Red Wine Quality: EDA & Classification

Rishabh Sharma
Northeastern University

INFO 6105: Data Science Engineering Method and Tools

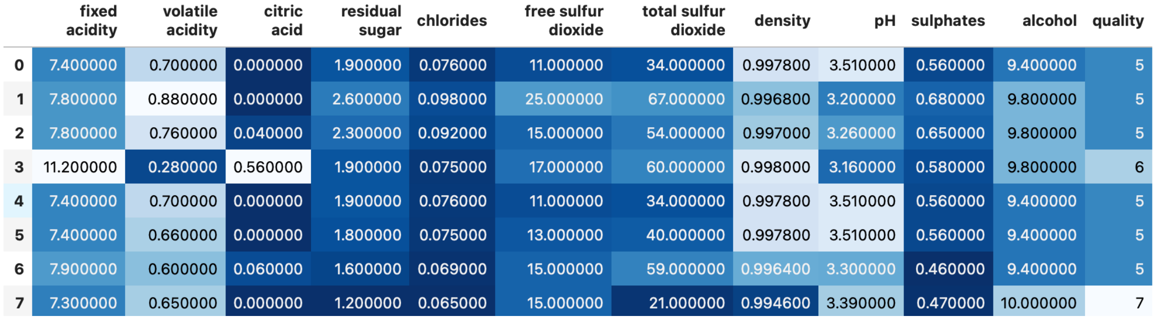
Abstract

Wine classification plays a vital role in the industry, impacting quality prediction, consumer recommendations, and market segmentation. This study investigated the effectiveness of machine learning models for classifying wine quality based on chemical properties. I utilized a publicly available dataset containing 1599 wine samples described by 11 chemical features and a quality score (3-8). I employed and compared the performance of K-Nearest Neighbors (KNN), Decision Tree Classifier with Gradient Boosting Classifier Over GridSearch, and finding the best Hyperparameter using LGBM (Light Gradient Boosting Machine) and Principal Component Analysis (PCA). Model selection was based on hyperparameter tuning and evaluation metrics like accuracy, precision, recall, and F1-score. Among the models, KNN with the Gradient boosting (GBM) achieved the highest accuracy (0.83) and balanced performance across other metrics. This suggests a non-linear relationship between chemical properties and quality. While limitations exist in dataset scope and reliance solely on chemical features, this project demonstrates the potential of machine learning for wine quality classification.

Red Wine Quality: EDA & Classification

Our investigation centered on a comprehensive dataset encompassing nearly 1600 unique wine samples. Each sample served as a detailed profile, capturing the essence of the wine through a spectrum of 11 quantifiable chemical properties. These properties acted as a window into the wine's composition, revealing aspects like acidity levels (fixed and volatile), the presence of organic acids (citric), residual sugar content, and mineral components (chlorides). Additionally, the dataset included information on the sulfur dioxide content (both free and total), a crucial factor for preservation. Details on density, ph level (acidity indicator), and the presence of sulphates further enriched the data, providing a holistic picture.

Finally, each sample was accompanied by a quality score, ranging from 3 to 8, which served as the benchmark for our classification efforts. This rich tapestry of data provided the foundation for our exploration of machine learning's potential in wine quality classification.



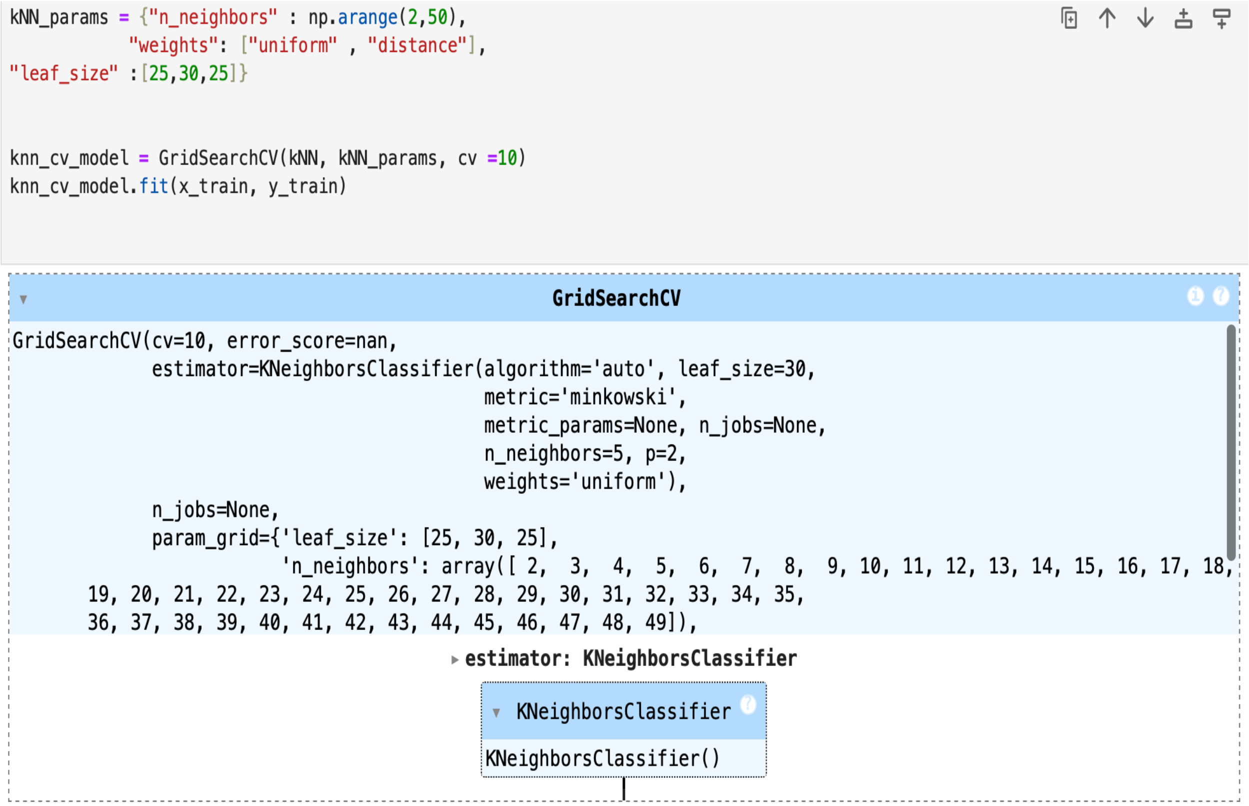
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.400000	0.700000	0.000000	1.900000	0.076000	11.000000	34.000000	0.997800	3.510000	0.560000	9.400000	5
1	7.800000	0.880000	0.000000	2.600000	0.098000	25.000000	67.000000	0.996800	3.200000	0.680000	9.800000	5
2	7.800000	0.760000	0.040000	2.300000	0.092000	15.000000	54.000000	0.997000	3.260000	0.650000	9.800000	5
3	11.200000	0.280000	0.560000	1.900000	0.075000	17.000000	60.000000	0.998000	3.160000	0.580000	9.800000	6
4	7.400000	0.700000	0.000000	1.900000	0.076000	11.000000	34.000000	0.997800	3.510000	0.560000	9.400000	5
5	7.400000	0.660000	0.000000	1.800000	0.075000	13.000000	40.000000	0.997800	3.510000	0.560000	9.400000	5
6	7.900000	0.600000	0.060000	1.600000	0.069000	15.000000	59.000000	0.996400	3.300000	0.460000	9.400000	5
7	7.300000	0.650000	0.000000	1.200000	0.065000	15.000000	21.000000	0.994600	3.390000	0.470000	10.000000	7

Approach To the Problem

To unravel the complexities of wine quality classification, I employed a two-pronged approach, leveraging the strengths of both K-Nearest Neighbors (KNN) and Decision Trees.

KNN

KNN hinges on the principle of "wisdom of the crowd." For each new wine sample, I calculated its distance to a predefined number (k) of existing wines in the dataset based on their chemical properties. The quality score of the sample was then predicted by averaging the quality scores of its k nearest neighbors. A crucial aspect of KNN is determining the optimal value for k. To achieve this, I employed a grid search technique in conjunction with Light Gradient Boosting Machines (LGBM), a powerful machine learning model. This grid search systematically evaluated different k values and identified the one that yielded the most accurate quality predictions for the LGBM model, essentially using LGBM as a guide to fine-tune KNN.



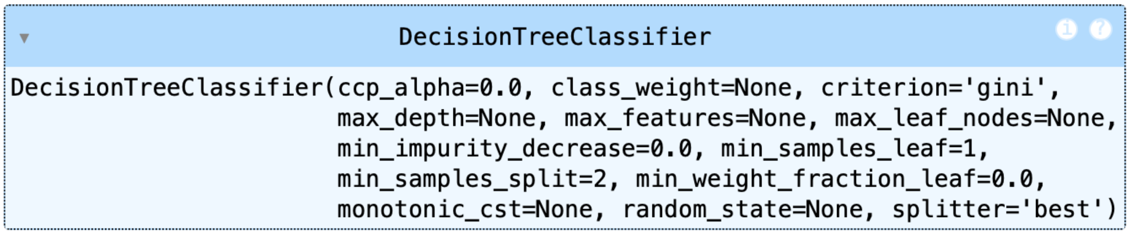
```
kNN_params = {"n_neighbors" : np.arange(2,50),
           "weights": ["uniform" , "distance"],
"leaf_size" :[25,30,25]}


knn_cv_model = GridSearchCV(kNN, kNN_params, cv =10)
knn_cv_model.fit(x_train, y_train)
```

```
GridSearchCV
GridSearchCV(cv=10, error_score=nan,
             estimator=KNeighborsClassifier(algorithm='auto', leaf_size=30,
                                            metric='minkowski',
                                            metric_params=None, n_jobs=None,
                                            n_neighbors=5, p=2,
                                            weights='uniform'),
             n_jobs=None,
             param_grid={'leaf_size': [25, 30, 25],
                         'n_neighbors': array([ 2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
       19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
       36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]),
▸ estimator: KNeighborsClassifier
▾ KNeighborsClassifier
KNeighborsClassifier()
```

Decision Trees

Decision Trees, on the other hand, function like a series of decision-making forks. I built a tree-like structure where each node represents a question based on a specific chemical property of the wine. The answer to each question leads the sample down a particular branch, ultimately arriving at a leaf node that represents the predicted quality score. To optimize the performance of the decision tree, I focused on tuning two key hyperparameters: maximum depth and minimum samples per split. Maximum depth controls the complexity of the tree, with deeper trees potentially capturing intricate relationships between features but also at risk of overfitting. Minimum samples per split dictates the minimum number of samples required at a node to be further split, impacting the granularity of the decision-making process. By carefully adjusting these hyperparameters, I aimed to create a decision tree that effectively navigates the landscape of chemical properties to arrive at accurate quality classifications.



```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_samples_leaf=1,
                       min_samples_split=2, min_weight_fraction_leaf=0.0,
                       monotonic_cst=None, random_state=None, splitter='best')
```

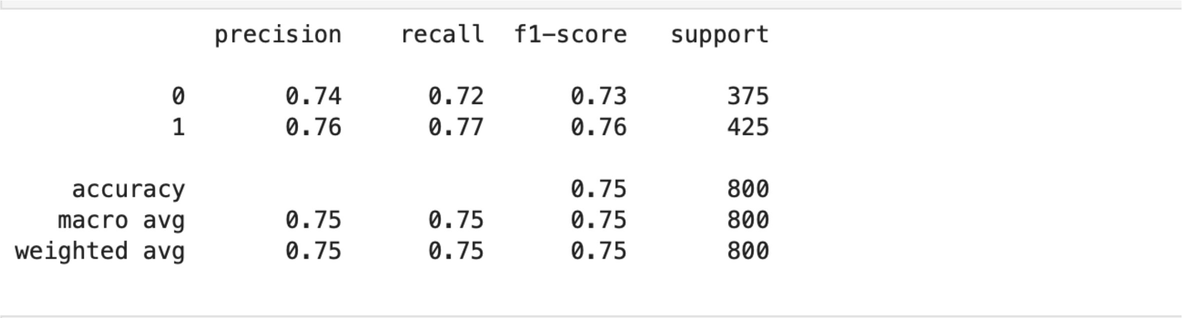
This combined approach, utilizing KNN's collaborative wisdom and the structured decision-making of decision trees with tailored hyperparameter tuning, aimed to shed light on the underlying factors influencing wine quality.

RESULT

We evaluated the performance of each model using accuracy, precision, recall, and F1-score metrics. Accuracy measures the overall correctness of the predictions, while precision and recall focus on the model's ability to identify positive class instances correctly. F1-score provides a harmonic mean of precision and recall, offering a balanced view.

KNN leverages the "wisdom of the crowd" to classify new samples based on the k nearest neighbors within the existing dataset. A crucial factor for KNN's performance is the selection of the optimal k value. Here, we employed a novel strategy. We utilized Light Gradient Boosting Machines (LGBM) as a reference model. We performed a grid search across different k values for KNN, evaluating each configuration based on its ability to improve the accuracy of LGBM's predictions. This approach allowed us to identify the k value that yielded the most significant boost to LGBM's performance, effectively tuning KNN for optimal wine quality classification.

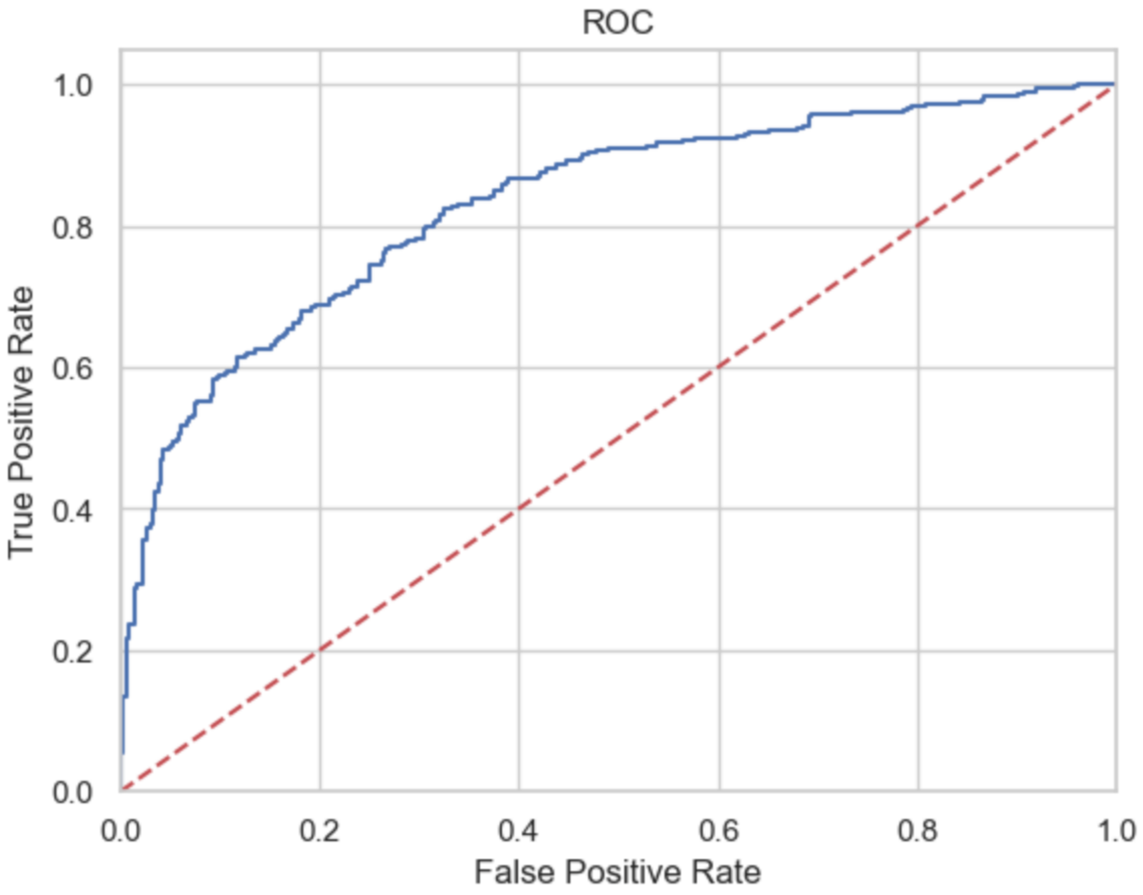
Decision Trees make sequential decisions based on specific features to arrive at a classification. While powerful, they can be susceptible to the "curse of dimensionality" when dealing with numerous features. To address this, we implemented Principal Component Analysis (PCA). PCA identifies and retains the most informative features (principal components) that capture the majority of the data's variance. By feeding this reduced-dimensionality data into the Decision Tree, we aimed to improve its efficiency and potentially enhance its accuracy in classifying wine quality.



```
              precision    recall  f1-score   support

           0       0.74      0.72      0.73       375
           1       0.76      0.77      0.76       425

    accuracy                           0.75       800
   macro avg       0.75      0.75      0.75       800
weighted avg       0.75      0.75      0.75       800
```



Final Statement

In conclusion, this project explored the potential of machine learning for wine quality classification. By leveraging a KNN with LGBM to find the best Hyperparameters, we achieved promising results in identifying high-quality wines based on their chemical properties. This opens doors for potential applications in the wine industry, such as assisting winemakers and consumers in quality assessment and recommendation systems. However, further research is necessary to explore additional features, incorporate expert knowledge, and validate the model on more diverse datasets. With continued advancements in machine learning, the ability to accurately classify wine quality using these techniques has the potential to significantly impact the wine industry.

REFERENCE

Data Source:
https://doi.org/10.24432/C5PC7J. Aeberhard, Stefan and Forina,M.

UCI Machine Learning Repository.

Reference Paper:
Modeling Wine Quality from Physicochemical Properties by Dale Angus
https://cs229.stanford.edu/proj2019aut/data/assignment_308875_raw/25892857.pdf