DEPLOYMENT REQUIREMENTS

1-Any Hadoop distribution on centos6 distribution (I used cloudera CDH 5) with YARN installed in at least 2 nodes server (2 nodes is optional you can also run standalane mode spark, please refer to calculator.py to **change from yarn-client mode to local mode/spark stand-alone mode**) Please create a folder as "jpmorgan/portfolios/ " (defined in calculator.py) under directory profile and place the following file as portfolio.tsv:



1-Spark 1.3 should be installed, Spark on should be installed in each node

*[root@bigdata1 ~]# cd spark-1.2.0.2.2.0.0-82-bin-2.6.0.2.2.0.0-2041*

*[root@bigdata1 spark-1.2.0.2.2.0.0-82-bin-2.6.0.2.2.0.0-2041]# export YARN\_CONF\_DIR=/etc/hadoop/conf*

*[root@bigdata1 spark-1.2.0.2.2.0.0-82-bin-2.6.0.2.2.0.0-2041]#  ./bin/spark-shell --master yarn-master --driver-memory 256m --executor-memory 256m*

3-Python 2.7 in all servers (please ensure that they are the primary python installations, because in remote node there might be version conflicts) Then please install the following packages:

Mandatory

pip2.7 install django

pip2.7 install django\_tables2

pip2.7 install numpy

pip2.7 install SimPy

Optional

pip2.7 install scipy

pip2.7 install statsmodels

pip2.7 install Matplotlib

pip2.7 install networkx

(If there is a problem please refer to me [anil.sener@student.ie.edu](mailto:anil.sener@student.ie.edu) )

4-Please install PyCharm IDE and open the /mortgagebalanceforecastingengine project file (file->open) in the package (pycham is run by run.sh). Please don’t forget to change python interpreter settings, by File->Settings->Project:jpmorgan->Project Interpreter as python2.7 and please press apply. Finally please check on right edge of pycharm there is database please check sql lite is available with its tables.

SQLlite Table SQLs are provided below:

CREATE TABLE default.portfolios(

ID INT,

Channel STRING,

Origination\_Date STRING,

State STRING,

Original\_Loan\_Amount BIGINT,

OLTV INT,

OFICO INT,

Term INT,

Annual\_Interest\_Rate DECIMAL,

Balance DECIMAL,

Age INT)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '\t'

LOCATION '/user/anil/jpmorgan/portfolios';

LOAD DATA INPATH '/user/anil/jpmorgan/portfolio.tsv' OVERWRITE INTO TABLE default.portfolios;

DROP TABLE forecastingengine\_macroecon;

CREATE TABLE forecastingengine\_macroecon

(

id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,

Date DATE NOT NULL,

HPI\_NY DECIMAL NOT NULL,

HPI\_CA DECIMAL NOT NULL,

MTG DECIMAL NOT NULL,

userID INTEGER NOT NULL,

isForecast BOOLEAN NOT NULL,

estimation DECIMAL DEFAULT 0

);

DROP TABLE auth\_user;

CREATE TABLE auth\_user

(

username VARCHAR NOT NULL,

password VARCHAR NOT NULL,

email VARCHAR NOT NULL,

last\_login DATETIME,

is\_superuser BOOLEAN,

first\_name VARCHAR NOT NULL,

last\_name VARCHAR NOT NULL,

is\_staff BOOLEAN,

is\_active BOOLEAN,

date\_joined datetime NOT NULL

);

DROP TABLE django\_session;

CREATE TABLE django\_session

(

id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,

session\_data VARCHAR NOT NULL,

session\_key VARCHAR NOT NULL,

expire\_date DATETIME

);

5- In case of multiple python distribution installations please change spark-env.sh as below (/opt/cloudera/parcels/CDH/lib/spark/conf or /etc/spark/conf) and after restart please restart spark masters and slaves with ./start-all.sh and ./start-all.sh (/opt/cloudera/parcels/CDH/lib/spark/sbin or /etc/spark/sbin)

#!/usr/bin/env bash

##

# Generated by Cloudera Manager and should not be modified directly

##

export SPARK\_HOME=/opt/cloudera/parcels/CDH-5.3.0-1.cdh5.3.0.p0.30/lib/spark

export DEFAULT\_HADOOP\_HOME=/opt/cloudera/parcels/CDH-5.3.0-1.cdh5.3.0.p0.30/lib/hadoop

export PYTHONPATH=$SPARK\_HOME/python:$SPARK\_HOME/python/build:$PYTHONPATH

export PYSPARK\_PYTHON=python2.7

export PATH=/usr/local/bin:$PATH

### Path of Spark assembly jar in HDFS

export SPARK\_JAR\_HDFS\_PATH=${SPARK\_JAR\_HDFS\_PATH:-/user/spark/share/lib/spark-assembly.jar}

### Let's run everything with JVM runtime, instead of Scala

export SPARK\_LAUNCH\_WITH\_SCALA=0

export SPARK\_LIBRARY\_PATH=${SPARK\_HOME}/lib

export SCALA\_LIBRARY\_PATH=${SPARK\_HOME}/lib

export HADOOP\_HOME=${HADOOP\_HOME:-$DEFAULT\_HADOOP\_HOME}

if [ -n "$HADOOP\_HOME" ]; then

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:${HADOOP\_HOME}/lib/native

fi

export HADOOP\_CONF\_DIR=${HADOOP\_CONF\_DIR:-/etc/hadoop/conf}

**USAGE GUIDELINES**

1-Build the application as follows (before you can change .setMaster to local or yarn-client in calculator.py, and you can improve memory and cpu configurations as your platform supports) I advice to firstly test in local mode because both environments should be identical in terms of spark and python installations to run yarn-client successfully as in calculator.py:

conf = (SparkConf()

.setMaster("yarn-client")

#.setMaster("local")

#you can shift between local and yarn-client mode, it is very important to have same python2.7 version in all servers

.setAppName("forecastingengine")

.set("spark.executor.memory", "512m")

.set("spark.driver.memory", "512m")

.set("spark.python.worker.memory", "4g") #My environment didnt support more please increase to 8gb

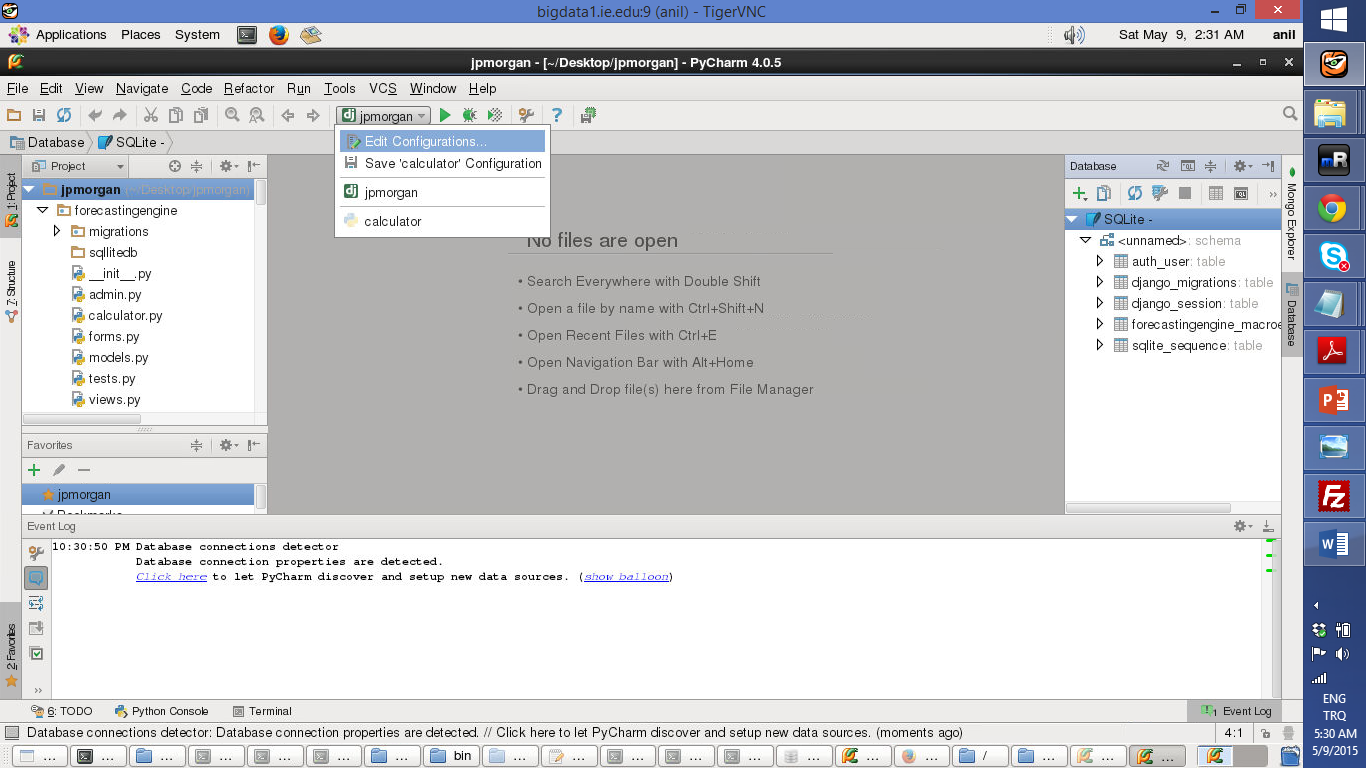
.set("spark.shuffle.memoryFraction", 0.4)

.set("spark.default.parallelism",2) #I am using only 2 nodes for execution

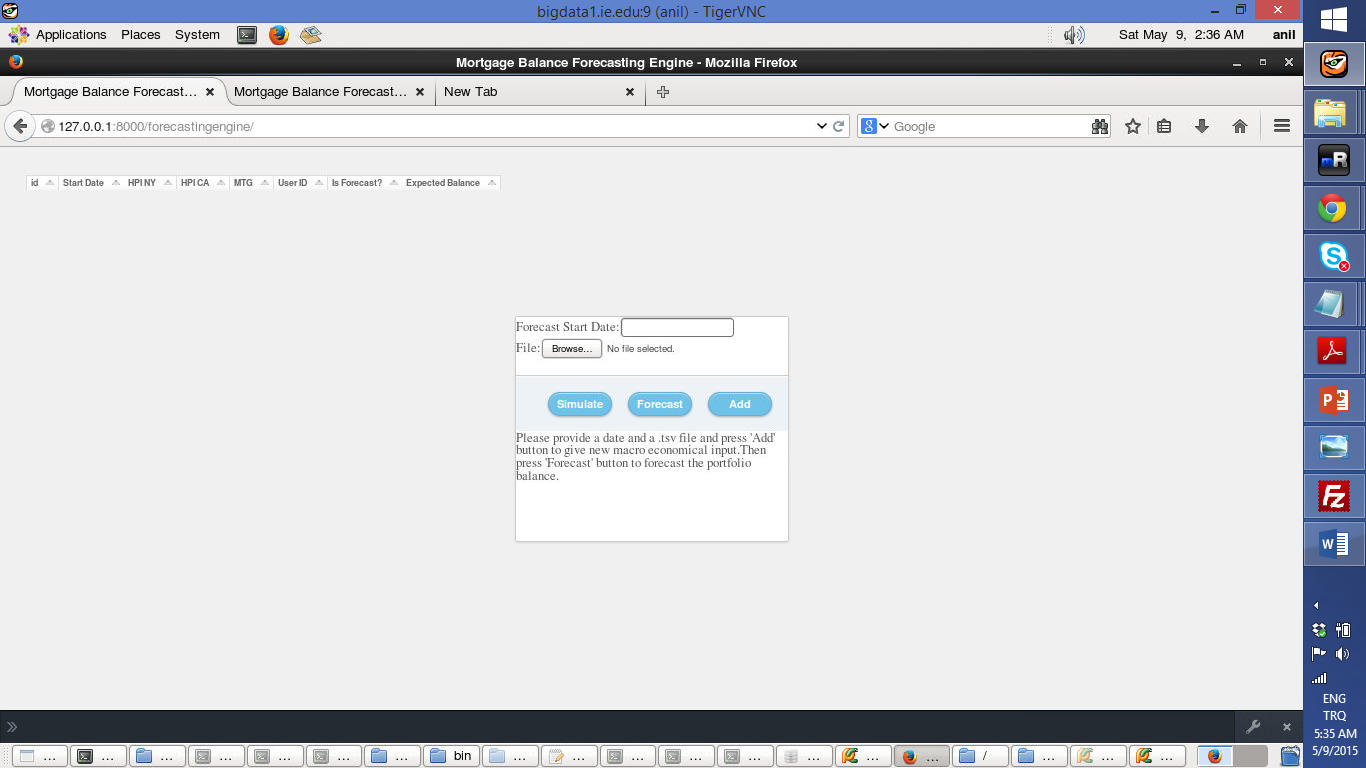
.set("spark.executor.instances", 2)

.set("spark.executor.cores", 2) #My environment didnt support more please increase to 4 cores

)

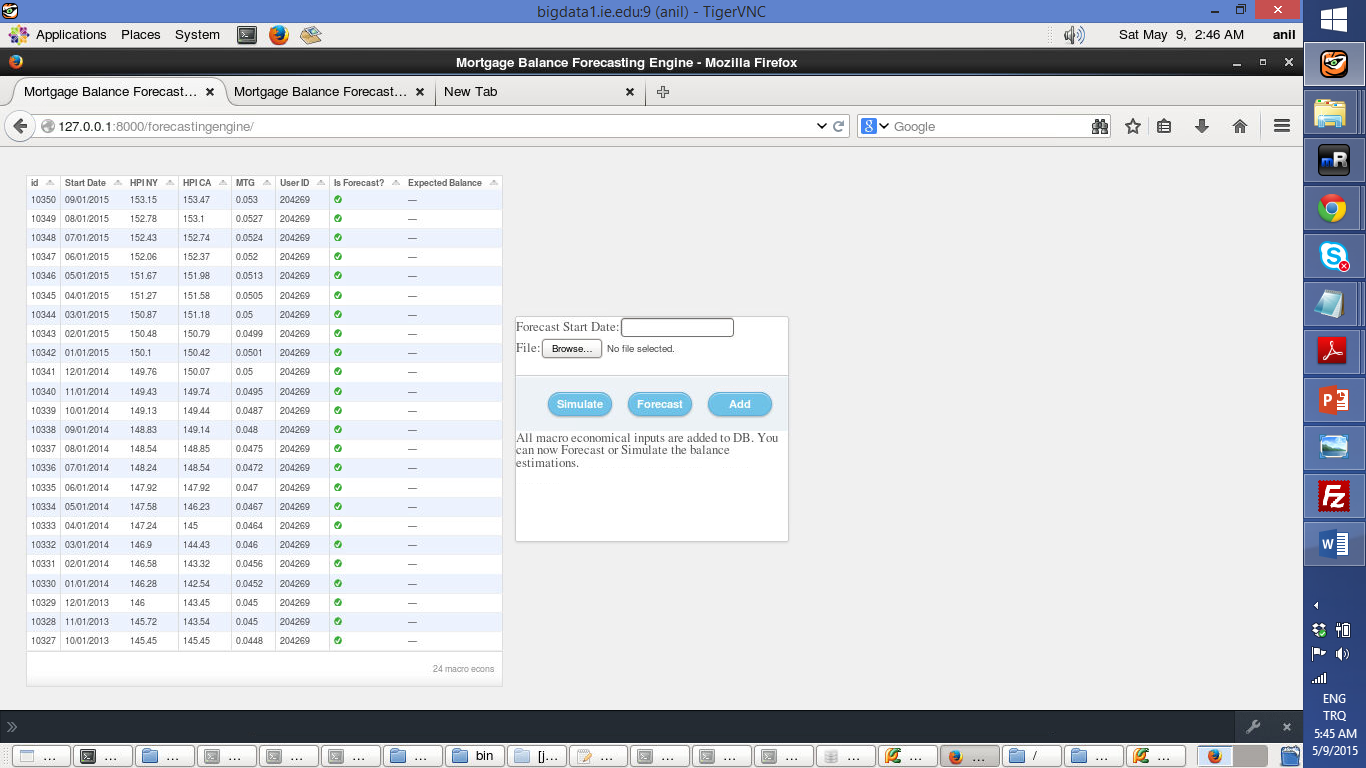


2-Please open Mozzila firefox and login to the application with <http://127.0.0.1:8000/forecastingengine/> link.

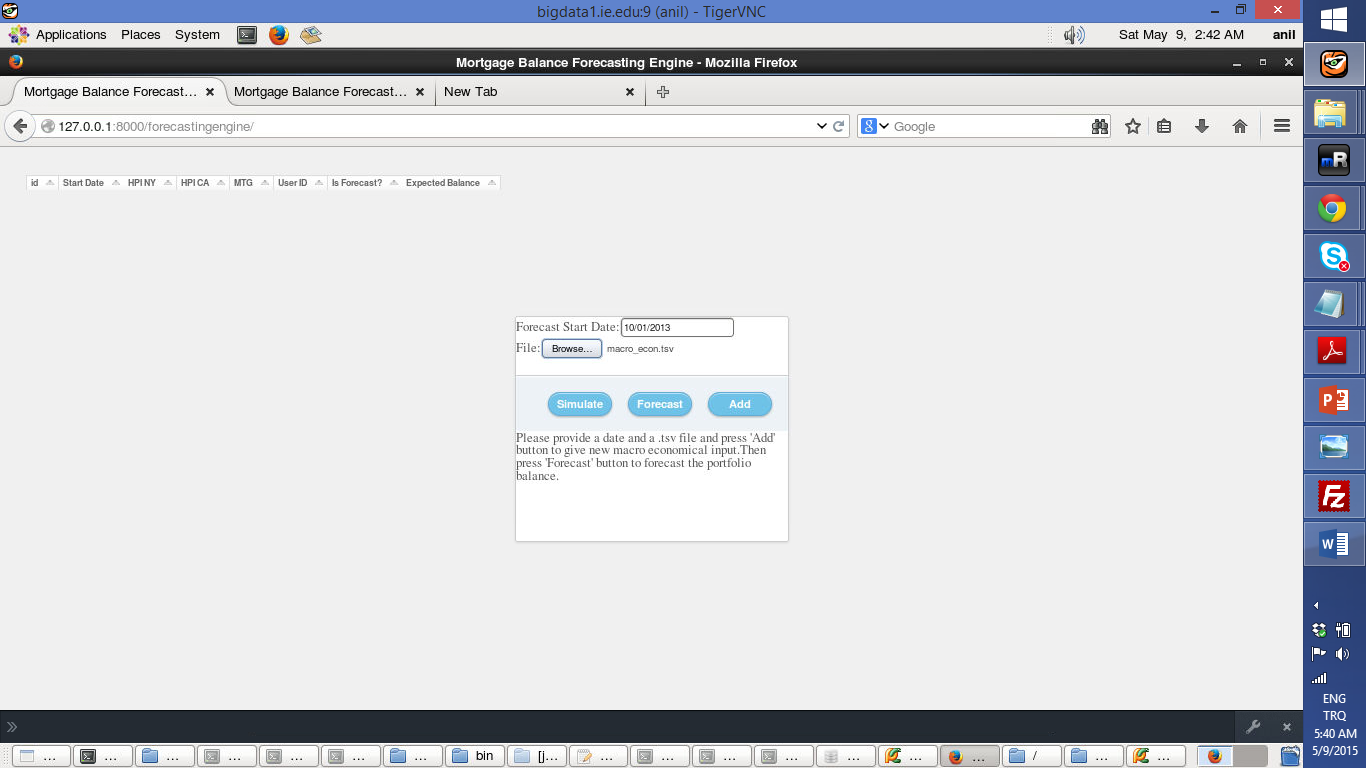


3-Please enter a forecast start as 10/01/2013 (Please us this format, I didn’t want to spend time on format checks other than Django provides by default) and browse macro\_econ.tsv file below to upload macro economic inputs and press Add.

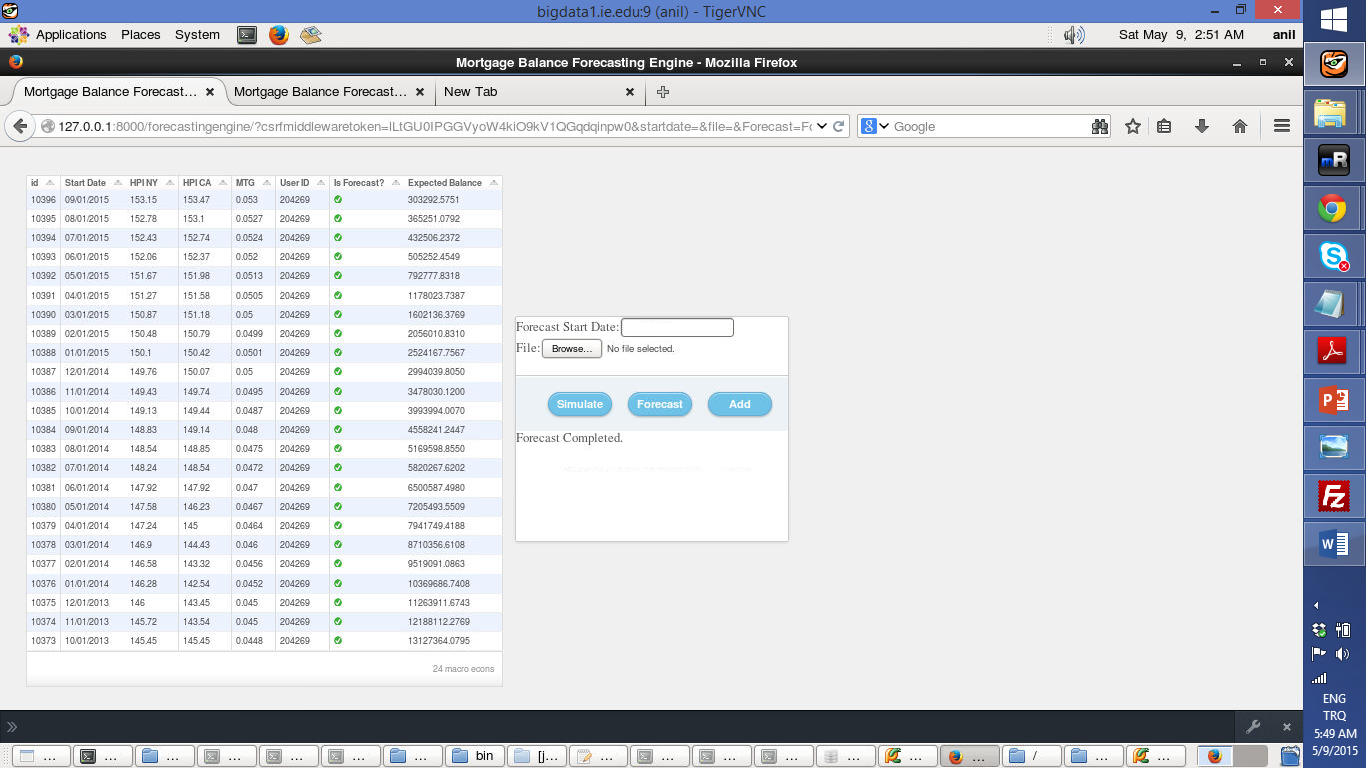




4-After the inputs are added you can press Forecast (Problem 2) or Simulate (Bonus) as you like:



5- The output for forecast:



6-The output for simulate:

