Untappd Beer Data

by: Alissa Trujillo

Importing Necessary Packages

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from ClusteringReport import cluster_report

from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeClassifier, export_text
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import MinMaxScaler, StandardScaler

from sklearn.model_selection import train_test_split, GridSearchCV, RandomizedS
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_regression
from sklearn.pipeline import Pipeline

from sklearn.decomposition import PCA
from sklearn.cluster import KMeans

from imblearn.over_sampling import RandomOverSampler

pd.options.display.float_format = '{:.2f}'.format

import warnings
warnings.filterwarnings('ignore')
```

Data Preparation

Importing Dataset

In [6]:
```
beer_df = pd.read_csv("beer_profile_and_ratings.csv")
```

In [4]:
```
beer_dff = pd.read_csv('beer_profile_and_ratings.csv')
beer_dff.head()
```

Out[4]:

	Name	Style	Brewery	Beer Name (Full)	Description	ABV	Min IBU	Max IBU	Astringency	B
0	Amber	Altbier	Alaskan Brewing Co.	Alaskan Brewing Co. Alaskan Amber	Notes:Richly malty and long on the palate, wit...	5.30	25	50	13	
1	Double Bag	Altbier	Long Trail Brewing Co.	Long Trail Brewing Co. Double Bag	Notes:This malty, full-bodied double alt is al...	7.20	25	50	12	
2	Long Trail Ale	Altbier	Long Trail Brewing Co.	Long Trail Brewing Co. Long Trail Ale	Notes:Long Trail Ale is a full-bodied amber al...	5.00	25	50	14	
3	Doppelsticke	Altbier	Uerige Obergärige Hausbrauerei GmbH / Zum Uerige	Uerige Obergärige Hausbrauerei GmbH / Zum Ueri...	Notes:	8.50	25	50	13	
4	Sleigh'r Dark Doüble Alt Ale	Altbier	Ninkasi Brewing Company	Ninkasi Brewing Company Sleigh'r Dark Doüble A...	Notes:Called 'Dark Double Alt' on the label.Se...	7.20	25	50	25	

5 rows × 25 columns

Dropping Unnecessary Columns

In [7]:
```python
beer_df.drop(['Name', 'Brewery', 'Description',
                   'Beer Name (Full)', 'number_of_reviews'],
                    axis=1, inplace=True)
```

This dataset thankfully only needs a few simple tweaks to make it model-ready. The first step will be to remove any features that are not relevant to either of our business problems. Here, we are removing the 'Name', 'Brewery', 'Description, 'Beer Name' and 'number_of_reviews' columns.

Consolidating IBU Variable

In [8]:
```python
IBU = beer_df.iloc[:,3:5].mean(axis=1)
beer_df.insert(3, 'IBU', IBU)
beer_df.drop(['Min IBU','Max IBU'], axis=1, inplace=True)
```

IBU is represented in the dataset by a range of values. In order to aid us in model creation, we will consolidate this range to the average of the Min and Max values for each beer.

Grouping Together Similar Styles

In [9]:
```python
beer_df['Style'] = beer_df['Style'].str.split(' - ').str[0]
```

The original dataset has 111 unique values in the 'Style' column. We will need to consolidate these to reduce dimensionality. A number of styles had extra descriptors (ex. 'Stout – English', 'Stout – Oatmeal' and 'Stout – Russian Imperial' were all unique styles). By splitting the styles by the "-" character and only keeping the broader category (e.g. 'Stout'), we are able to reduce the number of styles to 43.

Removing Correlated Values

In [10]:
```
beer_df.drop(columns=beer_df.columns[-5:-1], axis=1, inplace=True)
```

We have a number of different features that represent consumer's review ratings for different aspects of the beers (aroma, appearance, palate, taste and overall). Since all of the other review scores are condensed into a final overall review score, we will only keep that single overall review variable. This will ensure that none of our features are heavily correlated with each other.

In [12]:
```
beer_dfRF = beer_df.drop('review_overall', axis=1)
```

In [14]:
```
beer_dfRF.head()
```

Out[14]:

	Style	ABV	IBU	Astringency	Body	Alcohol	Bitter	Sweet	Sour	Salty	Fruits	Hoppy
0	Altbier	5.30	31.50	13	32	9	47	74	33	0	33	57
1	Altbier	7.20	31.00	12	57	18	33	55	16	0	24	35
2	Altbier	5.00	32.00	14	37	6	42	43	11	0	10	54
3	Altbier	8.50	31.50	13	55	31	47	101	18	1	49	40
4	Altbier	7.20	37.50	25	51	26	44	45	9	1	11	51

Separating Target Variable

In [14]:
```
featuresRF = beer_dfRF.drop('Style', axis=1)
targetRF = beer_dfRF['Style']
```

Our target variable for the classification problem is 'Style'. Our goal in creating this model is to see how accurately we are able to predict a beer's class based on its inherent properties. With 42 different styles, I anticipate it requiring a fine bit of tuning. However, the multitude of classes makes it feel like a more intricate real-life problem, which is more interesting to me than a dataset intended for learning with only 2 or 3 classes.

By separating the target variable, our featuresRF data frame is ready to go for clustering as well.

Model Creation: Linear Regression Model

We will first create a baseline linear regression model.

In [15]:
```
X_trainL, X_testL, y_trainL, y_testL = train_test_split(featuresLR_S, targetLR,
```

In [16]:
```
modelL = LinearRegression()
modelL.fit(X_trainL, y_trainL)
```

Out[16]:
```
▾ LinearRegression
LinearRegression()
```

In [17]:
```
modelL.score(X_testL, y_testL)
```

Out[17]:
```
0.45644990717469625
```

The baseline model has an R2 score of 0.46, meaning that 46% of the variance in review scores can be explained using our feature variables.

In [19]:
```
coef = pd.DataFrame(zip(featuresLR.columns, modelL.coef_))
coef.head(13)
```

Out[19]:

	0	1
0	ABV	1.55
1	IBU	0.37
2	Astringency	0.12
3	Body	0.60
4	Alcohol	-0.70
5	Bitter	-0.16
6	Sweet	-0.73
7	Sour	-0.61
8	Salty	-0.05
9	Fruits	0.54
10	Hoppy	0.26
11	Spices	0.40
12	Malty	0.51

We can see, looking at the coefficients, which features are positively or negatively correlated with our target variable. We can see most are positive, but there are negative correlations between alcohol taste, sweet taste, and sour taste with overall review scores.

Selecting Best Features

We will be able to reduce dimensionality and clean up our model by selecting only the best features for our model. We will set up a pipeline that will determine the best value of k for us, which will let us know how many features to select.

In [20]:
```
pipeline = Pipeline(
    [
     ('selector',SelectKBest(f_regression)),
     ('model',LinearRegression())
    ]
)
```

In [21]:
```
search = GridSearchCV(
    estimator = pipeline,
    param_grid = {'selector__k': range(56)},
    n_jobs=-1,
    scoring="r2",
    cv=5,
    verbose=3
)
```

In [22]:
```
search.fit(featuresLR_S, targetLR)
search.best_params_
```

```
Fitting 5 folds for each of 56 candidates, totalling 280 fits
```

Out[22]:
```
{'selector__k': 11}
```

The grid search shows us that the optimal value is k=11. This means that our optimal model will take into account only the 11 most useful variables.

Creating an Updated Model

In [23]:
```
fs = SelectKBest(score_func=f_regression, k='all')
fs.fit(X_trainL, y_trainL)

X_train_fs = fs.transform(X_trainL)
X_test_fs = fs.transform(X_testL)
```

In [24]:
```
modelL2 = LinearRegression()
modelL2.fit(X_train_fs, y_trainL)
```

Out[24]:
```
▾ LinearRegression
LinearRegression()
```

In [25]:
```
modelL2.score(X_test_fs, y_testL)
```

Out[25]:
```
0.4565277310363943
```

Our second model takes into account only the 11 most pertinent features. This does not significantly increase our R2 score. However, our model is now much less complex and renders easier to analyze.

Feature Scoring

We can take a look at the scores our selector assigned to each feature to see which ones were the most important in determining review score.

```
In [26]: kfeat = pd.DataFrame(zip(featuresLR.columns, fs.scores_))
         kfeat.set_index(0, inplace=True)
         kfeat.sort_values(by=[1], ascending=False).head(11)
```

Out[26]:

0	1
Style_Lager	366.81
IBU	231.87
Body	205.73
Fruits	141.47
Style_Low Alcohol Beer	124.02
Bitter	123.13
ABV	122.77
Sour	95.15
Malty	94.13
Sweet	84.73
Hoppy	51.11

The scores show us that a beer being classified as a lager is the most informative feature to our target variable. The only other style that was selected for modeling is whether a beer is 'low alcohol' or not. This style is unique as it designates that a beer was designed to have the taste of a beer but with no (or very little) alcohol content. IBU, or International Bitterness Units, is also a very informative factor for modeling purposes.

Analysis

Unforunately, our tuned linear regression model was only able to explain 46% of the variance in our target variable: review score. This leads me to believe that there must be other factors not accounted for in our dataset that would be useful to finding an answer to this business problem. It is possible that there might be interaction terms at play, or maybe personal differences between consumers are just too large to make a blanket statement regarding what makes a well-reviewed beer.

We are at least able to identify which qualities are most important in determining review score. We can see that the model rated Style_Lager as the most important feature. IBU, a beer's body, as well as a fruity flavor are all influential as well. Of all of the flavor profiles, fruity had the highest score determined by the selector.

Clustering Models

We will next take a look at our second business problem: "Can a clustering model group together the beers in a meaningful way to offer recommendations?". To answer this we will use a K-Means Clustering algorithm.

Standardizing Values

In [28]:
```
scaler = StandardScaler()
feat_c = scaler.fit_transform(featuresRF)
```

Principal Component Analysis

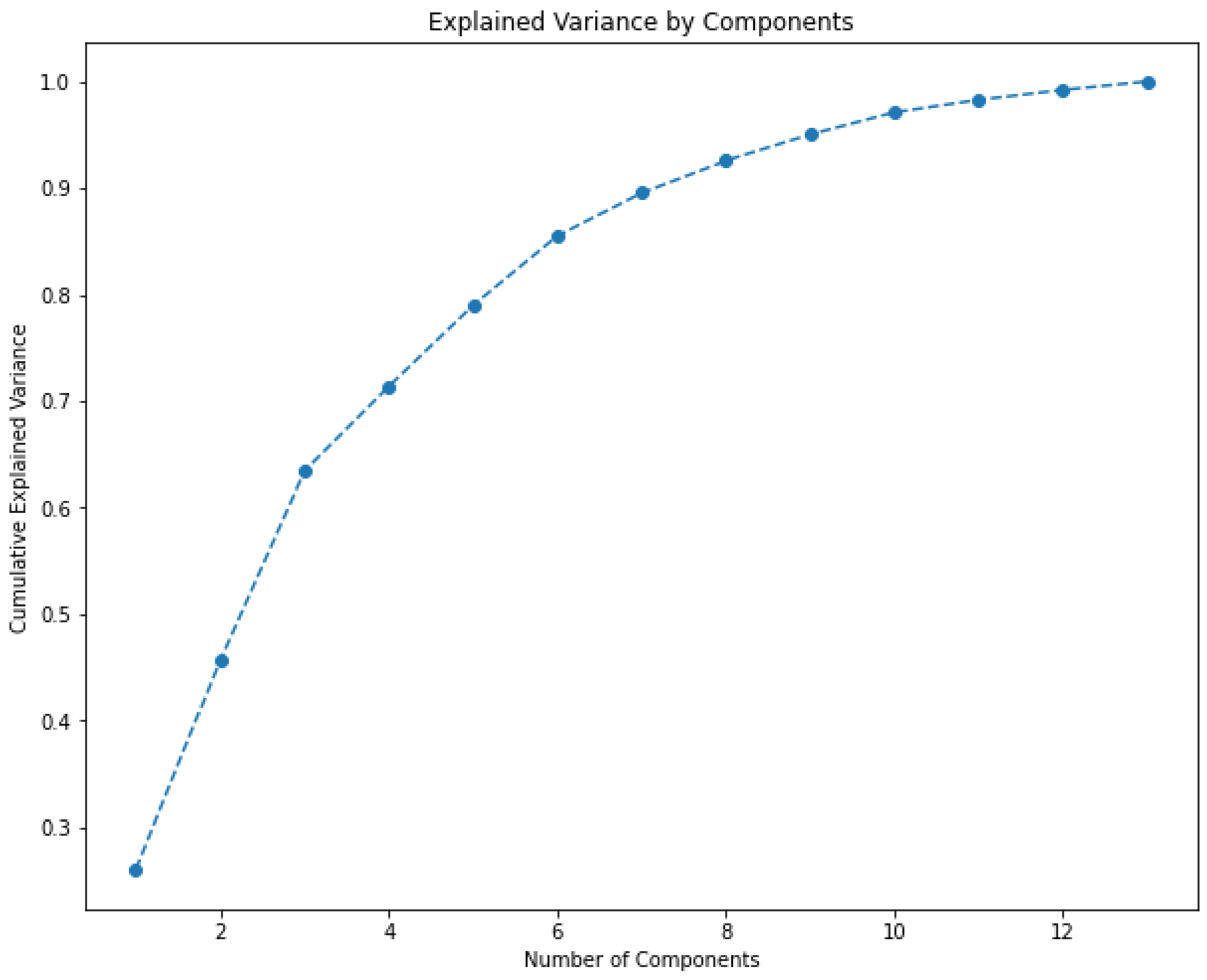
We will use PCA to reduce dimensionality of our data set.

In [29]:
```
pca = PCA()
pca.fit(feat_c)
```

Out[29]:
```
▾ PCA
PCA()
```

In [31]:
```
plt.figure(figsize = (10,8))
plt.plot(range(1,14), pca.explained_variance_ratio_.cumsum(),
         marker ='o', linestyle='--')
plt.title('Explained Variance by Components')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
```

Out[31]:
```
Text(0, 0.5, 'Cumulative Explained Variance')
```



Looking at the plot, we see that there is a slight but noticeable decline in explained variance at components = 6. We will use this information in order to reduce dimensionality to only 6 components rather than the current 14 features in our dataset.

In [39]:
```
pca = PCA(n_components=6)
pca.fit(feat_c)
pca.transform(feat_c)

scores_pca = pca.transform(feat_c)
```

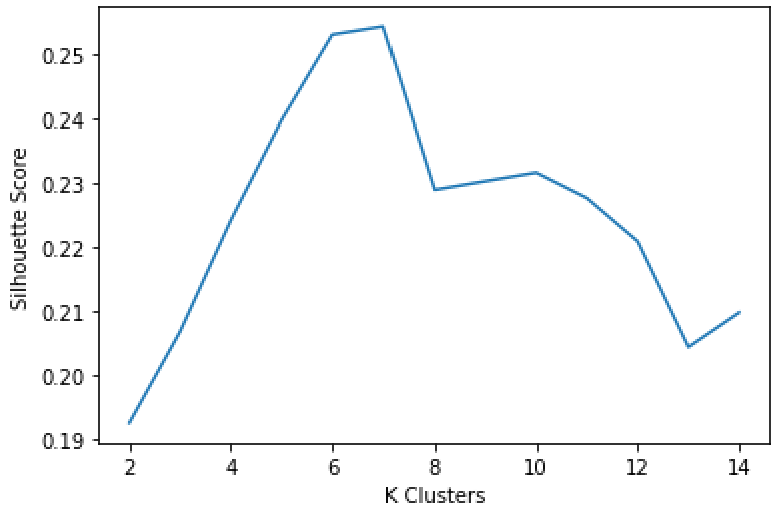
Determining K

In [42]:
```
silhouette = []
kx = []
kmax = 14

for k in range(2, kmax+1):
    kx.append(k)

for k in range(2, kmax+1):
    kmeans = KMeans(n_clusters = k).fit(scores_pca)
    labels = kmeans.labels_
    silhouette.append(silhouette_score(scores_pca, labels, metric = 'euclidean'
```

In [43]:
```
plt.plot(kx, silhouette)
plt.xlabel("K Clusters")
plt.ylabel("Silhouette Score")
plt.show()
```



This plot shows us the silhouette score for each possible value of k. We find the highest peak at k=7, showing that the optimal number of clusters for this dataset is 7. These visualizations provided us with the information necessary to effectively tune our hyperparameter k.

Creating the Model

In [44]:
```
kmeans7 = KMeans(n_clusters=7, random_state=23).fit(scores_pca)
kmeans7.fit(scores_pca)
```

Out[44]:
```
▾                KMeans
KMeans(n_clusters=7, random_state=23)
```

Appending Cluster Membership to DF

In [45]:
```
beer_df['Cluster'] = kmeans7.labels_
beer_df.head()
```

Out[45]:

	Style	ABV	IBU	Astringency	Body	Alcohol	Bitter	Sweet	Sour	Salty	Fruits	Hoppy
0	Altbier	5.30	31.50	13	32	9	47	74	33	0	33	57
1	Altbier	7.20	31.00	12	57	18	33	55	16	0	24	35
2	Altbier	5.00	32.00	14	37	6	42	43	11	0	10	54
3	Altbier	8.50	31.50	13	55	31	47	101	18	1	49	40
4	Altbier	7.20	37.50	25	51	26	44	45	9	1	11	51

Differences Between Clusters

In [47]:
```
beer_df.groupby('Cluster').mean()
```

Out[47]:

Cluster	ABV	IBU	Astringency	Body	Alcohol	Bitter	Sweet	Sour	Salty	Fruits	Hoppy
0	6.54	28.93	29.93	35.45	14.04	15.65	75.57	112.56	1.17	92.10	27.20
1	6.08	27.18	14.36	71.55	13.52	53.51	69.59	16.72	0.42	21.22	41.42
2	5.22	17.77	10.37	24.27	7.95	16.67	30.39	16.41	0.45	19.50	21.81
3	10.31	32.58	10.72	55.73	48.48	27.72	101.06	32.16	0.40	56.58	25.63
4	6.73	24.63	11.80	53.91	18.71	25.77	50.48	18.80	0.95	36.67	22.96
5	6.91	41.79	20.71	43.23	15.41	71.21	51.87	46.69	0.89	57.04	97.48
6	5.07	29.04	26.47	33.95	9.23	37.00	35.20	24.75	4.25	25.43	57.96

Looking at the descriptive statistics for our clusters, we can see the traits that differentiate them. Cluster 3 has a significantly higher ABV and presence of fruity flavors than the other clusters. Cluster 2 has a significantly lower IBU and fruity flavors than our other clusters. Cluster 1 has a much higher malty flavor than our other beers and ranks quite high in bitterness as well.

Clustering Report

In [72]:
```
cluster_report(featuresRF, beer_df['Cluster'], min_samples_leaf=100, pruning_le
```

	class_name	instance_count	rule_list
5	0	325	[0.7061855670103093] (Malty <= 86.5) and (Sour > 60.5)
0	1	769	[0.7674698795180723] (Malty > 86.5) and (Alcohol <= 30.5) and (Hoppy <= 80.5)
1	2	739	[0.7777777777777778] (Malty <= 86.5) and (Sour <= 60.5) and (Hoppy <= 49.5) and (Alcohol <= 20.5) and (Spices <= 40.5)
2	3	432	[0.5674740484429066] (Malty <= 86.5) and (Sour <= 60.5) and (Hoppy <= 49.5) and (Alcohol > 20.5) [0.7303370786516854] (Malty > 86.5) and (Alcohol > 30.5)
6	4	184	[0.7227722772277227] (Malty <= 86.5) and (Sour <= 60.5) and (Hoppy <= 49.5) and (Alcohol <= 20.5) and (Spices > 40.5)
3	5	389	[0.6053639846743295] (Malty <= 86.5) and (Sour <= 60.5) and (Hoppy > 49.5) and (Salty <= 1.5) [0.7008547008547008] (Malty > 86.5) and (Alcohol <= 30.5) and (Hoppy > 80.5)
4	6	359	[0.7563451776649747] (Malty <= 86.5) and (Sour <= 60.5) and (Hoppy > 49.5) and (Salty > 1.5)

This report breaks down the rules for membership in each of our clusters. Cluster 0 seems to be beers that are sour and low in malt. Cluster 1 is comprised of beers that are malty, yet low in alcohol flavor and low in hoppiness. Cluster 2 is made up of beers that are low in most flavor profiles, meaning they are milder in flavor. Cluster 3 has two paths to membership, both requiring a high level of alcohol flavor. Cluster 4 contains beers that are high in spicy flavors. Cluster 5 is comprised of beers that are hoppy and sometimes malty, but not salty. Lastly, Cluster 6 contains beers that are hoppy, not malty but indeed salty.

The classifications are definitely fascinating. Some make a lot of sense, such as C0 with all sour beers and C4 with spicy low-alcohol beers. However, the last two categories that are both hoppy but decided mainly by salt flavor seems like a bit of a strange distinction to me.

Analysis

After creating the clusters, I think the model did a great job at finding similarities between the beers in the dataset. Some of them seem similar to the ways humans would distinguish beer profiles, and some seem strange in practice. The clusters seem to be distinct and do not have really any overlap. The descriptive statistics show us the differences in each of our beer qualities and flavor profiles so we can really understand where the decisions came from.

Classification Models

For the last set of models, we will conduct a thought experiment. I am interested to see if we can use the beer's properties to create a model that can predict a beer's style. With the ultimate goal of creating a Random Forest Classifier, we will begin by creating a Decision Tree Classifier to get a closer look at the classification process before introducing an ensemble model.

Resample Data

Our dataset has 43 different possible values for our target variable: 'Style'. During the data cleaning process, we were able to consolidate the styles into broader groups, however, there are still a few groups that have <50 samples. We will oversample these smaller categories to ensure that they have representation in both our training and test sets when they are split.

In [55]:
```
ros = RandomOverSampler(random_state=23, sampling_strategy='minority')
features_rs, target_rs = ros.fit_resample(featuresRF, targetRF)
```

Split into Train/Test Set

In [56]:
```
X_trainD, X_testD, y_trainD, y_testD = train_test_split(features_rs, target_rs,
```

Decision Tree Classifier

In [57]:
```
dclf = DecisionTreeClassifier()
dclf = dclf.fit(X_trainD,y_trainD)
```

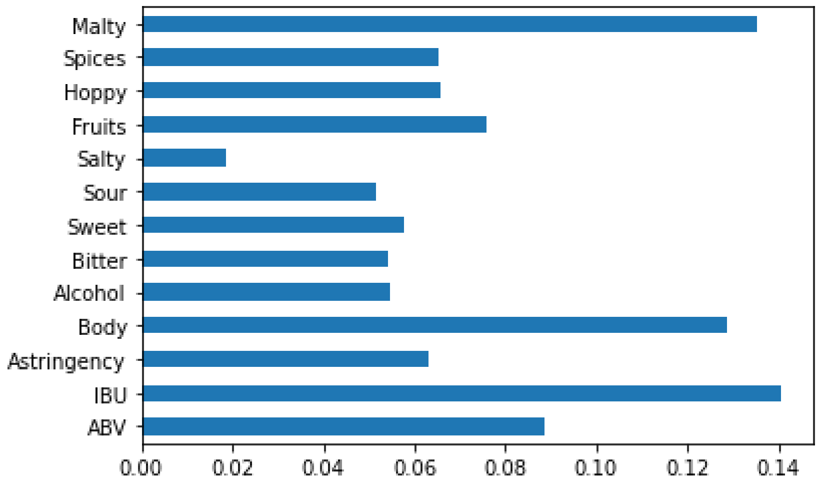
In [58]:
```
dclf.score(X_testD,y_testD)
```

Out[58]:
```
0.5287865367581931
```

Our initial Decision Tree Classifier is able to accurately classify a beer 53% of the time. This is fairly impressive for our base model, considering our target variable 'Style' has 43 different unique values.

In [59]:
```
imp = pd.Series(dclf.feature_importances_, index=X_trainD.columns)
imp.plot(kind='barh')
```

Out[59]:
```
<AxesSubplot:>
```



This chart shows us the most important features in classifying a beer's style. IBU (International Bitterness Units) is the most notable feature, followed by maltiness and the beer's body.

Visualizing the Tree

In [60]:
```
col = X_trainD.columns.tolist()

txt = export_text(dclf, feature_names=col, max_depth=3)
print(txt)
```

```
|--- Malty <= 27.50
|   |--- Body <= 32.50
|   |   |--- Sour <= 16.50
|   |   |   |--- IBU <= 7.25
|   |   |   |   |--- truncated branch of depth 9
|   |   |   |--- IBU >  7.25
|   |   |   |   |--- truncated branch of depth 11
|   |   |--- Sour >  16.50
|   |   |   |--- Malty <= 11.50
|   |   |   |   |--- truncated branch of depth 6
|   |   |   |--- Malty >  11.50
|   |   |   |   |--- truncated branch of depth 11
|   |--- Body >  32.50
|   |   |--- IBU <= 17.25
|   |   |   |--- Astringency <= 25.00
|   |   |   |   |--- class: Lambic
|   |   |   |--- Astringency >  25.00
|   |   |   |   |--- class: Brett Beer
|   |   |--- IBU >  17.25
|   |   |   |--- ABV <= 7.25
|   |   |   |   |--- truncated branch of depth 6
|   |   |   |--- ABV >  7.25
|   |   |   |   |--- truncated branch of depth 6
|--- Malty >  27.50
|   |--- Body <= 70.50
|   |   |--- Fruits <= 24.50
|   |   |   |--- IBU <= 34.25
|   |   |   |   |--- truncated branch of depth 19
|   |   |   |--- IBU >  34.25
|   |   |   |   |--- truncated branch of depth 10
|   |   |--- Fruits >  24.50
|   |   |   |--- IBU <= 18.75
|   |   |   |   |--- truncated branch of depth 10
|   |   |   |--- IBU >  18.75
|   |   |   |   |--- truncated branch of depth 19
|   |--- Body >  70.50
|   |   |--- IBU <= 32.25
|   |   |   |--- Sweet <= 99.50
|   |   |   |   |--- truncated branch of depth 11
|   |   |   |--- Sweet >  99.50
|   |   |   |   |--- truncated branch of depth 6
|   |   |--- IBU >  32.25
|   |   |   |--- Sour <= 27.50
|   |   |   |   |--- truncated branch of depth 9
|   |   |   |--- Sour >  27.50
|   |   |   |   |--- truncated branch of depth 8
```

This visualization shows us the top 3 levels of our decision tree model. We can see the first node splits our data depending if the beer has a maltiness rating greater than 27.5 or not. We then get further splits depending on the beer's body, which have different qualification limits depending on the maltiness rating.

Random Forest Classifier

Now that we have a bit of an idea of what a classifier looks like for this problem, we can build an ensemble model that will combine the efforts of many lower level models.

In [61]:
```
rf = RandomForestClassifier()
rf.fit(X_trainD, y_trainD)
```

Out[61]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [62]:
```
rf.score(X_testD,y_testD)
```

Out[62]:
```
0.6616474756421612
```

Our Random Forest Classifier is able to classify beer style with a 66% accuracy. Using the qualities in the dataset, we can determine a beer's style to a high level of precision (distinguishing stouts vs. porters, sours vs. wild ales) with a 66% accuracy which I consider to be a success. I believe that if we truncated some of the rarer styles from the dataset, we could create a model with an impressive accuracy.

In [63]:
```
y_predD = rf.predict(X_testD)
print(classification_report(y_predD, y_testD))
```

```
                                precision    recall  f1-score   support

                       Altbier       0.14      0.50      0.22         4
                    Barleywine       0.85      0.59      0.69        29
                        Bitter       0.48      0.53      0.50        19
Bière de Champagne / Bière Brut       0.00      0.00      0.00         0
                   Blonde Ale       0.17      0.33      0.22         9
                         Bock       0.56      0.49      0.52        51
                      Braggot       0.00      0.00      0.00         1
                   Brett Beer       1.00      1.00      1.00       175
                    Brown Ale       0.27      0.80      0.40        10
 California Common / Steam Beer       0.33      1.00      0.50         2
                   Chile Beer       1.00      0.86      0.92         7
                    Cream Ale       0.11      1.00      0.20         1
                       Dubbel       0.86      0.35      0.50        17
                Farmhouse Ale       0.27      0.55      0.36        11
         Fruit and Field Beer       0.67      1.00      0.80         6
    Gruit / Ancient Herbed Ale       0.20      1.00      0.33         1
                     Happoshu       0.80      0.57      0.67         7
          Herb and Spice Beer       0.00      0.00      0.00         0
                          IPA       0.91      0.70      0.79        70
                        Kvass       1.00      1.00      1.00         5
                       Kölsch       0.38      0.30      0.33        10
                        Lager       0.77      0.58      0.66       238
                       Lambic       0.82      0.71      0.76        38
            Low Alcohol Beer       1.00      1.00      1.00        11
                     Mild Ale       0.13      1.00      0.24         2
                      Old Ale       0.12      0.33      0.18         3
                     Pale Ale       0.41      0.38      0.39        29
                      Pilsner       0.52      0.68      0.59        19
                       Porter       0.58      0.49      0.53        59
                 Pumpkin Beer       1.00      0.79      0.88        14
             Quadrupel (Quad)       0.83      0.62      0.71         8
                      Red Ale       0.27      0.44      0.33        16
                     Rye Beer       0.17      0.50      0.25         2
        Scotch Ale / Wee Heavy       0.31      0.57      0.40         7
                 Scottish Ale       0.10      1.00      0.18         1
                  Smoked Beer       0.17      0.25      0.20         4
                         Sour       0.82      0.69      0.75        13
                        Stout       0.79      0.68      0.73        90
                   Strong Ale       0.62      0.49      0.55        57
                       Tripel       0.36      0.83      0.50         6
                   Wheat Beer       0.74      0.66      0.70        65
                     Wild Ale       0.00      0.00      0.00         2
               Winter Warmer       0.53      0.90      0.67        10

                     accuracy                           0.66      1129
                    macro avg       0.49      0.61      0.49      1129
                 weighted avg       0.74      0.66      0.69      1129
```

The classification report shows us the precision, recall, and F1 score for each class in our dataset. We can see that in this model, a number of our smaller categories (cream ale, mild ale, gruit/ancient herbed ale) have high recall but low precision. This indicates that some of our rarer beer styles are largely being predicted correctly, but the categories are also getting lots of false positives. The weighted averages being significantly higher than the macro averages indicate to us that our larger categories are getting predicted pretty well, but we are struggling a bit with the smaller categories. A good next step would be to collect more data on beers in these smaller categories.

Hyperparameter Tuning - RandomizedSearchCV

We will take a look at some of the hyperparameter choices and see if we can fine tune this model to gain more predictive power.

In [64]:
```
param_grid = {
    'n_estimators': [25, 50, 100, 150],
    'max_features': ['sqrt', 'log2', None],
    'max_depth': [10, 25, 100, None],
    'max_leaf_nodes': [5, 10, None],
}
```

In [67]:
```
random_search = RandomizedSearchCV(RandomForestClassifier(),
                                   param_grid)
random_search.fit(X_trainD, y_trainD)
print(random_search.best_estimator_)
```

```
RandomForestClassifier(max_depth=100, max_features=None)
```

Our randomized search indicates that the optimal hyperparameter values are max_features = 'log2' with the rest of the values maintaining their default.

Best Model

In [68]:
```
model_rnd = RandomForestClassifier(max_depth=100,
                                      max_features=None,
                                      max_leaf_nodes=None,
                                      n_estimators=100)
model_rnd.fit(X_trainD, y_trainD)
y_pred_rnd = model_rnd.predict(X_testD)
print(classification_report(y_pred_rnd, y_testD))
```

```
                                 precision    recall  f1-score   support

                        Altbier       0.29      0.57      0.38         7
                     Barleywine       0.90      0.69      0.78        26
                         Bitter       0.48      0.59      0.53        17
Bière de Champagne / Bière Brut       0.00      0.00      0.00         0
                     Blonde Ale       0.17      0.27      0.21        11
                           Bock       0.53      0.47      0.50        51
                        Braggot       0.33      1.00      0.50         1
                     Brett Beer       1.00      1.00      1.00       175
                      Brown Ale       0.30      0.90      0.45        10
 California Common / Steam Beer       0.50      0.75      0.60         4
                     Chile Beer       1.00      0.86      0.92         7
                      Cream Ale       0.11      1.00      0.20         1
                         Dubbel       0.86      0.43      0.57        14
                  Farmhouse Ale       0.32      0.50      0.39        14
           Fruit and Field Beer       0.56      0.71      0.63         7
     Gruit / Ancient Herbed Ale       0.40      1.00      0.57         2
                      Happoshu       0.80      0.57      0.67         7
            Herb and Spice Beer       0.00      0.00      0.00         0
                            IPA       0.87      0.69      0.77        68
                          Kvass       1.00      0.71      0.83         7
                         Kölsch       0.50      0.29      0.36        14
                          Lager       0.76      0.62      0.68       222
                         Lambic       0.79      0.68      0.73        38
               Low Alcohol Beer       0.91      1.00      0.95        10
                       Mild Ale       0.40      0.86      0.55         7
                        Old Ale       0.25      0.67      0.36         3
                       Pale Ale       0.37      0.36      0.36        28
                        Pilsner       0.48      0.71      0.57        17
                         Porter       0.62      0.53      0.57        59
                   Pumpkin Beer       0.91      0.91      0.91        11
               Quadrupel (Quad)       1.00      0.60      0.75        10
                        Red Ale       0.31      0.38      0.34        21
                       Rye Beer       0.33      1.00      0.50         2
         Scotch Ale / Wee Heavy       0.38      0.56      0.45         9
                   Scottish Ale       0.20      0.67      0.31         3
                    Smoked Beer       0.17      0.20      0.18         5
                           Sour       0.82      0.75      0.78        12
                          Stout       0.82      0.72      0.76        88
                     Strong Ale       0.58      0.49      0.53        53
                         Tripel       0.36      0.62      0.45         8
                     Wheat Beer       0.76      0.73      0.75        60
                       Wild Ale       0.33      0.25      0.29         8
                 Winter Warmer       0.71      1.00      0.83        12

                       accuracy                           0.68      1129
                      macro avg       0.54      0.63      0.55      1129
                   weighted avg       0.73      0.68      0.69      1129
```

Our final model increases slightly to a 68% accuracy rate. Looking at the precision and recall scores we can see an interesting difference. We actually have a decrease in the weighted average for precision, but an increase in the macro average for both precision and recall. This tells us that this model is better at classifiying the smaller classes of beers. We are receiving less false positives in some of the categories we discussed earlier (mild ales and gruit/ancient herbed ales).

In [70]:
```
pd.DataFrame({'Variable':col,
              'Importance':model_rnd.feature_importances_}).sort_values('Import
```

Out[70]:

	Variable	Importance
1	IBU	0.16
12	Malty	0.12
3	Body	0.11
0	ABV	0.08
9	Fruits	0.08
2	Astringency	0.07
11	Spices	0.07
7	Sour	0.06
10	Hoppy	0.06
6	Sweet	0.06
5	Bitter	0.06
4	Alcohol	0.05
8	Salty	0.02

Looking at the feature importance, we see similarities between the feature ranking for our classification model and our clustering model. IBU shows to be the top ranked feature once again, with maltiness coming in right behind it.

Analysis

We are able to classify beer to an impressive degree considering how specific some of the style categories are. I think the model could be greatly improved if we had larger sample sizes for the rarer categories. I would love to collect further data, but unfortunately Untappd has closed their API to new users at the moment. Our model is really great at predicting certain styles (such as pumpkin beers and chile beers) that have distinct flavor profiles. It struggles with other styles (smoked beers, bitter beers) that have a less defined set of flavors.

Recommendations

Our linear regression model indicates to breweries which qualities are important in determining whether a consumer will enjoy a beer. Having a higher IBU, ABV, a fuller body and a fruity flavor all are predictors of a higher review score. However, having an alcohol taste or a sweet taste are indicators that a beer will have lower review scores. These recommendations can assist brewers in the process to add flavors that consumers enjoy. Having higher review scores may entice more patrons to find the business (possibly on a platform such as Yelp or Untappd where they can read reviews and see photos).

Our clustering model is great for a few different purposes. Consumers can use it in order to find beers that are similar to their current favorites. Bartenders can also use it in order to recommend beers to their guests. If a guest is looking for a sour beer, you can easily pull all of the information from Cluster 0 and see what options you have. It is also particularly interesting because it ignores man-made labels, so beers that are considered 'sours' might not fall into this category if their sour flavor profile is not strong enough compared to other traits.

Our classification model was a good experiment, and is a fun way to brainstorm when creating flavors. If you are imagining creating a beer using mangoes, wheat and lactose, you can input those flavor profiles into the model and it will let you know what styles typically have that profile. That can assist in choosing what other flavors might go well with it, or determine the way you brew it. If the model determines a wheat beer is the best fit, you would probably use a different brewing style than if it classified the beer as an IPA.

Ethical Implications

This dataset does not include any personal data regarding the consumers who reviewed the beers. There are very few inherent risks as it contains strictly anonymous product reviews. There is possible concern for less-reviewed breweries, as the low sample size may skew the perception of a brewery. If there is a single negative review, it might not fully represent the views of their customers and might provide “bad press” so to speak for the brewery. I intend to take a look at anything with a very small sample size and tread carefully to avoid single pieces of data skewing my model and results.

Milestone 3

Data Quality

The two questions I will be focusing are: 1) What qualities are most highly correlated with a positively reviewed beer? And, 2) Can a clustering model group together the beers in a meaningful way to offer recommendations? What features, or components, are most relevant in the creation of the clusters?

My dataset has all of the qualities I need to create my two models and answer each of my questions. The dataset has no missing values, as each row represents one beer and combines all of its reviews into one observation. I checked for outliers as well and there was nothing that seemed out of line for any of our variables. I removed a couple of redundant columns, there were 3 separate columns that listed the beer’s name, the brewery, and then the combination of both. There was also a range for IBU (International Bitterness Units, the measure of a beer’s bitterness), so I consolidated it to the mean of the range so it was a singular variable. For the purposes of the linear regression, the only additional step I will need to complete is to create dummy variables for each of the ‘Style’ options.

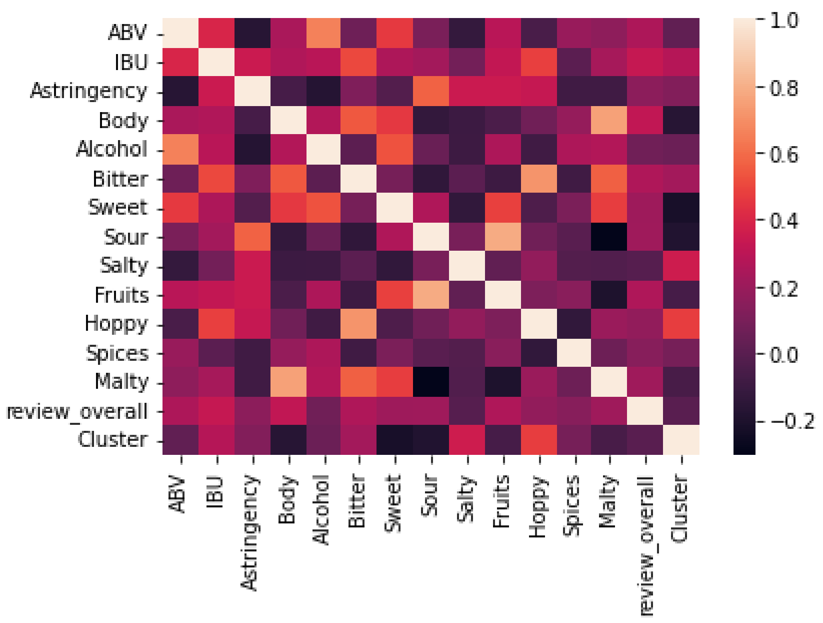
For the clustering model, I created another dataframe, removing a couple other columns. I removed the “Style” column. “Style” is a manmade label for types of beers (IPA, stout, sour, etc.), so I felt it would have undue influence on the model. I would like the model to cluster the data without the guidance of existing style labels. I also removed the review scores after some trial and error. My first K-Means model ended up clustering the beer by only positive review vs. negative review. This data did not seem to genuinely be helpful to create meaningful clusters, so I removed it in favor of only using inherent physical traits of the beer.

Visualizations

Before diving into creating a regression model, it is worth taking a look at the correlation matrix to see what relationships the features have.

```
In [76]: sns.heatmap(beer_df.corr())
```

```
Out[76]: <AxesSubplot:>
```



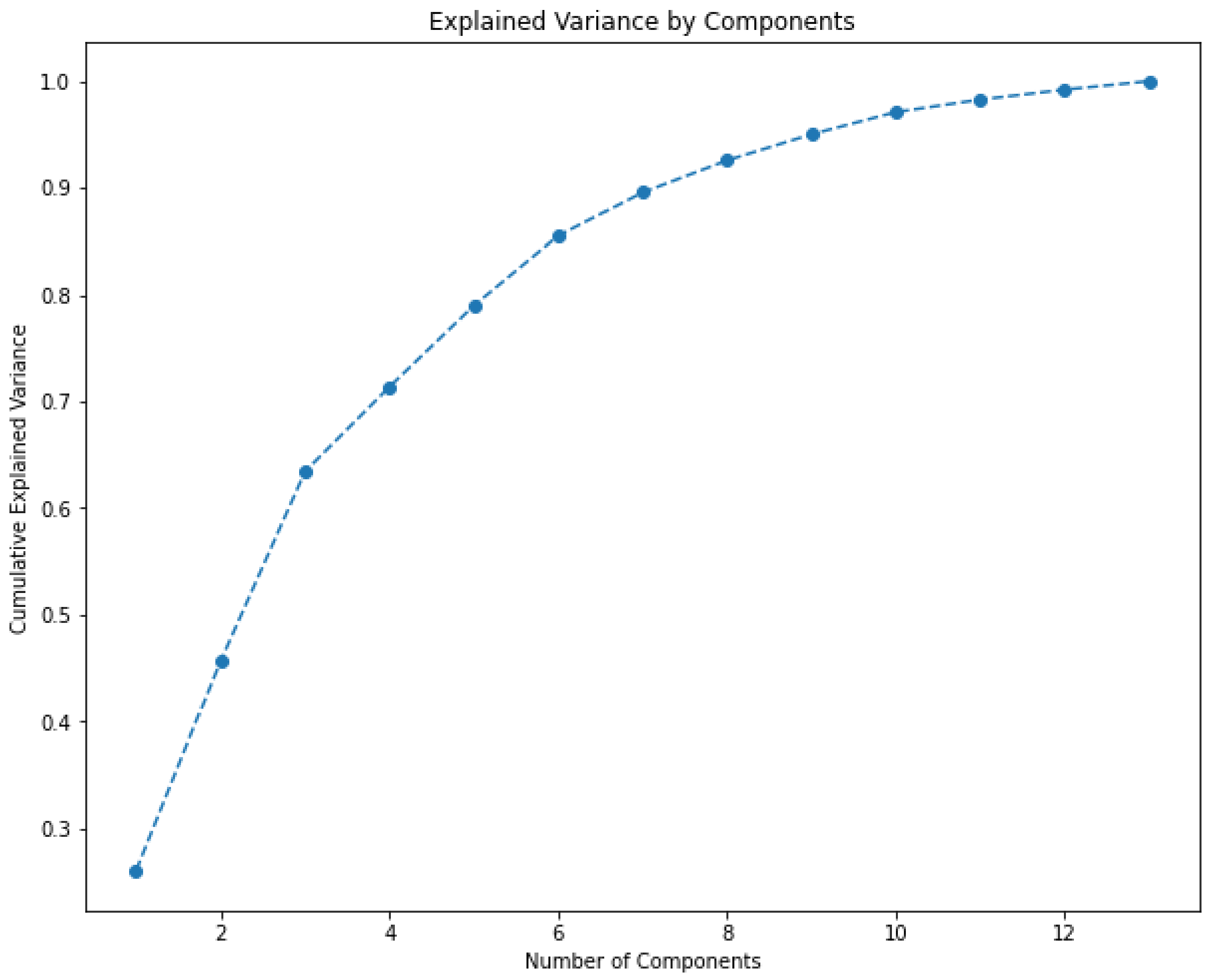
This matrix shows us the correlations between each of our variables. We can see that each variable is 100% correlated with itself in red along the diagonal axis. Our target variable, ‘review_overall’ looks like it is actually positively correlated with every feature variable (even if the correlation is small). There is a high correlation between ABV and the beer having an ‘alcohol’ taste, which makes sense as it would be more prominent. There is also a high correlation between a beer being full-bodied (having a high ‘body’ rating) and the ‘malty’ flavor profile. Body of a beer can be enhanced by adding “caramelized and roasted malts” so this is expected from a brewing perspective (Smith).

When working with our clustering model, we will need to create a couple of visualizations to help us define our hyperparameters. Firstly, we will need to determine the ideal number of components for our PCA. To determine this value, we will create a plot that shows us the cumulative explained variance of each additional component.

In [78]:
```
pca = PCA()
pca.fit(feat_c)

plt.figure(figsize = (10,8))
plt.plot(range(1,14), pca.explained_variance_ratio_.cumsum(),
         marker ='o', linestyle='--')
plt.title('Explained Variance by Components')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
```

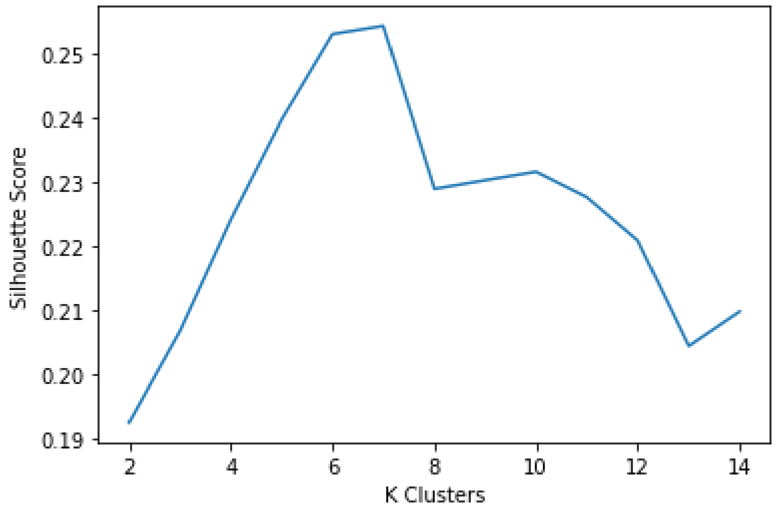
Out[78]:
```
Text(0, 0.5, 'Cumulative Explained Variance')
```



Looking at the plot, we see that there is a slight but noticeable decline in explained variance at components = 6. We will use this information in order to reduce dimensionality to only 6 components rather than the current 14 features in our dataset.

The next visualization we need to make is a silhouette score plot. This will give us the information we need in order to determine the ideal value for k for our K-Means Clustering Model.

In [79]:
```
plt.plot(kx, silhouette)
plt.xlabel("K Clusters")
plt.ylabel("Silhouette Score")
plt.show()
```



This plot shows us the silhouette score for each possible value of k. There is a noticeable peak at k=7, showing that the optimal number of clusters for this dataset is 7. These visualizations provided us with the information necessary to effectively tune our hyperparameters to create the best possible clustering model.

Adjustments to Data, Questions & Model

In Milestone 2, I was unsure of what models would be best suited for my data. I have fine-tuned my business problem to specifically aim for a classification model that would be able to recommend beers to consumers based on their current beer preferences. I will still also be conducting a simple Linear Regression model to satisfy my curiosity regarding what elements of a beer are most strongly tied to positive reviews. I will be creating both a supervised model, in the form of a Linear Regression, and an unsupervised model, in the form of a K-Means Clustering model.

I did do a fair bit of adjusting to the data to make it workable for these problems. I converted the IBU range to a single number for analysis. I removed a number of nominal variables, such as the beer's name and brewery's name. I simplified the 'Style' variable so sub-styles were all included together. This variable has been removed from the clustering model because I do not want human-created labels to influence the clusters the model is able to create. I also removed the review scores from the clustering model since they were making the results cloudy. However, the review score is my target variable for the regression model, so it does have its place in my analysis.

Expectations

My original expectations are reasonable. At first I was not entirely confident I would be able to create a clustering model, but I have been working on it over the last week and it is coming along nicely. Once I was able to determine the initial problem was the inclusion of the review score variables, I removed them and ended up with a much more robust model that categorizes beers based on their flavor profiles. I aim to take a look at each style of beer to see what clusters they are by cross-referencing the determined clusters with the original dataset. Because there are 43 styles (even after I consolidated substyles), I would not expect it to be able to distinguish each and every one. However, I am hoping that it is able to cluster the beer styles in a way that is logical, where beers from a singular style are clustered together.

Milestone 2

Problem Statement

I will be looking to answer two questions during my analysis. The first: what makes a good beer? Which qualities are most important to determining whether consumers will enjoy a certain brew and rate it highly? The second: can we use this data to create a clustering model that acts as a recommendation system for new beers based on their profiles and consumer’s preferences?

Significance of the Problem

This is an interesting problem not only for breweries who are looking to develop new flavors for their patrons, but also to both causal and dedicated beer drinkers. Larger breweries, such as Anheuser-Busch, may not necessarily find this information useful in creating profit as they have a very successful business model based on brewing a variety of many light, simple, similar tasting beers. However, smaller breweries who are focused more on small-batch brews with fun, interesting flavor profiles will find this information incredibly useful to determining the ideal flavor profiles for different types of beers.

Avid beer drinkers will also find this information to be interesting, as it provides a bird’s eye view of the craft beer industry. There are a number of very niche beer styles that might be new to them, introducing them to new types of beer or breweries. It also allows the audience to see the popularity of different styles, and whether certain flavor profiles were linked to positive or negative reviews. For example, “hoppy” might have a positive correlation with reviews of IPA beers, but a negative correlation with wheat beers. These nuances provide us valuable insight on the differences in expectation and execution of different types of beers.

By creating clusters of beers based on the data, we can see which beers are most similar to each other. This can assist bartenders in recommending new beers to patrons based on their tastes. It can also help with marketing the right beers to the right audience. As a consumer, it can allow you to see what beverages have similar qualities to the ones you enjoy and explore new breweries.

The Data

The dataset I will be using for this project is titled “Beer Profile and Ratings Data Set” and it is sourced from Kaggle. This dataset explores reviews of a number of craft beers, with detailed information about their taste profiles. The dataset has 25 columns, including ABV, IBU, and flavor notes of the brew. There are 3197 rows, each presenting information pertaining to an individual beer. Each of these features will give insight into consumer preferences and specific notes for each beer.

Models

I intend to create two models, one to solve each of my problem statements. The first model will be a linear regression model to determine which factors are most heavily linked to positive beer reviews. The second model I intend to create is a K-Means Clustering algorithm that will group beers based on the features in the data set (ABV, IBU and flavor notes such as “malty”, “fruity”, “spicy”).

Evaluating Results

To evaluate the linear regression model, I will split my data between a training and test set to ensure that the model is independent of the data it will be using during the evaluation process. When evaluating a linear regression model, the R2 score is likely the best metric to utilize. The R2 metric calculates the proportion of variance in our target variable that can be explained by the feature variables. The resulting score demonstrates the fit of the regression between 0 and 1, making it ideal for comparing linear models.

To evaluate the K-Means Clustering model, I will simulate a model with a number of different k values to see which version of the model has the highest Silhouette Score. This score will provide me with an understanding of how distinct the clusters are. This metric takes into account the mean distance between items in one cluster as well as the mean distance of the nearest cluster. With scores ranging from -1 to 1, a high score indicates that the clusters created by the model are well-defined and independent. Since K-Means Clustering is an unsupervised technique, there is no target variable to measure accuracy against.

Takeaways

From a less technical standpoint, I hope to learn more about the subtle differences between different types of beer. I would love to see results that support the idea that small breweries creating artisan brews can appeal to audiences better than large corporations who mass-produce flavorless beverages. I am also curious as to what factors are most highly linked to positively reviewed beers. There are certain flavors, such as the hoppiness of an IPA or the tartness of a sour beer, that consumers have very polarized opinions about.

On the technical side, I am interested to see what factors the model will determine are most important to clustering beers. Which features, or components, will provide us the clearest clusters? Will they be broken up by style (IPA, stout, sour, etc.) or will there be different driving factors? The style will of course be removed as a feature because it is a manmade label. Will the algorithm come up with similar labels?

Ethical Concerns

This dataset does not include any personal data regarding the consumers who reviewed the beers. There are very few inherent risks as it contains strictly anonymous product reviews. There is possible concern for less-reviewed breweries, as the low sample size may skew the perception of a brewery. If there is a single negative review, it might not fully represent the views of their customers and might provide “bad press” so to speak for the brewery. I intend to take a look at anything with a very small sample size and tread carefully to avoid single pieces of data skewing my model and results.

Contingency Plan

I am confident I will be able to create a Linear Regression model to determine what factors are best linked to positively reviewed beers. I am not quite as confident that I will be able to create a useful clustering model. If the clustering model falls through, I will rely on my Linear Regression model and go further in depth with models of that nature to fully flesh out my project. Even if the clustering model is not successful, it will be interesting to analyze why. We can determine what additional information might make it more successful in the future.

Other Notes

I am excited to delve into this data as craft beer is something that I am particularly interested in on a personal level. I have visited a number of breweries over the years throughout North America and have had the pleasure of being a patron at many of the breweries and taprooms in this dataset. I have my own data from Untappd that I would like to compare to some of the data to see if my information and flavor profiling matches up with that of the reviewers. This will not be something I use for my final project, but I believe that my passion for the subject will allow me to create a well thought out, useful model.

References

RuthGN. (2022, March). Beer Profile and Ratings Data Set, Version 8. Retrieved March 16, 2022 from https://www.kaggle.com/datasets/ruthgn/beer-profile-and-ratings-data-set

Smith, Brad. “Making Full Bodied Beer at Home.” BeerSmith Home Brewing Blog RSS, 27 Feb. 2008, https://beersmith.com/blog/2008/02/27/making-full-body-beer-at-home/.