FOREST COVER TYPE PREDICTION

USING CATOGRAPHIC VARIABLES TO CLASSIFY FOREST CATEGORIES

Done By:
N.Pranitha.

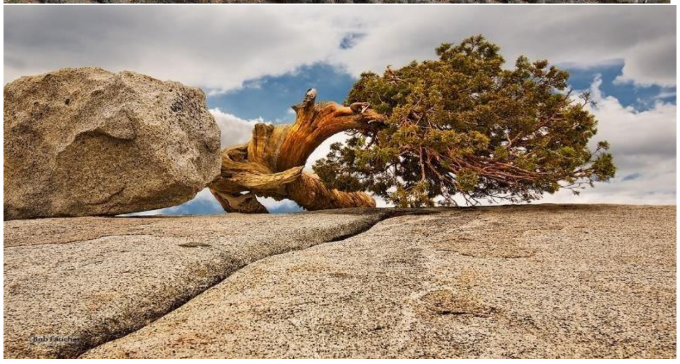
ABSTRACT

Ensemble learning approach methods find application in solving various problems ranging from a bit harder to severe complexity problems. These models in Machine Learning are known for their better predicting results for any given large data sets. Here in the project we worked on various ensemble learning approaches for solving problem on "Identifying Forest Cover". The dataset for the given problem comprises of some features collected on the forest cover and also explaining the fit of various models for the given problem.

Predict the forest cover type (the predominant kind of tree cover) from cartographic variables. The data is in raw form and contains binary columns of data for qualitativeʋ independent variables such as wilderness areas and soil type. This study area includes four wilderness areas located in the Rooseveltʋ National Forest of northern Colorado. These areas represent forests with minimal human-caused disturbances, soʋ that existing forest cover types are more a result of ecological processes rather than forest .

I.Introduction

We would like to introduce the problem in this section. In this experiment we will be analysing various machine learning algorithm on Forest Cover Type. We need to predict the cover type using the given data. We will also analyse various parameters of the applied algorithm and view the effect on the dataset. Parameter tuning using Grid search is done. The dataset is asking one to predict the type of forest cover from given cartographic variables. The actual forest cover was modelled into a 30 by 30-meter cell and was issued by USFS. The study includes four different types of wilderness areas and 7 different types namely Spruce, Lodgepole Pine, Ponderosa Pine, Cottonwood, Aspen, Douglas-fir, Krummholz.



I. Misdescriptions of dataset (Data Exploration)

The data is partially raw format where some of the variables are in binary as wilderness areas and soil type. The training set has a total of 15120 observations containing both the features and cover type whereas the test set contains only the features. It has a total of 565892 observations.

The following are the data fields:

- Elevation - Elevation in meters

- Aspect - Aspect in degrees azimuth
- Slope - Slope in degrees
- Horizontal_Distance_To_Hydrology - Horz Dist to nearest surface water features
- Vertical_Distance_To_Hydrology - Vert Dist to nearest surface water features
- Horizontal_Distance_To_Roadways - Horz Dist to nearest roadway
- Hillshade_9am (0 to 255 index) - Hillshade index at 9am, summer solstice
- Hillshade_Noon (0 to 255 index) - Hillshade index at noon, summer solstice
- Hillshade_3pm (0 to 255 index) - Hillshade index at 3pm, summer solstice
- Horizontal_Distance_To_Fire_Points - Horz Dist to nearest wildfire ignition points
- Wilderness_Area (4 binary columns, 0 = absence or 1 = presence) - Wilderness area designation
- Soil_Type (40 binary columns, 0 = absence or 1 = presence) - Soil Type designation
- Cover_Type (7 types, integers 1 to 7) - Forest Cover Type designation

The wilderness areas are:

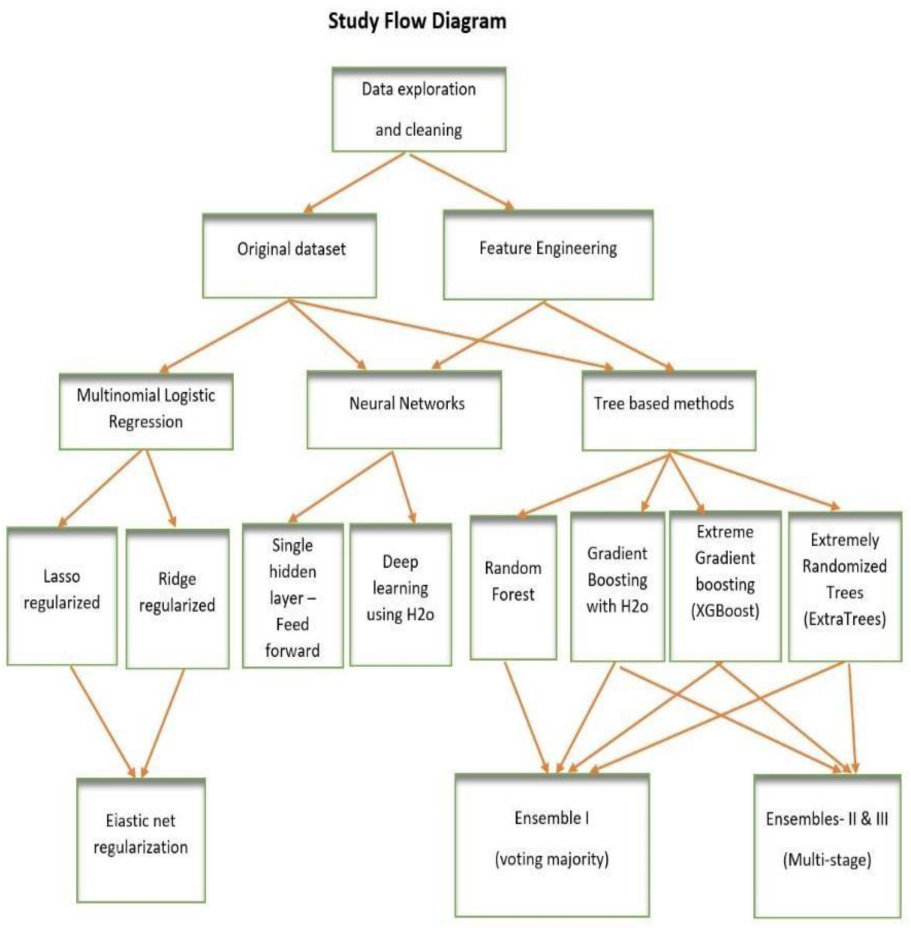
1-Rawah Wilderness Area

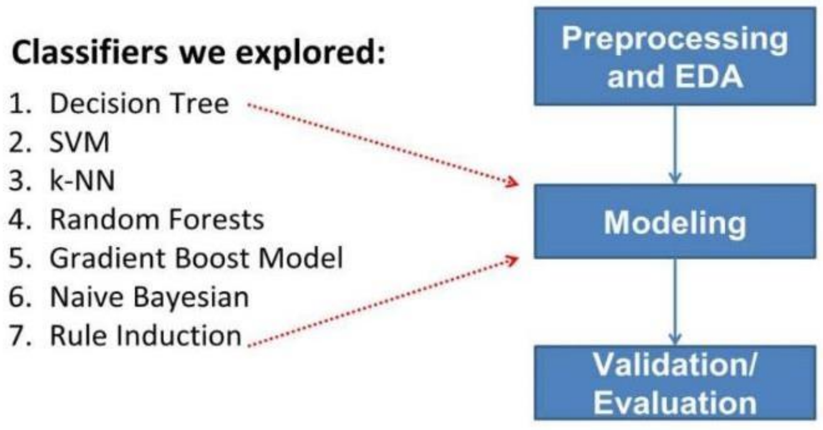
2-Neota Wilderness Area

3-Comanche Peak Wilderness Area

4- Cache la Poudre Wilderness Area

II. Method of approach


Study Flow Diagram
Data exploration and cleaning
Original dataset
Feature Engineering
Multinomial Logistic Regression
Neural Networks
Tree based methods
Lasso regularized
Ridge regularized
Single hidden layer – Feed forward
Deep learning using H2o
Random Forest
Gradient Boosting with H2o
Extreme Gradient boosting (XGBoost)
Extremely Randomized Trees (ExtraTrees)
Eiastic net regularization
Ensemble I (voting majority)
Ensembles- II & III (Multi-stage)



The data is already clean and there are no null values. We applied Random Forest Classifier algorithm as there are seven types of forests. We have got the initial accuracy of 0.859. In the next step we have calculated the feature importance. We have also dropped the features which contributed less to the accuracy. Now we got an accuracy of 0.862 after re-classifying the data again. We have added few new attributes using the already existing attributes. Calculating the accuracy score again, we got it as 0.90. We tried using boosting algorithms as Adaboost which gave us an accuracy of 0.91.

About Extra Tree Classifier:

It is similar to RandomForestClassifier only difference lies when choosing variables at a split, samples are drawn from the entire training set instead of a bootstrap sample of the training set. Computationally it is cheaper than RandomForest but can grow much bigger trees. It is a kind of descision tree wherein multiple weak learners are joined together to form a tree which gives a better classification. We calculated the n_estimators based on the GridSearch results.

About Decision Tree:

Like the name decision tree suggests, we can think of this model as breaking down our data by making decisions based on asking a series of questions. To divide the data into two sets, we need to have a function which divides the data at every node into two sub parts and this process continues.

There are 3 splitting criteria which are entropy,gini index and classification error

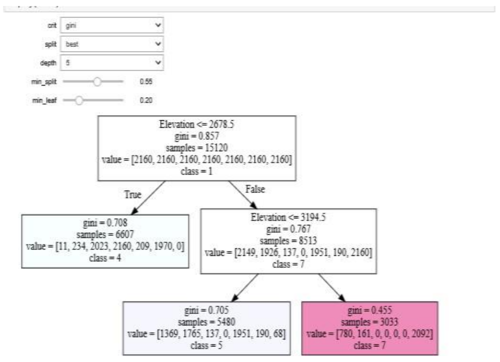
This is the formula for entropy

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t)$$

Entropy is maximal when the class distribution is equal and minimum when the class distribution is very unequally divided. Entropy criterion attempts to maximize the mutual information in the tree.

This is the formula for gini index

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(-p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2$$



crit: gini
split: best
depth: 5
min_split: 0.55
min_leaf: 0.20

It can be seen that normal decision tree was not even able to identify all the classes

Thus, many other essemble learning approaches were applied. But classification error can most of the times give same values for differently distributed data. So, entropy and gini are preferred over classification error

For the test data, the final accuracy obtained is 0.8

III. Packages:

import pandas as pd

import numpy as np

import matplotlib as pl

import matplotlib.pyplot as plt

import seaborn as sns

IV. Alternatives

Alternatives can be a voting classifier with more than ensemble approaches, Using Feature engineering and training with weights. Also, Ensembling and Stacking can improve the test results.

V. Conclusion:

The Accuracies we got for each model are:

KNN Classifier-83.9%

Naive Bayes-60.1%

Random Forest-86.4%

Gradient Boosting-84.5%

We got the highest accuracy for random forest algorithm.

VI. Future Scope:

Semi Supervised Learning Methods to increase the training data size.

Using two-way classification approaches to distinguish between majority class groups with minority.

Feature Engineering using principal component Analysis.

Apply of advanced classifiers and boosting methods.

VII. References:

1. G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental feedforward networks with arbitrary input weights," in Technical Report ICIS/46/2003, (School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore), Oct. 2003.
2. C.-C. Chang and C.-J. Lin, "LIBSVM - a library for support vector machines," in http://www.csie.ntu.edu.tw/~cjlin/libsvm/, Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, 2003.

3. Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in International Conference on Machine Learning, pp. 148-156, 1996.

4. R. Collobert, S. Bengio, and Y. Bengio, "A parallel mixtures of SVMs for very large scale problems," Neural Computation, vol. 14, pp. 1105-1114, 2002.