Final Project Chocolate Rating

By Anh Huynh

Table of Contents

1
- Overview
- Goals

2
- DATA description
- DATA Cleaning & EDA

3
- Database & MySQL
- Insights

4
- Deploy Machine Learning

Overview

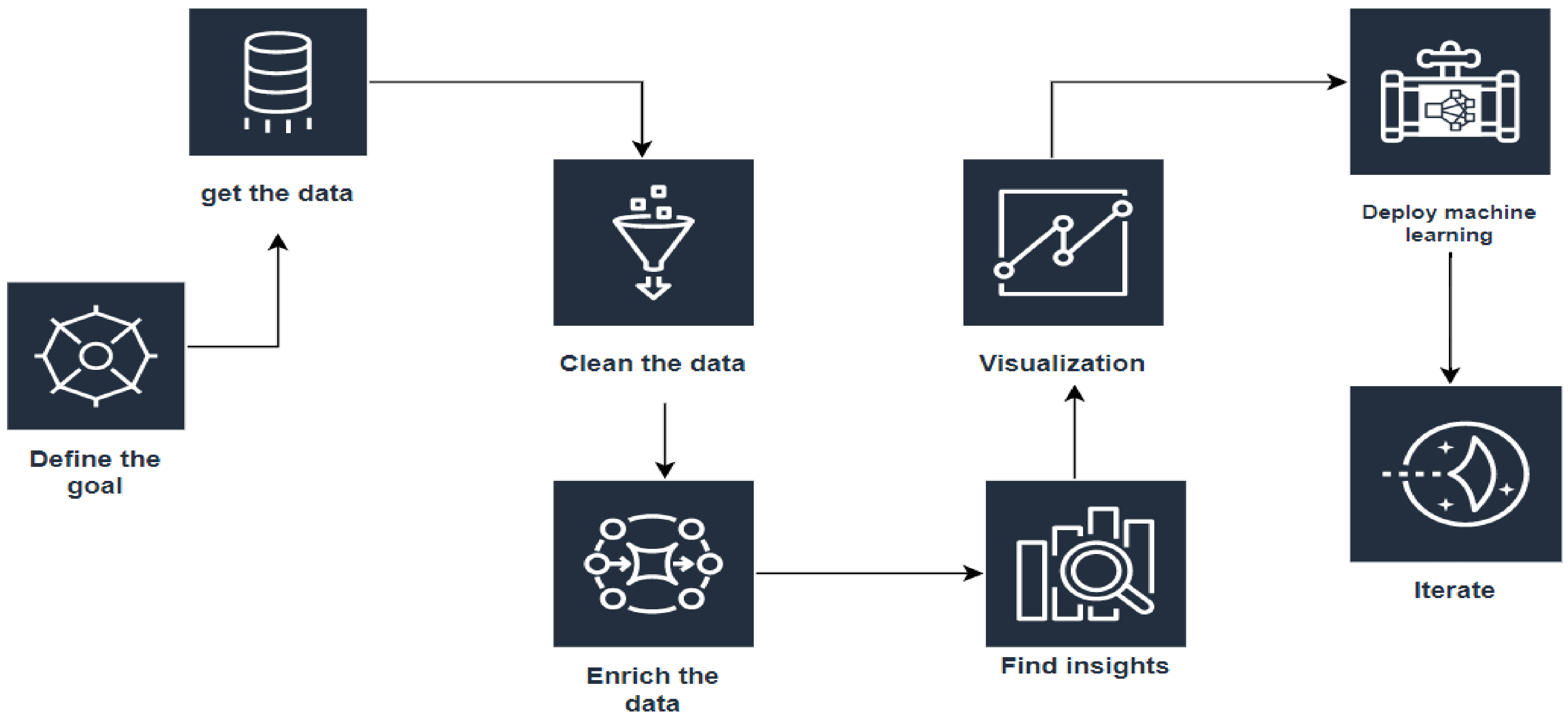
- The cacao tree was cultivated more than 3000 years ago by the Maya, Toltec and Aztec
- They use cocoa bean to prepare a beverage as a ceremonial and as a currency.
- The Maya considered chocolate to be the food of gods, held the cacao tree to be sacred for after life
- Today chocolate industry estimated to be an USD 130.56 billion-dollar industry

CHOCOLATE RATING

Project Main goals

focused on ***Rating*** on different chocolate bars with an aim of different **categories**

Main dataset ‘Chocolate rating	**2530 rows,10 columns**
Chocolate flavor	**1795 rows × 9 columns**
Countries and continents	**249 rows × 9 columns**



Define the goal

get the data

Clean the data

Enrich the data

Find insights

Visualization

Deploy machine learning

Iterate

Data Descriptions

Rating

- 3.5 - 4 = Outstanding
- 3.0 - 3.49 = Recommended
- 2.0 - 2.9 = Disappointing
- 1.0 - 1.9 = Unpleasant

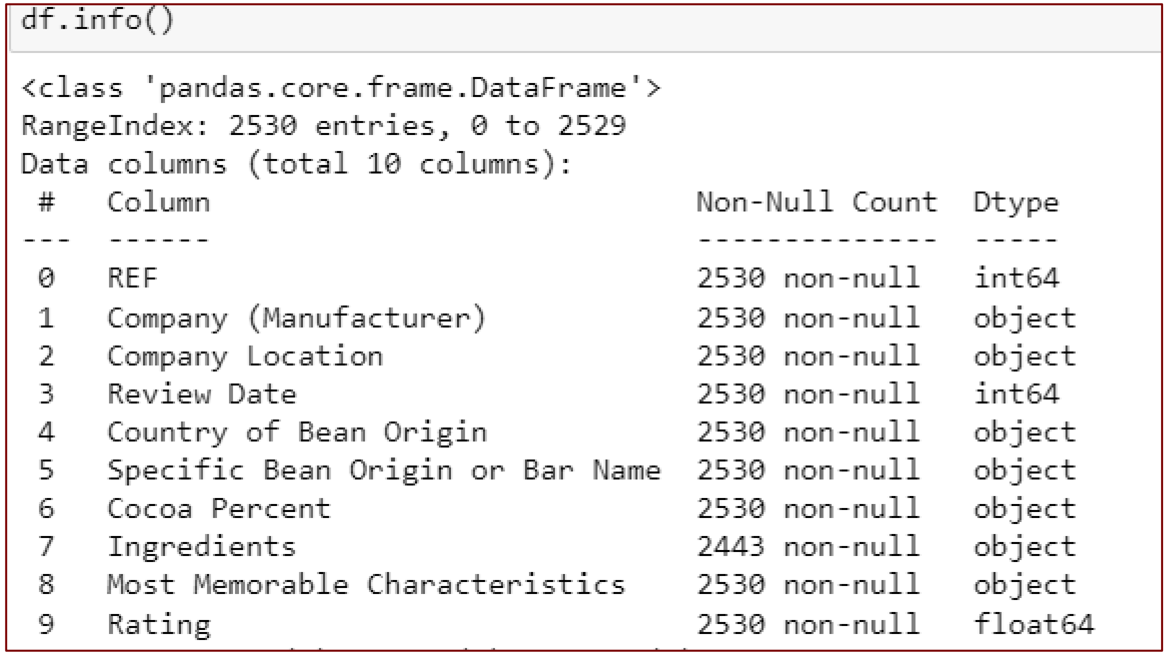
Ingredients

- B (Beans)
- S (Sugar)
- Sa (Salt)
- (L) Lecithin
- S* (Beet sugar)
- C (Cocoa Butter)
- V (Vanilla)

Data Descriptions

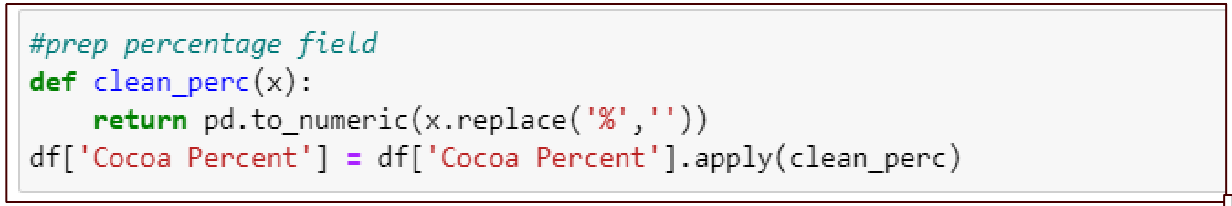
- Review day
- Companies (manufacturer)
- Company location
- Country of bean origin
- Cocoa percent
- Memorable Characteristics

Data cleaning (Main data)

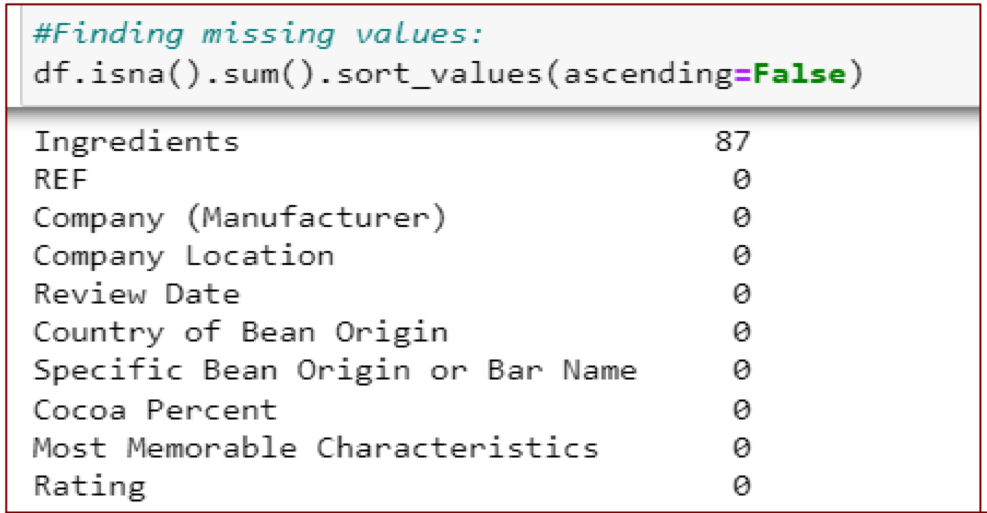


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2530 entries, 0 to 2529
Data columns (total 10 columns):
 #   Column                             Non-Null Count  Dtype  
---  ------                             --------------  -----  
 0   REF                                2530 non-null   int64  
 1   Company (Manufacturer)             2530 non-null   object 
 2   Company Location                   2530 non-null   object 
 3   Review Date                        2530 non-null   int64  
 4   Country of Bean Origin             2530 non-null   object 
 5   Specific Bean Origin or Bar Name   2530 non-null   object 
 6   Cocoa Percent                      2530 non-null   object 
 7   Ingredients                        2443 non-null   object 
 8   Most Memorable Characteristics     2530 non-null   object 
 9   Rating                             2530 non-null   float64
```

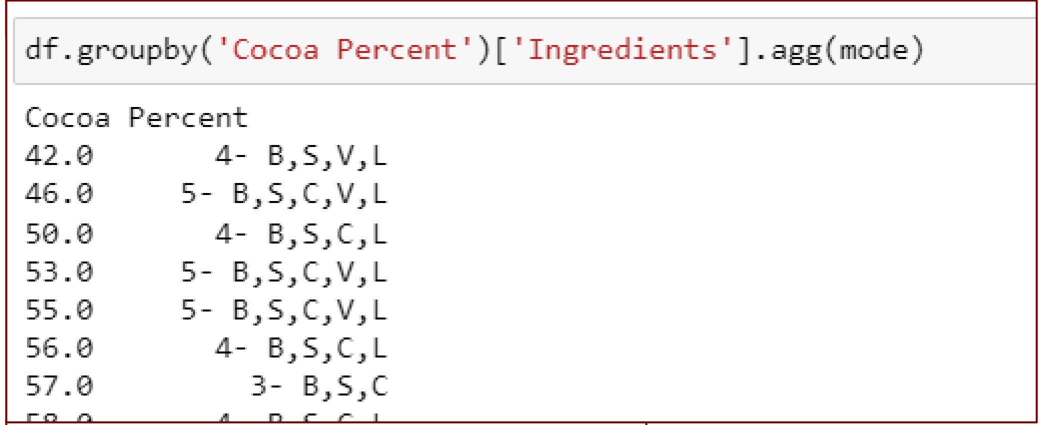


```
#prep percentage field
def clean_perc(x):
    return pd.to_numeric(x.replace('%',''))
df['Cocoa Percent'] = df['Cocoa Percent'].apply(clean_perc)
```



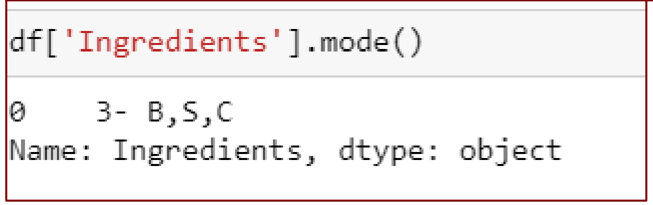
```
#Finding missing values:
df.isna().sum().sort_values(ascending=False)
```

```
Ingredients                         87
REF                                  0
Company (Manufacturer)               0
Company Location                     0
Review Date                          0
Country of Bean Origin               0
Specific Bean Origin or Bar Name     0
Cocoa Percent                        0
Most Memorable Characteristics       0
Rating                               0
```



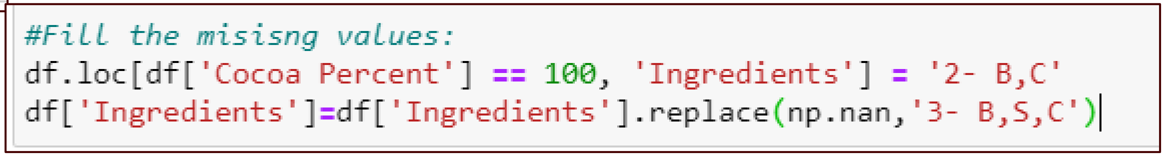
```
df.groupby('Cocoa Percent')['Ingredients'].agg(mode)
```

```
Cocoa Percent
42.0      4- B,S,V,L
46.0    5- B,S,C,V,L
50.0      4- B,S,C,L
53.0    5- B,S,C,V,L
55.0    5- B,S,C,V,L
56.0      4- B,S,C,L
57.0        3- B,S,C
58.0      4- B,S,C,L
```



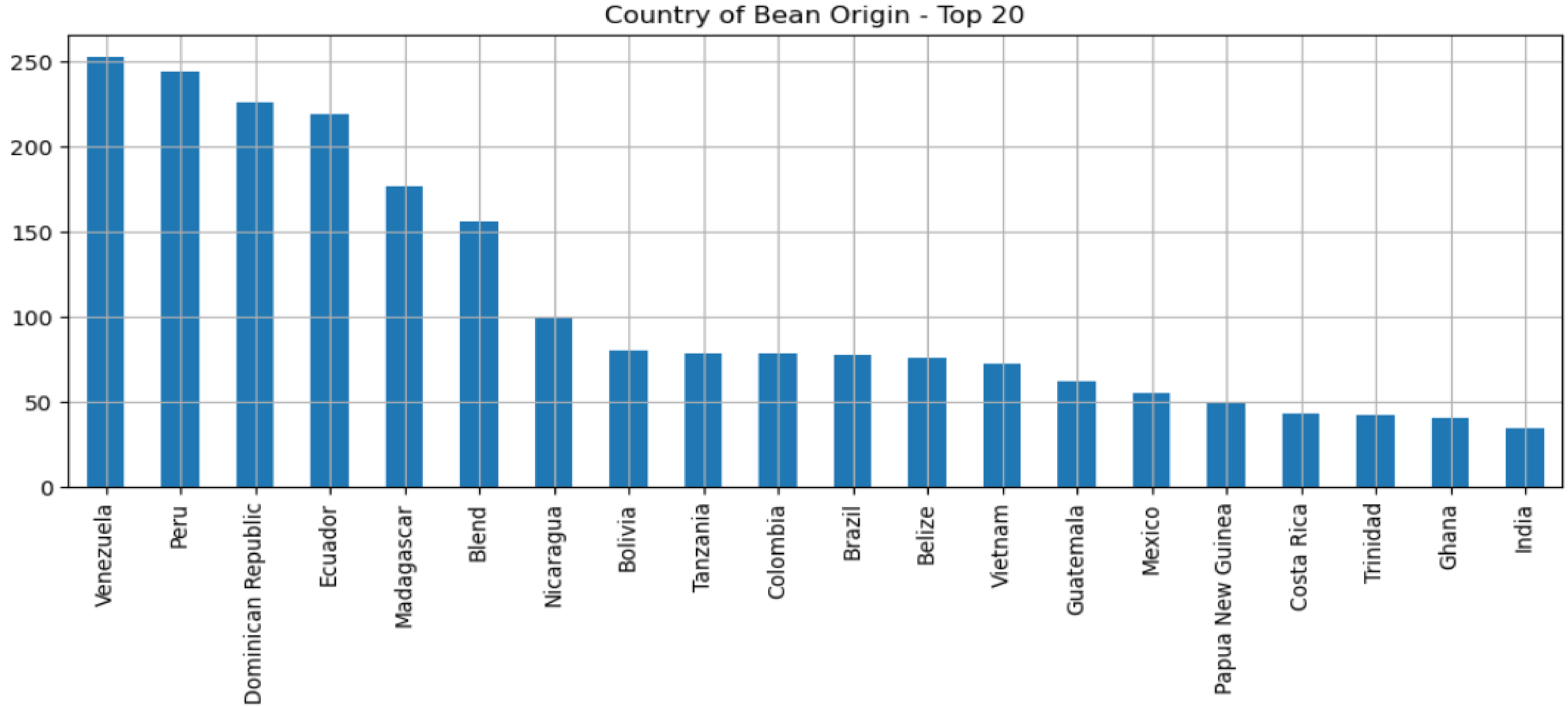
```
df['Ingredients'].mode()
```

```
0    3- B,S,C
Name: Ingredients, dtype: object
```

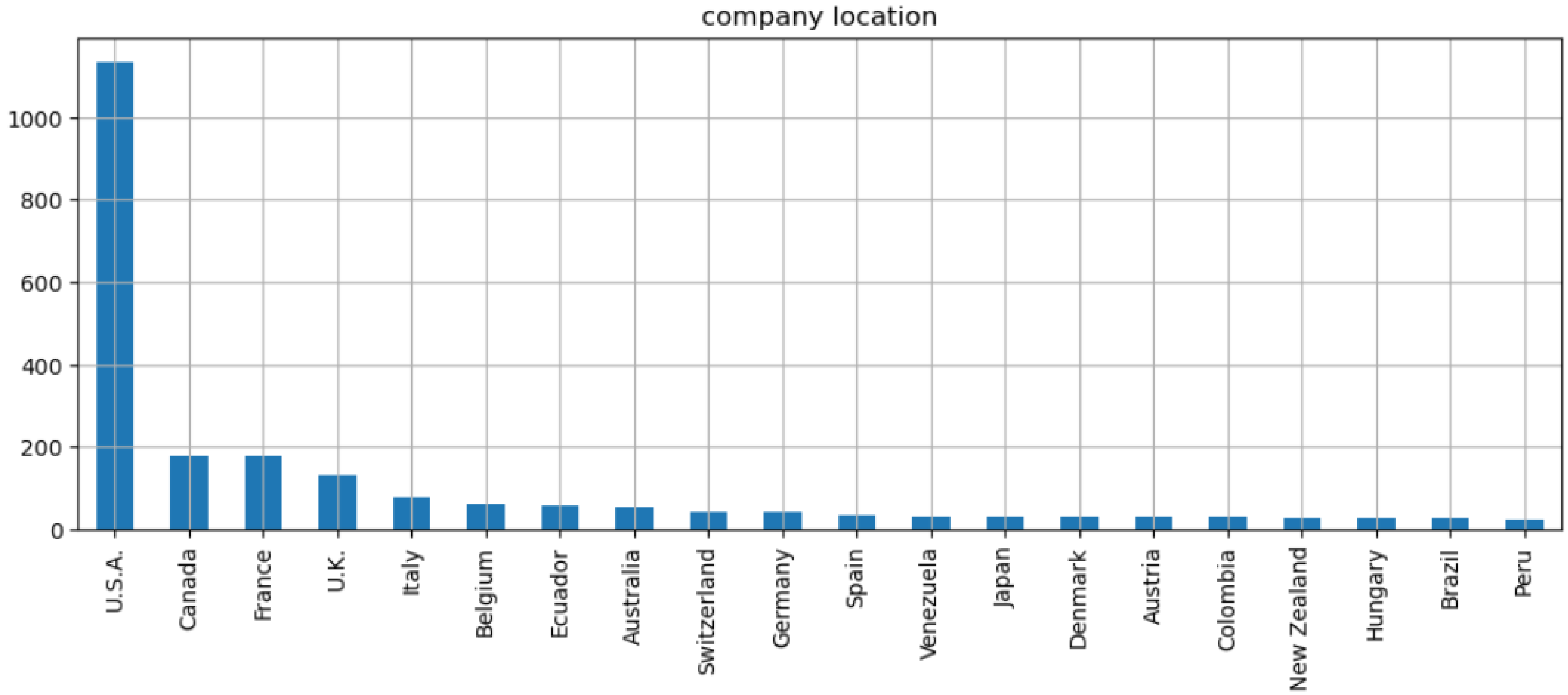


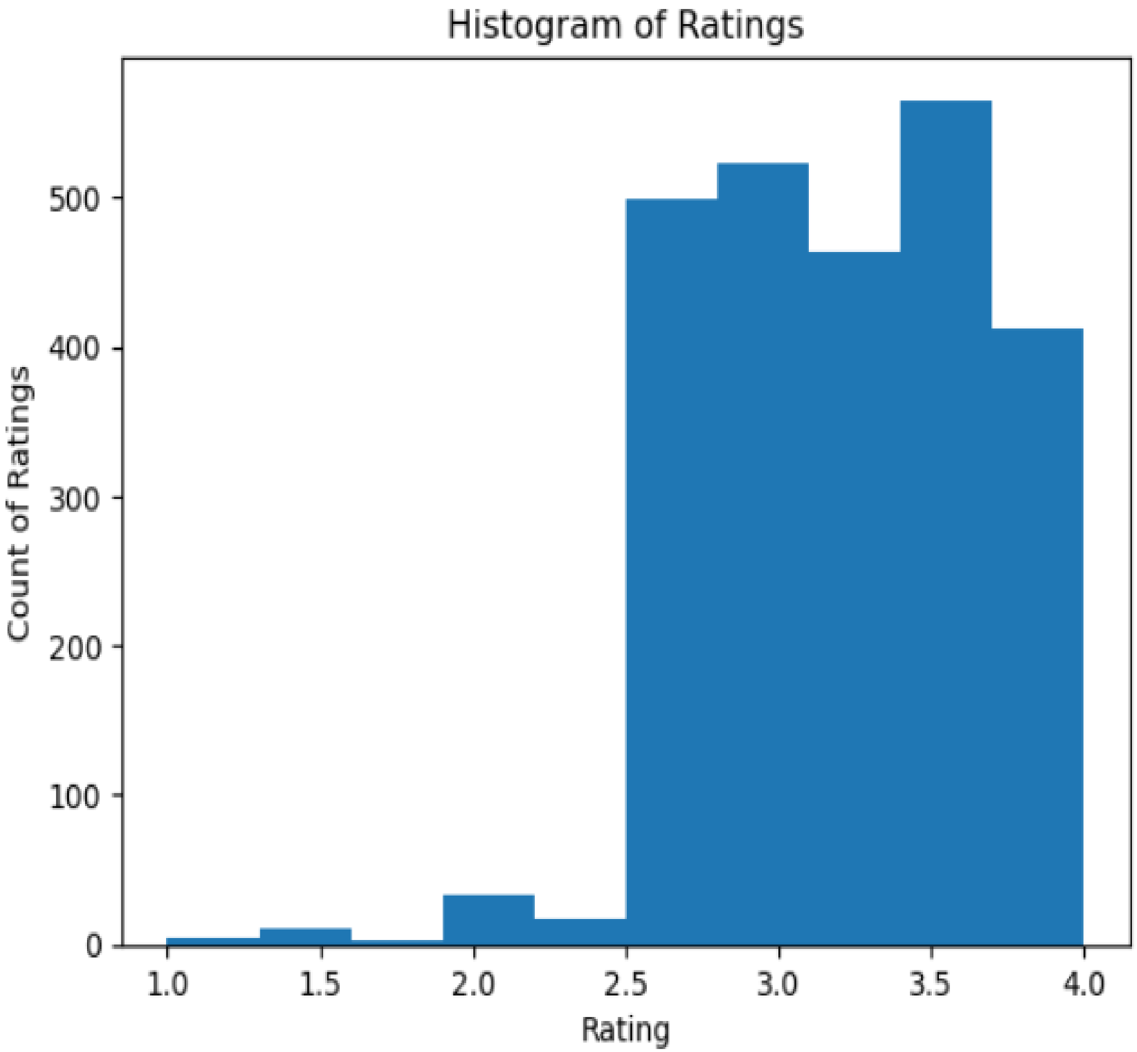
```
#Fill the misisng values:
df.loc[df['Cocoa Percent'] == 100, 'Ingredients'] = '2- B,C'
df['Ingredients']=df['Ingredients'].replace(np.nan,'3- B,S,C')
```

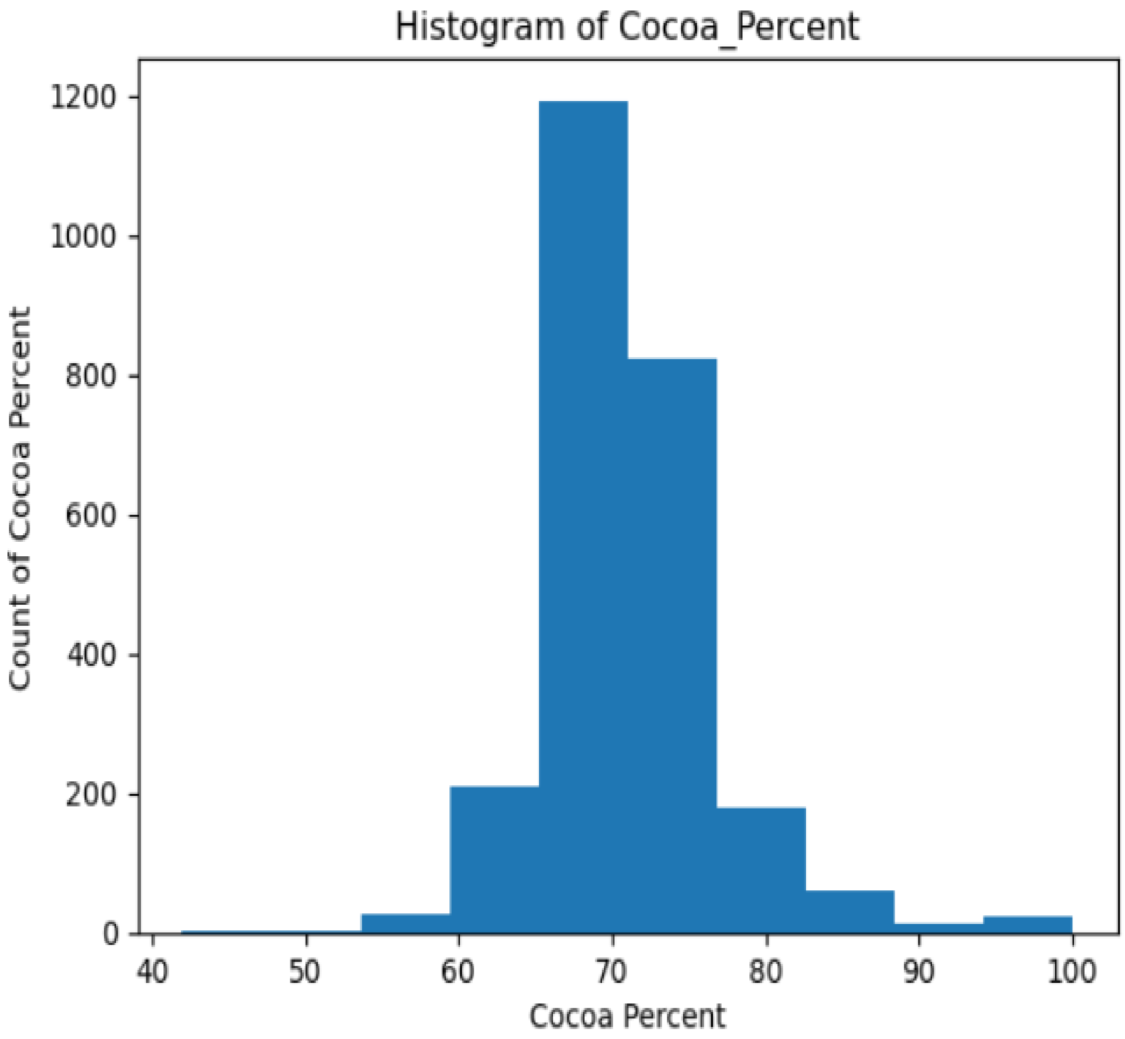
EDA


Country of Bean Origin - Top 20
0
50
100
150
200
250
Venezuela
Peru
Dominican Republic
Ecuador
Madagascar
Blend
Nicaragua
Bolivia
Tanzania
Colombia
Brazil
Belize
Vietnam
Guatemala
Mexico
Papua New Guinea
Costa Rica
Trinidad
Ghana
India

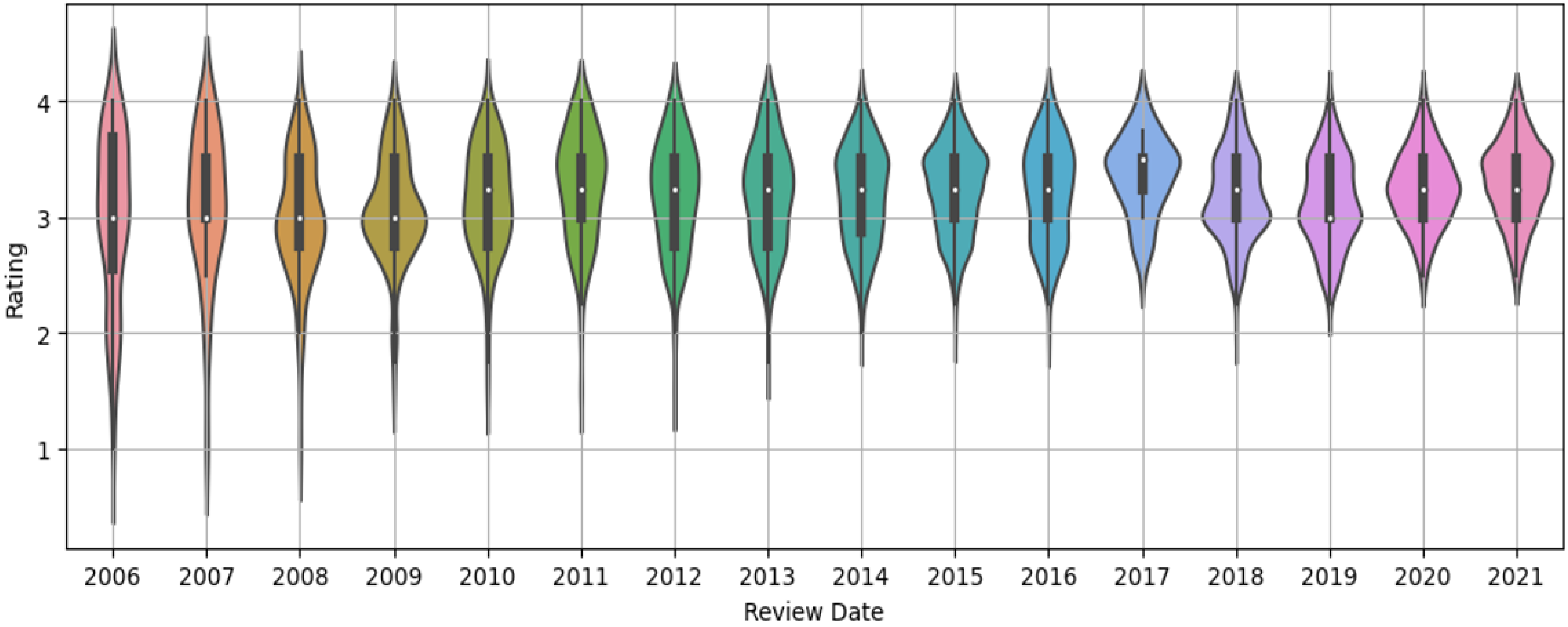
Company location



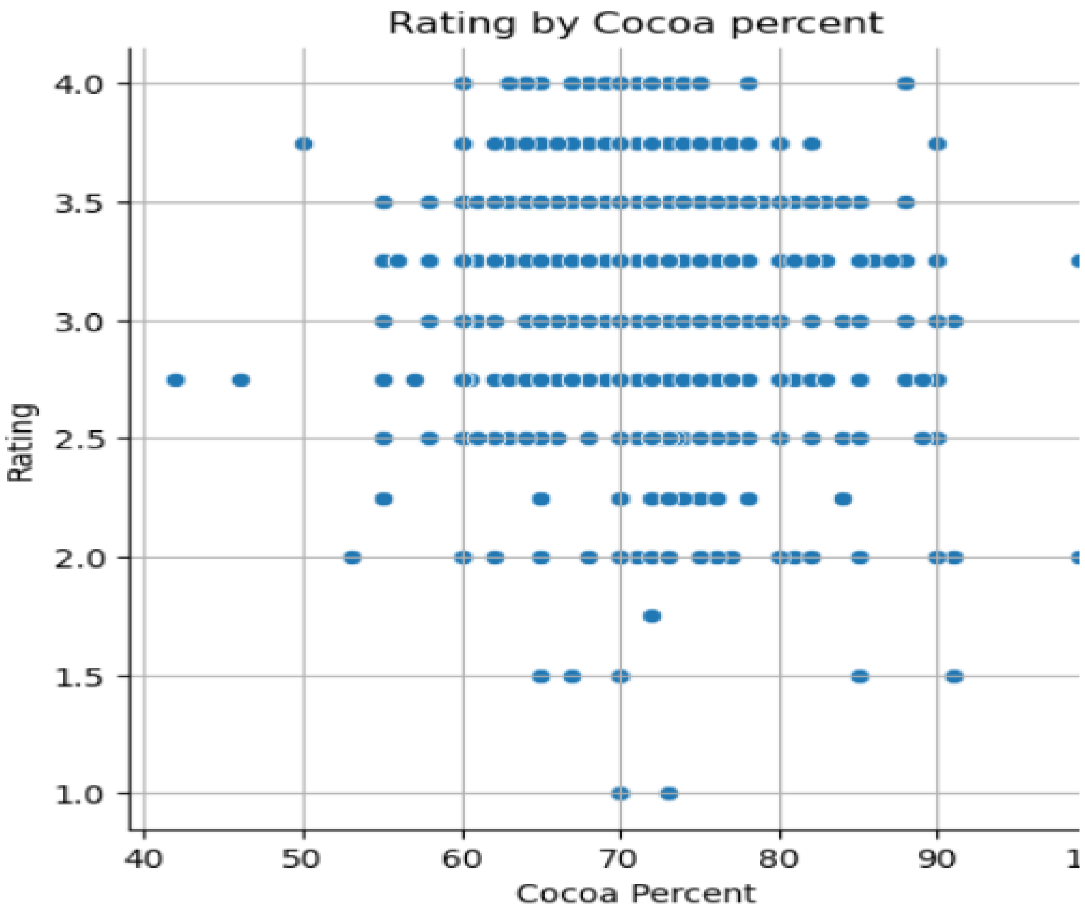

Histogram of Ratings
Count of Ratings
0
100
200
300
400
500
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Rating

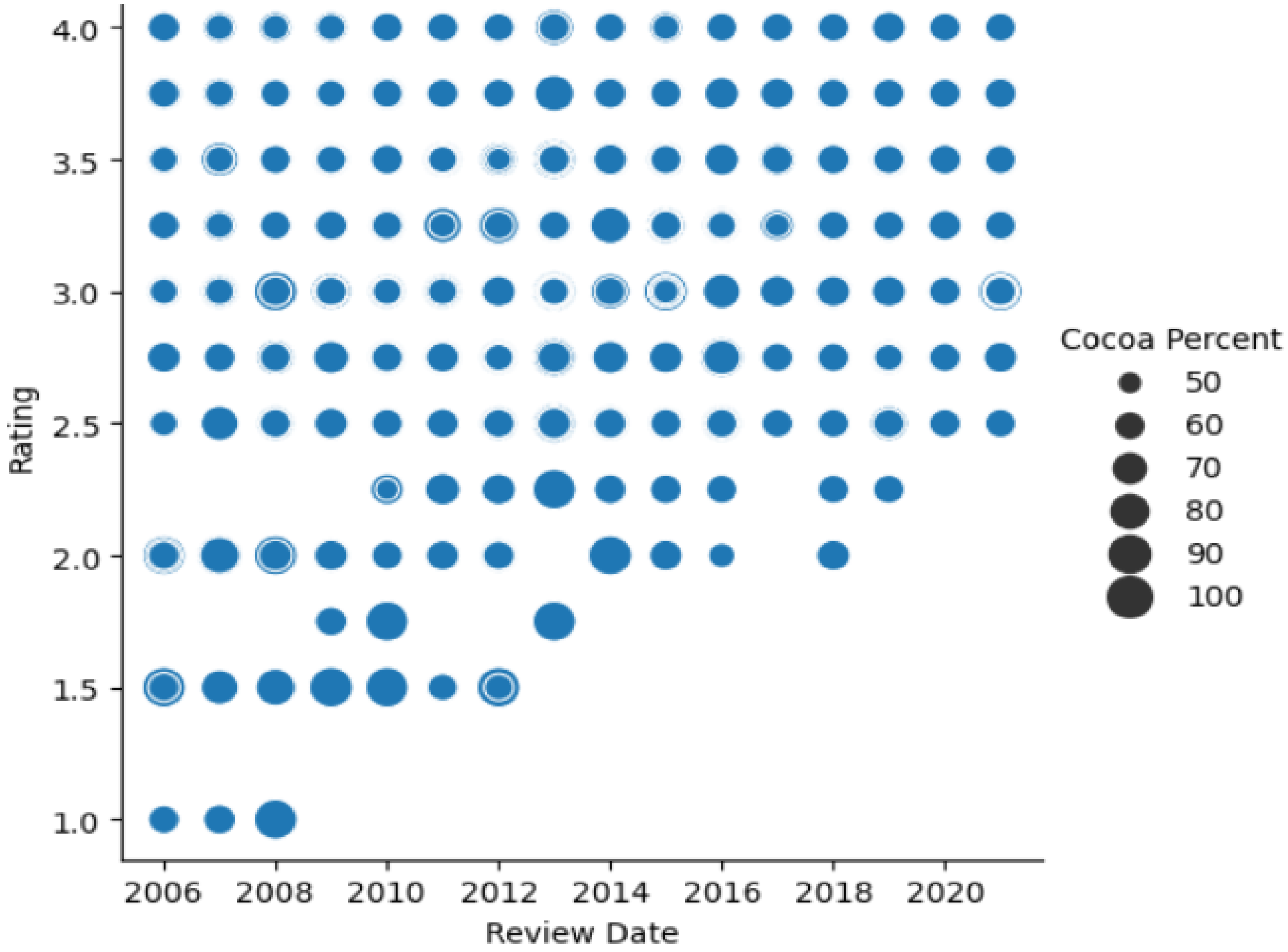

Histogram of Cocoa_Percent
Count of Cocoa Percent
0
200
400
600
800
1000
1200
40
50
60
70
80
90
100
Cocoa Percent

Rating by Year

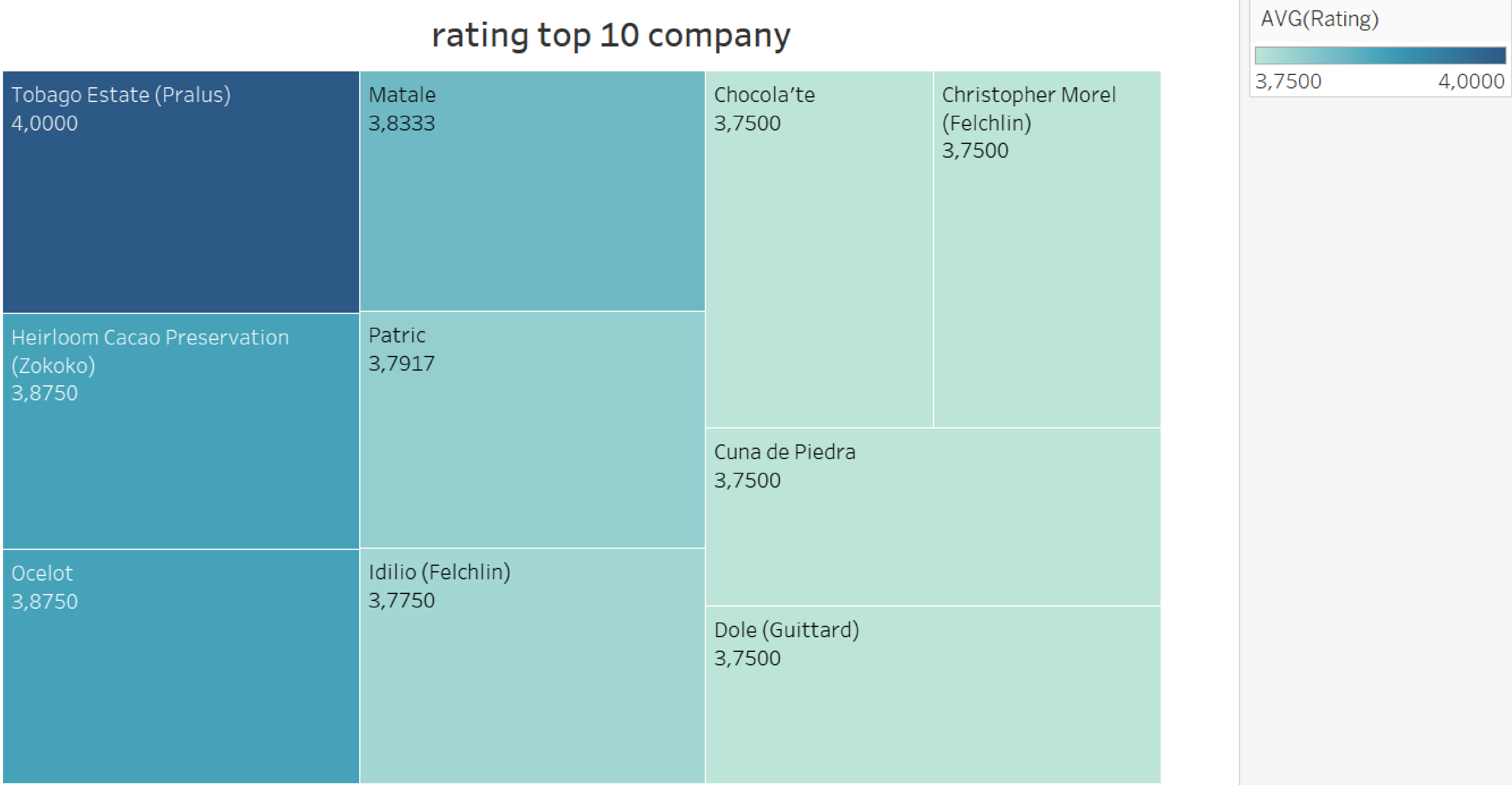


Rating by cocoa percent

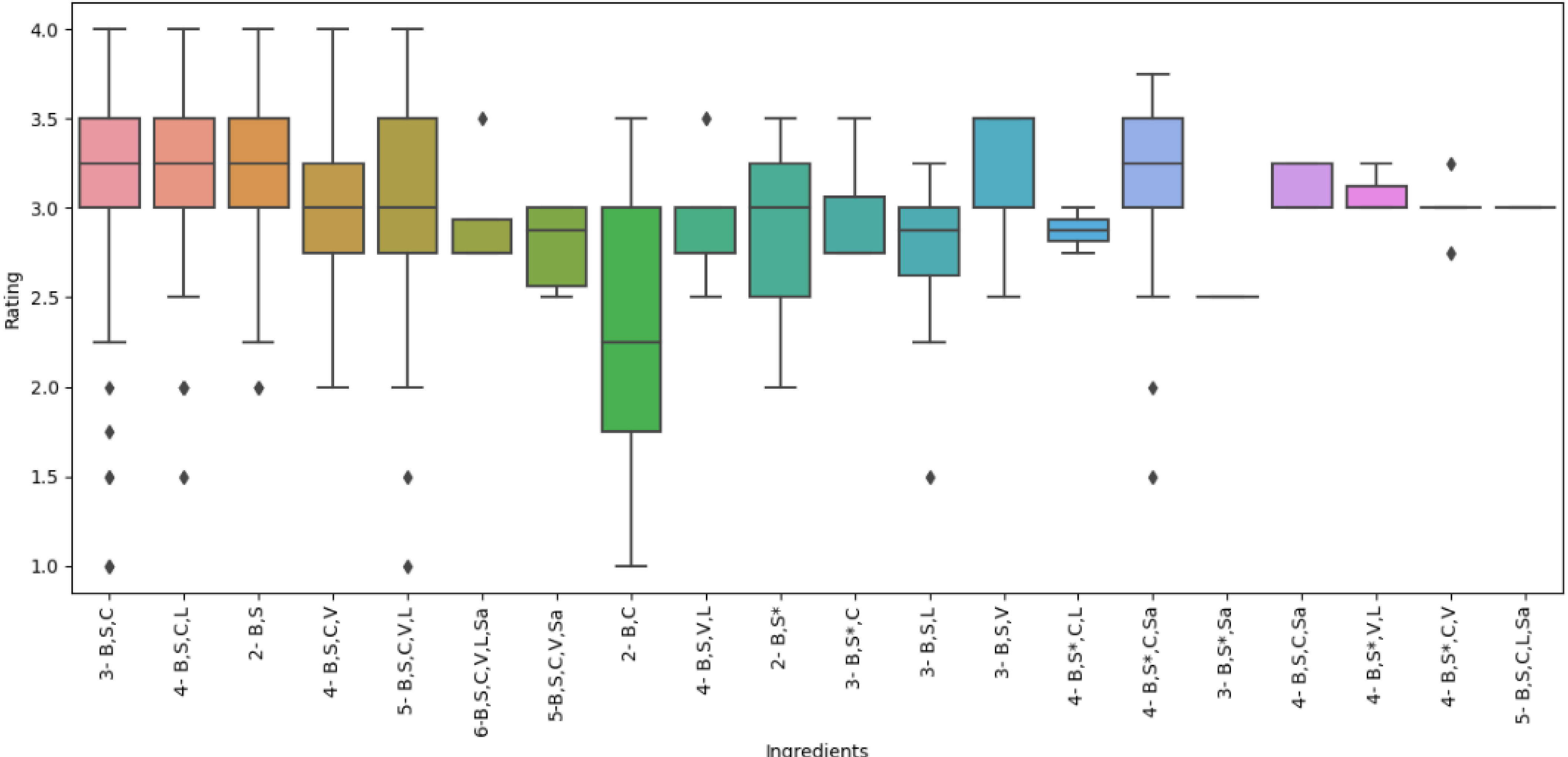




rating top 10 company

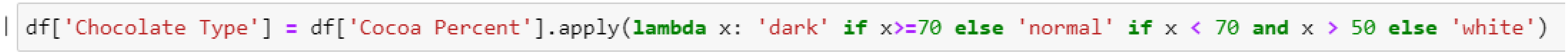


Rating by ingredients

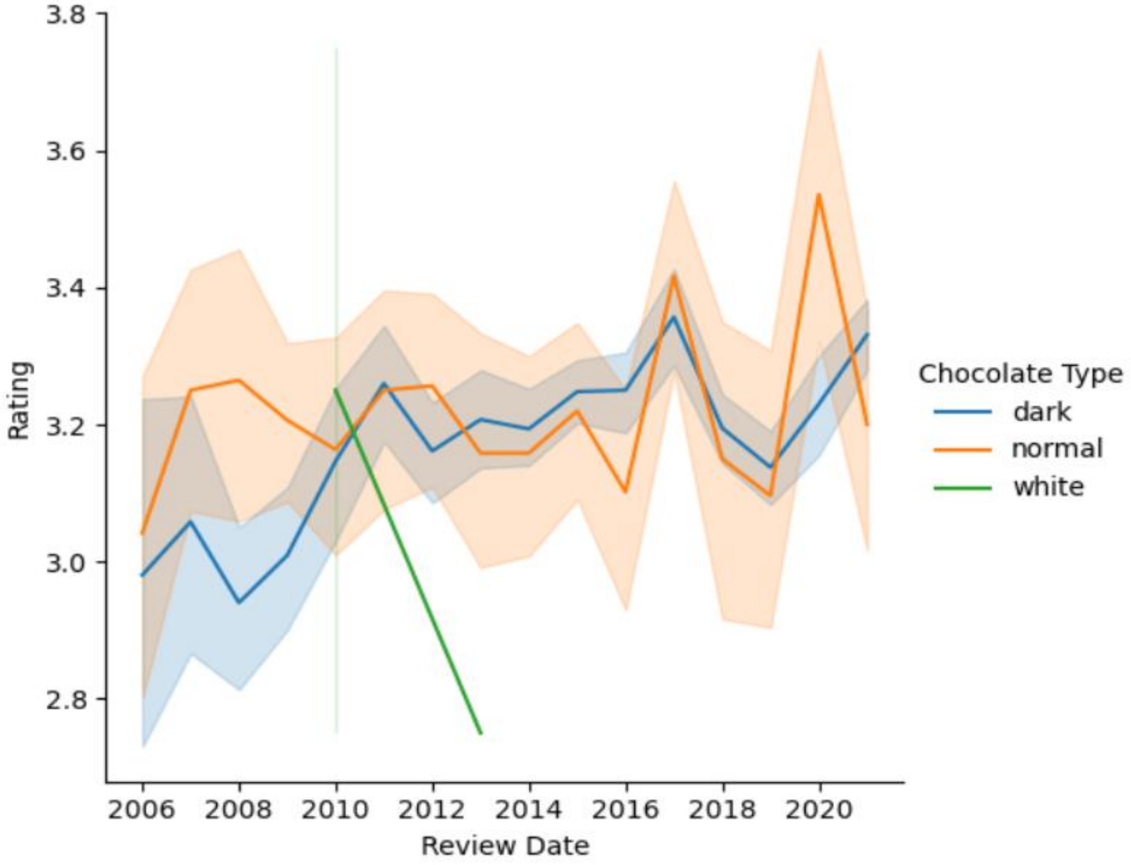


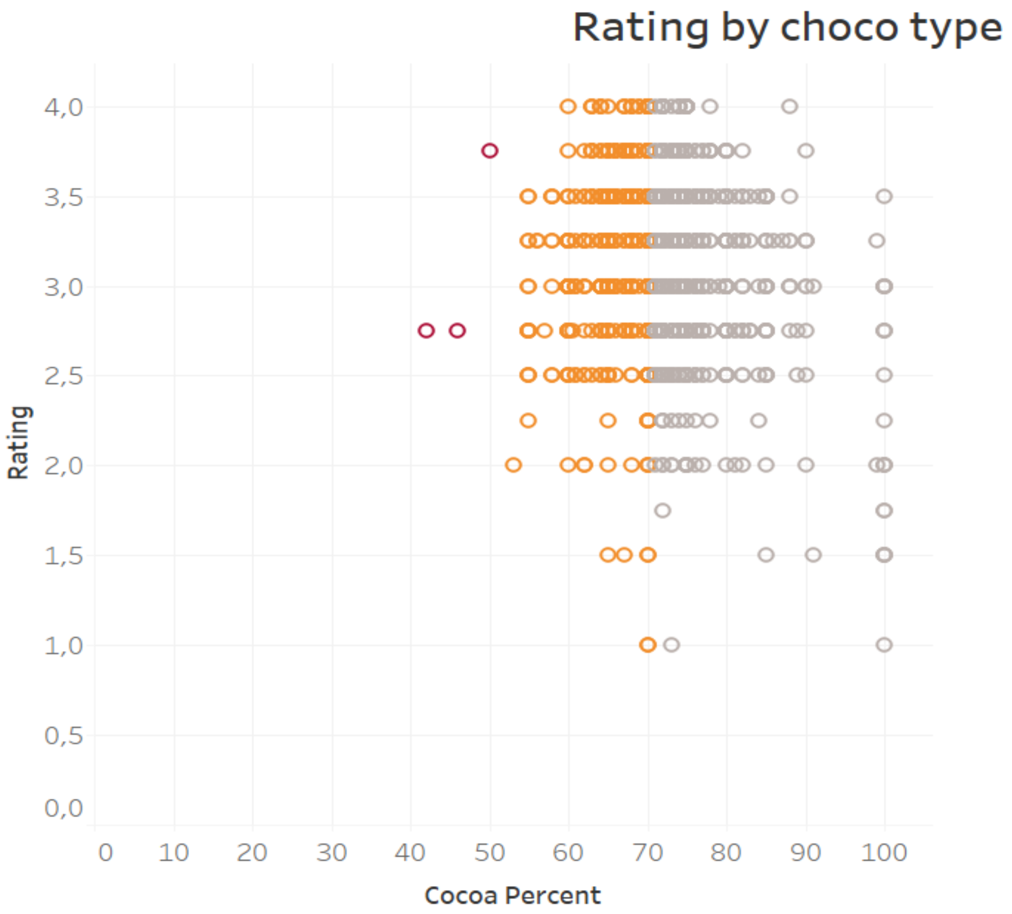
Further exploratory data analysis

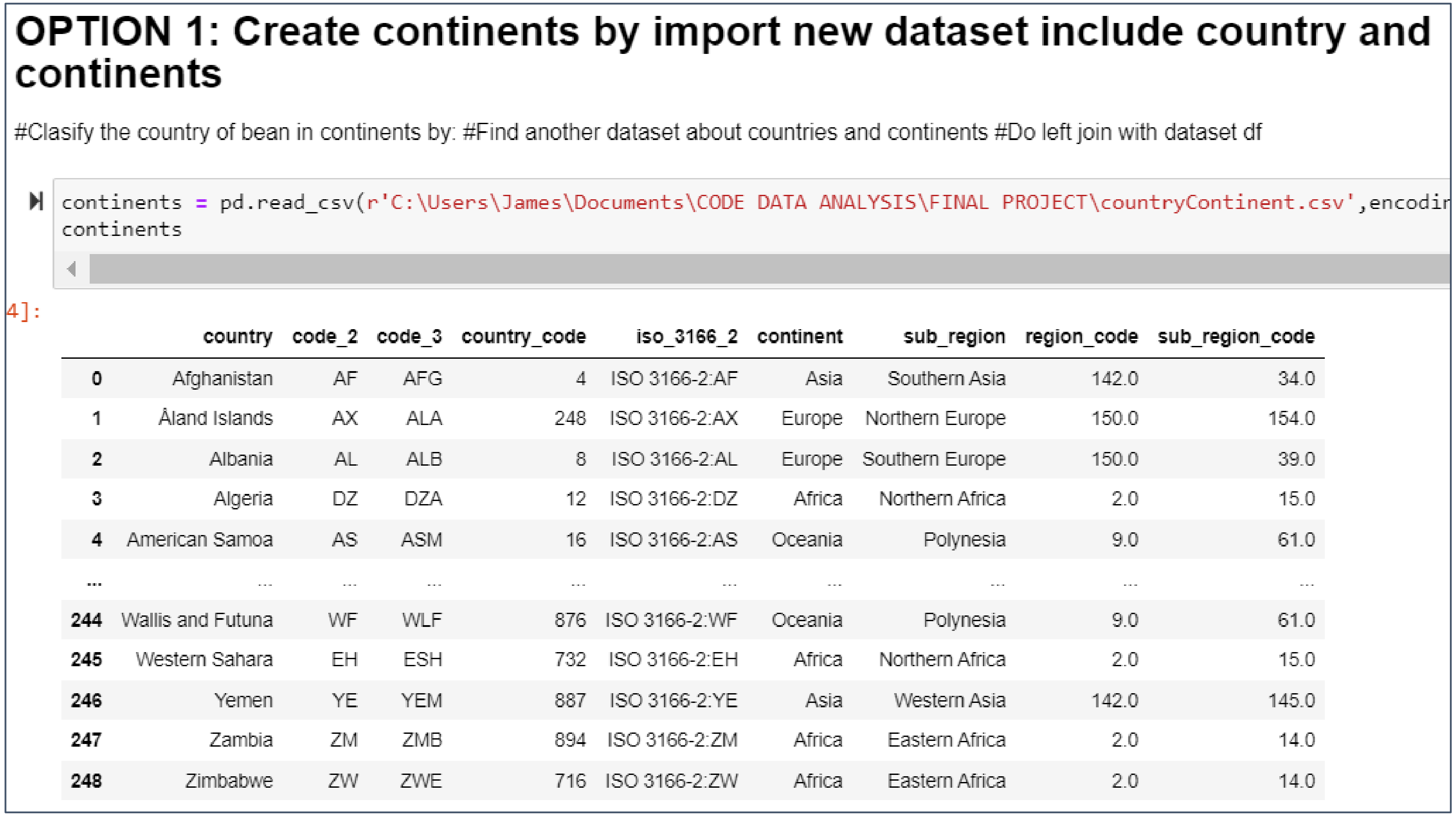
Rating by chocolate type



```
df['Chocolate Type'] = df['Cocoa Percent'].apply(lambda x: 'dark' if x>=70 else 'normal' if x < 70 and x > 50 else 'white')
```







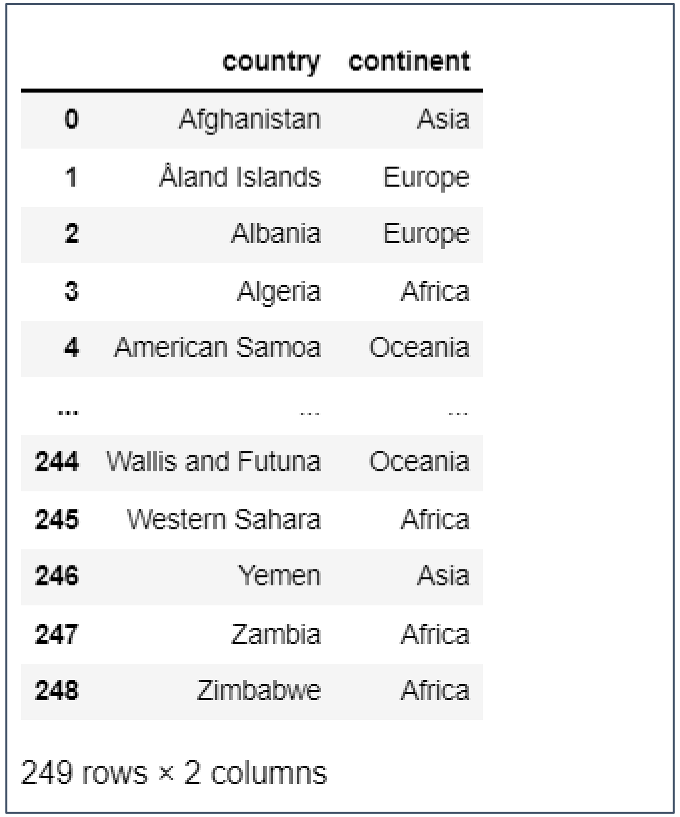
OPTION 1: Create continents by import new dataset include country and continents

#Clasify the country of bean in continents by: #Find another dataset about countries and continents #Do left join with dataset df

```
continents = pd.read_csv(r'C:\Users\James\Documents\CODE DATA ANALYSIS\FINAL PROJECT\countryContinent.csv',encodir
continents
```

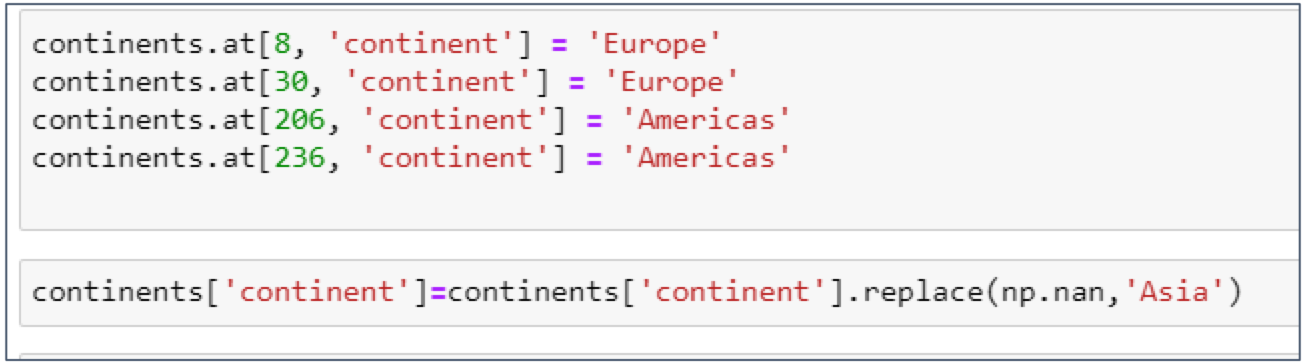
4]:

	country	code_2	code_3	country_code	iso_3166_2	continent	sub_region	region_code	sub_region_code
0	Afghanistan	AF	AFG	4	ISO 3166-2:AF	Asia	Southern Asia	142.0	34.0
1	Åland Islands	AX	ALA	248	ISO 3166-2:AX	Europe	Northern Europe	150.0	154.0
2	Albania	AL	ALB	8	ISO 3166-2:AL	Europe	Southern Europe	150.0	39.0
3	Algeria	DZ	DZA	12	ISO 3166-2:DZ	Africa	Northern Africa	2.0	15.0
4	American Samoa	AS	ASM	16	ISO 3166-2:AS	Oceania	Polynesia	9.0	61.0
...	...	...	...	...	...	...	...	...	...
244	Wallis and Futuna	WF	WLF	876	ISO 3166-2:WF	Oceania	Polynesia	9.0	61.0
245	Western Sahara	EH	ESH	732	ISO 3166-2:EH	Africa	Northern Africa	2.0	15.0
246	Yemen	YE	YEM	887	ISO 3166-2:YE	Asia	Western Asia	142.0	145.0
247	Zambia	ZM	ZMB	894	ISO 3166-2:ZM	Africa	Eastern Africa	2.0	14.0
248	Zimbabwe	ZW	ZWE	716	ISO 3166-2:ZW	Africa	Eastern Africa	2.0	14.0



	country	continent
0	Afghanistan	Asia
1	Åland Islands	Europe
2	Albania	Europe
3	Algeria	Africa
4	American Samoa	Oceania
...	...	...
244	Wallis and Futuna	Oceania
245	Western Sahara	Africa
246	Yemen	Asia
247	Zambia	Africa
248	Zimbabwe	Africa

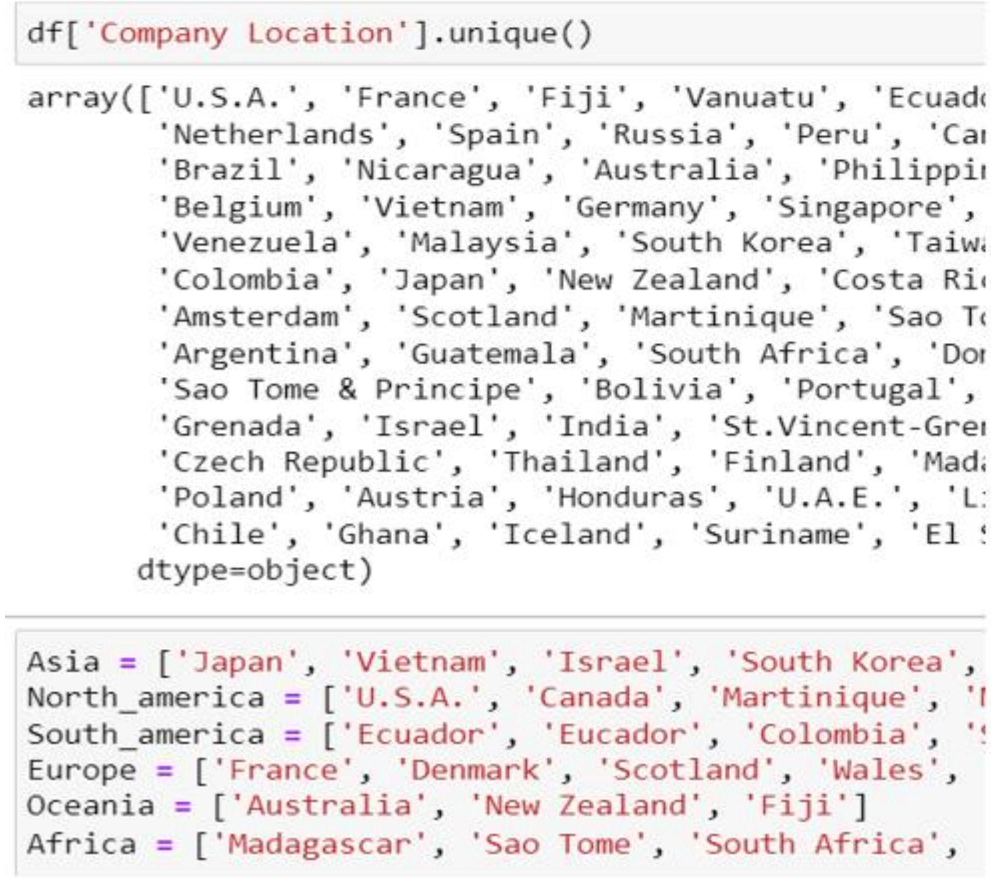
249 rows × 2 columns



```
continents.at[8, 'continent'] = 'Europe'
continents.at[30, 'continent'] = 'Europe'
continents.at[206, 'continent'] = 'Americas'
continents.at[236, 'continent'] = 'Americas'
```

```
continents['continent']=continents['continent'].replace(np.nan,'Asia')
```

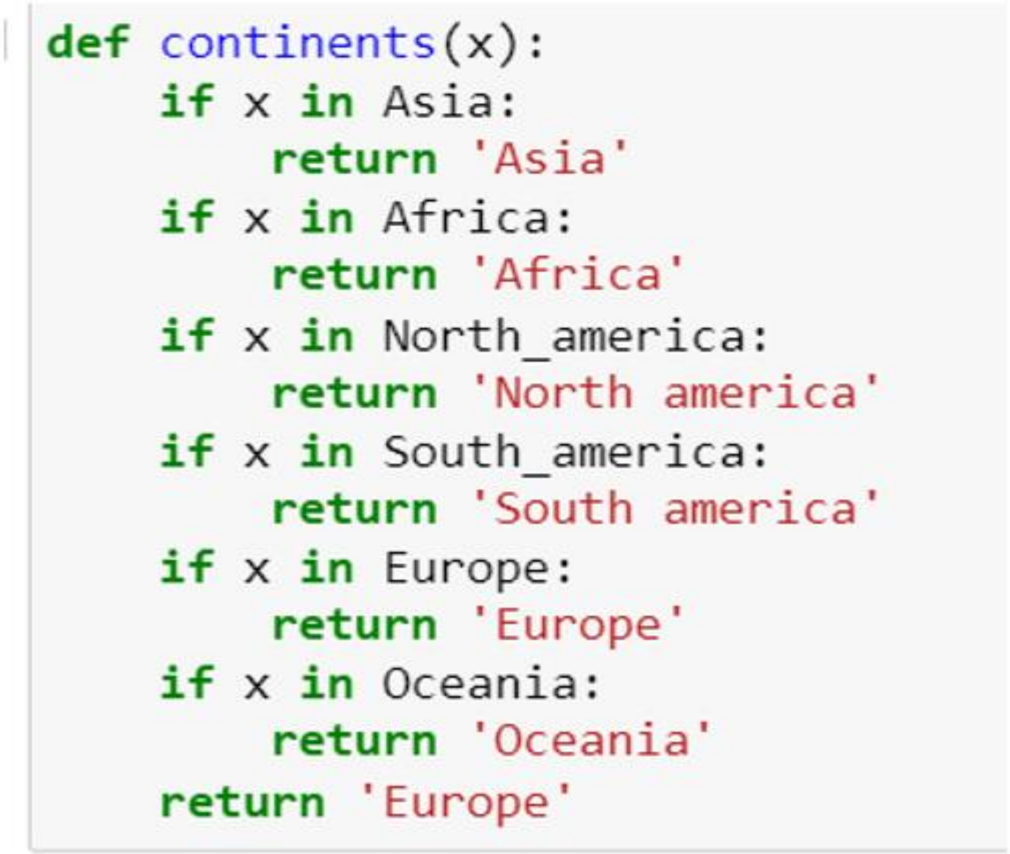
Option 2



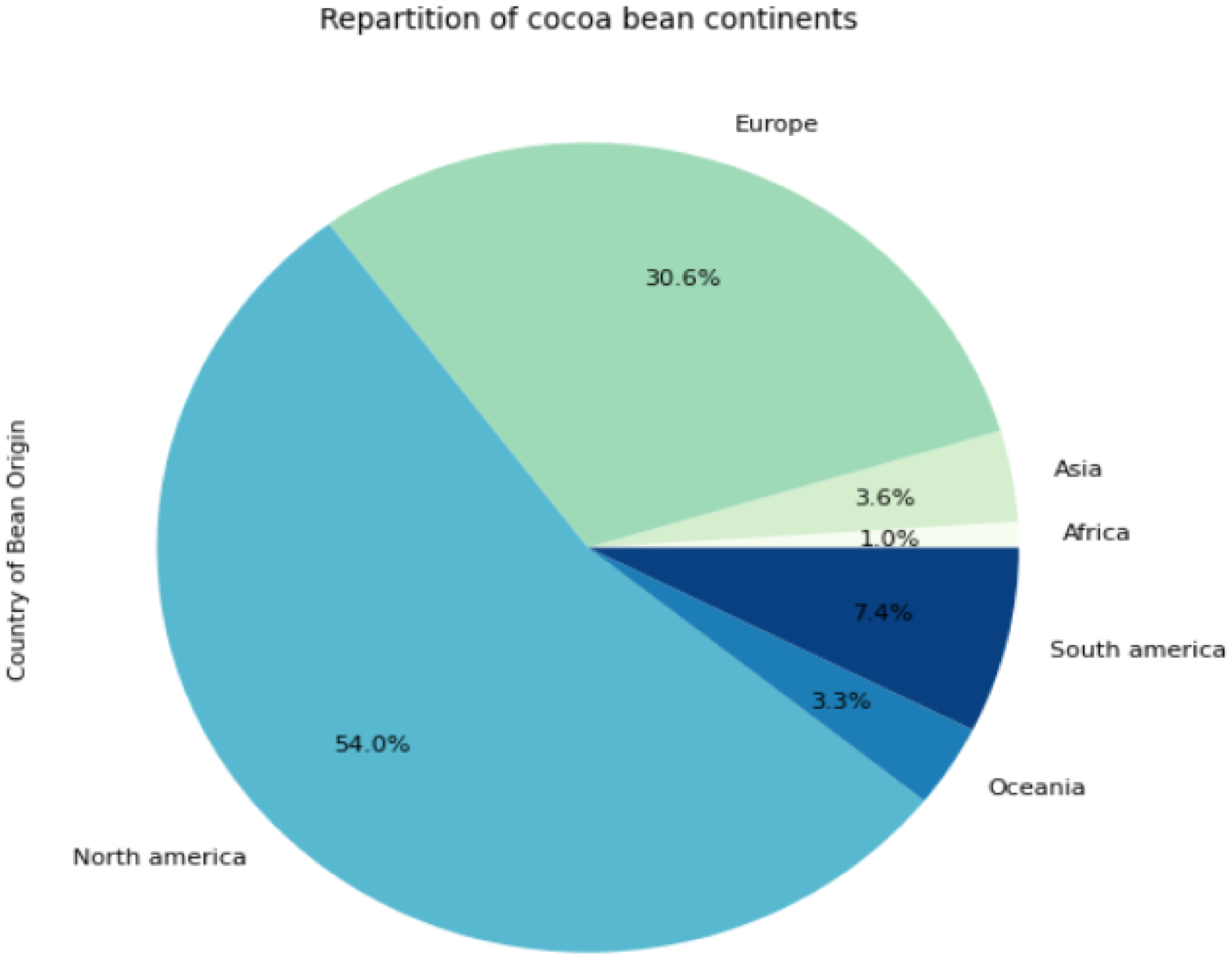
```
df['Company Location'].unique()
```

```
array(['U.S.A.', 'France', 'Fiji', 'Vanuatu', 'Ecuad
       'Netherlands', 'Spain', 'Russia', 'Peru', 'Ca
       'Brazil', 'Nicaragua', 'Australia', 'Philippi
       'Belgium', 'Vietnam', 'Germany', 'Singapore',
       'Venezuela', 'Malaysia', 'South Korea', 'Taiw
       'Colombia', 'Japan', 'New Zealand', 'Costa Ri
       'Amsterdam', 'Scotland', 'Martinique', 'Sao T
       'Argentina', 'Guatemala', 'South Africa', 'Do
       'Sao Tome & Principe', 'Bolivia', 'Portugal',
       'Grenada', 'Israel', 'India', 'St.Vincent-Gre
       'Czech Republic', 'Thailand', 'Finland', 'Mad
       'Poland', 'Austria', 'Honduras', 'U.A.E.', 'L
       'Chile', 'Ghana', 'Iceland', 'Suriname', 'El
      dtype=object)
```

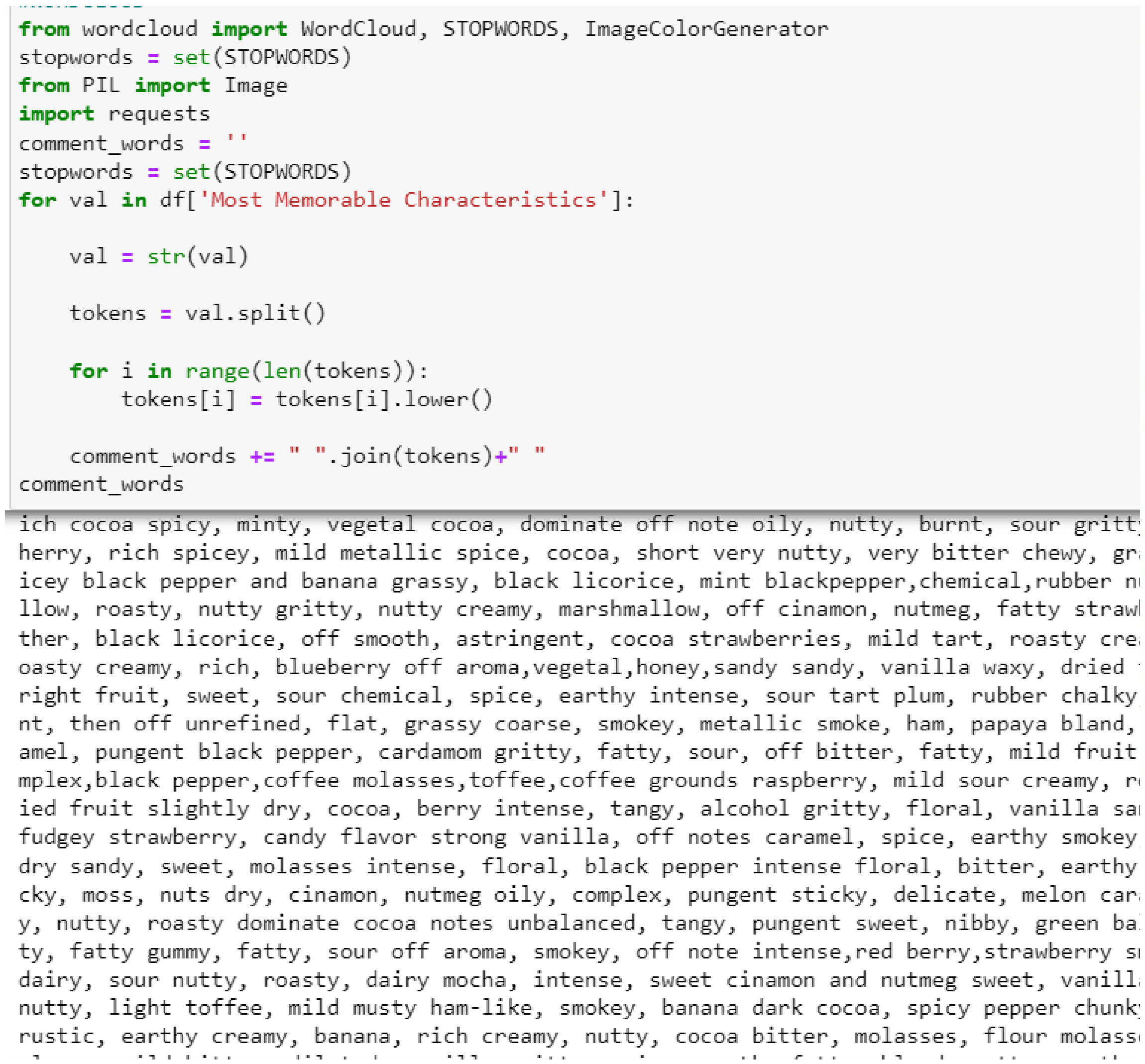
```
Asia = ['Japan', 'Vietnam', 'Israel', 'South Korea',
North_america = ['U.S.A.', 'Canada', 'Martinique', '
South_america = ['Ecuador', 'Eucador', 'Colombia', '
Europe = ['France', 'Denmark', 'Scotland', 'Wales',
Oceania = ['Australia', 'New Zealand', 'Fiji']
Africa = ['Madagascar', 'Sao Tome', 'South Africa',
```



```
def continents(x):
    if x in Asia:
        return 'Asia'
    if x in Africa:
        return 'Africa'
    if x in North_america:
        return 'North america'
    if x in South_america:
        return 'South america'
    if x in Europe:
        return 'Europe'
    if x in Oceania:
        return 'Oceania'
    return 'Europe'
```

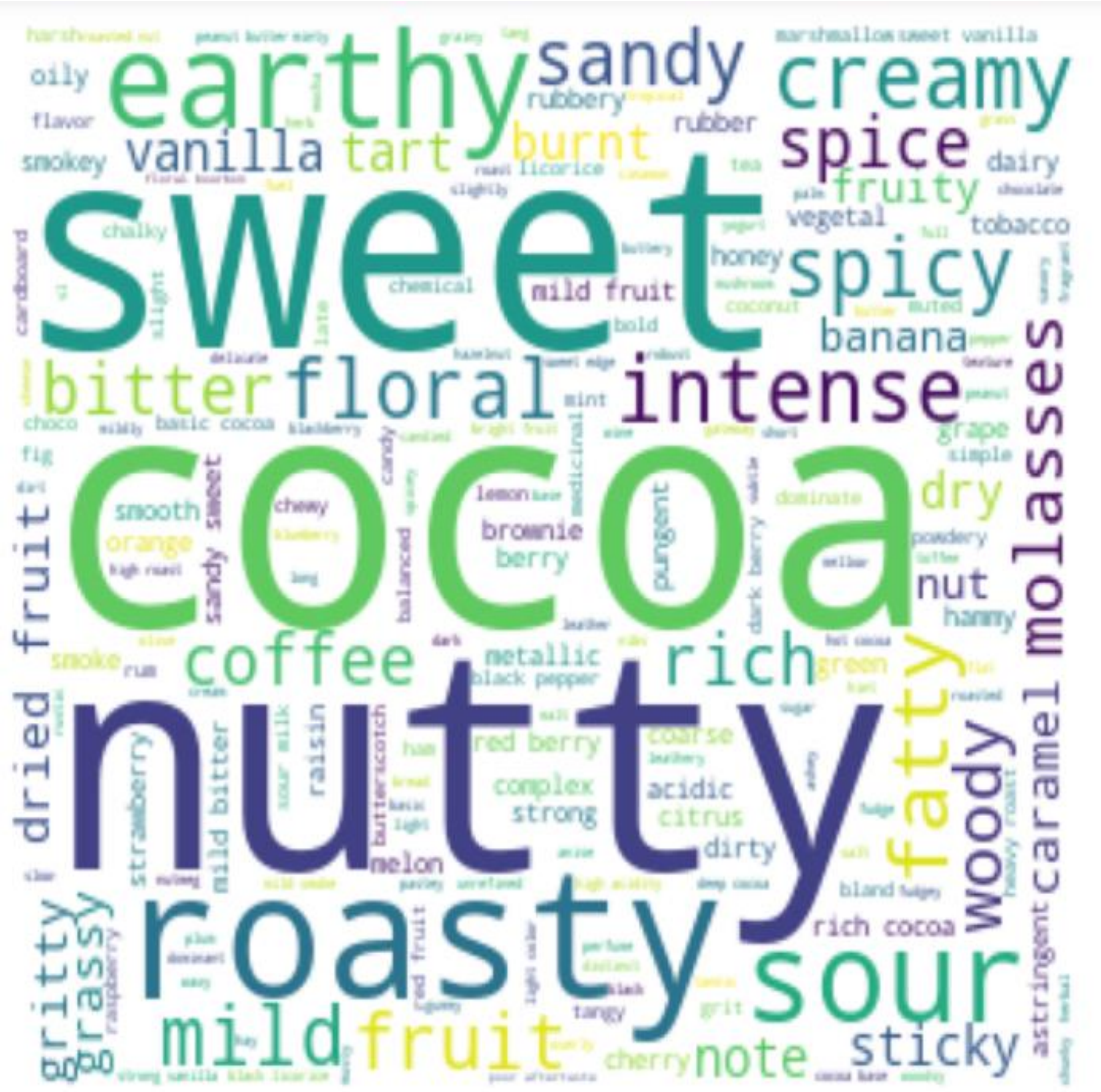

Repartition of cocoa bean continents
Country of Bean Origin
Europe
30.6%
Asia
3.6%
Africa
1.0%
South america
7.4%
Oceania
3.3%
North america
54.0%

Wordcloud



```
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
stopwords = set(STOPWORDS)
from PIL import Image
import requests
comment_words = ''
stopwords = set(STOPWORDS)
for val in df['Most Memorable Characteristics']:

    val = str(val)

    tokens = val.split()

    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()

    comment_words += " ".join(tokens)+" "
comment_words
```

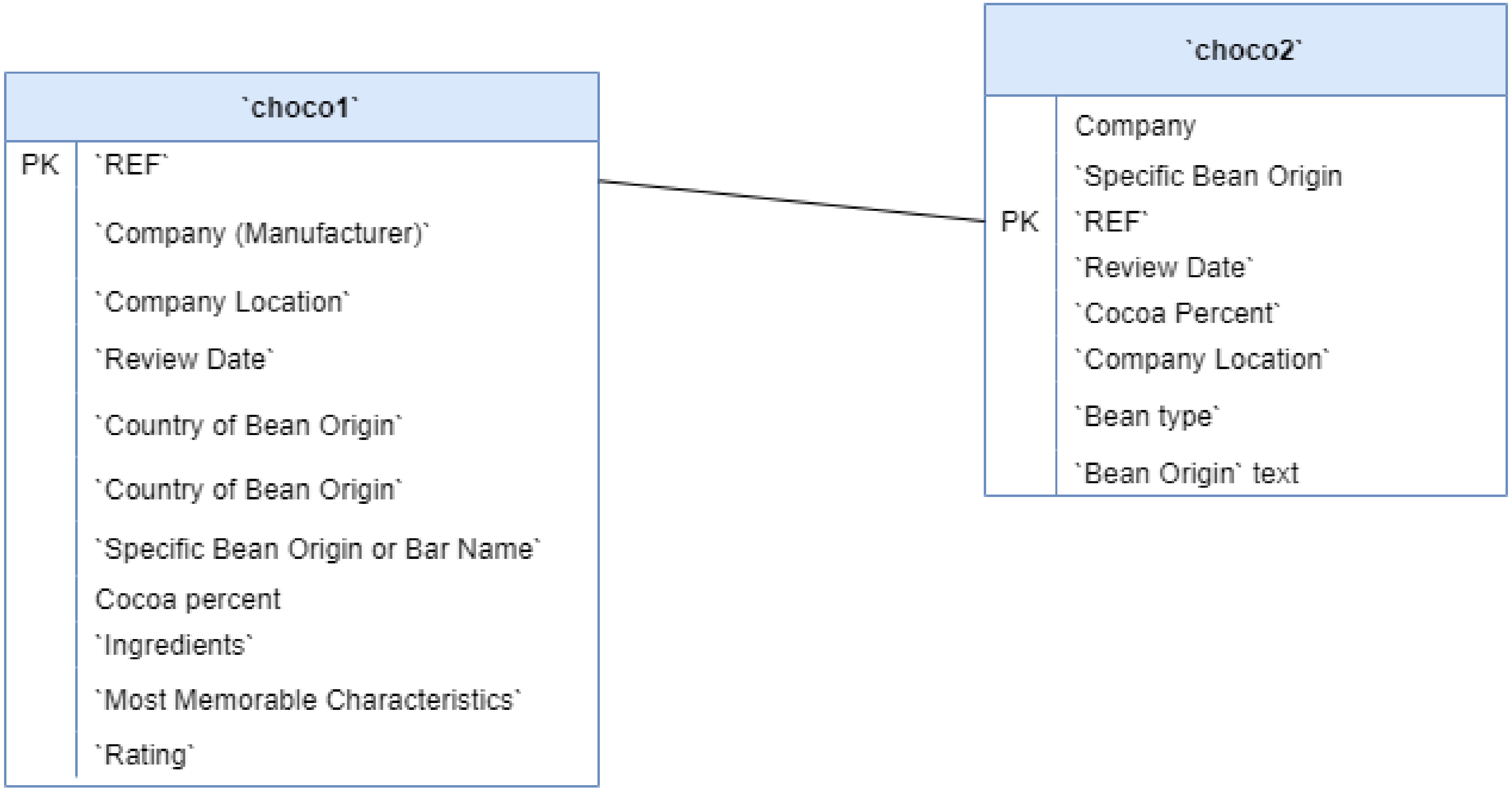
```
ich cocoa spicy, minty, vegetal cocoa, dominate off note oily, nutty, burnt, sour gritt
herry, rich spicey, mild metallic spice, cocoa, short very nutty, very bitter chewy, gr
icey black pepper and banana grassy, black licorice, mint blackpepper,chemical,rubber n
llow, roasty, nutty gritty, nutty creamy, marshmallow, off cinamon, nutmeg, fatty straw
ther, black licorice, off smooth, astringent, cocoa strawberries, mild tart, roasty cre
oasty creamy, rich, blueberry off aroma,vegetal,honey,sandy sandy, vanilla waxy, dried
right fruit, sweet, sour chemical, spice, earthy intense, sour tart plum, rubber chalky
nt, then off unrefined, flat, grassy coarse, smokey, metallic smoke, ham, papaya bland,
amel, pungent black pepper, cardamom gritty, fatty, sour, off bitter, fatty, mild fruit
mplex,black pepper,coffee molasses,toffee,coffee grounds raspberry, mild sour creamy, r
ied fruit slightly dry, cocoa, berry intense, tangy, alcohol gritty, floral, vanilla sa
fudgey strawberry, candy flavor strong vanilla, off notes caramel, spice, earthy smokey
dry sandy, sweet, molasses intense, floral, black pepper intense floral, bitter, earthy
cky, moss, nuts dry, cinamon, nutmeg oily, complex, pungent sticky, delicate, melon car
y, nutty, roasty dominate cocoa notes unbalanced, tangy, pungent sweet, nibby, green ba
ty, fatty gummy, fatty, sour off aroma, smokey, off note intense,red berry,strawberry s
dairy, sour nutty, roasty, dairy mocha, intense, sweet cinamon and nutmeg sweet, vanill
nutty, light toffee, mild musty ham-like, smokey, banana dark cocoa, spicy pepper chunk
rustic, earthy creamy, banana, rich creamy, nutty, cocoa bitter, molasses, flour molass
```



Database type comparation

SQL	NoSQL
- Relational - Use structured query language and predefined schema - Can work with smaller amount of data - Are table-based, multi-row transaction - OOP unfriendly (object-oriented programing)	- Non-relational - Have dynamic schemas for unstructured or semi structured data - Can work with big amount of data - Document key-value, graph, wide-column - OOP friendly

Entities. ERD



`choco1`	
PK	`REF`
	`Company (Manufacturer)`
	`Company Location`
	`Review Date`
	`Country of Bean Origin`
	`Country of Bean Origin`
	`Specific Bean Origin or Bar Name`
	Cocoa percent
	`Ingredients`
	`Most Memorable Characteristics`
	`Rating`

`choco2`	
	Company
	`Specific Bean Origin
PK	`REF`
	`Review Date`
	`Cocoa Percent`
	`Company Location`
	`Bean type`
	`Bean Origin` text

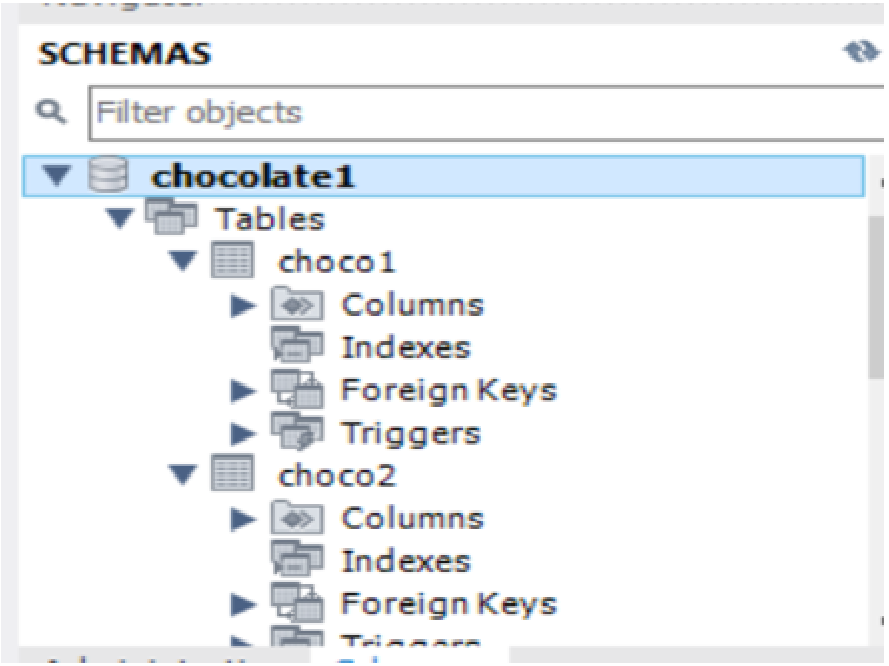
MySQL

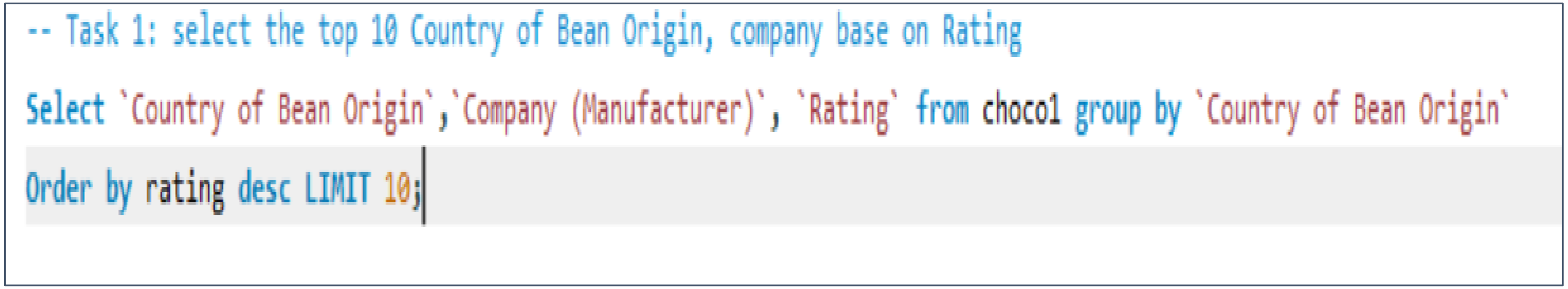
Database creation and data importation

After deciding on which type of database to use, I began creating my relational database on MySQL workbench with “create database if not exists”:

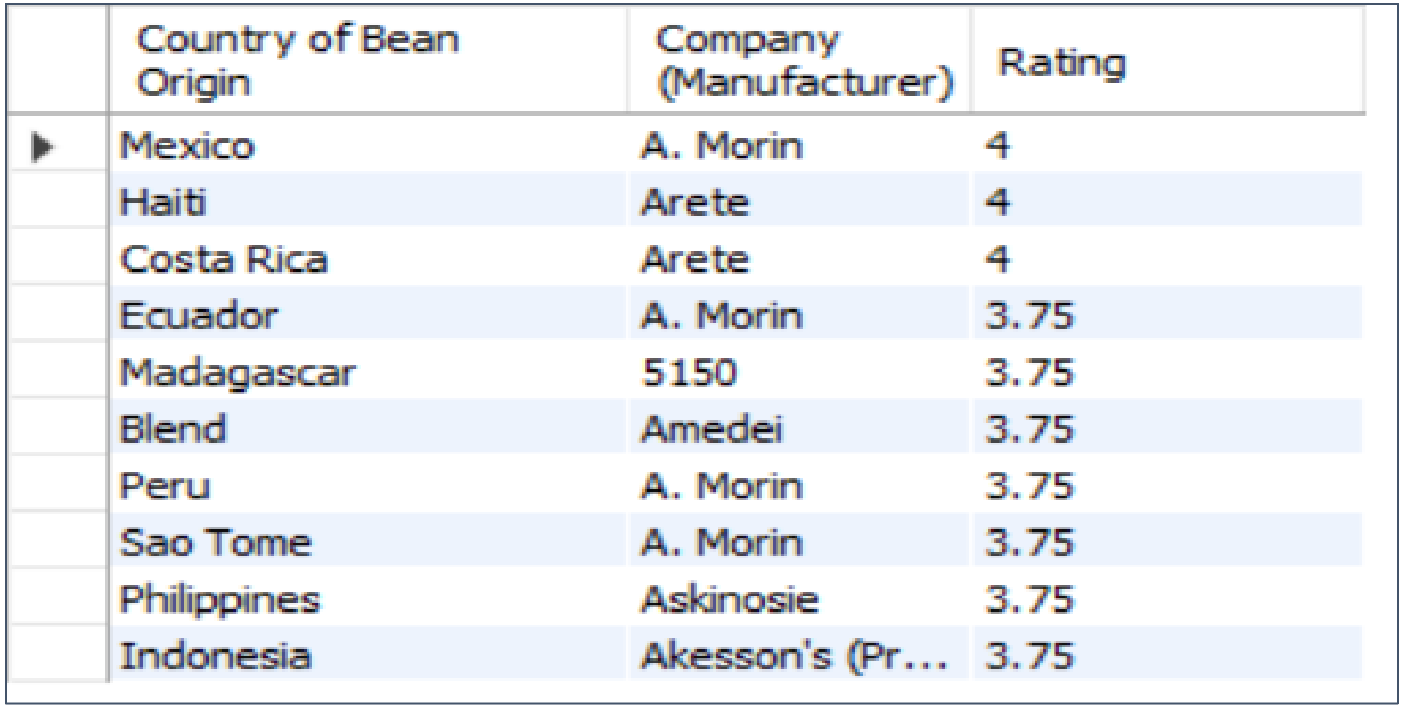
- `create database if not exists CHOCOLATE1;`
- `use CHOCOLATE1;`

Then I upload 2 table choco1 and choco2 by clicking on “*Table Data Import Wizard*”. After import successfully, I start to make some queries to response for 5 tasks above in order to receive various insights from my data.

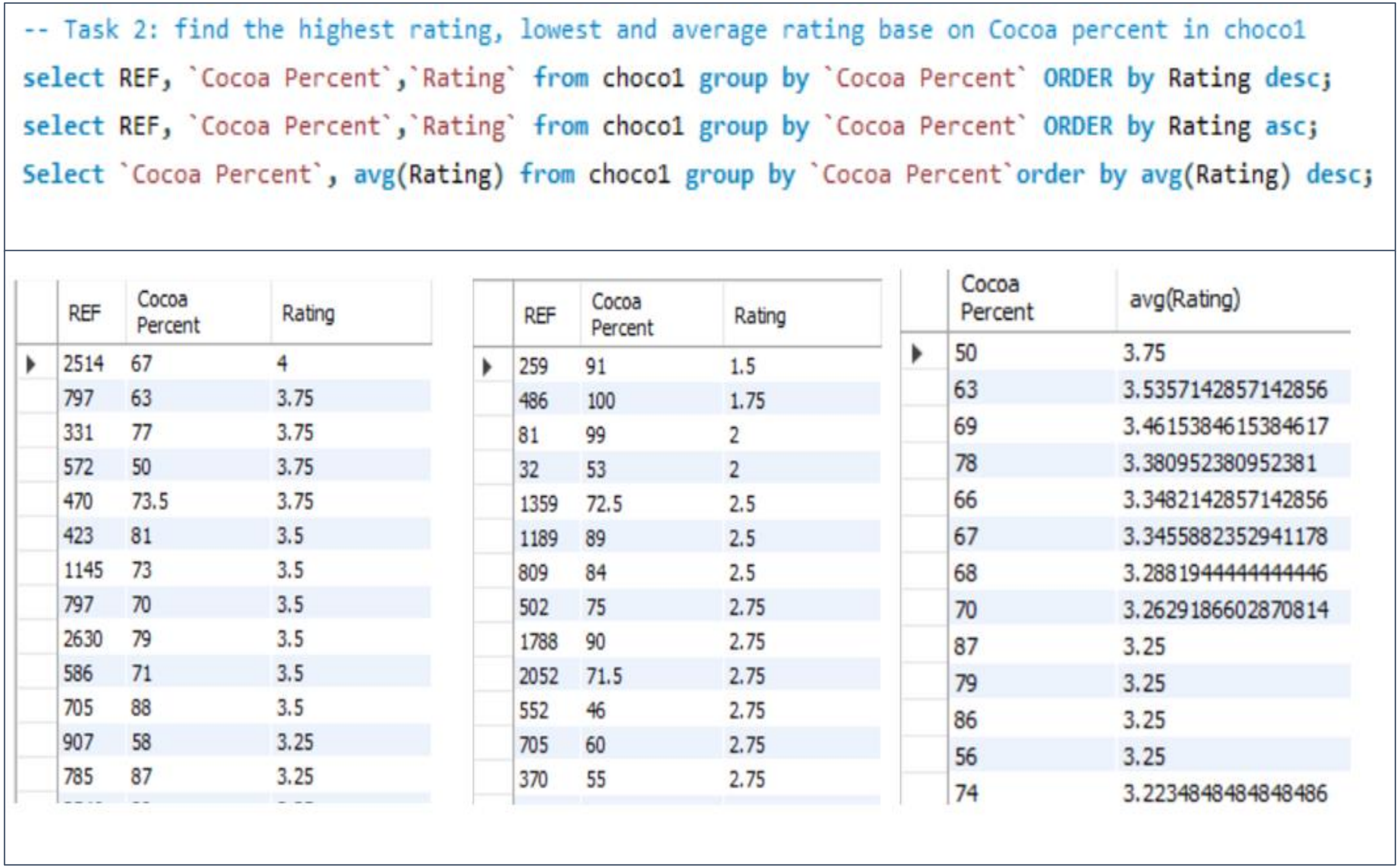




```
-- Task 1: select the top 10 Country of Bean Origin, company base on Rating
Select `Country of Bean Origin`,`Company (Manufacturer)`, `Rating` from choco1 group by `Country of Bean Origin`
Order by rating desc LIMIT 10;
```



	Country of Bean Origin	Company (Manufacturer)	Rating
▶	Mexico	A. Morin	4
	Haiti	Arete	4
	Costa Rica	Arete	4
	Ecuador	A. Morin	3.75
	Madagascar	5150	3.75
	Blend	Amedei	3.75
	Peru	A. Morin	3.75
	Sao Tome	A. Morin	3.75
	Philippines	Askinosie	3.75
	Indonesia	Akesson's (Pr...	3.75

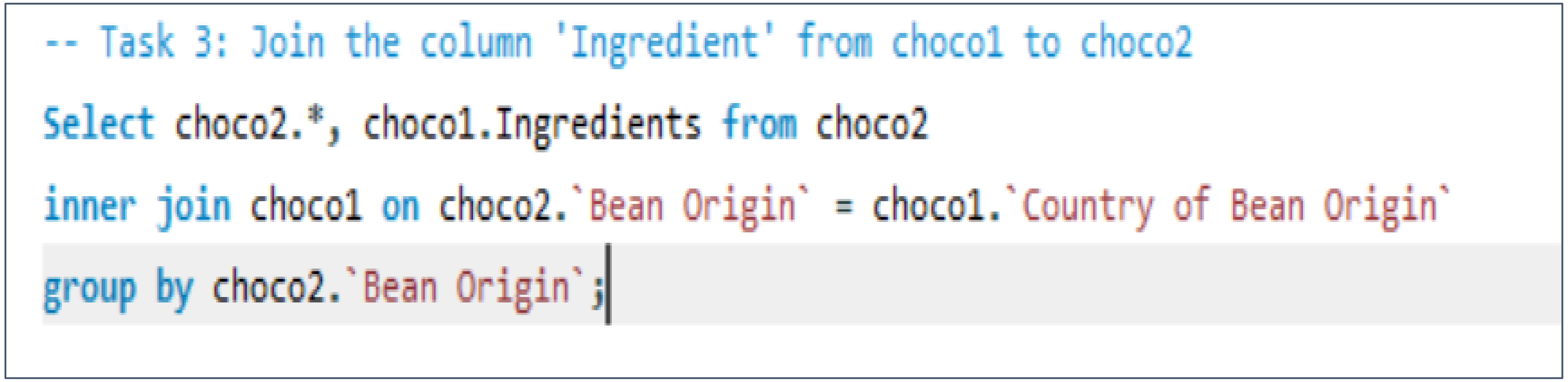


```
-- Task 2: find the highest rating, lowest and average rating base on Cocoa percent in choco1
select REF, `Cocoa Percent`,`Rating` from choco1 group by `Cocoa Percent` ORDER by Rating desc;
select REF, `Cocoa Percent`,`Rating` from choco1 group by `Cocoa Percent` ORDER by Rating asc;
Select `Cocoa Percent`, avg(Rating) from choco1 group by `Cocoa Percent`order by avg(Rating) desc;
```

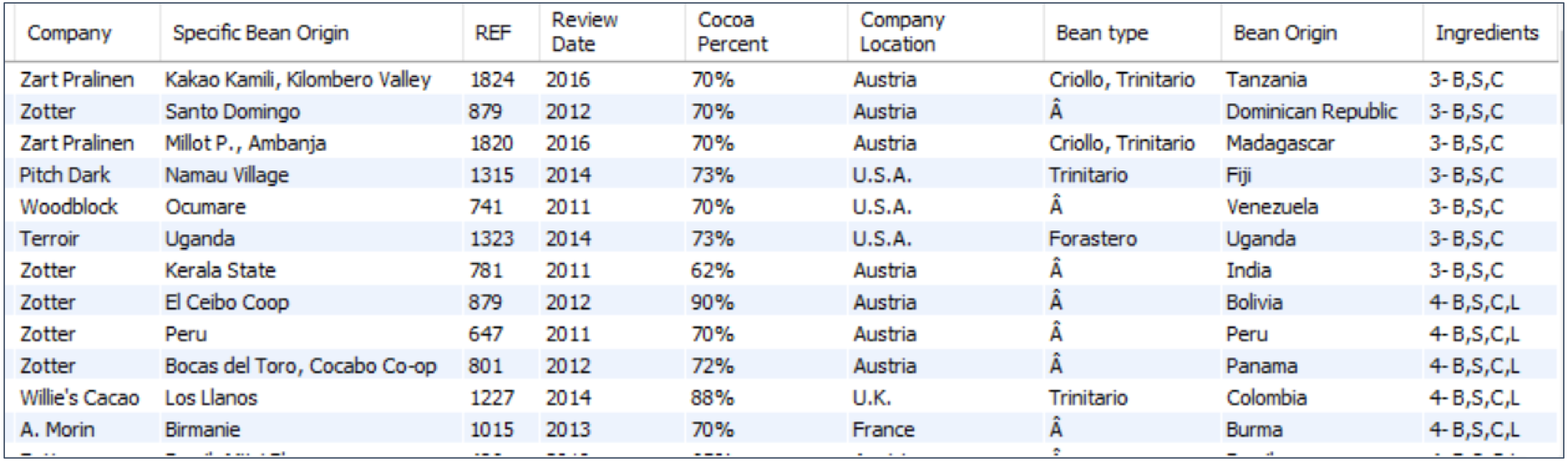
REF	Cocoa Percent	Rating
2514	67	4
797	63	3.75
331	77	3.75
572	50	3.75
470	73.5	3.75
423	81	3.5
1145	73	3.5
797	70	3.5
2630	79	3.5
586	71	3.5
705	88	3.5
907	58	3.25
785	87	3.25

REF	Cocoa Percent	Rating
259	91	1.5
486	100	1.75
81	99	2
32	53	2
1359	72.5	2.5
1189	89	2.5
809	84	2.5
502	75	2.75
1788	90	2.75
2052	71.5	2.75
552	46	2.75
705	60	2.75
370	55	2.75

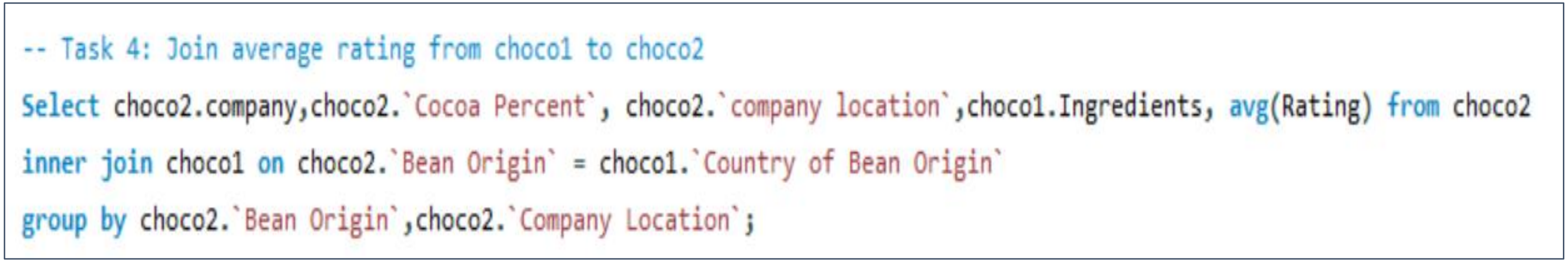
Cocoa Percent	avg(Rating)
50	3.75
63	3.5357142857142856
69	3.4615384615384617
78	3.380952380952381
66	3.3482142857142856
67	3.3455882352941178
68	3.2881944444444446
70	3.2629186602870814
87	3.25
79	3.25
86	3.25
56	3.25
74	3.2234848484848486



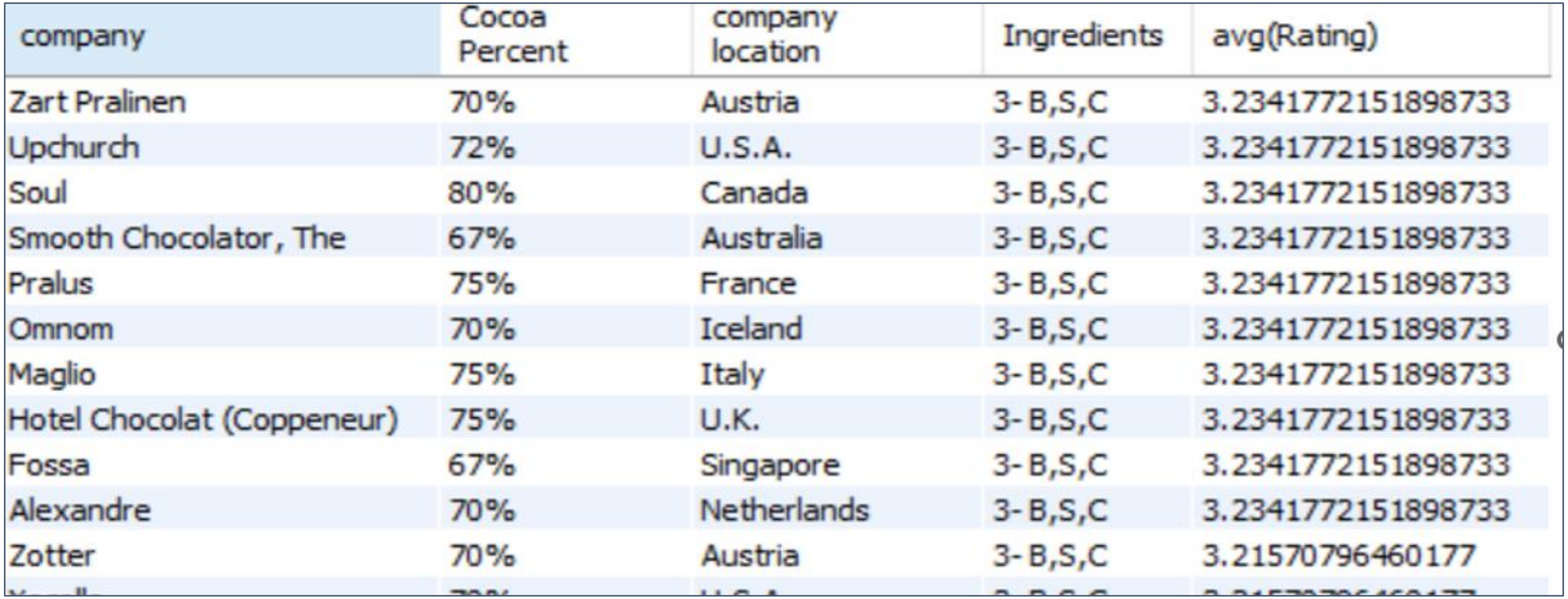
```
-- Task 3: Join the column 'Ingredient' from choco1 to choco2
Select choco2.*, choco1.Ingredients from choco2
inner join choco1 on choco2.`Bean Origin` = choco1.`Country of Bean Origin`
group by choco2.`Bean Origin`;
```



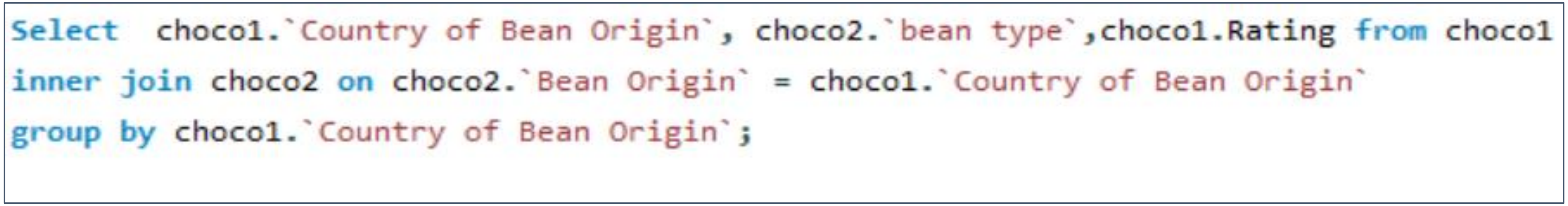
Company	Specific Bean Origin	REF	Review Date	Cocoa Percent	Company Location	Bean type	Bean Origin	Ingredients
Zart Pralinen	Kakao Kamili, Kilombero Valley	1824	2016	70%	Austria	Criollo, Trinitario	Tanzania	3- B,S,C
Zotter	Santo Domingo	879	2012	70%	Austria	Â	Dominican Republic	3- B,S,C
Zart Pralinen	Millot P., Ambanja	1820	2016	70%	Austria	Criollo, Trinitario	Madagascar	3- B,S,C
Pitch Dark	Namau Village	1315	2014	73%	U.S.A.	Trinitario	Fiji	3- B,S,C
Woodblock	Ocumare	741	2011	70%	U.S.A.	Â	Venezuela	3- B,S,C
Terroir	Uganda	1323	2014	73%	U.S.A.	Forastero	Uganda	3- B,S,C
Zotter	Kerala State	781	2011	62%	Austria	Â	India	3- B,S,C
Zotter	El Ceibo Coop	879	2012	90%	Austria	Â	Bolivia	4- B,S,C,L
Zotter	Peru	647	2011	70%	Austria	Â	Peru	4- B,S,C,L
Zotter	Bocas del Toro, Cocabo Co-op	801	2012	72%	Austria	Â	Panama	4- B,S,C,L
Willie's Cacao	Los Llanos	1227	2014	88%	U.K.	Trinitario	Colombia	4- B,S,C,L
A. Morin	Birmanie	1015	2013	70%	France	Â	Burma	4- B,S,C,L



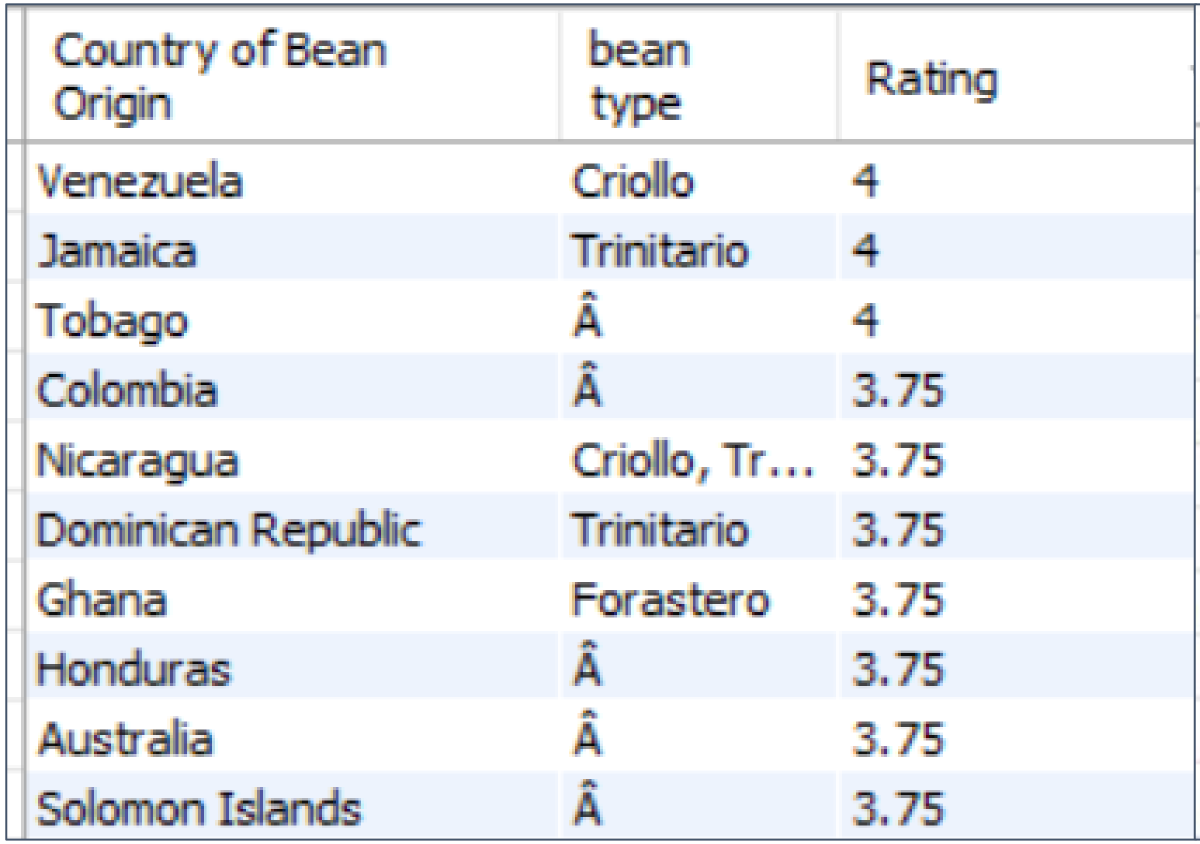
```
-- Task 4: Join average rating from choco1 to choco2
Select choco2.company,choco2.`Cocoa Percent`, choco2.`company location`,choco1.Ingredients, avg(Rating) from choco2
inner join choco1 on choco2.`Bean Origin` = choco1.`Country of Bean Origin`
group by choco2.`Bean Origin`,choco2.`Company Location`;
```



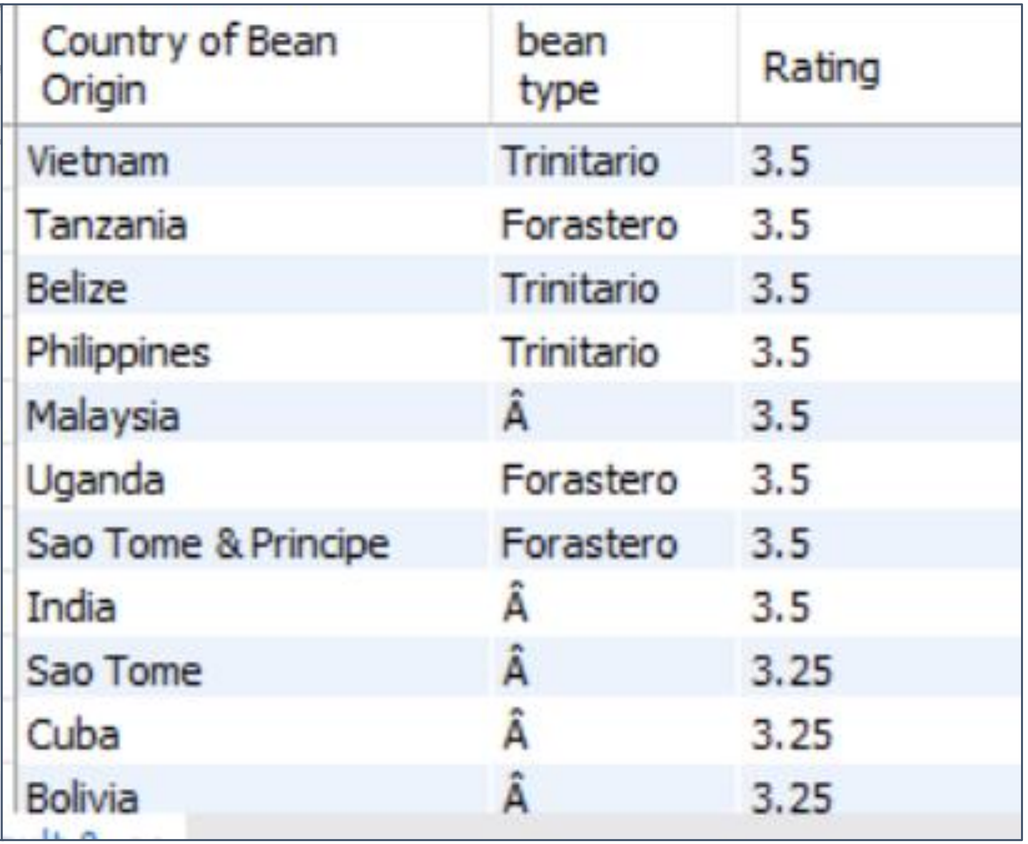
company	Cocoa Percent	company location	Ingredients	avg(Rating)
Zart Pralinen	70%	Austria	3- B,S,C	3.234177215189873
Upchurch	72%	U.S.A.	3- B,S,C	3.234177215189873
Soul	80%	Canada	3- B,S,C	3.234177215189873
Smooth Chocolator, The	67%	Australia	3- B,S,C	3.234177215189873
Pralus	75%	France	3- B,S,C	3.234177215189873
Omnom	70%	Iceland	3- B,S,C	3.234177215189873
Maglio	75%	Italy	3- B,S,C	3.234177215189873
Hotel Chocolat (Coppeneur)	75%	U.K.	3- B,S,C	3.234177215189873
Fossa	67%	Singapore	3- B,S,C	3.234177215189873
Alexandre	70%	Netherlands	3- B,S,C	3.234177215189873
Zotter	70%	Austria	3- B,S,C	3.2157079646017



```
Select  choco1.`Country of Bean Origin`, choco2.`bean type`,choco1.Rating from choco1
inner join choco2 on choco2.`Bean Origin` = choco1.`Country of Bean Origin`
group by choco1.`Country of Bean Origin`;
```

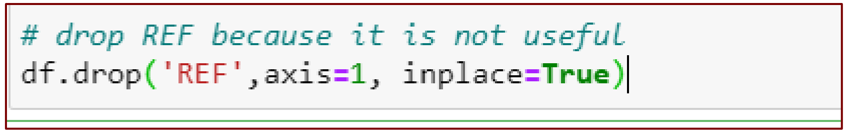


Country of Bean Origin	bean type	Rating
Venezuela	Criollo	4
Jamaica	Trinitario	4
Tobago	Â	4
Colombia	Â	3.75
Nicaragua	Criollo, Tr...	3.75
Dominican Republic	Trinitario	3.75
Ghana	Forastero	3.75
Honduras	Â	3.75
Australia	Â	3.75
Solomon Islands	Â	3.75

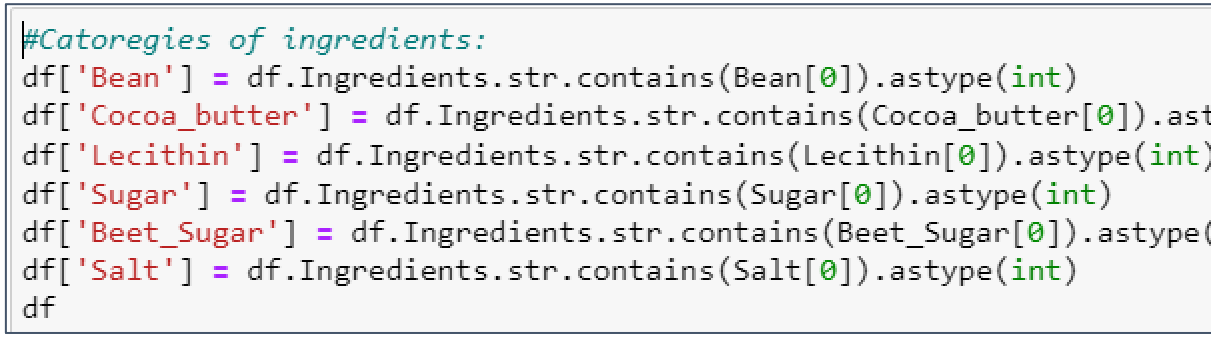


Country of Bean Origin	bean type	Rating
Vietnam	Trinitario	3.5
Tanzania	Forastero	3.5
Belize	Trinitario	3.5
Philippines	Trinitario	3.5
Malaysia	Â	3.5
Uganda	Forastero	3.5
Sao Tome & Principe	Forastero	3.5
India	Â	3.5
Sao Tome	Â	3.25
Cuba	Â	3.25
Bolivia	Â	3.25

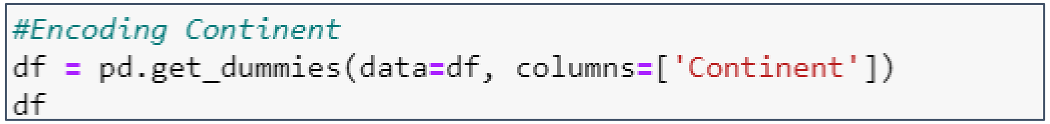
Data preparation



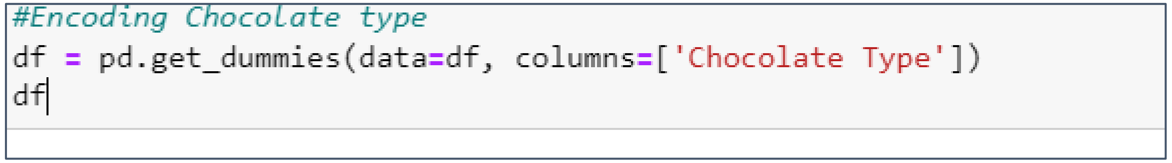
```
# drop REF because it is not useful
df.drop('REF',axis=1, inplace=True)
```



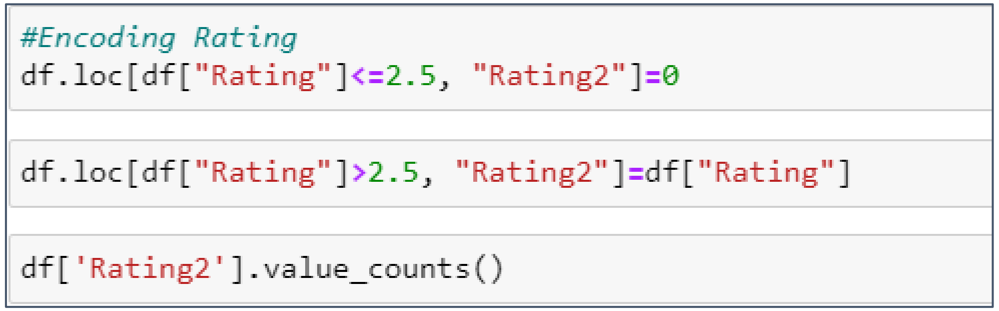
```
#Catoregies of ingredients:
df['Bean'] = df.Ingredients.str.contains(Bean[0]).astype(int)
df['Cocoa_butter'] = df.Ingredients.str.contains(Cocoa_butter[0]).ast
df['Lecithin'] = df.Ingredients.str.contains(Lecithin[0]).astype(int)
df['Sugar'] = df.Ingredients.str.contains(Sugar[0]).astype(int)
df['Beet_Sugar'] = df.Ingredients.str.contains(Beet_Sugar[0]).astype(
df['Salt'] = df.Ingredients.str.contains(Salt[0]).astype(int)
df
```



```
#Encoding Continent
df = pd.get_dummies(data=df, columns=['Continent'])
df
```



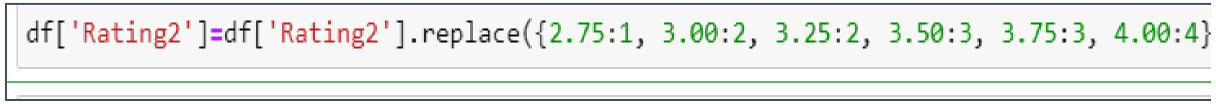
```
#Encoding Chocolate type
df = pd.get_dummies(data=df, columns=['Chocolate Type'])
df
```



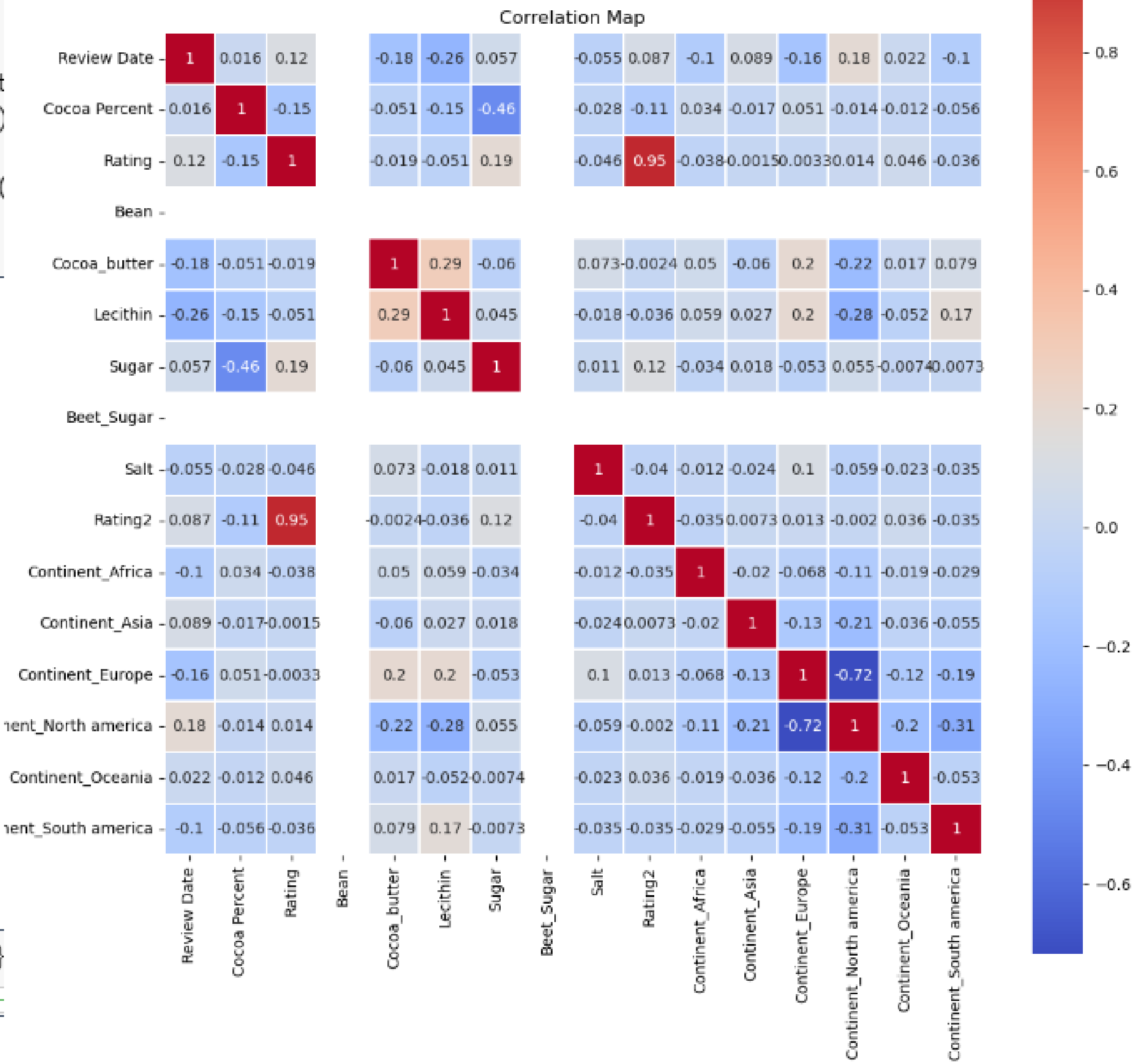
```
#Encoding Rating
df.loc[df["Rating"]<=2.5, "Rating2"]=0
```

```
df.loc[df["Rating"]>2.5, "Rating2"]=df["Rating"]
```

```
df['Rating2'].value_counts()
```



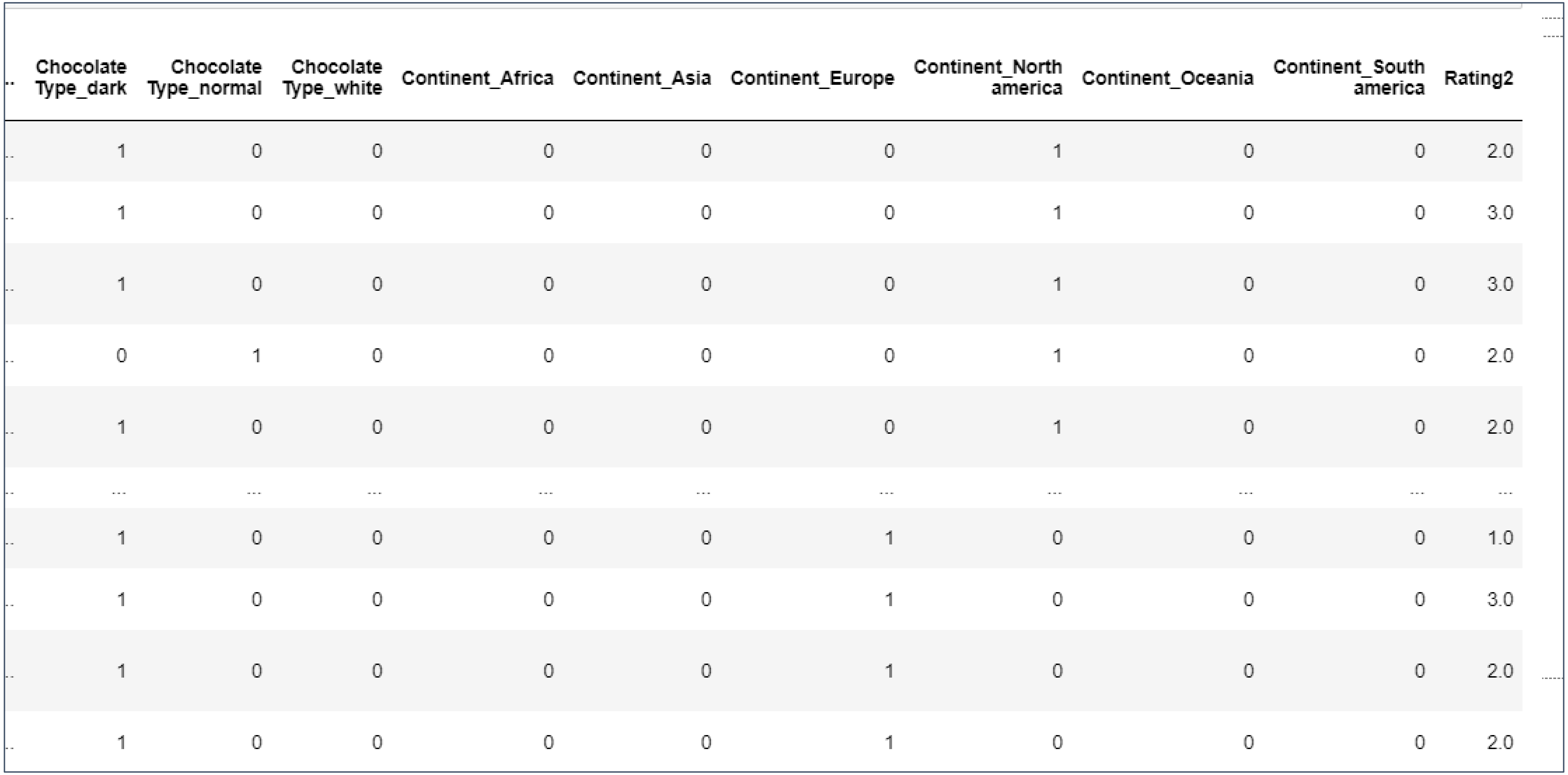
```
df['Rating2']=df['Rating2'].replace({2.75:1, 3.00:2, 3.25:2, 3.50:3, 3.75:3, 4.00:4}
```



Correlation Map

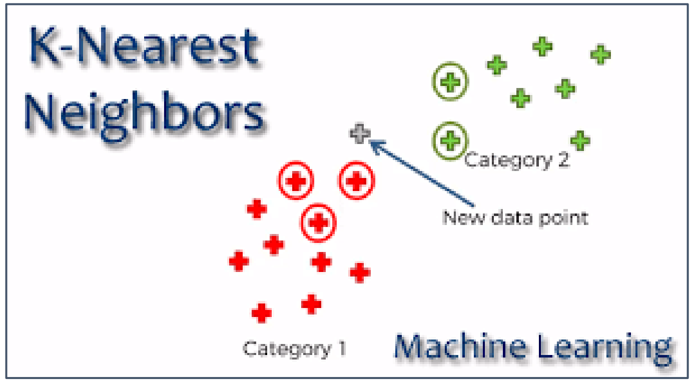
	Review Date	Cocoa Percent	Rating	Bean	Cocoa_butter	Lecithin	Sugar	Beet_Sugar	Salt	Rating2	Continent_Africa	Continent_Asia	Continent_Europe	Continent_North america	Continent_Oceania	Continent_South america
Review Date	1	0.016	0.12		-0.18	-0.26	0.057		-0.055	0.087	-0.1	0.089	-0.16	0.18	0.022	-0.1
Cocoa Percent	0.016	1	-0.15		-0.051	-0.15	-0.46		-0.028	-0.11	0.034	-0.017	0.051	-0.014	-0.012	-0.056
Rating	0.12	-0.15	1		-0.019	-0.051	0.19		-0.046	0.95	-0.038	-0.0015	-0.0033	0.014	0.046	-0.036
Bean																
Cocoa_butter	-0.18	-0.051	-0.019		1	0.29	-0.06		0.073	-0.0024	0.05	-0.06	0.2	-0.22	0.017	0.079
Lecithin	-0.26	-0.15	-0.051		0.29	1	0.045		-0.018	-0.036	0.059	0.027	0.2	-0.28	-0.052	0.17
Sugar	0.057	-0.46	0.19		-0.06	0.045	1		0.011	0.12	-0.034	0.018	-0.053	0.055	-0.0074	0.0073
Beet_Sugar																
Salt	-0.055	-0.028	-0.046		0.073	-0.018	0.011		1	-0.04	-0.012	-0.024	0.1	-0.059	-0.023	-0.035
Rating2	0.087	-0.11	0.95		-0.0024	-0.036	0.12		-0.04	1	-0.035	0.0073	0.013	-0.002	0.036	-0.035
Continent_Africa	-0.1	0.034	-0.038		0.05	0.059	-0.034		-0.012	-0.035	1	-0.02	-0.068	-0.11	-0.019	-0.029
Continent_Asia	0.089	-0.017	-0.0015		-0.06	0.027	0.018		-0.024	0.0073	-0.02	1	-0.13	-0.21	-0.036	-0.055
Continent_Europe	-0.16	0.051	-0.0033		0.2	0.2	-0.053		0.1	0.013	-0.068	-0.13	1	-0.72	-0.12	-0.19
ent_North america	0.18	-0.014	0.014		-0.22	-0.28	0.055		-0.059	-0.002	-0.11	-0.21	-0.72	1	-0.2	-0.31
Continent_Oceania	0.022	-0.012	0.046		0.017	-0.052	-0.0074		-0.023	0.036	-0.019	-0.036	-0.12	-0.2	1	-0.053
ent_South america	-0.1	-0.056	-0.036		0.079	0.17	-0.0073		-0.035	-0.035	-0.029	-0.055	-0.19	-0.31	-0.053	1

23 columns

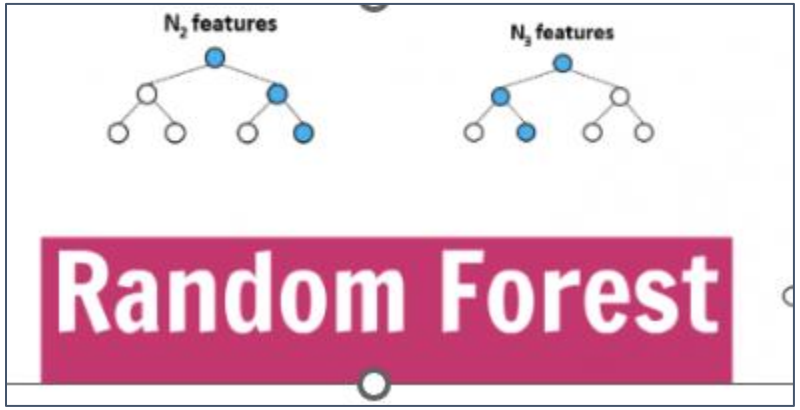


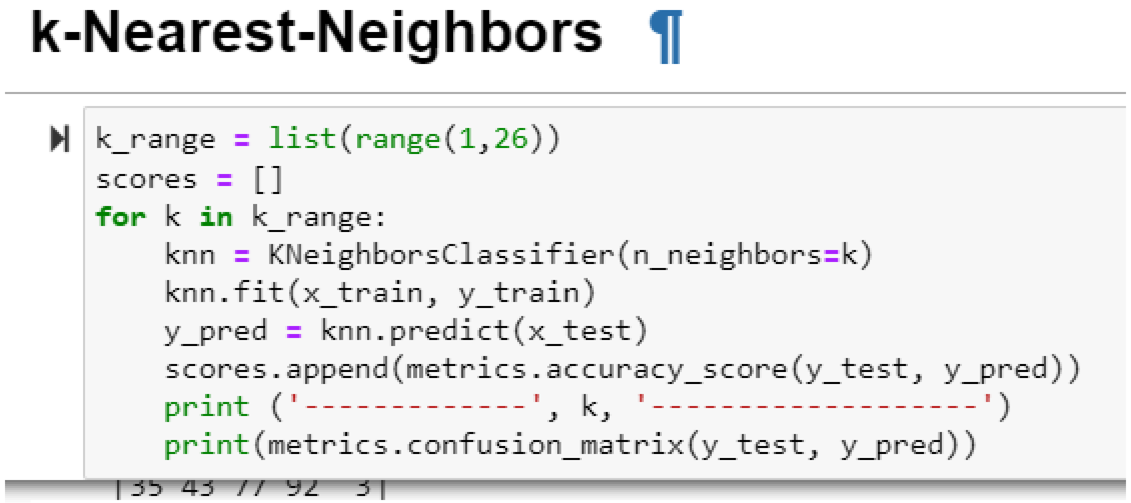
..	Chocolate Type_dark	Chocolate Type_normal	Chocolate Type_white	Continent_Africa	Continent_Asia	Continent_Europe	Continent_North america	Continent_Oceania	Continent_South america	Rating2
..	1	0	0	0	0	0	1	0	0	2.0
..	1	0	0	0	0	0	1	0	0	3.0
..	1	0	0	0	0	0	1	0	0	3.0
..	0	1	0	0	0	0	1	0	0	2.0
..	1	0	0	0	0	0	1	0	0	2.0
..	...	...	...	...	...	...	...	...	...	...
..	1	0	0	0	0	1	0	0	0	1.0
..	1	0	0	0	0	1	0	0	0	3.0
..	1	0	0	0	0	1	0	0	0	2.0
..	1	0	0	0	0	1	0	0	0	2.0

Deploy Machine learning



`sklearn.svm.`LinearSVC



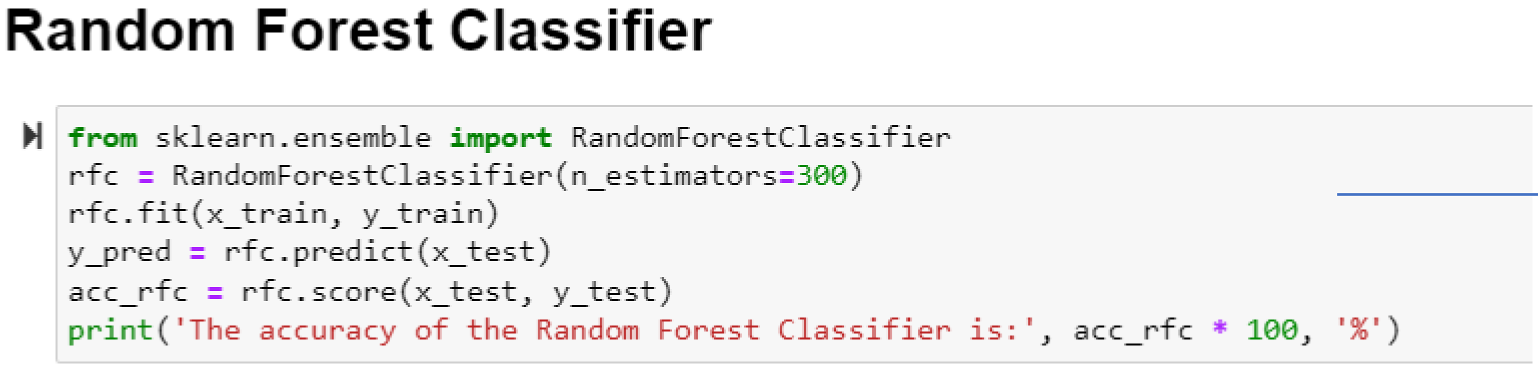


k-Nearest-Neighbors ¶

```
k_range = list(range(1,26))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(x_train, y_train)
    y_pred = knn.predict(x_test)
    scores.append(metrics.accuracy_score(y_test, y_pred))
    print ('-------------', k, '-------------------')
    print(metrics.confusion_matrix(y_test, y_pred))
```

```
 [35 43 77 92  3]
```

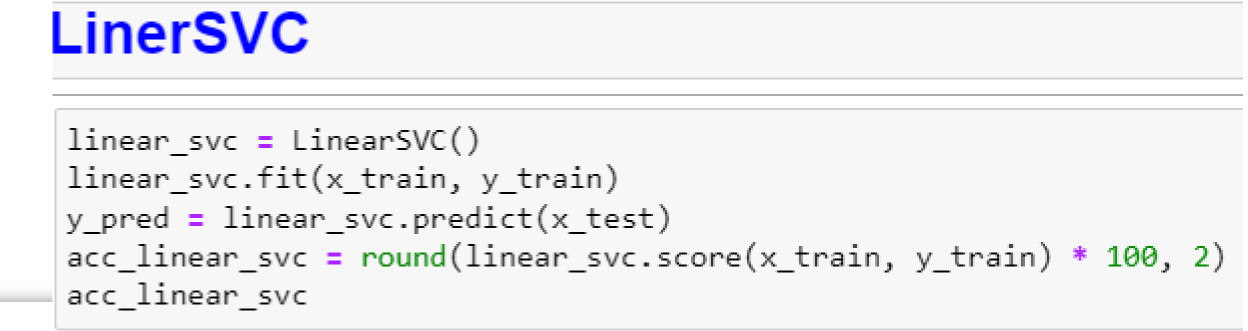
73%



Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=300)
rfc.fit(x_train, y_train)
y_pred = rfc.predict(x_test)
acc_rfc = rfc.score(x_test, y_test)
print('The accuracy of the Random Forest Classifier is:', acc_rfc * 100, '%')
```

66%



LinerSVC

```
linear_svc = LinearSVC()
linear_svc.fit(x_train, y_train)
y_pred = linear_svc.predict(x_test)
acc_linear_svc = round(linear_svc.score(x_train, y_train) * 100, 2)
acc_linear_svc
```

74%

Conclusion

Consumers prefer chocolate from **65-75%** cocoa

Chocolate brands can affect to the choices of customers

From review time, It show the quality of chocolate are improving

The biggest production of chocolate is in USA

Predict how well a chocolate bar will do in the near future to see if we can increase sales

Q &A
Thank you