

Term Project #4: Bioinformatics

Classification and diagnostic prediction of the recurrence of cancer using gene expression profiling and different classifiers.

The purpose of this project is to compare different classification models in an attempt to diagnose the recurrence of cancer in patients using gene expression profiles.

The data:

The data consists of 55 patients, or samples, with 54613 genes, or features. The values of the features express the level of expression of that gene in the patient. Each patient is known to have or not have a recurring case of cancer. This data is used to train a classifier so that given a set of features, it can tell whether the patient's cancer will recur.

Strategy:

A classifier needs to, given a set of features, give a probability of that sample having a recurring case of cancer (here on referred to as Class 1). Various algorithms exist for this, and a few will be tested. This data set presents two difficulties: there are very few samples and very many features.

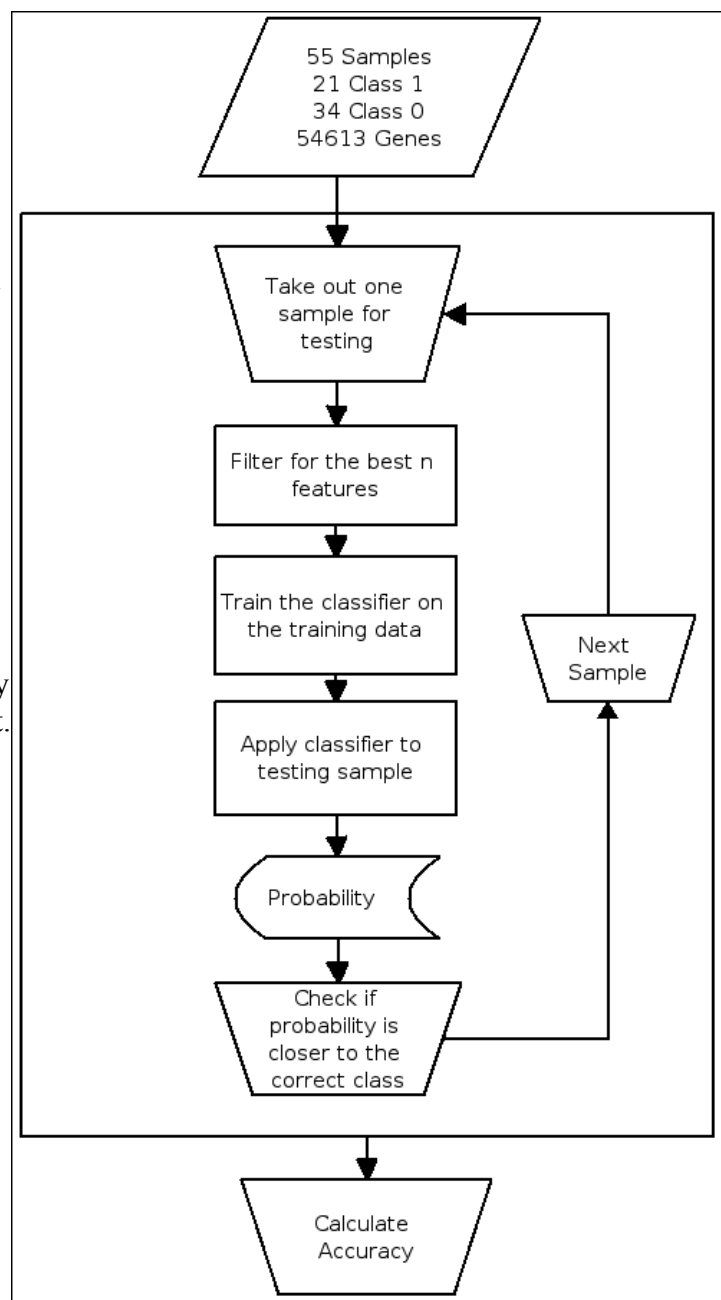
The more samples there are, the more there is to learn from, so the more accurate the prediction can get. This applies to machine learning as it does to human learning. This data set only has 55 samples, which limits the potential to find trends.

The amount of features greatly complicates computations on the data. First of all, it takes a lot of computing power to do computations on over 54000 features. Even efficient algorithms will take an exorbitant amount of time classifying such a large data set. The features need to be filtered out for such practical reasons, as well as some scientific ones. There are many features which are irrelevant and only introduce noise. These features need to be filtered out. More directly, the point of the investigation is to find the genes that determine the class of the patient. These are expected to be a certain relatively small subset of the features. It is important to use these features, and not other irrelevant ones for the classification.

Initial ideas and testing:

The initial idea was to use a decision tree and apply it to the entire data set. However, the available implementation of the decision tree relied on discrete data, and all values in this data set are continuous. It would not be reasonable to discretize the data using clustering and not knowing how many clusters there are.

The first test was performed with the Naive Bayes classifier and simple correlation feature selection. When tested directly on the training data with optimal parameters this provided a 98% accuracy (all but one sample were correct). This was tested with different thresholds for the



correlation of the features, and .51 was found to be the best (from a range of 0-1). This resulted in about a dozen features. However, this figure is meaningless as there is no testing data. Further tests were done with leave one out cross-validation.

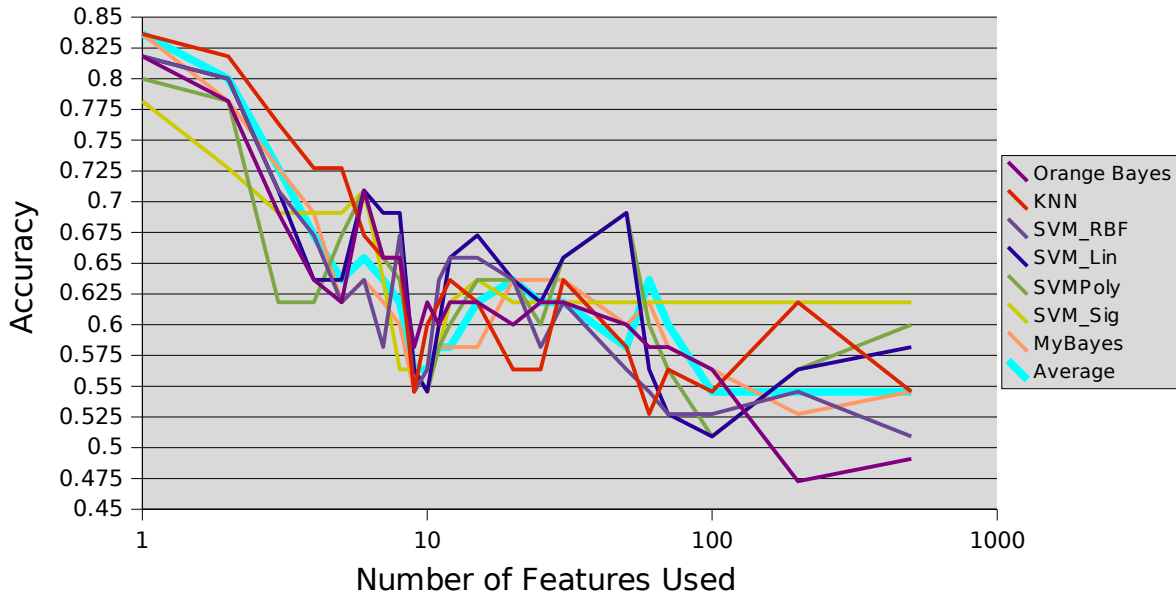
Feature selection:

All feature selection was done using simple correlation. Various amounts of best features between 1 and 500 were tested. Feature selection, being the only part of the algorithm that deals with the entire data set, was a resource limiting factor in testing. Other feature selection methods were only available for discrete data, and discretizing the data could not be done within a reasonable time with the available methods. So, practical limitations were the primary reason for only using the simplest feature selection algorithm and focusing on a variety of classifiers instead.

The classifiers:

Several classifiers were used. One was a self-implemented version of the Naive Bayes classifier. The rest of the classifiers were from the Orange data mining library¹. These include their version of the Naive Bayes classifier, k-Nearest Neighbor classifier, and SVM (support vector machines with C support vector classification) with linear, polynomial, radial basis function (RBF), and sigmoid kernels. An average of the probabilities given by the two Naive Bayes classifiers, k-NN, and SVM with linear and RBF kernels was taken in an attempt to get the best of all the classifiers.

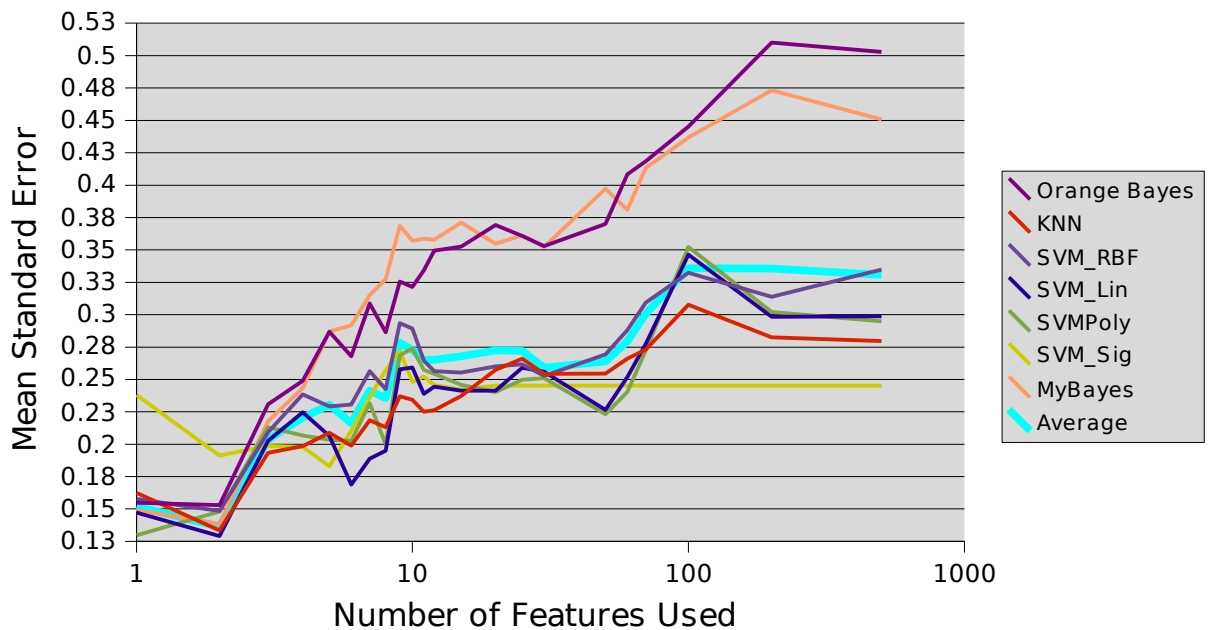
Accuracy vs. Features Used



Number of Features	Accuracy							
	Orange Bayes	k-NN	SVM_RBF	SVM_Lin	SVM_Poly	SVM_Sig	MyBayes	Average
1	0.82	0.84	0.82	0.82	0.8	0.78	0.84	0.84
2	0.78	0.82	0.8	0.8	0.78	0.73	0.78	0.8
3	0.69	0.76	0.71	0.71	0.62	0.69	0.73	0.73
4	0.64	0.73	0.67	0.64	0.62	0.69	0.69	0.67
5	0.62	0.73	0.62	0.64	0.67	0.69	0.62	0.64
6	0.71	0.67	0.64	0.71	0.71	0.71	0.64	0.65
7	0.65	0.65	0.58	0.69	0.65	0.64	0.62	0.64
8	0.65	0.65	0.67	0.69	0.64	0.56	0.6	0.62
9	0.58	0.55	0.55	0.56	0.56	0.56	0.55	0.56
10	0.62	0.6	0.56	0.55	0.55	0.55	0.56	0.56
11	0.6	0.62	0.64	0.6	0.58	0.58	0.58	0.58
12	0.62	0.64	0.65	0.65	0.6	0.62	0.58	0.58
15	0.62	0.62	0.65	0.67	0.64	0.64	0.58	0.62
20	0.6	0.56	0.64	0.64	0.64	0.62	0.64	0.64
25	0.62	0.56	0.58	0.62	0.6	0.62	0.64	0.62
30	0.62	0.64	0.62	0.65	0.65	0.62	0.64	0.62
50	0.6	0.58	0.56	0.69	0.69	0.62	0.6	0.58
60	0.58	0.53	0.55	0.56	0.6	0.62	0.62	0.64
70	0.58	0.56	0.53	0.53	0.56	0.62	0.58	0.6
100	0.56	0.55	0.53	0.51	0.51	0.62	0.56	0.55
200	0.47	0.62	0.55	0.56	0.56	0.62	0.53	0.55
500	0.49	0.55	0.51	0.58	0.6	0.62	0.55	0.55

Number Features	Mean Standard Error							
	Orange Bayes	k-NN	SVM_RBF	SVM_Lin	SVM_Poly	SVM_Sig	MyBayes	Average
1	0.15	0.16	0.16	0.15	0.13	0.24	0.15	0.15
2	0.15	0.13	0.15	0.13	0.15	0.19	0.14	0.14
3	0.23	0.19	0.21	0.2	0.21	0.2	0.22	0.2
4	0.25	0.2	0.24	0.22	0.21	0.2	0.24	0.22
5	0.29	0.21	0.23	0.21	0.2	0.18	0.29	0.23
6	0.27	0.2	0.23	0.17	0.2	0.21	0.29	0.22
7	0.31	0.22	0.26	0.19	0.23	0.24	0.32	0.24
8	0.29	0.21	0.24	0.2	0.2	0.26	0.33	0.24
9	0.33	0.24	0.29	0.26	0.27	0.27	0.37	0.28
10	0.32	0.23	0.29	0.26	0.27	0.25	0.36	0.27
11	0.33	0.23	0.26	0.24	0.26	0.25	0.36	0.26
12	0.35	0.23	0.26	0.24	0.25	0.25	0.36	0.27
15	0.35	0.24	0.26	0.24	0.25	0.24	0.37	0.27
20	0.37	0.26	0.26	0.24	0.24	0.25	0.35	0.27
25	0.36	0.27	0.26	0.26	0.25	0.25	0.36	0.27
30	0.35	0.25	0.25	0.26	0.25	0.25	0.35	0.26
50	0.37	0.25	0.27	0.23	0.22	0.25	0.4	0.26
60	0.41	0.27	0.29	0.25	0.24	0.25	0.38	0.28
70	0.42	0.27	0.31	0.28	0.27	0.25	0.41	0.3
100	0.45	0.31	0.33	0.35	0.35	0.25	0.44	0.34
200	0.51	0.28	0.31	0.3	0.3	0.25	0.47	0.34
500	0.5	0.28	0.33	0.3	0.29	0.25	0.45	0.33

Mean Standard Error vs. Features Used



The results:

The best results were with one feature, gene 1570141_at (feature #54346). This feature, while generally having low values, is relatively consistent in that it is expressed in samples with class 0 and not samples with class 1. This provided an accuracy up to 84% with k-NN and average. The mean standard error for the average of the probabilities was .15.

All classifiers showed a downward trend in accuracy with more features. Using two features reduced the mean standard error down to .14, but that increased from there on. Using simple correlation to find features, additional features do not help classify the data. k-NN gave the best accuracy with 1 to 5 features. SVM with a linear kernel was best with 15 to 50 features. With a sigmoid kernel, SVM no longer gave different results with more than 15 features. The Naive Bayes algorithm from the Orange library was the only one to ever fall below 50% accuracy. Since there are two classes, at that point the classifier was doing a worse job than a random guess. However, all the classifiers gave very low accuracy (mostly <60%) with more than 50 features.

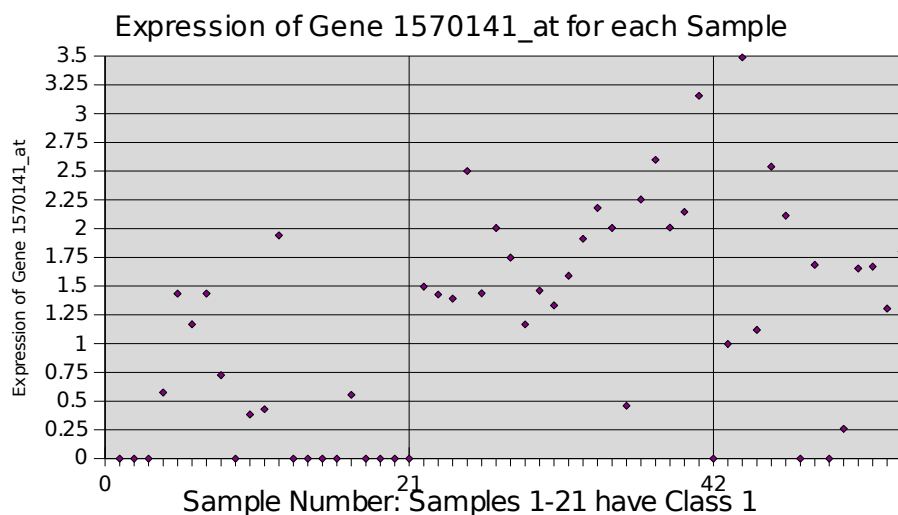
Although accuracy-wise Naive Bayes kept up with the other classifiers, it showed its lack of robustness in its mean standard error, which climbed much more quickly than the other classifiers'. The error with Naive Bayes was up to 90% worse than with k-NN, which had the least error with the greatest number of features (other than SVM with a sigmoid kernel which, as mentioned, mysteriously flattened out). SVM with a polynomial kernel started out with the lowest error. All the classifiers maintained a relatively decent and flat error relative to Naive Bayes.

Additional Testing

Preliminary testing was done with a decision tree from the Orange¹ library. With 55-fold cross validation, it yielded 74% accuracy.

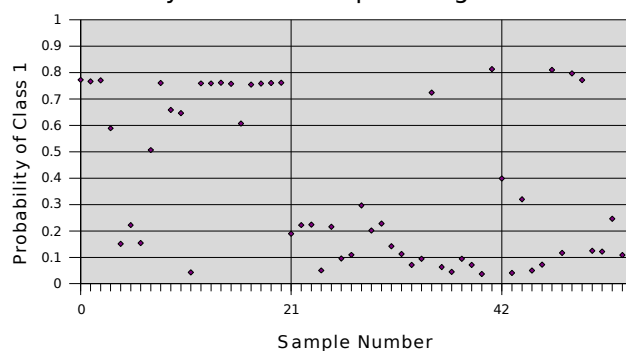
Further Investigation

Various classifiers were tested, but simple correlation was always used for feature selection. Other methods of feature selection need to be tested, along with refining the current feature selection method. For example, gene 1570141_at, while yielding the best results, generally had low values which might throw off the data. It might be helpful to filter out features with a very low average value and variance. Initial testing, by throwing out 1570141_at and using 239180_at, the second best feature, as the only feature shows this method is not particularly helpful, as it yielded lower accuracy in that test. Another improvement would be to calculate the variance of a feature for

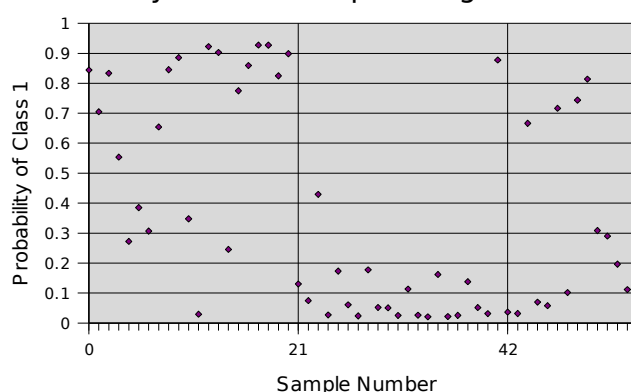


Most expression values for this gene are 0, or less than 1 in the samples with class 1. Most samples with class 0 have expression values greater than 1.

Probability for each sample using 1 feature



Probability for each sample using 2 features



Probabilities using 1 and 2 features: It is visible that with two features, the correct probabilities are closer to the correct value, while the wrong answers aren't as close to the wrong value. In fact, with one feature even the highest probabilities for class 1 are less than 80%. This explains why the mean standard error is lower with two features.

each class, and throw out samples that don't fit with the other 95% of samples in that class. This is one method to remove outliers that don't fit otherwise obvious trends. The effect of outliers is visible in the results, as some samples with class 1 yield a nearly 0% probability, and vice-versa, even when other samples are classified accurately with that feature set and classifier.

Technical Notes:

All programming done in Python. Simple correlation, Naive Bayes, and cross validation were implemented. The rest of the classifiers were from the Orange data mining library.

Citations:

¹Demsar J, Zupan B, Leban G (2004) Orange: From Experimental Machine Learning to Interactive Data Mining, White Paper (www.ailab.si/orange), Faculty of Computer and Information Science, University of Ljubljana.