Austin Prince

Data Mining Homework 2 Part B

**Preprocessing**

Much of the preprocessing for this assignment was similar to the last assignment. I used the same Gaussian imputation methods as we did for the last assignment to fill in missing variables. In order to use the data for association rule mining I had to translate the continuous gene expression values into a binary matrix where the value is 1 if the gene expression for a particular person was in the top 10% of expression for that particular gene (highest numerical value) and a 0 otherwise. In order to accomplish this, I loaded the data into a pandas DataFrame in order to accomplish this processing. I organized my DataFrame so that the individual genes are the columns and the individual patients are the rows. In order to convert the continuous gene expression values to binary data I sorted each column of the DataFrame in descending order with the highest values at the top and the lowest values at the bottom. Then I changed the top 10% of these values (in the case of Control this is 18 patients) into 1’s and the rest of the values into 0’s. I repeated this for every column (gene) in order to convert the continuous data to a binary matrix.

**Software**

For this assignment I used the mlxtend library in Python. This library allows me to plug in a DataFrame of the form created in preprocessing (binary matrix with 1’s for top 10% of gene expression) and generate frequent itemsets using the apriori algorithm with specified support percent and from there generate association rules with specified confidence percentage.

In order to generate frequent itemsets with 8% support I used the following line of code.

frequent\_itemsets = apriori(dataFrame, min\_support=0.08)

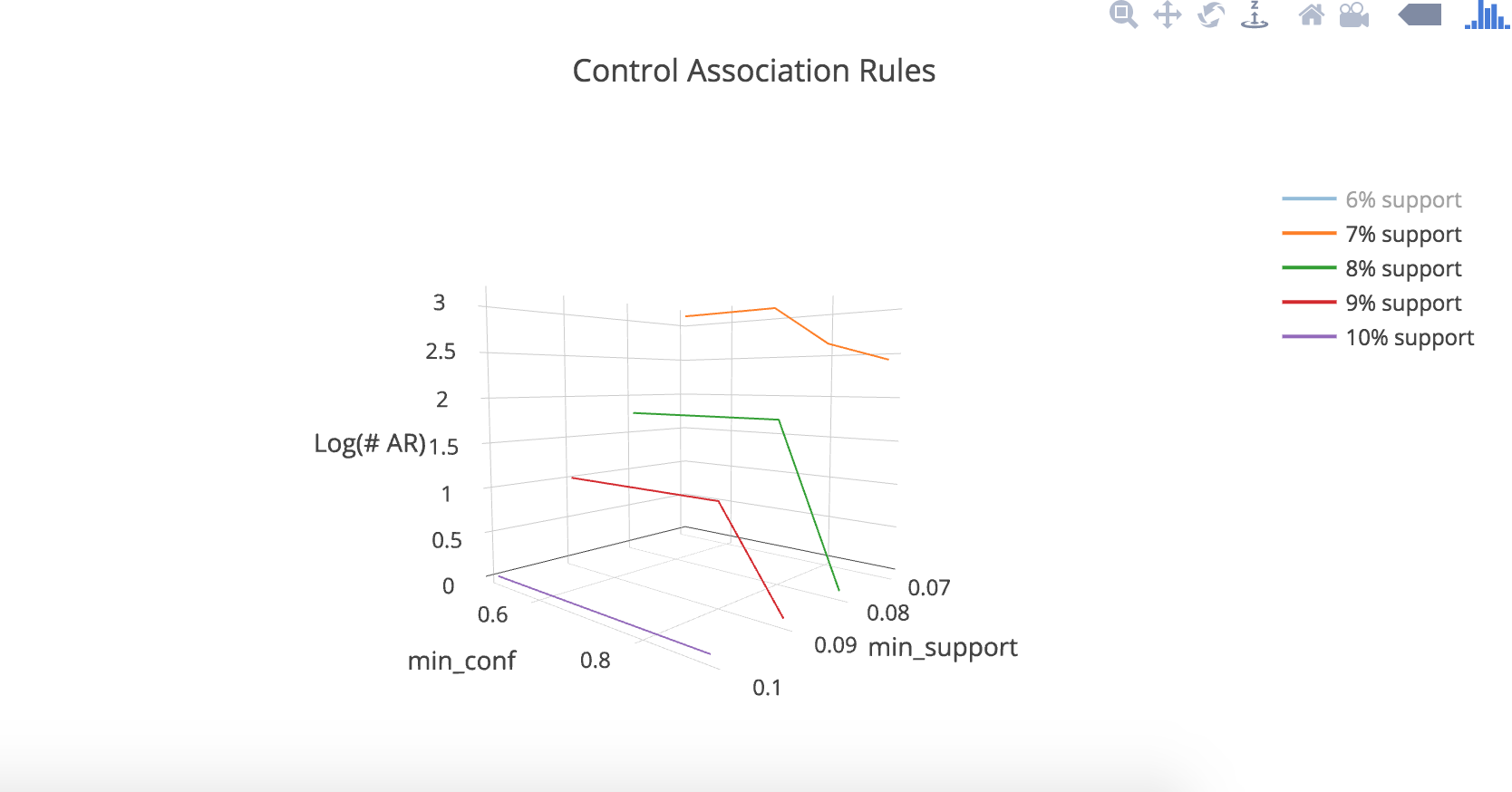
This code will return another DataFrame containing all of the frequent itemsets in the provided DataFrame with the specified min\_support. Once I have the frequent itemsets I can use the following line of code to generate association rules with confidence of 50%.

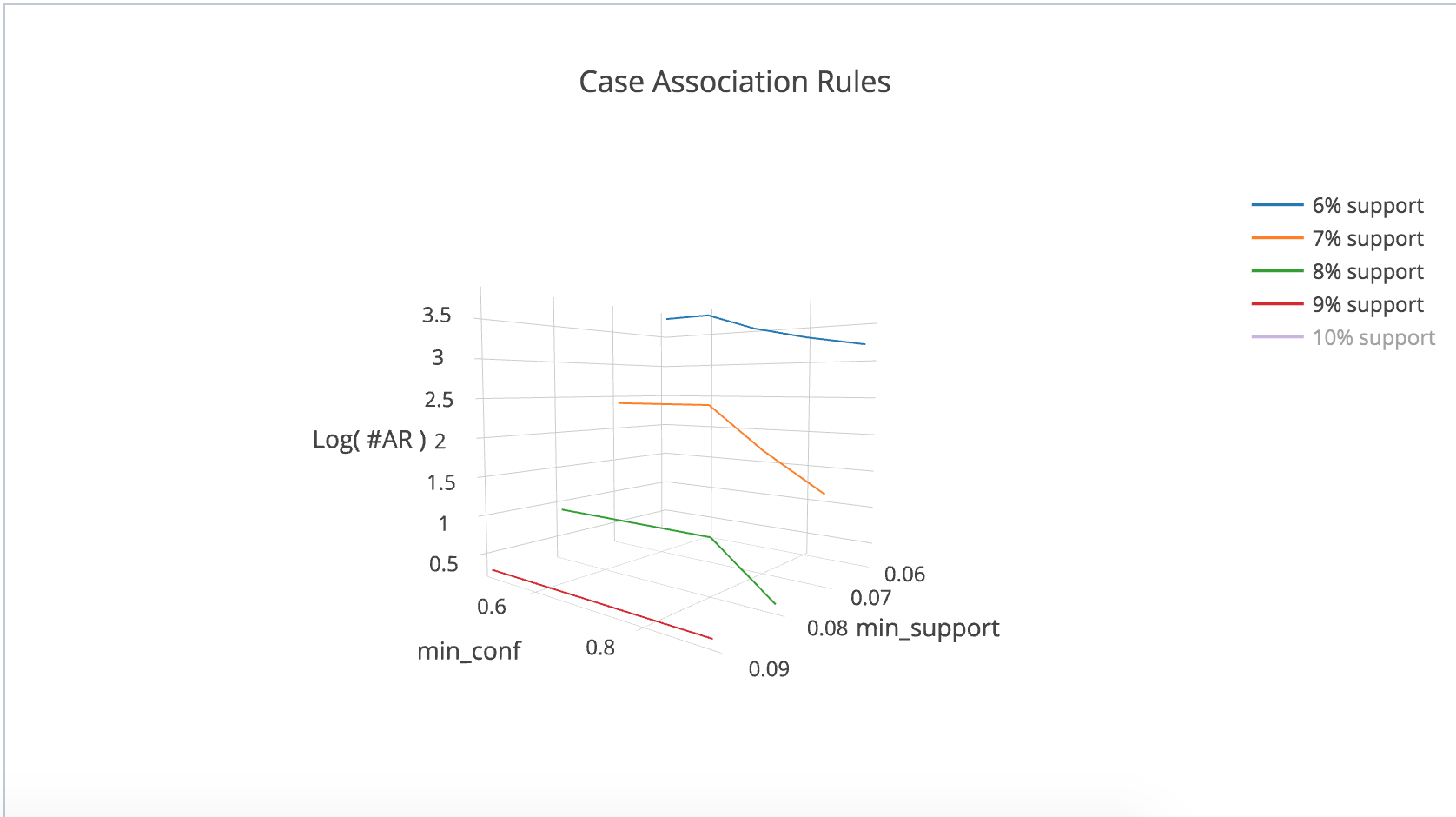
rules = association\_rules(frequent\_itemsets, metric=’confidence’, min\_threshold=0.5)

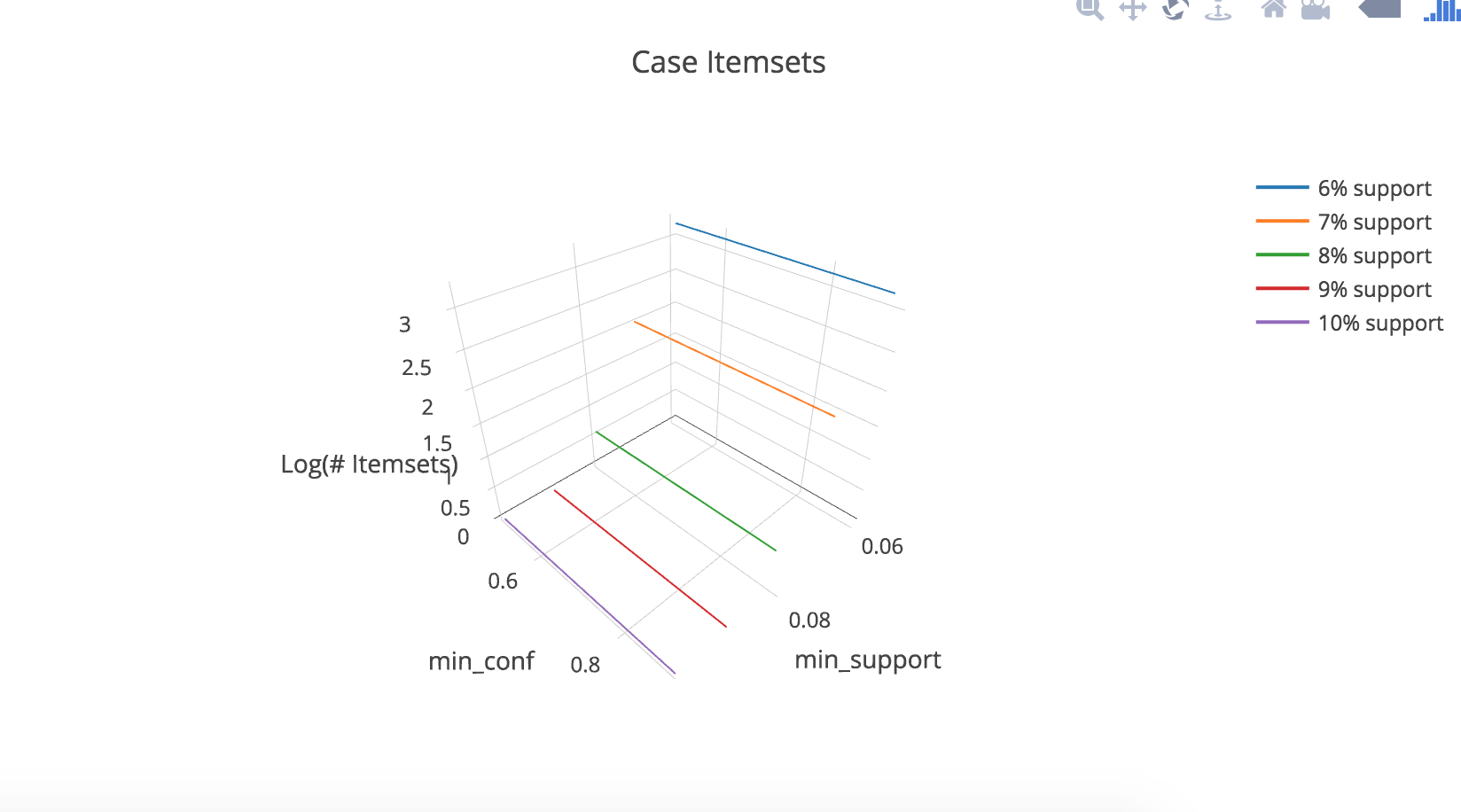
This code will return yet another DataFrame containing association rules with min\_support of 8% and min\_confidence of 50%. In order to store this data for usage in analysis I wrote both the frequent itemset and the association rules to csv files using pandas built in to\_csv function that can be applied to DataFrames.

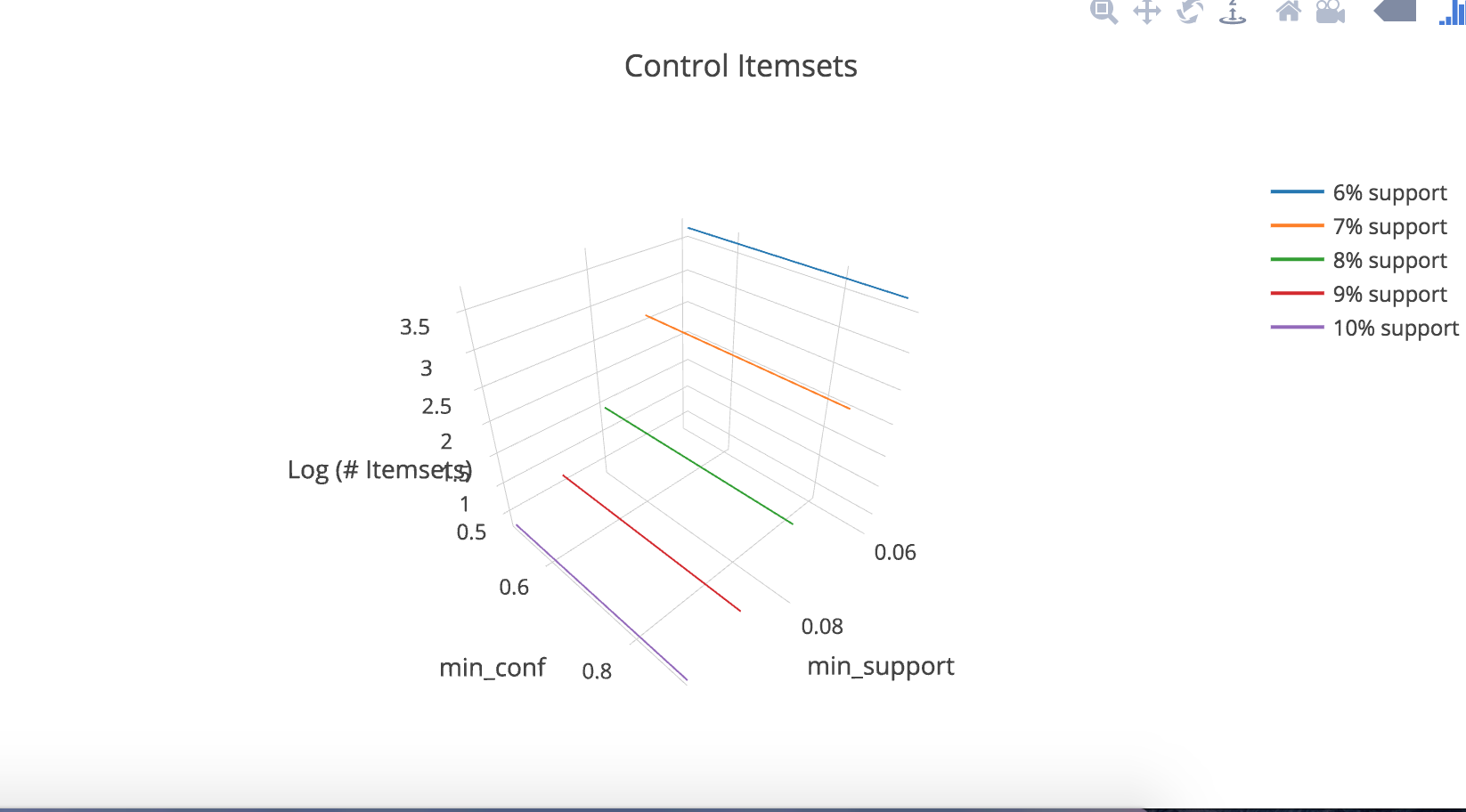
I ran the ctrl.gex and case.gex files using apriori separately because much of the insight given to us by association rules in this case will tell us how certain networks of gene expression are different in AD and non-AD patients.

1) Running the code and plotting the corresponding number of frequent itemsets and assocation rules for each combination of min\_support and min\_confidence I created the following graphs. I created these graphs using the plotly website which provides an easy interface to create 3d line graphs. The number of itemsets does not change as the min\_confidence value changes, which makes sense because the number of itemsets that are generated from apriori is only dependent on support, and has nothing to do with confidence. As the min\_support goes up the number of frequent itemsets goes down correspondingly which makes sense as at 10% the itemset must appear in more transactions (patients) so it is sensible that number of frequent itemsets will go down. For association rules the number of association rules is roughly linear (on a log scale) as the min\_confidence goes up.









2) I have decided to ignore the cast of single element itemsets because these are not instructive in any way. It is rather obvious that single element itemsets would have the highest frequency (each with 10%) because each individual gene appears in 10% of the total number of transactions (patients) as was specified in the preprocessing. Below is a list of the 50 most frequent itemsets for the control bucket of data.



Below is the list of the top 50 most frequent itemsets for case data.



Below is a list of the the top association rules for control data. These are the rules with the highest confidence meaning that they have the highest percentages of instances where when the antecedent exists the consequent also exists. In this case that means that for these instances if the genes in the antecedent are in the top 10% of expression for a patient then there is a very high probability that the gene in the consequent will also be expressed in the top 10%. I have chosen to put only the top 20 here because when running my program, I found upwards of 6000 Association rules within the suggested confidence and support percentages.



Below is a list of the of the top 20 association rules with the highest confidence for the case data set.



3) The meaning of association rules in this context gives us insight into cause and effect of gene regulation in patients with AD. An association rule defines a common rule that connects two transactions or in this case connects two item sets which are genes or groups of genes. In this case an association rule will tell us given that a patient has a certain set of genes expressed highly there is a high probability that another given set of genes will also be expressed. This definition is formalized in the notion of antecedent and consequent. We can say that within our minimum support and confidence levels the genes in the antecedent of our association rule will lead to the consequent of our association rule. In this context that means that given a certain set of expressed genes in the top 10% another set of genes will also be expressed in the top 10% of all patients in our dataset.

We have discovered association rules related to both case and control or patients with AD and patients without AD. In comparing these association rules we can learn more about gene expression and compare how gene expression (within these 1000 genes) functions for people with AD and without. This data provides valuable information on gene expression pathways that could be activated in the progression of AD that do not exist in non-AD patients which could perhaps help provide targeted treatments for AD patients.

4) In order to find independent features and features that are dependent on these independent features we can look at the frequent itemsets and compare them to the association rules. An association rule consists of an antecedent and a consequent, which are related to each other by the statement if an antecedent (frequent itemset) appears in a transaction then with a probability equal to confidence value the consequent (another frequent itemset) will also appear in this transaction. Another way to think of this is as an if-then statement. The antecedent will be the if and the consequent is the then so we can say with confidence that if the antecendent (set of genes) appears then the consequent will appear in a transaction (patient). If we want to find independent features and the features that are dependent on these we can look at genes or sets of genes in the frequent itemsets that appear in the antecedent that do not appear in the consequent of any association rules. This implies that this feature is independent because it is not the consequent or dependent on any other feature or group of features.

Psuedocode for this processs is described as follows.

assuming we have a list of association rules stored as association rules

for antecedents in association\_rules:

make a unique (contains one listing of each gene) list of all of the genes present in antecedents in the given association rules

for consequents in association\_rules:

make a unique (contains one listing of each gene) list of all the genes present in the consequents of the given association rules

for antecedents in antecendent\_list:

counter = 0

for consequents in consequent\_list:

if antecendent == consequent:

consequent += 1

if count = 0:

add antecedent to list of independent genes

This pseudocode will find all independent genes or genes that appear only in the antecedent of association rules which means that these genes are not dependent on other genes to appear.

If I want to find the genes that are dependent on these independent genes (genes that only appear in the antecedent of association rules) then I can simply look through my top association rules and find genes that are in the consequents of association rules containing these independent genes. Pseudocode for this follows.

for genes in independents\_genes:

for rule in association\_rules:

if gene is in association\_rule.antecedent:

add genes in consequent to dependent\_genes\_list

return dependent\_genes\_list

5) After translating the above pseudocode into a python script I ran it on my top association rules for both case and control (AR with confidence > 80% from top 50 most frequent item sets with support > 6%). Listed below are the independent genes for the case dataset.



For these genes I found that the following genes were dependent or the associated redundant features (genes) in the case dataset.



Below are the independent genes for the control dataset.



For the control dataset I found the following genes were dependent or the associated redundant features (genes).

