Project Report
<Student Intervention>

Jihun Mun Brian

1. Classification V.S. Regression. Which one is more proper to apply in this project?

In this supervised machine learning project, I will use Classifier algorithm. Our data set has two parts, features part and result part. Our result set can be classified into 2 groups, passed or not. Also given features set with unknown results, we will predict result passed or not. So it is classification problem.

2. Training and Evaluating Models

Our data set is high dimensions. It has 30 features. And also the number of data set is not that big. So I choose 3 models as following.

- KNearestNeighbors Algorithms
- Decision Tree Algorithms
- SVM - SVC Algorithms

From now on, I will analyze the pros and cons about these 3 models.

1) KNearestNeighbors Algorithm

PROS	CONS
No training involved : We don't need to train our model. New example can be added anytime without time cost!!	Instead, querying time of the predicting data is expensive and slow. => O(log(n) + k)
Very powerful, and complexity of parameter is simple. The only thing what we have to do is decide n_neighbors.	

To sum up, low train cost (O(n)) and (compared)expensive run time (O(log(n) + k)). But the result would be not bad. And number of our example is not large, so I choose this model.

2) DecisionTree Algorithm

PROS	CONS
Querying time(Running time)is very fast.	Training time and complexity can be expensive
Normally Decision tree is very interpretable. Other algorithm is back box algorithm, but this is white box algorithm.	It is easy to overfitting. So you should prune when you make decision tree algorithm.

DecisionTree algorithm is very interpretable and once it is trained, the running time would be very fast. But you should be careful about managing complexity(overfitting problem).

3) SVM-SVC Algorithm

PROS	CONS
It is very effective when data set has high dimensional space. Also memory effective.	Training time is very long when the data set is large. So normally It is very ineffective with large number of training sets.
It works really well when there are a clear margin space between classification.	It is very weak at noise and overlapping.

SVM-SVC algorithm is very expensive algorithm(training time is very long). But in our case, the data set has high dimensionality and it contains small number of examples. So this algorithm is very suitable for our situation.

3. Choosing The Best Model

At this section I will choose the best model among the above 3 algorithms. To do this Let's compare the three tables below.

KNearestNeighbors Algorithm	Training Set Size		
	100	200	300
Training Time(sec)	0.001	0.001	0.001
Prediction Time(sec)	0.002	0.003	0.005
F1 score for training set	0.823529411765	0.847457627119	0.842352941176
F1 score for test set	0.815789473684	0.828947368421	0.853333333333

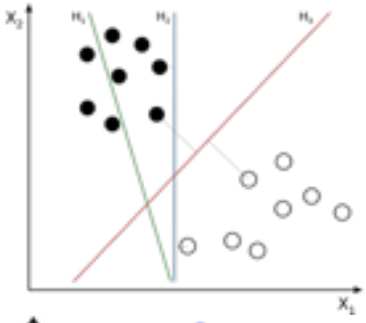
DecisionTree Algorithm	Training Set Size		
	100	200	300
Training Time(sec)	0.001	0.001	0.002
Prediction Time(sec)	0	0	0
F1 score for training set	1	1	1
F1 score for test set	0.736842105263	0.6875	0.682926829268

SVM-SVC Algorithm	Training Set Size		
	100	200	300
Training Time(sec)	0.002	0.003	0.009
Prediction Time(sec)	0.001	0.002	0.007
F1 score for training set	0.896551724138	0.854430379747	0.862921348315
F1 score for test set	0.85	0.832298136646	0.8375

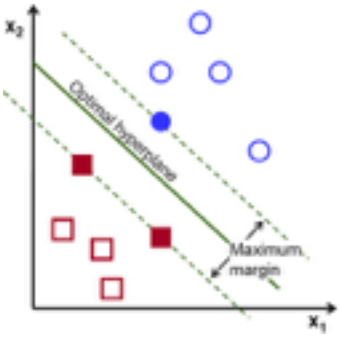
1) Their performance result are all similar except DecisionTree. DecisionTree suffers from overfitting problem. Because we didn't set any additional setting for max_depth. And with its default value, the tree will be expanded until all leaves are pure.
Both of KNN and SVM-SVC algorithms' perform very well. I can choose anything of them because they show the similar F1 score. However, considering the fact that SupportVectorClassifier is very suitable on small number of data set with high dimensionality, I will choose SVM-SVC algorithm for the best suitable model.

2) Now let's talk about SVM-SVC algorithm more detain in layman's term. Basically this algorithm classify the data set. Given the training data set which target is already classified, this algorithm involves finding the hyperplane (hyperplane can be line or 2D dimensional plane, …, more generally, hyperplane is n-1 dimensional space when the data set contains n features). That hyperplane should be clearly divide the data set to classified groups with maximum margin. We call the hyperplane with maximum margin is a optimal hyperplane of model. So finding the optimal hyperplane of model is the purpose of SVC algorithm. I will explain in more detail about hyperplane and maximum margin.

* (Optimal) hyperplane and maximum margin

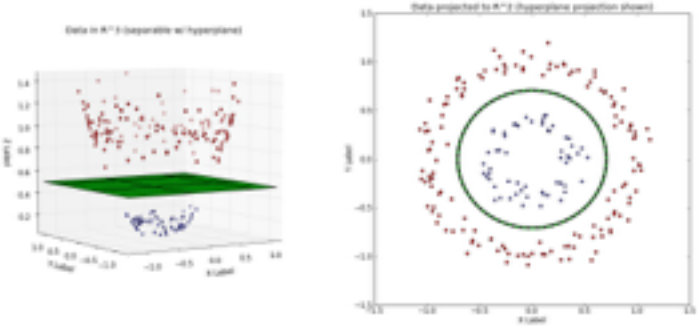


As you can see on left picture, H2 and H3 can be hyperplane. Both are exactly divide the data set which is already classified. But the line 'H3" is the optimal hyperplane. Because margin of H3 is bigger than of H2. So H3 has the maximum margin. So If we apply SVC algorithm in this case, we can obtain optimal hyperplane like H3.



Maximum margin can be sounds little vague. If so, you can check the left picture.
Maximum margin is the minimum orthogonal (to hyperplane) distance between support vector which belong to different classified groups.

Above cases are 2-dimensional features case. So the hyperplane has less then 2 dimensional space. Following examples show the high dimensional hyperplane example.



We talked about general concept and some concepts like hyperplane and maximum margin. From now on I will talk about more on training process using our student_intervention_project as an example.

[STEP1] Obtain Separating Hyperplane. In our case, we have two classification, passed or not. Hyperplane will divide these two groups and optimal hyperplane have maximal margin. Margin is the distances between support vectors and hyperplane!
(Support Vectors : data points that lie closest to hyperplane)
So based on result(passed), we can obtain hyperplane which is equation of features(sex, age, …)

In Step1, we usually can't separate perfectly with linear hyperplane. In this case we use 'Kernel Trick'. So Kernel function will transform the original data to map into new space so that data can be linearly separable. For example, let's see fourth picture. Let's say we transform like x->x^2, y->y^2. Then x^2 + y^2 = C will be optimal hyperplane. Now x^2, y^2 are new features, and with these new axis, we can separate nonlinear area.
Generally we use many kernel functions(polynomial, or gaussian kernel, …) so that we can divide area as we want.

[STEP2] Based on optimal hyperplane we will classify new data set(test_set). There is two group, above(outside) of hyperplane or below(inside) of hyperplane.

Result : If we follow above step, we can predict our test_set. As a result, if we calculate F1_score, we can obtain 0.8375.
If we use grid search algorithm, we can find more suitable model for our data set. Basically It will determine complexity terms(e.g. kernel function or gamma term).
F1_score(using grid_search) : 0.83950617284.
We can check out that the F1_score with grid_search algorithm perform slightly better than normal SVC algorithm.