DS 8002 Machine Learning Project 2 – Unsupervised and Supervised Learning (December 2016)

Najlis, Bernardo J. (Student Number: 500744793)

Abstract—**In this project, you will apply several algorithms to two data sets. Please answer each question in the order they appear. Do not skip to later steps to answer earlier questions that ask you to predict outcomes based on your analysis of the data and understanding of the algorithms.**
Submit your report in D2L by midnight on the due date.

I. Introduction

IN this project we perform analysis on two the same data sets (iris and contact lens) as in the previous project using five different types of machine learning algorithms and techniques:

- Clustering – Simple K-means
- PCA – Principal Component Analysis
- SVM – Support Vector Machines
- Random Forest
- AdaBoost

The analysis was done using R and libraries for each type of algorithm used.

This report is accompanied by an R markup document with all the code and its output to support all answers and reasoning presented here.

II. SVM

Which kernel works better? Why?

With **Iris**, using tune() to select the best cost for each of the four kernels evaluated (linear, polynomial, radial and sigmoid). As the accuracy varies depending on the test set, through several runs the best kernel is **Polynomial.** Here is one of the sample runs:

SVM Kernel	Best Cost	Accuracy
Linear	100	0.9623
Polynomial	10	1
Radial	1	0.98
Sigmoid	1	0.931

With **Contact Lenses** I also used tune() to select the best cost for each of the four kernels evaluated (linear, polynomial, radial and sigmoid), and the best kernel through multiple runs is also **Polynomial.**

SVM Kernel	Best Cost	Accuracy
Linear	1	0.6364
Polynomial	10	0.9091
Radial	10	0.7273
Sigmoid	10	0.6364

Polynomial is the best probably because it can provide a model to separate classes with higher complexity than linear, radial and sigmoid.

How did the SVM compare to the classifiers from Project 1 in terms of training time and performance?

The table below summarizes my conclusions when comparing SVM with the classifiers from project 1 (KNN, Decision Trees and Multilayer Perceptron). I also added the “Knowledge Extraction” as another variable to consider (if required by the use case). Overall, SVM provides higher performance than KNN and Decision trees but at a higher cost, still providing some degree of knowledge that can be extracted. For classifiers, SVM is probably the mid-range best option if the training time can be afforded (i.e. batch training use case): provides high performance at a reasonable increase in training time.

	Training Time	Performance	Knowledge Extraction
SVM	**MED**	**HIGH**	**MED**
KNN	**LOW**	**LOW**	**MED**
Decision Trees	**LOW**	**LOW**	**HIGH**
Perceptron	**HIGH**	**HIGH**	**LOW**

III. PCA – SVM

Run PCA and then run SVM on the reduced data.

With both data sets we first apply log() transformation on the data, and we center and scale when doing PCA analysis.

How many principal components did you pick? Why?

On Iris, I picked the first two components, which account for 96% of the total variance.

On Contact lenses, I picked three components as they all have the same variance, leading to 75% percent of total variance. Accuracy will suffer by losing 25% of the total variance, but it is worth trying to see by how much.

How did the SVM perform on the reduced data compared to the original data?

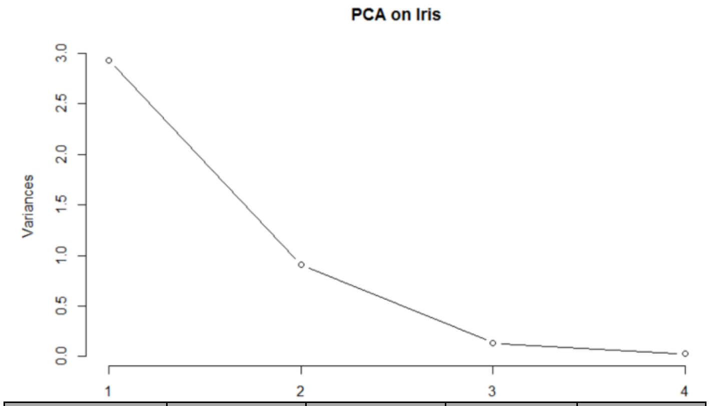
On Iris, as the best kernel is Polynomial, the comparison analysis is based just on that kernel. **The accuracy is about 10% lower than when using all original features.** This is expected as we are only using a subset of the features, but still a good tradeoff ratio: reducing 50 percent of features (which means we also reduced training time by half) we only lost 10 percent of accuracy.

On contact lenses, the best we can get by is trying with three principal components and 75% total variance. **The accuracy went down to 0.4545 (about half of using four components).** This indicates that for this data set (also considering its size) it is not worth using PCA to do dimensionality reduction. Also, any variance in accuracy are more related to the training/test data sets used than to the specific alternatives and tweaks done to the models.

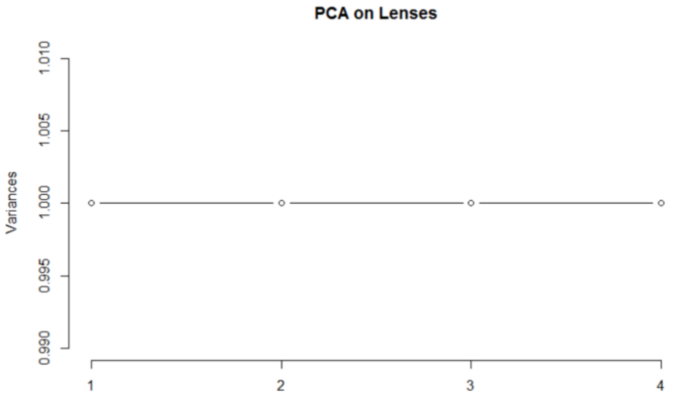
How much of the variance in the data is described by the first two or three principal components? Show visually.

In iris, after normalization and centering, the first two components describe 96% of the variance, and the first three describe 99.3% of the variance.

In Contact Lens, the variance is equal among all variables, the first two describe 50% and the first three describe 75% of the total variance.



	PC1	PC2	PC3	PC4
Cumulative Proportion	**0.7331**	**0.9599**	**0.99314**	**1.0000**



	PC1	PC2	PC3	PC4
Cumulative Proportion	**0.25**	**0.50**	**0.75**	**1.00**

IV. AdaBoost – Random Forest

How did the boosting or bagging compare to the J48 results from Project 1?

On **Iris**, J48 in Project 1 had 0.8936 accuracy, and using Random Forest I obtained 0.9362 accuracy. **Performance is better using Random Forest**.

On **lenses**, as the data set is too small, I used 3-fold cross validation to evaluate performance on J48. **Surprisingly accuracy went down to 0.6667 with Random Forest vs. 0.8333 using J48.**

V. Clustering (K-Means) / Decision Tree / SVM

Run clustering (k-means) and then apply decision tree and SVM on clustered data.

Compare the performance with the previous results.

With Iris, adding a cluster feature to the data set before running SVM varies the accuracy by a very small factor (reducing from 1 to 0.9821). In other cases, it may increase the performance as well. Results are very tight to draw a definitive conclusion. One important detail that can be noted is that **when adding the cluster feature** based on the columns suggested by NbClust() (two clusters) **the SVM accuracy is higher than when using 3 centroids**, even though the underlying data is really split into three classes.
Still with Iris, running decision trees with the additional cluster feature doesn’t affect the performance as the cluster feature is not used at all in any of the tree stumps.

With lenses, adding a cluster feature reduces accuracy significantly (down to 0.6667 from 0.9091). It is as if adding the additional cluster number feature (being based on two or three centroids) makes it more difficult to run SVM. Also, if running decision trees with the additional cluster feature does not make any difference, as the cluster feature is not used by any of the stumps in the decision tree.

This leads to the conclusion that adding clustering features to the set can or cannot improve accuracy, and it must be determined based on the data set.