Project 2 : Building a Student Intervention System

Yoonyoung Cho

February 21 2016

1 Classification vs. Regression

This problem is an instance of **classification**, because it identifies students with a boolean label: ones who need early intervention and ones who do not.

2 Exploring the Data

Field	Value
# Students	395
# Pass	265
# Fail	130
Graduation Rate	67.09%
# Features	30

Table 1: Characterization of Data

3 Preparing the Data

[Refer to student_intervention.ipynb]

4 Training and Evaluating Models

4.1 Support Vector Machine Classifier

Classification of data based on *support vectors*, which are the edge-cases that lie on the *margin*. A margin is the buffer zone that defines the separation of the two categories.

1. Strengths
   - (a) Works well for data with *well-defined* margins – in other words, a clear separation of data based on features.
   - (b) Tends to prevent overfitting, as maximizing the margins acts against edge-cases
2. Weaknesses
   - (a) Doesn't work well with big data with lots of noises, because of the tradeoff between maximizing the margin and best classifying the data.
3. Result

Table 2: Evaluation time and scores

Set Size	100	200	300
t_{train}	.002s	.004s	.010s
t_{test}	.001s	.001s	.004s
$F1_{train}$	.850	.846	.869
$F1_{test}$	.787	.803	.797

The dataset was relatively small, and it seemed that each of the features were clearly defined. To demonstrate, here is a visualization of the SVM for two of the representative features:

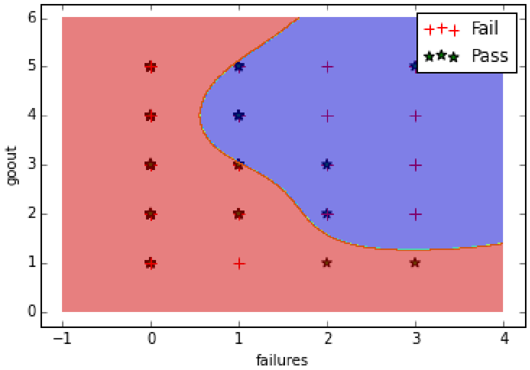


Figure 1: SVM Visualization for two features: the numerical(discrete) rating of a student's *failures* in class and social relationships(denoted as *goout*)

As shown, a clear margin is drawn in the correlation between the features of the data set and its labels.[1]

4.2 K-Nearest Neighbors Classifier

Classifies Data based on how similar a query is to the data it has already seen; hence the term *neighbor*.

1. Strengths
   (a) Fitting is Instantaneous (insertion of data).
   (b) Works well with big data, as it accumulates a lot of information.
2. Weaknesses
   (a) Assumes Weights on each feature are equal, when evaluating similarity.
3. Result

Table 3: Evaluation time and scores

Set Size	100	200	300
t_{train}	.001s	.001s	.001s
t_{test}	.002s	.002s	.003s
$F1_{train}$	.825	.822	.852
$F1_{test}$	.753	.778	.809

[1] It is remarkable that, of the 300 samples, none of the students with some of the combinations in the upper-right managed to graduate.

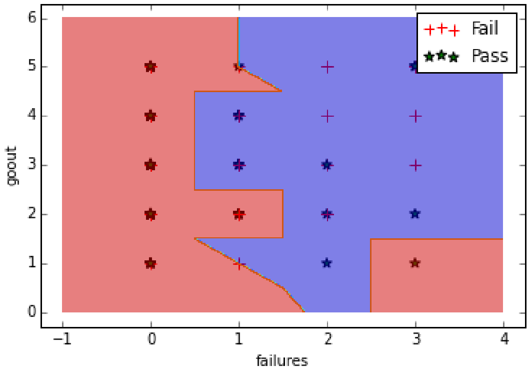


Figure 2: KNN visualization for two of the prominent features;

Considering that the data are placed in a lattice-like structure, it is completely natural that the contour seems jagged. This diagram indicates which areas are dominated by which type of students; it does not completely generalize, as – intuitively – students with 3 failures would perform worse than students with 2 failures (among the students whose social outings were rated 1. However, throughout the majority of data, the evaluation remains consistent with its neighbors.

4.3 Decision Tree Classifier

Classifies Data based on entropy; divides the data with respect to the attribute that best *splits* the data into homogeneous partitions

1. Strengths
   - (a) White Box Model : easy to understand the logic
   - (b) Logarithmic Query
2. Weaknesses
   - (a) Unstable : small variations can cause big differences in the tree generated
3. Result

Table 4: Evaluation time and scores

Set Size	100	200	300
t_{train}	.001s	.002s	.003s
t_{test}	.000s	.000s	.000s
$F1_{train}$	1.00	1.00	1.00
$F1_{test}$	.650	.71	.667

I applied the decision tree classifier in order to visualize which features, among quite a few, were the most prominent factors in the classification – i.e. most impactful. Cutting to the chase, here is the tree:

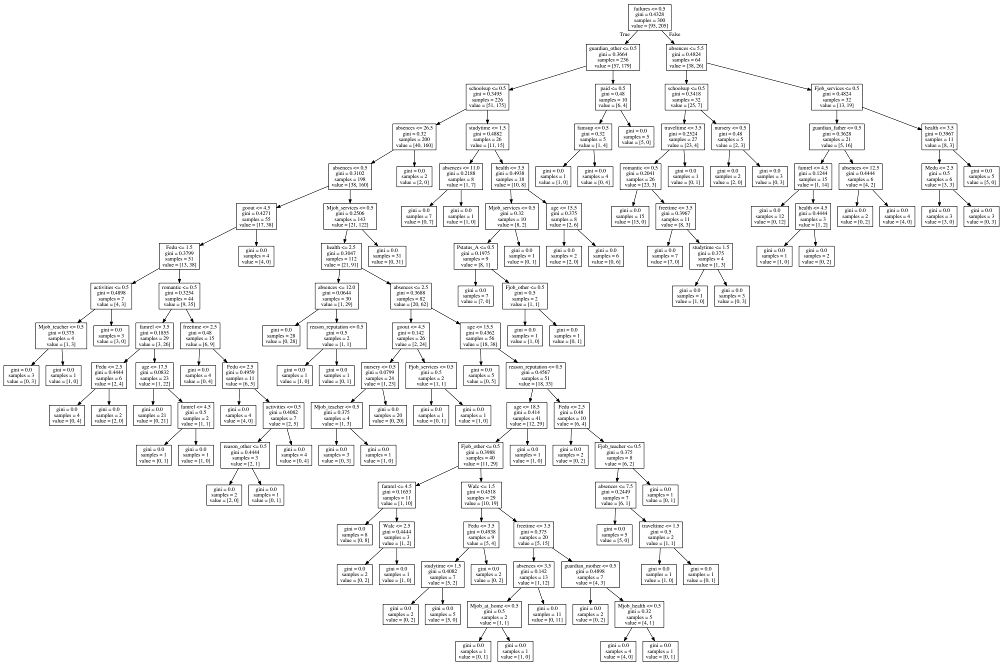


Figure 3: Full Tree

And, examining the top to find the attribute that contributes most to the information gain:

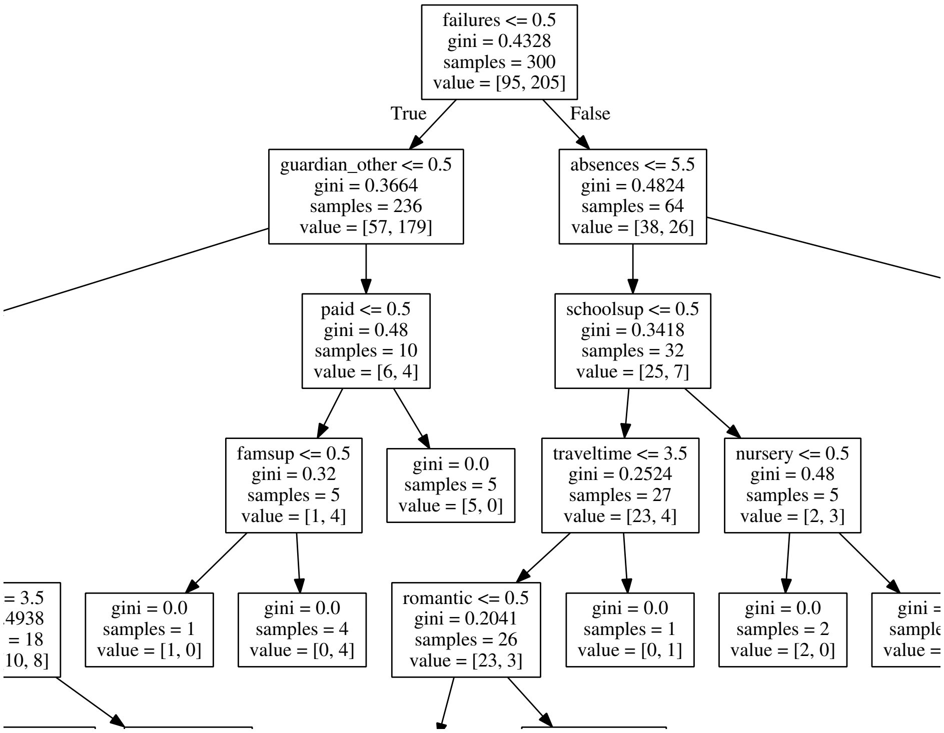


Figure 4: Top of the tree, showing the most "useful" features.

The advantage of the decision tree, as seen, is that its results are easy to interpret; for instance, it is consistent with intuition that each of the features at the top portion, such as the number of past failures, would contribute directly to whether or not a student will graduate.

4.4 Gaussian Naive Bayes Classifier

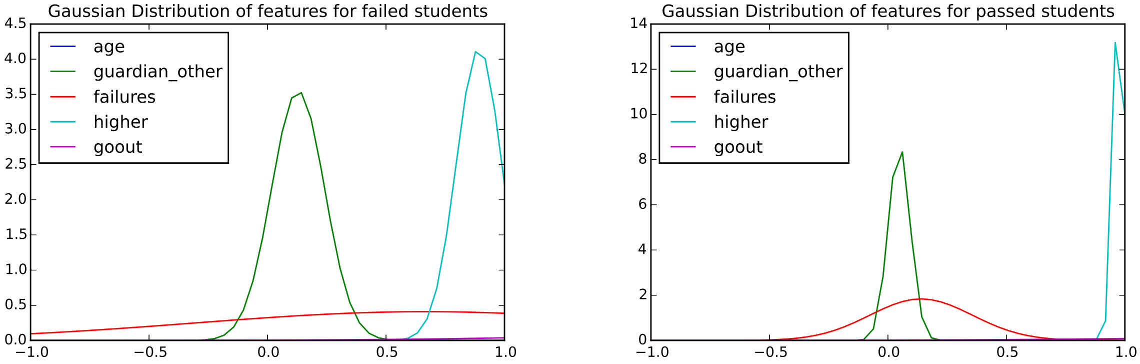
Classifies data based on how likely a *class* is to have a certain combination of values for each of the features.

1. Strengths
   (a) Logarithmic Query
2. Weaknesses
   (a) Makes assumptions about the data, such as Gaussian Noise
   (b) In theory, the features are constrained such that they are not interrelated.[2]
   (c) A feature that's not been seen in the training set associated with a particular label will yield a probability of zero, failing to generalize.
3. Result

Table 5: Evaluation time and scores

Set Size	100	200	300
t_{train}	.001s	.001s	.002s
t_{test}	.001s	.000s	.000s
$F1_{train}$	.815	.800	.799
$F1_{test}$	.683	.756	.761

The visual interpretation of the result:



(a) Failed

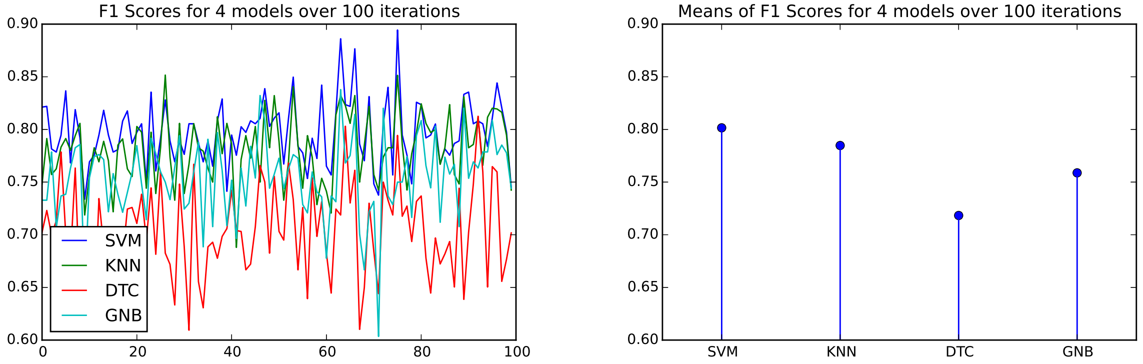
(b) Passed

Figure 5: Comparison of the distribution of five representative features

Numerically, the discrepancy is perhaps more apparent; for instance, the mean of the failures were .66 and .14, respectively, for the failed students and the graduated students.

[2] However, in practice, this constraint may often be overlooked.

5 Choosing the Best Model



(a) Scores

(b) Means

Figure 6: Scores over 100 iterations

5.1 Best Model

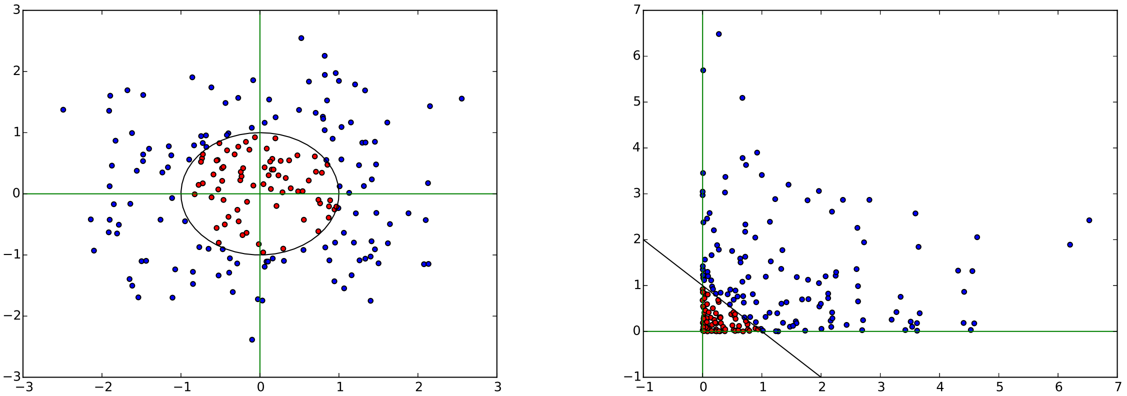
In the above figure, a clear trend can be seen that the Support Vector Machine consistently outperforms other models, overlapped occasionally by the K-Nearest Neighbor Classifier; it is also apparent that, as anticipated, the Decision-Tree Classifier is unstable, and fluctuates a slightly wider range(.60-.80) compared to other models. Gaussian Naive Bayes classifier generally had a higher score than the DTC, but it also had the lowest overall score in one occasion.

Considering that the time each model takes for fitting and predicting are all quite negligible, time is not a big concern; given the features we have, SVM is a reasonable choice, since it is relatively consistent, outperforms other models in terms of accuracy, and also performs best in its high tide.

5.2 How SVM Works

Conceptually, a Support Vector Machine operates by finding the division that best polarizes the data – in other words, putting each class of data as far apart from the other as possible. This is essentially the same as focusing on the hardest[3] samples and minimizing the ambiguity. To this end, the SVM finds the dividing "line" between the two clusters composed of each class.

[3] most difficult to categorize



(a) Original

(b) Transformed

Figure 7: Example of a Kernel Trick on nonlinear data

When the data is not linearly separable[4], through a technique called the **kernel trick**, SVM introduces another parameter (derived from the others) that allows the data to be linearly separable. For instance, in Fig.7, the data could be reformatted from parameters x,y into x^2,y^2 such that the data could be redefined with respect to the distance from the origin.

As shown, predicting the category of the input on a trained SVM is trivial: on a two-dimensional system, this is as simple as determining which side of the dividing line the point lies on; for three parameters, this would be equivalent to determining which side of the dividing plane the point lies on. For any number of features in the input, this can be generalized to which side the point lies on in an n-dimensional hyperplane.

5.3 Fine-Tuning

After fine-tuning the model, through grid-search, the final f1 score was .854. Although it had occasionally performed better in its 100 iterations (as seen from Fig.6a), this digression may be dismissed as trivial, as the score fluctuated quite a bit within the range of .75 to .9; however, it should also be noted that the fine-tuning, while it performed better than the mean f1 score in the 100 iterations, did not quite improve the model significantly.

[4] i.e. it cannot be split into two classes at opposite sides of a line