Alexander Rasho

The goal of this project is to redo (only) the cross-validation part of Project 1 but on an Apache Spark platform running on a local Linux machine. The two main tasks are:

1. Install all packages on a Linux computer (or virtual machine) required for Spark and connecting R to Spark
2. Run R code that uses Spark to perform 10-fold cross-validation of your best model from Project 1

Installing Apache Spark and R:

It is easiest to install into a Linux (virtual) machine. Then install R or Rstudio, install the SparkR package inside R/Rstudio, and then use SparkR's install.spark() function to do the Spark installation. The following instructions are for Ubuntu Linux[1].

1. To install R, from the Linux command line:
```
> sudo apt install r-base
```
2. To install RStudio:
   > Download the latest version (as a .deb file) from https://www.rstudio.com/products/rstudio/download/#download
```
> sudo apt install gdebi
> sudo gdebi <location of downloaded rstudio .deb file>
```
3. Install the SparkR package inside R/Rstudio:
```
> install.packages("SparkR")
```
4. Install spark from inside R/Rstudio[2]
```
> library(SparkR)
> install.spark()
```

Cross-validation in Spark

This code will be very similar to the code shown in class on May 2 (and attached here). You can edit this code for your specific model. Use the data file attached with this assignment (same data as from Project 1 but the ambiguity with column name "FF" removed with two names "FF1" and "FF2" - the goal is to predict FF2).

[1] You may want to try out "Tuffix", the Titan-branded version of Ubuntu 18.04. Instructions on how to install Tuffix or a Tuffix-based VM are in the Tuffix Titanium Community for Students, https://communities.fullerton.edu/course/view.php?id=1547 (also the best venue to receive help with Tuffix).

[2] The above instructions install spark to a local folder. You can also install to a system-wide folder, a install a specific version of Spark, or use an existing installation of Spark (email for help if you want to do this).

1. To Initialize a Spark session within R. Run the following commands:
   - > `library(SparkR)`
   - > `sparkR.session(master = "local[*]", sparkConfig = list(spark.driver.memory = "2g"))`
   - *Note: several warning messages are normal*
   - *Take note of the IP address of the Spark web UI that can be used to monitor the status of Spark jobs. Example:*

     `… using `***`10.0.2.15`***` instead ...`
2. Now you can run the commands in project2.R. Change the model variables to match the independent variables from your Project 1.

Submission:

1. Prepare a short report containing only the following:
   a. The variables that you selected
   b. The root mean squared error of the model prediction
   c. A screen capture image of the running Apache Spark web UI. It is by default at port 4040 (but maybe 4041, 4042, … if you have multiple Spark sessions active). The IP address of the UI is then:

      http://10.0.2.15:4040

REPORT

****

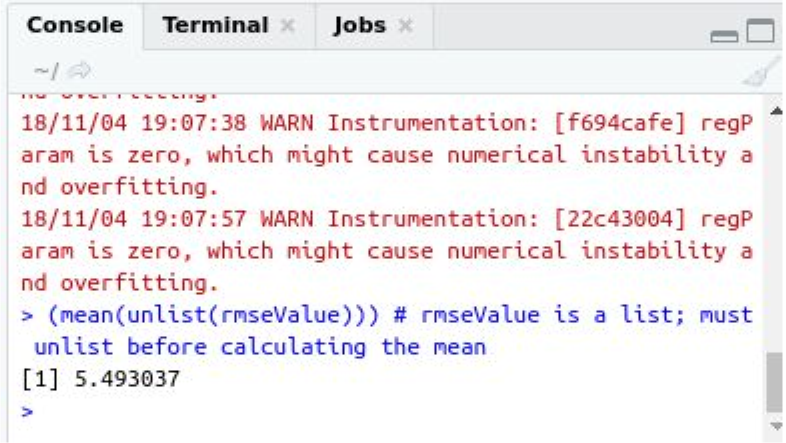
The variables selected were:
HR1,Wt,RPE1,VC,LGM,RGM,HR2,Ht,PL1,RF.2

R code:

```
> mydata <- read.df("/home/student/Downloads/Health_Sciences_Data_File_project1.csv", "csv", header="true", inferSchema = "true", na.strings = "NA")
> mydata1 <- select(mydata, "FF2", "HR 1", "Wt", "RPE 1", "VC", "LGM", "RGM", "HR 2", "Ht", "PL 1", "RF 2")
>colnames(mydata1)[2] <- "HR1"
>colnames(mydata1)[4] <- "RPE1"
>colnames(mydata1)[8] <- "HR2"
>colnames(mydata1)[10] <- "PL1"
>colnames(mydata1)[11] <- "RF2"
```

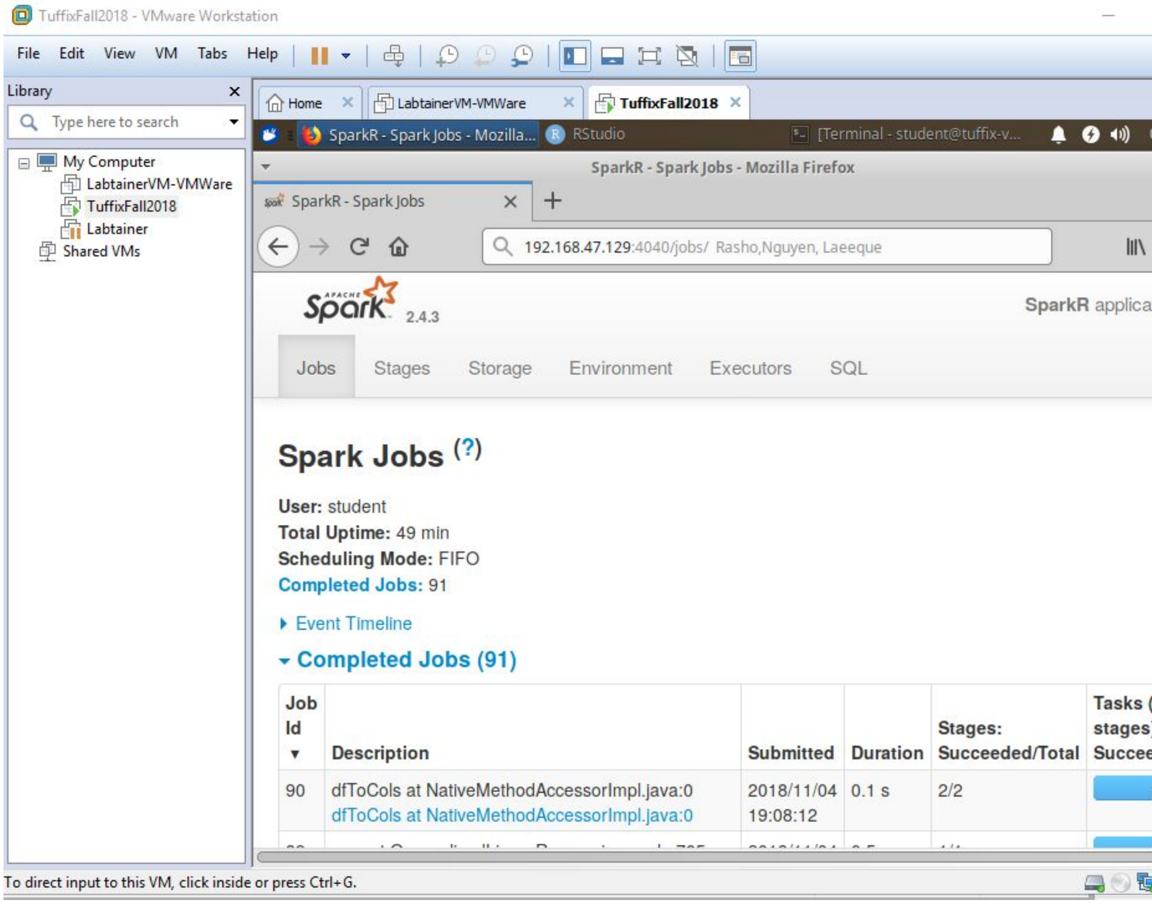
```
>traintest <- randomSplit(mydata1, c(1,1,1,1,1,1,1,1,1,1))

>> for (iter in 1:4) {
+         testdata <- traintest[[iter]]
+
+         #traindata <- rbind(traintest[[2]], traintest[[3]], traintest[[4]])
+         traindata <- do.call(rbind, traintest[-iter])
+
+         mod <-
glm(FF2~HR1+Wt+RPE1+VC+LGM+RGM+HR2+Ht+PL1+RF2,
data=traindata, family="gaussian")
+         preds <- predict(mod, testdata)
+         preds$err <- (preds$FF2 - preds$prediction)^2
+         rmseValue[iter] <- collect(select(preds, sqrt(mean(preds$err))))
+ }
> (mean(unlist(rmseValue)))
```

The root mean squared error is: **5.493037**



```
18/11/04 19:07:38 WARN Instrumentation: [f694cafe] regP
aram is zero, which might cause numerical instability a
nd overfitting.
18/11/04 19:07:57 WARN Instrumentation: [22c43004] regP
aram is zero, which might cause numerical instability a
nd overfitting.
> (mean(unlist(rmseValue))) # rmseValue is a list; must
 unlist before calculating the mean
[1] 5.493037
>
```

Screen Capture of Spark WebUI from VMWare running with Tuffix


TuffixFall2018 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
LabtainerVM-VMWare
TuffixFall2018
Labtainer
Shared VMs
Home
LabtainerVM-VMWare
TuffixFall2018
SparkR - Spark Jobs - Mozilla...
RStudio
[Terminal - student@tuffix-v...
SparkR - Spark Jobs - Mozilla Firefox
SparkR - Spark Jobs
192.168.47.129:4040/jobs/ Rasho,Nguyen, Laeeque
Spark 2.4.3
SparkR applica
Jobs
Stages
Storage
Environment
Executors
SQL
To direct input to this VM, click inside or press Ctrl+G.

Spark Jobs (?)

User: student
Total Uptime: 49 min
Scheduling Mode: FIFO
Completed Jobs: 91

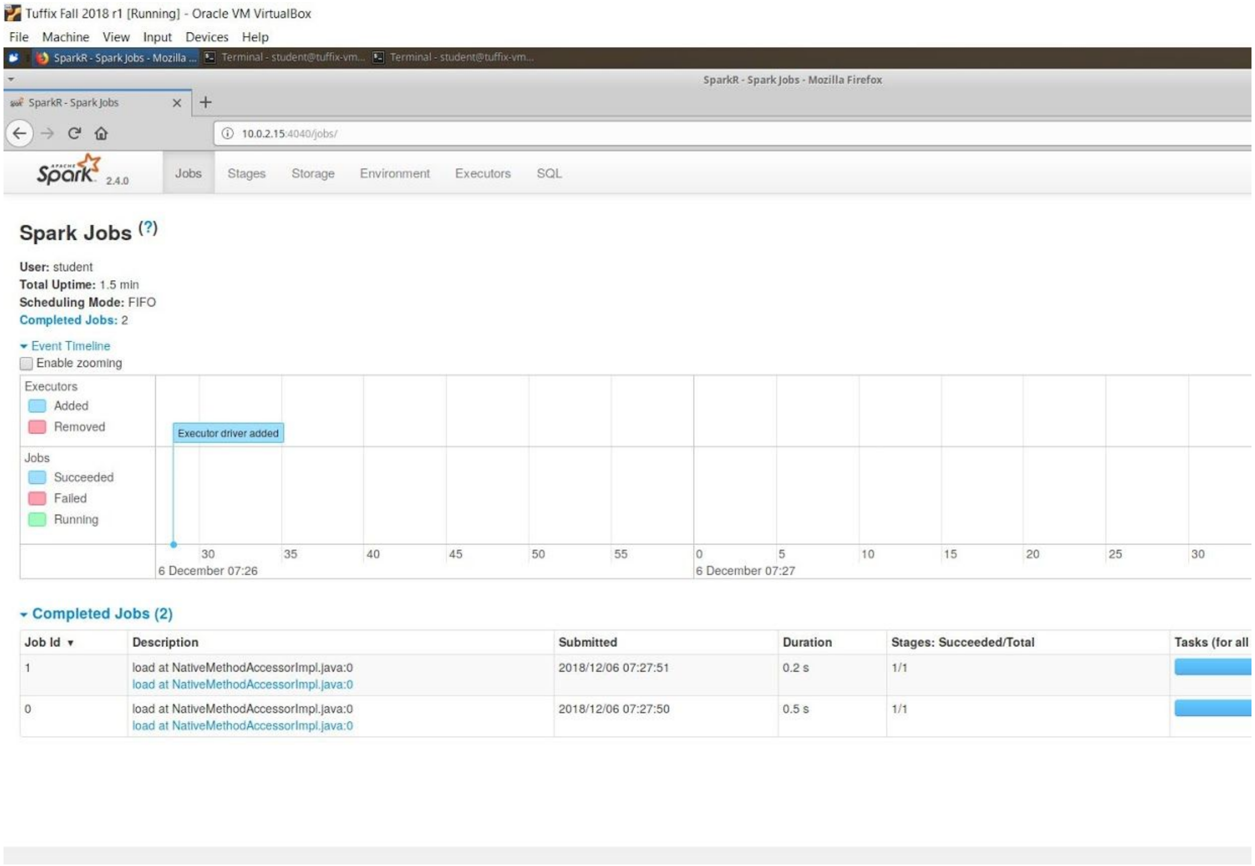
▸ Event Timeline

▾ **Completed Jobs (91)**

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (stages) Succee
90	dfToCols at NativeMethodAccessorImpl.java:0 dfToCols at NativeMethodAccessorImpl.java:0	2018/11/04 19:08:12	0.1 s	2/2	

****

An example is shown below:


Tuffix Fall 2018 r1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
SparkR - Spark Jobs - Mozilla ...
Terminal - student@tuffix-vm...
Terminal - student@tuffix-vm...
SparkR - Spark Jobs - Mozilla Firefox
SparkR - Spark Jobs
10.0.2.15:4040/jobs/
Spark 2.4.0
Jobs
Stages
Storage
Environment
Executors
SQL

Spark Jobs (?)

User: student
Total Uptime: 1.5 min
Scheduling Mode: FIFO
Completed Jobs: 2

Completed Jobs (2)

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all
1	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2018/12/06 07:27:51	0.2 s	1/1	
0	load at NativeMethodAccessorImpl.java:0 load at NativeMethodAccessorImpl.java:0	2018/12/06 07:27:50	0.5 s	1/1	

Due date:

Friday 5/17, 11:55 pm on Titanium. Submit a single PDF file.

Group work:

You may work in groups of 1-3. Include all group member names in the PDF file. Only one person in the group needs to submit to Titanium.