GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Big Data Technologies 17/18 – Implementation Phase

Taxi price prediction

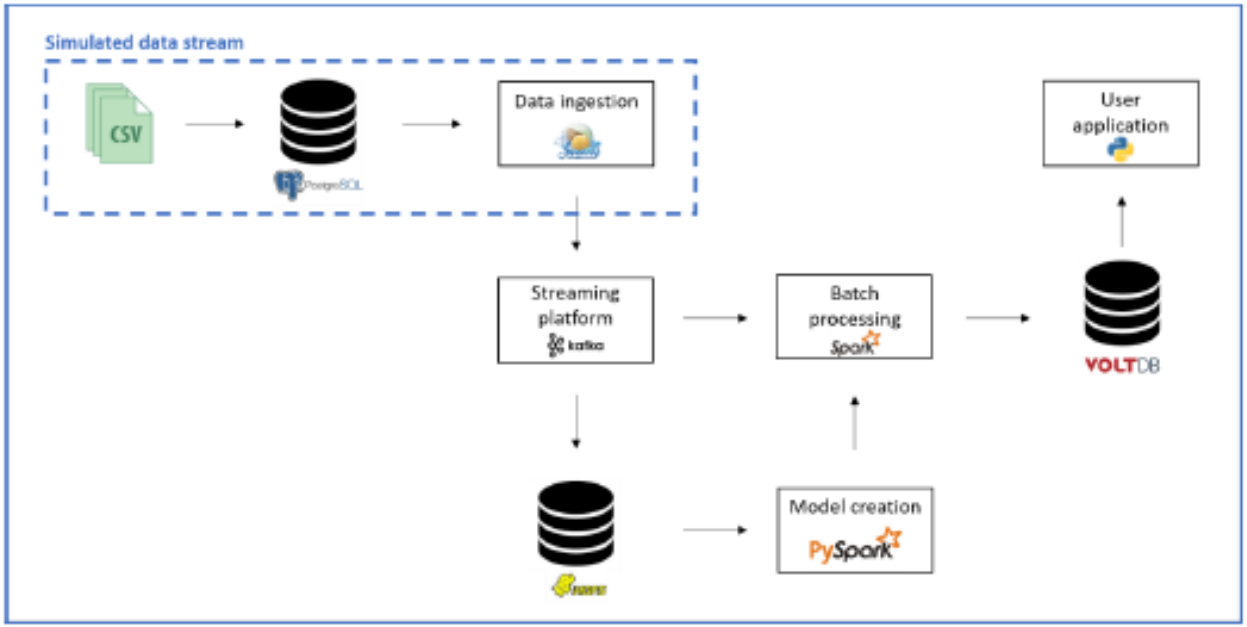
By Luis Severien Marcilla and Jelena Stankovic

Content

- Project Objective
- Implementation
  - Architecture and live presentation
  - Modelling
- Problems

Project Objective

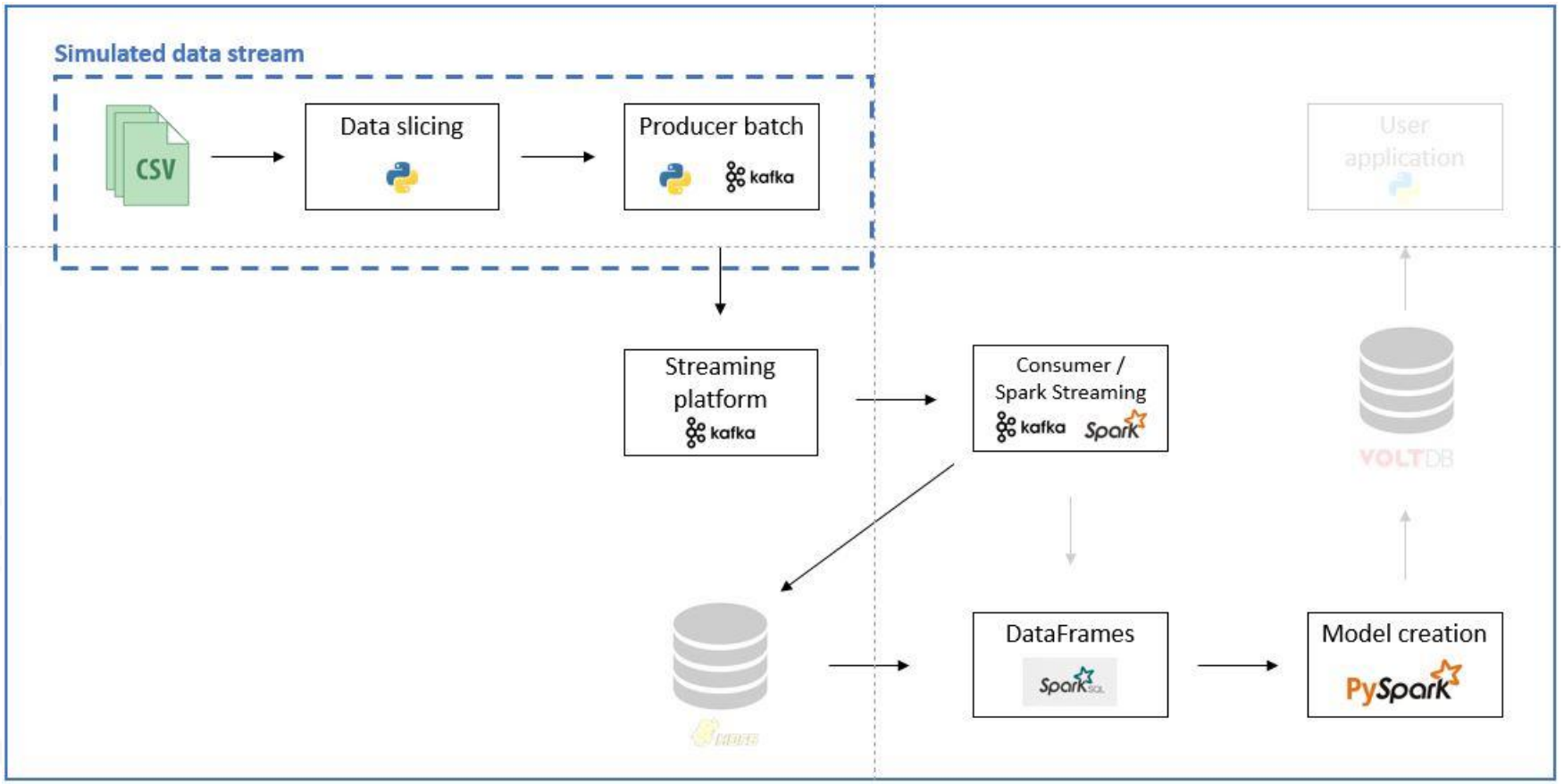
- Data Source: NYC Taxi Dataset → Static source with huge files
- The idea was to predict the total taxi price by decomposing it and estimating the time spent in traffic (→ problems later)
- Former idea for the architecture:



Basic project architecture

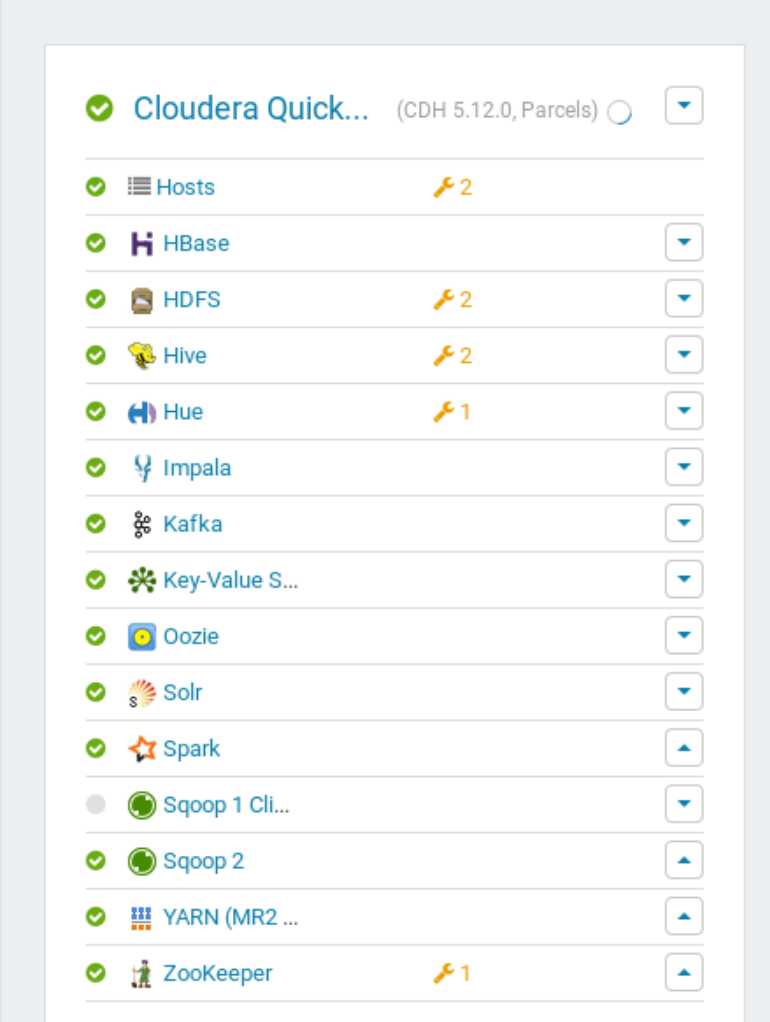
Implementation

- Actual architecture:



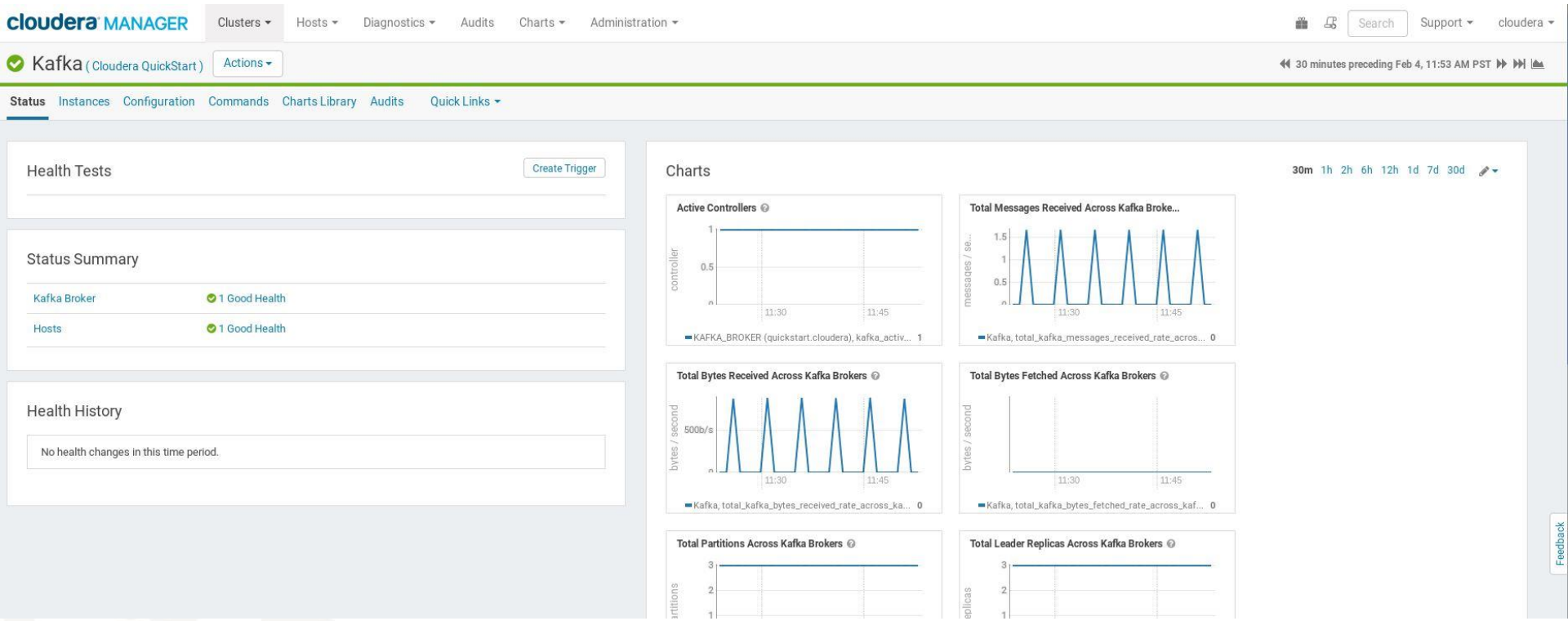
Implementation Prerequisites

- Cloudera Quickstart VM 5.12
- CDH 5.12, Kafka, Spark installed and aditionally Yarn and Zookeeper as active Services
- Python2.7 and the Jupyter Notebook (or Anaconda)
- Libraries and packages: pandas, sklearn, numpy, kafka and csv databricks (package for pyspark for the preliminary architecture)



Kafka Live

Live presentation



Spark – cannot be shown live ☹

```
df = sqlContext.read.load('taxidata_test.csv', format='com.databricks.spark.csv', header='true', inferSchema='true')
#df.dtypes - checks the types of the inferred schema - for this dates correct - pickup_datetime has to be a timestamp


df = (df.withColumnRenamed('PULocationID', 'pu_id').withColumnRenamed('DOLocationID', 'do_id'))


#This is the data relevant for modelling; The hour from datetime is extracted, as we assume this one relevant
#for possible delay in traffic

df_selection = df.select('pu_id', 'do_id', 'trip_distance', 'total_amount', hour('tpep_pickup_datetime').alias('hour'))

#the data is transformed into a pandas dataframe
df_pandas = df_selection.toPandas()
```

Modelling

- Data is processed in Spark data frames and the necessary pieces are fetched into a Pandas data frame.
- Regression with Y = Taxi fare and X = trip distance and a rushhour dummy
- Calculation of goodness of fit: R^2 and the mean squared error
- Dynamic (to be implemented): As new data comes in a new model is to be calculated and compared to the old one (calculate new R^2 and MSE and use the model with the better fit)

```
X_train = df_X[:-100]
X_test = df_X[-100:]

# Split the targets into training/testing sets
Y_train = df_Y[:-100]
Y_test = df_Y[-100:]

from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, mean_squared_error

lr = LinearRegression()

lr.fit(X_train,Y_train )
print('Coefficients: \n', lr.coef_)
```

Problems

- The VM required too much RAM. It usually became slow when the services were turned on and one tried to process any serious amount of data.
- Setting up the environment with compatible versions of the tools turned out to be the most time consuming task.
- Spark streaming could not be implemented as structured streaming was only available in later versions (Spark 2.2+)
- The properties of the data forced us to change our modelling ideas a couple of times (price could not be decomposed as it did not follow the formula we assumed for many data points)