NYC DATA SCIENCE ACADEMY

Association Rules & Naïve Bayes

Data Science Bootcamp

Outline

- Part 1: Association Rule Mining
- Part 2: Naïve Bayes
- Part 3: Review

PART 1

Association Rule Mining

Association Rule Mining

- Association rule mining is an unsupervised methodology for finding relationships and patterns among different variables in a dataset; it is particularly useful in assessing large databases because of its (eventual) computational efficiency.

- Because the algorithm is unsupervised, there is no need for it to be trained. While the method can be unleashed on data straight away, there is no objective way to measure performance aside from measuring qualitative usefulness.

- The main goal is to construct a set of rules that imply some sort of structure in the data. Generally, these rules take the form of “if X, then Y.”

Applications of Association Rule Mining

- Potential applications of association rule mining are quite widespread. Some examples are:
  - Genetics: evaluating frequently occurring DNA and protein sequences in the analysis of disease genomics.
  - Fraud: identifying patterns of medical claims or user purchases that appear alongside fraudulent insurance and credit card usage.
  - Subscription: assessing consumer behavior prior to purchasing or dropping a subscription to a premium service.
  - Market Basket Analysis: showing the relationship between items that a customer purchases.

- We will focus on the application to the market basket analysis framework; however, this methodology can be applied across all disciplines.

Market Basket Analysis

- Often association rule mining is synonymous with market basket analysis because of its clear application to the shopping framework.
    - A typical rule might be: if someone buys peanut butter and jelly, then that person is likely to buy bread as well.

- Retailers can use this information to inform decisions about their:
    - Marketing strategies: target customers that buy peanut butter and jelly with offers on bread in order to get them to spend more.
    - Store layout: place peanut butter, jelly, and bread relatively close to one another among the shelves in the aisle.

Dataset Complexity

- Sticking with the market basket framework, transactional data can be quite complex; the massive size of transactional datasets is the threat of market basket analysis.
  - ➢ The number of transactions (rows in our dataset) is likely to be large.
  - ➢ The number of available items (columns in our dataset) is likely to be large.

- The different combination of items that can appear in a specific purchase increases exponentially with the number of items available for purchase.
  - ➢ Given k different items in a store, there are (2^k - 1) different possible combinations of those items appearing in purchases.

- How do we get around dealing with this complexity?

The *A priori* Belief

- Rather than evaluating each of the individual purchase combinations one-by-one, the Apriori algorithm takes into account one big observation:
  - In reality, many of the potential combinations of items rarely (if ever) manifest in the data.

- By straightaway ignoring the rarer purchase combinations, it is possible to limit the scope of the problem to a more manageable realm of exploration.

- The Apriori algorithm employs a simple *a priori* belief as a guideline for reducing the association rule space. The notions boil down to:
  - All subsets of a frequent purchase must also be frequent.
  - All supersets of a infrequent purchase must also be infrequent.

An Overview of Notation

- A rule is typically written with set notation. A simple rule might look like the following: $\{X_i, X_j\} \rightarrow \{X_k\}$
  - $\{X_i, X_j\}$ represents the left hand side or LHS.
  - $\{X_k\}$ represents the right hand side or RHS.

- In the market basket framework, the statement made by this rule can be read as: “if the user purchases X_i and X_j, they are likely to purchase X_k as well.”

- Our previous example could be written in this notation as follows:
  - {Peanut Butter, Jelly} → {Bread}
  - Interpretation: Should a transaction include the items peanut butter and jelly, we should expect to see bread in the transaction more often than not.

An Overview of Terminology

- How can we determine whether an association rule is deemed interesting? This is usually a function of looking at three different parameters that surround a particular rule:
    - Support: the fraction of which each item appears within the dataset as a whole.
    - Confidence: the likelihood that a constructed rule is correct given the items on the left hand side of the transaction.
    - Lift: the ratio by which the confidence of a rule exceeds the expected outcome.

An Overview of Terminology: Support

- The support of an item X is the fraction of which each item appears within the total number of transactions:

$$Support(X) = \frac{Count(X)}{N}$$

- Generally speaking, it is desirable to identify rules that have a high support:
  - These rules will be applicable to a large number of transactions and are generally more interesting and profitable to evaluate from a business standpoint.
  - Rules with low support may appear by chance, and can be uninteresting or more costly to spend time evaluating.

An Overview of Terminology: Support Example

- Consider the following dataset of transactions:

Transaction	Flowers	Get Well Card	Soda	Stuffed Animal	Balloons	Candy Bar
#1	1	1	1	0	0	0
#2	1	0	0	1	1	1
#3	1	1	0	0	0	1
#4	0	0	1	1	1	0
#5	1	1	1	0	0	0

- The support of {Candy Bar} = ⅖ = 0.4.
  - The combination {Candy Bar} appears in 40% of the transactions.
- The support of {Flowers, Get Well Card} = ⅗ = 0.6.
  - The combination {Flowers, Get Well Card} appears in 60% of the transactions.

An Overview of Terminology: Confidence

- The confidence of a rule is the likelihood that a constructed rule is correct given the items on the left hand side of the transaction. In other words, it is the probability that the transaction also contains the items on the right hand side:

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

- Confidence measures the accuracy or reliability of inference made by the given rule. A higher level of confidence implies a higher likelihood that Y appears alongside transactions in which X appears.

- **NB**: Be careful with interpreting the confidence of an association rule. A high confidence merely suggests a strong association (co-occurrence) between items, not a causal relationship.

An Overview of Terminology: Confidence Example

- Consider the following dataset of transactions:

Transaction	Flowers	Get Well Card	Soda	Stuffed Animal	Balloons	Candy Bar
#1	1	1	1	0	0	0
#2	1	0	0	1	1	1
#3	1	1	0	0	0	1
#4	0	0	1	1	1	0
#5	1	1	1	0	0	0

- The confidence of {Flowers} → {Get Well Card} = 0.6/0.8 = 0.75.
  - The proportion of transactions where {Flowers} resulted in the presence of {Get Well Card} is 75%.
- The confidence of {Get Well Card} → {Flowers} = 0.6/0.6 = 1.00.
  - The proportion of transactions where {Get Well Card} resulted in the presence of {Flowers} is 100%.

An Overview of Terminology: Lift

- The lift of a rule is the ratio by which the confidence of a rule exceeds the expected outcome:

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

- In other words, it is the ratio of the support of the items on the LHS of the rule co-occurring with the items on the RHS divided by the probability that the LHS and RHS co-occur if the two are independent.

- When lift > 1, the presence of X seems to have increased the probability of Y occurring in the transaction.
- When lift < 1, the presence of X seems to have decreased the probability of Y occurring in the transaction.
- When lift = 1, X and Y are independent.

An Overview of Terminology: Lift Example

- Consider the following dataset of transactions:

Transaction	Flowers	Get Well Card	Soda	Stuffed Animal	Balloons	Candy Bar
#1	1	1	1	0	0	0
#2	1	0	0	1	1	1
#3	1	1	0	0	0	1
#4	0	0	1	1	1	0
#5	1	1	1	0	0	0

- Using the support formula:
  - The lift of {Flowers} → {Get Well Card} = 0.6/(.8 * .6) = 1.25.
- Using the confidence formula:
  - The lift of {Flowers} → {Get Well Card} = 0.75/.6 = 1.25.
  - The lift of {Get Well Card} → {Flowers} = 1.0/0.8 = 1.25.
- The items are found together 1.25 times more often than one would expect by chance.

The Apriori Algorithm

- Recall that the *a priori* belief was the solution to combing through a massive dataset of transactions because it takes advantage of the notion that:
  - All subsets of a frequent purchase must also be frequent.
  - All supersets of a infrequent purchase must also be infrequent.

- The Apriori algorithm that takes this frequency notion into account can be broken down into the following two steps in order to identify interesting rules:
  - Identify all subsets of items in the dataset that meet a minimum support threshold.
  - Create rules from these subsets that meet a minimum confidence threshold.

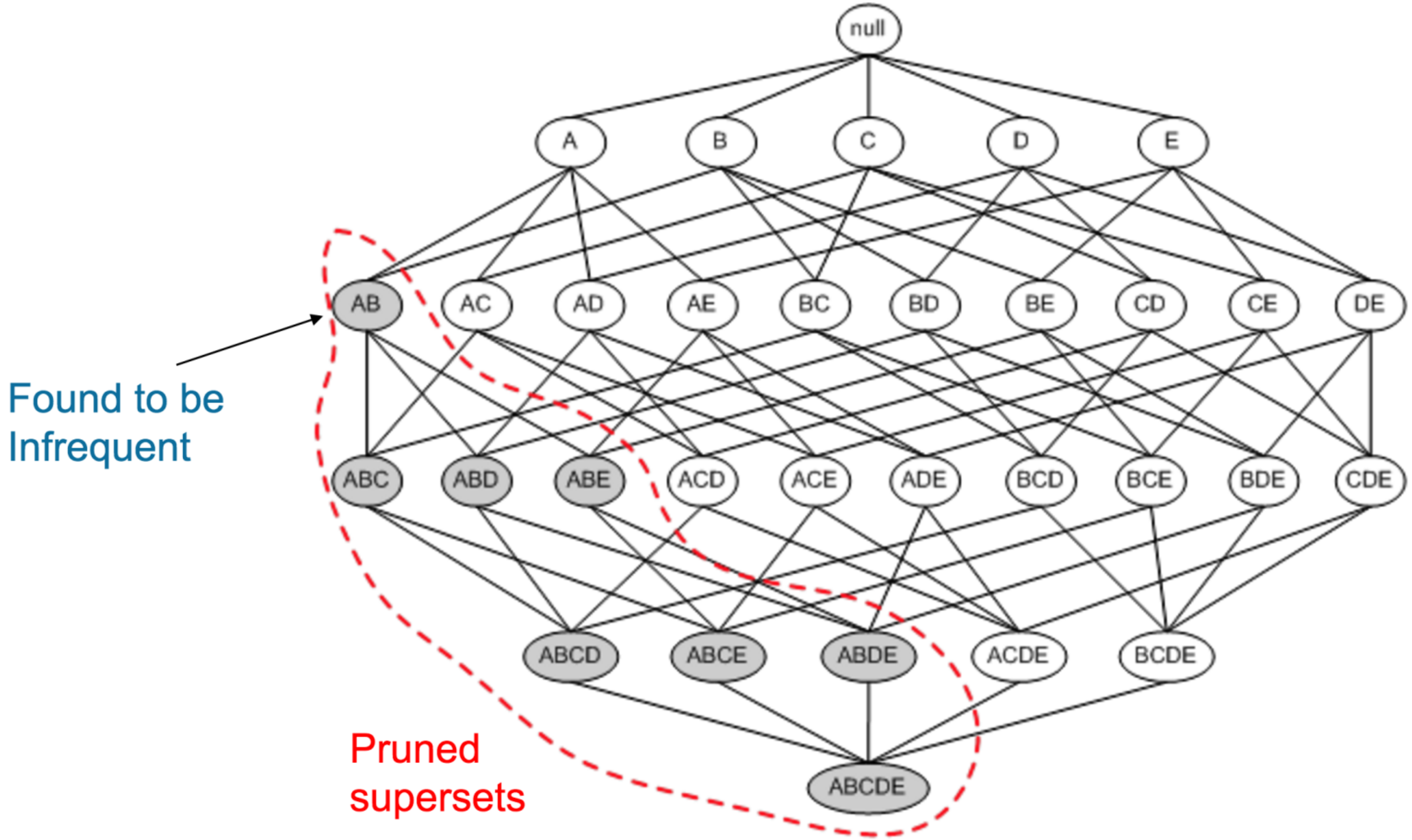
- After defining the minimum support and confidence, the possible rules to be traversed in the overall combinations can be limited in an efficient manner.

The Apriori Algorithm

- The first phase of the Apriori algorithm takes place in multiple iterations; each successive iteration evaluates the support of larger and larger sets of items:
  - Iteration 1 involves looking at the support of 1-item subsets.
  - Iteration 2 involves looking at the support of 2-item subsets.
  - Etc.; but don't we still end up looking at all subsets?

- The *a priori* belief can help us eliminate many combinations. Suppose we have items A, B, C, D, and E:
  - Iteration 1 finds the support of {A}, {B}, {C}, {D}, and {E} to surpass the threshold.
    - Now consider all 2-item subsets in iteration 2: {A, B}, {A, C}, …, {D, E}.
  - Iteration 2 finds the support of {A, B} to not surpass the threshold.
    - Now consider all 3-item subsets that do not include {A, B}: {A, C, D}, {A, C, E}, …, {C, D, E}.

The Apriori Algorithm: Visually



Tuning Support & Confidence

- Given the unsupervised nature of the association rule problem, there is no steadfast guideline for deciding the minimum support and confidence.

- It is important to recognize that different businesses and realms will have extremely varied distributions of items and therefore completely varied types of transactions.
  - Dependent on these distributions, different values of support and confidence will be applicable.

- High support and confidence will reduce the total number of rules generated, but tend to increase the individual rule prevalence and importance; try inspecting the results yielded by high parameter values and cutting back the support and confidence if necessary.

Pros & Cons of Association Rule Mining

- Pros:
  - Association rule mining is ideally suited for working with very large amounts of transactional data.
  - The results are rules that are generally easy to understand and have a high amount of interpretability.
  - The process is useful for data mining and uncovering unexpected knowledge within a dataset.

- Cons:
  - The outcome is usually not interesting when applied to smaller datasets.
  - It is difficult to separate actual insights from common sense notions.
  - The analyst might be compelled to draw spurious conclusions -- remember that correlation doesn't imply causation!

PART 2

Naïve Bayes

Naïve Bayes

- Naïve Bayes is a probabilistic supervised classification method concerned with describing uncertainty.

- In a nutshell, the Naïve Bayes method employs a frequentist perspective on data analysis; it uses information about prior events to estimate the probability of future events.

- The method is particularly useful when your data has many categorical variables with many possible values, or if you are concerned with the interrelated nature of your variables.
  - Many algorithms tend to ignore features that have weak effects on the outcome; Naïve Bayes utilizes all available evidence to make a prediction.
  - The idea is that many small effects "added together" could ladder up to have a meaningful impact.

Independent & Dependent Events

- If all events in the world were independent of one another, it would be impossible to accurately predict any future event based on data collected from another event.
  - In other words, if two events are independent, knowledge of one event does not inform knowledge of the other event.

- On the other hand, dependencies among events form the basis of predictive modeling; it can be helpful to use these dependencies in order to predict future events.
  - In other words, if two events are dependent, knowledge of one event does inform knowledge of the other event.

Conditional Probability: Bayes' Theorem

- The relationships between dependent events can be described using Bayes' Theorem. The conditional probability of of event A given event B is as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(A \cap B)}{P(B)}$$

- In other words, Bayes' Theorem states that the posterior is proportional to the likelihood times the prior.

- Unsurprisingly, Bayes' Theorem represents the overall underpinnings of the Naïve Bayes classifier.
  - How does this theorem work?

Conditional Probability: Bayes’ Theorem

- Suppose you wanted to predict whether or not an incoming email message was spam:
  - With no knowledge of the incoming email, the best guess you could give is the probability that any previous email was spam (i.e., the prior probability).

- What if you knew the incoming email contained the word “Viagra”? That knowledge should theoretically change the probability of the message being spam because of dependencies among the variables:
  - The probability that the word Viagra was used in previously observed spam messages is called the likelihood.
  - The probability that Viagra appeared in any given message is called the marginal likelihood.

Conditional Probability: Bayes' Theorem

- By implementing Bayes' Theorem, we can compute the posterior probability that measures the likelihood of the new message being spam given the evidence that the email contained the word Viagra:

$$P(spam|Viagra) = \frac{P(Viagra|spam)P(spam)}{P(Viagra)}$$

- If the posterior probability is greater than 50%, then it is more likely than not that the email is spam.
- If the posterior probability is less than 50%, then it is more likely than not that the email is not spam.

Conditional Probability: Bayes' Theorem

- Consider the following dataset of emails:

	Viagra	No Viagra	**Total**
Spam	4	16	20
Not Spam	1	79	80
Total	5	95	100

- What is the posterior probability that an email is spam given that it contains the word Viagra?
  - The likelihood P(Viagra | spam) = 4/20 = 0.2.
  - The prior P(spam) = 20/100 = 0.2.
  - The marginal likelihood P(Viagra) = 5/100 = 0.05.
  - The posterior probability P(spam | Viagra) = (0.2 * 0.2)/0.05 = 0.8; the probability that the message is spam given that it contains the word Viagra is 80%.

The Naïve Bayes Algorithm

- The Naïve Bayes algorithm uses Bayes' Theorem in order to perform classification. It is considered "naïve" because it makes a couple of unrealistic assumptions about the data at hand:
  - All of the features in the dataset are equally important.
  - All of the features in the dataset are independent of one another.

- Luckily, even though these assumptions are rarely true in real applications, the Naïve Bayes algorithm still performs quite well given assumption violations.
  - Because of its ease of application and versatility, Naïve Bayes is generally a good "quick and dirty" classification method.
  - Sometimes, as long as the resulting class labels are accurate, it is not as important to obtain a precise estimate of probability (e.g., spam filtering).

- Why do we need these assumptions?

The Naïve Bayes Algorithm

- Suppose we were looking at the presence of three different words when trying to classify an email as spam or not spam: W_1, W_2, and W_3. Applying Bayes' Rule, we would have:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1 \cap W_2 \cap W_3|spam)P(spam)}{P(W_1 \cap W_2 \cap W_3)}$$

- This formula can get unwieldy very quickly and be extremely computationally difficult to solve.
    - As more features are added, it is necessary to keep track of the probabilities of all combinations of intersecting events.
    - Extremely large training datasets would be required to ensure that all possible combinations are represented.

The Naïve Bayes Algorithm

- The Naïve Bayes algorithm attempts to overcome this problem by making a couple of assumptions. If we assume class-conditional independence, then the computation becomes much simpler:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1|spam)P(W_2|spam)P(W_3|spam)P(spam)}{P(W_1)P(W_2)P(W_3)}$$

- The resulting value of this formula can be compared to the probability that the same message is not spam given the same words.
  - Whichever probability is higher dictates the group to which the email message should belong.
  - **NB:** The denominator can be temporarily ignored for classification because it is the same in both cases; however, the denominator can help convert the values to probabilities.

The Naïve Bayes Algorithm

- What happens if a specific event never occurs in our dataset? For example, suppose that in our previous example, W_2 didn't appear in any of the spam messages at all:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1|spam)*0*P(W_3|spam)P(spam)}{P(W_1)P(W_2)P(W_3)}$$

- Because probabilities are multiplied in the Naïve Bayes algorithm, this 0 probability makes the posterior probability of spam 0 as well!
  - The absence of W_2 overruled and nullified the effects of all the other evidence.
  - Even if the email message was incredibly spammy with words W_1 and W_3 appearing in every spam email, it would still be classified as not spam!

The Laplace Estimator

- One simple solution to this problem is called the Laplace estimator:
  - The Laplace Estimator is a corrective measure that adds a small amount of error to each of the counts in the frequency table of words.
  - The addition of error ensures that each resulting probability of each event will necessarily be nonzero, even if the event did not appear in the training data.

- Typically, the Laplace estimator is chosen to be 1 so that each class/feature combination is “observed” at least once.

- With the addition of the Laplace estimator, we avoid the problem of having faulty probabilities that are strictly 0 even though we minorly artificially tamper with the raw data.

Pros & Cons of the Naïve Bayes Algorithm

- Pros:
  - It is relatively simple to understand the application of the algorithm because it is based on one mathematical equation/rule.
  - Training the classifier does not require many observations, and the method also works well with large amounts of data.
  - It is easy to obtain the estimated probability for a classification prediction.

- Cons:
  - The method relies upon the faulty assumptions that the features in the dataset are independent and equally important.
  - While easily attainable, the estimated probabilities are often less reliable than the predicted class labels themselves.

PART 3

Review

Review

- Part 1: Association Rule Mining
  - Association Rule Mining
    - Applications
  - Market Basket Analysis
  - Dataset Complexity
  - The *A priori* Belief
  - An Overview of Notation
  - An Overview of Terminology
    - Support
    - Confidence
    - Lift
    - Examples
  - The Aprori Algorithm
    - Visually
  - Tuning Support & Confidence
  - Pros & Cons of Association Rule Mining

- Part 2: Naïve Bayes
  - Independent & Dependent Events
  - Conditional Probability: Bayes’ Theorem
  - The Naïve Bayes Algorithm
  - The Laplace Estimator
  - Pros & Cons of the Naïve Bayes Algorithm

- Part 3: Review