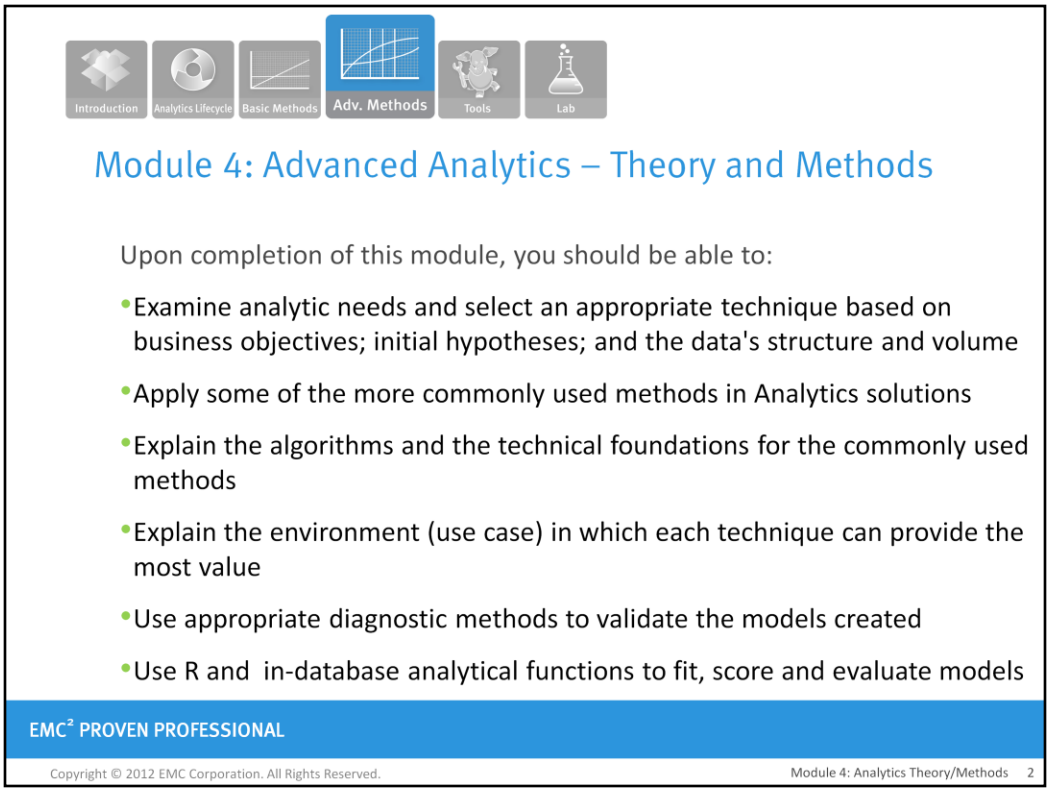
Adv. Methods

Module 4 – Advanced Analytics - Theory and Methods

EMC² PROVEN PROFESSIONAL



The objectives of this module are listed. The Analytical methods covered are:

Categorization (un-supervised) :

1.K-means clustering

2. Association Rules

Regression

3. Linear

4. Logistic

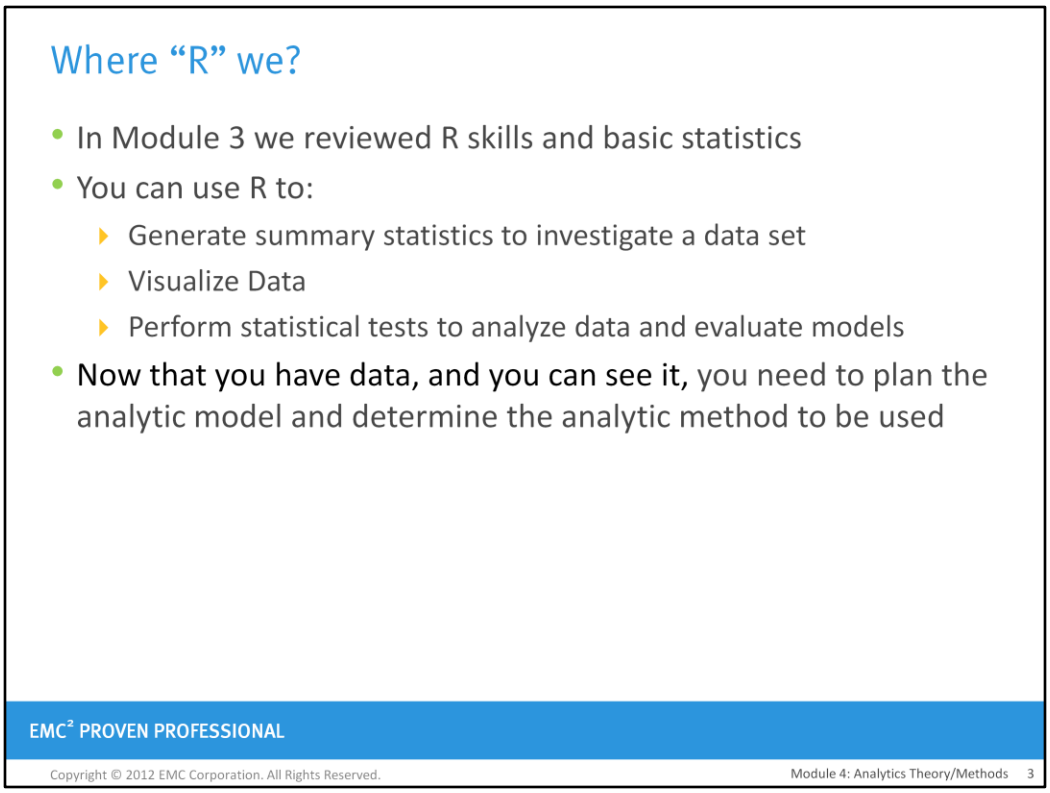
Classification (supervised)

5.Naïve Bayesian classifier

6. Decision Trees

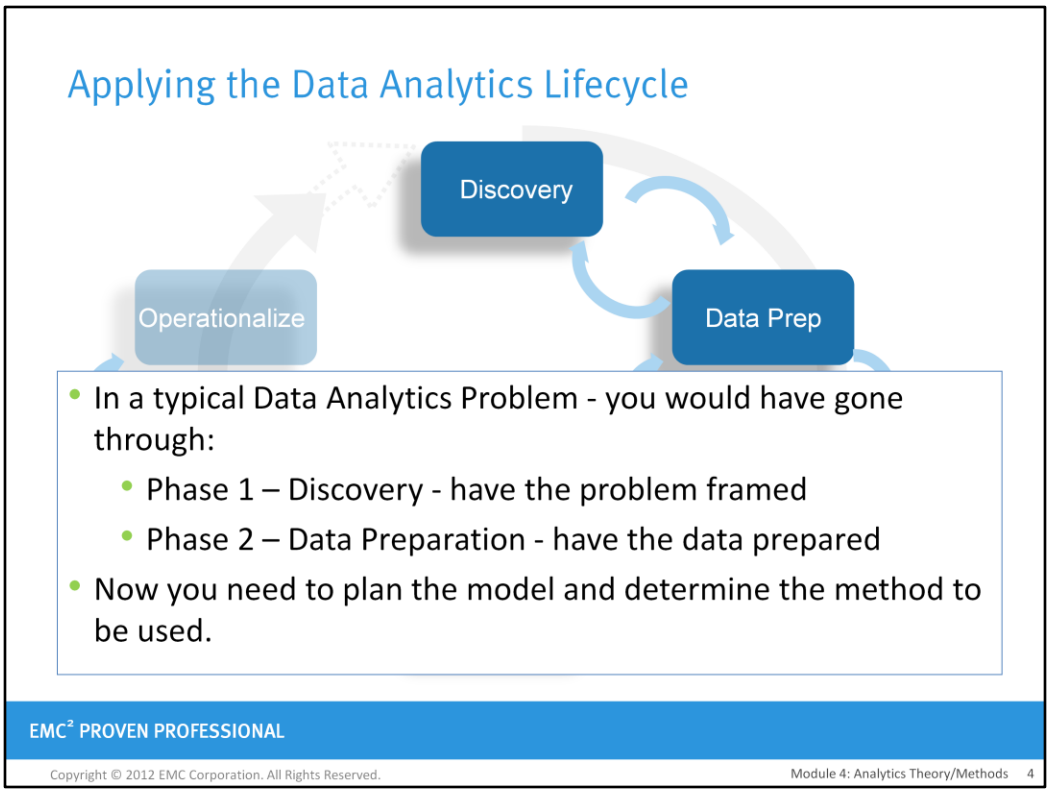
7. Time Series Analysis

8. Text Analysis

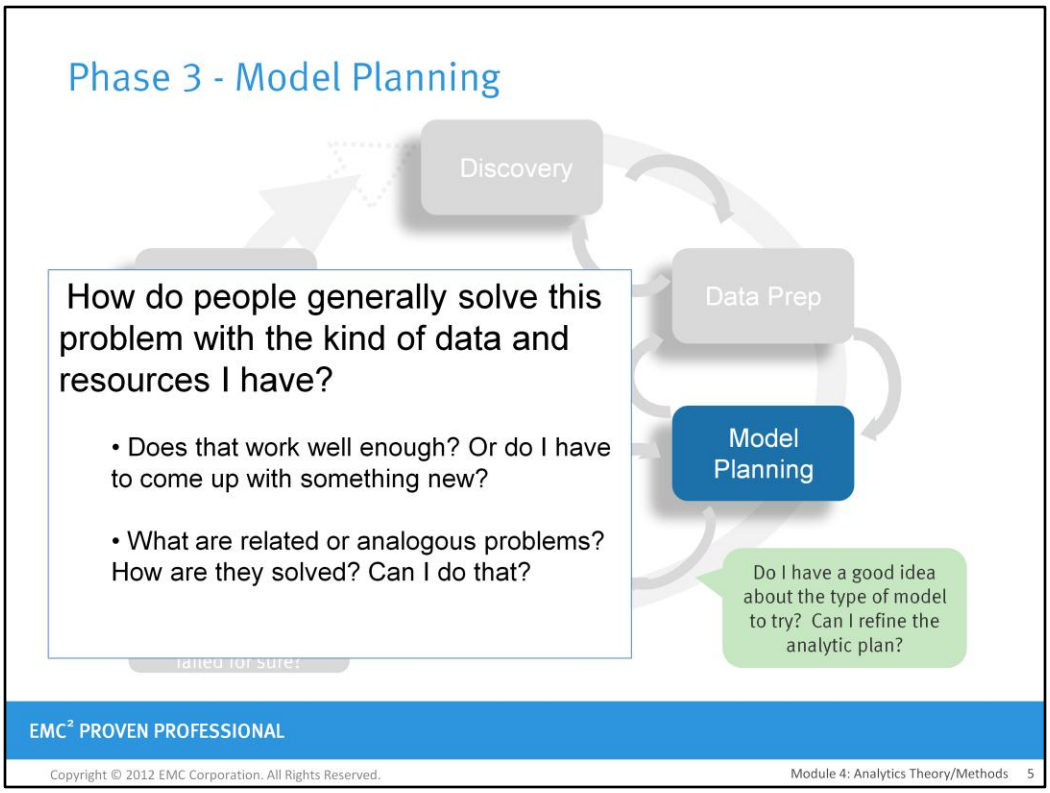


Module 4 focuses on the most commonly used analytic methods, detailing:

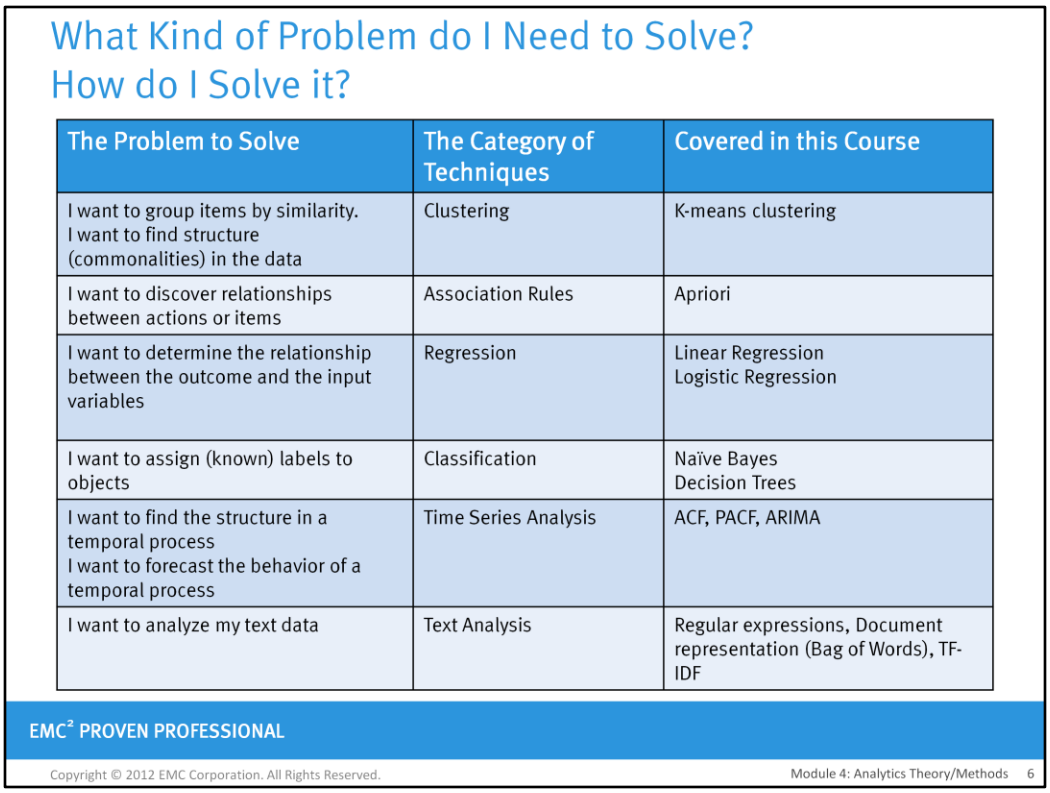
a) Prominent use cases for the method
b) Algorithms to implement the method
c) Diagnostics that are most commonly used to evaluate the effectiveness of the method
d) The Reasons to Choose (+) and Cautions (-) (where the method is most and least effective)



Here we recall phases of analytic life cycle we would have gone through before we plan for the analytic method we should be using with the data.



Model planning is the process of determining the appropriate analytic method based on the problem. It also depends on the type of data and the computational resources available.



This table lists the typical business questions (column 1) addressed by a category of techniques or analytical methods (column 2)

Some of the typical business questions for different category of techniques are listed below:

Clustering	How do I group these documents by topic? How do I group these images by similarity? (More businesslike questions)
Association Rules	What do other people like this person tend to like/buy/watch?
Regression	I want to predict the lifetime value of this customer. I want to predict the probability that this loan will default.
Classification	Where in the catalog should I place this product? Is this email spam?
Time Series Analysis	What is the likely future price of this stock? What will my sales volume be next month?
Text Analysis	Is this a positive product review or a negative one?

As it can be observed that these category of techniques overlap with each other with the type of problem they can be used to solve.

Questions such as "How do I group these documents?" and "Is this email spam?" , “Is this a positive product review" can all be answered with a “classification”. But these questions can also be considered as a Text analysis problem which we cover in this module. Text analysis is defined as term for the specific process of representing, manipulating, and predicting or learning over text. The tasks themselves can often be classified as clustering, or classification.

Similarly more than one method can be used to solve the same problem. For example Time Series Analysis can be used to predict prices over time. Time series is used in cases where the past is observable to the participants, which is often true of stock, and real estate. Sometimes we can use regression methods as well. However, regression is most effective when assigning effects to complicated patterns of treatment.

Column 3 in the table above lists the specific analytical methods that are detailed in the subsequent lessons in this module.

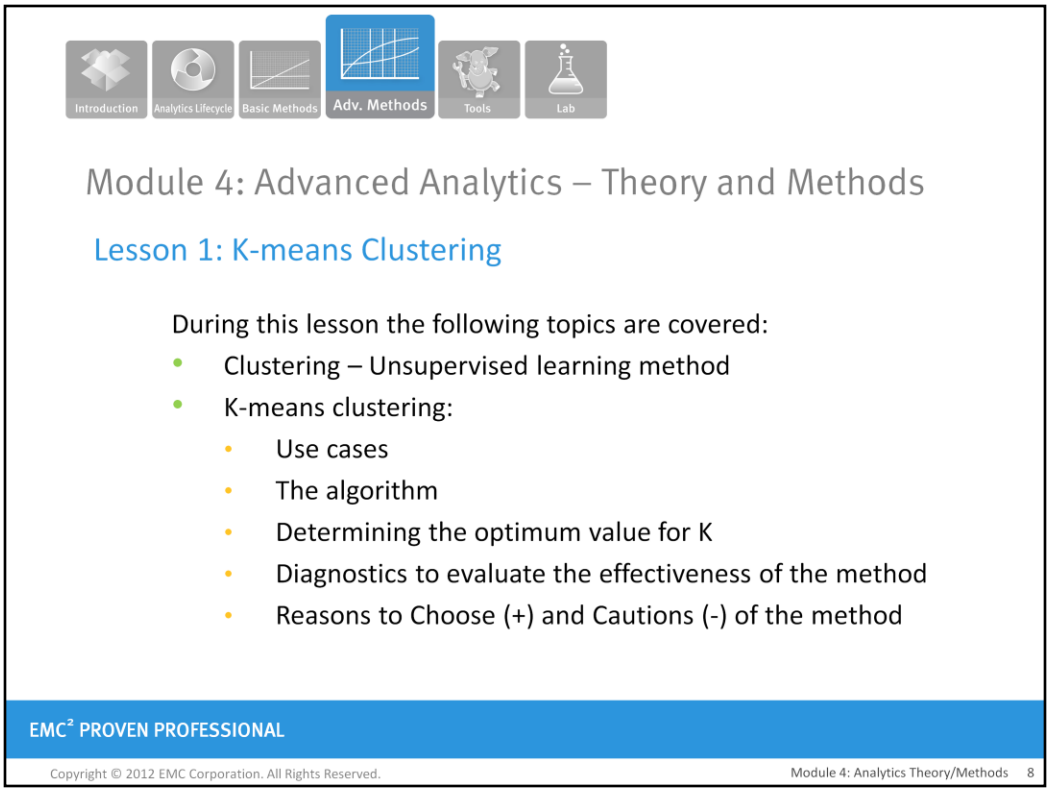
Why These Example Techniques?

- Most popular, frequently used:
  - Provide the foundation for Data Science skills on which to build
- Relatively easy for new Data Scientists to understand & comprehend
- Applicable to a broad range of problems in several verticals

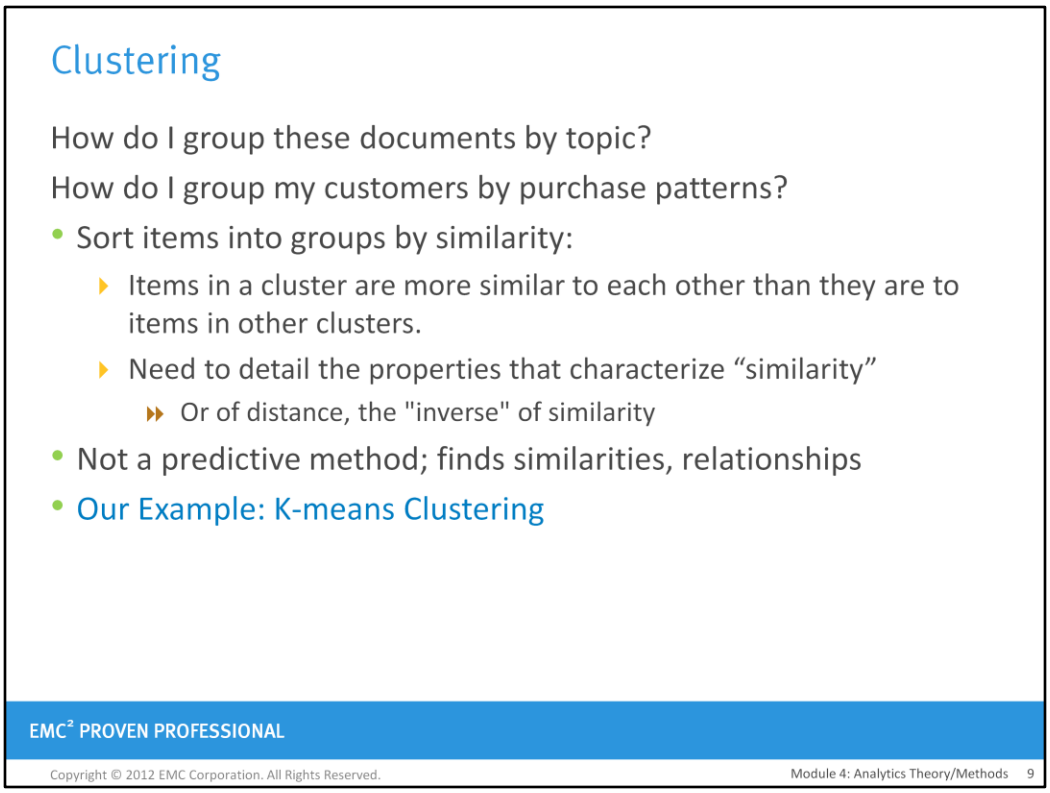
EMC² PROVEN PROFESSIONAL

We present in this module K-means clustering, Apriori algorithm for Association rules, Linear and logistic regression, Classification methods with Naïve Bayesian method and Decision Trees, Time Series Analysis with Box-Jenkins ARIMA modeling and key concepts such as TF-IDF.

Regular expressions and document representation methods with “bag of words” are chosen to be presented in this module among several techniques available for the Data Scientists to use to solve analytic problems. The reasons for which these techniques are chosen among all the available techniques are listed on this slide.



This lesson covers K-means clustering with these topics.



In machine learning, “unsupervised” refers to the problem of finding a hidden structure within unlabeled data. In this lesson and the following lesson we will be discussing two unsupervised learning methods clustering and Association Rules.

Clustering is a popular method used to form homogenous groups within a data set based on their internal structure. Clustering is a method often used for exploratory analysis of the data. There are no ”predictions” of any values done with clustering just finding the similarity between the data and grouping them into clusters

The notion of similarities can be explained with the following examples:

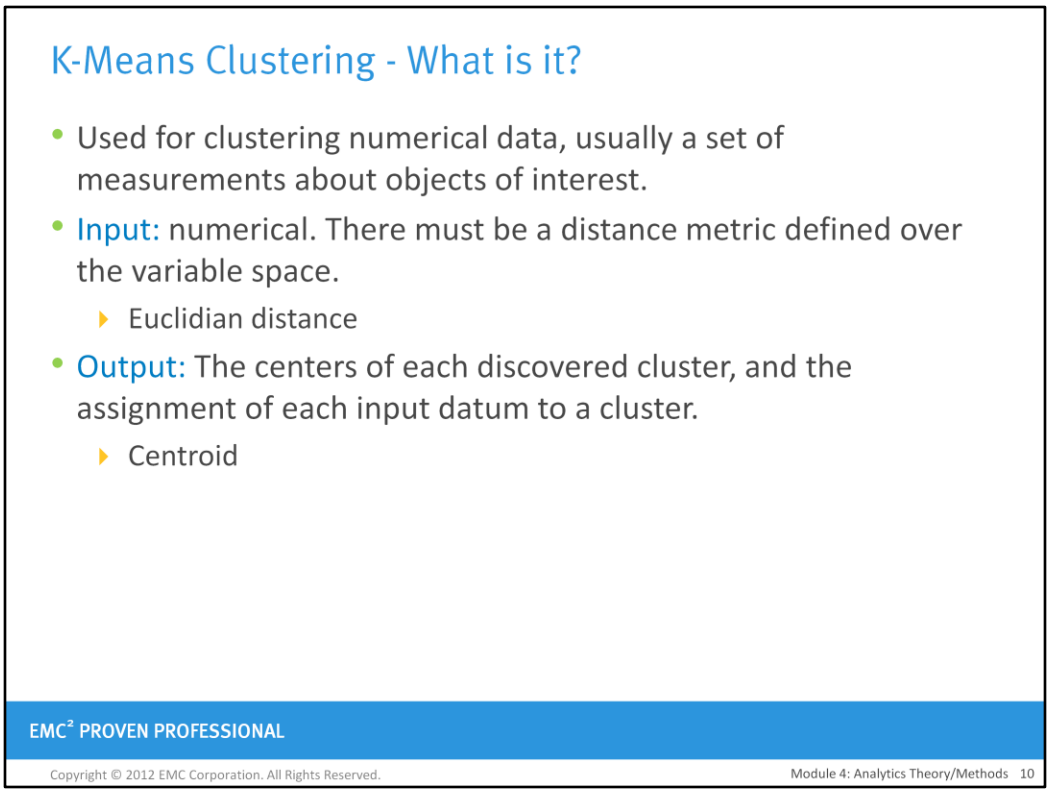
Consider questions such as

1. How do I group these documents by topic?
2. How do I perform customer segmentation to allow for targeted or special marketing programs.

The definition of “similarity” is specific to the problem domain. We are defining similarity as those data points with the same “topic” tag or customers who can be profiled in to a same “age group/income/gender” or a “purchase pattern”.

If we have a vector of measurements of an attribute of the data, the data points that are grouped into a cluster will have values for the measurement close to each other than to those data points grouped in a different cluster. In other words the distance, (an inverse of similarity) between the points within a cluster are always lower than the distance between points in a different cluster. **In a cluster we end up with a tight group (homogeneous) of data points that are far apart from those data points that end up in a different cluster.**

There are many clustering techniques and we are going to discuss one of the most popular clustering method known as “K-means clustering” in this lesson.



K-means clustering is used to cluster numerical data.

In K-means we define two measures of distances, between two data points(records) and the distance between two clusters. Distance can be measured (calculated) in a number of ways but four principles tend to hold true.

1. Distance is not negative (it is stated as an absolute value)
2. Distance from one record to itself is zero.
3. Distance from record I to record J is the same as the distance from record J to record I, again since the distance is stated as an absolute value, the starting and end points can be reversed.
4. Distance between two records can not be greater than the sum of the distance between each record and a third record.

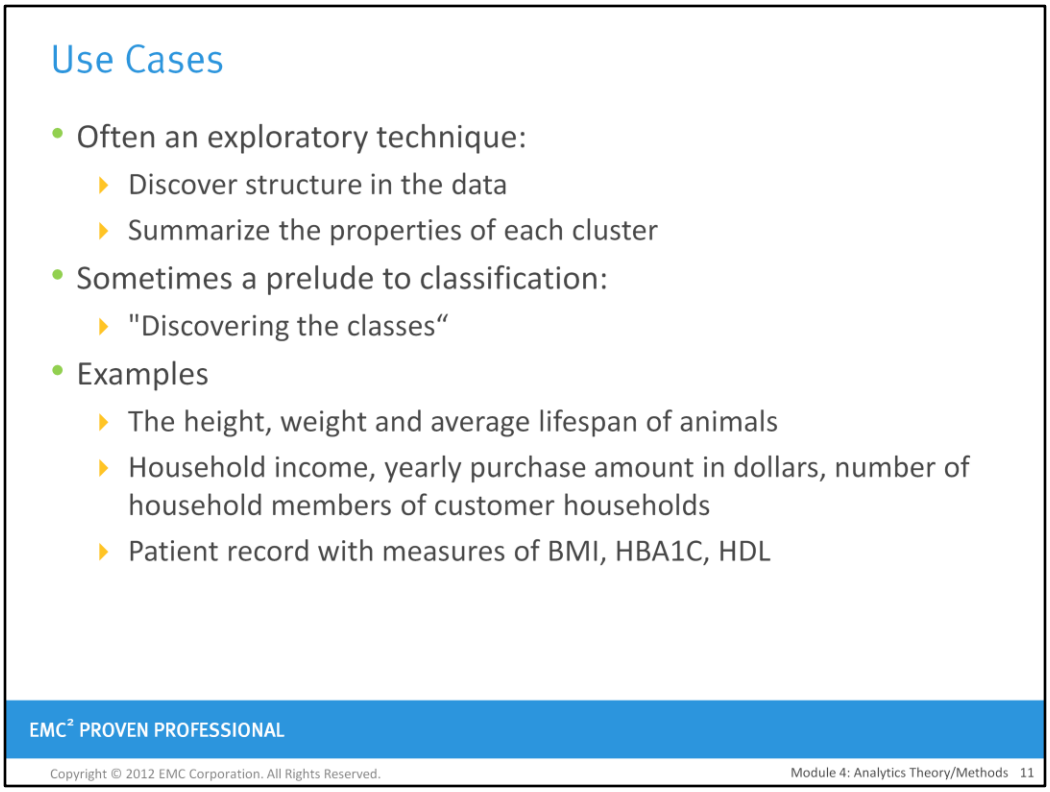
Euclidean distance is the most popular method for calculating distance. Euclidian distance is a "ordinary" distance that one could measure with a ruler. In a single dimension the Euclidian distance is the absolute value of the differences between two points. The straight line distance between two points. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)}$.

In N dimensions, the Euclidean distance between two points p and q is $\sqrt{(\sum_{i=1}^{N} (p_i - q_i)^2)}$ where p_i (or q_i) is the coordinate of p (or q) in dimension i.

Though there are many other distance measures, the Euclidian distance is the most commonly used distance measure and many packages use this measure.

The Euclidian distance is influenced by the scale of the variables. Changing the scale (for example from feet to inches) can significantly influence the results.Second, the equation ignores the relationship between variables. Lastly, the clustering algorithm is sensitive to outliers. If the data has outliers and removal of them is not possible, the results of the clustering can be substantially distorted.

The centroid is the center of the discovered cluster. K-means clustering provides this as an output. When the number of clusters is fixed to k, *K-means clustering* gives a formal definition as an optimization problem: find the *k* cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.



K-means clustering is often used as a lead-in to classification. It is primarily an exploratory technique to discover the structure of the data that you might not have notice before and as a prelude to more focused analysis or decision processes.

Some examples of the set of measurements based on which clustering can be performed are detailed in the slide.

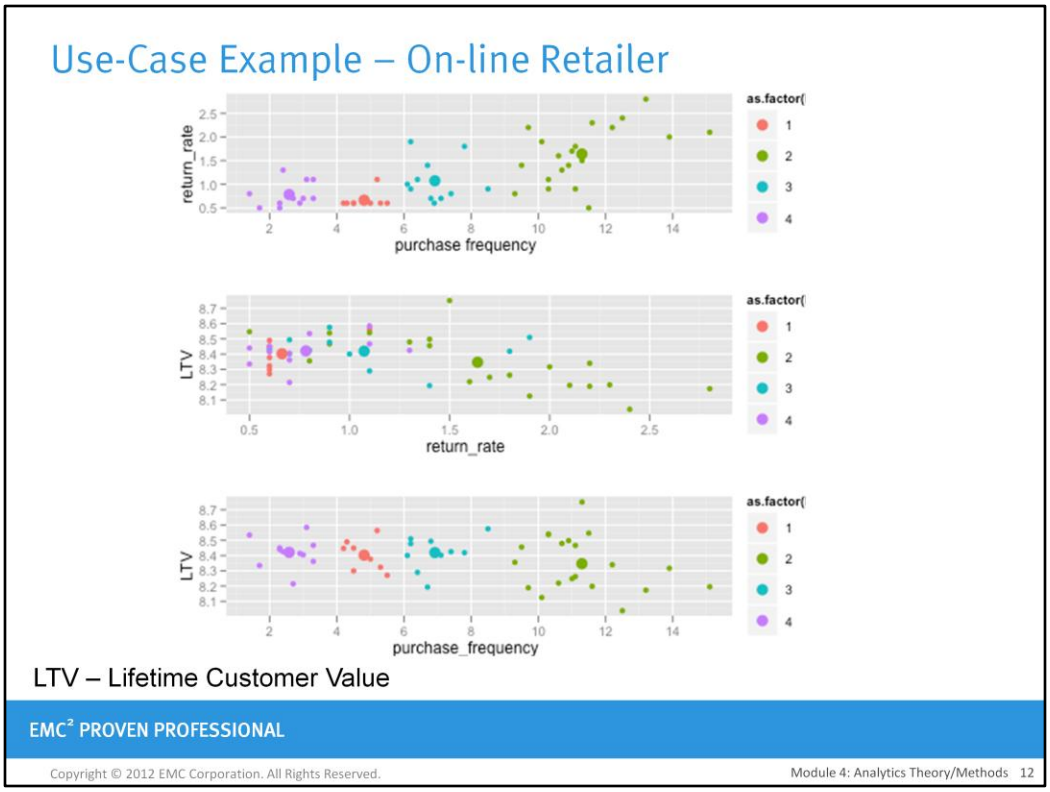
In the patient record where we have measures such as BMI, HBA1C, HDL with which we could cluster patients into groups that define varying degrees of risk of a heart disease.

In Classification the labels are known. Whereas in clustering the labels are not known. Hence clustering can be used to determine the structure in the data and summarize the properties of each cluster in terms of the measured centroids for the group. The clusters can define what the initial classes could be.

In low dimensions we can visualize the clusters. It gets very hard to visualize as the dimensions increase.

There are a lot of applications of the K-mean clustering, examples include pattern recognition, classification analysis, artificial intelligence, image processing, machine vision, etc.

In principle, you have several objects and each object has several attributes. You want to classify the objects based on the attributes, then you can apply this algorithm. For Data Scientists, K-means is an excellent tool to understand the structure of data and validate some of the assumptions that are provided by the domain experts pertaining to the data. We will look into a specific use-case in the following slide.



Here we present a fabricated example of an on-line retailer. The unique selling point of this retailer is that they make the "returns" simple with an assumption that this policy encourages use and "frequent customers are more valuable". So let us validate this assumption.

We took a sample set of customers clustered on purchase frequency, return rate, and lifetime customer value (LTV).

We define purchase frequency as the number of visits a customer made in a month on average that had a shopping cart transaction.

We can easily see that return rate has an important effect on customer value.

We clustered the customers into 4 groups, and the plotted 3 graphs taking two of the attributes in a graph. The data points are represented in the graphs by different colors for each cluster and larger "dot" represents the centroid for the group.

The groups can be defined broadly as follows:

GP1: Visit less frequently, low return rate, moderate LTV(ranked 3rd)

GP2: Visit often, return a lot of their purchases. Lowest avg LTV (counter intuitive)

GP3: Visit often, return things moderately, High LTV (ranked 2nd) (happy medium)

GP4: Visit rarely, don't return purchases. Highest avg LTV

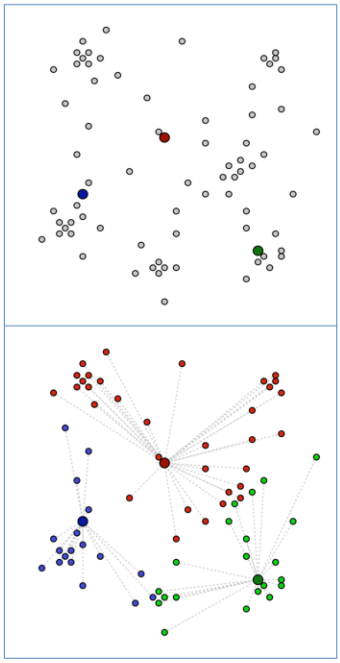
It appears that GP3 is the ideal group – they visit often, return things moderately, and are high value. The next questions are

- Why is it that GP3 is ideal?
- What are the people in these different groups buying?
- Is that affecting LTV?
- Can we raise the LTV of our frequent customers, perhaps by lowering the cost of returns, or by somehow discouraging customers who return goods too frequently?
- Can we encourage GP4 customers to visit more (without lowering their LTV?)
- Are more frequent customers more valuable?

You can see the range of questions that a Data Scientist can address with the initial analysis with k-means clustering.

The Algorithm

1. Choose *K*; then select K random "centroids"
   In our example, K=3
2. Assign records to the cluster with the closest centroid



EMC² PROVEN PROFESSIONAL

Step 1 - K-means clustering begins with the data set segmented into K clusters.

Step 2- Observations are moved from cluster to cluster to help reduce the distance from the observation to the cluster centroid.

The Algorithm (Continued)

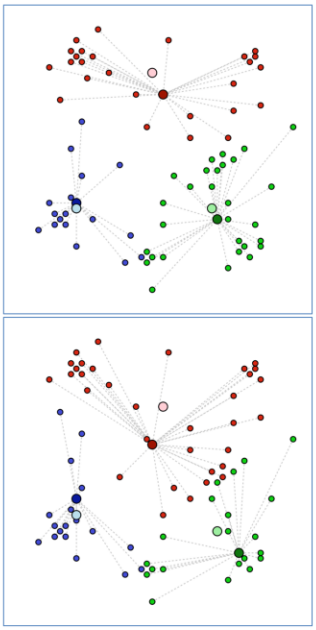
3. Recalculate the resulting centroids

   Centroid: the mean value of all the records in the cluster

4. Repeat steps 2 & 3 until record assignments no longer change

Model Output:

- The final cluster centers
- The final cluster assignments of the training data



EMC² PROVEN PROFESSIONAL

Step 3 - When observations are moved to a new cluster, the centroid for the affected clusters needs to be recalculated.

Step 4 - This movement and recalculation is repeated until movement no longer results in an improvement.

The model output is the final cluster centers and the final cluster assignments for the data.

Selecting the appropriate number of clusters, K, can be done upfront if you possess some knowledge on what the right number may be. Alternatively you can try the exercise with different values for K and decide which clusters best suit your needs. Since it is rare that the appropriate number of clusters in a dataset is known, it is good practice to select a few values for k and compare the results.

The first partitioning should be done with the same knowledge used to select the appropriate value of K, for example domain knowledge about the market or industries.

If K was selected without external knowledge, the partitioning can be done without any inputs.

Once all observations are assigned to their closest cluster, the clusters can be evaluated for their "in-cluster dispersion." Clusters with the smallest average distance are the most homogenous. We can also examine the distance between clusters and decide if it makes sense to combine clusters which may be located close together. We can also use the distance between clusters to assess how successful the clustering exercise has been. Ideally, the clusters should not be located close together as the clusters should be well separated.

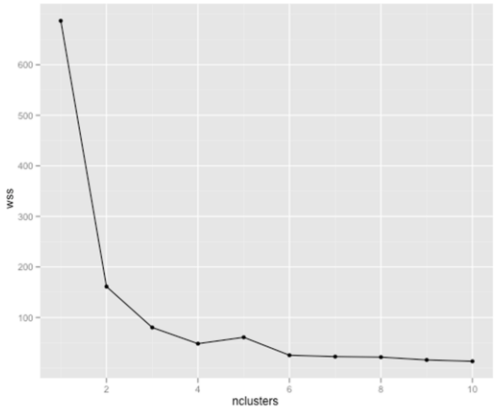
Picking K

Heuristic: find the "elbow" of the within-sum-of-squares (wss) plot as a function of K.

$$\text{WSS} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} |x_{ij} - c_i|^2$$

K: # of clusters
n_i: # points in i^{th} cluster
c_i: centroid of i^{th} cluster
x_{ij}: jth point of i^{th} cluster

"Elbows" at k=2,4,6



EMC² PROVEN PROFESSIONAL

Practically based on the domain knowledge, a value for K is picked and the centroids are computed. Then a different K is chosen and the model is repeated to observe if it enhanced the cohesiveness of the data points within the cluster group. However if there is no apparent structure in the data we may have to try multiple values for K. It is an exploratory process.

We present here one of the heuristic approaches used for picking the optimal “K” for the given dataset. “Within Sum of Squares” – WSS is a measure of how tight on average each cluster is. For k=1, WSS can be considered the overall dispersion of the data. WSS primarily is a measure of homogeneity. In general more clusters result in tighter clusters. But having too many clusters is over-fitting. The formula that defines WSS is shown. The graph depicts the value of WSS on the Y-axis and the number of clusters on the X-axis. The online retailer example data we reviewed earlier is the data with which the graph shown here is generated. We repeated the clustering for 12 different values .When we went from one cluster to two there is a significant drop in the value of WSS, since with two clusters you get more homogeneity. We look for the elbow of the curve which provides the optimal number of clusters for the given data.

Visualizing the data helps in confirming the optimal number of clusters. Reviewing the three pair-wise graphs we plotted for the online retailer example earlier you can see that having four groups sufficiently explained the data and from the graph above we can also see the elbow of the curve is at 4.

Diagnostics – Evaluating the Model

- Do the clusters look separated in at least some of the plots when you do pair-wise plots of the clusters?
  - Pair-wise plots can be used when there are not many variables
- Do you have any clusters with few data points?
  - Try decreasing the value of K
- Are there splits on variables that you would expect, but don't see?
  - Try increasing the value K
- Do any of the centroids seem too close to each other?
  - Try decreasing the value of K

EMC² PROVEN PROFESSIONAL

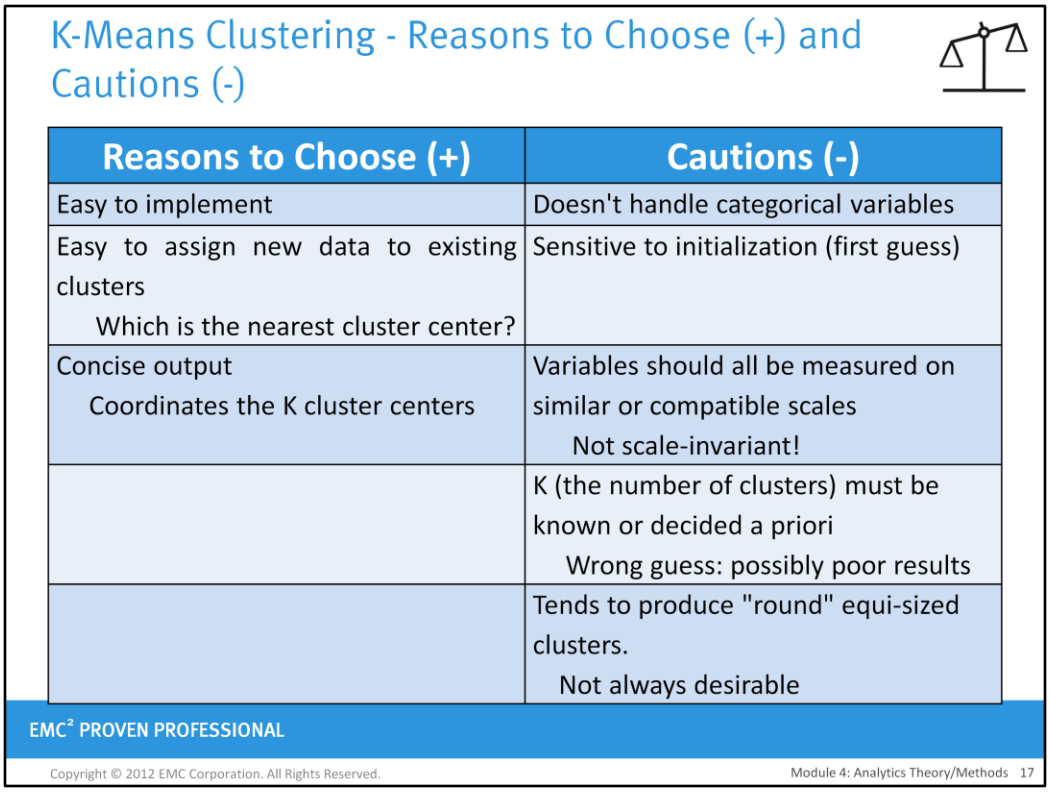
How do we know that we have good clusters?

Pair-wise plots of the clusters provide a good visual confirmation that the clusters are homogeneous. When the dimensions of the data are not significantly large this method helps in determining the optimal number of clusters. With these plots you should be able to determine if the clusters look separated in at least some of the plots. They won't be very separated in all of the plots. This can be seen even with the on-line retailer example we saw earlier. Some of the clusters get mixed in together in some dimensions.

If you feel that your clusters are too small it indicates that you have a large value for K and K needs to be reduced (try a smaller K). It may be the outliers in the data that tend to cluster into clusters with less data points.

Alternatively if you see there are splits that you expected but are not seen in the clusters, for example you expect two different income groups and you don't see them, you should try a bigger value for K.

If the centroids seem too close to each other then you should try decreasing the value of K.



K-means clustering is easy to implement and it produces concise output. It is easy to assign new data to the existing clusters by determining which centroid the new data point is closest to it.

However K-means works only on the numerical data and does not handle categorical variables. It is sensitive to the initial guess on the centroids. It is important that the variables must be all measured on similar or compatible scales. If you measure the living space of a house in square feet, the cost of the house in thousands of dollars (that is, 1 unit is $1000), and then you change the cost of the house to dollars (so one unit is $1), then the clusters may change**. K should be decided ahead of the modeling process.** Wrong guesses for K may lead to improper clustering.

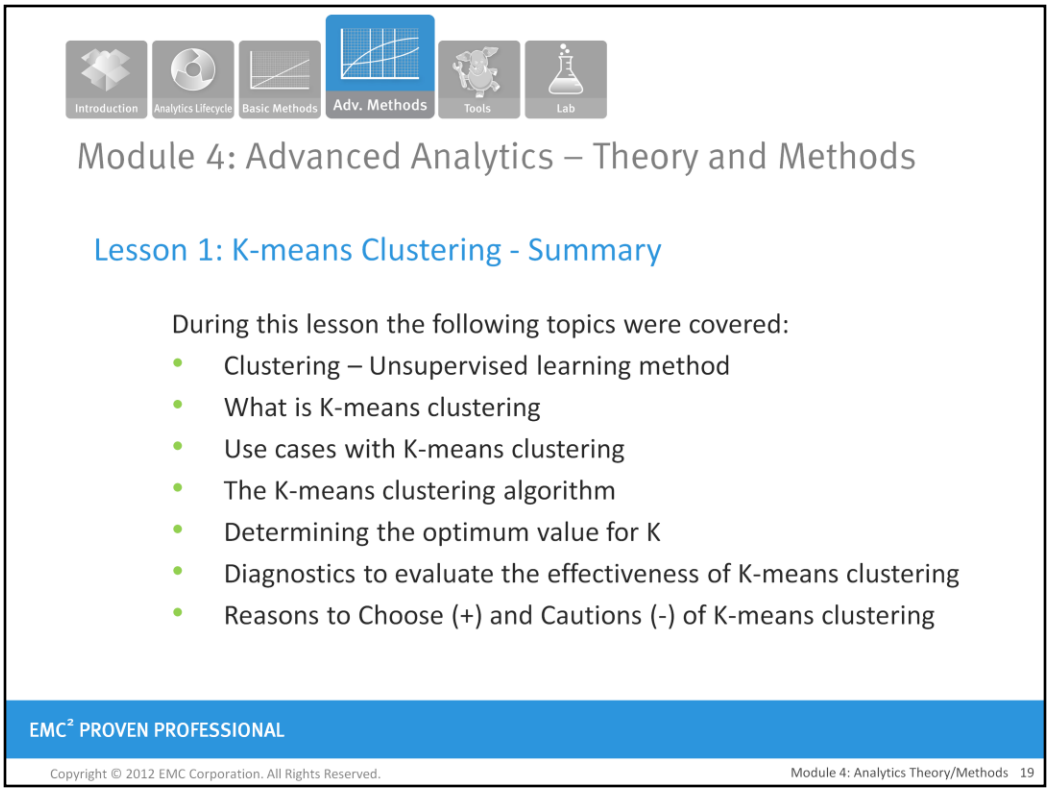
K-means tends to produce rounded and equal sized clusters. If you have clusters which are elongated or crescent shaped, K-means may not be able to find these clusters appropriately. The data in this case may have to be transformed before modeling.

Check Your Knowledge

1. Why do we consider K-means clustering as a unsupervised machine learning algorithm?
2. How do you use "pair-wise" plots to evaluate the effectiveness of the clustering?
3. Detail the four steps in the K-means clustering algorithm.
4. How do we use WSS to pick the value of K?
5. What is the most common measure of distance used with K-means clustering algorithms?
6. The attributes of a data set are "purchase decision (Yes/No), Gender (M/F), income group (<10K, 10-50K, >50K). Can you use K-means to cluster this data set?

EMC² PROVEN PROFESSIONAL

Record your answers here.



Summary of key-topics presented in this lesson are listed. Take a moment to review them.

Lab Exercise 4: K-means Clustering

This Lab is designed to investigate and practice K-means Clustering.

After completing the tasks in this lab you should be able to:

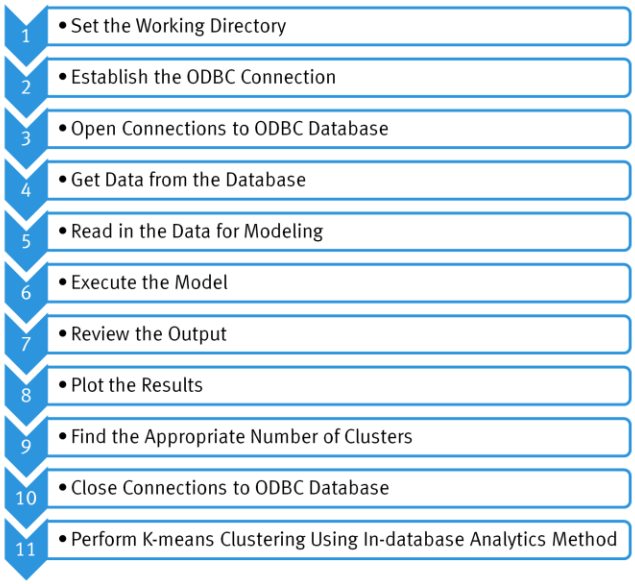
- Use R functions to create K-means Clustering models
- Use ODBC connection to the database and execute SQL statements and read database tables in an R environment
- Visualize the effectiveness of the K-means Clustering algorithm using graphic capabilities in R
- Use MADlib function for K-means Clustering

EMC² PROVEN PROFESSIONAL

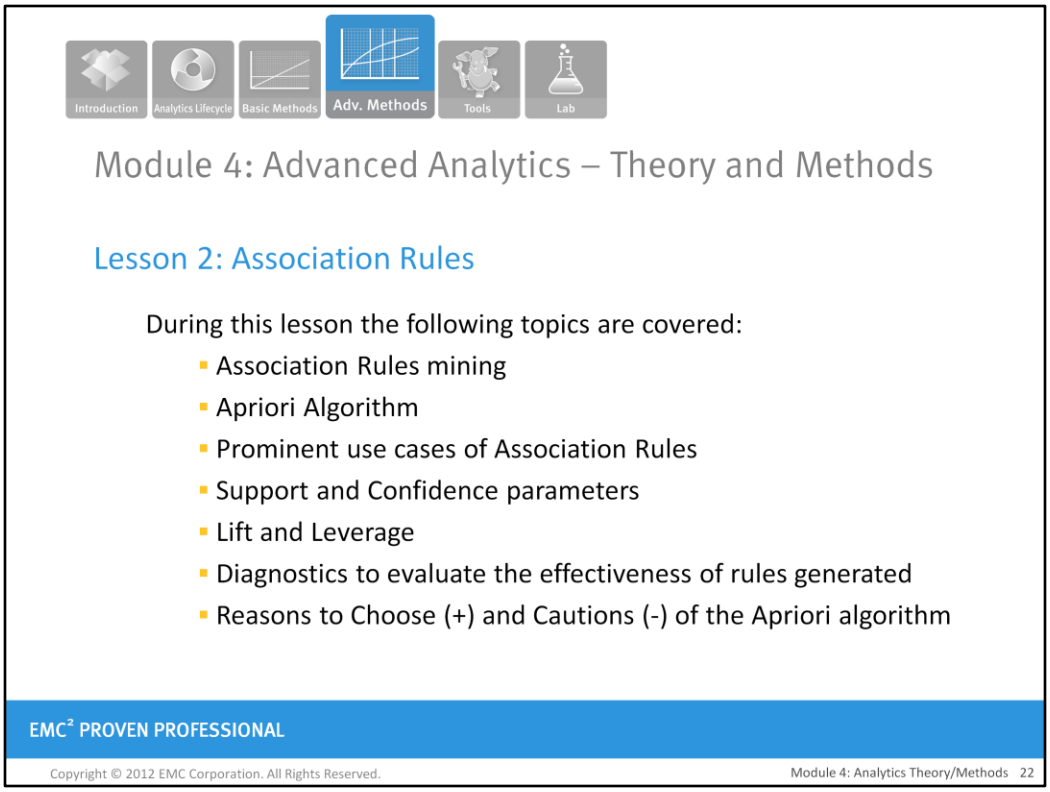
Tasks you will be completing in this lab include:

- Use RStudio environment to code K-means Clustering models
- Use the ODBC connection in the R environment to create the average household income from the census database as a test data for K-means Clustering
- Use R graphics functions to visualize the effectiveness of the K-means Clustering algorithm
- Use MADlib functions for K-means clustering

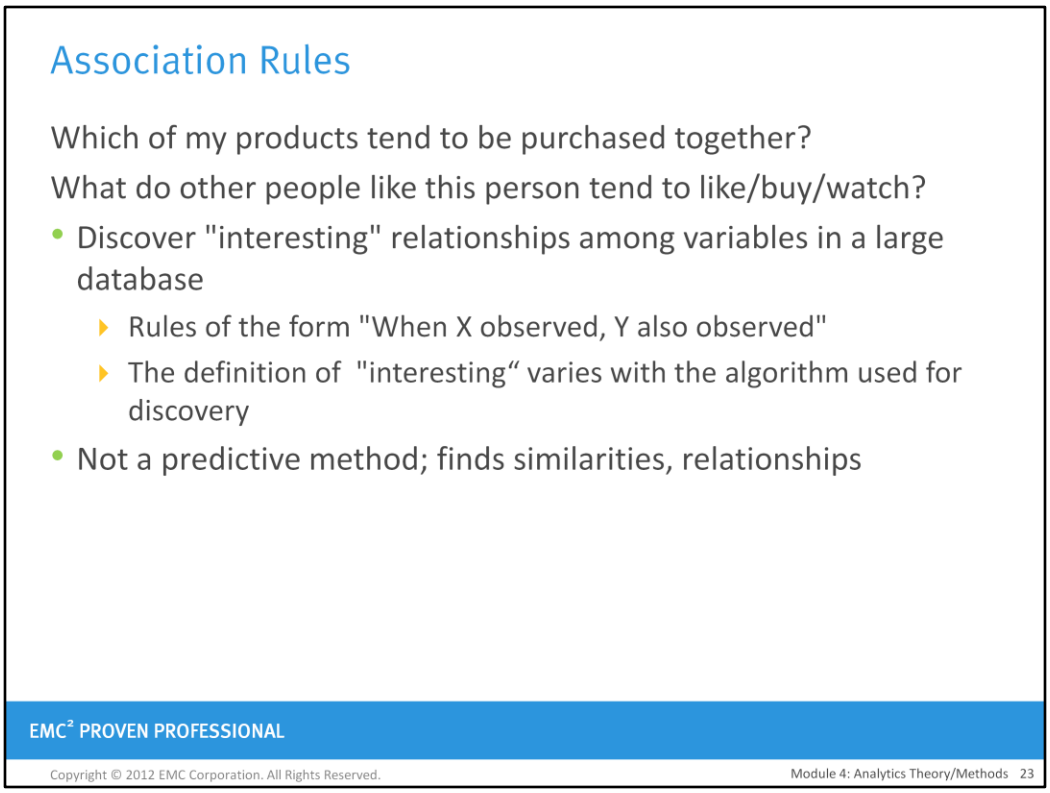
Lab Exercise 4: K-means Clustering - Workflow



EMC² PROVEN PROFESSIONAL



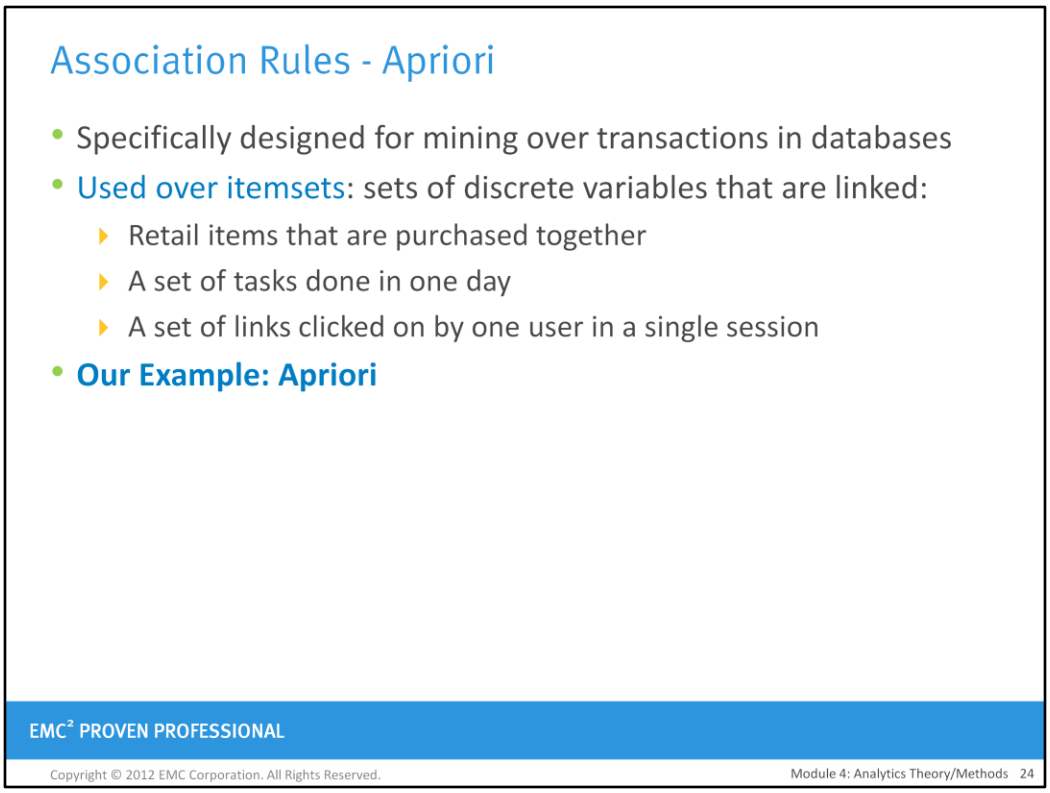
The topics covered in this lesson are listed.



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.



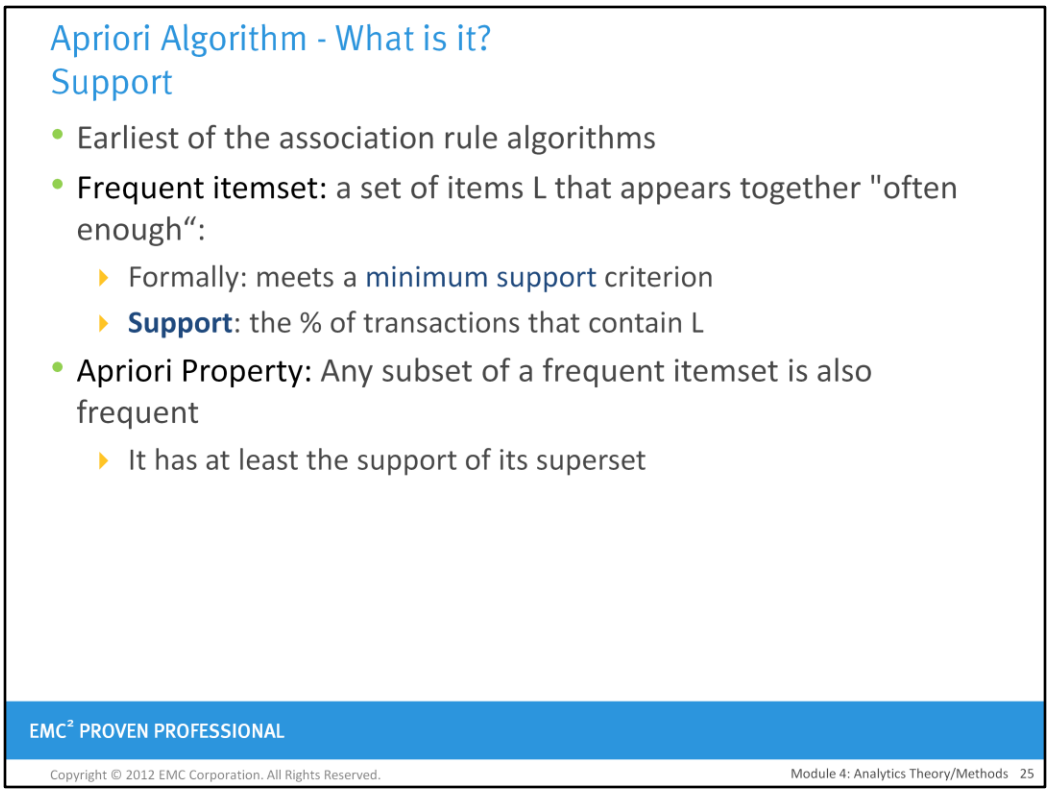
Association Rules are specifically designed for in-database mining over transactions in databases.

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.

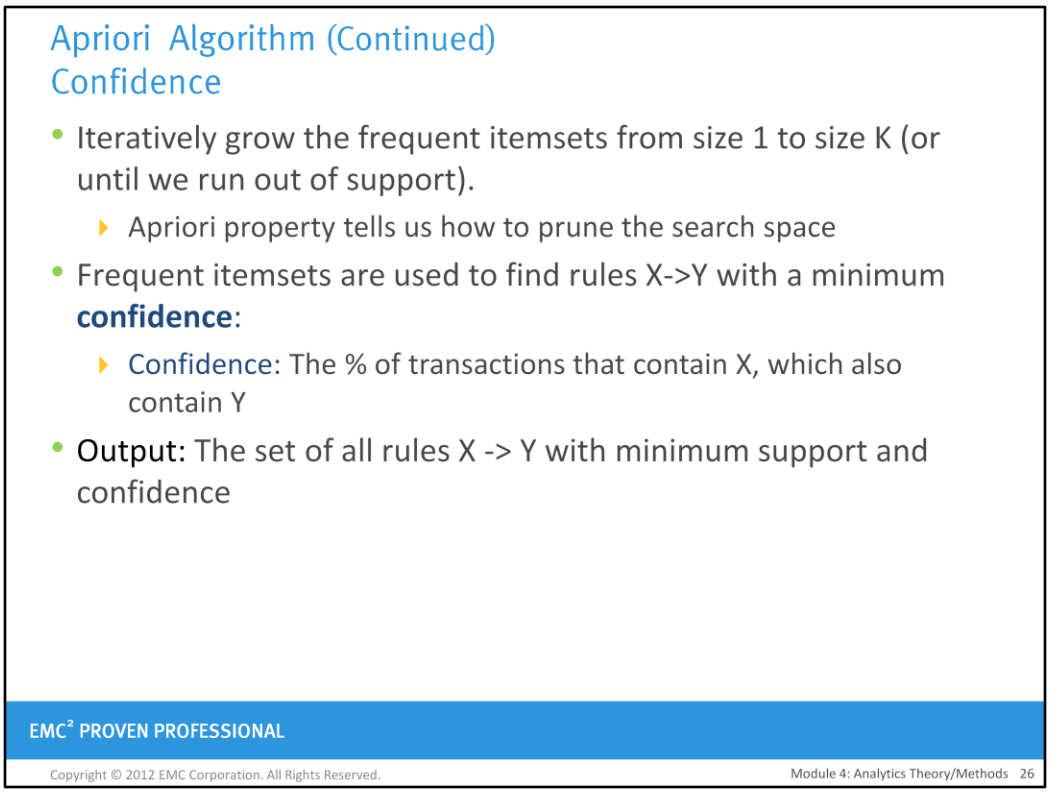


We will now detail the Apriori algorithm.

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 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

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

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

EMC² PROVEN PROFESSIONAL

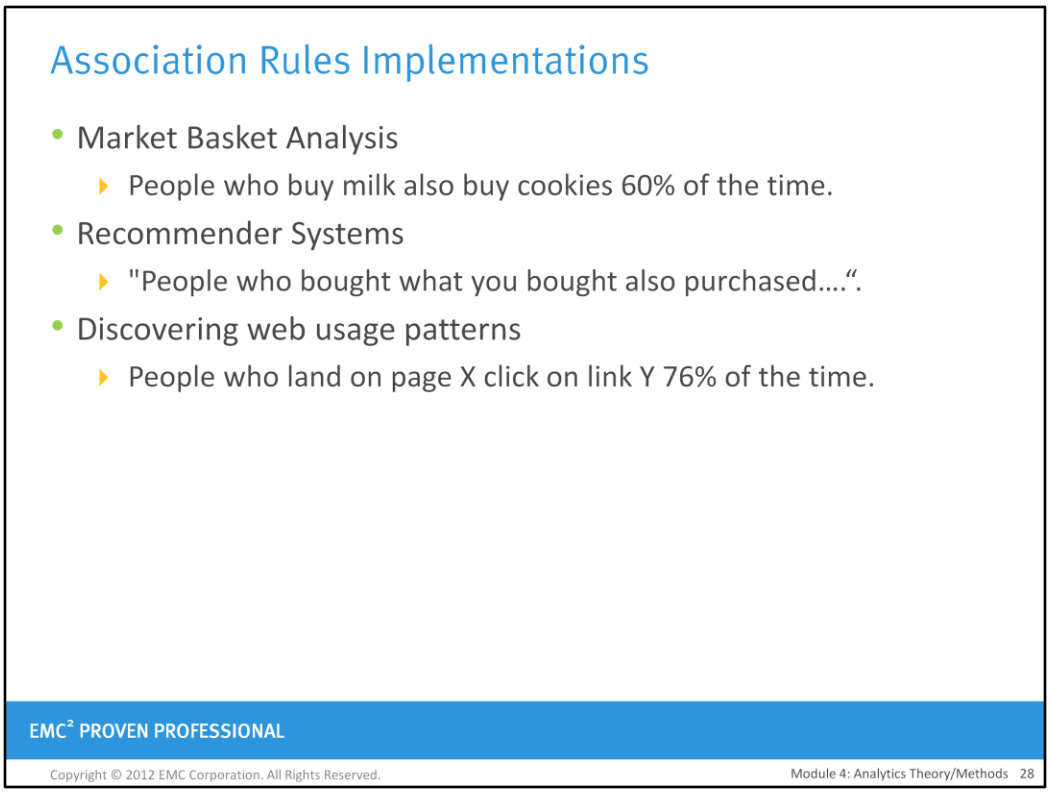
The common measures used by Apriori algorithm are **Support** and **Confidence** . We rank all the rules based on the support and confidence and filter out the most "interesting" rules. There are other measures to evaluate candidate rules and we will define two such measures **Lift** and **Leverage**.

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



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

EMC² PROVEN PROFESSIONAL

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

EMC² PROVEN PROFESSIONAL

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%

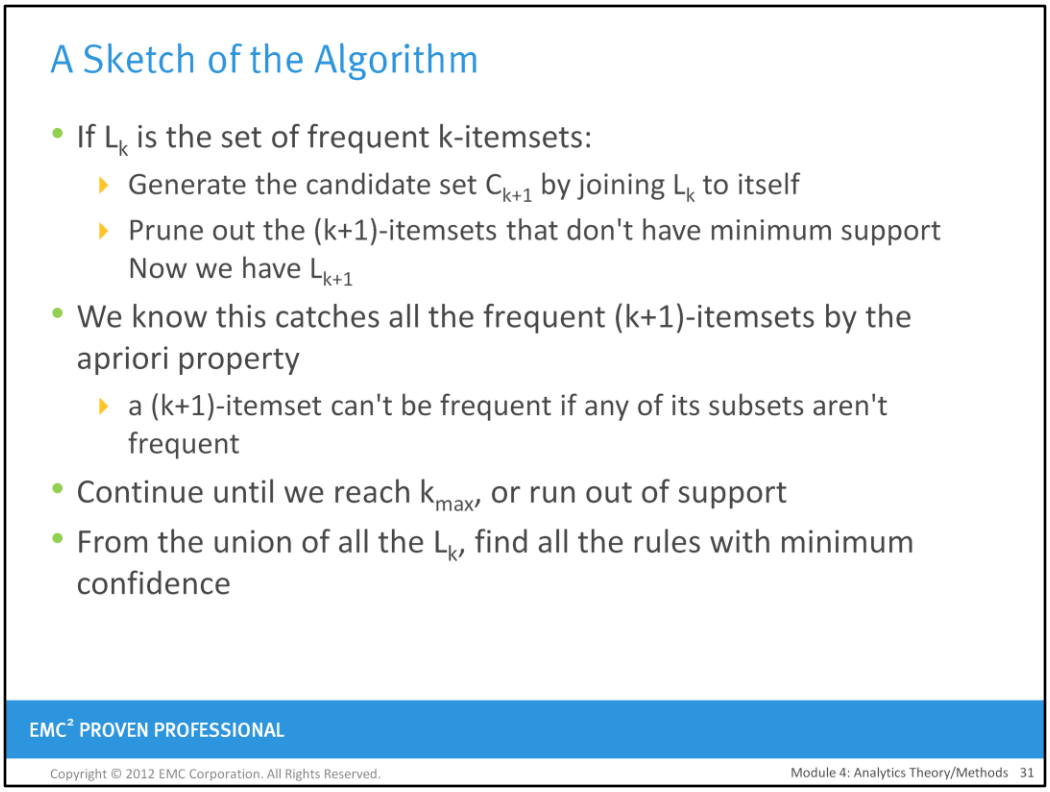
The confidence for the rule

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.



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~~

EMC² PROVEN PROFESSIONAL

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~~

EMC² PROVEN PROFESSIONAL

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

EMC² PROVEN PROFESSIONAL

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

EMC² PROVEN PROFESSIONAL

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 predict 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).

EMC² PROVEN PROFESSIONAL

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

EMC² PROVEN PROFESSIONAL

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.

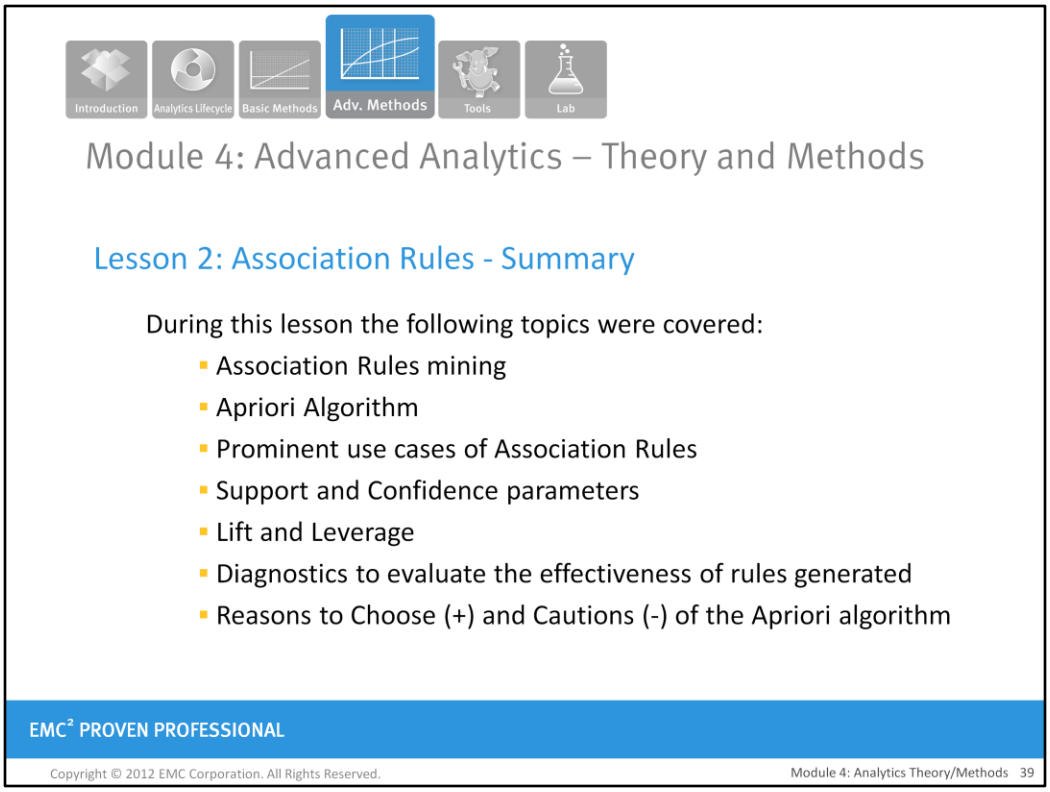
Lift and Leverage measures must be used to prune out these rules.

Check Your Knowledge

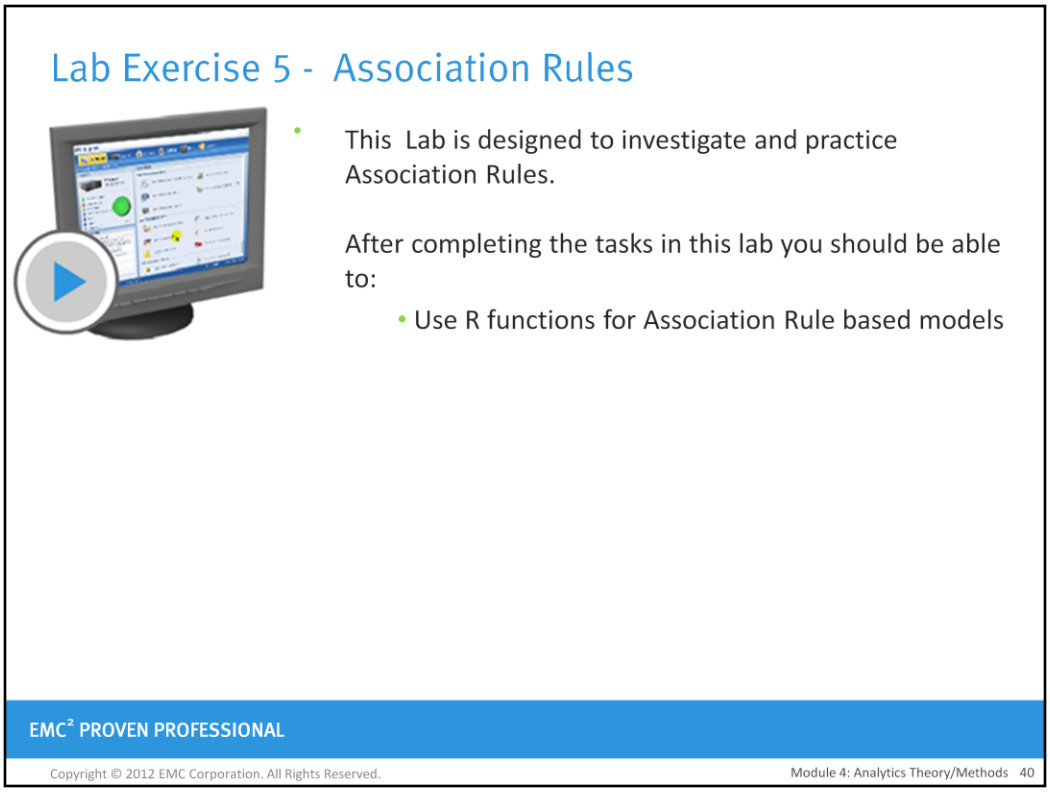
1. What is the Apriori property and how is it used in the Apriori algorithm?
2. List three popular use cases of the Association Rules mining algorithms.
3. What is the difference between Lift and Leverage. How is Lift used in evaluating the quality of rules discovered?
4. Define Support and Confidence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

EMC² PROVEN PROFESSIONAL

Record your answers here.



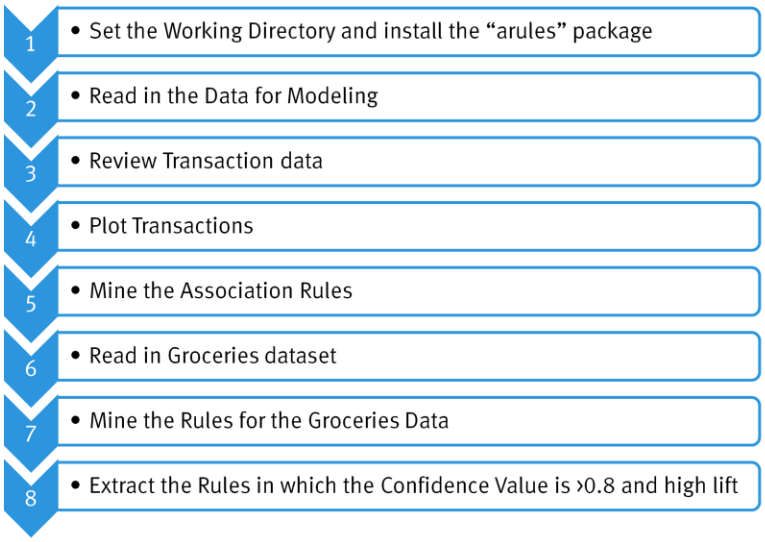
This lesson covered these topics. Please take a moment to review them.



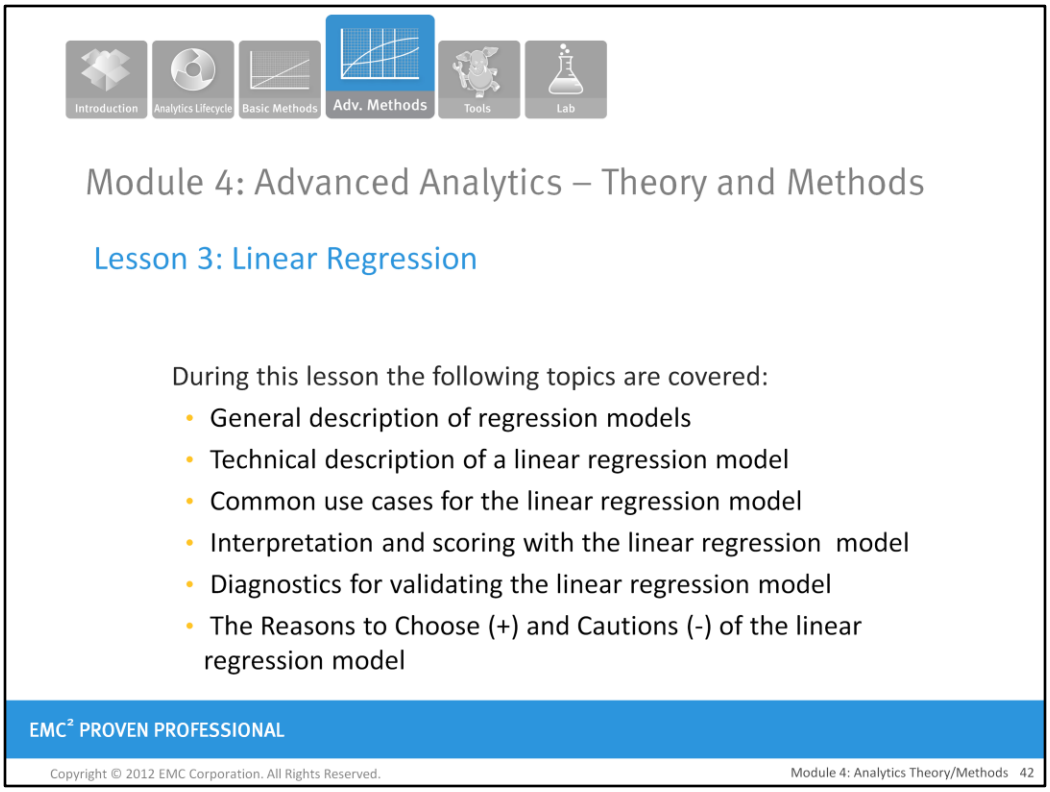
Tasks you will be completing in this lab include:

- Use RStudio environment to code Association Rule models.
- Apply constraints in the Market Basket Analysis methods such as minimum thresholds on support and confidence measures that can be used to select interesting rules from the set of all possible rules
- Use R graphics "arules" to execute and inspect the models and the effect of the various thresholds

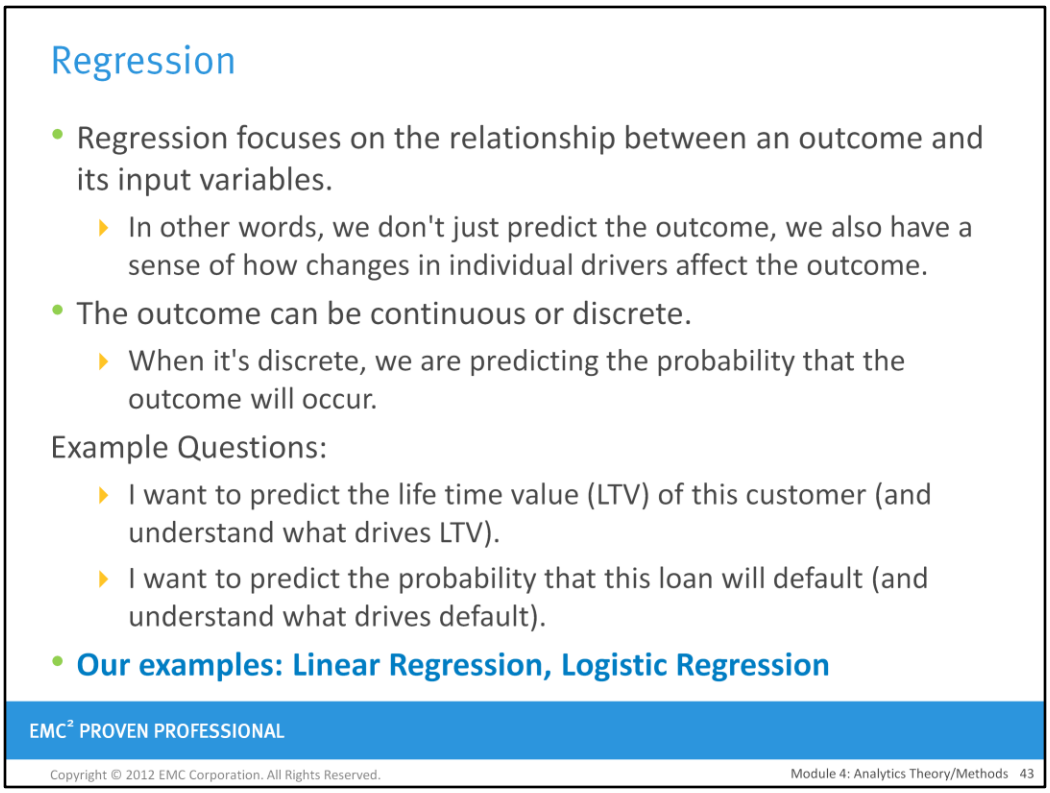
Lab Exercise 5 - Association Rules - Workflow



EMC² PROVEN PROFESSIONAL



The topics covered in this lesson are listed.



The term "regression" was coined by Francis Galton in the nineteenth century to describe a biological phenomenon. The phenomenon was that the heights of descendants of tall ancestors tend to regress down towards a normal average (a phenomenon also known as regression toward the mean).

Specifically, regression analysis helps one understand how the value of the dependent variable (also referred to as outcome) changes when any one of the independent variables changes (also referred as drivers), while the other independent variables are held fixed. Regression analysis estimates the conditional expectation of the dependent variable given the independent variables — that is, the mean value of the dependent variable when the independent variables are held fixed.

Some example questions are :

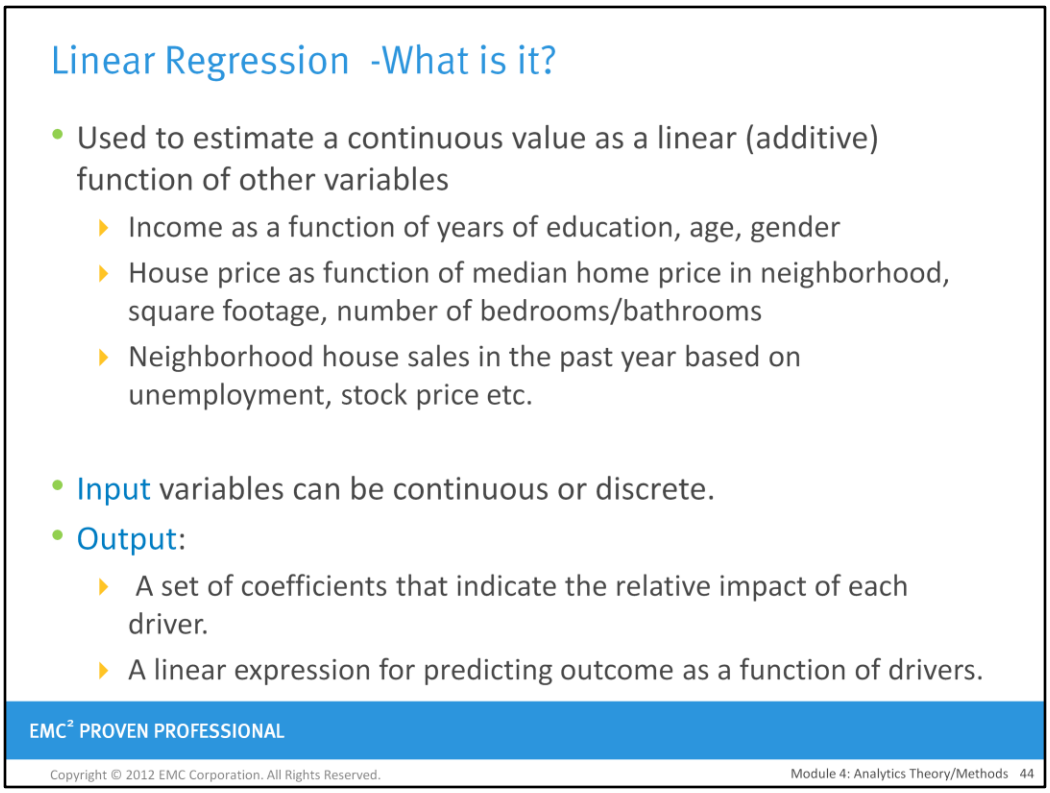
I want to predict the lifetime value of this customer and understand what drives LTV. What drives the LTV higher or lower?

I want to predict the probability that this loan will default and understand what drives default

Regression focuses on the relationship between the outputs and the inputs. It also provides a model that has some **explanatory value**, in addition to predicting outcomes. Social scientists used regression mainly for its explanatory value and it can be a fairly good predictor for which method is popular among Data Scientists.

The outcome can be continuous or discrete and when it is discrete we are predicting the probability that the outcome will occur.

Two types of regression methods will be discussed in this module. This lesson will focus on the Linear regression and the next lesson will detail Logistic regression.

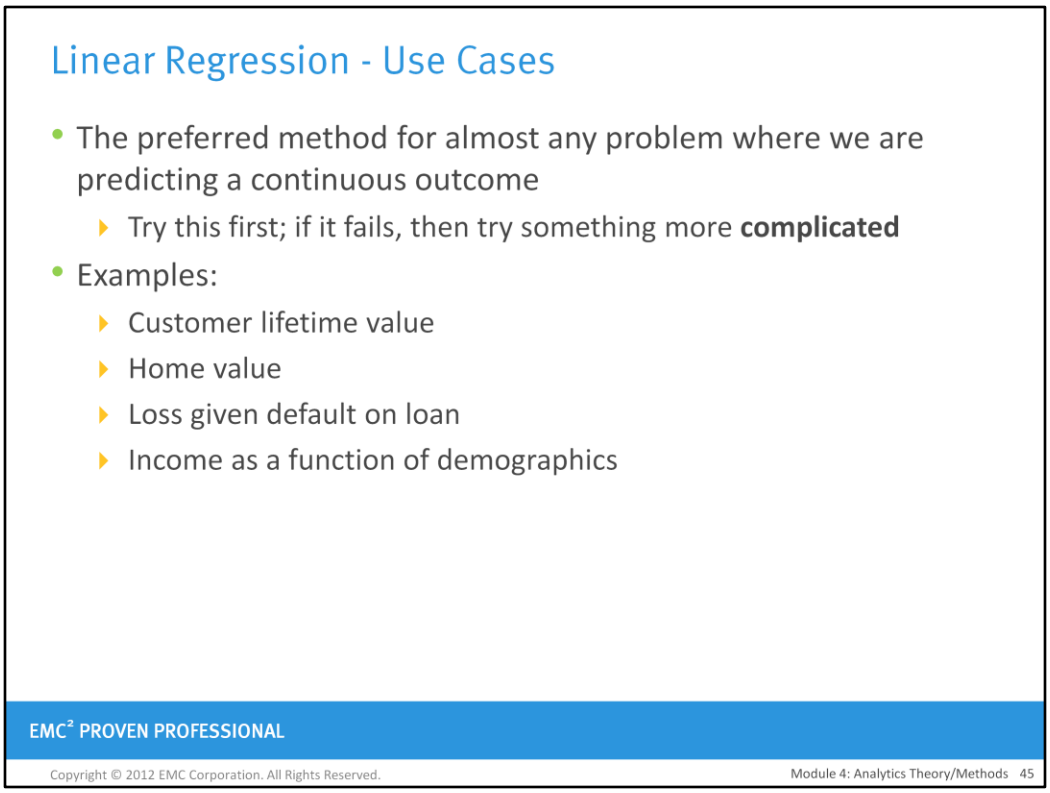


We use Linear regression to predict a continuous value as a linear or additive function of other variables. Some examples are

- Predicting income as a function of number of years of education, age and gender (drivers).
- House price (outcome) as a function of median home price in the neighborhood, square footage, number of rooms.
- Neighborhood house sales in the past year based on economic indicators.

The input variables can be continuous or discrete and the outputs are:

1) A set of coefficients that indicate the relative impact of each driver (possibly and how strongly the variables are correlated)
2) A linear expression predicting the outcome as a function of drivers.



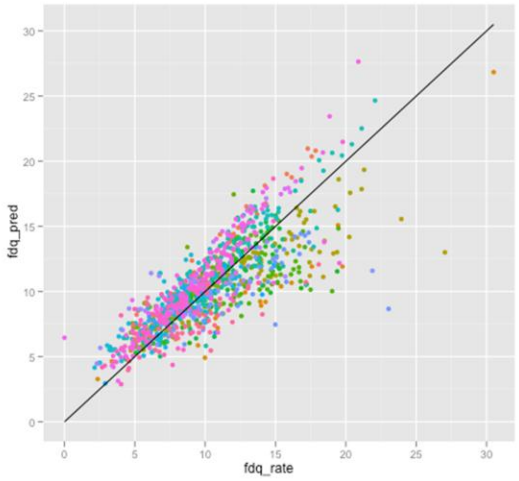
Linear Regression is the most frequently used technique for predicting a continuous outcome. It is simple and works well in most instances. It is recommended that Linear regression should be tried and if it is determined that the results are not reliable other complicated models should be used. Models such as kernelized ridge regression, local linear regression, regression trees, neural nets can be attempted. (all these models are out of scope for this course).

Some of the use cases are listed on the slide, others examples also are:

- Look at past years' sales orders and advertising campaigns to decide where and how you will spend this year's advertising budget
- Identify the relationship between important variables that affect your business or organization

Example: Predict Mortgage Foreclosure/Delinquency Rates

fdq_rate = -0.9 + 0.66 CurrentUnemp + 1.06 ChgInUnem1yr + 0.22 hicost_mort_rate



EMC² PROVEN PROFESSIONAL

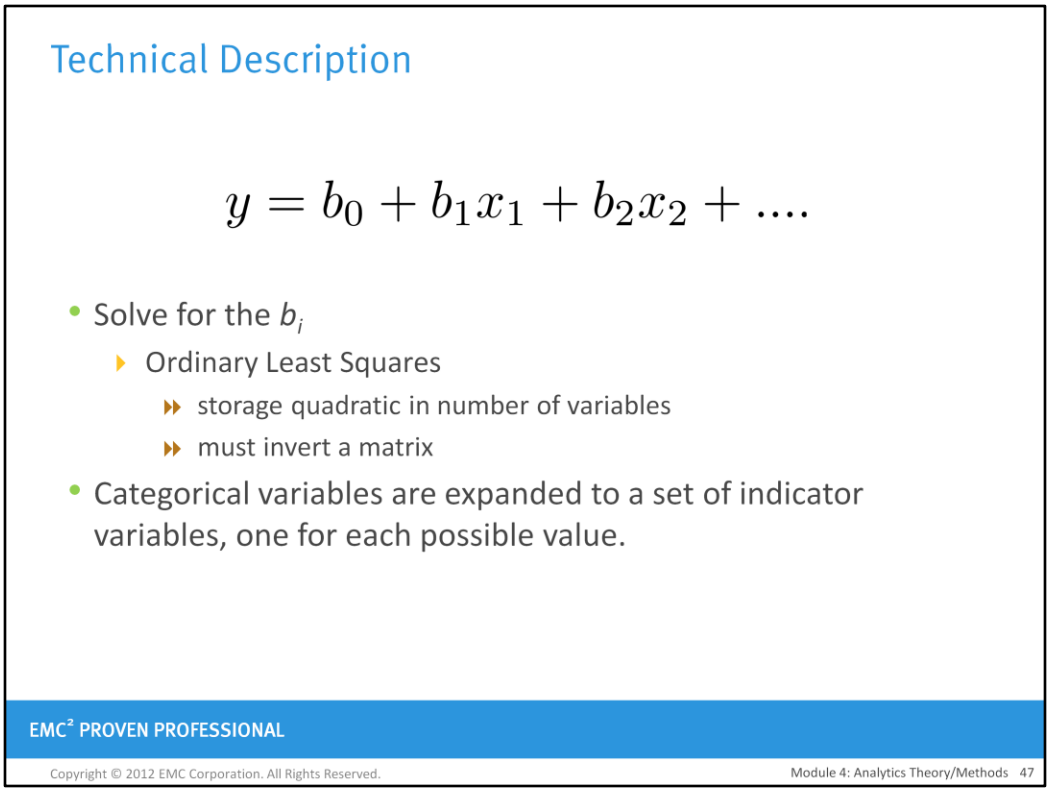
Here is an example of predicting mortgage foreclosure given delinquency rates:

A large bank is evaluating the performance of its mortgage portfolios across the nation. They are interested in predicting the rate of foreclosure or serious delinquency within the next year (going forward) in various regions that they serve.

This model shows that fdq is positively correlated with the unemployment rate in the region, and even more strongly correlated with the change in unemployment over the past year. The rate of "high cost" mortgages (mortgages that charge more interest than an equivalent treasury security would have earned) also affects the fdq rate.

The graph shows the true foreclosure rate on the x-axis and the y axis shows what we predicted from the model. Color coding is for the state, which isn't a part of this model.

The points that are bunched closer to the line (x=y , perfect prediction) are indications of good prediction. We see the spread out from the line for some states and it shows that the model does predict fdq better in some states than in others.



Linear regressions are of the form y is equal to a constant term + a linear combination of all the variables. The linear combinations are made up of a coefficient term "b_i" multiplied by the value of the corresponding variable.

The problem itself is solving for the b_i

It is a matrix inversion problem and the method is referred to as Ordinary Least Squares (OLS). The solution requires storage as the square of the number of variables and we need to invert a matrix. The complexity of the solution (both in storage and computation) increases as the number of variables increase.

When you have categorical variables, they are expanded as a set of indicator variables one for each possible value. We will explain this in the next slide in more detail with an example.

What we highlight here is that if we expand on categorical variables (ZIP codes as a categorical value) we will end up with lot of variables and the complexity of the solution becomes extremely high.

Representing Categorical Variable

$$\text{income} = b_0 + b_1\text{age} + b_2\text{yearsOfEducation} + b_3\text{gender} + b_4\text{state}$$

- *State* is a categorical variable: 50 possible values.
- Expand it to 49 indicator (0/1) variables:
  - The remaining level is the "default level"
  - This is done automatically by standard packages
- *Gender* is categorical, too, but binary
  - so one variable: *genderMale*, which is 0 for females

EMC² PROVEN PROFESSIONAL

Here we present another example:

We are predicting income given age, years of education, gender and state. There are 50 possible values for state and we will have to expand it to 49 indicator variables with 0 or 1 and the remaining level is the default level.

As Gender is a binary variable it is denoted as a single variable genderMale, which is 0 for females.

Usually, you want to solve for log income, but just discussing income makes the explanations easier. The implicit assumption with linear regression is that the variables are normally distributed. In reality "income" is not normally distributed. So it is a good practice to model for the log of the income.

What do the Coefficients b_i Mean?

- Change in y as a function of unit change in x_i
  - all other things being equal
- Example: income in units of \$10K, years in age, $b_{age} = 2$
  - For the same gender, years of education, and state of residence, a person's income increases by 2 units (20K)for every year older
- Standard packages also report the significance of the b_i: probability that, in reality, $b_i = 0$
  - b_i "significant" if P($b_i = 0$) is small

EMC² PROVEN PROFESSIONAL

The first coefficient, b_0, represents the value of the outcome in the "reference situation" – the situation that is represented by all the continuous variables set to zero, and the categorical variables at their reference.

For the example on the previous slide, if the reference value of state is Alabama, and the reference value of gender is Female, then b_0 would represent the income of a hypothetical female from Alabama, who is zero years old, with zero years of education. Obviously, this situation doesn't always make sense, but it does give us a reference point from which the income varies, as we change the values of the different drivers.

The coefficients b_i measure the change in outcome variable as a function of unit change in the input variable considering all other things being equal.

Consider our example in the previous slide. Let us say income in units of 10K and the coefficient b_{age}= 2 for years in age. If the other variables such as gender, years of education, and state of residence remain the same, a person's income increases 20K for every year older.

If the coefficient for $b_{stateNebraska}$ is 2.3, then a 30 year old male with 16 years of education tends to make 2.3 times higher income in Nebraska, than in Alabama.

In R (and in many other packages), the reference level of categorical variables is the level that comes first alphabetically. You can set the reference level explicitly, by using the relevel() command.

In Linear regression we said earlier that the coefficients are explanatory and not just predictive. So we want to know if the variable really makes a difference. For example if we ask the question "does age impact income?" and we conclude the answer is no, then the coefficient for age must be zero.

We use the measure of significance for this purpose. Standard packages report the significance of b_i.

The significance is the probability that b_i is zero. Or b_i " is significant" if P($b_i = 0$) is small.

Diagnostics

- Hold-out data
  - Does the model predict well on data it hasn't seen?
- N-fold cross-validation
  - Partition the data into N groups.
  - Fit N models, holding out each group, and calculate the residuals on the group.
  - Estimated prediction error is the average over all the residuals.
- R^2 : The fraction of the variance in the output variable that the model can explain.
  - It is also the square of the correlation between the true output and the predicted output. You want it close to 1.

EMC² PROVEN PROFESSIONAL

Creating a hold-out data set (we discussed this in Apriori diagnostics earlier in lesson 2 of this module) before you fit the model, and using that to estimate prediction error is by far the easiest thing to do.

There are two kinds of errors in predictive models One is the training error and the other is the prediction error. sets is the easiest way to estimate the prediction errors.

N-fold cross validation – it tells you if your set of variables is reasonable. This method is used when you don't have enough data to create a (test set) data.

Cross Validation is done by Splitting the dataset into, say, N non-overlapping subsets (fold) , Fit a model using N-1 folds and predict its performance using the fold that was left out. This can be done for all possible combination of folds (first leave 1st fold out, then 2nd, .. , then Nth and train with the remaining folds). After completing the fit on all possible folds you estimate the mean performance of all folds (maybe also the variance/standard deviation of the performance).

R^2 (goodness of fit metric) is reported by all standard packages. It is the fraction of the variance in the output variable that the model can explain.

The definition of R^2 is $1 - SSerr/SStot$ where

$SSerr = Sum[(y-y_{pred})^2]$ and

$SStot = Sum[(y-y_{mean})^2]$.

For the output of a correlated model, like regression, this definition will be the square of the correlation. For a good fit we want R^2 as close to 1 as possible.

Reference for n-fold cross validation is "Ensemble Methods in Data Mining", Seni and Elder. Nice succinct description.

Diagnostics (Continued)

- Sanity check the coefficients
  - Do the signs make sense? Are the coefficients excessively large?
    - Wrong sign is an indication of correlated inputs, but doesn't necessarily affect predictive power.
    - Excessively large coefficient magnitudes may indicate strongly correlated inputs; you may want to consider eliminating some variables, or using regularized regression techniques.
      - Ridge, Lasso
    - Infinite magnitude coefficients could indicate a variable that strongly predicts a subset of the output (and doesn't predict well on the rest).
      - Plot output vs. this input, and see if you should segment the data before regressing.

EMC² PROVEN PROFESSIONAL

Once we determine the fit is good we need to perform the sanity checks. Linear regression is an explanatory model and the coefficients provide the required details.

First check on the sign of the coefficients. Do the signs make sense? For example should the income increase with age or years of education? The coefficients should be positive. If not there might be something wrong. It is often an indicator that the variables are correlated to each other. Regression works best if all the drivers are independent. This does not in fact affect the predictive power but the explanatory capability is compromised here.

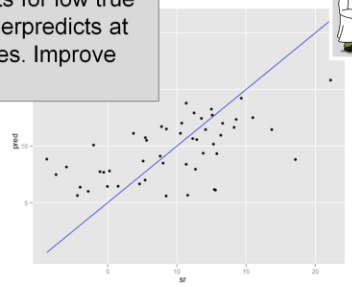
We also need to check if the magnitude of the coefficients make sense? They sometimes can become excessively large and we prefer them not to be very large. This is also an indication of strongly correlated inputs. In this case consider eliminating some variables or use other regularized regression techniques such as Ridge and Lasso (Out of scope for this course). These techniques impose a penalty function on large coefficients and keep them in a desirable range.

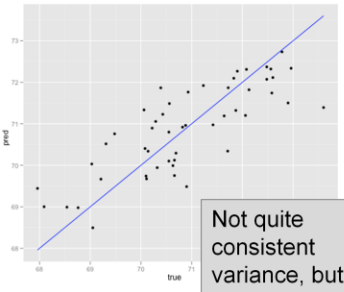
Sometimes you may get infinite magnitude coefficients (R package for OLS will report an error on this) which could indicate that there is a variable that strongly predicts a certain subset of the output and does not predict well on the rest. For example there is a range of age for which the output income is perfectly predicted. In such conditions plot the output vs. the input and determine the segment at which the prediction goes wrong. Then segment the data before fitting the model.

Diagnostics (Continued)

- **Plot it!**
  - Prediction vs. true outcome
- Look for:
  - Systematic over/under prediction
  - Non-consistent variance
    - The data cloud should be symmetric about the line of true prediction
  - Glaring outliers
- You will see other diagnostic plots in the lab

Overpredicts for low true values, underpredicts at higher values. Improve the model.





Not quite consistent variance, but much better.

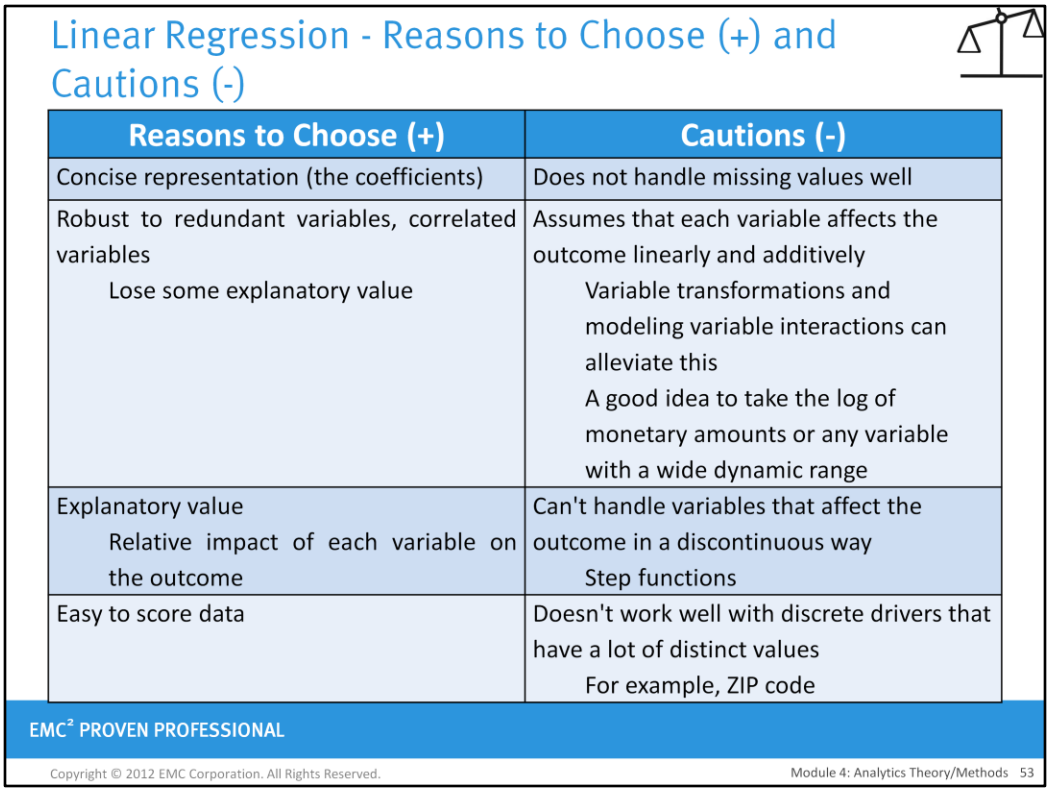
EMC² PROVEN PROFESSIONAL

Even if everything before looks fairly reasonable, it's still a good idea to plot the prediction vs. true outcome. R base package comes with standard graphs but developing plots as the one shown here (generated with ggplot2) is more intuitive for stakeholders to understand.

What you have to look for is that the model does not systematically over predict or under predict in certain ranges. We want the variance to be Consistent (the cloud around the line to be symmetrical). These plots also identify the obvious outliers in the data.

The two graphs show examples of a model in which we over predict for low true values and under predict at higher values. (Graph on the top of the slide).

The graph at the bottom shows an improvement of the model (selecting the correct range, eliminating correlated variables etc). Even in this plot we still do not see a consistent variance but the model fit seems to be better here.



Linear regressions have the explanatory values and we can easily determine how the variables affect the outcome. It is robust to redundant variables and correlated variables. The prediction is not impacted but we lose some explanatory value with the fitted model. Linear regression provides the concise representation of the outcome with the coefficients and it is easy to score the data with this model.

Cautions (-) are that Linear regression does not handle the missing values well. It assumes that each variable affects the outcome linearly and additively. So if we have some variables that affect the outcome non-linearly and the relationships are not actually additive the model does not fit well. Variable transformations and modeling variable interactions can address this to some extent.

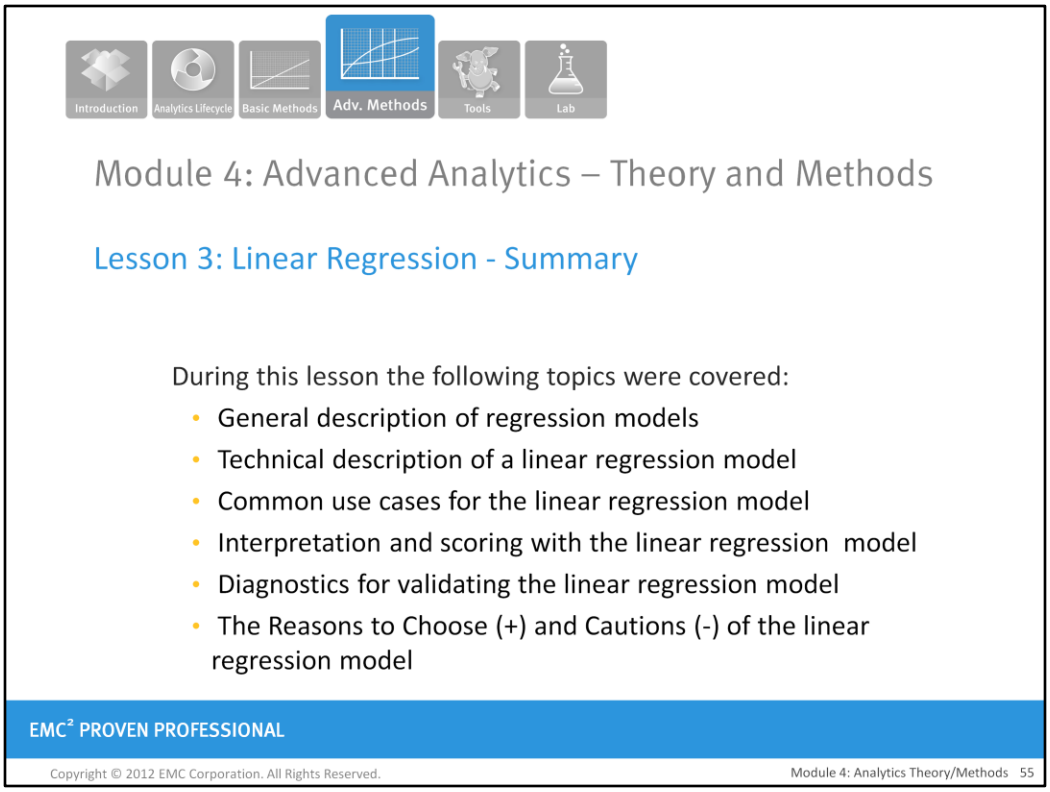
It is recommended to take the log of monetary amounts or any variable with a wide dynamic range. It cannot handle variables that affect the outcome in a discontinuous way. We discussed the issue of infinite magnitude coefficients earlier where the prediction is inconsistent in ranges. Also when you have discrete drivers with a large number of distinct values the model becomes complex and computationally inefficient.

Check Your Knowledge

1. How is the measure of significance used in determining the explanatory value of a driver with linear regression models?
2. Detail the challenges with categorical values in linear regression model.
3. Describe N-Fold cross validation method used for diagnosing a fitted model.
4. List two use cases of linear regression models.
5. List and discuss two standard sanity checks that you will perform on the coefficients derived from a linear regression model.

EMC² PROVEN PROFESSIONAL

Record your answers here.



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

Lab Exercise 6: Linear Regression

This Lab is designed to investigate and practice Linear Regression.

After completing the tasks in this lab you should be able to:

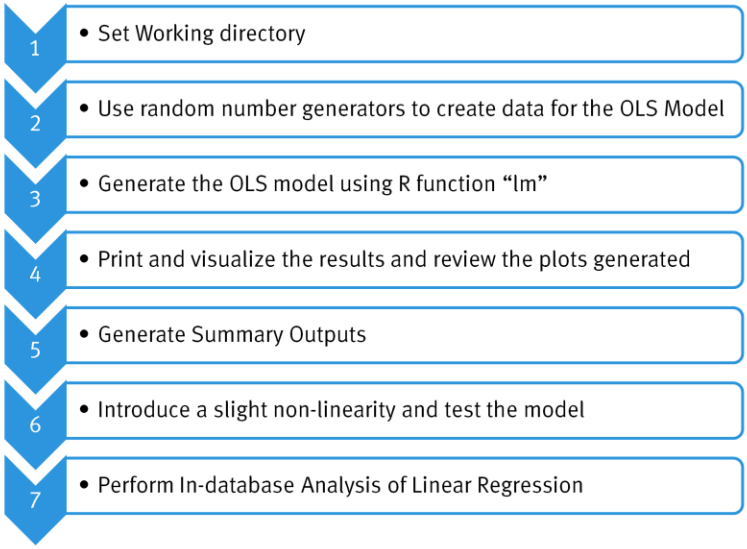
- Use R functions for Linear Regression (Ordinary Least Squares – OLS)
- Predict the dependent variables based on the model
- Investigate different statistical parameter tests that measure the effectiveness of the model

EMC² PROVEN PROFESSIONAL

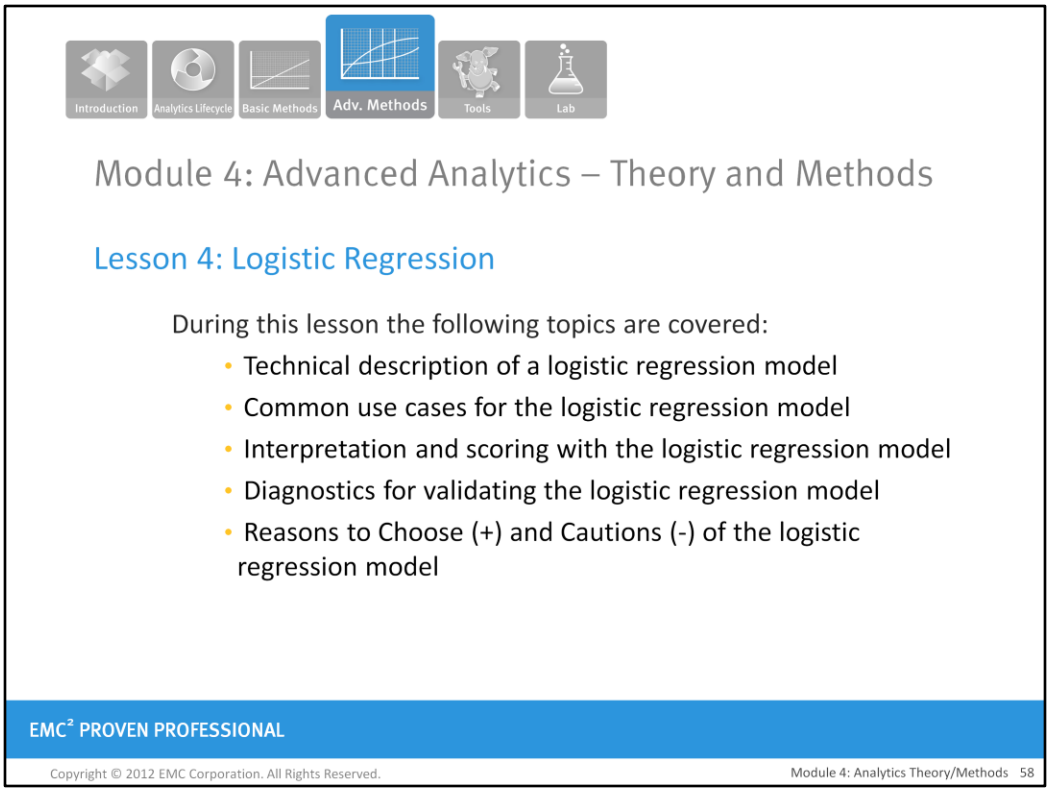
Tasks you will be completing in this lab include:

- Use RStudio environment to code OLS model
- Review the methodology to validate the model and predict the dependent variable for a set of given independent variables
- Use R graphics functions to visualize the results generated with the model
- Use MADlib in-db analytics functions for the linear regression model

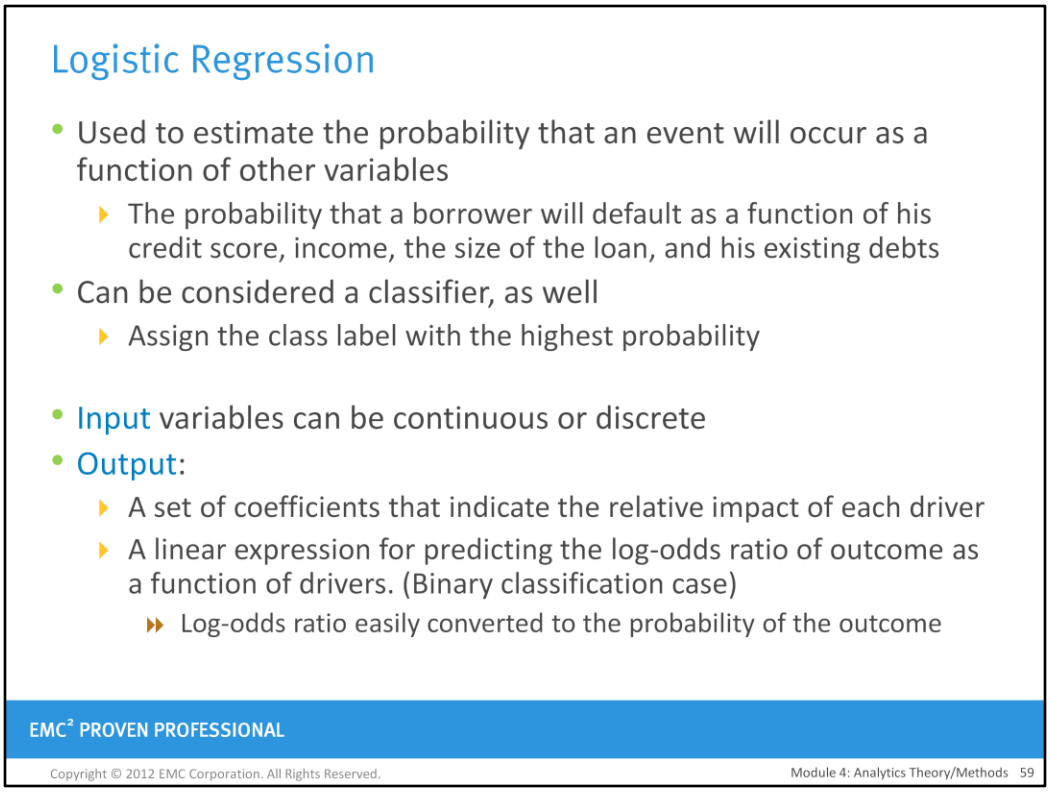
Lab Exercise 6: Linear Regression - Workflow



EMC² PROVEN PROFESSIONAL



The topics covered in this lesson are listed.

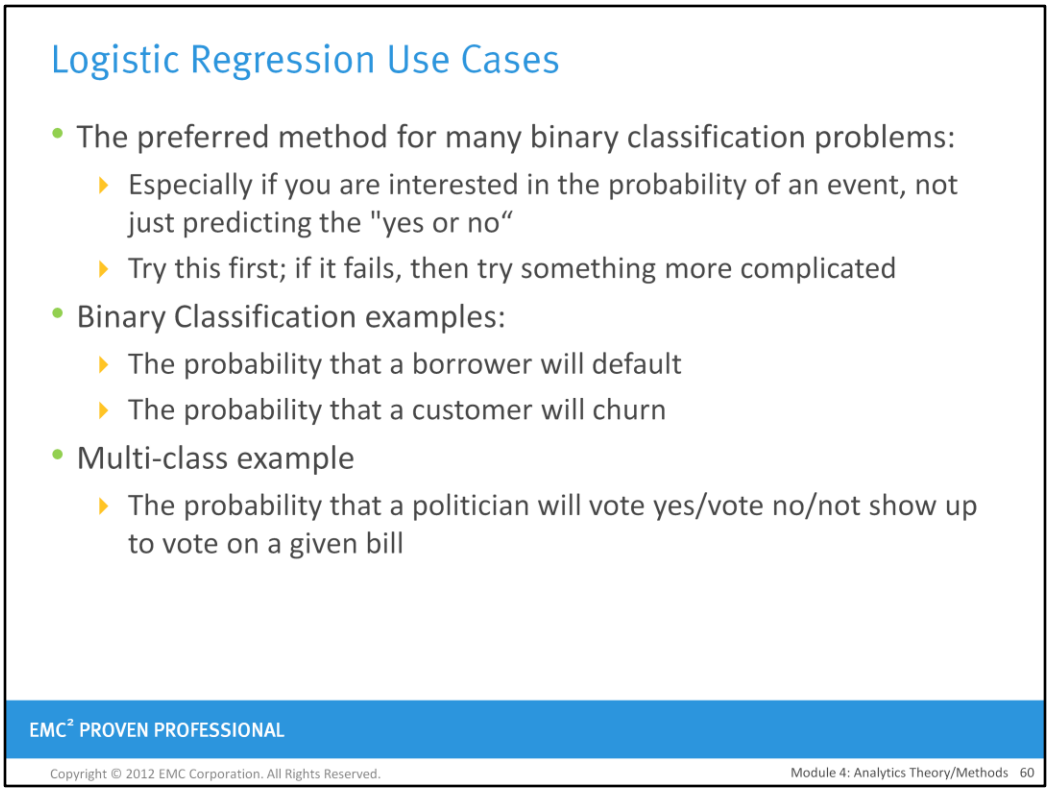


We use logistic regression to estimate the probability that an event will occur as a function of other variables. An example is that the probability that a borrower will default as a function of his credit score , income, loan size, and his current debts.

We will be discussing classifiers in the next lesson. Logistic regression can also be considered a classifier. Recall the discussions on classifiers in lesson 1 of this module(Clustering). Classifiers are methods to assign class labels (default or no_default) based on the highest probability.

In logistic regression input variables can be continuous or discrete. The output is a set of coefficients that indicate the relative impact of each of the input variables.

In a binary classification case (true/false) the output also provides a linear expression for predicting the **log odds** ratio of the outcome as a function of drivers. The log odds ratios can be converted to the probability of an outcome and many packages do this conversion in their outputs automatically.



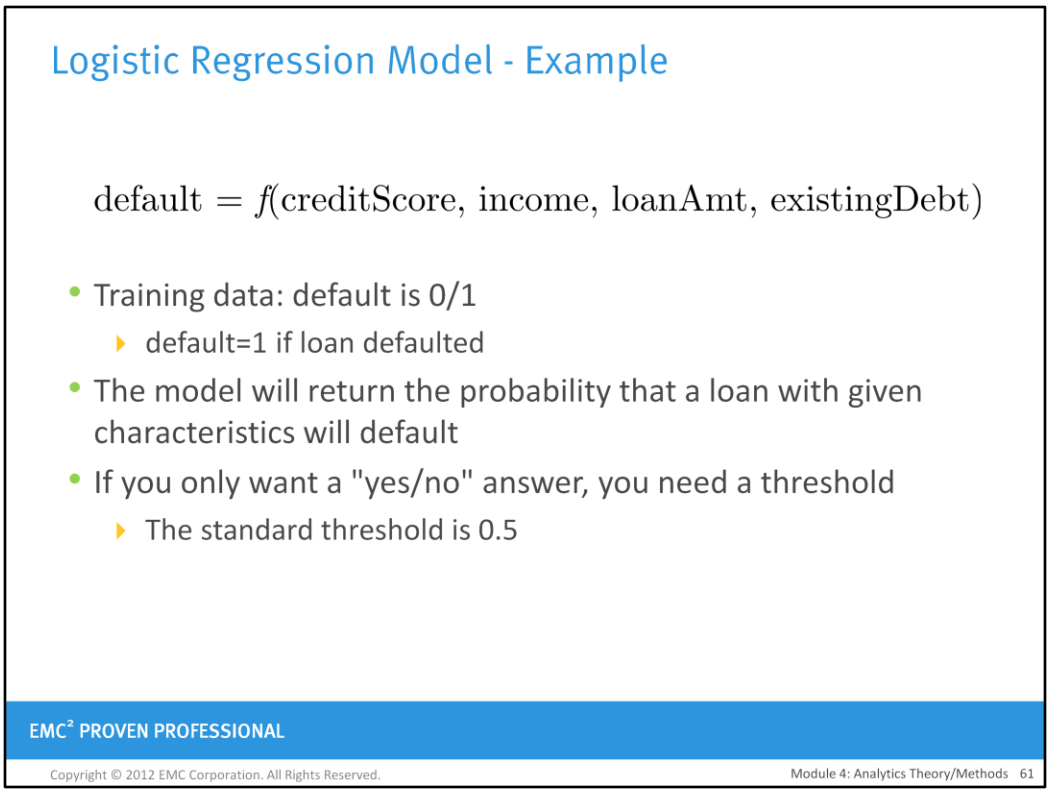
Logistic regression is the preferred method for many binary classification problems

Two examples of a binary classification problem are shown in the slide above. Other examples :

- true/false
- approve/deny
- respond to medical treatment/no response
- will purchase from a website/no purchase
- likelihood Spain will win the next World Cup

The third example on the slide " The probability that a politician will vote yes/vote no/not show up to vote on a given bill" is a multiclass problem. We will only discuss binary problems (such as loan default) for simplicity in this lesson.

Logistic regression is especially useful if you are interested in the probability of an event, not just predicting the class labels. In a binary class problem Logistic regression must be tried first to fit a model. And only if it does not work models such as GAMS (generalized additive methods), Support Vector Machines and Ensemble Methods are tried (these models are out of scope for this course).



The slide shows an example “Probability of Default”

Default (output for this model) is defined as a function of credit score, income, loan amount and existing debt.

The training data represents the default as either 0 or 1 where default = 1 if the loan is defaulted.

Fitting and scoring the logistic regression model will return the probability that a loan with a given value for each of the input variables will default.

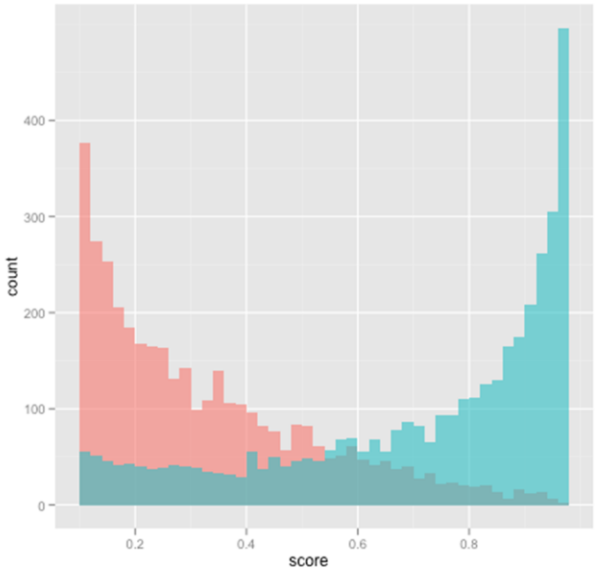
If only Yes/No type answer is desired a threshold must be set for the value of probability to return the class label. The standard threshold is 0.5.

Logistic Regression- Visualizing the Model

Overall fraction of default: ~20%

Logistic regression returns a score that estimates the probability that a borrower will default

The graph compares the distribution of defaulters and non-defaulters as a function of the model's predicted probability, for borrowers scoring higher than 0.1



Blue=defaulters

EMC² PROVEN PROFESSIONAL

This is an example of how one might visualize the model. Logistic regression returns a score that estimates the probability that a borrower will default. The graph compares the distribution of defaulters and non defaulters as a function of model's predicted probability for borrowers scoring higher than 0.1 and less than 0.98

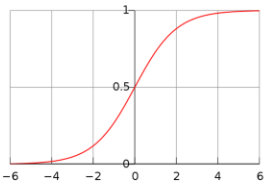
The graph is overlaid – think of the blue graph (defaulters) as being transparent and "in front of" the red graph (non defaulters).

The takeaway from the graph is that the higher a borrower scores, the more likely empirically that he will default.

The graph only considers borrowers who score > 0.1 and < 0.98 because this graph had large spikes near 0 and 1, so the graph becomes hard to read. We can see, however, that a fraction of low scoring borrowers do actually default. (the overlap)

Technical Description (Binary Case)

$$\ln \frac{P(y=1)}{1-P(y=1)} = b_0 + b_1 x_1 + b_2 x_2 \ldots$$



- y=1 is the case of interest: 'TRUE'
- LHS is called logit(P(y=1))
  - hence, "logistic regression"
- logit(P(y=1)) is inverted by the sigmoid function
  - standard packages can return probability for you
- Categorical variables are expanded as with linear regression
- Iterative, not closed form solution
  - "Iteratively re-weighted least squares"

EMC² PROVEN PROFESSIONAL

The quantity on LHS (Left Hand Side) is the log odds ratio. We first compute the ratio of probability of y equal to 1 vs. the probability of y not equal to 1 and take a log of this ratio. In logistic regression the log odds ratio is equal to linear additive combination of the drivers.

LHS is called logit(P(y=1)) and hence this method came to be known as logistic regression. The inverse of the logit is the sigmoid function. The output of the sigmoid is the actual probabilities. Standard packages give the inverse as a standard output.

Categorical values are expanded exactly the way we did in the linear regression. Computing the coefficients is also done as least square method but implemented as iteratively re-weighted least squares converging to the true probabilities with every iteration.

Logistic regression has exactly the same problems that a OLS method has and the computational complexity increases with more input variables and with categorical values with multiple levels.

What do the Coefficients b_j Mean?

- Invert the logit expression:

$$\frac{P(y=1)}{1-P(y=1)} = \exp(\sum_{j=0}^{K} b_j x_j)$$

$$= \prod_{j=0}^{K} \exp(b_j x_j)$$

- $\exp(b_j)$ tells us how the odds-ratio of y=1 changes for every unit change in x_j
- Example: $b_{creditScore}$ = -0.69
  - $\exp(b_{creditScore}) = 0.5 = 1/2$
  - for the same income, loan, and existing debt, the odds-ratio of default is halved for every point increase in credit score
- Standard packages return the significance of the coefficients in the same way as in linear regression

EMC² PROVEN PROFESSIONAL

If we invert the logit expression shown in the slide, we come up with the logit as a product of the exponents of the coefficients times the drivers.

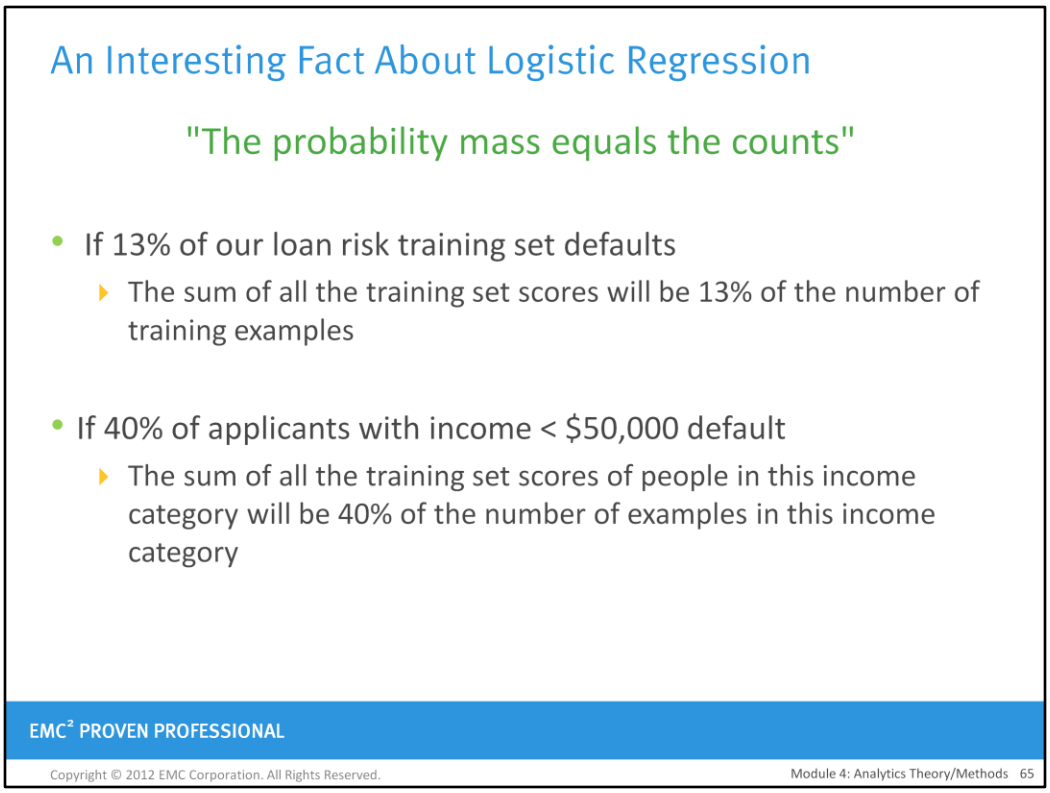
The exponent of the first coefficient, b_0, represents the odds-ratio of the outcome in the "reference situation" – the situation that is represented by all the continuous variables set to zero, and the categorical variables at their reference

That means the exponent of the coefficients $\exp(b_j)$ tells us how the odds-ratio of y=1 changes for every unit change in x_j

Suppose we have $b_{creditScore}$ = -3 implies exp(-69) = 0.5 = 1/2

This means for the same income, loan amount, existing debt, the odds ratio of default is cut in half for every point of increase of credit score. The negative number on the coefficient indicates that there is a negative relation between the credit score and the probability of default. Higher credit score implies lower probability of default.

Significance of the credit score is returned in the same way as in linear regression. So you should look for very low “p” values.



"Logistic regression preserves summary statistics of the training data" – in other words, logistic regression is a very good way of concisely describing the probability of all the different possible combination of features in the training data.

Two examples of this feature are shown in the slide. If you sum up everybody's score after putting them through the model the total computed will be equal to the sum of all the training set scores.

What this means is that it is almost like a continuous look up probability table. Assume that we have all categorical variables and you have the table of probability of every possible combination of variables, Logistic regression is a concise version of the table. This is what can be defined as a "well calibrated" model.

Reference: http://www.win-vector.com/blog/2011/09/the-simpler-derivation-of-logistic-regression/

Diagnostics

- Hold-out data:
  - Does the model predict well on data it hasn't seen?
- N-fold cross-validation: Formal estimate of generalization error
- "Pseudo-R^2" : 1 – (deviance/null deviance)
  - Deviance, null deviance both reported by most standard packages
  - The fraction of "variance" that is explained by the model
  - Used the way R^2 is used

EMC² PROVEN PROFESSIONAL

This is all very similar to linear regression. We use the hold-out data method, and N-fold cross validation on the fitted model. This is exactly what we did with linear regression to determine if the model predicts well.

The model should explain more than just this simple guess. Pseudo R^2 is the term we use in Logistic regression which we use the same way we use R^2 in linear regression. It is basically "the fraction" of the variance .

Deviance, for the purposes of this discussion, is analogous to "variance" in linear regression.

The null deviance is the deviance (or "error') that you would make if you always assumed that the probability of true were simply the global probability.

1 – (deviance/null deviance) is the "fraction" that defines Pseudo R^2 which is a measure of how well the model explains the data.

Diagnostics (Cont.)

- Sanity check the coefficients
  - Do the signs make sense? Are the coefficients excessively large?
    - Wrong sign is an indication of correlated inputs, but doesn't necessarily affect predictive power.
    - Excessively large coefficient magnitudes may indicate strongly correlated inputs; you may want to consider eliminating some variables**, or using regularized regression techniques.**
      - Unfortunately, regularized logistic regression is not standard.
    - Infinite magnitude coefficients could indicate a variable that strongly predicts a subset of the output (and doesn't predict well on the rest).
      - Try a Decision Tree on that variable, to see if you should segment the data before regressing.

EMC² PROVEN PROFESSIONAL

The sanity checks are exactly the same as what we discussed in linear regression.

Once we determine the fit is good we need to perform the sanity checks. Logistic regression is an explanatory model and the coefficients provide the required details.

First check the sign of the coefficients. Do the signs make sense. For example, should the income increase with age or years of education? The coefficients should be positive. If not there might be something wrong. It is often an indicator that the variables are correlated to each other. Regression works best if all the drivers are independent. This does not in fact affect the predictive power but the explanatory capability is compromised here.

We also need to check if the magnitude of the coefficients make sense? They sometimes can become excessively large and we prefer them not to be very large. This is also an indication of strongly correlated inputs. In this case consider eliminating some variables. **Note that unlike linear regression, where we have regularized regression techniques, there are not any standard methods with logistic regression. If there is a requirement one should implement one's own method.**

Sometimes you may get infinite magnitude coefficients which could indicate that there is a variable that strongly predicts a certain subset of the output and does not predict well on the rest. For example there is a range of age for which the output income is perfectly predicted. In such conditions plot the output vs. the input and determine the segment at which the prediction goes wrong. We should then segment the data before fitting the model. Decision Trees can be used on that variable, to see if you should segment the data before regressing.

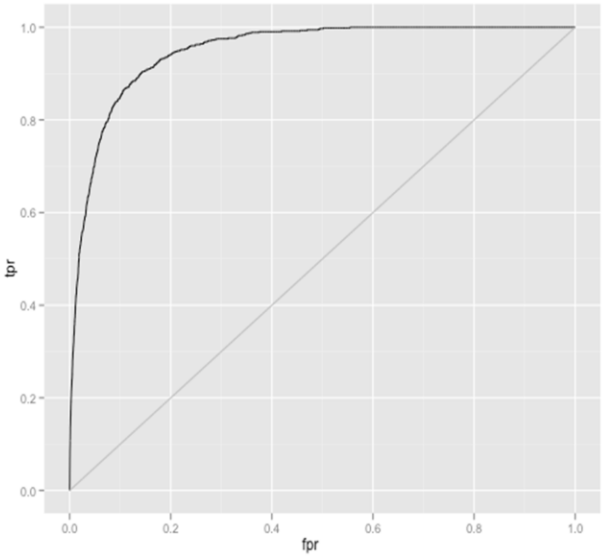
Diagnostics: ROC Curve

$$\text{FPR} = \frac{\#\text{ false positives}}{\text{all negatives}}$$

$$\text{TPR} = \frac{\#\text{ true positives}}{\text{all positives}}$$

Area under the curve (AUC) tells you how well the model predicts. (Ideal AUC = 1)

For logistic regression, ROC curve can help set classifier threshold



EMC² PROVEN PROFESSIONAL

Logistic models do very well at predicting class probabilities; but if you want to use them as a classifier you have to set a threshold. For a given threshold, the classifier will give false positives and false negatives. False positive rate (fpr) is the fraction of negative instances that were misclassified.

False negative rate (fnr) is the fraction of positive instances that were misclassified. True positive rate (tpr) = 1 – fnr

The ROC (Receiver Operating Characteristics) curve plots (fpr, tpr) as the threshold is varied from 0 (the upper right hand corner) to 1 (the lower left hand corner).

As the threshold is raised, the false positive rate decreases, but the true positive rate decreases, too.

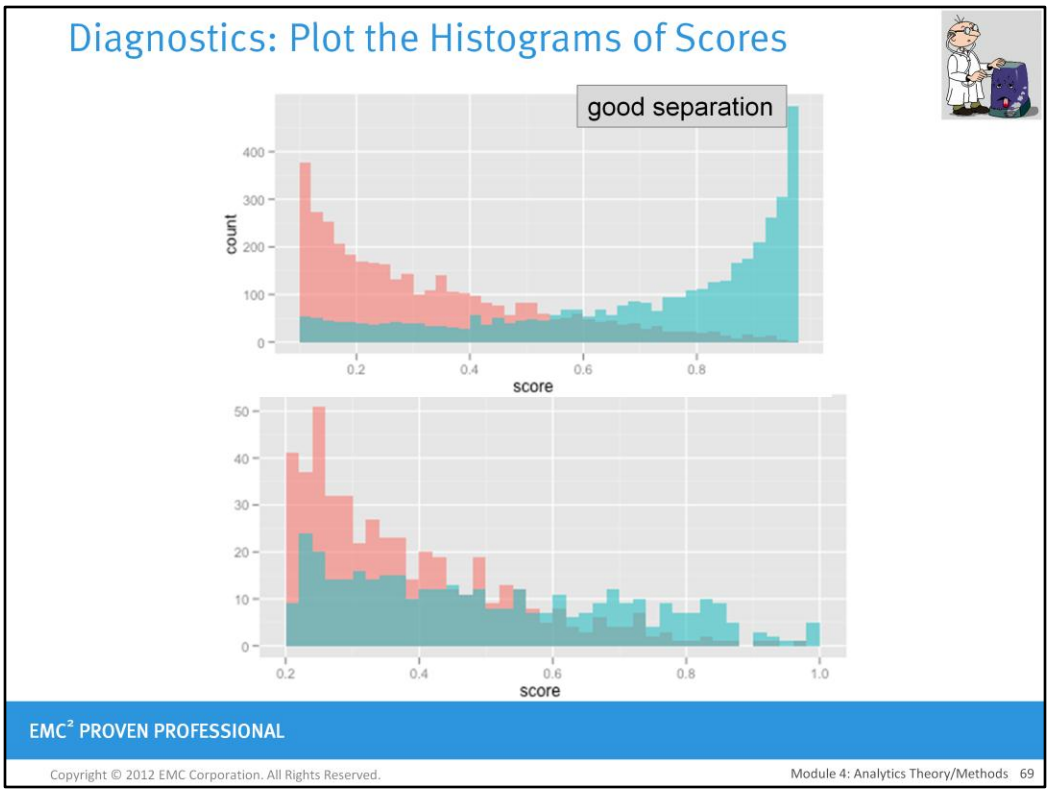
The ideal classifier (only true instances have probability near 1) would trace the upper left triangle of the unit square: as the threshold increases, fpr decreases without lowering tpr.

Usually, ROC curves are only used to evaluate prediction quality – how close the AUC is to 1. But they can also be used to set thresholds; if you have upper bounds on your desired fpr and fnr, you can use the ROC curve (or more accurately, the software that you use to plot the ROC curve) to give you the range of thresholds that meet those constraints.

For logistic regression, the ROC curve can help set the classifier threshold.

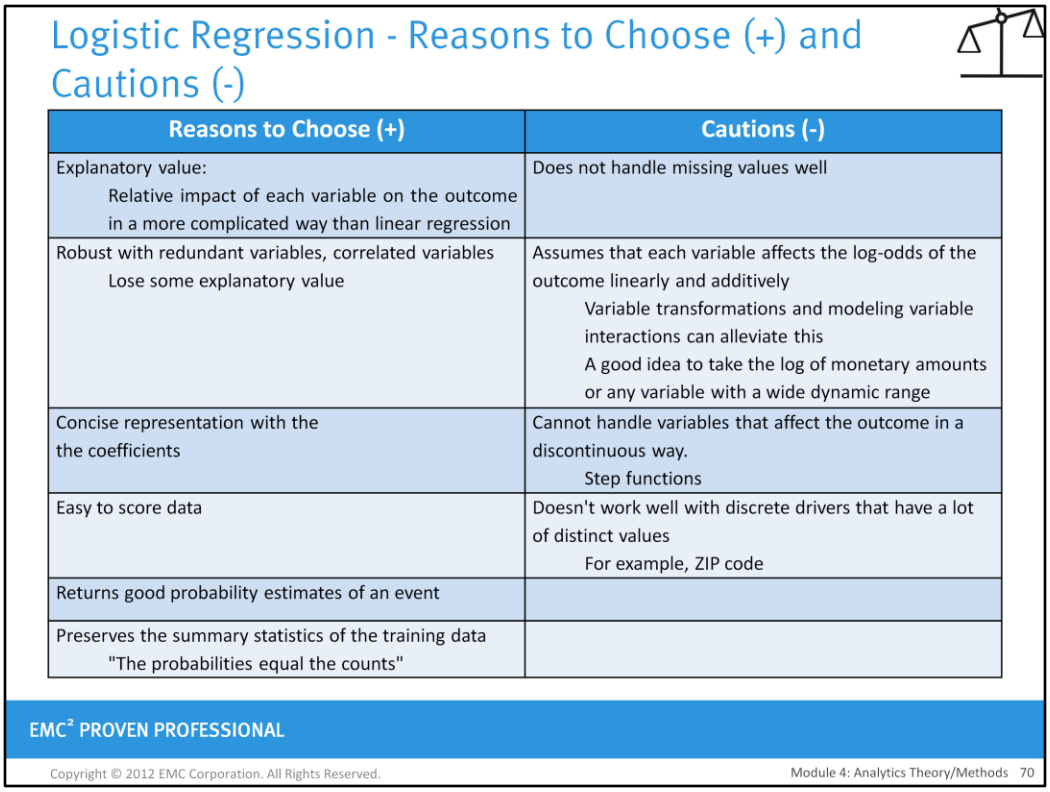
An excellent primer on ROC is available in the following reference:

http://home.comcast.net/~tom.fawcett/public_html/papers/ROC101.pdf



The next diagnostic method is plotting the histogram of the scores. The graph in the top half is what we saw earlier in the lesson. The graph tells us how well the model discriminates true instances from false instances. Ideally, true score high and false instances score low. If so, most of the mass of the two histograms are separated. That is what you see in the graph at the top.

The graph shown at the bottom shows substantial overlap. The model did not predict well. This means the input variables are not strong predictors of the output.



Logistic regressions have the explanatory values and we can easily determine how the variables affect the outcome. The explanatory values are a little more complicated than linear regression. It works well with (robust) redundant variables and correlated variables. In this case the prediction is not impacted but we lose some explanatory value with the fitted model. Logistic regression provides the concise representation of the outcome with the coefficients and it is easy to score the data with this model. Logistic regression returns probability estimates of an event. It also returns calibrated model it preserves the summary statistics of the training data.

Cautions (-) are that the Logistic regression does not handle missing values well. It assumes that each variable affects the log odds of the outcome linearly and additively. So if we have some variables that affect the outcome non-linearly and the relationships are not actually additive the model does not fit well.

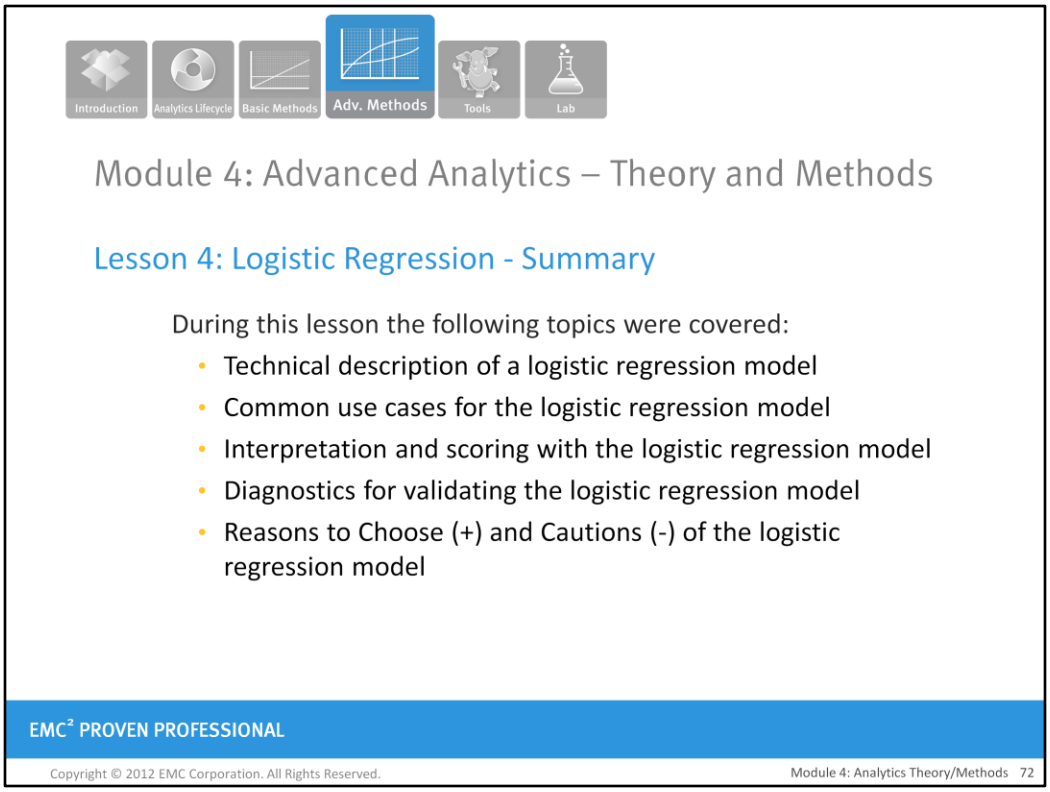
Variable transformations and modeling variable interactions can address this to some extent. It is recommended to take the log of monetary amounts or any variable with a wide dynamic range. It cannot handle variables that affect the outcome in a discontinuous way. We discussed the issue of infinite magnitude coefficients earlier where the prediction is inconsistent in ranges. Also when you have discrete drivers with a large number of distinct values the model becomes complex and computationally inefficient.

Check Your Knowledge

1. What is a logit and how do we compute class probabilities from the logit?
2. How is ROC curve used to diagnose the effectiveness of the logistic regression model?
3. What is Pseudo R^2 and what does it measure in a logistic regression model?
4. How do you describe a binary class problem?
5. Compare and contrast linear and logistic regression methods.

EMC² PROVEN PROFESSIONAL

Record your answers here.



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

Lab Exercise 7: Logistic Regression

This Lab is designed to investigate and practice Logistic Regression.

After completing the tasks in this lab you should be able to:

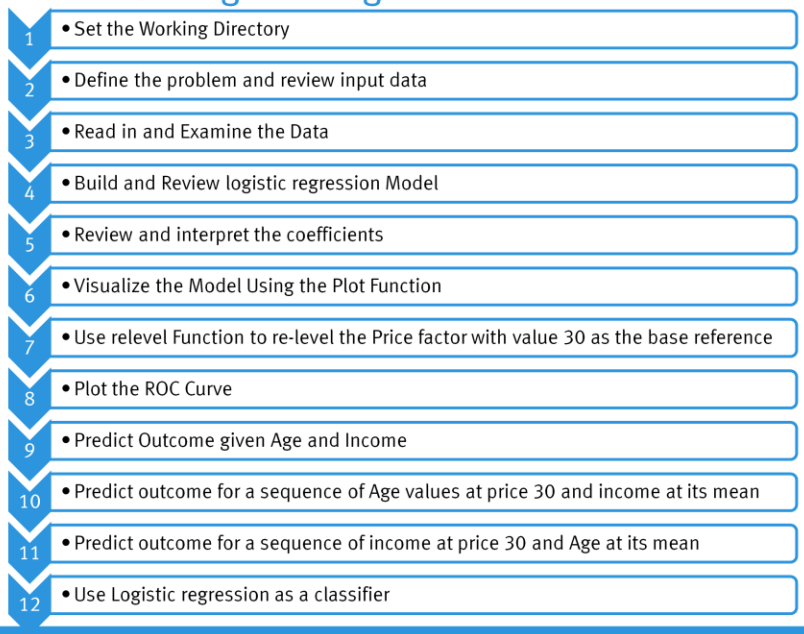
- Use R functions for Logistic Regression – (*also known as Logit*)
- Predict the dependent variables based on the model
- Investigate different statistical parameter tests that measure the effectiveness of the model

EMC² PROVEN PROFESSIONAL

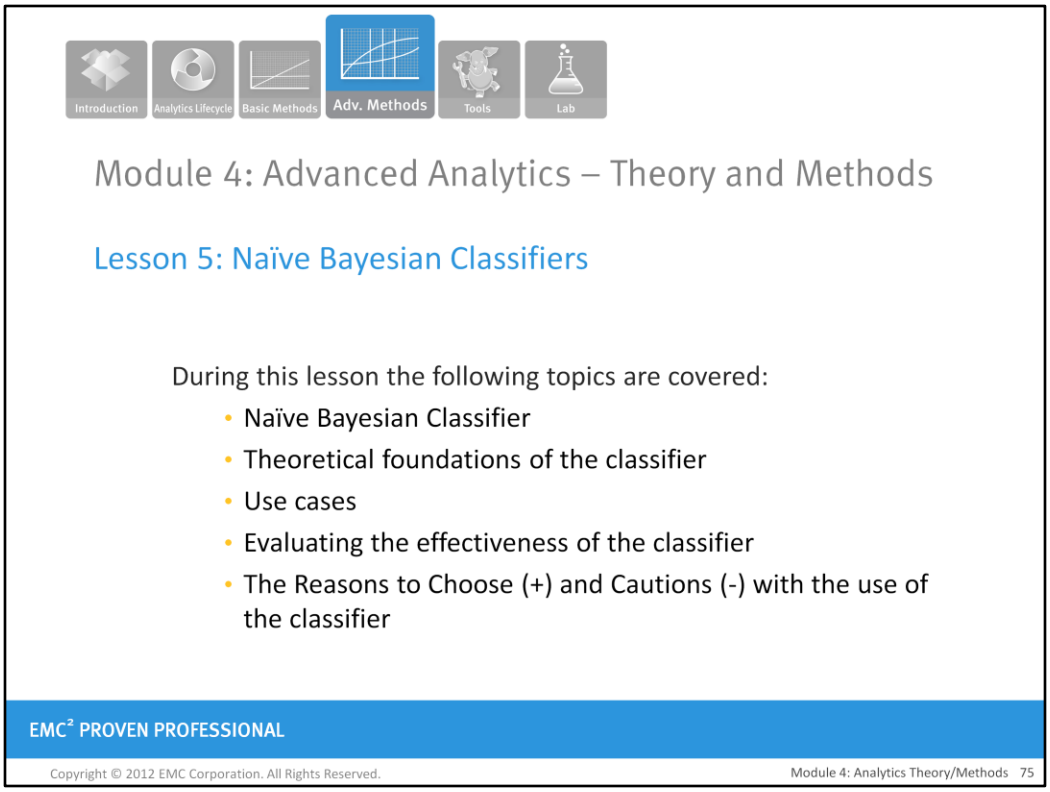
Tasks you will be completing in this lab include:

- Use RStudio environment to code Logit models
- Review the methodology to validate the model and predict the dependent variable for a set of given independent variables
- Use R graphics functions to visualize the results generated with the model

Lab Exercise 7: Logistic Regression - Workflow



EMC² PROVEN PROFESSIONAL

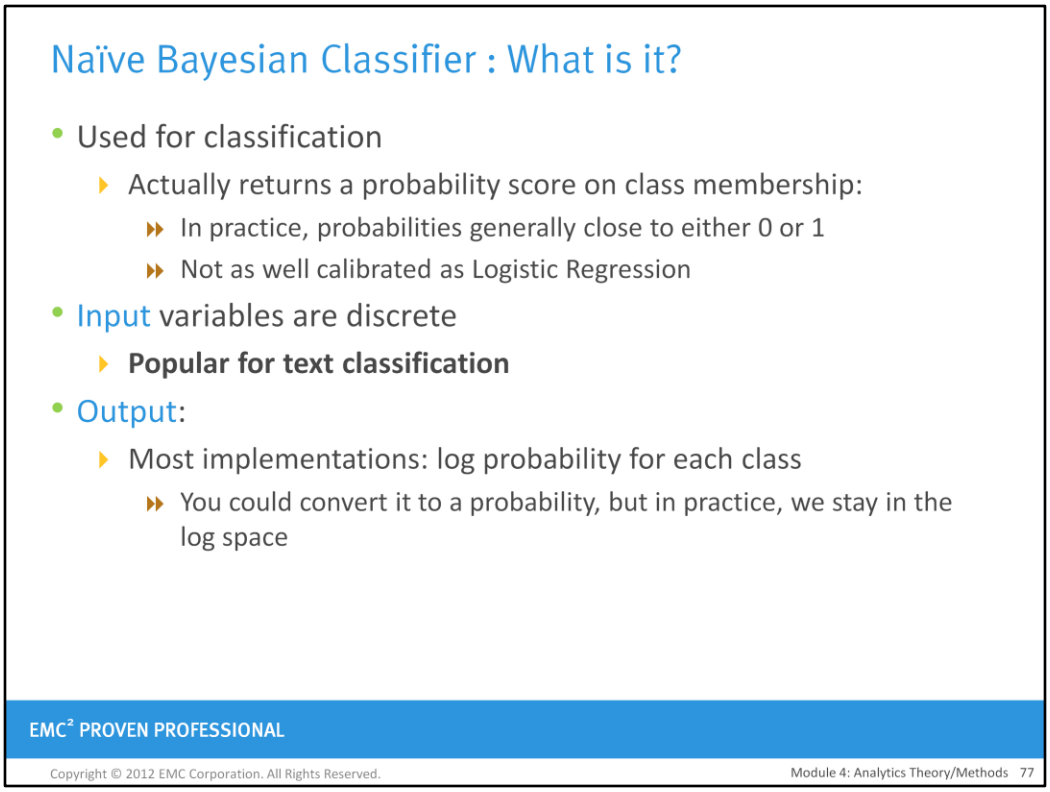


The topics covered in this lesson are listed.

The primary task performed by the Classifiers is assigning labels to objects. Labels in classifiers are pre-determined unlike in clustering we discover the structure and assign labels. Classifier problems are supervised learning methods. We start with a training set of pre-classified examples and with the knowledge of prior probabilities we assign class labels.

Some use case examples are shown in the slide. Based on the voting pattern on issues we could use them to classify whether a politician has an affiliation to a party or a principle. Retailers use classifiers to assign proper catalog entry locations for their products. Most importantly the classification of emails as spam is another useful application of this method.

Logistic regression that we discussed in the previous lesson can be viewed and used as a classifier. We will discuss Naïve Bayesian classifier in this lesson and Decision Trees, the example of another classifier covered in the next lesson.



The Naïve Bayesian Classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong *naïve independence assumptions*. In simple terms, a Naïve Bayes Classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.

For example, an object can be classified into a particular category based on its properties such as shape, size, and color. (A tennis ball is round, 4 inches in diameter and is yellow in color). Even if these features depend on each other or upon the existence of the other features, a Naïve Bayesian classifier considers all of these properties to independently contribute to the probability that the object is a tennis ball.

Depending on the precise nature of the probability model, Naïve Bayes Classifiers can be trained very efficiently in a supervised learning setting. Bayesian reasoning is applied to decision making and inferential statistics that deal with probability inference. It uses the knowledge of prior events to predict future events. Bayesian Classifiers are widely used for text classification.

The input variables are generally discrete but there are variations to the algorithms that work with continuous variables as well. The outputs are the probability scores. These scores are generally very close to 0 or 1 and are not as well calibrated as with logistic regression.

The output usually returns a probability score and class membership. The output from most of the implementations are log probabilities for the class (we will detail this later in the lesson) and we assign the class label based on the highest probability.

Naïve Bayesian Classifier - Use Cases

- Preferred method for many text classification problems.
  - Try this first; if it doesn't work, try something more complicated
- Use cases
  - Spam filtering, other text classification tasks
  - Fraud detection

EMC² PROVEN PROFESSIONAL

Naïve Bayesian Classifiers are among the most successful known algorithms for learning to classify text documents. Spam filtering is the best known use of Naïve Bayesian Text Classification. Bayesian Spam Filtering has become a popular mechanism to distinguish illegitimate spam email from legitimate email. Many modern mail clients implement Bayesian Spam Filtering.

Naïve Bayesian Classifiers are used to detect fraud. For example in auto insurance, based on a training data set with attributes (such as driver's rating, vehicle age/price, is it a claim by the policy holder, police report status, claim genuine) we can classify a new claim as genuine or not.

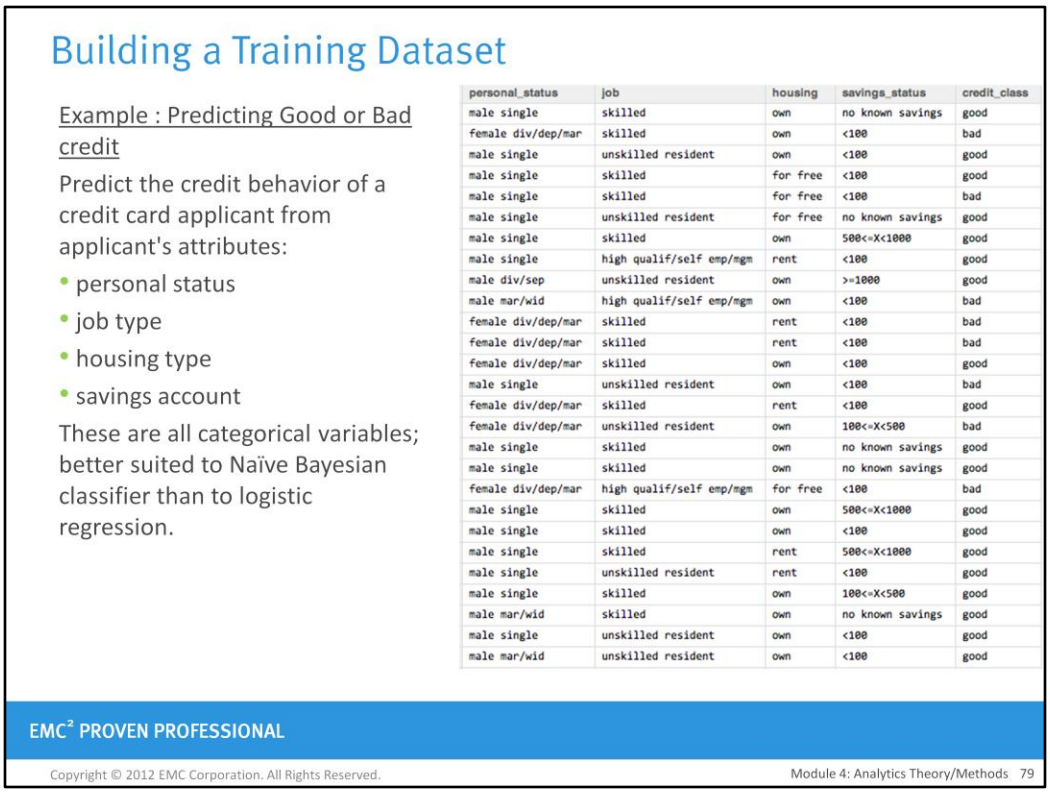
References:

Spam filtering (http://en.wikipedia.org/wiki/Bayesian_spam_filtering)

http://www.cisjournal.org/archive/vol2no4/vol2no4_1.pdf

Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering (http://eprints.ecs.soton.ac.uk/18483/

Online applications (http://www.convo.co.uk/x02/)



Let us look into a specific use case example. We present here the same example we worked with in Lesson 2 of this module with the Apriori algorithm. The training data set consists of attributes personal status, job type, housing type and amount of money in their savings account. They are represented as categorical variables which are better suited for Naïve Bayesian classifier.

With this training set we want to predict the credit behavior of a new customer. This problem could be solved with logistic regression as well. If there are multiple levels for the attribute you want to predict then Naïve Bayesian Classifier is a better solution.

Technical Description - Bayes' Law

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

- B is the class label:
  - $B \in \{b_1, b_2, \ldots b_n\}$
- A is the specific assignment of input variables
  - $A = (a_1, a_2, \ldots a_m)$

Reverend Thomas Bayes

EMC² PROVEN PROFESSIONAL

Bayes' Law says : P(B | A)*P(A) = P(A | B)*P(B) = P(A ^ B).

That is, the probability that B is true given that A is true, times the probability of A is the same as the probability that A is true given that B is true, times the probability of B. Both of these are the same as the probability P(A^B) that A and B are simultaneously true. If we divide all three terms by P(A), then we get the form shown on the slide.

If P(B |A) is the POSTERIOR probability of observing a specific class label, given that we have observed the input variables A, then Bayes law says that this posterior probability is the same as the probability of observing A given that we are in class B, times the PRIOR probability of being in class B, divided by the probability of being in class A.

The reason this is important is that we don't know P(B | A) (and we want to). We DO know P(A |B) and P(B) from the training data. We don't (usually) know P(A), but that turns out to be okay, as we will see.

An Example:

John flies frequently and likes to upgrade his seat to first class.

He has determined that, if he checks in for his flight at least two hours early, the probability that he will get the upgrade is .75; otherwise, the probability that he will get the upgrade is .35. With his busy schedule, he checks in at least two hours before his flight only 40% of the time.

Suppose John didn't receive an upgrade on his most recent attempt. What is the probability that he arrived late?

X – John Arrived late

Y – John did not receive an upgrade

P(X) = Prior probability of arriving late = .6

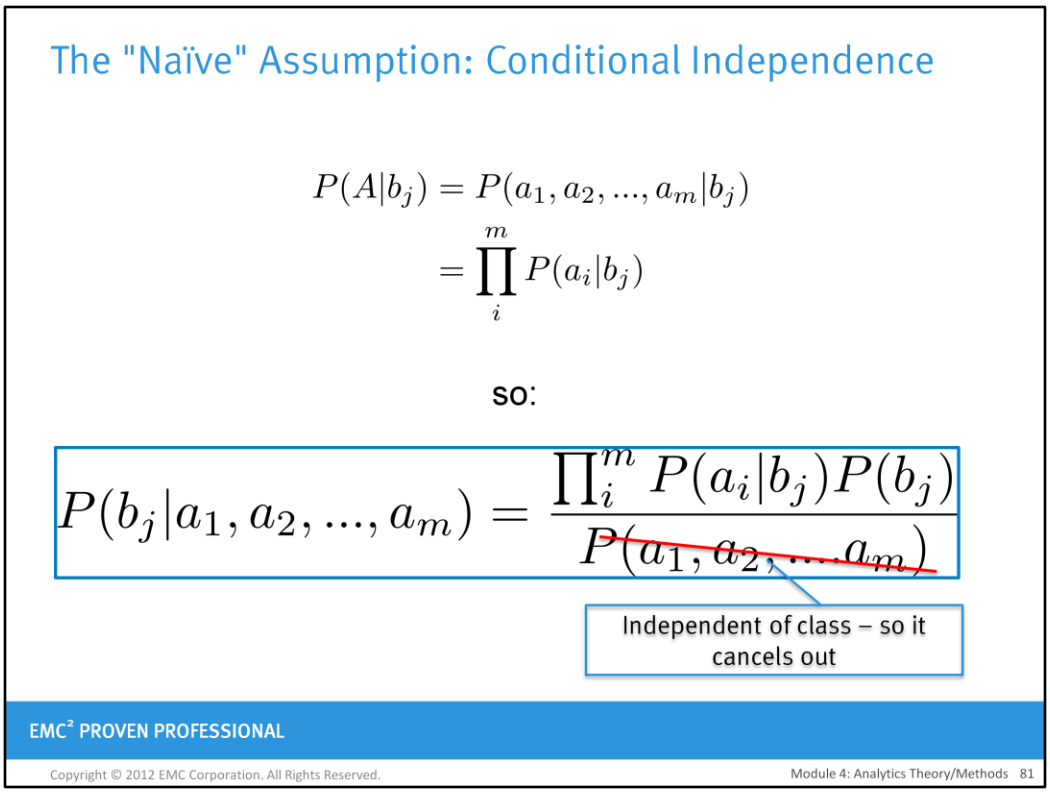
P(Y) = Prior probability of not receiving an upgrade = 1 – (.4 x .75 + .6 x .35) = 1 -.51 = .49

P(Y/X) = Probability that John did not receive an upgrade given that he arrived late = .65

P(X/Y) = Probability that John arrived late given that he did not receive his upgrade = ? = (.65 x .6)/.49 = .80 (approx)

Reference:

"In praise of Bayes." *The Economist* (September, 2000)
http://www.economist.com/node/382968?Story_ID=382968

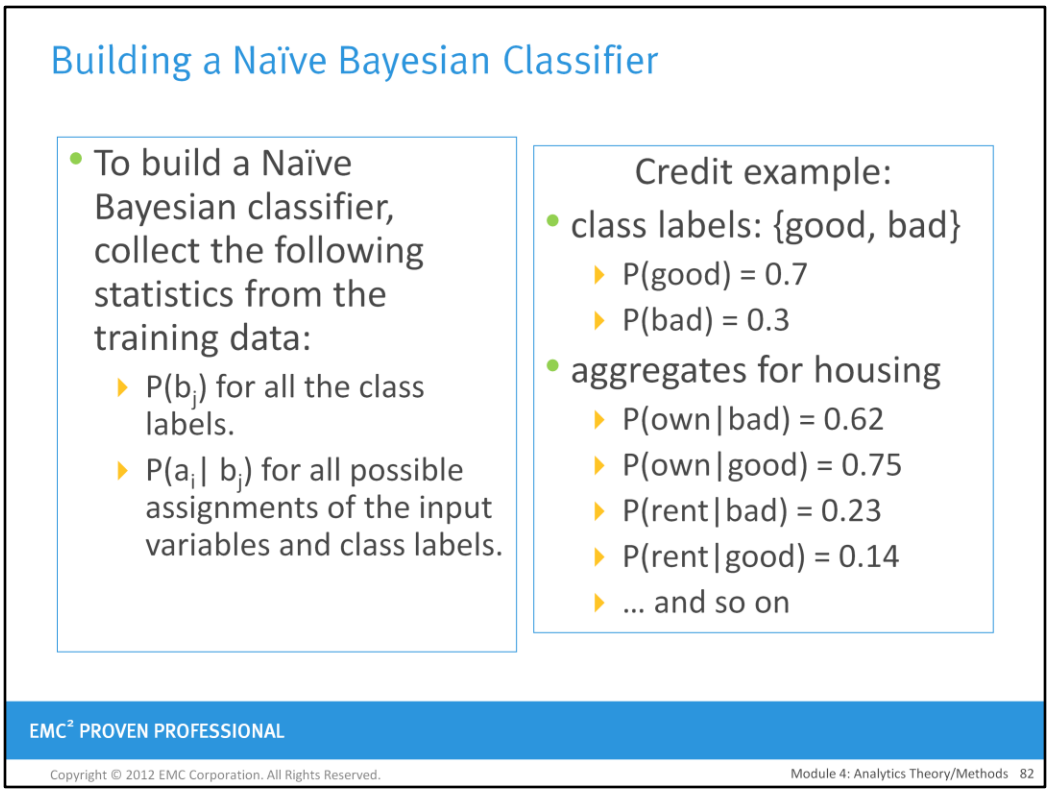


Here we present the equation that represents the Bayesian classifier.

For an attribute A we have m classes a_1,a_2,…..a_m

The probability of A given a set of j values of b is the product of the conditional probability of every a_i given b_j.

This product applying the Bayesian theorem is represented with the formula highlighted above (boxed). The Naïve Bayesian Conditional Independence Assumption comes into play here. The assumption here is that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(A|b_j)$.



To build a Naïve Bayesian classifier we need to collect the following statistics:

1. Probability of all class labels – Probability of good credit and probability of bad credit. From the all data available in the training set we determine P(good) = 0.7 and P(bad) = 0.3
2. We have several attributes in our training set. “own house”, “Job skilled”, “Female_Single” etc. For each assignment of input variable and class label for example, own house / credit good , own house/credit bad, job skilled/credit good, job skilled/ credit bad we compute the conditional probabilities as shown in the slide.

Building a Naïve Bayesian Classifier (Continued)

- Assign the label that maximizes the value

$$\prod_{i}^{m} P(a_i|b_j)P(b_j)$$

EMC² PROVEN PROFESSIONAL

Having computed the probabilities of the class labels and the probability of each attribute given the class label value, we compute the product of these two components.

The label is assigned based on the maximum of the values we get from the equation shown in the slide.

Let us review this with our credit example.

Back to Credit Example

P(good|X) ~ (0.28*0.75*0.14*0.06)*0.7 = 0.0012

P(bad|X) ~ (0.36*0.62*0.17*0.02)*0.3 = 0.0002

Credit Example: X

- female
- owns home
- Self-employed
- savings > $1000

P(good|X) > P(bad|X):
Assign X the label "good"

a_i	b_j	$P(a_i \mid b_j)$
female	good	0.28
female	bad	0.36
own	good	0.75
own	bad	0.62
self emp	good	0.14
self emp	bad	0.17
savings>1K	good	0.06
savings>1K	bad	0.02

EMC² PROVEN PROFESSIONAL

Here we have an example of a person X who is female, owns a home, is self-employed and has savings over $1000 in her savings account. How will we classify this person? Will she be scored as a person with good or bad credit?

Having built the classifier with the training set we find the P(good|X) which is equal to 0.0012 (see the computation on the slide) and P(bad|X) is 0.0002. The maximum of the two values is used to assign the label. P(good|X) is the maximum of the two and we assign the label “good”.

The score is only proportional to the probability,. It doesn't equal the probability, because we haven't included the denominator. But both formulas have the same denominator, so we don't need to calculate it in order to know which quantity is bigger.

In fact, if we normalize the values, we will get p(good|X)=84.4%, and p(bad|X)=15.5%

Notice, though, how small in magnitude these scores are. When we are looking at problems with a large number of attributes, or attributes with a very high number of levels, these values can become very small in magnitude.

Implementation Guideline

- High-dimensional problems are prone to numerical underflow and unobserved events; it's better to calculate the log probability (with smoothing).

$$\sum_{i}^{m} \log(P(a_i|b_j) + \epsilon) + \log(P(b_j) + \epsilon)$$

(Smoothing technique varies with implementation)

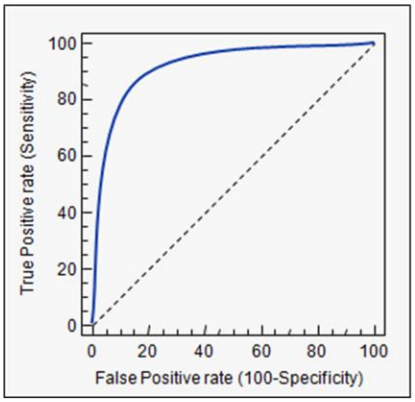
EMC² PROVEN PROFESSIONAL

Multiplying several probability values (< 1) invariably leads to the problem of numerical underflow. So an important implementation guideline is that the log of probability added with a smoothing value should be computed. This recommendation is not just for high-dimensional problems but for all implementations.

The R implementation of Naïve Bayes incorporates the smoothing directly into the probability tables, so we don't need to use an epsilon. Essentially, the Laplace smoothing that R uses adds one (or a small value) to every count, so for example, if we have 100 "good" customers, and 20 of them own their homes, the "raw" P(own | good) = 20/100 = 0.2; with Laplace smoothing using "1", the calculation would be P(own | good) = (20 + 1)/100 = 0.21.

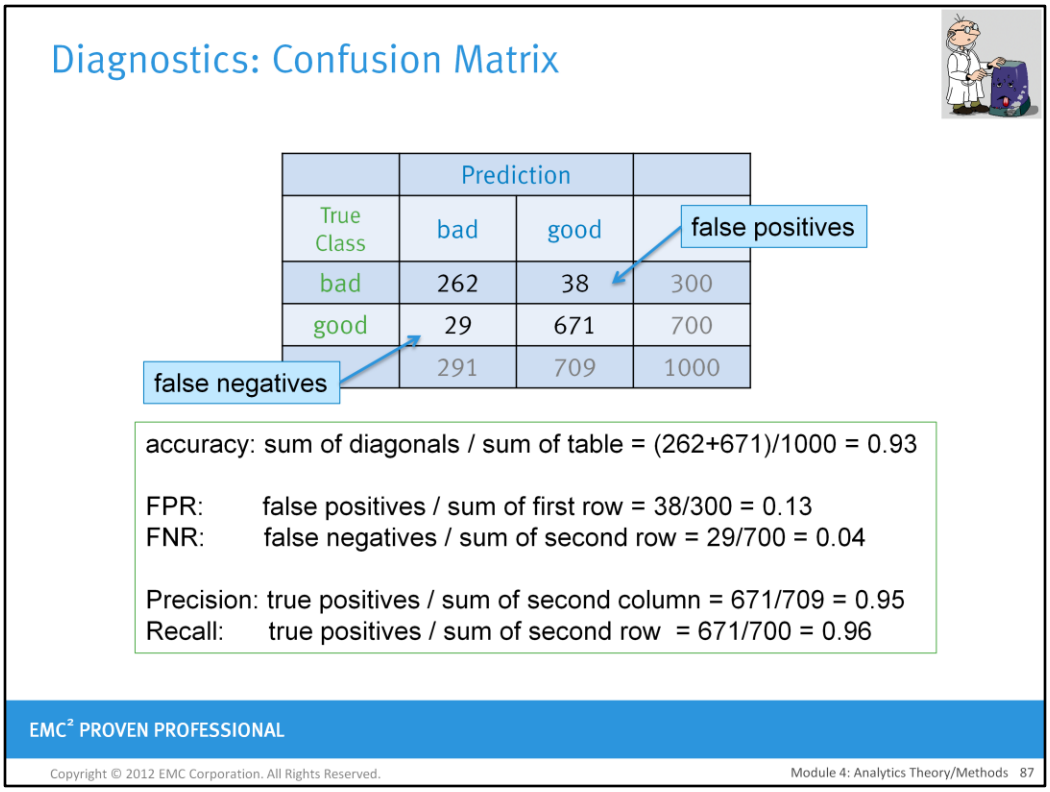
Diagnostics

- Hold-out data
  - How well does the model classify new instances?
- Cross-validation
- ROC curve/AUC



EMC² PROVEN PROFESSIONAL

The diagnostics we used in regression can be used to validate the effectiveness of the model we built. The technique of using the hold-out data and performing N-fold cross validations and using the ROC/Area Under the Curve methods can be deployed with Naïve Bayesian classifier as well.



A **confusion matrix** is a specific table layout that allows visualization of the performance of the model. In the hypothetical example of confusion matrix shown:

Of 1000 credit score samples, the system predicted that there were good and bad credit, and of the 700 good credits, the model predicted 29 as bad and similarly 38 of the actual bad credits were predicted as good. All correct guesses are located in the diagonal of the table, so it's easy to visually inspect the table for errors, as they will be represented by any non-zero values outside the diagonal.

We define accuracy as a metric defining – what we got right - which is the ratio between the sum of the diagonals vs. the sum of the table.

We saw a false positive rate and a false negative rate when we discussed ROC curves

FPR – what percent of negatives we marked positive.

FNR– what percent of positives we marked negative.

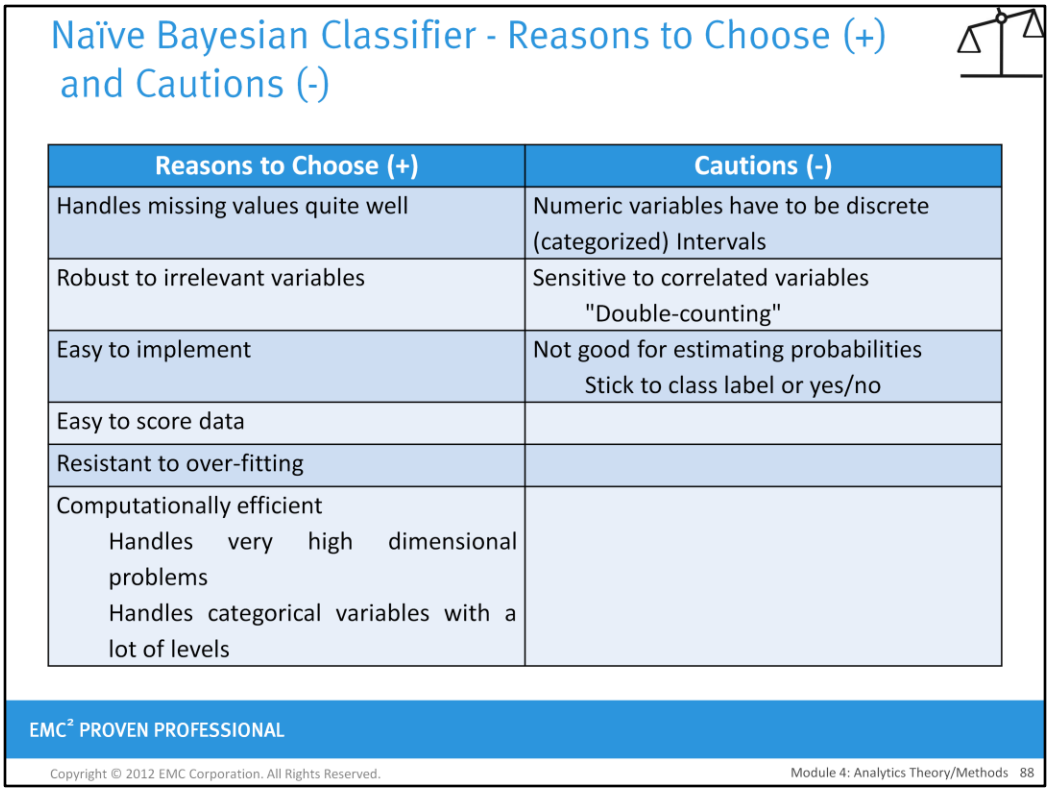
The computation of FPR and FNR are shown in the slide.

Precision and Recall are accuracy metrics used by the information retrieval community; they are often used to characterize classifiers as well. We will detail these metrics in lesson 8 of this module.

Note:

precision – what percent of things we marked positive really are positive

recall – what percent of positive instances did we correctly identify



The Reasons to Choose (+) and Cautions (-) of the Naïve Bayesian classifier are listed. Unlike Logistic regression, missing values are handled well by the Naïve Bayesian classifier. It is also very robust to irrelevant variables (irrelevant variables are distributed among all the classes and their effects are not pronounced).

The model is easy to implement and we will see how easily a basic version can be implemented in the lab without using any packages. Scoring data (predicting) is very simple and the model is resistant to over fitting. (Over fitting refers to fitting training data so well that we fit the idiosyncrasies such as the data that are not relevant in characterizing the data). It is computationally efficient and handles high dimensional problems efficiently. Unlike logistic regression Naïve Bayesian classifier handles categorical variables with a lot of levels.

The Cautions (-) are that it is sensitive to correlated variables as the algorithm double counts the effect of the correlated variables. For example people with low income tend to default and people with low credit tend to default. It is also true that people with low income tend to have low credit. If we try to score “default” with both low income and low credit as variables we will see the double counting effect in our model output and in the scoring.

Though the probabilities are provided as an output of the scored data, Naïve Bayesian classifier is not very reliable for the probability estimation and should be used for class label assignments only. Naïve Bayesian classifier in its simple form is used only with categorical variables and any continuous variables should be rendered discrete into intervals. You will learn more about this in the lab. However it is not necessary to have the continuous variables as “discrete” and several standard implementations can handle continuous variables as well.

Check Your Knowledge

Your Thoughts?

1. Consider the following Training Data Set:
   - Apply the Naïve Bayesian Classifier to this data set and compute

     $P(y = 1 | X)$ for $X = (1,0,0)$

     Show your work

Training Data Set

X1	X2	X3	Y
1	1	1	0
1	1	0	0
0	0	0	0
0	1	0	1
1	0	1	1
0	1	1	1

2. List some prominent Use Cases of the Naïve Bayesian Classifier.
3. What gives the Naïve Bayesian Classifier the advantage of being computationally inexpensive?
4. Why should we use log-likelihoods rather than pure probability values in the Naïve Bayesian Classifier?

EMC² PROVEN PROFESSIONAL

Record your answers here. More Check Your Knowledge questions are on the next page.

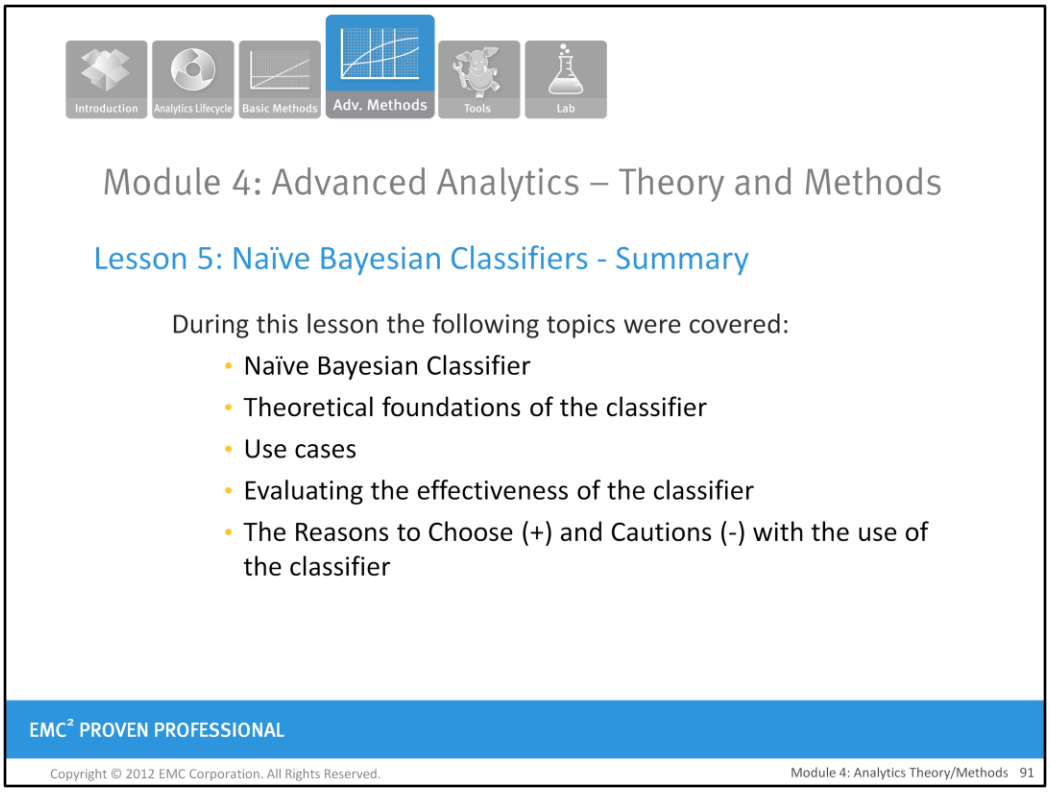
Check Your Knowledge (Continued)

5. What is a confusion matrix and how it is used to evaluate the effectiveness of the model?
6. Consider the following data set with two input features temperature and season
   - What is the Naïve Bayesian assumption?
   - Is the Naïve Bayesian assumption satisfied for this problem?

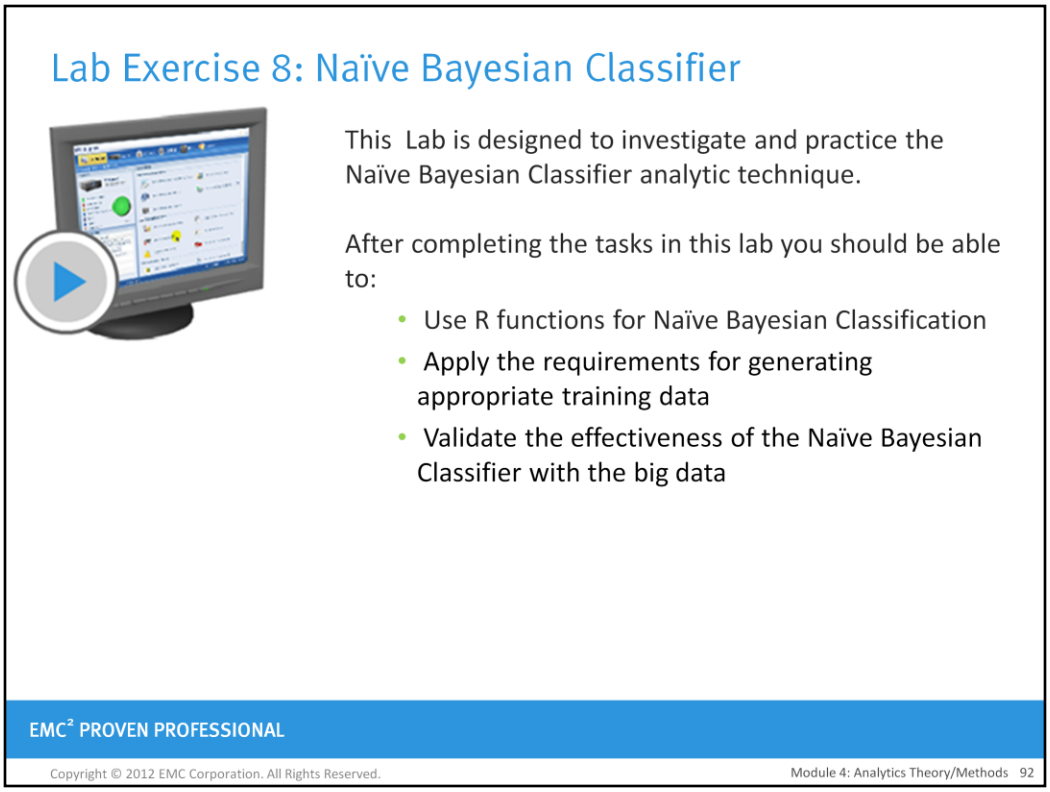
Temperature	Season	Electricity Usage (Class)
Below Average	Winter	High
Above Average	Winter	Low
Below Average	Summer	Low
Above Average	Summer	High

EMC² PROVEN PROFESSIONAL

Record your answers here.



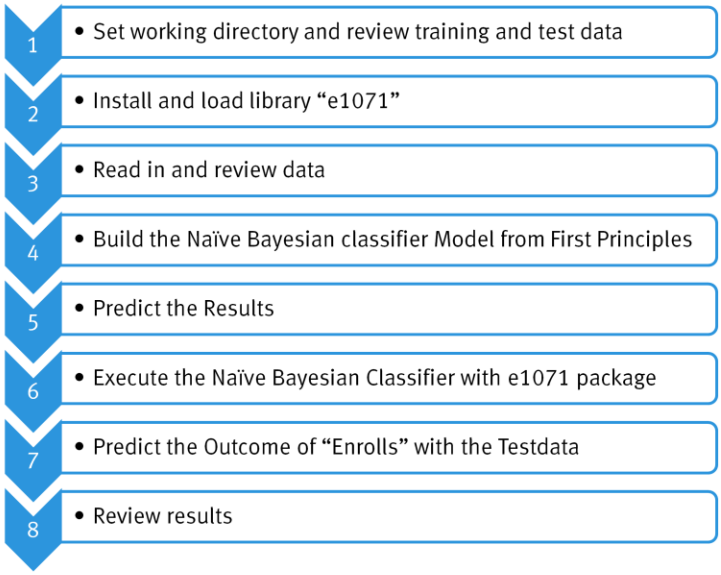
This lesson covered these topics. Please take a moment to review them.



Tasks you will be completing in this lab include:

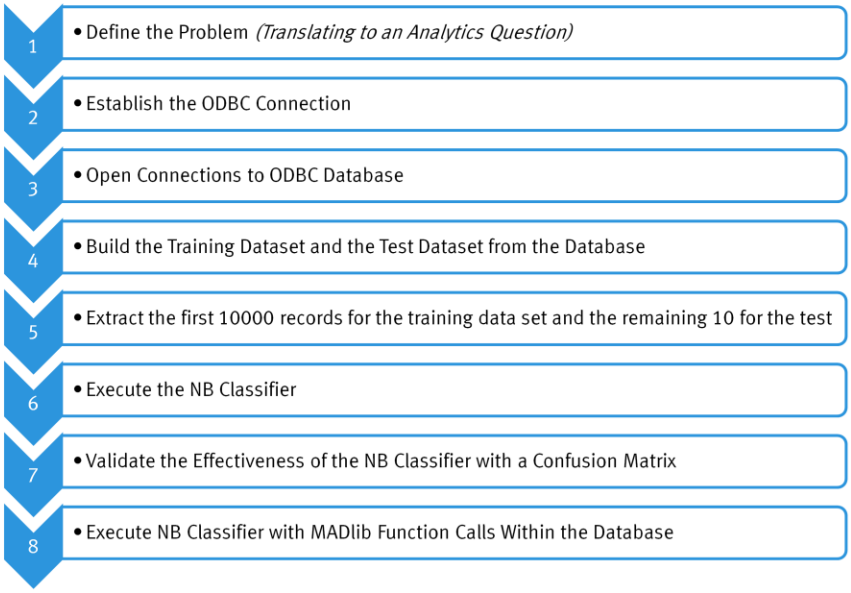
- Use RStudio environment to code Naïve Bayesian Classifier
- Use the ODBC connection to the "census" database to create a training data set for Naïve Bayesian Classifier from the big data
- Use the Naïve Bayesian Classifier program and evaluate how well it predicts the results using the training data and then compare the results with original data.
- Use the MADlib function for NB classifier (in-database analytics)

Lab Exercise 8: Naïve Bayesian Classifier Part1 - Workflow

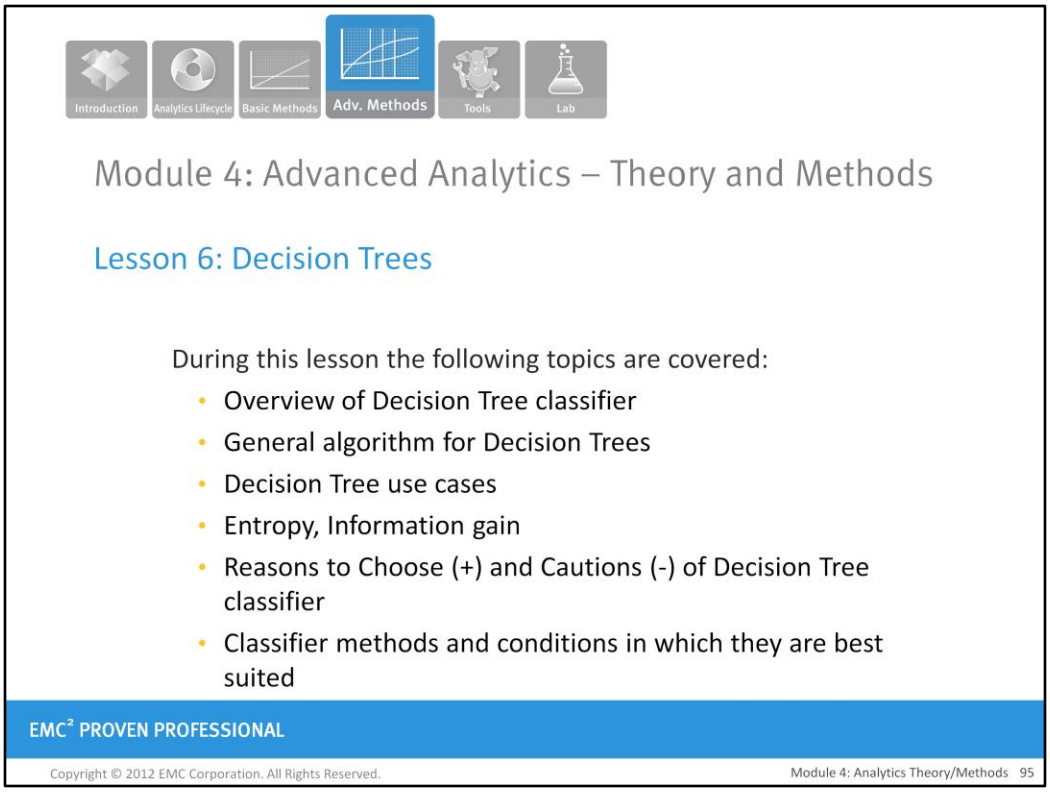


EMC² PROVEN PROFESSIONAL

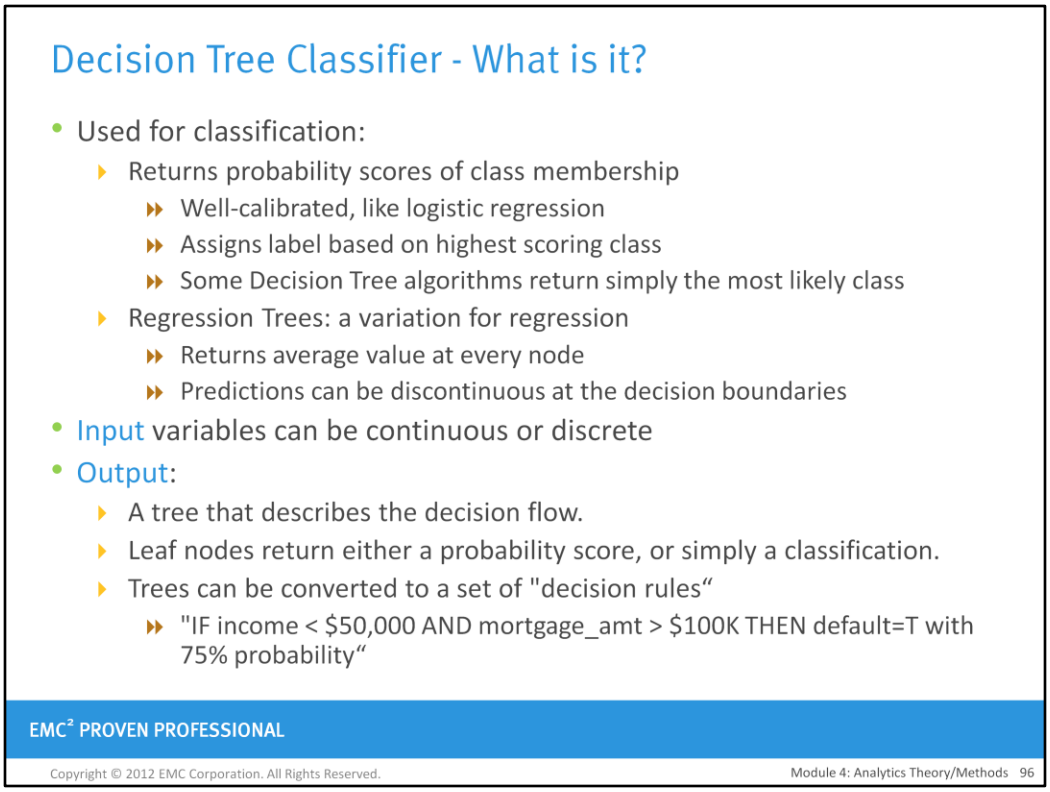
Lab Exercise 8: Naïve Bayesian Classifier Part2 - Workflow



EMC² PROVEN PROFESSIONAL



The topics covered in this lesson are listed.



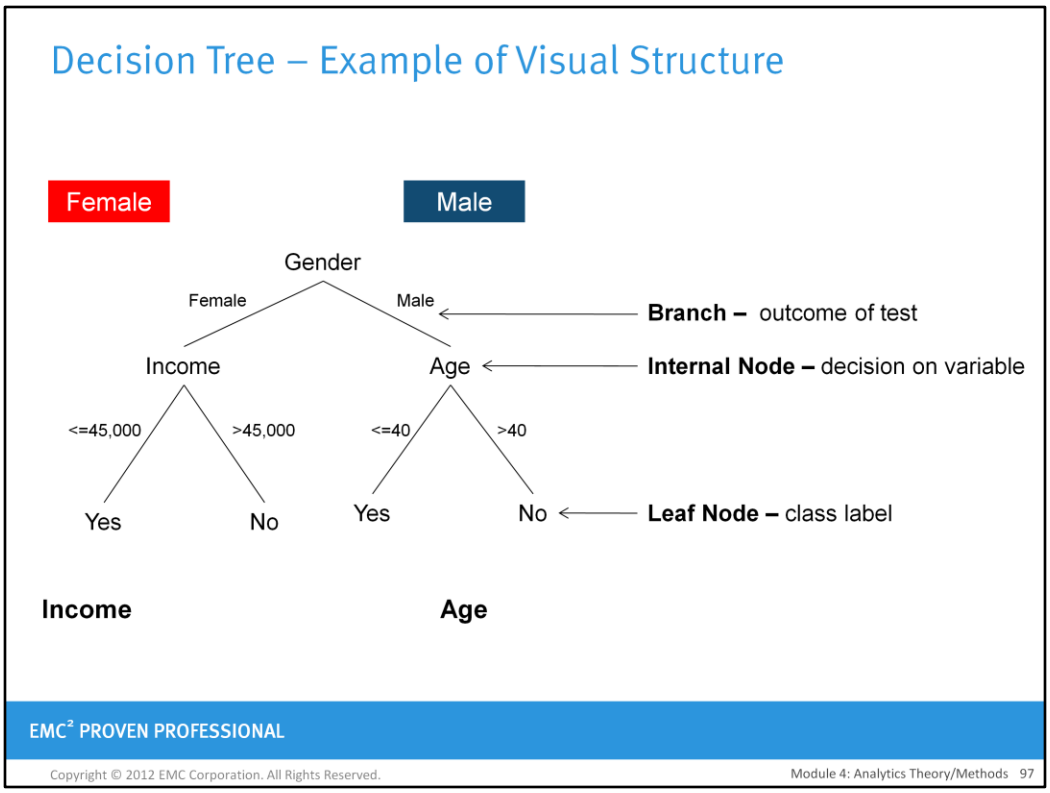
Decision Trees are a flexible method very commonly deployed in data mining applications. In this lesson we will focus on Decision Trees used for classification problems.

There are two types of trees; Classification Trees and Regression (or Prediction) Trees

- Classification Trees – are used to segment observations into more homogenous groups (assign class labels). They usually apply to outcomes that are binary or categorical in nature.
- Regression Trees – are variations of regression and what is returned in each node is the average value at each node (type of a step function with which the average value can be computed). Regression trees can be applied to outcomes that are continuous (like account spend or personal income).

The input values can be continuous or discrete. Decision Tree models output a tree that describes the decision flow. The leaf nodes return class labels and in some implementations they also return the probability scores. In theory the tree can be converted into decision rules such as the example shown in the slide.

Decision Trees are a popular method because they can be applied to a variety of situations. The rules of classification are very straight forward and the results can easily be presented visually. Additionally, because the end result is a series of logical “if-then” statements, there is no underlying assumption of a linear (or non-linear) relationship between the predictor variables and the dependent variable.



Decision Trees are typically depicted in a flow-chart like manner.

Branches refer to the outcome of a decision and are represented by the connecting lines here.

When the decision is numerical, the "greater than" branch is usually shown on the right and "less than" on the left.

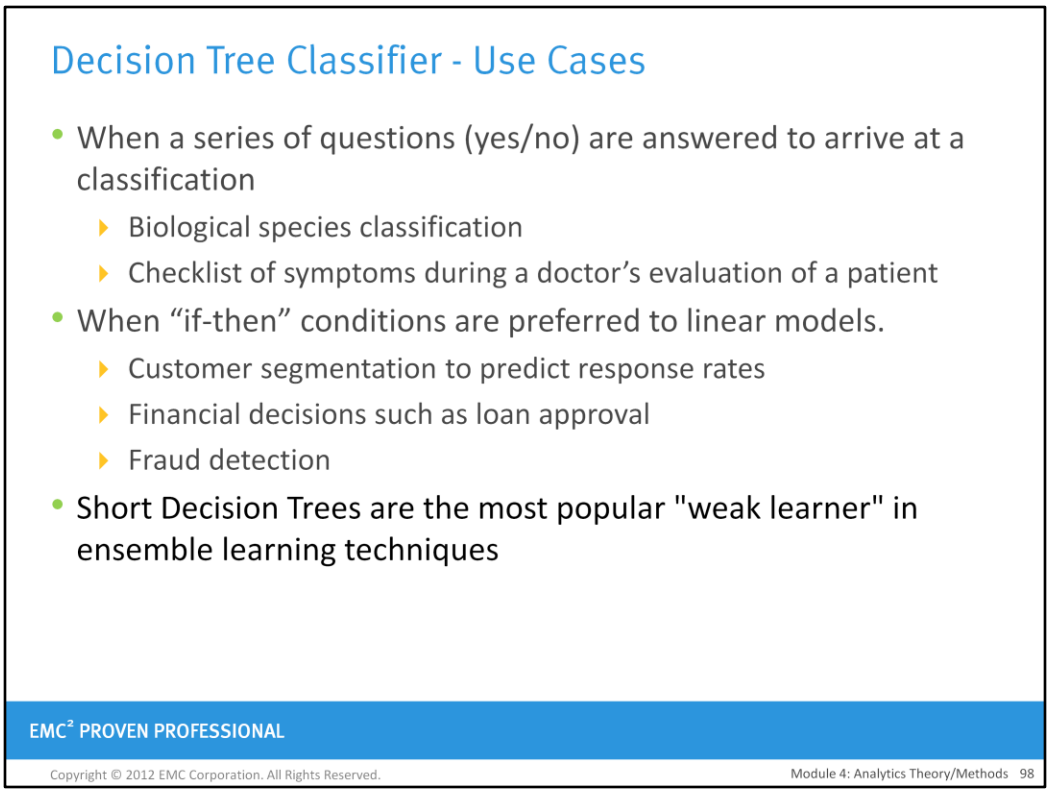
Depending on the nature of the variable, you may need to include an "equal to" component on one branch.

Internal Nodes are the decision or test points. Each refers to a single variable or attribute.

In the example here the outcomes are binary, although there could be more than 2 branches stemming from an internal node.

For example, if the variable was categorical and had 3 choices, you might need a branch for each choice.

The **Leaf Nodes** are at the end of the last branch on the tree. These represent the outcome of all the prior decisions. The leaf nodes are the class labels, or the segment in which all observations that follow the path to the leaf would be placed.

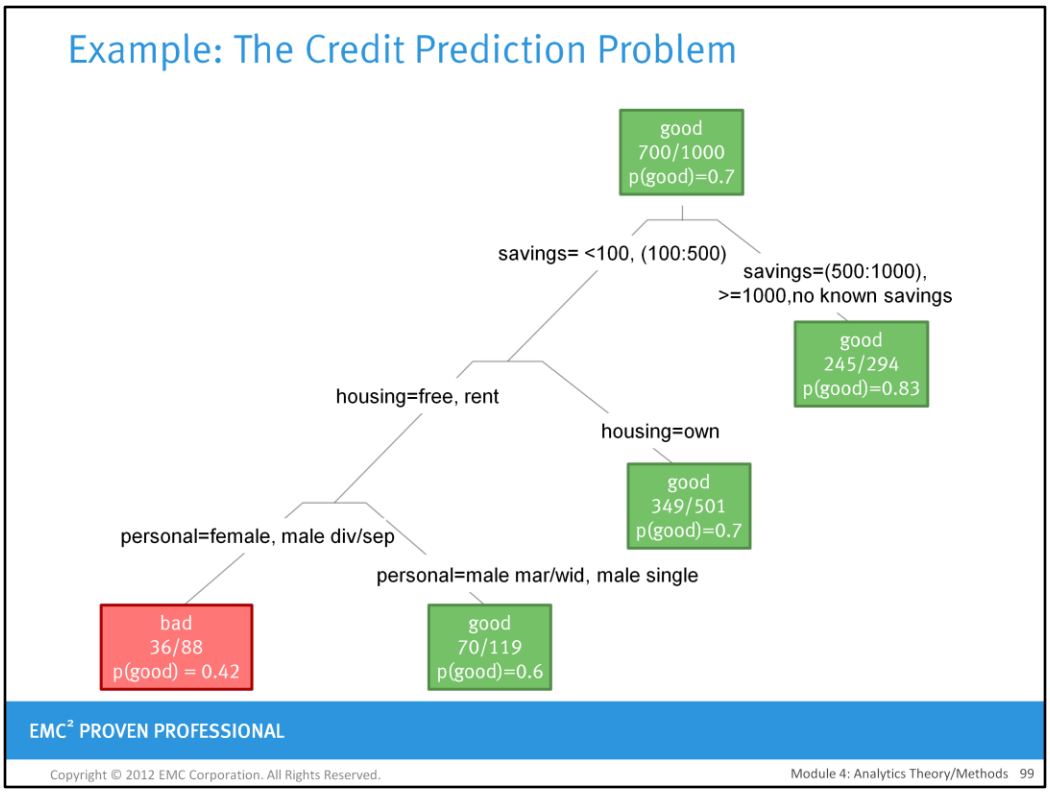


An example of Decision Trees in practice is the method for classifying biological species. A series of questions (yes/no) are answered to arrive at a classification.

Another example is a checklist of symptoms during a doctor's evaluation of a patient. People mentally perform these types of analysis frequently when assessing a situation.

Other use cases can be customer segmentation to better predict response rates to marketing and promotions. Computers can be "taught" to evaluate a series of criteria and automatically approve or deny an application for a loan. In the case of loan approval, computers can use the logical "if-then" statements to predict whether the customer will default on the loan. For customers with a clear (strong) outcome, no human interaction is required, for observations which may not generate a clear response, a human is needed for the decision.

Short Decision Trees (where we have limited the number of splits) are often used as components (called "weak learners" or "base learners") in ensemble techniques (a set of predictive models which will all vote and we take decisions based on the combination of the votes) such as Random forests, bagging and boosting (Beyond the scope for this class). The very simplest of the short trees are decision stumps: Decision Trees with one internal node (the root) which is immediately connected to the terminal nodes. A decision stump makes a prediction based on the value of just a single input feature.



We will use the same example we used in the previous lesson with Naïve Bayesian classifier.

For the people with good credit and we start at the top of the tree the probability is 70% (700 out of 1000 people have good credit). The process has decided that we are going to split how much is in the savings account into two groups.

One group with savings less than $100 or between $100 to $ 500.

The second group is the rest of the population which has savings of $500 to $1000 or greater than $1000 or no known savings.

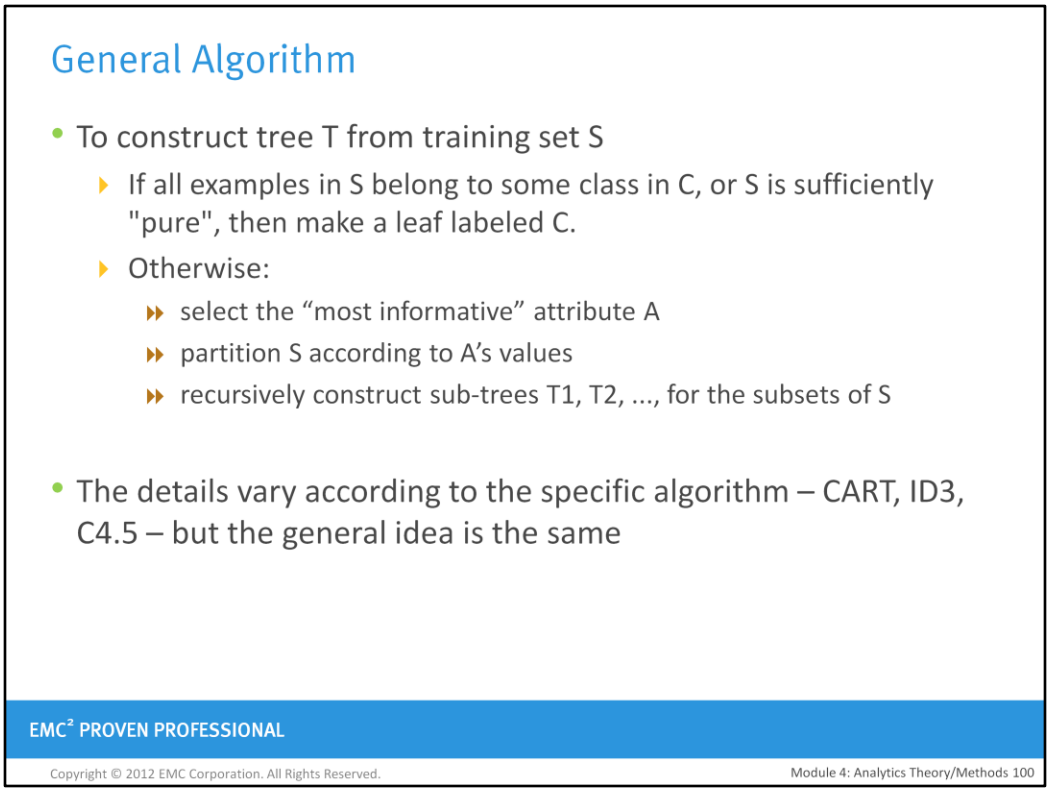
We compute the probability of good credit at the second node and we find in the second savings category 245 out of 294 have good credit and the probability at this node is 83%.

Looking at the other node (Savings <100 or Savings 100:500) we look into housing. We split this node into Housing (free,rent) as one group and Housing (own) as the other. Computing probability of good credit at housing (own) node we see that 349 out of 501 people have good credit, a 70% probability.

Traversing down the housing (free, rent) node we split now on the variable known as personal. The two groups are Personal (female, male divorced/ separated) and Personal (male,married/widowed,male_single). In the node on the right, the probability of good credit is 0.6; in the node on the left, the probability of good credit is 44% (which is less than 50%, so we label the node as a "bad credit" node).

We can see that for this case, we might want to work with the probabilities, rather than the class labels; this tree would only label 88 rows (out of 1000) of the training set as "bad", which is far less than the 30% "bad" rate of the training set, and of those cases labeled "bad", only 54% of them would truly be bad. Tuning the splitting parameters, or using a random forest or other ensemble technique (more on that later) might improve the performance.

Decision Trees are greedy algorithms. They take decisions based on what is available at that moment and once a bad decision is taken it is propagated all the way down. An ensemble technique may randomize the splitting (or even randomize data) and come up with multiple tree structures. It then assigns labels by looking at the average of the nodes in all the trees and assigns class labels or probability values.

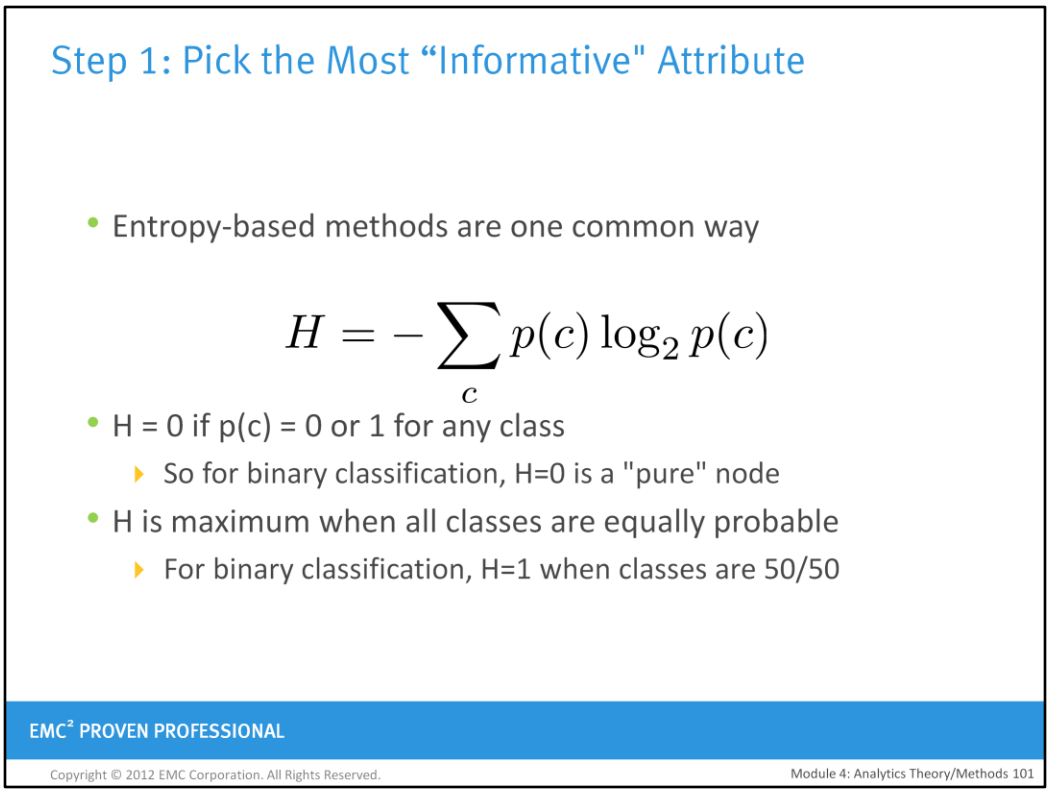


We now describe the general algorithm. Our objective is to construct a tree T from a training set S. If all examples in S belongs to some class “C “ (good_credit for example) or S is sufficiently “pure” (in our case node p(credit_good) is 70% pure) we make a leaf labeled “C”.

Otherwise we will select another attribute considered as the “most informative” (savings, housing etc.) and partition S according to A’s values. Something similar to what we explained in the previous slide. We will construct sub-trees T1,T2….. or the subsets of S recursively until

- You have all of the nodes as pure as required or
- You cannot split further as per your specifications or
- Any other stopping criteria specified.

There are several algorithms that implement Decision Trees and the methods of tree construction vary with each one of them. CART,ID3 and C4.5 are some of the popular algorithms.



The first step is to pick the most informative attribute. There are many ways to do it. We detail Entropy based methods.

Let p(c) be the probability of a given class. H as defined by the formula shown above will have a value 0 if p (c) is 0 or 1. So for binary classification H=0 means it is a “pure” node. H is maximum when all classes are equally probable. If the probability of classes are 50/50 then H=1 (maximum entropy).

Step 1: Pick the most "informative" attribute (Continued)

- First, we need to get the base entropy of the data

$$H_{credit} = -(0.7\log_2(0.7) + 0.3\log_2(0.3)) = 0.88$$

EMC² PROVEN PROFESSIONAL

In our credit problem p(credit_good) is 0.7 and p(credit_bad) is 0.3.

The base entropy H_{credit} = -(0.7 $\log_2(0.7)$ + 0.3$\log_2(0.3)$) = 0.88 (very close to 1)

Our unconditioned credit problem has fairly high entropy.

Step 1: Pick the Most "Informative" Attribute (Continued)
Conditional Entropy

$$H_{attr} = -\sum_{v} p(v) \sum_{c} p(c|v) \log_2 p(c|v)$$

- The weighted sum of the class entropies for each value of the attribute
- In English: attribute values (home owner vs. renter) give more information about class membership
  - "Home owners are more likely to have good credit than renters"
- Conditional entropy should be lower than unconditioned entropy

EMC² PROVEN PROFESSIONAL

Continuing with step 1 we now find the conditional entropy, which is the weighted sum of class entropies for each value of the attribute.

Let us say we choose the attribute "Housing" we have three levels for this attribute (free, rent and own). Intuitively we can say that home owners are more likely to have better credit than renters. So the attribute value Housing will give more information about the class membership for credit_good. The conditional entropy of attribute Housing should be lower than the base entropy.

At worst (in the case where the attribute is uncorrelated with the class label), the conditional entropy is the same as the unconditioned entropy.

Conditional Entropy Example

	for free	own	rent
P(housing)	0.108	0.713	0.179
P(bad | housing)	0.407	0.261	0.391
p(good | housing)	0.592	0.739	0.601

$$
\begin{aligned}
H_{(housing|credit)} = &-[0.108 * (0.407 \log_2(0.407) + 0.592 \log_2(0.592)) \\
&+ 0.713 * (0.261 \log_2(0.261) + 0.739 \log_2(0.739)) \\
&+ 0.179 * (0.391 \log_2(0.391) + 0.601 \log_2(0.601))] \\
= &\ 0.868
\end{aligned}
$$

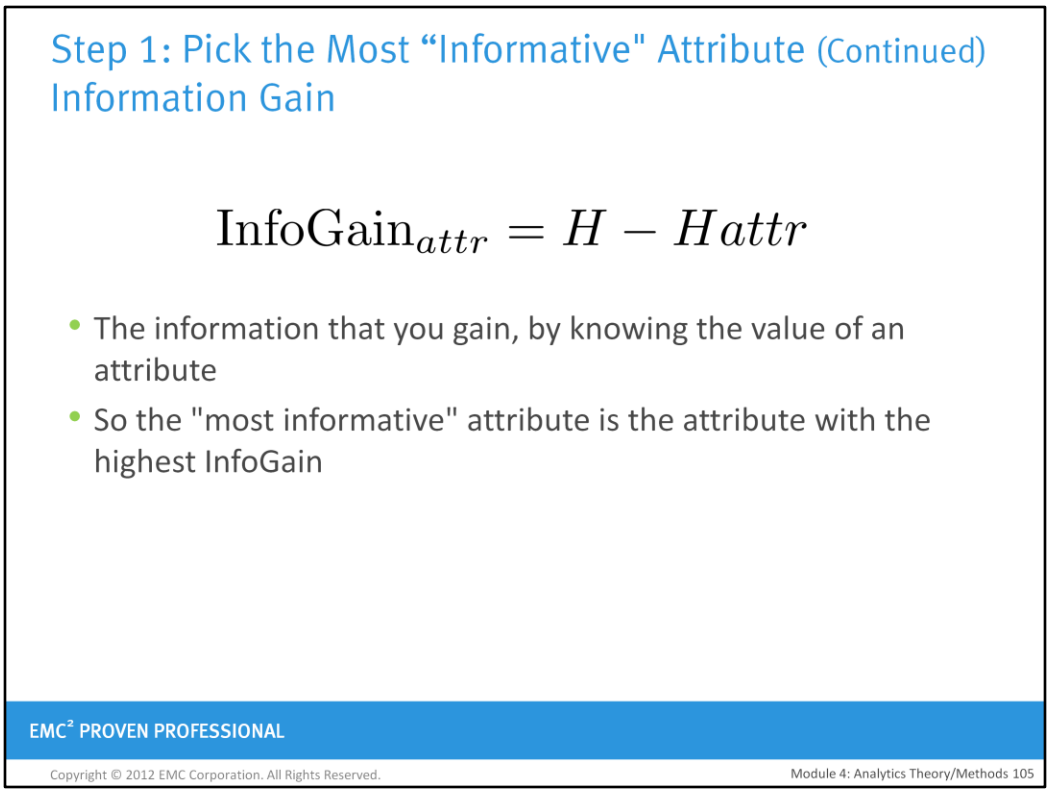
EMC² PROVEN PROFESSIONAL

Let's compute the conditional entropy of credit class conditioned on housing status.

In the top row of the table are the probabilities of each value. In the next two rows are the probabilities of the class labels conditioned on the housing value.

Note that each term inside parentheses is the entropy of the class labels within a single housing value.

The conditional entropy is still fairly high; but it is a little less than the unconditioned entropy.



Information Gain is defined as the difference between the base entropy and the conditional entropy of the attribute.

So the most informative attribute is the attribute with most information gain. Remember, this is just an example. There are other information/purity measures, but InfoGain is a fairly popular one for inducing Decision Trees.

Back to the Credit Prediction Example

$$\begin{aligned}\text{InfoGain}_{credit} &= H_{credit} - H_{housing|credit} \\ &= 0.88 - 0.86 \\ &\approx 0.013\end{aligned}$$

Attribute	InfoGain
job	0.001
housing	0.013
personal_status	0.006
savings_status	*0.028*

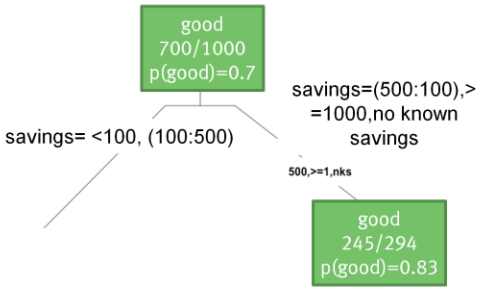
EMC² PROVEN PROFESSIONAL

If we compute the InfoGain for all of our input variables, we see that savings_status is the most informative variable.

We can see that savings_status gives the most infoGain and that is why it was the first variable on which the tree was split.

Step 2 & 3: Partition on the Selected Variable

- Step 2: Find the partition with the highest InfoGain
  - In our example the selected partition has InfoGain = 0.028
- Step 3: At each resulting node, repeat Steps 1 and 2
  - until node is "pure enough"
- Pure nodes => no information gain by splitting on other attributes



EMC² PROVEN PROFESSIONAL

The selected partitioning has InfoGain almost as high as using each savings value as a separate node. And InfoGain happens to be biased to many partitions, so this partition is basically as informative.

InfoGain can be used with continuous variables as well; in that case, finding the partition and computing the information gain are the same step.

"Pure enough" usually means that no more information can be gained by splitting on other attributes

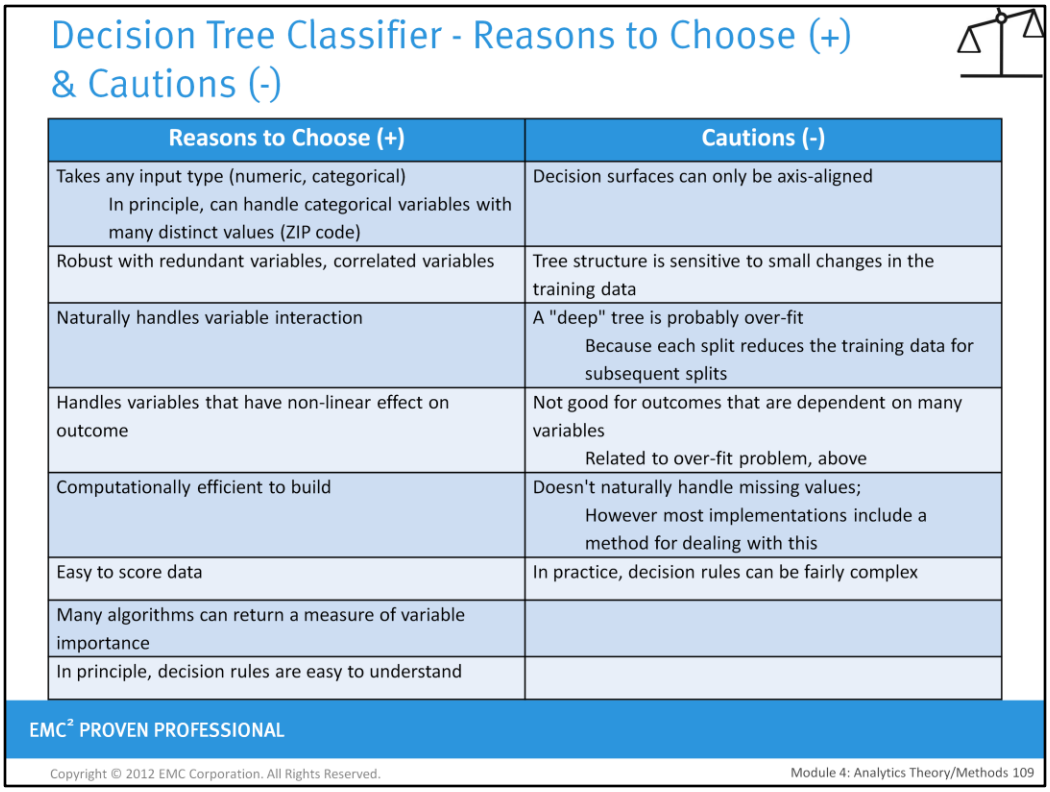
Diagnostics

- Hold-out data
- ROC/AUC
- Confusion Matrix
- FPR/FNR, Precision/Recall
- Do the splits (or the "rules") make sense?
  - What does the domain expert say?
- How deep is the tree?
  - Too many layers are prone to over-fit
- Do you get nodes with very few members?
  - Over-fit

EMC² PROVEN PROFESSIONAL

The diagnostics are exactly the same as the one we detailed for Naïve Bayesian classifier. We use the hold-out data /AUC and confusion matrix. There are sanity checks that can be performed such as validating the "decision rules" with domain experts and determining if they make sense.

Having too many layers and obtaining nodes with very few members are signs of over fitting.



Decision Trees take both numerical and categorical variables. They can handle many distinct values such as the zip code in the data.

Unlike Naïve Bayesian the Decision Tree method is robust with redundant or correlated variables. Decision Trees handles variables that are non-linear. Linear/logistic regression computes the value as b1*x1 + b2*x2 .. And so on.

If two variables interact and say the value y depends on x1*x2, linear regression does not model this type of data correctly.

Naïve Bayes also does not do variable interactions (by design). Decision Trees handle variable interactions naturally. Every node in the tree is in some sense an interaction.

Decision Tree algorithms are computationally efficient and it is easy to score the data. The outputs are easy to understand. Many algorithms return a measure of variable importance. Basically the information gain from each variable is provided by many packages.

In terms of Cautions (-), decision surface is axis aligned and the decision regions are rectangular surfaces. However, if the decision surface is not axis aligned (say a triangular surface), the Decision Tree algorithms do not handle this type of data well.

Tree structure is sensitive to small variations in the training data. If you have a large data set and you build a Decision Tree on one subset and another Decision Tree on a different subset the resulting trees can be very different even though they are from the same data set. If you get a deep tree you are probably over fitting as each split reduces the training data for subsequent splits.

<Continued>

Decision Tree Classifier - Reasons to Choose (+) & Cautions (-) (Continued)

Reasons to Choose (+)	Cautions (-)
Takes any input type (numeric, categorical) In principle, can handle categorical variables with many distinct values (ZIP code)	Decision surfaces can only be axis-aligned
Robust with redundant variables, correlated variables	Tree structure is sensitive to small changes in the training data
Naturally handles variable interaction	A "deep" tree is probably over-fit Because each split reduces the training data for subsequent splits
Handles variables that have non-linear effect on outcome	Not good for outcomes that are dependent on many variables Related to over-fit problem, above
Computationally efficient to build	Doesn't naturally handle missing values; However most implementations include a method for dealing with this
Easy to score data	In practice, decision rules can be fairly complex
Many algorithms can return a measure of variable importance	
In principle, decision rules are easy to understand	

EMC² PROVEN PROFESSIONAL

Decision Trees are not good for outcomes that are dependent on many variables. This may contradict the notion that they are robust with redundant variables and correlated variables.

If you have redundant variables, Decision Trees ignore them as the algorithm cannot detect any information gain. If there variables are important and if you split on these variables you will end up with less data with every split.

So if you are dependent on too many variables, Decision Trees may not work well. You will end up with over fit trees. You can compensate for the instability and potential over-fitting of deep trees by combining the decisions of several randomized shallow Decision Trees. Or other "weak learners" – but usually trees use an ensemble model for classification. This has been shown to improve predictive power compared to a single model.

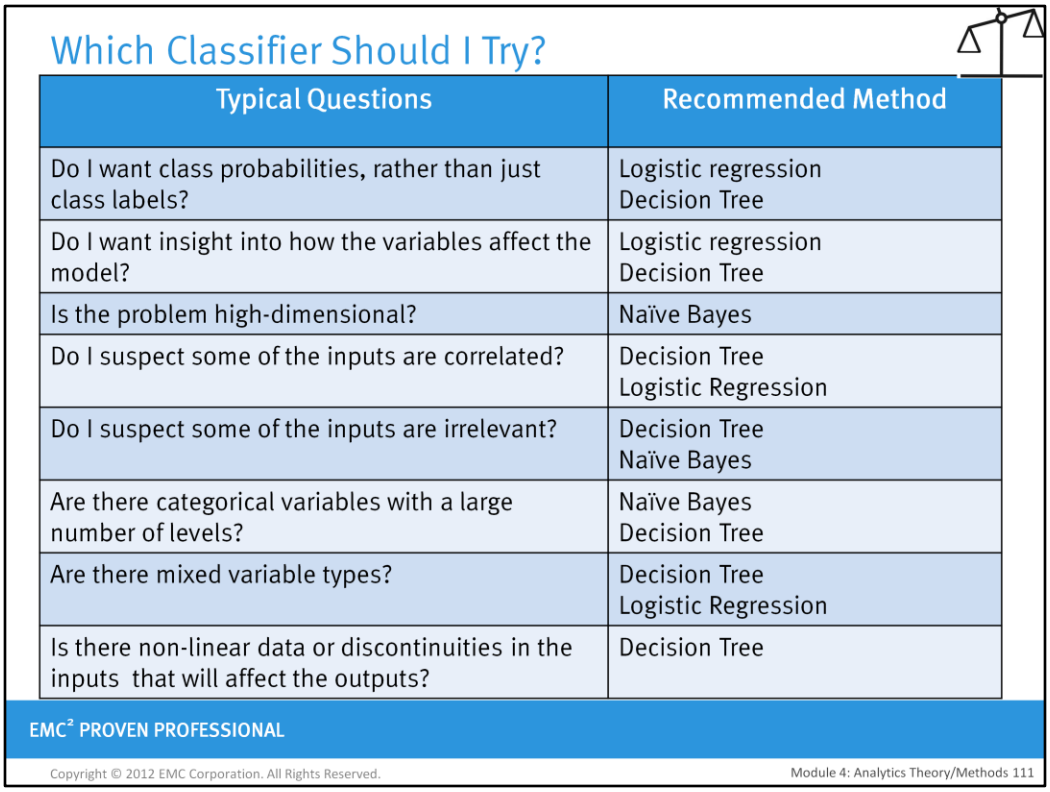
If you are modeling with logistic regression and you have 500 variables and you really do not know which ones to choose. You can use Decision Trees to determine which variables to select based on information gain. Then choose those variables for the logistic regression model. Decision Trees can be used to prune redundant variables.

Decision Trees don't naturally handle missing values though many implementations include a method for dealing with this. Even though we mentioned that decision rules are easy to understand, in practice they can be very complex.

References

Hastie, Tibshirani and Friedman, "The Elements of Statistical Learning"

Seni and Elder, "Ensemble Methods In Data Mining"



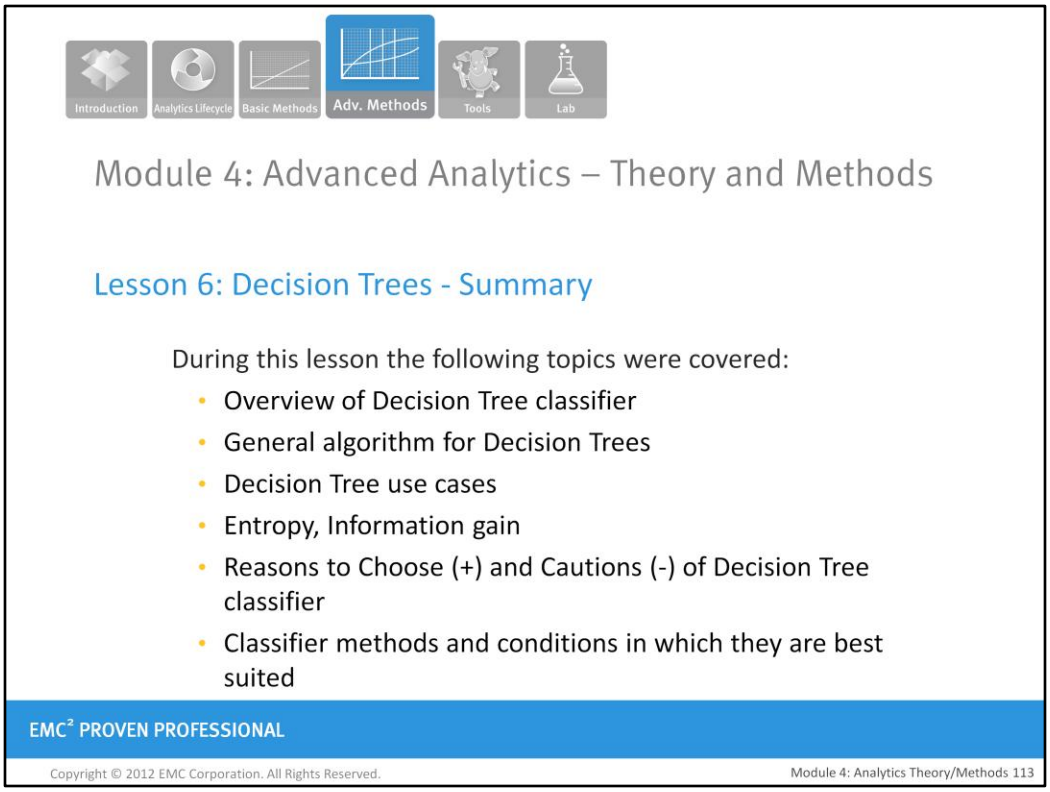
This is only advisory. It's a list of things to think about when picking a classifier, based on the Reasons to Choose (+) and Cautions (-) we've discussed.

Check Your Knowledge

1. How do you define information gain?
2. For what conditions is the value of entropy at a maximum and when is it at a minimum?
3. List three use cases of Decision Trees.
4. What are weak learners and how are they used in ensemble methods?
5. Why do we end up with an over fitted model with deep trees and in data sets when we have outcomes that are dependent on many variables?
6. What classification method would you recommend for the following cases:
   - High dimensional data
   - Data in which outputs are affected by non-linearity and discontinuity in the inputs

EMC² PROVEN PROFESSIONAL

Record your answers here.



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

Lab Exercise 9: Decision Trees

This Lab is designed to investigate and practice Decision Tree (DT) models covered in the course work.

After completing the tasks in this lab you should be able to:

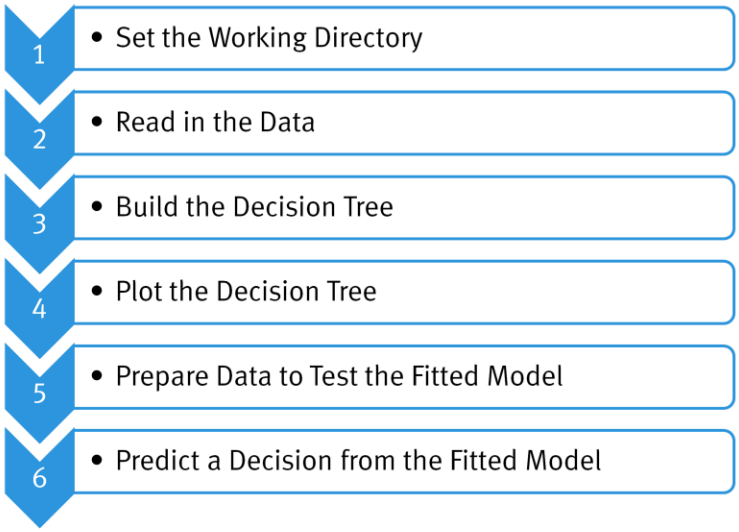
- Use R functions for Decision Tree models
- Predict the outcome of an attribute based on the model

EMC² PROVEN PROFESSIONAL

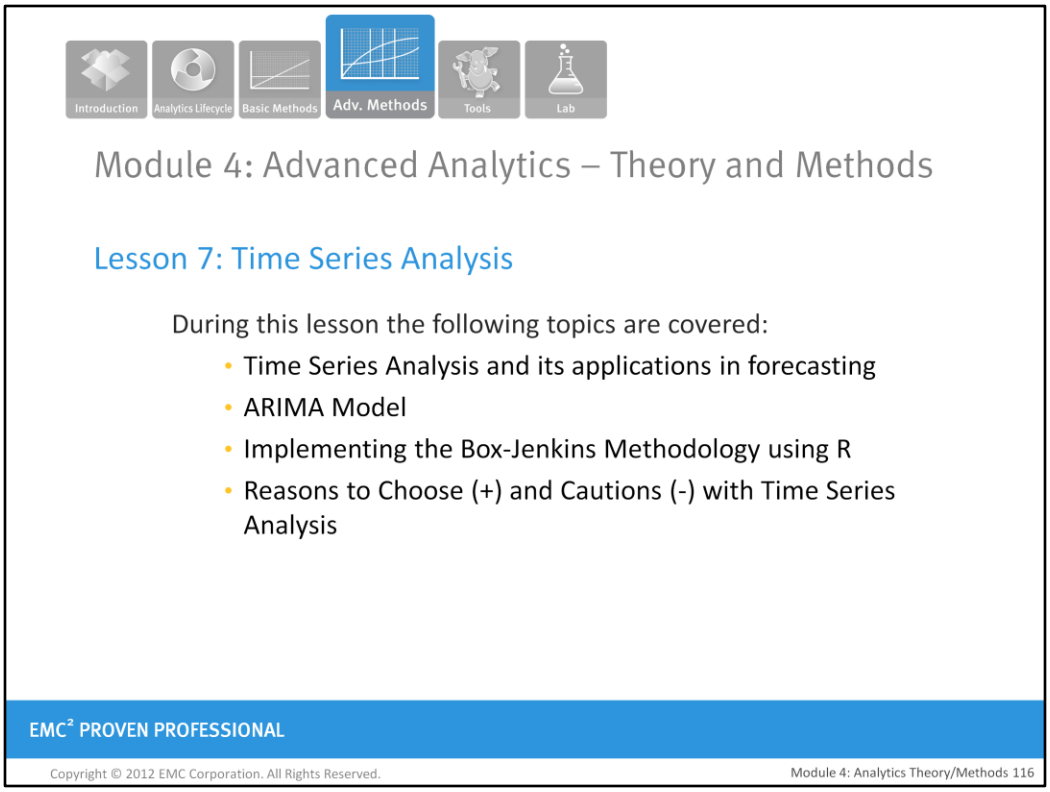
Tasks you will be completing in this lab include:

- Use the RStudio environment to code Decision Tree Models
- Build a Decision Tree Model based on data whose schema is composed of attributes
- Predict the outcome of one attribute based on the model

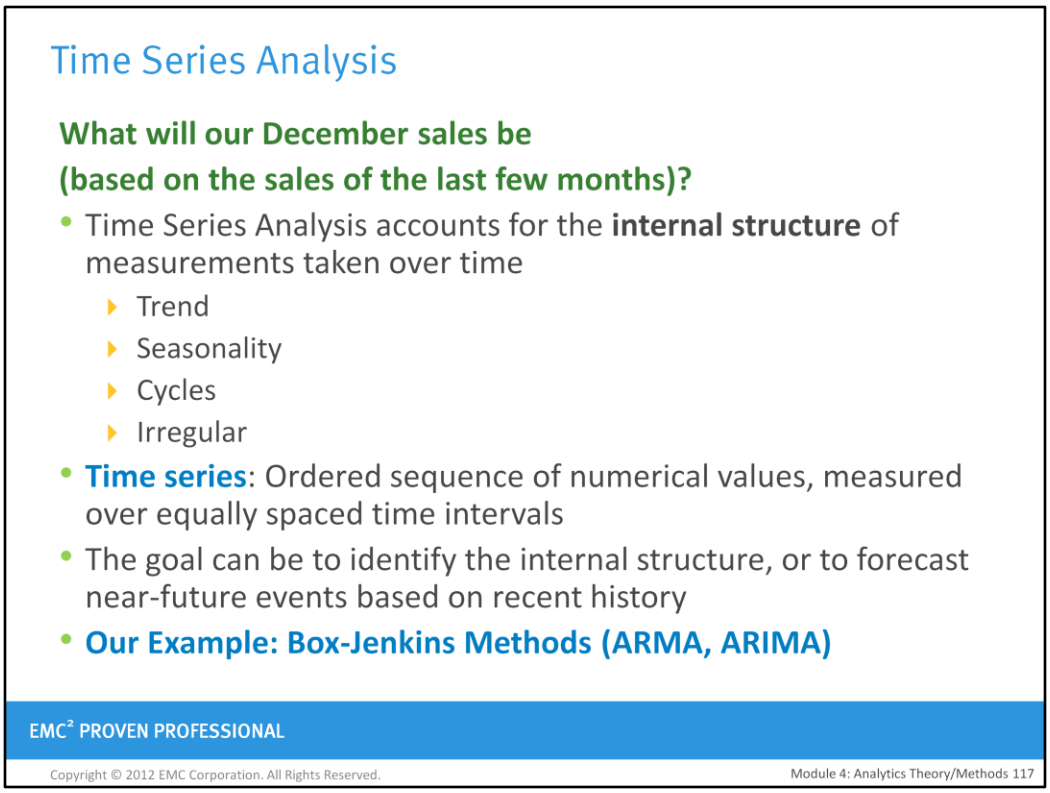
Lab Exercise 9: Decision Trees - Workflow



EMC² PROVEN PROFESSIONAL



The topics covered in this lesson are listed. ARIMA and Box-Jenkins methodology are explained in following slides.

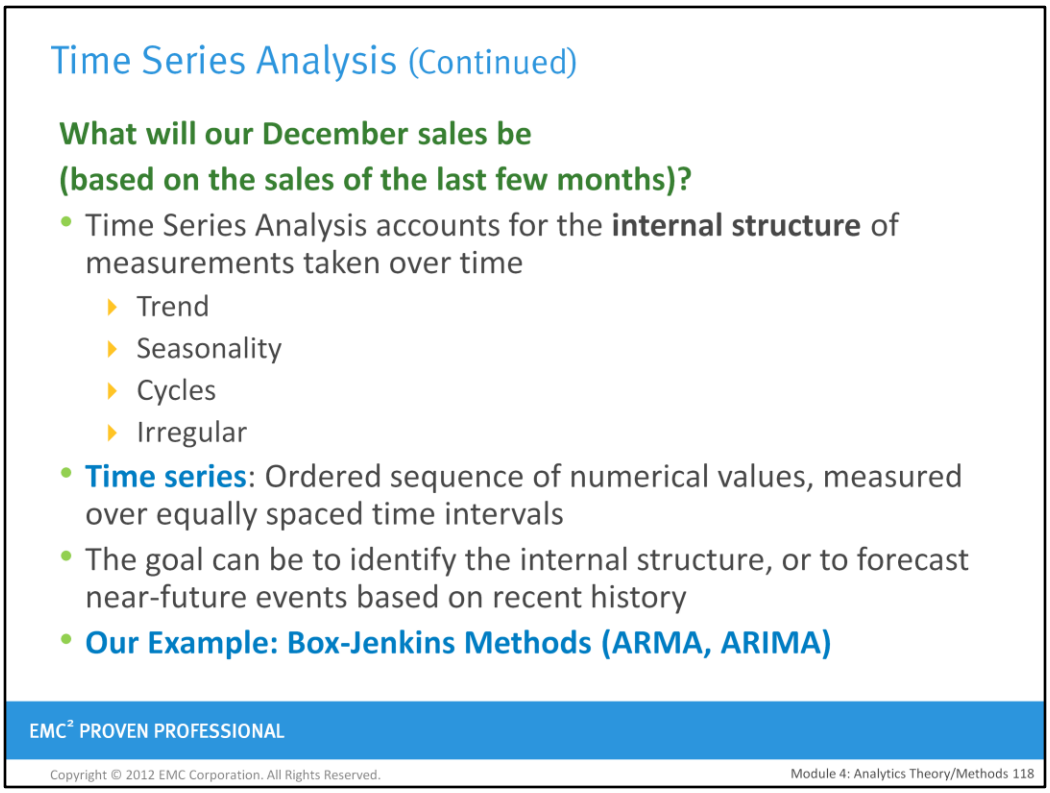


Businesses perform sales forecasting to look well ahead in order to plan their investments, launch new products, decide when to close or withdraw products and so on. The sales forecasting process is a critical one for most businesses. One of the inputs to the sales forecasting process is to look into the past. How well did we do in the last few months or what were our sales in the same time period for he last few years? Time Series Analysis provides a scientific methodology for sales forecasting.**Time Series Analysis** is the analysis of data organized across units of time. Time series is a basic research design in which data for one or more variables are collected for many observations at different time periods. The main objectives in Time Series Analysis are:

- To understand the underlying structure of the time series by breaking it down to its components.
- Fit a mathematical model and then proceed to forecast the future

The time periods are usually regularly spaced and the observations may be either univariate or multivariate. **Univariate** time series are those where only one variable is measured over time, whereas multiple time series are those, where multiple variables are measured simultaneously. The internal structure of the data may specify a trend, seasonality or cycles:

<Continued>



Trend component - Trend is a long term movement in a time series. It is the underlying direction (upward or downward) and rate of change in a time series, when allowance has been made for the other components.

Seasonal component - Seasonal fluctuations of known periodicity. It is the component of variation in a time series which is dependent on the time of the year. It describes any regular fluctuations with a period of less than one year. For example, the costs of various types of fruits and vegetables, and average daily rainfall, all show marked seasonal variation.

Cyclic component - Cyclical variations of non-seasonal nature, whose periodicity is un-known.

Irregular component - Random or chaotic noisy residuals left over when other components of the series (trend, seasonal and cyclical) have been accounted for.

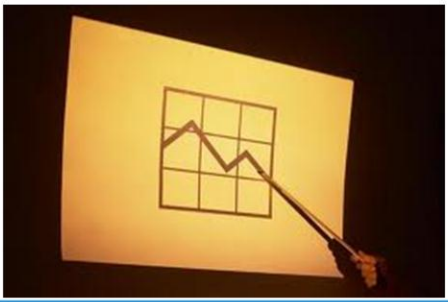
Trend and seasonality, though conceptually distinct, are essentially entangled. The value of the series at time t essentially depends on its value at time t−1, with the result that trend and periodic components are inextricably mixed. Hence, it is not possible to isolate one without trying to isolate the other.

In this lesson we will primarily focus on one Time Series Analysis methodology known as the "Box-Jenkins" method.

Box-Jenkins: What is it?

Used for predicting the next few observations in a time series, based on the last few observations.

- Input: *Trend and Seasonally-adjusted* time series
- Output: Expected future value of the time series
- Applies ARMA (Autoregressive Moving Averages) and ARIMA (Autoregressive Integrated Moving Averages) model



EMC² PROVEN PROFESSIONAL

Box Jenkins methodology developed by Professors G.E.P. Box and G.M. Jenkins, enables the forecasting with time series data with both high accuracy and low computational requirements.

The technique may be applied to quickly determine forecasts that are as uncomplicated in form as the simple smoothing methods, or that involve a number of economic variables. In either case, use of this technique enables efficient utilization of other predictive information contained in the data. It offers assurance of obtaining the highest forecasting accuracy possible in terms of the variables on which the forecast is based.

The input for the model is the trend and seasonality adjusted time series and the output is the expected future value of the time series.

Box Jenkins Methodology applies autoregressive moving average ARMA or ARIMA models to find the best fit of a time series to past values of this time series, in order to make forecasts.

Use Cases

- Forecast next month's sales
  - Based on last few months
- Forecast tomorrow's stock price
  - Based on last few days
- Forecast power demand in the near term
  - Based on last few days

EMC² PROVEN PROFESSIONAL

The key area of application of **Time Series Analysis** is in forecasting.

Economic and business planning, inventory and production control. Control and optimization of industrial processes are some of the key applications in which time series analysis is deployed.

Time Series data provide useful information about the physical, biological, social or economic systems generating the time series, such as:

Economics/ Finance: share prices, profits, imports, exports, stock exchange indices.

Sociology: school enrollments, unemployment, crime rate.

Environment: Amount of pollutants, such as suspended particulate matter (SPM), in the environment.

Meteorology: Rainfall, temperature, wind speed.

Epidemiology: Number of SARS cases over time.

Medicine: Blood pressure measurements over time for evaluating drugs to control hypertension.

Modeling a Time Series

- Let's model the time series as

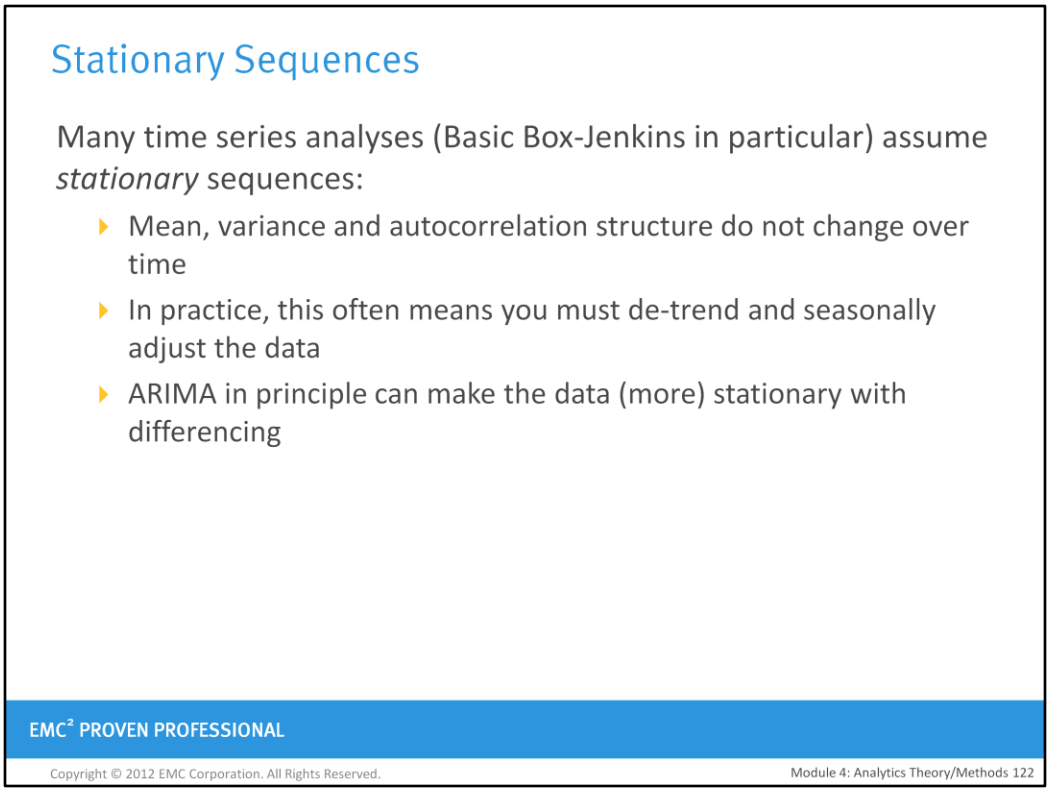
$$Y_t = T_t + S_t + R_t, \quad t=1,\ldots,n.$$

- T_t: Trend term
  - Sales of iPads steadily increased over the last few years: trending upward.
- S_t: The seasonal term (short term periodicity)
  - Retail sales fluctuate in a regular pattern over the course of a year.
    - Typically, sales increase from September through December and decline in January and February.
- R_t: Random fluctuation
  - Noise, or regular high frequency patterns in fluctuation

EMC² PROVEN PROFESSIONAL

We present a simple model for the time series with the trend, seasonality and a random fluctuation. There is often a low frequency cyclic term as well, but we are ignoring that for simplicity.

Examples of trend and seasonality are also detailed in the slide

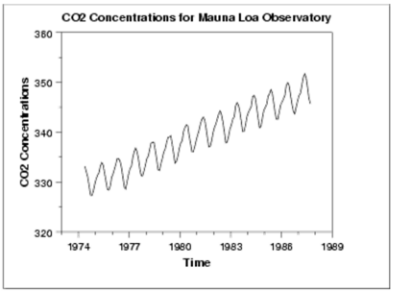


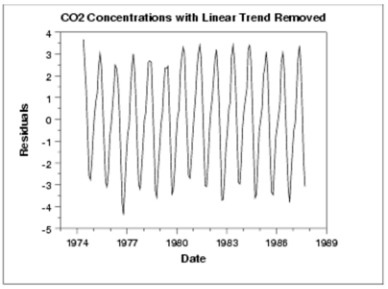
A Stationary sequence is a random sequence in which the joint probability distribution does not vary over time. In other words the mean, variance and auto correlations do not change in the sequence over time.

In order to render a sequence stationary we need to remove the effects of trend and seasonality. The ARIMA model (implemented with Box Jenkins) uses the method of differencing to render the data stationary.

De-trending

- In this example, we see a linear trend, so we fit a linear model
  - $T^*_t = mY_t + b$
- The de-trended series is then
  - $Y^1_t = Y_t - T^*_t$
- In some cases, may have to fit a non-linear model
  - Quadratic
  - Exponential





EMC² PROVEN PROFESSIONAL

Trend in a time series is a slow, gradual change in some property of the series over the whole interval under investigation.

De-trending is often applied to remove a feature thought to distort or obscure the relationships of interest.

In the example shown, the graph of CO2 concentrations measured over many years shows a linear upward trend. In climatology, for example, this CO2 trend due to urban warming might obscure a relationship between air temperature and CO2 concentration.

De-trending is a pre-processing step to prepare time series for analysis by methods that assume stationarity.

A simple linear trend can be removed by subtracting a least-squares-fit straight line. In the example shown we fit a linear model and obtain the difference. The graph shown next is a de-trended time series.

More complicated trends might require different procedures such a fitting a non-linear model such as a quadratic or a exponential model.

Use a ***Linear Trend Model*** if the first differences are more or less constant [$(y_2-y_1) = (y_3-y_2) = \ldots\ldots = (y_n-y_{n-1})$]

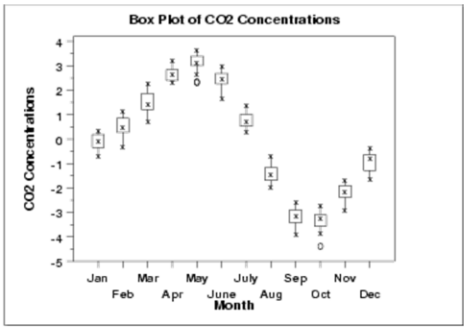
Use a ***Quadratic Trend Model*** if the second differences are more or less constant. [$(y_3-y_2) - (y_2-y_1) = \ldots\ldots\ldots = (y_n-y_{n-1})-(y_{n-1}-y_{n-2})$]

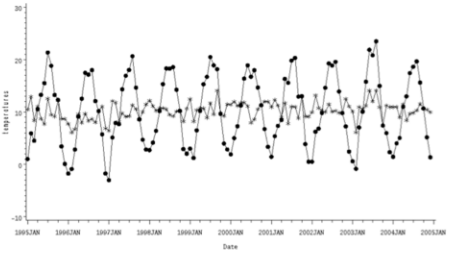
Use an ***Exponential Trend Model*** if the percentage differences are more or constant. [$((y_2-y_1)/y_1) * 100\% = \ldots\ldots((y_n-y_{n-1})/y_{n-1}) * 100\%$

Seasonal Adjustment

- Often, we know the "season"
  - For both retail sales and CO2 concentration, we can model the period as being a year, with variation at the month level
- Simple ad-hoc adjustment: take several years of data, calculate the average value for each month, and subtract that from Y^1_t

$$Y^2_t = Y^1_t - S^*_t$$



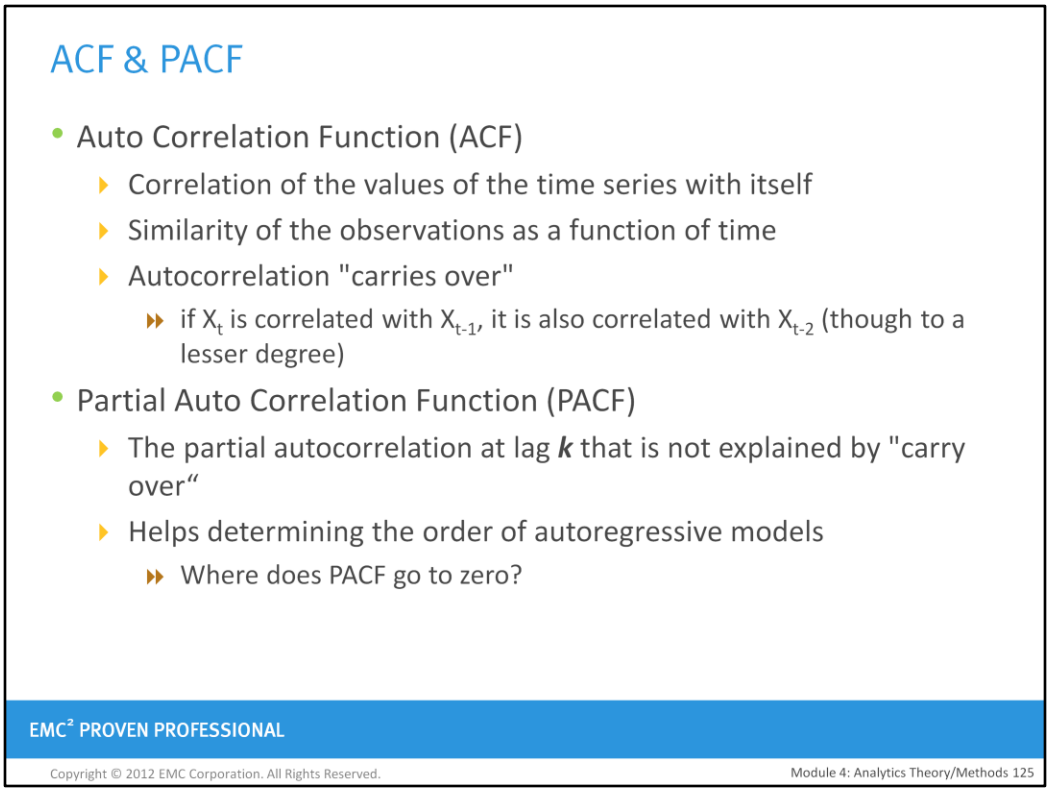


EMC² PROVEN PROFESSIONAL

Unlike the trend and cyclical components, seasonal components, theoretically, happen with similar magnitude during the same time period each year.

The holiday sales spike is an example of seasonality. The seasonal component of a series typically makes the interpretation of a series ambiguous. By removing the seasonal component, it is easier to focus on other components.

A simple adjustment for seasonality is done with taking several years of data, calculating average value for each month and subtracting them from the actual value.



A common assumption in many time series techniques is that the data are stationary. A stationary process has the property that the mean, variance and autocorrelation structure do not change over time.

An ACF plot provides an indication of the stationarity of the data. If the time series is not stationary, we can often transform it to stationarity with the simple technique of differencing.

PACF - The partial autocorrelation at lag ***k*** is the autocorrelation between $\boldsymbol{X_t}$ and $\boldsymbol{X_{t-k}}$ that is not accounted for by lags 1 through ***k*-1**.

One looks for the point on the plot where the partial autocorrelations for all higher lags are essentially zero.

We will look into ACF and PACF graphs in the next Lab.

ARMA Model

$$Y_t = \delta + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}$$

- The simplest Box-Jenkins Model
- Combination of two process models
  - **Autoregressive**: Y_t is a linear combination of its last *p* values
  - **Moving average**: Y_t is a constant value plus the effects of a dampened white noise process over the last *q* time steps

EMC² PROVEN PROFESSIONAL

Autoregressive (AR) models can be coupled with moving average (MA) models to form a general and useful class of time series models called *Autoregressive Moving Average (ARMA)* models. This is the simplest Box-Jenkins model.

AR model predicts y_t as a linear combination of its last p values. An autoregressive model is simply a linear regression of the current value of the series on one or more prior values of the same series. Several options are available for analyzing autoregressive models, including standard linear least squares techniques. They also have a straightforward interpretation.

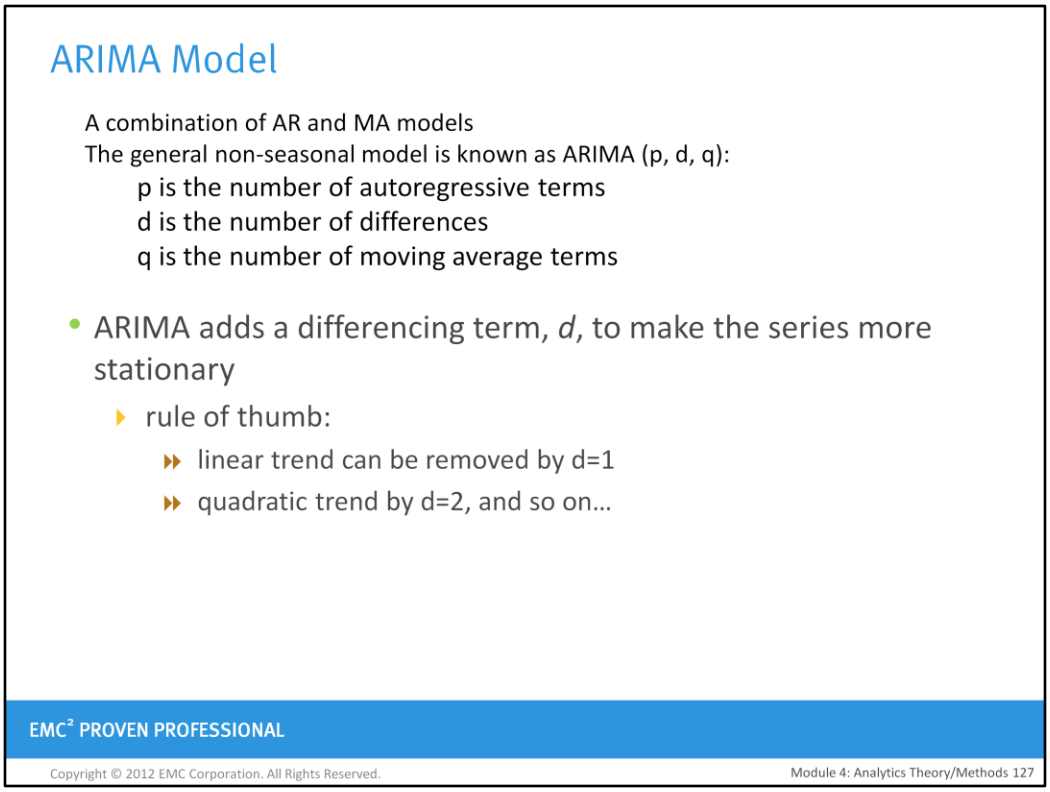
The time series y is called an autoregressive process of order p and is denoted as AR(p) process.

A MA model adds to y_t the effects of a dampened white noise process over the last q steps. This is a simple moving average or single moving average; it's probably the most basic of the forecasting methods.

What one does is to take the data from the last n periods, average the data, and use that as the forecast for the next period. We count backwards in time, minus 1, minus, 2, minus 3 and so forth until we have n data points, divide the sum of those by the number of data points, n, and that gives you the forecast for the next period. So it's called a single moving average or simple moving average. The forecast is simply a constant value that projects the next time period. “n” is also the order of the moving averages.

ARIMA – difference the Y_t d times to "induce stationarity". d is usually 1 or 2. "I" stands for integrated – the outputs of the model are summed up (or "integrated") to recover Y_t

moving average: like a random walk, or brownian motion



ARMA models can be used when the series is **weakly stationary**; in other words, the series has a *constant* variance around a *constant* mean.. This class of models can be extended to non-stationary series by allowing the differencing of the data series. These are called *Autoregressive Integrated Moving Average(ARIMA)* models. There are a large variety of ARIMA models.

ARIMA – difference the y_t d times to "induce stationarity". d is usually 1 or 2. "I" stands for integrated – the outputs of the model are summed up (or "integrated") to recover y_t

The general ARIMA (p, d, q) model gives a tremendous variety of patterns in the ACF and PACF, so it is not practical to state rules for identifying general ARIMA models. In practice, it is seldom necessary to deal with values p, d, or q that are larger than 0, 1, or 2. It is remarkable that such a small range of values for p, d, or q can cover such a large range of practical forecasting situations.

Model Selection

- The Data Scientist must pick *p, d* and *q*
  - An "art form" that requires domain knowledge, modeling experience, and a few iterations
  - A simple AR model (q = 0), or MA model (p=0) might be simpler for the novice

EMC² PROVEN PROFESSIONAL

Identification of the most appropriate model is the most important part of the process, where it becomes as much 'art' as 'science'.

The first step is to determine if the variable is stationary, this can be done with the correlogram. If it is not stationary it needs to be first-differenced. (it may need to be differenced again to induce stationarity)

The next stage is to determine the *p* and *q* in the ARIMA (*p*, *D, q*) model (the *D* refers to how many times the data needs to be differenced to produce a stationary series).

In the diagnostic stage we assess the model's adequacy by checking whether the model assumptions are satisfied. If the model is inadequate, this stage will provide some information for us to re-identify the model. We also perform: checking normality, constant variance, and independence assumption among residuals.

Time Series Analysis - Reasons to Choose (+) & Cautions (-)

Reasons to Choose (+)	Cautions (-)
Minimal data collection Only have to collect the series itself Do not need to input drivers	No meaningful drivers: prediction based only on past performance No explanatory value Can't do "what-if" scenarios Can't stress test
Designed to handle the inherent autocorrelation of lagged time series Compared to simple linear regression Once you've seasonally/trend adjusted	It's an "art form" to select appropriate parameters
	Suitable for short term predictions only

EMC² PROVEN PROFESSIONAL

The Reasons to Choose (+) and Cautions (-) of Time Series Analysis are listed.

Time Series Analysis is not a common “tool” in a Data Scientist’s tool kit. Though the models require minimal data collection and handle the inherent auto correlations of lagged time series, it does not produce meaningful drivers for the prediction.

The selection of (p,d,q) appropriately is not very straight forward. A complete understanding of the domain knowledge and very detailed analysis of trend and seasonality may be required. Further this method is suitable for short term predictions only.

Time Series Analysis with R

- Getting the data and plotting
- The function "*ts*" is used to create time series objects
  - Made into an *R* time series via mydata.data<- *ts*(mydata,start=c(1999,1),frequency=12)
  - Model building – use plot and box plot
- Differencing
  - diff(hstart.data,1,1)
  - acf: It computes (and by default plots) estimates of the autocovariance or autocorrelation function
  - pacf: It is the function used for the partial autocorrelations

EMC² PROVEN PROFESSIONAL

Important R functions and commands we will be using are listed here.

Time Series Analysis with R (Continued)

- ar: Fit an autoregressive time series model to the data
- arima: Fit an ARIMA model to a ***Univariate Time Series***
- predict: Do model predictions
  - *"predict"* is a generic function for predictions from the results of various model fitting functions. The function invokes particular methods which depend on the *class* of the first argument
- arima.sim: Simulate from an ARIMA model
- ARMAtoMA: Convert ARMA process to infinite MA process
- decompose:
  - Decompose a time series into seasonal, trend and irregular components using moving averages
  - Deals with additive or multiplicative seasonal component
  - stl: Decompose a time series into seasonal, trend and irregular components using loess

EMC² PROVEN PROFESSIONAL

Some additional commands in the *ts* package are listed.

We will use these commands in the lab.

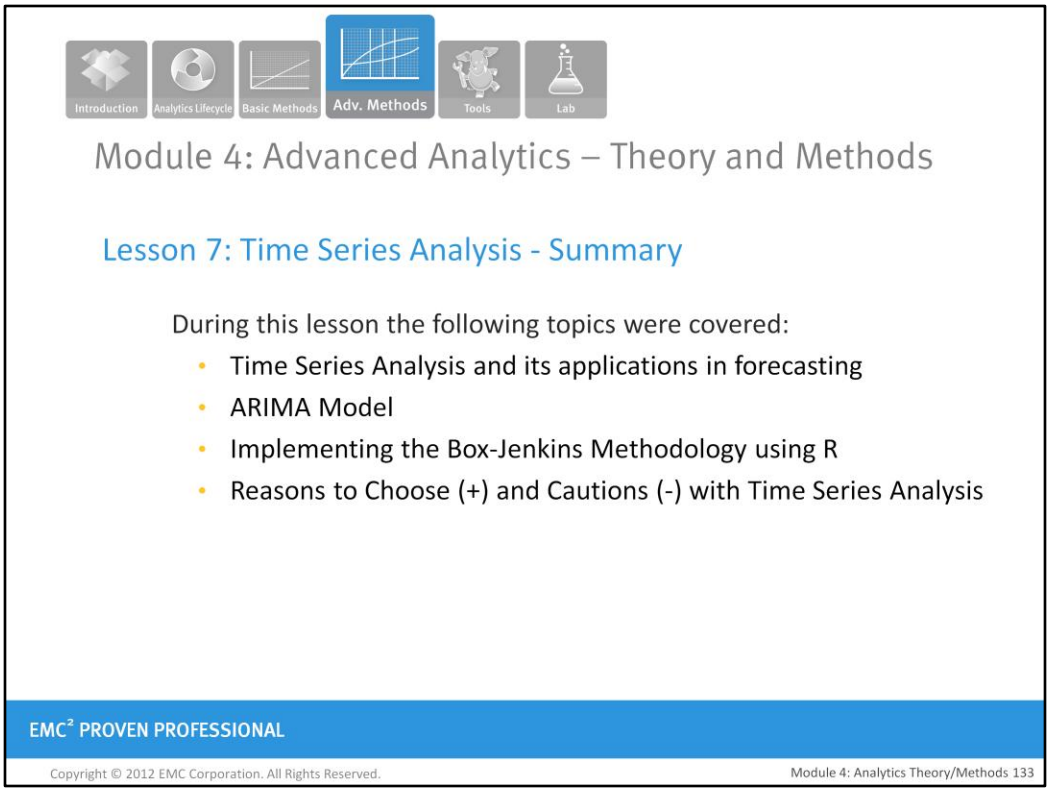
Check Your Knowledge

Your Thoughts?

1. What is a time series and what are the key components of a time series?
2. How do we "de-trend" a time series data?
3. What makes data stationary?
4. How is seasonality removed from the data?
5. What are the modeling parameters in ARIMA?
6. How do you use ACF and PACF to determine the "stationarity" of time series data?

EMC² PROVEN PROFESSIONAL

Record your answers here.



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

Lab Exercise 10: Time Series Analysis

This Lab is designed to investigate and practice Time Series Analysis with ARIMA models (Box-Jenkins-methodology).

After completing the tasks in this lab you should be able to:

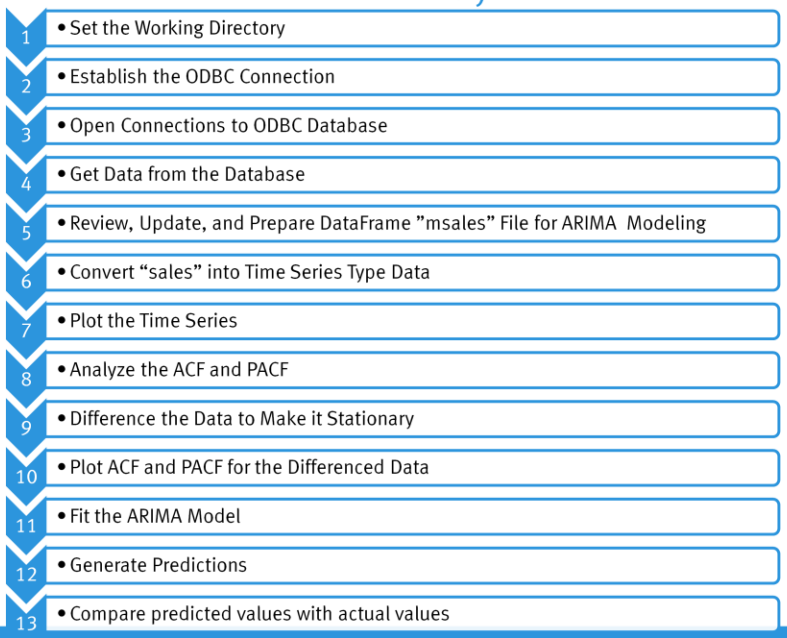
- Use R functions for ARIMA models
- Apply the requirements for generating appropriate training data
- Validate the effectiveness of the ARIMA models

EMC² PROVEN PROFESSIONAL

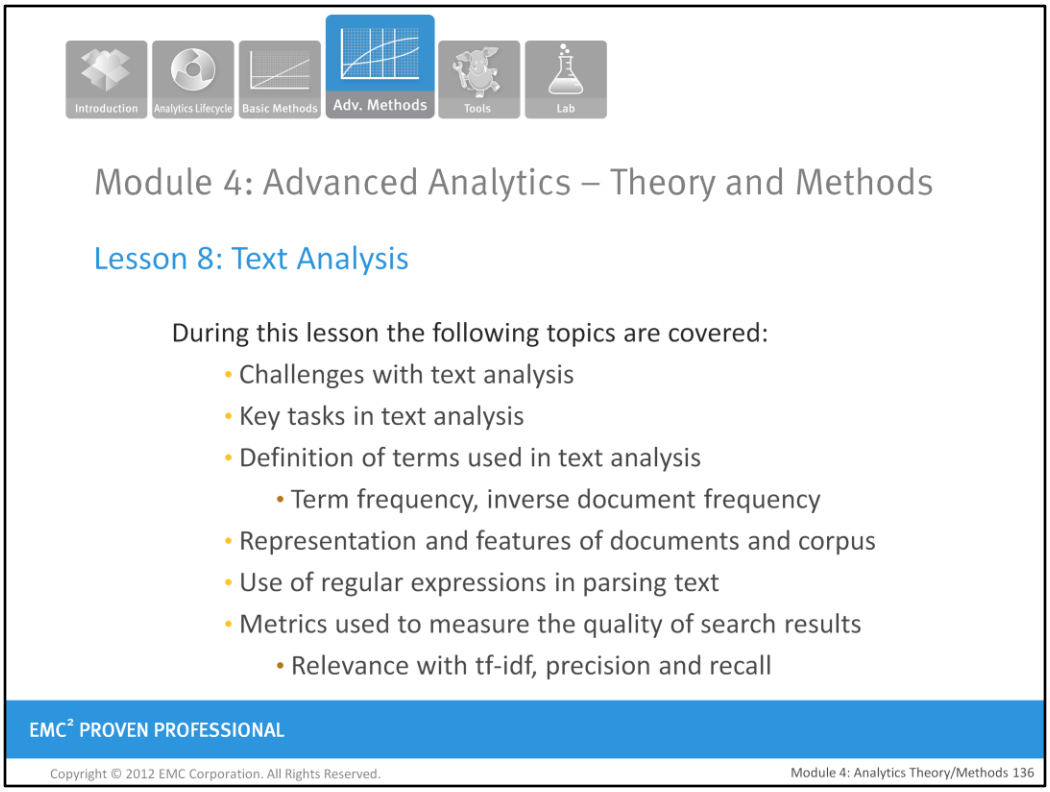
Tasks you will be completing in this lab include:

- Use RStudio environment to code ARIMA models.
- Review the methodology to create the weekly sales data from the retail database.
- Use generated model and evaluate how well it predicts the results using the model. Then compare the results with the original data.

Lab Exercise 10: Time Series Analysis - Workflow



EMC² PROVEN PROFESSIONAL

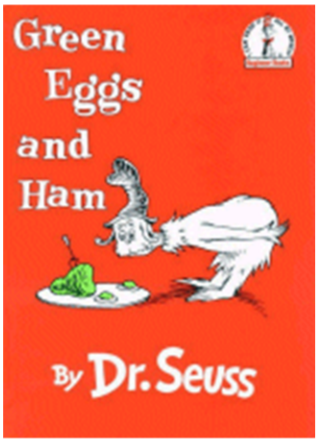


The topics covered in this lesson are listed.

Text Analysis

Encompasses the processing and representation of text for analysis and learning tasks

- **High-dimensionality**
  - Every distinct term is a dimension
  - *Green Eggs and Ham*: A 50-D problem!
- **Data is Un-structured**



EMC² PROVEN PROFESSIONAL

Text analysis is essentially the processing and representation of data that is in text form for the purpose of analyzing and learning new models from it.

The main challenge in text analysis is the problem of high dimensionality. When analyzing a document every possible word in the document represents a dimension.

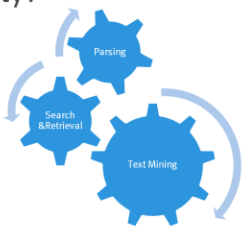
Consider the book 'Green Eggs and Ham' by Dr. Seuss, which he wrote responding to a challenge to write a book with just fifty different words.

(http://en.wikipedia.org/wiki/Green_Eggs_and_Ham). Even this book represents a 50 dimension problem if we consider vectors in a text space.

The other major challenge with text analysis is that the data is unstructured.

Text Analysis – Problem-solving Tasks

- Parsing
  - Impose a structure on the unstructured/semi-structured text for downstream analysis
- Search/Retrieval
  - Which documents have this word or phrase?
  - Which documents are about this topic or this entity?
- Text-mining
  - "Understand" the content
  - Clustering, classification
- Tasks are not an ordered list
  - Does not represent process
  - Set of tasks used appropriately depending on the problem addressed



EMC² PROVEN PROFESSIONAL

The process or the problem solving tasks in text analysis is composed of three important steps namely Parsing, Search/ Retrieval and Text mining.

Parsing is the process step that takes the un-structured or a semi-structured document and impose a structure for the downstream analysis. Parsing is basically reading the text which could be weblog, a RSS feed ,a XML or a HTML file or a word document. Parsing decomposes what is read in and renders it in a structure for the subsequent steps.

Once parsing is done, the problem focuses on **search and/or retrieval** of specific words or phrases or in finding a specific topic or an entity (a person or a corporation) in a document or a corpus (body of knowledge). All text representation takes place implicitly in the context of the corpus. All search and retrieval is something we are used to performing with search engines such as Google. Most of the techniques used in search and retrieval originated from the field of library science.

With the completion of these two steps, the output generated is a structured set of tokens or a bunch of key words that were searched, retrieved and organized. The third task is **mining the text** or understanding the content itself. Instead of treating the text as set of tokens or keywords, in this step we derive meaningful insights into the data pertaining to the domain of knowledge, business process or the problem we are trying to solve.

Many of the techniques that we mentioned in the previous lessons such as clustering and classification can be adapted to the text mining, with the proper representation of the text. We could use K-means clustering or other methods to tie the text into meaningful groups of subjects. Sentiment Analysis and Spam filtering are examples of a classification tasks in text mining. (recall that we listed Spam filtering as a prominent use case for Naïve Bayesian Classifier). In addition to traditional statistical methods, Natural Language processing methods are also used in this phase.

It should be noted the list of tasks are not ordered. One generally starts with the parsing, either with the intention of compiling them into a searchable corpus or catalog (maybe after some analytical tasks like tagging or categorization), OR specifically for the purpose of text mining. So it's not a process, it's a set of things that go into the text analysis task. Or maybe a tree, where you start with parsing, and go down to either search or to text-mining.

We will look into details of each of these steps in the rest of this lesson.

Example: Brand Management

- Acme currently makes two products
  - bPhone
  - bEbook
- They have lots of competition. They want to maintain their reputation for excellent products and keep their sales high.
- What is the buzz on Acme?
  - Search for mentions of Acme products
    - Twitter, Facebook, Review Sites, etc.
  - What do people say?
    - Positive or negative?
    - What do people think is good or bad about the products?

EMC² PROVEN PROFESSIONAL

Here we present an example “Brand Management” to detail the concepts in text analysis throughout this lesson.

The company Acme makes two products bPhone and bEbook. Acme is not the only one in the market making similar products. The competition is stiff and they want to maintain the reputation they have among e-book readers as an excellent product offering and also to enhance their sales.

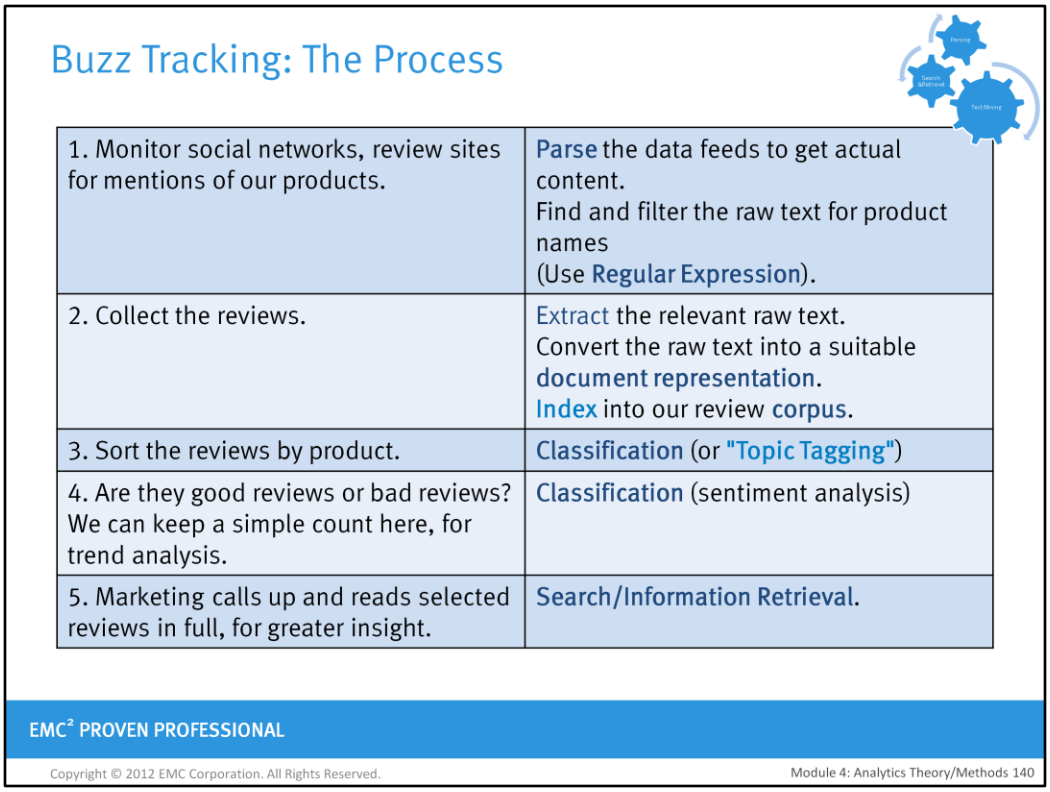
One of the ways they do this is to monitor what is being said about Acme products in the social media. In other words what is the buzz on Acme products. They want to search all that is said about Acme products in Twitter, Facebook and popular review sites (Amazon).

They want to know:

a) If people are mentioning their products?

b) What is being said – good or bad about the products. What people think is good or bad about Acme products. For example are they complaining about the battery life of the bPhone, or the latency in their bEbook.

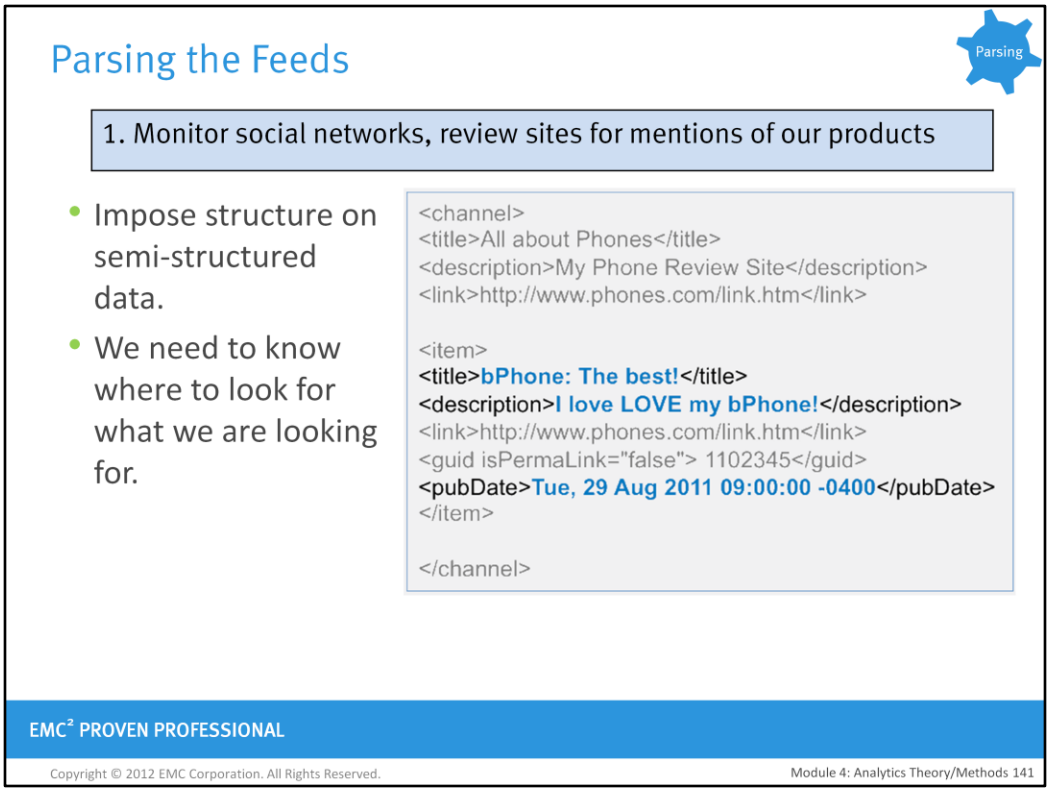
A full example would ask "how does bPhone compare to the competition, but let's keep the example simple.



Here we present a hypothetical and vastly oversimplified example of a process that you can adopt for the tracking what is said about Acme.

The first column of the table lists the tasks carried out for the buzz tracking and the second column lists the corresponding text analysis tasks associated with the established buzz tracking process.

The process is merely a way to organize the topics we present in this lesson, and to call out some of the difficulties that are unique to text mining.



Parsing in the linguistic sense means "to resolve a sentence into component parts of speech and explain syntactical relationships". (Merriam-Webster)

First, we want to monitor the data feeds, and parse them.

In this context, we are talking about parsing semi-structured data: html pages, RSS feeds, or whatever we may have.

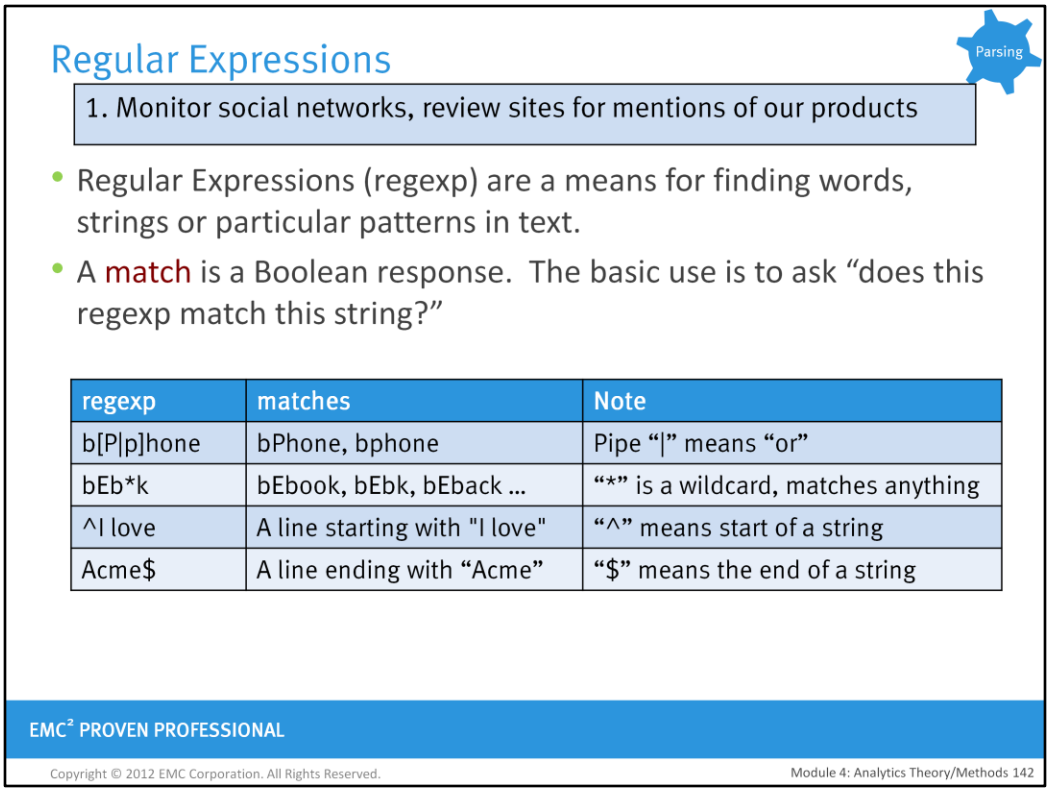
We need to impose enough structure so we can find the part of the raw text that we really care about -- in this case the actual content of review (including their titles), and when the reviews were posted.

This requires knowing the grammar of the data source. Sometimes it's relatively standard – HTML, RSS. Other times, it may not be quite as standard (web logs, for instance).

– **As an example**, An RSS (Really Simple Syndication) feed for a **smart phone review blog** is shown in the slide.

What is highlighted in the RSS feed shown here are the contents we are interested in. The “title”, “Description” and the “date”.

Once we know where to look, we can determine if it's what we are looking for.

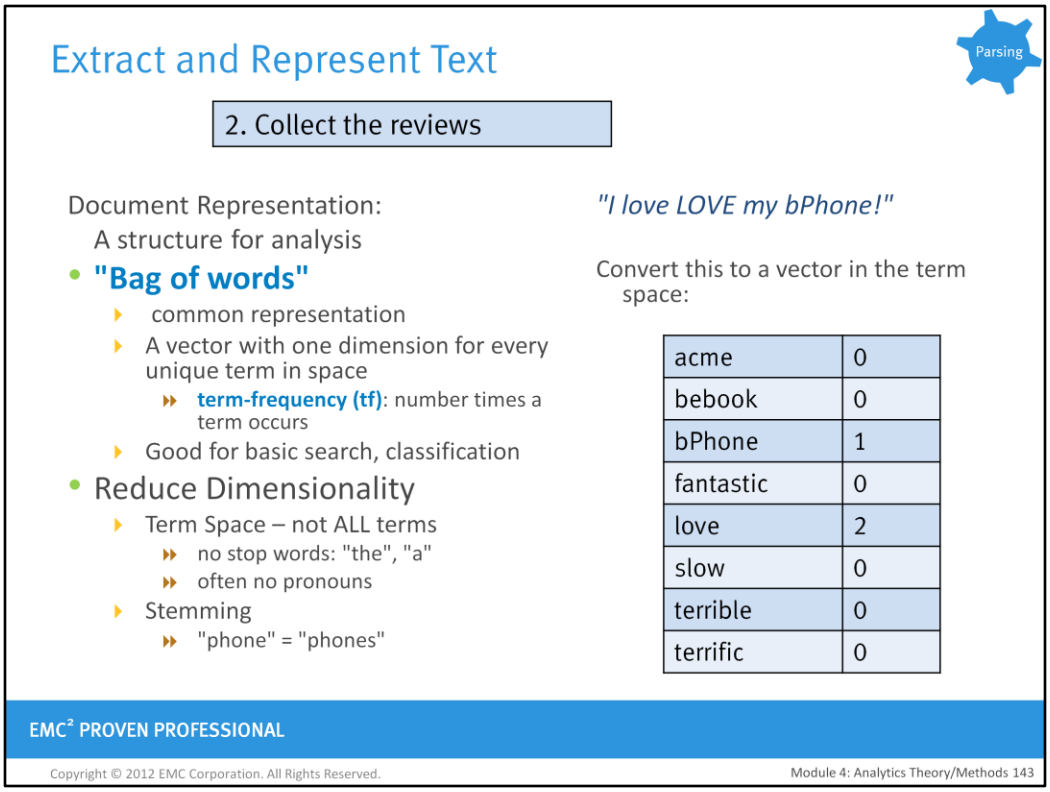


Regular Expressions is a popular technique used for finding words, strings or a particular patterns in the text. We will explore regular expression later in detail in Module 5.

The basic use is to determine if the regular expression (regexp) matches this string.

We have shown some examples of syntax used in regexp above. It is beyond the scope of this lesson to go into the details of the regexp syntax. But the general idea is that once we have the content from the fields of interest, we want to know if it is of interest to us. In this case: do those fields mention bPhone, bEbook, or Acme?

With regular expressions we can take into account capitalization (or lack of it), common misspellings, common abbreviations etc.



We are now in Step 2 . We have parsed all our data feeds and collected the phrases and words and we are ready to represent what we collected in a structured manner for down stream analysis.

The most common representation of the structure is known as the “bag of words”. The “Bag of words” is a vector with one dimension for every unique term in the space.

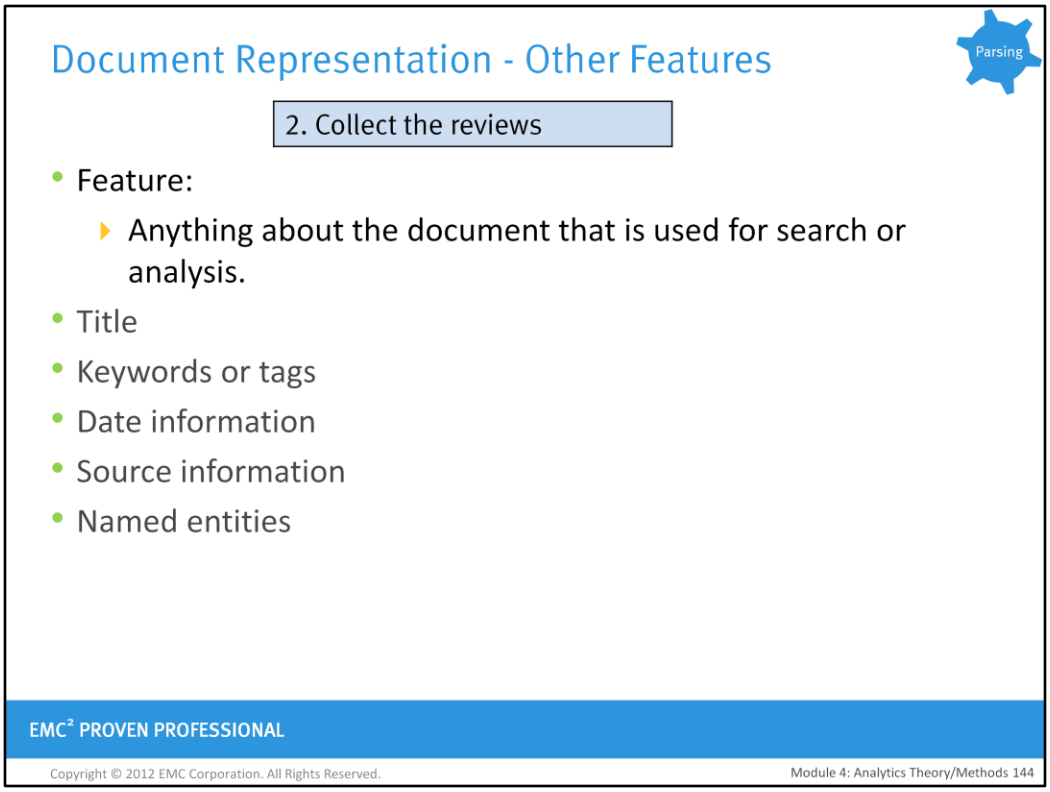
We also introduce the term “term-frequency” (tf) which is the number of times a term occurs in a vector.

Obviously the vector is VERY high-dimensional as we invariably end up with a significant number of unique words in a document. “Bag of words is a common representation and it is suited very well for search and classification. There are more sophisticated representations for sophisticated algorithms.

In the example above, the RSS feeds we parsed “I love LOVE my bPhone”, (we are only showing the part of our vector space).

We count the occurrences of the words in the text parsed and number of times the word is repeated and store word count as a part of the vector representation. In our example we see bPhone mentioned once and “love” mentioned twice.

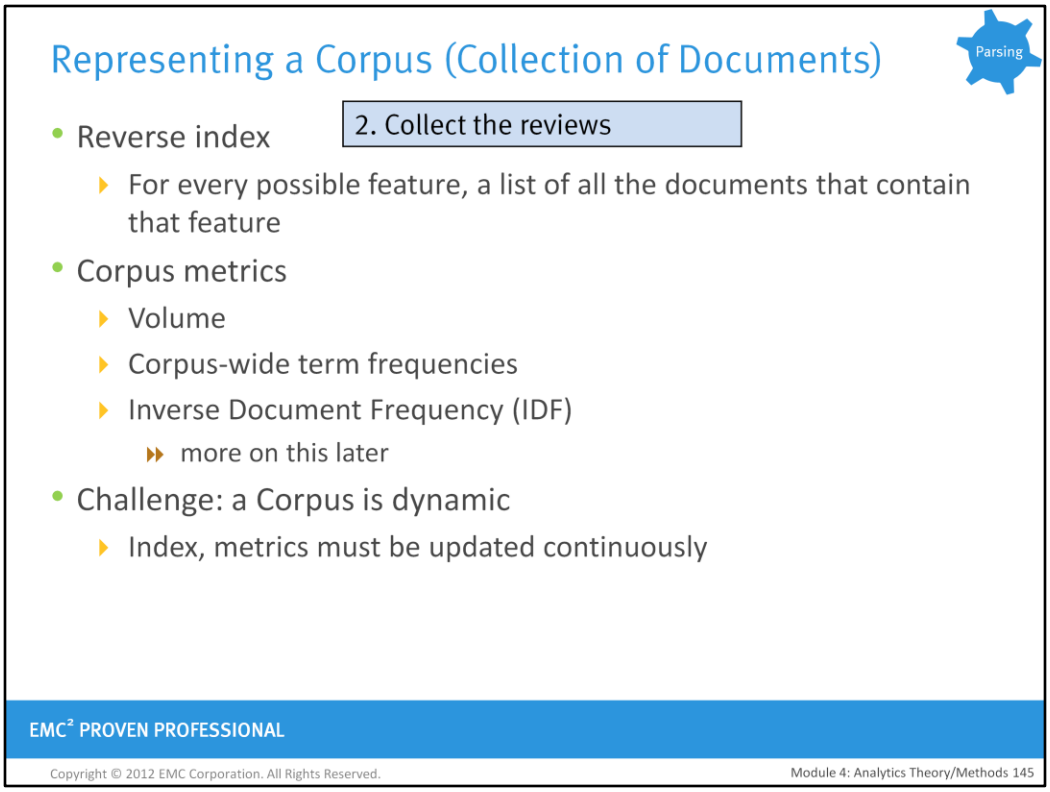
In order to reduce the dimensionality we do not include all words in the English language. Normally we ignore some “stop” words such as “the” “a” etc. There are other methods such as stemming the words and avoiding pronouns in the term space. Vector space must be managed in a way so that it only contains words that are essential for the analysis. Stemming is done based on the context and corpus. In a completely unstructured document techniques such as “parts of speech tagging” are used for parsing.



In addition to the "term, the features we store are the title of the document, any key words or tags attached to it, the date the document was created, the source from where the document was extracted (twitter, facebook, Amazon etc.) and some of the Named entities such as a mention of a competitor's name (do they compare bPhone to iPhone ?).

Sometimes creating these features is a text analysis task all to itself, like topic tagging. Companies invest significant resources in creating these tags as a separate activity. You see people tag their blogs to enable easy search and retrieval.

These features help with down stream analysis in classification or sentiment analysis.



It is important that we not only create a representation of the document but we also need to represent a corpus. What is the representation of a corpus?

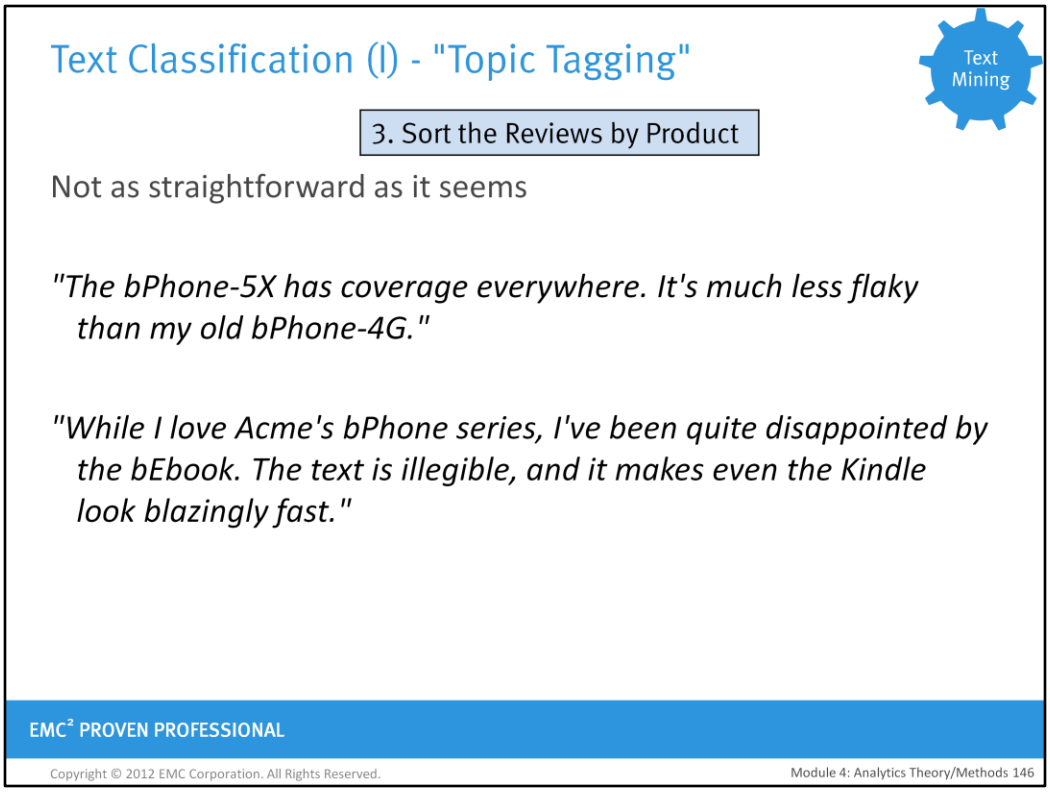
Now that we've collected the reviews and turned them into the proper representation, we want to archive them in a searchable archive for future reference and research. This is done with "reverse index" which provides a way of keeping track of list of all documents that contain a specific feature and for every possible feature.

The other corpus metrics such as volume and corpus-wide term frequency*, which specifies how the terms are distributed across the corpus, help with the down stream analysis of classification and searching. Search algorithms also* inverse document frequency which we define later in this lesson.

A fact that many people don't think about is that documents are often only relevant in the context of a corpus, or a specific collection of documents. Sometimes this is obvious, as in the case of search or retrieval. It is less obvious in the case of classification (for example, spam filtering, sentiment analysis) – but even in that case, the classifier has been trained on a specific set of documents, and the underlying assumption of all classifiers is that it will be deployed on a population that is similar to the population that it was trained on.

A primary challenge in text analysis and search is that a corpus changes constantly over time: not only do new documents get added (which means the metrics and indices must be updated), but word distributions can change over time (which will reduce the effectiveness of classifiers and filters, if they are not retrained – think about spam filters).

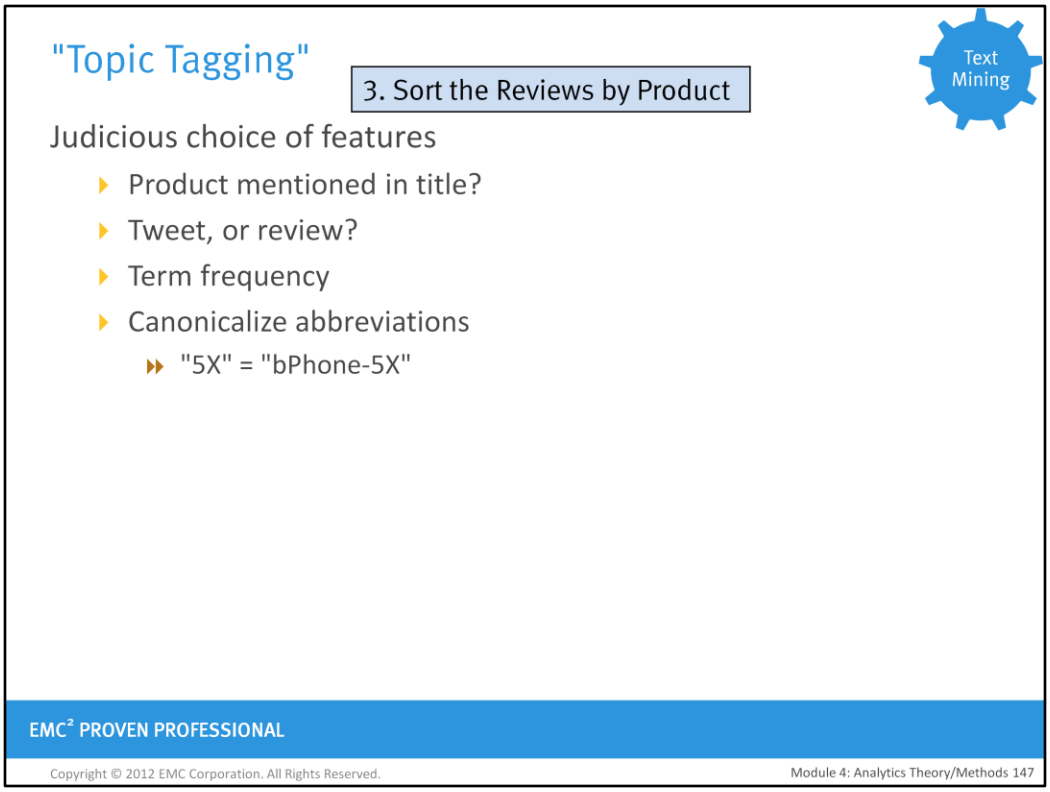
The corpus representation that we discuss here is primarily oriented towards search/retrieval, but some of the metrics, like IDF can also be relevant to classification as well.



Now all the reviews are collected and represented we want to sort them by product. This is done with topic tagging. For the two reviews shown:

- Is the first review about bPhone-5x or bPhone-4g?
- Is the second review is about bPhone or bEbook or Kindle?

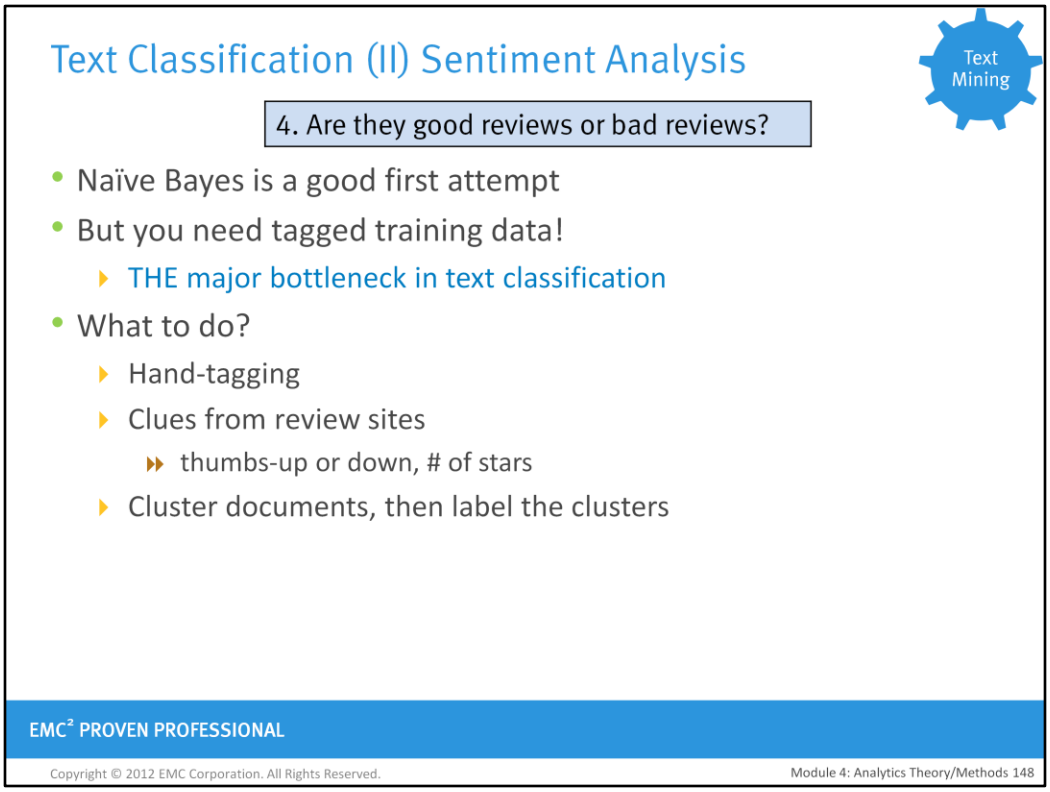
It is a complex problem to properly tag a document and it is not as straightforward as it appears. There are several methods available such as simply counting the number of occurrences of a product name to many sophisticated methods. More on this in the following slide.



There are rules you can come up with to determine how to sort a document (in a given context).

If the bphone5X is mentioned in the title, then the document is likely to be about the 5X, and mentions of the 4G in the text may or may not be relevant (to tagging). A tweet that mentions the product is probably about the product (whereas a review may mention many products as comparisons). More frequent mentions of the product in the document are a clue. Somewhere, you need to resolve abbreviations into the correct product (in the term space).

One could manually compile these rules (dirty secret – many folks do). Ideally, the Data Scientist should have a good idea what the relevant features are for a given task, and structure the document representation to fit both the explanatory features, and the algorithm that is used to do the classification/tagging. This process is part of the Data Analytics Lifecycle we discussed in Module 2.



At this point in the process, Acme already has a sentiment classification engine; here we are going to discuss how one might build one.

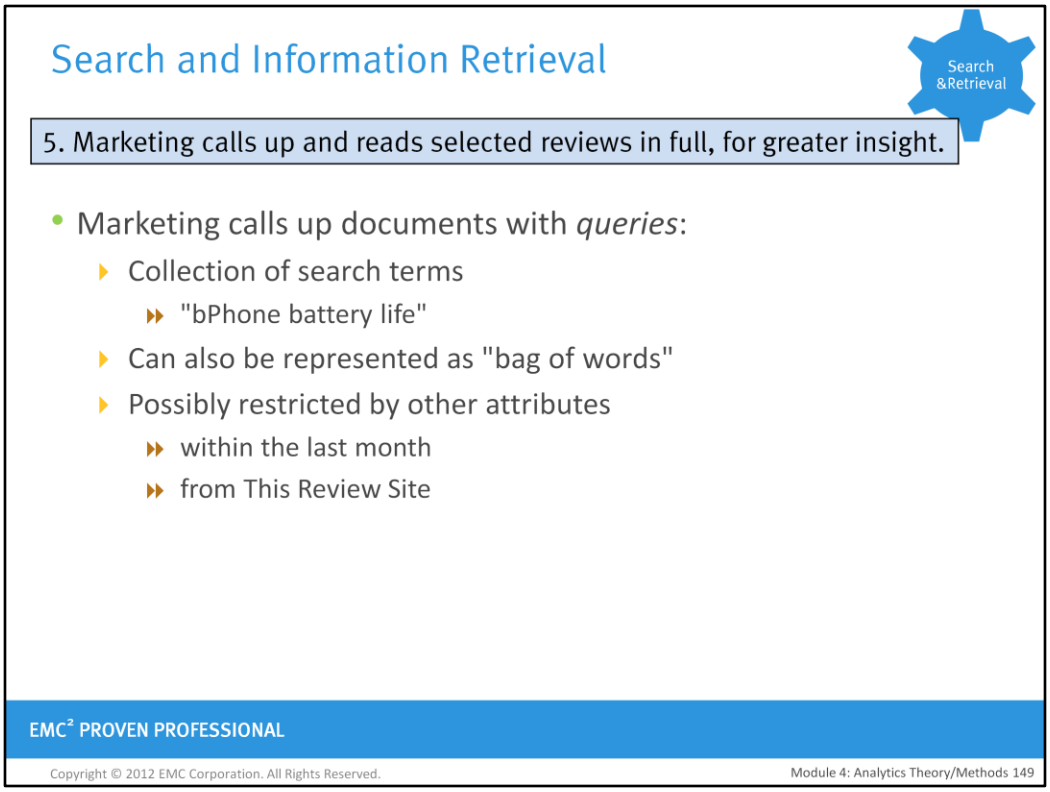
The take-away here is that the challenge in text classification is often not the algorithm; it's getting the tagged data.

Many companies, like Amazon or Shopping.com, rely on teams of hand-taggers to create training corpora to jump-start efforts in automated categorization. Hand-tagged data is slow to collect, and is prone to fatigue errors and inconsistent (subjective) tagging on the part of the taggers.

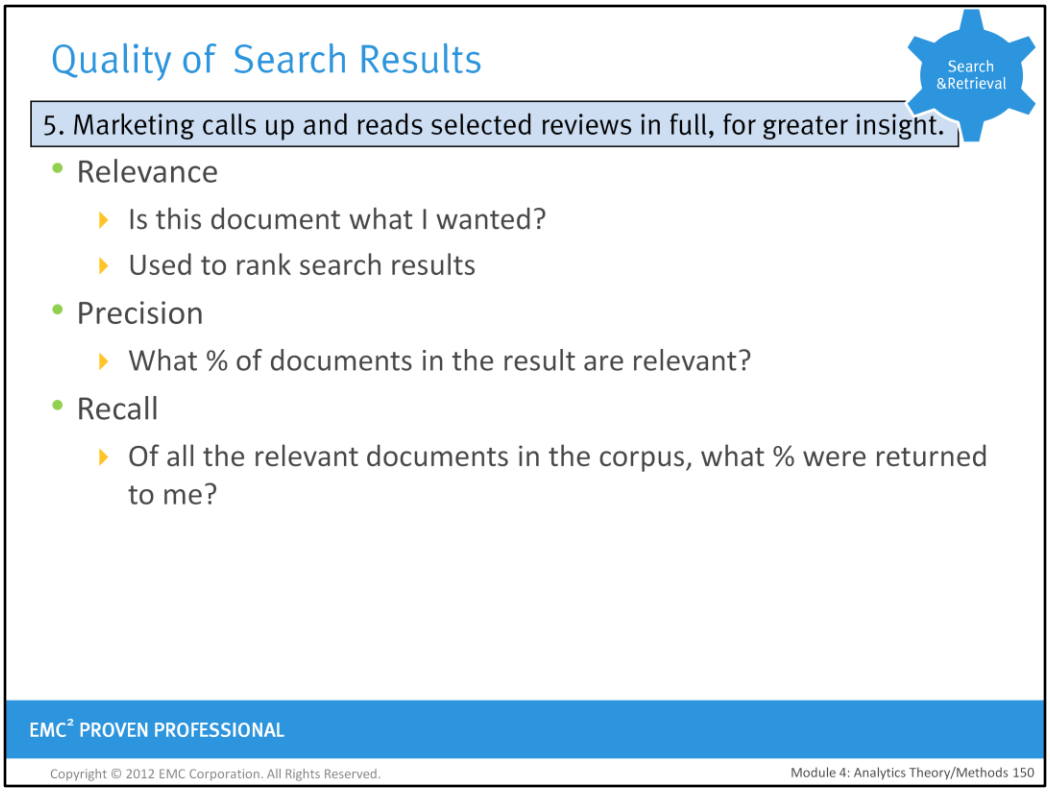
In the case of sentiment analysis, one could try creating training corpora based on sites that have quantitative ratings for the products; the resulting classifiers run the risk of only being effective on the reviews from sites that they came from (or for reviews from that product category), because of idiosyncratic terminology of the website community, or the product category. As an example, "lightweight" is a positive adjective for laptops, but not necessarily for wheelbarrows, or books. Classifiers built from reviews would almost definitely not work on tweets or blog comments.

Using unsupervised methods to cluster the documents, and then assigning labels based on whether or not the sampled documents from a cluster are positive or negative might work – but since the cluster is not built specifically on sentiment, it may not partition on sentiment.

There are other things you can do to track sentiment, besides classification: for instance, you can track the frequency with which certain words appear in reviews of your products, and then let a human decide if the overall trend looks positive or negative. The point of this discussion is not to cover all the possible ways of text mining, but to cover the basic concepts and issues.



Finally we got our corpus created tags and we have some sentiment analysis and the marketing team wants to call up these documents. This is typically done with a query which may specify calling up of documents from a particular site or reviews in a specific data range. This basically is a search problem, finding the document that meets the search criteria.



Let us now focus on the quality of search results. It basically is determining if the results you receive are indeed the ones you wanted or not. Relevance, precision and recall are the metrics that are used to determine the quality of search results.

We come up with an objective measure of relevance (Is this the document the user wanted) and rank the search results based on **Relevance** and provide users the most relevant documents ahead of those that score low on relevance.

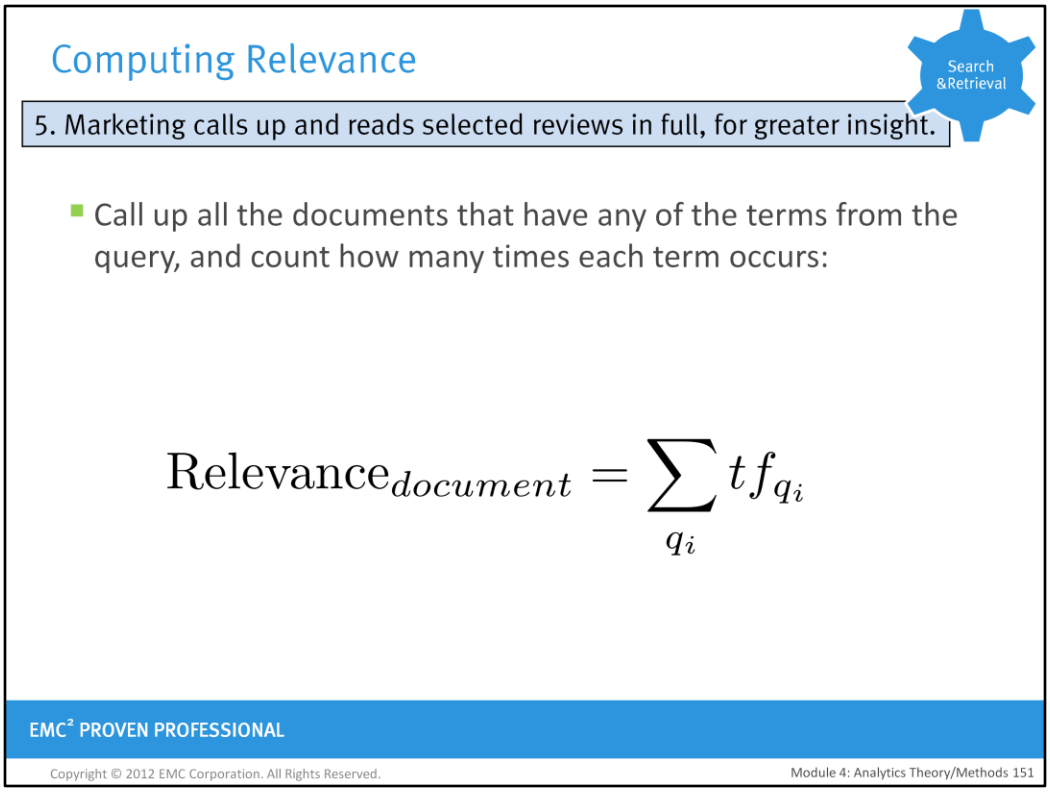
Precision and **Recall** are measures of accuracy of the search. Precision is defined as the % of documents in the results that are relevant. If we say bPhone and it gives back a 100 documents and 70 of them are relevant the precision is 70%.

Recall is the % of returned documents among all relevant documents in the corpus.

Relevance and Precision are always important concepts, whether you are talking about a web search or information retrieval from a finite corpus (like our review archive).

Recall is basically a meaningless concept when you are discussing general web search. Or to put it another way: it will probably always be low, you just hope it's not zero. But it may be relevant in finite corpus.

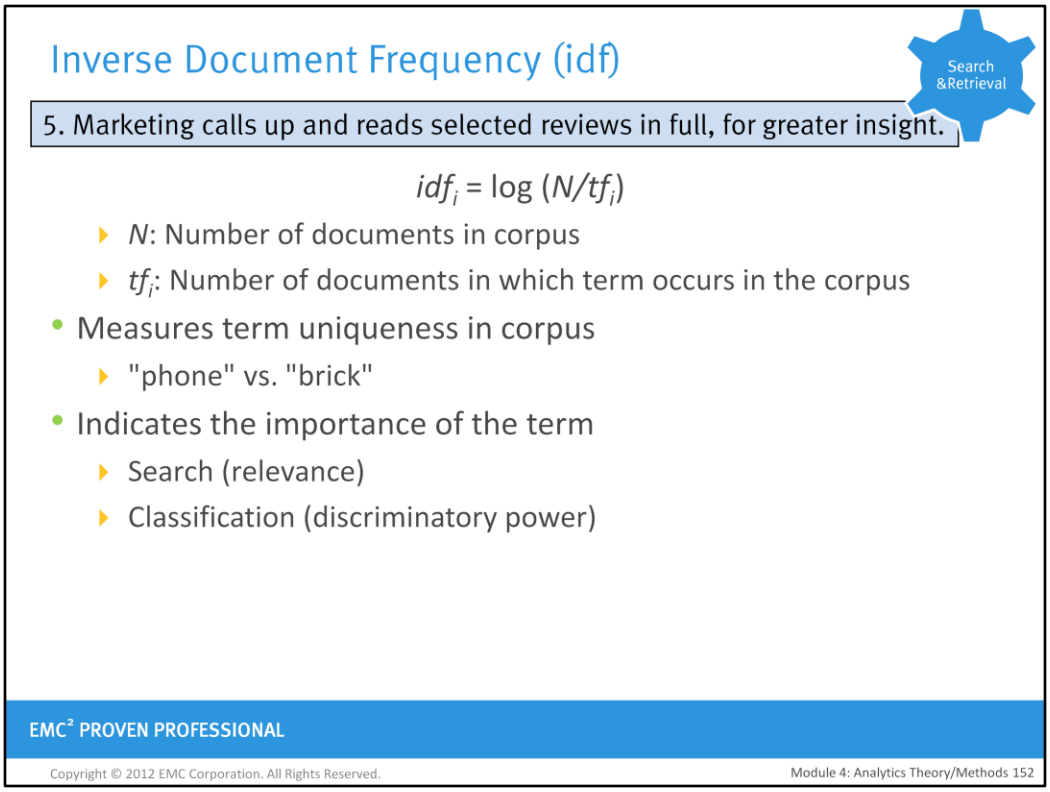
Search algorithms (and classification algorithms, in general) are usually evaluated in terms of precision and recall by the computer science community.



Here we present a simple example of how relevance might be computed.

We call up all the documents that have any of the terms from the query and count how many times each term occurs. For example the more often “bPhone” and “Battery Life” are mentioned in the document the more relevant the document is.

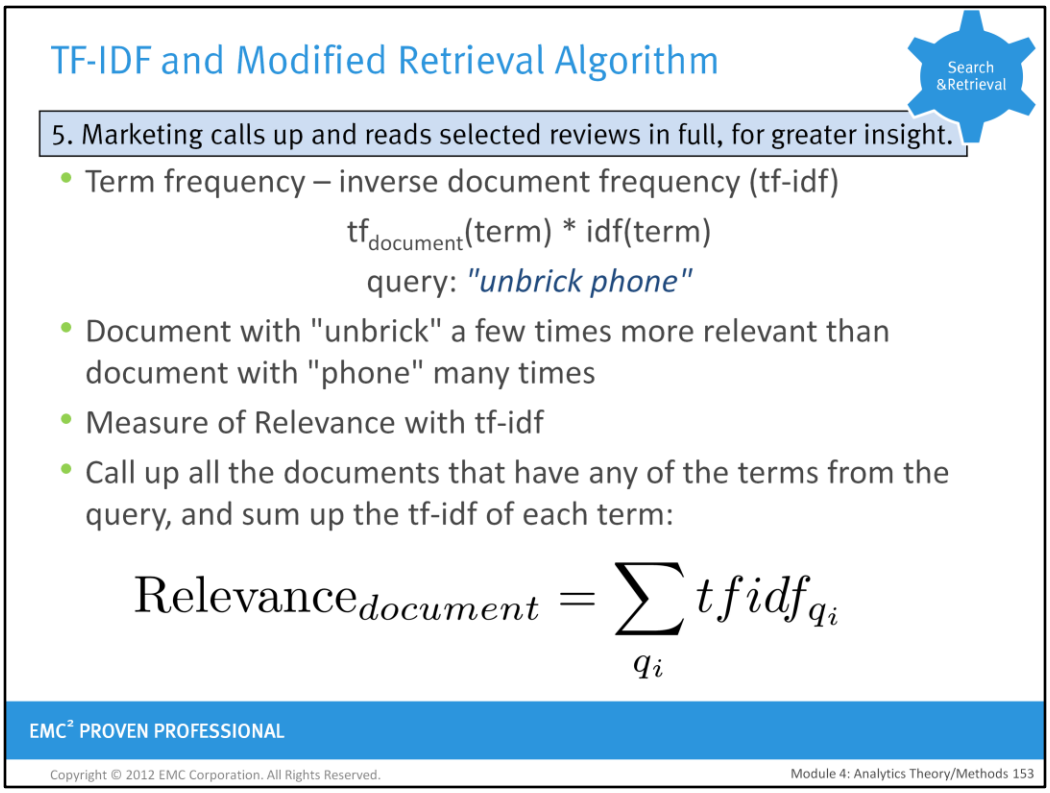
Obviously, there are ways to improve this method. For example, one might prefer documents that include ALL the terms, not just any. Also, one might want to limit the weight accorded to any one term ("Spam spam spam spam, wonderful spam….").



We now define Inverse Document Frequency and look into how we can improve our search algorithm with idf.

idf measures the uniqueness of a term in the corpus. If a term shows up only in 10% of the documents then it is unique. If a term shows up in 90% of the documents then it is not all that unique. It indicates the importance of the term (that appears in 10% of documents) and **provides relevance to the search by weighing the rare term higher**.

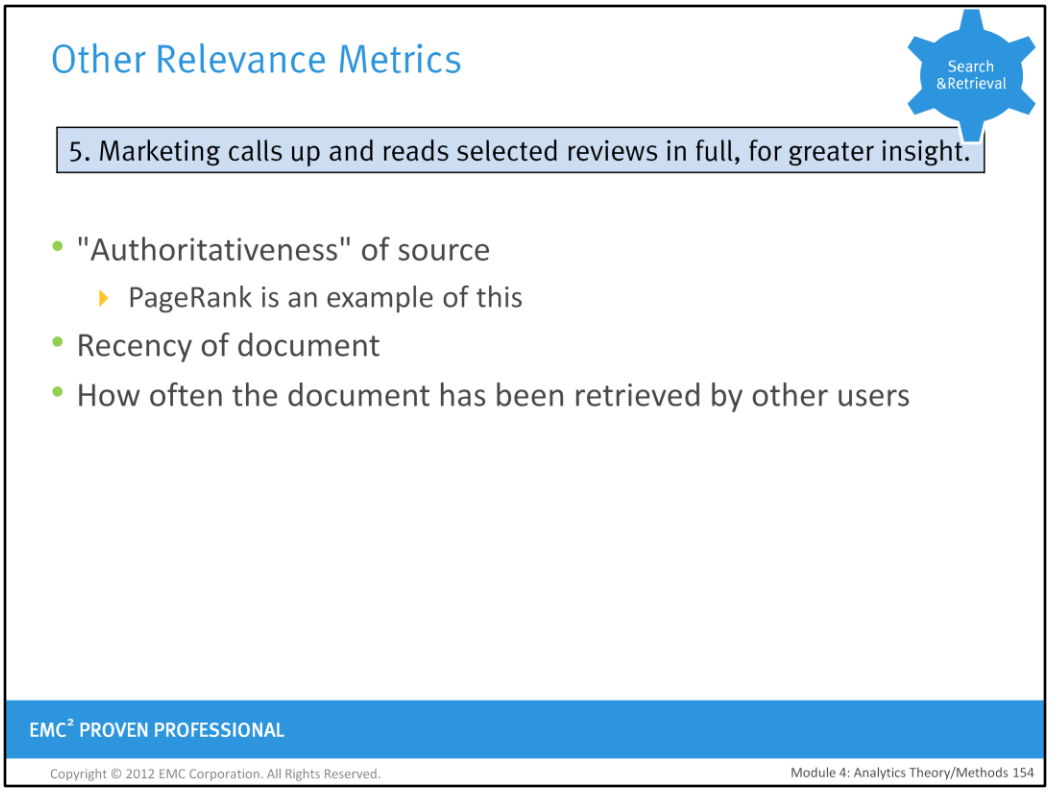
In a corpus of phone reviews, the word "phone" is probably pretty common; in particular it shows up in both good and bad reviews. The term "brick" is probably less common. So it is an important term when it shows up in a query (it discriminates relevant documents better than "phone" does), and potentially is distributed differently in good reviews and bad reviews. IDF reflects the fact that "brick" is potentially an interesting feature of a document.



tf-idf is the product of term frequency (tf) and inverse document frequency (idf). **It provides measure that will weight the presence of unusual terms in the query as higher indications of document relevance than the presence of more common terms.**

In our query example “unbrick phone” tf-idf ensures that documents with “unbrick” are made more relevant than the document with “phone”.

We use the relevance as the sum of tf-idf and this modification to the search algorithm will yield better results in this corpus.

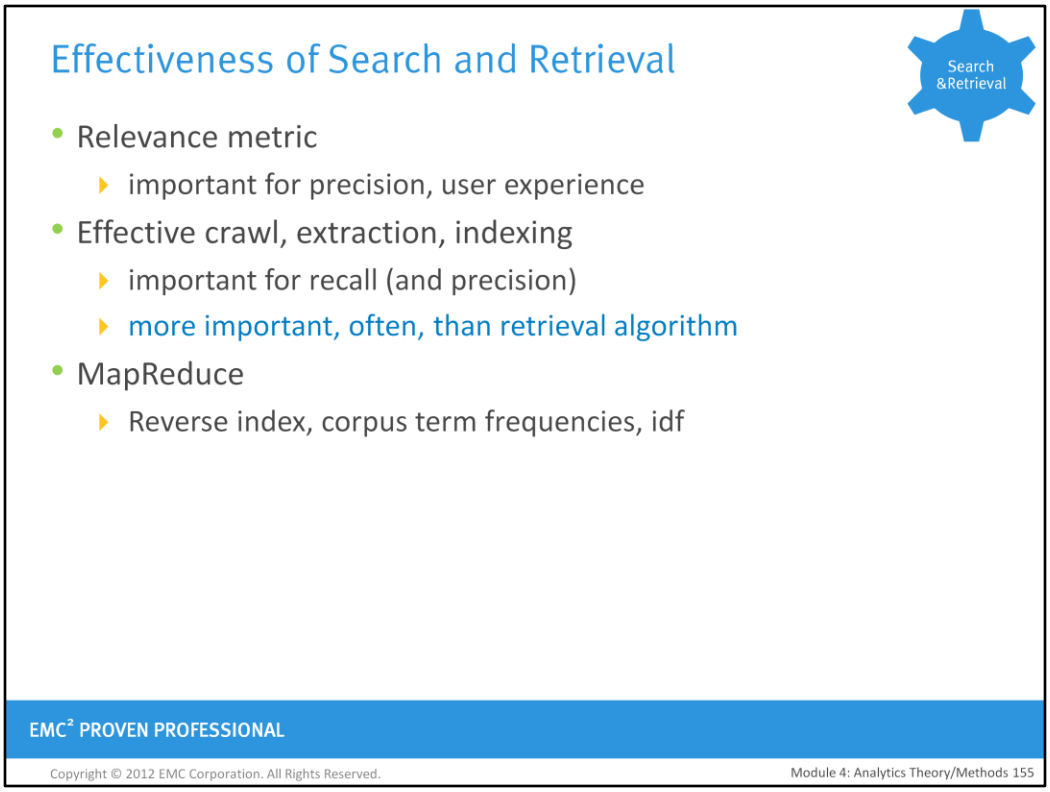


There are other measures of relevance that are usually used in conjunction with term-based (for example, tfidf) relevance.

Authoritativeness of source is one such measure (PageRank – used by Google is an example)

Recency – new documents are more relevant than old ones

Keeping records of how often a document is retrieved as part of the corpus metrics by other users also provides a relevancy measure.



There are other retrieval algorithms, probably more effective than the basic one that we described. But the important thing is that the documents be available for search.

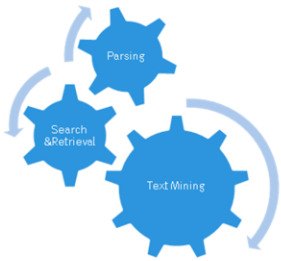
The relevance metric is important for the precision and user experience. Crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches.

The search engineers who provide the infrastructure for the search and retrieval process play a key role in “text analysis”. More so than played by Data Scientists.

The tasks such as reverse indexing, finding the idfs and corpus term frequencies are implemented effectively with map and reduce algorithms that we will detail in Module 5.

Challenges - Text Analysis

- Challenge: finding the right structure for your unstructured data
- Challenge: very high dimensionality
- Challenge: thinking about your problem the right way



EMC² PROVEN PROFESSIONAL

We again recap on the key challenges with text analysis.

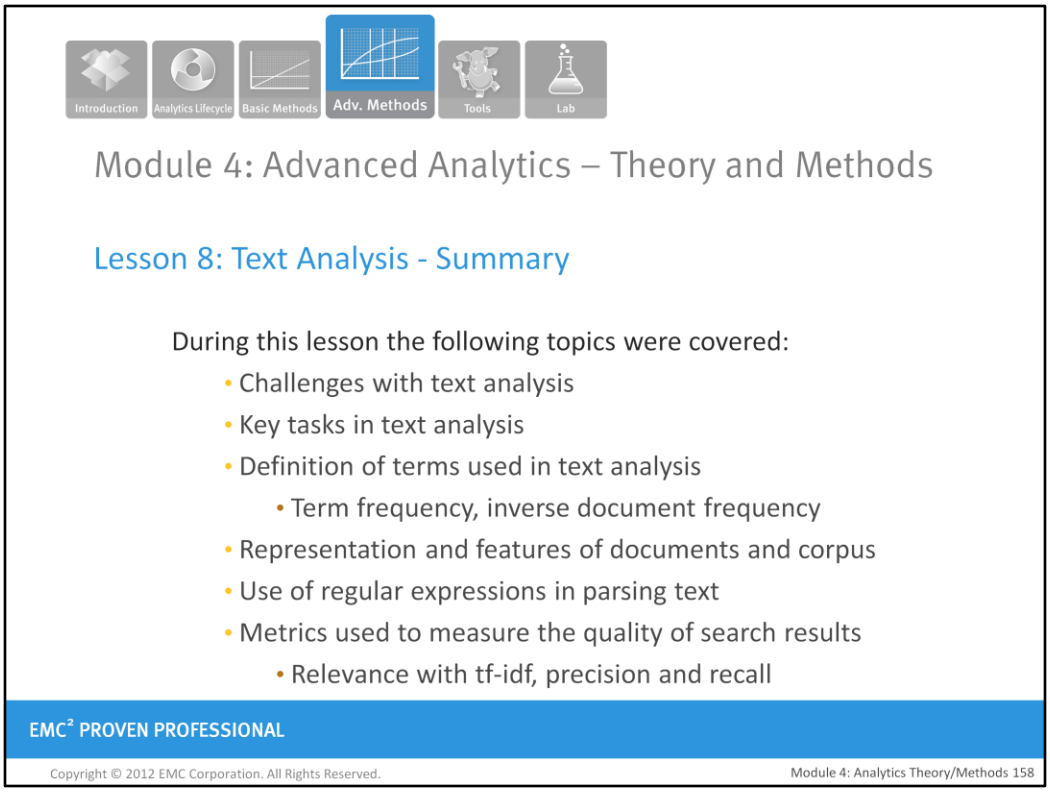
As we saw in Module 2, the most challenging aspect of data analytics problems often isn't the statistics or mathematical algorithms; it's formulating the problem, getting the data, and preparing the data. This is especially true for text analysis.

Check Your Knowledge

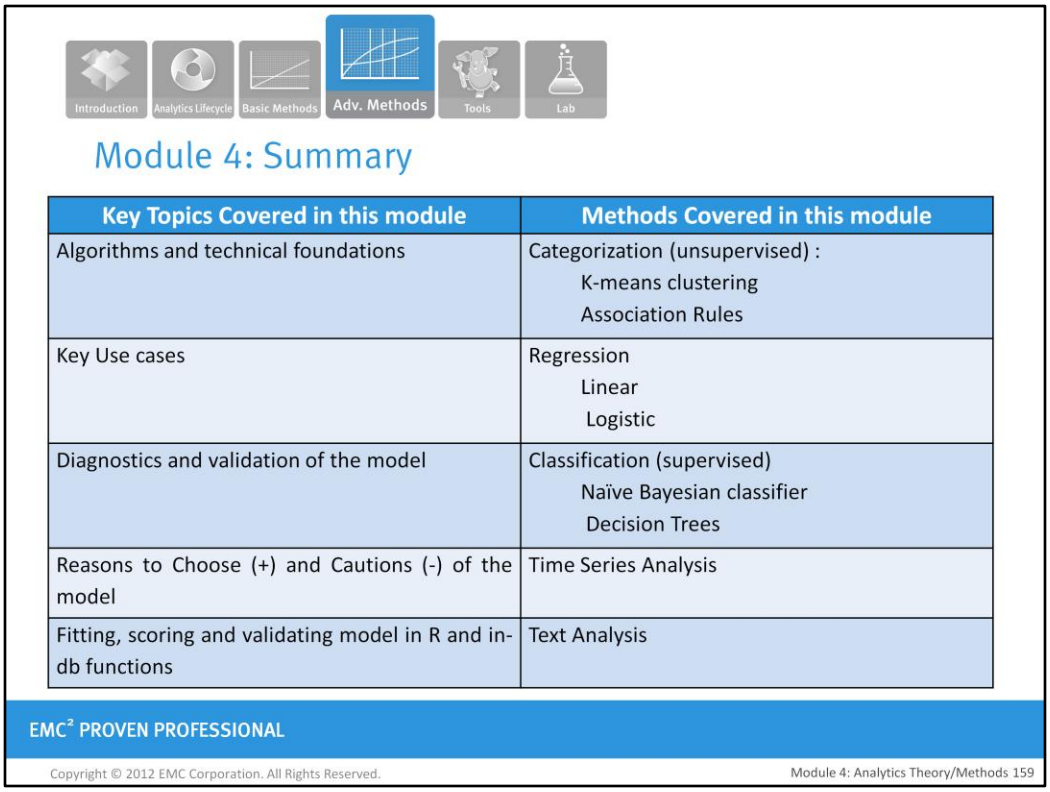
1. What are the two major challenges in the problem of text analysis?
2. What is a reverse index?
3. Why is the corpus metrics dynamic. Provide an example and a scenario that explains the dynamism of the corpus metrics.
4. How does tf-idf enhance the relevance of a search result?
5. List and discuss a few methods that are deployed in text analysis to reduce the dimensions.

EMC² PROVEN PROFESSIONAL

Record your answers here.



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



Summary of key-topics presented in this Module are listed.

This slide intentionally left blank.

EMC² PROVEN PROFESSIONAL