LIME in Spark

Tin Luu

The mission of LIME is to explain what machine learning classifiers are doing.

LIME stands for Locally Interpretable Model-Agnostic Explanation and, according to the paper on it, LIME "is a novel explanation technique that explains the predictions of any classifier in an interpretable and faithful manner, by learning an interpretable model locally around the prediction"

For detailed information on what the referred "explanation" means, please take a look at https://github.com/marcotcr/lime as it is very well defined there.

Here are a few points to note about LIME. First, it is built on scikit-learn and works most organically with it. Second, it can only generate individual explanations, thus its explainer can only handle one data point at a time. Third, LIME generates explanation both textually and visually (see library Github). Lastly, while it is well-integrated into platforms such as Anaconda's Jupyter Notebook, LIME is not well-integrated into big-data systems as such Spark. The consequence of the first two points is that ML pipelining, analyzing data points and generating explanation are slow given a large dataset, and this is ultimately inefficient.

https://github.com/marcotcr/lime/blob/master/CONTRIBUTING.md
The creators of LIME have suggested developing various options to improve LIME usability (link above). From my perspective, I think it is important for LIME to be well integrated into big-data systems. In particular, I will use the link below as a benchmark and 1) integrate LIME to be usable and parallelizable in Spark, 2) to expand on the example and improve its classifier accuracy.

There is a great deal of similarity between the functions of scikit-learn and Spark that allows building LIME in Spark to make sense. Notably, just as one builds ML pipelines with the various data transformers and classifier in scikit-learn, we can do similarly so with Spark's MLlib and ML libraries. One advantage of using Spark is the parallelization of the data and pipeline that ultimately speeds up the processing. As we choose speed and efficiency, we would be sacrificing the visualization of explanations that can be displayed with IPython in Jupyter Notebook. Nevertheless, what is the most important still remains, that is the generated textual explanations.

https://marcotcr.github.io/lime/tutorials/Lime%20-%20basic%20usage%2C%20two%20class%20case.html

<u>Requirements:</u>
Please use the attached .py file.
We will be using NYU Dumbo HPC and PySpark.
    Python 3.6.5, Spark 2.4.0, Numpy (versions available on Dumbo)
    Lime, Numpy (versions available on Dumbo)