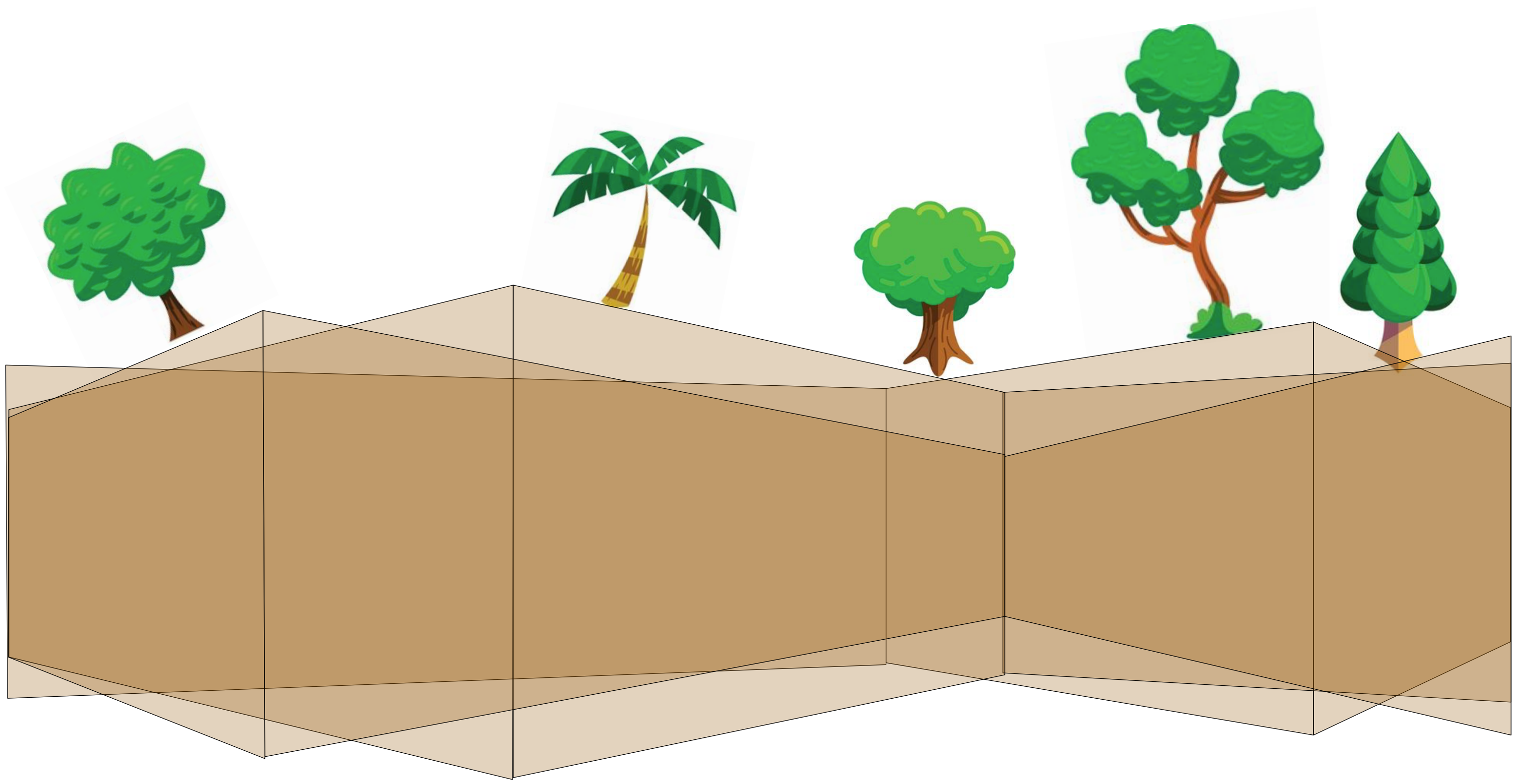
Forest Cover Type Prediction

David Albrecht



Problem Definition and Significance

The Roosevelt National Forest, located in north central Colorado, began in 1897 as part of the Medicine Bow Forest Reserve. Renamed in 1932 after President Theodore Roosevelt, its total area spans 813,799 acres[1] and is the subject of the following analysis. More specifically, the following pertains to a Kaggle competition[2] that uses the forest cover type dataset hosted by the UCI Machine Learning Repository[3]. The goal of the competition is to maximize classification accuracy of seven forest cover types given cartographic variables only.

Why should we care about this classification problem? Let's try to see the forest for the trees (pun very intended). Accurate forest cover identification is important because each type may have different requirements for growth, different values to people, and varying importance to local wildlife.[4] In addition, extending the use of cartographic variables to satellite imaging, accurate forest cover classification is crucial for conservation efforts as enormous amounts of forests can be kept track of in near real-time. For example, forest monitoring is used in the Amazon basin to quantify deforestation and forest biomass, which aid in identifying total forest carbon stocks.[5] It is our duty as citizens of this planet to lend a helping hand in the conservation of the environment, and it is our duty as data scientists to use technology to lend that helping hand!

Data and Description of Features

The study area from which the data was derived includes four wilderness areas in the Roosevelt National Forest where the areas represent forests with minimal human-caused disturbances. The data contains the following features:

---

[1] https://en.wikipedia.org/wiki/Roosevelt_National_Forest
[2] https://www.kaggle.com/c/forest-cover-type-prediction
[3] https://archive.ics.uci.edu/ml/datasets/Covertype
[4] http://youth.cnre.vt.edu
[5] http://globalforestatlas.yale.edu/amazon/conservation-initiatives/forest-inventory-and-monitoring

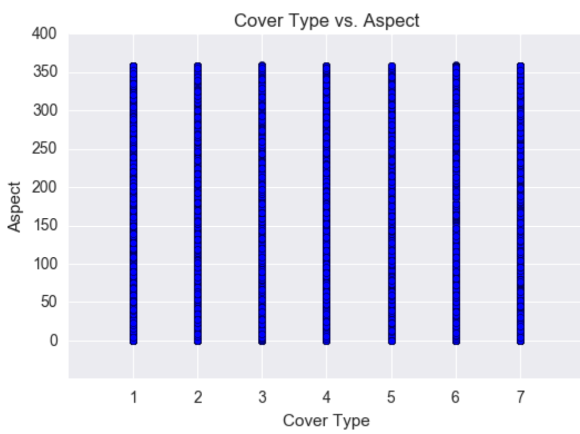
1. Instance identifier.
2. Elevation in meters.
3. Aspect in degrees azimuth.
4. Slope in degrees.
5. Horizontal distance to nearest surface water feature.
6. Vertical distance to nearest surface water feature.
7. Horizontal distance to nearest roadway.
8. Hillshade index at 9am, summer solstice.
9. Hillshade index at noon, summer solstice.
10. Hillshade index at 3pm, summer solstice.
11. Horizontal distance to nearest wildfire ignition points.
12. Wilderness area designation (binary):
    - Rawah Wilderness Area.
    - Neota Wilderness Area.
    - Comanche Peak Wilderness Area.
    - Cache la Poudre Wilderness Area.
13. Soil Type designation:
    - 40 binary features describing surrounding soil. The full list can be found in Appendix A: Soil Type Features.
14. Target Class: Forest Cover Type designation.

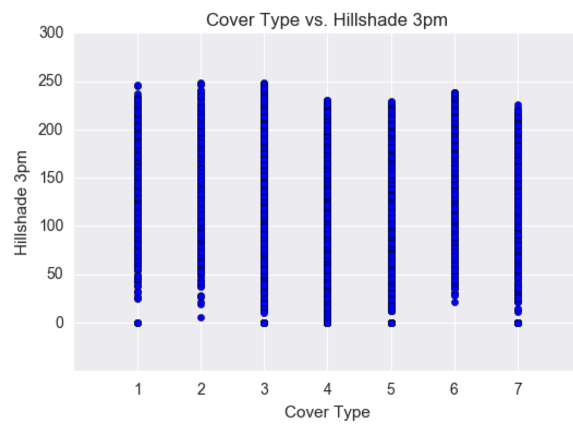
The training set contains 15,120 observations where each observation represents a 30m x 30m patch of forest. The forest cover type, as stored in the target class, was determined from US Forest Service (USFS) Region 2 Resource Information System data. We begin our forest classification task by performing exploratory data analysis on the training set.

Exploratory Data Analysis

After importing the training data and confirming that there are no missing values, we inspect each feature more closely to determine whether they all take values that we might expect given the descriptions above. Surely enough, all features take appropriate values, but not every feature looks to be helpful in being true predictors of forest cover type. Three features that reveal themselves are: 1) The instance identifier, 2) Soil Type #7: Gothic family, and 3) Soil Type #15: Unspecified in the USFS Soil and ELU Survey. The instance identifier is just that, and it is not a predictor of forest cover so it is dropped from the training data. Soil types #7 and #15 are empty features since these soils are not present for any forest cover type in the training data, and so they are dropped as well.

Next, we are interested in determining whether there are any clear outliers in the data by plotting boxplots of the first 11, non-binary features. Since all boxplots show there are no outliers, we turn to scatterplots to determine whether, for each forest cover type, different features take different values so as to distinguish between cover types. Inspecting whether any features help in distinguishing between cover types, we see that there are features that take similar ranges of values, and so are not good predictors. We see that 1) aspect in degrees azimuth and 2) hillshade index at 3pm, summer solstice features look to take the same values for each forest cover type:





We see that these two features will be distracting to a learning model because all cover types tend to have similar values, and so we drop these two features as well.

We have taken care of finding the features that may not be predictive, but what about redundancy in the data? With the number of features we have, it is plausible that we have too much noise in the data and, because of this, we use correlation to determine if any features are redundant or highly predictive of forest cover type. Using a cutoff value of 0.75 to reduce a correlation matrix between all features, we find that the Cache la Poudre Wilderness Area and elevation in meters features are correlated at approximately -0.784. Considering this is not very much higher than our cutoff of 0.75, we choose to keep these two features because there is not too much redundancy. Interestingly enough, per the UCI Machine Learning Repository, “Neota (area 2) probably has the highest mean elevational value of the 4 wilderness areas. Rawah (area 1) and Comanche Peak (area 3) would have a lower mean elevational value, while Cache la Poudre (area 4) would have the lowest mean elevational value." Given this information, it makes sense why we would see a high correlation between certain wilderness areas and elevation. In addition to finding redundancies in the data, correlation could also have helped us identify features that are strongly correlated with the forest cover type, but there were none in this case.

Finally, and very importantly, we inspect the target class to determine its distribution, and see that it shows a perfectly uniform distribution where each cover type is represented by 2,160 different instances. Clearly, since this data comes to us well cleaned, we do not have to worry about class imbalances. Now we can move on to building our model.

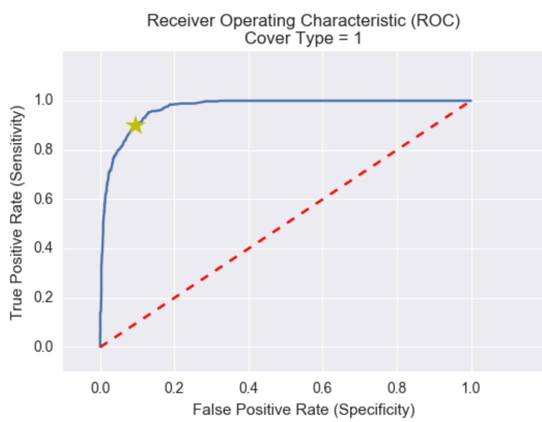
An Ensemble of Extra Trees

I chose to build seven different Extra Trees (or Extremely Randomized Trees) classifiers to create a single ensemble. The Extra Trees algorithm works by creating an ensemble of unpruned decision trees where randomized cut-points of randomly selected attributes form the entire structure. In its most extreme, the algorithm randomly picks a single attribute and cut-point at each node and therefore builds completely randomized trees that are independent of the target class. I chose this algorithm for all seven classifiers because it performed slightly better than the Random Forest algorithm on the public test set, which was one of the original baseline models. Specifically, Extra Trees introduces additional randomness over a Random Forest by randomizing tree splits when a Random Forest would seek to optimize those splits. This, in turn, leads to decreases in Extra Trees training time over Random Forests. Additionally, Extra Trees do not typically apply a bootstrap procedure to create training samples (as a Random Forest would) and instead uses the entire input training set to train all trees. As usual, there is no silver bullet and it is more or less happenstance that Extra Trees outperforms Random Forests in this case.[6]

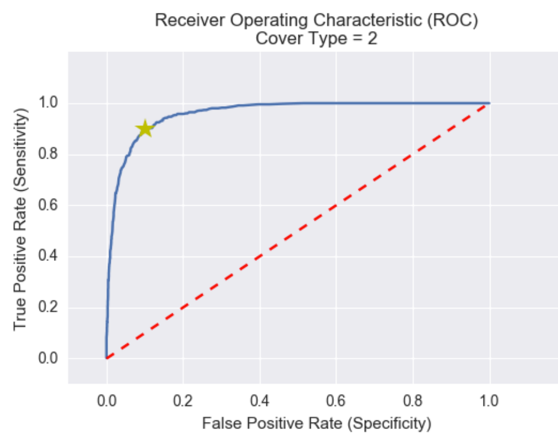
To build each classifier, I iteratively transformed the data from multi-class into binary. For example, the first classifier is trained on data where forest cover type 1 equals one and the other

[6] http://www.montefiore.ulg.ac.be/~ernst/uploads/news/id63/extremely-randomized-trees.pdf

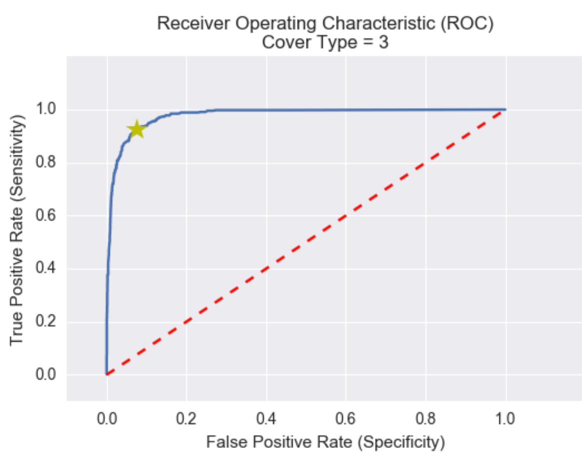
six cover types equal zero. This process is followed for each of the seven cover types to obtain seven different classifiers. Each classifier was trained using a grid search with 5-fold cross validation where I optimized for different numbers of trees using Receiver Operating Characteristic (ROC)/Area Under Curve (AUC). The number of trees in each final model varied between 180-280 where the search area contained a number of trees between 120-300 in increments of 10 (120, 130… 300). Each model was then used to return ROC, AUC, optimal ROC cutoff values, and predicted probabilities on the validation set. Below are the resulting ROC curves and related information for each cover type:



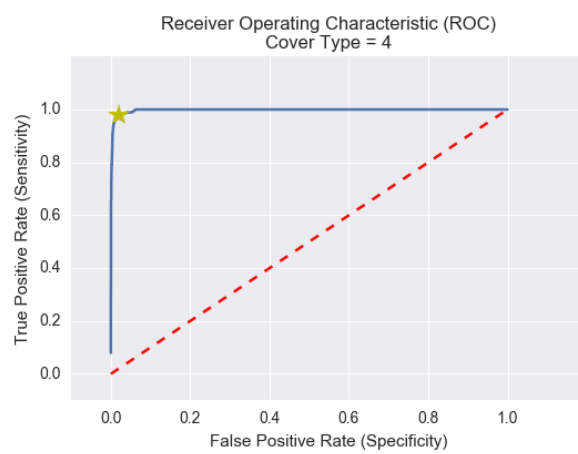

```
AUC: 0.9694
Optimal Threshold: 0.2542
```




```
AUC: 0.9603
Optimal Threshold: 0.2500
```

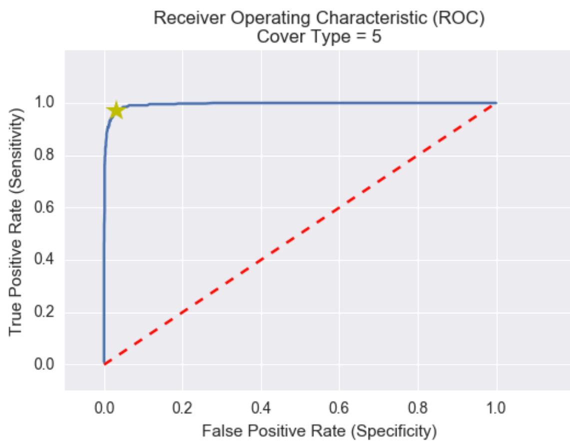



```
AUC: 0.9780
Optimal Threshold: 0.2679
```

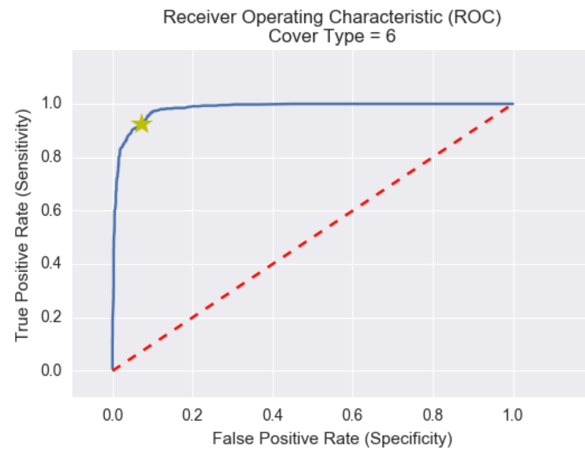



```
AUC: 0.9978
Optimal Threshold: 0.3412
```

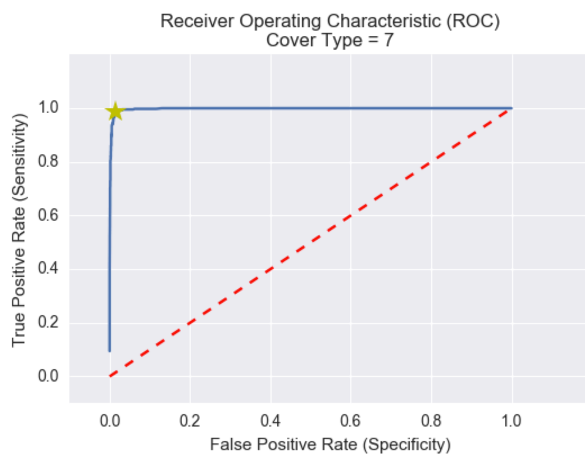
(Continued on the subsequent page)




```
AUC: 0.9953
Optimal Threshold: 0.3227
```




```
AUC: 0.9825
Optimal Threshold: 0.2778
```




```
AUC: 0.9980
Optimal Threshold: 0.3565
```

These ROC curves show us the tradeoff between the True Positive Rate (TPR) and False Positive Rate (FPR) over a wide range of cutoff values in class probabilities where the yellow stars indicate the optimal cutoff value. Judging by the above curves, we can see that the binary classifiers model the data very well in this one versus rest approach. Even though all AUC values are above 0.96, we see that the models have slightly more difficulty separating cover types 1 and 2 out from the other cover types (as exhibited by the lower AUC values) than they do with cover types 3-7. This could foreshadow a source of error between cover types 1 and 2 in the future.

A sample of the predicted, posterior probabilities on the validation set look as follows:

Posterior Probabilities by Forest Cover Type							
1	**2**	**3**	**4**	**5**	**6**	**7**	**True Label**
0.092308	0.000000	0.000000	0.000000	0.000000	0.000000	0.900000	7
0.000000	0.000000	0.338095	0.063158	0.000000	0.525000	0.000000	6
0.000000	0.000000	0.000000	0.000000	0.994444	0.000000	0.000000	5
0.392308	0.372222	0.000000	0.000000	0.116667	0.000000	0.016667	2

Before we decide on a solution to the ensemble, however, we have to account for the fact that the probabilities for each cover type were generated separately so there may not always be a clear maximum. We implement a softmax function to normalize across each instance, which, on the sample above, results in the following:

Softmax Posterior Probabilities by Forest Cover Type							
1	**2**	**3**	**4**	**5**	**6**	**7**	**True Label**
0.128175	0.116873	0.116873	0.116873	0.116873	0.116873	0.287461	7
0.122580	0.122580	0.171891	0.130572	0.122580	0.207217	0.122580	6
0.114900	0.114900	0.114900	0.114900	0.310600	0.114900	0.114900	5
0.183401	0.179754	0.123887	0.123887	0.139217	0.123887	0.125969	2

I chose to solve the ensemble using a comparison between the softmax probabilities and the optimal ROC threshold values for each classifier. Below is the pseudo-code for the decision function:

- For each row of posterior probabilities:
    - Store any forest cover type that has a probability greater than its optimal ROC threshold
        - If there is more than one forest cover type that exceeds its optimal ROC threshold, return the cover type with the greatest difference between probability and threshold
        - Else, if there are no forest cover types that exceed the optimal ROC threshold, return the cover type with the highest probability
        - Else, if there is only one forest cover type that exceeds its optimal ROC, return that cover type
    - **If the chosen forest cover type is 1 and the difference between cover types 1 and 3 probabilities is less than 0.03 then return cover type 3
    - **If the chosen forest cover type is 7 and the difference between cover types 7 and 1 probabilities is less than 0.05 then return cover type 1
    - **If the chosen forest cover type is 5 and the difference between cover types 5 and 2 probabilities is less than 0.03 then return cover type 2

** These points show exceptions in the ensemble solution process and act to help correct certain decisions. For example, when the determined forest cover type is 1, but its posterior probability does not exceed that of cover type 3 by more than 0.03, it has been determined that the forest cover type is more likely to be 3 and so the prediction is changed from cover type 1 to 3.

Following this approach, we achieve a validation set accuracy of ~86% and the following confusion matrix:

True Labels by Cover Type	Predictions by Cover Type: 1	2	3	4	5	6	7
1	355	77	1	0	7	0	13
2	83	330	11	0	19	9	1
3	0	6	365	25	3	55	0
4	0	0	7	442	0	5	0
5	1	23	3	0	423	4	0
6	2	4	50	15	3	380	0
7	31	0	0	0	0	0	423

We see that the result is very good where all forest cover types have a vast majority of correct classifications. It is also apparent that cover types 1 and 2 are the most difficult to distinguish between by far (highlighted in yellow). The UCI Machine Learning Repository shows a small bit of information that might help in further distinguishing between cover types: With regard to the original features, "As for primary major tree species in these areas, Neota would have spruce/fir (type 1), while Rawah and Comanche Peak would probably have lodgepole pine (type 2) as their primary species, followed by spruce/fir and aspen (type 5). Cache la Poudre would tend to have Ponderosa pine (type 3), Douglas-fir (type 6), and cottonwood/willow (type 4)."[7] In an attempt to take advantage of this knowledge, I implemented several tweaks in the above ensemble solution, but they were unsuccessful. For example, when taking into account the presence of the Neota area if the chosen cover type was 2, the model would correctly classify more cover type 1 while also

[7] https://archive.ics.uci.edu/ml/datasets/Covertype

incorrectly classifying more cover type 2. I chose to keep the final ensemble solution more balanced as opposed to favoring any single cover type, so these implementations were discarded.

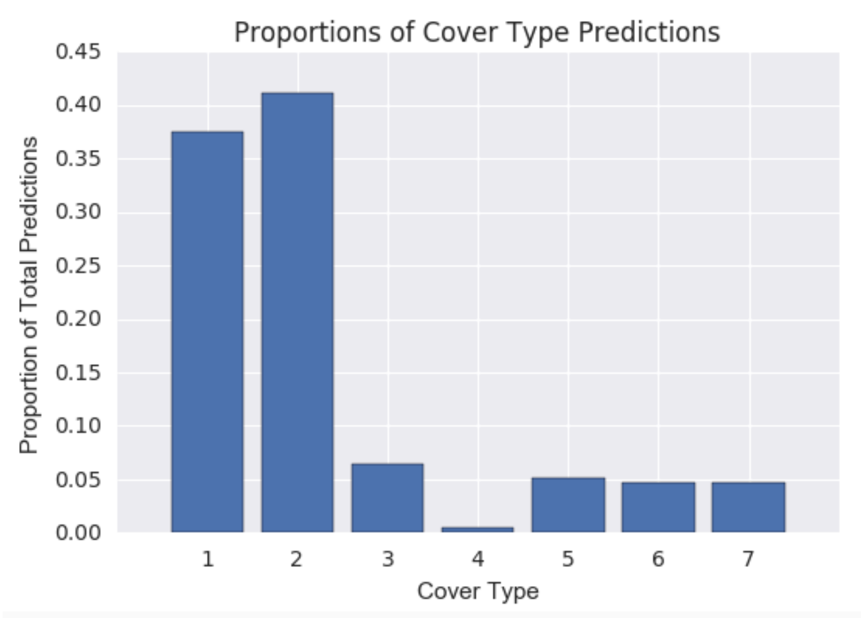
Now, since we tuned hyper parameters on the validation set, it is important to test the final model on an independent test set. Having set aside a labeled test set before training the ensemble on the training set and tuning on the validation set, we can estimate expected results when the model is introduced to new data. Taking the test set, we predict using our ensemble, use the softmax function to normalize the returned probabilities, and use the decision function to determine our predictions. We are able to achieve ~85% accuracy here, which shows us that our model holds. We have the following confusion matrix:

True Labels by Cover Type	Predictions by Cover Type: 1	2	3	4	5	6	7
1	492	118	1	0	12	1	24
2	115	449	18	0	37	24	5
3	0	4	510	42	5	87	0
4	0	0	10	630	0	8	0
5	2	29	18	0	591	8	0
6	0	2	64	12	2	568	0
7	32	1	0	0	0	0	615

We see here again that the result is very good where all forest cover types have a vast majority of correct classifications, but cover types 1 and 2 are still the most difficult to distinguish between by far (highlighted in yellow). It is important to note and remember here that the training data was originally uniformly distributed, and stratified splits to the data created equally uniform training, validation, and testing sets. This fact, coupled with the model's difficulty distinguishing between cover types 1 and 2, is a main reason why we might expect to see different results depending on the class distribution in the testing set. For example, a testing set weighted more heavily toward cover types 3-7 is expected to classify more easily (> 85% accuracy) than a testing set that is weighted more heavily toward cover types 1 and 2 (< 85% accuracy). We will see this phenomenon in action when we predict on Kaggle's public test set.

Final Testing and Conclusions

Once again, this time with Kaggle's set aside test set, we use our trained ensemble to predict on the unlabeled test set, use the softmax function to normalize the returned probabilities, and use the decision function to determine our predictions. We are able to achieve ~74.5% accuracy on the public leader board, which is ~10% lower than our original estimate on our held out test set. At this point we might think there could have been signal leakage between training, validation, or test sets, but there is something a bit different going on here as touched upon previously. Inspecting the proportions of submitted predictions for each of the seven cover types, we see:



It is clear, and we can be confident given our accuracy, that cover types 1 and 2 are far more heavily represented in the public testing set than in the training set where we saw a perfectly uniform distribution. Since our model had the most trouble distinguishing between these two cover types that are now the vast majority of the test set, it is understandable that we underperformed. Instead, had the public test set been weighted toward cover types 3-7, we may have outperformed our accuracy estimate.

This achieved accuracy on the public test set puts us at about rank 900 out of 1,700 where a good number of submissions achieve very high accuracies and even a single 100% accuracy. Digging deeper into how this is possible and after multiple model variations, it became apparent that, since the UCI Machine Learning Repository hosts the entire dataset, a model could be trained and tuned on the entire data set so as to over-fit the model and then predict on the redundant public test set. Given this information and the robustness of our models even in the face of imbalanced data, we can be confident that the generated models could be helpful in a real world scenario. We were able to achieve an accuracy that is much greater than simply guessing any of the seven forest cover types. You're welcome, Mother Nature!

Appendix A: Soil Type Features

1. Cathedral family - Rock outcrop complex, extremely stony.
2. Vanet - Ratake families complex, very stony.
3. Haploborolis - Rock outcrop complex, rubbly.
4. Ratake family - Rock outcrop complex, rubbly.
5. Vanet family - Rock outcrop complex complex, rubbly.
6. Vanet - Wetmore families - Rock outcrop complex, stony.
7. Gothic family.
8. Supervisor - Limber families complex.
9. Troutville family, very stony.
10. Bullwark - Catamount families - Rock outcrop complex, rubbly.
11. Bullwark - Catamount families - Rock land complex, rubbly.
12. Legault family - Rock land complex, stony.
13. Catamount family - Rock land - Bullwark family complex, rubbly.
14. Pachic Argiborolis - Aquolis complex.
15. Unspecified in the USFS Soil and ELU Survey.
16. Cryaquolis - Cryoborolis complex.
17. Gateview family - Cryaquolis complex.
18. Rogert family, very stony.
19. Typic Cryaquolis - Borohemists complex.
20. Typic Cryaquepts - Typic Cryaquolls complex.
21. Typic Cryaquolls - Leighcan family, till substratum complex.
22. Leighcan family, till substratum, extremely bouldery.
23. Leighcan family, till substratum - Typic Cryaquolls complex.
24. Leighcan family, extremely stony.
25. Leighcan family, warm, extremely stony.
26. Granile - Catamount families complex, very stony.
27. Leighcan family, warm - Rock outcrop complex, extremely stony.
28. Leighcan family - Rock outcrop complex, extremely stony.
29. Como - Legault families complex, extremely stony.
30. Como family - Rock land - Legault family complex, extremely stony.
31. Leighcan - Catamount families complex, extremely stony.
32. Catamount family - Rock outcrop - Leighcan family complex, extremely stony.
33. Leighcan - Catamount families - Rock outcrop complex, extremely stony.
34. Cryorthents - Rock land complex, extremely stony.
35. Cryumbrepts - Rock outcrop - Cryaquepts complex.
36. Bross family - Rock land - Cryumbrepts complex, extremely stony.
37. Rock outcrop - Cryumbrepts - Cryorthents complex, extremely stony.
38. Leighcan - Moran families - Cryaquolls complex, extremely stony.
39. Moran family - Cryorthents - Leighcan family complex, extremely stony.
40. Moran family - Cryorthents - Rock land complex, extremely stony.