BAIS:6100 Text Analytics

Document-Term Representation
Text Classification

Kang-Pyo Lee

Course Schedule (Subject to Change)

Week	Date	Topics	Due
1	Jan 28	Introduction to Text Analytics Introduction to Python, Jupyter Notebook, and UI Interactive Data Analytics Service (IDAS)	
2	Feb 4	Module 1. Python Basics for Text Processing, Part 1 : Strings, Collections, Built-in Functions, Flow Control, and User-Defined Functions	
3	Feb 11	Module 2. Python Basics for Text Processing, Part 2 : Files, Dataframes, and Pattern Matching Using Regular Expressions	HW 1
4	Feb 18	Module 3. Basic Natural Language Processing (NLP) Techniques : Tokenization, Part-of-Speech Tagging, Stemming, Lemmatization, N-grams, Noun Phrase Extraction, Language Detection and Translation, and Gender Prediction Module 4. Keyword Analysis and Visualization	HW 2
5	Feb 25	Test 1	HW 3 (Feb 24)
6	Mar 4	Modules 5 & 6. Text Data Collection Using Twitter APIs and Web Scraping Group Project Announcement	
7	Mar 11	Module 7. Document-Term Representation Module 8. Text Classification	HW 4
8	Mar 18	Module 9. Text Clustering and Topic Modeling	Project Proposal
9	Mar 25	Module 10. Text Similarity Module 11. Keyword Network Analysis	
10	Apr 1	Test 2	HW 5 (Mar 31)
11	Apr 8	Group Project Presentations and Course Wrap-Up	Project Deliverables

Reminder

No homework for the next two weeks!

Make sure to submit the group project proposal by 6 pm on Thu, Mar 18

Machine Learning with Python

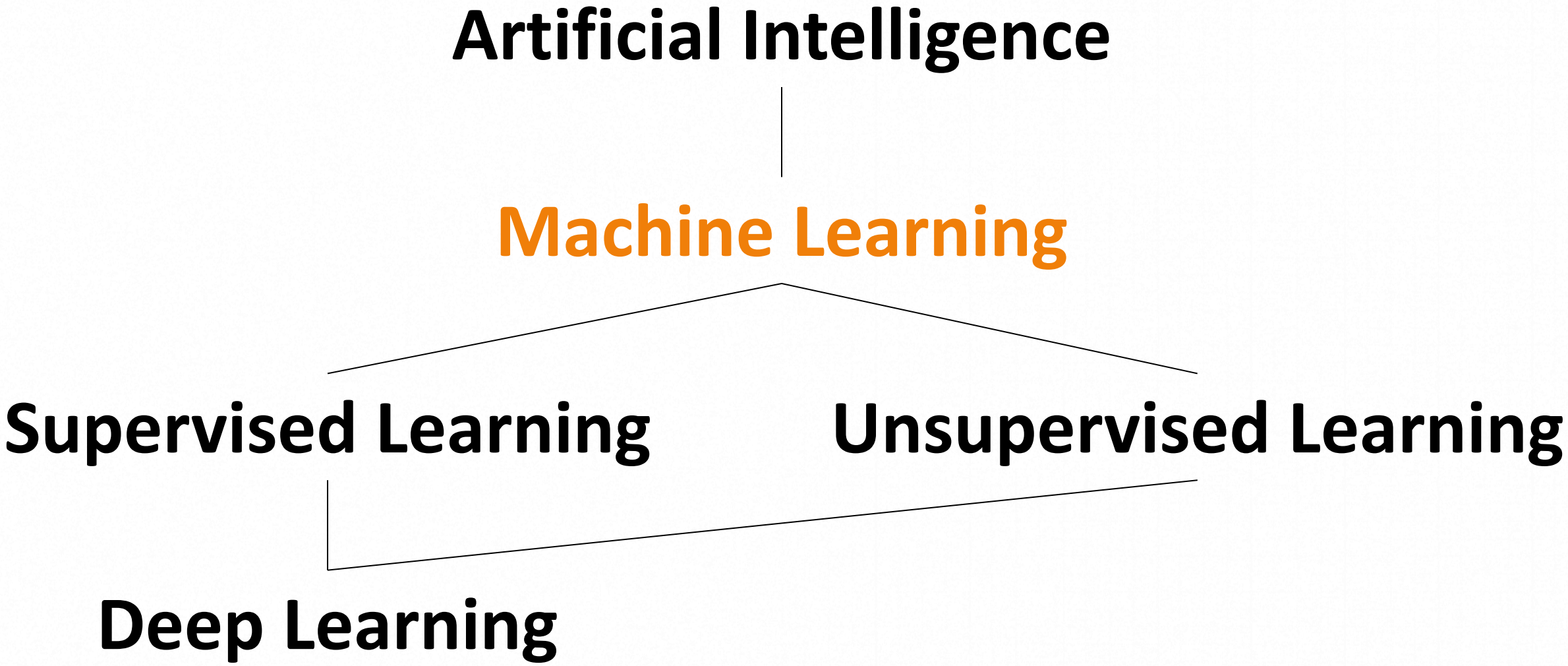
Machine learning is a complex topic

Python, however, can help you develop and evaluate machine learning models very quickly and easily

What Is Machine Learning?

Machine learning is a form of Artificial Intelligence (AI) that enables a machine to learn from data, just as humans learn from experience

Machine Learning Hierarchy



Supervised vs. Unsupervised Learning

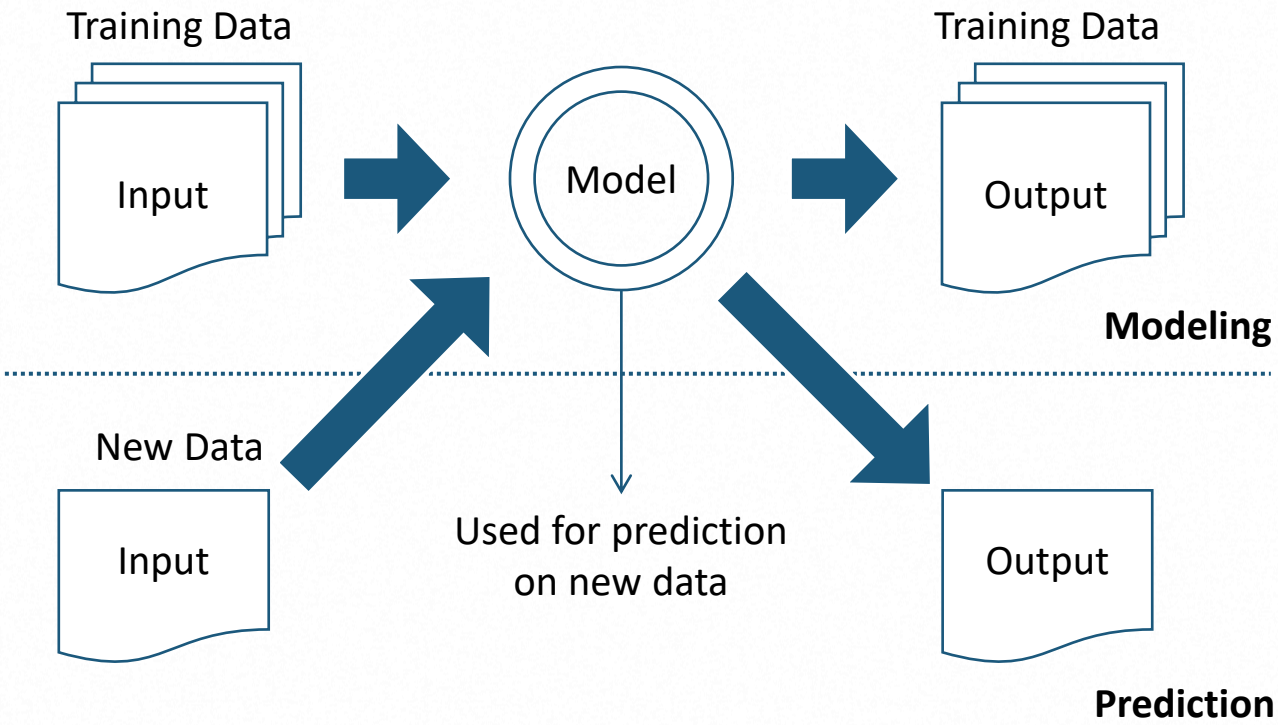
Supervised Learning vs. Unsupervised Learning

Criteria – whether or not there is feedback available to the learning system	
Learns from gold standard (a.k.a. training data, example data, labeled data, etc.)	Learns with no gold standard
Based on example inputs (*X*) and their outputs (*y*), aims to learn a general rule that maps new inputs (*X_new*) to their best possible outputs (*y_new*)	Learns on its own to find structures, or patterns, inherent in its inputs (*X*)
E.g., based on the customer profile data in a bank, build a model that predicts whether a new customer will leave within a year or not	E.g., given the customers in a bank, group them into several clusters of customers who share a similar profile
Regression, classification	Clustering, dimensionality reduction
Easier and more straightforward to evaluate	Harder to evaluate

prediction **pattern**

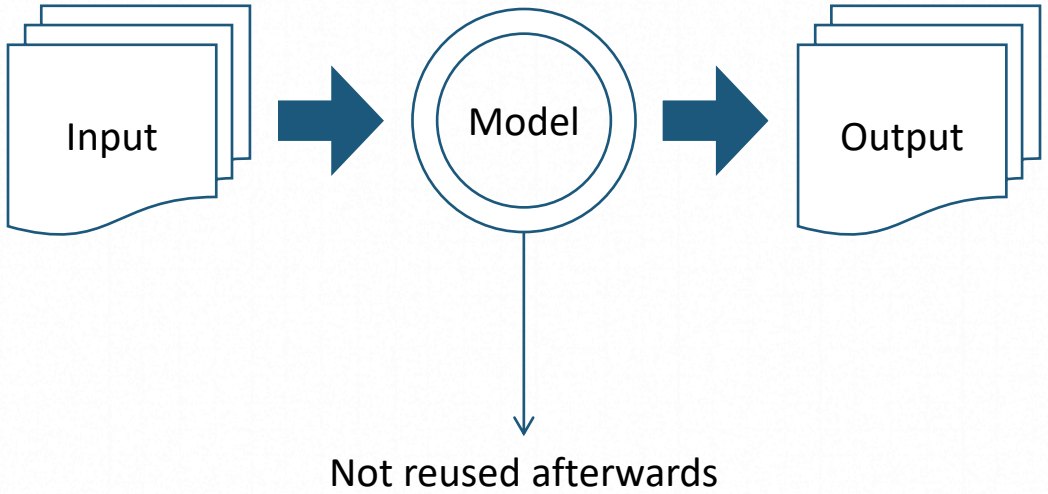
Supervised vs. Unsupervised Learning

Supervised Learning



vs.

Unsupervised Learning



Regression vs. Classification

Regression vs. Classification

Both supervised learning	
Criteria – whether or not there is continuity between possible outcomes	
Aims to predict a **continuous number**	Aims to predict a **class label**, which is a choice from a predefined list of possibilities
E.g., predicting a person's annual income from their education, their age, where they live, etc.	• Binary classification: only two classes (e.g., yes/no, negative/positive, survive/die, spam/nonspam) • Multiclass classification: more than two classes (e.g., weather as sunny, cloudy, rainy, or snowy)
k-Nearest Neighbors (k-NN), Linear Regression	k-Nearest Neighbors (k-NN), Logistic Regression, Support Vector Machines (SVMs), Naïve Bayes Classifiers, Decision Trees, Neural Networks

Classification vs. Clustering

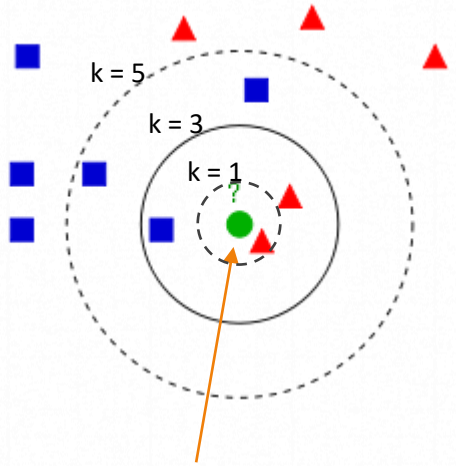
Classification vs. Clustering

Both aim to divide the data into meaningful segments	
Criteria – whether or not there are predefined classes	
Aims to classify the data into one of the **predefined categorical classes**	Aims to map the data into one of several clusters of **similar data items**
E.g., weather as sunny, cloudy, rainy, or snowy	E.g., clustering of similar news articles in a large news article data collection
Supervised learning → training sample provided	Unsupervised learning → no training sample
k-Nearest Neighbors (k-NN), Logistic Regression, Support Vector Machines (SVMs), Naïve Bayes Classifiers, Decision Trees, Neural Networks	k-Means Clustering, Agglomerative Clustering, DBSCAN, Topic Modeling

Supervised Learning – k-Nearest Neighbors (k-NNs)

- One of the simplest machine learning algorithms
- How it works
  - First, just stores the training data
  - For a new data point, finds the points in the training set that are **closest, or nearest,** to the new point
  - *k* = the number of the closest neighbors to consider
  - Makes a prediction using the majority class among these *k* nearest neighbors
- Used for both regression and classification
- Strengths
  - Very easy to understand
  - Often gives reasonable performance without a lot of adjustments
  - A good baseline method to try before considering advanced techniques
- Weaknesses
  - Slow in prediction for large training datasets
  - Does not perform well on datasets with many features (hundreds or more) or sparse datasets
  - Not often used in practice

Classification into Two Classes



Supervised Learning – Linear Regression

- Makes a prediction about a **continuous number** using a **linear function** of the input features
- Finds the parameters w and b that minimize the error between predictions ($\hat{y}$) and the true values (y)

Model → $\hat{y} = w[0] * x[0] + w[1] * x[1] + \cdots + w[p] * x[p] + b$

income, education, age

$where\ \hat{y}$: $target, or\ the\ prediction\ the\ model\ makes$

$x[0]\ to\ x[p]$: $features$

$w\ and\ b$: $parameters\ to\ be\ learned$

Training Dataset

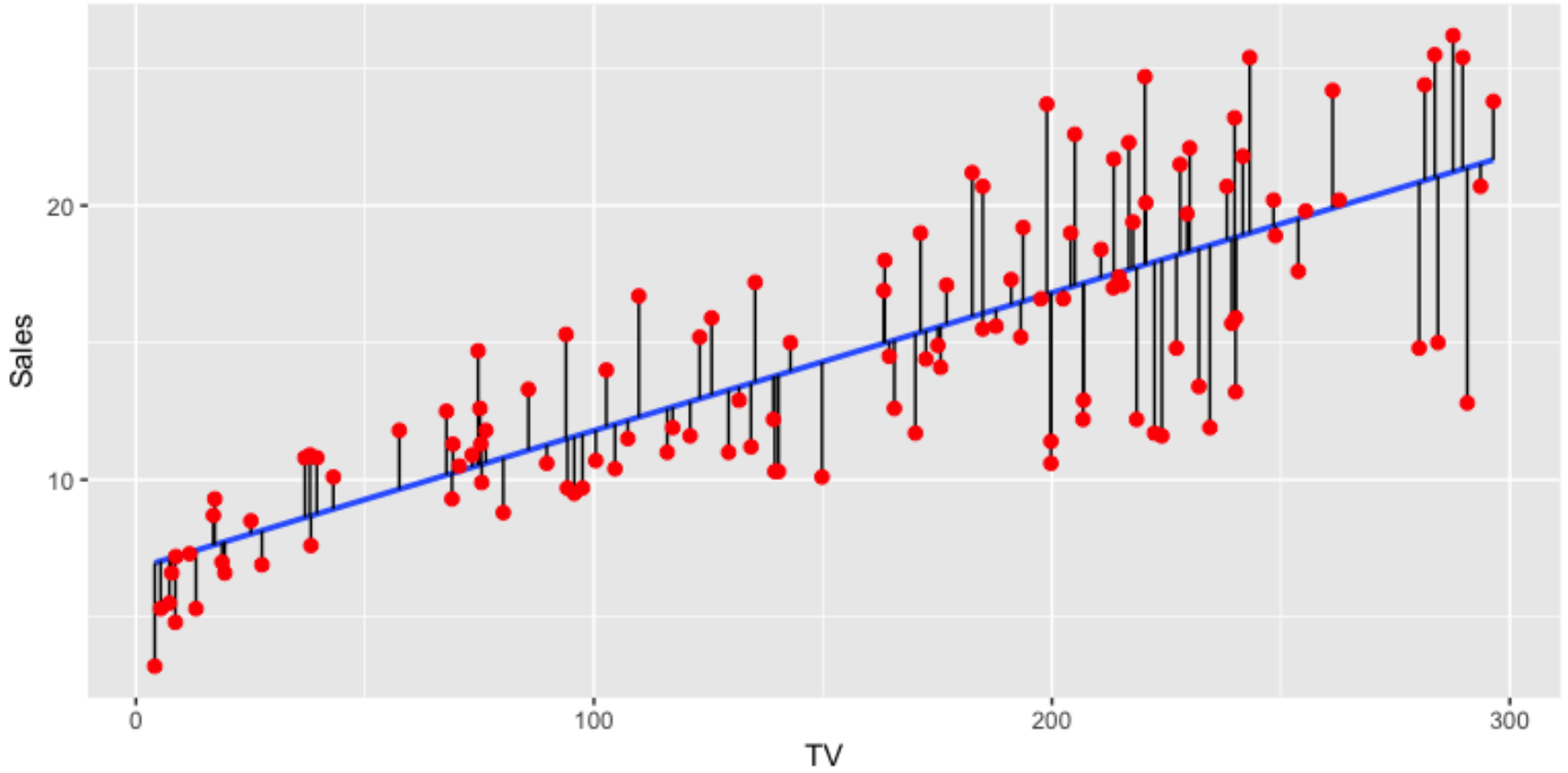
1st example → $y_0 = w[0] * x_0[0] + w[1] * x_0[1] + \cdots + w[p] * x_0[p] + b$

2nd example → $y_1 = w[0] * x_1[0] + w[1] * x_1[1] + \cdots + w[p] * x_1[p] + b$

⋮

Supervised Learning – Linear Regression

Find a strict line that minimizes the sum of squared errors between predictions ($\hat{y}$) and the true values (y)



Supervised Learning – Logistic Regression

- Makes a prediction about a **class label** using a **linear function** of the input features
- Finds the parameters w and b that minimizes the error between predictions ($\hat{y}$) and the true values (y)

Model → $\hat{y} = g(w[0] * x[0] + w[1] * x[1] + \cdots + w[p] * x[p] + b)$

spam/nonspam — length of the title — contains a URL

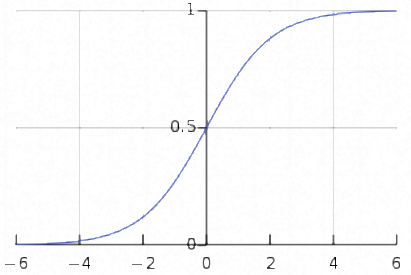
$where\ \hat{y}: the\ prediction\ the\ model\ makes$

$x[0]\ to\ x[p]: features$

$w\ and\ b: parameters\ to\ be\ learned$

If g is larger than 0.5, we predict the class as +1; if smaller than 0.5, we predict the class as -1

Binary classification



$g = sigmoid$

Training Dataset

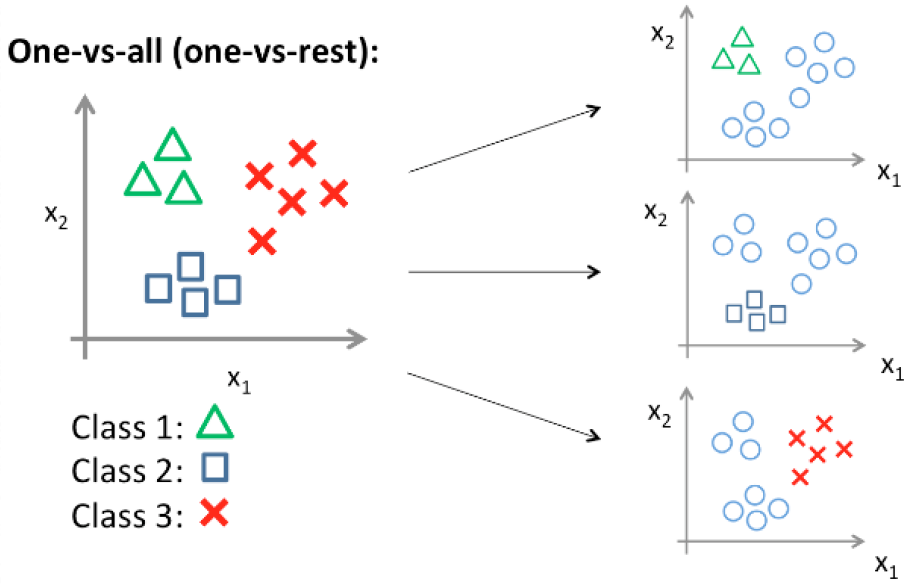
1st example → $y_0 = g(w[0] * x_0[0] + w[1] * x_0[1] + \cdots + w[p] * x_0[p] + b)$

2nd example → $y_1 = g(w[0] * x_1[0] + w[1] * x_1[1] + \cdots + w[p] * x_1[p] + b)$

⋮

Supervised Learning – Logistic Regression

- Multiclass classification
  - E.g., weather prediction as sunny, cloudy, rainy, or snowy
  - ***One-vs.-all*** (*one-vs.-rest*) approach
    - A multiclass classification problem with *n* classes can be decomposed into *n* binary classification problems
    - A binary model is learned for each class that separates that class from all other classes, resulting in as many binary models as there are classes (= *n*)
    - To make a prediction, all binary classifiers are run on a new point, and the classifier with the highest score on its single class wins, and this class label is returned as the prediction



Supervised Learning – Linear Models

- Strengths
  - Fast to train and to predict
  - Scale to very large datasets and work well with sparse data
  - Relatively easy to understand how a prediction is made using linear functions
  - Often perform well when the number of features is large compared to the number of training samples

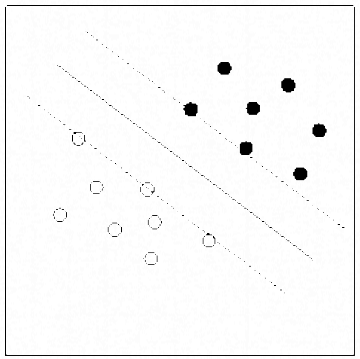
- Weaknesses
  - Based on an assumption that the target variable can be predicted by a linear combination of feature variables, which may be too weak to apply to real world problems
  - Other models may yield better generalization performance in lower-dimensional spaces

Supervised Learning – Support Vector Machines (SVMs)

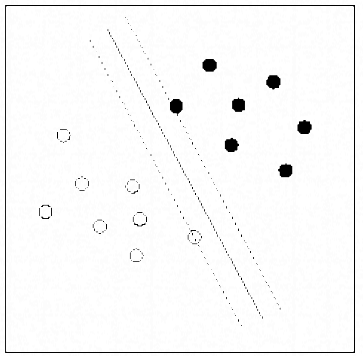
- Based on the **large margin** intuition
  - Find the maximum-margin hyperplane that represents the largest separation, or margin, between two classes
- Typically, only a subset of the training points matter for defining the decision boundary: the ones that lie on the border between the classes → called **support vectors**
- To make a prediction for a new data point,
  - The distance to each of the support vectors is measured
  - A classification decision is made based on the distances to the support vector and the weights of the support vector that were learned during training
- The distance between data points can be measured by Gaussian kernel:

$$k_{rbf}(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2)$$

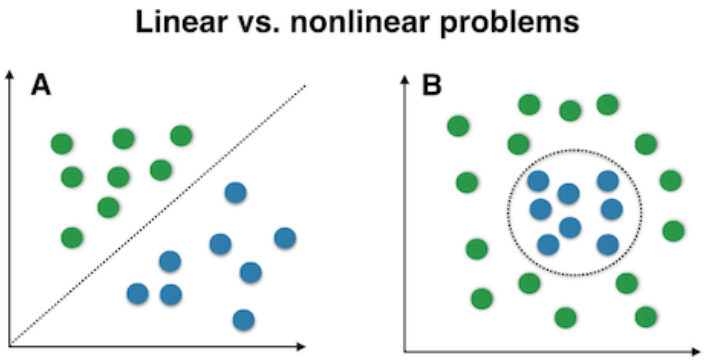
Controls the width of the Gaussian kernel Euclidean distance of two data points



(a) Larger margin



(b) Smaller margin



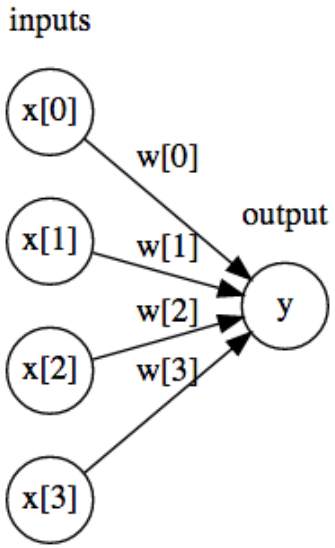
Supervised Learning – Support Vector Machines (SVMs)

- Strengths
  - Perform very well on a variety of datasets
  - Allow for complex decision boundaries, even if the data has only a few features
  - Work well on low-dimensional data with few features and high-dimensional data with many features

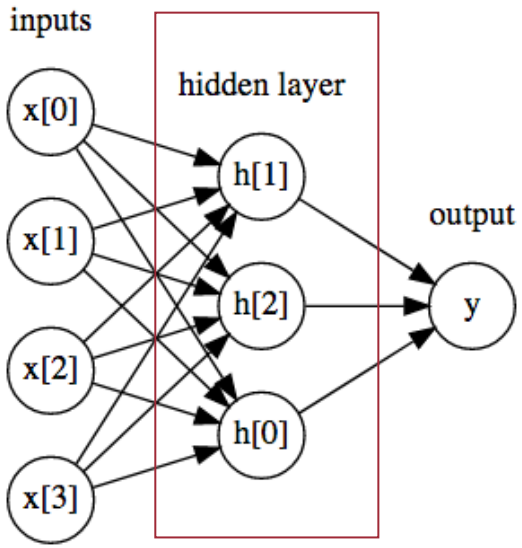
- Weaknesses
  - Very sensitive to the scaling of the data and the settings of the parameters
    - Do not scale very well with the number of training samples in terms of runtime and memory usage
    - Require careful preprocessing of the data and tuning of the parameters
  - Hard to understand why a particular decision was made

Supervised Learning – Neural Networks

- A.k.a. artificial neural networks (ANNs) or multilayer perceptrons (MLPs)
- Inspired by the biological neural networks that constitute animal brains
- Generalizations of linear models that perform **multiple stages of processing** to come to a decision



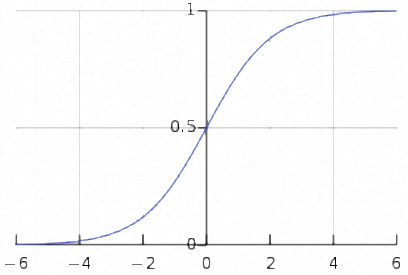
Logistic regression



MLPs with a single hidden layer

$$h[0] = g(w[0,0] * x[0] + w[1,0] * x[1] + w[2,0] * x[2] + w[3,0] * x[3])$$
$$h[1] = g(w[0,1] * x[0] + w[1,1] * x[1] + w[2,1] * x[2] + w[3,1] * x[3])$$
$$h[2] = g(w[0,2] * x[0] + w[1,2] * x[1] + w[2,2] * x[2] + w[3,2] * x[3])$$
$$y = g(v[0] * h[0] + v[1] * h[1] + v[2] * h[2])$$

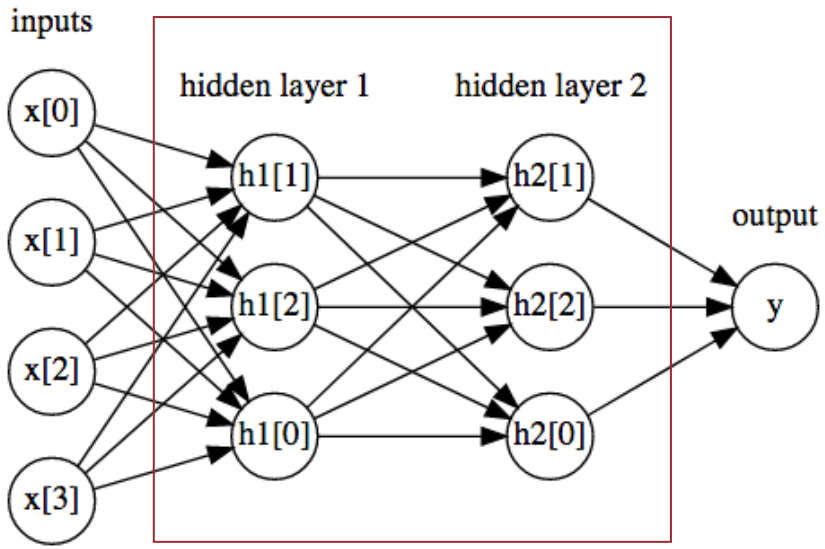
activation function



$g = sigmoid$

Supervised Learning – Neural Networks

- You can control the complexity of neural networks with
  - the number of hidden layers
  - the number of hidden units in each hidden layer



MLPs with two hidden layers

- Random initialization of weights
  - All initial weights are set randomly before learning is started → called random seeds
  - This random initialization can affect the model that is learned, particularly for small networks
  - One effective strategy for random initialization is to randomly select values uniformly in the range $[-\epsilon_{init}, \varepsilon_{init}]$, e.g., $[-0.12, 0.12]$, to ensure the weights are kept small and makes the learning more efficient

Supervised Learning – Neural Networks

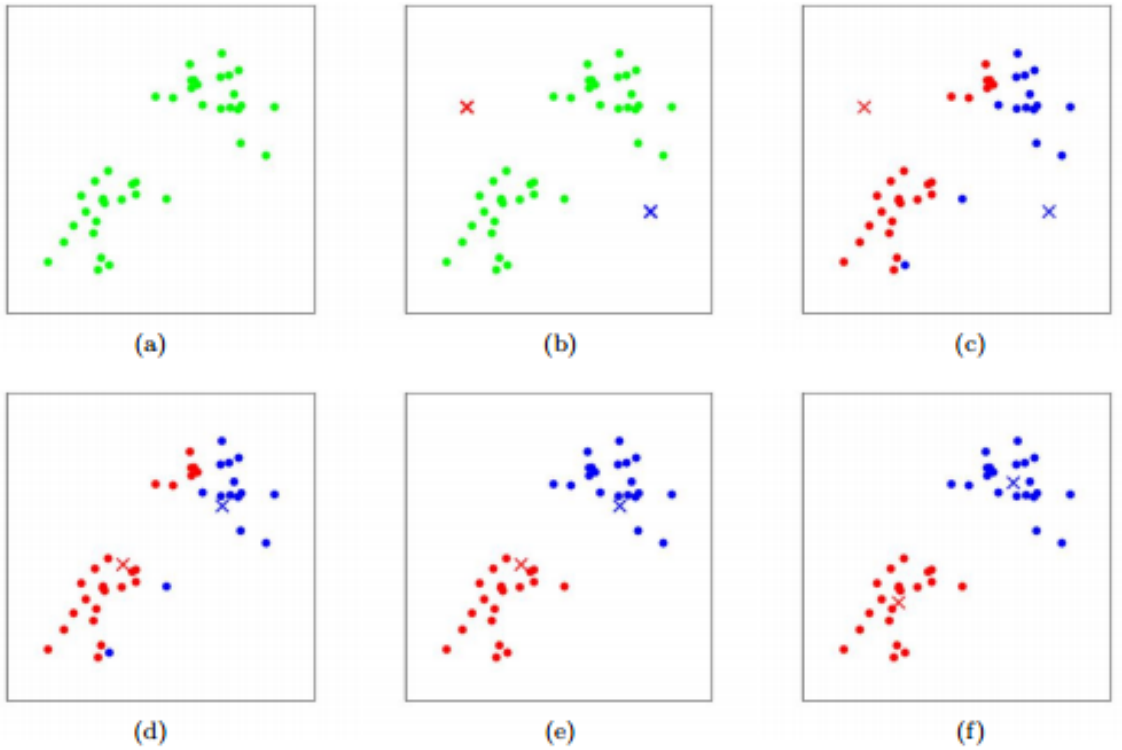
- Strengths
  - Able to capture information contained in large amounts of data and build incredibly complex models → the basis for deep learning
  - Often outperform other machine learning algorithms, given enough computation time, data and careful tuning of the parameters
  - Work best with homogeneous data, where all the features have similar meanings

- Weaknesses
  - Often takes long time to train
  - Require careful preprocessing of the data and tuning of parameters
  - Do not work well with heterogeneous data with very different kinds of features

Unsupervised Learning – k-Means Clustering

- One of the simplest and most commonly used clustering algorithms
- Finds cluster centers, or **centroids**, that are representative of certain regions of the data
- How it works
  - Given the number of clusters (*k*), randomly choose *k* centroids
  - Iterate between two steps:
    - Assign each data point to the closest centroid
    - Update each centroid as the mean of the data points that are assigned to it
  - Finish when the assignment of instances to clusters no longer changes

k = 2



Unsupervised Learning – k-Means Clustering

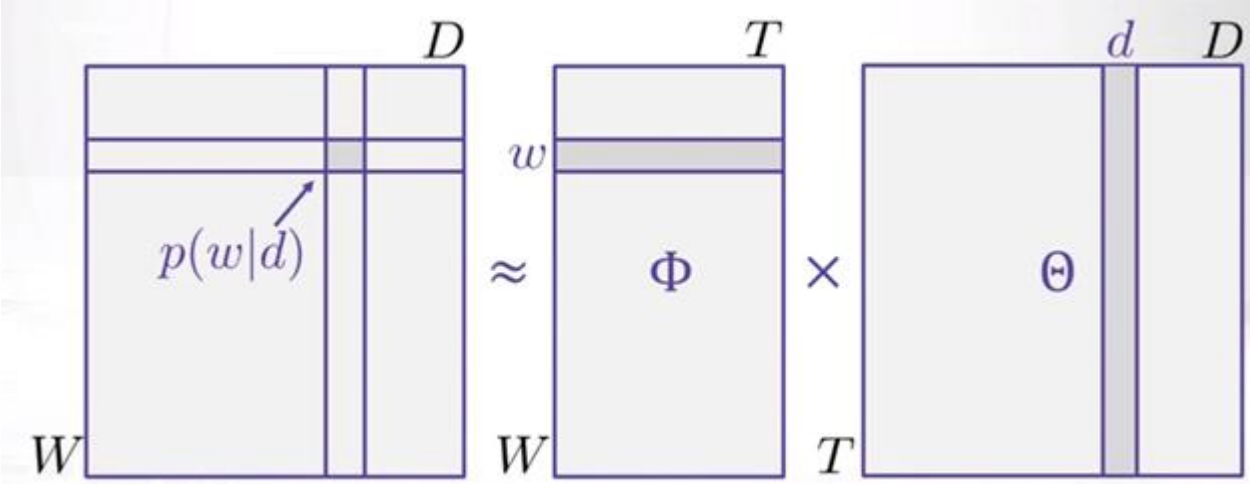
- Strengths
  - Relatively easy to understand and implement
  - Runs relatively quickly
  - Scales easily to large datasets

- Weaknesses
  - The outcome of the algorithm depends on the random initialization of centroids
  - Can only capture clusters of relatively simple shapes
  - Assumes that all directions are equally important for each cluster
  - Requires **the number of clusters (*k*)** you are looking for (which might not be known in a real-world application)
    - The number of clusters depends on the intended level of specialization

      Set *k* to a small number → fewer clusters of more data points → a wider view on the clustered data
      Set *k* to a large number → more clusters of fewer data points → a narrower view

Unsupervised Learning – Topic Modeling

- Topic modeling is a type of statistical modeling for **discovering the abstract, or latent, topics** that occur in a collection of documents
- Latent Dirichlet Allocation (LDA) is an example of topic model and is used to classify text in a document to a particular topic
- LDA works by making a key assumption: the way a document was generated was by picking a set of topics and for each topic picking a set of words
  → LDA reverse engineers this process to find topics



Word-doc matrix Word-topic matrix Topic-doc matrix

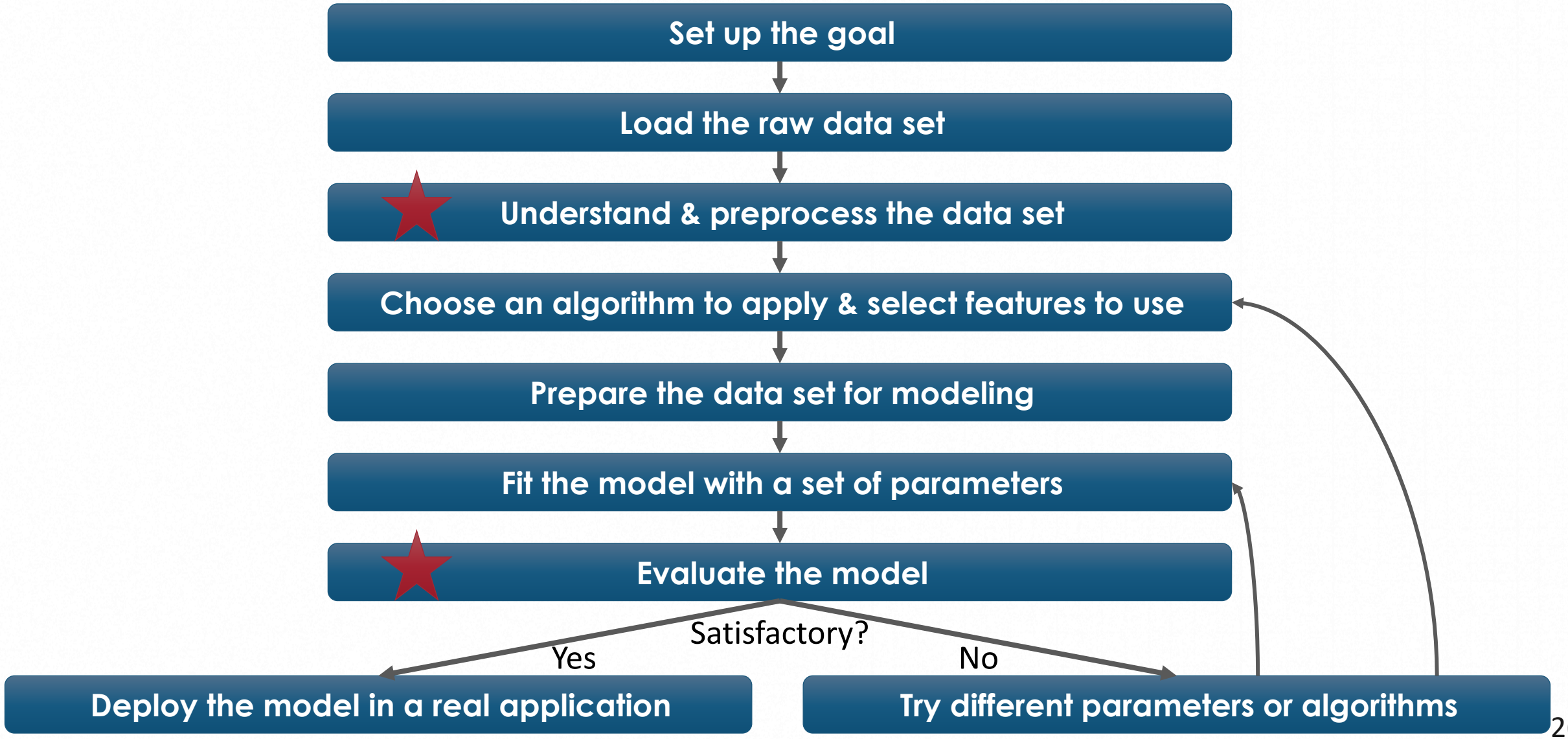
Matrix factorization or singular value decomposition (SVD), where it decomposes the probability distribution matrix of word in document into two matrices consisting of distribution of topic in a document and distribution of words in a topic

Topic modeling using Latent Dirichlet Allocation(LDA) and Gibbs Sampling explained!

Unsupervised Learning on Text Data

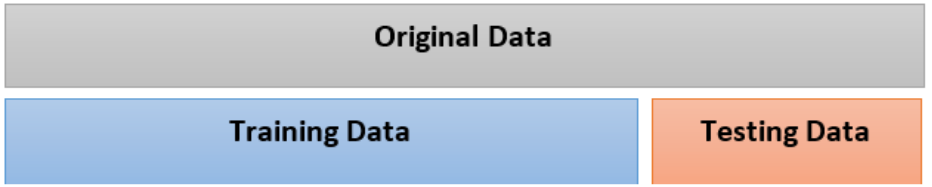
Document Clustering	vs.	Topic Modeling
Aims to find **clusters** of similar documents		Aims to find **topics** that occur across documents
Final outcome is the documents each labeled with a cluster number		Final outcome is the topics each with its contributing words and their scores
k-Means Clustering		LDA Topic Modeling
The number of clusters needs to be defined in advance		The number of topics needs to be defined in advance

Machine Learning Process

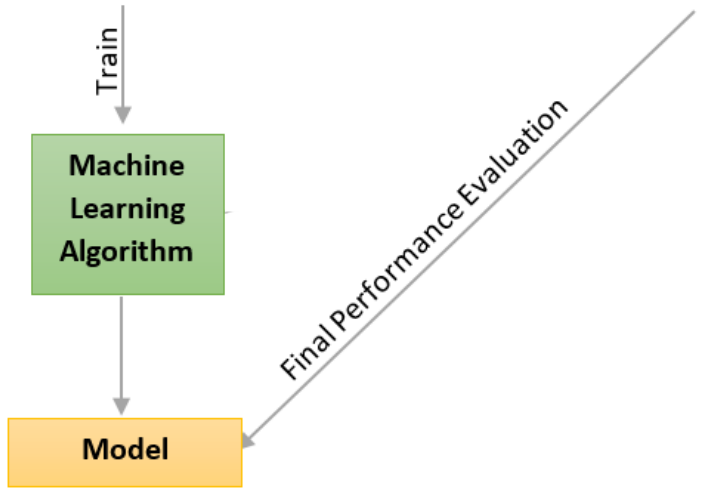


Evaluation of Supervised Models

- Randomly split the data into training and test sets to measure how well the model generalizes to new, previously unseen data
  - Randomly split the dataset into a **training set** (75%) and a **test set** (25%)
  - Build a model on the training set
  - Evaluate the model on the test set



- We are NOT interested in how well the model fits the training set, BUT rather in how well it can make predictions for the test set that was not observed during training



Evaluation of Supervised Models

Accuracy score on **training** data	Accuracy score on **test** data	
Low	Low	Underfitting
Low	High	Good, but rare
High	Low	Overfitting
High	High	Excellent

Machine Learning in Text Analytics

When applying machine learning to text analytics, words are used as features and documents as individual records