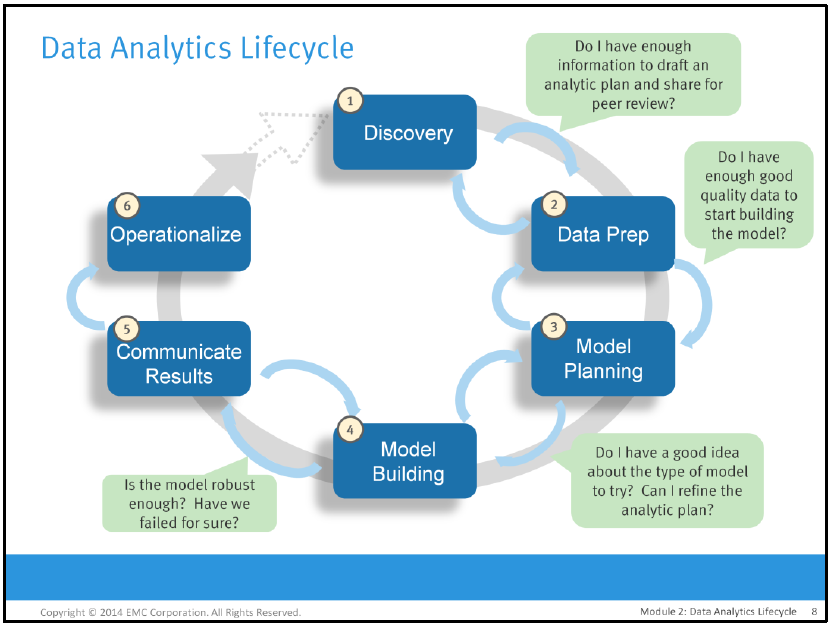
Data Science and Big Data Analytics projects

* Reframe business challenges as analytics challenges
* Design, implement and deploy statistical models and data mining techniques on big data
* Create insights the lead to actionable recommendations



Phase 1: Discovery

Learn the business domain, including relevant history, such as whether the organization or business unit has attempted similar projects in the past, from which you can learn. Assess the resources you will have to support the project, in terms of people, technology, time, and data. Frame the business problem as an analytic challenge that can be addressed in subsequent phases. Formulate initial hypotheses (IH) to test and begin learning the data.

Phase 2: Data Preparation

Prepare an analytic sandbox, in which you can work for the duration of the project. Perform ELT and ETL to get data into the sandbox, and begin transforming the data so you can work with it and analyze it. Familiarize yourself with the data thoroughly and take steps to condition the data.

In order to use this raw data in our models, in almost all cases, we need to perform preprocessing, which might include:

* Filtering data
* Dealing with missing, incomplete, or corrupted data
* Dealing with potential anomalies, errors, and outliers
* Joining together disparate data sources
* Aggregating data

Once we have performed initial preprocessing on our data, we often need to transform the data into a representation that is suitable for machine learning models. For many model types, this representation will take the form of a vector or matrix structure that contains numerical data. Common challenges during data transformation and feature extraction include:

Numerical Features

These features are typically real or integer numbers. So use in raw format, without any conversions.

Need to measure the **weights** of each feature. Like age has the direct relation in predicting the outcome variable in the movie recommendation example, so its weight is more compared to other features.

We often convert numerical data into categorical data to reduce the number of values a variable can take on. An example of this is converting a variable for age into buckets (such as 25-35, 45-55 …).

Transforming numerical features; for example, applying a log transformation to a numerical variable can help deal with variables that take on a very large range of values.

Normalizing and standardizing numerical features ensures that all the different input variables for a model have a consistent scale. Many machine learning models require standardized input to work properly.

Categorical Features

Categorical features cannot be used as input in their raw form, as they are not numbers; instead, they are members of a set of possible values that the variable can take.

To transform categorical variables into a numerical representation, we can use a common approach known as **1-of-k encoding**. An approach such as 1-of-k encoding is required to represent nominal variables in a way that makes sense for machine learning tasks. Ordinal variables might be used in their raw form but are often encoded in the same way as nominal variables.

Derived Features

Examples of features derived from raw data include computing average values, median values, variances, sums, differences, maximums or minimums, and counts.

Text features

In some ways, text features are a form of categorical and derived features.

Approaches for text-feature extraction - **bag-of-words**:

The bag-of-words approach treats a piece of text content as a set of the words, and possibly numbers, in the text (these are often referred to as terms). The process of the bag-of-words approach is as follows:

* Tokenization: First, some form of tokenization is applied to the text to split it into a set of tokens (generally words, numbers, and so on). An example of this is simple whitespace tokenization, which splits the text on each space and might remove punctuation and other characters that are not alphabetical or numerical.
* Stop word removal: Next, it is usual to remove very common words such as "the", "and", and "but" (these are known as stop words).
* Stemming: The next step can include stemming, which refers to taking a term and reducing it to its base form or stem. A common example is plural terms becoming singular (for example, dogs becomes dog and so on). There are many approaches to stemming, and text-processing libraries often contain various stemming algorithms.
* Vectorization: The final step is turning the processed terms into a vector representation. The simplest form is, perhaps, a binary vector representation, where we assign a value of one if a term exists in the text and zero if it does not. This is essentially identical to the categorical 1-of-k encoding we encountered earlier. Like 1-of-k encoding, this requires a dictionary of terms mapping a given term to an index number. As you might gather, there are potentially millions of individual possible terms (even after stop word removal and stemming). Hence, it becomes critical to use a sparse vector representation where only the fact that a term is present is stored, to save memory and disk space as well as compute time.

**Normalizing features**

Once the features have been extracted into the form of a vector, a common preprocessing step is to normalize the numerical data. The idea behind this is to transform each numerical feature in a way that scales it to a standard size. We can perform different kinds of normalization, which are as follows:

* Normalize a feature: This is usually a transformation applied to an individual feature across the dataset, for example, subtracting the mean (centering the feature) or applying the standard normal transformation (such that the feature has a mean of zero and a standard deviation of 1).
* Normalize a feature vector: This is usually a transformation applied to all features in a given row of the dataset such that the resulting feature vector has a normalized length. That is, we will ensure that each feature in the vector is scaled such that the vector has a norm of 1 (typically, on an L1 or L2 norm).

MLib APIs for normalization- *StandardScaler*, which applies the standard normal transformation, and *Normalizer*, which applies the same feature vector normalization

Phase 3: Model Planning

Determine the methods, techniques and workflow you intend to follow for the model scoring. Explore the data to learn about the relationships between variables, and subsequently select key variables/features and the models you are likely to use.

Phase 4: Model Building

Develop data sets for testing, training, and production purposes. Get the best environment you can for executing models and workflows, including fast hardware and parallel processing.

Phase 5: Communicate Results

Determine if you succeeded or failed, based on the criteria you developed in the Discovery phase, in collaboration with your stakeholders. Identify your key findings, quantify the business value and develop a narrative to summarize your findings and convey to stakeholders.

Phase 6: Operationalize

Deliver final reports, briefings, code, and technical documents. Run a pilot project, and implement your models in a production environment.