Project 2: Supervised Learning

Building a Student Intervention System

Project Report

1 Classification vs Regression

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

Answer: Identifying students for early intervention is a classification problem. We are trying to find if can predicts student result and if student is going to fail then intervention can be happened ahead of time to assist them being failed. Here are few reason to clarify that.

1. Identifying students for early intervention needs discrete identification, yes or no. It don't require any other information to support that. For discrete output, classification is a best way to solve the problem.
2. Regression has continuous output which is not the case in this problem.
3. Regression has some sort of order in output but in this case there is not order, what we are really looking is yes or no.

2 Exploring the Data

1. Total number of students: 395
2. Number of students who passed: 265
3. Number of students who failed: 130
4. Number of features: 30
5. Graduation rate of the class: 67.09%

3 Preparing the Data

Demonstrated in code.

4 Training and Evaluating Models

Problem Statement: Choose 3 supervised learning models that are available in scikit-learn, and appropriate for this problem. For each model:

- What are the general applications of this model? What are its strengths and weaknesses?
- Given what you know about the data so far, why did you choose this model to apply?
- 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.

Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

Answer:

1. **Gaussian Naive Bayes**

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

   Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

   In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters.

   On the flip side, although naive Bayes is known as a decent classifier, it is known to be a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.

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

   As this is simple, relatively quick to train and gives predictions very fast, I thought to use this as a first model to evaluate. This usually gives good result for classification problem, though I had intuition that this may not be best algorithm.

   ➔ Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

	Training Set Size			
	100	200	250	300
Training time (secs)	0.002	0.001	0.001	0.001
Prediction time (secs)	0.000	0.000	0.000	0.000

F1 score on training set	0.847	0.841	0.829	0.804
F1 score on test set	0.803	0.724	0.775	0.763

2. Bagging with K Nearest Neighbors

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

A bagging classifier is an ensemble of base classifiers that fit each classifier to a random subsets of the original data and averages the prediction of the individual classifiers to create the final output. Bagging classifier is used to increase accuracy by combining the weak learners (e.g. decision trees, knn ,etc) to provide a strong learner. K nearest neighbors is used as a base classifier in this example. k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. General application for KNN can be computer vision or recommending systems.

Strengths

- Reduces variance and avoids overfitting
- KNN successful in classification problems where the decision boundary is irregular

Weaknesses

- KNN stores instances of the training data and computes nearest neighbors of each point so running time is very slow compared to other models when predicting
- KNN is a lazy learner. It will be slow during prediction when too many data points
- KNN will not know which attributes are more important than others because all attributes have same weight
- A linear combination of KNN classification can be harder to interpret than one KNN classification

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

Based on the data, It looks like decision boundary would be not linear and since KNN is non parametric so the decision boundary is flexible. Since there are not a large data points so I thought the performance will not be that bad and I used a bagging method to improve the results even further.

➔ Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

	Training Set Size			
	100	200	250	300

Training time (secs)	0.018	0.015	0.015	0.048
Prediction time (secs)	0.005	0.011	0.019	0.022
F1 score on training set	0.817	0.865	0.863	0.868
F1 score on test set	0.741	0.760	0.770	0.769

3. Support Vector Machines

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

Support Vector Machine can be used for both classification and regression problems. For classification, general applications can be text categorization or image classification.

Strengths

- Memory efficient for using support vectors.
- Works well in high dimensional spaces and where the data is not linearly separable.
- Using a nonlinear kernel trick, you can capture a more complex relationship between the datapoints.

Weaknesses

- For a large dataset with a lot of noise, SVMs will be slow and possibly produce overfitting
- If there are too many features than samples, SVM will perform poorly

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

My believe is that SVM will work better specially non linear SVM because there are 30 features and based on that the data will not be linearly separable. Also Non Linear SVM works well for a high dimensional data like the student data.

➔ Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

	Training Set Size			
	100	200	250	300
Training time (secs)	0.002	0.004	0.006	0.009
Prediction time (secs)	0.001	0.003	0.004	0.008
F1 score on training set	0.878	0.868	0.872	0.876
F1 score on test set	0.775	0.781	0.773	0.784

4. Random Forest

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

Random Forest is an ensemble of decision trees. Random forest is mostly applied to answer classification problems and not usually great for regression because this don't do good with outside of the training data range. Some general applications could be predicting weather or deciding to go to a specific restaurant or not.

Strengths

- Accurate and runs efficiently on large datasets
- Random Forest corrects the decision tree's possible overfit to the training set.
- While only one decision tree is likely to suffer from high variance or high bias, random forest uses the technique to average its predictions to find the right balance.

Weaknesses

- Slow predictions
- For better accuracy, need more trees. This can slow down the training performance.
- Will not work as well with a small dataset

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

Although the dataset is small, It was reasonable to try the Random forest model. However, there is a possibility that the data could contain some outliers and this can affect the accuracy. Random forest is robust to outliers because it uses a subset of training sets with bagging and a subset of features to reduce the outliers effect on the result.

➔ Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

	Training Set Size			
	100	200	250	300
Training time (secs)	0.031	0.043	0.021	0.019
Prediction time (secs)	0.001	0.001	0.001	0.002
F1 score on training set	0.992	0.986	0.997	0.995
F1 score on test set	0.759	0.786	0.767	0.713

5 Choosing the best model

1. 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?

   After analyzing all the model, I guess Support Vector Machine Algorithm is the best model. Support Vector Machine is known to do well in high dimensional space, producing non linear decision boundaries, and due to its implementation it is memory efficient. On the contrary, it can perform badly when there are more features than samples, and if there are too many data points and noise, it will produce overfitting.

   Based on our test, SVM came out as winner and it look like most suitable for this project data. Here is my analysis based on result data.

   - Training Time – Gausian performed the best for training time, trailing by SVM. Other two are bad on training time.
   - Prediction Time – Gausian and Random Forest did very good. SVM follows very close to them. Bagging with KNN is not good and it increase as data increase.
   - F1 Score on Training Set – Random Forest did best followed by SVM.
   - F1 Score on Test Set – SVM did best followed by Bagging with KNN and Gausian.

   So based on above analysis SVM is the most appropriate as we have enough data to test. The F1 Score is the higher for training size 200 and 300 which shows it perform better even the training size increases. We have more samples than features, thus we do not have to worry about a bad performance. As for cost, SVM is memory efficient and faster than the K nearest neighbor. Although the random forest model is faster in training and prediction than SVM, its performance is not as great as SVM. You can see that the random forest model does very well (almost 1 as its F1 score) during training but the score drops by more than 0.2 for prediction. This is a clear symptom of overfitting. This can be fixed by growing a larger forest but prediction will be slower.

2. 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).

   The Support Vector Machine can classify whether a student needs early intervention or not by creating a line between the students based on their background information.

   Let’s say students who need early intervention and who don’t are all standing around on a field. They are positioned on the field based on their extracurricular activities and GPA. Our goal is to separate those students and create some kind of boundary. We can draw a straight

line to accurately separate those students. However if we add new students to the field, students who need early intervention may be on the wrong side of the line vice versa. What Support Vector Machine does in this situation is to draw a better line between the two groups that has a maximum gap on either side of the line.

However if we observe other factors that could affect the student's performance in school such as their parent's occupation, the students position will be more complicated on the field. It becomes harder to separate the students with just a line. Here, SVM is able to separate them with a line but in a higher dimension.

3. 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.

   Best parameters for the final tuned SVM model is {'kernel': 'rbf', 'C': 1, 'degree': 3}

4. What is the model's final F1 score?

   F1 score for test set: 0.783783783784