Building a Platform for Data Science Competitions on Data Streams

Author: Dihia BOULEGANE

Supervisor: Mr. Albert BIFET
Organize a competition

- Provide training and test data files to be downloaded
- Receive predictions in the form of csv files
- Evaluate results and determine winner at the end
OBJECTIVE

- Build a platform that can serve many streams to many competitors
- Build a platform that is able to receive many streams and process them
- Build an engine that is able to evaluate user submissions and provide results in real time
- Create a user friendly web application that is able to display results in real time
OFFLINE VS ONLINE LEARNING

OFFLINE

TRAINING STAGE

1 2 ... m  

m+1 m+2 ... n

1 2 ... t

TEST SET

PREDICTION STAGE

1 2 ... u  

u+1 u+2 ... u+k

CHANGE

RE-TRAINING STAGE

Target

Features

TRAINING STAGE

PREDICTION STAGE

RE-TRAINING STAGE

TRAINED MODEL

RE-TRAINED MODEL
DESIGN AND IMPLEMENTATION
CONCEPTUAL VIEW

HOW IT WORKS

Data Sources

Servers

Stream Server

Resource Server

Auth. Server

Data Layer

Application

User

Admin

Bi-directional Stream

Secret

Request + Token

Login

Token

Check login

Secret

Subscribe

Request + Token

Login

Token

Check login
TECHNICAL ARCHITECTURE

- **Data Sources**: CSV, Kafka Cluster, gRPC server, Flask
- **Servers**: Producer, Consumer, Resource Server, Auth. Server
- **Application**: HTML, CSS, JavaScript, AngularJS
- **Data Layer**: MongoDB, MySQL
Resource Server is the module (API) responsible for handling client web application requests to browse data in Two different ways:

- **REST:** handles calls of type get/post/update or delete
- **Server Sent Events:** initiates streams of data towards client web application, this is used to ensure real time updates of charts.
TECHNICAL ARCHITECTURE
 RESOURCE SERVER

REST API: Create a data stream

- Provide a unique name
- Provide a description of the data
- Provide a csv file to be uploaded on the server

Add Datastream

Name
RTC Rgression

Description
Daily Energy Consumption
24/255

Choose File previ_conso18-04-2017.csv

CANCEL ADD
TECHNICAL ARCHITECTURE

RESOURCE SERVER

REST API : Create a competition

- Provide a unique name
- Select a Data Stream from the list
- Specify all targets and for every target the measures
- Specify time parameters
- Select a Protobuf file to describe data instances and predictions
REST API: Subscribe to a competition

- User can subscribe to a competition as long as its end date hasn’t been reached
- User gets a secret token that must be used in authentication credentials to stream server
Server Sent Event: Competition results

- User sends a request to server
- Server initiates stream toward client web application
- Client treats every instance as an event and triggers a function
TECHNICAL ARCHITECTURE
STREAM SERVER

Stream Server is the module responsible for:
• pulling data from sources
• serving data streams to different consumers
• receiving data streams from multiple competitors

Technologies:
• Kafka as pub/sub messaging system
• gRPC to handle bi-directional stream
• Protobuf to serialize data
**TECHNICAL ARCHITECTURE**

**CLIENT APPLICATION**

**Web Application** is a Single Page Application that allows users to browse data through a user friendly UI.

**Stream Consumer** is a simple code that can be written in any language supported by gRPC. It’s used to maintain a bi-directional stream with
THANK YOU FOR YOUR ATTENTION
QUESTIONS ?