Student Intervention System - Supervised learning models

Marimuthu Ananthavelu

Udacity- Machine Learning Nanodegree Program-Project 2.

The questions and answers are inserted within the code, are reported here.

Abstract

The following are the Questions and answers for the Project-2 Student Intervention System which is a part of Machine Learning Nanodegree curriculum.

Keywords: Nanodegree, Machine learning, Udacity, Project-2, Student System Intervention

Student Intervention System - Supervised learning models

1. Classification vs Regression

Your goal is to identify students who might need early intervention - which type of supervised machine learning problem is this, classification or regression? Why?

The type of problem is 'Classification' as the objective is to find out whether the students will pass or not so to intervene in advance.

2. Exploring the Data:

Looking at the data how it looks the following are noted;

```
Total number of students: 395
Number of students who passed: 265
Number of students who failed: 130
Number of features: 30
Graduation rate of the class: 67.00%
```

3. Preparation of the Data

In this section, the given data is prepared well before training and testing the machine learning algorithms. The following are done on the data;

a. Identifying the feature columns and Target columns: The target column is whether the student pass or not. Here it is represented as ‘yes’ or ‘no’. Remaining columns are the features which all are labeled inputs.

b. Preprocess feature columns: Convert non-numeric values in the dataset into numeric value. It is easy with Machine learning algorithm when the numeric values are handled. Here we convert the ‘non-numeric’ values to either 1 or 0. In case if we have more non-numeric variables on the same column, they were splitted into new columns and combined with original, thus increasing the number of features. Add dummy variables with columns wherever necessary wherever there are more categorical variables in the given column of dataset.

c. Split data into training and test sets: We need to split the data into Training and Test for the following main reasons;

i. To ensure that the 'model' has learned to predict the target values for unknown features over the different possible scenarios at all the times (Gives an estimate of performance on an independent datasets)

ii. To ensure that the 'model' has not only to known to predict for given features but getting tested with different sets for the chosen problem, gives an advantage for another sets of features. (This helps in avoiding 'overfitting' while training the model.).

4. Training and Evaluating Models

The purpose of this section is to choose 3 different supervised learning methods (Classifiers) for a given dataset and provide explanations on the following.

- What are the general applications of this model? What are its strengths and weaknesses?

The 3 supervised models used for solving the problem. They are:

1. Decision Trees

2. Gaussian Naive Bayes

3. Support Vector Machines

1. Decision Trees

The Decision Tree model that predicts the value of a target variable with the help of learning simple decision rules from the data features.

The strengths and weaknesses of Decision trees are;

Strengths(I,II):

i. Decision Trees are very flexible, easy to understand, and easy to debug.One of the coolest things about Decision Trees is they only need a table of data and they will build a classifier directly from that data without needing any up front design work to take place. Able to handle both numerical and categorical data. Other techniques are usually specialised in analysing datasets that have only one type of variable.(I,IV)

ii. Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.(I)

iii. The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.(I)

iv. Able to handle multi-output problems.

v. Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.

vi. Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.

vii. Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

viii. Addresses non-linearity.

Weaknesses(I,III,IV):

i. They over fit. Splitting a lot leads to complex trees and raises probability you are overfitting. Decision-tree learners can create over-complex trees that do not generalise the data well. One require to prune the trees to overcome this issue or setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem. One does not have any upfront design cost due to its simplicity, but one will pay that back on tuning the trees performance. (I,IV)

ii. Instability: Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble. (III)

iii. The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.(I)

iv. There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.(I)

v. Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.(I)

References:

(I) http://scikit-learn.org/stable/modules/tree.html

(II) https://www.quora.com/What-are-the-advantages-of-using-a-decision-tree-for-classification

(III) https://www.quora.com/What-are-the-disadvantages-of-using-a-decision-tree-for-classification

(IV) http://stackoverflow.com/questions/10317885/decision-tree-vs-naive-bayes-classifier

Reasons for considering Decision Trees:

a. Decision trees is able to work with both numerical categorical variables unlike other models and Decision trees take little efforts for Data preparation. Due to the content and the efforts for data preparation, i preferred Decision trees as one of the algorithm.

b. Decision trees is simple to understand and interpret. With respect to the student data, i do visualize how the decision trees will create leafs and nodes with respect to number of features and 2 possible outcomes i.e. ‘yes' or 'no'. Easy to interpret overall.

2. Gaussian Naive Bayes

Strengths (I,III):

i. Simple in comparison, fast to train and fast to classify.

ii. Not sensitive to irrelevant features (see #1 in weaknesses)

iii. Good for a smaller dataset.

iv. Handles streaming data well(III)

v. Bayes is based upon the conditional probability. Their answers are in terms of probabilities i.e. P(yes)=95 %, P(no)= 5%.

Weaknesses (II,III):

i. Assumes independence of Features

ii. High bias classifier due to its split and narrowing. (III)

References:

(I) https://books.google.ca/books?id=3DPcCgAAQBAJ&pg=PT139&lpg=PT139&dq=naive+Bayes#v=onepage&q=naive%20Bayes&f=false

(II) http://www.cs.ucr.edu/~eamonn/CE/Bayesian%20Classification%20withInsect_examples.pdf

(III) https://www.quora.com/What-are-the-advantages-of-different-classification-algorithms

Reasons for considering Naïve Bayes:

a. They are based upon conditional probability. Since in our case, the student performances are expected to be classified, my intuition is that finding the probability whether the students will pass

or not based upon the input features which all are giving information about his nature/activity/age/etc.

For example, as one would say, the students who scored and passed have attended the class in an average of about 70%. So the intuition is, probability is high if the students attend more classes.

b. Our dataset is small with close to 400 samples. So i believe, Naive bayes is believed to yield good prediction.

3. Support Vector Machines

Strengths(I):

i. High accuracy. Effective in high dimensional spaces. Kernel trick is the strength of this algorithm. With an appropriate kernel they can work well even if the data isn't linearly separable in the base feature space. (I)

ii. Still effective in cases where number of dimensions is greater than the number of samples

iii. Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

Weaknesses:

i. Perhaps the biggest limitation of the support vector approach lies in choice of the kernel-selection of the kernel function parameters in high dimensional spaces. (II)

ii. A second limitation is speed and size, both in training and testing. (I)

iii. If the number of features is much greater than the number of samples, the method is likely to give poor performances. (II)

iv. SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. (II)

References:

(I) (https://www.quora.com/What-are-the-advantages-of-different-classification-algorithms

(II) http://www.svms.org/disadvantages.html

Reason for considering Support Vector Machines:

a. Due to its accuracy in prediction of the student performances. Due to its highly predictive ability, this was one of the most important reason to pick up.

b. Due to its advantage of being efficient at high dimensional spaces. So there is no constraint for not having linearly separable data.

- **Given what you know about the data so far, why did you choose this model to apply?**

  - Based on the experiments you performed earlier, in 1-2 paragraphs explain to the board of supervisors what single model you chose as the best model. Which model is generally the most appropriate based on the available data, limited resources, cost, and performance?

  - In 1-2 paragraphs explain to the board of supervisors in layman's terms how the final model chosen is supposed to work (for example if you chose a Decision Tree or Support Vector Machine, how does it make a prediction).

  - Fine-tune the model. Use Gridsearch with at least one important parameter tuned and with at least 3 settings. Use the entire training set for this.

  - What is the model's final F₁ score?

  - Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant.

Of all the chosen model, we should choose the model which performs optimally in processing time and F1 Score. The results are as below;

For Training set 100	Decision Tree	Gaussian Naïve Bayes	Support Vector Machines
Training time	0.002	0.004	0.003
Prediction time on training set	0.001	0.001	0.003
F1 Score for training set	1.000	0.832	0.868
Prediction time on testing set	0.001	0.001	0.003
F1 Score for testing set	0.656	0.797	0.810
For Training set 200	**Decision Tree**	**Gaussian Naïve Bayes**	**Support Vector Machines**
Training time	0.004	0.003	0.009
Prediction time on training set	0.001	0.001	0.007
F1 Score for training set	1.000	0.797	0.842
Prediction time on testing set	0.000	0.001	0.005
F1 Score for testing set	0.626	0.803	0.818
For Training set 300	**Decision Tree**	**Gaussian Naïve Bayes**	**Support Vector Machines**
Training time	0.006	0.003	0.021
Prediction time on training set	0.001	0.003	0.013
F1 Score for training set	1.000	0.796	0.847
Prediction time on testing set	0.000	0.001	0.005
F1 Score for testing set	0.699	0.797	0.810
F1 Score for training set -average	**0.660**	**0.799**	**0.813**

The observations from the above results for the 3 different models are as below;

1.Training time and prediction times are high for SVM in comparison to Decision Trees, Naïve Bayes. They increase considerably with respect to training set. And SVM comes with high average F1 score on test set.

2.Training time is not consistent for Naïve Bayes with respect to increases in Training set.

3. Average F1 Score on testing set with different training set for Decision trees is less than the other chosen classifiers.

4. Gaussian Naïve Bayes is fast in training and classifying. After NB, Decision trees take less time to learn the training set among the chosen algorithms for all the training size.

Since *'Processing time'* and *'Optimum F1 Score'* are the essentials to choose the best algorithm, here i propose Support Vector Machines algorithm for the given dataset where the accuracy is high though it comes at an expense of additional processing time in comparison.

There are 2 most important reasons i gave as a reason for choosing SVM. They are;

1.SVM's are accurate than the other chosen algorithms. Please look at the average F1 Score on testing set for all the chosen classifiers in the report. A difference in accuracy makes prediction for few or more students likely to join the program.

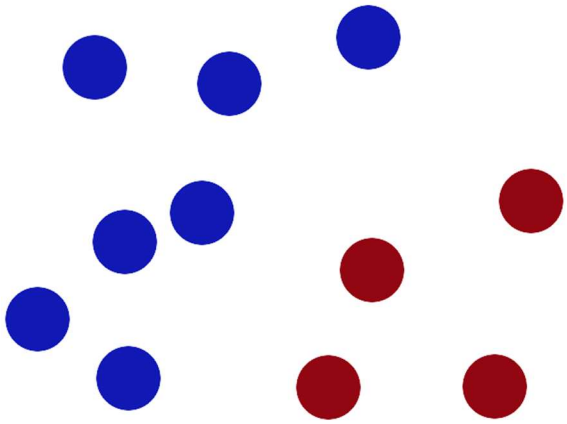
2.Though SVM take more processing time than other 2 classifiers, the following are the convincing reality which we need to make;

a. First, the student interventions system model is required to learn and predict in a given time which is not critical in nature. For example, a High School teacher can expect to wait for an additional hour to know the predictions from the machine learning model. Less computing power increases the wait time. More computing power improves the performance time of machine learning algorithm where one can draw an intuition upon AlphaGo whereas the thinking time is 2 seconds.https://en.wikipedia.org/wiki/AlphaGo

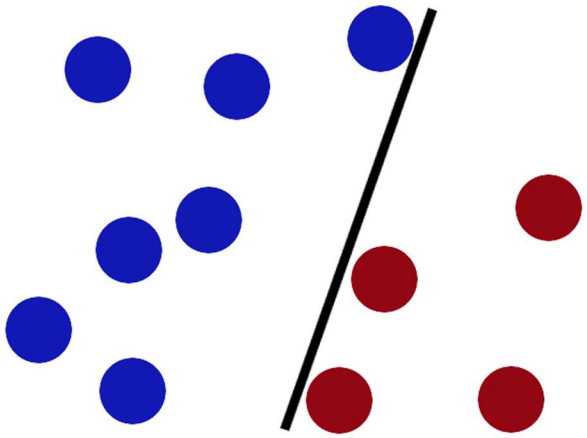
b. We are not dealing with very critical output from our model within a very short moment. For example, in chemical process industries or oil refinery applications, the data should be processed quickly and get processed as quick to avoid any unsafe activity.But in our case, prediction of student performance can take an another several minutes oran hour based upon dataset. Satisfying the more accuracy gives more confidence in SVM.

The chosen algorithm, Support Vector Machines work in the following way: (Help from Reddit)

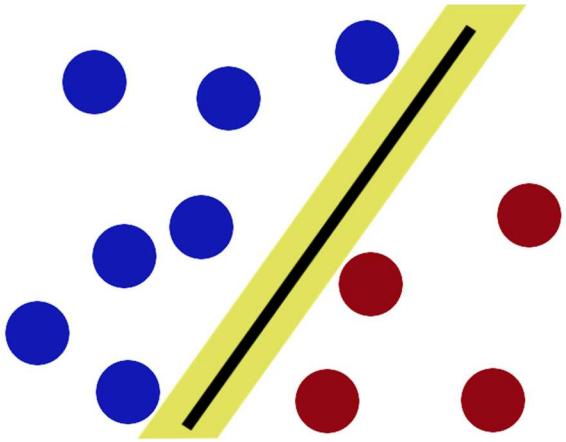
Let’s imagine, we place two colors of balls on the table and we want to segregate(classify) based upon the colors. You are given a scale / ruler to place on the table where an exact separation happens. How our intuition will decide where to place the ruler on the table between two different colors of balls.



Yes, we will look for a place where the maximum likelihood of one set of balls with same color on one side and the another set of balls with another color will go on another side. So, we find a place which is in between these 2 different colors. The following can be one way.

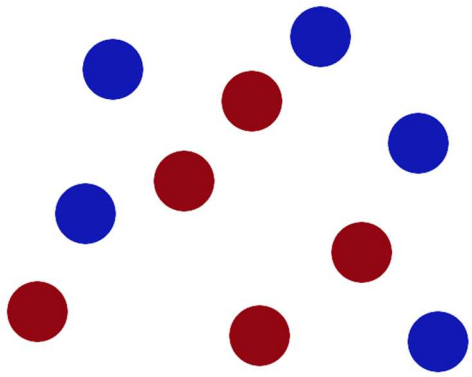


And as we see, that place has space where in multiple ways the ruler can be placed on different angles. to choose the best placement, lets think for a while how this ruler can best be placed in that space. Yes, we would prefer the ruler to be placed closest to the middle in the gap/space between two different colors as below which is best place to make a line.



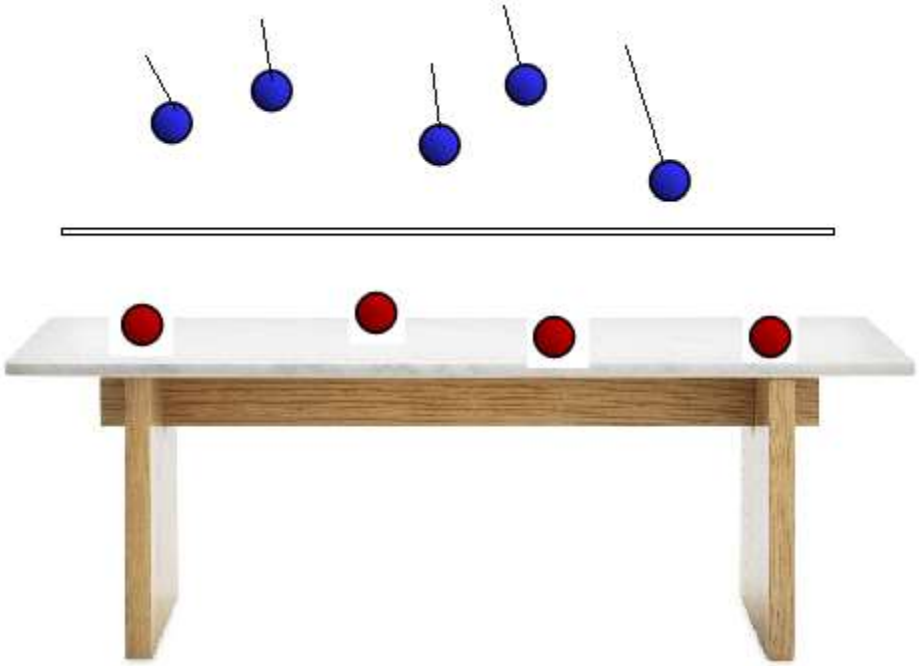
Also SVM comes with trick called 'Kernel' which is used for handling the data which all are not linearly separable.

In our example, lets say a kid comes and randomly plays the balls on the table and left it for you to palce a ruler to separate the two different colors of balls. Now there are no ordered balls on the table.



That would be difficult task as the balls are no more segregated nicely as before and Ruler can not be straight away be used. Let's think and see how we can classify.

One way to separate these balls is to increase the elevation of one set of balls with same color above. We can tie thread to balls with same color and lift above the table surface. This will give a different elevation from one set of balls with a color from another set of balls with different color. It will look as below;



Now, if we look horizontally, the two sets of balls with different colors can be separable with the help of straight ruler. Here we introduced a new hero called 'Thread' to make it happen.

As we can see, in Support Vector Machines, we add more features to existing set of features. In our above example, it is 'Thread'.

In our student sets as well, we can see for example, a set of new features can be introduced. Students study time and travel time can be used to create a new feature called 'time management' which may classify into 'Good' or 'little Good' for students.

I play an example video here for your reference.

https://www.youtube.com/watch?v=3liCbRZPrZA

We can see beautifully here how SVM works to play with features to convert the non-linearly separable data into linearly separable.

In our student data set as well, we intend to classify the students who require intervention and who does not, using the similar way of drawing a best decision boundary. As we see, the data is not so linear where we can draw conclusion easily but requires techniques like kernel to use which comes along with SVM.

References:

https://www.udacity.com/course/viewer#!/c-ud726-nd/l-5447009165/e-2428048554/m-2436168579.

Based upon the above explanations and inferences, the tuned model is expected to have the

Final F1 score for test set as: 0.802816901408