FOREST COVER TYPE PREDICTION

W207 (MIDS) Section 2

Submitted by:

- **Eric Le**
- **Jherson Fuentes**
- **Jash Sompalli**
- **Aaron Tang**

Github: https://github.com/aatang16/W207-project---Machine-Learning-

Kaggle Dataset: https://www.kaggle.com/c/forest-cover-type-prediction

---

Problem Motivation: Besides global warming, deforestation is an increasing concern for many. In fact the world has lost one-third of its forest, an area twice the size of the United States. By identifying forest covers we would be able to protect endangered forests as well as better monitor our forests.

Objective: In this competition we 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 4 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.
Information about the **7 possible types** of forest cover (coded as integers from 1 to 7), as well as the 54 features, can be found at https://www.kaggle.com/c/forest-cover-type-prediction/data.

As mentioned there, the training set contains 15,120 observations, whereas the test set contains 565,892 observations (about 37 times larger than the training set). Both sets contain information about 10 continuous features (listed below), and 2 categorical features (wilderness area and soil type), with 4 and 40 categories each (coded as 4 and 40 binary features, respectively, where 1 indicates presence and 0 indicates absence).

- Elevation (in meters)
- Aspect (in degrees azimuth)
- Slope (in degrees)
- Horizontal_Distance_To_Hydrology (i.e., to nearest surface water features)

- Vertical_Distance_To_Hydrology (ditto)
- Horizontal_Distance_To_Roadways (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 (i.e., to nearest wildfire ignition points)

Experiments: We used the most typical classifiers that we covered in the course:

- **K Nearest Neighbors**
- **XGBoost**
- **Light Gradient Boosting Machine**
- **Random Forests**
- **AdaBoost**
- **Extra Trees Classifier**

Results: As we'll see later, **Light Gradient Boosting Machine** yielded the best results, followed by **Random Forests**, **XGBoost, Extra Trees, K Nearest Neighbors**, and **AdaBoost**.

We assembled the results of all models, weighting them by their accuracy.

Conclusion:

From our experiment, it seems that the Light Gradient Boosting Machine provided the best results in terms of testing accuracy at 90%.

The other two algorithms Random Forest Classifier & XGBoost seem to provide similar results and if accounting for variations in testing data sets, the difference from naked eye numbers seems rather negligible. ETC does slightly worse, and drops off significantly, with inferior results from KNN, Neural Networks and ADA. It seems that Decision Tree class algorithms provide the highest degree of accuracy in this experiment.

The KNN will require us to run through all the data when provided with a new datapoint to classify, this will be very costly and may degrade the quality of the algorithm. There are hundreds of thousands of data points with 100+ features, so it may be inappropriate to use KNN with the amount of computational power required.

Additionally, we should consider the pros and cons of each from Gradient Boosting & Random Forest. Random Forest is generally easier to tune than GBM and is more difficult to overfit - however, a large number of trees as in our case may make the algorithm slow for real-time prediction. However, real time prediction may not be necessary in our case, if we are simply just classifying forest type according to the parameters given.

The results above show that the decision tree is the best choice for most accurate classification. In the future, we will try to include more parameters to test - e.g. adjust learning rates, number of k's in KNN to see if we can find an algorithm with higher accuracy - additionally, we should consider neural networks with different types of parameters to find the most optimized accuracy e.g. different learning rates, number of epochs etc.