Meaningful Biomarkers for Predicting Pathological Tumor Stage in Prostate Cancer

60-473/574: Advanced Topics in AI - Machine Learning/Pattern Recognition

William Briguglio, Tajinder Dhillon, Alexandru Filip, Jai Priyadarshi

December 16th, 2018

Summary: In this report we consider a dataset from cBioPortal of 494 samples(patients) containing clinical variables (such as Pathological Tumor Stage) and the gene expression data of approximately 60,000 genes. Multiple feature selection and classification techniques are applied, and their results analyzed, in order to determine the most predictive subset of genes. We were able to obtain a relatively small feature set that could predict the Pathological Tumor Stage with a very high precision. In light of our results, we concluded that the genes we've found are meaningful biomarkers for predicting Pathological Tumor Stage.

1. Introduction:

With the recent resurgence of machine learning in both academia and the industry, multiple machine learning and pattern recognition algorithms have been applied to a host of problems from multiple disciplines. One such discipline has been the medical field where neural networks have been used to identify cancerous tissues and other anomalies in diagnostic images, while other machine learning algorithms have been applied as pattern recognition to assist pathologists in diagnoses. Pathology is the medical discipline that is concerned with the diagnosis of disease based on analysis of bodily tissues and fluids in a laboratory. Our research is one such example of the application of pattern recognition in pathology. We examined a data set containing clinical variable data and the gene expression data for approximately 60,000 genes for 487 patients. One of the clinical variables involved was Pathological Tumor Stage(a.k.a Path-T-Stage). We focused on determining the meaningful biomarkers (i.e. genes) for predicting Path-T-Stage. Path-T-Stage is normally determined by examination of a removed tumor, thus accurately predicting Path-T-Stage with gene expression data could provide a non-invasive approach for doctors and clinicians to obtain potentially important data, prior to any surgery taking place.

2. Problem Description:

The datasets is from cBioPortal[1], and contained two large files. The first contains clinical variables (such as Path-T-Stage, Gleason score, progression, etc.) for 494 samples(patients). The second contains the gene expression data of approximately 60,000 genes for 499 patients. The 494 patients in the first file are a subset of the patient set in the second file, so we first removed the extra gene expression samples from the data set. Our intent is on finding a minimal subset of the gene

expression data to train a classifier to detect Path-T-Stage, so we removed all other clinical variables from the data set. Additionally we concerned ourselves only with the Path-T-Stage base class (i.e. 'T2', 'T3', and 'T4') and ignored the subclasses ('Tx*a*', 'Tx*b*', or 'Tx*c*').We hope to find a classifier which can accurately predict the Path-T-Stage of a tumor based on its' gene expression data. In this way we can find and verify the most important biomarkers(genes) for predicting Path-T-Stage.

To this end we set out to thoroughly test multiple classification algorithms, being: Support Vector Machines (with linear, polynomial, and radial basis function kernels) k-Nearest Neighbor, Naïve Bayes, and Random Forest. We tested a variety of classification schemes (discussed later) and parameters to ensure thorough results. Additionally, due to the high dimensionality of the data (60,483 features and 487 samples) we employed several feature selection and normalization techniques to decrease computation time and increase the practicality of the resulting classifier. In sections 3 and 4 we go into more detail about how these techniques and algorithms were implemented. After thoroughly testing each classifier, we compared results to determine the most efficient classifier and identified the features it used as most meaningful. The efficiency of the classifier was determined by its' weighted precision since there was high class imbalance amongst our data.

Class	Number of Samples	Percentage
T2	186	38.19%
T3	291	59.75%
T4	10	2.05%

3. Method:

We chose to test multiple classification schemes as this would give us more thorough results as well as determine what classifier worked best for identifying a specific class and thus which features were most important for distinguishing that class. The first four classification schemes we tested were; T2 vs. T3 vs. T4, T2 vs. {T3,T4}, T3 vs. {T2,T4}, and T4 vs. {T2,T3}.

Next, for each classification scheme, the following work flow was repeated to determine the most effective classifiers.

1. Cleaning Data: The data was loaded from the original files provided by Dr. Rueda and any samples with invalid labels for Path-T-Stage were removed, the extra gene expression samples were removed, and the extra clinical variables were removed.
2. Macro Feature Reduction: The 60,483 features were reduced by removing features with zero or near zero variance among the samples with sckikit-learn's VarianceThreshold()[2] function. Next using scikit-learns SelectFromModel()[3] function, the features were further reduced to between 400-3500 features depending on the classification scheme being tested.
    a. The estimator passed to the function was scikit-learn's Random Forest classifier[4] with 100 trees and the class weights set to 'balanced'. In scikit's implementation, setting the class weights to 'balanced' will automatically adjust weights inversely proportional to class frequency. In this way we eliminated the bias towards one class the feature selection would have had from class imbalance in the dataset.

   b. SelectFromModel with Random Forest was used because it is much quicker than other feature selection approaches and provided good results. This allowed us to efficiently reduce the size of the feature set to something manageable by a more thorough feature selection algorithm that more often finds optimal results.
3. Micro Feature Reduction: Next, with a feature selection algorithm dependent on the classifier being tested, we iteratively determined the $n<300$ most important features for multiple values of n.
4. Obtain Best Parameters: For each value of n, we used grid search with cross validation to find the best parameters for the classifier being tested on those n features. Then tested the classifier using cross validation and recorded the precision.
5. Evaluate Classification: Comparing the precisions, we determined the most effective classifier for the structure being tested and extracted the features used by said classifier as our meaningful biomarkers.

After the above process was finished for all 4 structures, we determined that class T4 was very easily separated from the other classes and so we focused our effort on separating T3 from T2. The above process was repeated for this final structure by ignoring samples with class T4 by removing it from the dataset. We did not test T2 vs. T4 or T3 vs. T4 because we were not able to separate T3 from the rest, or T2 from the rest with sufficiently high precision or sufficiently low number of features.

Note: all estimators used balanced class weights to counter class imbalances, and grid search and classifications were all done using 3-fold cross validation to ensure the features/classifiers found were not over fitted to the test data.

4. Classifier Performance:

4.1. SVM with Linear Kernel:

Implementation:

This classifier was implemented using scikit-learn's SVC() function[5]. The classification took place in two phases and was repeated four times for each of the schemes discussed earlier. First scikit-learn's SelectFromModel() was used to obtain the most important features with a Random Forest Classifier with class weight set to 'balanced' and 100 estimators(i.e. trees) but otherwise default parameters. This would vary between 400-1500 features depending on the structure being tested.

Our program would then enter a loop which iterated over different values of n (multiples of 10 from 10 to at least 100) which would determine the number of features used for the next phase of the classification which took place inside the loop. Within the loop, the following steps took place:

1. The subset of features obtained from the select from model approach was further reduced using scikit-learn's RFE()[6] function, with an SVM using linear kernel as the estimator, to obtain the n most important features.

Note: RFE was used because it selects features by recursively considering smaller and smaller sets of features. It is slower (hence why it wasn't used from the start) but much more thorough and thus provided us good results.

2. The optimal parameters where then obtained using Scikit-learn's GridSearchCV()[7] function with 3-fold cross validation, weighted precision as the scoring function, and the features obtained from step 1.
3. The parameters obtained in step 2 and the features obtained in step 1 where then used with 3-fold cross validation to obtain the confusion matrix from classifying our dataset with the optimal parameters and features found in steps 1 and 2.

In this way we obtained the confusion matrices for the optimal parameters on the n most important features for n = [10, 20, 30, …]. We then compared the precision score of each of these classifications to determine the optimal number of features.

Results:

After repeating this entire process 4 times, once for each scheme, we found that the optimal performance for this classifier was achieved when classifying T4 against the rest, with linear kernel and C = 1, scoring a 100% weighted precision. Similar results were obtained for the other classification schemes (weighted precisions between 93-98%) however this required between 250 and 350 features and we believed we could achieve better results by using two classifiers; one which determined membership in T4 vs. {T2, T3} and one which determined membership in T2 vs. T3 alone. Considering our results, we proceeded to repeat the same process for T2 against T3. We were able to obtain a weighted precision of 91.7% with only 40 features using 3-fold cross validation and the SVM's C parameter set to 1.

4.2. SVM with Polynomial Kernel:

Implementation:

Scikit learn's SVC was used to implement the SVM classifier with polynomial kernel. The classification was performed on the four classification schemes discussed earlier in order to find the best precision or PPV (positive predicted value) resulting in obtaining the most accurate decision boundary. The reason behind classifying based on the earlier mentioned schemes is to find an optimal subset of features (using feature selection algorithms) for the main classes (T2, T3, T4) which is hard to visualize, thus obtaining an optimal decision boundary. The implemented code can be explained as follows:

1. **Dataset Pre-Processing:** The first part of the code deals with data preprocessing for removing noise, variations and discrepancies that had occurred while recording the data. The noise, variations and discrepancies were removed both separately and conjointly from the clinical and gene data in order to keep the same patient IDs/number of samples in both the files. This not only improves PPV performance measure but is also the first step of pattern recognition procedure. The pre-processing also considers the conversion of the sub-labels such as {T2a, T2b, T2c} into T2 label depending on schemes explained earlier. The loadData() method

performs the pre-processing of the datasets. For example: after preprocessing gene dataset = (487, 60,483) and clinical dataset = (487,).

2. **Feature Selection:** Feature selection is the second step of pattern recognition procedure. As the gene dataset contains 60, 483 features, the objective was to find the smallest subset of features giving the highest PPV after classification. Two known feature selection algorithms were implemented to obtain optimal subset of features between 10- 60 as follows:
    a. **Random Forest:** Random forest Scikit's tree-based feature selection approach was used to find the best subset of features using the ranking of the features. The number of trees in the forest used were 50 (n_estimators = 50) coupled with the 'select from model' and 'transform' methods to obtain best reduced set of features. The best subset of features obtained were approximately 4000.
    b. **Minimum Redundancy and Maximum Relevance (mRMR):** mRMR algorithm accurately selects features relating to genes. The algorithm tends to select a subset of features having the most correlation with class and the least correlation between themselves. It ranks features based on the mRMR criterion which is based on mutual information. Though the algorithm is slow limiting the selection up to 60 features because it selects features one by one by applying a greedy search to maximize the objective function, which is the function of relevance and redundancy. mRMR was implemented using a GitHub implementation[8] that uses mutual information to obtain the given set of optimal features. Parameters passed to the mRMR are number_of_features = {10, 20, 30, 40, 50, 60} and verbose = 2 which gives the mRMR criterion score (mutual information) coupled with fit and transform methods.

3. **Classification:** SVM with polynomial kernel was used as classification algorithm to classify the dataset with selected subset of features i.e., {10, 20, 30, 40, 50, 60} to compare and obtain the best PPV performance measure. SVM with polynomial kernel finds a quadratic decision boundry in original space and linear decision boundary in higher dimension space without explicit mapping. Grid search has been implemented to tune the parameters and find the best set of params giving the best PPV value. Scikit learn's grid search class with cross validation = 3 was used to implement the functionality. The params used were degree = 2 and C = {1.0, 10.0}. We also print the confusion matrix in addition to PPV to obtain extra insight about classification of predicted vs actual class labels.

Results:

The above-mentioned implementation steps were used on all the four classification schemes I.e., T4 vs T2 vs T3, T2 vs {T3, T4}, T3 vs {T2, T4} and T4 vs {T2, T3}. After implementing grid search with SVM polynomial kernel, C = {1.0, 10.0}, degree = 2 and cross validation = 3 we found that the PPV value was the best i.e., 97% for T4 vs {T2, T3} for subset of 10 best features obtained after feature selection. Considering our results, we proceeded with classifying T2 vs T3 with the same implementation steps and found that the PPV value was 64.8% for the subset of 10 best features obtained after feature selection.

4.3. SVM with Radial Basis Function Kernel:

Implementation:

Before starting classification, the number of features (genes) was reduced from 60,483 to 32,086 by removing all features with variance less than 0.01. Scikit's VarianceThreshold class was used for this

At first feature selection was performed using chi-squared. However, due to its univariate nature, chi-squared proved to be a bad method to use for feature selection on this dataset. We settled on using a Random Forest Classifier along with scikit's SelectFromModel class to select the most important features. The class weight was set to balanced and 100 estimators were used. This resulted in 5,070 important features which were used for further classification. L2 normalization was applied and the features were scaled to have a mean of 0 and variance of 1 using scikit's normalize and scale functions respectively. Doing this improved classification results and speed.

Classification:
One vs. all classification was done with T2, T3 and T4, by considering one class to be positive and all other classes collectively to be negative.

Scikit-feature's implementation of mRMR was used to perform further feature selection. The number of features chosen started at 10 and was increased by steps of 10 up to 70.

At every step an SVM classifier was used with a radial basis function kernel. Grid search was performed using the GridSearchCV object. When performing grid search the values of C and gamma were manipulated. Both variables used were powers of 10. The values of C were 1, 10, 100, ... 10000. The values of gamma were 1, 0.1, 0.01, ... 0.00001.

The scikit's SVC object was used to perform SVM-rbf classification.

The best results were $C = 10$ and $\text{gamma} = 0.01$

After performing grid search the best values of C and gamma were used to perform the 1 vs. all classification. Cross validation was performed to find the precision of the classifier. The number of features chosen were those that resulted in the highest precision, in this case 40 with a precision of 97.2%.

The same process was used to perform classification between T2 and T3 since the SVM linear classifier was able to achieve 100% precision and accuracy. In this case 66.5% accuracy was achieved using 60 features.

4.4. k-Nearest Neighbors:

The same process used for the SVM with RBF kernel was also performed using K-nearest neighbour where the number of neighbours and p (the exponent for the Minkowski distance) were varied.

4.5. Naïve Bayes:

Implementation: For this classifier scikit-learn's GaussianBC()[9] was used. Before classification scikit-learn's SelectFromModel() with a Random Forest Classifier was used to select the most important features (genes). The Random Forest Classifier with the parameters set to 'balanced' and 100 estimators, all other parameters were set to their default settings.
Similarly to the previous classifier our program would loop through different values of n (increments of 10 from 10 to 100), which represented the number of features used for the classification. Within the loop the following steps took place:

1. A subset of features is selected containing n features was produced from the SelectFromModel function where the max_features was set to n.
2. Once the features are selected then the classifier was used with 3-fold cross validation, the weighted precision was recorded for that n number of features.

Results: This was repeated for the 4 time, once for each scheme, this performed best in the scheme when it was T4 against all, scoring 98.3% weighted precision with 10 features. Which is much better than the classifier trying to distinguish each class alone, T2 vs T3 vs T4, which only had a weighted precision of 72.7% with 300 features. For T2 vs T3 the same implementation strategy was used as explained in SVM with Polynomial kernel section with the only difference of classification algorithm and removing of T4 labels obtaining 477 patient samples. The classification algorithm used was Naïve Bayes with cross validation = 3. The best PPV value obtained was 68.81% for the subset of best 10 features after feature selection.

4.6. Random Forest:

Implementation: For this classifier scikit-learn's RandomForestClassifier() was used. Before classification scikit-learn's SelectFromModel() with the Random Forest Classifier was used to select the most important features (genes). The Random Forest Classifier with the parameters set to 'balanced' and 100 estimators, all other parameters were set to their default settings, this reduced the number of features before further reducing them using RFE.
Similarly to the previous classifier our program would loop through different values of n (increments of 10 from 10 to 100), which represented the number of features used for the classification. Within the loop the following steps took place:

1. A subset containing n features was produced from the RFE() function where the number feature to select was set to n.
2. Once the features were selected then grid search was used to obtain the best parameters for the Random Forest Classifier. In the grid search the parameter max_depth with values [10, 25, 50, 75, None], min_samples_split with values [2, 5, 10] , min_samples_leaf with values [1, 2, 4], and n_estimators with values [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 200], were used.
3. Once the optimal parameters for Random Forest with n features were obtained, 3-fold cross validation was preformed, then the weighted precision where recorded with the parameters.

Results: This was repeated 4 times, once for each scheme. The scheme that performed best was T4 against all, scoring 98.7% weighted precision with 20 features. The grid search obtained max_depth

75, min_samples_split 10, min_samples_split 4, and n_estimators 600. Which is much better than the classifier trying to distinguish each class alone, T2 vs T3 vs T4, which only had a weighted precision of 68.7% with 60 features. For T2 vs T3 the same implementation strategy was used as explained above. Classification of T2 vs. T3 using Random Forest resulted in a precision of 73.75% with 30 features, with parameters max_depth 10, min_samples_split 2, min_sapmles_leaf 2, and n_estimators 400.

5. Discussion of Results:

After each classifier was tested with each classification scheme, we recorded the best precision achieved and the number of features it used, for each classifier and scheme combination. Following are our results:

Key: # of Feat : Precision		Classifier					
		SVM(linear)	SVM(poly)	SVM(rbf)	k-NN	Random Forest	Naïve Bayes
Structure	T2 vs. T3 vs. T4	250 : 93.4%	60 : 60.0%	70 : 57.4%	40 : 55.0%	60 : 68.7%	300 : 72.7%
	T2 vs. {T3,T4}	250 : 96.5%	10 : 67.0%	60 : 66.8%	60 : 60.4%	80 : 69.9%	150 : 77.0%
	T3 vs. {T2,T4}	350 : 97.5%	60 : 65.0%	40 : 64.6%	30 : 56.9%	60 : 68.4%	80 : 72.8%
	T4 vs. {T2,T3}	14 : 100%	10 : 97.0%	40 : 97.2%	10 : 95.9%	20 : 98.7%	10 : 98.3%
	T2 vs. T3	40 : 91.6%	10 : 64.8%	60 : 66.5%	50 : 59.0%	30 : 73.0%	10 : 68.8%

Nearly all the classifiers had insufficient precision when working with the first three classification schemes with the exception of SVM with linear kernel. However sufficient precision was only achieved when using well over 150 features, with the best precisions achieved at 250-300 features. The fourth scheme achieved >95% precision for all the classifiers, with the linear kernel SVM being able to correctly distinguish samples belonging to class T4 with 100% precision, 14 features, and 3-fold cross validation. This high precision is suspicious, for example it could be that this classifier does not generalize well because the data set it was trained on contained too few T4 class samples which were not representative of the entire class. However, there is no way to know for sure without more data.

Because of our results, we decided to focus on separating T2 from T3 next. Only one classifier performed well with this classification scheme, being the SVM with linear kernel with a 91.6% precision with only 40 features.

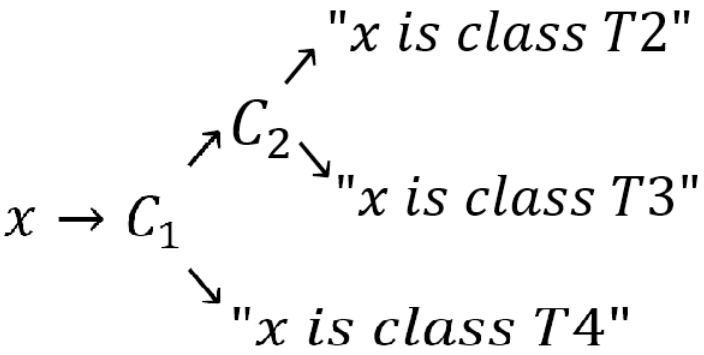
In theory, the two linear kernel SVM classifiers which achieved respectively 100% and 91.6% precision with the 4^{th} and 5^{th} classification schemes can be combined to form a single two step classifier. In this way the classification precision would become the product of the two precisions achieved, $(91.6\% \times 100\%) = 91.6\%$, and the classifier would thus still maintain a high level of precision, while only using 54 features. Because of this, we consider these 54 features to be the most meaningful in predicting Pathological Tumor Stage of prostate tumors using gene expression data. Following are the gene ID's of the 54 genes used by the two-step classifier:

Genes Useful for Distinguishing T2 from T3

Genes Useful for Distinguishing T4 from T2 and T3
ENSG00000143294.13
ENSG00000159239.10
ENSG00000255114.1
ENSG00000143363.14
ENSG00000231368.1
ENSG00000263862.1
ENSG00000232936.4
ENSG00000214207.2
ENSG00000267225.1
ENSG00000197044.9
ENSG00000174469.16
ENSG00000204055.4
ENSG00000255571.5
ENSG00000270617.1

The following figure illustrates the two-stage classification process:

- x is an unclassified sample
- C_1 is the SVM with linear kernel using 14 features
- C_2 is the SVM with linear kernel using 40 features



ENSG00000241757.3
ENSG00000167004.11
ENSG00000227309.1
ENSG00000222640.1
ENSG00000240687.1
ENSG00000212938.3
ENSG00000197217.11
ENSG00000277945.1
ENSG00000253773.2
ENSG00000232842.2
ENSG00000106610.13
ENSG00000277130.1
ENSG00000279034.1
ENSG00000165169.9
ENSG00000183662.9
ENSG00000068781.19
ENSG00000064886.12
ENSG00000239392.2
ENSG00000256944.1
ENSG00000142945.11
ENSG00000171223.5
ENSG00000233918.1
ENSG00000230227.4
ENSG00000278746.1
ENSG00000214894.5
ENSG00000265403.1
ENSG00000220721.1
ENSG00000274825.1
ENSG00000175471.18
ENSG00000199609.1
ENSG00000130227.15
ENSG00000270560.1
ENSG00000157965.11
ENSG00000237398.1
ENSG00000212580.1
ENSG00000228615.1
ENSG00000124713.5
ENSG00000255495.1
ENSG00000241542.3
ENSG00000228339.1

6. Conclusion:

By testing several feature selection and classification algorithms, we were able to obtain a classifier with 91.6% precision that used only 54 features. Thus, we were able to determine the 54 most meaningful biomarkers in our original feature set of 60,483 features, thereby reducing our feature set size by 99.91%. In future research there remains the possibility that other classification algorithms not tested here would perform better with higher precision or fewer features, however we consider our research a proof of concept that pathological tumor stage can be reliably determined using gene expression data. We would also like to have tested our final two-staged classifier on another, not seen before, dataset. However, we believe we competently countered the effects of class imbalance and overfitting using balanced class weights and cross validation.

7. Role of Group Members:

Group Member	Tasks	Contribution Percentage
William Briguglio	1. Ran all tests (feature selection, grid search, classification) and wrote related code that concerned the SVM with linear kernel, as well as the test concerning SVM with RBF kernel with classification scheme 5 2. Wrote Topic and Description 3. Wrote Summary and Sections 1,2,3,4.1,5,6 of report	25%
Jai Priyadarshi	1. Ran all tests that and wrote related code that concerned the SVM with polynomial kernel, as well as the test concerning Naïve Bayes with classification scheme 5 2. Wrote section 4.2 of the report	25%
Alexandru Filip	1. Ran all tests and wrote related code that concerned the SVM with rbf kernel for classification schemes 1-4, as well as all tests concerning k-NN 2. Wrote section 4.3 and 4.4 of the report	25%
Tajinder Dhillon	1. Ran all tests and wrote related code that concerned the Naïve Bayes for classification schemes 1-4, as well as all tests concerning Random Forest as a classifier (all participants used random forest for feature selection) 2. Wrote section 4.5 and 4.6 of the report	25%

8. References:

[1]http://www.cbioportal.org/study?id=prad_tcga#summary

[2]https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.Variance Threshold.html

[3]https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel .html

[4]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier. html#sklearn.ensemble.RandomForestClassifier

[5]https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[6]https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html# sklearn.feature_selection.RFE

[7] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html #sklearn.model_selection.GridSearchCV

[8] https://github.com/danielhomola/mifs

[9] https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html# sklearn.naive_bayes.GaussianNB