Project 2: Supervised Learning

Building a student intervention system

Classification Vs Regression

This is a classification supervised learning problem. Because we are trying to predict whether students are likely to pass or not. So this is a binary classification problem.

Exploring the Data

Statistics of the given dataset is as follows

- Number of data points: 395
- Number of features: 30
- Number of graduates: 265
- Number of non-graduates: 130
- Graduation rate: 67.09%

Training and Evaluating Models

The 3 supervised learning models I choose are

1. Logistic Regression
2. Support Vector Machine
3. Gaussian Naive Bayes

Logistic regression converts the dependent variable from a classification type to a probability and then uses linear regression to predict the probability of the dependent variable. Eventually a threshold is used to convert the probability of dependent variable to a classification variable by using the logic if the probability is above threshold then true otherwise false. Logistic regression can be used to solve binary classification problem. Its strength is that it can give probability of the dependent variable, so we can have a measure of likelihood of the dependent variable. Its weakness is that given a threshold, it will not be able to determine class of some dependent variable that are equal to the threshold. Another major limitation of logistic regression is that as the number of features increase, larger sample sizes are to make prediction. Since in the current data we are trying to predict pass or no pass for a student, which is a binary classification, logistic regression fits the need. Besides the number of features is also not that big, so logistic regression can perform prediction with the given sample.

Support Vector Machine models the features as points in space and tries to form boundary lines between different classes of variables, thus separating each class of features into different planes. Its strength is in dividing features that have a clear separation. It can quickly separate different classes of features when they are distinctively apart. Its weakness is that it fails in situation when points are

uniformly distributed. Since the current data can be divided into two classes, support vector machine can separate them into two planes. One issue with SVMs is having to choose a kernel function. Finding the right kernel function is not trivial in many cases. SVMs are known to have good generalization performance, but can be slow in test phase. SVMs are effective in high dimensional spaces. Even when number of features exceed the number of samples SVM can still give good results, but can have poor performance when number of features is much greater than number of samples. Unlike logistic regression, SVM does not give probability and finding probability after classifying labels using SVM requires complex k-fold algorithms, which can have poor performance. SVM is suitable for current scenario, because we only need to determine pass or no pass and we are not required to give probability estimates and the number of features are not that large.

Gaussian Naive Bayes gives weight to each feature depending on how it relates to the label and then uses the weighted sum of features as a model to calculate the dependent variable. Naive bayes can be simple and fast when feature don't have interdependency. But its weakness is when two features depend on each other, it would put stronger weight on those features and hence will be biased. Also naive bayes cannot optimize as well as logistic regression. A subtle issue with Naïve Bayes is that if there are no occurrences of class label and a class label and certain attribute value together, then the frequency based probability estimate will be 0 and this will affect posterior probability estimates. Lagrange correction or Laplace smoothing is typically used to avoid this issue, where a small correction is added to the probability, so it's never exactly 0. Gaussian Naïve Bayes could be suitable for current scenario since the features refer to attributes that don't have obvious dependency on each other.

Below are 3 tables for model training and prediction results

Model: Logistic Regression

Training Size	Training Time(s)	Training Prediction Time(s)	Training F1	Testing Prediction Time(s)	Testing F1
100	0.004	0.001	0.910	0.001	0.708
200	0.004	0.000	0.842	0.000	0.788
300	0.004	0.001	0.843	0.000	0.782

Model: Support Vector Machine

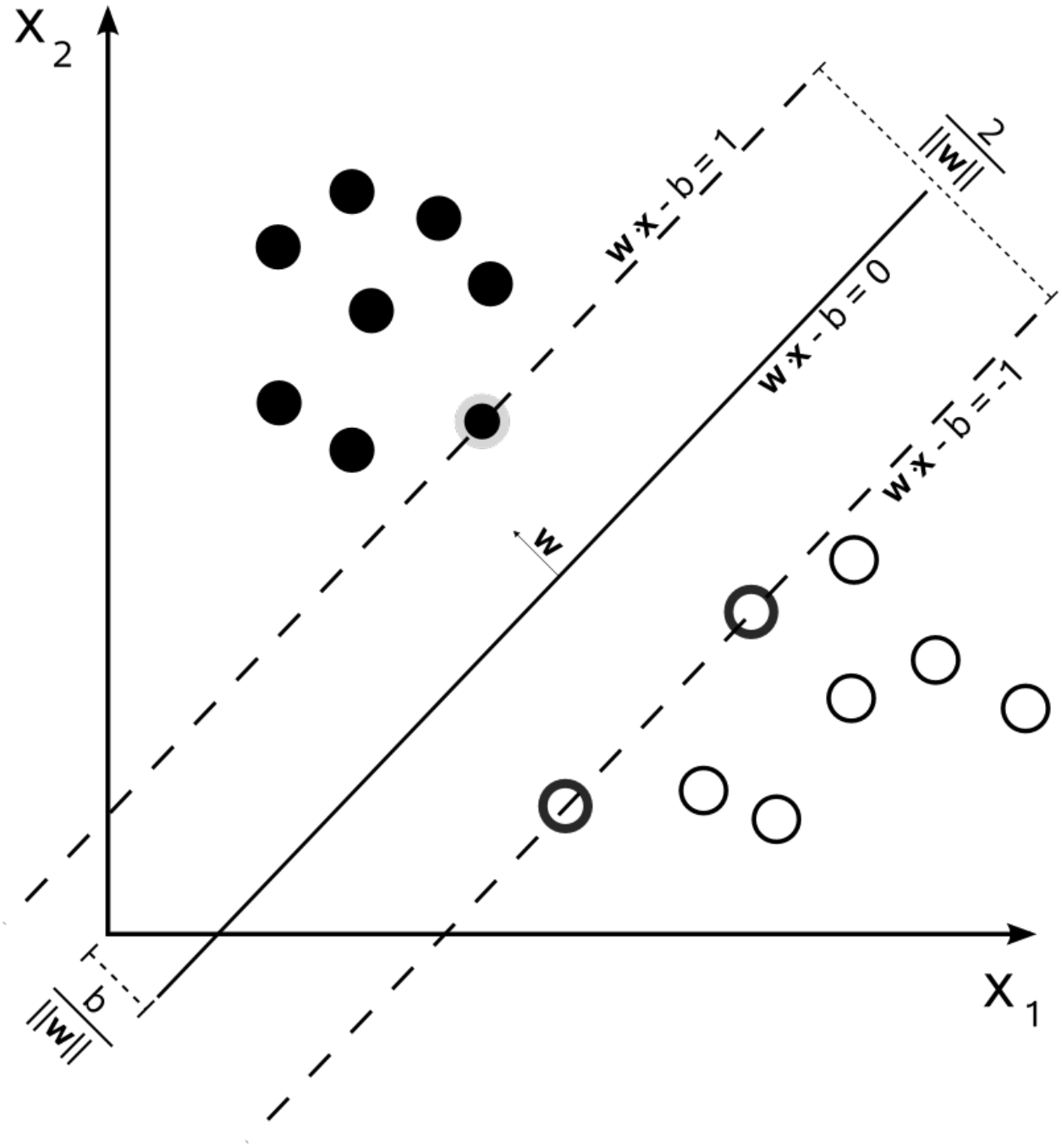
Training Size	Training Time(s)	Training Prediction Time(s)	Training F1	Testing Prediction Time(s)	Testing F1
100	0.002	0.001	0.877	0.001	0.774
200	0.004	0.003	0.868	0.002	0.781
300	0.008	0.006	0.876	0.003	0.784

Model: Gaussian Naive Bayes

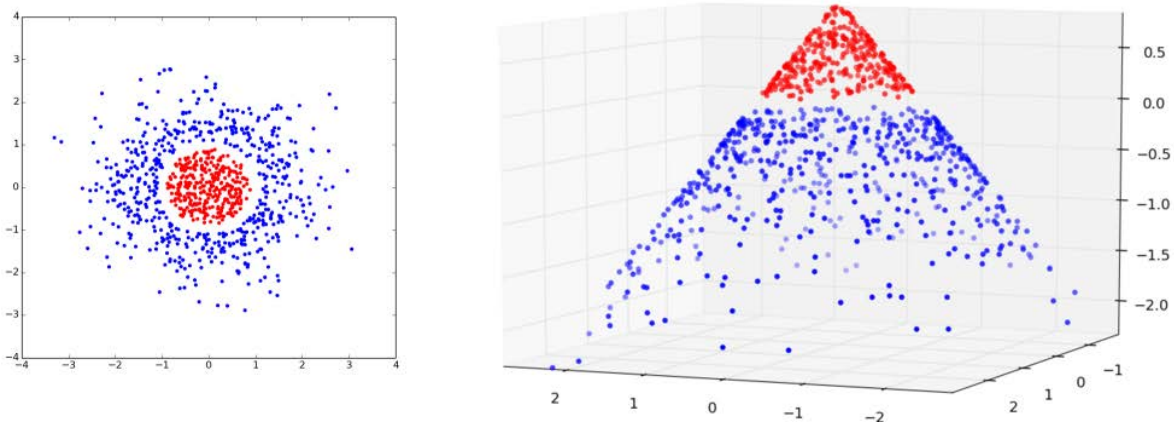
Training Size	Training Time(s)	Training Prediction Time(s)	Training F1	Testing Prediction Time(s)	Testing F1
100	0.002	0.001	0.846	0.000	0.803
200	0.001	0.000	0.840	0.000	0.724
300	0.001	0.001	0.804	0.001	0.763

Choosing the best model

Based on the results I choose support vector machine as the model for prediction. After testing logistic regression and Gaussian naive bayes, it seems support vector machine has the best testing F1 score with maximum training size 300 (0.784 as opposed to 0.782 for logistic regression and 0.763 for Gaussian Naïve Bayes). Besides having high testing F1 score SVM F1 score also seems to be in a closer range with 0.774 for training size 100 and 0.784 for training size 300. On the other hand, testing F1 score for Logistic Regression ranges from 0.708 to 0.782 and for Gaussian Naïve Bayes the testing F1 score ranges from 0.803 to 0.763. This shows that SVM generalizes better than other models. Although training time for support vector machine is a bit higher than other 2 algorithms, it is not significantly higher. Besides the process of grouping customers into different classes is not likely to be done in real time so reducing processing time is not that important.



Support vector machine maps features as points in space and tries to separate different classes of points as far apart as possible. A vanilla SVM is a type of linear separator. It finds the maximum margin line or the line in the middle that separates two different classes of labels. For example, suppose we want to split black and white circles in the above figure. There are infinite number of lines that separate the two types of points. SVM would find the maximum-margin line, which is the line in the middle. This works well because it allows noise on either side. Given a new point or feature, SVM would be able to predict which class it belongs to depending on which side of the line, the point would be mapped to.



Sometimes we need a curve instead of a straight line to separate different classes of points. In the left figure above a circle around the red dots would separate them from blue dots and in the right figure a plane parallel to ground would separate the blue dots from the red ones. This is achieved not by drawing curves or planes but by lifting the features we observe into higher dimension. For example if we can't draw a line in space (x,y) then we may try adding a third dimension (x,y,xy). If we project the line, called a hyperplane in higher dimension down to original dimension, it looks like a curve. Doing this kind of lift is called a kernel trick.

In the current dataset SVM would map the labels pass or no pass using feature attributes as coordinates, and then try to divide them into groups of pass or no pass.

Using grid search the final F1 score of the model with training data is 0.876 and with test data is 0.784