Nawar Mikha
CSC 5800 Course Project

# Overview

The data mining and analysis project I chose to work on is the promoter gene sequence data set. The background of this dataset is that it was used to evaluate possible contact regions as well as confirmations of the Escherichia coli bacteria or E. coli. Since contracting this illness can be life threatening in some cases, the motivation for pulling valuable information from the data set is clear. Aside from E. coli there is a wealth of information that can be derived from genetic data in general. The field of mining and analyzing data sets similar to the one used in this project has now grown into the pervasive field of bioinformatic.

Gene sequences are comprised of chains of four bases adenine, cytosine, guanine, and thymine or ACGT as it is commonly referred to[1]. The gene sequence is the set of bonds that hold together two strands of DNA; the entire structure forms a helix shape. The two strands are complementary and can be used as templates for each other. The bonds are made up of the bases, where adenine and thymine can only bond together, and cytosine and guanine can only bond together. A gene is the specific sequence within a particular region of the DNA structure. Genetic information is essentially a biological blueprint for the cells of a living organism's body.

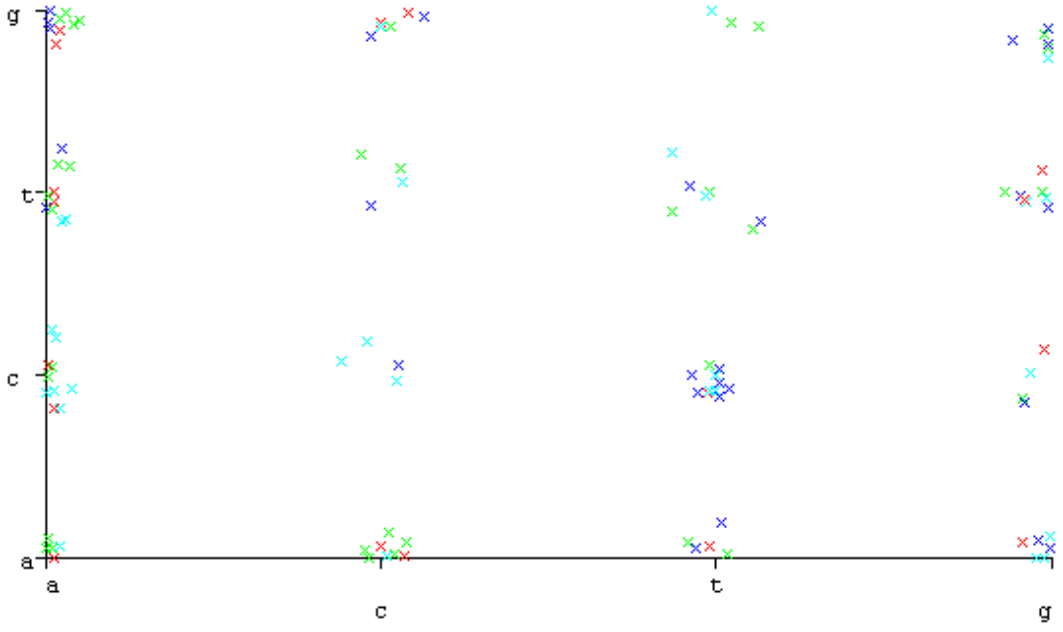
If we dig further into a particular gene (sub region of the DNA) there is a sub region within that gene that is called the promoter. The promoter is what initiates the transcription of the gene that it is located within. The task here is to predict what values at specific points in the sequence, as well as what patterns within the sequence elude to possible cases of an E. coli infection.

# Data Overview

The data set contains a total of one hundred and six instances, fifty-nine attributes, and two distinct class labels. Each instance is a different gene sequence with its own promoter sub region. The second attribute is the name of each specific gene (S10, UVRB_P2,1169, RECA, 521 etc…). The first attribute is the class name, denoted by a "+" or a "-". The plus sign represents a class that has a promoter for the E. coli bacteria, the minus sign represents a non-promoter instance. The remaining fifty-seven attributes are the gene sequence with possible values of A, C, G, or T. Each index within the sequence has a unique name associated with it here, the range is from P-50 up to P7 (negative up to positive). This would make the sequence fifty-eight units long rather than the actual fifty-seven. After doing some secondary research I wasn't able to find a reference for the index names anywhere; the only rational conclusion I was left with was not to include the P0 label. Below is a graphic depicting the data in weka after it has been converted to .arff format.

| No. | 1: class Nominal | 2: instanceName Nominal | 3: p-50 Nominal | 4: p-49 Nominal | 5: p-48 Nominal | 6: p-47 Nominal | 7: p-46 Nominal | 8: p-45 Nominal | 9: p-44 Nominal | 10: p-43 Nominal | 11: p-42 Nominal | 12: p-41 Nominal | 13: p-40 Nominal |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | + | S10 | t | a | c | t | a | g | c | a | a | t | a |
| 2 | + | AMPC | t | g | c | t | a | t | c | c | t | g | a |
| 3 | + | AROH | g | t | a | c | t | a | g | a | g | a | a |
| 4 | + | DEOP2 | a | a | t | t | g | t | g | a | t | g | t |
| 5 | + | LEU1_TRNA | t | c | g | a | t | a | a | t | t | a | a |
| 6 | + | MALEFG | a | g | g | g | g | c | a | a | g | g | a |
| 7 | + | MALK | c | a | g | g | g | g | g | t | g | g | a |
| 8 | + | RECA | t | t | t | c | t | a | c | a | a | a | a |
| 9 | + | RPOB | c | g | a | c | t | t | a | a | t | a | t |
| 10 | + | RRNAB_P1 | t | t | t | t | a | a | a | t | t | t | c |
| 11 | + | RRNAB_P2 | g | c | a | a | a | a | a | t | a | a | a |
| 12 | + | RRNDEX_P2 | c | c | t | g | a | a | a | t | t | c | a |
| 13 | + | RRND_P1 | g | a | t | c | a | a | a | a | a | a | a |
| 14 | + | RRNE_P1 | c | t | g | c | a | a | t | t | t | t | t |
| 15 | + | RRNG_P1 | t | t | t | a | t | a | t | t | t | t | t |

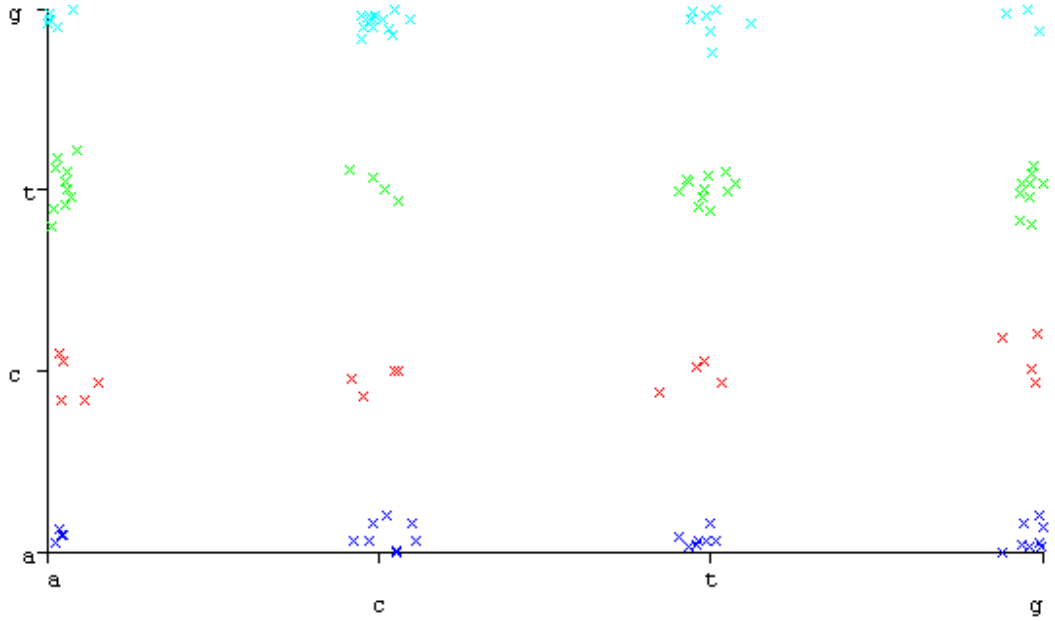
Since all of our attributes, excluding class label and instance name are discrete and only contain one of four possible values, visualizing this data before running it through any algorithms produces fairly boring results, even when jitter is incorporated. No real trends or correlations are visible. Almost every feature plotted against any other feature produces essentially the same graph.




The one exception is P7, it seems as though the values within this attribute appears to dominate every other attribute that they are plotted against.

**(a c t g)** **X-axis = P-49** **Y-axis = P6**

I am not entirely certain why this is the case, but as far as I can tell after reviewing the data visualization graphs this is the only attribute where this is occurring. It is something to keep in mind during the analysis phase.




**(a c t g)** **X-axis = P6 Y-axis = P7**

# Data Preprocesing

This particular type of data and more specifically this data set is not very malleable in terms of preprocessing. As far as cleaning is concerned, there are no missing values; and since the vast majority of the attributes can only have one of four distinct discrete values, noise is essentially non existent here. Inconsistencies are also not present; one example of this would be if any of the gene were something other than A, C, G, or T, which there are none of. Since every attribute in the data set is nominal there is also no need to discretize anything. The same applies for data transformations. We have no numerical values so normalizing is irrelevant with this data. Aggregating and generalizing like-wise have no purpose as we do not have missing values or inconsistency. Lastly, there is no need to integrate the data as there are no conflicting representations.

The one possible category of data preprocessing that may apply to our data set is that of data reduction. Though there are only one hundred and six instances, we may need to reduce the number of attributes. This might sound counterintuitive as one might think that this would completely distort the information encoded within the sequence, but as the documentation states certain patterns contiguous or not are what tell us the information we desire to gain, not the entire gene sequence. That being said, it may prove valuable to cut down the number of attributes in the sequence to see if we can improve our mining and analysis tasks. This will be explored in the following algorithms section. According to the documentation, two areas within the sequence that are of particular interest are around P-35 and P-10. In these regions a chain rather than a

grouping of dispersed indexes within the sequence are signifiers that we may have a positive instance.

```
% The following rules describe the compositions of possible contact regions.
minus_35 :- p-37=c, p-36=t, p-35=t, p-34=g, p-33=a, p-32=c.
minus_35 :-         p-36=t, p-35=t, p-34=g,         p-32=c, p-31=a.
minus_35 :-         p-36=t, p-35=t, p-34=g, p-33=a, p-32=c, p-31=a.
minus_35 :-         p-36=t, p-35=t, p-34=g, p-33=a, p-32=c.

minus_10 :- p-14 t, p-13 a, p-12=t, p-11=a, p-10=a, p-9=t.
minus_10 :-         p-13 t, p-12=a,         p-10=a,        p-8=t.
minus_10 :-         p-13 t, p-12=a, p-11=t, p-10=a, p-9=a, p-8=t.
minus_10 :-                 p-12=t, p-11=a,                       p-7=t.
```

Within the documentation there is also information on rules that depicted a combination of non-contiguous gene indexes that produce confirmations of the bacteria we are investigating.

```
% The following rules describe sequence characteristics that produce
% acceptable conformations.
conformation :- p-47=c, p-46=a, p-45=a, p-43=t, p-42=t, p-40=a, p-39=c,
                p-22=g, p-18=t, p-16=c, p-8=g,  p-7=c,  p-6=g,  p-5=c,
                p-4=c,  p-2=c,  p-1=c.
conformation :- p-45=a, p-44=a, p-41=a.
conformation :- p-49=a, p-44=t, p-27=t, p-22=a, p-18=t, p-16=t, p-15=g,
                p-1=a.
conformation :- p-45=a, p-41=a, p-28=t, p-27=t, p-23=t, p-21=a, p-20=a,
                p-17=t, p-15=t, p-4=t.
```

# Algorithms

Before I begin to try and manipulate (preprocess) the instances to improve my results, I first need a baseline for the algorithms that I will be running on this data. I will run the data through five different classifiers, two clustering methods, and the a priori association rule algorithm. The first table below illustrates the clustering models that the data was put through. I thought about using hierarchical clustering at first, but that really isn't applicable to this data. The second table is for the classification algorithms, where I decided to use 10-fold cross validation on all five of the classifiers. The third includes some association rules from the A priori algorithm.

| | **K-Means** | **Density Based Clustering** |
|---|---|---|
| **Error rate** | 0.415 | 0.415 |
| **Confusion Matrix** | 0 1<br>20 33 \| + 0: 49<br>29 24 \| - 1: 57 | 0 1<br>21 32 \| + 0: 51<br>30 23 \| - 1: 55 |

| * top row a & b is what the instance is classified as (conf mat) | Naïve Bayes | Neural Network | Nearest Neighbor | Support Vector Machine | boosting |
|---|---|---|---|---|---|
| **Error rate** | 0.094 | 0.066 | 0.179 | 0.066 | 0.142 |
| **TP rate** | 0.906 | 0.934 | 0.821 | 0.934 | 0.858 |
| **FP rate** | 0.094 | 0.066 | 0.179 | 0.066 | 0.142 |
| **Precision** | 0.906 | 0.934 | 0.830 | 0.934 | 0.859 |
| **Recall** | 0.906 | 0.934 | 0.821 | 0.934 | 0.858 |
| **F-measure** | 0.906 | 0.934 | 0.819 | 0.934 | 0.858 |
| **ROC Area** | 0.967 | 0.980 | 0.886 | 0.934 | 0.951 |
| **Confusion Matrix** | a b<br>47 6 \| a = +<br>4 49 \| b = - | a b<br>49 4 \| a = +<br>3 50 \| b = - | a b<br>48 15 \| a = +<br>14 39 \| b = - | a b<br>50 3 \| a = +<br>4 49 \| b = - | a b<br>47 7 \| a = +<br>8 45 \| b = - |

| A Priori | Association Rule | Confidence | Lift |
|---|---|---|---|
| **Rule 1** | p-45=a p-35=t 30 ==> class=+ 30 | 1.00 | 2.00 |
| **Rule 2** | p-35=t p-34=g 35 ==> class=+ 34 | 0.97 | 1.94 |
| **Rule 3** | p-45=a p-36=t 28 ==> class=+ 27 | 0.96 | 1.93 |
| **Rule 4** | p-35=t p-10=a 27 ==> class=+ 26 | 0.96 | 1.93 |
| **Rule 5** | p-36=t p-35=t p-34=g 27 ==> class=+ 26 | 0.96 | 1.93 |
| **Rule 6** | p-36=t p-34=g 34 ==> class=+ 32 | 0.94 | 1.88 |
| **Rule 7** | p-36=t p-10=a 30 ==> class=+ 28 | 0.93 | 1.87 |
| **Rule 18** | p-35=t p-11=a 29 ==> class=+ 27 | 0.93 | 1.86 |

```
Ranked attributes:
 1           2 instanceName
 0.3473     17 p-36
 0.32044    19 p-34
 0.28252    18 p-35
 0.23516    41 p-12
 0.17894    20 p-33
 0.14797     8 p-45
 0.11913    43 p-10
 0.11418    22 p-31
 0.10865    51 p-2
 0.09973    42 p-11
 0.0838     12 p-41
 0.07871    40 p-13
 0.07692    10 p-43
 0.07666    21 p-32
```

Next, I decided to run a few attribute selection algorithms, to try and assist in the process of reducing the data set's dimensionality and see if they correlate with the information provided in the documentation. First, I implemented principle component analysis, but I was unable to obtain any meaningful results with it. After that I tried the information gain attribute evaluation method, to determine which features contributed the most to the reduction of entropy. Using the full training set, the following results were produced in the chart to the left. Instance name being irrelevant.

Nawar Mikha
CSC 5800 Course Project

## ----------Post Dimensionality Reduction----------

Now I wanted to try and improve the results of my classification and clustering tasks. The most obvious way to do this, is to go reduce the number of attributes in the data set based on the findings from the information gain algorithm. Trying to manipulate or interrupt the data for the confirmation rules would simply be too difficult of a process without writing a lot of code, so I only tried to optimize for the contact region codes which were sequential and not sporadic like the confirmation rules. The confirmation rules also didn't show up in my attribute selection analysis, or in any other method that I implemented the data into. I decided to eliminate everything besides the class, instance name, and the twelve attributes with the highest information gain, roughly twenty percent of the fifty-seven sequence attributes (P-36 to p-13 in the chart above). Below I have listed the same tables that I included above except now they include the values of the reduced data set implementation. To the left is a table of the clustering algorithms, and below is the one for the classification task.

| | **K-Means** | **Density Based Clustering** |
|---|---|---|
| **Error rate** | 0.123 | 0.113 |
| **Confusion Matrix** | 0 1<br>1 52 \| + 0: 42<br>41 12 \| - 1: 64 | 0 1<br>1 52 \| + 0: 43<br>42 11 \| - 1: 63 |

| * top row a & b is what the instance is classified as (conf mat) | **Naïve Bayes** | **Neural Network** | **Nearest Neighbor** | **Support Vector Machine** | **boosting** |
|---|---|---|---|---|---|
| **Error rate** | 0.075 | 0.057 | 0.123 | 0.057 | 0.142 |
| **TP rate** | 0..925 | 0.943 | 0.877 | 0.943 | 0.858 |
| **FP rate** | 0.075 | 0.057 | 0.123 | 0.057 | 0.142 |
| **Precision** | 0.927 | 0.943 | 0.902 | 0.943 | 0.860 |
| **Recall** | 0.925 | 0.943 | 0.877 | 0.943 | 0.858 |
| **F-measure** | 0.924 | 0.943 | 0.875 | 0.943 | 0.858 |
| **ROC Area** | 0.988 | 0.992 | 0.965 | 0.943 | 0.958 |
| **Confusion Matrix** | a b<br>47 6 \| a = +<br>2 51 \| b = - | a b<br>50 3 \| a = +<br>3 50 \| b = - | a b<br>53 0 \| a = +<br>13 40 \| b = - | a b<br>50 3 \| a = +<br>3 50 \| b = - | a b<br>47 6 \| a = +<br>9 44 \| b = - |

# Analysis and Results

Overall, I think my methodology for mining this data was fairly sound, and I am pleased with the results as they are improvements on the error rates, without having to overfit the model. I believe that the association rules, and the attribute selection function correlated well with the information that they revealed about the data set, and this information was a good resource to use as this data has no numeric attributes.

The clustering algorithms improved significantly. K-means dropped from 41.5% to 12.3%, the classes were more evenly distributed (there are fifty-three of each) with the original data in k-mean, but they were massively miss labeled. Density based clustering or DBS dropped from to 41.5% to 11.3%, which was an even bigger improvement than k-means. The same is true for DBS regarding the class distributions.

All of the classification algorithms improved in terms of error rate, except for adaBoost which had no change. Nearest neighbor improved the most with roughly a 5% decrease. Though it also had the highest error rate to start with, so it had the most room for improvement. Not surprisingly the neural network classification algorithm performed very well with a low of 5.7%, which was at 6.6% with the original data (1 more correctly labeled class). The support vector machine also had almost the exact same statistics as the neural network.

# Conclusion

For anyone who is interested in mining genetic sequence data I would recommend first attempting to reduce the dimensionality of the data. This will be of particular salience if you are dealing with a larger data set than the one that I had, as it only contained one hundred and six instances. It may be computationally infeasible to try to explore the data at all without reducing the feature set first, with larger data sets. One possible solution to this is to first consult a domain expert before attempting to extract any information. The way that I would recommend reducing the dimensionality, given the inherent nature of the data is to discover association rules. The specific algorithm that I would suggest for the association rule analysis or ARA is foils information gain, which weka has built into it.

What we are looking for in our data is a pattern or sequence (the difference being contiguity) that signifies the presence of a promoter for the E. coli virus. Knowing this will assist in providing treatment to people who are suffering from this condition and being able to detect it as early as possible ensures the highest chance of mitigating any harm to the patient. Finding these associations allows up to alter the data to improve on the supervised learning methods, which allows analysts to provide a more accurate diagnosis. I was not able to come up with any sort of methodology for extracting the confirmation rules for the bacteria, but I did still improve on the learning simply with the

contact region rules that were obtained through the A priori algorithm. I tried a few different implementations of the supervised learning tasks with slightly varying sets of attributes, as the association rule mining and feature extraction algorithms while largely in sync but did somewhat deviate from each other. This particular version of the reduced data set work the best from my findings.

**References:**

1. https://www.genome.gov/glossary/index.cfm?id=1
2. https://archive.ics.uci.edu/ml/machine-learning-databases/molecular-biology/promoter-gene-sequences/promoters.theory