Student Intervention Project Report
Levin Jian, April 2016

1. Classification vs Regression

It is a classification problem. The output "passed" of the learned model is a discreet value. It can be either "yes" or "no".

2. Exploring the Data

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: 67.09%

3. Preparing the Data

See codes.

4. Training and Evaluating Models

Below three supervised learning models are tried for the problem, Decision Tree, KNN, SVM.

4.1. Decision Tree

1. What is the theoretical O(n) time & space complexity in terms of input size? [(1)]
   In terms of input size,
   Time complexity during training: O(N)
   Space complexity during training: O(N)
   Time complexity during prediction: it depends on the final tree, like tree depth
   Space complexity during prediction: it depends on the final tree, like tree depth
2. What are the general applications of this model? What are its strengths and weaknesses? [(2,3,4)]

Decision Trees are a non-parametric supervised learning method for classification Decision trees are non-parametric supervised learning method for classification and regression. It creates a model that predicts the value of target variable by learning simple decision tree rules, in the form of decision tree, inferred from the data features.

Its strengths:

- Simple to understand and to interpret. Trees can be visualized.
- Decision trees implicitly perform variable screening or feature selection
- Decision trees require relatively little effort from users for data preparation, no need to do feature scaling/normalization

Its weaknesses:

- They can be extremely sensitive to small perturbations in the data: a slight change can result in a drastically different tree.
- They can easily overfit. This can be negated by validation methods and pruning, but this is a grey area.

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

Decision tree allows us to answer an interesting question, that is, which input features have bigger impact on the output? In our case, the answer is as below:

Ranked features: Index([u'absences', u'failures', u'famrel', u'Medu', u'goout', u'Fedu', u'age',
u'romantic', u'paid', u'nursery', u'health', u'Fjob_other',
u'schoolsup', u'Dalc', u'Mjob_other', u'famsize_LE3',
u'guardian_father', u'Fjob_at_home', u'studytime', u'Walc', u'freetime',
u'traveltime', u'Mjob_teacher', u'reason_other', u'Fjob_health',
u'internet', u'famsize_GT3', u'guardian_other', u'Mjob_health',
u'higher', u'reason_reputation', u'guardian_mother', u'famsup',
u'sex_F', u'Mjob_at_home', u'sex_M', u'address_R', u'school_MS',
u'address_U', u'reason_course', u'Pstatus_A', u'Pstatus_T',
u'Mjob_services', u'Fjob_services', u'Fjob_teacher', u'reason_home',
u'activities', u'school_GP'],
dtype='object')

Ranked importance: [0.11940667 0.08990799 0.05929865 0.05666606 0.05337846 0.05127233
0.04484056 0.03672725 0.03312865 0.03108381 0.03060736 0.02875481
0.02664474 0.02604071 0.0241335 0.02191744 0.02182285 0.02167504
0.02115897 0.02053915 0.02053915 0.01705483 0.01566605 0.01420512
0.01329302 0.01321048 0.01312872 0.01239685 0.01232349 0.01132674
0.01095421 0.0103307 0.01026958 0.00320924 0.00308681 0. 0.
0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0.]

4. 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.

	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.002

Prediction time (secs)	0.001	0.001	0.001
F1 score for training set	1.0	1.0	1.0
F1 score for test set	0.649572649573	0.692307692308	0.615384615385

4.2. KNN

1. What is the theoretical O(n) time & space complexity in terms of input size?[5]
   In terms of input size,
   Time complexity during training: O(N)
   Space complexity during training: O(N)
   Time complexity during prediction: it depends on K, distance metric
   Space complexity during prediction: it depends on K, distance metric

2. What are the general applications of this model? What are its strengths and weaknesses?[6]
   k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametric method used for classification and regression.
   Its strengths:
   - Simple algorithm, easy to understand
   - Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.

   Its weakness:
   - For large training sets, requires large memory and is slow when making a prediction
   - Prediction accuracy can quickly degrade when number of attributes grows due to curse of dimensionality
3. Given what you know about the data so far, why did you choose this model to apply?
   KNN is known as non-generalizing machine learning methods, since they simply "remember" all of its training data (possibly transformed into a fast indexing structure such as a Ball Tree or KD Tree.). It's interesting to see how it might perform in this problem.
4. 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.

	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.001
Prediction time (secs)	0.004	0.008	0.011
F1 score for training set	0.805970149254	0.88	0.880898876404
F1 score for test set	0.724637681159	0.769230769231	0.780141843972

4.3. SVM

- What is the theoretical O(n) time & space complexity in terms of input size?[7]
  Time complexity during training: $O(N^2)$
  Space complexity during training: $O(N^2)$
  Time complexity during prediction: it depends on the number of support vectors in the final model
  Space complexity during prediction: it depends on the number of support vectors in the final model
- What are the general applications of this model? What are its strengths and weaknesses?[8,9]
  Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.
  Its strengths:
  - Effective in high dimensional spaces.
  - Still effective in cases where number of dimensions is greater than the number of samples.
  - Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
  - Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

  Its weaknesses:
  - If the number of features is much greater than the number of samples, the method is likely to give poor performances.
  - SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation
- Given what you know about the data so far, why did you choose this model to apply?
  SVM is known to work well for small to middle sized dataset, so it's worthwhile to try it out.
- 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.

	Training set size		
	100	200	300
Training time (secs)	0.105	0.005	0.01
Prediction time (secs)	0.002	0.005	0.01
F1 score for training set	0.780487804878	0.816568047337	0.811881188119
F1 score for test set	0.774193548387	0.774193548387	0.774193548387

5. Choosing the Best Model

Based on the experiments performed earlier, SVM model is chose as the best model for below reasons:

- Decision tree has the worst performance, so it's excluded
- Using default hyper parameters ,KNN and SVM has similar performance, SVM is preferred
    a) KNN requires more memory during run time
    b) Theoretically speaking, SVM is more powerful than KNN, and thus we can have a better chance of fine tuning an SVM model to get good prediction result

In layman's term, all the training examples are interpreted as points in a high dimensional space, and then SVM model constructs a hyper plane that divides the examples of the separate categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. In SVM, kernel is often used to automatically transform input features into much higher dimensional space to allow for more complex, non-linear decision boundary.

After fine tuning via GridSearch, the F1 Score is 0.795, it performs slightly better than by using default parameters (0.774)

6. Reference:

1. http://stackoverflow.com/questions/22397485/what-is-the-o-runtime-complexity-of-adaboost
2. http://www.simafore.com/blog/bid/62333/4-key-advantages-of-using-decision-trees-for-predictive-analytics
3. http://scikit-learn.org/stable/modules/tree.html
4. http://stats.stackexchange.com/questions/1292/what-is-the-weak-side-of-decision-trees
5. http://nlp.stanford.edu/IR-book/html/htmledition/time-complexity-and-optimality-of-knn-1.html
6. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
7. http://stackoverflow.com/questions/16585465/training-complexity-of-linear-svm
8. http://scikit-learn.org/stable/modules/svm.html#svm-regression
9. https://en.wikipedia.org/wiki/Support_vector_machine