Nathan Horak
6/18/2022
Term Project
BU MET CS 699

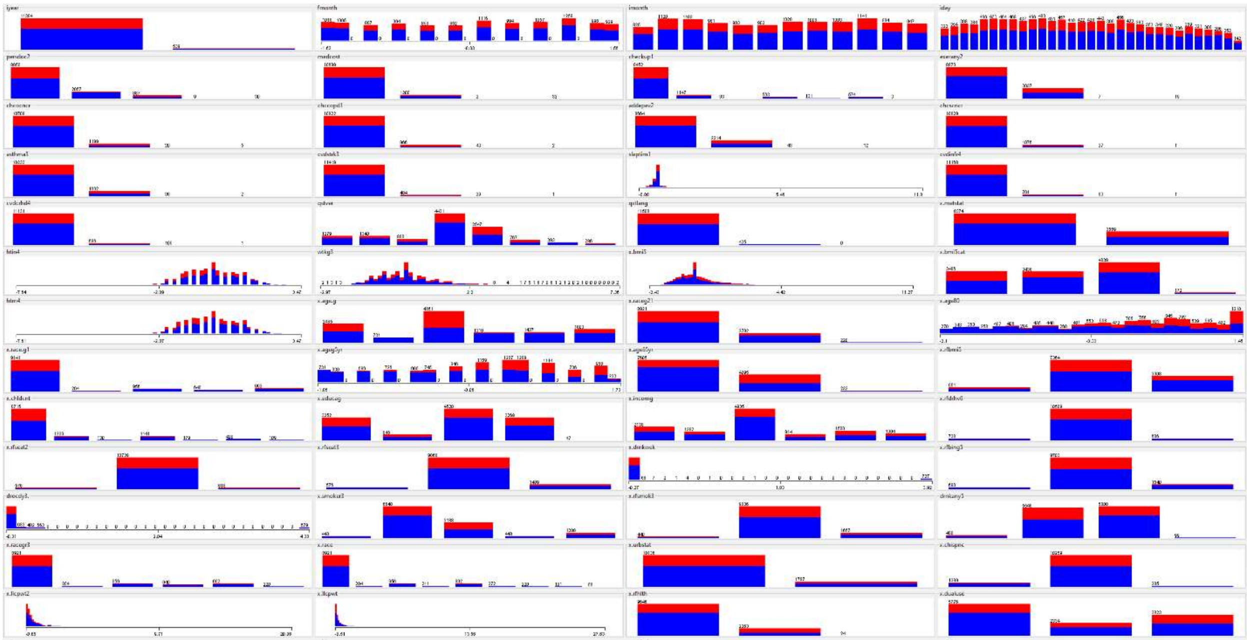
I used Weka to preprocess the data, create training/testing splits, and run classification algorithms. Relevant screenshots included. Link to data: https://www.cdc.gov/brfss/annual_data/annual_2018.html

Preprocessing-

The first step I undertook with regards to the dataset was preprocessing. Per module notes, preprocessing consists of five steps- data cleaning, data integration, data transformation, data reduction, and data discretization[1]. I will briefly go over each of these steps in relation to the preprocessing work I did.

In the first step of Data Cleaning, I replaced missing values. I did so by replacing missing values within the training dataset with mean and mode for the respective attribute. I chose this over removing rows containing missing data or replacing missing values with a constant as I wanted to include as much of the data as possible. Both other approaches would be acceptable, but I chose to go about it in this way. There were no actions I needed to take with regards to Data Integration, as the data was already consolidated in a single file. Concerning Data Transformation, I knew one of the algorithms I would want to use for classification was kNN. Given that this algorithm compares distances between points, it made sense to normalize the data. Data normalization is a common procedure in preprocessing but is essentially required when one will evaluate the data with distance metrics. For Data Reduction, I took no steps here. I knew that I was going to utilize attribute selection at a later point in time and thus determined that additional up-front reduction was not necessary. Lastly, for Data Discretization the data was already in a suitable format for analysis and as such, no action was necessary here.

A quick point related to noise/binning: I briefly experimented with utilizing binning to smooth over some of the data. However, I was unable to get much of a meaningful improvement. Upon examining the data, there are some outliers but there was nothing out of the ordinary. I came to this conclusion after attempting to incorporate binning as well as by examining breakdown of the attributes. See below:



In addition to the preprocessing done mentioned above, I also randomized the data before I created a train/test split. While Weka will do this automatically, it is important that the training/test datasets are drawn randomly.

Attribute Selection Methods-

When running a model with as many attributes as this one (108!), there are discrepancies between the effectiveness of incorporating a single attribute compared to another one (or concerning its absence). Given this, it makes sense to trim down the number of attributes if possible, to only run the algorithm with the most meaningful attributes. Per the module notes, attributes that do not add the accuracy of a model can be removed[1]. Excess attributes can unnecessarily increase training time and/or lower classification accuracy. I picked five different class selection methods and I aimed to pick a wide assortment of types to get unique recommended attributes.

The first attribute evaluator/search method pair I used was CorrelationAttributeEval + Ranker. According to Weka, the attribute evaluator works by measuring Pearson's correlation coefficient between it and the predicted class[2]. This seemed like a good attribute selector to use given that it incorporates Pearson's correlation coefficient- one of the most cited metrics in statistics. The search method used was Ranker, which ranks attributes according to a measure (in this case, Pearson's correlation coefficient). When I received the results, I had to independently decide how many attributes to include. Considering that the dataset originally had 107 attributes, I wanted to cut the number significantly while still leaving room for differing attribute selection methods to pick differing attributes. I settled on 20. I would use this same number going forward for all but the last attribute evaluator/search method pair. The top 20 attributes corresponded with indices 64, 22, 66, 2, 67, 87, 97, 20, 102, 46, 95, 31, 29, 25, 62, 6, 24, 27, 53, and 69. Names: employl, children, pneuvac4, diffwalk, diffdres, diffalon, rmvteth4, diabete3, physhlth, chccopdl, cvdcrhd4, x.age.g, x.age80, x.ageg5yr, x.age65yr, x.chldcnt, x.rfhlth, x.phys14d, x.hcvu651, and x.exteth3.

```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 108 havarth3):
        Correlation Ranking Filter
Ranked attributes:
 0.3663     64 x.age80
 0.34889    22 diffwalk
 0.34848    66 x.ageg5yr
 0.31775     2 employ1
 0.27431    67 x.age65yr
 0.23261    87 x.rfhlth
 0.22539    97 x.hcvu651
 0.21439    20 pneuvac4
 0.21368   102 x.exteth3
 0.20434    46 chccopdl
 0.18345    95 x.phys14d
 0.18281    31 physhlth
 0.17573    29 diabete3
 0.17271    25 diffalon
 0.16885    62 x.age.g
 0.16783     6 children
 0.16265    24 diffdres
 0.15687    27 rmvteth4
 0.15371    53 cvdcrhd4
 0.15242    69 x.chldcnt
 0.15185   104 x.michd
 0.15115    43 checkup1
 0.14602    13 deaf
 0.13933    98 x.totinda
 0.13932    44 exerany2
 0.13821    59 x.bmi5
 0.12912    26 smoke100
 0.12846    45 chcocncr
 0.12688    36 chckdny1
 0.12647    47 addepev2
 0.1243     41 persdoc2
 0.12129    16 flushot6
 0.11955    14 decide
 0.1172     48 chcscncr
 0.11601    10 sex1
 0.11263    52 cvdinfr4
 0.11197    50 cvdstrk3
 0.10937    49 asthma3
 0.10937   105 x.ltasthl
 0.1083     34 genhlth
 0.1077     84 x.chispnc
 0.10664    78 x.smoker3
 0.10585   103 x.asthmsl
 0.10498   106 x.casthml
 0.10456    57 htin4
 0.10437    21 alcday5
 0.10227    61 htm4
 0.10063    68 x.rfbmi5
 0.09693    86 x.llcpwt
 0.0968     88 x.dualuse
 0.09666    80 drnkany5
 0.09659    76 x.rfbing5
 0.0964     15 x.drnkdrv
 0.09316     8 blind
 0.09149    33 hlthpln1
 0.08348     9 renthom1
 0.08284    93 x.strwt
 0.0799     90 x.hispanc
 0.07845    63 x.raceg21
 0.07837    30 x.psu
```

The second attribute evaluator/search method pair I decided to use was InfoGainAttributeEval + Ranker. At this point, I knew I wanted to use at least two classification algorithms which used decision trees (J48 [Decision Tree] and RandomForest) and given that these algorithms work through attributes down through the top of the tree dependent on information gain, this seemed like a good evaluator to include. Per Weka, the evaluator works by evaluating the worth of an attribute, measuring the information gain with respect to the class. According to module notes, Info of a dataset D, Info(D) is the amount of knowledge needed to classify a tuple in D[1]. Information gain is simply the improvement in Info value from incorporating an attribute. One of the methods (and the main one) that decision trees use to gauge attributes is by comparing information gain across attributes. Just as above, I decided to use the top 20 attributes by information gain. These correspond to indices 64, 66, 62, 2, 22, 34, 67, 97, 31, 20, 95, 27, 87, 11, 102, 46, 69, 43, 6, and 29. Names: employl, children, marital, pneuvac4, diffwalk, rmvteth4, diabete3, physhlth, gnhlth, checkupl, chccopdl, x.age.g, x.age80, x.ageg5yr, x.age65yr, x.chldcnt, x.rfhlth, x.phys14d, x.hcvu651, and x.exteth3.

```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 108 havarth3):
        Information Gain Ranking Filter

Ranked attributes:
 0.11056061    64 x.age80
 0.10943741    66 x.ageg5yr
 0.10705785    62 x.age.g
 0.09497417     2 employl
 0.08384525    22 diffwalk
 0.06230612    34 genhlth
 0.05776225    67 x.age65yr
 0.05369539    97 x.hcvu651
 0.0532772     31 physhlth
 0.04960514    20 pneuvac4
 0.04852862    95 x.phys14d
 0.04475247    27 rmvteth4
 0.03745101    87 x.rfhlth
 0.03686287    11 marital
 0.03612048   102 x.exteth3
 0.02858777    46 chccopdl
 0.0261898     69 x.chldcnt
 0.02531043    43 checkupl
 0.02484826     6 children
 0.02420432    29 diabete3
 0.02353052    41 persdoc2
 0.02096672    54 qstver
 0.02028265    25 diffalon
 0.01808301    24 diffdres
 0.01661073    53 cvdcrhd4
 0.01630316     3 income2
 0.01560325   104 x.michd
 0.01525656    59 x.bmi5
 0.01492015    86 x.llcpwt
 0.01458789    13 deaf
 0.01436932    78 x.smoker3
 0.01389676    60 x.bmi5cat
 0.01378431    44 exerany2
 0.01375881    98 x.totinda
 0.01339996    71 x.incomg
 0.01281497    88 x.dualuse
 0.01223944    26 smoke100
 0.01173858    77 drocdy3.
 0.011655      47 addepev2
 0.01164035    82 x.race
 0.01146206    45 chcocncr
 0.01095419    36 chckdnyl
 0.01072602    16 flushot6
 0.01021096    14 decide
 0.00991437    10 sexl
 0.0097195     84 x.chispnc
 0.00948248    48 chcscncr
 0.00936112    89 x.imprace
 0.0093343     75 x.drnkwek
 0.00916422    68 x.rfbmi5
 0.00915904    93 x.strwt
 0.00887032    52 cvdinfr4
 0.00886591    76 x.rfbing5
 0.00882178   103 x.asthmsl
 0.00863993    21 alcday5
 0.00863581    49 asthma3
 0.00858883    50 cvdstrk3
 0.00847031   105 x.ltasthl
 0.0080264     80 drnkany5
```

The next pair I decided to use was SymmetricalUncertAttributeEval + Ranker. From Weka, this attribute evaluator/search method works by measuring the symmetrical uncertainty with respect to the class and then ranking them. The formula for Symmetrical Uncertainty is[2]:

2 * (H | Class) - H(Class | Attribute) / H(Class) + H(Attribute)

According to an abstract posted on sciencedirect.com[3], symmetrical uncertainty measures the relevance between feature and class label. The average normalized interaction gain between all features (including the predicted class) is calculated. Again, I decided to use the top 20 by rank and chose (by index) the attributes corresponding to 22, 64, 2, 62, 66, 67, 97, 87, 20, 31, 46, 95, 34, 102, 27, 24, 25, 29, 6, and 53. Names: employl, children, pneuvac4, diffwalk, diffdres, diffalon, rmvteth4, diabete3, physhlth, genhlth, chccopdl, cvdcrhd4, x.age.g, x.age80, x.ageg5yr, x.age65yr, x.rfhlth, x.phys14d, x.hcvu651, and x.exteth3.

```
=== Attribute Selection on all input data ===

Search Method:
	Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 108 havarth3):
	Symmetrical Uncertainty Ranking Filter

Ranked attributes:
 0.1056368    22 diffwalk
 0.0659474    64 x.age80
 0.0656272     2 employ1
 0.0649798    62 x.age.g
 0.0599647    66 x.ageg5yr
 0.0586527    67 x.age65yr
 0.048818     97 x.hcvu651
 0.0457244    87 x.rfhlth
 0.0448497    20 pneuvac4
 0.0445897    31 physhlth
 0.0420769    46 chccopdl
 0.0415452    95 x.phys14d
 0.0409477    34 genhlth
 0.0351759   102 x.exteth3
 0.0332137    27 rmvteth4
 0.0307201    24 diffdres
 0.0307077    25 diffalon
 0.0280744    29 diabete3
 0.027554      6 children
 0.0252204    53 cvdcrhd4
 0.0249933    11 marital
 0.0248461    43 checkup1
 0.0239389    41 persdoc2
 0.0230171   104 x.michd
 0.0228007    69 x.chldcnt
 0.0211613    13 deaf
 0.0185678    36 chckdny1
 0.0163481    45 chcocncr
 0.0157163    44 exerany2
 0.0157047    98 x.totinda
 0.014377     50 cvdstrk3
 0.0141011    47 addepev2
 0.0140947    14 decide
 0.0138995    48 chcscncr
 0.0137356    52 cvdinfr4
 0.0134616    59 x.bmi5
 0.0130849    86 x.llcpwt
 0.0125883    26 smoke100
 0.0122471    84 x.chispnc
 0.01217      54 qstver
 0.0113575    49 asthma3
 0.0112512    16 flushot6
 0.0111451   105 x.ltasth1
 0.0111208    78 x.smoker3
 0.010835     60 x.bmi5cat
 0.0108087   106 x.casthm1
 0.0106572   103 x.asthms1
 0.0106224    88 x.dualuse
 0.010268    100 x.prace1
 0.0102154     3 income2
 0.0101969    10 sex1
 0.0099677    76 x.rfbing5
 0.0099426    77 drocdy3.
 0.0098849    33 hlthpln1
 0.0098613    82 x.race
 0.0097427    75 x.drnkwek
 0.0097074     8 blind
 0.0090413    21 alcday5
 0.0090204   101 x.mrace1
```

The fourth pair I used was OneRAttributeEval + Ranker. According to Weka, this evaluator ranks each attribute using the 'OneR' classifier[2]. Per the module, the 1R (or one rule) algorithm generates rules based on a single attribute[1]. The algorithm generates rules based on each attribute with respect to the predicted class and then selects the attribute which minimizes classification error. Then, a set of rules are created from that single attribute. Again, I chose to use the top 20 by rank. By index, these are 22, 2, 31, 95, 46, 87, 34, 25, 66, 24, 27, 53, 67, 29, 11, 104, 62, 36, 13, and 50. Names: employl, marital, deaf, diffwalk, diffdres, diffalon, rmvteth4, diabete3, physhlth, genhlth, chckdnyl, chccpodl, cvdstrk3, cvdcrhd4, x.age.g, x.ageg5yr, x.age65yr, x.rfhlth, x.phys14d, and x.michd.

```
=== Attribute Selection on all input data ===

Search Method:
	Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 108 havarth3):
	OneR feature evaluator.

	Using 10 fold cross validation for evaluating attributes.
	Minimum bucket size for OneR: 6

Ranked attributes:
73.4891    22 diffwalk
70.2514     2 employl
69.8324    31 physhlth
69.4769    95 x.phys14d
69.3499    46 chccopdl
69.0452    87 x.rfhlth
69.0325    34 genhlth
68.5754    25 diffalon
68.4992    66 x.ageg5yr
68.4104    24 diffdres
68.1691    27 rmvteth4
68.1056    53 cvdcrhd4
68.0675    67 x.age65yr
68.0295    29 diabete3
67.9152    11 marital
67.8771   104 x.michd
67.8517    62 x.age.g
67.8136    36 chckdnyl
67.6866    13 deaf
67.4962    50 cvdstrk3
67.4327    64 x.age80
67.2676    52 cvdinfr4
67.0772    45 chcocncr
66.9756     8 blind
66.8994    51 sleptiml
66.874     49 asthma3
66.8614   105 x.ltasthl
66.8487    94 x.rawrake
66.8233    42 medcost
66.8233    43 checkupl
66.8233     1 x.aidtst3
66.8233    39 imonth
66.8233   106 x.casthml
66.8233    40 iday
66.8233    28 lastden4
66.8233    38 fmonth
66.8233    37 iyear
66.8233    10 sexl
66.8233     9 renthoml
66.8233     7 veteran3
66.8233     6 children
66.8233     3 income2
66.8233    12 educa
66.8233    15 x.drnkdrv
66.8233    18 hivtst6
66.8233    33 hlthplnl
66.8233    35 dispcode
66.8233    56 x.metstat
66.8233    20 pneuvac4
66.8233    21 alcday5
66.8233    55 qstlang
66.8233    54 qstver
66.8233    82 x.race
66.8233    83 x.urbstat
```

The last pair I used was CfsSubsetEval + Greedy Stepwise (first non-Ranked Search Method). According to Weka, this works by evaluating the worth of attribute subsets by considering the individual predictive ability of each future along with the degree of redundancy between them[2]. Feature subsets with high correlation and low intercorrelation are preferred. Unlike the four above selection methods, this method uses Greedy Stepwise instead of Ranking. According to Weka, Greedy Stepwise works by either progressing forward or backward from a single point (which could have no attribute at the beginning or a number of arbitrary starting attributes) and continuing to progress through additions/subtractions until a change in attribute leads to a lower evaluation[2]. The most recent change is undone, and that attribute set is returned. The method is called as such because it incorporates the Greedy Method. The Greedy Method is an algorithmic approach in which a solution is constructed part-by-part, choosing the next part to maximize the benefit, according to tutorialspoint.com[4]. This Attribute Evaluator/Search Method leaves us with a grouping of 13 attributes- indexed 2, 10, 20, 22, 24, 29, 31, 34, 43, 46, 64, 57, and 102. Names: employl, sex, pneuvac4, diffwalk, diffdres, diabete3, physhlth, genhlth, checkupl, chccopdl, x.age80, x.age65yr, and x.exteth3.

```
=== Attribute Selection on all input data ===

Search Method:
        Greedy Stepwise (forwards).
        Start set: no attributes
        Merit of best subset found:    0.137

Attribute Subset Evaluator (supervised, Class (nominal): 108 havarth3):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 2,10,20,22,24,29,31,34,43,46,64,67,102 : 13
                     employ1
                     sex1
                     pneuvac4
                     diffwalk
                     diffdres
                     diabete3
                     physhlth
                     genhlth
                     checkup1
                     chccopd1
                     x.age80
                     x.age65yr
                     x.exteth3
```

Classifier Algorithms-

As mentioned previously, I had an idea of 3 classification algorithms I wanted to use: Decision Tree (J48 in Weka), RandomForest, and kNN. Given that I wanted to use a wide assortment of algorithm types, I decided on Logistic Regression and Naive Bayesian as my final two choices.

The first classification algorithm I used was Decision Tree (J48 in Weka). As briefly discussed above, a Decision Tree works in the following way, per module notes: A decision tree is a classification algorithm which works by branching off from internal nodes by way of an attribute as the input datapoint traverses down the tree[1]. Initially, the entire training dataset is associated with the root node. A test attribute is chosen (one of three ways- information gain, gain ratio, and Gini index) and the dataset is split into subsets based on the value of the test attribute. These two steps are then repeated indefinitely until a stop condition is met- either due to running out of new attributes to split along or from a parameter-based limitation- such as hitting a maximum depth. Once the tree is created from training data, test data can be inputted through the top of the tree, and it is classified dependent on where it ends up at the bottom of the tree.

The second classification algorithm method I used was RandomForest. RandomForest is an evolution of the Decision Tree method listed above. Per module notes, the algorithm builds multiple trees and then combines their classifications to make a final classification prediction[1]. When a test object is classified, each tree gives their opinion, and the final class prediction is decided by vote. Of note, RandomForest has three main benefits over traditional Decision Trees, per lecture notes[1]. First, the algorithm is often less affected by outliers and errors. Second, overfitting is not much of an issue compared to Decision Tree and lastly, only subsets of attributes are considered at each node- which makes the algorithm run faster than Decision Tree.

The next classification algorithm to use is Logistic Regression. Unlike traditional Linear Regression, Logistic Regression can be used for classification by incorporating the logistic response function[1]. To have a linear expression on the right-side of the equation, odds is used (referring to the ratio of the probability of belonging to one class versus another)- odds(Y = 1) = (p / (1 - p)). By substituting 'p' with the logistic function and applying the natural logarithm to both sides, we are left with an equation that is suitable for regression.

log(odds) = B0 + B1X1 + B2X2 + ... + BnXn

The fourth classification algorithm I used is Naive Bayesian. According to lecture nodes, this classifier predicts classification based on the probability of class membership[1]. The theorem is based upon Bayes' theorem computing the posteriori probability of hypothesis P(H | X):

P(H | X) = P(X | H) * P(H) / (P(X)

Naive Bayesian drastically reduces calculating time by assuming that there are no dependent relationships among attributes (within a tuple)[1]. It is quite an assumption to make that all attributes are independent of one another, but it is a necessary assumption to make. This is where the 'naive' comes from with respect to Naive Bayesian.

The fifth and final classification algorithm I used is kNN (k Nearest Neighbors). As mentioned previously, I normalized the data in preprocessing step due to wanting to use this algorithm (although I probably would've been so regardless). The algorithm works by assigning a classification label to a point by its proximity to the nearest 'k' data points[1]. Similarity is used with a distance measure- often Euclidean or Manhattan distance, from lecture notes. The class of an unknown tuple is decided by majority voting of the nearest 'k' points by distance[1]. After briefly iterating through odd 'k' values (such that there could be no ties) I decided on a k value of k = 7. From my experimenting, this was a suitable number to pick to maximize accuracy and minimize evaluation time (although increasing 'k' to be too large of a value began to reduce accuracy).

Test Results-

See the 25 lots of test results before. Each test result is labelled according to its Attribute Selection Method and Classification Model in the following way:

Attribute Selection -	
CorrelationAttributeEval / Ranker	a
InfoGainAttributeEval / Ranker	b
SymmetricalUncertAttribute Eval / Ranke	c
OneRAttributeEval / Ranker	d
CfsSubsetEval / GreedyStepwise	e
Models-	
J48 (Decision Tree)	1
RandomForest	2
Logistic	3
Naïve Bayes	4
Ibk (kNN)	5

A1

```
=== Summary ===

Correctly Classified Instances        2954               72.8124 %
Incorrectly Classified Instances      1103               27.1876 %
Kappa statistic                          0.3655
Mean absolute error                      0.3459
Root mean squared error                  0.4389
Relative absolute error                 77.0362 %
Root relative squared error             92.0531 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.856    0.511    0.758      0.856    0.804      0.373    0.748     0.815     2
                 0.489    0.144    0.645      0.489    0.556      0.373    0.748     0.573     1
Weighted Avg.    0.728    0.383    0.719      0.728    0.718      0.373    0.748     0.731

=== Confusion Matrix ===

    a    b   <-- classified as
 2263  380 |    a = 2
  723  691 |    b = 1
```

A2

```
=== Summary ===

Correctly Classified Instances        2899               71.4567 %
Incorrectly Classified Instances      1158               28.5433 %
Kappa statistic                          0.3467
Mean absolute error                      0.3345
Root mean squared error                  0.443
Relative absolute error                 74.4928 %
Root relative squared error             92.9187 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.824    0.491    0.758      0.824    0.790      0.350    0.750     0.838     2
                 0.509    0.176    0.608      0.509    0.554      0.350    0.750     0.598     1
Weighted Avg.    0.715    0.381    0.706      0.715    0.708      0.350    0.750     0.755

=== Confusion Matrix ===

    a    b   <-- classified as
 2179  464 |    a = 2
  694  720 |    b = 1
```

A3

```
=== Summary ===

Correctly Classified Instances        3013               74.2667 %
Incorrectly Classified Instances      1044               25.7333 %
Kappa statistic                          0.3843
Mean absolute error                      0.3365
Root mean squared error                  0.415
Relative absolute error                 74.9255 %
Root relative squared error             87.042  %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.894    0.540    0.756      0.894    0.819      0.401    0.796     0.873     2
                 0.460    0.106    0.699      0.460    0.555      0.401    0.796     0.663     1
Weighted Avg.    0.743    0.389    0.736      0.743    0.727      0.401    0.796     0.800

=== Confusion Matrix ===

    a    b   <-- classified as
 2363  280 |    a = 2
  764  650 |    b = 1
```

A4

```
=== Summary ===

Correctly Classified Instances        2894               71.3335 %
Incorrectly Classified Instances      1163               28.6665 %
Kappa statistic                          0.4015
Mean absolute error                      0.2907
Root mean squared error                  0.5008
Relative absolute error                 64.7396 %
Root relative squared error            105.0235 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.717    0.293    0.820      0.717    0.765      0.408    0.783     0.868     2
                 0.707    0.283    0.572      0.707    0.632      0.408    0.783     0.631     1
Weighted Avg.    0.713    0.290    0.734      0.713    0.719      0.408    0.783     0.786

=== Confusion Matrix ===

    a    b   <-- classified as
 1895  748 |    a = 2
  415  999 |    b = 1
```

A5

```
=== Summary ===

Correctly Classified Instances        2926               72.1223 %
Incorrectly Classified Instances      1131               27.8777 %
Kappa statistic                          0.3469
Mean absolute error                      0.3344
Root mean squared error                  0.4352
Relative absolute error                 74.4778 %
Root relative squared error             91.2823 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.855    0.529    0.751      0.855    0.800      0.355    0.758     0.831     2
                 0.471    0.145    0.635      0.471    0.541      0.355    0.758     0.605     1
Weighted Avg.    0.721    0.395    0.711      0.721    0.710      0.355    0.758     0.752

=== Confusion Matrix ===

    a    b   <-- classified as
 2260  383 |    a = 2
  748  666 |    b = 1
```

B1

```
=== Summary ===

Correctly Classified Instances        2988               73.6505 %
Incorrectly Classified Instances      1069               26.3495 %
Kappa statistic                          0.3801
Mean absolute error                      0.3381
Root mean squared error                  0.4357
Relative absolute error                 75.2974 %
Root relative squared error             91.3896 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.871    0.516    0.760      0.871    0.812      0.390    0.749     0.814     2
                 0.484    0.129    0.668      0.484    0.562      0.390    0.749     0.587     1
Weighted Avg.    0.737    0.381    0.728      0.737    0.725      0.390    0.749     0.735

=== Confusion Matrix ===

    a    b   <-- classified as
 2303  340 |    a = 2
  729  685 |    b = 1
```

B2

```
=== Summary ===

Correctly Classified Instances        2935               72.3441 %
Incorrectly Classified Instances      1122               27.6559 %
Kappa statistic                          0.3703
Mean absolute error                      0.334
Root mean squared error                  0.4338
Relative absolute error                 74.3884 %
Root relative squared error             90.9808 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.826    0.467    0.767      0.826    0.795      0.373    0.763     0.845     2
                 0.533    0.174    0.620      0.533    0.573      0.373    0.763     0.609     1
Weighted Avg.    0.723    0.365    0.716      0.723    0.718      0.373    0.763     0.762

=== Confusion Matrix ===

    a    b   <-- classified as
 2182  461 |    a = 2
  661  753 |    b = 1
```

B3

```
=== Summary ===

Correctly Classified Instances        3033               74.7597 %
Incorrectly Classified Instances      1024               25.2403 %
Kappa statistic                          0.4019
Mean absolute error                      0.3325
Root mean squared error                  0.4127
Relative absolute error                 74.0533 %
Root relative squared error             86.5587 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.888    0.514    0.763      0.888    0.821      0.415    0.800     0.876     2
                 0.486    0.112    0.698      0.486    0.573      0.415    0.800     0.671     1
Weighted Avg.    0.748    0.374    0.741      0.748    0.734      0.415    0.800     0.804

=== Confusion Matrix ===

    a    b   <-- classified as
 2346  297 |    a = 2
  727  687 |    b = 1
```

B4

```
=== Summary ===

Correctly Classified Instances        2909               71.7032 %
Incorrectly Classified Instances      1148               28.2968 %
Kappa statistic                          0.4112
Mean absolute error                      0.2875
Root mean squared error                  0.4992
Relative absolute error                 64.029  %
Root relative squared error            104.7067 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.716    0.281    0.827      0.716    0.767      0.418    0.786     0.871     2
                 0.719    0.284    0.575      0.719    0.639      0.418    0.786     0.639     1
Weighted Avg.    0.717    0.282    0.739      0.717    0.723      0.418    0.786     0.790

=== Confusion Matrix ===

    a    b   <-- classified as
 1892  751 |    a = 2
  397 1017 |    b = 1
```

B5

```
=== Summary ===

Correctly Classified Instances        2927               72.1469 %
Incorrectly Classified Instances      1130               27.8531 %
Kappa statistic                          0.3568
Mean absolute error                      0.3334
Root mean squared error                  0.4346
Relative absolute error                 74.2544 %
Root relative squared error             91.145  %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.840    0.499    0.759      0.840    0.797      0.362    0.762     0.834     2
                 0.501    0.160    0.625      0.501    0.556      0.362    0.762     0.600     1
Weighted Avg.    0.721    0.381    0.712      0.721    0.713      0.362    0.762     0.753

=== Confusion Matrix ===

    a    b   <-- classified as
 2219  424 |    a = 2
  706  708 |    b = 1
```

C1

```
=== Summary ===

Correctly Classified Instances        2990               73.6998 %
Incorrectly Classified Instances      1067               26.3002 %
Kappa statistic                          0.3903
Mean absolute error                      0.34
Root mean squared error                  0.436
Relative absolute error                 75.7069 %
Root relative squared error             91.4399 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.856    0.485    0.767      0.856    0.809      0.396    0.754     0.822     2
                 0.515    0.144    0.656      0.515    0.577      0.396    0.754     0.581     1
Weighted Avg.    0.737    0.366    0.729      0.737    0.728      0.396    0.754     0.738

=== Confusion Matrix ===

    a    b   <-- classified as
 2262  381 |    a = 2
  686  728 |    b = 1
```

C2

```
=== Summary ===

Correctly Classified Instances        2889               71.2103 %
Incorrectly Classified Instances      1168               28.7897 %
Kappa statistic                          0.3451
Mean absolute error                      0.3324
Root mean squared error                  0.4396
Relative absolute error                 74.027  %
Root relative squared error             92.1942 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.816    0.482    0.760      0.816    0.787      0.347    0.756     0.842     2
                 0.518    0.184    0.601      0.518    0.557      0.347    0.756     0.610     1
Weighted Avg.    0.712    0.378    0.704      0.712    0.707      0.347    0.756     0.761

=== Confusion Matrix ===

    a    b   <-- classified as
 2156  487 |    a = 2
  681  733 |    b = 1
```

C3

```
=== Summary ===

Correctly Classified Instances        3029               74.6611 %
Incorrectly Classified Instances      1028               25.3389 %
Kappa statistic                          0.3989
Mean absolute error                      0.3333
Root mean squared error                  0.4126
Relative absolute error                 74.2301 %
Root relative squared error             86.5339 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.888    0.518    0.762      0.888    0.820      0.413    0.801     0.878     2
                 0.482    0.112    0.697      0.482    0.570      0.413    0.801     0.667     1
Weighted Avg.    0.747    0.376    0.740      0.747    0.733      0.413    0.801     0.804

=== Confusion Matrix ===

    a    b   <-- classified as
 2347  296 |    a = 2
  732  682 |    b = 1
```

C4

```
=== Summary ===

Correctly Classified Instances        2922               72.0237 %
Incorrectly Classified Instances      1135               27.9763 %
Kappa statistic                          0.4143
Mean absolute error                      0.2844
Root mean squared error                  0.4937
Relative absolute error                 63.3441 %
Root relative squared error            103.5386 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.726    0.290    0.824      0.726    0.772      0.420    0.789     0.873     2
                 0.710    0.274    0.581      0.710    0.639      0.420    0.789     0.640     1
Weighted Avg.    0.720    0.285    0.739      0.720    0.725      0.420    0.789     0.792

=== Confusion Matrix ===

    a    b   <-- classified as
 1918  725 |    a = 2
  410 1004 |    b = 1
```

C5

```
=== Summary ===

Correctly Classified Instances        2958               72.911  %
Incorrectly Classified Instances      1099               27.089  %
Kappa statistic                          0.3676
Mean absolute error                      0.3301
Root mean squared error                  0.4326
Relative absolute error                 73.5021 %
Root relative squared error             90.7227 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.857    0.511    0.758      0.857    0.805      0.375    0.762     0.832     2
                 0.489    0.143    0.647      0.489    0.557      0.375    0.762     0.612     1
Weighted Avg.    0.729    0.382    0.720      0.729    0.719      0.375    0.762     0.756

=== Confusion Matrix ===

    a    b   <-- classified as
 2266  377 |    a = 2
  722  692 |    b = 1
```

D1

```
=== Summary ===

Correctly Classified Instances        2975               73.33   %
Incorrectly Classified Instances      1082               26.67   %
Kappa statistic                          0.3658
Mean absolute error                      0.345
Root mean squared error                  0.4339
Relative absolute error                 76.8274 %
Root relative squared error             90.9971 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.880    0.542    0.752      0.880    0.811      0.379    0.753     0.817     2
                 0.458    0.120    0.672      0.458    0.545      0.379    0.753     0.585     1
Weighted Avg.    0.733    0.395    0.724      0.733    0.719      0.379    0.753     0.736

=== Confusion Matrix ===

    a    b   <-- classified as
 2327  316 |    a = 2
  766  648 |    b = 1
```

D2

```
=== Summary ===

Correctly Classified Instances        2932               72.2702 %
Incorrectly Classified Instances      1125               27.7298 %
Kappa statistic                          0.3685
Mean absolute error                      0.3344
Root mean squared error                  0.4396
Relative absolute error                 74.4656 %
Root relative squared error             92.1925 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.825    0.469    0.767      0.825    0.795      0.371    0.754     0.835     2
                 0.531    0.175    0.619      0.531    0.572      0.371    0.754     0.594     1
Weighted Avg.    0.723    0.366    0.715      0.723    0.717      0.371    0.754     0.751

=== Confusion Matrix ===

    a    b   <-- classified as
 2181  462 |    a = 2
  663  751 |    b = 1
```

D3

```
=== Summary ===

Correctly Classified Instances        3028               74.6364 %
Incorrectly Classified Instances      1029               25.3636 %
Kappa statistic                          0.3984
Mean absolute error                      0.3343
Root mean squared error                  0.4132
Relative absolute error                 74.4416 %
Root relative squared error             86.6588 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.888    0.518    0.762      0.888    0.820      0.412    0.800     0.877     2
                 0.482    0.112    0.697      0.482    0.570      0.412    0.800     0.662     1
Weighted Avg.    0.746    0.376    0.739      0.746    0.733      0.412    0.800     0.802

=== Confusion Matrix ===

    a    b   <-- classified as
 2346  297 |    a = 2
  732  682 |    b = 1
```

D4

```
=== Summary ===

Correctly Classified Instances        2967               73.1329 %
Incorrectly Classified Instances      1090               26.8671 %
Kappa statistic                          0.4159
Mean absolute error                      0.2766
Root mean squared error                  0.4654
Relative absolute error                 61.6057 %
Root relative squared error             97.6147 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.779    0.358    0.803      0.779    0.791      0.416    0.795     0.876     2
                 0.642    0.221    0.609      0.642    0.625      0.416    0.795     0.649     1
Weighted Avg.    0.731    0.310    0.735      0.731    0.733      0.416    0.795     0.797

=== Confusion Matrix ===

    a    b   <-- classified as
 2059  584 |    a = 2
  506  908 |    b = 1
```

D5

```
=== Summary ===

Correctly Classified Instances        2940               72.4673 %
Incorrectly Classified Instances      1117               27.5327 %
Kappa statistic                          0.3523
Mean absolute error                      0.3338
Root mean squared error                  0.4358
Relative absolute error                 74.3362 %
Root relative squared error             91.3927 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.862    0.533    0.752      0.862    0.803      0.362    0.760     0.837     2
                 0.467    0.138    0.645      0.467    0.542      0.362    0.760     0.600     1
Weighted Avg.    0.725    0.395    0.714      0.725    0.712      0.362    0.760     0.754

=== Confusion Matrix ===

    a    b   <-- classified as
 2279  364 |    a = 2
  753  661 |    b = 1
```

E1

```
=== Summary ===

Correctly Classified Instances        2990               73.6998 %
Incorrectly Classified Instances      1067               26.3002 %
Kappa statistic                          0.3677
Mean absolute error                      0.3491
Root mean squared error                  0.4325
Relative absolute error                 77.7461 %
Root relative squared error             90.7192 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.895    0.558    0.750      0.895    0.816      0.386    0.744     0.805     2
                 0.442    0.105    0.692      0.442    0.539      0.386    0.744     0.568     1
Weighted Avg.    0.737    0.400    0.730      0.737    0.720      0.386    0.744     0.722

=== Confusion Matrix ===

    a    b   <-- classified as
 2365  278 |    a = 2
  789  625 |    b = 1
```

E2

```
=== Summary ===

Correctly Classified Instances        2902               71.5307 %
Incorrectly Classified Instances      1155               28.4693 %
Kappa statistic                          0.3525
Mean absolute error                      0.3318
Root mean squared error                  0.4415
Relative absolute error                 73.8949 %
Root relative squared error             92.6008 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.818    0.477    0.762      0.818    0.789      0.355    0.756     0.843     2
                 0.523    0.182    0.606      0.523    0.562      0.355    0.756     0.603     1
Weighted Avg.    0.715    0.374    0.708      0.715    0.710      0.355    0.756     0.760

=== Confusion Matrix ===

    a    b   <-- classified as
 2162  481 |    a = 2
  674  740 |    b = 1
```

E3

```
=== Summary ===

Correctly Classified Instances        3035               74.809  %
Incorrectly Classified Instances      1022               25.191  %
Kappa statistic                          0.4068
Mean absolute error                      0.3319
Root mean squared error                  0.412
Relative absolute error                 73.9216 %
Root relative squared error             86.4135 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.881    0.501    0.767      0.881    0.820      0.418    0.802     0.875     2
                 0.499    0.119    0.692      0.499    0.580      0.418    0.802     0.670     1
Weighted Avg.    0.748    0.368    0.741      0.748    0.736      0.418    0.802     0.804

=== Confusion Matrix ===

    a    b   <-- classified as
 2329  314 |    a = 2
  708  706 |    b = 1
```

E4

```
=== Summary ===

Correctly Classified Instances        2965               73.0836 %
Incorrectly Classified Instances      1092               26.9164 %
Kappa statistic                          0.4204
Mean absolute error                      0.2803
Root mean squared error                  0.4559
Relative absolute error                 62.4206 %
Root relative squared error             95.6241 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.767    0.337    0.810      0.767    0.788      0.422    0.796     0.874     2
                 0.663    0.233    0.604      0.663    0.632      0.422    0.796     0.654     1
Weighted Avg.    0.731    0.301    0.738      0.731    0.733      0.422    0.796     0.798

=== Confusion Matrix ===

    a    b   <-- classified as
 2028  615 |    a = 2
  477  937 |    b = 1
```

E5

```
=== Summary ===

Correctly Classified Instances        2930               72.2209 %
Incorrectly Classified Instances      1127               27.7791 %
Kappa statistic                          0.3558
Mean absolute error                      0.332
Root mean squared error                  0.4365
Relative absolute error                 73.9354 %
Root relative squared error             91.5452 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.845    0.507    0.757      0.845    0.798      0.361    0.756     0.826     2
                 0.493    0.155    0.630      0.493    0.553      0.361    0.756     0.601     1
Weighted Avg.    0.722    0.384    0.713      0.722    0.713      0.361    0.756     0.747

=== Confusion Matrix ===

    a    b   <-- classified as
 2233  410 |    a = 2
  717  697 |    b = 1
```

Best Attribute Selection Method/Classification Algorithm-

To evaluate the results and choose the best pair, I constructed two 5x5 tables- the first of each model's Weighted Avg. F-Measure:

Weighted F Measure

	a	b	c	d	e
1	0.718	0.725	0.728	0.719	0.72
2	0.708	0.718	0.707	0.717	0.71
3	0.727	0.734	0.733	0.733	0.736
4	0.719	0.723	0.725	0.733	0.733
5	0.71	0.713	0.719	0.712	0.713

and the second- of the Recall metric for Class 1:

Recall Class 1

	a	b	c	d	e
1	0.489	0.484	0.515	0.458	0.442
2	0.509	0.533	0.518	0.531	0.523
3	0.46	0.486	0.482	0.482	0.499
4	0.707	0.719	0.71	0.642	0.663
5	0.471	0.501	0.489	0.467	0.493

I chose these classification metrics for two reasons. I chose Weighted Avg. F-Measure as one metric as I wanted a more standard performance metric that demonstrated how the model performed generally. F Measure combines precision (which, per module, measures exactness) and recall (which measures completeness)[1]. When examining the F-Measure table above, there aren't really any standout values- the data mostly hovers around 0.71 to 0.74.

Given the context of our dataset, I wanted to look at an additional performance metric that would be more in line with why the data was originally gathered in the first place. The original dataset is from a survey and relates to whether an individual was ever told they had some form of arthritis, rheumatoid arthritis, gout, lupus, or fibromyalgia. Given that we are dealing with medicine (and specifically diagnosing harmful medical conditions) it would be safe to say that the most important aspect of a classifier model built from this dataset would be able to correctly predict whether an individual which should be classified with one of these ailments is (we want to minimize False Negatives FNs). Given this, I chose Recall with respect to Class 1 as my second performance metric. As mentioned above, Recall examines the completeness, or per module notes, "what percentage of positive tuples are correctly classified as positive tuples."[1]

When evaluating the above table, the three values that stick out are 'A4', 'B4', and 'C4' (Naive Bayesian performs quite well). I decided that the best combination is 'B4'- which represents InfoGainAttributeEval + Ranker Attribute Selection with the Naive Bayesian model. 'B4' has an F-Measure on the higher side of 0.723 and the highest Recall metric for Class 1 at 0.719.

The attributes used in 'B4' are: employl, children, marital, pneuvac4, diffwalk, rmvteth4, diabete3, physhlth, gnhlth, checkupl, chccopdl, x.age.g, x.age80, x.ageg5yr, x.age65yr, x.chldcnt, x.rfhlth, x.phys14d, x.hcvu651, and x.exteth3.

B4

```
=== Summary ===

Correctly Classified Instances        2909               71.7032 %
Incorrectly Classified Instances      1148               28.2968 %
Kappa statistic                          0.4112
Mean absolute error                      0.2875
Root mean squared error                  0.4992
Relative absolute error                 64.029  %
Root relative squared error            104.7067 %
Total Number of Instances             4057

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.716    0.281    0.827      0.716    0.767      0.418    0.786     0.871     2
                 0.719    0.284    0.575      0.719    0.639      0.418    0.786     0.639     1
Weighted Avg.    0.717    0.282    0.739      0.717    0.723      0.418    0.786     0.790

=== Confusion Matrix ===

    a    b   <-- classified as
 1892  751 |    a = 2
  397 1017 |    b = 1
```

As mentioned above, when evaluating the criteria used for selecting the best model I wanted to focus on Recall metric for Class 1 while not discounting Weighted Avg. F Measure. Given that the dataset pertains to identifying adverse medical conditions, it makes sense to use an evaluation metric that focuses on limiting FNs and Recall metric for Class 1 works well for this purpose. There were some other criteria I considered with respect to classifying Class 1, but I believe that Recall is the most solid metric in this respect.

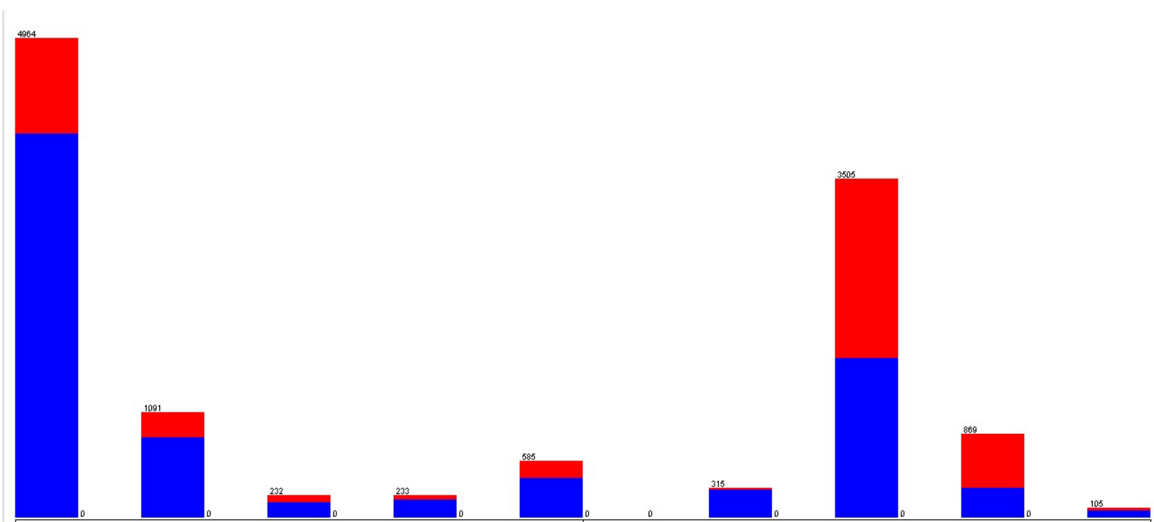
Five attributes which I think are most relevant to the class attribute are: employ1, diffwalk, genhlth, x.age80, and x.hcvu651 (2, 22, 34, 64, 97). I selected these attributes by carefully examining the attributes chosen across the five selections (with extra weight placed on both attributes ranked at the top and those present from the InfoGainAttributeEval + Ranking pair selection given that the model I identified as best used these selected attributes).

The employ1 attribute refers to a user's response to a question of their current employment. According to LLCP 2018 Codebook Report, answers are as follows[5]:

LLCP 2018 Codebook Report
Overall version data weighted with _LLCPWT
Behavioral Risk Factor Surveillance System

Label: Employment Status
Section Name: Demographics
Core Section Number: 8
Question Number: 15
Column: 172
Type of Variable: Num
SAS Variable Name: EMPLOY1
Question Prologue:
Question: Are you currently...?

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Employed for wages	180,094	41.29	47.69
2	Self-employed	39,400	9.03	9.47
3	Out of work for 1 year or more	8,266	1.90	2.40
4	Out of work for less than 1 year	8,507	1.95	2.53
5	A homemaker	21,280	4.88	5.84
6	A student	11,688	2.68	5.50
7	Retired	130,603	29.94	18.45
8	Unable to work	32,629	7.48	7.11
9	Refused	3,733	0.86	1.01
BLANK	Not asked or Missing	1,236	.	.



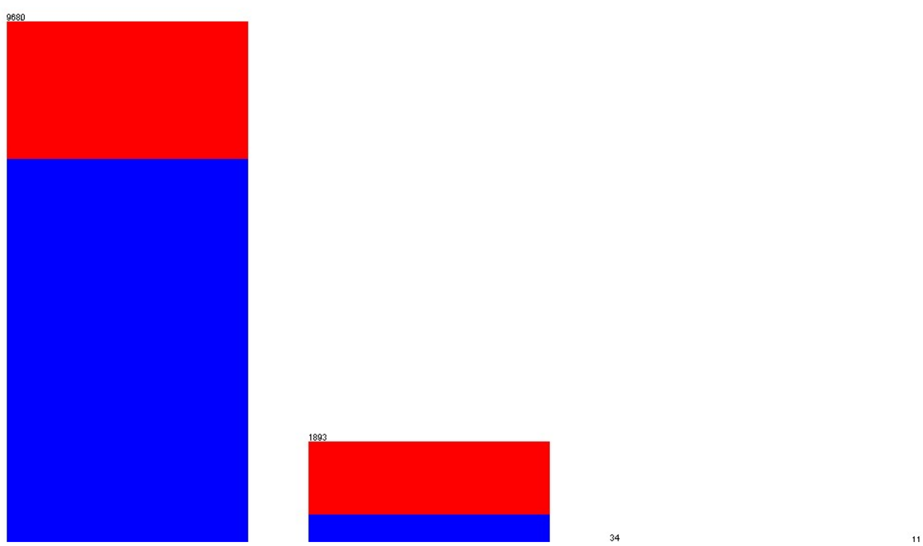
Examining the data, there is a relatively high red bias for individuals who stated that they were retired or unable to work, which makes intuitive sense given that such a condition may either force the employee to not work or usher them to an early retirement. Furthermore, there is a high blue bias for individuals who are students or self-employed, which makes sense given that young people are more likely to be self-employed or students.

'Diffwalk' refers to an individual having difficulty walking. From the Codebook Report[5]:

LLCP 2018 Codebook Report
Overall version data weighted with _LLCPWT
Behavioral Risk Factor Surveillance System

Label: Difficulty Walking or Climbing Stairs
Section Name: Demographics
Core Section Number: 8
Question Number: 24
Column: 189
Type of Variable: Num
SAS Variable Name: DIFFWALK
Question Prologue:
Question: Do you have serious difficulty walking or climbing stairs?

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Yes	71,832	16.87	13.70
2	No	352,556	82.78	86.01
7	Don’t know/Not Sure	1,146	0.27	0.20
9	Refused	374	0.09	0.09
BLANK	Not asked or Missing	11,528	.	.



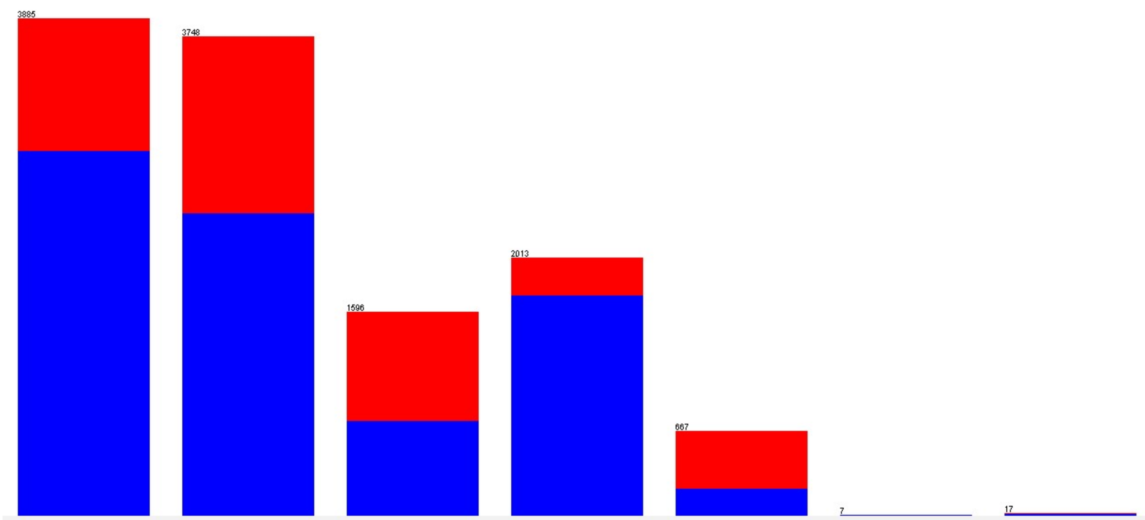
A vast majority of the individuals who stated they had difficulty walking belong to the positive class. This makes sense as conditions such as arthritis, gout, and fibromyalgia do most definitely impair someone's ability to walk.

The third attribute, 'genhlth' refers to an individual's assessment of their general health, per the Codebook Report[5]:

LLCP 2018 Codebook Report
Overall version data weighted with _LLCPWT
Behavioral Risk Factor Surveillance System

Label: General Health
Section Name: Health Status
Core Section Number: 1
Question Number: 1
Column: 90
Type of Variable: Num
SAS Variable Name: GENHLTH
Question Prologue:
Question: Would you say that in general your health is:

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Excellent	71,893	16.44	17.74
2	Very good	142,197	32.51	31.35
3	Good	138,321	31.62	32.16
4	Fair	60,762	13.89	13.73
5	Poor	23,120	5.29	4.78
7	Don't know/Not Sure	800	0.18	0.17
9	Refused	318	0.07	0.07
BLANK	Not asked or Missing	25	.	.



As one moves from one response category down the next, the percentage of red within the candlestick increases. Starting at 1 'Excellent' - the fourth candlestick from the left is largely blue. But going to the next candlestick (2, leftmost), and the next (3, to the right of 2), to the next (4, to the right of 3) and finally 5 (representing 'Poor'- at fifth from the left) one can see that the prevalence of Class 1 increases as reported general health declines. It makes intuitive sense that those who say they are worse off will be more likely to have been diagnosed with arthritis or any of the other flagged conditions.

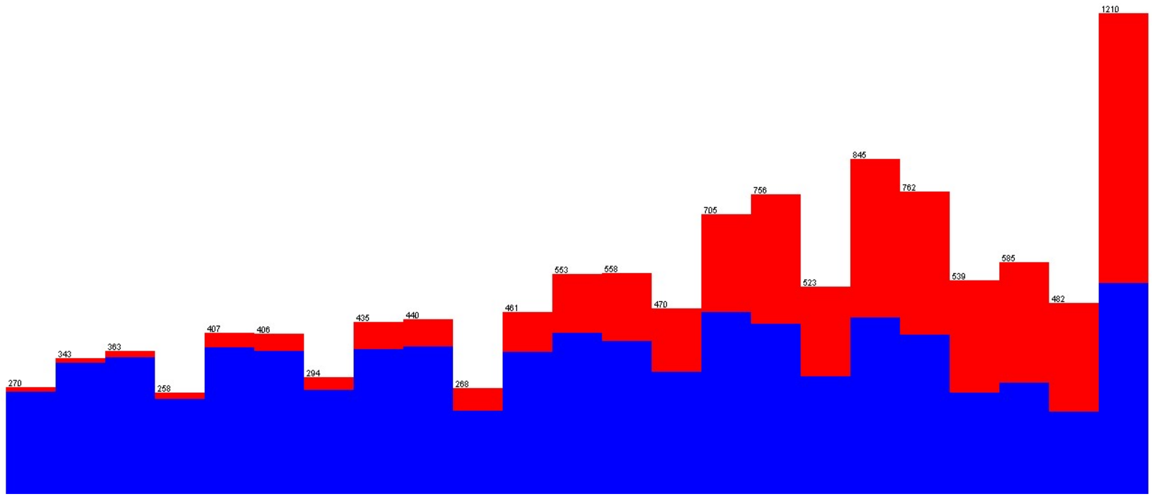
The fourth attribute, 'x.age80', asks if an individual is over the age of 80 or which sub-bracket of 5 years below they belong to[5]:

Label: Reported age in two age groups calculated variable
Section Name: Calculated Variables
Module Section Number: 8
Question Number: 12
Column: 1980
Type of Variable: Num
SAS Variable Name: _AGE65YR
Question Prologue:
Question: Two-level age category

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Age 18 to 64 Notes: 18 <= AGE <= 64	277,321	63.40	77.58
2	Age 65 or older Notes: 65 <= AGE <= 99	151,643	34.67	20.70
3	Don't know/Refused/Missing Notes: 7 <= AGE <= 9	8,472	1.94	1.73

Label: Imputed Age value collapsed above 80
Section Name: Calculated Variables
Module Section Number: 8
Question Number: 13
Column: 1981-1982
Type of Variable: Num
SAS Variable Name: _AGE80
Question Prologue:
Question: Imputed Age value collapsed above 80

Value	Value Label	Frequency	Percentage	Weighted Percentage
18 - 24	Imputed Age 18 to 24	26,012	5.95	12.35
25 - 29	Imputed Age 25 to 29	22,296	5.10	8.13
30 - 34	Imputed Age 30 to 34	24,308	5.56	9.25
35 - 39	Imputed Age 35 to 39	26,376	6.03	7.95
40 - 44	Imputed Age 40 to 44	26,089	5.96	8.32
45 - 49	Imputed Age 45 to 49	30,331	6.93	7.37
50 - 54	Imputed Age 50 to 54	37,505	8.57	9.01
55 - 59	Imputed Age 55 to 59	42,613	9.74	8.05
60 - 64	Imputed Age 60 to 64	47,982	10.97	8.61
65 - 69	Imputed Age 65 to 69	49,319	11.27	6.81
70 - 74	Imputed Age 70 to 74	41,179	9.41	5.66
75 - 79	Imputed Age 75 to 79	28,694	6.56	3.92
80 - 99	Imputed Age 80 or older	34,732	7.94	4.59



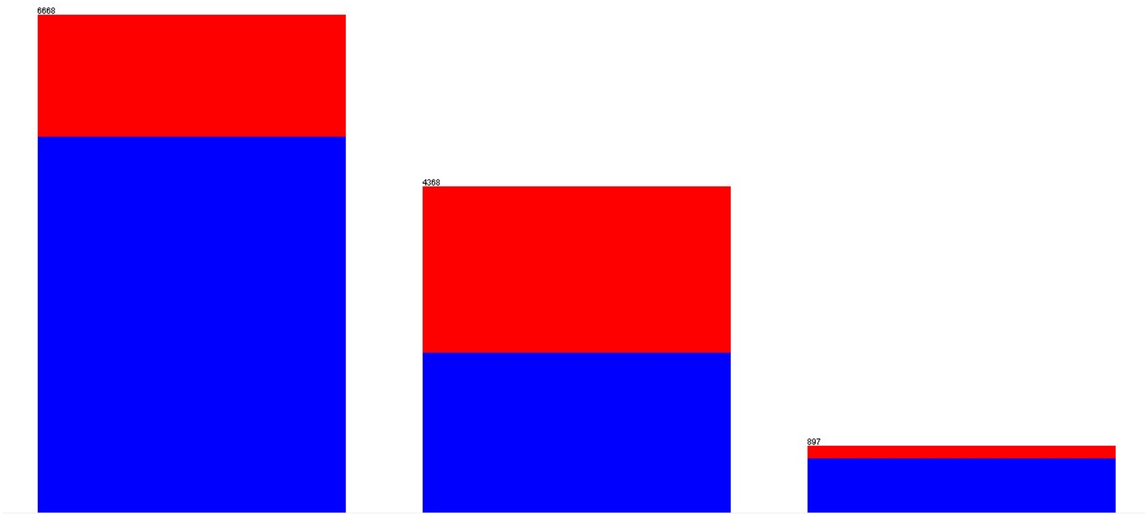
As the 5-year wide age gap increases, so does the prevalence of Class 1. This makes intuitive sense that the occurrence of arthritis, gout, and fibromyalgia would be correlated with age given that as individuals age, they are more likely to have health problems (especially arthritis). It also makes sense that elderly people are more likely to be diagnosed, given that they more than likely go see a physician more often than a younger person does.

The final attribute is 'x.hcvu651' which refers to whether respondents aged 18-64 have any form of health care coverage, per the LLCP 2018 Codebook Report[5]:

LLCP 2018 Codebook Report
Overall version data weighted with _LLCPWT
Behavioral Risk Factor Surveillance System

Label: Respondents aged 18-64 with health care coverage
Section Name: Calculated Variables
Module Section Number: 3
Question Number: 1
Column: 1901
Type of Variable: Num
SAS Variable Name: _HCVU651
Question Prologue:
Question: Respondents aged 18-64 who have any form of health care coverage

Value	Value Label	Frequency	Percentage	Weighted Percentage
1	Have health care coverage Notes: 18 <= AGE <=64 and HLTHPLN1 = 1	243,549	55.68	65.56
2	Do not have health care coverage Notes: 18 <= AGE <=64 and HLTHPLN1 = 2	32,464	7.42	11.50
9	Don't know/Not Sure, Refused or Missing Notes: AGE > 64 or AGE = Missing or HLTHPLN1 = 7 or 9 or Missing	161,423	36.90	22.94



While the previous attribute covered age quite well, this attribute deals with whether individuals not eligible for Medicare (age 18-64) have some form of health insurance. Persons who say that they do not have any form of health insurance are more likely to have been diagnosed with one of the prior referenced adverse conditions than those who have health insurance. Those who either didn't know, are not sure, or refused to answer have a lower prevalence of Class 1- If you are not even concerned with having health insurance, it doesn't seem too likely that you have a serious chronic health condition.

While working on this project, I observed a few things. First, it seems as though there are around 15 (plus or minus) attributes which were picked up by every single Attribute Selection Method and there was only variation in the last couple. This makes sense that the differing methods would prioritize many of the same attributes, and after examining the five attributes listed above, the attributes chosen by the method make intuitive sense. Secondly, as I touched on earlier, Naïve Bayesian was by far the best classifier concerning Recall Class 1 performance (which I mentioned earlier as a key target metric). Naïve Bayesian is a powerful algorithm, and it doesn't surprise me that it performed so well. I am a little surprised that beyond Naïve Bayesian, the second-best algorithm metric-wise was Logistic Regression. I assumed that Decision Tree or RandomForest would work quite well with this large data set and feature selection, and it's a little disappointing that they didn't fare better.

References-

Boston University Metropolitan College. (Accessed 2022, June 19). *Printable Lectures*. https://onlinecampus.bu.edu/ultra/courses/_85840_1/cl/outline

Weka Wiki Documentation. (Accessed 2022, June 19). *Docs -> Documentation*. https://waikato.github.io/weka-wiki/documentation/

Xiaohui Lin, Chao Li, Weijie Ren, Xiao Luo, Yanpeng Qi, A new feature selection method based on symmetrical uncertainty and interaction gain, Computational Biology and Chemistry, Volume 83, 2019, 107149, ISSN 1476-9271, https://doi.org/10.1016/j.compbiolchem.2019.107149. https://www.sciencedirect.com/science/article/pii/S1476927118303736

TutorialsPoint. (Accessed 2022, June 19). *Design and Analysis Greedy Method*. https://www.tutorialspoint.com/design_and_analysis_of_algorithms/design_and_analysis_of_algorithms_greedy_method.htm#:~:text=Greedy%20algorithms%20build%20a%20solution,used%20to%20solve%20optimization%20problems

LLCP 2018 Codebook Report. (2019, November 21). *Behavioral Risk Factor Surveillance System*. https://www.cdc.gov/brfss/annual_data/2018/pdf/codebook18_llcp-v2-508.pdf