BVRIT HYDERABAD

College of Engineering for Women

FOREST COVER

Team No: 12
Team Members:
A.Vennela , 21WH5A1201
V.Sushma , 21WH5A1202
Ch. Sandhya, 21WH5A1203
P. Sai Varshitha, 21WH5A1204
N. Jahnavi, 21WH5A1205
E. Mamatha, 21WH5A1206

VISHNU
UNIVERSAL LEARNING

AGENDA

- Problem statement
- Python Packages used
- Algorithm
- output
- Comparison table
- Execute the Code

VISHNU
UNIVERSAL LEARNING

PROBLEM STATEMENT

The study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. Each observation is a 30m x 30m patch. We are asked to predict an integer classification for the forest cover type. The seven types are:

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

Problem Statement

File descriptions:

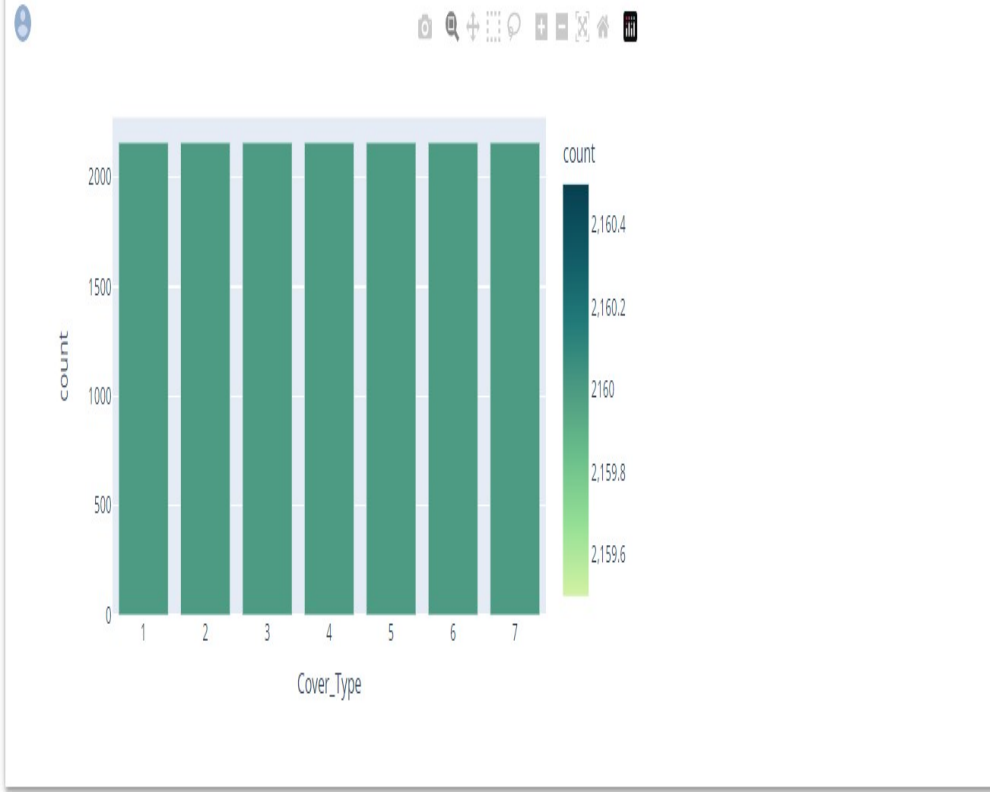
train.csv- the training set contains 15120 observations including both features and the Cover Type
test.csv- the test set contains 565892 observations including features and to predict the Cover Type
samplesubmission.csv - a sample submission file in the correct format
Data fields
id - id of the creature CoverType - type of the forest that the id belongs to

Python Packages used

- numpy
- pandas
- matplotlib.pyplot
- lightgbm
- gc
- plotly.express
- sklearn

Types of Forests in given training data

```
target_df = pd.DataFrame(train_df['Cover_Type'].value_counts()).reset_index()
target_df.columns = ['Cover_Type', 'count']
fig = px.bar(data_frame =target_df, x = 'Cover_Type', y = 'count', color = "count", color_continuous_scale = "Emrld", width=700, height=400)
fig.show()
```



VISHNU
UNIVERSAL LEARNING

Algorithms

- Decision Tree
- KNN
- Ada Boost
- Gaussian
- Support Vector Machine
- LIGHTGBM

VISHNU
UNIVERSAL LEARNING

Decision Tree

The decision tree algorithm is a popular machine learning technique used for classification and regression problems. In this forest cover project, the goal is to predict the forest cover type for each observation in the test set based on the features in the training set. The decision tree algorithm works: Splitting the Data , Building the Tree, Making Predictions. The decision tree algorithm recursively splits the data based on the values of the features to create a tree-like model that can be used to make predictions on new data. The goal is to create a tree that is as accurate as possible on the training data while avoiding overfitting. Accuracy:80

VISHNU
UNIVERSAL LEARNING

K-Nearest Neighbour

KNN (K-Nearest Neighbors) is a machine learning algorithm used for classification and regression analysis. In this this project, KNN can be used to predict the forest cover type based on the given features. In this project, KNN can be used to build a predictive model using the provided training set. Once the model is trained, it can be used to predict the forest cover type for each observation in the test set, and the predictions can be submitted as the final solution. The accuracy of the model can be evaluated using a suitable metric, such as accuracy, precision, recall, F1 score, etc. Accuracy:81

VISHNU
UNIVERSAL LEARNING

Adaptive Boosting

AdaBoost (Adaptive Boosting) is a machine learning algorithm that is often used in classification problems. The basic idea of AdaBoost is to combine several weak classifiers. The algorithm works by iteratively adjusting the weights of misclassified observations, such that subsequent weak classifiers focus more on the misclassified observations. In each iteration, a new weak classifier is trained on the re-weighted data and added to the ensemble. The final strong classifier is a weighted combination of the weak classifiers, where the weights are determined based on their performance on the training data.Overfitting occurs.
Accuracy:43

VISHNU
UNIVERSAL LEARNING

Gaussian Algorithm

The Gaussian algorithm, also known as the Gaussian Naive Bayes algorithm, is a probabilistic algorithm used for classification tasks in machine learning. It is based on Bayes' theorem, which states that the probability of a hypothesis (in this case, the class label) given some observed evidence (the feature values) is proportional to the probability of the evidence given the hypothesis times the prior probability of the hypothesis. Accuracy:63

VISHNU
UNIVERSAL LEARNING

Support Vector Machine

Support Vector Machines are a type of supervised learning algorithm that can be used for classification or regression tasks. In classification tasks, the algorithm learns a decision boundary that separates the different classes in the data. The decision boundary would be used to separate the seven different forest cover types. Accuracy:67

Light Gradient Boosting

It is similar to other popular gradient boosting frameworks like XGBoost and CatBoost, but it has some unique features that make it popular in machine learning competitions.LightGBM is designed to be faster and more memory-efficient than other gradient boosting frameworks. It achieves this by using a technique called "Gradient-based One-Side Sampling" (GOSS) to select only the most informative samples for each tree in the boosting process. This reduces the memory required for training and speeds up the process.LightGBM also uses a technique called "Exclusive Feature Bundling" (EFB) to group together features that are frequently used together. Memory can be reduced.LightGBM is a powerful and efficient gradient boosting framework that is often used in machine learning competitions and real-world applications. It can handle large datasets and a wide range of data types, and it has many useful features for feature engineering and model tuning. Accuracy:86

VISHNU
UNIVERSAL LEARNING

Light Gradient Boosting

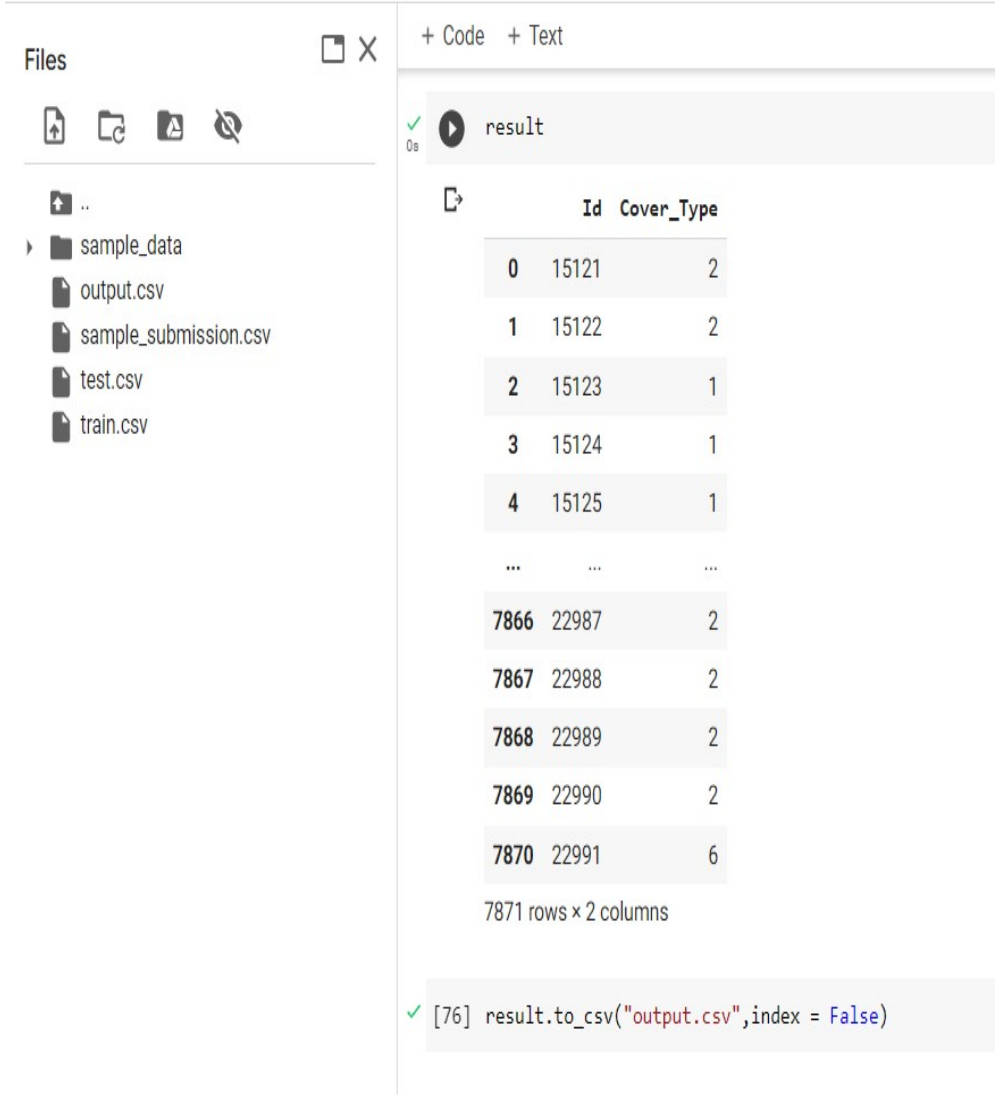
- Import the necessary libraries and load the training and test data.
- Split the training data into training and validation sets for model selection and tuning.
- Define the features and target variables.
- Encode the categorical variables using one-hot encoding.
- Define the lightgbm model and set the hyperparameters. You may want to use cross-validation to find the optimal hyperparameters.
- Train the model on the training set and validate on the validation set. Use the feature importance plot to identify the most important features.
- Use the trained model to predict on the test set and save the predictions to a file in the required format.

Confusion Matrix

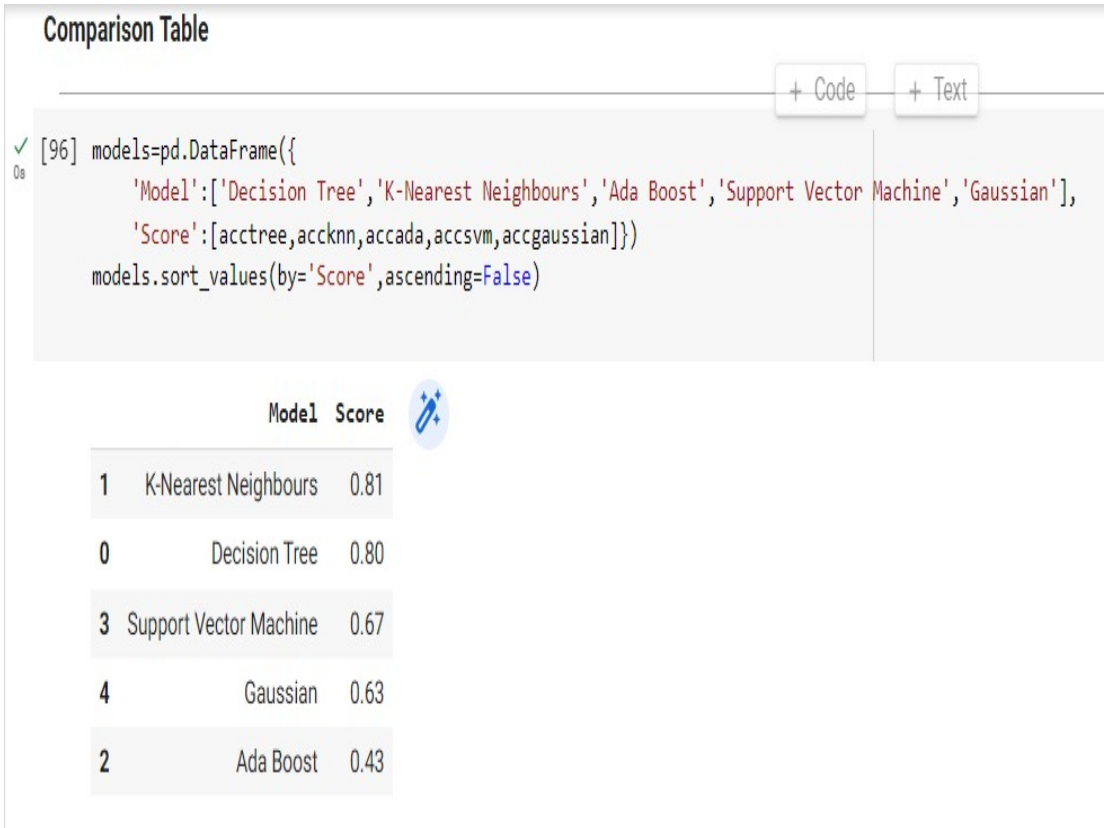
```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
cm = confusion_matrix(y_val, preds_valid)
print(cm)
print(classification_report(y_val, preds_valid))
```

```
[[326  94   0   0   4  16]
 [ 72 343  12   0   9   5]
 [  0   2 356  18  58   0]
 [  0   0   9 439   2   0]
 [  0   5  30   6 368   0]
 [ 12   1   0   0   0 405]]
              precision    recall  f1-score   support

           1       0.80      0.74      0.77       440
           2       0.77      0.78      0.77       441
           3       0.87      0.82      0.85       434
           4       0.95      0.98      0.96       450
           6       0.83      0.90      0.87       409
           7       0.95      0.97      0.96       418

    accuracy                           0.86      2592
   macro avg       0.86      0.86      0.86      2592
weighted avg       0.86      0.86      0.86      2592
```

Output


Files
..
sample_data
output.csv
sample_submission.csv
test.csv
train.csv
+ Code + Text
result
0s
Id Cover_Type
0 15121 2
1 15122 2
2 15123 1
3 15124 1
4 15125 1
... ...
7866 22987 2
7867 22988 2
7868 22989 2
7869 22990 2
7870 22991 6
7871 rows × 2 columns
[76] result.to_csv("output.csv",index = False)

Comparison Table



Comparison Table

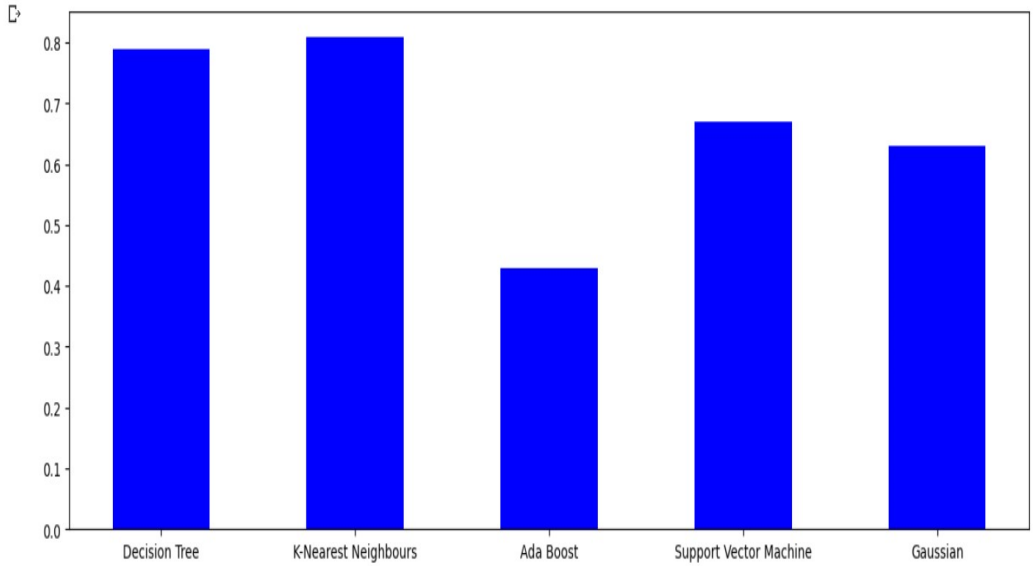
+ Code + Text

```
[96] models=pd.DataFrame({
         'Model':['Decision Tree','K-Nearest Neighbours','Ada Boost','Support Vector Machine','Gaussian'],
         'Score':[acctree,acckknn,accada,accsvm,accgaussian]})
     models.sort_values(by='Score',ascending=False)
```

	Model	Score
1	K-Nearest Neighbours	0.81
0	Decision Tree	0.80
3	Support Vector Machine	0.67
4	Gaussian	0.63
2	Ada Boost	0.43

ComparisonTable Graph

```
data = {'Decision Tree':0.79,'K-Nearest Neighbours':0.81,'Ada Boost' :0.43,'Support Vector Machine':0.67,'Gaussian':0.63}
models = list(data.keys())
score = list(data.values())
plt.figure(figsize=(15,5))

plt.bar(models,score, color ='blue',width=0.5)
plt.show()
```



Execute the Code

VISHNU
UNIVERSAL LEARNING

THANK YOU