IS733 – DATA MINING
HOMEWORK 2

Submitted By
PRATHYUSHA HARISH KUMAR
[JB24771]

PART 1. REFLECTIONS ON HOMEWORK 1 (10 POINTS)

From the feedback you received, what are the takeaways/lessons learned you could apply to future analysis?

I received feedback about including a GitHub link for my work in my Homework-1 document. By providing a link to my repository, I can ensure that others can easily review my analysis, understand my approach. It's a small but significant step toward making my projects more transparent and useful. Moving forward, I will make it a habit to always include relevant links and references so that my work is well-documented and easy to access.

Aside from this, I didn't receive much feedback on other aspects of my analysis, which is encouraging and suggests that my approach was generally on the right track. However, I know there is always room for improvement. I'll continue to refine my work by paying closer attention to the finer details, ensuring my visualizations are well-labeled and easy to interpret, and making my explanations as clear as possible. My goal is to not just present data but to communicate insights effectively, making sure that anyone who looks at my work can understand the key takeaways without confusion.

PART 2. CREATE A MODEL CARD (30 POINTS)

The model card contains information about the properties of the models. This is one way of organizing the knowledge about the model, which becomes handy in data science problem-solving. Prepare a table summarizing the properties of each base model we learned so far (Decision tree, Naive Bayes, K-nearest neighbor, logistic regression, SVM) with respect to the following properties:

1. *parametric or non-parametric*
2. *Input (continuous or discrete or both or mixed)*
3. *Output (continuous or discrete or both)*
4. *Can the model handle missing value*
5. *Model representation*
6. *Model Parameters*
7. *How to make the model more complex*
8. *How to make the model less complex*
9. *Is the model interpretable or transparent*

Your table can be organized into rows and columns, rows are the properties and column are the models. This table can be expanded in the future when you learn additional new models. You may use the given template for your model card.

Sl.No	Property	Decision Tree	Naive Bayes	K-Nearest Neighbour	Logistic Regression	Support Vector Machine (SVM)
1.	Parametric/Non-parametric	Non-Parametric [It doesn't assume any fixed equation, it builds rules from data to make decisions.]	Parametric [Looks at how likely things are based on past data. It assumes the data follows a probability distribution.]	Non-Parametric [It just remembers all the data and finds the closest match, no fixed formula.]	Parametric [Draws a straight (or curved) line to split things. It assumes a mathematical relationship between features and the result.]	Can be both. [Finds the best divider (a line or a curve) between groups. If using a simple line, it's parametric. If using curves (kernels), it's non-parametric.]
2.	Input	Both (Continuous & Discrete) [A tree can split any type of data easily.]	Both (mostly categorical) [It works best when features are separate (like "Rainy" vs. "Sunny"), not mixed numbers.]	Both [It just measures distances, so it doesn't care if it's numbers or labels.]	Both [Uses math equations, so it works well with numbers & categories.]	Both [Uses math to draw boundaries, so it can handle both.]
3.	Output	Both (continuous & discrete) [It can predict a number (like price) or a category (like "Yes/No").]	Discrete (classification only) [It's used for classification, so the answer is always a category.]	Both (continuous & discrete) [It can return the most common neighbor (category) or average value (number).]	Discrete (classification) [Used for classification, so output is a label.]	Discrete (classification) [Designed to separate categories, so the output is a label.]
4.	Handle Missing Value	Yes, but better if	No, it	No,	No, needs	No, needs

		you fix missing data. [It can skip missing values or use "backup" rules.]	struggles with missing values. But Yes, to some extent (by ignoring values or through imputation). [It relies on exact probability calculations, which need complete data.]	missing values cause problems. [If data is missing, it can't compare distances properly.]	complete data. [Since it uses formulas, missing data makes calculations impossible.]	complete data. [Missing values affect the boundary calculations]
5.	Model Representation	Tree structure [It builds a decision tree to classify things.]	Probabilistic table [It stores probabilities for each category and uses them to predict.]	Memory-based (data points) [It keeps all the training data and checks the closest matches.]	Math equation [It creates a formula that best fits the data.]	Hyperplane [It finds a divider (line or curve) that best separates the groups.]
6.	Model Parameters	Splitting criterion, Maximum tree depth, Minimum sample split and sample leaf. [You can change how deep the tree grows or what conditions it uses to split data.]	X	Number of neighbors, distance metric, weighting function. [You can decide how many neighbors to consider or how to measure closeness.]	Weights, Bias, Regularization Parameter, Learning Rate. [Weights determine how important each input is.]	Kernel function, support vectors, Weight, Bias. [The kernel controls the shape of the divider]
7.	Make the Model More Complex	Increase tree depth. [A deeper tree learns more patterns but may overfit.]	X	Increase K, use better distance rules. [More	Add polynomial features, decrease regularization. [	Use non-linear kernels (RBF, polynomial). [Non-

				neighbors mean better smoothing but slower predictions.]	More features let it find complex patterns.]	linear kernels allow it to separate data in more complex ways.]
8.	Make the Model less Complex	Limit the tree depth, Use fewer features of prevent overfitting. (cut extra branches) [Cutting unnecessary parts makes it generalize better.]	X	Use fewer neighbors, simpler distance measures. [Using fewer neighbors makes it react faster but less accurately.]	Increase regularization (L1/L2). [More regularization prevents overfitting.]	Use linear kernel, increase margin. [A linear boundary is easier to understand and compute.]
9.	Interpretability/Transparency	Highly interpretable and transparent. [You can follow the tree like a simple set of "Yes/No" questions.]	Moderately interpretable. [It's based on probabilities, which are easy to understand but require math.]	Lightly interpretable and transparent. [Since it stores all data and works by comparisons, it's not easy to explain.]	Moderately interpretable and transparent. [It involves mathematical equations, so it's not as intuitive.]	Less interpretable especially with non linear kernels. [The math behind SVM is complex, especially with fancy curves.]

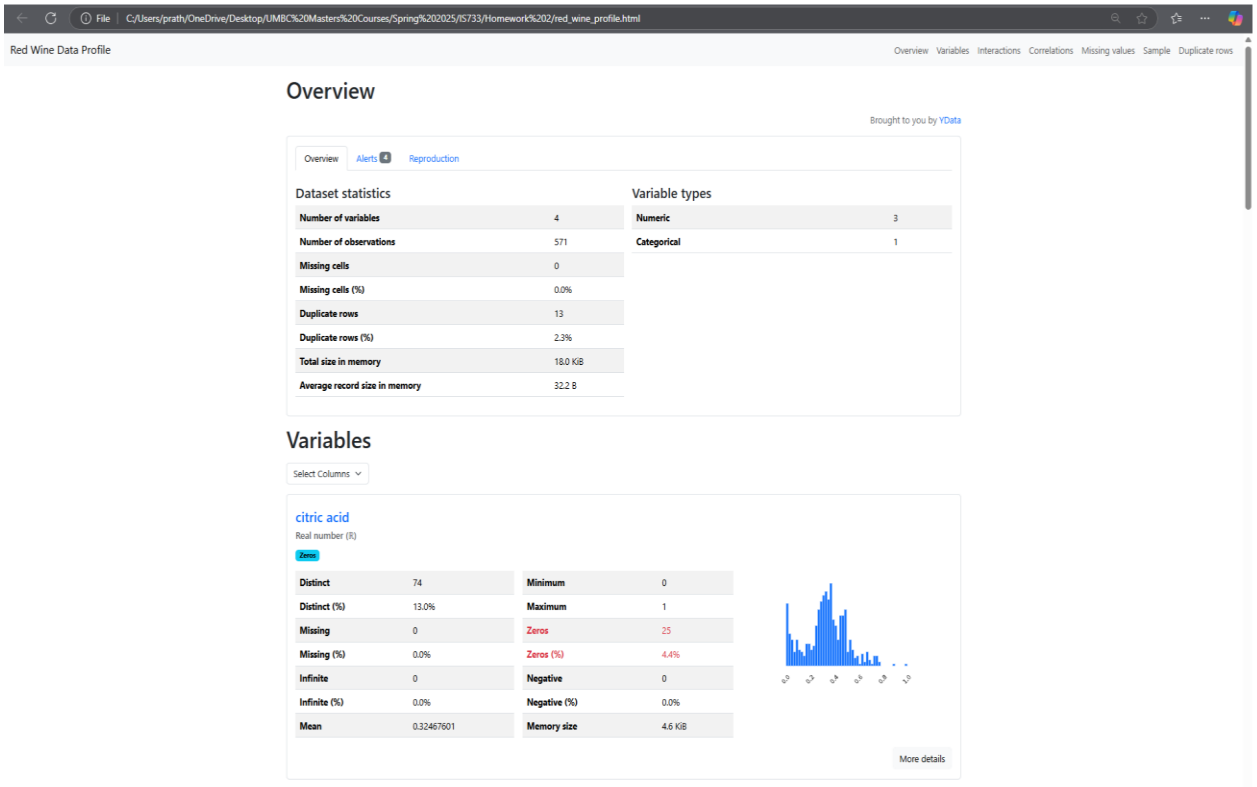
PART 3. WINE-TASTING MACHINE

In this task, we will practice building supervised machine learning with Logistic Regression (LR), Naïve Bayes (NB), Support Vector Machine (SVM), and Decision tree (DT), Random Forest (RF) classifiers, as compared with simple/baseline methods. The data for this exercise comes from the wine industry. Each record represents a sample of a specific wine product, the input attributes include its organoleptic characteristics, and the output denotes the quality class of each wine: {high, low}. The labels have been assigned by human wine-tasting experts, and we can treat that information as "ground truth" in this exercise. Your job is to build the best model to predict wine quality from its characteristics so that the winery can replace the costly services of professional *sommeliers with your automated alternative to enable quick and effective quality tracking of their wines at production facilities. They need to know whether such change is feasible and what extent inaccuracies may be involved in using your tool.*

You will be asked to run experiments in Python.
You are given two datasets red-wine.csv and white-wine.csv

PYTHON TASKS (60 POINTS)

1. *Read* ***red-wine.csv*** *into Python as a data frame, use a pandas profiling tool (https://github.com/pandas-profiling/pandas-profiling) to create an HTML file, and paste a screenshot of the HTML file here (10 points)*



Overview

Dataset statistics

Number of variables	4
Number of observations	571
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	13
Duplicate rows (%)	2.3%
Total size in memory	18.0 KiB
Average record size in memory	32.2 B

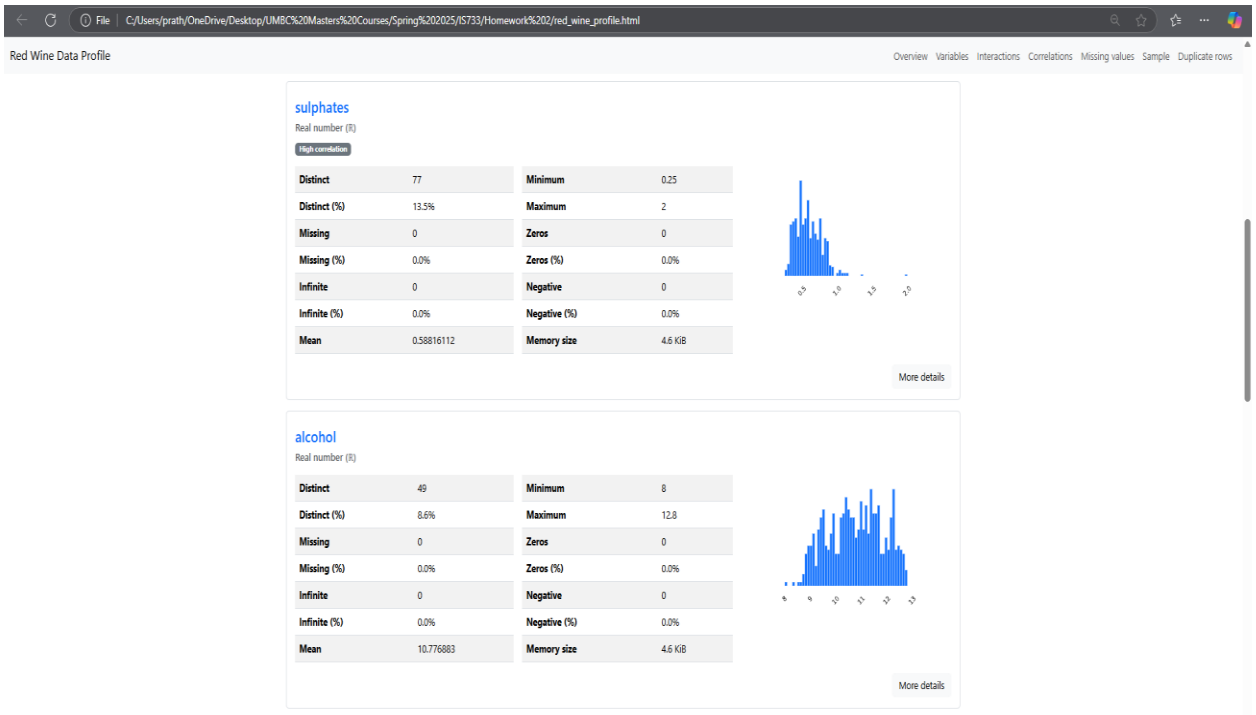
Variable types

Numeric	3
Categorical	1

Variables

citric acid
Real number (ℝ)

Distinct	74	Minimum	0
Distinct (%)	13.0%	Maximum	1
Missing	0	Zeros	25
Missing (%)	0.0%	Zeros (%)	4.4%
Infinite	0	Negative	0
Infinite (%)	0.0%	Negative (%)	0.0%
Mean	0.32467601	Memory size	4.6 KiB


C:/Users/prath/OneDrive/Desktop/UMBC%20Masters%20Courses/Spring%202025/IS733/Homework%202/red_wine_profile.html
Red Wine Data Profile
Overview Variables Interactions Correlations Missing values Sample Duplicate rows

sulphates
Real number (ℝ)
High correlation

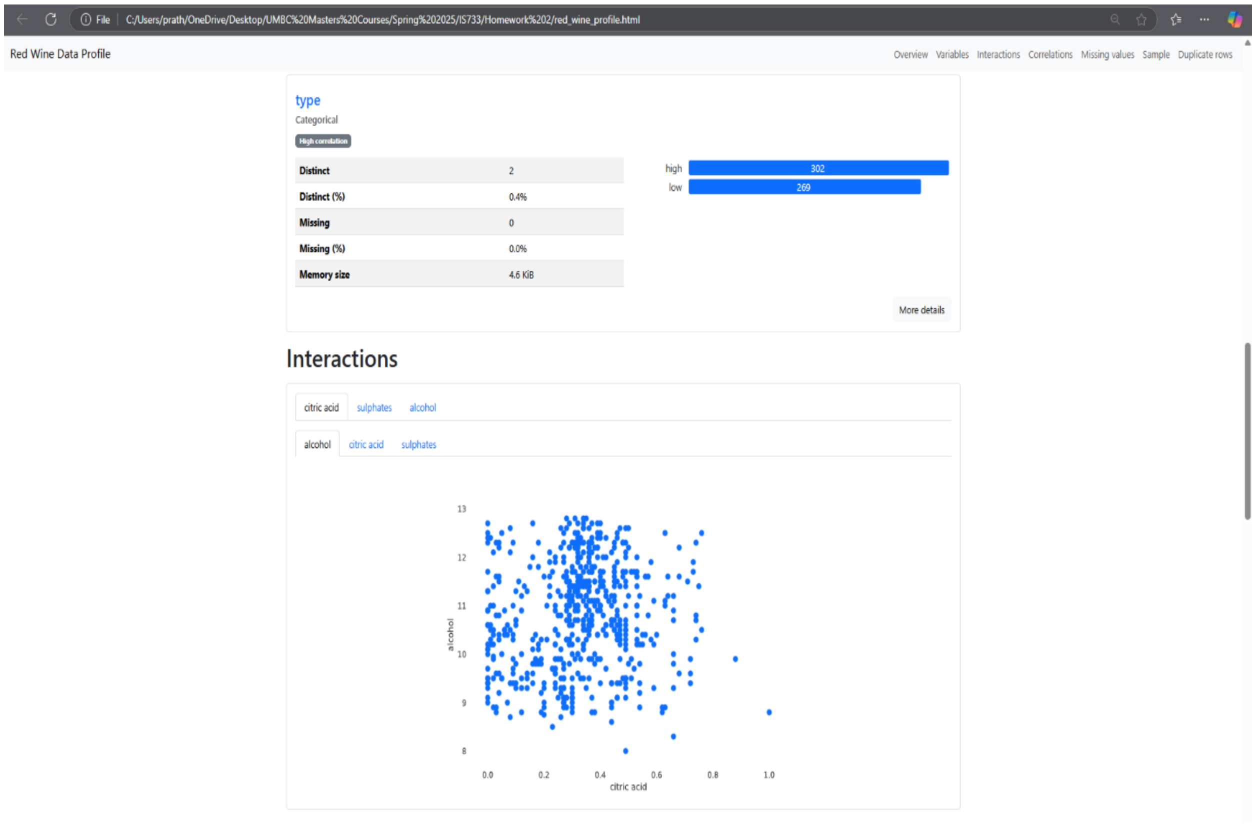
Distinct	77	Minimum	0.25
Distinct (%)	13.5%	Maximum	2
Missing	0	Zeros	0
Missing (%)	0.0%	Zeros (%)	0.0%
Infinite	0	Negative	0
Infinite (%)	0.0%	Negative (%)	0.0%
Mean	0.58816112	Memory size	4.6 KiB

0.5
1.0
1.5
2.0
More details

alcohol
Real number (ℝ)

Distinct	49	Minimum	8
Distinct (%)	8.6%	Maximum	12.8
Missing	0	Zeros	0
Missing (%)	0.0%	Zeros (%)	0.0%
Infinite	0	Negative	0
Infinite (%)	0.0%	Negative (%)	0.0%
Mean	10.776883	Memory size	4.6 KiB

8
9
10
11
12
13
More details


C:/Users/prath/OneDrive/Desktop/UMBC%20Masters%20Courses/Spring%202025/IS733/Homework%202/red_wine_profile.html
Red Wine Data Profile
Overview Variables Interactions Correlations Missing values Sample Duplicate rows

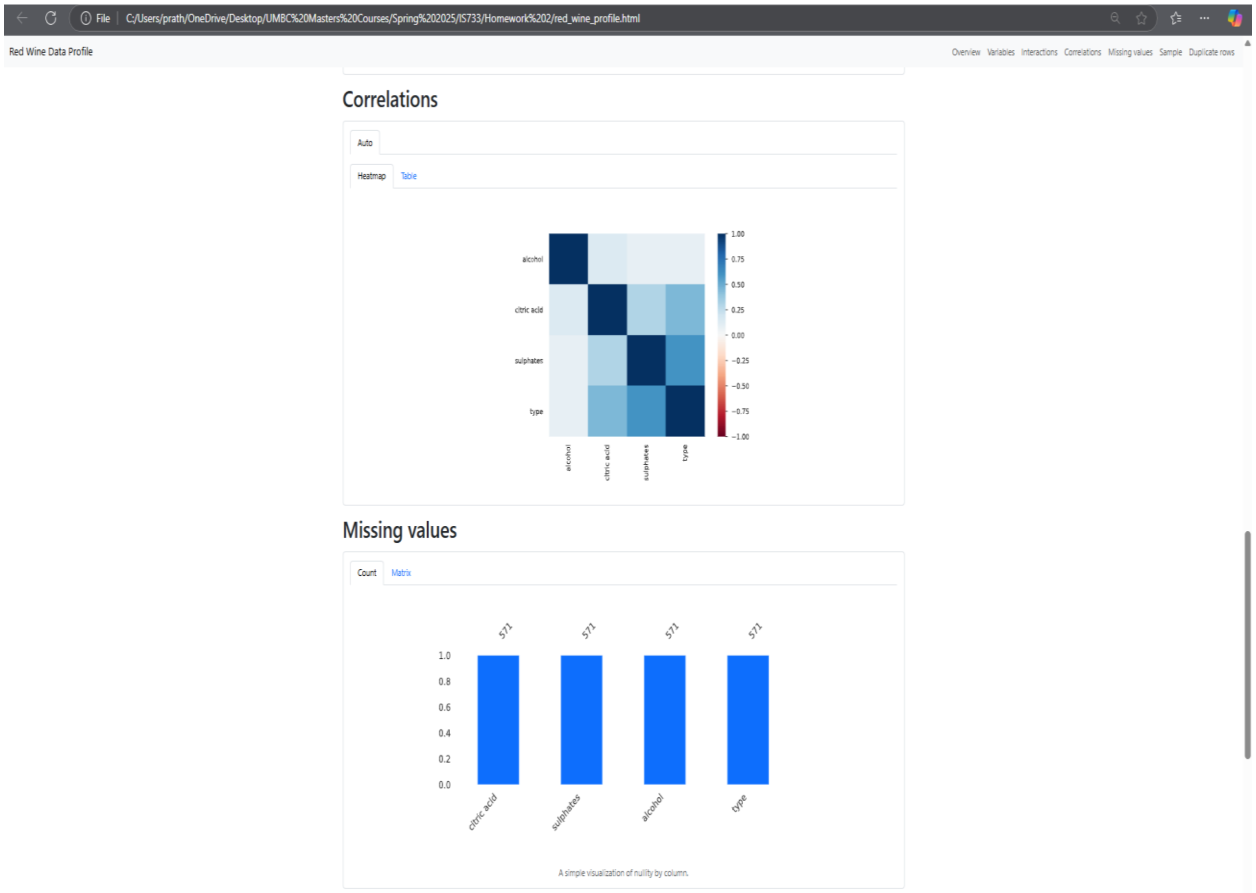
type
Categorical
High correlation

Distinct	2
Distinct (%)	0.4%
Missing	0
Missing (%)	0.0%
Memory size	4.6 KiB

high 302
low 269
More details

Interactions

citric acid
sulphates
alcohol
alcohol
citric acid
sulphates
alcohol
13
12
11
10
9
8
0.0
0.2
0.4
0.6
0.8
1.0
citric acid



C:/Users/prath/OneDrive/Desktop/UMBC%20Masters%20Courses/Spring%202025/IS733/Homework%202/red_wine_profile.html

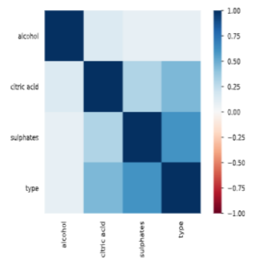
Red Wine Data Profile

Overview Variables Interactions Correlations Missing values Sample Duplicate rows

Correlations

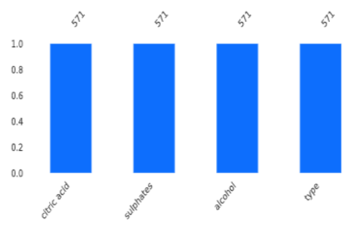
Auto

Heatmap Table

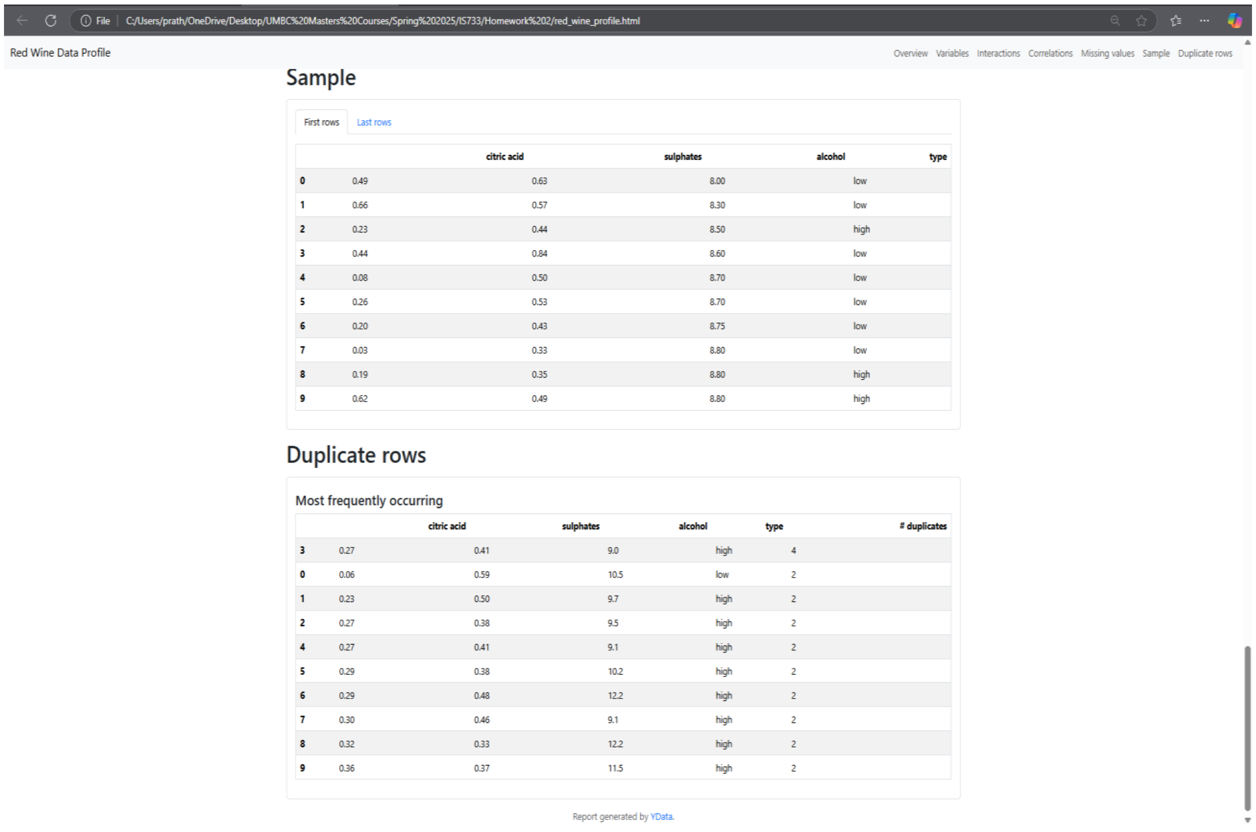


Missing values

Count Matrix



A simple visualization of nullity by column.



C:/Users/prath/OneDrive/Desktop/UMBC%20Masters%20Courses/Spring%202025/IS733/Homework%202/red_wine_profile.html

Red Wine Data Profile

Overview Variables Interactions Correlations Missing values Sample Duplicate rows

Sample

First rows Last rows

	citric acid	sulphates	alcohol	type
0	0.49	0.63	8.00	low
1	0.66	0.57	8.30	low
2	0.23	0.44	8.50	high
3	0.44	0.84	8.60	low
4	0.08	0.50	8.70	low
5	0.26	0.53	8.70	low
6	0.20	0.43	8.75	low
7	0.03	0.33	8.80	low
8	0.19	0.35	8.80	high
9	0.62	0.49	8.80	high

Duplicate rows

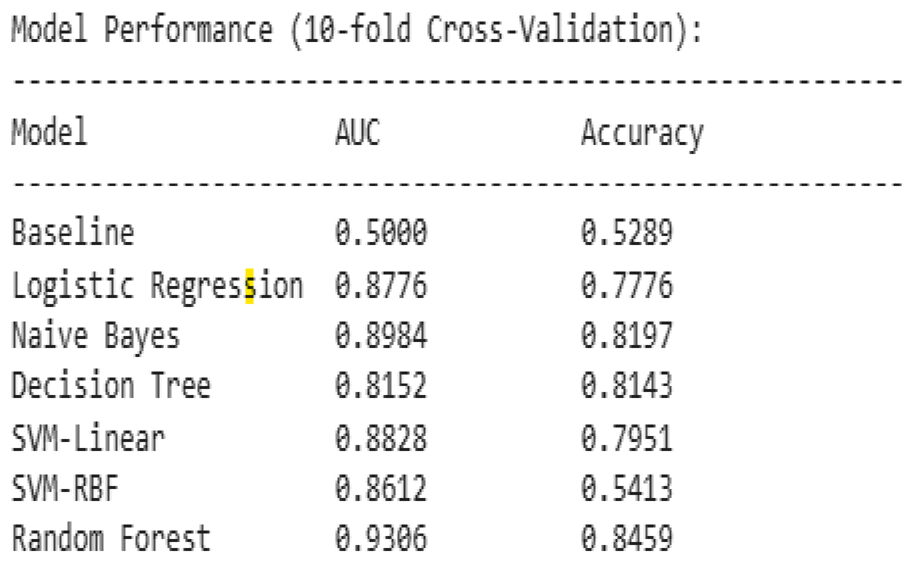
Most frequently occurring

	citric acid	sulphates	alcohol	type	# duplicates
3	0.27	0.41	9.0	high	4
0	0.06	0.59	10.5	low	2
1	0.23	0.50	9.7	high	2
2	0.27	0.38	9.5	high	2
4	0.27	0.41	9.1	high	2
5	0.29	0.38	10.2	high	2
6	0.29	0.48	12.2	high	2
7	0.30	0.46	9.1	high	2
8	0.32	0.33	12.2	high	2
9	0.36	0.37	11.5	high	2

Report generated by YData.

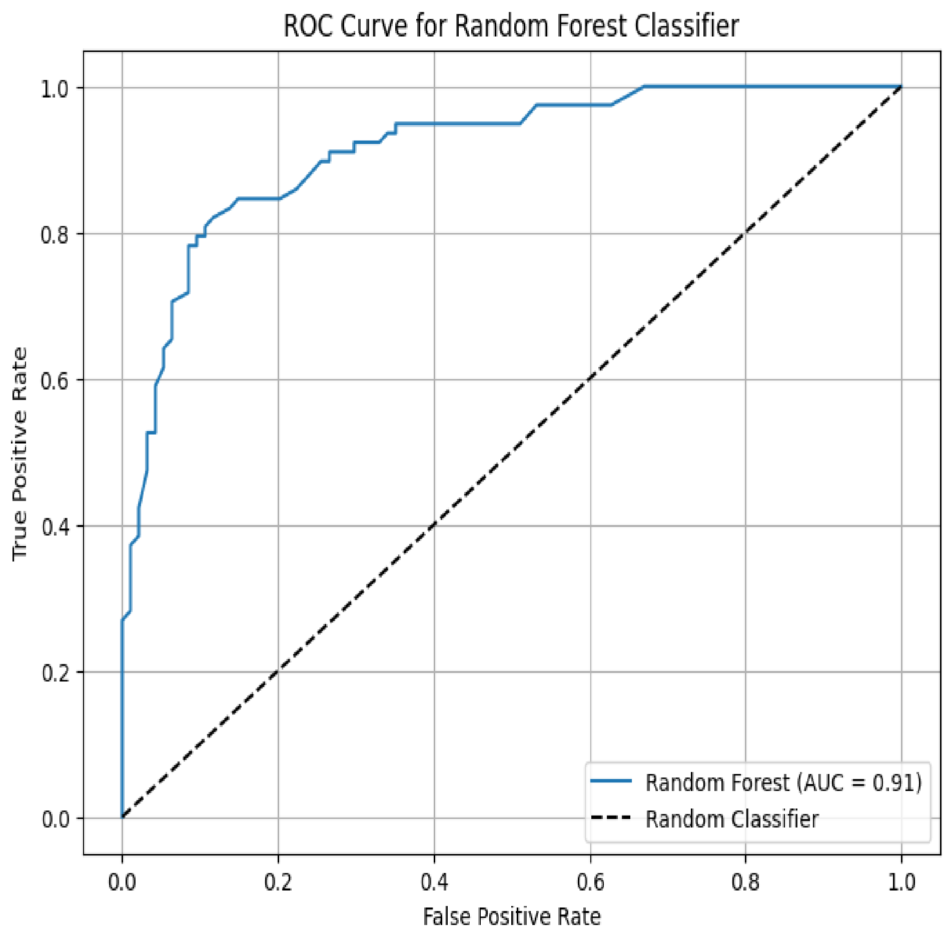
2. *Fit a model using each of the following methods and report the performance metrics of 10-fold cross-validation using* ***red-wine.csv*** *as the training set (30 points).*
   Note:
   - *You are not required to tune the parameter for this homework assignment.*
   - *You can use the default parameter for each model.*
   - *Baseline model accuracy is the accuracy when predicting the majority class; Baseline model AUC is the random classifier AUC*

Model	**Baseline**	**Logistic Regression**	**Naive Bayes**	**Decision Tree**	**SVM-Linear**	**SVM-RBF**	**Random Forest**
AUC	**0.50**	**0.8776**	**0.8984**	**0.8152**	**0.8828**	**0.8612**	**0.9306**
Accuracy	**0.52**	**0.7776**	**0.8197**	**0.8143**	**0.7951**	**0.5413**	**0.8459**



```
Model Performance (10-fold Cross-Validation):
------------------------------------------------------------
Model                AUC            Accuracy
------------------------------------------------------------
Baseline             0.5000         0.5289
Logistic Regression  0.8776         0.7776
Naive Bayes          0.8984         0.8197
Decision Tree        0.8152         0.8143
SVM-Linear           0.8828         0.7951
SVM-RBF              0.8612         0.5413
Random Forest        0.9306         0.8459
```

3. *Plot the ROC curve of the Random Forest classifier from the Python package, and paste a screenshot of your ROC curve here (10 points)*



4. *Using the best model obtained above in Q2 (according to AUC), running the model on* ***white-wine.csv,*** *and reporting the AUC score, comment on the performance. (5 points)*

```
AUC score on white wine dataset: 0.9740
```

The Naive Bayes AUC score on the white wine dataset is 0.9740.
This indicates excellent performance, suggesting that the model trained on the red wine dataset generalizes remarkably well to the white wine dataset. An AUC score this high implies that the model is highly capable of distinguishing between high and low-quality wines, even when applied to a different type of wine.

Comparatively, the performance on the red wine dataset, where the AUC was relatively lower, this result indicates that Naive Bayes is better suited for the white wine dataset. Naive Bayes seemed to "understand" the white wine data better than the red wine data.

5. *Suppose all the models have comparable performance. Which model would you prefer if the wine-tasting experts would like to gain some insights into the model? Note: there could be multiple model types fitting this criterion. (5 points)*

If all models perform similarly on the wine dataset, I would choose a model that prioritizes interpretability. A transparent model allows experts to better understand the results and gain valuable insights. According to me, **Logistic Regression** and **Decision Tree** are the best choices, as they are easier to interpret.

Comparison Of Model Performance & Interpretability:

1. Logistic Regression (My Preferred Choice)

- Achieves a strong **AUC of 0.8776** and **Accuracy of 0.7776**
- Offers clear insights into how each feature influences wine quality through its coefficients.
- Its simplicity makes it easy to understand and apply, helping wine experts identify key factors affecting quality.

2. Decision Tree (Another Good Option)

- Has a slightly lower **AUC (0.8152)** and **Accuracy (0.8143)** compared to Logistic Regression.
- Provides a rule-based, step-by-step decision-making process that can be visualized.
- Ideal for experts who prefer a structured, intuitive approach to interpreting model decisions.

3. **Random Forest (Less Preferred for Interpretability)**

- Achieves the **highest AUC (0.9306) and Accuracy (0.8459)**
- Can highlight feature importance but, as an ensemble method, lacks the direct interpretability of a single decision tree.

FINAL RECOMMENDATION

According to me, the most suitable models for wine-tasting experts are:

1. **Logistic Regression** – My top choice as it provides both high performance and interpretability, helping experts understand the impact of each feature.

2. **Decision Tree** – A strong alternative for those who prefer a clear, rule-based decision-making process.
3. **Random Forest** – A highly accurate model, but its complexity makes it harder to interpret.

Ultimately, **I would choose Logistic Regression**, as it strikes the best balance between accuracy and interpretability, making it the most practical option for wine experts.