Capstone Project Group 3:

Bassem Fadlia, David Brunik, Jared Allerson, James Cobb, and Suraj Rajak

Written Analysis

Introduction to the Topic:

Brewing beer isn't as straightforward as some might think. Sometimes a recipe for a beer turns out to be exactly what the brewer intended. In other cases, a beer ended up being wildly different than expected. Even at a commercial level, it's not uncommon for a brewery to pivot on the original plan for a batch of beer. Often, this pivot ultimately results in labeling a beer as different style than the one intended. At a homebrew level, most brewers are inexperienced when it comes to designing their own recipes. Their lack of experience is most noticeable when it comes to objectively thinking about beer. Drinking beer is a subjective experience, brewing beer and judging beer is objective. All beer styles have certain parameters that are used to distinguish themselves from other styles. Most of the time when homebrewers are trying to figure out what it is that they brewed, they go to their local homebrew club and sit in a circle sipping the beer and describing the flavors that they taste. We want to take a more analytical approach.

Brewing beer is a science that involves a lot of math, and frankly, a lot of data. Flavor is important, but so are the stats on a beer when it comes to objectively classifying it within a style. These stats include Original Gravity (OG), Final Gravity (FG), Alcohol Content (ABV), Standard Reference Method (SRM), and International Bitterness Units (IBU). Our goal is to use these parameters (along with flavor) to create a beer classifier. Users will be able to input the data on their batch of beer and then our model will compare it to the Beer Judge Certification

Program dataset. This will ultimately tell users what the closest style of beer their batch falls under. Then the model will use another dataset that contains thousands of known commercial beers within the US that are the same style. There's another dataset that has the top 250 commercial beers produced. The goal of including known and reputable beers will be for the users of our model to be able to go out on their own and purchase a beer of the same style to compare it to the beer that they made. Developing a beer recipe is all about trying other beers and finding things that you like and don't like and then adjusting from there.

Inspiration:

Craft beer sales have been increasing dramatically over the past decade (and even more so since the pandemic). Doing a project on beer sounded like a great idea since it is so relevant for most people. We wanted to have a project that would pique the interest of people in the class. The original idea came from the Beer Judge Certification Program (BJCP) mobile app. It's a fun and useful app to have on your phone! The app contains over a hundred recognized beer styles and explains their categories and parameters for the user. It makes it easy to understand more about a certain beer and compare the descriptive data to a beer that you are drinking. Eventually, we were able to find a dataset on this app on Kaggle! The dataset was perfectly organized and clean; we immediately knew we could use this to classify beer. Afterwards, we found a dataset containing over 75,000 homebrewer recipes from a recipe-builder website called Brewer's Friend. This final dataset ultimately led to the project becoming a craft beer style classifier for homebrewers.

Hypotheses or Expectations:

Users of our Craft Beer Style Classifier for Homebrewers will effectively be able to input parameters of OG, FG, ABV, SRM, and IBU into our app and the output will tell them what style that beer is.

<u>EDA Section:</u>

The exploratory data analysis for our datasets wasn't too hard. We had two main datasets initially: guidelines.csv (BJCP) and recipeData.csv (Brewer's Friend). Our first dataset guidelines.csv was already really organized. The data in the columns were all lined up and the same throughout the rows. The only thing we had to do was decide what columns were important and necessary for our analysis. We knew we needed the following columns: *#, Styles, Style Family, Origin, ABV min, ABV max, IBUs min, IBUs max, SRM min, SRM max, OG min, OG max, FG min,* and *FG max*. Most of these were numeric and were related to the parameters that we were going to use in our classifier. We chose to drop the following columns: *BJCP Categories* and *Style History*. The following columns are maybe options: *Overall Impression, Aroma, Appearance, Flavor, Mouthfeel, History, Characteristic Ingredients, Style Comparison,* and *Commercial Examples*. These could come in handy if we chose to display information about the style output on our website (see Appendix A).

Our second dataset recipeData.csv was huge and took a lot of time to organize. This dataset had over 75,000 rows of recipes in it and some columns had more than one type in it's categories. First step again was to find the columns that we wanted to use and ones that we didn't want. The following columns were the ones that we wanted: *BeerID, Name, Style, StyleID, OG, FG, ABV, IBU, Color, SugarScale,* and *BrewMethod*. SugarScale had two options, Plato and Specific Gravity. These are just two different types of ways to measure sugar (the same idea of how kilograms and pounds measure weight but in a different way). A vast majority of our data was in specific gravity, so we dropped all the rows that were done in plato. For BrewMethod, there was All Grain, BIAB, Partial Mash, and Extract. We dropped Extract because that is the only brewing technique that doesn't involve the use of grain. Extract recipes would be outliers since they're so far from the traditional method of brewing. The following are the columns that we didn't want: *URL, Size(L), BoilSize, BoilTime, BoilGravity, Efficiency, MashThickness, PitchRate, PrimaryTemp, PrimingMethod*, and *PrimingAmount*. These columns were mostly about steps in the brewing process, which isn't necessary for the parameters of our classifier (see Appendix B).

For our Craft Beer Style Classifier for Homebrewers, we used five parameters to filter our output. Our first parameter was **Original Gravity (OG)**. This numeric datapoint represents how much sugar is in your beer before fermentation (brewers extract sugars from grain). Our second parameter was **Final Gravity (FG)**. This numeric datapoint represents how much sugar is in your beer after fermentation. All beer styles have a certain range of where the FG should be. The higher the FG, the more body the beer is going to have. Aka, how thick the beer is due to the sugars left over in the beer. Thickness/body can best be described as how it feels in your mouth when you're drinking it. Orange juice for example is very thick compared to a glass of water. A light lager is going to have fewer leftover sugars than a stout for example. Our third parameter was **Alcohol Content (ABV)**. During fermentation, yeast turns those sugars into alcohol. How much alcohol is in your beer after fermentation is determined by measuring how much sugar is leftover; FG (final gravity). Attenuation is the process of comparing OG to FG to determine how much alcohol (ABV) is in your beer with a numeric value. Our fourth parameter was **Standard Reference Method (SRM).** This numeric value is North America's gold standard for measuring the color of your beer. Beer colors help distinguish styles from one another, such as a light-colored pilsner verses a dark-colored stout. These colors provide a visual experience for beer drinkers and enhances the drinking experience. Our fifth parameter was **International Bitterness Units (IBU).** During the brewing process, hops are added to beer during the boil. These hops create bitterness, which is used to help balance out sugars in a beer and can be measured as a numeric value. All beer styles have an ideal range for what their IBUs should be.

Database Section:

Once we had our data cleaned, we needed to store our data in a database. The advantage of using a database is that it reduces data redundancy to a large extent, facilitates sharing of data among users and ensures the security of the data. For our database, we chose to use SQLite because it is serverless, lightweight, and requires zero-configuration. This type of database is considered a Relational Database model. This type of model organizes the data into tables. We created four tables inside our Beer_Analysis.db (final_guidelines, final_recipes_dataset, final_style, and final_style_family). Essentially, we imported our csv files into each respective table that we created. These tables include data type, primary keys, and foreign keys. To visually see how we organized our data, check out our ERD in the Appendix section (see Appendix C).

ML Section:

For us to make our datasets usable under a Machine Learning module, we had to fix some overfitting issues with our data. There were over a 100 beer styles that we could use to classify a user input beer as, which is way too many and the accuracy would have been awful. So, our first step was to use categorization for beer style families. Our guidelines.csv had a column called Style Family and we created a numeric ID called Style_Family_ID. Our initial total count for our

Style_Family_ID was 15. These categories were: *Pale Lager, Pale Ale, Wheat Beer, Amber Ale, Porter, IPA, Brown Ale, Stout, Strong Ale, Sour, Bock, Pilsner, Dark Lager, Amber Lager,* and *Specialty Beer*. After that, we created a new CSV to host this new column called Style_Family.csv. Then we added the Style_Family_ID to the recipeData.csv (where we have the 75,000 recipes). Doing this, we had successfully connected our datasets together through Style Family (see Appendix D). We now had the ability to count how many of the recipes in recipeData.csv were in each of the Style_Family_ID categories. We ended up dropping 6 of the 15 categories because there were under 1,000 recipes brewed under these styles, which would make them outliers. These outliers were: *Sour, Bock, Pilsner, Dark Lager, Amber Lager,* and *Specialty Beer*. For our Machine Learning model, we initially had over a 100 beer style categories and we successfully got it down to 9.

Our next step was to figure out what machine learning model would work best for us. We imported pycaret.regression import * and ran a modeling algorithm that set the target to Style_Family_ID. Our model showed that Light Gradient Boosting Machine model would be our ideal method of machine learning. Our R^2 for that model was 0.9832 (98%) accurate (see Appendix E). To create the LightGBM we set our "y" to [['Style_Family_ID']] and we set our "X" to [['OG','FG','ABV','IBU','Color']] and did some split, test, and train. Our X_train and y_train model had a score of 0.76 (76%) accuracy. Our X_test and y_test model had a score of 0.65 (65%) accuracy. When we used our Light Gradient Boosting Machine model on our dataset, we were easily able to get an accuracy of 81% with some sample data (see Appendix F). Our machine learning model can successfully take user input that involves numeric data points (*OG, FG, ABV, IBU*, and *Color*) and classify that data as one of the nine Style_Family_IDs (*Pale Lager, Pale Ale, Wheat Beer, Amber Ale, Porter, IPA, Brown Ale, Stout,* and *Strong Ale).*

Tableau Section:

For our project, we created a total of two Tableau dashboards. The first dashboard contains a total of four interactive visuals about our beer datasets. The first visual is called "Beer count by Origin, Style Family and Brew Method" and uses our recipeData.csv. With this visual you can adjust the filter by changing which Style Family you want displayed. The Count of Beers shows the total number of homebrewer recipes for each Style Family in the recipeData.csv. This is useful because it can help you understand which Style Family is brewed most often by homebrewers. It's easy to see that homebrewers love making IPAs and Pale Ales! You can also filter this output by country of origin. This filter can be changed to numerous other countries. Origin gives you insight into which country's beers homebrewers like to make. The United States ranks first, Belgium ranks second, England ranks third, Germany ranks fourth, and Ireland ranks fifth. The Brew Method is a nice addition to the visual as well. With this feature, we can see which method of brewing is most popular for homebrewers! All Grain is by far the most popular, it's the cooler way to brew (see Appendix G).

The second visualization on this dashboard is called "Top 3 Styles per Style Family" and uses our recipeData.csv. This one shows users the three most popular styles of an individual Style Family (remember, Style Familys are made up of dozens of other beer styles). So for the style family of Amber Ales, we can see that the American Amber Ale is the most popular. Irish Red Ale and Belgian Dubbel come in second and third (see Appendix H). The third visualization is called "US vs Non US Origins by Style Family and Style." This visual separates each Style Family by Non US Origin and US Origin. After that, it tells you what beer styles are in each of those two categories. It's a great way to see how many beer styles originated in the US or abroad.

The visual also tells users the total count of beer recipes in the dataset for all beer styles (see Appendix I).

The fourth visualization is called "Beer Detail." This visual helps users understand how the search parameters work with our application by showing them the recipes in our dataset. It provides a detailed view of the metrics we are using in our classification: Color, Original Gravity, Final Gravity, ABV, and IBU. The visual can also be filtered by Style Family. The fun part about this visual is that it tells you names of other homebrewer recipes that are similar in regards to Style Family and even Styles. If users want to, they could look the names of these recipes up on Brewer's Friend and find them (see Appendix J)!

The second dashboard contains a total of two visualization on it. The first visual is called "Top Family Style by Origin." This one is really cool! It's a map that shows users where the beer styles in the recipeData.csv originated. So if we only selected IPA, we can see that only the United States and the United Kingdom had an influence on the styles of IPAs. It also gives users a total count of the beer recipes that are a beer style from that country of origin (see Appendix K). The second visualization is called "Beer Origins." This visual helps users understand which countries have the most popular beer styles and which countries are known for their beer. Homebrewers are most known for brewing beers of the United States origin and also of the UK, Belgium, and Germany. The map is a great way to show what we mean when we say country of origin for the style families (see Appendix L).

Web App Section:

In order for our group to make our Craft Beer Style Classifier easily available to users, we needed to host it on a website builder. We chose to use Flask! After doing some html work and creating an app.py file, we started designing the layout of our website. We wanted our webpage to be fun and colorful, so we went with a design concept called Neubrutalism. This style has big colorful buttons with nice rounded edges on everything. It has a candy-like style which almost appears to give it a bit of a glow. It's colorful shadows and subtle, pretty gradients really makes everything pop. It can best be described as having an arcade game feel!

The first thing users see when they open our website is an input box (see Appendix M). There're five possible sections where users can type in parameters on a beer that they've brewed. The first parameter is **Original Gravity (OG)**. An example of how this numeric value should be formatted is: 1052. The second parameter is **Final Gravity (FG)**. An example of this would be: 1007. Final gravity is always lower than original gravity. The third parameter is **Alcohol Content (ABV)**. An example of this would be 5.3. Don't add a percentage sign to this input, it won't match the dataset. The fourth parameter is **Standard Reference Method (SRM)**. Generally, this value scales from 1-40. Some recipes might have this number listed higher than 40. The fifth parameter is **International Bitterness Units (IBU)**. Most recipes are listed somewhere between 1-100 for this value. Once a user gives us all of these parameters, we can use our machine learning model on our datasets. The website then opens another window and tells users the output of what Style Family the app would classify the beer as. Below this output, there is a lot of descriptive data about the style. Each paragraph is related to a certain aspect about the style in the following order: *Overall Impression, Aroma, Flavor,* and then *Appearance*.

Conclusion/Call to Action:

Our project was able to successfully classify beer using our five parameters! This was a really cool way to analyze beer contrary to the traditional way. Most of the time, brewers use taste and subjective methods to dissect beer. In addition, we gained some insight on beer categorization through the use of Style Family. In beer school, brewers are taught to fixate on the historical origin of beer styles and categorize beer by countries. The Style Family is different than this because it does categorization by flavors and brewing methods. For example, the traditional categorization method wouldn't put an American Brown Ale and a British Brown Ale in the same category because they come from other countries. In our method, these two styles are in the same category because we're more interested in the fact that they're brown ales. Each Style Family is made up of numerous beer styles and it's interesting that we're able to use machine learning on so many different styles and get a decent prediction on what style a beer is based on five parameters.

Limitations and Future Work:

Our project could be greatly improved if we had more funding and time to work on it. The first improvement would be to have an interactive color chart on our website that displayed SRM colors of beer. A user would simply select a color on the chart that looks closest to the color of their beer and then it would return a numeric value for the SRM parameter we use for our machine learning. Currently our website just has users type in what they believe the color of their beer is. Sometimes people don't know what the SRM is and this would be a fun way to have users interact with our website.

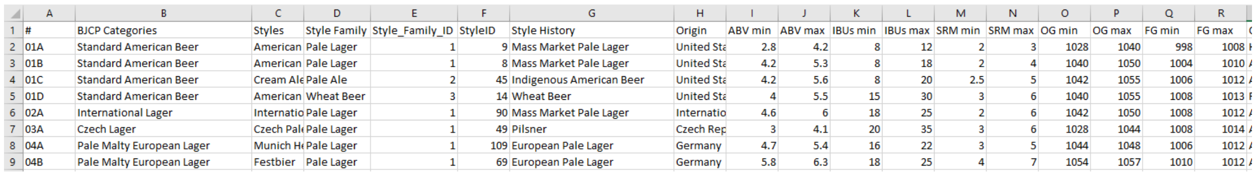
The main limitation of our project is that the accuracy could be better for our prediction. If we had more time to get more familiar with our Light Gradient Boosting Machine model, we could fine tune it. Working with that machine learning model, we were able to improve the accuracy by a few percentage points, but we were unsuccessful in getting the accuracy over 80%.

Another factor to this might be our data cleaning. There are still beer styles in certain style families that are probably outliers for our dataset. Narrowing down our search even more would probably help the machine learning model utilize the five parameters.

In the future, it would be really cool to have the model go beyond just predicting Style Family. Once the model knows what style family a beer potentially is, it would be nice to have the model search the five parameters within the style family and then output what beer style it most likely is.

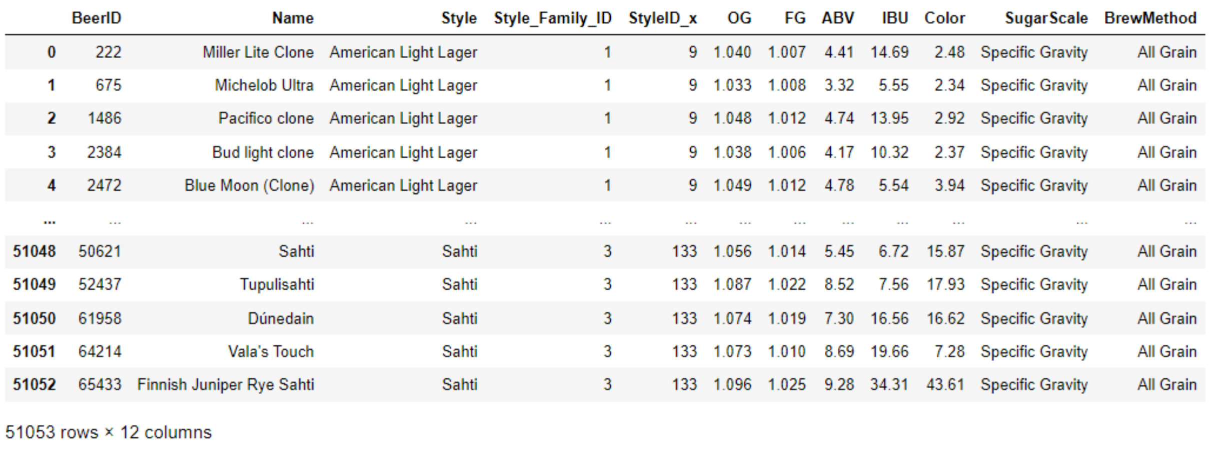
<u>Appendix:</u>

Appendix A (guidelines.csv)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	#	BJCP Categories	Styles	Style Family	Style_Family_ID	StyleID	Style History	Origin	ABV min	ABV max	IBUs min	IBUs max	SRM min	SRM max	OG min	OG max	FG min	FG max
2	01A	Standard American Beer	American	Pale Lager	1	9	Mass Market Pale Lager	United Sta	2.8	4.2	8	12	2	3	1028	1040	998	1008
3	01B	Standard American Beer	American	Pale Lager	1	8	Mass Market Pale Lager	United Sta	4.2	5.3	8	18	2	4	1040	1050	1004	1010
4	01C	Standard American Beer	Cream Ale	Pale Ale	2	45	Indigenous American Beer	United Sta	4.2	5.6	8	20	2.5	5	1042	1055	1006	1012
5	01D	Standard American Beer	American	Wheat Beer	3	14	Wheat Beer	United Sta	4	5.5	15	30	3	6	1040	1055	1008	1013
6	02A	International Lager	Internatio	Pale Lager	1	90	Mass Market Pale Lager	Internatio	4.6	6	18	25	2	6	1042	1050	1008	1012
7	03A	Czech Lager	Czech Pale	Pale Lager	1	49	Pilsner	Czech Rep	3	4.1	20	35	3	6	1028	1044	1008	1014
8	04A	Pale Malty European Lager	Munich He	Pale Lager	1	109	European Pale Lager	Germany	4.7	5.4	16	22	3	5	1044	1048	1006	1012
9	04B	Pale Malty European Lager	Festbier	Pale Lager	1	69	European Pale Lager	Germany	5.8	6.3	18	25	4	7	1054	1057	1010	1012

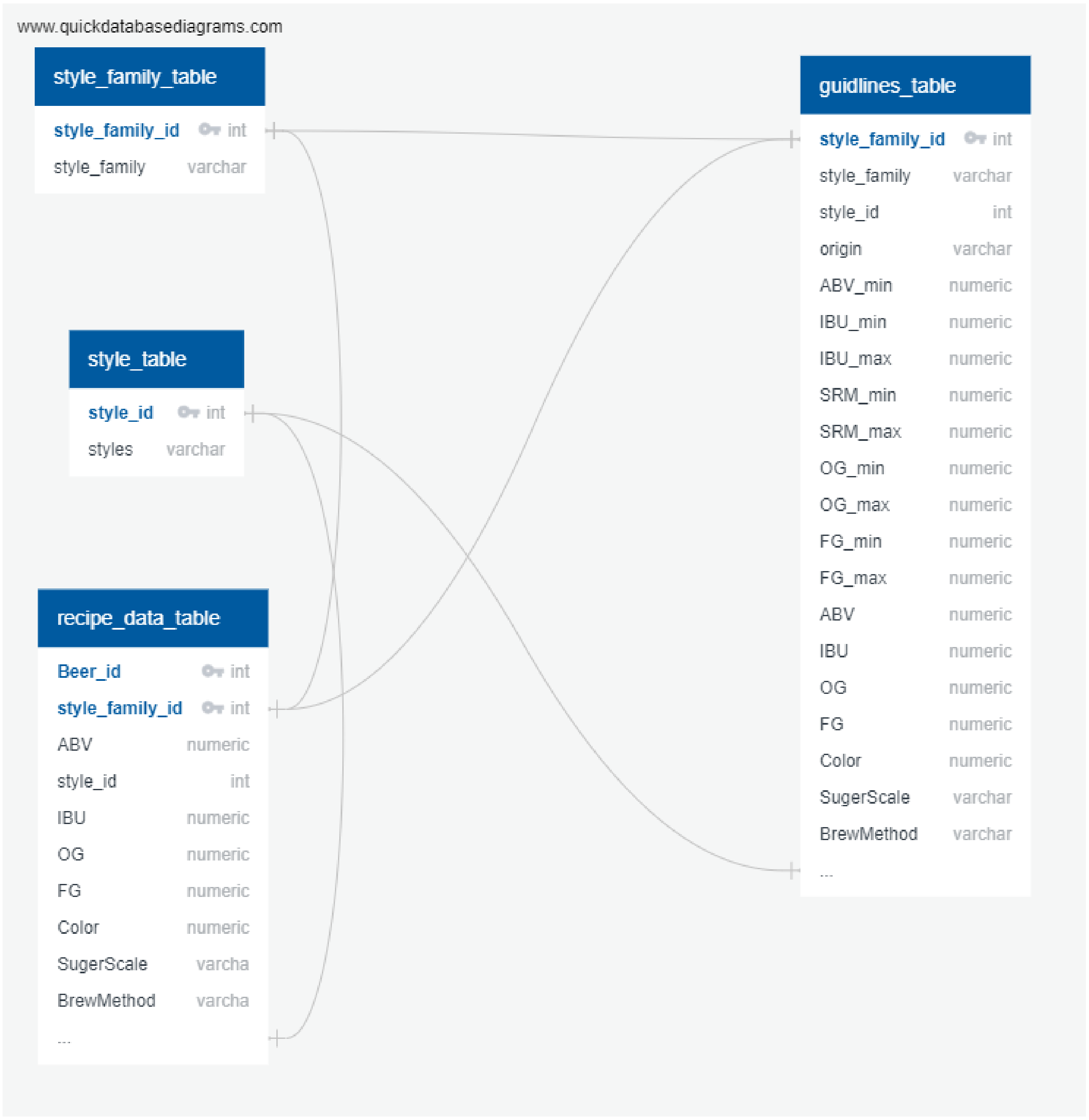
Appendix B (recipeData.csv)



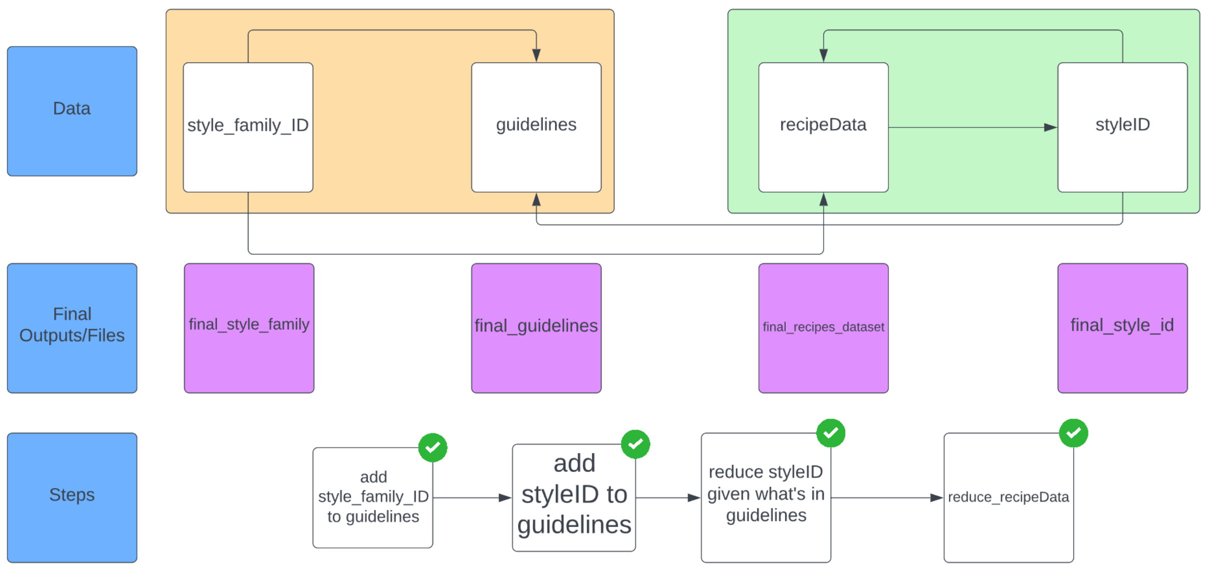
	BeerID	Name	Style	Style_Family_ID	StyleID_x	OG	FG	ABV	IBU	Color	SugarScale	BrewMethod
0	222	Miller Lite Clone	American Light Lager	1	9	1.040	1.007	4.41	14.69	2.48	Specific Gravity	All Grain
1	675	Michelob Ultra	American Light Lager	1	9	1.033	1.008	3.32	5.55	2.34	Specific Gravity	All Grain
2	1486	Pacifico clone	American Light Lager	1	9	1.048	1.012	4.74	13.95	2.92	Specific Gravity	All Grain
3	2384	Bud light clone	American Light Lager	1	9	1.038	1.006	4.17	10.32	2.37	Specific Gravity	All Grain
4	2472	Blue Moon (Clone)	American Light Lager	1	9	1.049	1.012	4.78	5.54	3.94	Specific Gravity	All Grain
...	...	...	...	...	...	...	...	...	...	...	...	...
51048	50621	Sahti	Sahti	3	133	1.056	1.014	5.45	6.72	15.87	Specific Gravity	All Grain
51049	52437	Tupulisahti	Sahti	3	133	1.087	1.022	8.52	7.56	17.93	Specific Gravity	All Grain
51050	61958	Dúnedain	Sahti	3	133	1.074	1.019	7.30	16.56	16.62	Specific Gravity	All Grain
51051	64214	Vala's Touch	Sahti	3	133	1.073	1.010	8.69	19.66	7.28	Specific Gravity	All Grain
51052	65433	Finnish Juniper Rye Sahti	Sahti	3	133	1.096	1.025	9.28	34.31	43.61	Specific Gravity	All Grain

51053 rows × 12 columns

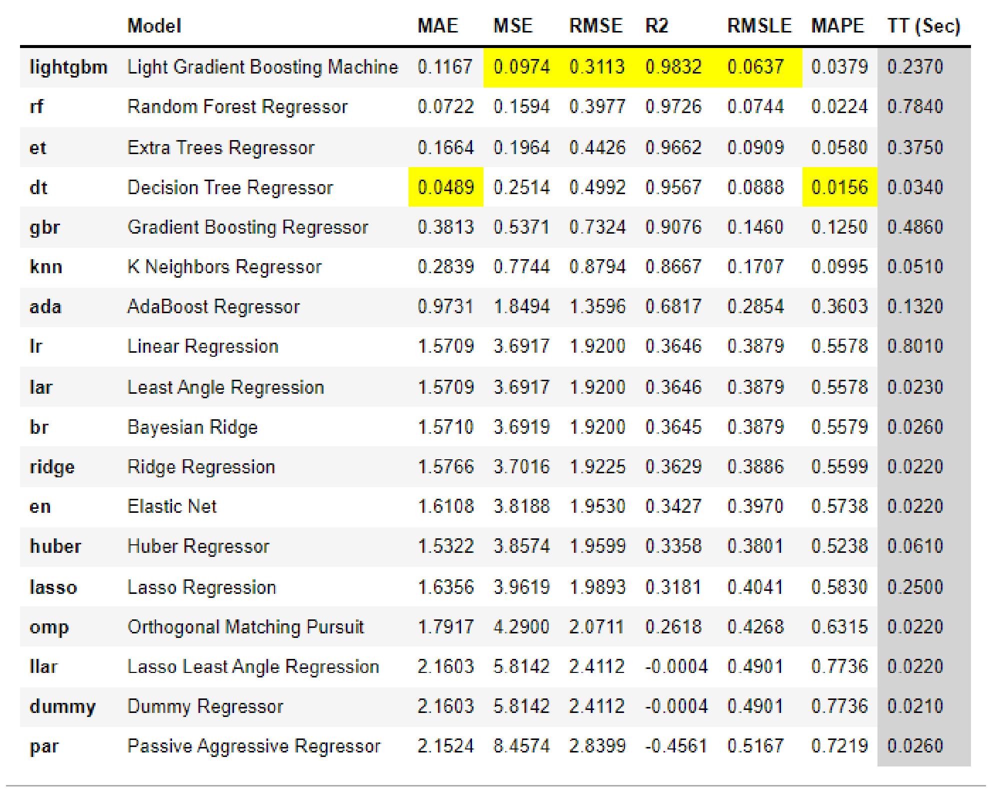
Appendix C


www.quickdatabasediagrams.com
style_family_table
style_family_id int
style_family varchar
style_table
style_id int
styles varchar
recipe_data_table
Beer_id int
style_family_id int
ABV numeric
style_id int
IBU numeric
OG numeric
FG numeric
Color numeric
SugerScale varcha
BrewMethod varcha
...
guidlines_table
style_family_id int
style_family varchar
style_id int
origin varchar
ABV_min numeric
IBU_min numeric
IBU_max numeric
SRM_min numeric
SRM_max numeric
OG_min numeric
OG_max numeric
FG_min numeric
FG_max numeric
ABV numeric
IBU numeric
OG numeric
FG numeric
Color numeric
SugerScale varchar
BrewMethod varchar
...

Appendix D

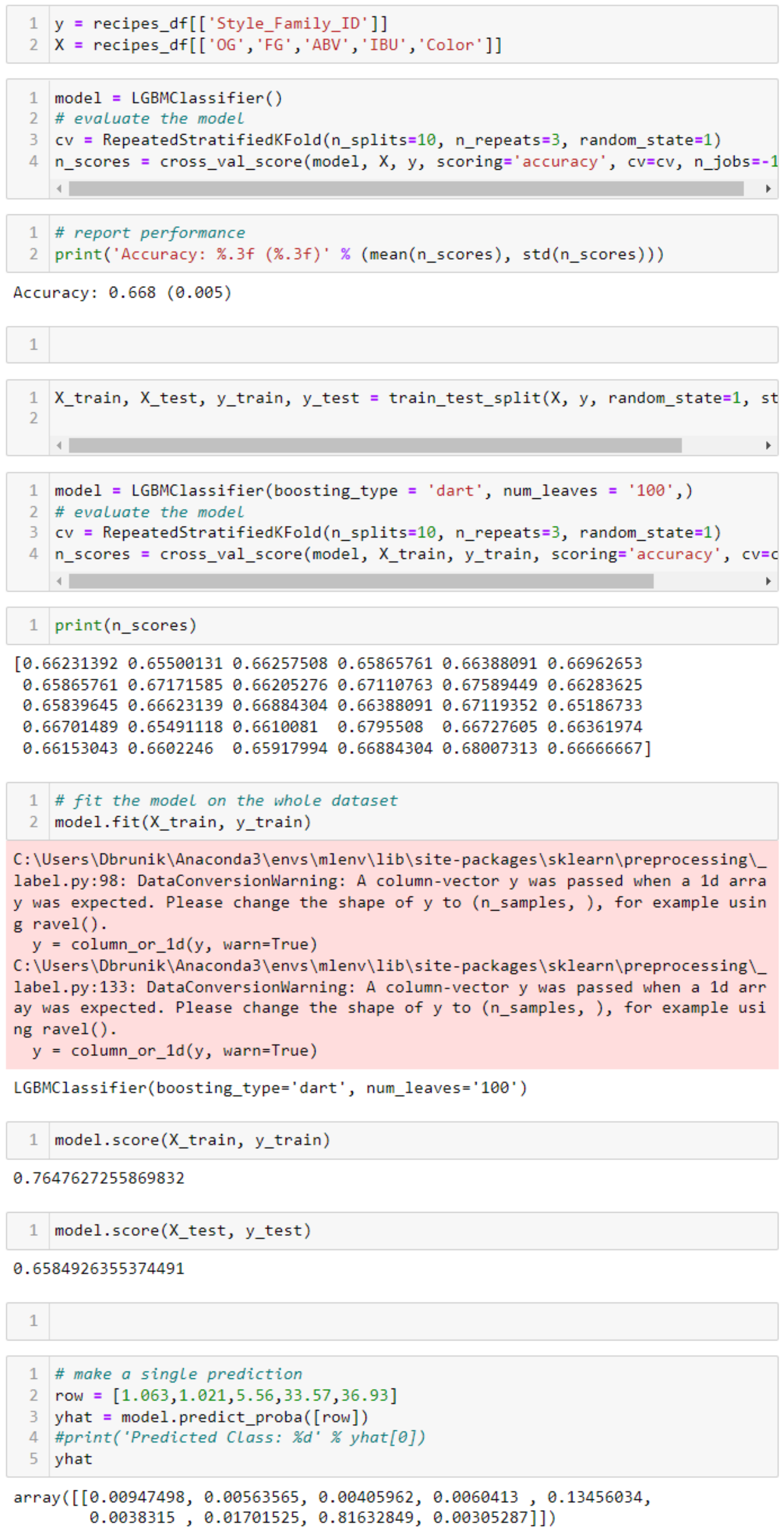


Appendix E



	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.1167	0.0974	0.3113	0.9832	0.0637	0.0379	0.2370
rf	Random Forest Regressor	0.0722	0.1594	0.3977	0.9726	0.0744	0.0224	0.7840
et	Extra Trees Regressor	0.1664	0.1964	0.4426	0.9662	0.0909	0.0580	0.3750
dt	Decision Tree Regressor	0.0489	0.2514	0.4992	0.9567	0.0888	0.0156	0.0340
gbr	Gradient Boosting Regressor	0.3813	0.5371	0.7324	0.9076	0.1460	0.1250	0.4860
knn	K Neighbors Regressor	0.2839	0.7744	0.8794	0.8667	0.1707	0.0995	0.0510
ada	AdaBoost Regressor	0.9731	1.8494	1.3596	0.6817	0.2854	0.3603	0.1320
lr	Linear Regression	1.5709	3.6917	1.9200	0.3646	0.3879	0.5578	0.8010
lar	Least Angle Regression	1.5709	3.6917	1.9200	0.3646	0.3879	0.5578	0.0230
br	Bayesian Ridge	1.5710	3.6919	1.9200	0.3645	0.3879	0.5579	0.0260
ridge	Ridge Regression	1.5766	3.7016	1.9225	0.3629	0.3886	0.5599	0.0220
en	Elastic Net	1.6108	3.8188	1.9530	0.3427	0.3970	0.5738	0.0220
huber	Huber Regressor	1.5322	3.8574	1.9599	0.3358	0.3801	0.5238	0.0610
lasso	Lasso Regression	1.6356	3.9619	1.9893	0.3181	0.4041	0.5830	0.2500
omp	Orthogonal Matching Pursuit	1.7917	4.2900	2.0711	0.2618	0.4268	0.6315	0.0220
llar	Lasso Least Angle Regression	2.1603	5.8142	2.4112	-0.0004	0.4901	0.7736	0.0220
dummy	Dummy Regressor	2.1603	5.8142	2.4112	-0.0004	0.4901	0.7736	0.0210
par	Passive Aggressive Regressor	2.1524	8.4574	2.8399	-0.4561	0.5167	0.7219	0.0260

Appendix F



```
y = recipes_df[['Style_Family_ID']]
X = recipes_df[['OG','FG','ABV','IBU','Color']]
```

```
model = LGBMClassifier()
# evaluate the model
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
n_scores = cross_val_score(model, X, y, scoring='accuracy', cv=cv, n_jobs=-1
```

```
# report performance
print('Accuracy: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
```

```
Accuracy: 0.668 (0.005)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1, st
```

```
model = LGBMClassifier(boosting_type = 'dart', num_leaves = '100',)
# evaluate the model
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
n_scores = cross_val_score(model, X_train, y_train, scoring='accuracy', cv=c
```

```
print(n_scores)
```

```
[0.66231392 0.65500131 0.66257508 0.65865761 0.66388091 0.66962653
 0.65865761 0.67171585 0.66205276 0.67110763 0.67589449 0.66283625
 0.65839645 0.66623139 0.66884304 0.66388091 0.67119352 0.65186733
 0.66701489 0.65491118 0.6610081  0.6795508  0.66727605 0.66361974
 0.66153043 0.6602246  0.65917994 0.66884304 0.68007313 0.66666667]
```

```
# fit the model on the whole dataset
model.fit(X_train, y_train)
```

```
C:\Users\Dbrunik\Anaconda3\envs\mlenv\lib\site-packages\sklearn\preprocessing\_
label.py:98: DataConversionWarning: A column-vector y was passed when a 1d arra
y was expected. Please change the shape of y to (n_samples, ), for example usin
g ravel().
  y = column_or_1d(y, warn=True)
C:\Users\Dbrunik\Anaconda3\envs\mlenv\lib\site-packages\sklearn\preprocessing\_
label.py:133: DataConversionWarning: A column-vector y was passed when a 1d arr
ay was expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
```

```
LGBMClassifier(boosting_type='dart', num_leaves='100')
```

```
model.score(X_train, y_train)
```

```
0.7647627255869832
```

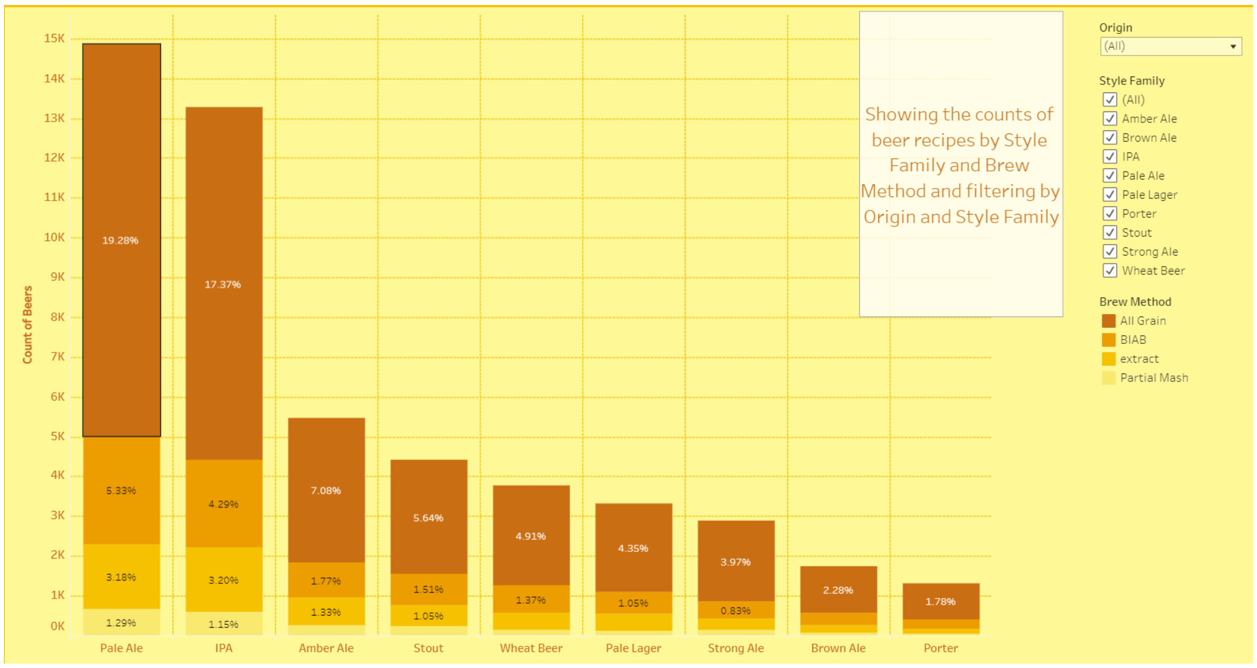
```
model.score(X_test, y_test)
```

```
0.6584926355374491
```

```
# make a single prediction
row = [1.063,1.021,5.56,33.57,36.93]
yhat = model.predict_proba([row])
#print('Predicted Class: %d' % yhat[0])
yhat
```

```
array([[0.00947498, 0.00563565, 0.00405962, 0.0060413 , 0.13456034,
        0.0038315 , 0.01701525, 0.81632849, 0.00305287]])
```

Appendix G

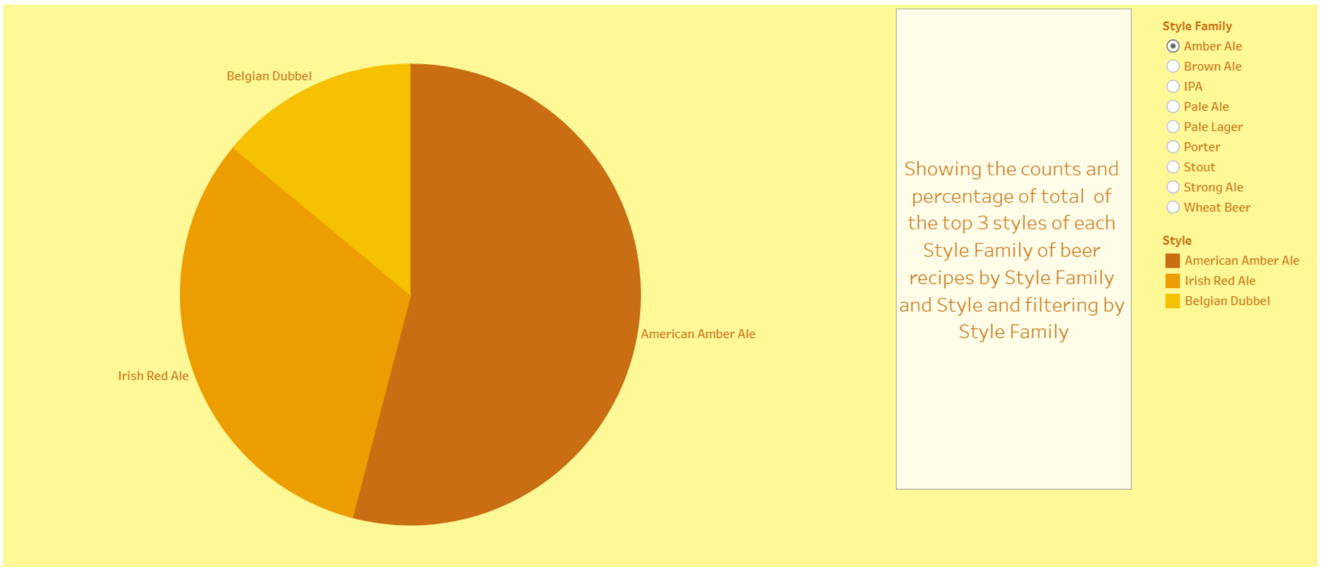


Showing the counts of beer recipes by Style Family and Brew Method and filtering by Origin and Style Family

Origin
(All)

Style Family
✓ (All)
✓ Amber Ale
✓ Brown Ale
✓ IPA
✓ Pale Ale
✓ Pale Lager
✓ Porter
✓ Stout
✓ Strong Ale
✓ Wheat Beer

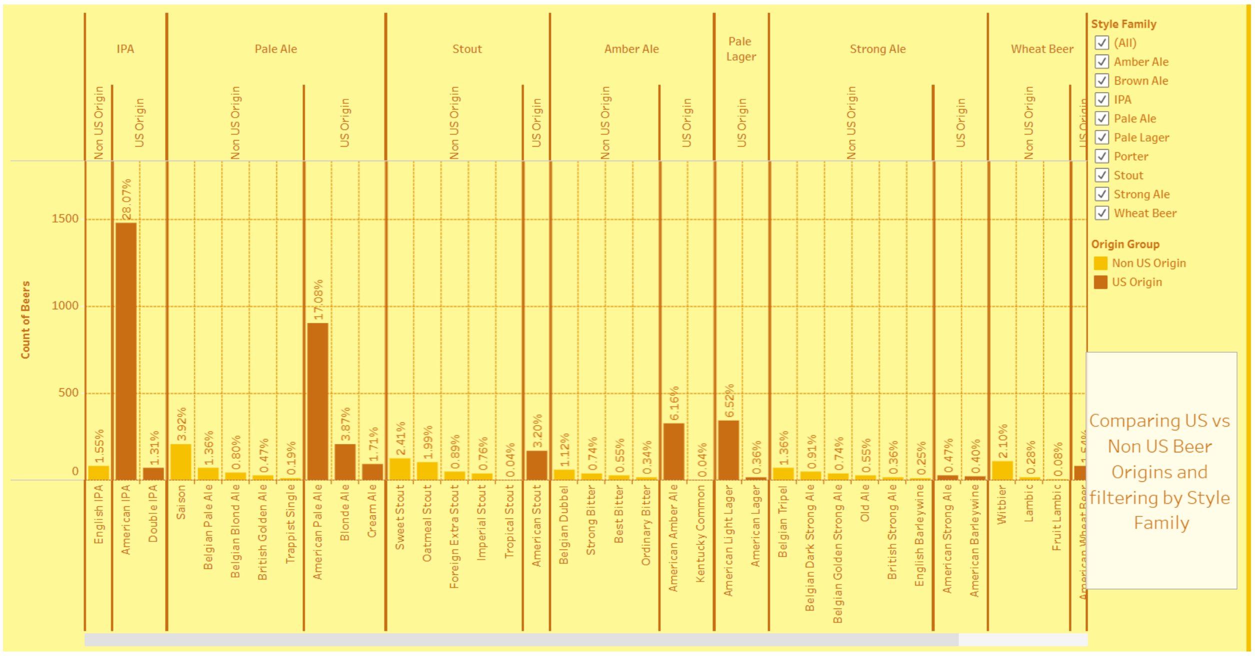
Appendix H



Showing the counts and percentage of total of the top 3 styles of each Style Family of beer recipes by Style Family and Style and filtering by Style Family

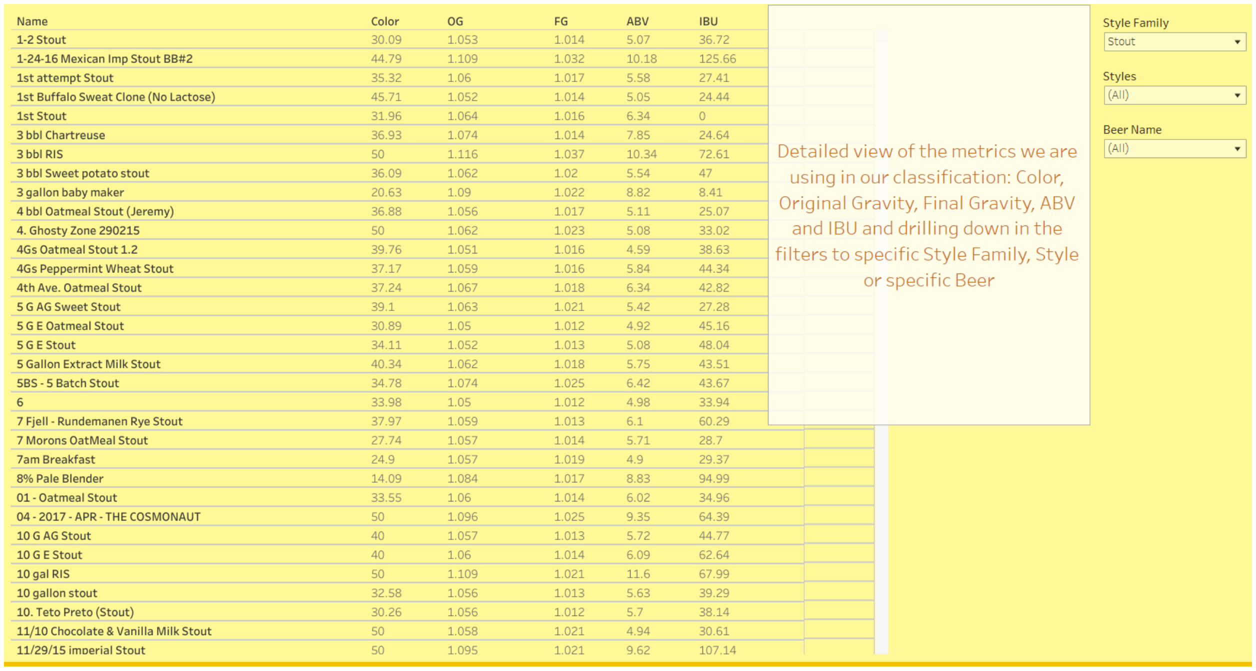
Style Family
● Amber Ale
○ Brown Ale
○ IPA
○ Pale Ale
○ Pale Lager
○ Porter
○ Stout
○ Strong Ale
○ Wheat Beer

Appendix I



Comparing US vs Non US Beer Origins and filtering by Style Family

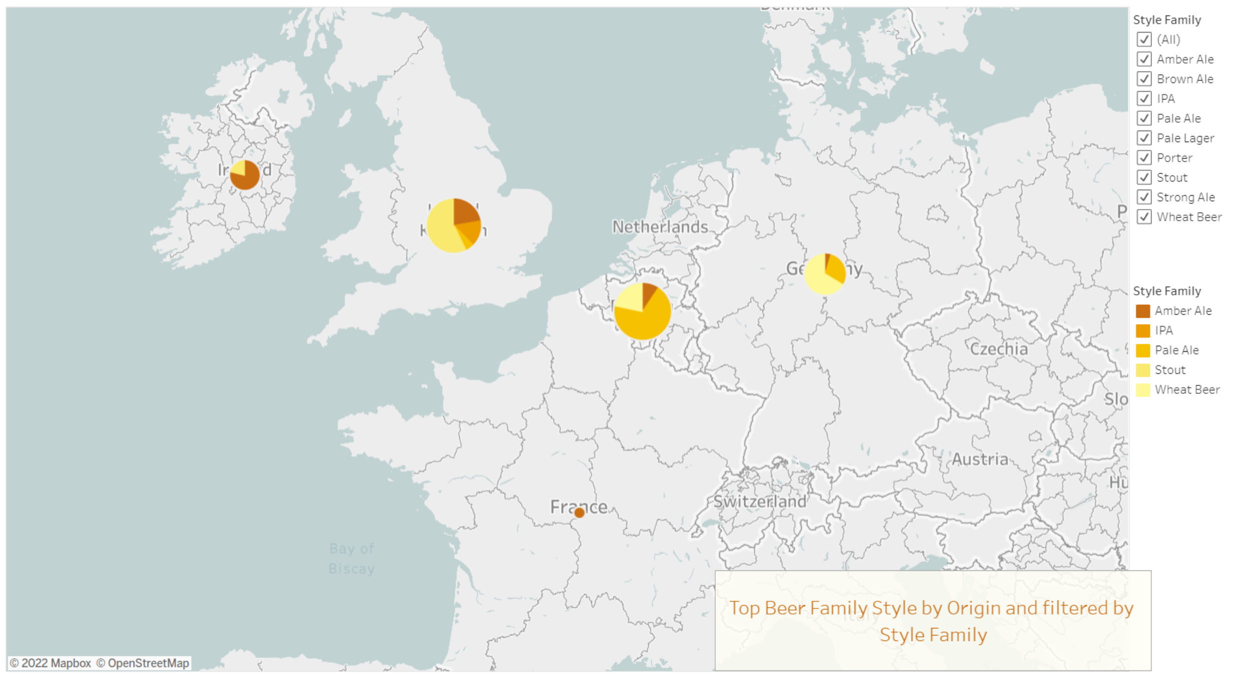
Appendix J



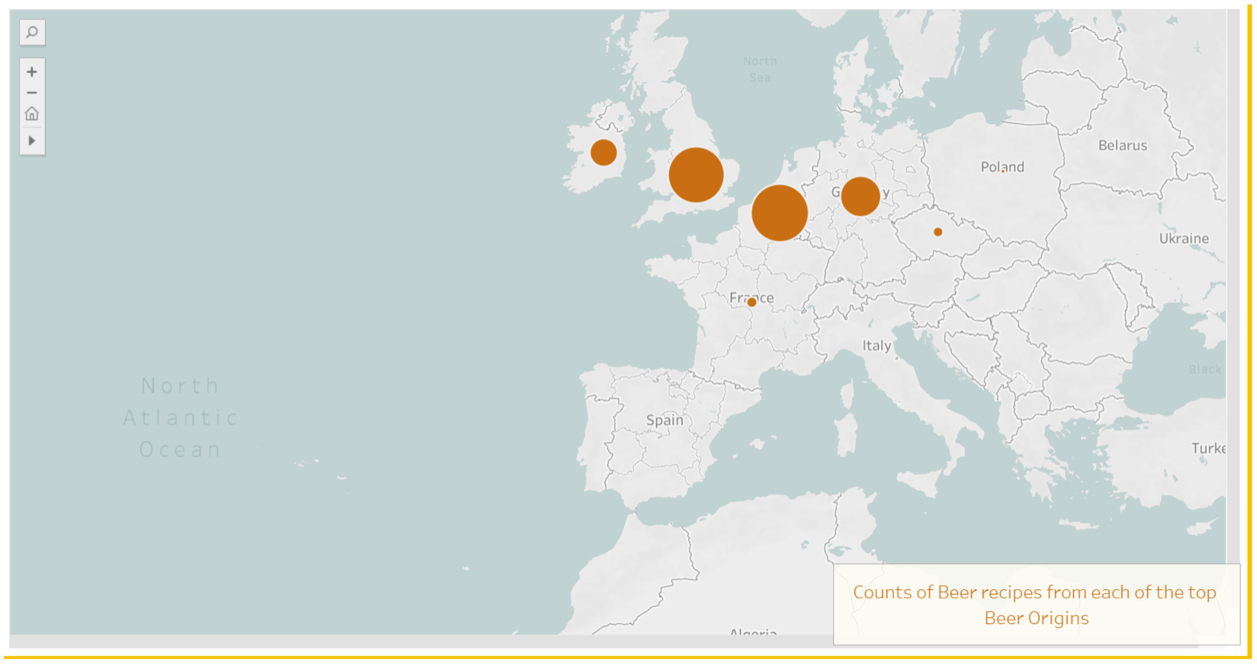
Name	Color	OG	FG	ABV	IBU
1-2 Stout	30.09	1.053	1.014	5.07	36.72
1-24-16 Mexican Imp Stout BB#2	44.79	1.109	1.032	10.18	125.66
1st attempt Stout	35.32	1.06	1.017	5.58	27.41
1st Buffalo Sweat Clone (No Lactose)	45.71	1.052	1.014	5.05	24.44
1st Stout	31.96	1.064	1.016	6.34	0
3 bbl Chartreuse	36.93	1.074	1.014	7.85	24.64
3 bbl RIS	50	1.116	1.037	10.34	72.61
3 bbl Sweet potato stout	36.09	1.062	1.02	5.54	47
3 gallon baby maker	20.63	1.09	1.022	8.82	8.41
4 bbl Oatmeal Stout (Jeremy)	36.88	1.056	1.017	5.11	25.07
4. Ghosty Zone 290215	50	1.062	1.023	5.08	33.02
4Gs Oatmeal Stout 1.2	39.76	1.051	1.016	4.59	38.63
4Gs Peppermint Wheat Stout	37.17	1.059	1.016	5.84	44.34
4th Ave. Oatmeal Stout	37.24	1.067	1.018	6.34	42.82
5 G AG Sweet Stout	39.1	1.063	1.021	5.42	27.28
5 G E Oatmeal Stout	30.89	1.05	1.012	4.92	45.16
5 G E Stout	34.11	1.052	1.013	5.08	48.04
5 Gallon Extract Milk Stout	40.34	1.062	1.018	5.75	43.51
5BS - 5 Batch Stout	34.78	1.074	1.025	6.42	43.67
6	33.98	1.05	1.012	4.98	33.94
7 Fjell - Rundemanen Rye Stout	37.97	1.059	1.013	6.1	60.29
7 Morons OatMeal Stout	27.74	1.057	1.014	5.71	28.7
7am Breakfast	24.9	1.057	1.019	4.9	29.37
8% Pale Blender	14.09	1.084	1.017	8.83	94.99
01 - Oatmeal Stout	33.55	1.06	1.014	6.02	34.96
04 - 2017 - APR - THE COSMONAUT	50	1.096	1.025	9.35	64.39
10 G AG Stout	40	1.057	1.013	5.72	44.77
10 G E Stout	40	1.06	1.014	6.09	62.64
10 gal RIS	50	1.109	1.021	11.6	67.99
10 gallon stout	32.58	1.056	1.013	5.63	39.29
10. Teto Preto (Stout)	30.26	1.056	1.012	5.7	38.14
11/10 Chocolate & Vanilla Milk Stout	50	1.058	1.021	4.94	30.61
11/29/15 imperial Stout	50	1.095	1.021	9.62	107.14

Detailed view of the metrics we are using in our classification: Color, Original Gravity, Final Gravity, ABV and IBU and drilling down in the filters to specific Style Family, Style or specific Beer

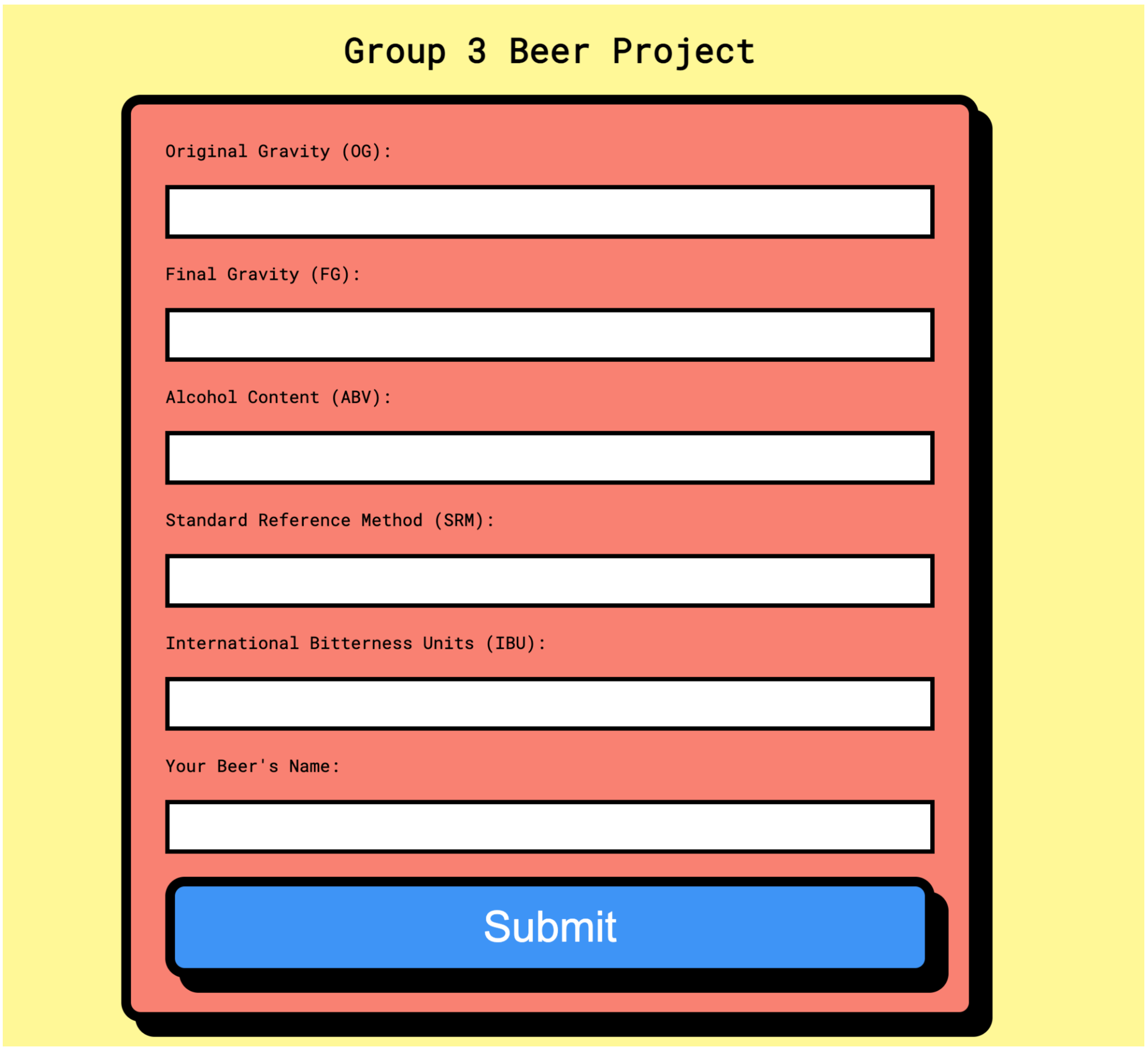
Appendix K



Appendix L



Appendix M



Group 3 Beer Project

Original Gravity (OG):

Final Gravity (FG):

Alcohol Content (ABV):

Standard Reference Method (SRM):

International Bitterness Units (IBU):

Your Beer's Name: