Seeing the Forest for the Trees: Utilizing Random Forest Machine Learning to Classify Forest Coverage

Peter McKay and Daniel Walinsky
Department of Computer Science and Information Science, University of Oregon
Eugene, Oregon
{pem,dwalinsky}@cs.uoregon.edu

ABSTRACT

problem, motivation, solution, takeaway

In this paper, we consider the problem of automatic forest coverage classification, as posed by the Kaggle competition: "Forest Cover Type Prediction." In short, we wish to construct a model capable of reliably predicting the predominant type of tree cover for a small region of forest, given a collection of cartographic variables. We create such a model by leveraging a voting body of a numnber of different classifiers. We supplement this technique by carrying out a number of feature transformations and subsequent feature scaling, making use of a number of statistical relationships discovered through judicious use of graphical data analysis.

After carrying out such modifications to the feature set, our ensemble classifer could dependably reach 80% accuracy. The structure of our classification setup implies is such that we gain some insight into a the data set, and the problem domain in general. PLACEHOLDER

1. INTRODUCTION

It is popular to model education as a process of ever-increasing specialization. Preschool acclimatizes larval humans to basic scholarly etiquette (listening, sharing, naptime), while Kindergarten begins a focus on more distinct and practical skills such as reading, writing, and simple arithmetic. high-school allows one to select a few electives, and the undergraduate degree program finally allows for the selection of the glorious Major. By the time one has reached the PhD program, one's course of study has narrowed down to a mere fraction of the breadth of earlier, sometimes a single problem.

When dealing with questions on the frontiers of science, one sometimes encounters a number of subproblems that fall within the purview of many such narrow fields. You start adding expert after expert to the team until you've got a 2-foot organizational chart and a half dozen management types trying to tell everyone what to do. Interdisciplinary research doesn't seem to scale very well.

One of the real draws for machine learning lies in allowing us to compartmentalize our need for understanding. So long as our model "understands" a problem, we can ignore a certain level of complexity and move on to the interesting part. This lets a computer scientist with a remarkable aversion to even thinking too deeply about the outdoors make a number of staggeringly correct judgemnts about forests, all from the comfort of their office.

Elevation	Elevation in meters
Aspect	Aspect in degrees azimuth
Slope	Slope in degrees
Horizontal_Distance_To_Hydrology	Meters to nearest water-body
Vertical_Distance_To_Hydrology	Meters to nearest water-body
Horizontal_Distance_To_Roadways	Meters to nearest roadway
Hillshade_9am (0 to 255 index)	Hillshade index at 9am
Hillshade_Noon (0 to 255 index)	Hillshade index at noon
Hillshade_3pm (0 to 255 index)	Hillshade index at 3pm
Horizontal_Distance_To_Fire_Points	Meters, wildfire ignition points
Wilderness_Area	Wilderness area designation
Soil_Type	Soil Type designation

Table 1: Feature List

We know a few things about a bit of forest, and an accomplished expert in the fields of ecology or geography, specializing in the climate of that forest, might be able to tell us quite a bit of information based on that starting data. In the noble tradition of computer science, we would like to avoid doing that much work, and we will apply machine learning techniques to accomplish this task. To that end, we consider the following multiclass classification problem. We construct a classifier that, when presented with a number of cartographic variables that apply to a 30–30 meter cell of forest, classifies such a cell by the predominant type of tree cover found therein.

In this paper, we utilize a multi-level ensemble method of classification. We add a number of new features, scale the feature vector appropriately, and then train a collection of different classifiers on that data. The classifiers vote to determine the classification of an example, and in some cases, pass on that vote into another layer of machines trained on a smaller subset of the training data. We accomplish these tasks using python and a series of libraries designed for data analysis and machine learning. In so doing, our model reaches approximately 80% accuracy on the Kaggle data set.

2. BACKGROUND

2.1 Problem Structure

The forest coverage classification problem clearly fits into the category of a multiclass classification problem, in which we describe a classifier that maps from a feature vector to one of a series of possible output labels. This feature vector, as outlined in table **??**, contains a mix of continuous and discrete values. The discrete features, as visualized in figures **??** and **??**, account for multiclass features encoded with the one-hot encoding method.

Given these features, we wish to classify our example with a choice from the labels below:

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

Unlike a binary classifier, it is quite easy to end up with an error rate well over 50% when dealing with a multiclass classifier.

2.2 Solution

The classification solution we assemble to solve this problem involves a number of different machine learning models. We began the project by attempting a Random Forest classifier, but eventually expanded to also make use of a Gradient Boost Machine, a Support Vector Machine, and a k-Nearest Neighbor classifier. Each of these machines has its own strengths and weaknesses, which we will discuss in the following section.

3. METHODOLOGY

3.1 Feature engineering

The first step in any machine learning problem is to take a look at the data, and see what we've gotten ourselves into. If we start by looking at the continuous variables, we observe a number of interesting correlations.

Aspect appears to be somewhat of an outlier in its shape, which makes sense when one considers the variable is given in terms of degrees: we should be working in modulo 360. There are a number of transformations that could solve this issue: we choose to simply split aspect into two different features scaled to 180°.

We know that Elevation is measured in meters, as are the 5 other "distance to nearest item of interest" features. It seems like we should be able to do something with that. In our case, we add the differences between Elevation and the Hydrology Distance features. We also integrate the horizontal and vertical distance features, for a Euclidian distance measurement. We notice there are some forest cells that appear to be nearly underwater, according to the Vertical Distance To Hydrology feature. We introduce a new boolean feature to denote that possibility.

While the graph of wilderness areas appears initially promising (look at all of those solid blocks of color!), we note that this is an artifact of the graphing process: there are only a few wilderness area types, and they seem to be spread pretty evenly among a the cover types.

The soil type graph shows us a preview of one of our more interesting discoveries, which we cover more thoroughly in the next subsection. For now, we simply observe that some cover types are more distinct from the group than others, when we consider solely the soil types.

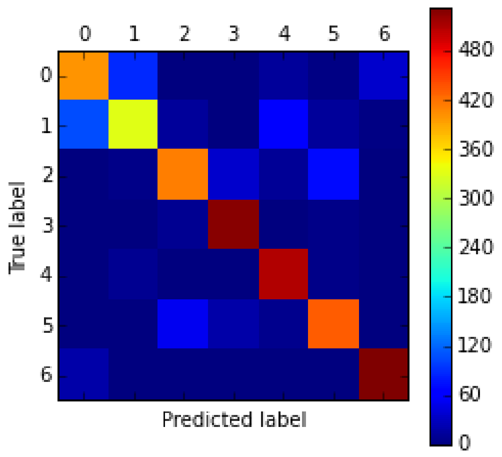

Figure 4: Confusion Matrix

3.2 final model

As we add new features and massage existing features into new shapes, we iteratively apply our classifier scheme to the feature vector to test the efficacy of those features. This, however, is not an exact science. A change that increases accuracy over some decisions could very well lead to a loss of accuracy over other decisions. In order to more fully examine the consequences of our actions, we make use of a graphical tool called a confusion matrix [**?**]. A confusion matrix, or, error matrix, is a tool suited for specifying exactly what kind of missclassifications lead to a loss of accuracy. In our case, as we see in Figure **??**, our model does a pretty good job until we start trying to differentiate between the Spruce/Fir type and the Lodgepole Pine type (and, to a lesser extent, between Lodgepol Pine and Douglas-fir).

Adding a mechanism to our ensemble to correct for this error results in a modest accuracy increase for our validation set, but a significantly greater accuracy gain for the Kaggle competition itself. If we count up the estimated classificaitons, we observe that our classifiers predict more 1s and 2s than any other cover type, by nearly an order of magnitude. This suggests that the unexpected gain in accuracy is due to a heavier preponderance of Spruce/Fir and Lodgepole Pine trees in the testing region. Informed by these features, we then tested a number of classifiers with default settings in order to

a Support Vector Machine, and a Gradient Boosting Machine on that data. We train a Random Forest classifier to settle on a list of feature weights, which we use to optimize the performance of a k-Nearest-Neighbor classifier. The GBM, SVM, k-NN classifier, thus assembled, use majority voting to determine which classification is selected for a given collection of features. If that vote indicates that we're dealing with cover type 1 or 2, we fall back to a Random Forest trained only on examples with cover types 1 and 2.

4. RESULTS

From the first attempt at a Random Tree classifier, up to our final, somewhat convoluted approach to We'll want to talk about final accuracy, about the effects of various changes we made to the model on accuracy, about

Interestingly, some of the more puzzling and impenetrable features were revealed to be the most important across all of the classifiers.

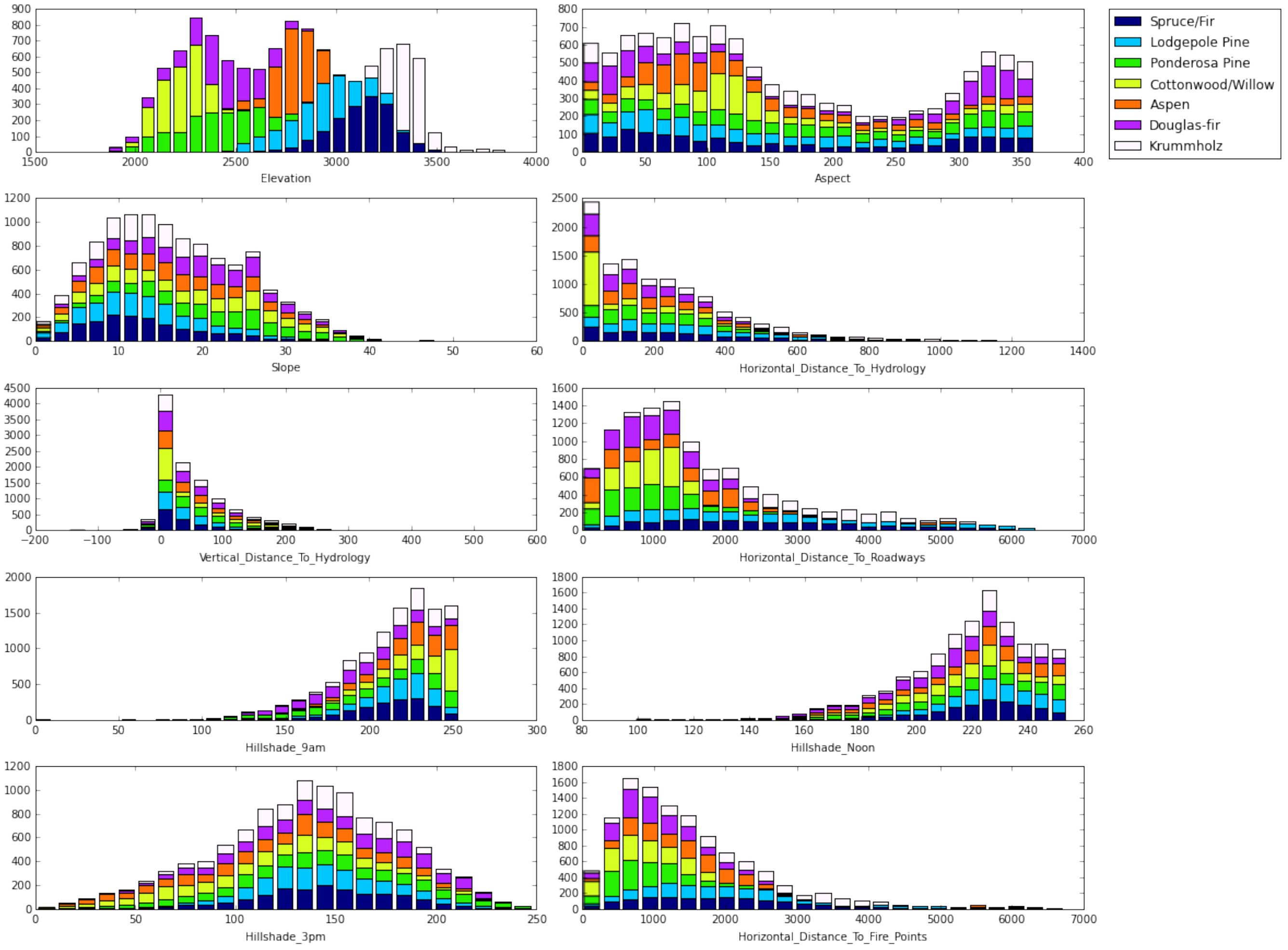

Figure 1: Continuous variables

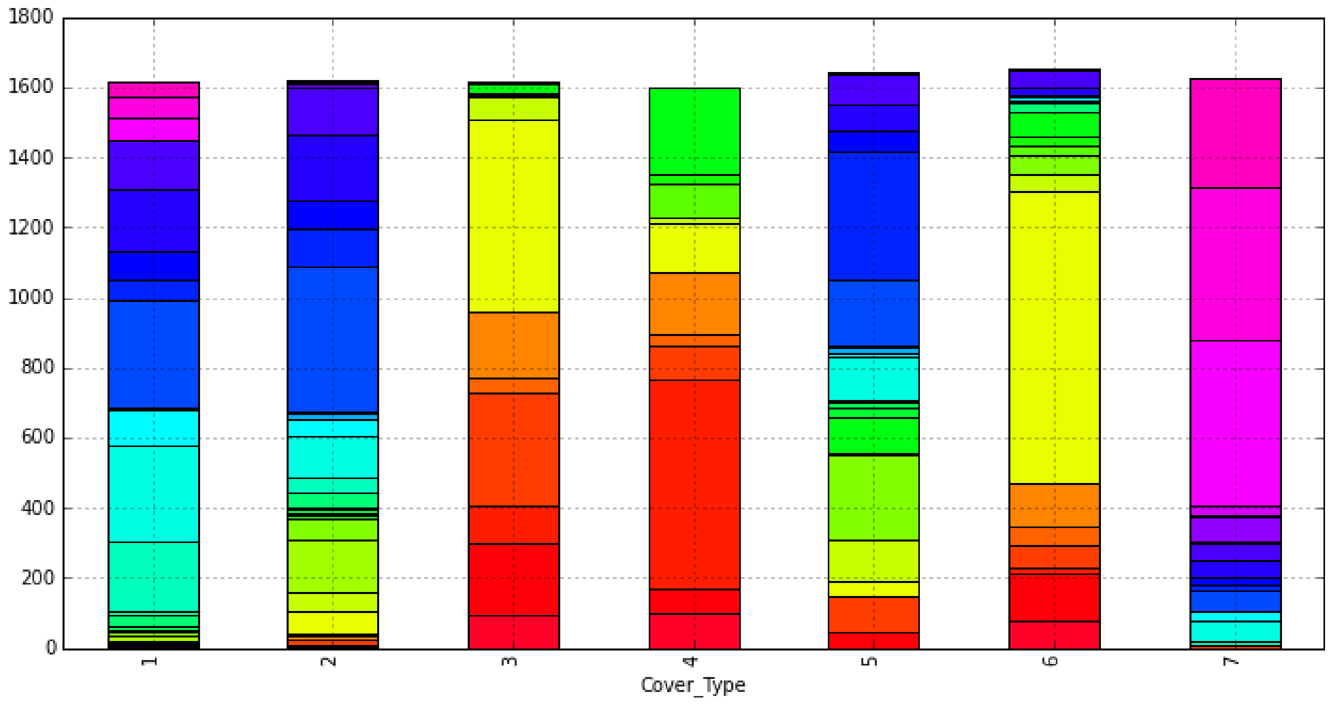

Figure 2: Soil types and accompanying cover types

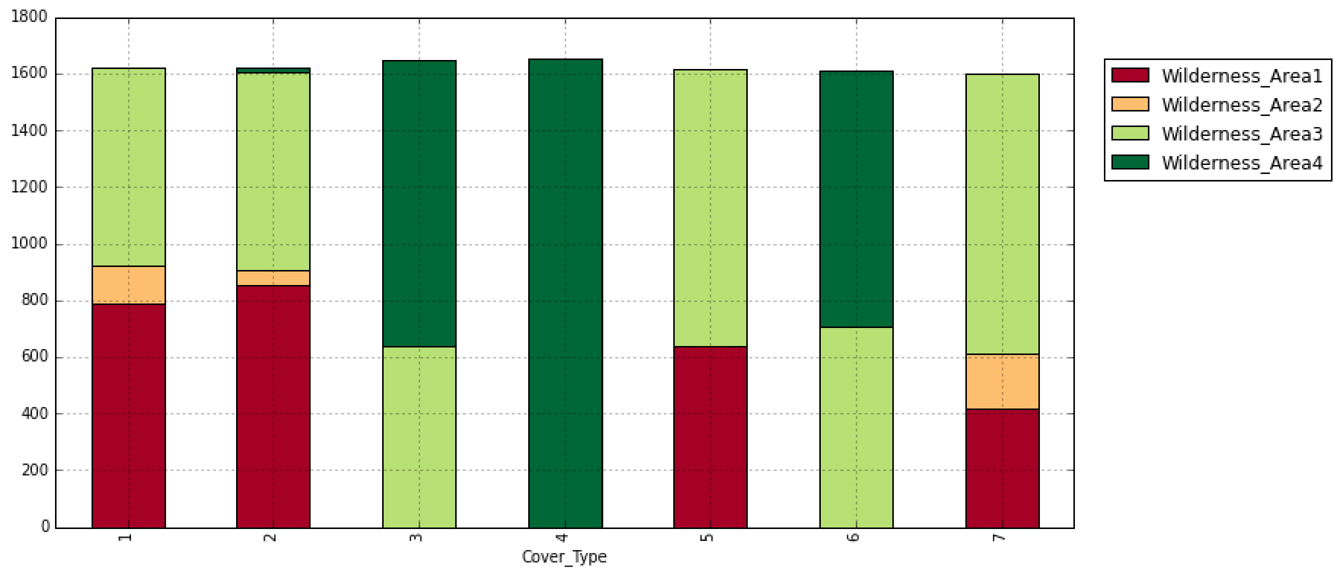

Figure 3: Wilderness types and accompanying cover types

5. CONCLUSION

6. ACKNOWLEDGMENTS