

# Comparison of Association Rule Learning applied to Diabetes

Mittelman  
CS 2051  
Georgia Tech  
Atlanta, Georgia

Hellman  
CS 2051  
Georgia Tech  
Atlanta, Georgia

Banerjee  
CS 2051  
Georgia Tech  
Atlanta, Georgia

**Abstract**—Association rule learning is a method to find relationships and correlation within data. The Apriori Algorithm is a very common method for finding relationships by looking at items that are repeated many times throughout a dataset. In this paper we explore the Apriori algorithm and look at other measures of correlation such as conviction to see the differences in their results when attempting to diagnose diabetes.

## I. INTRODUCTION

Association Rule Learning is the process of mining data for associations between subsets of items. This means finding rules that allow us to deduce "if this set of items is in this set we can reasonably expect this other set of items to be in this set". This is very abstract so it makes sense to use a concrete example.

The standard example for this is analyzing transactions at a super market. A super market has a dataset of all of its sales and it can run an Association Rule Learning algorithm to find what things are commonly bought together. For example, it might find if someone buys toothbrushes they are highly likely to buy floss so it is a good idea to place them next to each other in the store.

Also note that these techniques don't just apply to supermarkets but many data sets can be treated in this way. For example: apps might find content that is rated highly by a group of people and so if somebody watches one video the other video should be recommended.

Now we will explain some measures that can be used to find associations.

For these examples let:

set of items  $I = i_1, i_2, \dots, i_m$

set of transactions  $T = t_1, t_2, \dots, t_n$ , where each transaction is a set of items

Our goal is to find rules of the form  $X \rightarrow Y$ , where  $X$  and  $Y$  are subsets of  $I$ , that are considered to be strongly associated within the transactions. Meaning if we find the elements of set  $X$  in a transaction we can reasonably expect the set of items in  $Y$  to be there too. Going to our supermarket example the set of items would be everything the grocery store sells and the set of transactions would be receipts from customers buying items.

- **Support:** This measures how frequently these two items are bought in general it is calculated as the fraction of transactions that have  $X$  and  $Y$ . In probability notation:  $P(X \in t \wedge Y \in t)$

$$\text{Support}(X \rightarrow Y) = \frac{|t \in T \mid X \subseteq t \text{ and } Y \subseteq t|}{|T|}$$

Items	Frequency	Support
Eggs	5	5/30=0.166
Bacon	8	8/30=0.266
Milk	10	10/30=0.333
Cheese	4	4/30=0.133
Juice	3	3/30=0.1
Total	30	1.0

- **Confidence:** This measures how confident we can be that if  $X$  appears then  $Y$  appears. It's calculated as the fraction of transactions containing  $X$  that also contain  $Y$ . Note that this is not commutative. In probability notation  $P(Y \in t \mid X \in t)$

$$\text{Confidence}(X \rightarrow Y) = \frac{|t \in T \mid X \subseteq t \text{ and } Y \subseteq t|}{|t \in T \mid X \subseteq t|}$$

Receipt 1	Milk, eggs, juice
Receipt 2	Bacon, milk, chicken
Receipt 3	Milk, eggs
Receipt 4	Juice, chicken, bacon
<p>Confidence(Milk <math>\rightarrow</math> Eggs) = (Number of receipts with milk and eggs) / (Number of receipts with milk) = 2/3</p> <p>Confidence(Eggs <math>\rightarrow</math> Milk) = (Number of receipts with milk and eggs) / (Number of receipts with eggs) = 1</p>	

- Lift: When developing an association  $X \rightarrow Y$  it is helpful to know if X appearing actually makes Y more likely to appear. This is what lift does it is calculated as the ratio of Confidence to Support. Lift is commutative.
  - if lift  $> 1$  that means X appearing makes Y more likely to appear
  - if lift  $= 1$  that means it has no affect
  - if lift  $< 1$  it means X appearing makes Y less likely to appear.

$$\text{Lift}(X \rightarrow Y) = \frac{\text{Confidence}(X \rightarrow Y)}{\text{Support}(Y)}$$

Receipt 1	Milk, eggs, juice
Receipt 2	Bacon, milk, chicken
Receipt 3	Milk, eggs
Receipt 4	Juice, chicken, bacon
<p>Lift(Chicken <math>\rightarrow</math> Juice) = Confidence (Chicken <math>\rightarrow</math> Juice) / Support(Juice) = 0.5/0.5 = 1</p> <p>Lift(Eggs <math>\rightarrow</math> Milk) = Confidence(Eggs <math>\rightarrow</math> Milk) / Support(Milk) = 1/0.75=1.333</p>	

- Conviction: Measures the ratio between the probability an element B does not appear in general and the probability B does not appear if A is present. Unlike Lift Conviction is directed meaning Conviction from A to B is not equal to Conviction from B to A. Just like lift:

- if conviction  $> 1$  that means X appearing makes Y more likely to appear
- if conviction  $= 1$  that means it has no affect
- if conviction  $< 1$  it means X appearing makes Y less likely to appear.

$$\text{Conviction}(A \rightarrow B) = \frac{1 - \text{Support}(B)}{1 - \text{Confidence}(A \rightarrow B)}$$

Receipt 1	Milk, eggs, juice
Receipt 2	Bacon, milk, chicken
Receipt 3	Milk, eggs
Receipt 4	Juice, chicken, bacon
<p>Conviction(Bacon <math>\rightarrow</math> Milk) = (1 - Support (Milk))/(1-Confidence(Bacon <math>\rightarrow</math> Milk)) = (1-0.75)/(1-0.5) = 0.5</p> <p>Conviction(Milk <math>\rightarrow</math> Bacon) = (1 - Support (Bacon))/(1 - Confidence(Milk <math>\rightarrow</math> Bacon)) = (1-0.5)/(1-0.333) = 0.75</p>	

These four values allow us to measure associations in a large data set, but these individual measures can't give us any data without a way to apply them. A store owner can't apply conviction to every possible combination of items to find correlation. This is where algorithm come in, specifically we will talk about Apriori algorithm which uses support to find correlations in datasets.

## II. MAIN RESULT

### A. Apriori Algorithm

Apriori algorithm is an unsupervised algorithm, meaning it is given a dataset and simply operates on it with no idea what the output should be as opposed to a supervised algorithm which must be given human labeled data to operate. It is used for finding commonly occurring sets of items in a dataset, as described in the supermarket example from the previous section.

On a high level, the way Apriori works is it first identifies individual items that appear frequently in the dataset, and gradually builds bigger itemsets that appear frequently by adding more items. The algorithm sets a minimum support threshold, with which it decides if items from the previous iteration are continued on to the current iteration. It does this until it reaches a plateau, where there are no more new frequent itemsets that can be found. The resulting itemsets at the end of the algorithm can be used to draw important conclusions.

In the first iteration, Apriori will look at the individual items and see if they pass a set support threshold. If they do, then they will move on to the second iteration. In the second iteration, every possible pair of items that passed the first iteration is checked against the support threshold and this continues until no more items can be added.

The reason that this works is because adding an extra element to calculate support with can only decrease the support value. If the support of milk is 0.5, then the support of milk and eggs is less than or equal to 0.5 because support calculator the amount with both items in the transaction divided by total items. And the amount of transactions with milk and eggs is at most the amount of transactions with milk.

An example of how the algorithm operates on restaurant sales with a support threshold of 0.6 is shown on the right:

Apriori applied to restaurant sales		
1st Iteration		
Items	Support	Decision
Burger	0.65	✓
Fries	0.5	✗
Pop	0.8	✓
Hotdog	0.6	✓
Any items with a support over or equal to 0.6 are kept and paired to another item		
Apriori applied to restaurant sales		
2nd Iteration		
Items	Support	Decision
Burger/Pop	0.6	✓
Burger/Hotdog	0.3	✗
Hotdog/Pop	0.4	✗
There are no more elements to be added for a third iteration because only one pair passed the support threshold, meaning the algorithm stops here.		

### III. APRIORI ALGORITHM AND CONVICTION APPLIED TO DIABETIC INDICATORS

#### A. Our Dataset

For our extension we are using a **dataset** that contains a lot of information from people with and without diabetes. The data includes information such as blood pressure, cholesterol level, smoking history, physical activity, and others. We plan on using Apriori Algorithm on this data to try and find correlation between these health indicators and whether or not the person has diabetes. This is useful and will give us groups with a high correlation, but if we want to see how much a certain indicator can predict whether or not a person will have diabetes then conviction should be used.

Below are some images of the responses within the dataset:

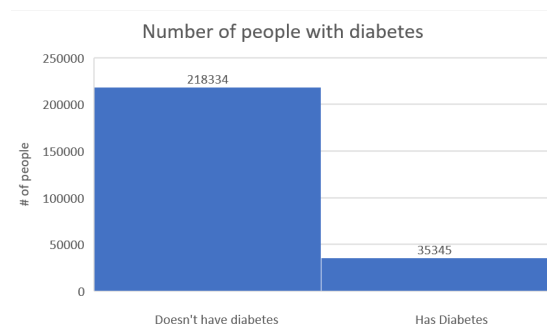


Fig. 1. We have a large amount of people in both categories so this data set does not fail due to small size

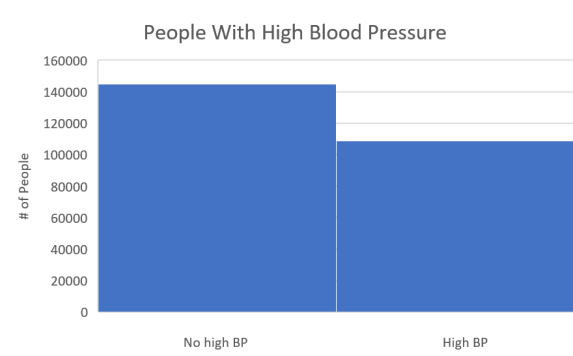


Fig. 2. Categorical data works very well with Association Rule learning which is why this dataset can be applied to conviction and Apriori algorithm.

#### B. Cleaning the Dataset

Most of the data was already ready to use because it was binary questions. For example, do you have high blood pressure, this could either be a 0, no, or a 1, yes. This works perfectly with Apriori algorithm and Conviction which need binary data. However, we also needed to clean some of the data, since some of it was not categorical. For example, one survey question asked how many days they felt they had poor mental health. Answers could range from 0-30 but we needed categorical data, so we counted anything above 0 as a yes mental health problems and 0 as a no mental health problems.

Conviction measures the expected amount that certain elements should appear without other elements compared to the actual amount they actually appear without another element. In the supermarket example, a conviction of 1.8 from milk to eggs means that eggs appears in a much higher amount of milk transactions than expected. This is a directed measure meaning the conviction from milk to eggs can be different than the conviction from eggs to milk.

This is really useful in our case because we can apply conviction to certain health indicators and see if they show up much more in people with diabetes than expected. We plan on calculating the conviction between the health indicators within the dataset and whether or not the person has diabetes and seeing if they are any values that are very low or very high. These can show strong correlation within the data specifically from one indicator.

Our goal from this is to show that certain Association Rules are much more useful in specific circumstances. When applying Apriori Algorithm we won't see directional movement from indicators to diabetes and will instead only see strong correlation between certain pairs or groups. But conviction can show us directed movement from an indicator to a diagnosis which is much more useful when looking at causes or effects or diabetes. Conviction is also better than confidence because conviction compares our results with the expected amount, meaning instead of a cutoff like we would need for confidence, we can just check if conviction is greater than 1.

#### D. Our version of Apriori and Conviction

Our goal is to find links between people who have diabetes and specific health indicators. But because the Apriori algorithm only looks at support and in our dataset most people don't have diabetes, we needed to make some changes to the Apriori Algorithm. Otherwise, diabetes would never be linked to any indicator due to it having low support.

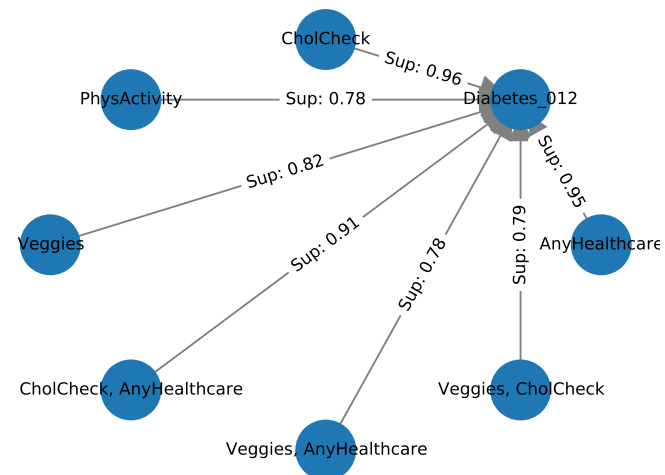
To fix this issue we removed all of the non-diabetics from the dataset and then used the Apriori Algorithm. This means that Apriori will just select health indicators that have a high support from people who have diabetes. We also created our own algorithm using conviction, this is how it worked:

- 1) Calculate conviction on every individual health indicator to diabetes and store every indicator with a conviction higher than 1.01, otherwise some values that were too close to 1 could be kept, even though they were much closer to neutral than correlated.
- 2) Then add another health indicator and see if that would raise or lower conviction
- 3) If conviction went up, keep that set to return as one that implies diabetes and repeat. Regardless, keep this set to try supersets of, because even though a subset might not have high enough conviction it's superset might.
- 4) Repeat until largest possible set is found

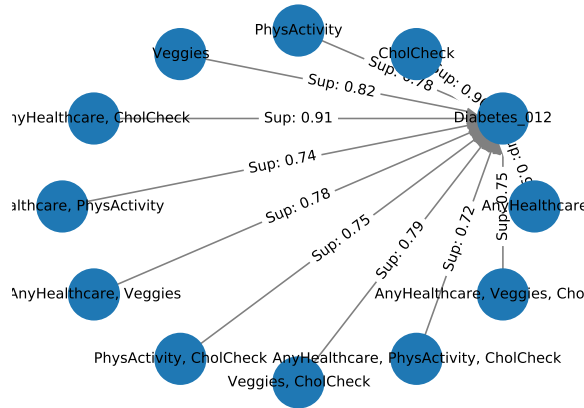
#### A. Apriori

We used the aforementioned Apriori's algorithm with varying support thresholds. The following diagram is a graph with edges directed from the antecedent groups to consequent groups. We set the confidence and lift to zero in the library implementation we used because they were not included as metrics in the default version of Apriori's algorithm that we analyzed.

It can be seen that there is some correctness to the results, with there being correlations between cholesterol checks and diabetes, and healthcare and diabetes, for example. However, there are also some questionable results, for example with veggies and physical activity being antecedents to diabetes. Note that the results in the graph correspond to Apriori's with a minimum support of 0.76, on the entire dataset of roughly 250,000 people.



Here is the same graph but with a minimum support of 0.70:



### B. Conviction Algorithm

In the first iteration of our algorithm we found that the following had a conviction higher than 1.01 with diabetes:

- High Blood Pressure
- High Cholesterol
- History of smoking
- Has had a stroke
- Has heart disease or had a heart attack
- Low physical activity
- No fruit in diet
- No vegetables in diet
- Trouble affording doctor visit
- Reporting good general health
- Poor mental health
- Poor physical health
- Difficulty walking
- Being a man
- Being 60-79 years old
- Not having a Bachelor's degree
- Low Income

We unfortunately could not run further iterations due to limited computing power and the large number of potential combinations, but even this initial iteration shows the power of our algorithm. Had we run more we could have found factors that alone don't imply diabetes but together they do.

### C. Comparing the two algorithms

We can see that the Apriori algorithm definitely left in some more questionable results, such as eating vegetables and having healthcare. Neither of these were kept in our conviction algorithm, and our conviction algorithm found symptoms that apriori didn't such as high blood pressure, high cholesterol and old age, which are notable. While this can't definitely put our algorithm above Apriori, we can say in this case that conviction gave us data closer to what was expected. Also Apriori algorithm requires a support threshold to be set which can make changes to the data

depending on the value used, so Apriori could be less consistent. The conviction algorithm sets a flat threshold of 1.01 which means it is less prone to change depending on the user's parameters.

## APPENDIX

### A. Code: Hyperlink

- [Apriori's Algorithm using support](#)
- [Conviction Algorithm](#)

## REFERENCES

- [1] Dobilas, S. (2021, July 10). Apriori algorithm for Association Rule learning - how to find clear links between transactions. Medium. Retrieved April 9, 2023, from <https://towardsdatascience.com/apriori-algorithm-for-association-rule-learning-how-to-find-clear-links-between-transactions-bf7ebc22cf0a>
- [2] Kim, C. (2022, September 6). Market basket analysis with association rules and network Graphing with python. Medium. Retrieved April 9, 2023, from <https://medium.com/@chyun55555/market-basket-analysis-with-association-rules-and-network-graphing-with-python-96319585fd27>
- [3] Teboul, A. (2021, November 8). Diabetes health indicators dataset. Kaggle. Retrieved April 9, 2023, from <https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset>