# TW5: Naive Bayesian

* Full names of your team members who work on the assignment.
  + Xiaomei Xie
  + Lili Hao
* URL links to the notebook of each student on GitHub repo.
  + <https://github.com/xiaomeiX/NaiveBayes_TW5>
  + <https://github.com/lhaoSeattleu/TW5-NB>
* A summary of what you learned from the teamwork assignment.

**Learning objectives:**

* Be able to understand Bayesian classifier.
* Be able to develop a ML framework and test it.
* Be able to evaluate classification models.
* Be able to understand parameters of classification models.

**Part 1:** Construct a ML framework and develop classification models.

* Dataset: ***iris.csv*** is stored in a folder
* Your modeling analysis should be done on two different datasets:
  + The original dataset
  + Normalized data using min-max normalization.
* Apply Naive Bayes classifiers
* Apply KNN classifiers
* A framework of the k-cross validation (k = 10)
* Display confusion matrix (a matrix with numbers)
* Print a summary of the performance metrics.
* Plot ROC curves

***Submission(s)***

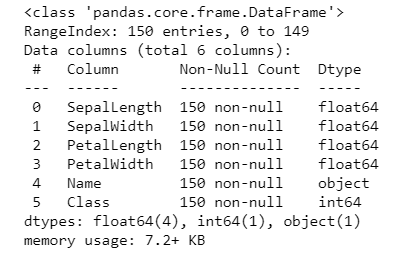
*Each student should make individual submissions.*

* **Part 2**:
  + Submit a summary of your learning to Canvas. Your document should include:
    - Full names of your team members who work on the assignment.
    - URL links to the notebook of each student on GitHub repo.
    - A summary of what you learned from the teamwork assignment.

Your summary should include the comparisons of the two models and the model performance based on parameters (e.g., k value in k-NN classifier).

# Iris Dataset:

This dataset includes 4 columns as floats, 1 column as an int and 1 column as a string:



The dataset has no missing data, and a few outliers.

|  |  |
| --- | --- |
|  |  |

The numeric columns range from 0 to 8.

|  |  |
| --- | --- |
|  |  |

# Naive Bayes classifiers

**KFold(n\_splits=10):** we first applied Naïve Bayes classifiers. We used 10 splits for K fold cross validation: 6 of them have 100% accuracy, 1 of them has 86.7% accuracy, and the remaining 3 of them are all above 90% accuracy.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| k | **Confusion matrix** |  | K | **Confusion matrix** |  |
| 1 |  | Inserting image... | 6 |  |  |
| 2 |  |  | 7 |  | Inserting image... |
| 3 |  | Inserting image... | 8 |  |  |
| 4 |  |  | 9 |  |  |
| 5 |  |  | 10 |  |  |

### **ROC Curve:**

We use binarize three class labels for the Roc Curve. The receiver operating characteristic curve shows the performance of a classification model at all thresholds using True Positive Rate and False Positive rate. We have the accuracy for each class as:

* Class\_0: 1.0
* Class\_1: 0.98148
* Class\_1: 0.98214

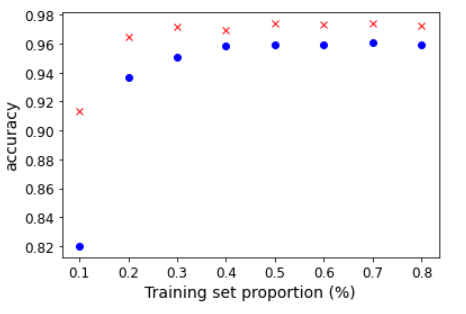
|  |  |
| --- | --- |
|  | Roc Curve |
| Class\_0 | Inserting image... |
| Class\_1 |  |
| Class\_2 |  |

# KNN Classifier

We tried different values for n\_neighbors, such as 1, 5, 15, 20. When n\_neighbors = 5, the model has the best training and testing accuracy.

|  |  |
| --- | --- |
|  |  |
|  |  |

We also tried different training and testing data splits, and found the best performance is between 70% and 80%. So, we chose default 75% as the split percentage.



# Comparison between Naïve Bayes and KNN classifiers

Below is the training and testing accuracy comparison between Naïve Bays and KNN classifier based on both original data (training testing data split at 75%) and normalized data (MinMaxScaler, training testing data split at 75%):

* The KNN classifier has better training accuracy than Naïve Bayes classifier. However, the testing result is worse. In contrast, Naïve Bayes classifiers has better accuracy in testing data.
* For Naïve Bayes classifier, the training and testing data accuracy stays same between original data and normalized data. However, for the KNN classifier the normalized data has better performance on the training dataset.
* The Naïve Bayes classifier has overall better performance even on original data. This is because Naïve Bayes algorithm is a classification technique based on Bayes’ Theorem. It has the assumption that each predictor is independent, and the calculation is based only on the occurrence of a particular feature in a class.
* KNN classifier has performance improvement on normalized data. This is because KNN finds the distances between data points and selects (K) amount of the closest points. Therefore, KNN’s performance requires scaled and centered data to improve the performance.

|  |  |  |
| --- | --- | --- |
| Accuracy | **Naive Bayes classifier** | **KNN (neighbor = 5) classifier** |
| Original Data Accuracy | Train: 0.9378787878787878  Test: 1.0 | Train: 0.9545454545454547  Test: 0.9736842105263158 |
| Normalized Data Accuracy | Train: 0.9378787878787878  Test: 1.0 | Train: 0.9643939393939395 Test: 0.9736842105263158 |