Leaf Classification Methods Using the Scikit-Learn Library

CMPT 404

Dr. Pablo Rivas

John Randis

12 December 2016

Abstract

This paper details an application project intended to classify different species of leaves using different classification methods. This is part of a Kaggle data science competition. The competition measures results using the logarithmic loss algoritm. The two classification methods used in this project are part the Scikit-learn library for python. They are the kneighbors classifier and the logistic regression classifier. These classification methods will be compared to see what works the best on this data set. They will be compared using the logarithmic loss metric, because that is what Kaggle will use to judge.

Introduction

The implications and benefits of automatic plant classification are wide and far reaching. Human error often leads to misclassified or duplicate classifications. Automated classification can help discoveries in biology and medicine. All data collected and code produced will be submitted to Kaggle for judging in their leaf classification competition. This paper will include the background and related works, the methodology of the approach taken for this project, and a careful analysis of results.

Background

In very recent history, biological information has never been more readily available. Databases exists across the web full of biological information on plant life. Online biodiversity databases such as http://www.gbif.org/ and http://www.pfaf.org/ have made this information more accessible than ever. Biologists have been classifying plant and animal species for years. This is why introducing machine learning to this problem could be so revolutionary. The problem of plant classification has not been tackled much in the data science community. The Kaggle competition has been ongoing since August 30th of this year. There have been over 900 submissions made with a few claiming to have gotten perfect

results. I am eager and excited to be a part of this project and hopefully can contribute some meaningful data to Kaggle.

Methodology

There will be two approaches taken for classification in this project. The first is logistic regression. Logistic regression is a simple approach to this problem and is less prone to overfitting. It assigns probability values corresponding to each leaf and how likely it is to be each species. The second approach used will be k nearest neighbors. This is also an easy to implement method using the sklearn kneighbors classifier. The python library sklearn is being implemented for these tasks. Fortunately Kaggle provides a csv of pre-extracted features from the leaf images. The extracted features contain 64 attribute vectors for margin, shape, and texture. The classifier stores all these features in a pandas data frame, and builds a logistic regression model based off of the training data. Then, the model is applied to the test data. Each leaf is given a probability 0-1 for belonging to a specific species. It is outputted to the .csv submission file. For judging, Kaggle is using logarithmic loss, and that is the metric I will be using to compare the classifier. The function is defined as:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

Logarithmic loss is the logarithm of the likelihood function for a Bernoulli Random Distribution.

Experiments

1584 pieces of data were used in this experiment. They are divided up into 991 data points used for training and 593 used for testing. They all corresponded to a different image of a leaf. The classifier checks for features like margin, shape, and texture. The first script

that I ran, I built with the intent to use the KNeighbors classifier. The KNeighbors classifier works by importing the sklearn library, giving it a number of neighbors, and then fitting it for the training values. The first time I ran the classifier, the logarithmic loss ended up being 1.37956. It got past the uniform probability benchmark, and I was happy with the score for a first result.

Before moving on to another classifier, I did not want to give up on KNeighbors. I thought it would be good to try to massage the data a bit and run it through the classifier again. I used the sklearn standard scaler preprocessor to standardize the features of the leaves by removing the mean and scaling to unit variance. This gave me a result of .10093, which was a significant improvement.

I then tried another approach: to use logistic regression to classify the data set. There exists open source code for this on Kaggle, under the Apache 2.0 license. The data is put into a pandas data frame, is scaled, regularized, and then the sklearn logistic regression classifier is run. The support vector machine is given the species index (x_train) and its attributes (y_train). They are given to the classifier to accurately create a model without overfitting. We call the predict_probability method on the classifier now to run the test data and get percentage values for each of the plant ids and their possible species. The code compiled in under 10 seconds. The attached csv submission file includes all of the results.

Analysis

Looking at the results of the experiment, it seems as though logistic regression is the more reliable method for classification of this kind. However, there are a few caveats. The author of the open source code that wrote the logistic regression is a more experienced data scientist. The hyperparameters of the logistic regression classifier were fine tuned for this specific data set. On the other hand, I used most the default values for the hyperparameters

for the kneighbors classifier. This may skew the results a bit in favor of logistic regression. An advantage of the kneighbors classifier though is that logarithmic loss favors values that read absolutely true (1) or absolutely false (0). Since I used 3 neighbors, the values can either be 1, .67, .33, or 0. If there is a misclassified true or false (1 or 0), the logarithmic loss value increases exponentially resulting in severe penalties. If I had more time to do the project, I would have liked to use neural networks using Keras. The metric used to compare the two classifiers, logarithmic loss, gave the kneighbors classifier a .10093 and the logistic regression classifier a .03806.

Conclusion

Using logistic regression and k nearest neighbors, the classifiers both provided an accurate way of sorting the plant species. I think that logistic regression beat out k nearest neighbors for two main reasons. One, the hyperparameters for logistic regression were better tuned for this specific data set. With kneighbors, I picked 3 neighbors and left the rest of the parameters as default. In addition, kneighbors has absolute true and false (1 and 0) values, in which misclassified points incur a more strict penalty. If I had more time to do this project, I would have also liked to compare the two classifiers using other metrics. Hopefully the data that I contributed to Kaggle will be useful. Online databases today make accessing biological data easier than ever. Automating the organization of this data is an important step we can take to improve the free flow of information. I am glad to have been a part of this process and contributed to the Kaggle competition and hopefully something useful will result of it.

References

benjo. Sept. 2016. *Logistic Regression*. Python 3.4. Apahace 2.0 Open Source License. Source Code. *Kaggle*.

"Global Biodiversity Information Facility." *Free and Open Access to Biodiversity Data*. GBIF, n.d. Web. 07 Nov. 2016.

"PFAF." *PFAF*. Plants For A Future, n.d. Web. 07 Nov. 2016.