FOREST COVER PREDICTION

Using Machine Learning

Abstract

Building a system that can predict the type of forest cover using analysis data for a 30m x 30m patch of land in the forest

Moe Htet Min

Moehtetmin004@gmail.com

Table of Contents

Forest Cover Prediction - Project Report

GitHub: https://github.com/Moehtetmin28/Forest-Cover-Prediction

1. Introduction

Forest cover types require prediction based on geographical and environmental features within this project analysis. The dataset comprises information including elevation data as well as aspect analysis and slope measurements and types of soil present. The task involves creating an automated system able to forecast the forest cover type from analysis data collected across 30m x 30m forest land areas.

---

2. Dataset Overview

The analysis dataset originated from the forest department in the northern Colorado-area Roosevelt National Forest. The dataset contains different environmental characteristics including elevation, aspect, slope and soil types. The target variable Cover_Type functions as an integer classification system which defines different forest cover types:

1. Spruce/Fir
2. Lodgepole Pine
3. Ponderosa Pine
4. Cottonwood/Willow
5. Aspen
6. Douglas-fir
7. Krummholz

2.1 Feature Description

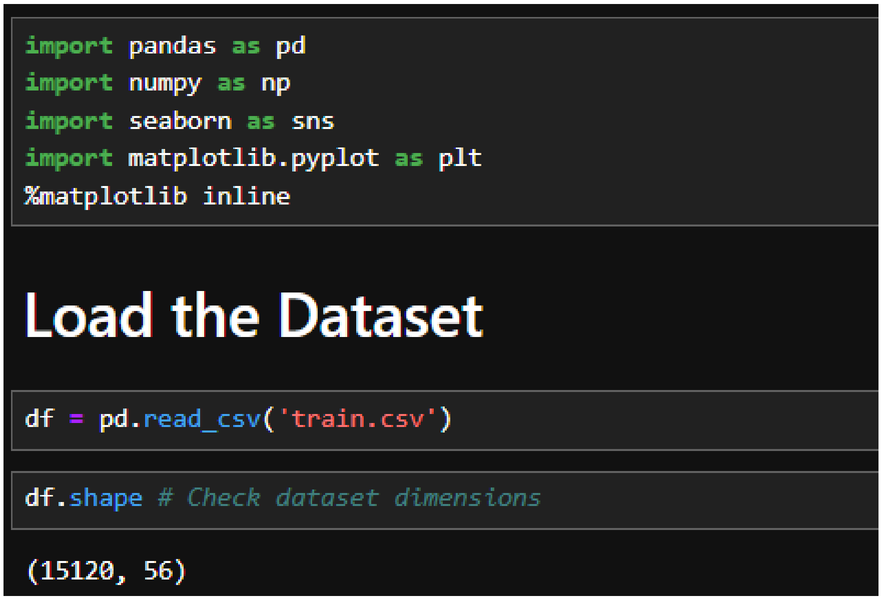
- **Elevation:** Elevation in meters
- **Aspect:** Aspect in degrees azimuth

- **Slope:** Slope in degrees
- **Horizontal_Distance_To_Hydrology:** Distance to nearest surface water features
- **Vertical_Distance_To_Hydrology:** Distance to nearest surface water features (vertical)
- **Horizontal_Distance_To_Roadways:** Distance to nearest roadway
- **Hillshade (morning, noon, evening):** Hillshade index (0 to 255) at different times of the day
- **Horizontal_Distance_To_Fire_Points:** Distance to nearest wildfire ignition points
- **Wilderness_Area (4 binary columns):** Wilderness area designation
- **Soil_Type (40 binary columns):** Soil type designation
- **Cover_Type:** Target variable representing forest cover type

---

3. Data Loading

The **pandas** program initiates the process of dataset loading:



```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Load the Dataset

```
df = pd.read_csv('train.csv')
```

```
df.shape # Check dataset dimensions
```

```
(15120, 56)
```

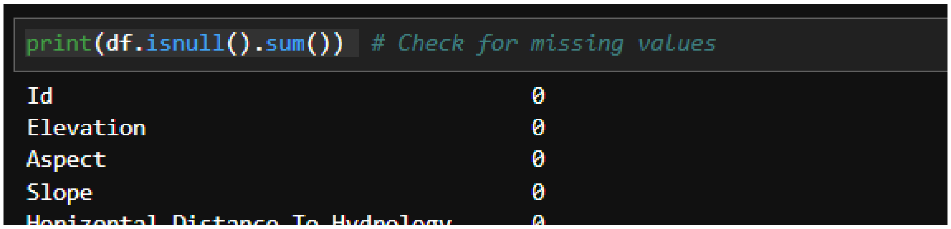
Explanation:

- Data manipulation functions within **pandas** allow users to work with their data.
- The numerical operations within the program run through **numpy**.
- The visualization tasks utilize **seaborn** together with **matplotlib**.
- The program uses pandas to read the dataset which shows the data attributes in the DataFrame alongside displaying its dimension structure.
- The machine learning model deployment occurs through the **sklearn** implementation framework.

---

4. Exploratory Data Analysis (EDA)

4.1 Checking Missing Values



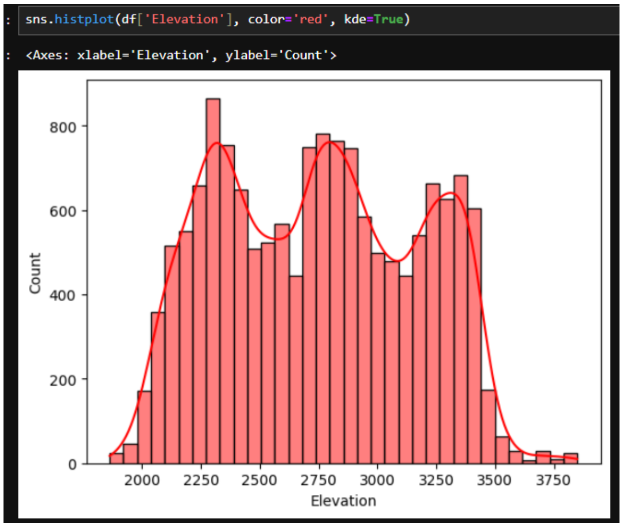
```
print(df.isnull().sum())  # Check for missing values
```

```
Id                                  0
Elevation                           0
Aspect                              0
Slope                               0
Horizontal_Distance_To_Hydrology    0
```

Explanation:

- A complete data set requires verification to determine the presence of missing data values.

4.2 Data Distribution – Elevation



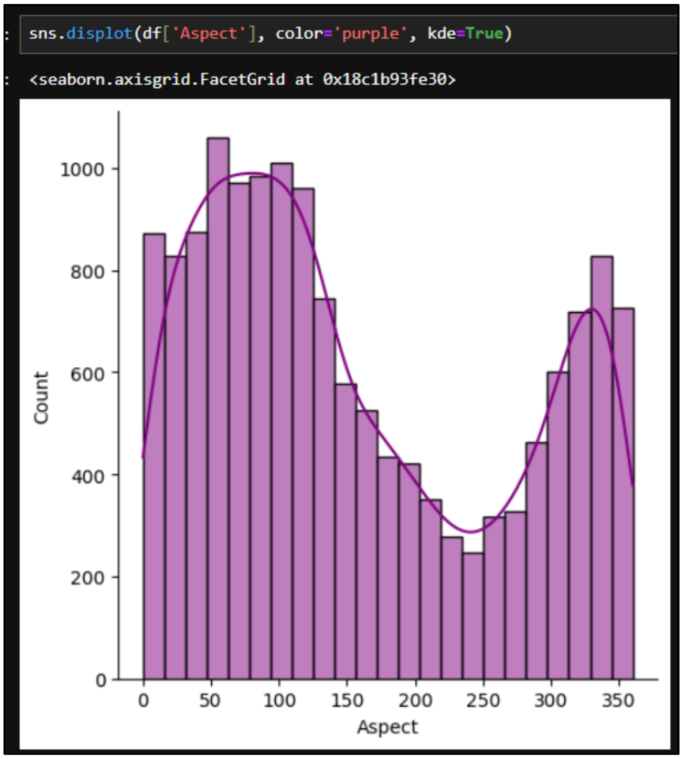
```
sns.histplot(df['Elevation'], color='red', kde=True)
```

```
<Axes: xlabel='Elevation', ylabel='Count'>
```

Explanation:

- The **Elevation** feature distribution appears in **histplot** through its Kernel Density Estimate (KDE) overlay display.

4.3 Aspect Analysis



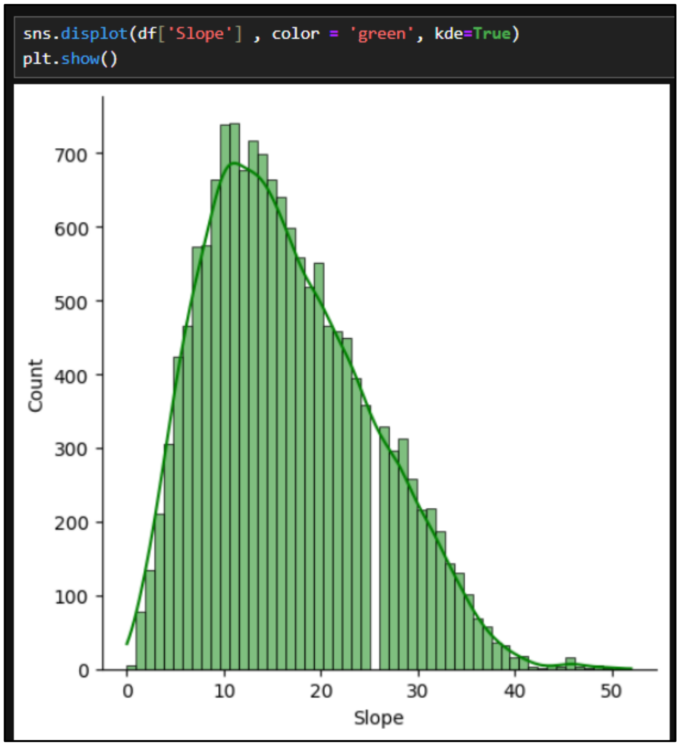
```
sns.displot(df['Aspect'], color='purple', kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x18c1b93fe30>
```

Explanation:

- The analysis of **Aspect** distribution between different **Cover_Type** categories relies on boxplots for interpretation.

4.4 Slope Analysis



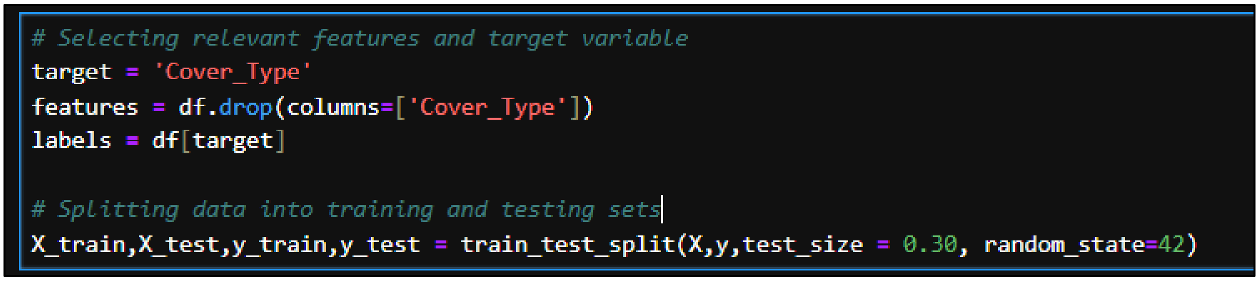
```
sns.displot(df['Slope'] , color = 'green', kde=True)
plt.show()
```

Explanation:

- The analysis of slope value changes between various forest cover types becomes possible through boxplot comparisons.

5. Machine Learing Model

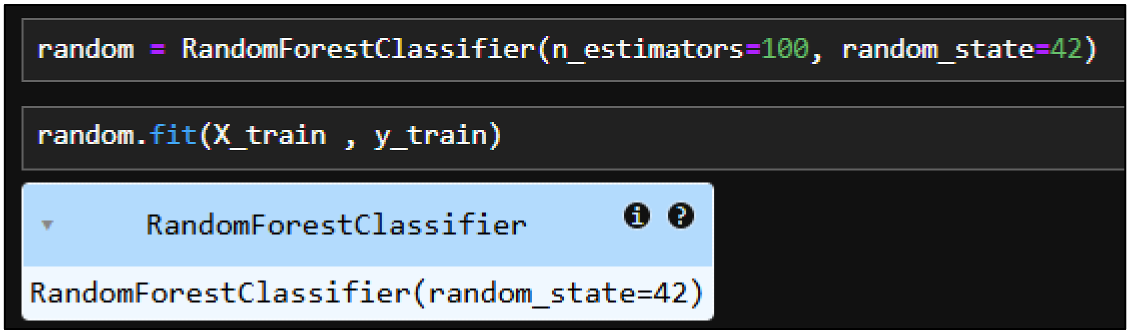
5.1 Feature Selection and Data Preparation



```
# Selecting relevant features and target variable
target = 'Cover_Type'
features = df.drop(columns=['Cover_Type'])
labels = df[target]

# Splitting data into training and testing sets
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.30, random_state=42)
```

5.2 Training a Random Forest Classifier

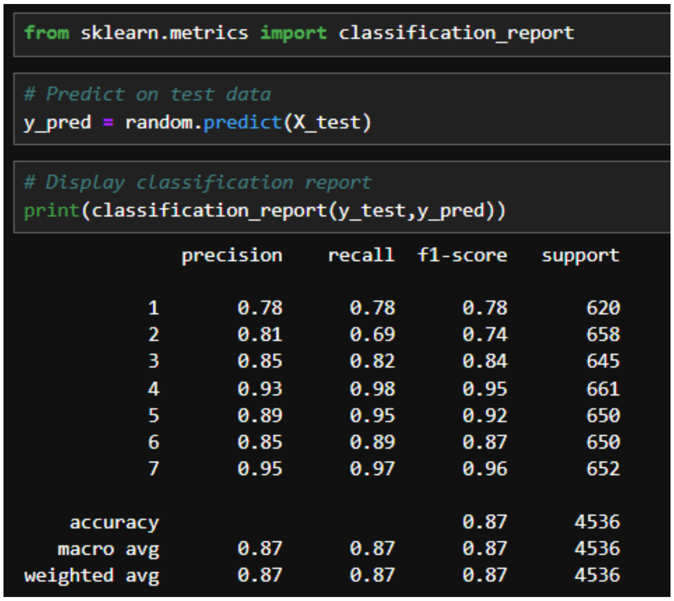


```
random = RandomForestClassifier(n_estimators=100, random_state=42)
```

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

```
RandomForestClassifier(random_state=42)
```

5.3 Evaluating the Model



```
from sklearn.metrics import classification_report
```

```
# Predict on test data
y_pred = random.predict(X_test)
```

```
# Display classification report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.78      0.78      0.78       620
           2       0.81      0.69      0.74       658
           3       0.85      0.82      0.84       645
           4       0.93      0.98      0.95       661
           5       0.89      0.95      0.92       650
           6       0.85      0.89      0.87       650
           7       0.95      0.97      0.96       652

    accuracy                           0.87      4536
   macro avg       0.87      0.87      0.87      4536
weighted avg       0.87      0.87      0.87      4536
```

Explanation:

- The information exists in separated training and testing groups.
- The predictive model selects **a Random Forest Classifier** for its operation.
- The evaluation of the model takes place through accuracy assessment and classification report analysis.

6. Conclusion

A study of the distribution patterns for important geographical variables including elevation aspect and slope was performed through exploratory data analysis. This evaluation allows us to determine the effects that geographical features have on defining forest cover types. The implemented Random Forest model creates an efficient system to forecast the forest cover type of specific 30m x 30m land areas. Future improvements could involve:

- The performance of the model can be improved through proper selection of hyperparameters.
- The development of deep learning neural networks requires testing to advance their accuracy potential.
- The process of transforming raw features into new constructs which better suits analytical models for representation enhancement.

==