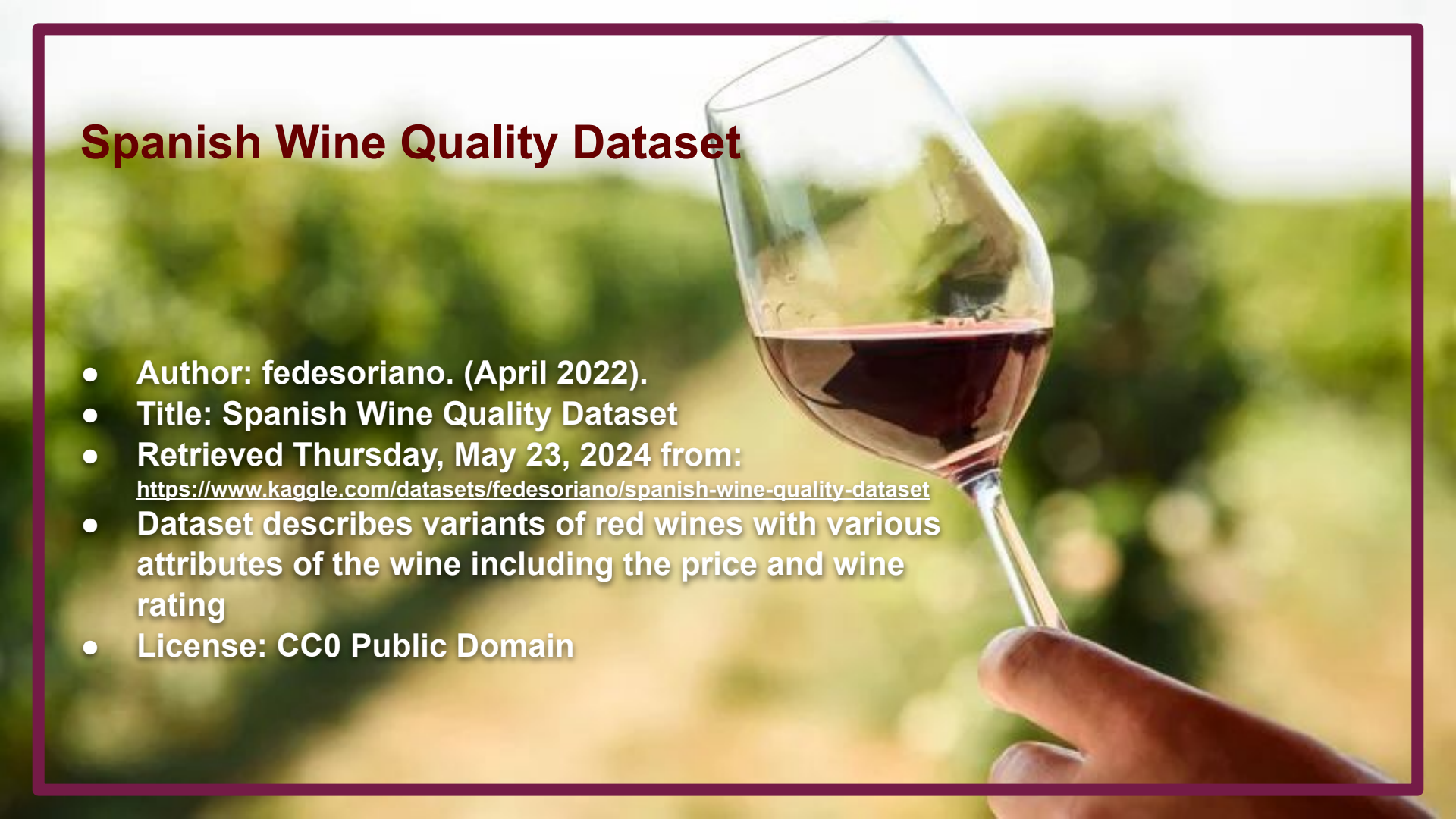

Wine Classification
Chris Covill
Keegan Nohavec
Mike Sheridan

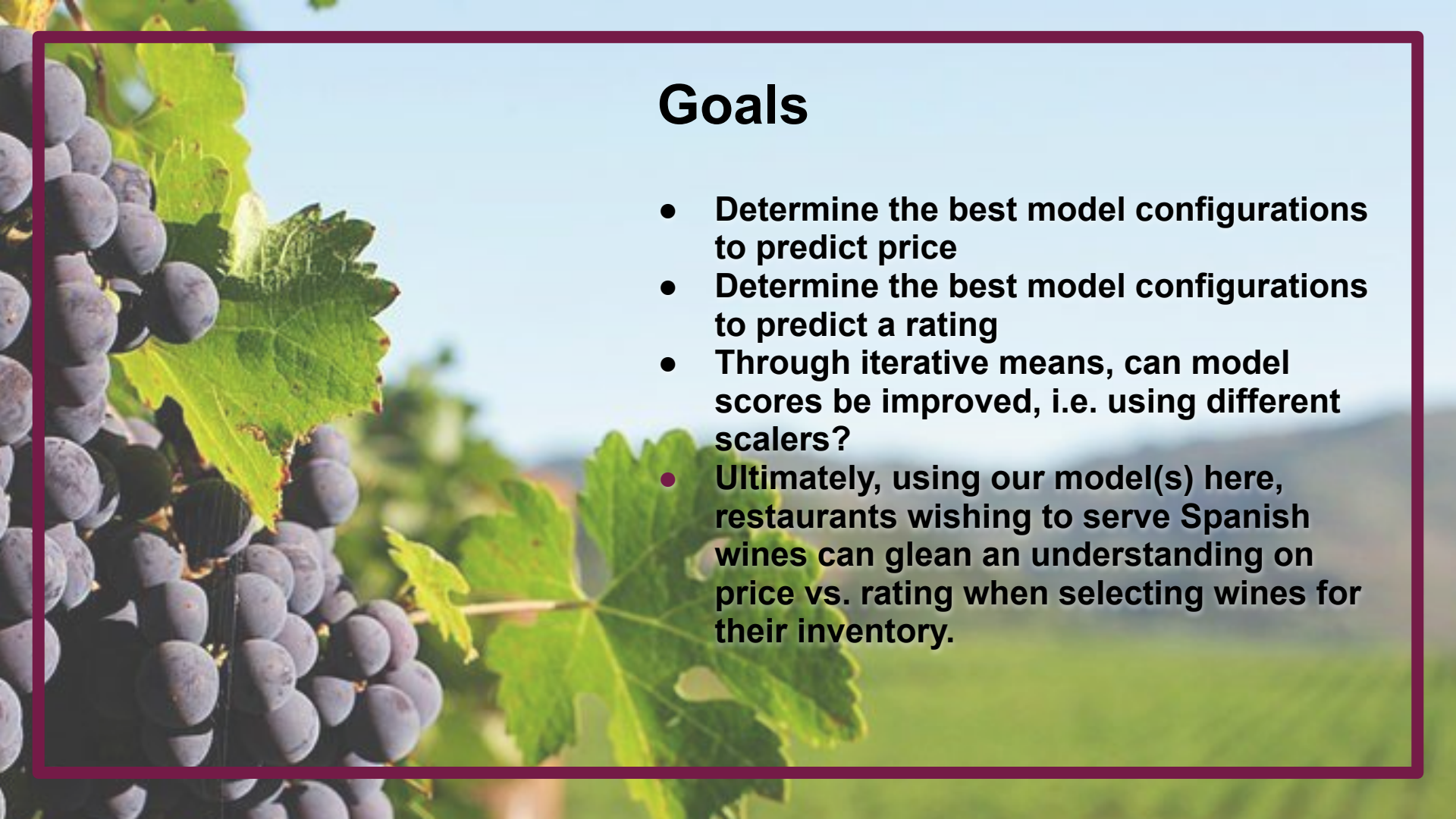
Spanish Wine Quality Dataset



- Author: fedesoriano. (April 2022).
- Title: Spanish Wine Quality Dataset
- Retrieved Thursday, May 23, 2024 from: https://www.kaggle.com/datasets/fedesoriano/spanish-wine-quality-dataset
- Dataset describes variants of red wines with various attributes of the wine including the price and wine rating
- License: CC0 Public Domain

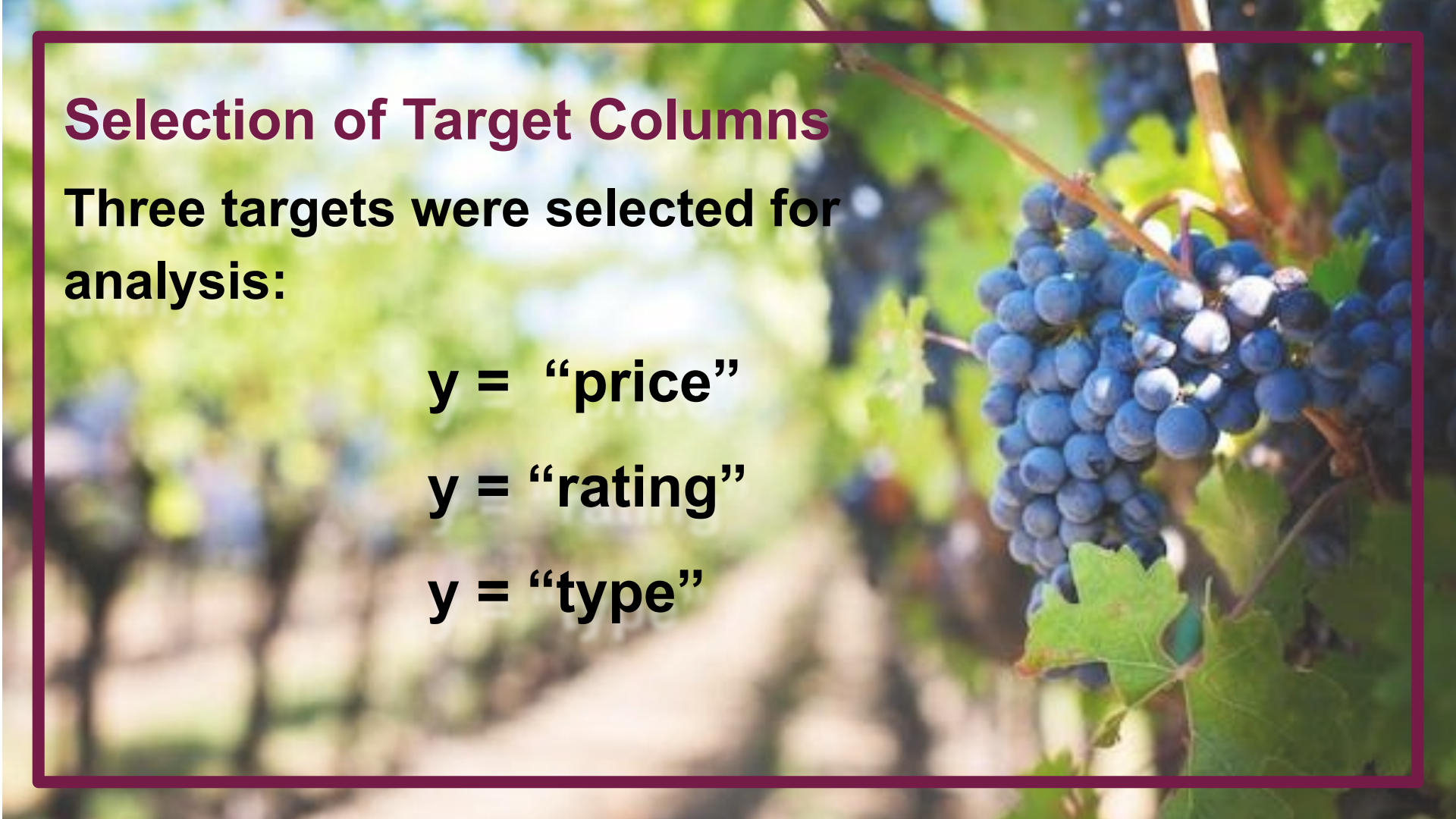
Dataset Features

1. winery: Winery name
2. wine: Name of the wine
3. year: Year in which the grapes were harvested
4. rating: Average rating given to the wine by the users [from 1-5]
5. num_reviews: Number of users that reviewed the wine
6. country: Country of origin [Spain]
7. region: Region of the wine
8. price: Price in euros [€]
9. type: Wine variety
10. body: Body score, defined as the richness and weight of the wine in your mouth [from 1-5]
11. acidity: Acidity score, defined as wine's “pucker” or tartness; [from 1-5]



Goals

- Determine the best model configurations to predict price
- Determine the best model configurations to predict a rating
- Through iterative means, can model scores be improved, i.e. using different scalers?
- Ultimately, using our model(s) here, restaurants wishing to serve Spanish wines can glean an understanding on price vs. rating when selecting wines for their inventory.



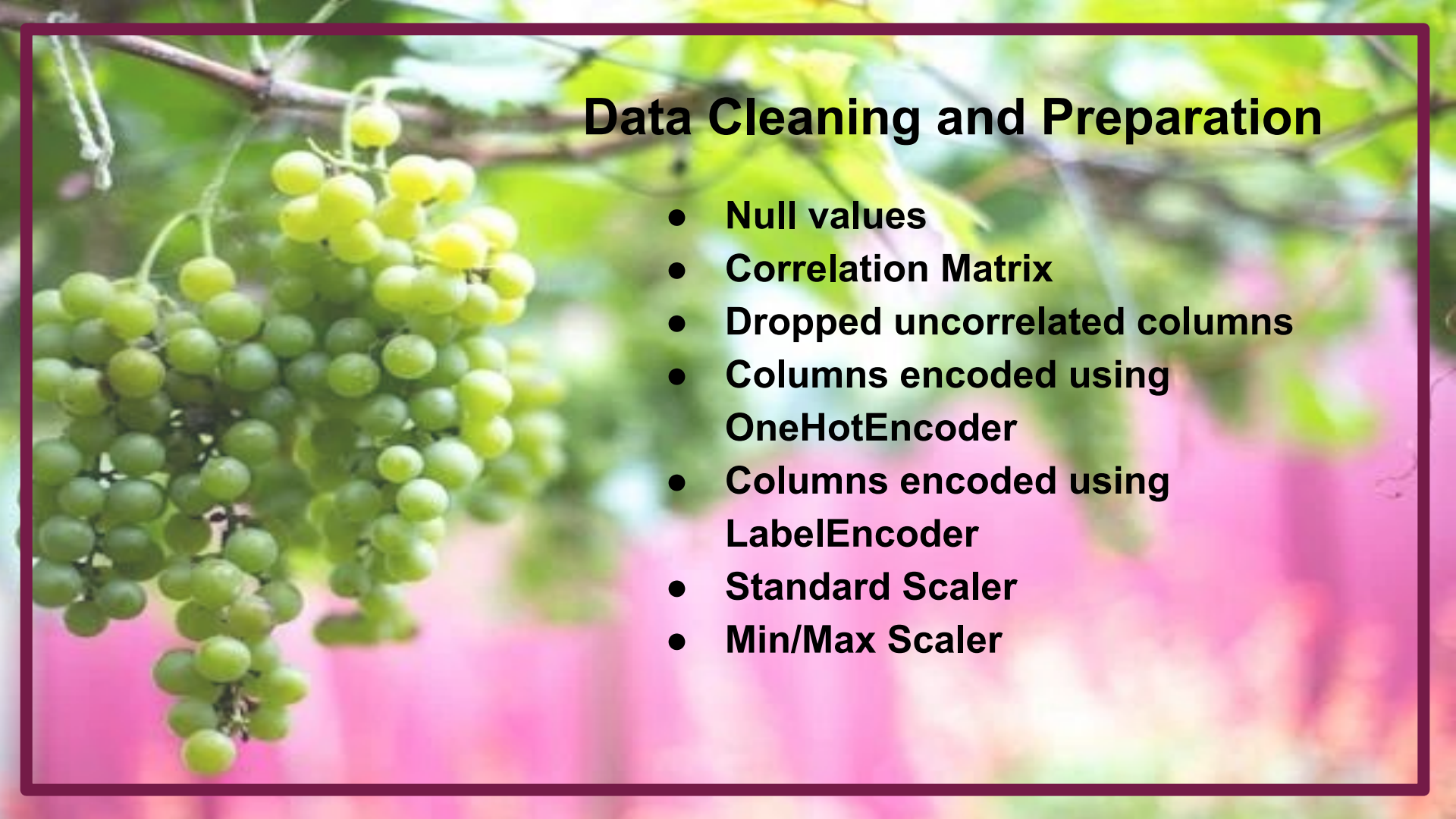
Selection of Target Columns

Three targets were selected for analysis:

y = “price”

y = “rating”

y = “type”



Data Cleaning and Preparation

- Null values
- Correlation Matrix
- Dropped uncorrelated columns
- Columns encoded using OneHotEncoder
- Columns encoded using LabelEncoder
- Standard Scaler
- Min/Max Scaler

Creation of Model Pipeline

- Cleaning and PreProcessing
- Dropna
- Train_test_split
- Drop target column
- K Nearest Neighbor
- Standard Scaler
- OneHotEncoder

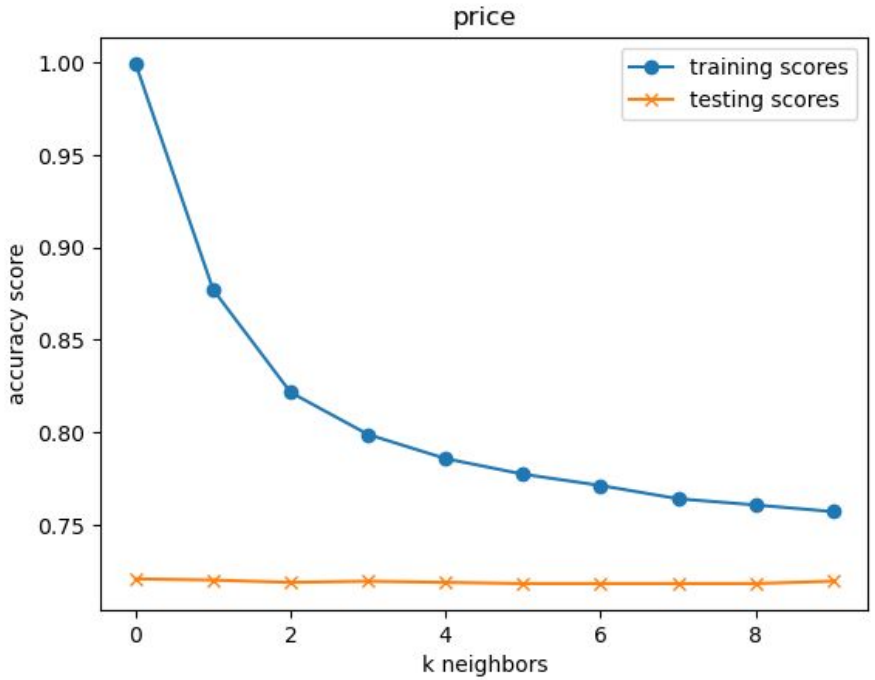
Models

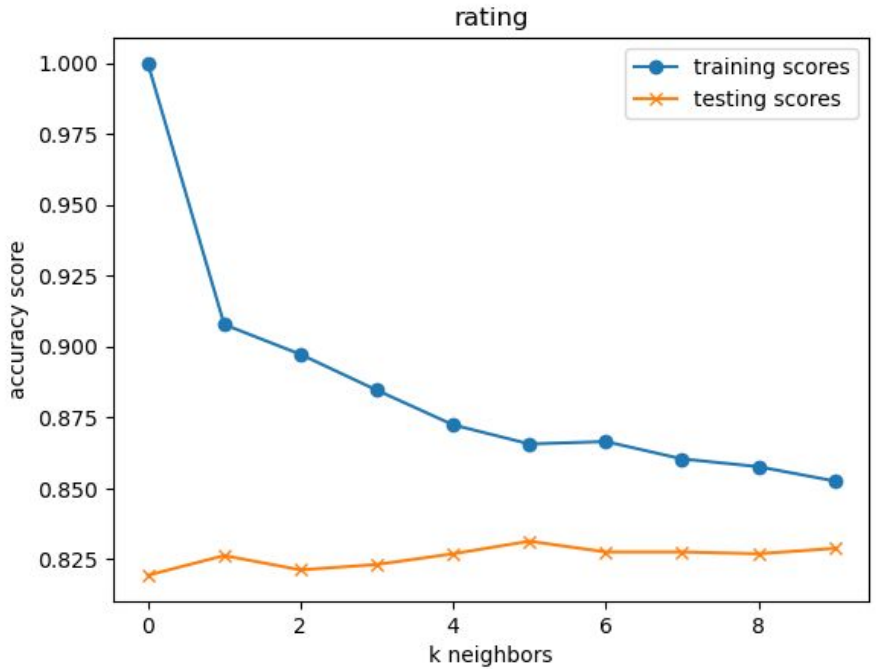
- OneHotEncoder
- Random Forest
- Gradient Boosting
- ADA Boost
- SVM
- Linear Regression

Model Optimization Techniques

- KNN Elbow Graphs
- OneHoteEncoder(‘winery’, ‘wine’, ‘region’, ‘type’, ‘year’)
- LabelEncoder (‘type’)
- StandardScaler
- MinMax Scaler
- Dropping uncorrelated columns (‘country’)

KNN Elbow Analysis





y=’type’: Highly Predictable

KNN - Training Score: 0.9810, R^2 Score: 0.9810, MSE: 0.2517
KNN - Testing Score: 0.9494, R^2 Score: 0.9494, MSE: 0.6739

Random Forest - Training Score: 0.9971, R^2 Score: 0.9971, MSE: 0.0383
Random Forest - Testing Score: 0.9577, R^2 Score: 0.9577, MSE: 0.5635

Gradient Boosting - Training Score: 0.9637, R^2 Score: 0.9637, MSE: 0.4795
Gradient Boosting - Testing Score: 0.9319, R^2 Score: 0.9319, MSE: 0.9068

AdaBoost - Training Score: 0.6630, R^2 Score: 0.6630, MSE: 4.4553
AdaBoost - Testing Score: 0.6371, R^2 Score: 0.6371, MSE: 4.8312

SVM - Training Score: 0.9097, R^2 Score: 0.9097, MSE: 1.1943
SVM - Testing Score: 0.8606, R^2 Score: 0.8606, MSE: 1.8559

Linear Regression - Training Score: 1.0000, R^2 Score: 1.0000, MSE: 0.0000
Linear Regression - Testing Score: 0.9574, R^2 Score: 0.9574, MSE: 0.5673

```python
from ClassificationDataPipeline import train_and_evaluate_models
df = cleaned_df
target_column = 'rating'
Randomstate = 50
accuracy_scores = train_and_evaluate_models(df, target_column, Randomstate)
print(accuracy_scores)
```

✓ 13.1s

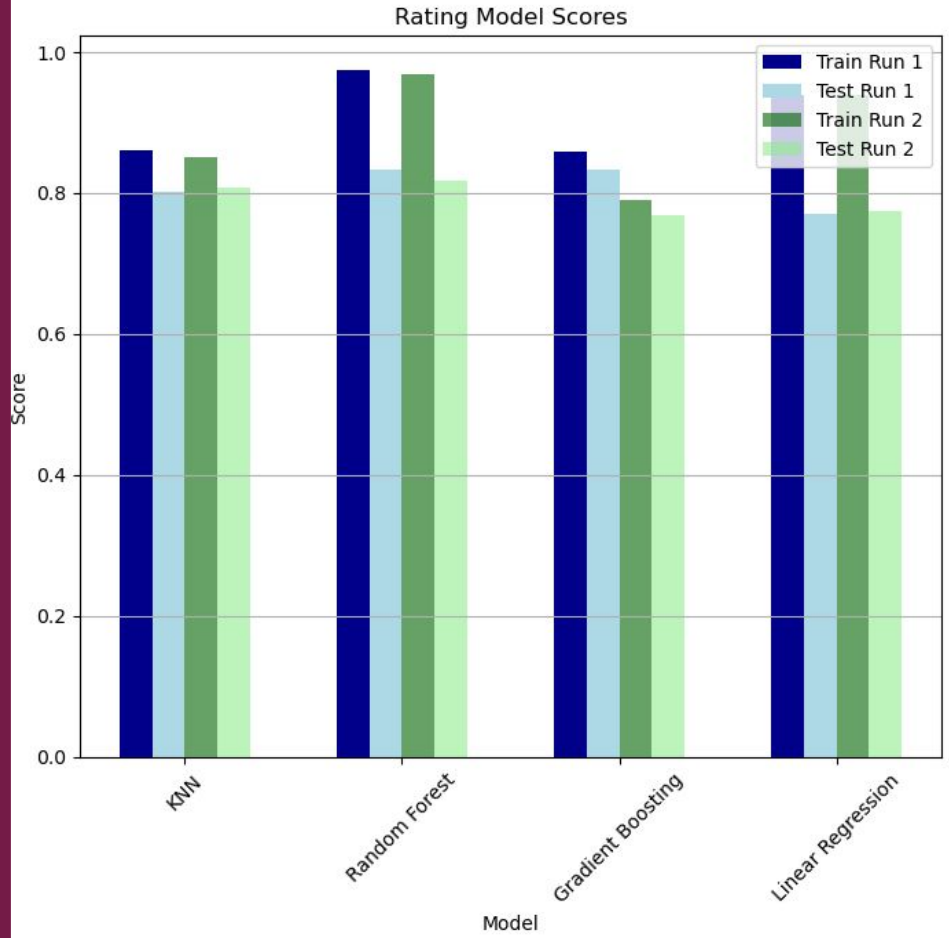
```
KNN - Training Score: 0.8512, R² Score: 0.8512, MSE: 0.0024
KNN - Testing Score: 0.8088, R² Score: 0.8088, MSE: 0.0027
--------------------------------------------------
Random Forest - Training Score: 0.9696, R² Score: 0.9696, MSE: 0.0005
Random Forest - Testing Score: 0.8167, R² Score: 0.8167, MSE: 0.0026
--------------------------------------------------
Gradient Boosting - Training Score: 0.7892, R² Score: 0.7892, MSE: 0.0033
Gradient Boosting - Testing Score: 0.7677, R² Score: 0.7677, MSE: 0.0033
--------------------------------------------------
AdaBoost - Training Score: 0.4684, R² Score: 0.4684, MSE: 0.0084
AdaBoost - Testing Score: 0.4390, R² Score: 0.4390, MSE: 0.0080
--------------------------------------------------
SVM - Training Score: 0.4376, R² Score: 0.4376, MSE: 0.0089
SVM - Testing Score: 0.2898, R² Score: 0.2898, MSE: 0.0102
--------------------------------------------------
Linear Regression - Training Score: 0.9401, R² Score: 0.9401, MSE: 0.0009
Linear Regression - Testing Score: 0.7753, R² Score: 0.7753, MSE: 0.0032
--------------------------------------------------
Final Accuracy, R², and MSE Scores:
Train Scores:
  KNN: Accuracy Score: 0.8512, R² Score: 0.8512, MSE: 0.0024
  Random Forest: Accuracy Score: 0.9696, R² Score: 0.9696, MSE: 0.0005
  Gradient Boosting: Accuracy Score: 0.7892, R² Score: 0.7892, MSE: 0.0033
  AdaBoost: Accuracy Score: 0.4684, R² Score: 0.4684, MSE: 0.0084
  SVM: Accuracy Score: 0.4376, R² Score: 0.4376, MSE: 0.0089
...
  SVM: Accuracy Score: 0.2898, R² Score: 0.2898, MSE: 0.0102
  Linear Regression: Accuracy Score: 0.7753, R² Score: 0.7753, MSE: 0.0032

({'train': {'KNN': 0.8511997263806768, 'Random Forest': 0.9695688300808258, 'Gradient Boosting': 0.7892436840594999, 'AdaBoost': 0.46835435988812824, 'SVM': 0.43759199890118194, 'Linear Regression':
```

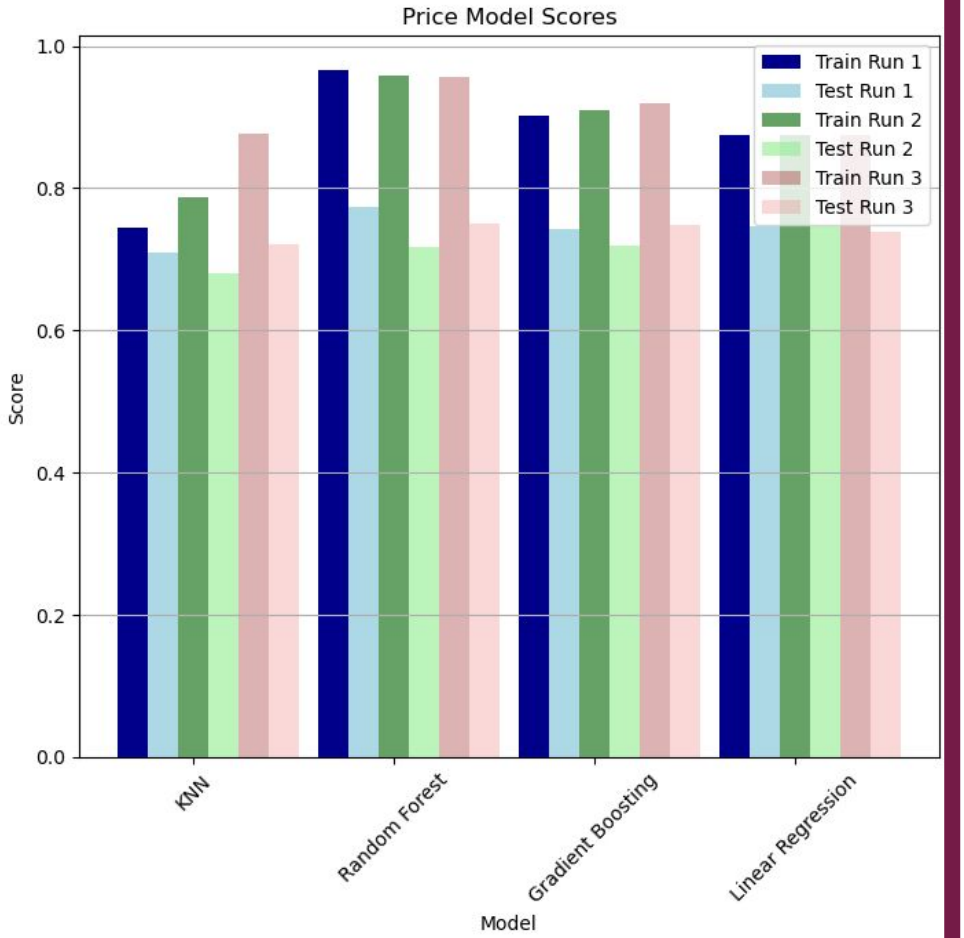
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Linear Regression: 0.7393

AdaBoost: 0.7748

Initial Results vs Model Optimization: y='rating', y='price'





Predicting Price w/Prophet

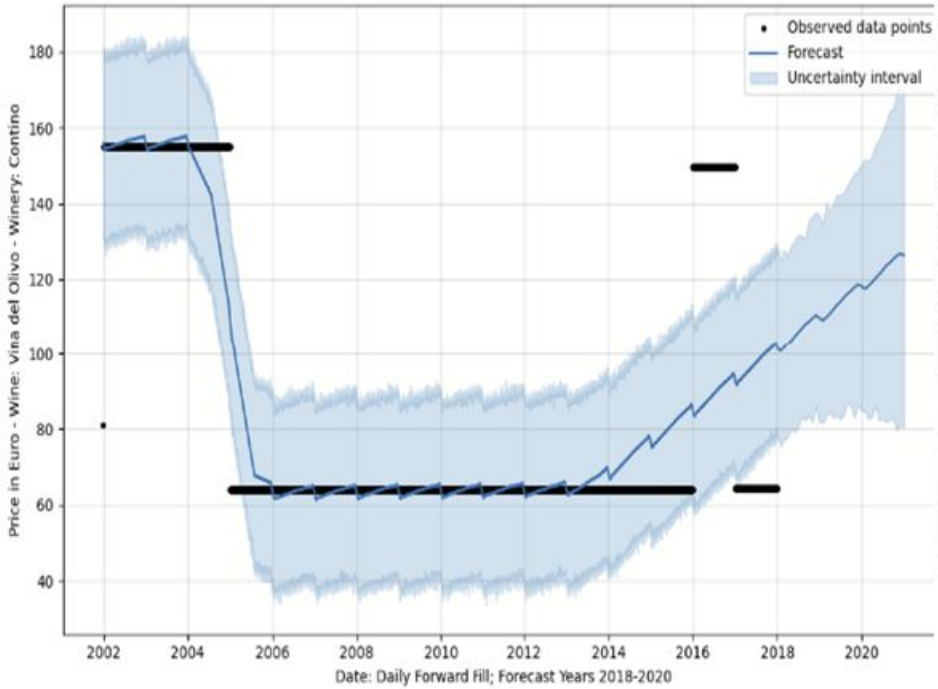
- ✓ **Extensive Data Preparation on Vintage Year .....**

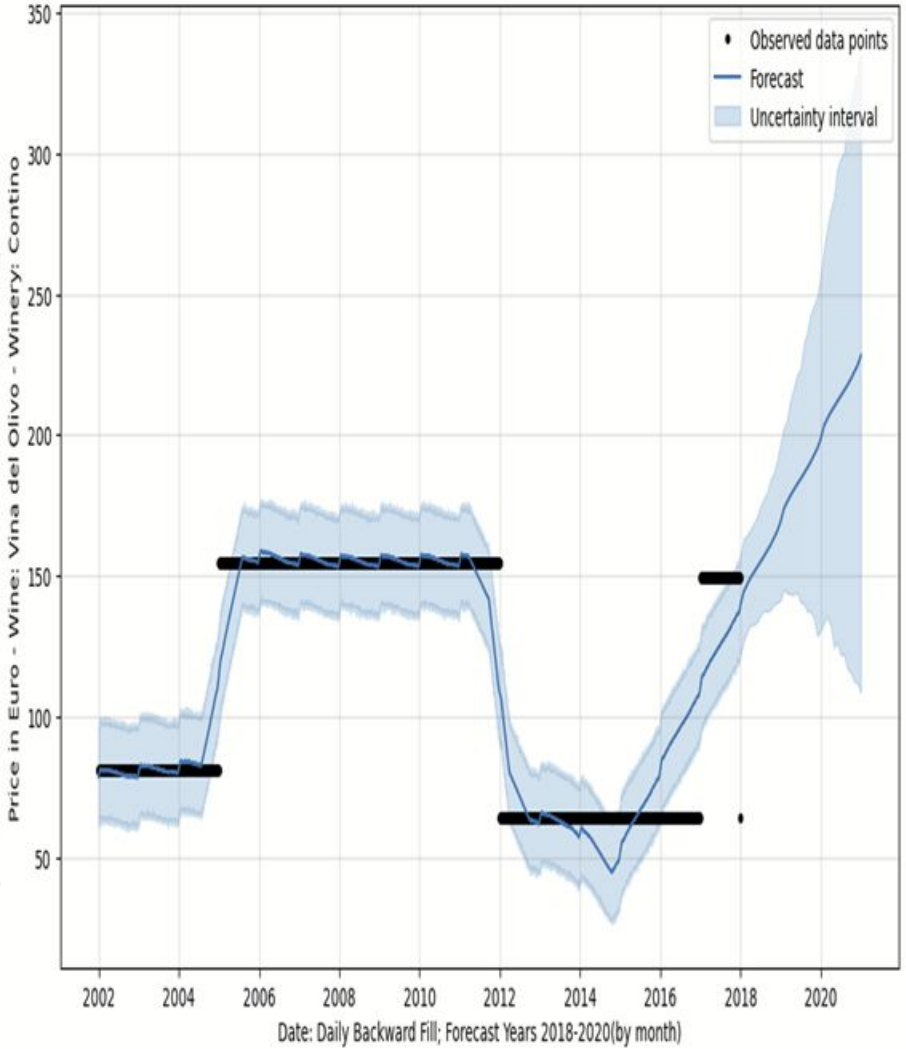
  Dropna
  Convert to string and append month/day to year: YYYY-12-31
  Convert to DateTimeIndex for resample
  Resample by adding days throughout year for each year
  and bfill price from next year, or ffill price from last year

- ✓ **Select Winery: Contino**

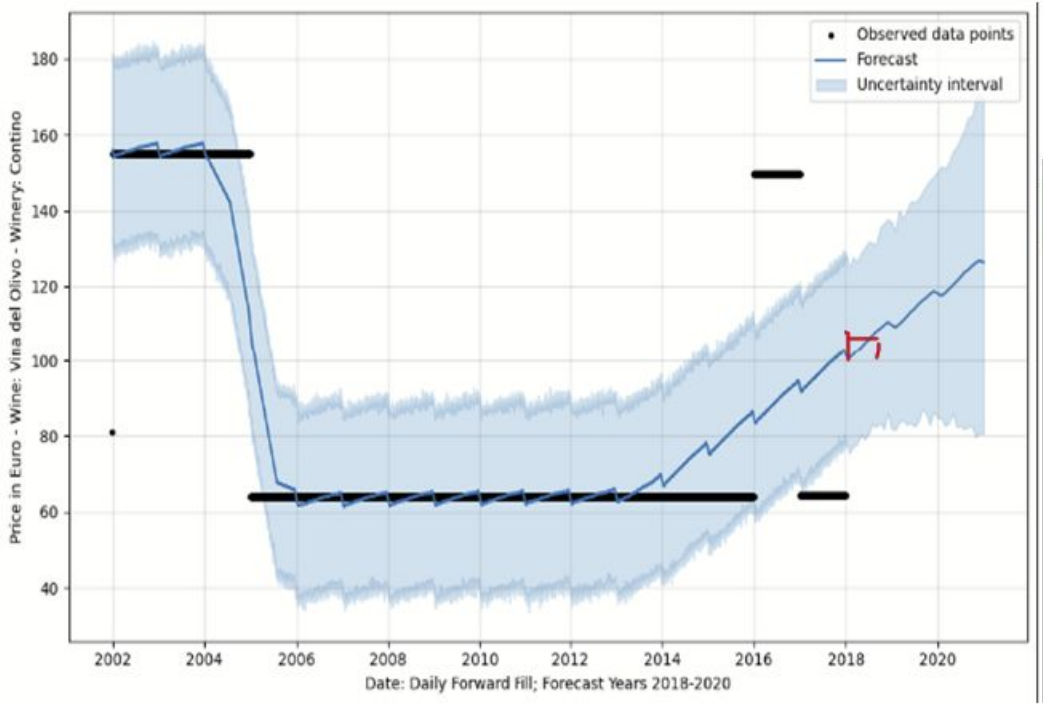
- ✓ **Select Wines: (1) Vino del Olivo (6 rows)**
  (2) Rioja Graciano (202 rows)

Olivio – by month (Ffill/Bfill)



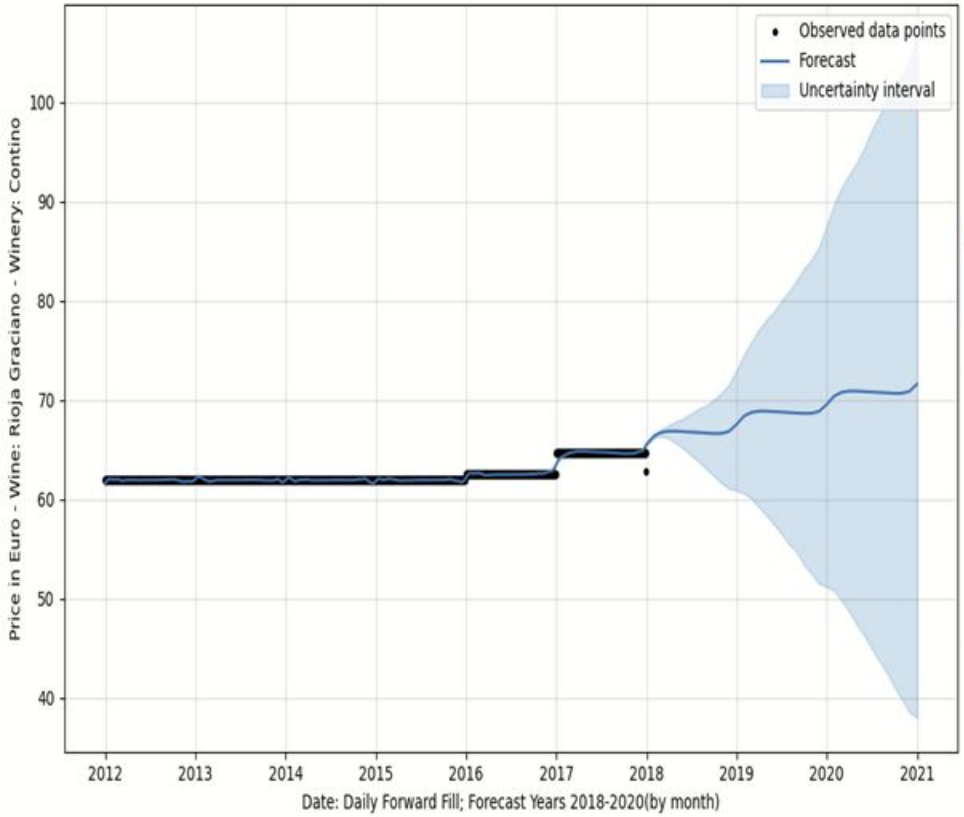


2018 Vintage Olivo Price Forecast



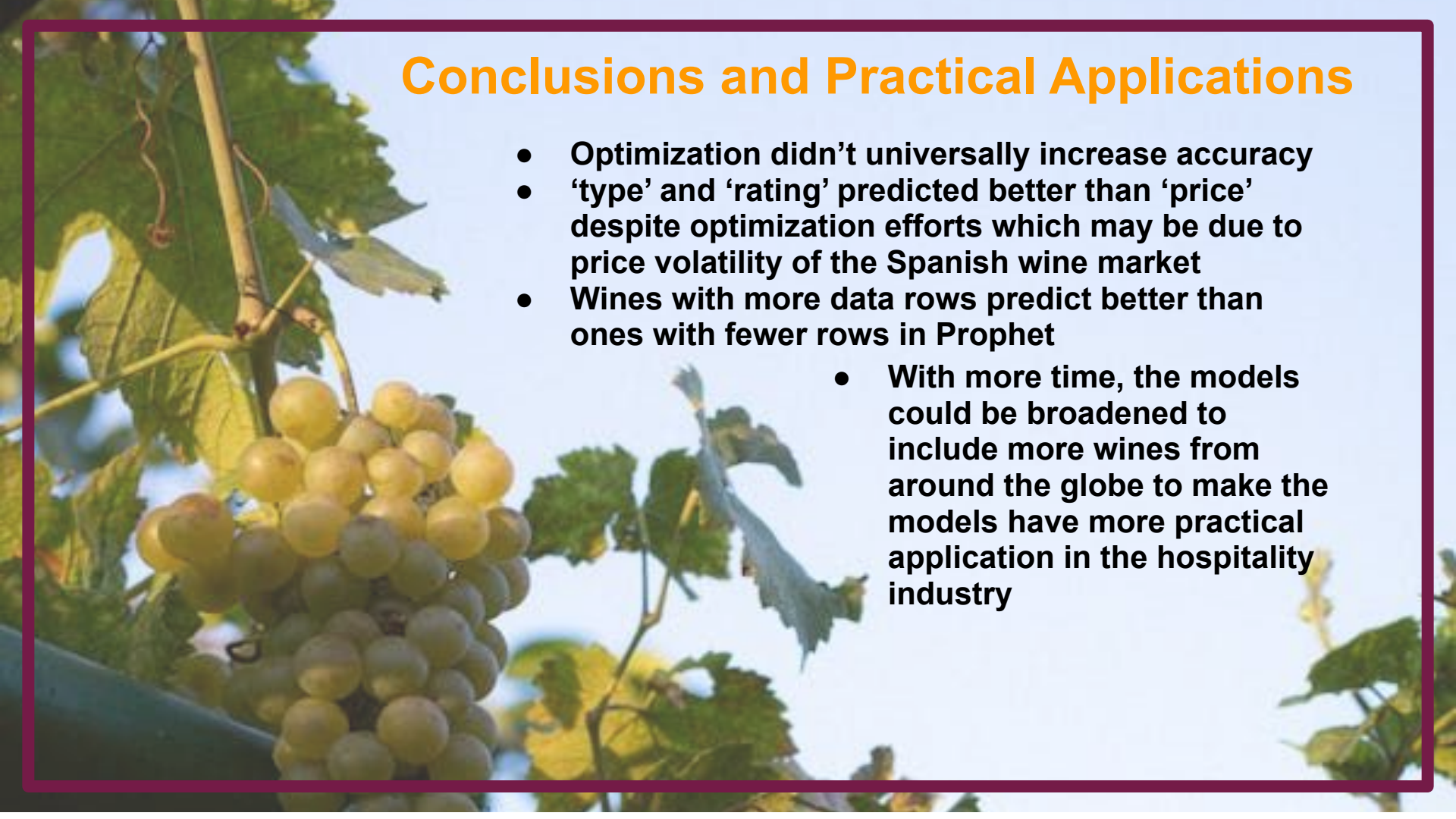
👎 Prophet: $100-105 Euros
Wine Searcher: $66 Euros

2018 Vintage Graciano Price Forecast



👍 Prophet: $65-67 Euros

Wine Searcher: $64 Euros



Conclusions and Practical Applications

- Optimization didn't universally increase accuracy
- 'type' and 'rating' predicted better than 'price' despite optimization efforts which may be due to price volatility of the Spanish wine market
- Wines with more data rows predict better than ones with fewer rows in Prophet
    - With more time, the models could be broadened to include more wines from around the globe to make the models have more practical application in the hospitality industry