Assignment-1: Forest Cover Type Prediction

Rahul Tallamraju(201450841) Saket Maheswari(201407670)*

rahul.t@research.iiit.ac.in saket.maheshwary@research.iiit.ac.in

KaggleTeamName: RTSM, Usernames: SaketMaheshwary, RahulTallamraju
Primary Account: SaketMaheshwary
Highest Rank: 57

Problem Statement

In this competition you are asked to predict the forest cover type (the predominant kind of tree cover) from strictly cartographic variables (as opposed to remotely sensed data). The actual forest cover type for a given 30 x 30 meter cell was determined from US Forest Service (USFS) Region 2 Resource Information System data. Independent variables were then derived from data obtained from the US Geological Survey and USFS. The data is in raw form (not scaled) and contains binary columns of data for qualitative independent variables such as wilderness areas and soil type.
This study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. These areas represent forests with minimal human-caused disturbances, so that existing forest cover types are more a result of ecological processes rather than forest management practices.

I. Introduction

The given problem is a multi-class classification problem. An integer classification model needs to be developed/identified in order to predict the forest cover type. Each observation in the train and test datasets is a 30m x 30m patch of forest. There are seven types of forest covers presented, which represent the seven classes in the classification problem. The training set has a total of 15120 observations containing both features and cover-types. Prediction must be performed on the test set which has 565892 observations. There are 10 numeric(quantifiable) attributes such as slope, elevation, aspect .. etc . A total of 44 binary columns are present. Four columns indicate the presence or absence of a specific widerness type and the other 40 represent the presence or absence of a specific soil type. Several approaches to process, model and evaluate the data have been explored. The next section provides a brief overview of the approach used to solve the problem.

II. Overview of Approach

Our approach to this problem involves the following steps:

1. Data Pre-processing

   This step involves statistical analysis of each attribute in the data, visualization of the attributes using histograms and scatter plots, data transformations and feature engineering. The statistical and visual analysis help in the understanding of intrinsic details of the data. They also help in choosing and engineering the right groups of features for good classification.

*Kaggle Team Name: RTSM

2. Modeling the Problem

   The pre-processed data is divided into training and validation sets and fed to data mining algorithms. 5-fold cross validation is performed until satisfactory accuracies are reported. Various Data Mining algorithms were explored and implemented in order to determine which algorithm best models the data.

3. Evaluation

   In this step the test data is fed to the trained data mining algorithm in order to predict the forest cover types for the test observations

4. Submission

   A csv file containing the test ID number and the predicted Forest Cover ID of all test observations is submitted on kaggle.com to obtain a rank and a score for the model.

III. Data Analysis and Pre-Processing

All the pre-processing steps used, were applied to both training and test data attributes.
The data analysis and some preprocessing steps employed for the given data are as described below:

I. Statistical Analysis for Attribute Selection

Statistics of attributes were first extracted to understand the intrinsic details of data. The mean and standard deviation for each attribute of training data were observed. Attributes with a relatively low standard deviation w.r.t other attributes were not considered for classification(for some experiments) This helped in significant improvement in classification accuracy. Table 1 indicates the mean and standard deviation of the given attributes. Table 2 indicates the selected attributes from the training data based on relative standard deviations w.r.t mean and other standard deviation of other attributes.

Table 1: *Mean & Standard Deviation of Features*

	Features									
Measure	Elevation	Aspect	Slope	HDTH	VDTH	HDTR	HS9	HSN	HS3	HDFP
Mean	2749	156	16.5	227	51	1714	212	218	135	1511
std	417	110	9	210	61	1325	30	22	45	1099

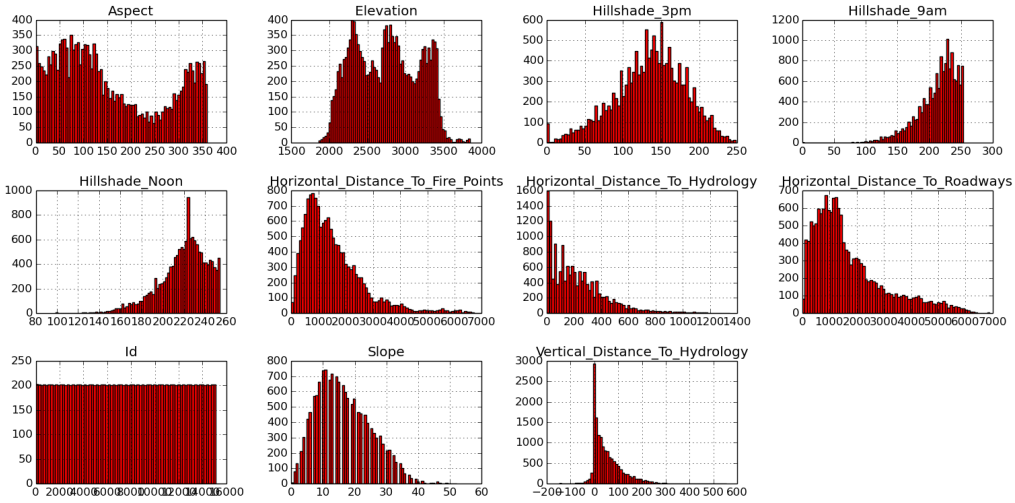
Table 2: *Selected Features*

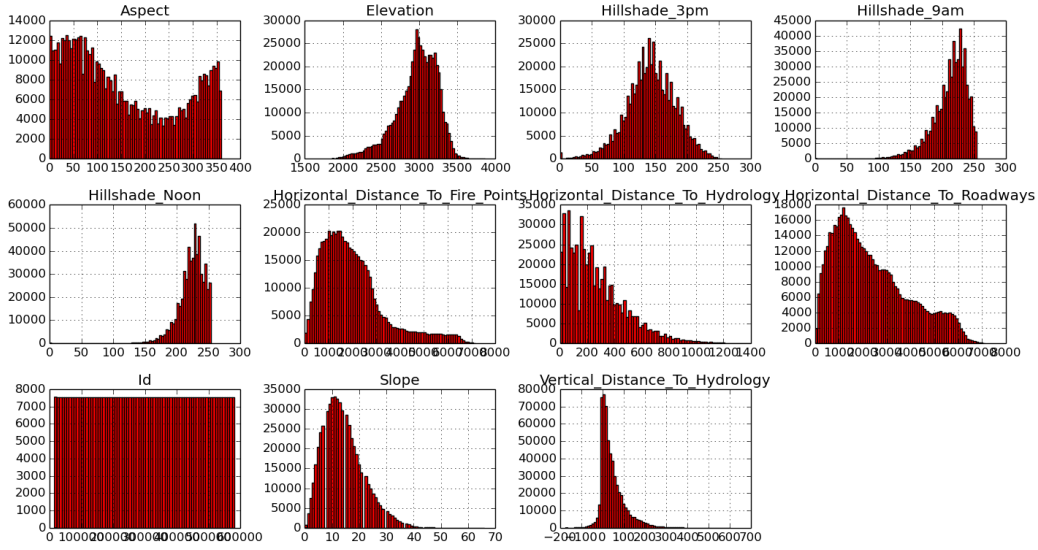
Features								
Elevation	Aspect	HDTH	VDTH	HDTR	HS3	HDFP	Wilderness(1-4)	SoilType(1-40)

II. Visual Analysis

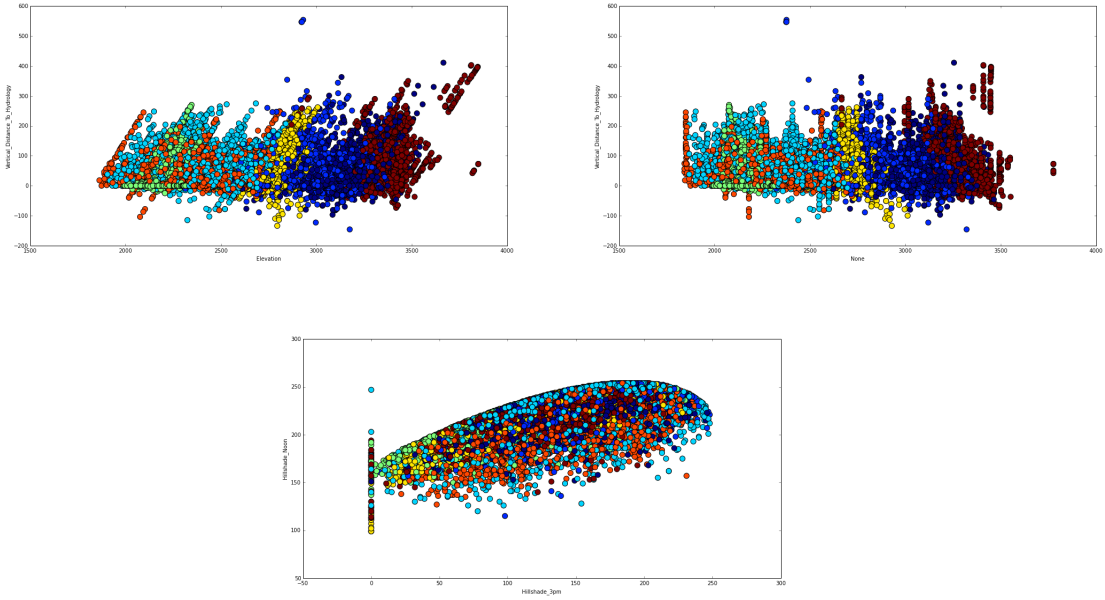
Features were visualized using histograms in order to get more insight into the values of each attribute. We observe from the given figures the distribution of values of different attributes in

training data. This also helps in feature engineering. For example, 'Vertical Distance To Hydrology' has negative values a new binary feature can be engineered. This feature can indicate if a value is positive or negative. The below figures provide histograms of features of training and test data.





Additional Visualizations of features were done using scatter-plots of features individually or in-the-form-of feature-1 vs feature-2. Below are a few examples of such scatterplots:



The above plots help in feature engineering, feature selection and denoising. Plot-(1,1) suggests that the values of both Elevation and VDTH are similar and can be linearly combined to give a new feature. Plot-(1,2) suggests that (Elevation - VDTH) (vs VDTH)is a very good feature as it almost has clear distinct bands of values for each class, and hence is a very good discriminator of the classes. Plot (2,1) suggests that Hillshade3pm has many zero values and hence those have to be replaced by mean, median or mode.

III. Feature Transformation

There a total of 54 features of which 44 are binary features. Only one of 40 columns of soil is present for a given observation, similarly only one of four wilderness types is present for a given observation. To reduce the dimensionality caused by the 42 zero attribute values, the following method was tested. The four wilderness columns and 40 soil columns were each observation were considered as a binary string and converted to decimal. Therefore we now have 12 attributes in total. This method provided satisfactory results and there was a considerable increase in Kaggle rank. The feature transformation of Wilderness is as shown in Table 3. Soil Types are also mapped in a similar way.

Table 3: *Eg: Mapping of Wilderness*

Features	
Binary	Decimal
0001	1
0010	2
0100	4
1000	8

IV. Feature Engineering

The features were engineered based on the visualized dependency and similarity in feature values. Various linear combinations of attributes led to creation of more attributes. Engineering features appended with all attributes of training data gave us the highest rank for our assignment in kaggle. More detailed information on all the list experiments performed have been provided in the Experiments and rResults section. A list of engineered features are as given below

Table 4: *Eg: Mapping of Wilderness*

Features					
FeatureName	Feature-1	Feature-2	Operation	dtype	std
Aspect2	NULL	NULL	Map(0,180)	int64	-
Highwater	VDTH	NULL	c1>0 | c1<0	binary	-
EVDtH	Elevation	VDTH	c1 - c2	int64	414
EHDth	Elevation	HDTH	c1-0.2*c2	int64	402
HydroFire1	HDTH	HDTFP	c1 + c2	int64	215
HydroFire2	HDTH	HDTFP	c1 - c2	int64	115
HydroRoad1	HDTH	HDTR	c1 + c2	int64	138
HydroRoad2	HDTH	HDTR	c1 - c2	int64	129
FireRoad1	HDTFP	HDTR	c1 + c2	int64	209
FireRoad2	HDTFP	HDTR	c1 - c2	int64	884

Listing the features based on their importance on a scale of 0 to 1 based on 'gini importance' a function from sklearn : feature-importances-

Table 5: *Eg: Mapping of Wilderness*

Features	
FeatureName	ImportanceValue
EHDtH	0.097486
Elevation	0.096896
EVDtH	0.092564
WildernessArea4	0.046237
FireRoad1	0.033677
HydroRoad2	0.032912
HorizontalDistanceToRoadways	0.031300
HydroRoad1	0.030773
DistansetoHydrolody	0.028715
HorizontalDistanceToHydrology	0.027839

IV. Data Mining Algorithms Used

Different algorithms used in experimentation are:

1. K-Nearest Neighbor
2. Gaussian Naive Bayes
3. Support Vector Machines
4. Gradient Tree Boosting
5. Decision Trees
6. Random Forests
7. Extremely Randomized Trees

As a simple rule for comparison, all algorithms were initially trained using all attributes of training data.
5-fold cross validation was performed on the data to identify the accuracy of each classifier. Further experiments of feature selection, feature engineering and transformation were applied to the top performing classifier.

I. Supervised K-Nearest Neighbor

The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be a user-defined constant (k-nearest neighbor learning). The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice.
Table displaying experimental results with 2,3 and 4 nearest neighbors is as given below.

Table 6: *KNN results*

	k-values		
Measure	k=2	k=3	k=4
Accuracy	70.8	71.2	69.3
Precision	0.81	0.81	0.80
f1-score	0.81	0.81	0.80

II. Naive Bayes

Supervised learning algorithm based on applying Bayes theorem with the naive assumption of independence between every pair of features. Given a class variable y and a dependent feature vector x1 through xn, Bayes theorem states the following relationship:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1,\ldots x_n|y)}{P(x_1,\ldots,x_n)}$$

Gaussian Naive Bayes algorithm was used for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Table 7: *Gaussian Naive Bayes results*

Measure	Value
Accuracy	46
Precision	0.46
f1-score	0.47

III. Support Vector Machines

Support vector machines (SVMs) are a set of supervised learning methods used for classification. Effective in high dimensional spaces and in cases where number of dimensions is greater than the number of samples. Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.Different Kernel functions can be specified for the decision function.

SVM with a linear and radial basis function kernels were used for classification.

Table 8: *SVM results*

kernels		
Measure	Linear	RBF
Accuracy	28.1	14
Precision	0.29	0.31
f1-score	0.23	0.14

IV. Decision Tree

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Table 9: *Decision Tree results*

Measure	Value
Accuracy	71
Precision	0.77
f1-score	0.77

V. Gradient Tree Boosting

Gradient Tree Boosting or Gradient Boosted Regression Trees (GBRT) is a generalization of boosting to arbitrary differentiable loss functions. GBRT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems. Gradient Tree Boosting models are used in a variety of areas including Web search ranking and ecology. Supports both binary and multi-class classification.

Table 10: *Gradient Tree Boosting results*

	N Trees		
Measure	N=100	N = 300	N = 500
Accuracy	75	75	75
Precision	0.82	0.82	0.82
f1-score	0.82	0.83	0.82

VI. Random Forests

In random forests, each tree in the ensemble is built from a sample drawn with replacement from the training set. In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model.

Table 11: *Random Forests results*

	N Trees		
Measure	N=100	N = 300	N = 500
Accuracy	78	79	79
Precision	0.86	0.86	0.85
f1-score	0.86	0.86	0.86

VII. Extremely Randomized Trees

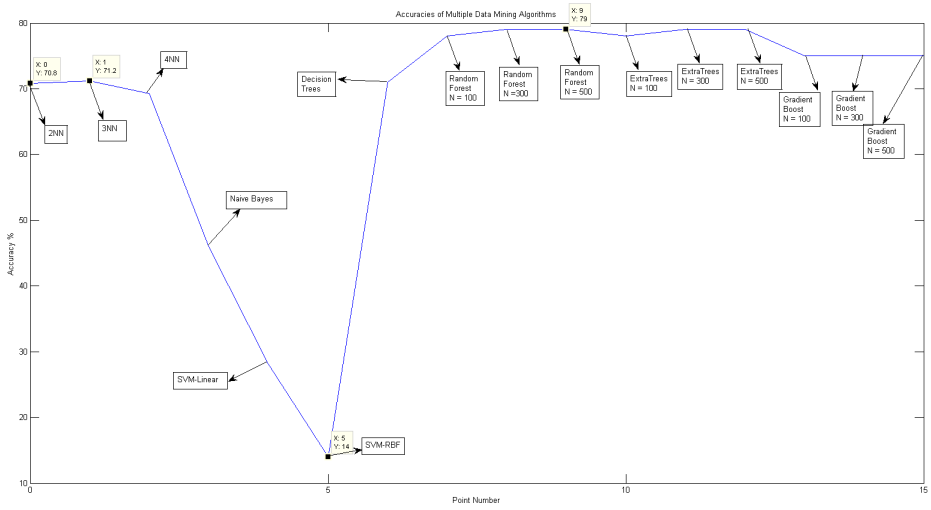
In extremely randomized trees, randomness goes one step further in the way splits are computed. As in random forests, a random subset of candidate features is used, but instead of looking for the most discriminative thresholds, thresholds are drawn at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting rule. This usually allows to reduce the variance of the model a bit more, at the expense of a slightly greater increase in bias.

Table 12: *ExtraTrees results*

	N Trees		
Measure	N=100	N = 300	N = 500
Accuracy	78	79	79
Precision	0.86	0.86	0.86
f1-score	0.86	0.86	0.86

V. Experiments and Results

The below figure shows the plot of accuracy graph for various classifiers discussed in the previous section.



It is observed from the figure and f1-scores, from the previous section, that the best performing classifiers are Random Trees and Extra Trees Classifiers. The Extra Trees Classifier with 500 trees yielded a rank of **491** on Kaggle. Random Forests yielded **896** (accuracy=87%). Therefore Extremely Randomized Trees Classifier has been used for further experimentation.

.1 Experiment 1

Using the reduced set of features from statistical analysis section, and reducing the total number of features to 12 features (using feature transformation of the wilderness and soil types) the Extra trees classifier was trained. **With 500 trees the kaggle rank obtained was 441**.

.2 Experiment 2

Appending engineered features (described in the feature engineering subsection) to the reduced set of features we obtained an **a rank of 150 on kaggle**.

.3 Experiment 3

Using features based on their importance rank as given in feature engineering section, **a kaggle rank of 65 was obtained**.

.4 Experiment 4

Using engineered features appended with all the features provided in training data (without applying any feature transformation or feature selections) we got a **rank of 57 on kaggle**.

Table 13: *Final results using 500 trees in Extra Trees Classifier*

Final Accuracy	
Measure	Value
Accuracy	82.2%

VI. Conclusions and Future Improvements

With **Extra Trees classifier, N=500 trees** our model obtained **a rank of 57 on kaggle.com**.There remains a lot of scope for improvement, some ideas for further improvements could be creating more engineered features, selecting features which have high inter-class variance and low intra-class variance. We could also construct a new Ensemble Classifier of 3 best performing classifiers in this problem and use a majority voting scheme to obtain correct prediction.

References

[1] Kaggle. 2014. Kaggle. [ONLINE] Available at: https://www.kaggle.com/c/forest-cover-type-prediction. [Accessed 05 September 15].

[2] scikit-learn. 2014. scikit-learn. [ONLINE] Available at: http://scikit-learn.org/stable/documentation.html. [Accessed 05 September 15].