CSC371 Machine Learning
Cancer Classification Data Analysis

Eric Xu and **Kostas Mateer**
{erxu,komateer}@davidson.edu
Davidson College
Davidson, NC 28035
U.S.A.

Abstract

Given data files from The Cancer Genome Atlas (cite) that is a database of 2895 patients with 1882 features of miRNA profile. Each data file had a higher level folder corresponding to the type of cancer the patient has. Each data file comprised of miRNA ID marker and the reads per million marker of that specific RNA. The data was organized by cancer type folder, then another folder, then the data file on specific patient. Within that data file, the rows of the full data file comprised of one row in the compiled matrix with the markers corresponding to the read files. The 'reads_per_million_miRNA_mapped' column normalized value indicating the level to which the specified miRNA was expressed in this patient's tissue — these are the values of the features. We built two tuned classifier models to predict the cancer type. We were able to build two models that utilized different algorithms of classifying: SVM model and a Decision Tree Classifier model. Support Vector Machine models find a hyperplane in an N-dimensional space, where N is the number of features, that classifies the data points. It finds the maximum margin between each class of data and finds the optimal hyperplane. A Decision Tree Classifier model utilizes yes/no questions on each feature to split the dataset until the data points are belonging to each class. We built two models for comparison purposes with the scikit-learn package for SVM and Decision Tree Classifier. The SVM model had an accuracy score of .97 and the Decision Tree Classifier model had an accuracy score of .938.

1 Introduction

Being able to accurately predict the outcomes of possible events, allows for a seemingly endless amount of opportunities to open up in terms of growing success, increasing productivity, building ideas once thought to be impossible, and so on. Even beyond the computer science and business fields, companies and people of all areas would benefit from the ability to do so, as the best possible decision could always be predicted. Using the concepts within machine learning, companies and other professions may be able to predict prices, human tendencies, images, and even disease. Even people at-home who just have knowledge of machine learning concepts and processes have been able to make programs for their benefit.

From a medical sense, the use of machine learning has proven to be an extremely important factor in the prediction of diseases, helping doctors, and making the medical industry more efficient. Hospitals could help patients more quickly and reduce the load on doctors, especially during times like the COVID-19 pandemic that has burdened hospitals and doctors.

Using svm and Decision Tree Classification models, we used thousands of patients data with their miRNA markers to predict the classification of cancer. These are vastly different models with different approaches to classification.

2 Data Analysis

We did data analysis on The Cancer Genome Atlas. At first, we failed to fully understand the structure of the data. We did not realize each specific file was a patient example, so our initial analysis was not productive. We created averages of all the read counts of each miRNA marker and the `'reads_per_million_miRNA_mapped'` column and created plots for some patients refer to Figure 1. This was a failed plot. However, the columns in the individual patient data file are not columns in the compiled data file. Instead, the column `'reads_per_million_miRNA_mapped'` is an individual row in the compiled data matrix.

In order to compile the data into one file, we had to use a python data structure to walk through the directory. Using the `python os` package with `os.walk` and `os.path.join` is the methodology we used. The `os.walk` method walks through the whole directory, and we open each path that is returned, extract the `'reads_per_million_miRNA_mapped'` column into an array; add it as a new row with a classification number from 0-5. 0 was Breast Invasive Carcinoma, 1 was Kidney Renal Clear Cell Carcinoma, 2 was Lung Adenocarcinoma, 3 was Lung Squamous Cell Carcinoma, 4 was Pancreatic Adenocarcinoma, and 5 was Uveal Melanoma.

The amount of patients is 2895, and the compiled matrix is 2895 x 1882. We noticed that some features had 0 values for every patient, so we created a structure to remove 0 valued features. This is done by going through each column, summing it, checking to see if the column sum is 0, then dropping that column if it is 0. The amount of features that are 0 valued were 187. The final shape of the compiled data matrix is 2895 x 1695.

Normalization

There are two ways of normalizing the data:

- Transform original data into a new set of data, in which mean value is 0 and standard deviation is 1.
- Reduce the scale of the data to a range of (0,1)

We finally chose the second method over the first one for faster operation speed. Because there are many zeros in the data even after we processed the raw data, it will be the best to keep these zeros in order to make operation faster.

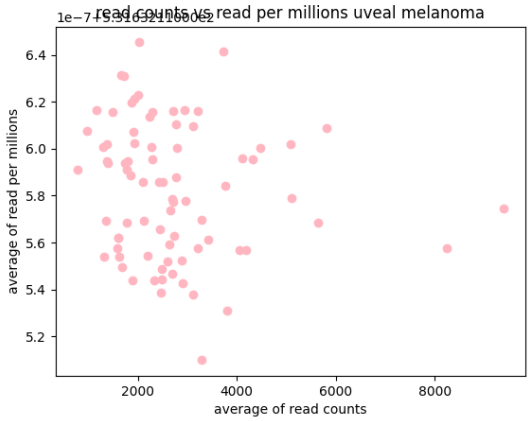

Figure 1: Failed Scatter Plot

3 Experiments

In this section we discuss our two models that we utilized: svm Model and Decision Tree Classifier Model. We talk about what each model is and how it works, and we discuss the hyper-parameter tuning and selection methods that we used to choose hyper-parameters.

Support Vector Machine(svm) Model

A support vector machine constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for data training for classification. A good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class, because larger margin can produce lower error when we input the model with new, unknown data. In SVC, given training vectors:

$$x_i \in R^p (i = 1, ..., n), y \in \{1, -1\}^n$$

The goal is to find $w \in R^p, b \in R$ such that the prediction is made by the following function:

$$w^T \phi(x) + b$$

We implemented scikit-learn package in Python library to build the model for our dataset. Specifically, we used:

```
from sklearn import svm
clf = svm.SVC(kernel = 'linear', C = 1)
clf.fit(x_train, y_train)
predicted = clf.predict(x_test)
```

This estimator implements C-Support Vector Classification. The implementation is based on libsvm. In the classifier, there are two hyper-parameters, which are choice of kernel and regularization value(C). Tuning these parameters will be elaborated in hyper-parameter tuning section.

Decision Tree Classifier Model

For our experimenting purposes, we chose the Decision Tree Classifier model. Following the selection of a data set to use, the first step is to split the data set into a training set and a test set. The training set is used to build and train the model until it is accurately making predictions, whereas the test set is used to test the performance of the model created by comparing the predicted values of a test set to the actual values.

For the Decision Tree Classifier Model, we used the decision tree classification algorithm. This model organizes the data into a tree structure. The basic intuition is that each node is a yes/no question on a specific feature. The algorithm runs as follows: select a feature with the highest gain (measure of information gain), drop that feature from dataset, repeat till entire data is pure/run out of features. The main equations used for Decision Tree Classifier Model is as follows:

This is the entropy equation (measure of purity of dataset)

$$S(D) = P_y \log_2 P_y - P_n \log_2 P_n$$

The symbols and their meanings are as follows: $P_n = \frac{n}{n+y}$, $P_y = \frac{y}{n+y}$, n = #of no's, y = #of yeses

This is the gain equation (information gain)

$$G(S', feature) = S(D) - (\frac{\#D_y}{\#D} S(D_y) + \frac{\#D_n}{\#D} S(D_n))$$

The symbols and their meanings are as follows: $S(D)$ = original entropy, $\#D_y$ = number of examples in D_y (dataset of yeses), $\#D$ = number of original dataset, $S(D_y)$ = entropy of D_y

The `sklearn` module that we used is `tree`. The code to build the model is as follows:

```
from sklearn import tree
clf = tree.DecisionTreeClassifier(max\_depth=i)
clf.fit(X_train, y_train)
y\_predict = clf.predict(X_test)
```

Hyper-parameter Tuning

The determination of hyper-parameter of svm model is done by a gridsearch on two variables: choice of kernel and regularization value(C). The result of gridsearch is shown in Table 1.

Therefore, it is obvious to see that linear model has the best performance with accuracy of 0.93448276, and it seems that value of C does not influence linear model's result. So we ended up choosing linear model as kernel and C = 1 for convenience.

The hyper-parameter of a Decision Tree Classifier model is how in depth the tree goes. The depth of the tree means how specific it fits to each feature, so a tree can continue to

Table 1: Gridsearch Results

kernel&C	accuracy	performance rank
linear & 1	0.93448276	1
poly & 1	0.62758621	20
linear & 2	0.93448276	1
poly & 2	0.69655172	19
linear & 3	0.93448276	1
poly & 3	0.73793103	18
linear & 4	0.93448276	1
poly & 4	0.748275865	17
linear & 5	0.93448276	1
poly & 5	0.78965517	16
linear & 6	0.93448276	1
poly & 6	0.82413793	12
linear & 7	0.93448276	1
poly & 7	0.82413793	12
linear & 8	0.93448276	1
poly & 8	0.82758621	11
linear & 9	0.93448276	1
poly & 9	0.82413793	12
linear & 10	0.93448276	1
poly & 10	0.82413793	12

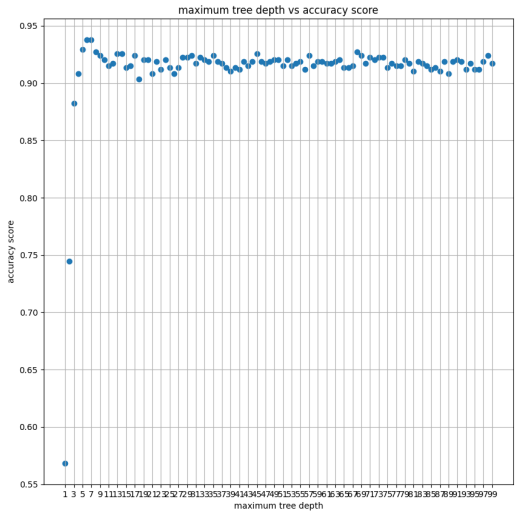

Figure 2: max_depth vs accuracy_score

go till every feature is used to classify, or it can go till whatever level the tree is set to using the `max_depth=i` where i equals how deep. Generally, the less in depth the tree is the better it is for testing data because it is not overfitted to the specific training data.

We ran the Decision Tree Classifier with varying levels of depth from 1-99. It was found that the Decision Tree Classifier was most accurate with an `accuracy_score` of .938 with a `max_depth=7`. Figure 2 shows the `accuracy_score` as a function of the `max_depth`.

Validation

We did a K-fold validation for svm model using

```
K_fold = svm.SVC(kernel = 'linear')
validation_result = cross_val_score(K_fold, x_val, y_val, cv = 5)
```

It is found that the mean accuracy of 5-fold validation of our svm model is around 0.91, which is acceptable and adequate.

We did not do any validation for the Decision Tree Classifier model because we did not feel a need for it.

4 Results

In this section we discuss the different models and their performances. We noticed that the svm Model performed better than the Decision Tree Classifier model.

After testing our data from the svm algorithm, the average accuracy over the test dataset is 0.97, which is higher than expected. Figure 3 shows this below. Also, the operation time for building SVC model is about 8 to 10 seconds, which is fast enough.

Our sci-kit learn Decision Tree Classifier model `tree` performed not as well as the svm algorithm model. The highest accuracy recorded was .938; however, this is not a low performing model in any sense. We got this score from using sci-kit's `accuracy_score` function. This function compares a matrix of predicted classifications to the actual classifications. Figure 2 shows this once again. The algorithm runs well, and does not take as long as other algorithms such as K-Nearest Neighbors (KNN) which was another classifier model that was in debate to be used. The We decided to not use KNN because of the nature of it. KNN would not work because Euclidean distances become less characteristic with the more features since the feature space becomes too crowded. A feature space of about 1800 makes it difficult to use KNN accurately. We also utilized the package `graphviz` in order to visualize the tree. A portion of the tree structure is shown in Figure 4. Every arrow to the left is a 'yes' answer to the question posed in the node, and every arrow to the right is a 'no' answer.

Classification report for classifier SVC(C=1, kernel='linear'):

	precision	recall	f1-score	support
0.0	1.00	0.99	0.99	218
1.0	1.00	0.97	0.99	116
2.0	0.96	0.95	0.96	110
3.0	0.91	0.96	0.93	95
4.0	0.93	1.00	0.97	28
5.0	1.00	0.92	0.96	12
accuracy			0.97	579
macro avg	0.97	0.96	0.97	579
weighted avg	0.97	0.97	0.97	579

Figure 3: SVM model results

5 Conclusions

In this section we discuss the conclusion from our group's model building.

Support Vector Machine(svm) model is not a new thing for identifying disease types from DNA/RNA profiles. According to an experiment analysing if the presence/absence

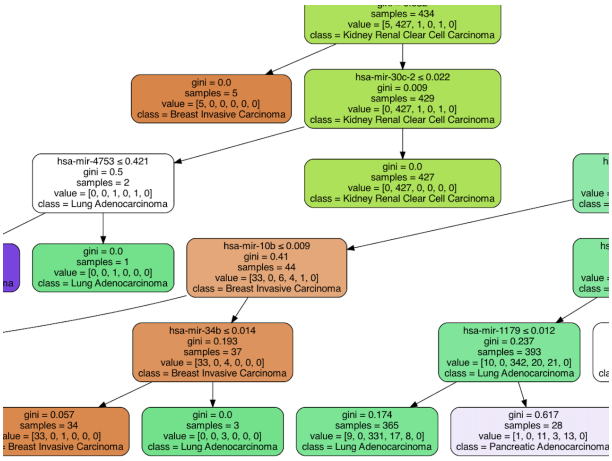

Figure 4: Partial Tree Structure

of certain miRNA could identify cancer type, researchers build their model using svm and found that the mean accuracy for identifying different cancer types is 91%.(Telonis et al. 2017) The accuracy of our model is 97%, which is higher than 91%. This could mean that the difficulty of identifying cancers using our data is easier, or we have found a better way to build model to identify cancers. Since the experiment was using a different svm model and different data set, further experiments and discussions are needed.

Decision Tree has also been used in similar problems. Researchers has tried to determine if an existing classification algorithm would be effective in differentiating ovarian cancer from benign diseases and healthy controls.(Vlahou et al. 2003) The result shows that a decision tree, using five protein peaks, resulted in an accuracy of 81.5% in the cross-validation analysis and 80% in a blinded set of samples in differentiating the ovarian cancer from the control groups. Again, the same conclusion for the svm model could be drawn here: our result is better, but the data and specific method are different. We can compare the effectiveness between different models when they are using the same data and solving the same problem.

According to our result for svm model and decision tree, svm model has a better accuracy than decision tree, which means that svm model might be a better way to identify cancer types using this data. However, this is not to say that we have to fully abandon decision tree. The fact that these two models are not 100% correct to identify cancers indicates that they have certain defects. It is unknown that whether decision tree algorithm can give some insights for improving svm model, vice versa.

6 Ethics

In this section we discuss some ethical concerns that can be found if a model like this is used in a real world medical application.

A major ethical concern is the accuracy of the each model. Even though both models preformed ”well”, one must look at the importance of accuracy when it comes to something as dire as predicting cancer type. The svm Model has an accuracy score of .97 and the Decision Tree Model had an accuracy of .938. Even though an error rate of 3% and 6.2% do not sound too bad, but when looking at the amount of wrong diagnoses and deaths that could be a result of the model, then it can look not as ideal.

With the current error rates there is a 3% chance and 6.2% chance that the diagnosis will be wrong. When thinking about life and death those are high stakes to leave up to a machine. A better practice may be to use the models in conjunction with human intuition in order to get the best diagnosis possible.

Another ethical concern that can arise from these models is on the data itself. We do not know what the demographics of the data are, so different people could possibly have different markers in their blood due to environments. Something that may be a marker for cancer may not be cancer in different blood types or different environments; however, we are not too sure about this because we are not biologists or chemists.

7 Contributions

K.M. created the python program that consolidated all the data into one file. K.M wrote the Ethics section. K.M. wrote the Abstract section. E.X. and K.M. contributed to the Data Analysis section. E.X. and K.M. contributed to the Experiments section. E.X. and K.M. contributed to the Results section. K.M. wrote the Introduction section. K.M. wrote the Decision Tree Classifier model python program and created the tree image. E.X. wrote the svm model python program and created the relevant tables and images. E.X. contributed to the Conclusion section.

References

Telonis, A. G.; Magee, R.; Loher, P.; Chervoneva, I.; Londin, E.; and Rigoutsos, I. 2017. Knowledge about the presence or absence of mirna isoforms (isomirs) can successfully discriminate amongst 32 tcga cancer types. *Nucleic acids research* 45(6):2973–2985.

Vlahou, A.; Schorge, J. O.; Gregory, B. W.; and Coleman, R. L. 2003. Diagnosis of ovarian cancer using decision tree classification of mass spectral data. *Journal of Biomedicine and Biotechnology* 2003(5):308–314.