Analysis on used car dataset from Kaggle

Problem Background:

A large investment firm that is contemplating to invest in a used car business and task is to provide data driven advice to the stakeholders, that will enable them to make a sound investment decision.

Data Collection:

The used car dataset was downloaded from https://www.kaggle.com/mirosval/personal-cars-classifieds.

Data Dictionary:

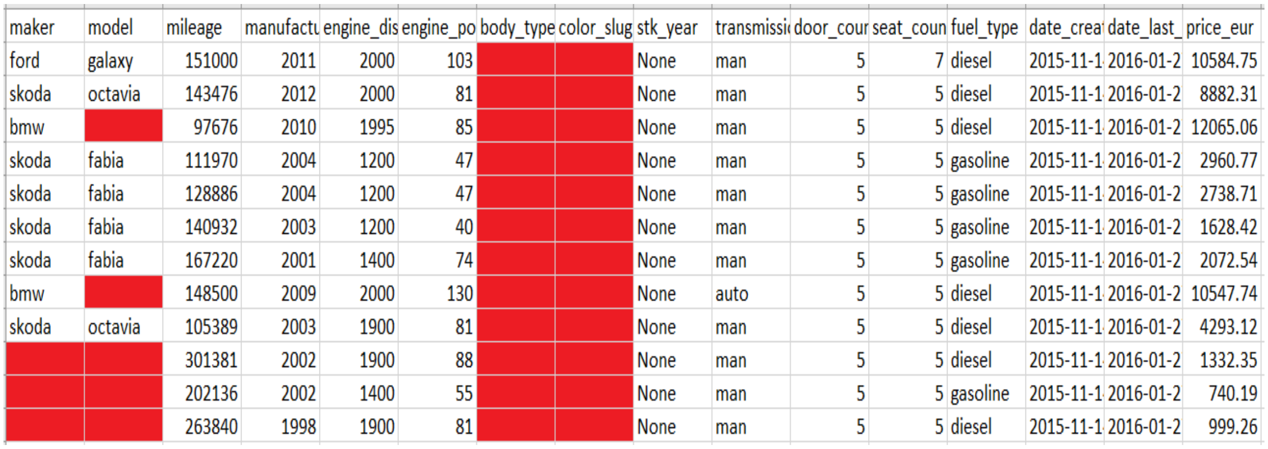
Attributes	Datatype
maker	STRING
model	STRING
mileage	FLOAT
manufacture_year	INTEGER
engine_displacement	FLOAT
engine_power	STRING
body_type	STRING
color_slug	STRING
stk_year	FLOAT
Transmission:	STRING
door_count	INTEGER
seat_count	INTEGER
fuel_type	STRING
date_created	DATE
date_last_seen	DATE
price_eur	FLOAT

Table:1 Data Dictionary of used car dataset

Features:

Initial records: 3552912 rows x 16 columns

Raw Data:



maker	model	mileage	manufactu	engine_dis	engine_po	body_type	color_slug	stk_year	transmissi	door_cour	seat_coun	fuel_type	date_creat	date_last_	price_eur
ford	galaxy	151000	2011	2000	103	🟥	🟥	None	man	5	7	diesel	2015-11-1	2016-01-2	10584.75
skoda	octavia	143476	2012	2000	81	🟥	🟥	None	man	5	5	diesel	2015-11-1	2016-01-2	8882.31
bmw	🟥	97676	2010	1995	85	🟥	🟥	None	man	5	5	diesel	2015-11-1	2016-01-2	12065.06
skoda	fabia	111970	2004	1200	47	🟥	🟥	None	man	5	5	gasoline	2015-11-1	2016-01-2	2960.77
skoda	fabia	128886	2004	1200	47	🟥	🟥	None	man	5	5	gasoline	2015-11-1	2016-01-2	2738.71
skoda	fabia	140932	2003	1200	40	🟥	🟥	None	man	5	5	gasoline	2015-11-1	2016-01-2	1628.42
skoda	fabia	167220	2001	1400	74	🟥	🟥	None	man	5	5	gasoline	2015-11-1	2016-01-2	2072.54
bmw	🟥	148500	2009	2000	130	🟥	🟥	None	auto	5	5	diesel	2015-11-1	2016-01-2	10547.74
skoda	octavia	105389	2003	1900	81	🟥	🟥	None	man	5	5	diesel	2015-11-1	2016-01-2	4293.12
🟥	🟥	301381	2002	1900	88	🟥	🟥	None	man	5	5	diesel	2015-11-1	2016-01-2	1332.35
🟥	🟥	202136	2002	1400	55	🟥	🟥	None	man	5	5	gasoline	2015-11-1	2016-01-2	740.19
🟥	🟥	263840	1998	1900	81	🟥	🟥	None	man	5	5	diesel	2015-11-1	2016-01-2	999.26

Table:2 Raw data of used car dataset

Data Loading in Hive:

Steps to load data into hive:

1. Downloading dataset from the following link: [https://www.kaggle.com/mirosval/personal-cars-classifieds using](https://www.kaggle.com/mirosval/personal-cars-classifieds)
2. Copy downloaded dataset to folder in HDFS
3. Creating database named cars_db
4. Using created database
5. Creating table in cars_db database
6. Load downloaded dataset using load command.

Data Cleaning:

1. Checking for the missing and null values.
2. Finding out incorrect entries
3. Finding out of range values

After performing data cleaning, the final dataset contains following columns:

1. maker: all the maker records excluding null values.
2. model: all the model records excluding null values.
3. mileage: records with value<=250000
   - dataset contains lots of values that were not feasible i.e maximum mileage value was 9999999.0. in addition, customers generally would not prefer cars that has been driven more than certain amount.

4. manufacture_year: considering manufacture_year > 2008
    - Considering factors such as warranty, extended warranty and individual maintenance, customers tend to buy cars no older than 10 years. In reference to that, cars manufactured later than year 2008 are considered for analysis.

5. engine_displacement: all the records between 500 and 8000.

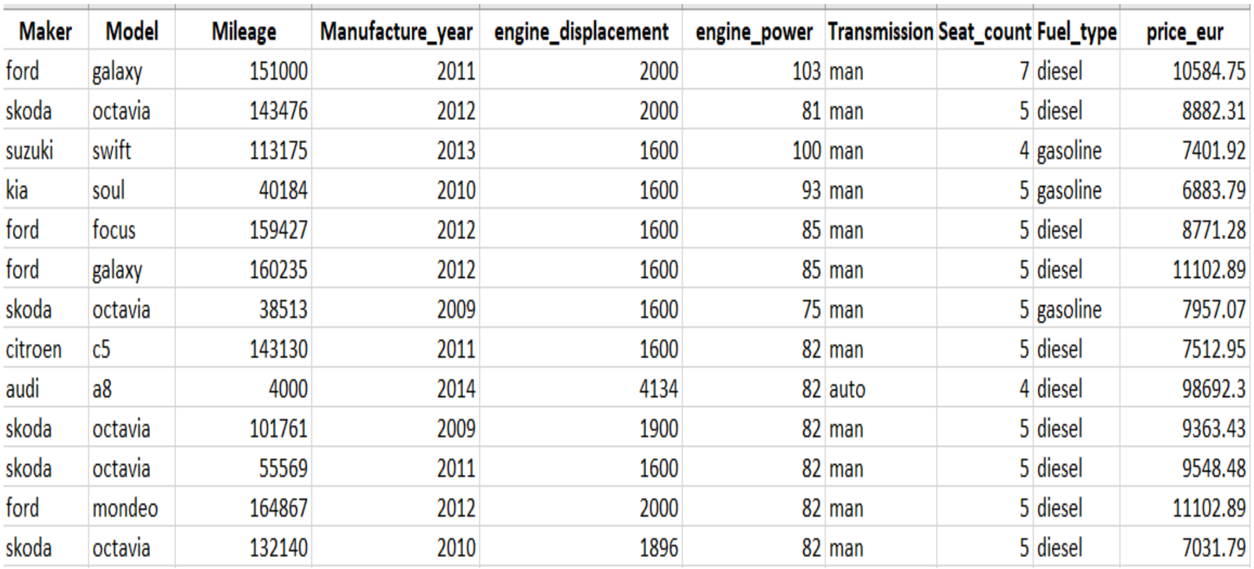
    - Researching on maximum engine_displacement of any car, the largest engine_displacement value found is 8.4. therefore, value between 500 and 8000 are considered. Dataset contains lots of infeasible values which are removed.

6. engine_power: all the records of engine_power.
7. transmission: all the records of transmission
8. seat_count: all the records of seat_count
9. fuel_type: all the records of fuel_type
10. price_eur: price in euro between 500 and 500000.

    - Dataset contains large no. of records having car price in millions which is not appropriate. One would seldom invest such large amount in used cars.

Records: 974335 rows x 10 columns.

Cleansed data:



Maker	Model	Mileage	Manufacture_year	engine_displacement	engine_power	Transmission	Seat_count	Fuel_type	price_eur
ford	galaxy	151000	2011	2000	103	man	7	diesel	10584.75
skoda	octavia	143476	2012	2000	81	man	5	diesel	8882.31
suzuki	swift	113175	2013	1600	100	man	4	gasoline	7401.92
kia	soul	40184	2010	1600	93	man	5	gasoline	6883.79
ford	focus	159427	2012	1600	85	man	5	diesel	8771.28
ford	galaxy	160235	2012	1600	85	man	5	diesel	11102.89
skoda	octavia	38513	2009	1600	75	man	5	gasoline	7957.07
citroen	c5	143130	2011	1600	82	man	5	diesel	7512.95
audi	a8	4000	2014	4134	82	auto	4	diesel	98692.3
skoda	octavia	101761	2009	1900	82	man	5	diesel	9363.43
skoda	octavia	55569	2011	1600	82	man	5	diesel	9548.48
ford	mondeo	164867	2012	2000	82	man	5	diesel	11102.89
skoda	octavia	132140	2010	1896	82	man	5	diesel	7031.79

Table 3: cleansed data of used car dataset

Data Analysis and Data Visualization:

Business Question:

- **Which car makers are dominant in market, hence best to invest on?**
  - **Total cars per maker**

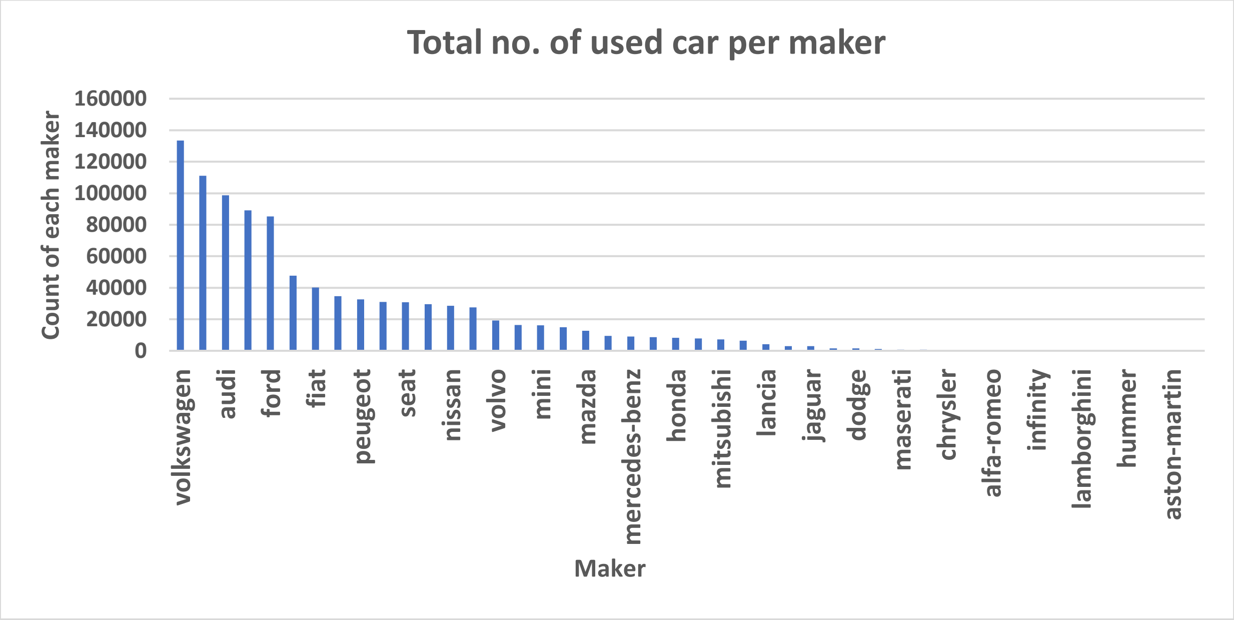


Figure 1: Bar chart of total no. of car per maker

The above analysis shows the no of cars listed for each maker. The chart indicates that certain car makers have very high count of cars, implying that investing on these car makers would be beneficial.

Considering top 15 makers for further analysis.

- **What is the significance of car pricing and how it helps to determine potential car makers for investment?**

  - **Average price per maker**

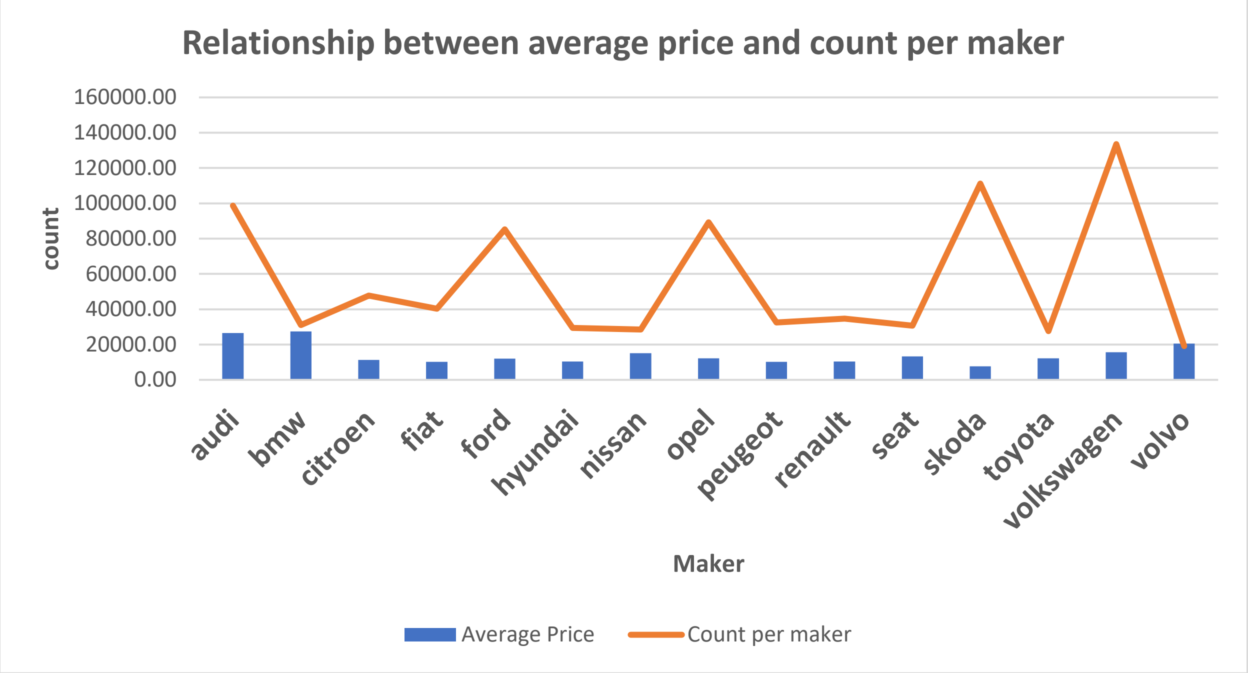


Figure 2: bar chart and line chart of relationship between average price and count per maker

Generally dividing average car price into 3 range: high, mid and low, it can be seen that Audi is the highest selling car in high range Category. Ford, Opel and Skoda have similar no of units, which falls under low range category. Volkswagen has highest no of units in all categories, which falls under mid-range category.

High range car makers to invest on: Audi

Mid-range car makers to invest on: Volkswagen

Low range car makers to invest on: Ford, Opel, Skoda

- **Which car models are best to invest on in dominant car makers?**
  - Finding top five car model from all five car makers from previous question.

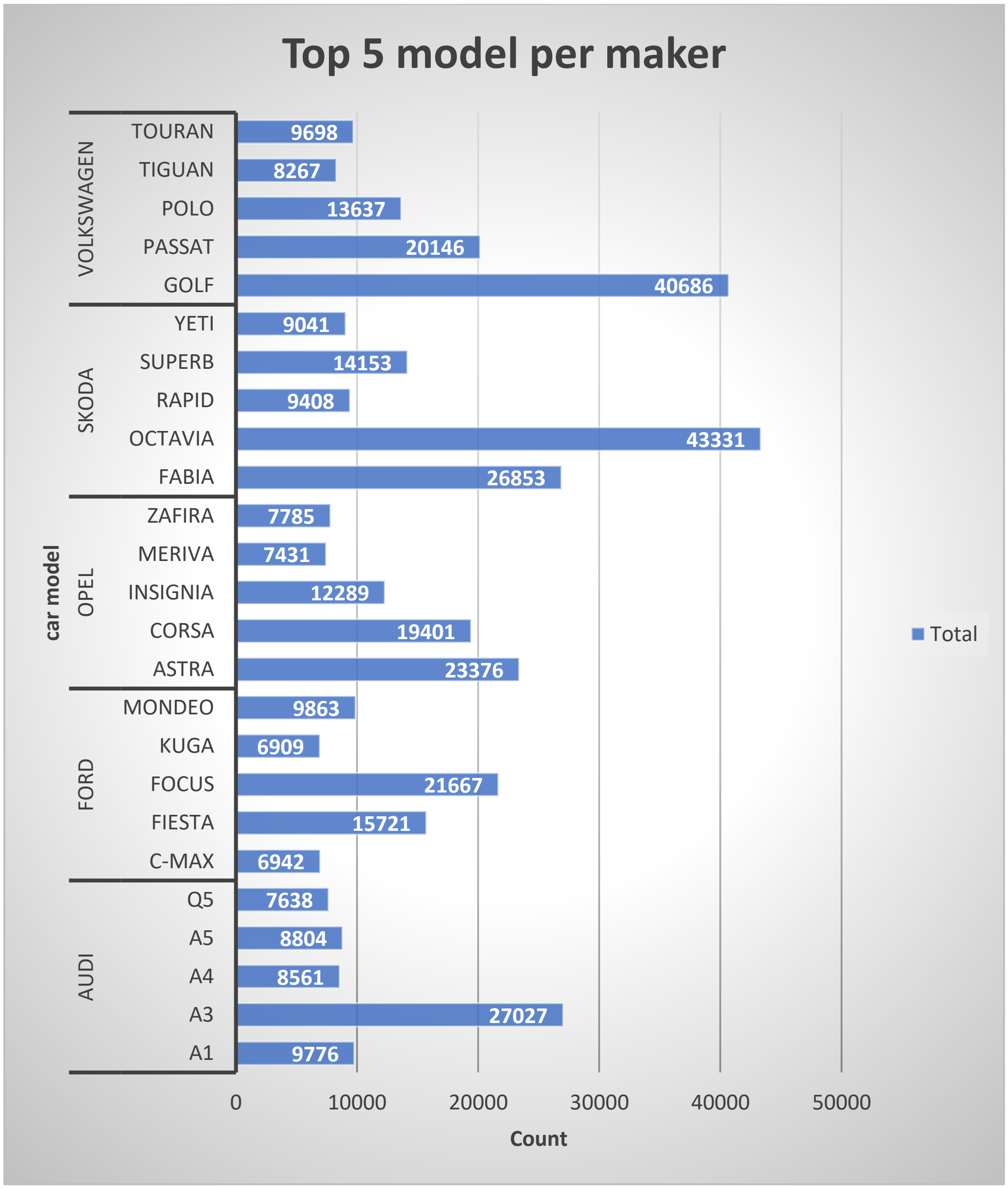


Figure 3: Clustered bar chart of top 5 model per maker

The above analysis shows the no of cars listed for each car model. The chart indicates that certain car models have very high count of cars, implying that investing on these car models would be beneficial.

- **Does price and mileage have any impact while selecting a car?**

- **Price v/s Mileage**

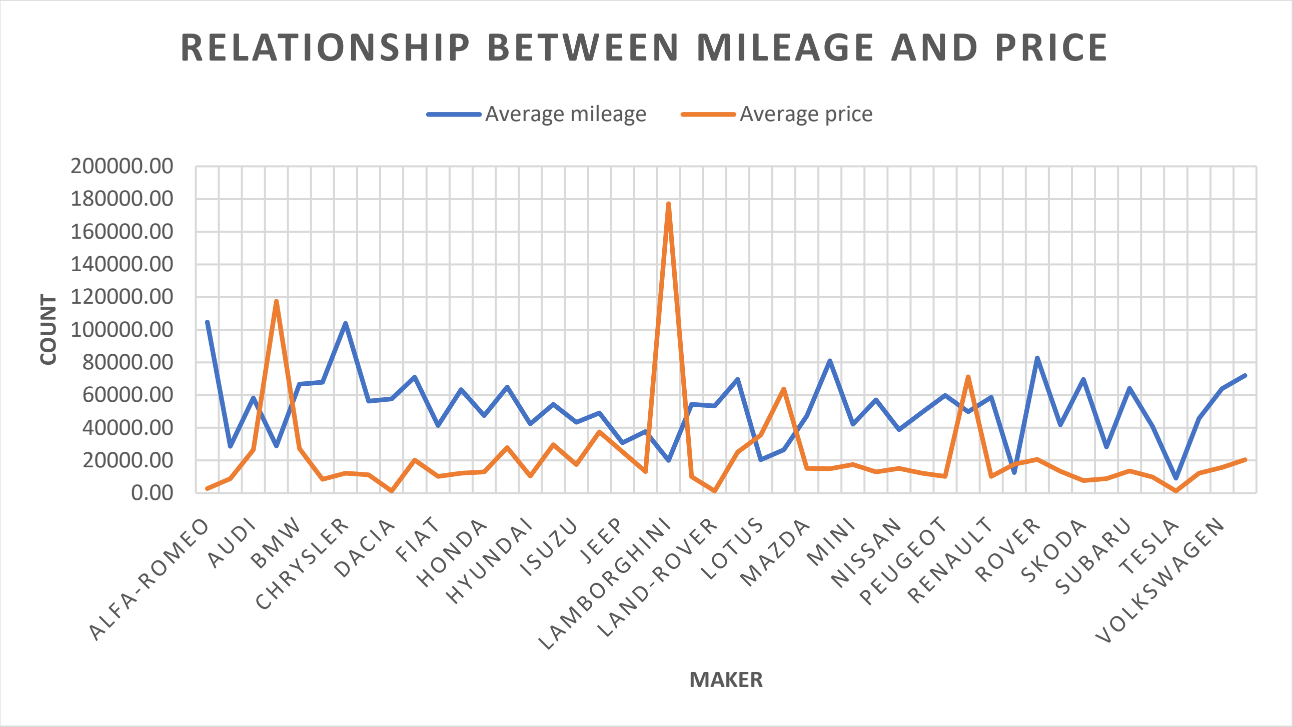


Figure 4: Line chart showing relationship between price and mileage for each maker

The above chart shows the relationship between average price and average mileage of all car makers and it clearly indicates that both the variables are inversely proportional. For example, the price of car is higher if it has less mileage and vice versa.

- **What is the most common engine displacement value found in modern car?**

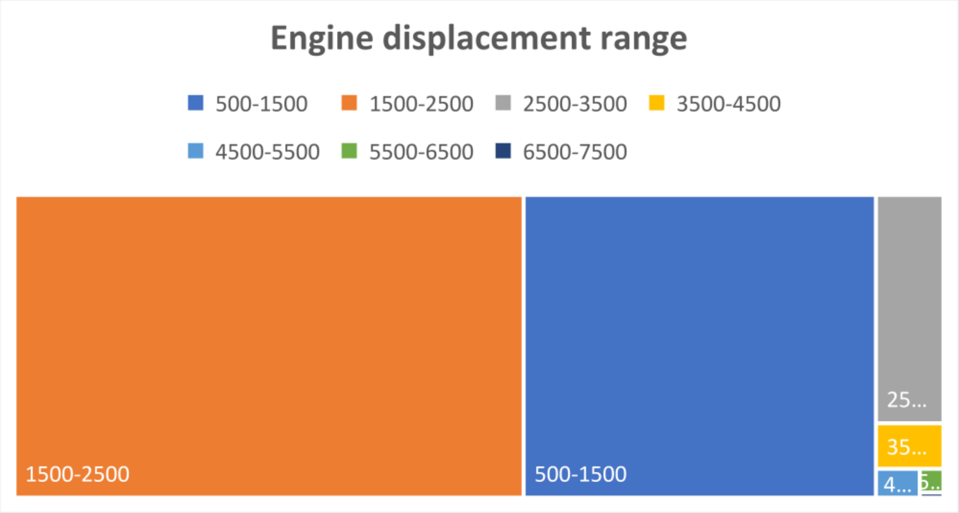


Figure 5: tree map showing maximum car available is between 1500-2500 engine displacement range

Dividing engine displacement value into various class range and counting total car available in each class shows that, maximum cars have engine_displacement value between 1500-2500. Most modern cars run on 2000 cc engine displacement.[1]

Therefore, investing on cars with engine_displacement range between 1500-2500 would be advisable.

Conclusion:

Based on the analyses, it can be concluded that investing on car models of dominant car makers would be beneficial for an organization. Furthermore, price, mileage and engine_displacement features play an important role while selecting any car model for investment.

Appendix

Commands and queries executed to extract data for further analysis

- **Steps to load data into hive:**

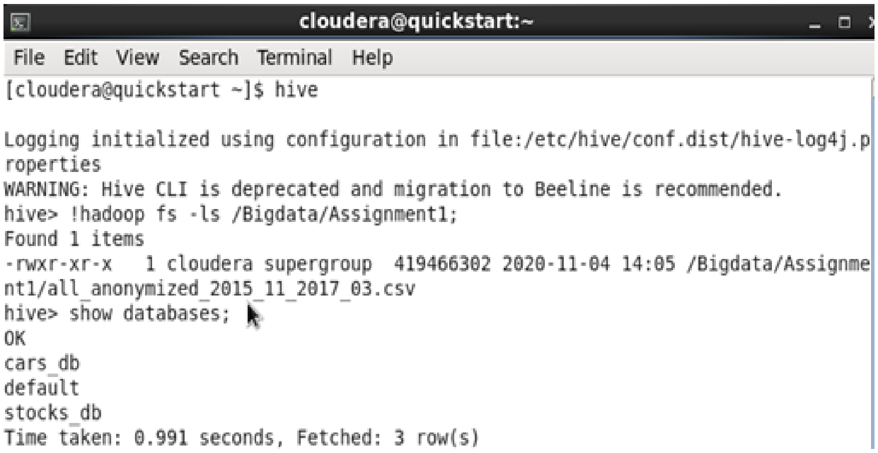
1. Downloading dataset from the following link: [https://www.kaggle.com/mirosval/personal-cars-classifieds using](https://www.kaggle.com/mirosval/personal-cars-classifieds)

2. Copy downloaded dataset to folder in HDFS

```
hadoop fa -copyFromLocal
/home/cloudera/Downloads/all_anonymized_2015_11_2017_03.csv
/Bigdata/Assignment1/all_anonymized_2015_11_2017_03.csv
```

3. Creating database named cars_db

```
hive> CREATE DATABASE cars_db;
```

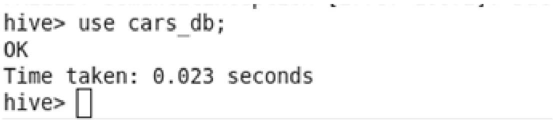


```
[cloudera@quickstart ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.p
roperties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> !hadoop fs -ls /Bigdata/Assignment1;
Found 1 items
-rwxr-xr-x   1 cloudera supergroup  419466302 2020-11-04 14:05 /Bigdata/Assignme
nt1/all_anonymized_2015_11_2017_03.csv
hive> show databases;
OK
cars_db
default
stocks_db
Time taken: 0.991 seconds, Fetched: 3 row(s)
```

Figure 1: result of show databases command

4. Using created database



```
hive> use cars_db;
OK
Time taken: 0.023 seconds
hive> 
```

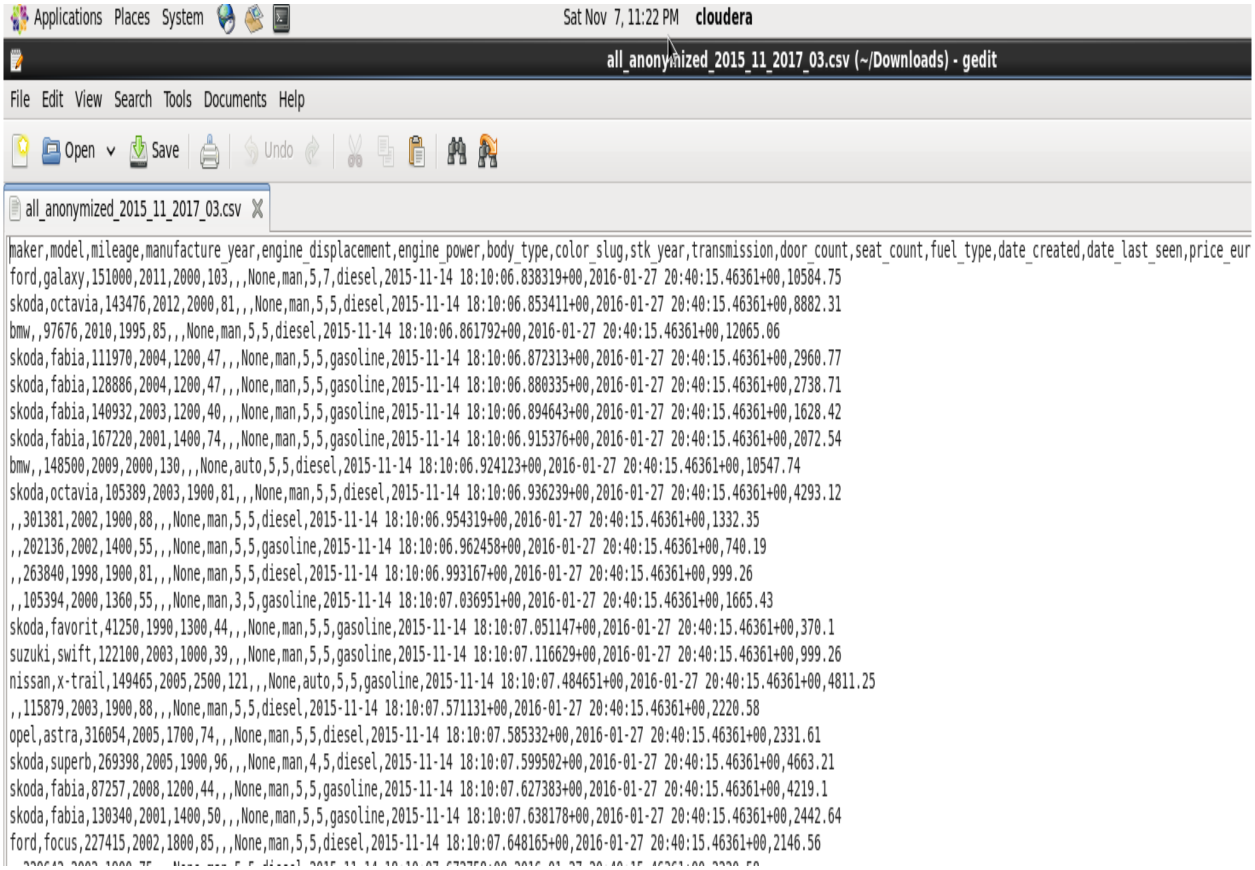
5. Creating table in cars_db database.

```
Hive> CREATE EXTERNAL TABLE IF NOT EXISTS cars (
maker STRING,
model STRING,
mileage FLOAT,
manufacture_year INT,
engine_displacement FLOAT,
engine_power STRING,
body_type STRING,
color_slug STRING,
stk_year FLOAT,
transmission STRING,
door_count INT,
seat_count INT,
fuel_type STRING,
date_created DATE,
date_last_seen DATE,
price_eur FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/Bigdata/Assignment1/'
TBLPROPERTIES ("skip.header.line.count"="1",
'creator'='Karishma', 'created_on'='11/04/2020',
'description'='dataset for classfied Ads of cars in Germany
and Czech Republic');
```

6. Load dataset in hive using load command.

```
LOAD DATA INPATH
'/Bigdata/Assignment1/all_anonymized_2015_11_2017_03' INTO
TABLE cars;
```

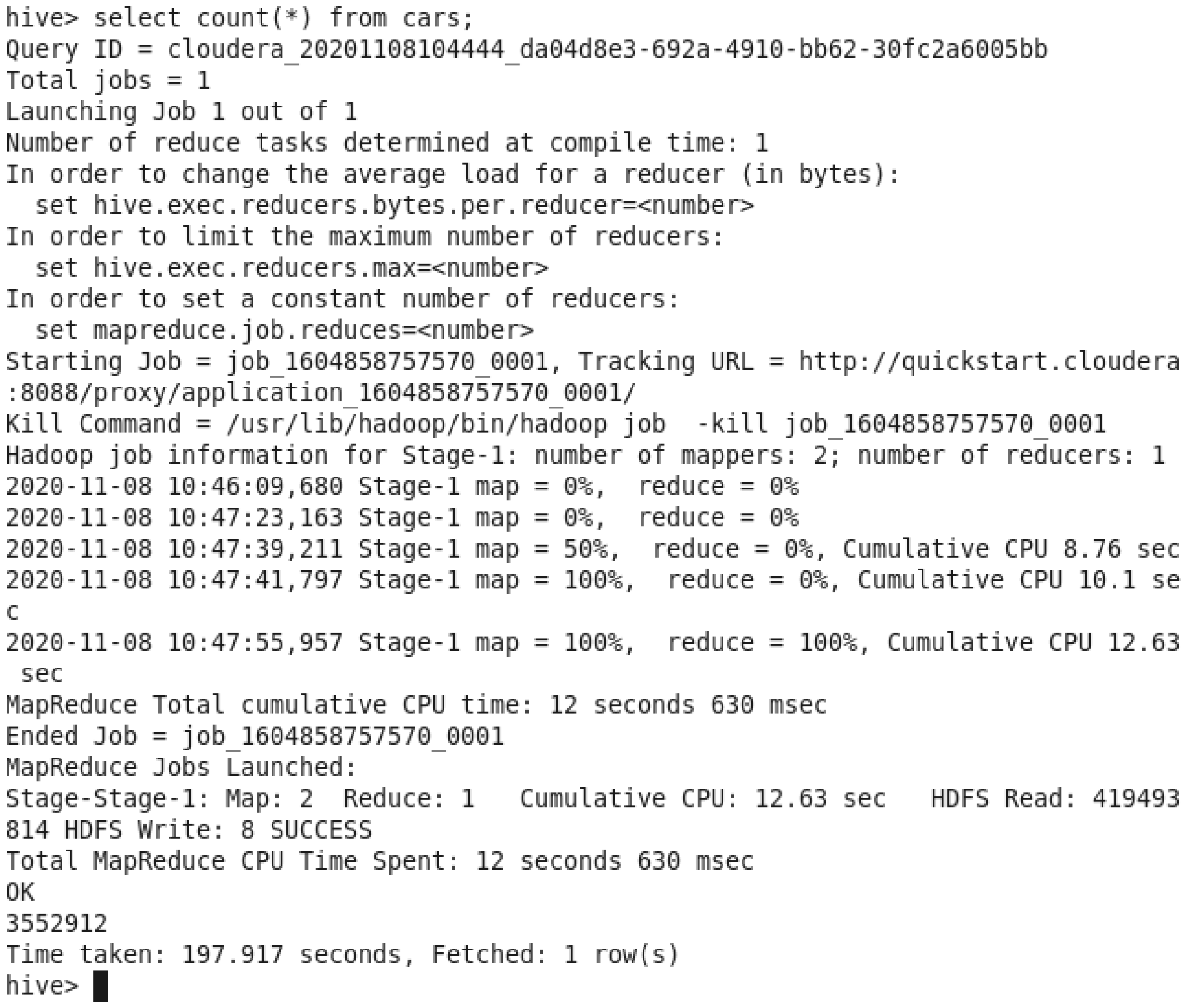
Raw data:


Applications Places System
Sat Nov 7, 11:22 PM cloudera
all_anonymized_2015_11_2017_03.csv (~/Downloads) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
all_anonymized_2015_11_2017_03.csv

```
maker,model,mileage,manufacture_year,engine_displacement,engine_power,body_type,color_slug,stk_year,transmission,door_count,seat_count,fuel_type,date_created,date_last_seen,price_eur
ford,galaxy,151000,2011,2000,103,,,None,man,5,7,diesel,2015-11-14 18:10:06.838319+00,2016-01-27 20:40:15.46361+00,10584.75
skoda,octavia,143476,2012,2000,81,,,None,man,5,5,diesel,2015-11-14 18:10:06.853411+00,2016-01-27 20:40:15.46361+00,8882.31
bmw,,97676,2010,1995,85,,,None,man,5,5,diesel,2015-11-14 18:10:06.861792+00,2016-01-27 20:40:15.46361+00,12065.06
skoda,fabia,111970,2004,1200,47,,,None,man,5,5,gasoline,2015-11-14 18:10:06.872313+00,2016-01-27 20:40:15.46361+00,2960.77
skoda,fabia,128886,2004,1200,47,,,None,man,5,5,gasoline,2015-11-14 18:10:06.880335+00,2016-01-27 20:40:15.46361+00,2738.71
skoda,fabia,140932,2003,1200,40,,,None,man,5,5,gasoline,2015-11-14 18:10:06.894643+00,2016-01-27 20:40:15.46361+00,1628.42
skoda,fabia,167220,2001,1400,74,,,None,man,5,5,gasoline,2015-11-14 18:10:06.915376+00,2016-01-27 20:40:15.46361+00,2072.54
bmw,,148500,2009,2000,130,,,None,auto,5,5,diesel,2015-11-14 18:10:06.924123+00,2016-01-27 20:40:15.46361+00,10547.74
skoda,octavia,105389,2003,1900,81,,,None,man,5,5,diesel,2015-11-14 18:10:06.936239+00,2016-01-27 20:40:15.46361+00,4293.12
,,301381,2002,1900,88,,,None,man,5,5,diesel,2015-11-14 18:10:06.954319+00,2016-01-27 20:40:15.46361+00,1332.35
,,202136,2002,1400,55,,,None,man,5,5,gasoline,2015-11-14 18:10:06.962458+00,2016-01-27 20:40:15.46361+00,740.19
,,263840,1998,1900,81,,,None,man,5,5,diesel,2015-11-14 18:10:06.993167+00,2016-01-27 20:40:15.46361+00,999.26
,,105394,2000,1360,55,,,None,man,3,5,gasoline,2015-11-14 18:10:07.036951+00,2016-01-27 20:40:15.46361+00,1665.43
skoda,favorit,41250,1990,1300,44,,,None,man,5,5,gasoline,2015-11-14 18:10:07.051147+00,2016-01-27 20:40:15.46361+00,370.1
suzuki,swift,122100,2003,1000,39,,,None,man,5,5,gasoline,2015-11-14 18:10:07.116629+00,2016-01-27 20:40:15.46361+00,999.26
nissan,x-trail,149465,2005,2500,121,,,None,auto,5,5,gasoline,2015-11-14 18:10:07.484651+00,2016-01-27 20:40:15.46361+00,4811.25
,,115879,2003,1900,88,,,None,man,5,5,diesel,2015-11-14 18:10:07.571131+00,2016-01-27 20:40:15.46361+00,2220.58
opel,astra,316054,2005,1700,74,,,None,man,5,5,diesel,2015-11-14 18:10:07.585332+00,2016-01-27 20:40:15.46361+00,2331.61
skoda,superb,269398,2005,1900,96,,,None,man,4,5,diesel,2015-11-14 18:10:07.599502+00,2016-01-27 20:40:15.46361+00,4663.21
skoda,fabia,87257,2008,1200,44,,,None,man,5,5,gasoline,2015-11-14 18:10:07.627383+00,2016-01-27 20:40:15.46361+00,4219.1
skoda,fabia,130340,2001,1400,50,,,None,man,5,5,gasoline,2015-11-14 18:10:07.638178+00,2016-01-27 20:40:15.46361+00,2442.64
ford,focus,227415,2002,1800,85,,,None,man,5,5,diesel,2015-11-14 18:10:07.648165+00,2016-01-27 20:40:15.46361+00,2146.56
```

Figure 2: raw data of used car dataset

- **Queries performed to clean dataset:**



```
hive> select count(*) from cars;
Query ID = cloudera_20201108104444_da04d8e3-692a-4910-bb62-30fc2a6005bb
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1604858757570_0001, Tracking URL = http://quickstart.cloudera
:8088/proxy/application_1604858757570_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1604858757570_0001
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2020-11-08 10:46:09,680 Stage-1 map = 0%,  reduce = 0%
2020-11-08 10:47:23,163 Stage-1 map = 0%,  reduce = 0%
2020-11-08 10:47:39,211 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 8.76 sec
2020-11-08 10:47:41,797 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 10.1 se
c
2020-11-08 10:47:55,957 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 12.63
 sec
MapReduce Total cumulative CPU time: 12 seconds 630 msec
Ended Job = job_1604858757570_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 1   Cumulative CPU: 12.63 sec   HDFS Read: 419493
814 HDFS Write: 8 SUCCESS
Total MapReduce CPU Time Spent: 12 seconds 630 msec
OK
3552912
Time taken: 197.917 seconds, Fetched: 1 row(s)
hive> 
```

Figure 3: total no. of record available before performing data cleaning.

Above image shows the initial count of record loaded in cars table in cars_db database.

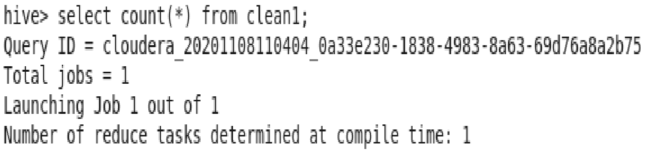
Initial records:3552912 records x 16 columns

1. Removing null values from maker variable.

```
hive> Create table if not exists clean1 as select * from
clean where maker! = ‘ ‘;
```

2. **Removing null values from model variable.**

```
hive>Create table if not exists clean1 as select * from clean
where model! = ‘ ‘;
```



```
hive> select count(*) from clean1;
Query ID = cloudera_20201108110404_0a33e230-1838-4983-8a63-69d76a8a2b75
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
```

Output: 2419551 records

3. **Finding minimum and maximum values of mileage**

```
hive>Select min(mileage) as min_mileage, max(mileage) as max-
mileage from clean1;
```

Output: min_mileage = 0.0
max_mileage = 9999999.0

4. **Considering records with mileage <=300000 and manufacture year > 1990**

```
hive>Create table if not exists clean2 as select * from clean1
where mileage <=300000 and manufacture_year>1990;
```

```
hive>select count (*) from clean2;
```

Output: 1914890 records

5. **Considering records with mileage <=300000 and manufacture year > 2005**

```
hive>Create table if not exists clean3 as select * from clean2
where mileage <=250000 and manufacture_year>2005;
```

```
hive>select count (*) from clean3;
```

Output:1436459 records

6. **Removing body_type and color_slug from the table and creating new table.**

```
hive>Create table if not exists clean4 as select maker, model,
mileage, manufacture_year, engine_displacement, engine_power,
stk_year, transmission, door_count, seat_count, fuel_type,
date_last_created, date_last_seen, price_eur from clean3;
```

7. **Considering records with manufacture year > 2008.**

```
hive>Create table if not exists clean5 as select * from clean4
where manufacture_year>2008;
```

```
hive>select count (*) from clean5;
```

Output: 1134786 records

8. **Removing stk_year from the table.**

```
hive>Create table if not exists clean6 as select maker, model,
mileage, manufacture_year, engine_displacement, engine_power,
transmission, door_count, seat_count, fuel_type,
date_last_created, date_last_seen, price_eur from clean5;
```

9. **Considering engine_displacement between 500 and 8000.**

```
hive>Create table if not exists clean7 as select * from clean6
where engine_displacement between 500 and 8000;
```

```
hive>select coun(*) from clean7;
```

Output: 974532 records

10. **Considering price_eur between 500 and 500000.**

```
hive>Create table if not exists clean8 as select * from clean7
where price_eur between 500 and 500000;
```

```
hive>select coun(*) from clean8;
```

Output: 974345 records

- **Queries executed to perform data analysis.**

1. **Which car makers are dominant in market, hence best to invest on?**

   Finding count of each makers.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY
'/home/cloudera/result' row format delimited fields
terminated by ‘,’ select maker, count(maker) as
count_of_maker from clean8 group by maker;
```

Output:

Maker	Count of maker
volkswagen	133527
skoda	111133
audi	98658
opel	89144
ford	85270
citroen	47716
fiat	40217
renault	34665
peugeot	32530
bmw	31070
seat	30733
hyundai	29498
nissan	28605
toyota	27586
volvo	19116
kia	16270
mini	16233
smart	14878
mazda	12733
porsche	9370
mercedes-benz	9072
suzuki	8638
honda	8282
chevrolet	7739
mitsubishi	7232
jeep	6490
lancia	4255
subaru	2950
jaguar	2889
lexus	1524
dodge	1465
dacia	1017
maserati	715
land-rover	689
chrysler	491
rover	465
alfa-romeo	378
bentley	293
infinity	276
isuzu	248
lamborghini	148
lotus	53
hummer	38
rolls-royce	25
aston-martin	10
tesla	1

Table 1: output of query (1)

2. **What is the significance of car pricing and how it helps to determine potential car makers for investment?**

   Calculating average price of top 15 car maker and finding relationship between car makers and price.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY
'/home/cloudera/result' row format delimited fields
terminated by ‘,’ select maker,avg(price_eur) as
average_price, from clean8 group by maker;
```

Output:

Maker	Average price
audi	26440.46
bmw	27363.81
citroen	11210.33
fiat	10265.20
ford	12086.47
hyundai	10463.27
nissan	15088.24
opel	12147.40
peugeot	10298.04
renault	10306.99
seat	13336.92
skoda	7736.10
toyota	12187.85
volkswagen	15725.82
volvo	20488.43

Table 2: output of query (2)

3. **Which car models are best to invest on in dominant car makers?**

   Finding top 5 car model for all dominant makers.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY
'/home/cloudera/result' row format delimited fields
terminated by ‘,’ select maker, model, count(model) as
count_of_model from clean8 group by maker, model;
```

Output:

Maker Model	count
audi	**61806**
a1	9776
a3	27027
a4	8561
a5	8804
q5	7638
ford	**61102**
c-max	6942
fiesta	15721
focus	21667
kuga	6909
mondeo	9863
opel	**70282**
astra	23376
corsa	19401
insignia	12289
meriva	7431
zafira	7785
skoda	**102786**
fabia	26853
octavia	43331
rapid	9408
superb	14153
yeti	9041
volkswagen	**92434**
golf	40686
passat	20146
polo	13637
tiguan	8267
touran	9698

Table 3: output of query (3)

4. **Finding average price and average mileage of each maker using avg() aggregate function and group by.**

```
hive> INSERT OVERWRITE LOCAL DIRECTORY
'/home/cloudera/result' row format delimited fields
terminated by ‘,’ select maker, avg(price_eur) as
average_price, avg(mileage) as average_mileage from
clean8 group by maker;
```

Output:

maker	Average mileage	Average price
alfa-romeo	104668.72	2807.52
aston-martin	28676.00	8863.81
audi	58175.09	26440.46
bentley	28813.53	117508.88
bmw	66653.00	27363.81
chevrolet	67801.80	8431.97
chrysler	103847.40	12217.59
citroen	56190.01	11210.33
dacia	57608.72	1295.34
dodge	70924.87	20196.25
fiat	41413.45	10265.20
ford	63408.71	12086.47
honda	47424.47	13060.89
hummer	64821.58	27882.88
hyundai	42413.67	10463.27
infinity	54258.41	29654.11
isuzu	43419.68	17500.85
jaguar	48969.70	37449.69
jeep	30758.82	25270.67
kia	37651.50	13122.72
lamborghini	20042.30	177141.17
lancia	54298.73	10076.25
land-rover	53379.37	1295.34
lexus	69624.64	25086.74
lotus	20417.34	35521.03
maserati	26569.99	63741.51
mazda	47239.25	15110.30
mercedes-benz	81011.11	14928.45
mini	42144.43	17437.44
mitsubishi	56992.20	13063.84
nissan	38832.56	15088.24
opel	49444.63	12147.40
peugeot	59870.49	10298.04
porsche	49908.63	71206.29
renault	58723.54	10306.99
rolls-royce	12610.00	17649.15
rover	82808.88	20684.21
seat	41799.03	13336.92
skoda	69614.02	7736.10
smart	28272.30	8780.43
subaru	64093.01	13542.42
suzuki	40706.87	9845.73
tesla	9100.00	1295.34
toyota	45673.52	12187.85
volkswagen	63920.55	15725.82
volvo	71873.62	20488.43

Table 4: output of query (4)

5. What should be the range of engine_displacement?

```
hive > select count (*) from clean8 where engine_displacement
between 500-1500;

hive > select count (*) from clean8 where engine_displacement
between 1500-2500;

hive > select count (*) from clean8 where engine_displacement
between 2500-3500;

hive > select count (*) from clean8 where engine_displacement
between 3500-4500;

hive > select count (*) from clean8 where engine_displacement
between 4500-5500;

hive > select count (*) from clean8 where engine_displacement
between 5500-6500;

hive > select count (*) from clean8 where engine_displacement
between 6500-7500;
```

Output:

engine-displacement	count
500-1500	370700
1500-2500	535856
2500-3500	53554
3500-4500	10677
4500-5500	4415
5500-6500	1963
6500-7500	427

Table 5: output of query (5)

Reference:

[1] https://www.carpart.com.au/blog/educational/what-is-engine-displacement-and-why-does-it-matter#:~:text=Engine%20displacement%20is%20expressed%20in,one%2Dhalf%20litre%20or%20500cc.