Project 1: Bayes Classifier vs. Winnow_2 Algorithm

Nazanin Yari
Johns Hopkins University

Introduction to Machine Learning - Section 8VL

Abstract

In the project reported here, we present the accuracy comparison of two machine learning algorithms: Naive Bayes classifier and Winnow_2 algorithm. Seven datasets were used for performance assessment and the accuracy results have been calculated by measuring the number of times the algorithms correctly predicted the class to which the observation belongs. The analysis of approximately 6 thousands predictions or 33% of the data have led to very similar performance in accuracy for both classifiers.

Solution Method

Winnow_2 Algorithm. The Winnow_2 algorithm is a technique for learning a linear classifier from labeled examples.[1] It is an example of supervised learning which was invented by Nick Littlestone in the late 1980s.[2] In this algorithm, we train the model using a set of attributes and the related classification to help the machine learn to predict classes associated with the features.

In the winnow2.py file, we first created a discriminant function `h_x(row, weight_list)` which takes a data point of the train dataset and a weight list as its arguments. It sets the weights equal to 1 if the weight list is empty. It then returns the sum of the dot product of the data point and the weight list. The `prediction(h, theta)` function is then used to predict the class based on the given threshold theta. If the prediction is correct, it makes no adjustments. Otherwise, it updates the weights. If the predicted value was higher than the desired outcome, the algorithm demotes h(x) using the `demote(row, alpha, weight_list)` function. The function divides the weight by learning parameter alpha if the data point equals to 1. If the predicted class was lower than the actual class, the program promotes the weights using the `promote(row, alpha, weight_list)` function which multiplies the old weight to alpha where the data point is 1.

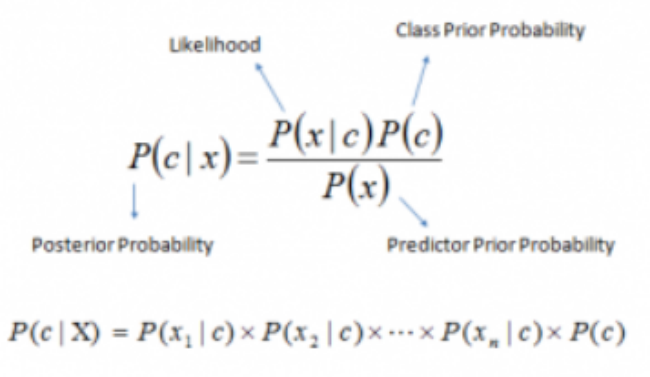
The last function in our file is `winnow_accuracy_test(x_test, y_test, x_train, y_train, alpha, theta)` which trains the model using a train data set to learn boolean functions and to divide the feature space into two d-dimensional areas by calculating the final weights. It also creates a comparison table which is simply a copy of the test dataset as well as a prediction column. The program compares the predicted value by each classifier in each data file to the actual classification number to measure the error of the classification. The function returns both a comparison table and the error rate.

For this assignment, we assumed a threshold of $\theta = 0.5$ and a learning parameter of $\alpha = 2$.

Naive Bayes Classifier. Naive Bayes algorithm is a classification technique based on Bayes' theorem with the assumption that the predictors are independent.[3] This algorithm returns the posterior probability of a class given an observation by the equation below:

---

[1] "Winnow (algorithm) - Wikipedia." https://en.wikipedia.org/wiki/Winnow_(algorithm). Accessed 19 Oct. 2020.
[2] "Winnow2 Algorithm From Scratch | Machine Learning ...." 16 Jun. 2019, https://automaticaddison.com/winnow2-algorithm-from-scratch-machine-learning/. Accessed 19 Oct. 2020.
[3] "Learn Naive Bayes Algorithm | Naive Bayes Classifier Examples." 11 Sep. 2017, https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/. Accessed 19 Oct. 2020.



In this model, the train dataset has been used to calculate the mean, sigma and class prior probability. The function `data_train_test(data_set: np.array)` has been used to create the test and train datasets. As suggested on the project handout, 67% of each data file have been used to train the classifiers, and the rest used to test the accuracy of the algorithm's classification prediction. The function `train_stats(x_train, y_train)` takes both x_train and y_train as its arguments and returns the mean, sigma and class prior probability of the train dataset. The calculated mean and sigma is then used as the arguments of the `class_probability(obs, mean, sigma, prob)` function. This function measures the likelihood or the probability of class given an observation using the test dataset. It returns a dictionary of the class probabilities where class numbers are keys and probabilities are the values of the dictionary. It also returns the highest probability and the associated class number.

The last function `bayes_error(x_test, y_test, x_train, y_train)` creates a comparison table which includes a copy of x_test, y_test and the predicted class. It then calculates the error rate by comparing the actual class column with the predicted class.

In order to validate our findings, the sklearn built-in Bayes function has been used in our program. The `bayes_error(x_test, y_test, x_train, y_train)` returns the sklearn error rate as True-error. The comparison tables as well as the error rates are written in the output_files folder.

What We Know About the Datasets

For this assignment, we used three datasets, each with 6000 observations with 2 features as well as a "Vote" dataset with 435 data points and 16 key votes .

Preprocessing data. Each of the three datasets have two classes. The class names have been updated to 0 and 1 in each file. In order to do an analysis on the Winnow_2 algorithm, three datasets have been created based on the original ones, but with changing the real-valued attributes to boolean numbers. We converted the data to boolean using the mean of the attributes: 0 if the data point was less than feature mean, and 1 otherwise.

The "Vote" dataset has been manipulated as: 1) the votes of 'y' and 'n' has been updated to 1 and 0 respectively. 2) the classes "republican" and "democrat" have been changed to 0 and 1 respectively and the class column has been moved to the last. 3) to handle the missing values, a random integer number between 0 and 1 have been selected.

Data Statistics. The mean and covariance of the datasets are calculated using function `data_stats(dataset)` in the P1_data_to_boolean.py and P1_votedata_cleansing.py files.

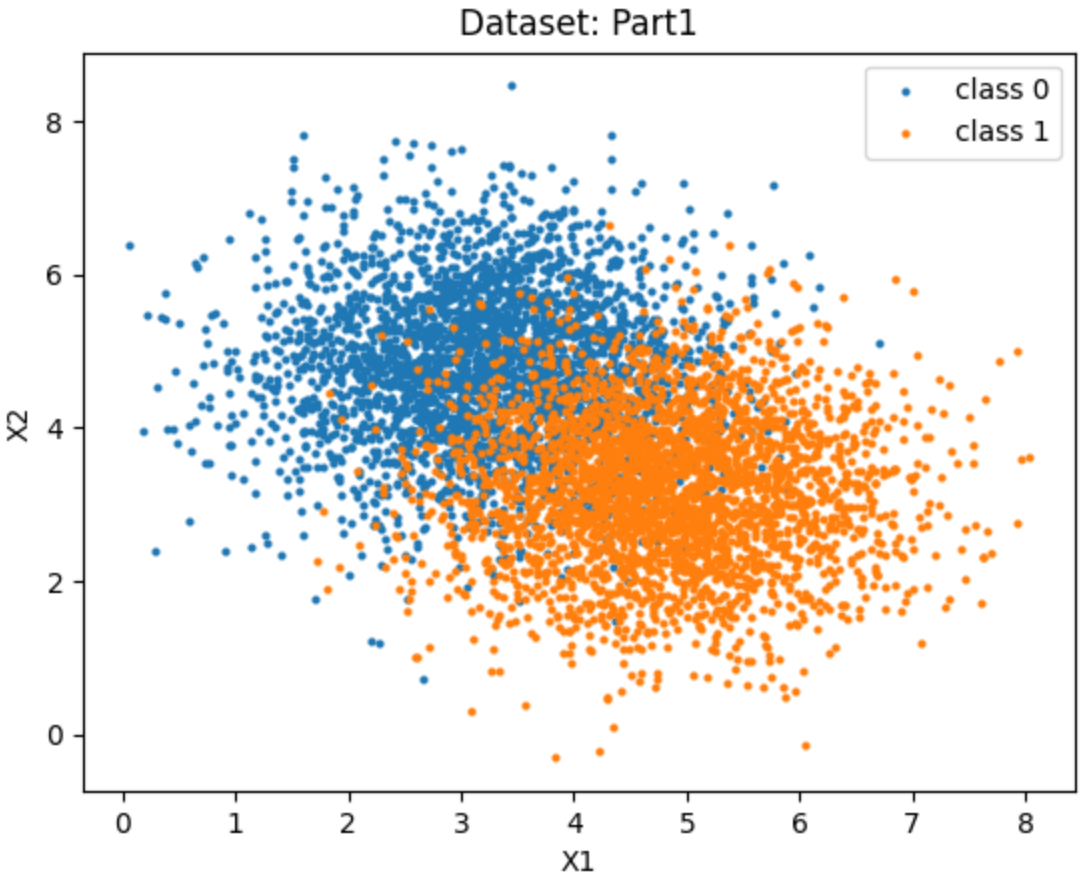
blackboard_part1_original. Below is the mean of the part1 dataset per class where the keys represent

classes and the values are means:

{0.0: [3.21661249, 4.80483994], 1.0: [4.80999843, 3.20070285]}

The covariance of the dataset is shown below. Both classes have variance of almost 1 and correlation of almost 0:

{0.0: [[0.960772, 0.00385086], [0.00385086, 1.02628706]], 1.0: [[0.98999247,0.00756763], [0.00756763, 0.96834425]]}



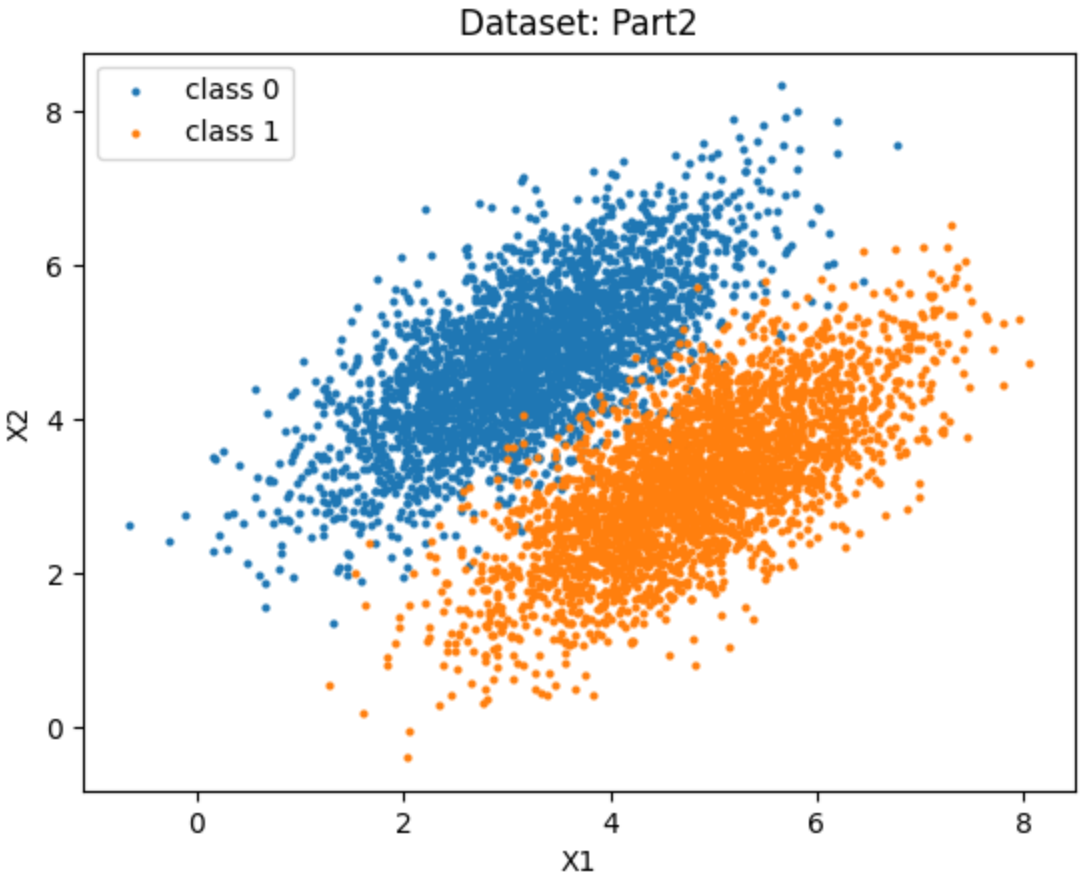
blackboard_part2_original. The mean of the part2 dataset per class where the keys represent classes and the values are means is:

{0.0: [3.19344699, 4.79341019], 1.0: [4.79962757, 3.20616983]}

The covariance of the dataset is shown below. Both classes have variance of almost 1 but correlation of around 0.7:

{0.0: [[0.9803292 , 0.67445753], [0.67445753, 0.96127209]], 1.0: [[1.02888678, 0.70811563], [0.70811563, 0.98803643]]}

The correlation can be observed in the below graph.

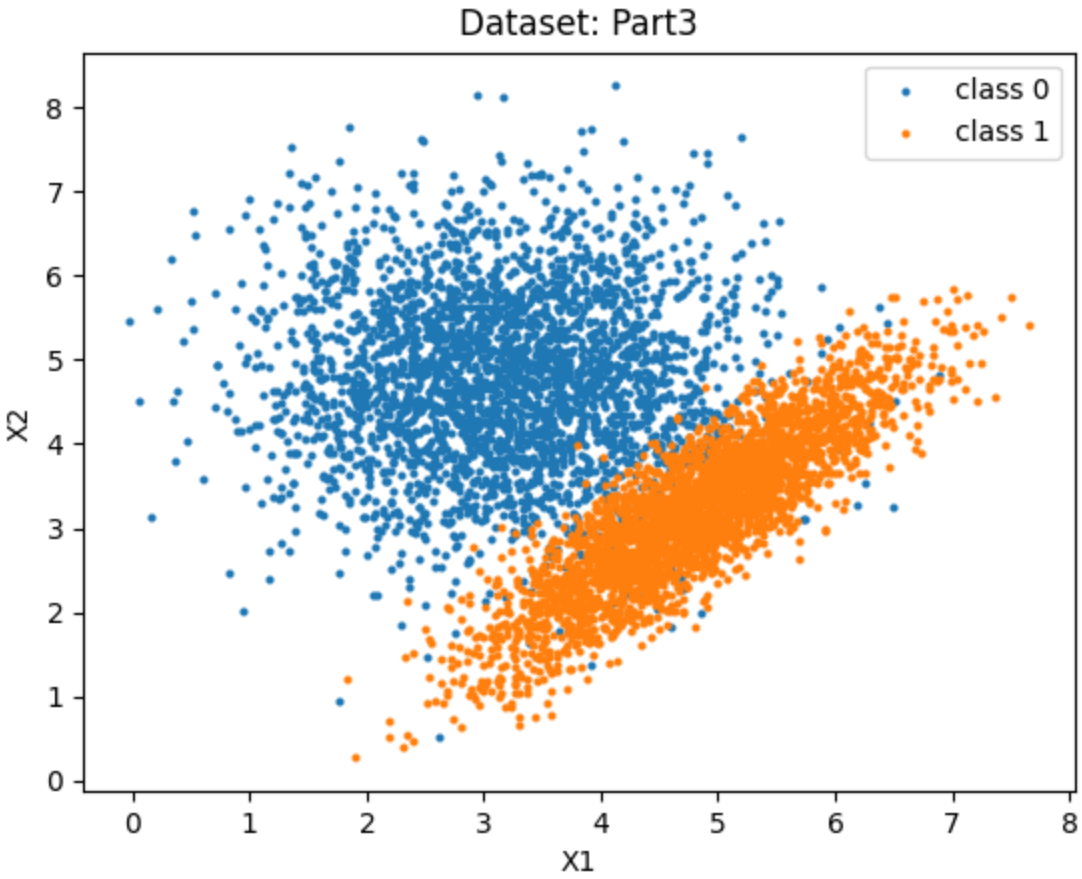


blackboard_part3_original. The mean of the part3 dataset per class where the keys represent classes and the values are means is:

{0.0: [3.18471218, 4.7714468], 1.0: [4.80251837, 3.19922164]}

The covariance of the dataset is shown below. The first class has a variance of almost 1 and correlation of 0 as shown with the blue dots in the graph below. However, the correlation between the points in the second class is around 0.7:

{0.0: [[1.01458111, -0.01512651], [-0.01512651, 1.03761314]]), 1.0: [[0.76737592, 0.68256884], [0.68256884, 0.80368669]]}



blackboard_part1_boolean. The mean of the part1 boolean dataset per class where the keys represent classes and the values are means is:

{0:[0.207, 0.78933333], 1: [0.79166667, 0.20466667]}

The covariance of the dataset is as below:

{0: [[0.16420574, 0.00094165], [0.00094165, 0.16634167]], 1: [[0.16498555, -0.00369568], [-0.00369568, 0.1628325]]}

blackboard_part2_boolean. The mean of the part2 boolean dataset per class where the keys represent classes and the values are means is:

{0: [0.20333333, 0.8], 1: [0.78866667, 0.211]}

The covariance of the dataset is:

{0: [[0.1620429 , 0.03867956], [0.03867956, 0.16005335]], 1: [[0.16672713, 0.04227209], [0.04227209, 0.16653451]]}

blackboard_part3_boolean. The mean of the part3 boolean dataset per class where the keys represent classes and the values are means is:

{0: [0.21666667, 0.781], 1: [0.82566667, 0.19233333]}

The covariance is:

{0: [[0.16977882, 0.00045015], [0.00045015, 0.17109603]], 1: [[0.14398922, 0.03320785], [0.03320785, 0.15539302]]}

Vote_data. The mean of the vote dataset per class where the keys represent classes and the values are means is:

{0: [0.18452381, 0.44642857, 0.13095238, 0.9702381 , 0.93452381, 0.88690476, 0.23214286, 0.14285714, 0.11309524, 0.54761905, 0.125, 0.80357143, 0.80952381, 0.94047619, 0.08333333, 0.57142857], 1: [0.58426966, 0.4494382 , 0.86516854, 0.05243446, 0.20599251, 0.46067416, 0.74906367, 0.8164794 , 0.70411985, 0.46441948, 0.48314607, 0.13483146, 0.27340824, 0.33707865, 0.59925094, 0.64794007]}

The standard deviation or the square root of the diagonal of the covariance matrix of the vote dataset is:

{0: [0.38791078, 0.49712182, 0.33734827, 0.16992979, 0.24736422, 0.31670918, 0.42219966, 0.34992711, 0.31670918, 0.49772726, 0.33071891, 0.39729635, 0.39267673, 0.23660246, 0.2763854 , 0.49487166], 1: [0.49284747, 0.49743694, 0.34154346, 0.22290151, 0.40442502, 0.49845108, 0.43355194, 0.38709274, 0.45643739, 0.49873242, 0.49971586, 0.34154346, 0.44570862, 0.472712 , 0.49005025, 0.47761254]}

Results

Seven datasets have been used for performance assessment and the accuracy results have been calculated by measuring the number of times the algorithms correctly predicted the class to which the observation belongs. The results of running both programs on all seven datasets are listed in the table below.

#	**Dataset Used**	**Bayes Classifier Error**	**Winnow_2 Error**
1	blackboard_part1_original	0.1282	NA
2	blackboard_part2_original	0.0196	NA
3	blackboard_part3_original	0.0707	NA
4	blackboard_part1_boolean	0.2187	0.2277
5	blackboard_part2_boolean	0.2101	0.2080
6	blackboard_part3_boolean	0.2095	0.1884
7	vote_data	0.0625	0.1527

As shown in the graph below, the accuracy of the Bayes' classifier for the original 3 datasets are significant. The numbers are shown in the first 3 blue bars where the Winnow_2 algorithm could not be

applied for comparison purposes. The performance of the algorithms using boolean versions of the three datasets as well as the "Vote" dataset are shown on the right section of the graph. For the non-manipulated "Vote" dataset, Bayes' classifier did a better class prediction; 6.25% vs. 15.27% by the Winnow_2 algorithm. For the other three boolean datasets, Bayes's classifier was slightly better in classifying the datasets.