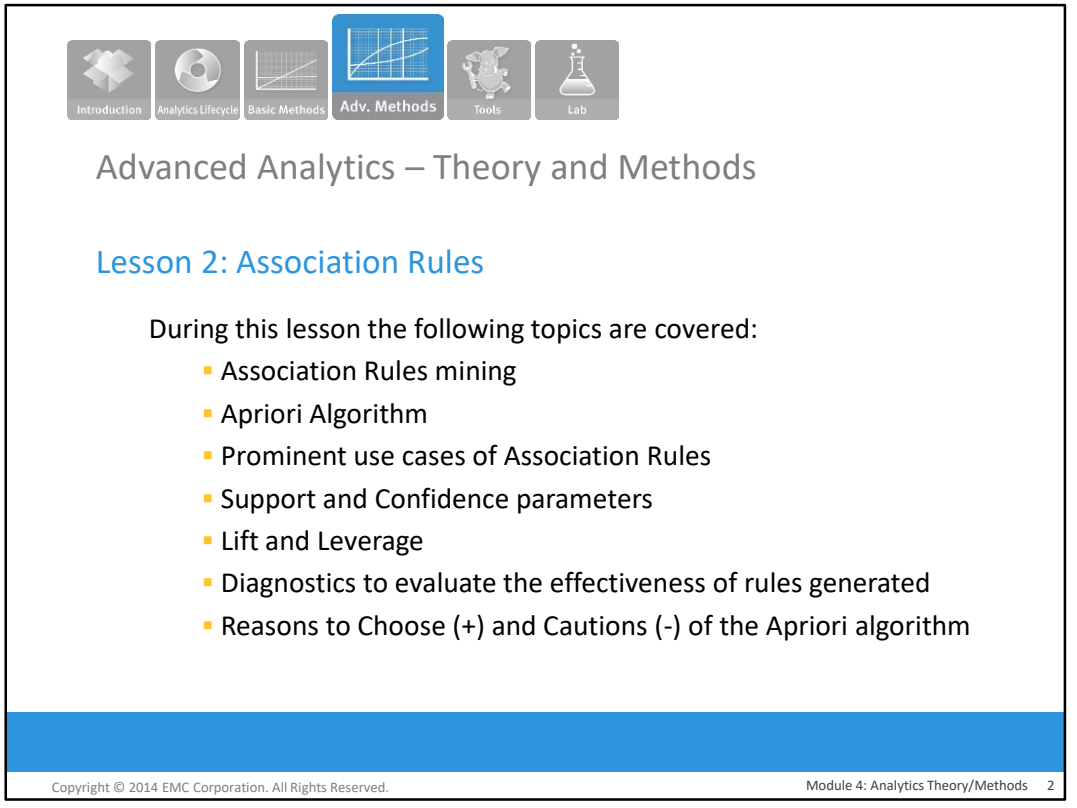
Adv. Methods

Advanced Analytics - Theory and Methods



The topics covered in this lesson are listed.

Association Rules

Which of my products tend to be purchased together?

What do other people like this person tend to like/buy/watch?

- Discover "interesting" relationships among variables in a large database
  - Rules of the form “If X is observed, then Y is also observed"
  - The definition of "interesting“ varies with the algorithm used for discovery
- Not a predictive method; finds similarities, relationships

Association Rules is another unsupervised learning method. There is no “prediction” performed but is used to discover relationships within the data.
The example questions are

- Which of my products tend to be purchased together?
- What will other people who are like this person or product tend to buy/watch or click on for other products we may have to offer?

In the online retailer example we analyzed in the previous lesson, we could use association rules to discover what products are purchased together within the group that yielded maximum LTV. For example if we set up the data appropriately, we could explore to further discover which products people in GP4 tend to buy together and derive any logical reasons for high rate of returns. We can discover the profile of purchases for people in different groups (Ex: people who buy high heel shoes and expensive purses tend to be in GP4 or people who buy walking shoes and camping gear tend to be in GP2 etc).
The goal with Association rules is to discover “interesting” relationships among the variables and the definition of “interesting” depends on the algorithm used for the discovery.
The **rules** you discover are of the form that when I observe X I also tend to observe Y.
An example of “interesting” relationships are those rules identified with a measure of “confidence” (with a value >= a pre-defined threshold) with which a rule can be stated based on the data.

Association Rules - Apriori

- Specifically designed for mining over transactions in databases
- Used over itemsets: sets of discrete variables that are linked:
  - Retail items that are purchased together
  - A set of tasks done in one day
  - A set of links clicked on by one user in a single session
- **Our Example: Apriori**

Association Rules are specifically designed for in-database mining over transactions in databases.

Association rules are used over transactions that Consists of “itemsets”.

Itemsets are discrete sets of items that are linked together. For example they could be a set of retail items purchased together in one transaction. Association rules are sometimes referred to as **Market Basket Analysis** and you can think of a itemset as everything in your shopping basket.

We can also group the tasks done in one day or set of links clicked by a user in a single session into a basket or an itemset for discovering associations.

“Apriori” is one of the earliest and the most commonly used algorithms for association rules and we will focus on Apriori in the rest of our lesson.

Apriori Algorithm - What is it?
Support

- Earliest of the association rule algorithms
- Frequent itemset: a set of items L that appears together "often enough“:
  - Formally: meets a minimum support criterion
  - **Support**: the % of transactions that contain L
- Apriori Property: Any subset of a frequent itemset is also frequent
  - It has at least the support of its superset

We will now detail the Apriori algorithm.

Apriori algorithm uses the notion of Frequent Itemset. As the name implies the frequent itemsets are a set of items “L” that appear together “often enough”. The term “often enough” is formally defined with a support criterion where the support is defined as the percentage of transactions that contain “L”.

For example:

If we define L as a itemset {shoes, purses} and we define our “support” as 50%. If 50% of the transactions have this itemset, then we say the L is a “frequent itemset”. It is apparent that if 50% of itemsets have {shoes,purses} in them, then at least 50% of the transactions will have either {shoes} or {purses} in them. This is an **Apriori property** which states that **any subset of a frequent itemset is also frequent**. Apriori property provides the basis for the Apriori algorithm that we will detail in the subsequent slides.

Apriori Algorithm (Continued)
Confidence

- Iteratively grow the frequent itemsets from size 1 to size K (or until we run out of support).
  - Apriori property tells us how to prune the search space
- Frequent itemsets are used to find rules X->Y with a minimum **confidence**:
  - Confidence: The % of transactions that contain X, which also contain Y
- Output: The set of all rules X -> Y with minimum support and confidence

Apriori is a bottom-up approach where we start with all the frequent itemsets of size 1 (for example shoes, purses, hats etc) first and determine the support. Then we start pairing them. We find the support for say {shoes,purses} or {shoes,hats} or {purses,hats}.

Suppose we set our threshold as 50% we find those itemsets that appear in 50% of all transactions. We scan all the itemsets and "prune away" the itemsets that have less than 50% support (appear in less than 50% of the transactions), and keep the ones that have sufficient support. The word "prune" is used like it would be in gardening, where you prune away the excess branches of your bushes.

Apriori property provides the basis to prune over the transactions (search space) and to stop searching further if the support threshold criterion is **not** met. If the support criterion is met we grow the itemset and repeat the process until we have the specified number of items in a itemset or we run out of support.

We now use the frequent itemsets to find our rules such as X implies Y. Confidence is the percent of transactions that contain X that also contain Y. For example if we have frequent itemset {shoes,purses, hats} and consider subsets {shoes,purses}. If 80% of the transactions that have {shoes,purses} also have {hats} we define Confidence for the rule that {shoes,purses} implies {hats} as 80%.

The output of the apriori are the rules with minimum support and confidence.

Lift and Leverage

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

$$\text{Leverage}(X \rightarrow Y) = \text{Support}(X \wedge Y) - \text{Support}(X) * \text{Support}(Y)$$

The common measures used by Apriori algorithm are **Support** and **Confidence** . We rank all the rules based on the support and confidence and filter out the most "interesting" rules.
There are other measures to evaluate candidate rules and we will define two such measures **Lift** and **Leverage**.
Lift measures how many times more often X and Y occur together than expected if they were statistically independent. It is a measure of how X and Y are really related rather than coincidentally happening together.
Leverage is a similar notion but instead of a ratio it is the difference.
Leverage measures the difference in the probability of X and Y appearing together in the data set compared to what would be expected if X and Y were statistically independent.

For more measures refer to:
http://michael.hahsler.net/research/association_rules/measures.html

Association Rules Implementations

- Market Basket Analysis
  - People who buy milk also buy cookies 60% of the time.
- Recommender Systems
  - "People who bought what you bought also purchased….".
- Discovering web usage patterns
  - People who land on page X click on link Y 76% of the time.

Listed are some example use cases with Association Rules.

Market basket analysis is an implementation of Association Rules mining that many companies use (to list a few among many) for

- Broad-scale approach to better merchandising
- Cross-merchandising between products and high-margin or high-ticket items
- Placement of product (in racks) within related category of products
- Promotional programs - Multiple product purchase incentives managed through loyalty card program

Recommender systems are used by all “on-line” retailers such as Amazon.

Web usage log files generated on web servers contain huge amounts of information and association rules can potentially give useful knowledge to the web usage data analysts.

Use Case Example: Credit Records

Credit ID	Attributes
1	credit_good, female_married, job_skilled, home_owner, …
2	credit_bad, male_single, job_unskilled, renter, …

Minimum Support: 50%

Frequent Itemset	Support
credit_good	70%
male_single	55%
job_skilled	63%
home_owner	71%
home_owner, credit_good	53%

The itemset {home_owner, credit_good} has minimum support.

The possible rules are

credit_good -> home_owner

and

home_owner -> credit_good

We present an example to detail the Apriori algorithm. We have a set of artificially created transaction records detailing several attributes of people. Let's say that we found records in which Credit_good, male_single, job_skilled, home_owner and {home_owner,credit_good} have a support of over 50%.

As the itemset {home_owner,credit_good} has a minimum support of over 50% we can state the following rules:

Credit _good -> home_owner

Home_owner-> credit_good

Let us compute the confidence and Lift

Computing Confidence and Lift

Suppose we have 1000 credit records:

	free_housing	home_owner	renter	total
credit_bad	44	186	70	**300**
credit_good	64	527	109	**700**
	108	**713**	**179**	

713 home_owners, 527 have good credit.
home_owner -> credit_good has confidence 527/713 = 74%

700 with good credit, 527 of them are home_owners
credit_good -> home_owner has confidence 527/700 = 75%

The lift of these two rules is

0.527 / (0.700*0.713) = 1.055

Consider we have 1000 credit records of individuals and the table of pair-wise attributes shows the number of individuals that have a specific attribute. We can see that among the 1000 individuals 700 have credit_good and 300 have credit_bad.

We also see among the 713 home owners 527 have good credit. The confidence for the rule

Home_owner -> credit_good is 527/713 = 74%

The confidence for the rule

Credit_good -> home owner is 527/700= 75%

The Lift is the ratio of Probability of home_owner with credit_good/probability of home_owner) x probability of credit_good

Which is 0.527/(0.700*0.713) = 1.055

The lift being close to the value of 1 indicates that the rule is purely coincidental and with larger values of Lift (say >1.5) we may say the rule is “true” and not coincidental.

A Sketch of the Algorithm

- If L_k is the set of frequent k-itemsets:
  - Generate the candidate set C_{k+1} by joining L_k to itself
  - Prune out the (k+1)-itemsets that don't have minimum support Now we have L_{k+1}
- We know this catches all the frequent (k-1)-itemsets by the apriori property
  - a (k+1)-itemset can't be frequent if any of its subsets aren't frequent
- Continue until we reach k_{max}, or run out of support
- From the union of all the L_k, find all the rules with minimum confidence

Here we formally define the Apriori algorithm.

Step 1 is identifying the frequent itemsets by starting with each item on the transactions that meet the support level. Then we grow each item set joining another itemset and determine the support for the new grown itemset.

Prune all the itemsets that do not meet the minimum support.

We repeat the growing and pruning until we reach the specified number of items in a itemset or we run out of support.

Then form rules with the union of all the itemsets that we retained that meets the minimum confidence threshold.

We will go back to our credit records example and understand the algorithm.

Step 1: 1-itemsets (L1)

- let min_support = 0.5
- 1000 credit records
- Scan the database
- Prune

Frequent Itemset	Count
credit_good	700
~~credit_bad~~	~~300~~
male_single	550
~~male_mar_or_wid~~	~~92~~
~~female~~	~~310~~
job_skilled	631
~~job_unskilled~~	~~200~~
home_owner	710
~~renter~~	~~179~~

The first step is to start with 1 element itemset and let the support be 0.5. we scan the database and count the occurrences of each attributes.

The itemsets that meet the support criteria are the ones that are not pruned (struck off).

Step 2: 2-itemsets (L2)

- Join L1 to itself
- Scan the database to get the counts
- Prune

Frequent Itemset	Count
~~credit_good, male_single~~	~~402~~
credit_good, job_skilled	544
credit_good, home_owner	527
~~male_single, job_skilled~~	~~340~~
~~male_single, home_owner~~	~~408~~
~~job_skilled, home_owner~~	~~452~~

The itemsets that we end up with at step 1 are {credit_good}, {male_single}, {home_owner} and {job_skilled}.

In step 2 we join (grow) these itemsets with 2 elements in each itemset as {credit_good,male_single}, {credit_good,home_owner}, {credit_good,job_skilled}, {male_single,job_skilled},{male_single,home_owner) and {job_skilled,home_owner} and determine the support for each of these combinations.

What survives the pruning are {credit_good,job_skilled) and {credit_good,home_owner}

Step 3: 3-itemsets

Frequent Itemset	Count
~~credit_good, job_skilled, home_owner~~	~~428~~

- We have run out of support.
- Candidate rules come from L2:
  - credit_good -> job_skilled
  - job_skilled -> credit_good
  - credit_good -> home_owner
  - home_owner -> credit_good

When we grow the itemsets to 3 we run out of support.
We stop and generate rules with results in step 2
The rules that come from step 2 are shown.
Obviously, depending on what we are trying to do (predict who will have good credit, or identify the characteristics of people with good credit), some rules are more useful than others, independently of confidence.

Finally: Find Confidence Rules

Rule	Set	Cnt	Set	Cnt	Confidence
IF credit_good THEN job_skilled	credit_good	700	credit_good AND job_skilled	544	544/700=77%
IF credit_good THEN home_owner	credit_good	700	credit_good AND home_owner	527	527/700=75%
IF job_skilled THEN credit_good	job_skilled	631	job_skilled AND credit_good	544	544/631=86%
IF home_owner THEN credit_good	home_owner	710	home_owner AND credit_good	527	527/710=74%

If we want confidence > 80%:
IF job_skilled THEN credit_good

Once we have the rules we compute the confidence for each rule. The table lists the rules and the computation of confidence.

We see that job_skilled -> credit_good has a 86% confidence.

Diagnostics

- Do the rules make sense?
  - What does the domain expert say?
- Make a "test set" from hold-out data:
  - Enter some market baskets with a few items missing (selected at random). Can the rules determine the missing items?
  - Remember, some of the test data may not cause a rule to fire.
- Evaluate the rules by lift or leverage.
  - Some associations may be coincidental (or obvious).

The first check on the output is to determine if the rules make any sense. The domain expertise provide inputs for this.

In the example of credit records we had 1000 transactions that we worked with for the discovery of rules. Let us assume that we had 1500 transactions, we can randomly select 500 transactions out of this and keep it aside as hold-out data and run the discovery of rules on the remaining 1000 transactions. The 500 records we kept aside are known as the **hold-out data**.

We can use the data as a test set and drop some items from the transactions randomly. When we run the Association rules again on the test set determine if the algorithm predicts the missing data or the items dropped. It should be noted that the some of the test data may not cause the rule to fire.

It is important to evaluate the rules with “Lift” or “Leverage”. While mining data with Association Rules several rules are generated that are purely coincidental.

If 95% of your customers buy X and 90% of customers buy Y, then X and Y occur together 85% of the time, even if there is no relationship between the two. The measure of Lift ensures “interesting” rules are identified rather than the coincidental ones.

Apriori - Reasons to Choose (+) and Cautions (-)

Reasons to Choose (+)	Cautions (-)
Easy to implement	Requires many database scans
Uses a clever observation to prune the search space •Apriori property	Exponential time complexity
Easy to parallelize	Can mistakenly find spurious (or coincidental) relationships •Addressed with Lift and Leverage measures

While Apriori algorithm is easy to implement and parallelize, it is computationally expensive. One of the major drawbacks with the algorithm is that many spurious rules tend to get generated that are practically not very useful. These spurious rules are generated due to coincidental relationships between the variables.

Lift and Leverage measures must be used to prune out these rules.

Check Your Knowledge

1. What is the Apriori property and how is it used in the Apriori algorithm?
2. List three popular use cases of the Association Rules mining algorithms.
3. What is the difference between Lift and Leverage. How is Lift used in evaluating the quality of rules discovered?
4. Define Support and Confidence
5. How do you use a "hold-out" dataset to evaluate the effectiveness of the rules generated?

Record your answers here.

Module 4: Advanced Analytics – Theory and Methods

Lesson 2: Association Rules - Summary

During this lesson the following topics were covered:

- Association Rules mining
- Apriori Algorithm
- Prominent use cases of Association Rules
- Support and Confidence parameters
- Lift and Leverage
- Diagnostics to evaluate the effectiveness of rules generated
- Reasons to Choose (+) and Cautions (-) of the Apriori algorithm

This lesson covered these topics. Please take a moment to review them.