# Laboratory 4 - Machine Learning with Spark MLlib

In this lab, we will learn to use different aspects of Machine Learning applied to a Big Data framework. We will apply classification and clustering techniques to measurements of the Internet traffic. We provide a network log trace file summarizing the traffic generated by thousands of users while browsing the web. In this laboratory, you need to perform a complete pipeline for applying machine learning. In the first part, for each network TCP connection, you have to identify and classify the service (e.g., Google, Amazon) that generated the traffic. In the second part, you will cluster the users according to their behaviour on the internet.

For this lab, you are required to **use the MLlib of Pyspark with the API for DataFrames (in pyspark.ml)**.

## 1. Input Data

In this lab you will use a Tstat log file. It reports per-TCP connection statistics. In other words, each line represents a TCP connection. Beside to the connection identifiers (client and server IP addresses and ports), Tstat reports dozens of features, such as number of packets, bytes uploaded and download, etc.

The file is stored at the path `/data/students/bigdata_internet/lab4/log_tcp_complete_classes.txt`. Note that you have to read it as a DataFrame. In this file, there is the header and the separator used to define the fields is the space `' '`.

Take a look at http://tstat.tlc.polito.it/measure.shtml#log_tcp_complete (http://tstat.tlc.polito.it/measure.shtml#log_tcp_complete) for details about the content of the columns. Then take confidence with the columns of the dataframe and analyze their content and data type. **Hint**: you can use the attributes of DataFrame `columns` and `describe` or the method `printSchema()` to analyze the name and data type of columns.

**Question:** How many columns/features has the file? How many TCP connections?

Notice that some variables are categorical. In particular, these columns are categorical:

- '#31#c_ip:1'
- 'c_port:2'
- 's_ip:15'
- 's_port:16'
- 'con_t:42'
- 'p2p_t:43'
- 'http_t:44'
- 'p2p_st:59'
- 'c_tls_SNI:116'
- 's_tls_SCN:117'
- 'c_npnalpn:118'
- 's_npnalpn:119'
- 'fqdn:127'
- 'dns_rslv:128'
- 'class:207'
- 'fqdn:127'
- 'dns_rslv:128'
- 'class:207'

For the possible values, we refer to Tstat manual (see, for example, the possible values for the protocol used in column `con_t:42`). Take care that other columns are binary (only 0/1 possible values), like for example:

- 'c_isint:38'
- 's_isint:39'
- 'c_iscrypto:40'
- 's_iscrypto:41'

## 2. Classify TCP connections

In this lab you have to classifiy each TCP connection, i.e., each line of the log, to the corresponding service that generated the line.

The class label is provided in the `class:207` column. There are different possible classes, corresponding to popular web services, such as Amazon, Google, YouTube, etc.

**Question:** How many classes are there in the file? Can you list all of them? How many connections per web service are present in the DataFrame?

You should follow the next steps, creating a `Pipeline` of transformers and training them. Motivate each choice that you take in each step of the pipeline.

### 2.1 Read and split the data

Read the data and split the dataset into a train and a test set. The test set will be used to asses the final performance that your final chosen classifier can reach. Take care that the connections are taken in random order.

### 2.2 Pre-process the dataset

You are free to select the features that you consider more useful for this task among all the available features.

You cannot use IP addresses as features as they are categorical attributes with very high cardinality, leading to overfitted results. Moreover, you cannot use information about the server providing the service that would make the problem trivial, e.g., its domain name. In details you **CANNOT** use the following features: `#31#c_ip:1`, `c_port:2`, `s_ip:15`, `s_port:16`, `c_tls_SNI:116`, `s_tls_SCN:117`, `fqdn:127`.

**Hint**: Start with a limited subset of features to perform the pipeline, and then add other features later.

Take care of preprocessing each feature. You can use for example `VectorAssembler`, `StandardScaler`, `StringIndexer`, `OneHotEncoderEstimator`.

### 2.3 Train at least two different models

Try with the training data at least two different classifier models from `pyspark.ml.classification`. One of the models should be the `RandomForestClassifier`. Try also different parameters (e.g., max tree depth).

### 2.4 Evaluate the performance of the models

Create one or more `evaluator` to compute performance evaluation metrics. Compute at least the global accuracy and the f-measure for all the classes in the dataset.

## 2.5 Tune the parameters of the models

Now that you have a model and a way to evaluate the performance, do a proper parameter tuning. Tune both the classifiers used at 2.3. In order to avoid overfitting, validate your results on a validation set by using one of the method of `pyspark.ml.tuning`. Build a `ParamGridBuilder` and evaluate a grid of parameters for all the transformers in the pipeline. Choose the best model according to global accuracy. Report the accuracy results for all the parameters you tried.

## 2.5 Return the best possible model and estimate its performance on new data

Choose the best possible model with the best configuration of parameters and report it (model type and parameters), with its performance evaluated on the training and validation sets. Then, at the end, you should assess the performance also using the test set. Report the expected results performance and comment on the results obtained.

# 3. Cluster users (BONUS TASK)

In this exercise you will cluster clients (i.e., users), according to their behaviour on the web. In particular we want to group together users for which we observe similar internet traffic.

Clients are determined by the IP address reported in the column `#31#c_ip:1`.

**Question:** How many clients are there in the file? What is their average number of connections?

For each user (i.e., for each unique value of `#31#c_ip:1`), you are required to compute statics on some specific columns/features. You will compute the following characteristics for each client:

1. Number of TCP connections.
2. Sum of all uploaded bytes, i.e., sum of `c_bytes_all:9` for all the connections.
3. Sum of all downloaded bytes, i.e., sum of `s_bytes_all:23` for all the connections.
4. Sum of the number of retransmitted bytes from server, i.e., sum of `s_bytes_retx:25` for all the connections.
5. Average round trip time from server to client, computed as the average of `s_rtt_avg:52` in all the connections.
6. Average time for processing and returning the first segment with payload since the first flow segment from server to client, i.e., average of `s_first:33` in all the connections.

The first three features measure how much traffic the users are generating, while the last three features measure the quality of the connections they have.

Now you are required to cluster users according to these 6 features. Perform the following steps within a pipeline:

- Preprocess the dataset and manage accordingly the 6 features.
- Try to use the following two models for creating the cluster from `pyspark.ml.clustering`: k-means and GMM.
- Evaluate its performance with an `evaluator`, in particular focusing on `silhouette`.
- Tune the parameters of the clustering algorithms.
- Try different values for the number of clusters (i.e., k) and provide comments, intuitions and `interpretation on the obtained clusters.`

## How to write and submit the report

In your report, you must answer to all questions, report the code and scripts the that you have written, and show the output.

You are are required to comment each instruction (or group of instructions) - i.e., what it the goal of the piece of code. You must follow the order in which questions and exercises are posed.

Your report must be a PDF. It can be directly generated from a Jupyter notebook. Go on `File-> Export Notebook As ... -> Export Notebook As PDF`. It must be submitted through the Teaching Portal course page (didattica.polito.it).

**Naming convention:** The file must be named with the following schema:

```
s<id>_lab<n>.pdf
```

For example, if you student ID is 123456 and you are submitting lab 4, the file must be: `s123456_lab4.pdf` Reports with a wrong file name will **NOT** be considered.

**Report lenght:** the report length must not exceed 10 pages, otherwise it will **NOT** be considered valid.