Homework 2

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?
Answer:-

In my previous homework I have completed the task of doing an explorative data analysis to a dataset, The dataset I chose was crimes in Baltimore. I analyzed the data and visualized certain trends which depicts the crime rate in certain parts of Baltimore and how it progressed over the years. I have learned a lot during my time with the assignment, I learned how to tell a story using my data and how to visualize interactive plots of my data. I got a 9.5 on my homework 1. The feedback I received was not regarding the technical part, rather how I submitted my assignment. I have submitted my assignment in a pdf format rather than a google slide which caused the decrease in my marks. I have learned to follow those minute instructions also. Hopefully I won't lose my marks in that area from this time onwards.

Part 2. Create a model card (30 points)

you learn additional new models. You may use the given template for your model card. 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

ANSWER :-

Model Card

Property	**Decision**	**Naive Bayes**	**K-Nearest**	**Logistic**	**Support**

	Tree		**Neighbour**	**Regression**	**Vector Machine (SVM)**
1. Parametric/Non-parametric	Non-parametric (it does not assumes any fixed form)	Parametric (assumes probability distribution for feature)	Non-parametric	Parametric(it can learn a fixed number of parameters)	It can be both, When the svm uses a kernel function like radial basis it is non-parametric, When it is a linear svm the svm can be parametric.
2. Input type	Both continuous and discrete	Both continuous and discrete	Both continuous and discrete	Both continuous and discrete(uses categorical data through one-hot encoding	Both continuous and discrete(requires feature scaling for numerical data)
3. Output type	Both continuous and discrete(used for classification and regression)	Discrete(Primarily used for classification problems)	Discrete	Discrete(used for binary and multi-class classification)	Both discrete and continous (svm for classification and svr for regression).
4. Handle Missing Value	Yes(can handle missing data by figuring out how to split the data)	No(requires pre-processing)	No(requires pre-processing)	No(requires pre-processing)	No(requires pre-processing)
5. Model Representation	Tree structure(A hierarchy of decision based on feature splits)	Probabilistic model (bayes theorem)	Instance-based(distance metric)(Stores datapoints and classifies based on nearest	Linear equation(sigmoid function)	Uses a hyperplane to separate classes in the feature space.

			neighbors)		
6. Model Parameters	Tree maximum depth, split criteria(it decides the minimum sample requires to split)	X	Number of neighbours (K), Distance metric (defines the scope of neighbor influence)	Weights, bias	Kernal type, regularization (c), Margin width(controls flexibility and generalization).
7. Make the Model More Complex	Increasing tree depth, reduce pruning(More nodes allow for finer decision boundaries)	X	Increases K value, Advanced distance metrics	Adding polynomial features, reduce regularization	Use non linear kernals(RBF, polynomial)
8. Make the Model less Complex	Do pruning, restricting tree depth(limits overfitting)	X	Reduce K value(lower k increases sensitivity to individual points)	Reduce features and Increase regularization (Fewer features reduces complexity)	Using linear kernels and reducing feature count.
9. Interpretability/Transparency	Yes(Easy to interpret), Visualization is good.	Yes,(naïve bayes calculates probabilities and probabilities are explainable)	No (predictions depend on stored data and no explicit model representation)	Yes(easy because decision boundaries are pretty straight forward).	Hard to interpret, Complex kernels make it harder to interpret.

Part 3. Wine-Tasting Machine

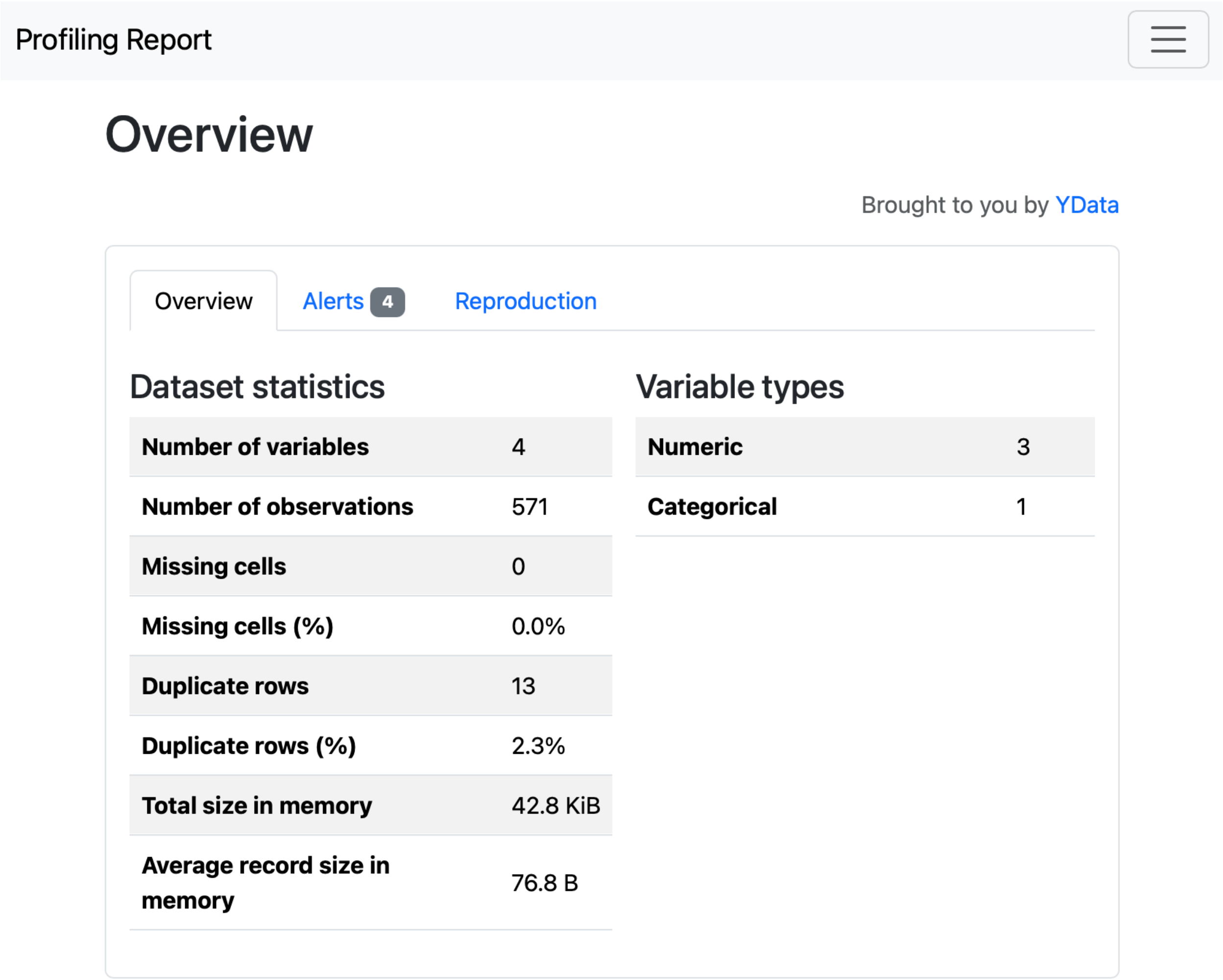
In this task, we will practice building supervised machine learning with Logistic Regression (LR), Naïve Bayes (NB), Support Vector Machine (SVM), and Decision tree (DT), Random Forest (RF) classifiers, as compared with simple/baseline methods. The data for this exercise comes from the wine industry. 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

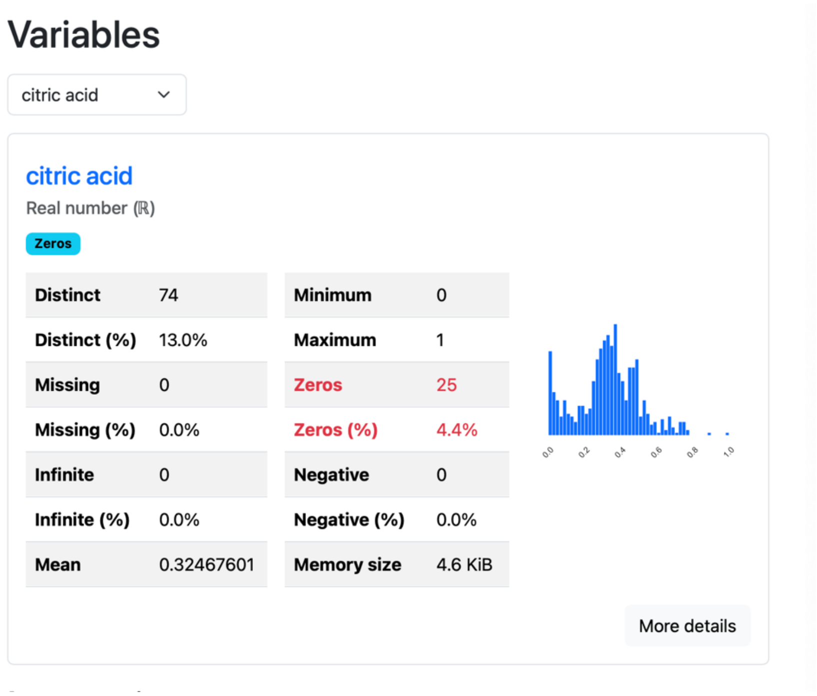
Brought to you by YData

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	42.8 KiB
Average record size in memory	76.8 B

Variable types

Numeric	3
Categorical	1



Variables

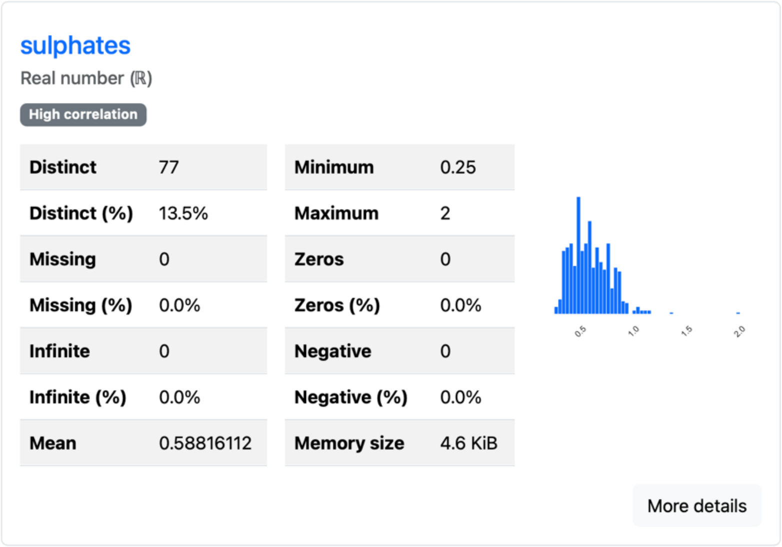
citric acid

citric acid

Real number (ℝ)

Zeros

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

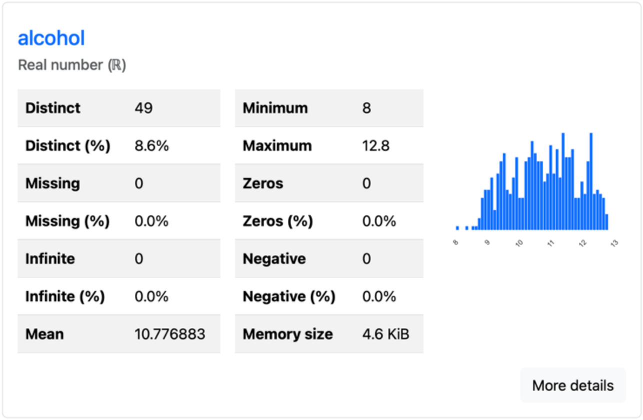


sulphates

Real number (ℝ)

High correlation

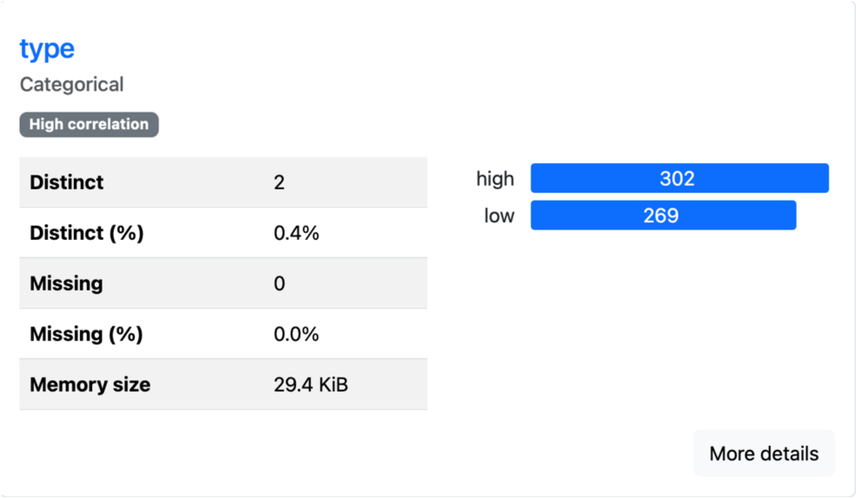
Distinct	77	**Minimum**	0.25
Distinct (%)	13.5%	**Maximum**	2
Missing	0	**Zeros**	0
Missing (%)	0.0%	**Zeros (%)**	0.0%
Infinite	0	**Negative**	0
Infinite (%)	0.0%	**Negative (%)**	0.0%
Mean	0.58816112	**Memory size**	4.6 KiB



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



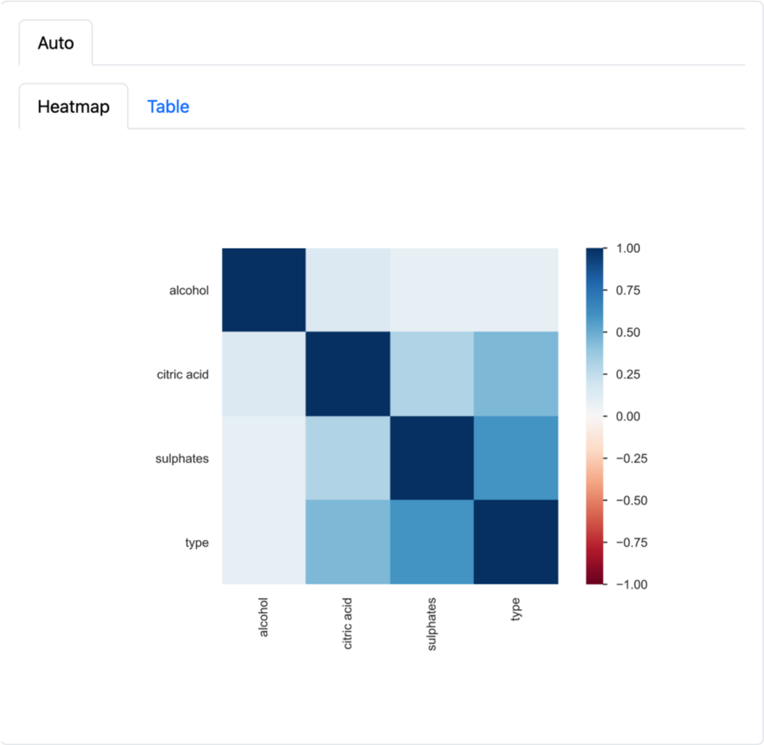
type

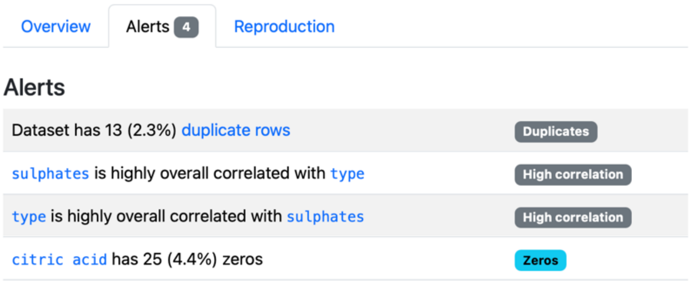
Categorical

High correlation

Distinct	2
Distinct (%)	0.4%
Missing	0
Missing (%)	0.0%
Memory size	29.4 KiB

high	302
low	269





Alerts

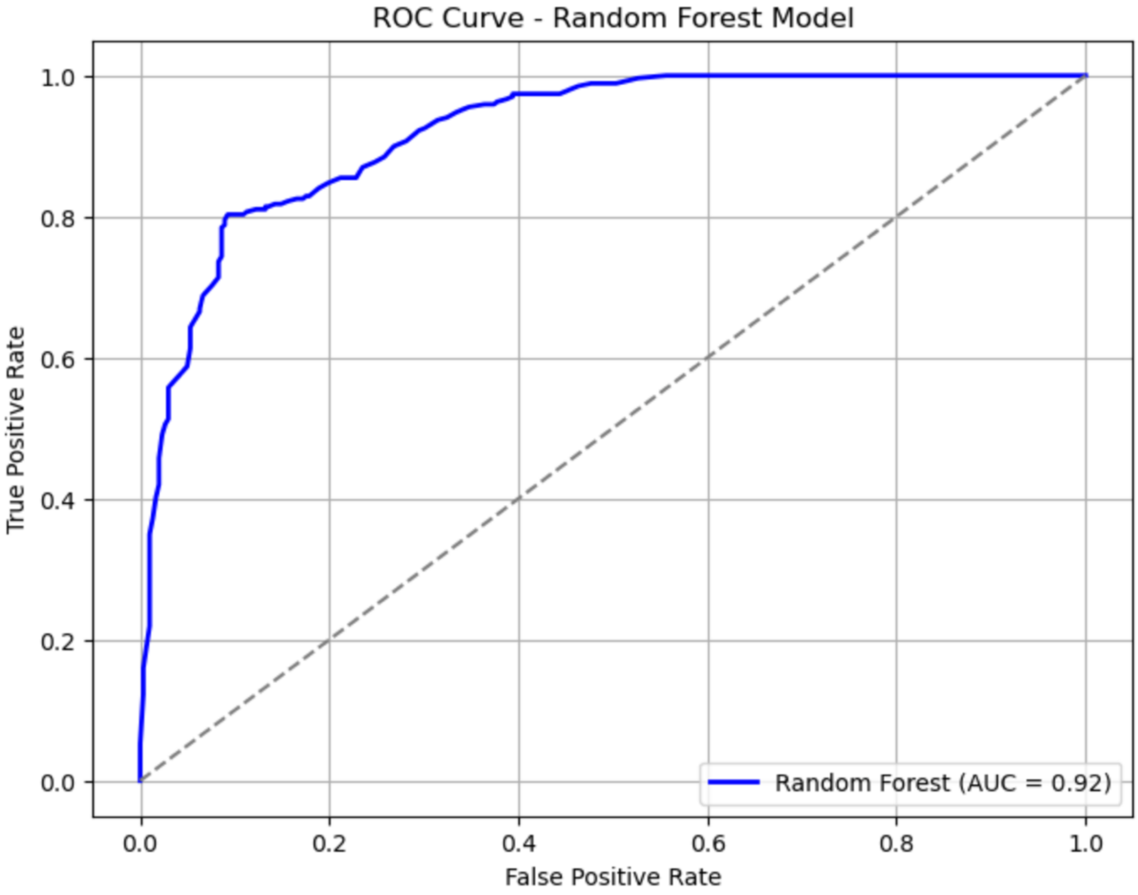
Dataset has 13 (2.3%) duplicate rows	Duplicates
`sulphates` is highly overall correlated with `type`	High correlation
`type` is highly overall correlated with `sulphates`	High correlation
`citric acid` has 25 (4.4%) zeros	Zeros

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	0.87	0.89	0.797	0.87	0.85	0.92
Accuracy	0.52	0.78	0.82	0.798	0.79	0.53	0.84

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 point)

ANSWER:-
The best model obtained in q2 is random forest, I got the accuracy of 0.84 and the AUC (area under the curve) of 0.92. I have saved the weights of the random forest model and tested the pretrained random forest model on the white wine dataset. I got the AUC of 0.97 which is far better than the AUC I got on the red wine dataset.

Based on my observation I can say that my model is performing well on unseen 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)

Answer:- When all the models have comparable performance, and as the wine tasting experts are laymen in the field AI. I would prefer showing them a decision tree model. The graphical representation of the hierarchical tree structure and the splits are more transparent and interpretable. Which can easily show the wine tasting experts how the model is deciding on the type, Which will make them to understand the process better.

Naïve bayes is also a good option when it comes to interpretability, Naïve bayes model decides the type by calculating the probabilities. Which can be shown to the experts to make them easily understand.

One more option is the logistic regression, even though the model name has regression in it. The tasks it performs are mainly classification tasks. Logistic regression uses decision boundaries to clearly separate outcome which makes it easy to see how each input affects the final result.

These three are few models which boosts interpretability and help the experts gain some insights into the models.

GPT STATEMENT:-

I have only used ChatGPT to code the training part of the model, even that I have not just copy pasted rather I learned the working of the model and read the documentation available only, I have changed the code wherever necessary and used ChatGPT when I needed to debug and to format some syntaxes.

GPT policy:

You are allowed to use GPT to complete this assignment, if you did, please make sure you summarize your GPT usage (GPT statement), and share a link to the chat history.

Deliverable:

- An editable link to Google Doc with answers to Parts 1, 2, and 3 (i.e. anyone with the link can edit)
- A link to the Python Notebook uploaded to GitHub
- If you use GPT, the GPT statement and link to the chat history

Reference (Python)

- Run K-fold cross-validation experiment
  - https://www.askpython.com/python/examples/k-fold-cross-validation

- Fitting model and compute AUC/ROC

  https://www.youtube.com/watch?v=uVJXPPrWRJ0

- Baseline model OneR and ZeroR - you may refer to https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyClassifier.html

  Or you may implement your own version

- Model fitting
  - You should be able to find all the following model fitting functions from the sklearn package

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

https://scikit-learn.org/stable/modules/naive_bayes.html

https://scikit-learn.org/stable/modules/svm.html#svm-classification

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://scikit-learn.org/stable/modules/tree.html