Data Scientist Exercise

This dataset consists of approximately 1.5 million beer reviews from Beer Advocate. Please use this dataset to answer the following questions.

1. Which brewery produces the strongest beers by ABV%?
2. If you had to pick 3 beers to recommend using only this data, which would you pick?
3. Which of the factors (aroma, taste, appearance, palette) are most important in determining the overall quality of a beer?
4. Lastly, if I typically enjoy a beer due to its aroma and appearance, which beer style should I try?

In [1]:
```
# import necessary packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:
```
# read data into the beer data file
beers = pd.read_csv('beer_reviews.csv')
```

In [3]:
```
# copy the original file to a file I can manipulate without losing original data
beers_clean = beers.copy()
```

In [4]:
```
# drop review time from the dataset, it has significance in my investigation
beers_clean = beers_clean.drop('review_time', axis=1)
```

In [5]:
```
beers_clean.head()
```

Out[5]:

	brewery_id	brewery_name	review_overall	review_aroma	review_appearance	review_profilename
0	10325	Vecchio Birraio	1.5	2.0	2.5	stcules
1	10325	Vecchio Birraio	3.0	2.5	3.0	stcules
2	10325	Vecchio Birraio	3.0	2.5	3.0	stcules
3	10325	Vecchio Birraio	3.0	3.0	3.5	stcules
4	1075	Caldera Brewing Company	4.0	4.5	4.0	johnmichaelsen

In [6]:
```
# check for duplicated rows in the data
sum(beers_clean.duplicated())
```

Out[6]: 774

In [7]:
```
beers_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1586614 entries, 0 to 1586613
Data columns (total 12 columns):
brewery_id            1586614 non-null int64
brewery_name          1586599 non-null object
review_overall        1586614 non-null float64
review_aroma          1586614 non-null float64
review_appearance     1586614 non-null float64
review_profilename    1586266 non-null object
beer_style            1586614 non-null object
review_palate         1586614 non-null float64
review_taste          1586614 non-null float64
beer_name             1586614 non-null object
beer_abv              1518829 non-null float64
beer_beerid           1586614 non-null int64
dtypes: float64(6), int64(2), object(4)
memory usage: 145.3+ MB
```

In [8]:
```
# remove duplicate rows from the data
beers_clean.drop_duplicates(inplace = True)
```

In [9]:
```
# check row counts to make sure I deleted 774 rows
beers_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1585840 entries, 0 to 1586613
Data columns (total 12 columns):
brewery_id            1585840 non-null int64
brewery_name          1585825 non-null object
review_overall        1585840 non-null float64
review_aroma          1585840 non-null float64
review_appearance     1585840 non-null float64
review_profilename    1585493 non-null object
beer_style            1585840 non-null object
review_palate         1585840 non-null float64
review_taste          1585840 non-null float64
beer_name             1585840 non-null object
beer_abv              1518078 non-null float64
beer_beerid           1585840 non-null int64
dtypes: float64(6), int64(2), object(4)
memory usage: 157.3+ MB
```

In [10]:
```
# check for null values in the dataset
beers_clean.isnull().values.any()
```

Out[10]: True

I will remove all null values from the dataset. With 1.5M data points it will not affect any resuls in a statistically significant manner

In [11]:
```
# remove all rows that contain null values.  With 1.5M rows of data this will not h
ave a significant affect on calculated numbers
beers_clean = beers_clean.dropna(how='any',axis=0)
```

In [12]:
```
beers_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1517728 entries, 0 to 1586613
Data columns (total 12 columns):
brewery_id            1517728 non-null int64
brewery_name          1517728 non-null object
review_overall        1517728 non-null float64
review_aroma          1517728 non-null float64
review_appearance     1517728 non-null float64
review_profilename    1517728 non-null object
beer_style            1517728 non-null object
review_palate         1517728 non-null float64
review_taste          1517728 non-null float64
beer_name             1517728 non-null object
beer_abv              1517728 non-null float64
beer_beerid           1517728 non-null int64
dtypes: float64(6), int64(2), object(4)
memory usage: 150.5+ MB
```

In [13]:
```
# view the cleaned dataset
beers_clean.head()
```

Out[13]:

	brewery_id	brewery_name	review_overall	review_aroma	review_appearance	review_profilename
0	10325	Vecchio Birraio	1.5	2.0	2.5	stcules
1	10325	Vecchio Birraio	3.0	2.5	3.0	stcules
2	10325	Vecchio Birraio	3.0	2.5	3.0	stcules
3	10325	Vecchio Birraio	3.0	3.0	3.5	stcules
4	1075	Caldera Brewing Company	4.0	4.5	4.0	johnmichaelsen

With the cleaned date I will explore Question 1

Which brewery produces the strongest beers by ABV%?

In [14]:
```
# group by brewery name and calculate the mean of all numerical columns
beers_brewgroup = beers_clean.groupby('brewery_name').mean()
```

In [15]:
```
# sort the new table by ABV% to determine highest alcohol content
beers_brewgroup.sort_values(by='beer_abv', ascending=False)
```

Out[15]:

	brewery_id	review_overall	review_aroma	review_appearance	review_palate	re
brewery_name						
Schorschbräu	6513.0	3.411765	3.529412	3.558824	3.470588	3.5
Shoes Brewery	14060.0	3.000000	3.000000	3.750000	3.500000	3.2
Rome Brewing Company	2873.0	4.100000	3.600000	3.800000	3.900000	4.4
Hurlimann Brewery	736.0	3.805556	4.333333	3.916667	4.083333	4.2
Alt-Oberurseler Brauhaus	10038.0	4.000000	4.500000	4.000000	4.500000	4.0
Rascal Creek Brewing Co.	21755.0	5.000000	5.000000	5.000000	5.000000	5.0
Monks Porter House	24215.0	3.833333	4.000000	3.833333	3.833333	3.8
Brasserie Grain d' Orge (Brasserie Jeanne d'Arc SA)	36.0	3.229299	3.466561	3.652866	3.485669	3.4
Tugboat Brewing Company	3452.0	3.500000	3.625000	3.687500	3.437500	3.6
United Brands Company	21678.0	1.884615	2.307692	2.846154	2.057692	2.0
Morgan Street Brewery	593.0	3.850000	3.700000	3.950000	3.750000	4.0
Snowy Mountain Brewery	19602.0	4.250000	4.250000	3.750000	3.500000	4.5
Rinkukių Aluas Darykla	23345.0	3.136364	3.545455	3.136364	2.954545	3.0
Brauerei Schloss Eggenberg	285.0	3.595552	3.824547	3.754256	3.820977	3.8
Etna Brewery	8540.0	4.250000	4.250000	4.375000	4.250000	4.3
Nasu Kogen Beer Co. Ltd.	5040.0	4.500000	4.250000	4.500000	4.000000	4.5
Brasserie Dubuisson Frères sprl	604.0	3.714949	3.908542	3.881151	3.846797	3.8
Kuhnhenn Brewing Company	2097.0	3.964975	4.161008	3.922802	4.009292	4.1
Main Street Brewery / Turoni's Pizza	639.0	3.916667	3.750000	3.916667	3.750000	3.9
Water Street Brewing & Ale House	10226.0	2.500000	2.500000	4.000000	4.000000	2.5
Wibblers	10191.0	4.000000	3.500000	4.000000	4.000000	4.5
Zago S.r.l.	19362.0	4.500000	4.000000	4.000000	4.000000	4.5

In [16]:
```
beers_clean[beers_clean['beer_abv']==beers_clean['beer_abv'].max()]
```

Out[16]:

	brewery_id	brewery_name	review_overall	review_aroma	review_appearance	review_profiler
12919	6513	Schorschbräu	4.0	4.0	4.0	kappldav123

In [17]:
```
# sort the original sheet by ABV% to determine which brewery the single beer with t
he most alcohol
beers.sort_values(by='beer_abv', ascending=False)
```

Out[17]:

	brewery_id	brewery_name	review_time	review_overall	review_aroma	review_appearance
12919	6513	Schorschbräu	1316780901	4.0	4.0	4.0
12939	6513	Schorschbräu	1309974178	4.0	4.0	3.5
12940	6513	Schorschbräu	1274469798	3.5	4.0	4.0
746385	16315	BrewDog	1285808609	3.5	4.0	4.0
746387	16315	BrewDog	1285274059	3.0	3.0	3.0
746386	16315	BrewDog	1285665487	2.5	3.0	3.5
746384	16315	BrewDog	1288121648	2.0	3.0	2.5
746358	16315	BrewDog	1307999104	3.0	2.5	3.0
746359	16315	BrewDog	1307202043	4.5	5.0	4.5
746360	16315	BrewDog	1307142237	1.0	3.5	2.0
746365	16315	BrewDog	1302716098	5.0	5.0	4.5
746366	16315	BrewDog	1301350582	3.5	4.0	4.5
746357	16315	BrewDog	[illegible]	3.5	5.0	4.5

Schorschbräu brewery has both the highest alcohol content by volume across all beers produced at 19.22% and the beer with the single highest alchol content with the Schorschbräu Schorschbock at 57%

If you had to pick 3 beers to recommend using only this data, which would you pick?

I tend to prefer an ale over most other beer types. I'll query the data frame and build a new dataset that only contains Ale's. Once I do that I can group the data by the beer name, calculate the mean reviews across all categories and sort the information based on the returned data.

In [18]:
```
beer_ale = beers_clean[beers_clean['beer_style'].str.contains("Ale")]
```

In [19]:
```
# build a new data file grouped by beer_name
beers_choice = beer_ale.groupby('beer_name').mean()
```

In [20]:
```
# create a total_review column that includes all rating categories
beers_choice['tot_review'] = ((beers_choice['review_overall'] + beers_choice['review_aroma'] + beers_choice['review_appearance'] + beers_choice['review_palate'] + beers_choice['review_taste'])/5)
```

In [21]:
```
beers_choice.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 14977 entries, ! (Old Ale) to ÜberSun (Imperial Summer Wheat Beer)
Data columns (total 9 columns):
brewery_id           14977 non-null float64
review_overall       14977 non-null float64
review_aroma         14977 non-null float64
review_appearance    14977 non-null float64
review_palate        14977 non-null float64
review_taste         14977 non-null float64
beer_abv             14977 non-null float64
beer_beerid          14977 non-null float64
tot_review           14977 non-null float64
dtypes: float64(9)
memory usage: 1.1+ MB
```

In [22]:
```
# sort new data sheet for the highest total_review
beers_choice.sort_values(by='tot_review', ascending=False)
```

Out[22]:

	brewery_id	review_overall	review_aroma	review_appearance	review_palate	review_ta
beer_name						
Great Lakes Truth Justice And The American Ale	73.0	5.000000	5.000000	5.000000	5.000000	5.000000
Edsten Triple-Wit	387.0	5.000000	5.000000	5.000000	5.000000	5.000000
Engelbert Moonbeam	642.0	5.000000	5.000000	5.000000	5.000000	5.000000
Lips Of Faith - Eric's Ale (Bourbon Barrel Aged)	192.0	5.000000	4.750000	4.750000	5.000000	5.000000
Opus Altar Boy	30.0	5.000000	5.000000	5.000000	5.000000	4.500000
Dry Hopped Abominable Ale	16353.0	5.000000	5.000000	5.000000	4.500000	5.000000
De Dolle Stille Nacht Special Reserva 2008	201.0	5.000000	5.000000	5.000000	5.000000	4.500000
Tetley's Mild	8535.0	5.000000	5.000000	5.000000	4.500000	5.000000
Love Child Belgiweizen	17282.0	5.000000	4.750000	4.750000	5.000000	5.000000
Divine Vamp 3	3599.0	5.000000	4.500000	4.750000	4.750000	5.000000
Dark Funk	9139.0	5.000000	5.000000	4.500000	4.500000	5.000000
Red, Wheat, And Blue	1709.0	4.500000	5.000000	5.000000	4.500000	5.000000
Leaf	18189.0	5.000000	4.500000	5.000000	4.500000	5.000000
Cascade Straight Bourbonic	2391.0	4.750000	4.875000	4.750000	4.875000	4.750000
Red Truck Ale	6508.0	5.000000	4.500000	5.000000	5.000000	4.500000
Hops Bandit	7402.0	5.000000	5.000000	5.000000	4.500000	4.500000
Bruno	23476.0	4.500000	5.000000	5.000000	5.000000	4.500000

After the manipulation is complete, we only end up with 3 beers at the top with a total_review rating of 5 across all categories. These are the beers I would recommend trying.

Great Lakes Truth Justice And The American AleEdsten Tripl_Wit and Engelbert Moonbeam

Which of the factors (aroma, taste, appearance, palette) are most important in determining the overall quality of a beer?

In [23]:
```
# create a new dataset than only includes the numerical data so I can run a correla
tion test
corr_test = beers_clean[['review_overall','review_aroma','review_taste','review_app
earance','review_palate']]
```

In [24]:
```
# create a correlation matrix that includes values and coloring for heatmap
corr = corr_test.corr()
corr.style.background_gradient(cmap='coolwarm')
```

Out[24]:

	review_overall	review_aroma	review_taste	review_appearance	review_palate
review_overall	1	0.612669	0.787111	0.498401	0.698925
review_aroma	0.612669	1	0.714677	0.558925	0.614781
review_taste	0.787111	0.714677	1	0.544432	0.73211
review_appearance	0.498401	0.558925	0.544432	1	0.564407
review_palate	0.698925	0.614781	0.73211	0.564407	1

This heatmap shows the relationship of the variables to each other. Reading the top row we gain the knowledge we need. From shades of blue to red, we see all these variables are positively correlate with review_overall. Taste has the highest coefficient at .79. All of these values however are higher than one might typically see.

It will be interesting to see how these interact with each other. I will use R to explore those relationships

Lastly, if I typically enjoy a beer due to its aroma and appearance, which beer style should I try?

In [25]:
```
# create new dataset for beer preference
beers_pref = beers_clean[['brewery_name', 'beer_style', 'beer_name', 'review_aroma'
, 'review_appearance', 'review_overall']]
```

In [26]:
```
#view new datasheet to cofirm accuracy
beers_pref.head()
```

Out[26]:

	brewery_name	beer_style	beer_name	review_aroma	review_appearance	review_overall
0	Vecchio Birraio	Hefeweizen	Sausa Weizen	2.0	2.5	1.5
1	Vecchio Birraio	English Strong Ale	Red Moon	2.5	3.0	3.0
2	Vecchio Birraio	Foreign / Export Stout	Black Horse Black Beer	2.5	3.0	3.0
3	Vecchio Birraio	German Pilsener	Sausa Pils	3.0	3.5	3.0
4	Caldera Brewing Company	American Double / Imperial IPA	Cauldron DIPA	4.5	4.0	4.0

In [27]:
```
# sort new dataset descending by aroma, appearance then review_overall
beers_pref1 = beers_pref.groupby('beer_style').mean()
```

In [28]:
```
#view new datasheet to cofirm accuracy
beers_pref1.head()
```

Out[28]:

	review_aroma	review_appearance	review_overall
beer_style			
Altbier	3.635412	3.815662	3.832017
American Adjunct Lager	2.478577	2.785754	3.010280
American Amber / Red Ale	3.653032	3.829129	3.802779
American Amber / Red Lager	3.220063	3.533167	3.577330
American Barleywine	4.022201	4.040175	3.898819

In [30]:
```
#sort datasheet to findest beer with the higest values
beers_pref1.sort_values(['review_overall','review_aroma', 'review_appearance'], asc
ending=False)
```

Out[30]:

	review_aroma	review_appearance	review_overall
beer_style			
American Wild Ale	4.134354	4.010932	4.100018
Gueuze	4.116157	4.037312	4.087034
Quadrupel (Quad)	4.133515	4.119829	4.073141
Lambic - Unblended	4.126564	3.918191	4.060635
American Double / Imperial Stout	4.161199	4.164013	4.030252
Russian Imperial Stout	4.077615	4.212620	4.024439
Weizenbock	4.049476	4.013302	4.011139
American Double / Imperial IPA	4.099739	4.080414	3.999935
Flanders Red Ale	4.045718	4.003505	3.995733
Rye Beer	3.907953	4.013913	3.988838
Keller Bier / Zwickel Bier	3.689478	3.848335	3.987670
Eisbock	4.157348	3.962977	3.975444
American IPA	3.901598	3.973841	3.971846
Saison / Farmhouse Ale	3.935383	4.002434	3.965143
Belgian IPA	3.984351	4.081147	3.963471
Gose	3.780581	3.905963	3.961774
Baltic Porter	3.947894	4.040809	3.957308
Hefeweizen	3.782597	3.851052	3.953560
Oatmeal Stout	3.863330	4.020571	3.951972
Roggenbier	3.853165	3.834177	3.950633
American Black Ale	3.933901	4.115964	3.934753
Dubbel	3.906931	4.008503	3.928987
English Porter	3.844733	3.937301	3.921067
Tripel	3.916956	3.956684	3.917723
Belgian Strong Dark Ale	3.972578	4.010457	3.914133
American Porter	3.847471	3.965436	3.908985
Flanders Oud Bruin	3.939616	3.902387	3.904325
Old Ale	4.008995	3.942790	3.899435
American Barleywine	4.022201	4.040175	3.898819
Belgian Strong Pale Ale	3.912220	3.962177	3.895648
...	...	...	...
Sahti	3.837151	3.652390	3.697211
Irish Red Ale	3.425632	3.770677	3.690294
English Pale Ale	3.423185	3.698856	3.689955
Scottish Gruit / Ancient Herbed Ale	3.742946	3.646940	3.689813
American Dark Wheat Ale	3.523451	3.680486	3.668342

R code is copied and paste below:

````
---
title: "Untitled"
author: "Whitmire"
date: "March 15, 2019"
output: html_document
editor_options:
  chunk_output_type: console
---

```{r echo=FALSE, load_packages}
#load packages need for analysis
library(ggplot2)
library(GGally)
library(scales)
library(memisc)
library(lattice)
library(MASS)
library(car)
library(reshape)
library(plyr)
library(dplyr)
```
---
title: "Untitled"
author: "Whitmire"
date: "March 15, 2019"
output: html_document
editor_options:
  chunk_output_type: console
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

```{r echo=FALSE, unpack_file}
# Need to untar the file to extract the data for working
untar("beer_reviews.tar.gz",list=TRUE)
untar("beer_reviews.tar")
```

```{r echo=FALSE, Load_the_Data}
# read the file into the directory and add columns as needed
beers <- read.csv("beer_reviews.csv")
beers$total_review <- (beers$review_overall + beers$review_aroma + beers$review_appearance + beers
$review_palate + beers$review_taste)/5

#create new dataset to work with
beers_clean <- beers[ -c(3) ]
```{r}
ggplot(aes(x = review_taste, y =review_palate, color = review_overall), data = beers) +
  geom_point(alpha = .2, position = 'jitter') +
      ggtitle('Palate/Taste ~ Overall Rating')
```

```{r}
ggplot(aes(x = review_taste, y =review_aroma, color = review_overall), data = beers) +
  geom_point(alpha = .2, position = 'jitter') +
      ggtitle('Aroma/Taste ~ Overall Rating')
````