Vinho Verde: Machine Learning with Wine Data

August 31, 2021

About Vinho Verde:

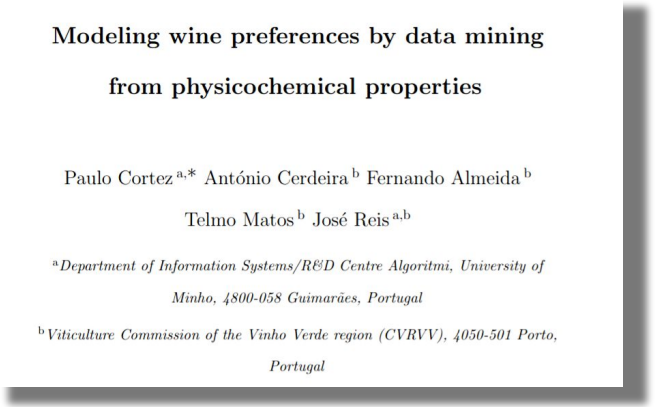
- Not a grape or a wine blend, but a region - Vinho verde gets its name from the lush region of Portugal it is produced in.
- The Viticulture Commission of the Vinhos Verdes Region (CVRVV) is the accredited organization that governs vinho verde certification
- The Minho Valley region in northwestern Portugal is the main home of vinho verde
- Grapes vary locally, but Alvarinho and Loureiro are the most frequent.
- Vinho verde can be white, rosé, red, sparkling, and aged.

Info Source:
https://www.vinhoverde.pt/pt/sobre-o-vinho-verde

Image Source:
https://pixabay.com/photos/lapela-mon%c3%a7%c3%a3o-vineyard-autumn-4569930/

Overview of the Wine Data Set

- Data sourced from the Viticulture Commission of the Vinhos Verdes Region (CVRVV)
- 11 physicochemical attributes (acidity, density, alcohol) and one attribute for overall quality
- About 5000 white wine entries and 1600 red
- Scored on a 0 to 10 scale. Quality in the data ranged from 3 to 9.



Modeling wine preferences by data mining from physicochemical properties

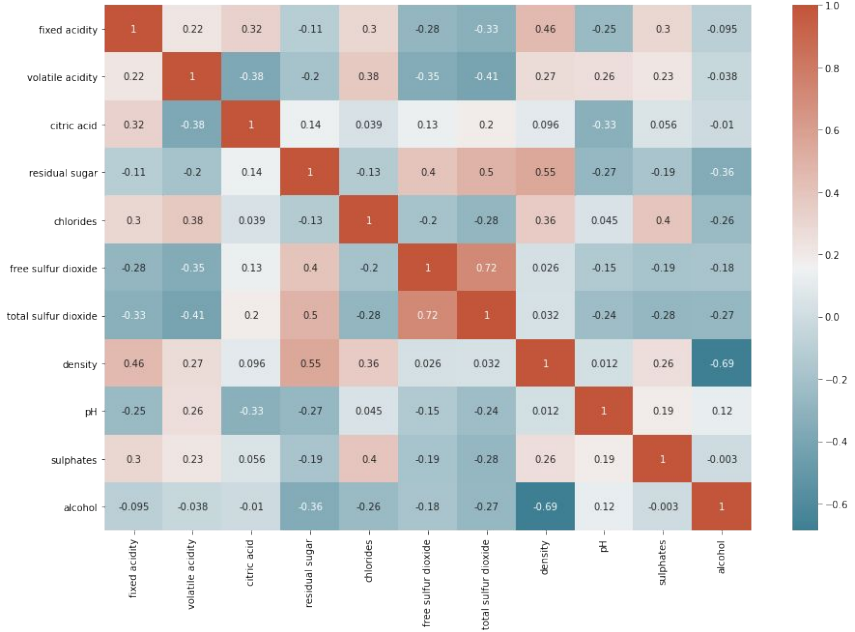
Paulo Cortez [a,*] António Cerdeira [b] Fernando Almeida [b]
Telmo Matos [b] José Reis [a,b]

[a] *Department of Information Systems/R&D Centre Algoritmi, University of Minho, 4800-058 Guimarães, Portugal*

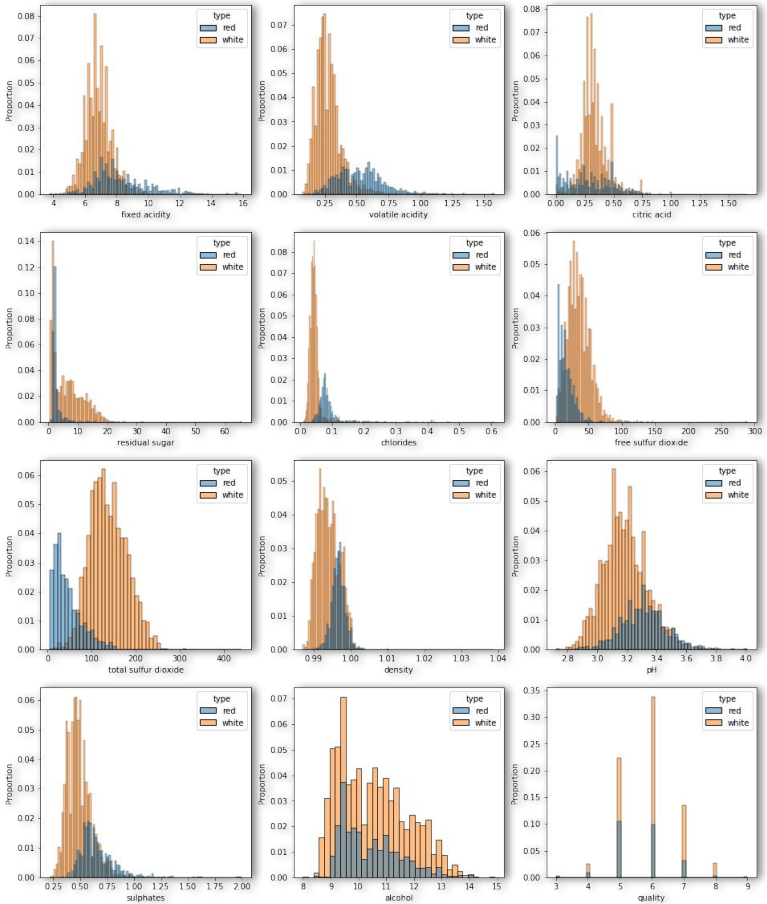
[b] *Viticulture Commission of the Vinho Verde region (CVRVV), 4050-501 Porto, Portugal*

Figures 1-2

Heat Map Correlation of All Wines



Distribution of Variables by Wine Type



Our Project objective:

Find and select the best features that correlate with the outcome of wine quality and create a machine learning model that can predict wine quality given new wine feature data, with reasonable accuracy.

Why Study Wine Data?

Consumers want to drink quality wines. Wine producers want to sell quality wine. Having a machine learning algorithm that can analyze and predict wine quality helps consumers better understand and trust the wine quality ranking process, and it benefits producers who can focus their production towards reaching ideal physicochemical levels.

Image Source: https://pixabay.com/photos/wine-green-portugal-alcohol-360300/

Methods and Process

Working in the Python programming language, we used the SciKit Learn, Numpy, Pandas, Seaborn, SciPy, and Matplotlib modules to conduct our analysis and run our models.

Python 3: Libraries

- **Numpy**
- **Pandas**
- **Seaborn**
- **Matplotlib**
- **Scikit Learn**
- **Scipy**

Step 1) Perform initial analysis of correlation and run models over the data as presented

Step 2) Reclassify quality into three levels representing more comprehensive scores such as “poor, decent, good.”

Step 3) Run supervised models on datasets where the colors are separated as well as where they are combined.

Three of Our Chosen Machine Learning Models

Random Forest

An ensemble model that uses averaging of randomized decision trees to carry out prediction.

Logistic Regression

A linear model that classifies through quality level probability calculation.

SVM

An algorithm model that creates a hyperplane over varying dimensions to separate and classify data points.

Our Modeling Qualities, X:

All 11 features from the data:

fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, PH, sulphates, and alcohol.

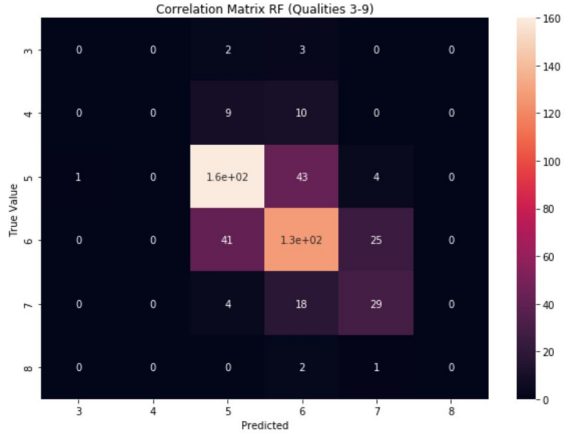
Our Reassigned Categories, y:

3-4 Poor, 5-6 Decent, and 7+ Good

Random Forest

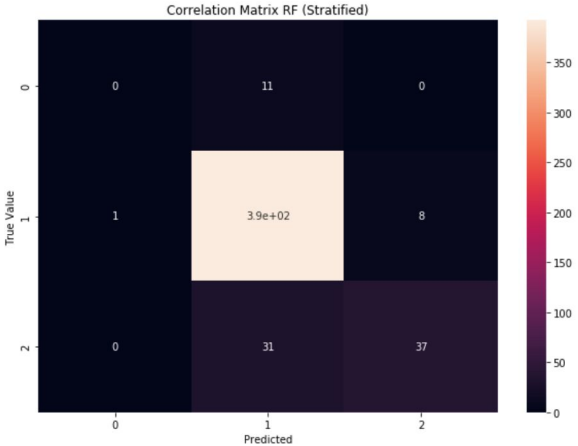
```
X = red.iloc[:,:-1].values
y = red.iloc[:,-1].values

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.3)
from sklearn.ensemble import RandomForestClassifier
random_forest = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='sqrt', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=150, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
random_forest.fit(x_train, y_train)
Y_pred_rf = random_forest.predict(x_test)
accuracy_score(y_test,Y_pred_rf)
```

```
0.6895833333333333
```




```
X = red_copy.iloc[:,:-1].values
y = red_copy.iloc[:,-1].values

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.3)

from sklearn.ensemble import RandomForestClassifier
random_forest = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='sqrt', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=150, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
random_forest.fit(x_train, y_train)
Y_pred_rf = random_forest.predict(x_test)
accuracy_score(y_test,Y_pred_rf)
```

```
0.8833333333333333
```



Figures 3-4

Random Forest on White Wine Dataset

```
from sklearn.ensemble import RandomForestClassifier
random_forest = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='sqrt', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=400, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
random_forest.fit(x_train, y_train)
Y_pred_rf = random_forest.predict(x_test)
accuracy_score(y_test,Y_pred_rf)
```

```
0.8369230769230769
```

Random Forest on Combined Dataset

```
from sklearn.ensemble import RandomForestClassifier
random_forest = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='sqrt', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=400, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
random_forest.fit(x_train, y_train)
Y_pred_rf = random_forest.predict(x_test)
accuracy_score(y_test,Y_pred_rf)
```

```
0.8231292517006803
```

Logistic Regression

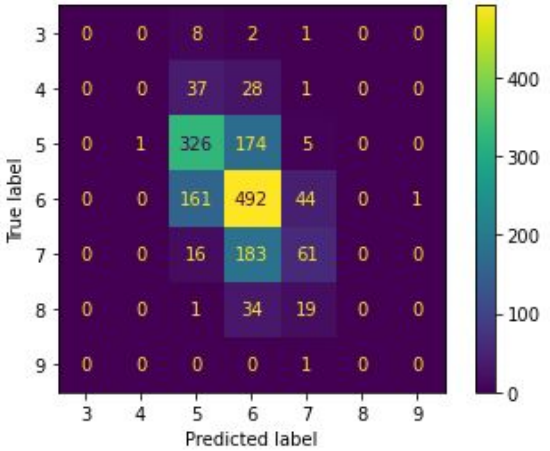
Findings

All Wines: 76.69%

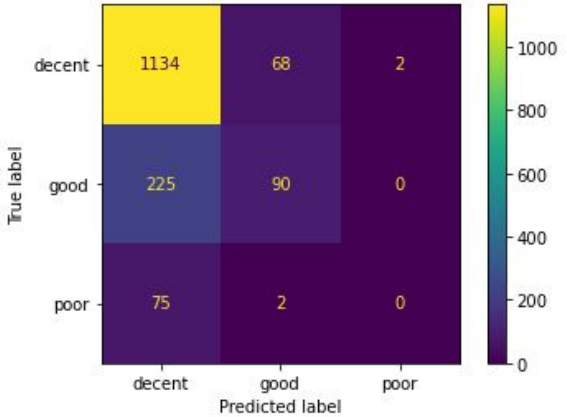
Red: 85.78%

White: 76.20%

***See Code**



Confusion Matrix with Qualities 3-9

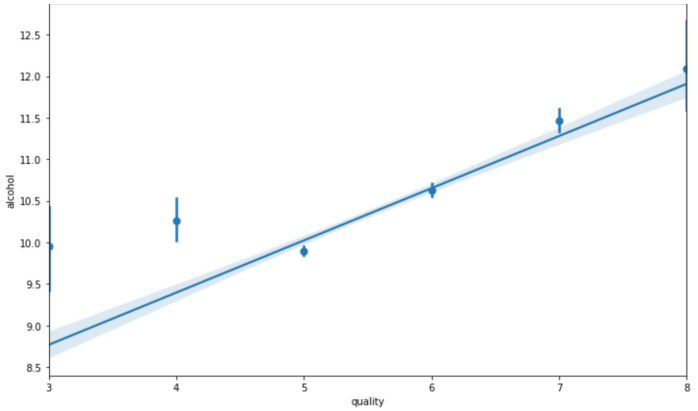


Confusion Matrix with Alternate Scale

Figures 5-6

SVM: Polynomial Kernel

```
#Correlation with output variable
cor = wine.corr()
cor_target = abs(cor["quality"])
#Selecting highly correlated features
relevant_features = cor_target[cor_target>0.45]
relevant_features

Group_names =['bad', 'good', 'great'] #Binning the quality data into 3 characteristics
Categories = pd.cut(wine['quality'], bins = [2,4,6,8], labels= Group_names)
wine['quality'] = Categories

from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC

X = wine.loc[:, ['alcohol']] #Pre-processing step: splitting data into x and y
y = wine.loc[:, ['quality']]

from sklearn.preprocessing import LabelEncoder #Encoding out target: Quality Column
labelencoder_y = LabelEncoder()
y = labelencoder_y.fit_transform(y)

from sklearn.model_selection import train_test_split #splitting dataset into train and test. Tests 20% of data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

from sklearn.preprocessing import StandardScaler #Standardizes the data
SC = StandardScaler()
X_train = SC.fit_transform(X_train)
X_test = SC.fit_transform(X_test)
```



Figures 7-8


```
from sklearn.svm import SVC #Using the poly kernel of SVM to classify the data
CL = SVC(kernel = 'poly', degree = 2)
CL.fit(X_train, y_train)
y_pred = CL.predict(X_test)

from sklearn.metrics import confusion_matrix
ax = plt.subplot()
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm,annot=True,fmt='2.0f')
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels')
ax.set_title('Confusion Matrix');
ax.xaxis.set_ticklabels(['bad', 'good', 'great']); ax.yaxis.set_ticklabels(['bad', 'good', 'great']);

from sklearn.model_selection import cross_val_score #Cross validation to assess the effectiveness of model
accuracies = cross_val_score(estimator = CL, X = X_train,
                             y = y_train, cv = 10)
accuracies.mean() #model accuracy
0.8147022637795276
```

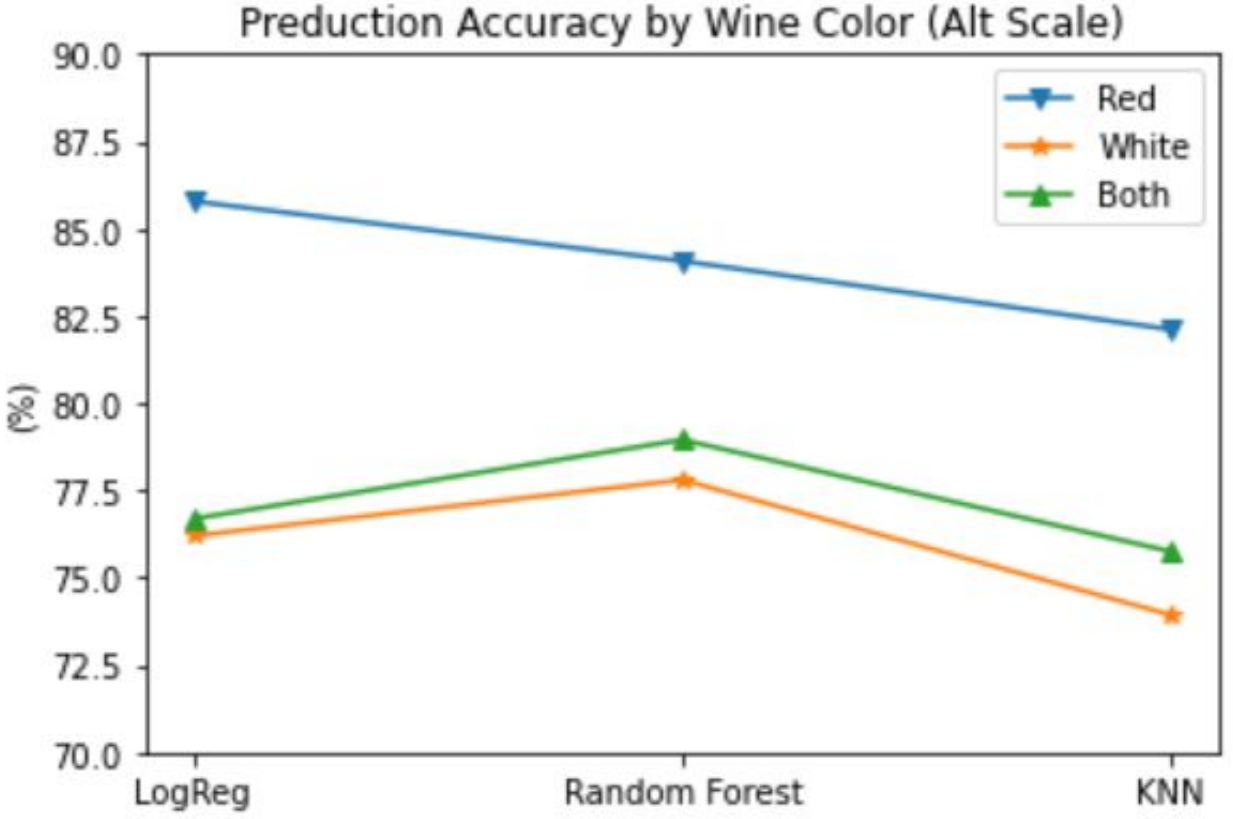
Results Summary

All Wines: From our three models using Random Forest, Logistic Regression, and KNN Modeling:

- We found that while the Random Forest model provides an accuracy score of 56.64% with the 3-9 scale, accuracy shoots to 78.95% with the alternate scale. Specialized RF produced best result: 82.31%

- For Logistic Regression and K-Nearest Neighbors models, the alternate scale sees accuracy scores of 76.69% and 75.75% for the test set, increases of nearly 21% and 30%, respectively.

Figure 9


Prediction Accuracy by Wine Color (Alt Scale)
90.0
87.5
85.0
82.5
80.0
77.5
75.0
72.5
70.0
(%)
Red
White
Both
LogReg
Random Forest
KNN

Other Key Findings

White Wines Only:

- Our models have less success in accurately classifying the alternate scale while performing comparably to "all wines" models with the 3-9 scale. Under the alternate scale, the Logistic Regression, Random Forest, and K-Nearest Neighbors models score 76.20 , 78.55%, and 73.93% respectively.

Red Wines Only:

- Logistic Regression, Random Forest, SVM, and K-Nearest Neighbors models with respect to red wine score 85.78%, 83.82%, 81.47%, and 82.11% under the alternate scale. These figures, as before, considerably outperform those of the 3-9 scale.

Comparison of Time Efficiencies

```
times = [.69, 1.31, .07]

pie, ax = plt.subplots()
ax.pie(times,labels=['LogReg (.69)','Random Forest (1.31)','KNN (.07)'],
       explode=[0,.05,.35], shadow=True, normalize=True)

print(pie)
```

```
Figure(432x288)
```

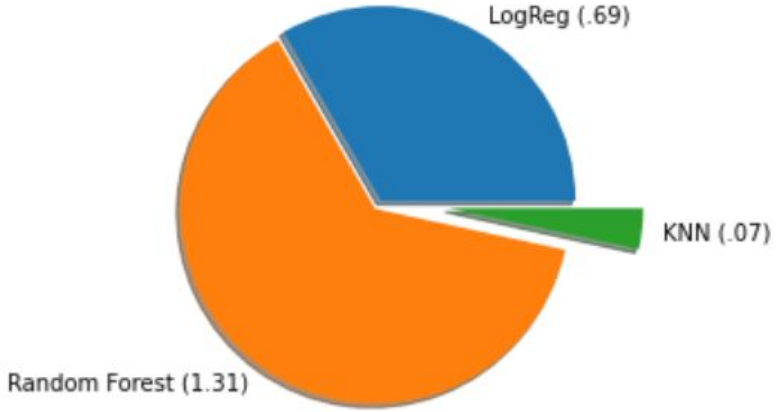


Figure 10

Questions?

Thank you team:

Annie Dang, William Cameron Edge, Nic Landry, Abigail Loy, Erik Martinez, Vladimir Potapenko, Wenhao Zhao