HIV Viral Load Prediction Model

# Introduction

The human immunodeficiency virus (HIV) attacks cells used to fight off infections. Once introduced into the body, HIV can rapidly replicate itself and is known to adapt after treatment has been prescribed. Understanding the genetics of HIV and finding genetic markers that can change the severity of infection can help predict the response a patient will have when treatment is introduced. Using classification and logistic regression I will attempt to create models that will predict the patient’s response (whether they will see an improvement in their viral load and the white blood cells count) before starting treatment for HIV.

# Literature Review

In “Exploiting HIV-1 protease and reverse transcriptase cross-resistance information for improved drug resistance prediction by means of multi-label classification”[[1]](#footnote-1) the authors performed two types of mutli labeled classification on the protien sequences of the protease and reverse transcriptase. The data that was taken werer for patients that were perscribed protease inhibitors and non-cuvleaside reverse transcriptase inhibitors. These two classes of drugs are used to help slow down the spread and replication of HIV. By using multi label classification to build their model, they were able to improve the overall prediction of not only the specific set of drug classes that was in their training dataset, but for all different types of medication prescribed to HIV postive patients.

“Accurate prediction of HIV-1 drug response from the reverse transcriptase and protease amino acid sequences using sparse models created by convex optimization”[[2]](#footnote-2) the authors performed two regression techniques to linear model and a non linear model. Again the reverse transcriptase and the protease sequences were taken to build their models. The patients in the dataset that were used were perscribed protease inhibitors and non-cuvleaside reverse transcriptase inhibitors for their treatments. Once the authors had built their models, they found their models were more accurate then drug responses of patients better than other models that were built in previous studies.

“An Analysis on the Prediction of HIV Progression”[[3]](#footnote-3), the authors are using the same Kaggle dataset as I am, and attempt to build the model using different methods to which method would yield the best accuracy. With each model that was built different attributes from the dataset where included or excluded to try and improve the accuracy. There were seven different methods built altogether, with the lowest accuracy of 41%, highest accuracy of 62%, and a mean accuracy of 51%. From their experiments and building the different models the authors concluded that

# Dataset

In April of 2010 Kaggle provided a dataset of 1000 patients who where recently diagnosed with HIV and have not begun any medical treatment for it. The dataset contains the following attributes for a patient record: response status, protease nucleotide sequence, reverse transcriptase nucleotide sequence, viral load at the beginning of therapy, and CD4 count at the beginning of therapy. Response status will contain a “1” if the patient improved with their viral load, and “0” otherwise. The reverse transcriptase and protease are enzymes that are used in the replication of the virus, in which we are provided their genetic sequences. The reverse transcriptase is responsible for copy the HIV gnome within the infected cell. The HIV gnome is translated in the infected cell as one long string of amino acids. The protease protein cuts this string into functional units required by for the HIV life cycle. The CD4 count is an estimated number of white blood cells in 1 mL of blood, while the viral load is the number of viral particles in the same sample of blood.

After reviewing the data the following results have been discovered about the data set given:

* There are 794 had a response code of 0 and 206 had a response code of 1
* Lengths of protease string range from 0 to 297 characters long, with mean of 272.64 and median of 297
* The reverse transcriptase attribute has string length between 579 to 1482 characters, with mean of 951.96, and a median of 903
* The viral load count has a value range from 2.7 to 6, with an average of 4.3 and a median of 4.3
* The cd4 count attribute have a value range from 0 (which means some is missing) to 1589, with an average value is 279.64 and median of 249

# Approach 1 - Classification with Decision Tree

## Step 1: Read in Data File

Read in comma separated file from was downloaded from Kaggle and save the data into a R data frame.

## Step 2: Calculate Number of NGrams

Using R, calculate the maximum number of NGrams of length three that will be created from the reverse transcriptase attribute in the data frame. Sample code is taken from Wikipedia page “K-mer”[[4]](#footnote-4)

## Step 3: Create New Data Frame

Create a new data frame with following attributes viral load and cd4 counts, and split the reverse transcriptase attribute into maximum number of NGrams. If the reverse transcriptase is too short for to populate all of NGrams, then the remaining NGrams with empty string. If cd4 attribute is equal to zero, then update that value with the mean value. Add response code for each record, if the original response code was 0 set new response code to be “no”, else response code will be “yes”.

## Step 4: Export New Data Frame to CSV

Export the new data frame into a comma separated value file that will then be read into Weka program.

## Step 5: Import CSV into Weka

Import comma separated value file into Weka. Use Weka program to J48 decision tree. Decision tree will give most optimal algorithm to predict if the patient will see an improvement when starting treatment.

# Approach 2 – Text Mining of Reverse Transcriptase

## Step 1: Read in Data File

Read in comma separated file from was downloaded from Kaggle and save the data into a R data frame.

## Step 2: Create Corpus of NGrams

Take reverse transcriptase and split the attribute for each record into Ngrams of length three. Combine all the NGrams into one attribute separated by a space for each record. Create new data frame of corpus.

## Step 3: Convert Corpus into Term Matrix

Clean up corpus of any extra white spaces. From corpus, create term matrix of the all the Ngrams that have appeared at least ten times. Combined the term matrix with the response code from the original data frame that was read in.

## Step 4: Perform Logistic Regression

From term matrix perform logistic regression using the RWeka package, and print out the results performance with this model.

# Approach 3 – Logistic Regression of CD4 and Viral Load

## Step 1: Read in Data File

Read in comma separated file from was downloaded from Kaggle and save the data into a R data frame.

## Step 2: Create Formula

Create a formula that will be used in the logistic regression that will predict patient response to HIV treatment. Formula will use only the cd4 and the viral load count attributes to predict the response code.

## Step 3: Split Data

Split the data that was read in into training and testing. Use 80% of the data for training and the remaining 20% for testing the formula.

## Split 4: Perform Logistic Regression

Perform logistic regression on the training data frame. Use results from the training on the test data and print out the accuracy of the performance from this model.

# Approach 4 – Text Mining and Logistic Regression Combined

## Step 1: Read in Data File

Read in comma separated file from was downloaded from Kaggle and save the data into a R data frame.

## Step 2: Create Corpus of NGrams

Take reverse transcriptase and split the attribute for each record into Ngrams of length three. Combine all the NGrams into one attribute separated by a space for each record. Create new data frame of corpus.

## Step 3: Convert Corpus into Term Matrix

Clean up corpus of any extra white spaces. From corpus, create term matrix of the all the Ngrams that have appeared at least ten times. Combined the term matrix with the response code, cd4 and viral load counts from the original data frame that was read in.

## Step 4: Perform Logistic Regression

From term matrix perform logistic regression using the RWeka package, and print out the results performance with this model.

# Results

## Approach 1

When first attempting to build a model using this approach, I noticed that the lengths of the reverse transcriptase attribute varied in each record, the protease had missing values for 8% of the population that was being tested, and the cd4 had a few zero values. From this observation I omitted the protease attribute, and used the minimum number of Ngrams that would be created when I split the reverse transcriptase attribute up. I tested this method with Ngrams lengths of 2, 3, 5 and 7, with length 3 yielding the highest accuracy rate.

On the second and third attempts at this model I decided to use the average value for the cd4 count for the records that had a zero value, add the protease attribute back as Ngram as the same length, and use the maximum number of Ngrams that can be created. When using the average cd4 value and maximum number of Ngrams, and after using the same length numbers as before, I found that a Ngram of length 3 had the best results. Table 1 shows the accuracy, precision, and recall values for this model. What is more interesting, was that when the protease variable was added back in and the decision tree was created, the variable was not used by Weka and those having no affect on the accuracy rate of the model.

Figure 1 shows a simplified version of the J48 decision tree that was created in Weka where the Ngram length was three and using the maximum number of Ngrams from the reverse transcriptase attribute. The following definitions are noted for the decision tree:

* rtPos is the position reverse transcriptase split into Ngrams
* NA is the reverse transcriptase value if at the rtPos is blank
* VL stands for viral load count
* Yes and No values represent if the patient will see an improvement after treatment has started.

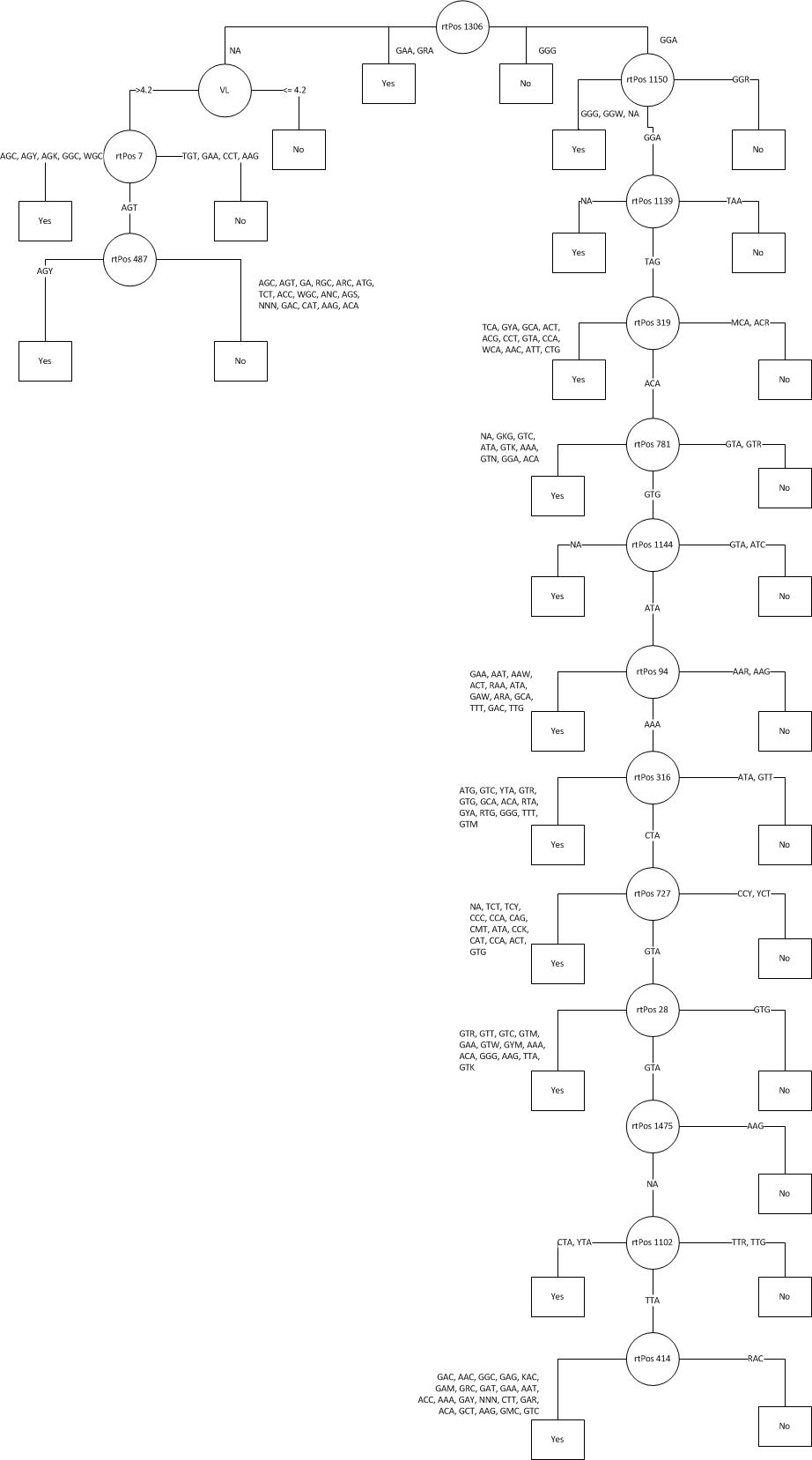


Figure Decision Tree of NGram of Length 3

## Approach 2

This approach at predicating a patient’s response code only used the reverse transcriptase attribute that would again be split into Ngram lengths of 3, 5, and 7. First attempt at text mining the reverse transcriptase attribute, I took a look at all the Ngrams that were created and see if the frequency of all these words had response code. This method only yielded an accuracy rate of 61%.

The second approach I took, was to take the Ngrams that appeared the most frequent. At first I started out with a minimum frequency of 5 and noticed that this did perform better than my first method. Figure 2 shows frequency of the top 5 Ngrams that were comprised of the reverse transcriptase attribute.



Figure 2

I began to play with the minimum frequency until I found that the words that had a minimum frequency of 10 gave the highest accuracy rate, which can be seen in Table 1.

After running both these approaches using Ngrams of length 3, it was attempted to try and use lengths 5 and 7. Unfortunately due to the capabilities of the machine that I was using, I was getting an error message informing me that I had run out of virtual memory. Because of this set back, I am not able to fully determine that if increasing the lengths of Ngrams would have a positive or negative results on the overall accuracy returned by the model.

## Approach 3

The third approach was to use logistic regression only using the numeric values from the cd4 and viral load counts. Instead of keeping the cd4 values of zero, I again decided to use the average values like I in the first approach. I found this method to be very simple and straightforward. The data was split into 80% training and 20% for testing. After splitting the data logistic regression was applied to the training data the following formula was the result:

*Resp = VL\*1.4851893 + CD4\*0.0004998 – 8.0752746*

The formula was then used on the remaining data to predict its accuracy which can be found in Table 1. Because the data was randomly split into test and training data, the accuracy would very each time, the accuracy that is shown in Table 1 is the rate from the last run.

## Approach 4

The final approach I took was a combination of the second and third approach using logistic regression to help predict a patients response code. I used the same minimum frequency limit of 10 of Ngrams of length 3, and added the viral load and cd4 counts to test if they had any affect on the accuracy. Since this was a combination of two approaches, not much extra work was needed to be complete this model. The only difference was that the data was not split into a training and test data set. I was expecting the resulting accuracy to be somewhere between the Approach 2 and 3, possibly in the 70 percentile. Table 1 shows that accuracy was actually not far Approach 2, and not an improvement in overall predicition.

## Overall Accuracy

The following table shows the accuracy, precision, and recall rates for the top performing methods in each of the approaches.

Table

|  |  |  |  |
| --- | --- | --- | --- |
| Approach Model | Accuracy Rate | Precision | Recall |
| Approach 1 | 81.1 % | 92.6 % | 94.9 % |
| Approach 2 | 64.4 % | 89.2 % | 72.4 % |
| Approach 3 | 80.5 % | 96.6 % | 84.6 % |
| Approach 4 | 64.9 % | 90.0 % | 73.5 % |

# Conclusions

Through the different approaches that were developed and used for predicating a patient’s response outcome to treatment, classification and the development of a decision tree provided the best results. Though the logistic regression model that was built did have an accuracy that was close to that of the decision tree, it did have a lower recall rate. It also appears that having a program or a machine-learning tool such as Weka helps improve the model building. This take away from any human inferring and possible misunderstanding of the data. I feel that if the protease attribute was not omitted for most of these approaches, or if I was able to predict the protease value if it was blank depending on the reverse transcriptase, the models that were developed would have a higher accuracy rate and would be off with the prediction.

All code can be found at the following location: https://github.com/c2rayman/cmke136/tree/final/final

1. Exploiting HIV-1 protease and reverse transcriptase cross-resistance information for improved drug resistance prediction by means of multi-label classification, <http://biodatamining.biomedcentral.com/articles/10.1186/s13040-016-0089-1> [↑](#footnote-ref-1)
2. <http://bioinformatics.oxfordjournals.org/content/22/5/541.long> [↑](#footnote-ref-2)
3. <http://www.ranjaykrishna.com/pdf/HIV_progression_prediction.pdf> [↑](#footnote-ref-3)
4. https://en.wikipedia.org/wiki/K-mer [↑](#footnote-ref-4)