Forest Cover Type Prediction

a Kaggle Competition

Karl Glaub

Competition brief

In this competition you are asked to predict the forest cover type from cartographic variables

Data comes from the US Forest Service and US Geological Survey

The data was 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

The Data

The study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado. Each observation is a 30m x 30m patch. You are asked to predict an integer classification for the forest cover type. The seven types are:

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

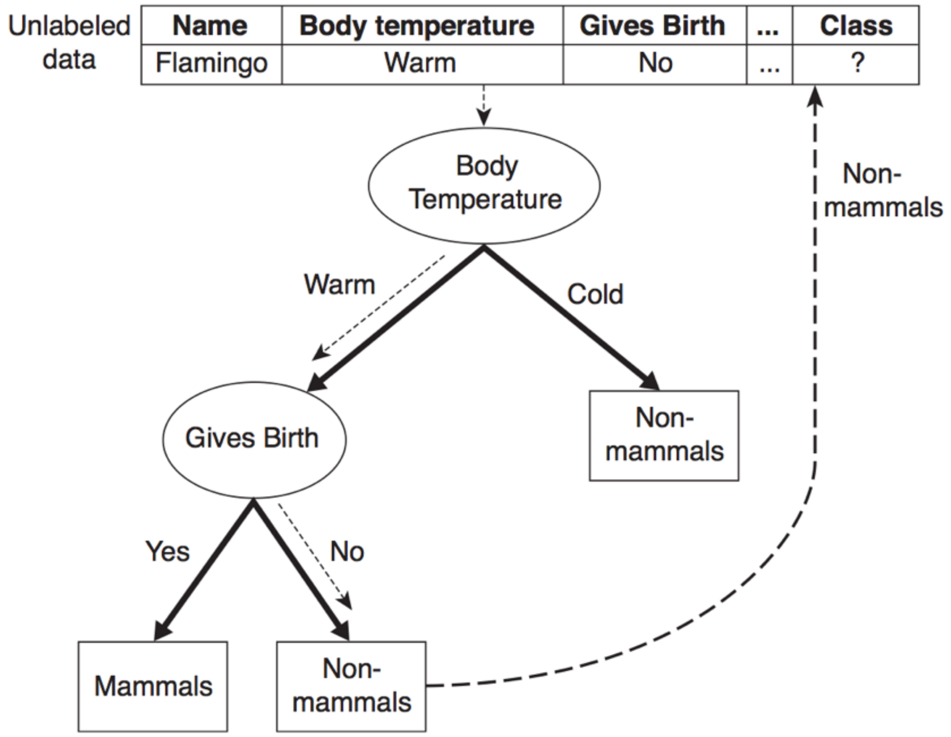
The training set (15120 observations) contains both features and the Cover_Type. The test set contains only the features. You must predict the Cover_Type for every row in the test set (565892 observations)

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

Approach

I used random forests as that was being used by others on Kaggle and I forked their code. Supervised classification also makes a lot of sense considering the task at hand, which is similar to our vertebrate example where we used decision trees:



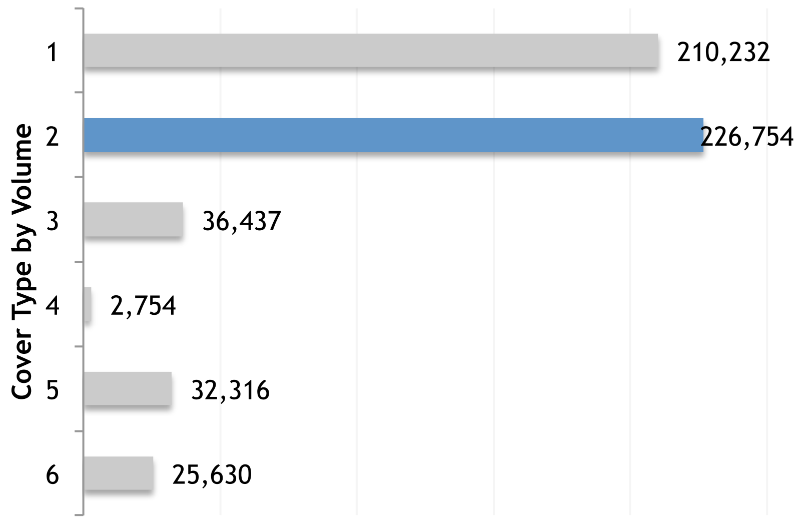
Unlabeled data	Name	Body temperature	Gives Birth	...	Class
	Flamingo	Warm	No	...	?

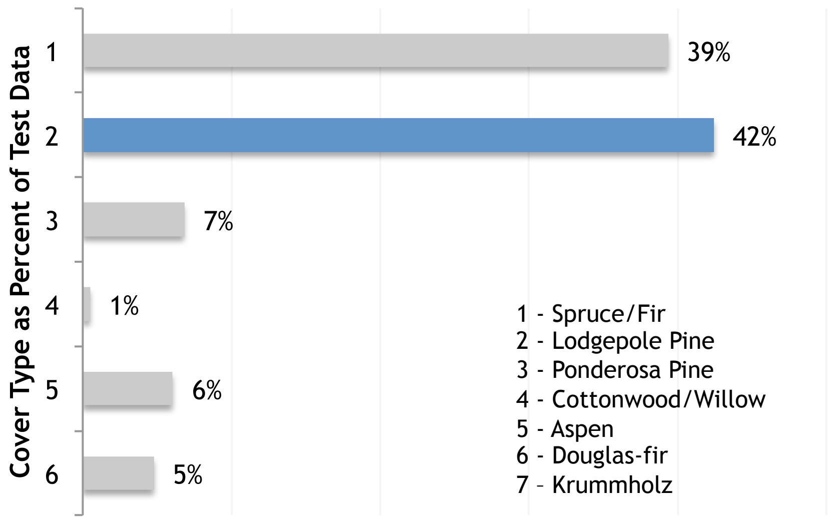
Approach

The sklearn ensemble random forests classifier is supposed to enhance the model by using a best split between a random subset of features, and then averaging it. This increases bias initially, but reduces variance and delivers a better result. Best of all, it allows people with minimal coding skills to use decision trees with more confidence.

I added in matplotlib to graph the results in a histogram.

Results





Lodgepole pine is the most common forest cover in this region

Going forward

I plan to take coding classes and become more familiar with python at a slower pace.

```
import pandas as pd

from sklearn import ensemble


if __name__ == "__main__":

  loc_train = "../input/train.csv"

  loc_test = "../input/test.csv"

  loc_submission = "kaggle.RF500.submission.csv"


  df_train = pd.read_csv(loc_train)

  df_test = pd.read_csv(loc_test)


  feature_cols = [col for col in df_train.columns if col not in ['Cover_Type','Id']]


  X_train = df_train[feature_cols]

  X_test = df_test[feature_cols]

  y = df_train['Cover_Type']

  test_ids = df_test['Id']


  clf = ensemble.RandomForestClassifier(n_estimators = 500, n_jobs = -1)


  clf.fit(X_train, y)


  with open(loc_submission, "w") as outfile:

    outfile.write("Id,Cover_Type\n")

    for e, val in enumerate(list(clf.predict(X_test))):

      outfile.write("%s,%s\n"%(test_ids[e],val))
```

```
import seaborn as sns

import matplotlib.pyplot as plt

from pylab import savefig

#Histogram
#create a new figure 10 X 10 inches

fig = plt.figure(figsize=(10,6))

fig.suptitle('Forest Test Data Histogram', size=24)


#subplots - (column, row, plot index)

ax = fig.add_subplot(2,2,1)


#Nan values don't play nice with histogram function

ax.hist(y)

#Can specify a range of bins

ax.set_title('Histogram', size=18)

savefig("my_plots.png”)
```