Naïve Bayes Learning and Learning the KNN Classifier

Task 1 Programming and Evaluation on A Small Dataset :

Given a university's football game data for the last two seasons, please construct Naïve Bayes classification models to predict game results on games, and evaluate the model performance.

- Data
  - Each data object (or called instance) is a game. We have three attributes: (1) "Is Home/Away?", a 2-value attribute ("Home", "Away"), (2) "Is Opponent in AP Top 25 at Preseason?", a 2-value attribute ("In", "Out"), (3) "Media", a 5-value attribute ("1-NBC", "2- ESPN", "3-FOX", "4-ABC", "5-CBS"). The label "Win/Lose" is binary ("Win", "Lose").
- Training set
  - 24 games. Please use game ID 1-24 to construct classification models.
- Testing set
  - 12 games. Please use your classification models to predict labels of game ID 25-36 and evaluate the performance of the classification models.
- Predictive labels
  - Suppose "Win" is the positive label and "Lose" is the negative label. Keep it in mind when you use Precision and Recall to evaluate the models.

Q1: Programming (you can implement from scratch, use open-sourced code, or use machine learning platforms)**:** Use Naïve Bayes and KNN to predict labels of instances in the testing set (12 games) based on the training set (24 games). Calculate Accuracy, Precision, Recall, and F1 score on the testing result. This posting discusses the four measurements: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

Dropped Date and ID features for both classifiers
Features for NB were described using integer categories

Naive Bayes Accuracy: 0.750000
Naive Bayes Precision: 0.875000
Naive Bayes Recall: 0.777778
Naive Bayes F1: 0.823529

Features for KNN were described using one hot encoding (since the categorical data has no explicit order). The following was with k = 3.

KNN Accuracy: 0.916667
KNN Precision: 0.900000
KNN Recall: 1.000000
KNN F1: 0.947368

In this example the KNN actually performed better than the NB algorithm, which is not what I expected after reading the paper for the latest discussion. Each of these measures of performance were higher for the KNN. I even tried using one hot encoding on both datasets and still the KNN was higher. Though the NB was a less computationally expensive model with similar precision to that of the KNN.

Q2: Write down the prediction labels of the 12 testing games in the PDF.

Naive Bayes Predictions:
['Win' 'Win' 'Win' 'Lose' 'Win' 'Lose' 'Win' 'Win' 'Win' 'Lose' 'Win' 'Lose']

KNN Predictions:
['Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Win' 'Lose']

Test Labels:
['Win', 'Lose', 'Win', 'Win', 'Win', 'Win', 'Win', 'Win', 'Win', 'Lose', 'Win', 'Lose']

Task 2 Programming and Evaluation on A Large Dataset (Titanic):

Q1: Test your naïve Bayesian classification on the Titanic dataset. Report the average Accuracy, Precision, Recall, and F1 score of your five-fold cross validation. The five-folds of the Titanic data are split randomly. What do you observe and learn by applying Bayesian learning to small datasets and larger datasets?

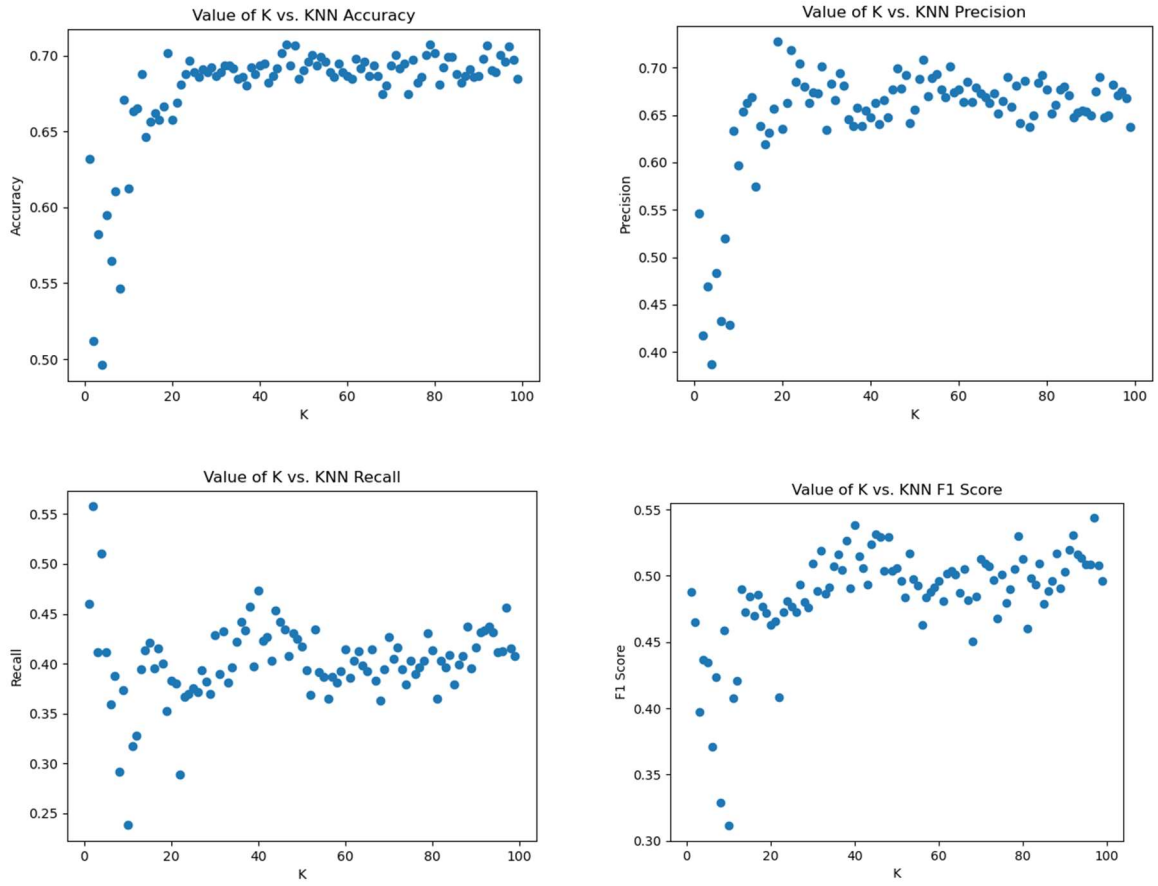
Using banded Age, embarked, pclass, and sex features

Naive Bayes Accuracy: 0.676768
Naive Bayes Precision: 0.599265
Naive Bayes Recall: 0.476608
Naive Bayes F1: 0.530945

Though the accuracy is not that far off, the rest of the performance measures are much worse for the larger dataset. Since I read the latest discussion I know this should be the case, but it makes sense that the Bayesian learning will perform much worse on larger datasets.

Q2: Implement KNN classification from scratch, and evaluate how K impacts the overall accuracy of kNN on the dataset. Plot the accuracies of kNN over k, and identify the best K. You can read sample code and try to implement by yourself. Below are some sample implementations from Github for your fast references:

Features were all converted to one hot encoded data, except for age and pclass, since those should remain ordinal.



Max KNN Accuracy [5-fold, k=79]: 0.707112
Max KNN Precision [5-fold, k=19]: 0.727173
Max KNN Recall [5-fold, k=2]: 0.557610
Max KNN F1 [5-fold, k=97]: 0.543615

The max of each statistic has a different value of K. The average ratio of false positives to false negatives was much different than 1, so we'd probably want to choose the k with the maximum F1 score, so choose K = 55.

I used this link as a reference: https://github.com/sagarmk/Knn-from-scratch

Q3: According to your algorithm analysis, which machine learning model performs better, Naïve Baysian or KNN on the Titanic dataset?

According to my algorithm analysis, the KNN algorithm performed better on the titanic dataset, but only slightly. These were done using 5-fold cross validation so I'm pretty confident in the answers, but I expected KNN to do much better on this dataset since there are much more samples. Even after improving the distance measurement by normalizing the data, the KNN performance was still only slightly better. The only other idea I have is to find an optimal weighting for the features. Either way, the KNN still wins in this case.

GITHUB LINK:

https://github.com/malleyconnor/cap5610/tree/master/hw3