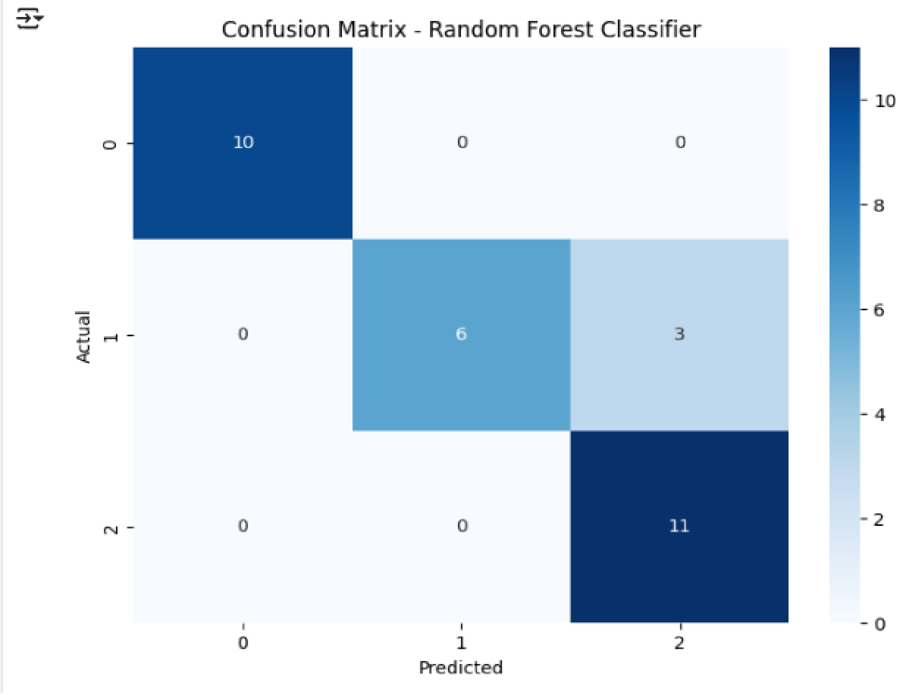
```
Linear Regression:
R-squared: 0.9165749856447738
Mean Squared Error: 0.05830481558826366

Decision Tree Regressor:
R-squared: 0.8569157392686804
Mean Squared Error: 0.1
```

Confusion Matrix

For classification tasks, plot the confusion matrix and compute the following metrics:

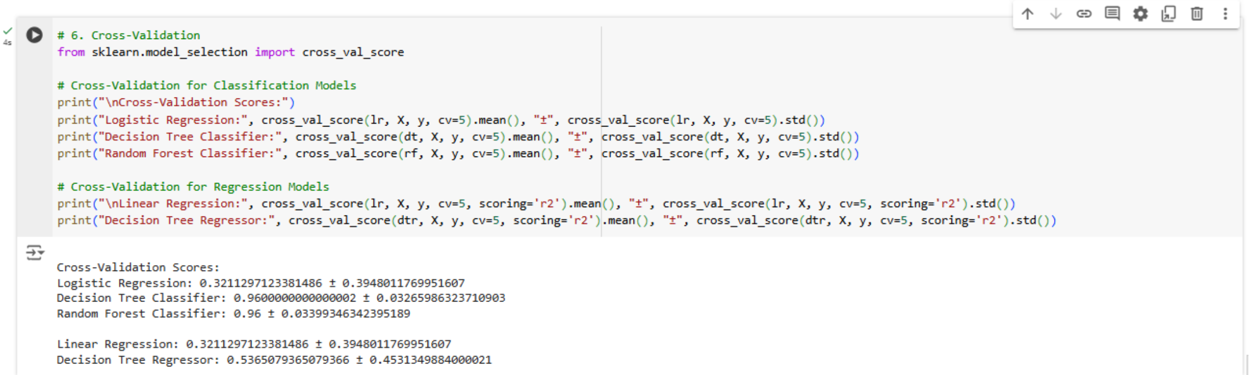
- Accuracy
- Precision
- Recall
- F1 Score



	0	1	2
0	10	0	0
1	0	6	3
2	0	0	11

Cross-Validation

- Implement k-fold cross-validation for both classification and regression models.
- Report the mean and standard deviation of the cross-validation scores.



```python
# 6. Cross-Validation
from sklearn.model_selection import cross_val_score

# Cross-Validation for Classification Models
print("\nCross-Validation Scores:")
print("Logistic Regression:", cross_val_score(lr, X, y, cv=5).mean(), "±", cross_val_score(lr, X, y, cv=5).std())
print("Decision Tree Classifier:", cross_val_score(dt, X, y, cv=5).mean(), "±", cross_val_score(dt, X, y, cv=5).std())
print("Random Forest Classifier:", cross_val_score(rf, X, y, cv=5).mean(), "±", cross_val_score(rf, X, y, cv=5).std())

# Cross-Validation for Regression Models
print("\nLinear Regression:", cross_val_score(lr, X, y, cv=5, scoring='r2').mean(), "±", cross_val_score(lr, X, y, cv=5, scoring='r2').std())
print("Decision Tree Regressor:", cross_val_score(dtr, X, y, cv=5, scoring='r2').mean(), "±", cross_val_score(dtr, X, y, cv=5, scoring='r2').std())
```

```
Cross-Validation Scores:
Logistic Regression: 0.3211297123381486 ± 0.3948011769951607
Decision Tree Classifier: 0.9600000000000002 ± 0.03265986323710903
Random Forest Classifier: 0.96 ± 0.03399346342395189

Linear Regression: 0.3211297123381486 ± 0.3948011769951607
Decision Tree Regressor: 0.5365079365079366 ± 0.4531349884000021
```

Conclusion

This documentation outlines the process of loading and preprocessing a dataset, performing exploratory data analysis (EDA), implementing classification and regression models, evaluating the models using various metrics, and visualizing results. The dataset used is assumed to have numerical and categorical features, with a target variable for prediction.