Exercise #10 - Introduction to Machine Learning

Computational Physics Lab

Prof. Sean Dobbs & Daniel Lersch

April 17, 2020

1 Instructions

Goal of this assignment is to make you familiar with machine learning techniques in (physics) data analysis. You will have one weeks to solve this homework (starting from this Friday: 04/17/2020). It is highly encouraged to work in groups, but the answers to this homework have to be submitted individually. Please do not hesitate to contact Prof. Dobbs (`sdobbs@fsu.edu`) or Daniel (`dlersch@jlab.org`) if you have any questions or need further assistance.

1.1 What is provided

You will be given a .csv file, accessible at: `http://hadron.physics.fsu.edu/~dlersch/ml_homework/`, containing a data set similar to the one discussed in the lecture. Furthermore, you will be given an analysis-skeleton which shell help you to solve this assignment.

1.2 What you need to provide

A document (latex, word, excel, whatever you prefer ..) summarizing your answers and conclusions. You should provide all calculations you make (if any!) in detail and describe the plots you produce.

1.3 Submitting your Homework

You will be evaluated based on the files contained in your remote GitHub repository at the due date. You should and commit all the files you created to your local repository on a regular basis. You should do this by adding any new or modified files using "git add", and then finalizing changes using "git commit". Remember that "git status" will give you information on which files in your repository have been changed. Remember to add short, but useful comments when performing a commit. When you are finished with the exercise, push the current status of your local repository to the remote repository on GitHub using the command "git push -u". You are encouraged to push your local files to the remote repository periodically before you are done, and certainly well before the deadline, if possible.

2 The Data Set

The .csv file is called: **fsu_ml_hwk_data.csv** and contains $\sim 225\,\mathrm{k}$ events with three particles species, each defined by three features and labeled: 0 (for species 1), 1 (for species 2) and 2 (for species 3).

This data is imbalanced, i.e. #Events with species 1 > #Events with species 3 > #Events with species 2. Or in other words: species 1 is the majority and species 2 the minority.

3 Analysis

The analysis-skeleton is called: **run_rf_classifier.py** and performs, when executed, the following taks:

1.) Read-in the given .csv file and create a DataFrame

2.) Create training data for classification

3.) Setup and train a random forest classifier

4.) Generate a handful of monitoring plots (e.g. classifier output, ROC-curve,..) which will be automatically saved with the prefix RF_ as .png file

It is up to you whether to use this code or not. It shall simply provide a guideline, but no one prevents you to write your own script from scratch.

It might become handy to save your machine learning model, once it has been trained and you decide to use it afterwards. You can do this by either using pythons pickle or joblib. A brief description on this can be found here `https://scikit-learn.org/stable/modules/model_persistence.html`.

4 Inspecting the Data

One of the first things you should always do, is having a close look at your data. NEVER apply any machine learning algorithm on an unknown data set.

4.1 Getting Started (1 pt)

Print out the first 20 lines of the DataFrame and check whether it has the right format (i.e. three feature columns and one label column). This is a quick way to ensure that you either look at the right DataFrame, or that you setup your own DataFrame properly.

4.2 Correlation Plots (1 pt)

These are the most helpful tools in any analysis, as they contain useful information and help you to better understand your data. Bearing this in mind, you should generate for each particle species the following correlation plots: Variable 2 vs. Variable 1, Variable 3 vs. Variable 1 and Variable 3 vs. Variable 2. Ideally, you should end up with nine plots in total. What do these plots tell you?

5 Classification via a Multi-Layer Perceptron (MLP)

A brief description of the scikit MLP classifier can be found here: `https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html`. This link might help you to understand which parameters can be tuned and how.

5.1 Training the MLP (3 pts)

Setup the MLP, like demonstrated in the lecture. You might choose the architecture (i.e. the number of neurons, layers), or any other parameter, freely. Afterwards, train the multilayer perceptron on the given data set. Hint: you might use the example-script and modify the classifier specific parts. Plot the training- and validation- (you decide whether to use a validation set or not) curves. Feel free to try different mlp settings and get a feeling on how the algorithm "behaves" with respect to the given data.

5.2 MLP Responses (3 pts)

Generate again the feature correlation plots for each species after classification. Do you see any weird fragments / structures? Plot all three mlp outputs for each species (i.e. output 1 for species 1,2,3, output 2 for species 1,2,3, etc.). What do you observe?

5.3 MLP ROC-Curve (3 pts)

Generate the ROC-curve for each species. What do these plots tell you?

5.4 MLP Confusion Matrix (3 pts)

Generate the confusion matrix plot for your mlp. How would you judge your classifiers performance? Compare the information gained from: the confusion matrix, ROC-curve and classifier outputs. Do all plots provide the same information? Are your observations consistent?

6 Classification via a Random Forest Classifier

When using machine learning in your analysis, it is sometimes helpful to try different algorithms and compare them. You might encounter, that the algorithm you initially chose is outperformed by an other algorithm you tested. There are also cases where the data needs to be analyzed by an ensemble of different algorithms, because each of them shows a high performance only on a portion of the data set.

This link: `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestC` `html` summarizes the important parameters for the random forest classifier and provides a few examples.

6.1 Know your classifier (3 pts)

Get familiar with the basic working principles of a decision tree and a random forest classifier. Train the random forest classifier on the given data and vary its settings (i.e. the number of trees, the depth of each tree,..). Again, you might do this several times, in order to better understand the random forest classifier. Of all settings you try choose three, based on your own judgement. Plot for each of the three settings the ROC-curves and compare them. What do you conclude?

6.2 Application in Data Analysis (3 pts)

Pick the random forest classifier that shows the best performance (judged by the ROC-curve) and: generated the feature-correlations plots after classification, plot all three classifier outputs for each species, generate the confusion matrix. What do you conlcude from all these plots? How does the random forest classifier compare to the mlp?