Hadoop and Hive

by Igor Baranov

Abstract We are using a dataset was scraped from several websites in Czech Republic and Germany over a period of more than a year. The dataset contains over 3.5 million records and has lots of missing data. The goal is to apply such big data tools as Hadoop and Hive to load and query the data and prepare it to the analysis. Diring the analysis the questions like what is the most advertised vs sol car, car maker and car model vere answered.

$if(highlighting - macros)\ highlighting - macros\ endif$

Introduction

The goals of the project are: * To use apache HIVE as an analytic tool to analyze realistic data. * To acquare experience working with opened end problems, that are similar to problems that we will face in our career as data professional. * At the end of this project we should: - Gain sufficient confidence in using Hadoop and Apache Hive - Gain an appetite for working with large datasets. - Be aware of the potential and benefits of analyzing large datasets.

Environment Preparation

Hadoop

HIVE

Loading Classified Ads for Cars Data to Hadoop

Data understanding

The dataset (UCI Wine Data Set) 16 attributes and 3.5 million instances. The data was scraped from several websites in Czech Republic and Germany over a period of more than a year. The scrapers were tuned slowly over the course of the year and some of the sources were completely unstructured, so as a result the data is dirty, there are missing values and some values are very obviously wrong (e.g. phone numbers scraped as mileage etc.) There are roughly 3,5 Million rows and the following columns:

- maker - normalized all lowercase
- model - normalized all lowercase
- mileage - in KM
- manufacture_year
- engine_displacement - in ccm
- engine_power - in kW
- body_type - almost never present, but I scraped only personal cars, no motorcycles or utility vehicles
- color_slug - also almost never present
- stk_year - year of the last emission control
- transmission - automatic or manual
- door_count
- seat_count
- fuel_type - gasoline, diesel, cng, lpg, electric
- date_created - when the ad was scraped
- date_last_seen - when the ad was last seen. Our policy was to remove all ads older than 60 days
- price_eur - list price converted to EUR

This is the link fo the dataset. Download it locally:

Login to the Hortonworks VM and create folder for the project:

```
ssh maria_dev@127.0.0.1 -p 2222
cd used-cars/
exit
```

From the Git bash or Putty console copy the file to the Hortonworks VM:

```
scp -P 2222 classified-ads-for-cars.zip maria_dev@127.0.0.1:/home/maria_dev/used-cars
```

Login to VM, unzip the file and count the number of lines in the created file (should be about 3.5M):

```
ssh maria_dev@127.0.0.1 -p 2222
cd used-cars/
gzip -d --suffix=.zip *.*
wc -l classified-ads-for-cars
```

Copy first line of the file to 'headers' file

```
 head -1 classified-ads-for-cars > headers
```

Split the file into 100 chunks and remove headers line from the first file:

```
mkdir chunks
cd chunks
split --number=l/100 ../classified-ads-for-cars classified-ads-for-cars_
sed -i 1d classified-ads-for-cars_aa
```

Copy the files to hdfs:

Create a directory in hadoop called baranov/cars/classified by using the following command:

```
hdfs dfs -mkdir -p  baranov/cars/classified
```

You can now copy the event files you downloaded earlier to the hdfs directory you just created by running the following commands. Those commands for each file will print the name of the file (to see the progress), then load the file to HDFS and then move the processed file to folder ../loaded-files:

```
mkdir ../loaded-files
for file in *; do echo $file;  hdfs dfs -put $file baranov/cars/classified/; mv $file -f ../loaded-files; done
```

To check how many unloaded files left, run the following commabd from another(!) bash window:

```
ls events/ | wc -l
```

List files copied to hadoop by running the following command:

```
hdfs dfs -ls baranov/cars/classified/
```

Remove chunks

```
cd ..
rm -r -f chunks
rm -f loaded-files/*
rm -r -f loaded-files
```

Creating HIVE database

```
cd ~
mkdir hive
cd hive
vi create-db.sql
```

```
CREATE DATABASE
    IF NOT EXISTS used_cars
    COMMENT 'This is the used cars database'
    With dbproperties ('Created by' = 'baranov','Created on' = 'August-2018');

SHOW DATABASES;

vi create-table.sql

CREATE EXTERNAL TABLE IF NOT EXISTS used_cars.events (
    maker STRING,
    model STRING,
    mileage INT,
    manufacture_year INT,
    engine_displacement INT,
    engine_power INT,
    body_type STRING,
    color_slug STRING,
    stk_year STRING,
    transmission STRING,
    door_count INT,
    seat_count INT,
    fuel_type STRING,
    date_created TIMESTAMP,
    date_last_seen TIMESTAMP,
    price_eur DECIMAL(13,2)
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/maria_dev/cars/classified';

ALTER TABLE used_cars.events SET SERDEPROPERTIES ("timestamp.formats"="yyyy-MM-dd HH:mm:ss.SSSSSSZ");

hive
hive> DESCRIBE used_cars.events;
OK
maker                   string
model                   string
mileage                 int
manufacture_year        int
engine_displacement     int
engine_power            int
body_type               string
color_slug              string
stk_year                string
transmission            string
door_count              int
seat_count              int
fuel_type               string
date_created            timestamp
date_last_seen          timestamp
price_eur               decimal(13,2)
Time taken: 0.59 seconds, Fetched: 16 row(s)

hive> select count (*) from used_cars.events;
Query ID = maria_dev_20180903015614_32d5f61c-9297-497e-83eb-5f6cbb1e3d6b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1535776685382_0027)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
```

```
Map 1 ..........   SUCCEEDED     20         20        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 102.01 s
--------------------------------------------------------------------------------
OK
3552912
Time taken: 110.189 seconds, Fetched: 1 row(s)

hive> quit
```

Analyzing Data

To perform the analysis, certain R libraries were used. The code below was used to load and initialize the library, then loads the data. To pretty-print the tables in this report we used xtable (Dahl, 2016) library.

```
set.seed(42)
library(ggplot2)
library(reshape2)
library(plyr)
library(readr)
library(fpc)
library(data.table)
library(ggplot2)
```

Connecting to USED_CARS database in HIVE running on Hortonworks Sandbox VM

```
options( java.parameters = "-Xmx8g" )
library(rJava)
library(RJDBC)

cp = c("//d:/tools/apache-hive-1.2.2/lib/hive-jdbc-1.2.2-standalone.jar",
       "//d:/tools/hadoop-2.7.7/share/hadoop/common/hadoop-common-2.7.7.jar")
.jinit(classpath=cp)

drv <- JDBC(driverClass = "org.apache.hive.jdbc.HiveDriver",
         classPath = "//d:/tools/apache-hive-1.2.2/lib/hive-jdbc-1.2.2-standalone.jar",
            identifier.quote="`")

conn <- dbConnect(drv, "jdbc:hive2://127.0.0.1:10000/used_cars", "maria_dev", "maria_dev")
```

Discover the USED_CARS database

First we are getting the EVENTS table description. Full description of the attributes presented on Classified Ads for Cars home page.

```
descr <- dbGetQuery(conn, "describe events")
kable(descr)
```

Code below selecting first 10 rows for EVENTS table and saves them to 'cars' dataframe. Note that we rename cars columns for better presentation. Also note that it looks that the dataset has some missing values.

```
cars <- dbGetQuery(conn, "select * from events limit 6")
```

The EVENTS table has 3,552,912 records which could be confirmed by running the HQL statement below.

```
dbGetQuery(conn, "select count (*) from events")
```

For the analysis we will extract not more than 30K rows from EVENT table randomly using the HQL statement below. We skip the rows mising the most important attributes:

	events.maker	events.model	events.mileage	events.manufacture_year	events.engine_displacement	events.eng
1	ford	galaxy	151000.00	2011.00	2000.00	
2	skoda	octavia	143476.00	2012.00	2000.00	
3	bmw		97676.00	2010.00	1995.00	
4	skoda	fabia	111970.00	2004.00	1200.00	
5	skoda	fabia	128886.00	2004.00	1200.00	
6	skoda	fabia	140932.00	2003.00	1200.00	

Table 1: Car Ads Dataset - first rows

```
filter <- paste(
      " maker <> ''",
      " AND model <> ''",
      " AND mileage is not NULL",
      " AND manufacture_year is not NULL",
      " AND price_eur is not NULL"
)
count <- dbGetQuery(conn, paste("select count(*) from events", " WHERE", filter))
cars.sample.totalFilered <- count$`_c0`

car.sample.maxSize <- 30000
lim <- car.sample.maxSize/cars.sample.totalFilered

sample_HQL <- paste(
  "select * from events WHERE", filter,
      " AND rand(123) < ", lim,
  " limit ", car.sample.maxSize
)

cars.sample <- dbGetQuery(conn, sample_HQL)
colnames(cars.sample) <- c(
  "Maker", "Model","Mileage","Year", "Disp", "Pwr", "Body", "Color", "Sticker",
  "Trans", "Doors", "Seats", "Fuel", "Listed", "Removed", "Price")

nrow(cars.sample)
```

```
#> [1] 29958
```

Check for missing values

The dataset has no missing values. Code below calculate number of rows with missing values and checks if there is at list one.

```
any(is.na(cars.sample))
```

```
#> [1] TRUE
```

Creating additional columns for analysis

```
cars.sample$ListedTS <- strptime(cars.sample$Listed, '%Y-%m-%d %H:%M:%OS')
cars.sample$RemovedTS <- strptime(cars.sample$Removed, '%Y-%m-%d %H:%M:%OS')

cars.sample$Age <- as.integer(ceiling(
  difftime(cars.sample$ListedTS, strptime(cars.sample$Year,'%Y'), units = "days")/365))

cars.sample$DaysListed <- as.integer(ceiling(
  difftime(cars.sample$RemovedTS, cars.sample$ListedTS, units = "days")))
```

How long the cars are usually listed?

```
ggplot(cars.sample, aes(x=DaysListed)) +
  geom_density(fill="#FF6666", alpha=.1) +
  geom_vline(aes(xintercept=42), color="blue", linetype="dashed", size=1) +
  geom_vline(aes(xintercept=60), color="red", linetype="dashed", size=1)
```

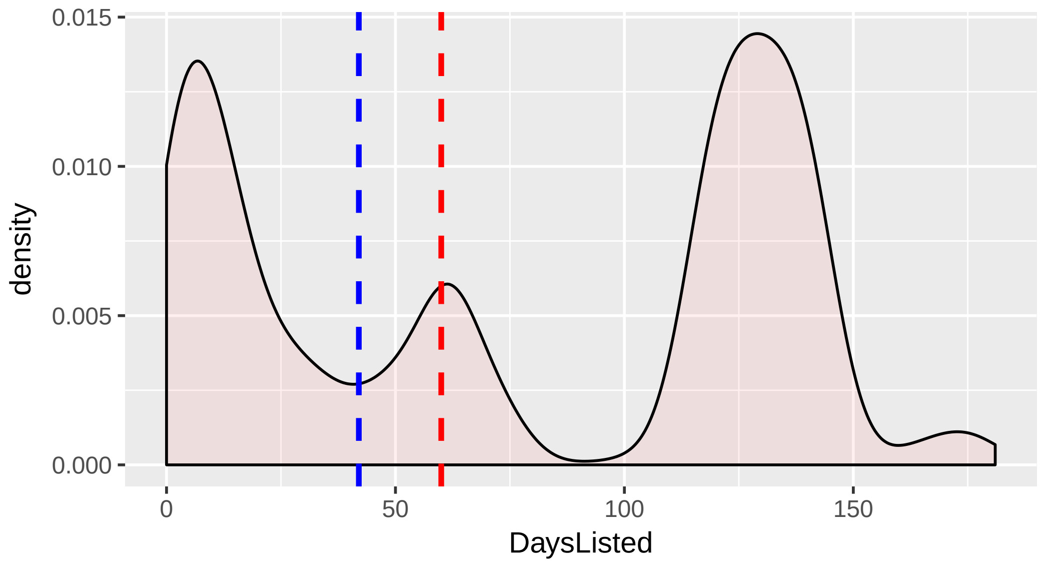


Figure 1: Days Cars Listed

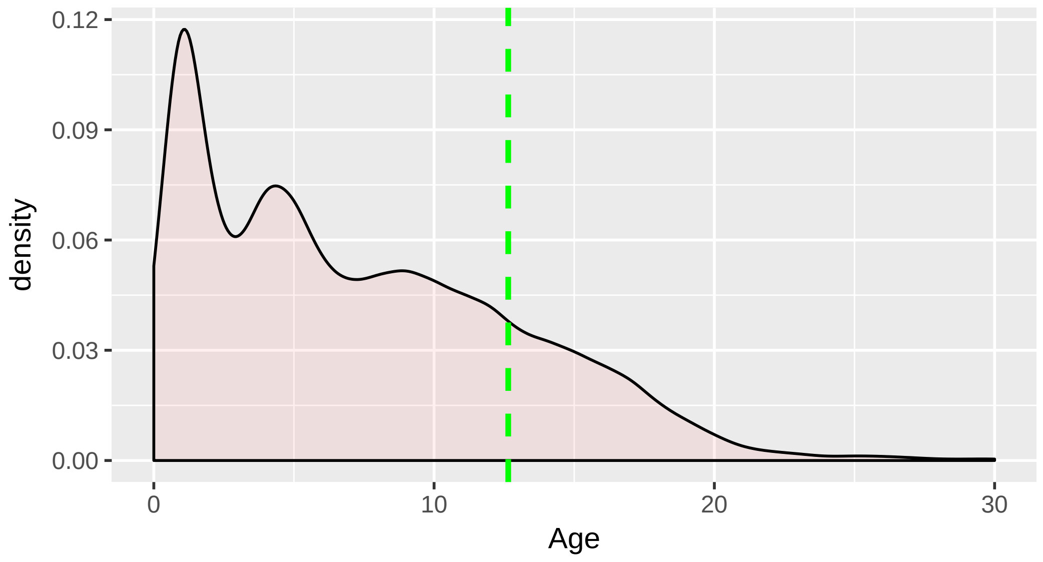


Figure 2: Number of Ads by Maker

Let's consider cars listed less than 42 days (6 weeks) to be sold

```
cars.sample$Sold <- cars.sample$DaysListed <= 42
```

What is the distribition of advertized cars age?

```
ggplot(cars.sample, aes(x=Age)) +
  geom_density(fill="#FF6666", alpha=.1) +
  scale_x_continuous(limits = c(0, 30))+
    geom_vline(aes(xintercept=mean(Age, na.rm=T)),
               color="green", linetype="dashed", size=1)
```

What is the distribition of mileage of the sold cars?

```
ggplot(cars.sample, aes(x=Mileage)) +
  geom_density(fill="#FF6666", alpha=.1) +
  scale_x_continuous(limits = c(0, 250000))+
    geom_vline(aes(xintercept=mean(Mileage, na.rm=T)),
               color="green", linetype="dashed", size=1)
```

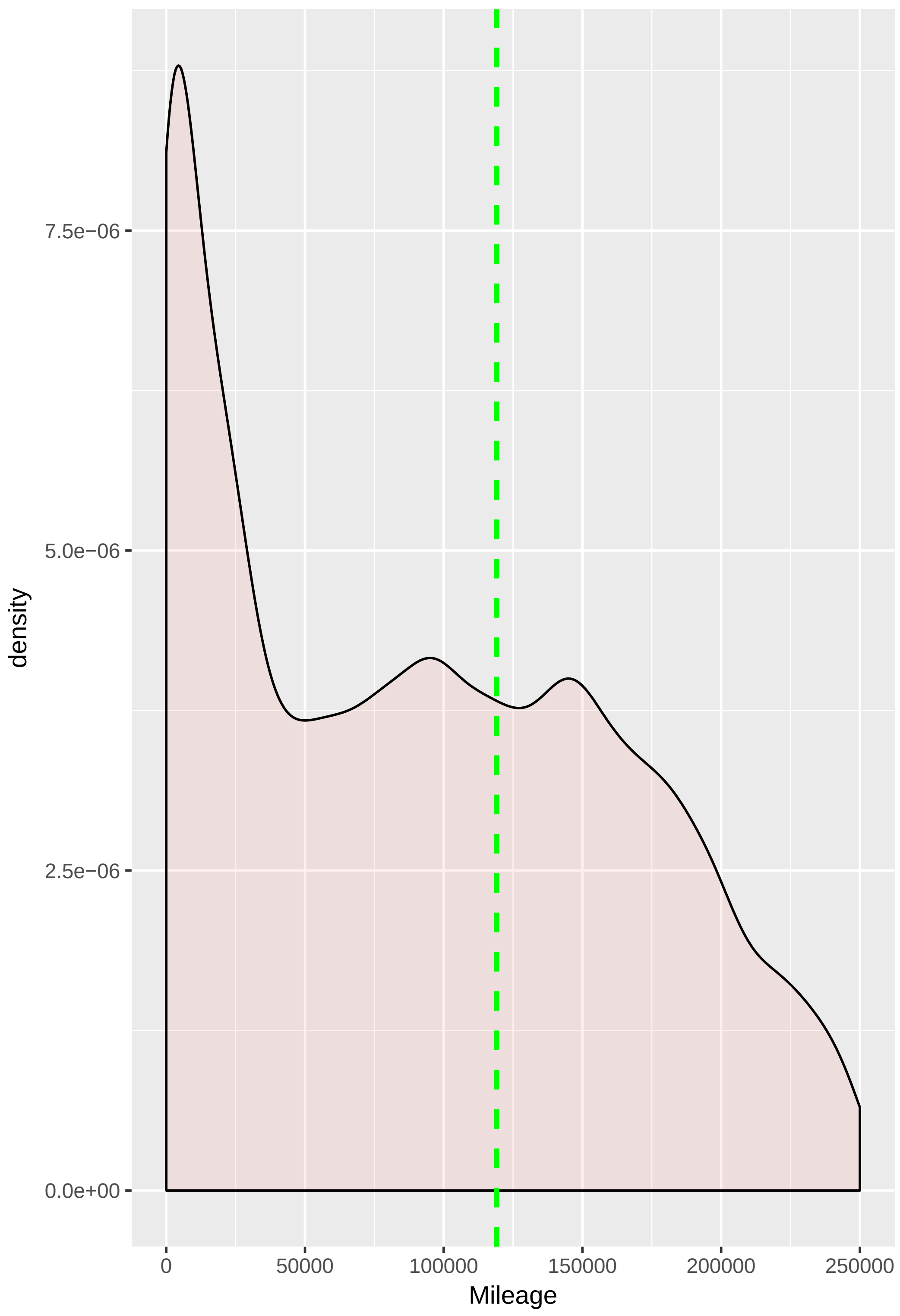


Figure 3: Mileage distribution

	Maker	Model	Year	Mileage	Age	DaysListed	Sold
1	skoda	citigo	2014	10349.00	2	75	FALSE
2	fiat	marea	2000	300017.00	16	75	FALSE
3	skoda	octavia	2003	145665.00	13	75	FALSE
4	skoda	citigo	2015	9800.00	1	75	FALSE
5	kia	sportage	2001	1.00	15	75	FALSE
6	skoda	superb	2002	234000.00	14	75	FALSE

Table 2: `Sample Car Ads Dataset - first rows`

```
# summary(cars.sample)
```

What is the most advertized vs sold car maker?

```
require(forcats)
```

```
#> Loading required package: forcats
```

```
total <- nrow(cars.sample)
ggplot(cars.sample, aes(fct_rev(fct_infreq(Maker)), fill=Sold)) +
       geom_bar() +
       labs(x="", y="Percent of Ads") +
        scale_y_continuous(labels = function(x) sprintf("%.0f%%",x/total*100)) +
       coord_flip()
```

What is the 20 best advertived vs sold car models?

```
require(forcats)
total <- nrow(cars.sample)
cars.sample$Car <- paste(cars.sample$Maker, cars.sample$Model)
betsCarsList <- fct_infreq(cars.sample$Car)
cars.sample.bestCars <- cars.sample[cars.sample$Car %in%  levels(betsCarsList)[1:20],]
ggplot(cars.sample.bestCars, aes(fct_rev(fct_infreq(Car)), fill=Sold)) +
       geom_bar() +
       labs(x="", y="Percent of Ads in the Sample Set") +
       scale_y_continuous(labels = function(x) sprintf("%.0f%%",x/total*100)) +
      coord_flip()
```

What is the best 20 advertised vs sold cars?

```
require(forcats)
total <- nrow(cars.sample)
cars.sample$Car1 <- paste(cars.sample$Maker, cars.sample$Model, cars.sample$Year)
betsCarsList <- fct_infreq(cars.sample$Car1)
cars.sample.bestCars <- cars.sample[cars.sample$Car1 %in%  levels(betsCarsList)[1:20],]
ggplot(cars.sample.bestCars, aes(fct_rev(fct_infreq(Car1)), fill=Sold)) +
       geom_bar() +
       labs(x="", y="Percent of Ads in the Sample Set") +
       scale_y_continuous(labels = function(x) sprintf("%.2f%%",x/total*100)) +
      coord_flip()
```

What is the distribution of car prices in the ads for the cars that were not sold?

```
ggplot(cars.sample[!(cars.sample$Sold),], aes(x=Price)) +
  geom_density(fill="#FF6666", alpha=.1) +
  scale_x_continuous(limits = c(0, 80000)) +
    geom_vline(aes(xintercept=mean(Price, na.rm=T)), color="red", linetype="dashed", size=1)
```

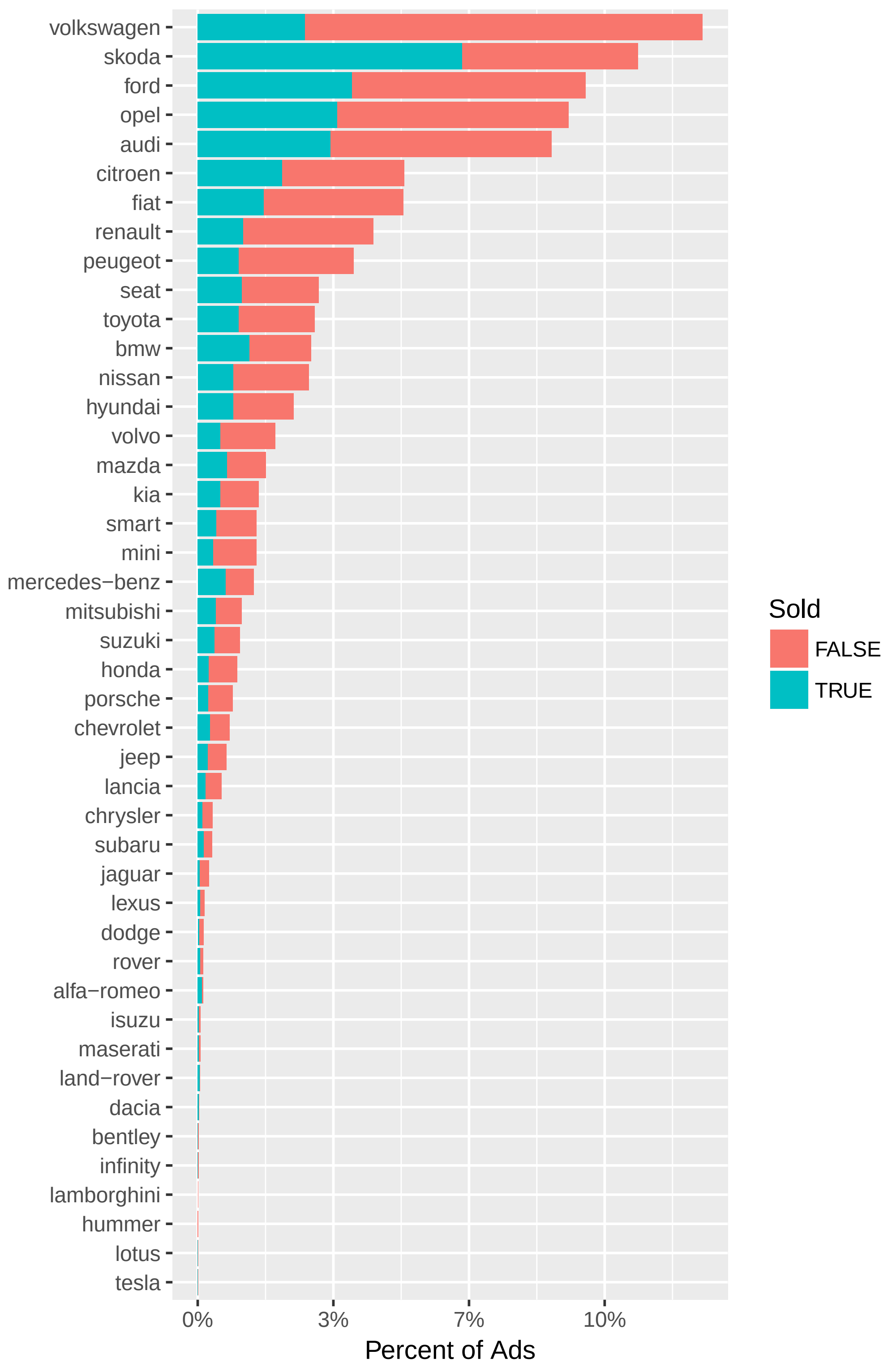


Figure 4: Number of Ads by Maker

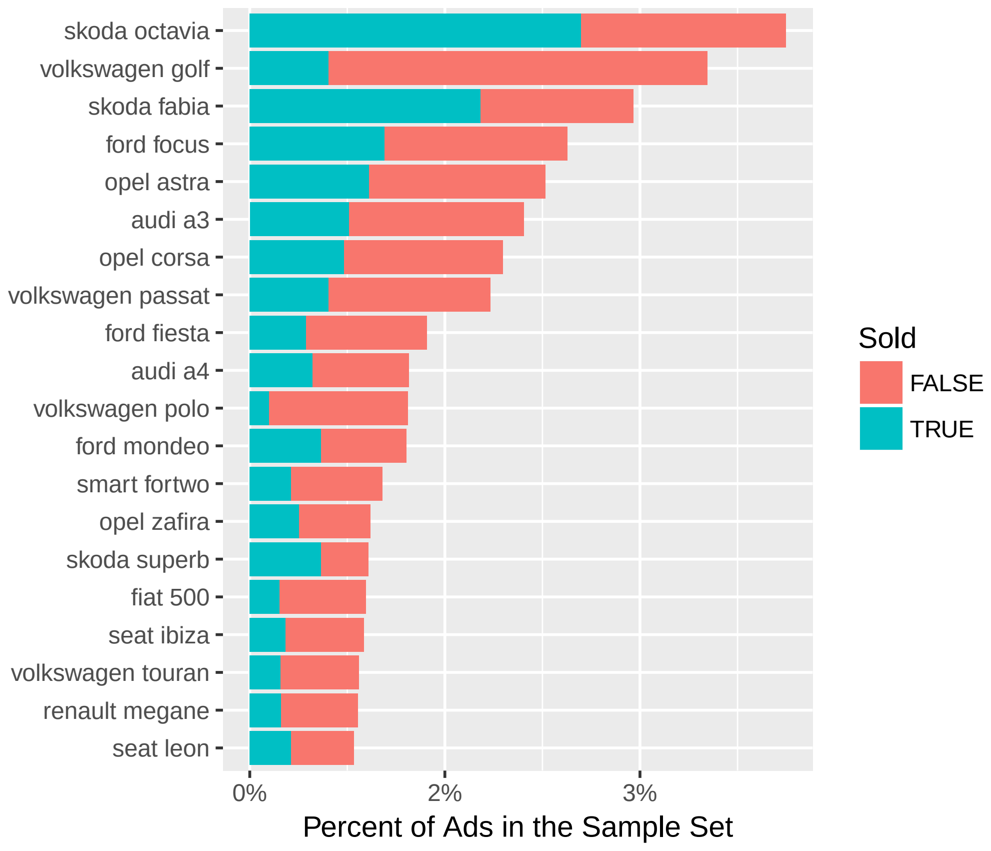


Figure 5: 40 Best Car Models

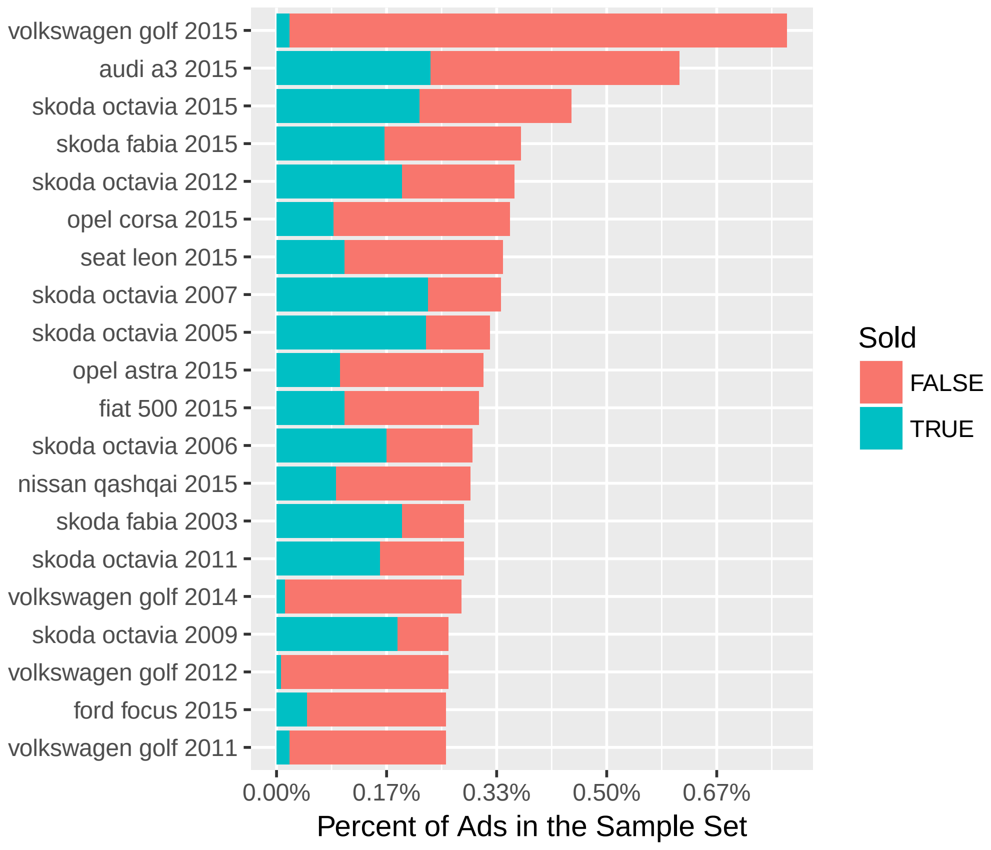
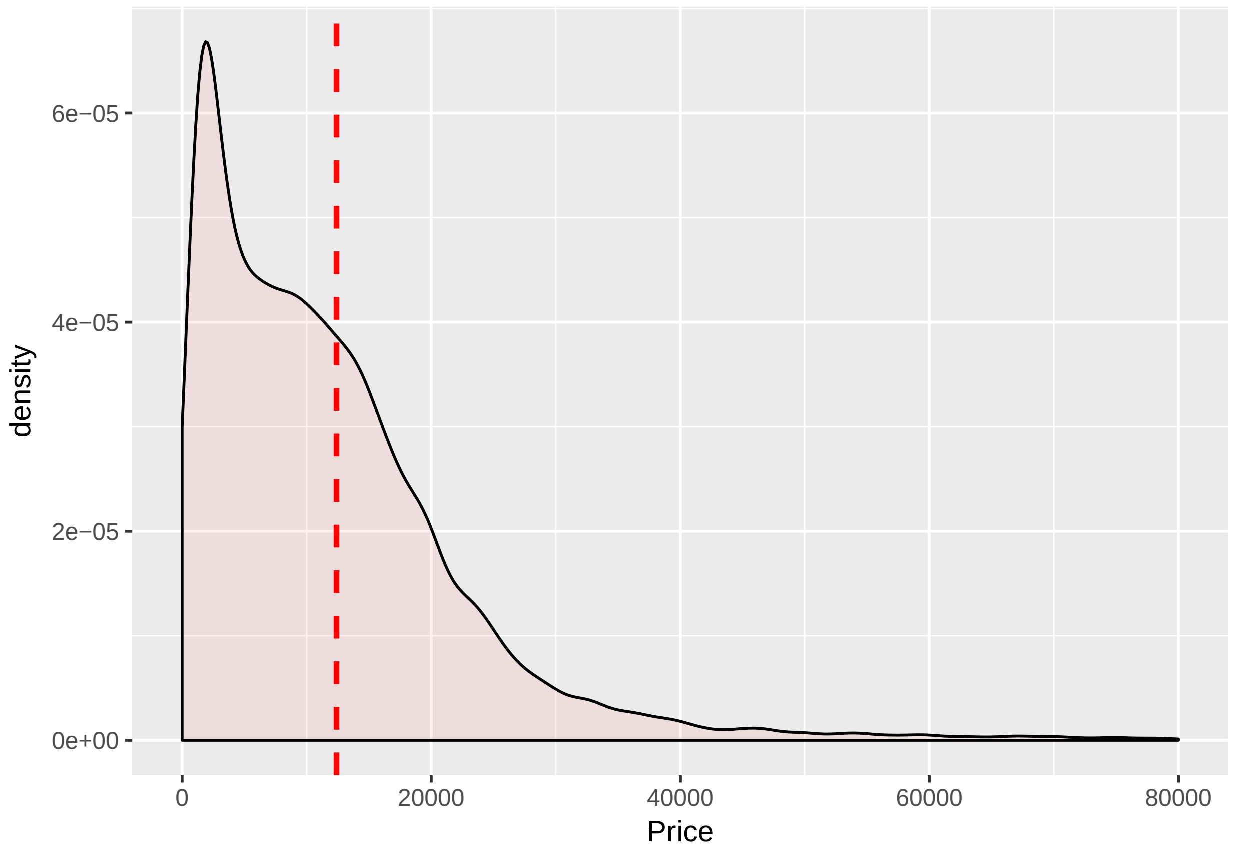
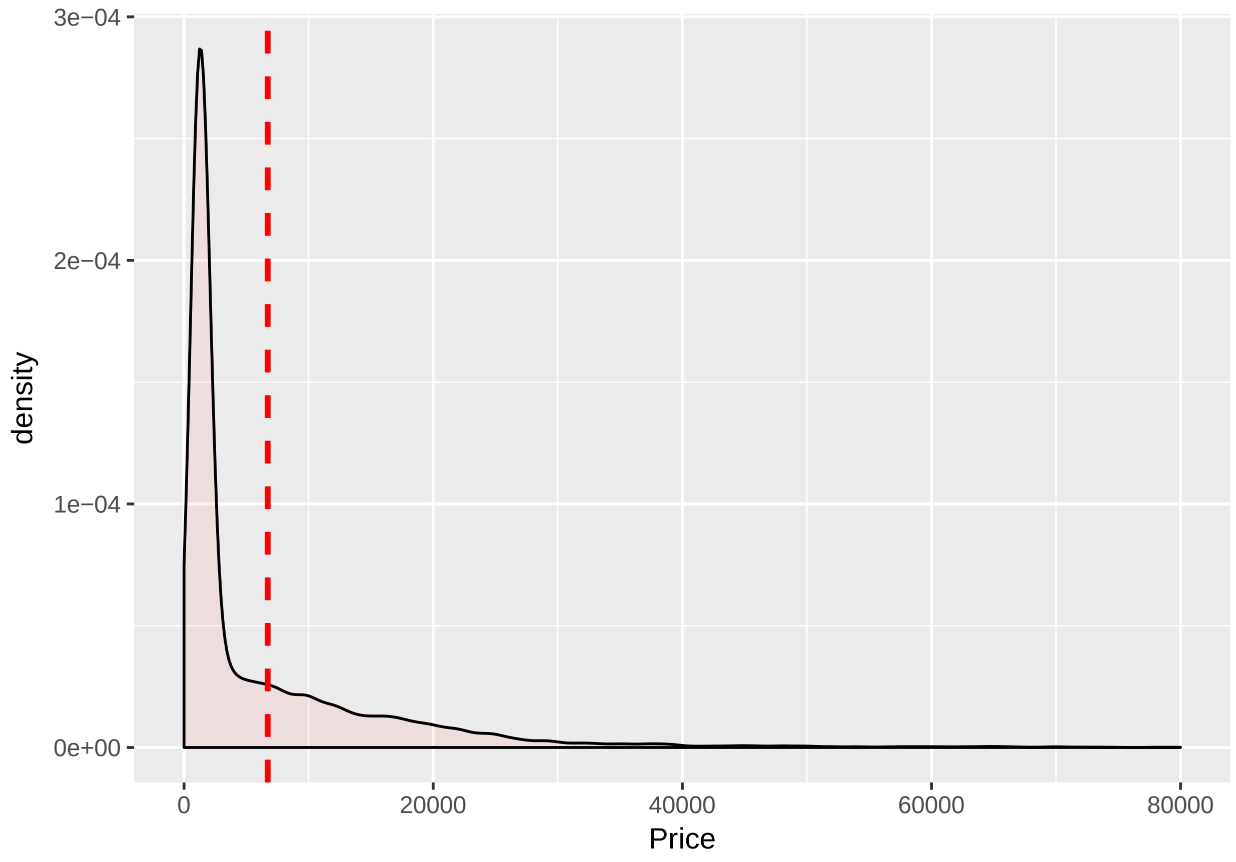


Figure 6: 20 Best Cars



What is the distribution of car prices of the cars that were sold?

```
ggplot(cars.sample[cars.sample$Sold,], aes(x=Price)) +
  geom_density(fill="#FF6666", alpha=.1) +
  scale_x_continuous(limits = c(0, 80000)) +
    geom_vline(aes(xintercept=mean(Price, na.rm=T)), color="red", linetype="dashed", size=1)
```



Disconnecting from the HIVE

It is very important to disconnect from the HIVE at the end of the session:

```
dbDisconnect(conn)
```

```
#> [1] TRUE
```

Conclusion

The project was a success.

Bibliography

D. B. Dahl. *xtable: Export Tables to LaTeX or HTML*, 2016. URL https://CRAN.R-project.org/package=xtable. R package version 1.8-2. [p4]

UCI Wine Data Set. Uci machine learning repository: wine quality data set. URL http://archive.ics.uci.edu/ml/datasets/Wine+Quality. [p1]

Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

Igor Baranov
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/profile.php?id=21219