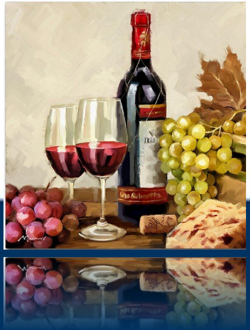
Wine Reviews & Rating Analysis

By: Ben Lew, Flora Choi, Janine Yoro, Patrick Po, Ahsan Tebha



CODE AVENGERS

UC Berkeley Extension

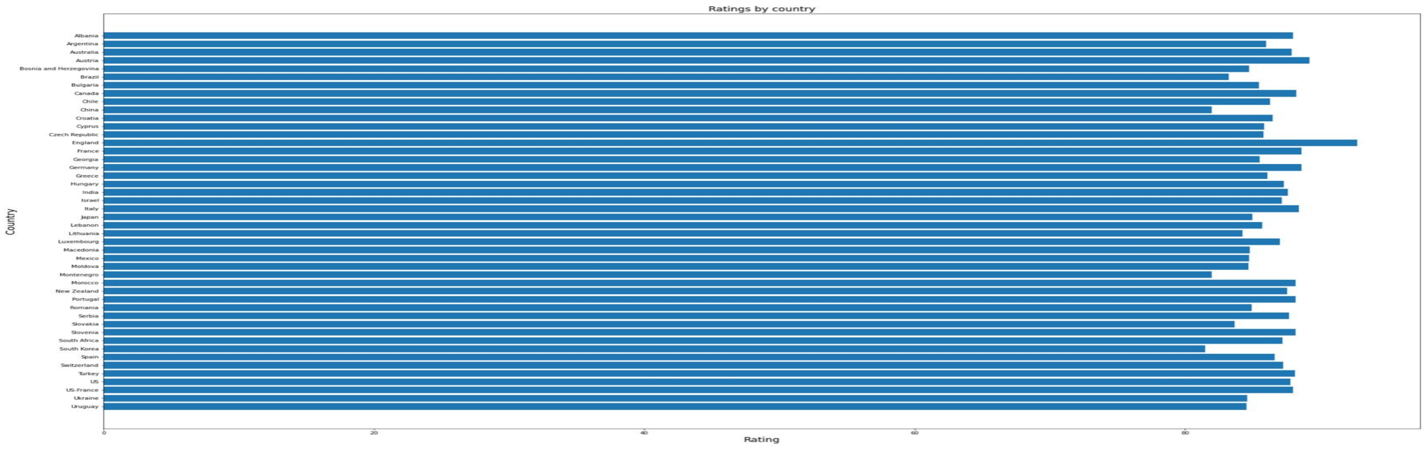
AGENDA

- PURPOSE
- PREPROCESSING
- ANALYSIS/MACHINE LEARNING
- TECHNOLOGY OVERVIEW
- RESULTS/DASHBOARD
- CONCLUSION
- Q&A

CODE AVENGERS

After a long day, we all unwind in some way….

- We choose the topic of wine
- Reason being many people like drinking, especially during a global pandemic



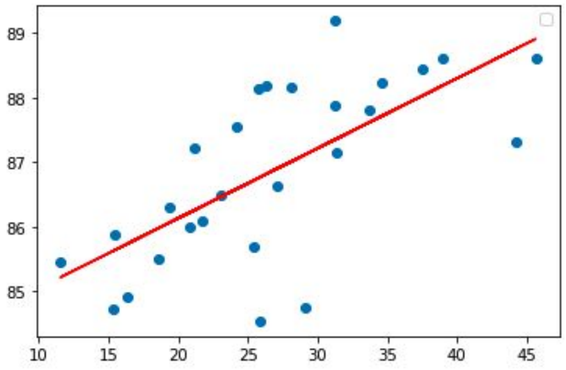
Drunk and Giving Directions

- Our source data originated from winemag.com through a web scrape that an avid wine enthusiast scraped up.
- The question we hoped to answer was *“Is rating predictive of future pricing?”*
- Challenges experienced with this question were the following:
  - We couldn’t make a unique identifier.
  - We couldn’t map the wines year to year.
  - Too much drift in the 2018 data set.
- Ultimately 2017 had enough data points to run through the model.
- This made the ASK evolve to:
  - **“Is rating predictive of price for the year?”**

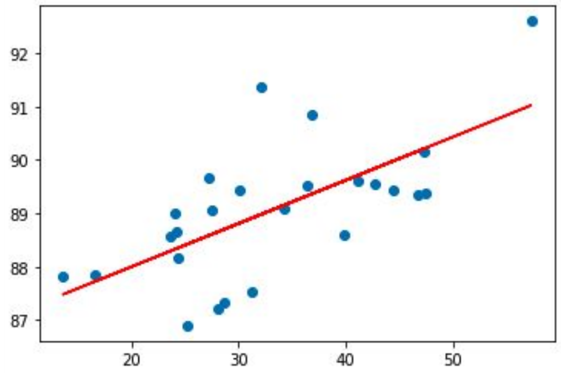
Better Preparation Leads to Better Execution

- What we did:
  - Downloaded 2017 Dataset (150k Rows)
  - Scraped WineMag 2018 Dataset (22k Rows)
  - Dropped columns not relevant to analysis
  - Dropped NaN values
  - Created Target Columns

2017



2018



Data Exploration & Analysis

- Our methodology for feature engineering was to first get a high level view, address the nulls, outliers and then organize/prepare the data.

- Analysis Overview
    - 59,790 rows
    - Few modifications applied to dataframe
        - Change columns names for readability
        - Change numerical data types to integers
        - Removed all the commas in the wine_type column to avoid creating additional columns in the csv

Preprocessing: Dataframes/Tables

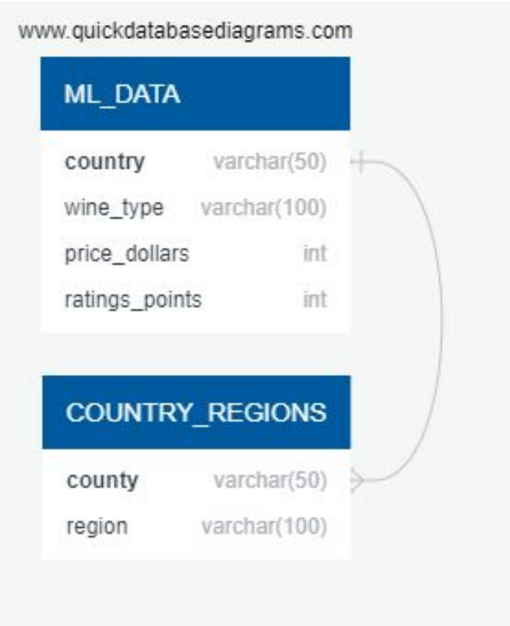
```
# This is for 2017 DF
orig_url = "https://drive.google.com/file/d/1zFHNHw6mTh4kyx8pVd27rh2ttV8b7zfU/view?usp=sharing"
file_id = orig_url.split('/')[-2]
dwn_url='https://drive.google.com/uc?export=download&id=' + file_id
url = requests.get(dwn_url).text
csv_raw = StringIO(url)
csv2017_df = pd.read_csv(csv_raw)
csv2017_df.head(5)
```

	Unnamed: 0	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum SelecciÃ³n Especial Reserva	96	110.0	Northern Spain	Toro	NaN	Tinta de Toro	Bodega Carmen RodrÃguez
2	2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley
3	3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi
4	4	France	This is the top wine from La BÃ©gude, named af...	La BrÃ»lade	95	66.0	Provence	Bandol	NaN	Provence red blend	Domaine de la BÃ©gude

	country	wine_type	price_dollars	ratings_points
0	US	Cabernet Sauvignon	235.0	96
1	Spain	Tinta de Toro	110.0	96
2	US	Sauvignon Blanc	90.0	96
3	US	Pinot Noir	65.0	96
4	France	Provence red blend	66.0	95

CSV FILE

1 **DATABASE**



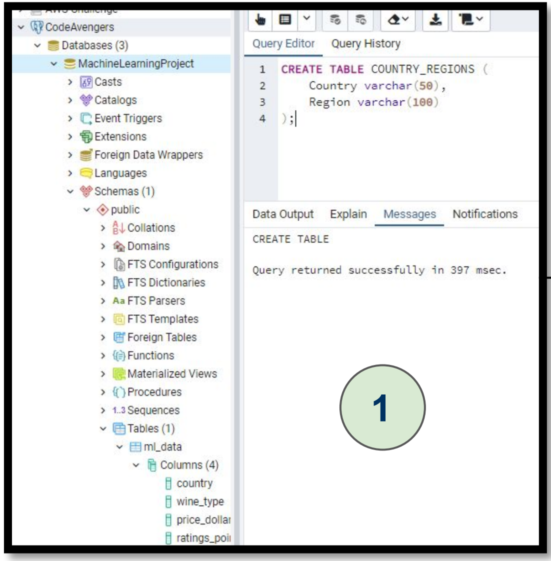
2 **CONTINUES FOR PROCESSING FOR *MACHINE LEARNING***

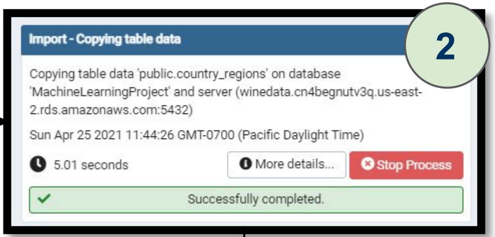
Database

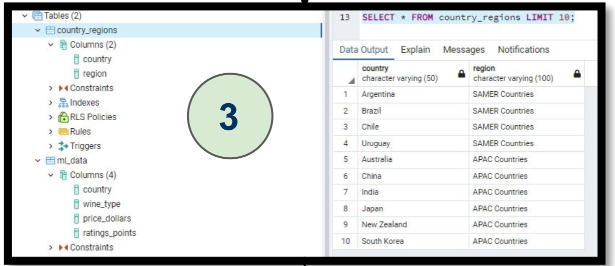
- Establish a connection between AWS RDS and PostgreSQL

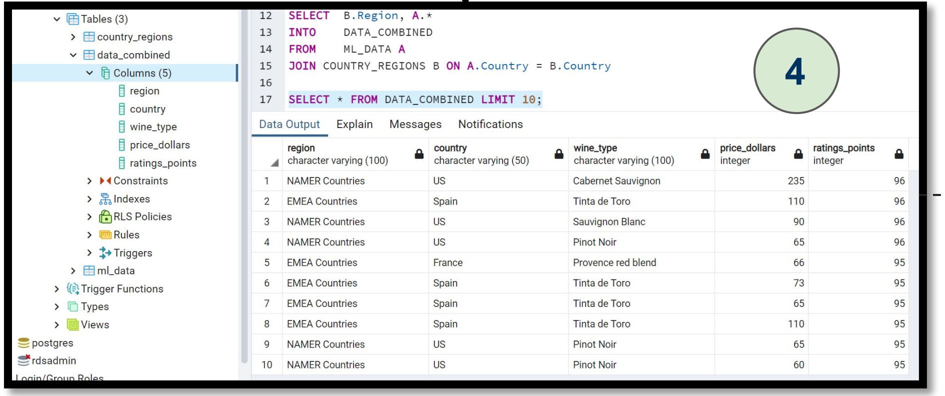
- Establish a connection between our notebook code to both AWS RDS & PostgreSQL

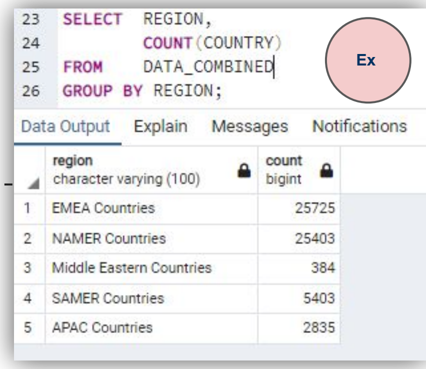
- Bring in the **COUNTRY_REGIONS** dataset for the purpose of joining and future reporting.











Machine Learning - Initial Model

MAIN OBJECTIVE: *GET A MODEL TO WORK*

Begin by building a working model at its **simplest form**.
Eliminate any noise related to descriptive columns & keep only the numerical values.

Create a new categorical value

GOOD WINE: Ratings 89 or Below
GREAT WINE: Ratings 90 or Above

	price_dollars	ratings_desc
1	110	1.0
2	90	1.0
3	65	1.0
4	66	1.0
5	73	1.0

Wine Spectator's Wine Ratings 100-Point Scale:

- **95-100** — Classic; a great wine
- **90-94** — Outstanding; superior character and style
- **80-89** — Good to very good; wine with special qualities
- **70-79** — Average; drinkable wine that may have minor flaws
- **60-69** — Below average; drinkable but not recommended
- **50-59** — Poor; undrinkable, not recommended

Source: http://www.winewins.com/wine-ratings/

FEATURES VS TARGET

Feature: PRICE_DOLLARS
Target: RATINGS_DESC
("0" For Good Wine, "1" For Great Wine")

DISTRIBUTION

★ Check the spread of price
★ Normal distribution?
★ Address Outliers (Remove top 1%)

```
#get stats of data -- notice the max
new2017_df.describe()
```

	price_dollars	ratings_desc
count	59750.000000	59750.000000
mean	36.216268	0.360854
std	36.442206	0.480252
min	4.000000	0.000000
25%	17.000000	0.000000
50%	27.000000	0.000000
75%	45.000000	1.000000
max	2013.000000	1.000000

74%

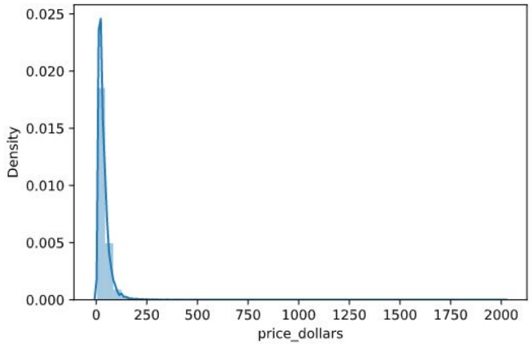
LOGISTIC REGRESSION MODEL

FINAL DATAFRAME

1. Split the data into Training and Testing
2. Created the Logistic Regression Model
3. Fit / Trained the model
4. Made Predictions
5. Validated model using confusion matrix
6. Generated an accuracy score

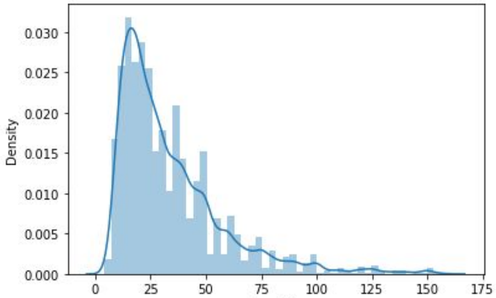
Machine Learning - Model Optimization

<AxesSubplot:xlabel='price_dollars', ylabel='Density'>



Full Dataset Distribution

<AxesSubplot:xlabel='price_dollars', ylabel='Density'>



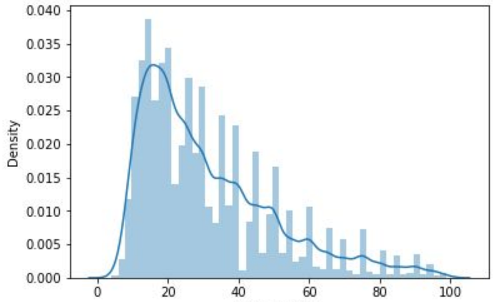

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, predictions)
```

0.7366107654855288

Outliers re-defined to >$100 price

Standard Scalar applied

<AxesSubplot:xlabel='price_dollars', ylabel='Density'>




```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, predictions)
```

0.7368128431440684

	Predicted good_wine	Predicted great_wine
Actual good_wine	8585	2943
Actual great_wine	952	2308

```
from imblearn.metrics import classification_report_imbalanced
print(classification_report_imbalanced(y_test,predictions))
```

	pre	rec	spe	f1	geo	iba	sup
0.0	0.74	0.90	0.44	0.82	0.63	0.41	9537
1.0	0.71	0.44	0.90	0.54	0.63	0.38	5251
avg / total	0.73	0.74	0.60	0.72	0.63	0.40	14788

	Predicted good_wine	Predicted great_wine
Actual good_wine	8360	2674
Actual great_wine	1113	2242

```
from imblearn.metrics import classification_report_imbalanced
print(classification_report_imbalanced(y_test,predictions))
```

	pre	rec	spe	f1	geo	iba	sup
0.0	0.76	0.88	0.46	0.82	0.63	0.42	9473
1.0	0.67	0.46	0.88	0.54	0.63	0.39	4916
avg / total	0.73	0.74	0.60	0.72	0.63	0.41	14389

Machine Learning - Validation Models

	US	Other Countries
Accuracy Score	73%	80%
Confusion Matrix		

US

	Predicted Good Wine	Predicted Great Wine
Actual Good Wine	9,433	3,193
Actual Great Wine	921	1,907

Other Countries

	Predicted Good Wine	Predicted Great Wine
Actual Good Wine	12,558	2,907
Actual Great Wine	751	2,240

Classification Reports

```
                pre       rec       spe        f1       geo       iba       sup

        0.0       0.81      0.94      0.44      0.87      0.64      0.43     13309
        1.0       0.75      0.44      0.94      0.55      0.64      0.39      5147

avg / total       0.79      0.80      0.58      0.78      0.64      0.42     18456
```

```
                pre       rec       spe        f1       geo       iba       sup

        0.0       0.81      0.94      0.44      0.87      0.64      0.43     13309
        1.0       0.75      0.44      0.94      0.55      0.64      0.39      5147

avg / total       0.79      0.80      0.58      0.78      0.64      0.42     18456
```

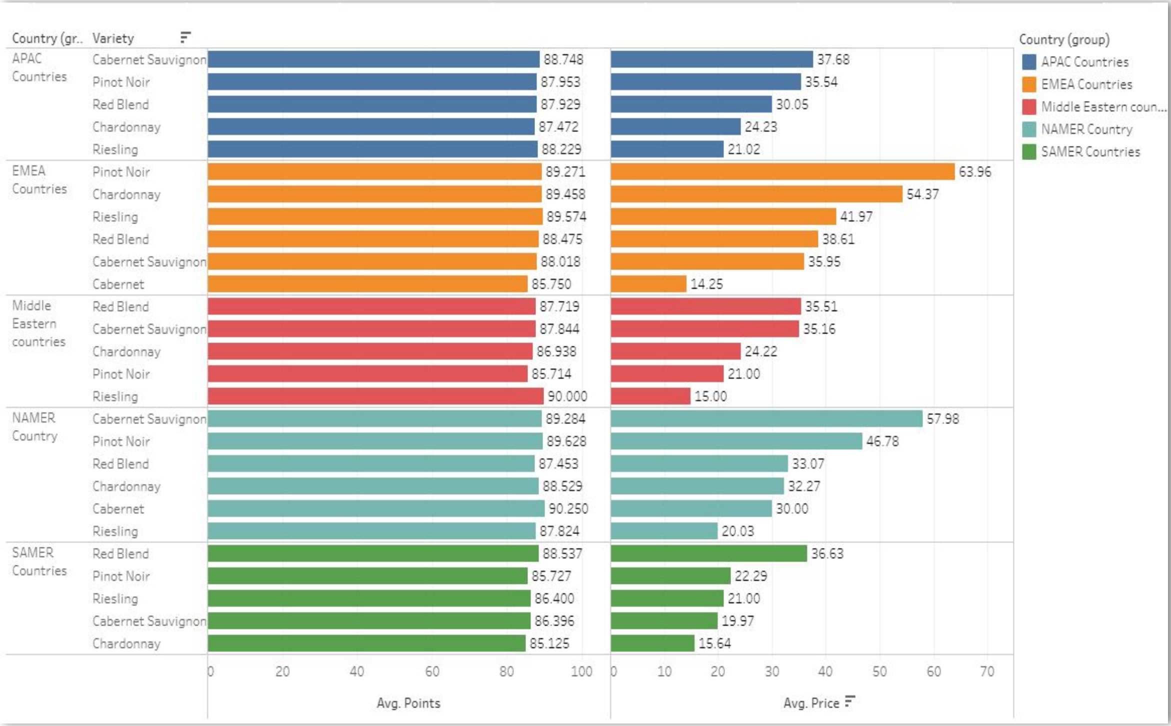
Tech Used & Results

Technology Overview:

- ➢ Python, Tableau, Postgres
- ➢ Google Colab, Github

Dashboard:

- ➢ Link to our Tableau project: [Link to Tableau]



Further Analysis & Alternative Actions

- The data set could be further drilled down…
- Run multiple regression models both linear and logistic to find best correlating features

Q&As

Thank You

Powered By:

CODE AVENGERS

B

A

F