Big Data Project – UE18CS322

Yelp Restaurant Review Analysis

TEAM-16

Ajitesh Nair– PES2201800681

Skandan K A– PES2201800064

Ishan Padhy – PES2201800158

Project done under the guidance of Prof. Sudaroli Vijayakumar

OBJECTIVE :

The objective of this project is to perform analysis and gain useful insights from the Yelp Restaurant Review Dataset.We try to analyse the data and plot various graphs to gain some valuable insights.We also use Natural language processing to process the reviews given by users.We make use of technologies such as hadoop hdfs,Hive and Spark to do the analysis and processing.We also try to find the main factors based on which people give reviews.

DATASET:

Source: Yelp's Dataset | Kaggle

Our dataset consists of two files:

- Yelp_business.csv (30.29 MB)
  It consists of the following columns:

```
|-- Business_id
|-- name
|-- neighborhood
|-- address
|-- city
|-- state
|-- postal_code
|-- latitude
|-- longitude
|-- stars
|-- review_count
|-- is_open
|-- categories
```

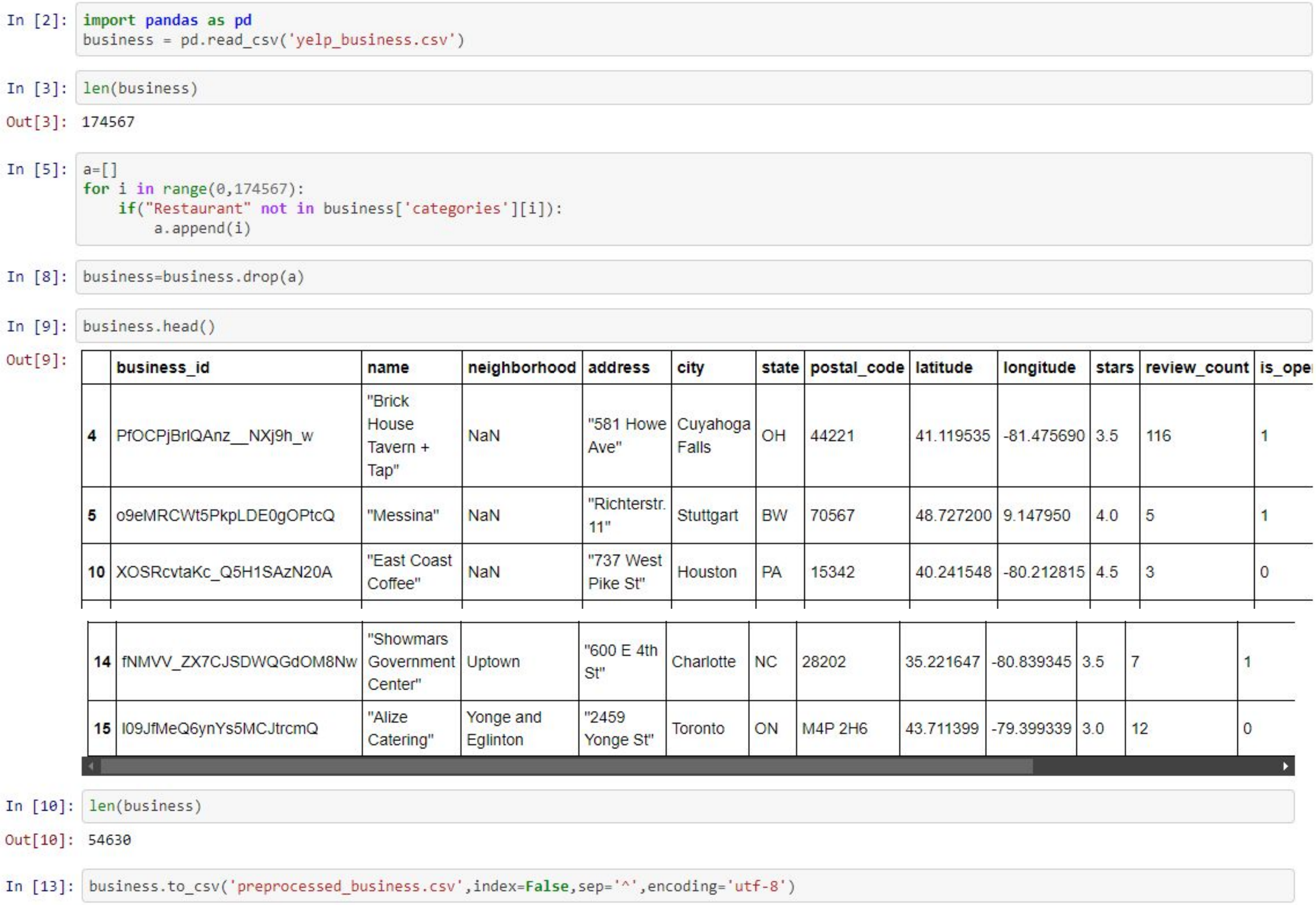
- Yelp_review.csv (3.53 GB)

  It consists of the following columns:

```
|-- review_id
|-- user_id
|-- business_id
|-- stars
|-- date
|-- text
|-- useful
|-- funny
|-- cool
```

PROJECT CODE AND EXECUTION:

We first do some preprocessing using Jupyter Notebook(Python)

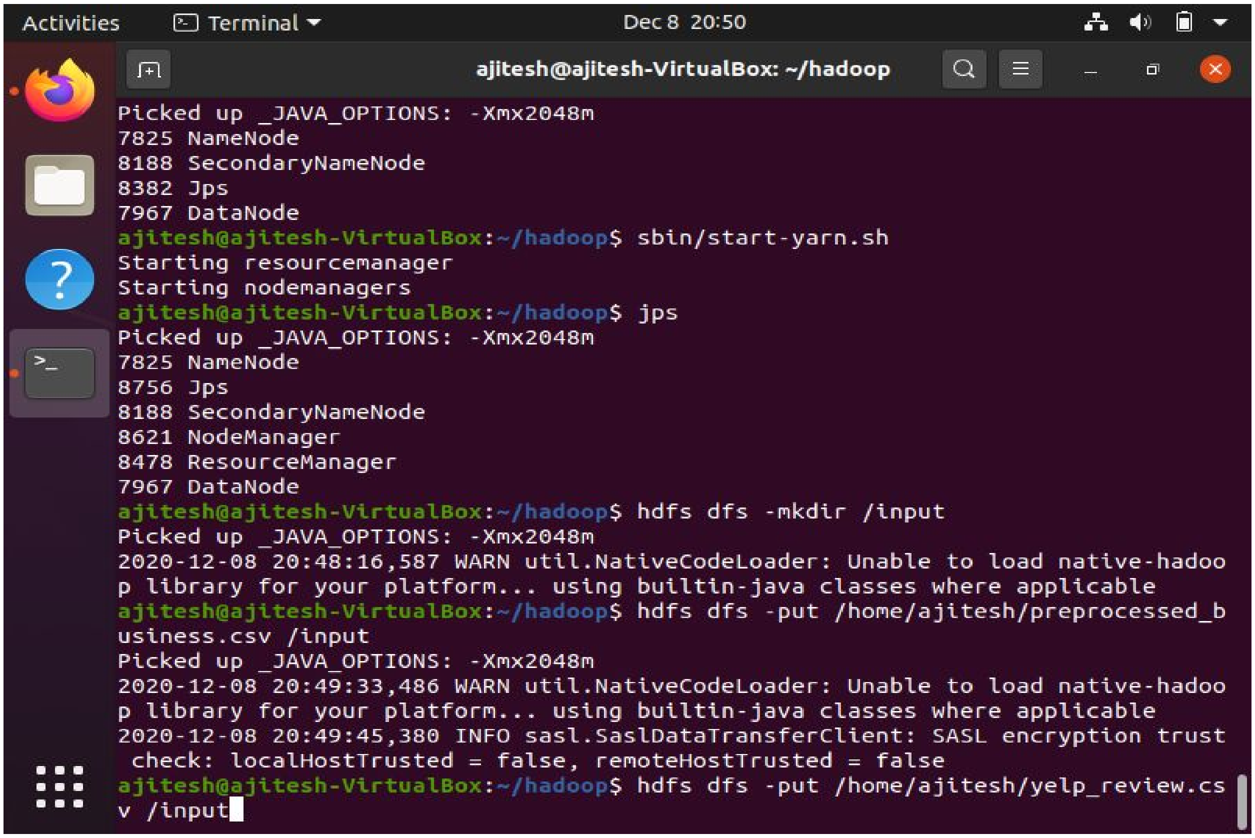


```
In [2]: import pandas as pd
        business = pd.read_csv('yelp_business.csv')

In [3]: len(business)

Out[3]: 174567

In [5]: a=[]
        for i in range(0,174567):
            if("Restaurant" not in business['categories'][i]):
                a.append(i)

In [8]: business=business.drop(a)

In [9]: business.head()

Out[9]:
```

	business_id	name	neighborhood	address	city	state	postal_code	latitude	longitude	stars	review_count	is_ope
4	PfOCPjBrlQAnz__NXj9h_w	"Brick House Tavern + Tap"	NaN	"581 Howe Ave"	Cuyahoga Falls	OH	44221	41.119535	-81.475690	3.5	116	1
5	o9eMRCWt5PkpLDE0gOPtcQ	"Messina"	NaN	"Richterstr. 11"	Stuttgart	BW	70567	48.727200	9.147950	4.0	5	1
10	XOSRcvtaKc_Q5H1SAzN20A	"East Coast Coffee"	NaN	"737 West Pike St"	Houston	PA	15342	40.241548	-80.212815	4.5	3	0
14	fNMVV_ZX7CJSDWQGdOM8Nw	"Showmars Government Center"	Uptown	"600 E 4th St"	Charlotte	NC	28202	35.221647	-80.839345	3.5	7	1
15	l09JfMeQ6ynYs5MCJtrcmQ	"Alize Catering"	Yonge and Eglinton	"2459 Yonge St"	Toronto	ON	M4P 2H6	43.711399	-79.399339	3.0	12	0

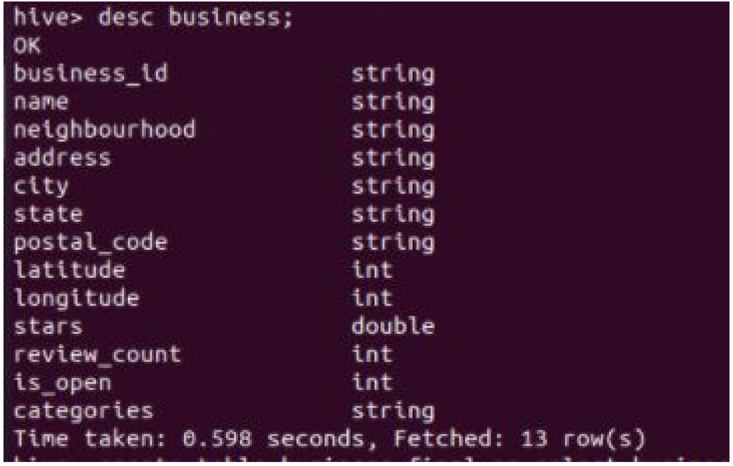
```
In [10]: len(business)

Out[10]: 54630

In [13]: business.to_csv('preprocessed_business.csv',index=False,sep='^',encoding='utf-8')
```

We then put the data into hdfs.

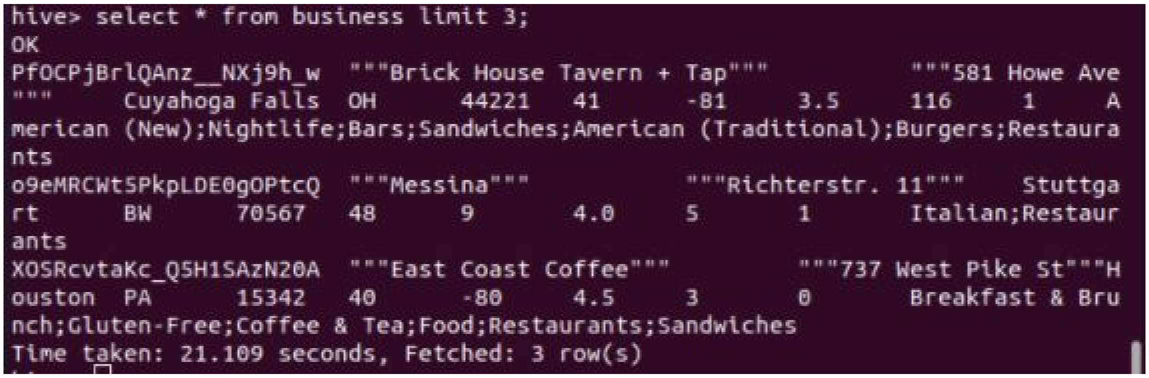


```
Picked up _JAVA_OPTIONS: -Xmx2048m
7825 NameNode
8188 SecondaryNameNode
8382 Jps
7967 DataNode
ajitesh@ajitesh-VirtualBox:~/hadoop$ sbin/start-yarn.sh
Starting resourcemanager
Starting nodemanagers
ajitesh@ajitesh-VirtualBox:~/hadoop$ jps
Picked up _JAVA_OPTIONS: -Xmx2048m
7825 NameNode
8756 Jps
8188 SecondaryNameNode
8621 NodeManager
8478 ResourceManager
7967 DataNode
ajitesh@ajitesh-VirtualBox:~/hadoop$ hdfs dfs -mkdir /input
Picked up _JAVA_OPTIONS: -Xmx2048m
2020-12-08 20:48:16,587 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
ajitesh@ajitesh-VirtualBox:~/hadoop$ hdfs dfs -put /home/ajitesh/preprocessed_b
usiness.csv /input
Picked up _JAVA_OPTIONS: -Xmx2048m
2020-12-08 20:49:33,486 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
2020-12-08 20:49:45,380 INFO sasl.SaslDataTransferClient: SASL encryption trust
 check: localHostTrusted = false, remoteHostTrusted = false
ajitesh@ajitesh-VirtualBox:~/hadoop$ hdfs dfs -put /home/ajitesh/yelp_review.cs
v /input
```

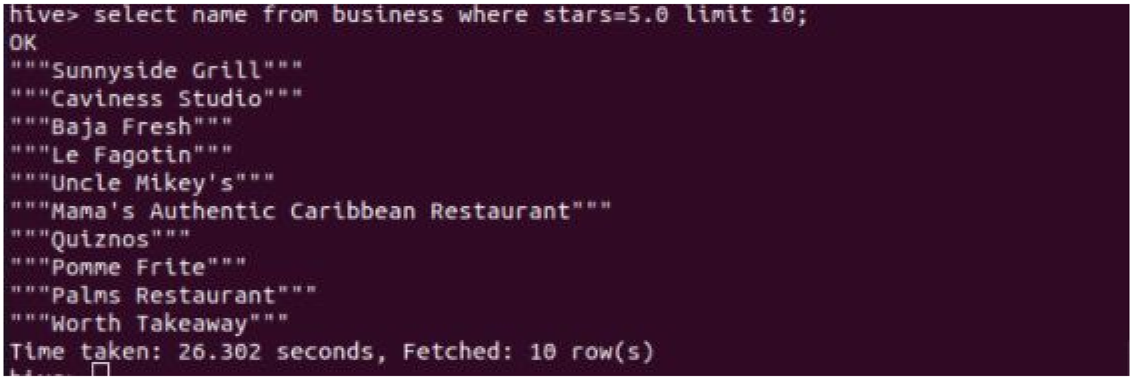
We then do some basic analysis using Hive.



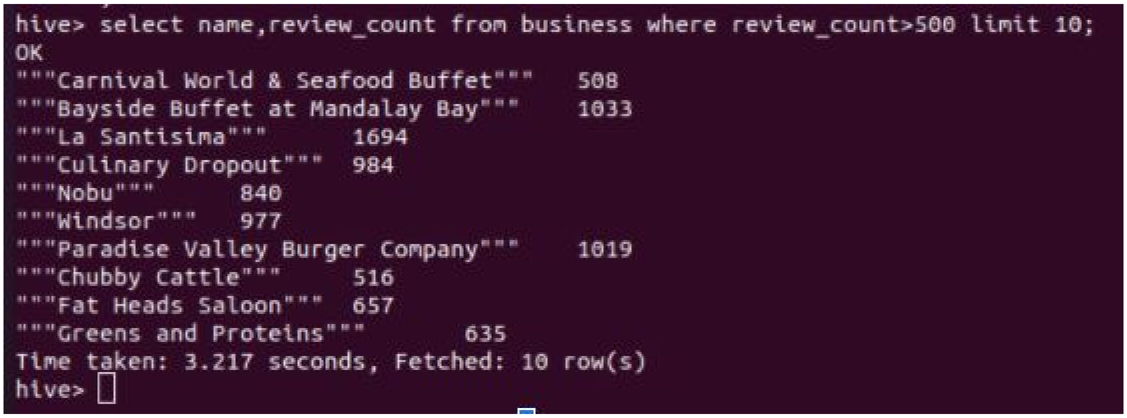
```
hive> desc business;
OK
business_id             string
name                    string
neighbourhood           string
address                 string
city                    string
state                   string
postal_code             string
latitude                int
longitude               int
stars                   double
review_count            int
is_open                 int
categories              string
Time taken: 0.598 seconds, Fetched: 13 row(s)
```



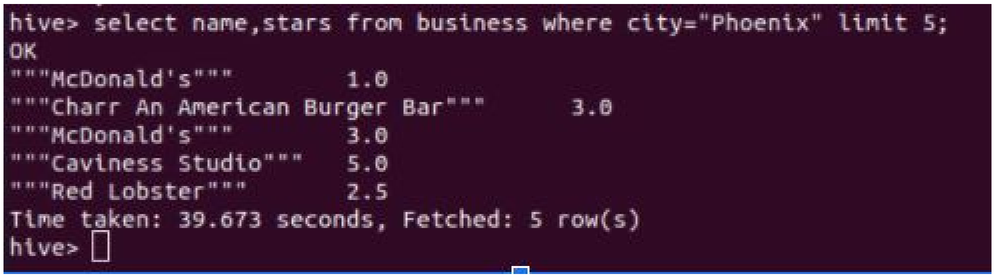
```
hive> select * from business limit 3;
OK
PfOCPjBrlQAnz__NXj9h_w  """Brick House Tavern + Tap"""          """581 Howe Ave
"""     Cuyahoga Falls  OH      44221   41      -81     3.5     116     1       A
merican (New);Nightlife;Bars;Sandwiches;American (Traditional);Burgers;Restaura
nts
o9eMRCWt5PkpLDE0gOPtcQ  """Messina"""           """Richterstr. 11"""    Stuttga
rt      BW      70567   48      9       4.0     5       1       Italian;Restaur
ants
XOSRcvtaKc_Q5H1SAzN20A  """East Coast Coffee"""         """737 West Pike St"""H
ouston  PA      15342   40      -80     4.5     3       0       Breakfast & Bru
nch;Gluten-Free;Coffee & Tea;Food;Restaurants;Sandwiches
Time taken: 21.109 seconds, Fetched: 3 row(s)
```



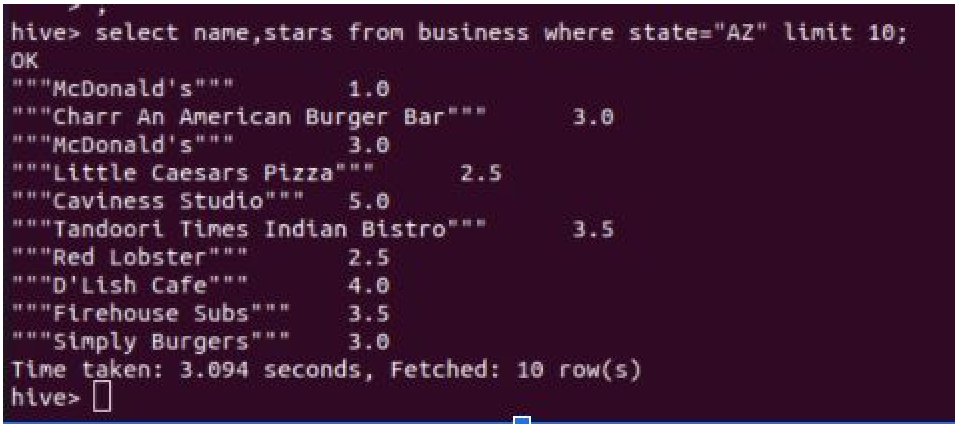
```
hive> select name from business where stars=5.0 limit 10;
OK
"""Sunnyside Grill"""
"""Caviness Studio"""
"""Baja Fresh"""
"""Le Fagotin"""
"""Uncle Mikey's"""
"""Mama's Authentic Caribbean Restaurant"""
"""Quiznos"""
"""Pomme Frite"""
"""Palms Restaurant"""
"""Worth Takeaway"""
Time taken: 26.302 seconds, Fetched: 10 row(s)
hive>
```



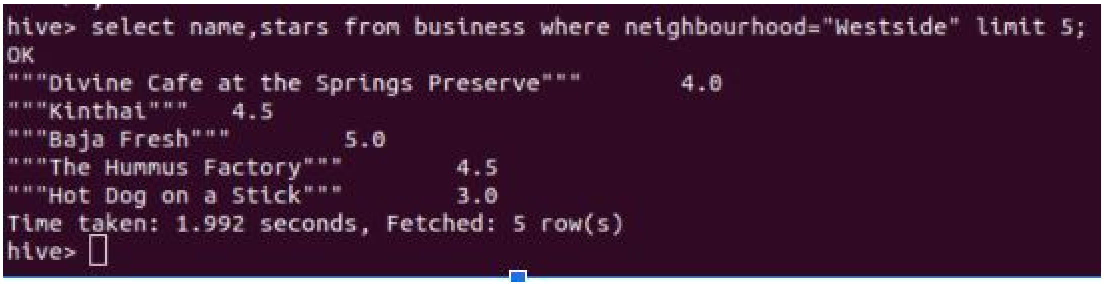
```
hive> select name,review_count from business where review_count>500 limit 10;
OK
"""Carnival World & Seafood Buffet"""   508
"""Bayside Buffet at Mandalay Bay"""    1033
"""La Santisima"""      1694
"""Culinary Dropout"""  984
"""Nobu"""      840
"""Windsor"""   977
"""Paradise Valley Burger Company"""    1019
"""Chubby Cattle"""     516
"""Fat Heads Saloon"""  657
"""Greens and Proteins"""       635
Time taken: 3.217 seconds, Fetched: 10 row(s)
hive>
```



```
hive> select name,stars from business where city="Phoenix" limit 5;
OK
"""McDonald's"""        1.0
"""Charr An American Burger Bar"""      3.0
"""McDonald's"""        3.0
"""Caviness Studio"""   5.0
"""Red Lobster"""       2.5
Time taken: 39.673 seconds, Fetched: 5 row(s)
hive>
```

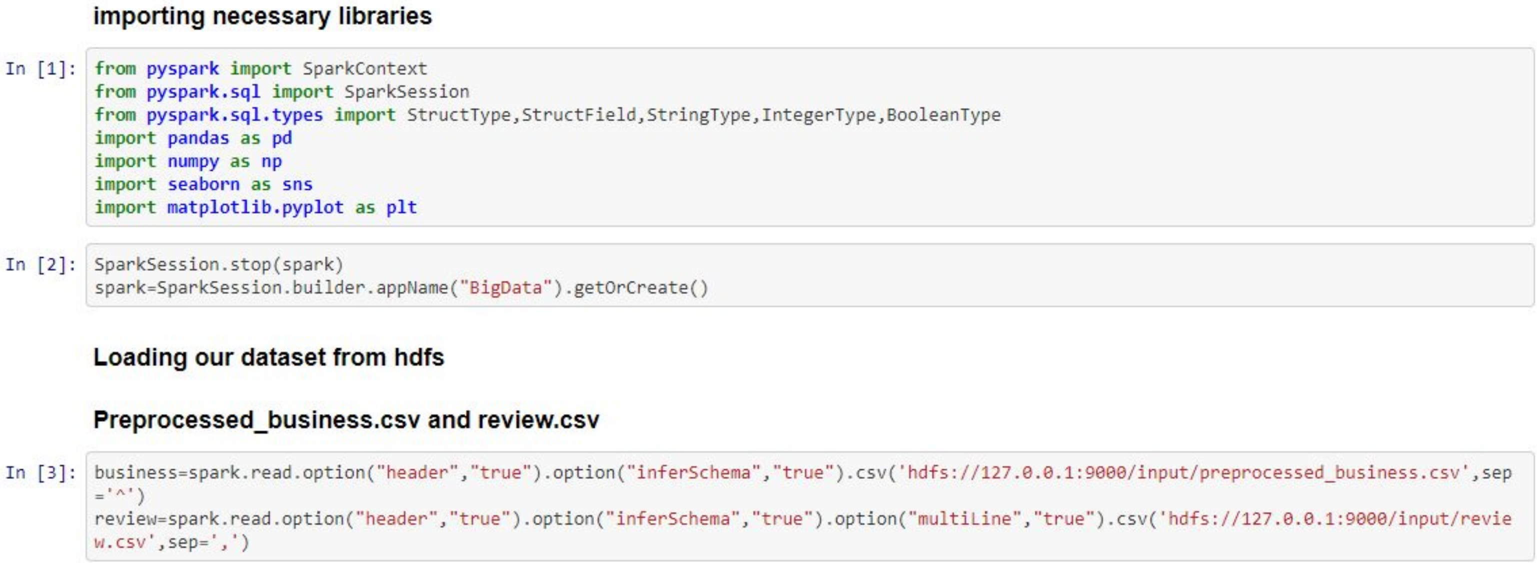


```
hive> select name,stars from business where state="AZ" limit 10;
OK
"""McDonald's"""        1.0
"""Charr An American Burger Bar"""      3.0
"""McDonald's"""        3.0
"""Little Caesars Pizza"""      2.5
"""Caviness Studio"""   5.0
"""Tandoori Times Indian Bistro"""      3.5
"""Red Lobster"""       2.5
"""D'Lish Cafe"""       4.0
"""Firehouse Subs"""    3.5
"""Simply Burgers"""    3.0
Time taken: 3.094 seconds, Fetched: 10 row(s)
hive>
```



```
hive> select name,stars from business where neighbourhood="Westside" limit 5;
OK
"""Divine Cafe at the Springs Preserve"""       4.0
"""Kinthai"""   4.5
"""Baja Fresh"""        5.0
"""The Hummus Factory"""        4.5
"""Hot Dog on a Stick"""        3.0
Time taken: 1.992 seconds, Fetched: 5 row(s)
hive> 
```

We now start our analysis and processing using Spark.



importing necessary libraries

```
In [1]: from pyspark import SparkContext
        from pyspark.sql import SparkSession
        from pyspark.sql.types import StructType,StructField,StringType,IntegerType,BooleanType
        import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
```

```
In [2]: SparkSession.stop(spark)
        spark=SparkSession.builder.appName("BigData").getOrCreate()
```

Loading our dataset from hdfs

Preprocessed_business.csv and review.csv

```
In [3]: business=spark.read.option("header","true").option("inferSchema","true").csv('hdfs://127.0.0.1:9000/input/preprocessed_business.csv',sep
        ='^')
        review=spark.read.option("header","true").option("inferSchema","true").option("multiLine","true").csv('hdfs://127.0.0.1:9000/input/revie
        w.csv',sep=',')
```

Lets see what our business dataset looks like

```
In [4]: business.printSchema()
```

```
root
 |-- business_id: string (nullable = true)
 |-- name: string (nullable = true)
 |-- neighborhood: string (nullable = true)
 |-- address: string (nullable = true)
 |-- city: string (nullable = true)
 |-- state: string (nullable = true)
 |-- postal_code: string (nullable = true)
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
 |-- stars: double (nullable = true)
 |-- review_count: integer (nullable = true)
 |-- is_open: integer (nullable = true)
 |-- categories: string (nullable = true)
```

One row from our business dataset

```
In [5]: business.show(1)
```

```
+--------------------+--------------------+------------+------------------+--------------+-----+-----------+----------+-----------+-----+
------------+-------+--------------------+
|         business_id|                name|neighborhood|           address|          city|state|postal_code|  latitude|  longitude|stars|
review_count|is_open|          categories|
+--------------------+--------------------+------------+------------------+--------------+-----+-----------+----------+-----------+-----+
------------+-------+--------------------+
|PfOCPjBrlQAnz__NX...|"""Brick House Ta...|        null|"""581 Howe Ave"""|Cuyahoga Falls|   OH|      44221|41.1195346|-81.4756898|  3.5|
116|      1|American (New);Ni...|
+--------------------+--------------------+------------+------------------+--------------+-----+-----------+----------+-----------+-----+
------------+-------+--------------------+
only showing top 1 row
```

Now lets look at our review dataset

```
In [6]: review.printSchema()
```

```
root
 |-- review_id: string (nullable = true)
 |-- user_id: string (nullable = true)
 |-- business_id: string (nullable = true)
 |-- stars: string (nullable = true)
 |-- date: string (nullable = true)
 |-- text: string (nullable = true)
 |-- useful: string (nullable = true)
 |-- funny: string (nullable = true)
 |-- cool: string (nullable = true)
```

```
In [7]: review.show(1)
```

```
+--------------------+--------------------+--------------------+-----+----------+--------------------+------+-----+----+
|           review_id|             user_id|         business_id|stars|      date|                text|useful|funny|cool|
+--------------------+--------------------+--------------------+-----+----------+--------------------+------+-----+----+
|vkVSCC7xljjrAI4UG...|bv2nCi5Qv5vroFiqK...|AEx2SYEUJmTxVVB18...|    5|2016-05-28|Super simple plac...|     0|    0|   0|
+--------------------+--------------------+--------------------+-----+----------+--------------------+------+-----+----+
only showing top 1 row
```

Now lets start our analysis

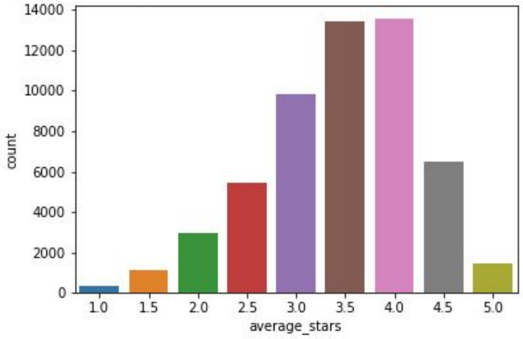
Lets group the businesses based on their average rating

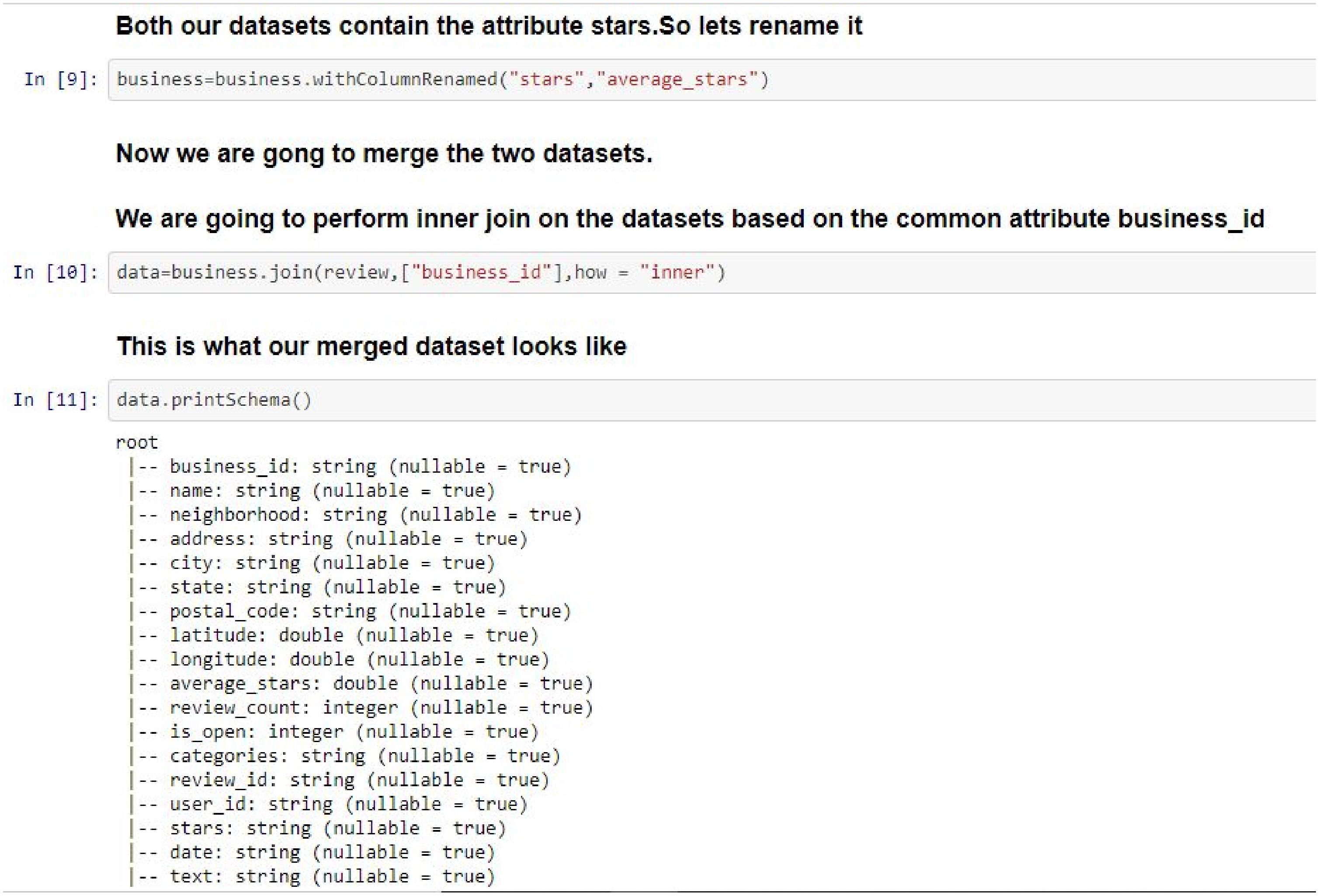
```
In [8]: business.groupby("stars").count().show()
```

```
+-----+-----+
|stars|count|
+-----+-----+
|  3.5|13391|
|  4.5| 6519|
|  2.5| 5448|
|  1.0|  348|
|  4.0|13529|
|  3.0| 9826|
|  2.0| 2946|
|  1.5| 1131|
|  5.0| 1492|
+-----+-----+
```

```
In [19]: reviewcount=business.groupby("average_stars").count().toPandas()
         sns.barplot(reviewcount['average_stars'],reviewcount['count'])
         plt.show()
```

```
/home/ajitesh/.local/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables as keyword args:
x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will
result in an error or misinterpretation.
  warnings.warn(
```





Both our datasets contain the attribute stars.So lets rename it

```
In [9]: business=business.withColumnRenamed("stars","average_stars")
```

Now we are gong to merge the two datasets.

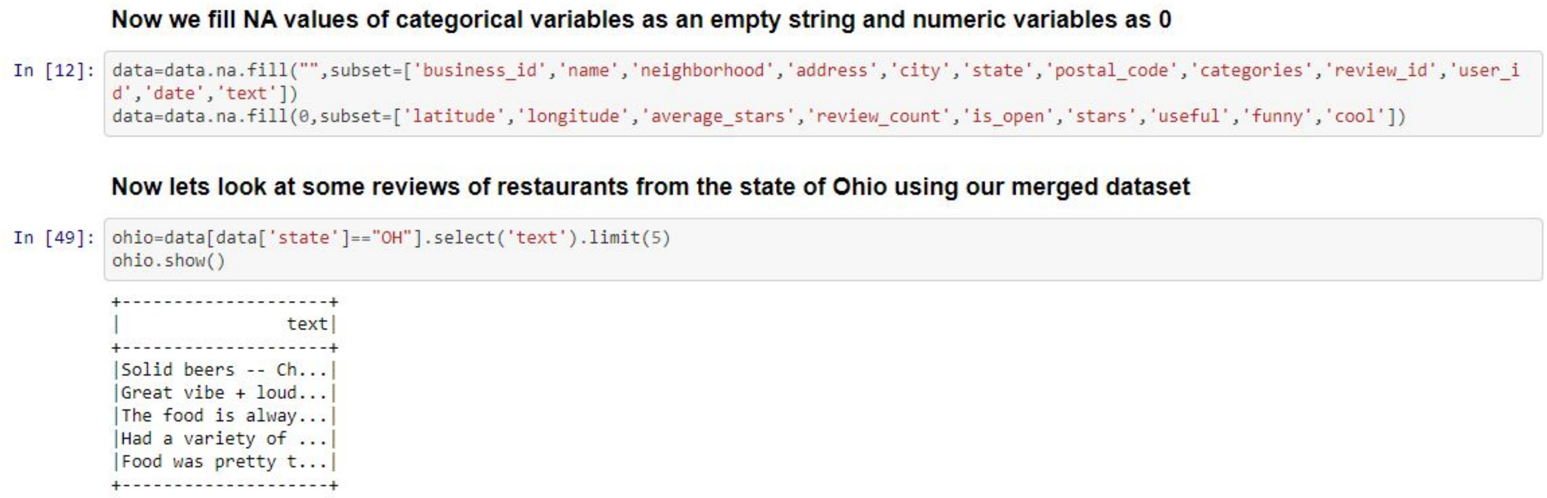
We are going to perform inner join on the datasets based on the common attribute business_id

```
In [10]: data=business.join(review,["business_id"],how = "inner")
```

This is what our merged dataset looks like

```
In [11]: data.printSchema()
```

```
root
 |-- business_id: string (nullable = true)
 |-- name: string (nullable = true)
 |-- neighborhood: string (nullable = true)
 |-- address: string (nullable = true)
 |-- city: string (nullable = true)
 |-- state: string (nullable = true)
 |-- postal_code: string (nullable = true)
 |-- latitude: double (nullable = true)
 |-- longitude: double (nullable = true)
 |-- average_stars: double (nullable = true)
 |-- review_count: integer (nullable = true)
 |-- is_open: integer (nullable = true)
 |-- categories: string (nullable = true)
 |-- review_id: string (nullable = true)
 |-- user_id: string (nullable = true)
 |-- stars: string (nullable = true)
 |-- date: string (nullable = true)
 |-- text: string (nullable = true)
```



Now we fill NA values of categorical variables as an empty string and numeric variables as 0

```
In [12]: data=data.na.fill("",subset=['business_id','name','neighborhood','address','city','state','postal_code','categories','review_id','user_id','date','text'])
         data=data.na.fill(0,subset=['latitude','longitude','average_stars','review_count','is_open','stars','useful','funny','cool'])
```

Now lets look at some reviews of restaurants from the state of Ohio using our merged dataset

```
In [49]: ohio=data[data['state']=="OH"].select('text').limit(5)
         ohio.show()
```

```
+--------------------+
|                text|
+--------------------+
|Solid beers -- Ch...|
|Great vibe + loud...|
|The food is alway...|
|Had a variety of ...|
|Food was pretty t...|
+--------------------+
```

Now lets count the number of restaurants in each city

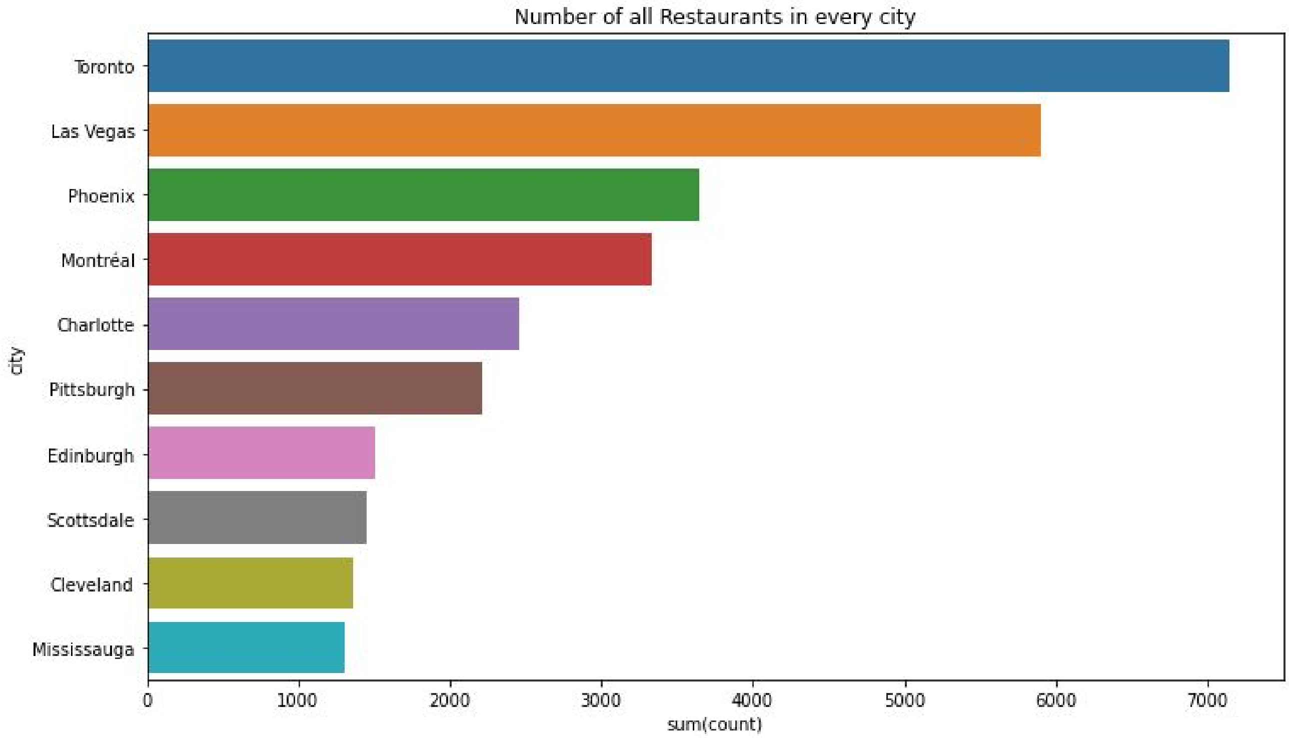
```
In [13]: city_business=business.groupby('city').count()
         city_business=city_business.groupby('city').sum()
```

```
In [14]: city_business=city_business.sort('sum(count)',ascending=False)
         city_business.show(5)
```

```
+---------+----------+
|     city|sum(count)|
+---------+----------+
|  Toronto|      7148|
|Las Vegas|      5902|
|  Phoenix|      3652|
| Montréal|      3332|
|Charlotte|      2461|
+---------+----------+
only showing top 5 rows
```

```
In [15]: city_business=city_business.limit(10).toPandas()
         plt.figure(figsize=(12,7))
         sns.barplot(y='city',x='sum(count)',data=city_business)
         plt.title('Number of all Restaurants in every city')
```

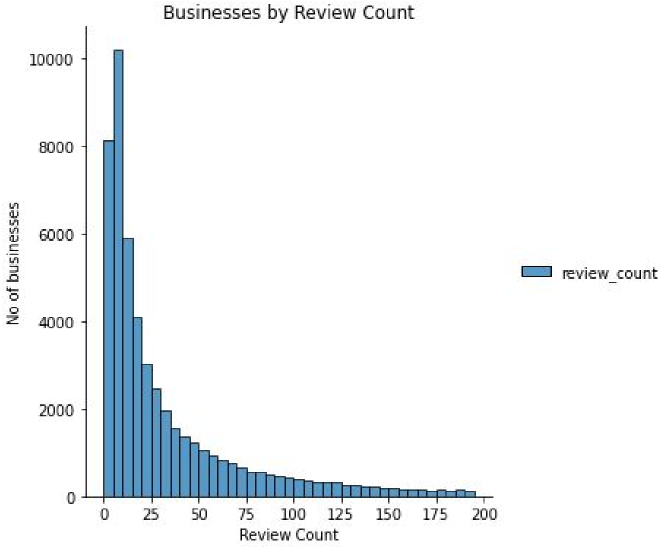
```
Out[15]: Text(0.5, 1.0, 'Number of all Restaurants in every city')
```



Distribution of Review Count

```
In [16]: reviewcount=business.select('review_count').toPandas()
         sns.displot(reviewcount,bins=range(0,200,5))
         plt.xlabel('Review Count')
         plt.ylabel('No of businesses')
         plt.title('Businesses by Review Count')
```

```
Out[16]: Text(0.5, 1.0, 'Businesses by Review Count')
```

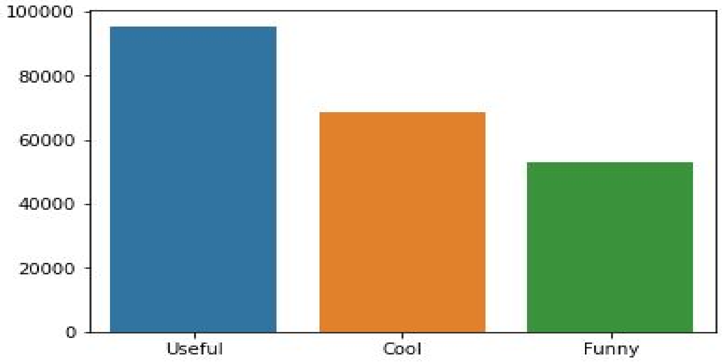


Lets analyse the reviews marked as useful,cool and funny

```
In [17]: useful_review=review[review['useful']=='1'].count()
         cool_review=review[review['cool']=='1'].count()
         funny_review=review[review['funny']=='1'].count()
```

```
In [18]: x=['Useful','Cool','Funny']
         y=[useful_review,cool_review,funny_review]
         sns.barplot(x,y)
         plt.show()
```

```
/home/ajitesh/.local/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pa
x, y. From version 0.12, the only valid positional argument will be `data`, and passing other
result in an error or misinterpretation.
  warnings.warn(
```



Lets analyse the reviews based on rating

We select 1000 reviews from our dataset which had the rating of 1/5.This will be considered as negative reviews.

Similarly we consider 5/5 stars as positive reviews.

In [29]:

```
stars_review1=review[review['stars']=='1'].select('stars','text').limit(1000).toPandas()
stars_review5=review[review['stars']=='5'].select('stars','text').limit(1000).toPandas()
```

Lets replace NA values with empty string

In [30]:

```
stars_review1['text'].fillna('',inplace=True)
stars_review5['text'].fillna('',inplace=True)
```

Now we remove Punctuations,Digits and special characters

In [31]:

```
import re
def remove_punctuation_digits_specialchar(line):
    return re.sub('[^A-Za-z]+',' ',line).lower()
```

In [32]:

```
stars_review1['clean_review']=stars_review1['text'].apply(remove_punctuation_digits_specialchar)
stars_review1[['text','clean_review']].head()
```

Out[32]:

	text	clean_review
0	I thought Tidy's Flowers had a great reputatio...	i thought tidy s flowers had a great reputatio...
1	I too have been trying to book an appt to use ...	i too have been trying to book an appt to use ...
2	really excited to hear of this restaurant comi...	really excited to hear of this restaurant comi...
3	Food is very bland - not authentic at all.\n\n...	food is very bland not authentic at all meant ...
4	If you have not yet tried Wasabi - don't bothe...	if you have not yet tried wasabi don t bother ...

In [33]:

```
stars_review5['clean_review']=stars_review5['text'].apply(remove_punctuation_digits_specialchar)
```

Now we remove the stop words

In [34]:

```
import nltk
from nltk.util import ngrams
from nltk.corpus import stopwords
def tokenize_no_stopwords(line):
    tokens=nltk.tokenize.word_tokenize(line)
    tokens_no_stop=[w for w in tokens if w not in stopwords.words('english')]
    return " ".join(tokens_no_stop)
```

In [35]:

```
stars_review1['final_review']=stars_review1['clean_review'].apply(tokenize_no_stopwords)
stars_review1[['clean_review','final_review']].head()
```

Out[35]:

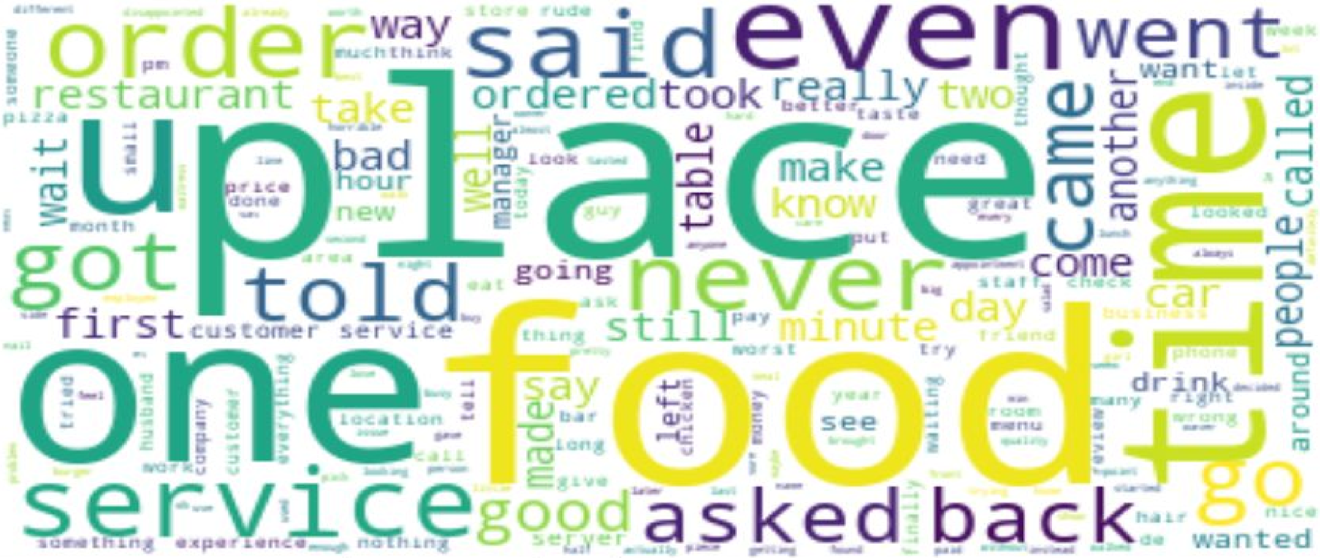
	clean_review	final_review
0	i thought tidy s flowers had a great reputatio...	thought tidy flowers great reputation florist ...
1	i too have been trying to book an appt to use ...	trying book appt use voucher months countless ...
2	really excited to hear of this restaurant comi...	really excited hear restaurant coming toronto ...
3	food is very bland not authentic at all meant ...	food bland authentic meant cater customers nev...
4	if you have not yet tried wasabi don t bother ...	yet tried wasabi bother expensive food disgust...

In [36]:

```
stars_review5['final_review']=stars_review5['clean_review'].apply(tokenize_no_stopwords)
```

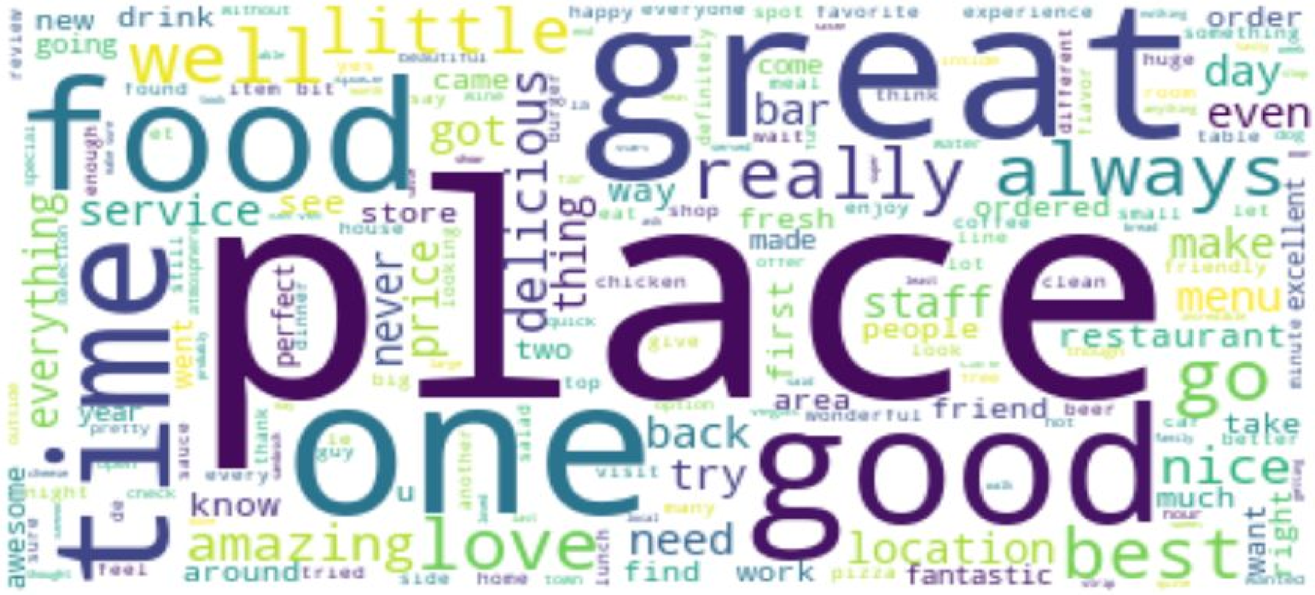
Now lets look at the 200 most commonly used words in negative reviews using Wordcloud

```
In [38]: import wordcloud
         from wordcloud import WordCloud,ImageColorGenerator
         text=" ".join(str(each) for each in stars_review1.final_review)
         wordcloud=WordCloud(max_words=200,background_color="white").generate(text)
         plt.figure(figsize=(15,10))
         plt.imshow(wordcloud,interpolation='bilinear')
         plt.axis("off")
         plt.show()
         pass
```



And the 200 most common words used in positive reviews.

```
In [39]: text=" ".join(str(each) for each in stars_review5.final_review)
         wordcloud=WordCloud(max_words=200,background_color="white").generate(text)
         plt.figure(figsize=(15,10))
         plt.imshow(wordcloud,interpolation='bilinear')
         plt.axis("off")
         plt.show()
         pass
```


new
drink
little
happy
everyone
spot
favorite
experience
order
going
well
great
something
review
food
came
come
meal
bar
think
different
huge
day
even
got
say
delicious
burger
definitely
really
always
table
special
enough
service
see
store
house
way
fresh
shop
eat
enjoy
ordered
coffee
small
let
everything
thing
made
line
lot
make
friendly
time
place
never
price
perfect
dinner
chicken
clean
staff
menu
excellent
atmosphere
still
went
look
people
restaurant
two
give
first
top
big
go
year
pretty
one
area
wonderful
friend
beer
take
better
back
try
nice
much
night
check
every
visit
know
thank
guy
u
many
awesome
amazing
love
need
location
best
want
right
around
tried
side
home
town
lunch
find
work
pizza
fantastic
feel
sure
minute
salad
another
sauce