# [Classification]: Prediction of Forest Cover Type

Prerna Singh
2013149

Virender Singh
2013117

## Abstract

*In this report, we build various classifiers to predict the forest cover type and compare their performances. This report gives a detailed description of the classifiers and ensembling techniques used to improve the accuracy of prediction.*

## 1. Introduction

Accurate natural resource information is very vital to any land management agency. Forest Cover Type is one basic characteristic recorded by such agencies. Cover type data may be recorded by a field personnel or by some sensors. But these methods may prove to be very expensive and time consuming. Hence, a predicting model can be very useful for obtaining data.

So, in this project we build a Classification model which can predict the Forest Cover Type. We use different classification models like SVM,ANN, Random Forest and KNN to improve the accuracy of prediction.
So our aim is that given a data set having forest features and forest type, we want to predict for a test data set what will be it's forest type.

### 1.1. Related Work

The forest cover type prediction problem has been explored by Kevin Crain and Graham Davis[5] from Stanford University as their course project on a smaller subset of this data. They used multi class SVM for prediction. They also performed Principal Component Analysis and k-mean clustering for visualization. In our project, we not only did multi class SVM but also achieved higher accuracy using Random Forest, KNN and neural network classifiers.

### 1.2. Dataset

In this project, we used UCI's Forest Covertype data set. This data set contains tree observations from four areas of the Roosevelt National Forest in Colorado. All observations are cartographic variables (no remote sensing) from 30 meter x 30 meter sections. The data set details are shown in table I. The feature vector included Elevation(in meters), Aspect, Slope, Horizontal Distance To Hydrology, Horizontal Distance To Roadway, Hillshade(at 9am, noon and 3pm), Horizontal Distance To Fire Points, Wilderness Area and Soil Type. The wilderness is described using 4 columns and soil type is represented using 40 columns, hence the total length of input feature is 54. The output consists of 7 classes corresponding to different types of forest cover refer to Figure 1.

### 1.3. Evaluation

The size of the data set is 0.1 million. We divided these into training and testing data sets. Training size = 70000 Testing size = 30000. The classifiers were evaluated using accuracy and confusion matrices.

**Table 1: Dataset description**

| Data field | Description |
|---|---|
| elevation | Elevation in meters |
| aspect | Aspect in degrees azimuth |
| slope | Slope in degrees |
| horz_hydro | Horz Dist to nearest surface water features |
| vert_hydro | Vert Dist to nearest surface water features |
| horz_road | Horz Dist to nearest roadway |
| hillshade_9am | Hillshade index at 9am, summer solstice (0 to 255 index) |
| hillshade_noon | Hillshade index at noon, summer solstice (0 to 255 index) |
| hillshade_3pm | Hillshade index at 3pm, summer solstice (0 to 255 index) |
| horz_fire | Horz Dist to nearest wildfire ignition points |
| wild | Wilderness area designation (4 binary columns) |
| soil_type | Soil Type designation (40 binary columns) |

Figure 1. Attributes Description

Above table shows the description of various features present in Forest Cover Type Database.

**Figure 2** Represents the number of entries per type of Forest Cover in the Database.

## 2. Pre-Processing

### 2.1. Data Analysis

Following steps were done in order to pre-process and analyze the data:

- CSV file was read and each column was given suitable title.

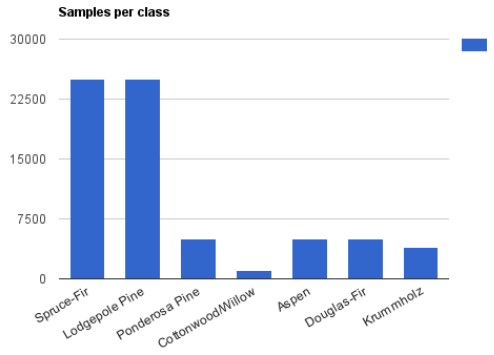

Figure 2. Distribution of Cover Types

- After checking the correlation of various columns with target column. We could see the essential features of the data for prediction.
- Normalization is done to have the same range of values for each of the inputs to the ANN model. This can guarantee stable convergence of weight and biases.

## 3. Methodology

After preprocessing the data set, we performed the following steps:

- Trained the model using some simple models such as
  i) KNN Classifer
  ii)SVM classifier.
- Trained the model on complex models such as
  i)Random Forest Classifier
  ii)Artificial Neural Network(ANN).
- Found best parameters by doing grid search and checked for over fitting on training data using k-fold cross validation.
- Accuracy and confusion matrices were computed for test data set on each of the above models. The results were compared.
- Used ensembling techniques(stacking and boosting) to improve the results.

The details about the implementation of the above steps is presented in the sections that follow.

### 3.1. KNN Classifier

kNN classifier finds the k nearest neighbors of any data entry and does voting in between those labeled neighbors to assign the label to that data entry.
We implemeted the sklearn's inbuilt KNN classifier. Our model used K=3. The accuracy obtained on test data was 93.39%.

### 3.2. SVM classifier

The multi-class Support Vector Machine implemented in this report was imported from the sklearn library. This SVM uses the one vs. one approach to multi-class classification. Our multi class SVM made use of the Radial Basis Function (RBF) Kernel.
To improve the accuracy of the multi-class SVM with respect to our data set, we wanted to optimize two hyperparameters of our model: C and $\gamma$ (RBF Kernel coefficient). In order to do so, we used both grid search and cross validation over the hyperparameters. We found the best results at C = 10000 and $\gamma$ = 0.05. The accuracy obtained on the test data is 83.5%

### 3.3. Random Forest Classifier

Random Forest Classifier is a collection of decision trees which assigns the label to a particular entry as per the majority by decision trees.
We imported the Random Forest classifier provided by sklearn. Our model used number of trees as 100 and we set the max feature to 'auto'. The accuracy on test data was 96.59%.

### 3.4. Artificial Neural Network

Although not proposed earlier we implemented ANN as part of additional work. We used keras which is a Deep Learning Library for Theano and Tensorflow.
Our model had 2 hidden layers using tanh as activation function where each layer had 100 neurons. The batch size(no of samples per gradient update) was 128 and no. of epochs(to train the model) were 100. We used Stochastic Gradient Descent as optimizer. The final layer used softmax for generating the predicted class.The accuracy obtained on test data was 82.58%.

Table 1. Accuracy from different classifiers

| Classifier | Accuracy |
|---|---|
| SVM | 83.5% |
| Random Forest | 96.59% |
| KNN Classifier | 93.39% |
| Neural Network | 82.58% |

## 4. Ensemble Approaches

Ensembling is a powerful technique to improve the accuracy of the model. In this method, we combine different models to form one single model that improves accuracy and results. In our project, we explored the following ensembling techniques:

### 4.1. Boosting

Boosting is a general ensemble method that creates a strong classifier from a number of weak classifiers. This

is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added. We explored AdaBoosting technique in this project.
**AdaBoosting** - In order to implement this, we created a model using sklearn's Random Forest Classifier. We then used AdaBoostClassifier to improve the accuracy. The accuracy value obtained on test data is 95.6 %.

### 4.2. Stacking

Building multiple models (typically of differing types) and supervisor model that learns how to best combine the predictions of the primary models. Here we use a learner to combine output from different learners. This can lead to decrease in either bias or variance error depending on the combining learner we use.
**Implementation**:
We took the outputs from the Random Forest classifier and KKN classifier and stacked them using voting technique.

**Voting** - This classifier picked the majority label as the final label. This improved the accuracy to 96.94%.

Table 2. Metrics for Ensembling

| **Ensembling** | **Accuracy** |
|---|---|
| Stacking(Voting) | 96.94% |
| AdaBoosting | 95.68% |

## 5. Challenges

During the course of the project, we faced the following challenges:

- Grid search took a lot of time to find the best parameters(2 days in our case).
- It took a lot of time to train multiclass SVM and Neural Network on this large dataset.

## 6. Results

Table 3. Precision Values from Classifiers

| Precision | KNN | Random Forest | SVM |
|---|---|---|---|
| Class 1 | 0.89 | 0.99 | 0.73 |
| Class 2 | 0.90 | 0.99 | 0.64 |
| Class 3 | 0.95 | 1 | 0.81 |
| Class 4 | 0.93 | 1 | 0.90 |
| Class 5 | 0.95 | 0.99 | 0.92 |
| Class 6 | 0.93 | 0.99 | 0.87 |
| Class 7 | 0.96 | 0.99 | 0.96 |

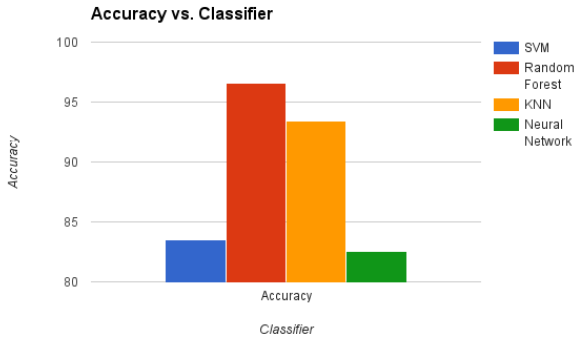


Figure 3. Accuracy of various Classifiers

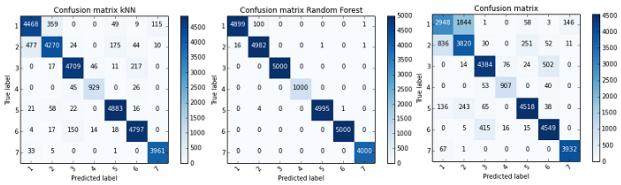


Figure 4. Confusion Matrices of various Classifiers

Table 4. Recall Values for Different Classifiers

| Recall | KNN | Random Forest | SVM |
|---|---|---|---|
| Class 1 | 0.89 | 0.97 | 0.58 |
| Class 2 | 0.85 | 0.99 | 0.76 |
| Class 3 | 0.94 | 1 | 0.87 |
| Class 4 | 0.92 | 1 | 0.90 |
| Class 5 | 0.97 | 0.99 | 0.90 |
| Class 6 | 0.95 | 1 | 0.90 |
| Class 7 | 0.99 | 1 | 0.98 |

Table 5. F1 Measures

| | KNN | Random Forest | SVM |
|---|---|---|---|
| F1 Measure | 0.93 | 0.99 | 0.83 |

All the results stated are achieved by taking 0.1M samples from the data set. These were further divided into training and testing. From the results stated in the above tables, following can be concluded:

- The accuracy obtained by Random Forest Classifier is maximum as compared to other classifiers used. Hence, we can say that Random Forest classifier works best for our data set.
- Based on ensembling results, we may conclude that stacking(voting) the results of Random Forest and KNN classifier works better for our dataset. The accuracy of KNN improved from 93.39% to 96.94%.
- From Table 3, we can see that precision for each class is very high for Random Forest classifier. This is also

verifies the high accuracy results obtained from Random Forest.

## 7. References


[1] Code can be found : **here**

[2] Ensemble Methods Foundations and Algorithms:Zhi-Hua Zhou

[3] https://www.kaggle.com/c/forest-cover-type-prediction

[4] **DataSet :** *http://mlr.cs.umass.edu/ml/datasets/Covertype*

[5] ttp://cs229.stanford.edu/proj2014/Kevin%20Crain,%20Graham %20 Davis,%20Classifying%20Forest%20Cover%20Type%20using%20Cartographic%20Features.pdf

[6]http://scikit-learn.org/stable/modules/ensemble.html

[7]http://machinelearningmastery.com/tutorial-first-neural-network-python-keras/


## 8. Contribution

The work division was as follows:

- Virender - Implemented KNN and SVM classifiers. Also explored AdaBoosting technique for ensembling.
- Prerna - Implemented Random Forest and Neural Network. Performed stacking ensembling technique.
- Results and report were complied by both.