BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Introduction to Classification

Srikanth Gunturu

In this segment

Introduction to Classification

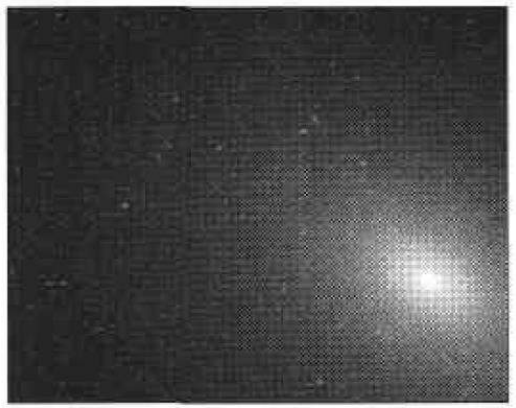
- What is a Classification ?
- Classification – Process and a few terms
- Classification Model – Uses & General Approach

Introduction to Classification

What is Classification

- The task of assigning objects to one of several predefined categories (classes)
- Real world applications of classification
  - Detecting spam email messages based upon the message header and content
  - Categorizing tumours as malignant or benign based upon the results of MRI scan
  - Classifying galaxies based upon their shapes

(a) A spiral galaxy.

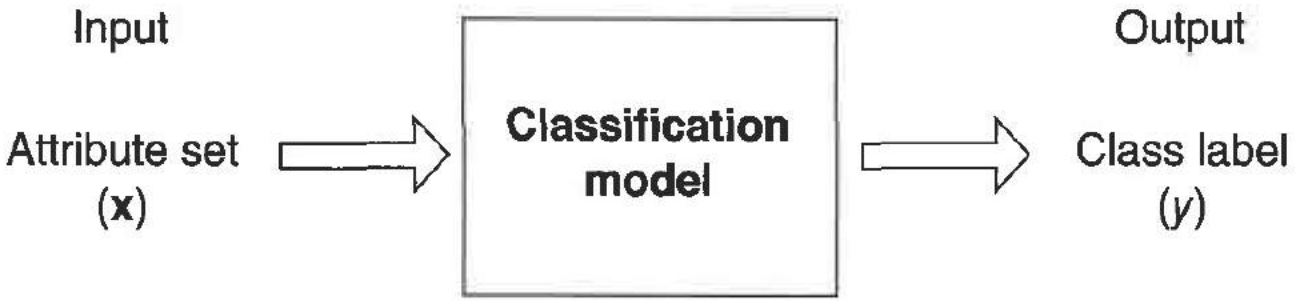


(b) An elliptical galaxy.

Introduction to Classification

Classification – Process and a few terms

- Classification is the task of learning a target function ***f*** that maps each attribute set ***x*** to one of the predefined class labels *y*



- Feature / Attribute – Input attribute that's used in classifying the output
- Feature set – Set of attributes that govern the output class
- Label – Output class that the object is assigned to

Introduction to Classification

Classification Model – Uses

- Descriptive - Serves as an explanatory tool to distinguish between objects of different classes

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
human	warm-blooded	hair	yes	no	no	yes	no	mammal
python	cold-blooded	scales	no	no	no	no	yes	reptile
salmon	cold-blooded	scales	no	yes	no	no	no	fish
whale	warm-blooded	hair	yes	yes	no	no	no	mammal
frog	cold-blooded	none	no	semi	no	yes	yes	amphibian
komodo dragon	cold-blooded	scales	no	no	no	yes	no	reptile
bat	warm-blooded	hair	yes	no	yes	yes	yes	mammal
pigeon	warm-blooded	feathers	no	no	yes	yes	no	bird
cat	warm-blooded	fur	yes	no	no	yes	no	mammal
leopard shark	cold-blooded	scales	yes	yes	no	no	no	fish
turtle	cold-blooded	scales	no	semi	no	yes	no	reptile
penguin	warm-blooded	feathers	no	semi	no	yes	no	bird
porcupine	warm-blooded	quills	yes	no	no	yes	yes	mammal
eel	cold-blooded	scales	no	yes	no	no	no	fish
salamander	cold-blooded	none	no	semi	no	yes	yes	amphibian

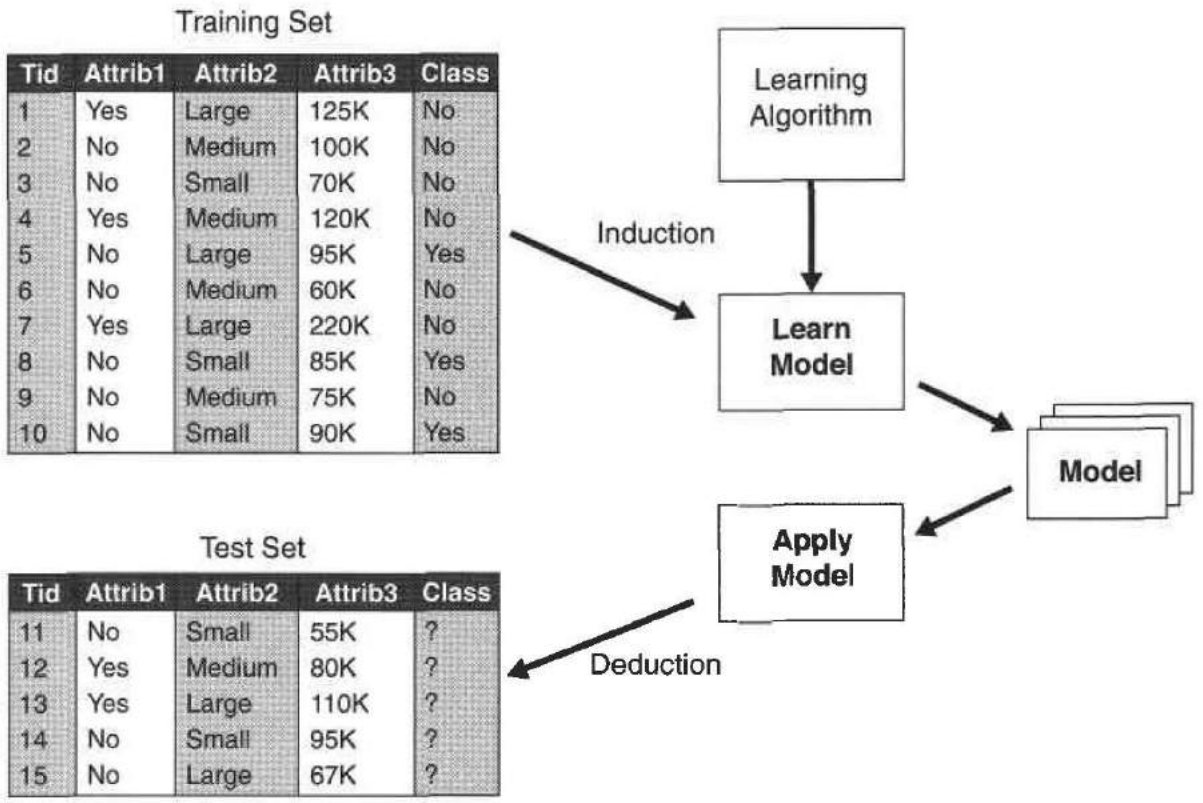
- Predictive – Used to predict the class label of unknown records

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
gila monster	cold-blooded	scales	no	no	no	yes	yes	?

Introduction to Classification

Classification Model – General Approach

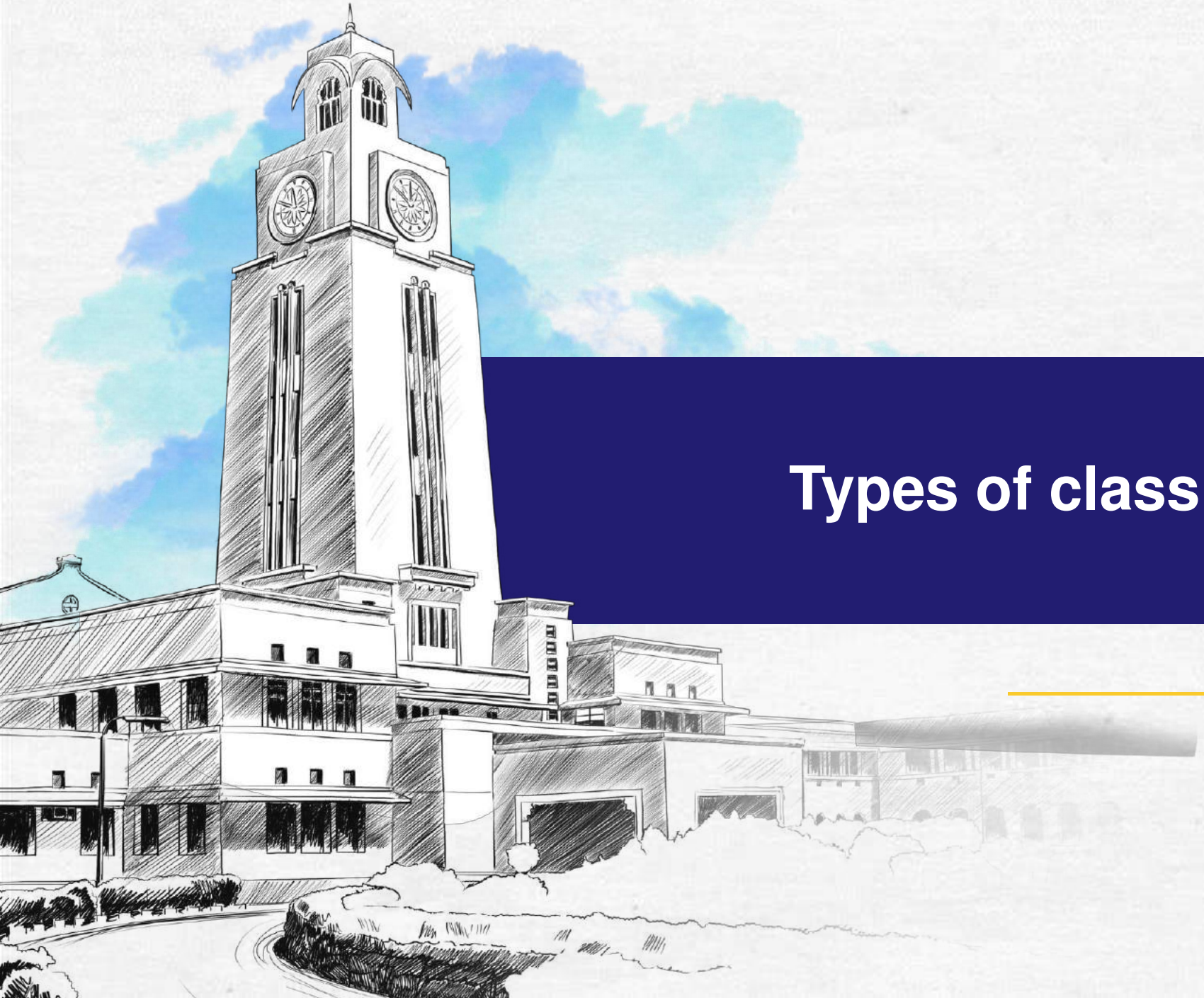
- Should fit the input data well
- Should correctly predict the class labels of records it has never seen before



Thank You!

In our next session:

Types of classification algorithms



BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Types of classification algorithms

Srikanth Gunturu

Guest Faculty
BITS, WILP

In this segment

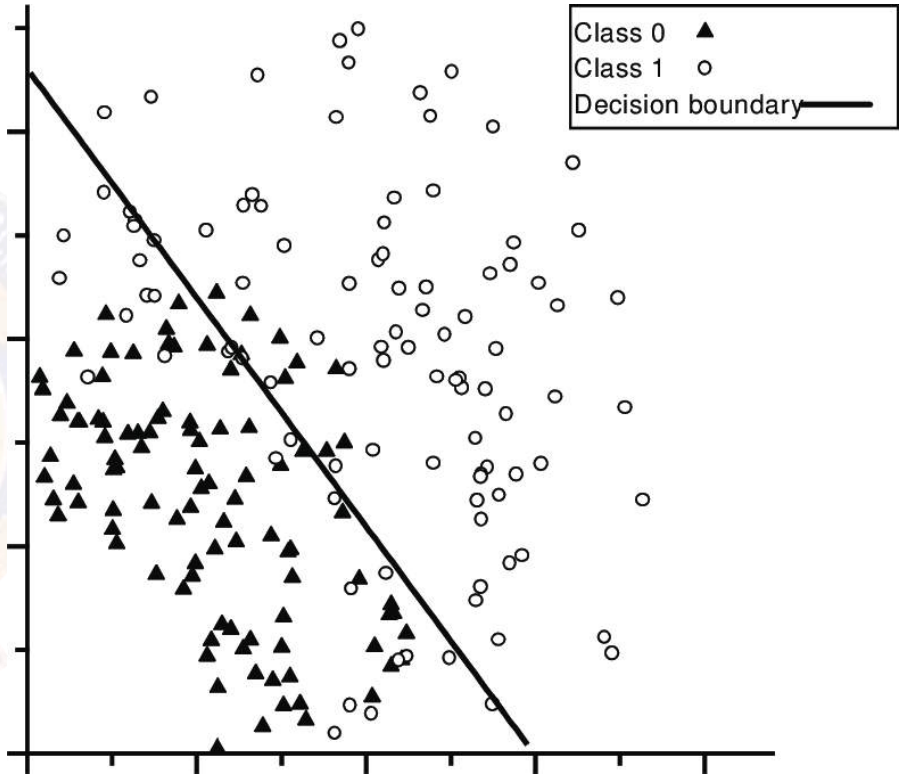
Types of classification algorithms

- Discriminant functions
- Probabilistic Generative models
- Probabilistic Discriminative models
- Tree based models

Types of classification algorithms

Discriminant functions - Overview

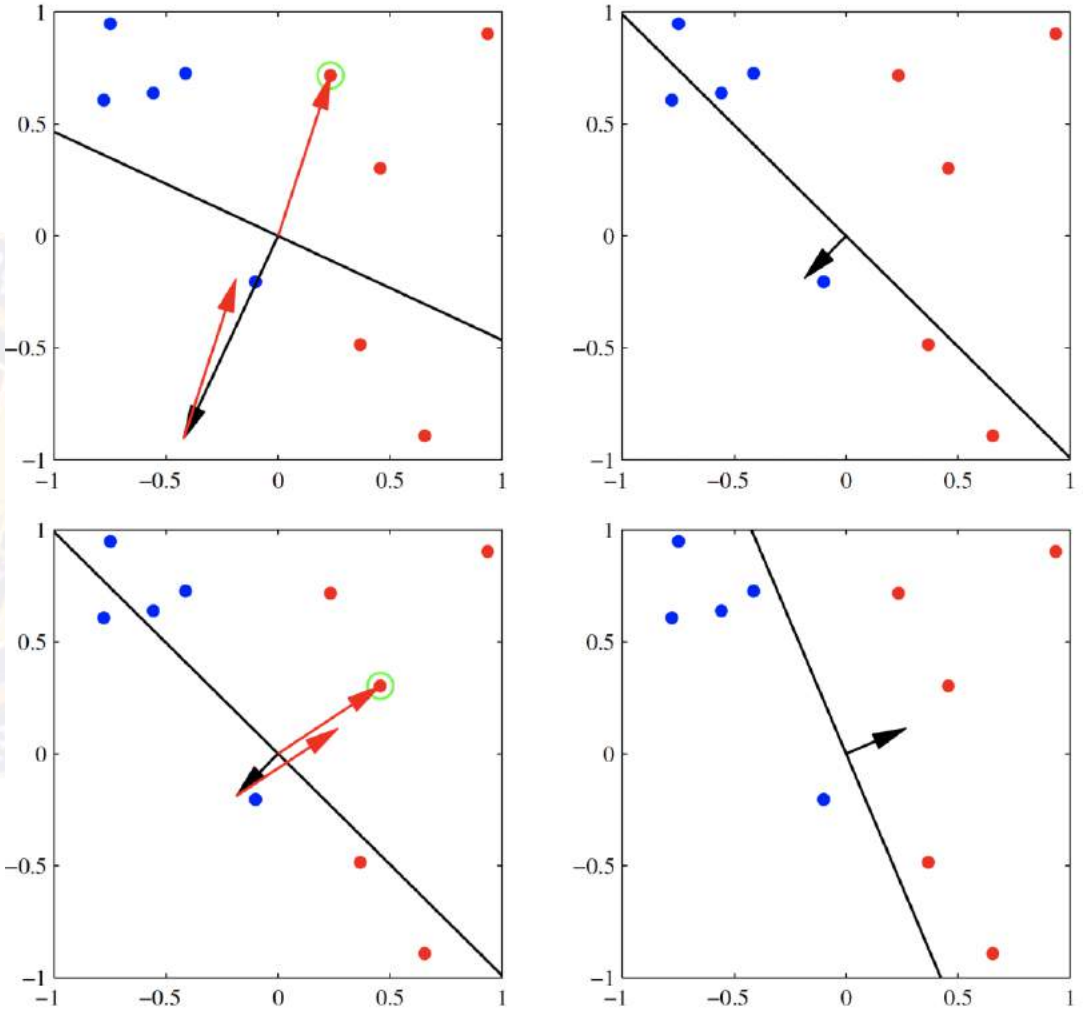
- Both inference and decision are performed together
- Discriminant is a function that maps input vector directly to a class label
- Linear discriminant is a linear function, whose decision surface is a hyperplane
- Examples:
  - Linear Discriminant
  - The Perceptron



Types of classification algorithms

Discriminant functions – The Perceptron

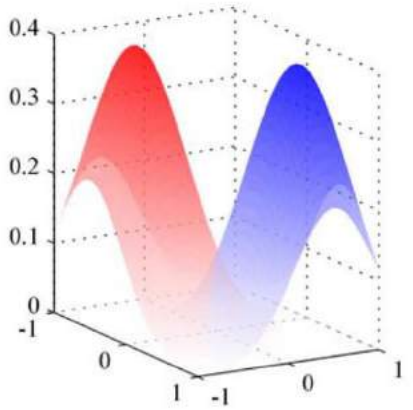
- Proposed by Frank Rosenblatt in 1962
- Corresponds to a two class model
- Associates zero error with correctly classified patterns
- Tries to minimise the criterion quantity for misclassified pattern
- Perceptron convergence (fig.)

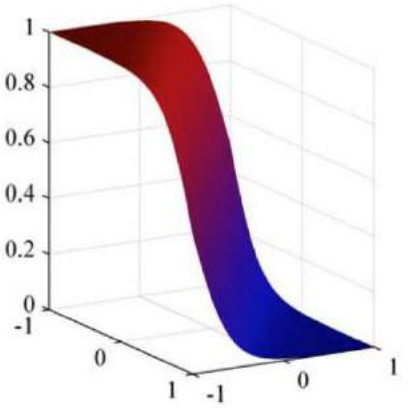


Types of classification algorithms

Probabilistic Generative Models - Overview

- Uses probability theory to generate joint distribution of input and output variables, P(X,C) and then use that distribution to predict the class for a given new input set
- Approach
  - Inference - Model joint distribution of input vector X and class C_k, using probability
  - Decision – Apply the model to given new set of input to determine probability of class C_k
- Two-class – probability $$p(\mathcal{C}_1|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)}$$
- Examples: Naïve Bayes, Gaussian Mixture Model etc.





Types of classification algorithms

Probabilistic Discriminative Models – Overview

- Uses conditional probability of the class variable C_k given the input vector X, i.e. $P(C_k|X)$
- Approach
  - Inference – Use conditional probability to model the probability of class variable given the input set, using training data
  - Decision – Use decision theory to assign each new input set to one of the classes
- Examples – Logistic Regression

Types of classification algorithms

Probabilistic Generative Vs Probabilistic Discriminative Models

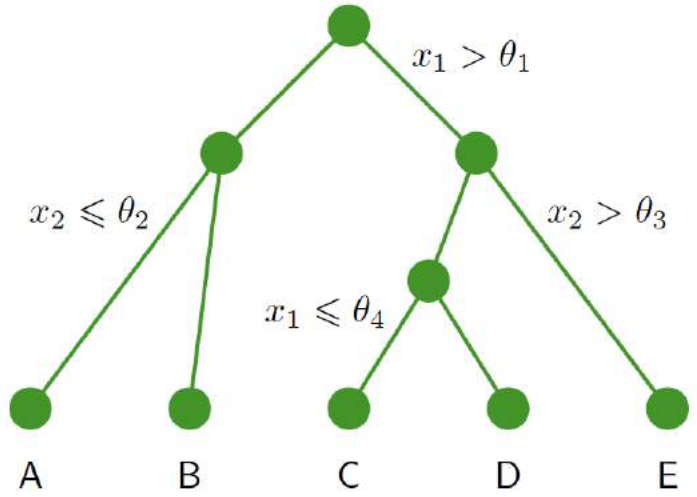
Generative Model	Discriminative Model
Derive joint probability distribution using training data	Derive conditional probability from training data
Requires considerably large training set to generate an decent model	Can work with moderate sized data as well, since the conditional probability is drawn from given input set directly
Models the actual distribution of each class	Models the decision boundary between classes
Use Bayesian theory to assign input to a class	Use maximum likelihood to assign an input to a class

Types of classification algorithms

Tree based Models - Overview

- These models separate input space into multiple partitions based on certain conditions and apply different modelling techniques independently on each sub space (ex: binary tree)

- For a tree with regions R and T number of leaf nodes, optimal prediction is $y_\tau = \frac{1}{N_\tau} \sum_{\mathbf{x}_n \in \mathcal{R}_\tau} t_n$
- Pruning is used for building optimal tree



Thank You!

In our next session:

Classification algorithms covered in this course

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Classification Algorithms in scope for this course

Srikanth Gunturu

Guest Faculty
BITS, WILP

In this segment

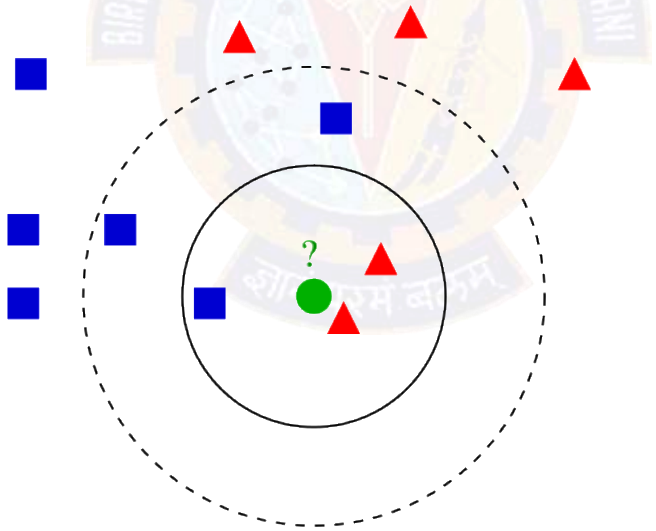
Classification Algorithms in scope for this course

- K-Nearest Neighbor
- Naïve Bayes
- Logistic Regression
- Decision Tree
- Support Vector Machines
- Ensemble Methods

Classification Algorithms in scope for this course

k-Nearest Neighbour Classifier (kNN)

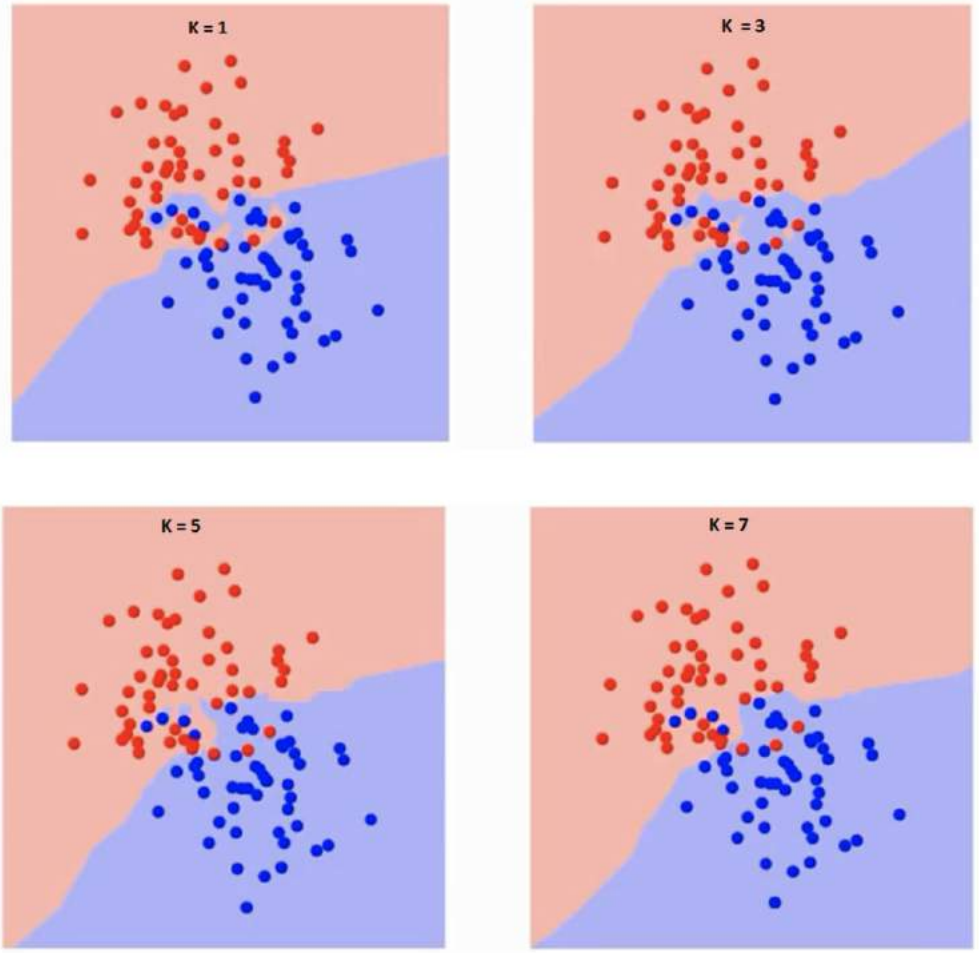
- Instance based classification
- An object is assigned to the class most common among its *k* nearest neighbours
- How to determine optimal k ?

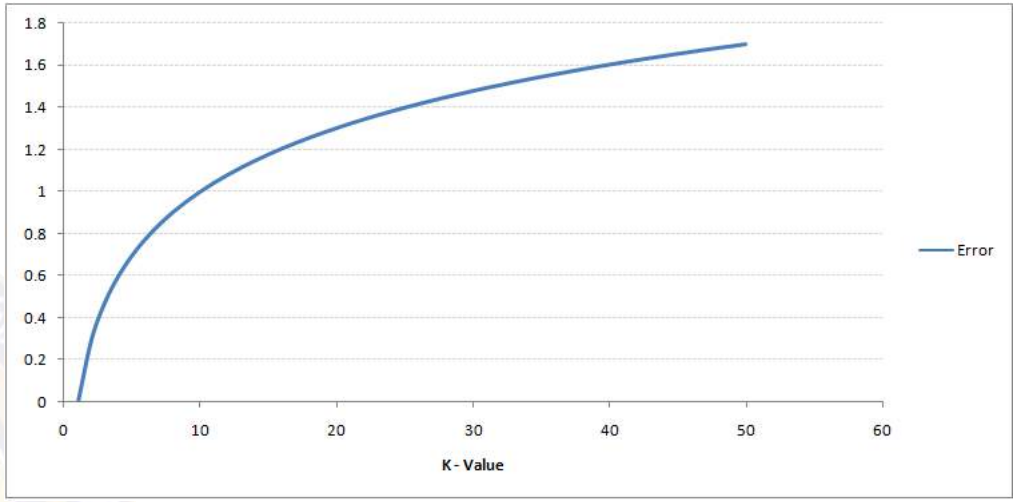


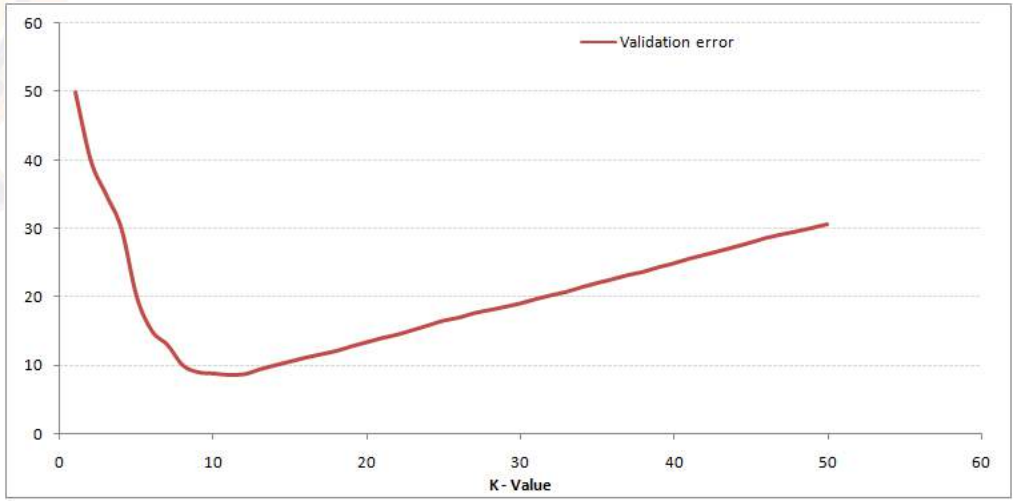
Classification Algorithms in scope for this course

k-Nearest Neighbour Classifier (kNN)

- As k increases boundaries get smoother, but..







Classification Algorithms in scope for this course

Naïve Bayes Classifier

- Bayes Theorem

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

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

$$P(y|x_1, ..., x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

- Naïve Bayes Classifier

$$y = argmax_y P(y)\prod_{i=1}^{n} P(x_i|y)$$

- *Gaussian Naïve Bayes – continuous data*

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

- Multinomial Naïve Bayes – feature vector representing frequency

$$p(\mathbf{x} \mid C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p_{ki}{}^{x_i}$$

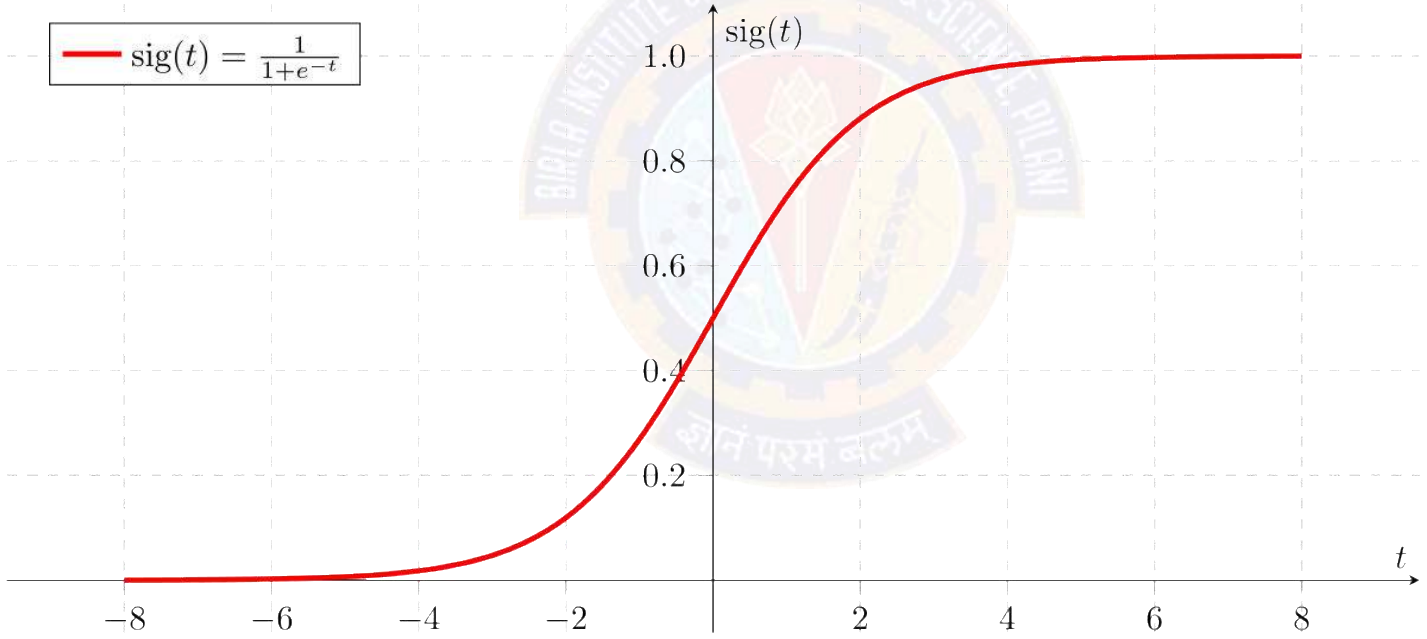
- Bernoulli Naïve Bayer – Binary valued features

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i}(1 - p_{ki})^{(1-x_i)}$$

Classification Algorithms in scope for this course

Logistic Regression

- Why “regression” for classification ?
- Logistic regression (logit) minimizes uncertainties, with differentiable decision function



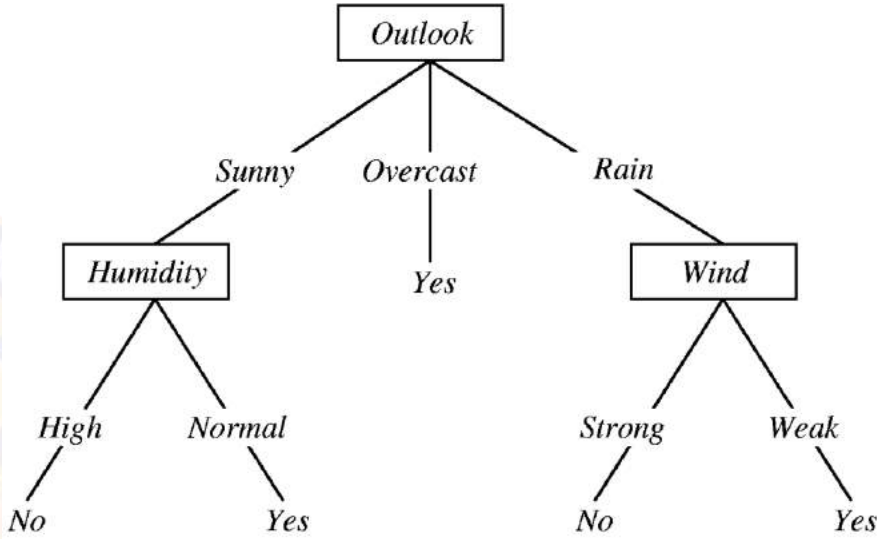
- Posterior probability – log of odds

$$log\left[\frac{P(Y|X)}{1-P(Y|X)}\right] = x_0 + x_1\beta_1 + x_2\beta_2 + ... + x_k\beta_k + \varepsilon = f(x)$$

Classification Algorithms in scope for this course

Decision Tree

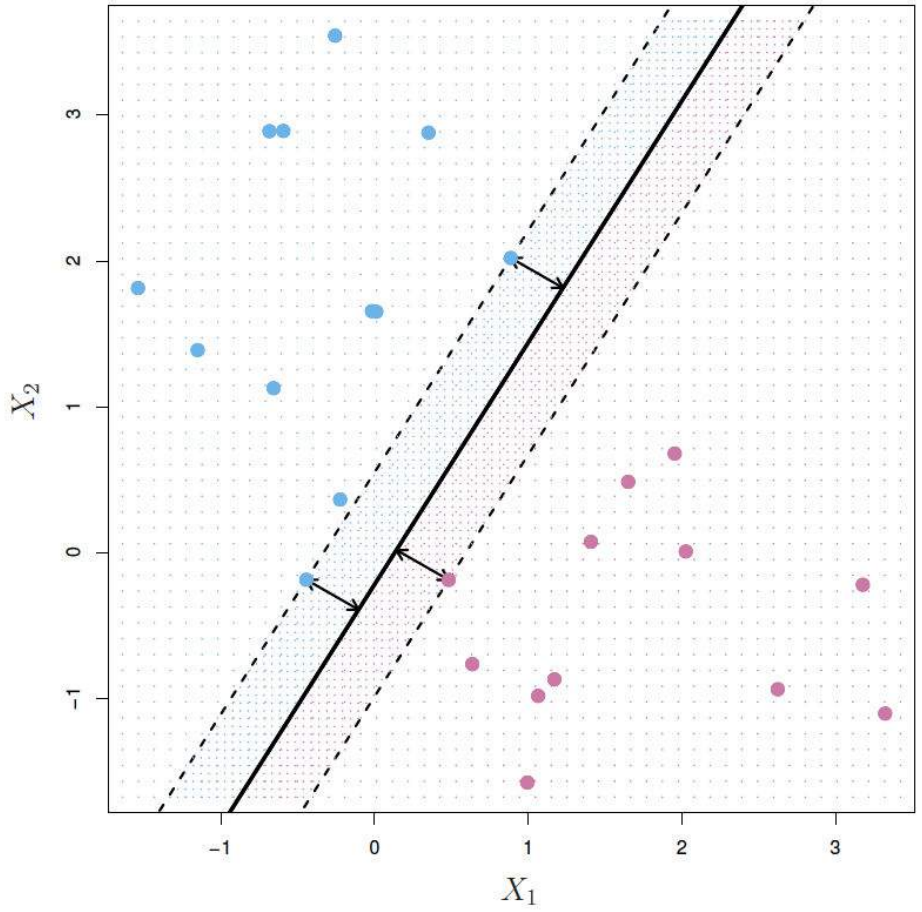
- Leaf represents class label
- Decisions are (internal) nodes/ branches
- Supervised learning for classification / regression
- Can handle categorical and continuous data
- Little preprocessing needed
- Prone to overfitting
  - Pruning
  - Early stop of tree building

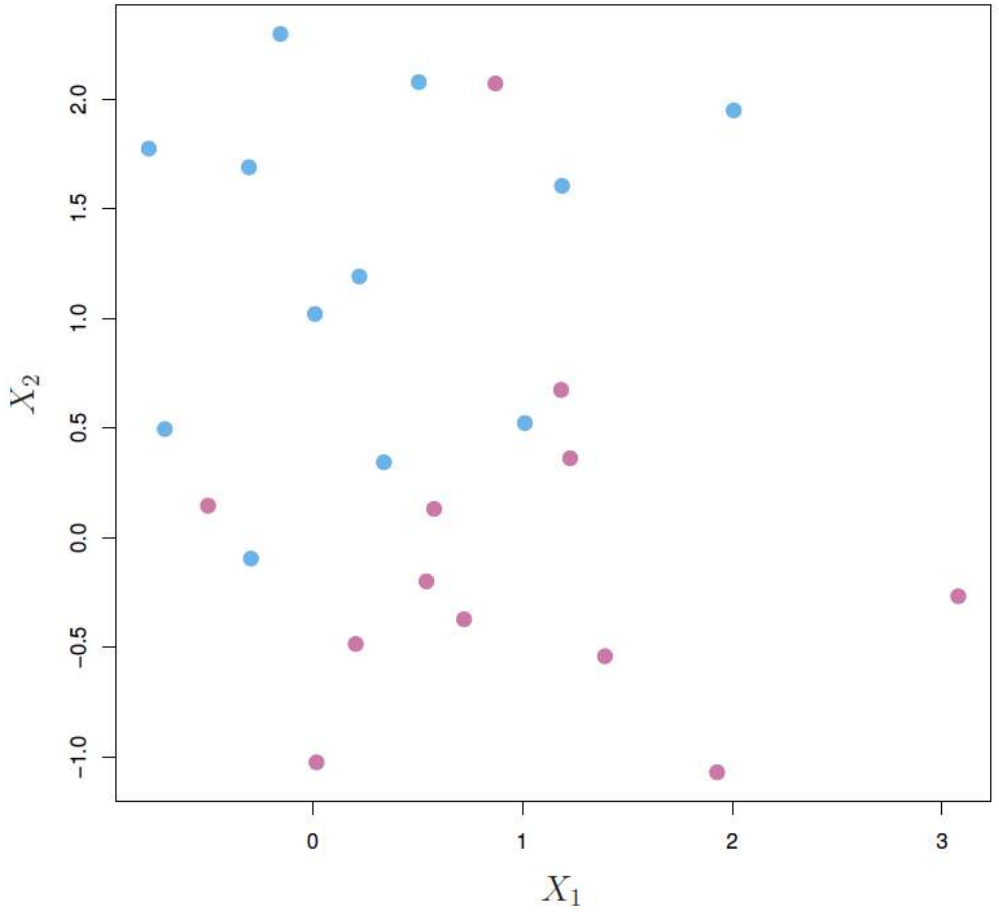


Classification Algorithms in scope for this course

Support Vector Machines (SVM)

- Maximum margin classifier – does it work always ?

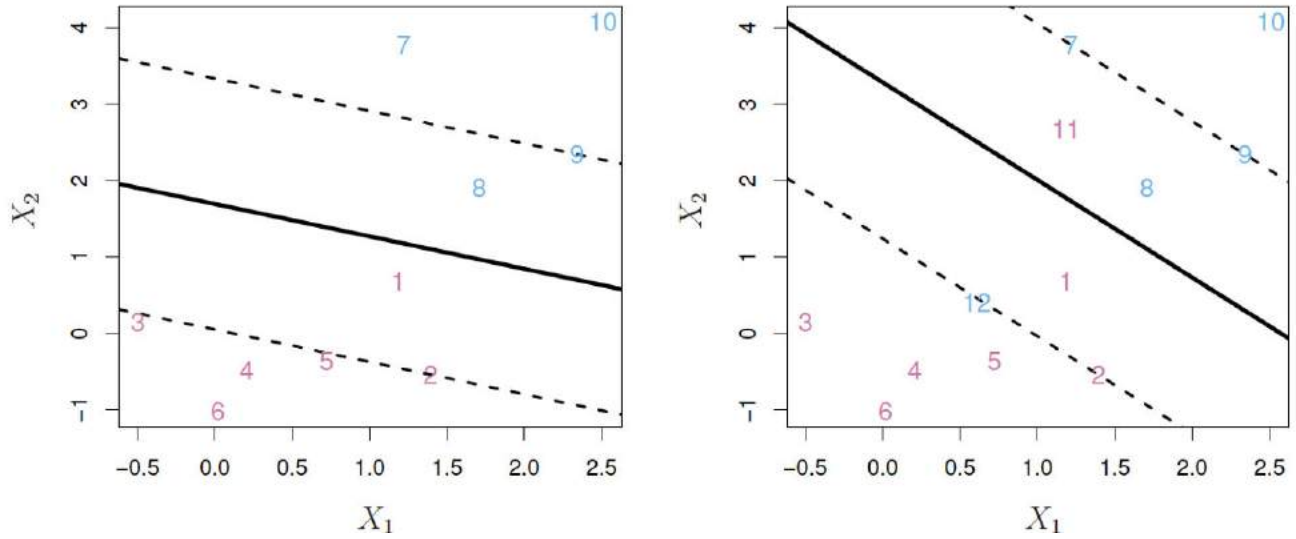


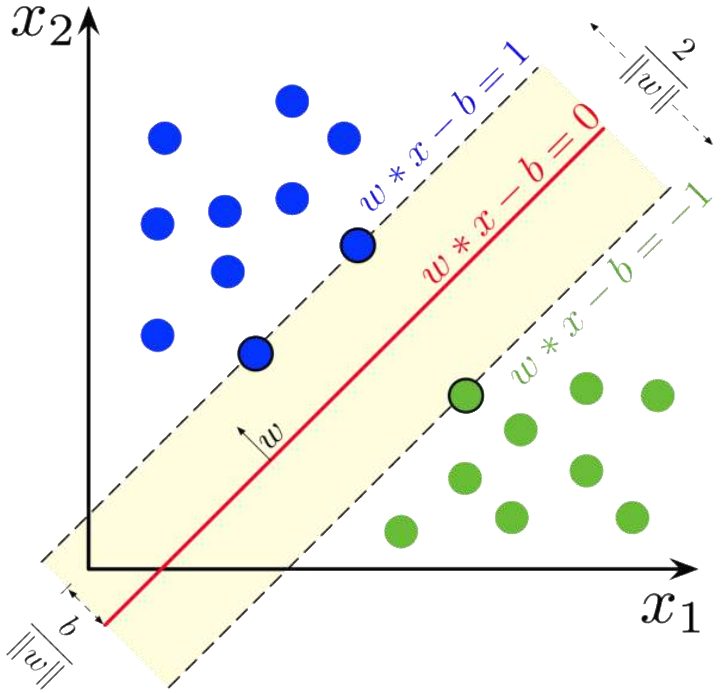


Classification Algorithms in scope for this course

Support Vector Machines (SVM) – Linear classifier

- Support Vector classifier – makes things better but not always
  - Hard-margin
  - Soft-margin

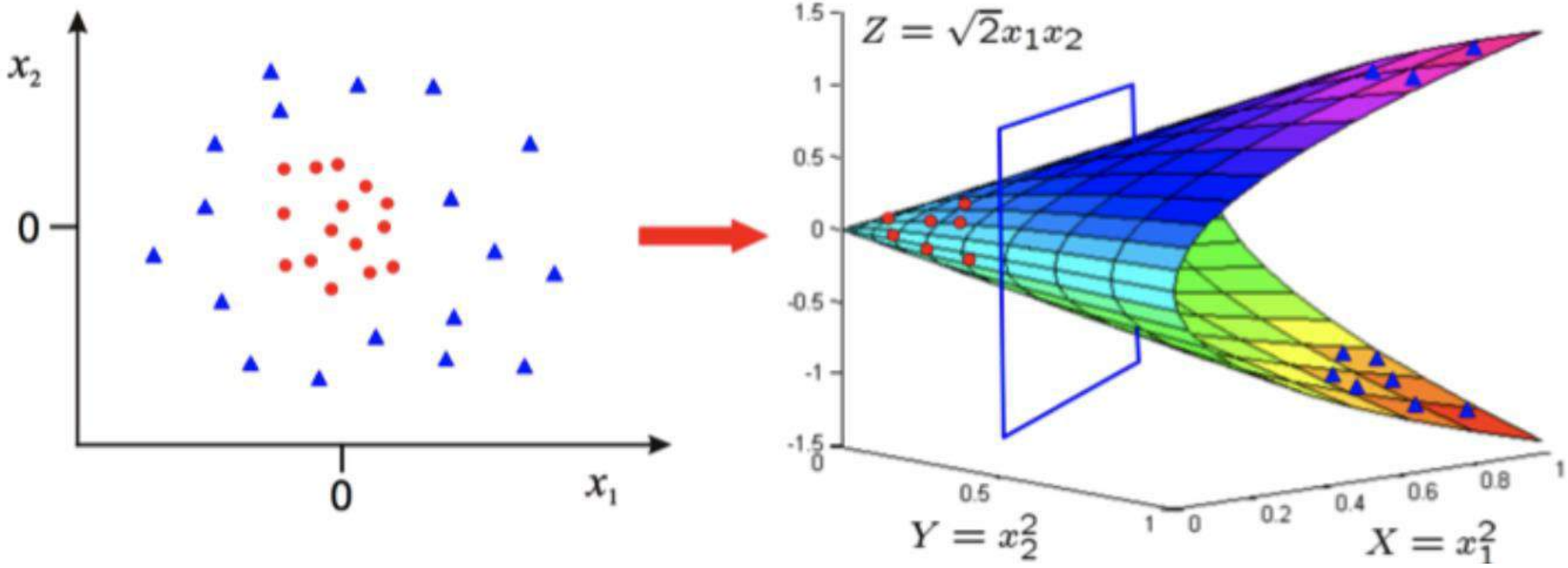




Classification Algorithms in scope for this course

Support Vector Machines (SVM) – Non-linear classifier (kernel trick)

- Transforms the feature space to allow fitment of a linear hyperplane
  - Polynomial
  - Radial basis function (rbf)
  - Hyperbolic Tangent (tanh)



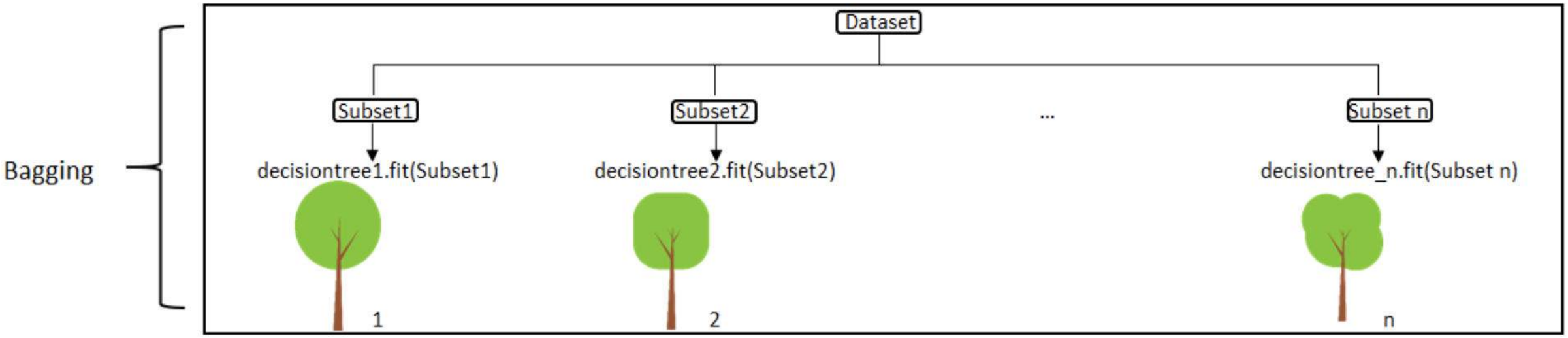
Classification

Ensemble Methods

- Bagging (Bootstrap AGGregating)
- Boosting
  - AdaBoost (Adaptive Boosting)
  - XGBoost (eXtreme Gradient Boosting)
- Random Forest

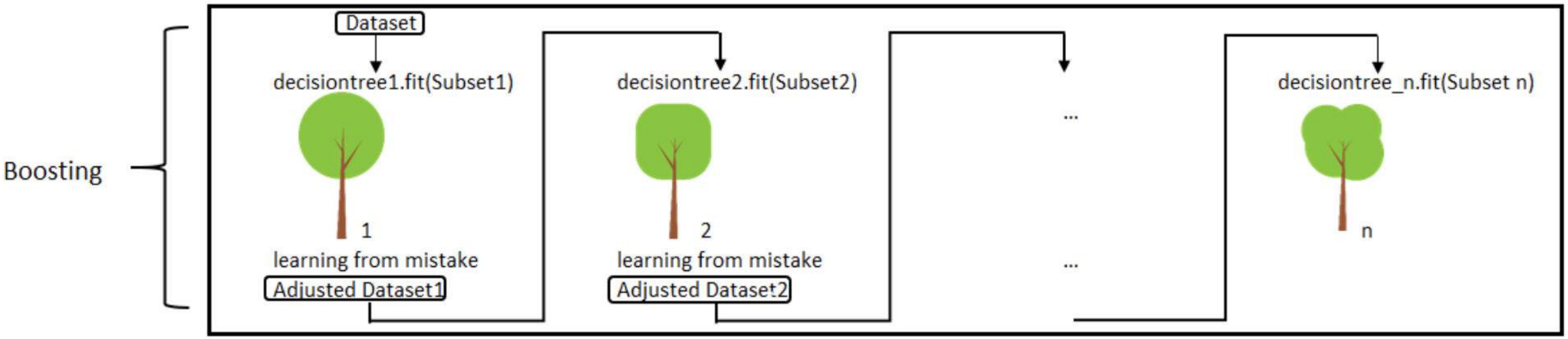
Classification

Bagging



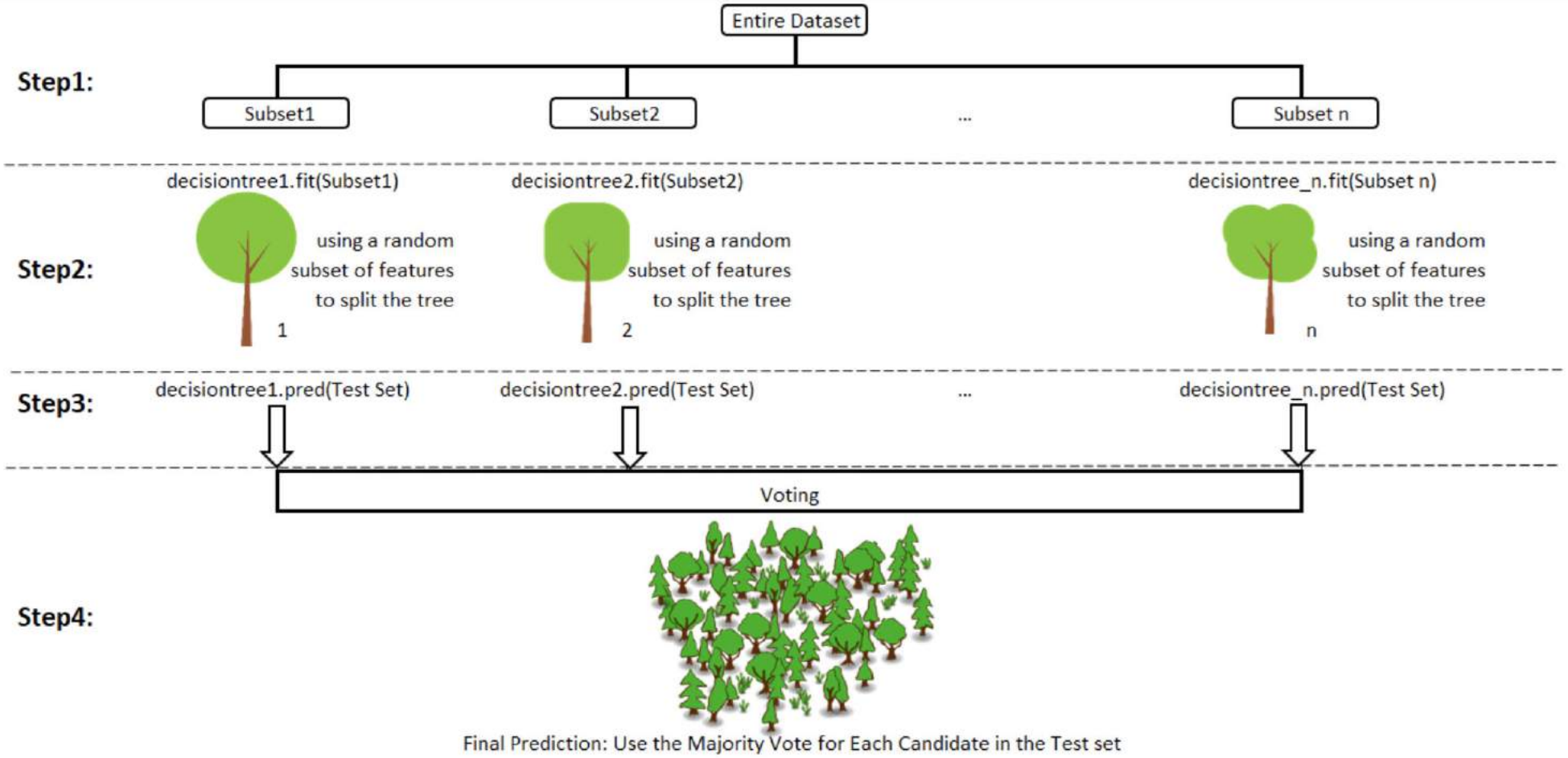
Classification

Boosting



Classification

Random Forest



Thank You!

In our next session:

Application of Classification and Introduction to case study

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Applications of Classification – Case Study

Srikanth Gunturu

Guest Faculty
BITS, WILP

In this segment

Applications of Classification – Case Study Overview

- Application of Classification
  - Healthcare
  - Finance
  - eCommerce
- Case study of the course - Overview

Applications of Classification – Case Study Overview

Applications of Classification - Healthcare

- Tumour classification – Malignant Benign (image based)
- Risk prediction for Diabetes, Heart disease, Alzheimer's etc.

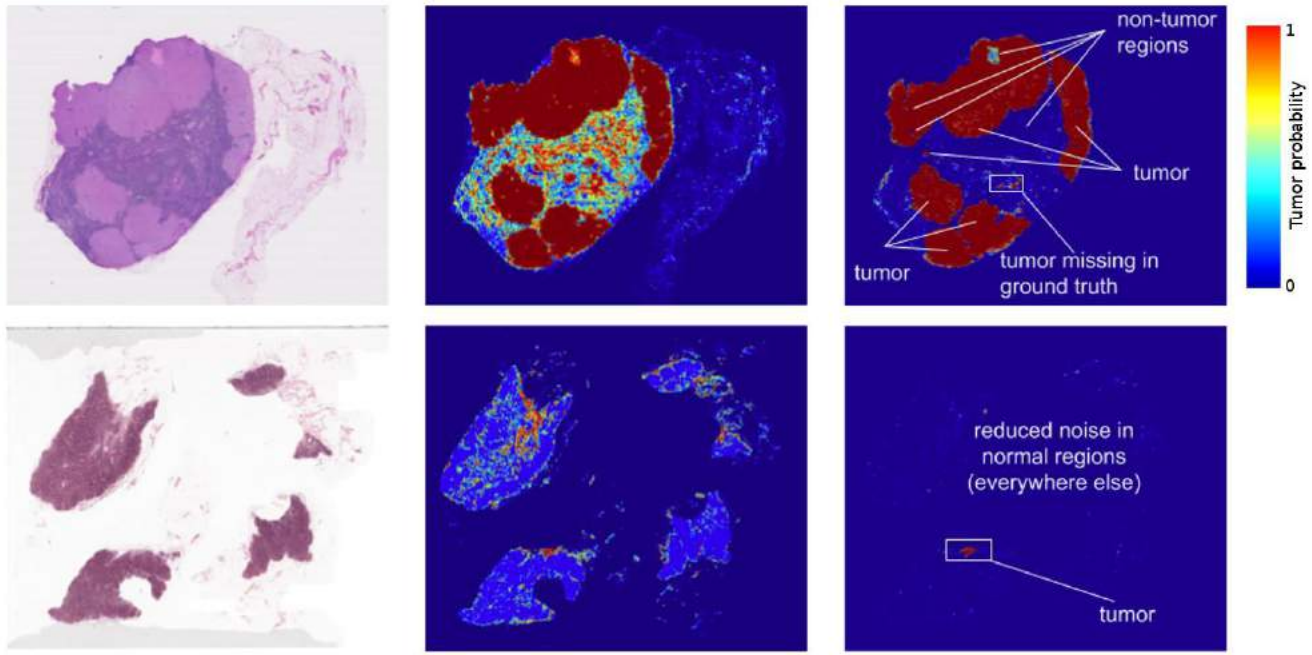


Image courtesy: Google AI blog

Applications of Classification – Case Study Overview

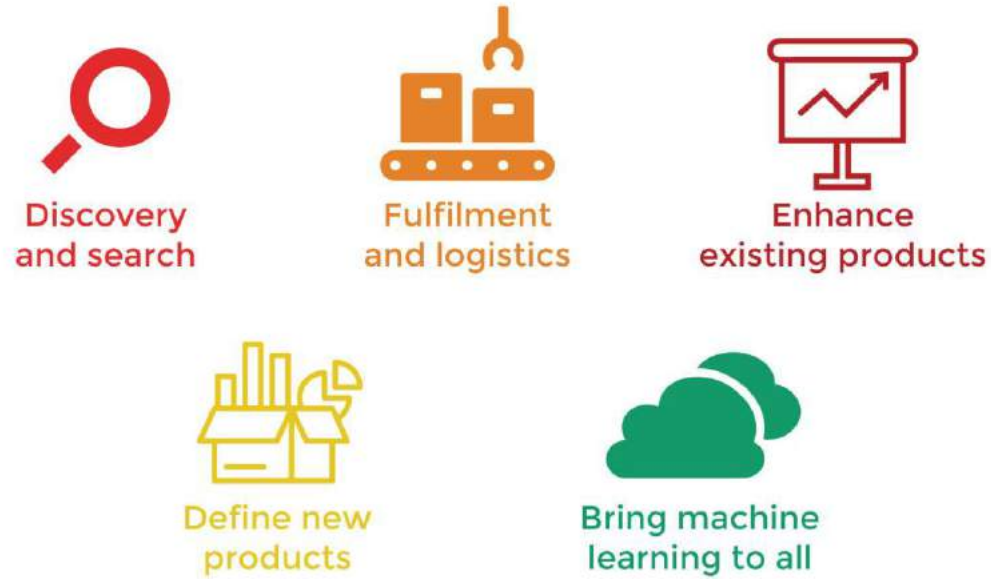
Applications of Classification - Finance

- Determining credit worthiness rating of corporates in lending
- Determine stock worthiness of the portfolio (buy/don't)
- Fraud detection in bulk transactions (stocks/loans etc.)

Applications of Classification – Case Study Overview

Applications of Classification - eCommerce

- Product Recommendations
- Smart search engines – image search
- Warehouse automation
- Sales forecasting
- Inventory forecast
- Dynamic pricing and price bundling



Applications of Classification – Case Study Overview

Case Study 1 (classroom) - Overview

- Diabetes Mellitus prediction in women
- Two-class problem (yes/no classification)
- Data set – 768 patients with health records and family history as parameters
- Input attribute set
  - Pregnancies
  - Glucose
  - Blood Pressure
  - Skin Thickness
  - Insulin
  - BMI
  - Age
- Label
  - Outcome (diabetic or non-diabetic)

Applications of Classification – Case Study Overview

Case Study 2 (exercises) - Overview

- Predict presence of Liver disease in a patient based on vitals
- Two-class problem (yes/no classification)
- Data set – Data 583 patients with health parameters
- Input attribute set
  - Age
  - Gender
  - Bilirubin (total & direct)
  - Alkaline Phosphatase
  - Alamine Aminotransferase
  - Aspartate Aminotransferase
  - Total Proteins
  - Albumin
  - Albumin & Globulin Ratio
- Label
  - Outcome (liver disease present or not)

Thank You!

In our next session:

k-Nearest Neighbor Classifier

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

k-Nearest Neighbor Classifier

Dr. Chetana Gavankar

Associate Professor, BITS Pilani
Chetana.gavankar@pilani.bits-pilani.ac.in

In this segment

k-Nearest Neighbor Classifier

- Nearest Neighbor – Overview
- kNN classifier – Introduction
- kNN classifier – example
- Variations with k

k-Nearest Neighbor Classifier

Instance based learning – Nearest Neighbor Classifiers

- Nearest Neighbour classifier is an instance based classifier
- Also called ‘lazy learning’, as learning is postponed until a new instance is encountered
- Constructs a local approximation to the target function, applicable (better suited) in the neighbourhood of new instance
- Suitable in cases where target function is complex over the entire input space, but easily describable in local approximations
- Caveat is the high cost of classification, which happens at the time of processing rather than before hand (there’s no training phase)

Lazy vs. Eager Learning

- Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
- Eager learning (the above discussed methods): Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting

k-Nearest Neighbor Classifier

kNN Classifier - Introduction

- Considers all instances as members of n-dimensional space
- Nearest neighbours of an instance is determined based on Euclidean distance
- Distance between two n-dimensional instances x_i and x_j is given by:

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

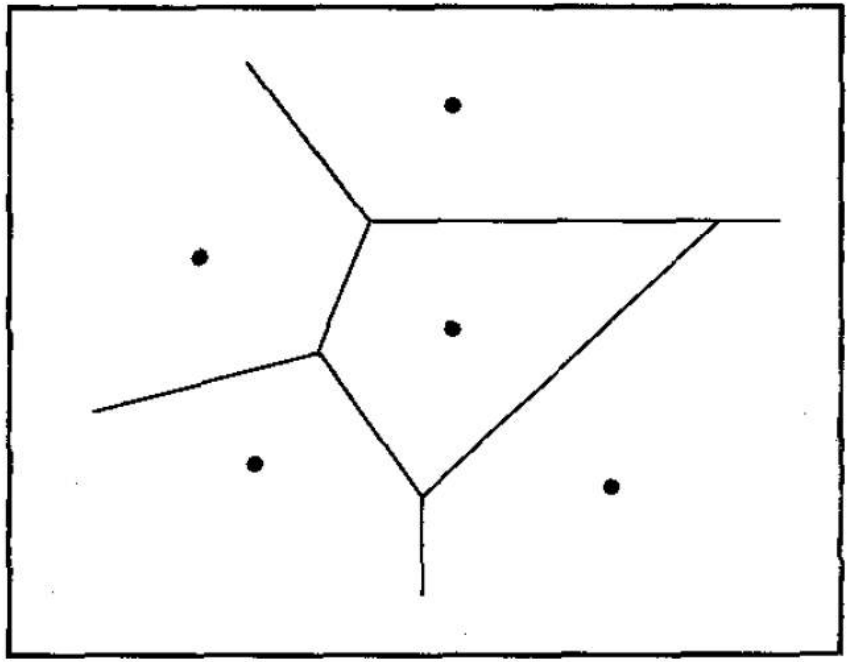
- For NN classifier, target function can be discrete or continuous

Steps involved in kNN

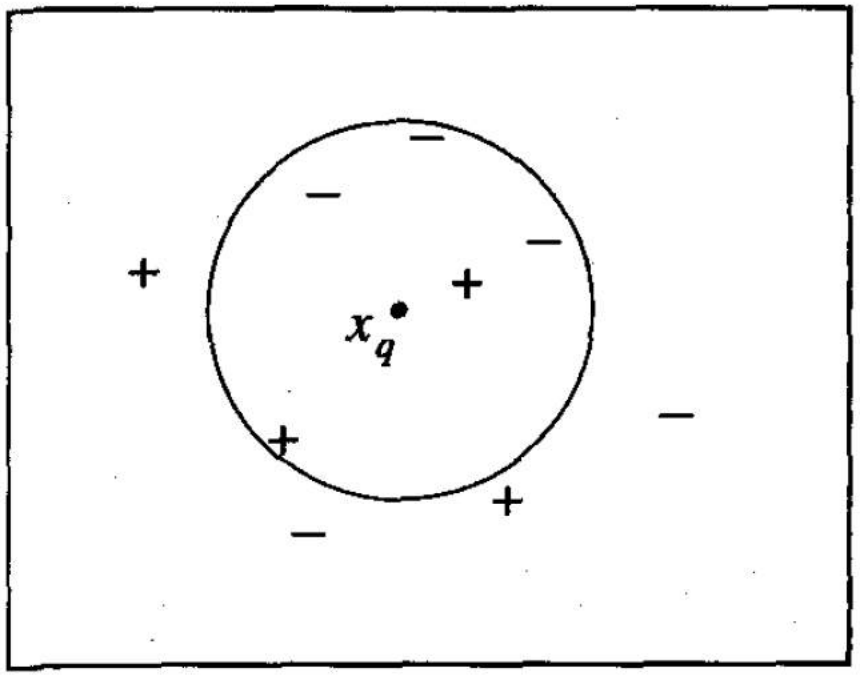
- Divide the data into training and test data.
- Select a value K.
- Determine which distance function is to be used.
- Choose a sample from the test data that needs to be classified and compute the distance to its n training samples.
- Sort the distances obtained and take the k-nearest data samples.
- Assign the test class to the class based on the majority vote of its k neighbours.

k-Nearest Neighbor Classifier

kNN Classifier – Example(s)



K=1



K=5

k-Nearest Neighbor Classifier

kNN Classifier - Algorithm

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

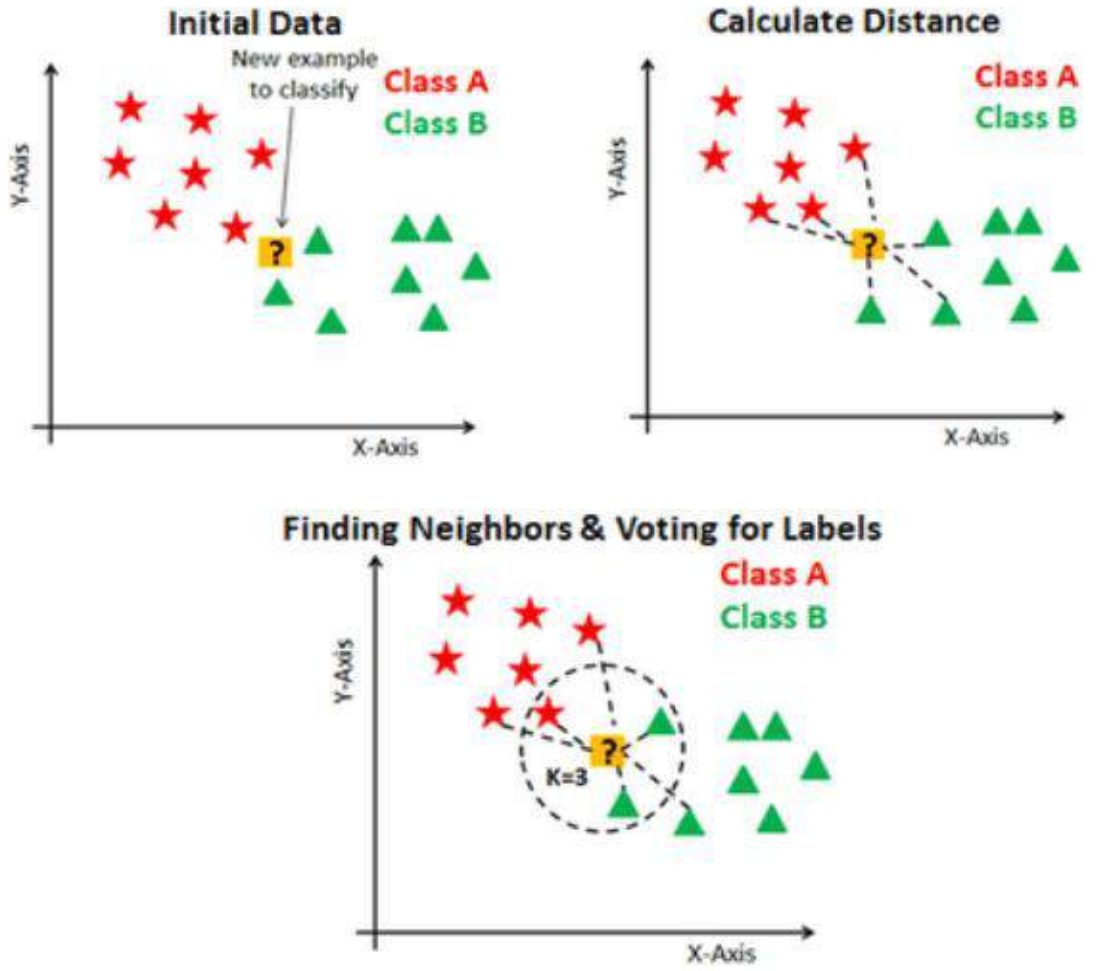
Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = 1$ if $a = b$ and where $\delta(a, b) = 0$ otherwise.

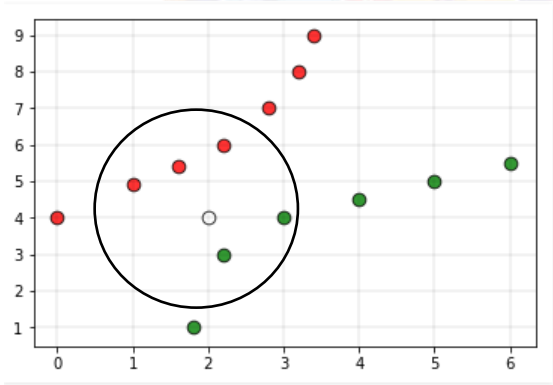
k-Nearest Neighbor Classifier



Source: DataCamp

Challenges in k-NN

- Performance of a classifier largely depends on the of the hyperparameter k
  - Choosing smaller values for K, noise can have a higher influence on the result.
  - Larger values of k are computationally expensive
- Assigning the class labels can be tricky. For example, in the below case, for (k=5) the point is closer to ‘green’ classification, but gets classified as ‘red’ due to higher red votes/majority voting to ‘red’



k-Nearest Neighbor Classifier- Distance weighed

- This refinement adds a ”weight” factor to kNN algorithm
- Weight would be based on the distance (ex: inverse square of distance from query instance), which gives more weightage to the closer neighbours
- Modified function would be:

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

where $w_i \equiv \frac{1}{d(x_q, x_i)^2}$

- This method is robust to noisy training data and effective for large set of training data
- kNN is the well-known “curse of dimensionality”
  - Can be overcome by weighing each attribute differently while calculating distances or eliminating irrelevant attributes

Factors that affect kNN Performance

Performance of the K-NN algorithm is influenced by the following factors:

- The distance function or distance metric used to determine the nearest neighbours.
- The number of neighbours (=k) used to classify the new example.

PROS and CONS of kNN

Pros

- The K-NN algorithm is very easy to implement.
- Nearly optimal in the large sample limit.
- Uses local information, which can yield highly adaptive behaviour.

Cons

- Large storage requirements.
- Computationally intensive recall.
- Highly susceptible to the curse of dimensionality.
- Consider all attributes of instances to retrieve similar training examples from memory

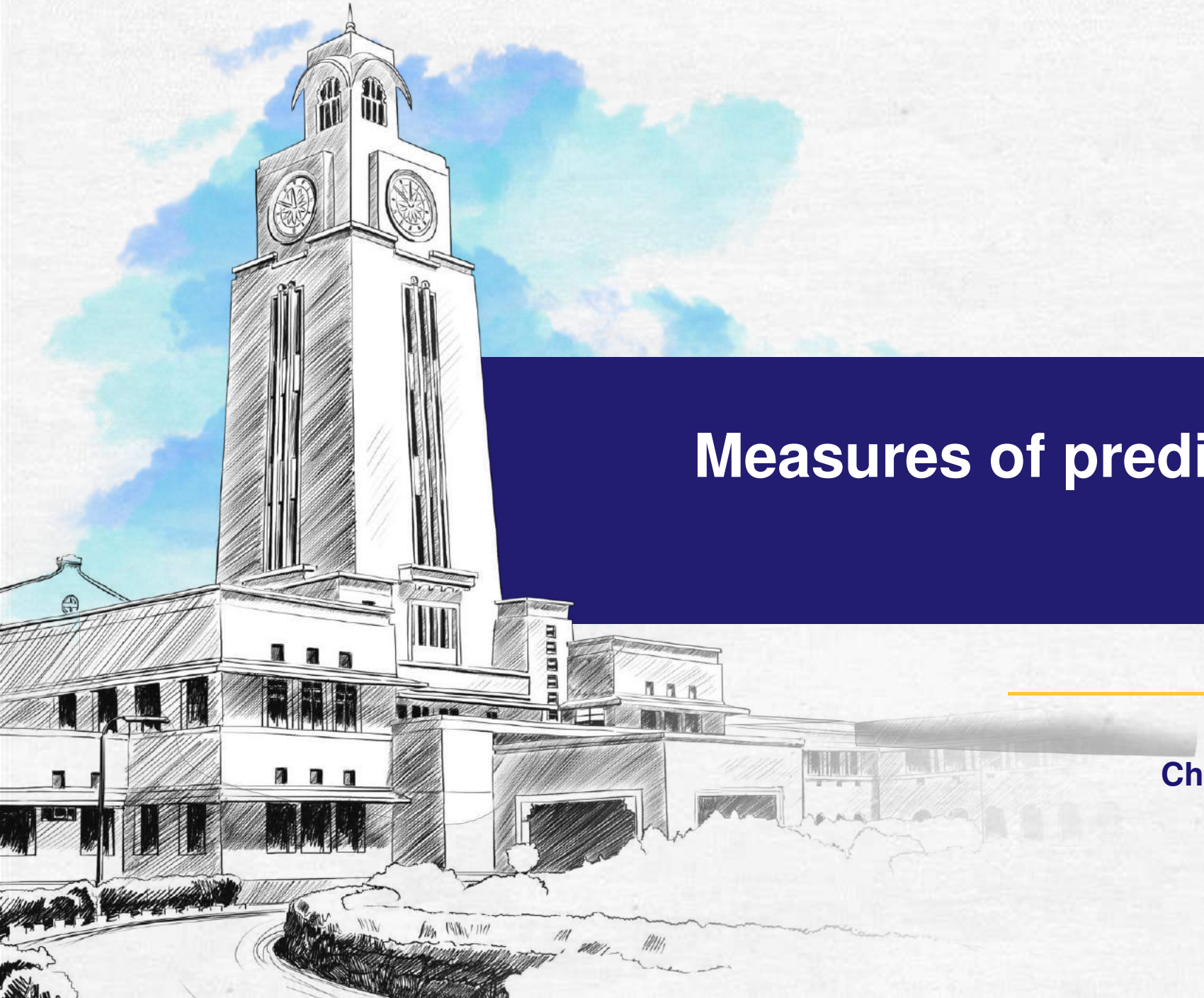
Real world applications

- Finance — financial institutes will predict the credit rating of customers.
- Healthcare
- Political Science — classifying potential voters in two classes will vote or won't vote.
- Handwriting detection.
- Image Recognition.

Thank You!

In our next session:

Measures of prediction accuracies of classifiers



BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Measures of prediction accuracies of classifiers

Dr. Chetana Gavankar

Associate Professor, BITS Pilani
Chetana.gavankar@pilani.bits-pilani.ac.in

In this segment

Measures of prediction accuracies of classifiers

- Metrics
  - Precision
  - Recall
  - F1
- ROC Curve – AUC of ROC
- Generating ROC Curve

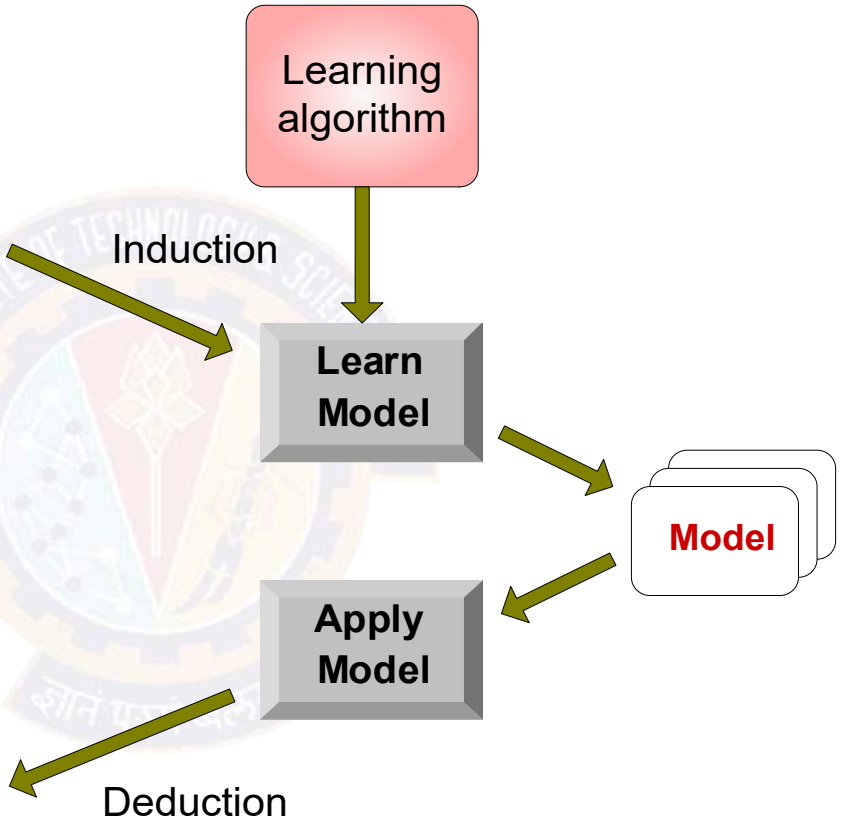
Classification

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Confusion Matrix

- **True Positive (TP):** It refers to the number of
•predictions where the classifier correctly predicts
•the positive class as positive.
- **True Negative (TN):** It refers to the number of
•predictions where the classifier correctly predicts
•the negative class as negative.
- **False Positive (FP):** It refers to the number of
•predictions where the classifier incorrectly
•predicts the negative class as positive.
- **False Negative (FN):** It refers to the number of
•predictions where the classifier incorrectly predicts
•the positive class as negative.

Predicted class ->	C_1	$\neg C_1$
Actual class⇓		
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

4

Classifier Evaluation Metrics: Accuracy, Error Rate,

- **Classifier Accuracy,** or recognition rate: percentage of test set tuples that are
- correctly classified

  Accuracy = (TP + TN)/All

  most effective when the class distribution is relatively balanced

- **Classification Error/ Misclassification rate:** *1 – accuracy*, or

  = (FP + FN)/All

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

5

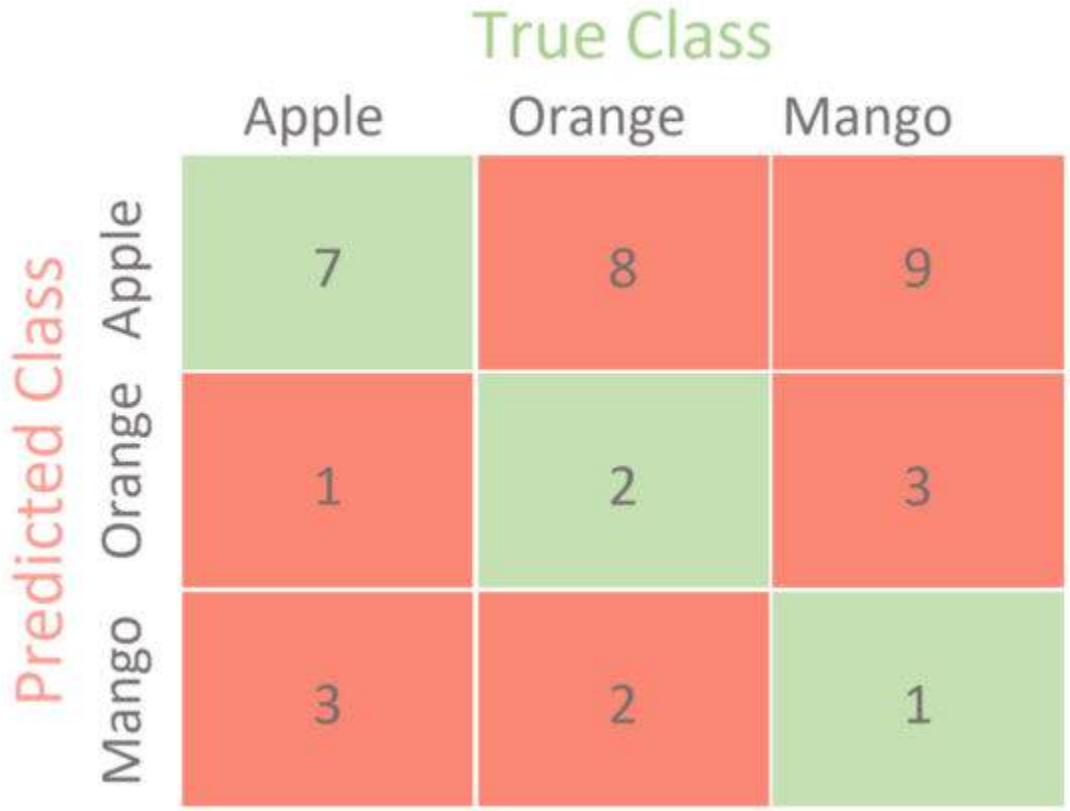
Accuracy

- Most widely-used metric:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Multiclass



- Find TP, TN, FP and FN for each individual class.
- For example, if we take class Apple,
- ***TP = 7***
- ***TN = (2+3+2+1) = 8***
- ***FP = (8+9) = 17***
- ***FN = (1+3) = 4***

Class Imbalance Problem

- Find needle in haystack
- Lots of classification problems where the classes are skewed (more records from one class than another)
  - Credit card fraud
  - Intrusion detection
  - Defective products in manufacturing assembly line

Approaches to solve Class imbalance problem

- Up-sample minority class
  - randomly duplicating observations from a minority class
- Down-sample majority class
  - removing random observations.
- Generate Synthetic Samples
  - new samples based on the distances between the point and its nearest neighbors
- Change the performance metric
  - Use Recall, Precision or ROC curves instead of accuracy
- Try different algorithms
  - Some algorithms as Support Vector Machines and Tree-Based algorithms are better to work with imbalanced classes.

Problem with Accuracy

- Consider a 2-class problem
  - Number of Class NO examples = 990
  - Number of Class YES examples = 10

- If a model predicts everything to be class NO, accuracy is 990/1000 = 99 %
  - This is misleading because the model does not detect any class YES example
  - Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)

Alternative Measures

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Precision (p)} = \frac{a}{a+c} \qquad precision = \frac{TP}{TP+FP}$$

$$\text{Recall (r)} = \frac{a}{a+b} \qquad recall = \frac{TP}{TP+FN}$$

$$\text{F - measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

Alternative Measures

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	10	0
	Class=No	10	980

$$\text{Precision (p)} = \frac{10}{10+10} = 0.5$$

$$\text{Recall (r)} = \frac{10}{10+0} = 1$$

$$\text{F - measure (F)} = \frac{2*1*0.5}{1+0.5} = 0.62$$

$$\text{Accuracy} = \frac{990}{1000} = 0.99$$

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	1	9
	Class=No	0	990

$$\text{Precision (p)} = \frac{1}{1+0} = 1$$

$$\text{Recall (r)} = \frac{1}{1+9} = 0.1$$

$$\text{F - measure (F)} = \frac{2*0.1*1}{1+0.1} = 0.18$$

$$\text{Accuracy} = \frac{991}{1000} = 0.991$$

Example

Given below is a confusion matrix for medical data where the class values are yes and no for a class label attribute, cancer. Calculate precision and recall of the classifier.

Classes	*yes*	*no*	*Total*	*Recognition (%)*
yes	**90**	**210**	300	30.00
no	**140**	**9560**	9700	98.56
Total	230	9770	10,000	96.40

Confusion matrix for the classes *cancer* = *yes* and *cancer* = *no*.

Reference: Data Mining: concepts and techniques, Han kamber, 3rd edition

Measures of prediction accuracies of classifiers

Metrics - Basics

- True positive rate (TPR) or **sensitivity** is defined as the fraction of positive examples predicted correctly by the model

$$TPR = TP/(TP + FN)$$

- True negative rate (TNR) or **specificity** is defined as the fraction of negative examples predicted correctly by the model

$$TNR = TN/(TN + FP)$$

- False positive rate (FPR) is the fraction of negative examples predicted as a positive class

$$FPR = FP/(TN + FP)$$

- False negative rate (F N R) is the fraction of positive examples predicted as a negative class

$$FNR = FN/(TP + FN)$$

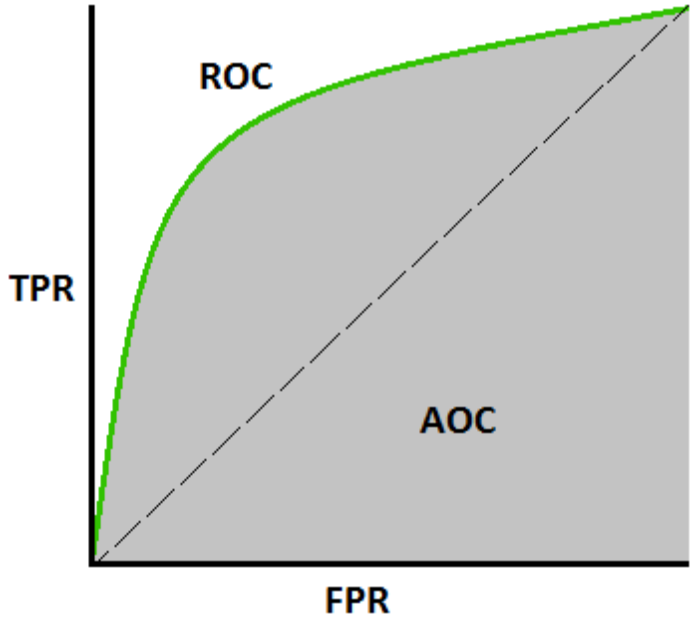
ROC (Receiver Operating Characteristic)

- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - Performance of a model represented as a point in an ROC curve
  - Changing the threshold parameter of classifier changes the location of the point

ROC

Receiver Operating Characteristic (ROC) Curve

- ROC curve plots TPR and FPR, to graphically represent their trade-off
- Area Under Curve (AUC) of ROC – evaluates model performance on average
  - AUC of ROC = 1, for a perfect model
  - AUC of ROC = 0.5, if the model is random
- For model comparison, AUC of ROC should be larger for the model to be superior or better performing



$$TPR = TP/(TP + FN)$$

$$FPR = FP/(TN + FP)$$

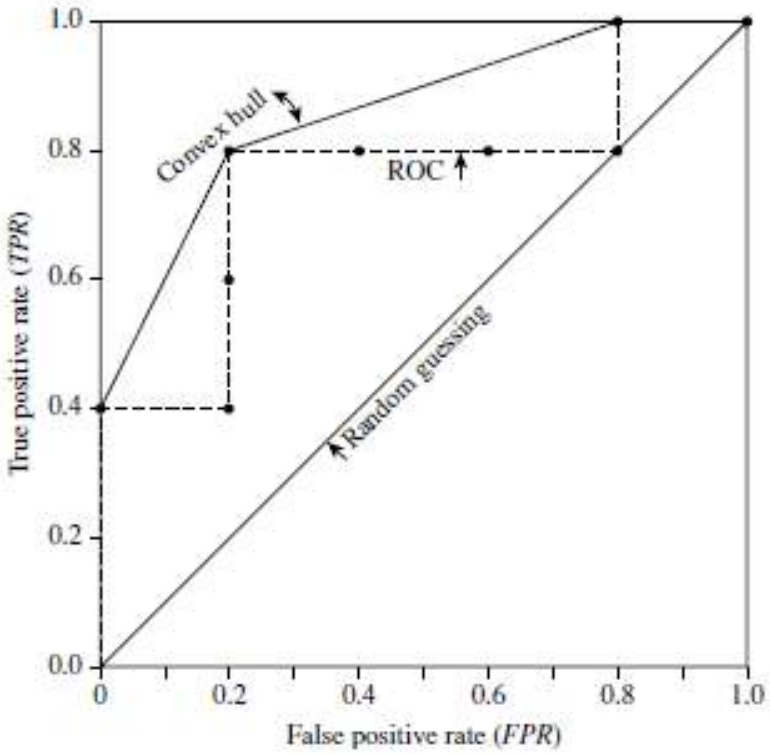
Example

- The table below shows the probability value (column 3) returned by a probabilistic classifier for each of the 10 tuples in a test set, sorted by decreasing probability order. The corresponding ROC is given on right hand side.

$$TPR = TP/(TP + FN)$$

$$FPR = FP/(TN + FP)$$

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	P	0.90	1	0	5	4	0.2	0
2	P	0.80	2	0	5	3	0.4	0
3	N	0.70	2	1	4	3	0.4	0.2
4	P	0.60	3	1	4	2	0.6	0.2
5	P	0.55	4	1	4	1	0.8	0.2
6	N	0.54	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.51	4	4	1	1	0.8	0.8
9	P	0.50	5	4	0	1	1.0	0.8
10	N	0.40	5	5	0	0	1.0	1.0



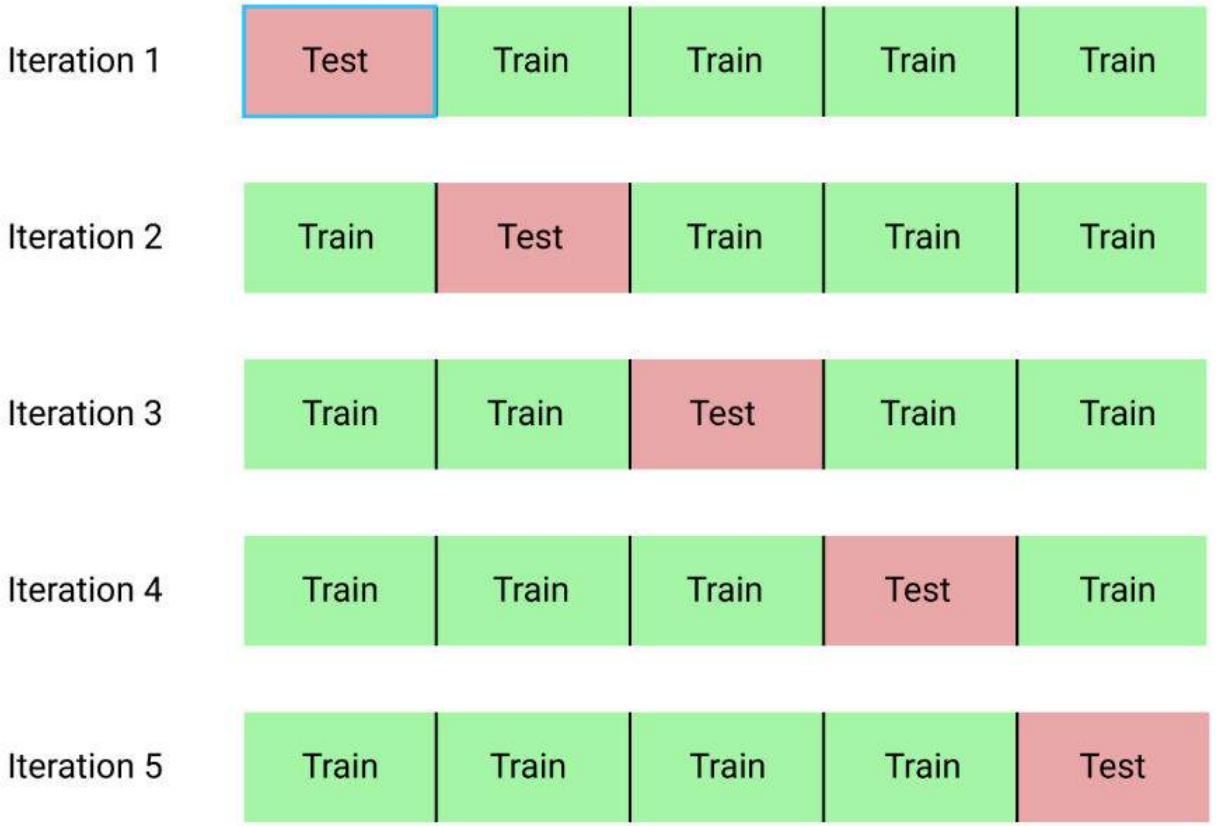
Evaluating Classifier Accuracy

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- **Cross-validation** (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - The Accuracy of the model is the average of the accuracy of each fold.

18

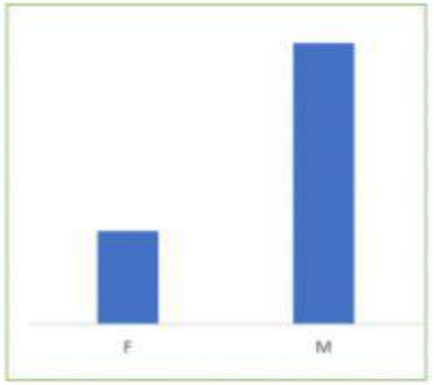
Cross Validation

k-Fold Cross Validation:

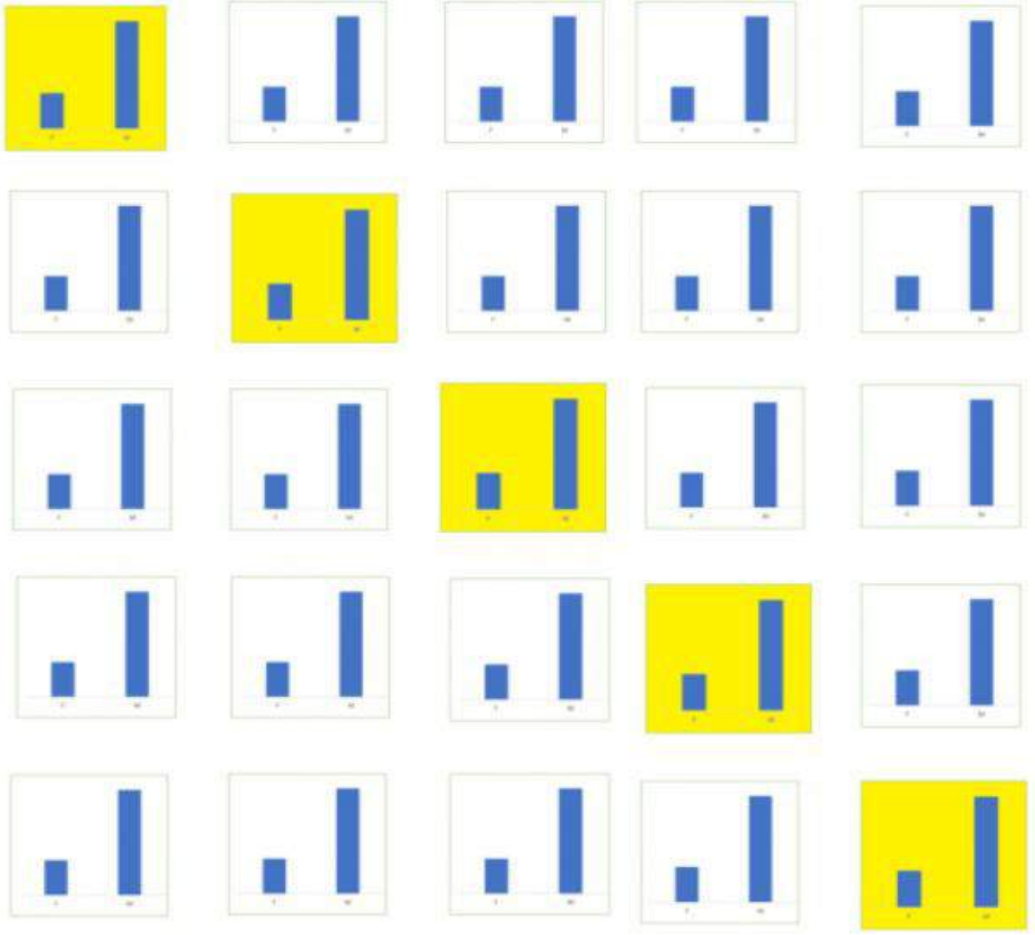


Stratified Cross Validation

Stratified K-Fold Cross Validation (K=5)



Class Distributions



Fold 1 Fold 2 Fold 3 Fold 4 Fold 5

Thank You!

In our next session:

Finding optimal k for kNN classifiers

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Finding optimal k for kNN classifiers

Dr. Chetana Gavankar

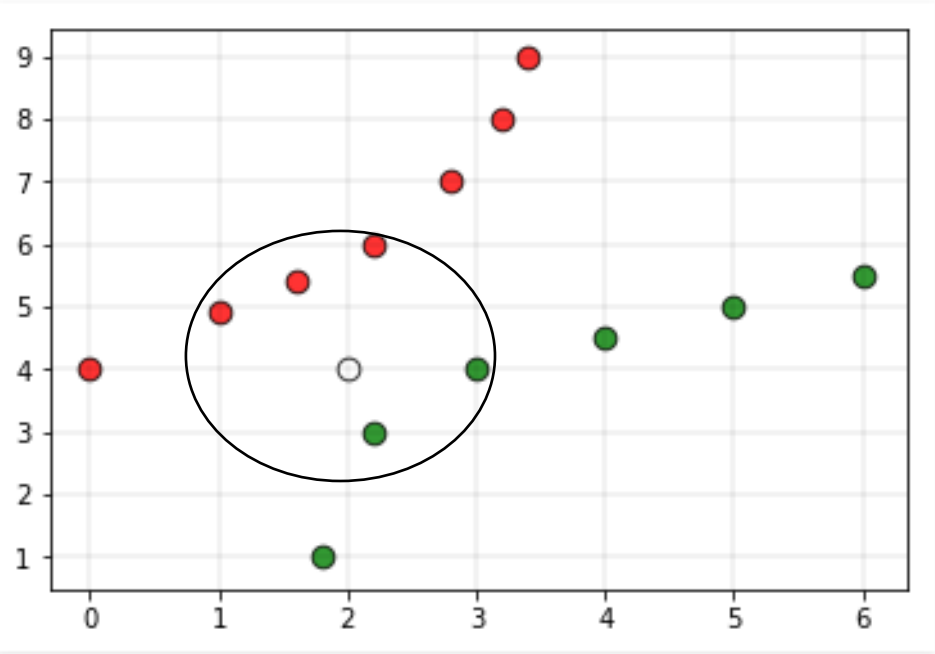
In this segment

Finding optimal k for kNN classifiers

- The problem of finding K
- Elbow method – Optimal K

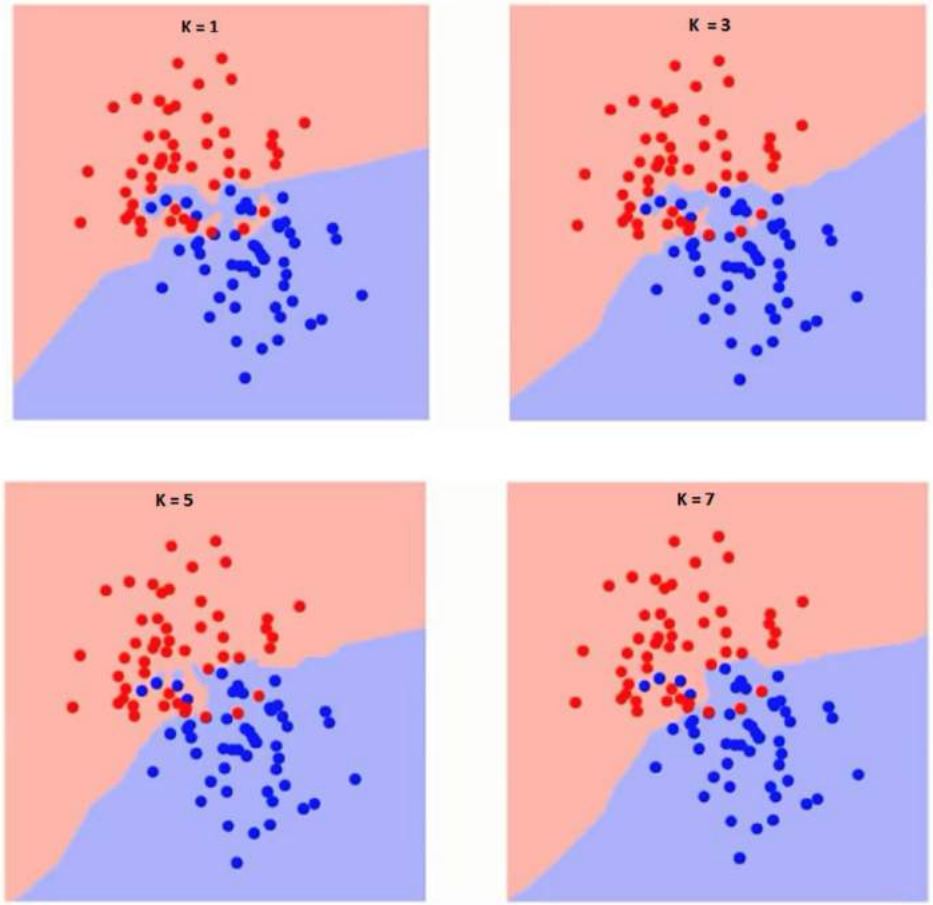
Various issues that affect the performance of kNN

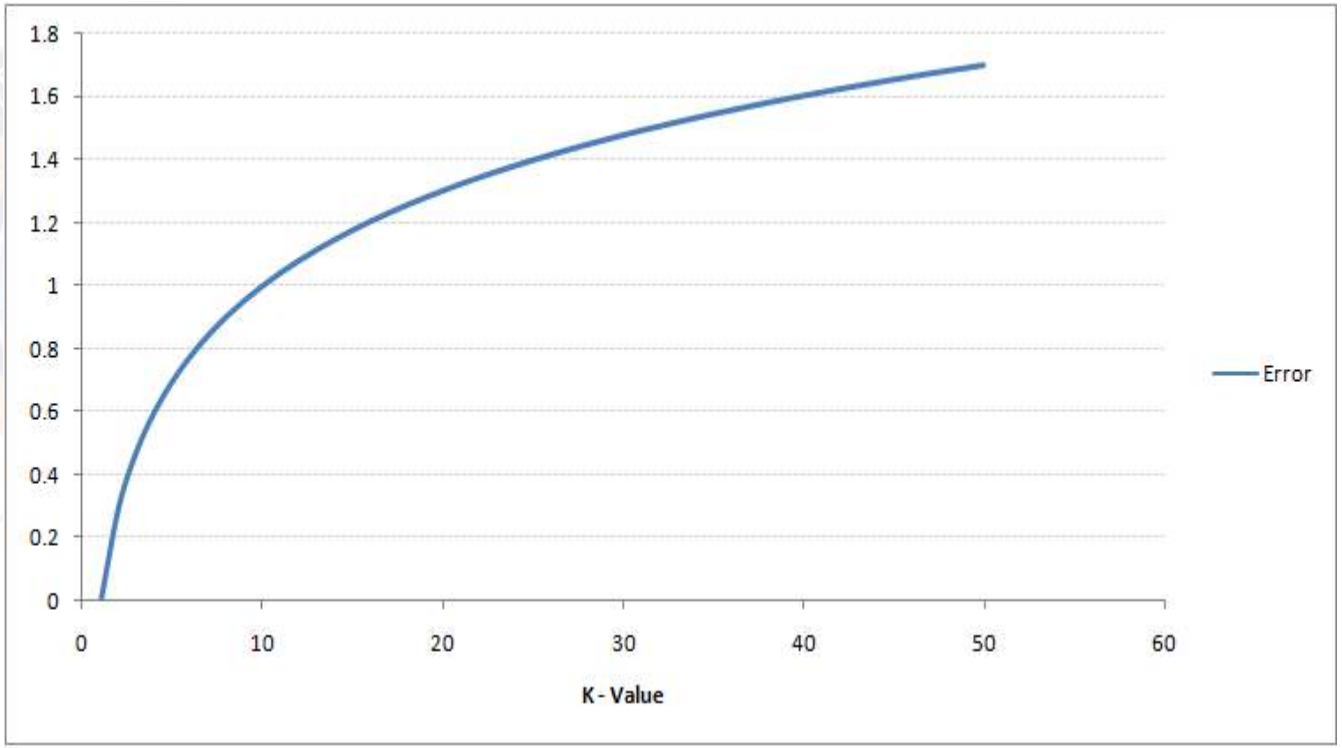
- Performance of a classifier largely depends on the of the hyperparameter k
  - Choosing smaller values for K, noise can have a higher influence on the result.
  - Larger values of k are computationally expensive
- Assigning the class labels can be tricky. For example, in the below case, for (k=5) the point is closer to ‘green’ classification, but gets classified as ‘red’ due to higher red votes/majority voting to ‘red’



Finding optimal k for kNN classifiers

What is the ideal value of ‘K’?

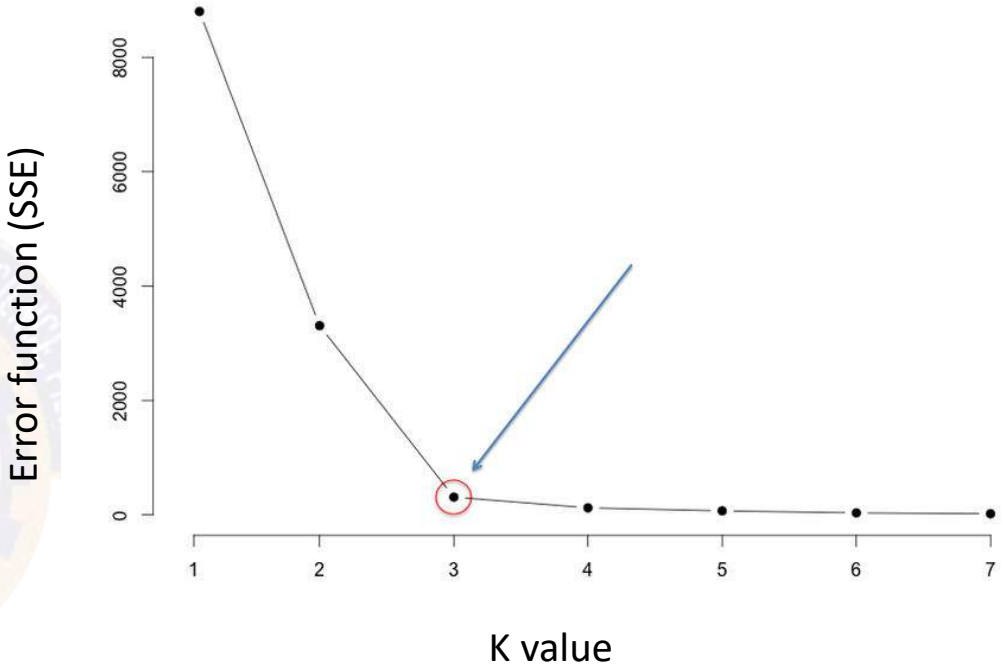




Finding optimal k for kNN classifiers

Finding K - Elbow method

- Compute sum of squares error (SSE) or any other error function for varying values of K (1 to a reasonable X) and plot against K
- In the plot, the elbow (see pic) gives the value of K beyond which the error function plot almost flattens
- As K approaches the total number of instances in the set, error function drops down to '0', but beyond optimal K, the model becomes too generic



Thank You!

In our next session:

Python implementation of kNN classifier

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Probability Foundations

Dr. Chetana Gavankar

In this segment

Probability Foundations

- Probability - Terminology
- Discrete Random Variable
- Continuous Random Variable
- Conditional Independence

Probability basics

Random Experiment

- A random experiment is an experiment or a process for which the outcome cannot be predicted with certainty
- Example:
  - Tossing a coin
  - number of calls to a communication system during a fixed length interval of time
  - Rolling a die
  - Number of dishwashers sold in India on a particular date

Sample Space:

- The sample space of a random experiment is the set of all possible outcomes. Denoted by omega Ω or sample space S
- Example: Random Experiment: Toss a fair coin once.
  Sample Space: Ω ={Head, Tail}

Probability basics

Events

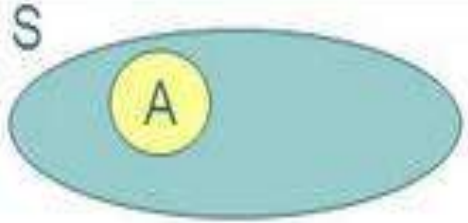
- Events are the subsets of the sample space
  - Simple Events: If an event consists of only one outcome, it is a simple event
- Union and Intersection: If A and B are events, then A∪B and A∩B are also events.
- Examples:
  - A= {the outcome that the dice is even} = {2,4,6}
  - B= {at least 2 tails}

Outcome: A result of a random experiment.
Sample Space: The set of all possible outcomes.
Event: A subset of the sample space.

Probability Theory

Let A be an event, then we denote

$P(A)$ the probability for A



It always hold that $0 \leq P(A) \leq 1$ $P(\emptyset) = 0$ $P(S) = 1$

Consider an experiment which has N equally likely outcomes, and let exactly n of these events correspond to the event A. Then

$$P(A) = \frac{n}{N} = \frac{\#\text{ successful outcomes}}{\#\text{ possible outcomes}}$$

Example:
Rolling a dice

$P(\text{even number})$

$$= \frac{3}{6} = \frac{1}{2}$$

Random Variable

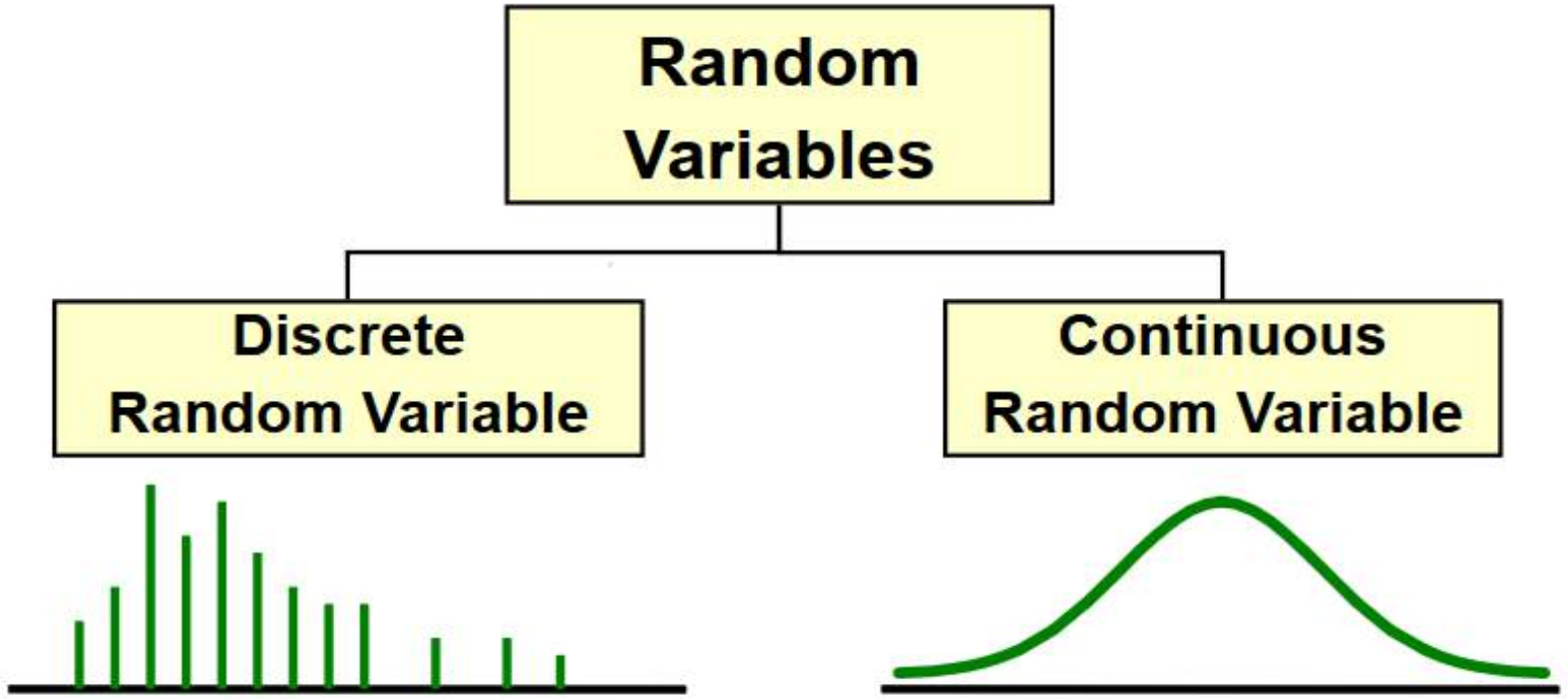
- A **random variable**, usually written X, is a **variable** whose possible values are numerical outcomes of a **random** phenomenon or experiment.

Examples

- ✓ X = number of heads when the experiment is flipping a coin 20 times.
- ✓ C = the daily change in a stock price.
- ✓ R = the number of kilometers per litre you get on your car during a family vacation.

Random Variable

Represents a possible numerical value from a random event



Random Variable

Discrete Random Variable

- one that takes on a ***countable*** number of values
- usually count data [Number of]
- list **all** possible outcomes without missing any of them

Example:

✓ X = sum of values on the roll of two dice:
X has to be either 2, 3, 4, …, or 12.

✓ Y = number of students in AIML:
Y has to be 60,65,70 ……

Random Variable

Continuous Random Variable

- Variable that takes on an uncountable number of values
- Usually measurement data [time, weight, distance, etc.]
- You can never list all possible outcomes even if you had an infinite amount of time

Example:

X = time it takes you to drive home from work place: X > 0, might be 30.1 minutes measured to the nearest tenth but in reality the actual time is 30.100001……. minutes?)

Probability Theory

Notation

- A **random variable X** represents outcomes or states of the world.
- We will write p(x) to mean Probability(X = x)
- **Sample space**: the space of all possible outcomes (may be discrete, continuous, or mixed)
- p(x) is the **probability mass (density) function**
  - Assigns a number to each point in sample space
  - Non-negative, sums (integrates) to 1
  - Intuitively: how often does x occur, how much do we believe in x.

Probability Densities

Continuous Probability Distribution

Let X be a continuous rv. Then a *probability distribution or probability density function (pdf)* of X is a function $f(x)$ such that for any two numbers a and b,

$$P(a \le X \le b) = \int_a^b f(x)dx$$

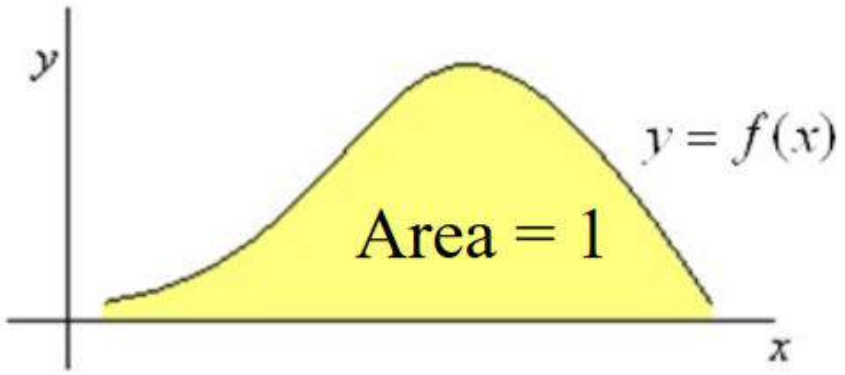
The graph of f is the *density curve*.

Probability Densities

For $f(x)$ to be a pdf

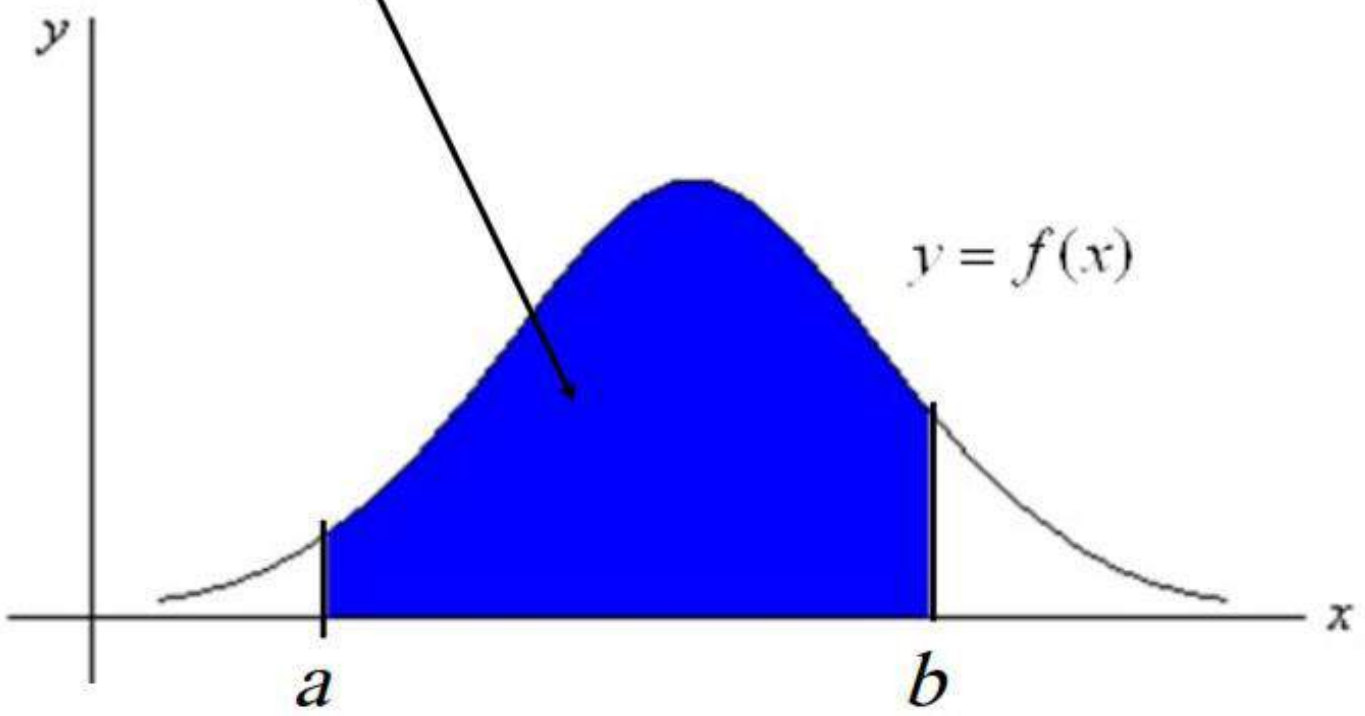
1. $f(x) > 0$ for all values of x.

2. The area of the region between the graph of f and the $x-$ axis is equal to 1.



Probability Densities

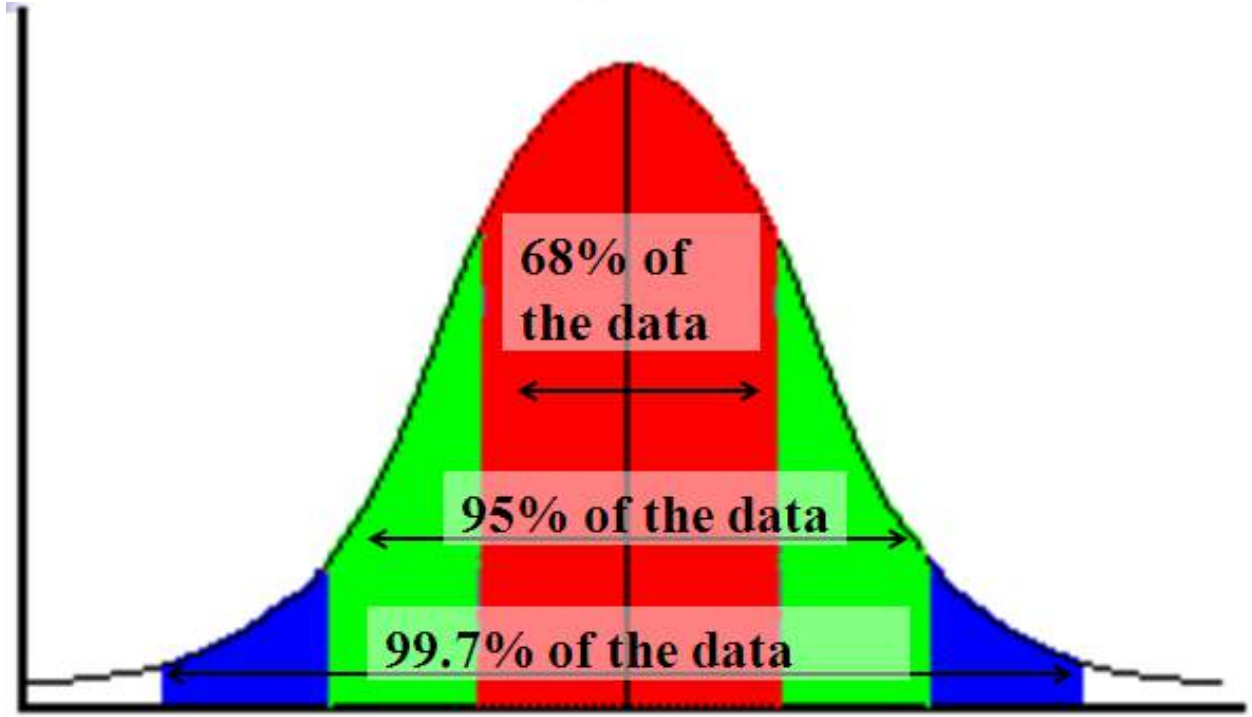
$P(a \leq X \leq b)$ is given by the area of the shaded region.



Gaussian Distribution

Empirical Rule:

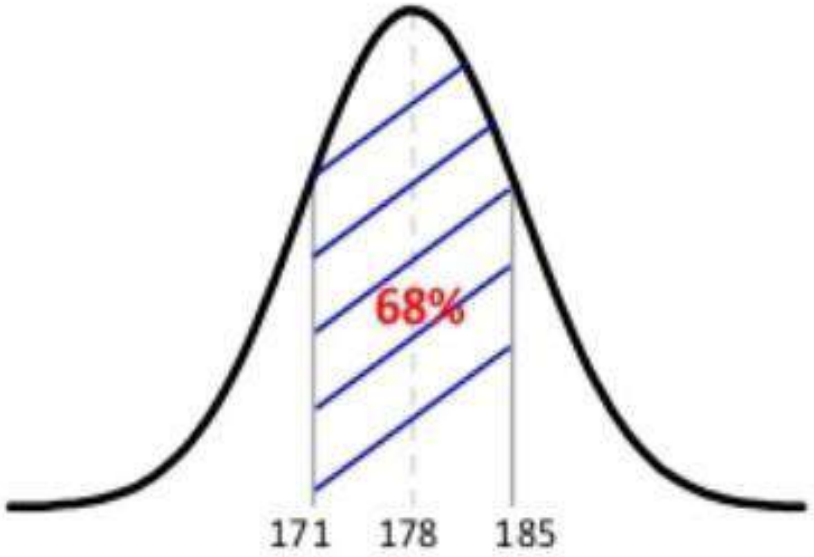
- For any normally distributed data:
  - **68%** of the data fall within **1** standard deviation of the mean.
  - **95%** of the data fall within **2** standard deviations of the mean.
  - **99.7%** of the data fall within **3** standard deviations of the mean.



Gaussian Distribution

- Suppose that the heights of a sample men are normally distributed.
- The mean height is **178** cm and a standard deviation is **7** cm.

- **We can generalize that:**
  - **68%** of population are between **171** cm and **185** cm.
  - This might be a generalization, but it's true if the data is normally distributed.



Mean, Variance & Standard Deviation

✓The mean of a discrete random variable is the ***weighted average*** of all of its values. The weights are the probabilities.

✓This parameter is also called the expected value of X and is represented by E(X).

$$E(X) = \mu = \sum_{all\ x} xP(x)$$

✓The variance is

$$V(X) = \sigma^2 = \sum_{all\ x} (x-\mu)^2 P(x)$$

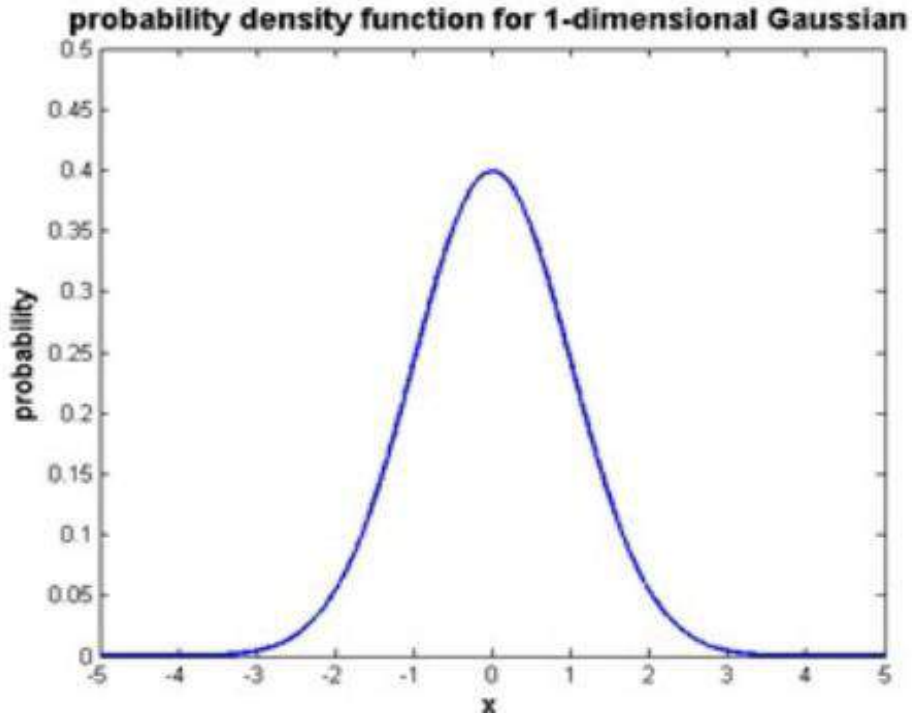
✓The standard deviation is

$$\sigma = \sqrt{\sigma^2}$$

Gaussian Distribution

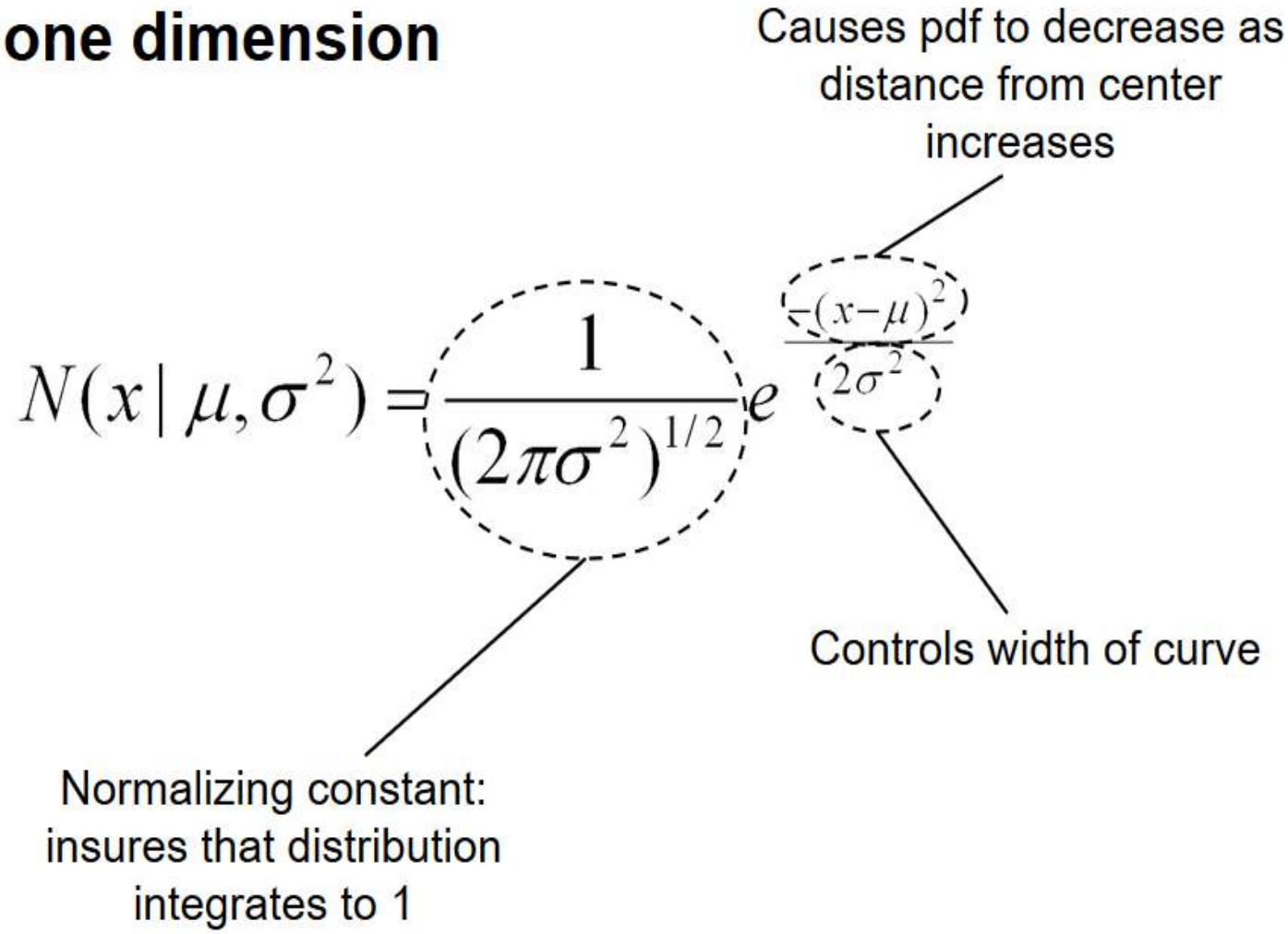
In one dimension

$$N(x \mid \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

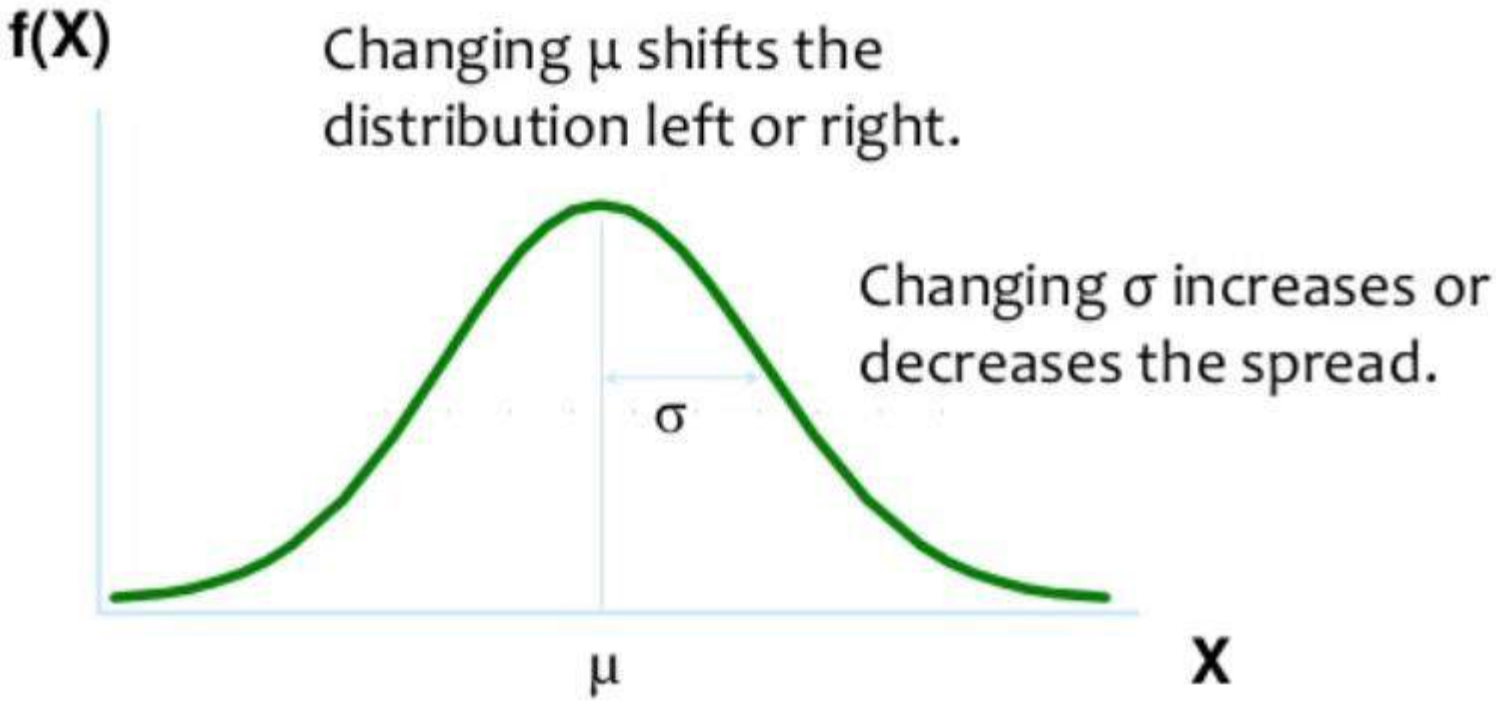


Gaussian Distribution

In one dimension

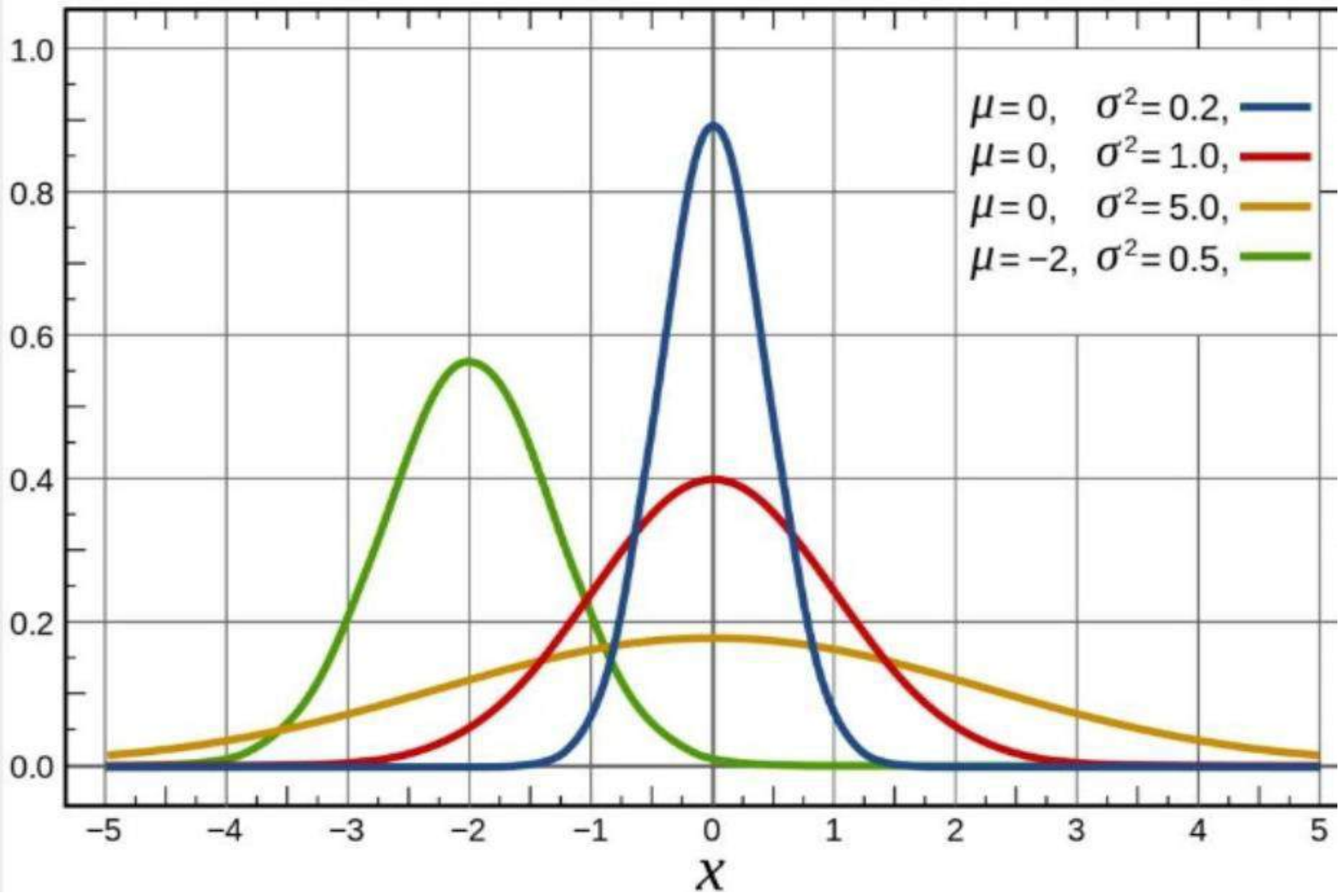


Gaussian Distribution



The normal curve is not a single curve but a family of curves, each of which is determined by its mean and standard deviation.

Gaussian Distribution



Joint Probability

- **Joint probability** is the probability of two events happening together. The two events are usually designated *event A* and *event B.* In probability terminology, it can be written as:
- P(X and Y) or P(A n B) or P(X,Y)

Example:

- The probability that a card is a five and black,

  p(five and black) = 2/52 = 1/26,

(There are two black fives in a deck of 52 cards, the five of spades and the five of clubs)

Independence

- Two events A and B are independent if and only if $P(A \cap B)=P(A)P(B)$.

$$\begin{aligned} P(A|B) &= \frac{P(A \cap B)}{P(B)} \\ &= \frac{P(A)P(B)}{P(B)} \\ &= P(A). \end{aligned}$$

- If two events A and B are independent and $P(B) \neq 0$, then $P(A|B)=P(A)$.
- In general, for n events $A_1, A_2, \cdots, A_n$ to be independent we must have

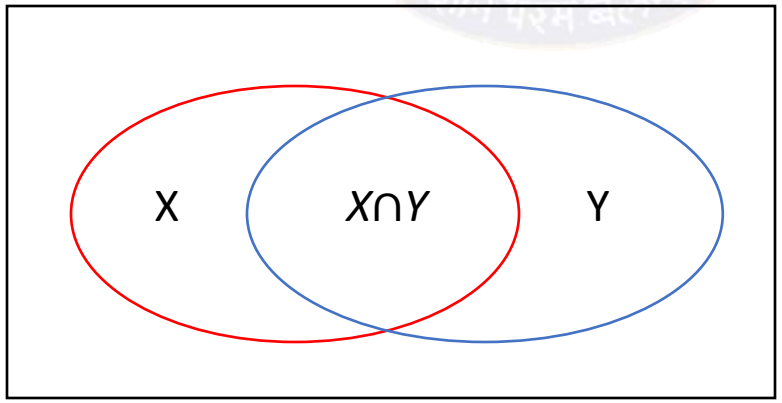
$$P(A_i \cap A_j) = P(A_i)P(A_j), \text{ for all distinct } i, j \in \{1, 2, \cdots, n\};$$

$$P(A_i \cap A_j \cap A_k) = P(A_i)P(A_j)P(A_k), \text{ for all distinct } i, j, k \in \{1, 2, \cdots, n\};$$

Conditional Probability

- Two random variables X and Y are said to be independent if their occurrence is not dependent on each other
- In that case, probability of both events occurring $P(X \cap Y) = P(X).P(Y)$
  - Ex: Outcome of 2 coin tosses
- If random variable X is dependent on Y, then the probability of X given Y occurs is termed as 'conditional probability' $P(X \mid Y)$, which can be expressed as the fraction of times X and Y occurs over the occurrence of Y, i.e.

$$P(X \mid Y) = P(X \cap Y) / P(Y)$$



Bayes Theorem

- Conditional probability - given random variables X and Y, probability of X given Y can be expressed as:

$$P(X \mid Y) = P(X \cap Y) / P(Y)$$

- The same can be written for Y given X as:

$$P(Y \mid X) = P(Y \cap X) / P(X)$$

- Since $P(X \cap Y) = P(Y \cap X)$, solving both equations gives:

$$P(X \cap Y) = P(X \mid Y)\, P(Y) = P(Y \mid X)\, P(X)$$

- We can rewrite conditional probability of X given Y as:

$$P(X \mid Y) = P(Y \mid X)\, P(X) / P(Y)$$

- **This is also known as Bayes theorem**

- In plain English, this can be written as

$$posterior = \frac{likelihood \times prior}{evidence}$$

Bayes Theorem Example 1

if patient has meningitis, then very often a stiff neck is observed
$P(S|M) = 0.8$ (can be easily determined by counting)

observation: 'I have a stiff neck! Do I have meningitis?' (is it reasonable to be afraid?)
$P(M|S) = ?$

we need to now: $P(M) = 0.0001$ (one of 10000 people has meningitis)
and $P(S) = 0.1$ (one out of 10 people has a stiff neck).

then:

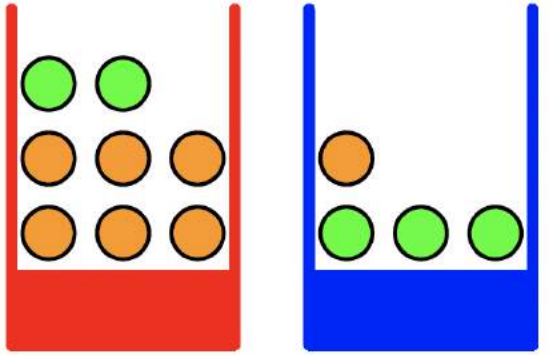
$$P(M|S) = \frac{P(S|M)P(M)}{P(S)} = \frac{0.8 \times 0.0001}{0.1} = 0.0008$$

Keep cool. Not very likely

Bayes Theorem Example 2

- Consider two boxes of fruit – red and blue, each containing apples (green color) and oranges
- Assume the probabilities of selecting red box (B=r) and blue box(B=b) at random is:

  $p(B = r) = 4/10$ $p(B = b) = 6/10$



- If we select a box at random and then select a fruit at random, the conditional probabilities for fruit selection thus become:

  $p(F = a \mid B = r) = 2/8=1/4$ $p(F = o \mid B = r) = 6/8=3/4$

  $p(F = a \mid B = b) = 3/4$ $p(F = o \mid B = b) = 1/4$

- Lets find out probability of picking an apple in general, using sum and products rule of probability

  $p(F = a) = p(F = a \mid B = r) \,.\, p(B = r) + p(F = a \mid B = b) \,.\, p(B = b)$

  $= 1/4 \times 4/10 + 3/4 \times 6/10 = 11/20$

- Since $p(F = a) + p(F = o) = 1$, Probability of choosing an orange becomes:

  $p(F = o) = 1 - p(F = a) = 1\text{-}11/20 = 9/20$

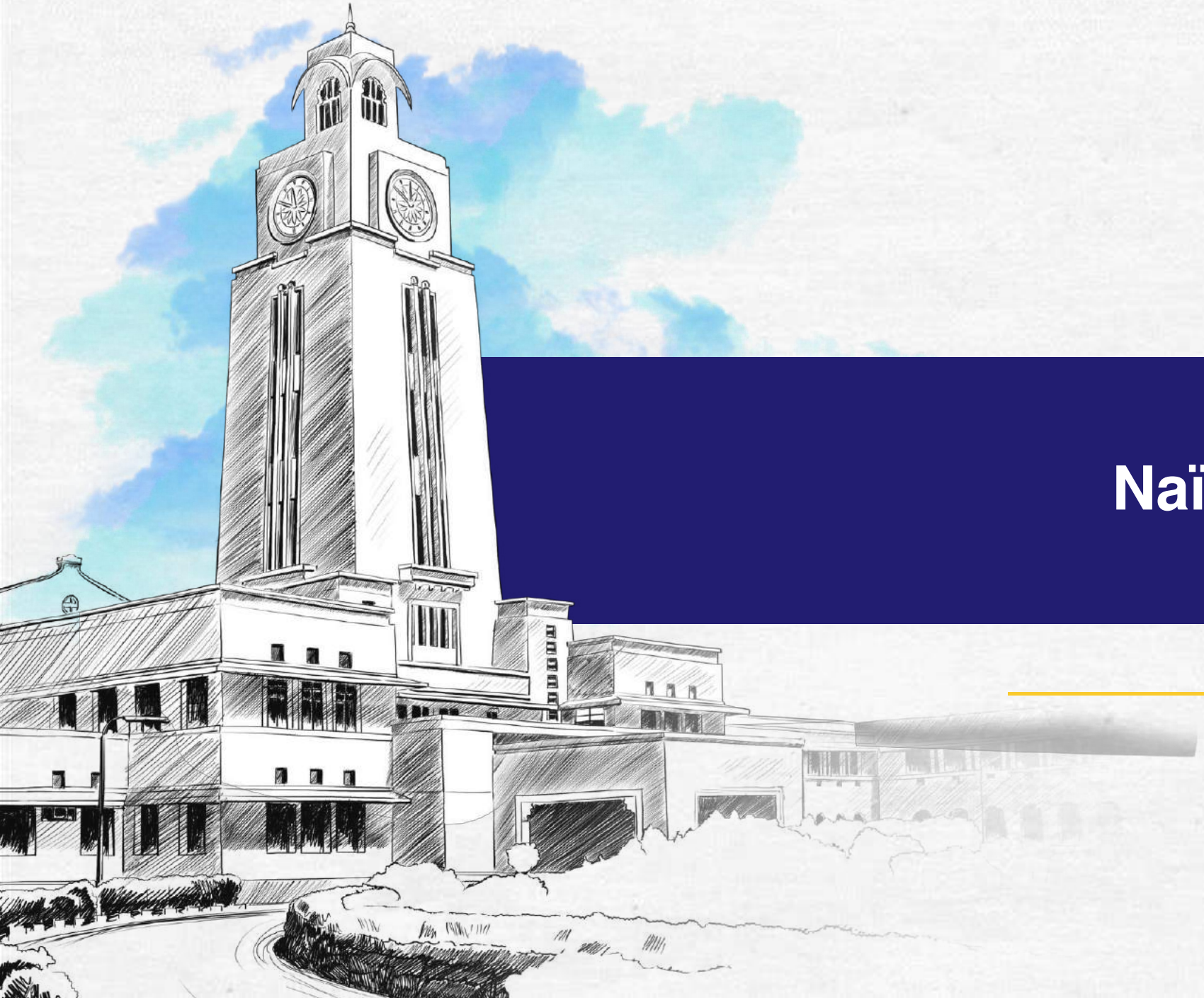
- **Now, if the chosen fruit is orange, using Bayes theorem, the probability that it came from red box can be determined as:**

  $\mathbf{p(B = r \mid F = o) = p(F = o \mid B = r).p(B = r) / p(F = o) = 3/4 \times 4/10 / (9/20) = 2/3}$

Thank You!

In our next session:

Naïve Bayes Classifier



BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Naïve Bayes Classifier

Dr. Chetana Gavankar

In this segment

- Bayes' Theorem
- Conditional Independence
- Naïve Bayes Classifier – Derivation
- Naïve Bayer Classification – Illustrative Examples

Naïve Bayes Classifier

- Bayesian learning algorithms calculate explicit probabilities for each class.
- Naive Bayes classifier, are among the most practical approaches to certain types of learning problems
- For example: Problem of learning to classify text documents such as electronic news articles. For such learning tasks, the naive Bayes classifier is among the most effective algorithms.

Independence

- Two events A and B are independent if and only if $P(A \cap B)=P(A)P(B)$.
- If two events A and B are independent then $P(A|B)=P(A)$

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

For example, the probability that a fair coin shows "heads" after being flipped is 1/2.
What if we knew the day was Tuesday? Does this change the probability of getting "heads?“
The probability of getting "heads," given that it's a Tuesday, is still 1/2.

So the result of a coin flip and the day being Tuesday are independent events; knowing it was a Tuesday didn't change the probability of getting "heads."

Independence of events

In general, for n events A1,A2,⋯,An to be independent we must have

$$P(A_i \cap A_j) = P(A_i)P(A_j), \text{ for all distinct } i, j \in \{1, 2, \cdots, n\};$$

$$P(A_i \cap A_j \cap A_k) = P(A_i)P(A_j)P(A_k), \text{ for all distinct } i, j, k \in \{1, 2, \cdots, n\};$$

Conditional Probability

- Conditional Probability is a measure of the probability of an event given that another event has already occurred.
- Let the event A assumed to have occurred.
- If the event of interest is B and is dependent on occurrence of A
- This probability is written *P(B|A)* notation for the *probability of B given A*.
- The joint probability of event A and B occurring is:

$$P(A \cap B) = P(A) \times P(B|A)$$

Conditional Probability Example

- **Event A** is drawing a King first, and **Event B** is drawing a King second.
- For the first card the chance of drawing a King is 4 out of 52 (there are 4 Kings in a deck of 52 cards):

  $P(A) = 4/52$

- But after removing a King from the deck the probability of the 2nd card drawn is **less** likely to be a King (only 3 of the 51 cards left are Kings):

  $P(B|A) = 3/51$

- And so:

  $\mathbf{P(A \text{ and } B) = P(A) \times P(B|A)} = (4/52) \times (3/51) = 12/2652 = \mathbf{1/221}$

- So the chance of getting 2 Kings is 1 in 221, or about 0.5%

Law of Total Probability

If $B_1, B_2, B_3, \cdots$ is a partition of the sample space S, then for any event A we have

$$P(A) = \sum_i P(A \cap B_i) = \sum_i P(A|B_i)P(B_i).$$

Example: A person has undertaken a mining job. The probabilities of completion of job on time with and without rain are 0.42 and 0.90 respectively. If the probability that it will rain is 0.45, then determine the probability that the mining job will be completed on time.

Solution:

Let A be the event that the mining job will be completed on time and B be the event that it rains.

P(B) = 0.45,

P(no rain) = P(B′) = 1 − P(B) = 1 − 0.45 = 0.55

P(A|B) = 0.42, and P(A|B′) = 0.90

Since, events B and B′ form partitions of the sample space S, by total probability theorem, we have

P(A) = P(B) P(A|B) + P(B′) P(A|B′)

=0.45 × 0.42 + 0.55 × 0.9= 0.189 + 0.495 = 0.684

So, the probability that the job will be completed on time is 0.684

Bayes' theorem

Using Conditional Probability

$$P(A|B)P(B) = P(A,B) = P(B|A)P(A)$$

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

Bayes' Rule

- For any two events A and B, where $P(A) \neq 0$, we have

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

- If $B_1, B_2, B_3, \cdots$ form a partition of the sample space S, and A is any event with $P(A) \neq 0$, we have

$$P(B_j|A) = \frac{P(A|B_j)P(B_j)}{\sum_i P(A|B_i)P(B_i)}.$$

Bayes' theorem example

Given:

- A doctor knows that meningitis causes stiff neck 50% of the time
- Prior probability of any patient having meningitis is 1/50,000
- Prior probability of any patient having stiff neck is 1/20

If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Conditional independence

Definition: X is conditionally independent of Y given Z, if the probability distribution governing X is independent of the value of Y, given the value of Z

$$(\forall i, j, k)\ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z_k)$$

$$P(X|Y, Z) = P(X|Z)$$

Example:

$$P(\text{Thunder}|\text{Rain}, \text{Lightning}) = P(\text{Thunder}|\text{Lightning})$$

Bayesian Classifiers

Let the attributes $X_1, X_2,\ldots,X_n$ and class labels $Y_1, Y_2,..Y_m$ be random variables

- Given a record with attributes $(X_1, X_2,\ldots,X_n)$
  - Goal is to predict class $Y=Y_k$
  - We want to find the value of Y that maximizes $P(Y| X_1, X_2,..,X_n)$

- Can we estimate $P(Y| X_1, X_2,..,X_n)$ directly from data?

Bayesian Classifiers

- Compute the posterior probability $\mathbf{P(Y| X_1, X_2,.., X_n)}$ for all values of Y using the Bayes theorem

$$\mathbf{P(Y| X_1, X_2,.., X_n) = \frac{P(X1, X2,..,Xn \mid Y)P(Y)}{P(X1, X2,..,Xn)}}$$

- Choose value of C that maximizes $\mathbf{P(Y| X_1, X_2,.., X_n)}$
- Equivalent to choosing value of Y that maximizes $\mathbf{P(X_1, X_2,.., X_n \mid Y)P(Y)}$

How to estimate $\mathbf{P(X_1, X_2,.., Xn \mid Y)}$?

Chain Rule for Conditional Probability

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$

Now we can extend this formula to three or more events:

$$P(A \cap B \cap C) = P\big(A \cap (B \cap C)\big) = P(A)P(B \cap C|A)$$

$$P(B \cap C) = P(B)P(C|B).$$

Conditioning both sides on A, we obtain

$$P(B \cap C|A) = P(B|A)P(C|A, B)$$

Combining Equation 1.6 and 1.7 we obtain the following chain rule:

Chain rule for conditional probability:

$$P(A_1 \cap A_2 \cap \cdots \cap A_n) = P(A_1)P(A_2|A_1)P(A_3|A_2, A_1) \cdots P(A_n|A_{n-1}A_{n-2} \cdots A_1)$$

https://www.probabilitycourse.com/chapter1/1_4_0_conditional_probability.php

Applying conditional independence

Naïve Bayes assumes X_i are conditionally independent given Y

e.g., $P(X_1|X_2, Y) = P(X_1|Y)$

$$P(X_1, X_2|Y) = P(X_1|X_2, Y)P(X_2|Y) = P(X_1|Y)P(X_2|Y)$$

General form: $P(X_1, \cdots, X_n|Y) = \prod_{j=1}^{n} P(X_j|Y)$

How many parameters to describe $P(X_1, \cdots, X_n|Y)$?

Naïve Bayes Independence assumption

Assumption:

$$P(X_1, \cdots, X_n|Y) = \prod_{j=1}^{n} P(X_j|Y)$$

i.e., X_i and X_j are conditionally independent given Y for $i \neq j$

Naïve Bayes Classifier

- Bayes rule:

$$P(Y = y_k | X_1, \cdots, X_n) = \frac{P(Y = y_k) P(X_1, \cdots, X_n | Y = y_k)}{\sum_j P(Y = y_j) P(X_1, \cdots, X_n | Y = y_j)}$$

- Assume conditional independence among X_i's:

$$P(Y = y_k | X_1, \cdots, X_n) = \frac{P(Y = y_k) \Pi_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \Pi_i P(X_i | Y = y_j)}$$

- Pick the most probable (MAP) Y

$$\hat{Y} \leftarrow \underset{y_k}{\operatorname{argmax}} P(Y = y_k) \Pi_i P(X_i | Y = y_k)$$

Prior Probability

MLE

Naïve Bayes Classifier

- Assume independence among attributes X_i when class is given:
  - $P(X_1, X_2, \ldots, X_d | Y_j) = P(X_1| Y_j)\, P(X_2| Y_j) \ldots P(X_d| Y_j)$
  - Now we can estimate $P(X_i| Y_j)$ for all X_i and Y_j combinations from the training data
  - New point is classified to Y_j if $P(Y_j) \prod P(X_i| Y_j)$ is maximal.

Slide adopted from "Introduction to Data mining" Vipin Kumar

Naïve Bayes Classifier

- Start with some belief about the system, called a prior.
- Then we obtain some data and use it to update our belief.
- The outcome is called a posterior.

Naïve Bayes Classifier – Example 1

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

A: attributes

M: mammals

N: non-mammals

$$P(A \mid M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A \mid N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A \mid M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A \mid N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

Naïve Bayes Classifier – Example 2

- Use case is to predict the chances of playing tennis given a few parameters
  - Weather outlook
  - Temperature
  - Humidity
  - Wind
- Data is given for 14 days

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Naïve Bayes Classifier – Example 2

- New input set to be classified is –

{Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong}

- Output parameter (*PlayTennis*) has 2 classes – Yes, No

- Step-1 Calculate conditional probabilities of each input parameter given each of the classes (likelihood)

P(Outlook=sunny | PlayTennis=yes) = 2 / 9
P(Outlook=sunny | PlayTennis=no) = 3 / 5
P(Temp=cool | PlayTennis=yes) = 3 / 9
P(Temp=cool | PlayTennis=no) = 1 / 5
P(Humidity=high | PlayTennis=yes) = 3 / 9
P(Humidity=high | PlayTennis=no) = 4 / 5
P(Wind=strong | PlayTennis=yes) = 3 / 9
P(Wind=strong | PlayTennis=no) = 3 / 5

- Step-2 Calculate total probability of each of the classes (prior)

P(PlayTennis=yes) = 9/14
P(PlayTennis=no) = 5/14

Naïve Bayes Classifier – Example 2

- Step-3 Calculate probability for each class given the input conditions in new input set (posterior)

P(PlayTennis=yes | Outlook=sunny, Temp=cool, Humidity=high, Wind=strong)

= P(yes). P(sunny | yes). P(cool | yes). P(high | yes). P(strong | yes)
= 9/14 * 2/9 * 3/9 * 3/9 * 3/9 = 0.00529

P(PlayTennis=no | Outlook=sunny, Temp=cool, Humidity=high, Wind=strong)
= P(no). P(sunny | no). P(cool | no). P(high | no). P(strong | no)
= 5/14 * 3/5 * 1/5 * 4/5 * 3/5 = 0.02057

Using Naïve Bayes classifier, argmax (P) = no, which mean the classier assign PlayTennis = No

Discrete and Continuous value attributes

To compute, $P(x_k|C_i)$

- A_k is categorical:

$$P(x_k|C_i) = \frac{\text{the number of tuples of class } C_i \text{ in } D \text{ having the value } x_k \text{ for } A_k}{\text{the number of tuples of class } C_i \text{ in } D.}$$

- A_k is continuous:

A continuous-valued attribute is typically assumed to have a Gaussian distribution with a mean μ and standard deviation σ

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}).$$

Estimate Probabilities from Continuous Data

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:
  - One for each (A_i,c_i) pair

$$P(A_i \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(A_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 3179.16

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}(56.38)} e^{-\frac{(120-110)^2}{2(3179)}} = 0.0069$$

Example of Naïve Bayes Classifier

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

- $P(X|\text{Class=No}) = P(\text{Refund=No}|\text{Class=No}) \times P(\text{Married}|\text{ Class=No}) \times P(\text{Income=120K}|\text{ Class=No})$
  $= 4/7 \times 4/7 \times 0.0069 = 0.0023$

- $P(X|\text{Class=Yes}) = P(\text{Refund=No}|\text{ Class=Yes}) \times P(\text{Married}|\text{ Class=Yes}) \times P(\text{Income=120K}|\text{ Class=Yes})$
  $= 1 \times 0 \times 1.2 \times 10^{-9} = 0$

Since $P(X|\text{No})P(\text{No}) > P(X|\text{Yes})P(\text{Yes})$

Therefore $P(\text{No}|X) > P(\text{Yes}|X)$

=> Class = No

Thank You!

In our next session:

Naïve Bayes Classifier –Issues, Applications

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Naïve Bayes –Interpretability

Dr. Chetana Gavankar

In this segment

Naïve Bayes – Interpretability

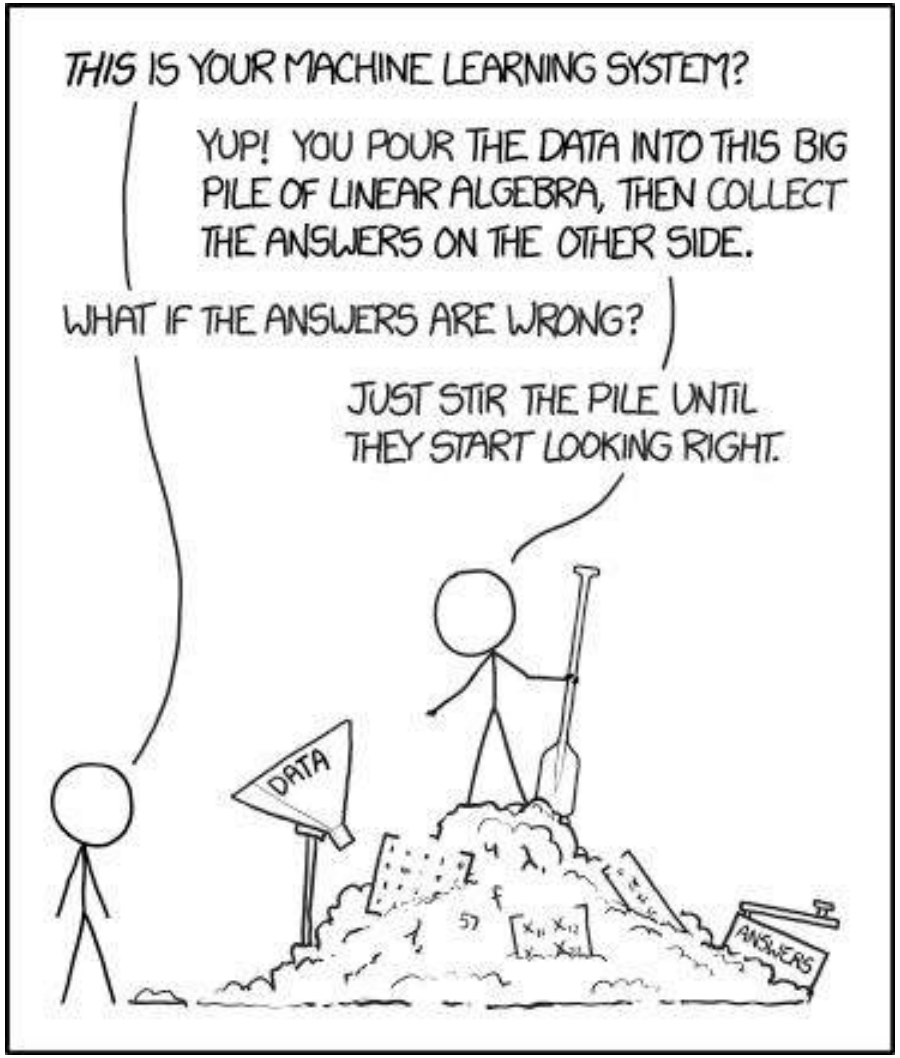
- Interpretability of Machine learning algorithm
- Interpretability of Naïve Bayes Classifier

Interpretability of Machine Learning Model

- Do we just want to know what is predicted?

OR why the prediction was made

- how effective some drug will be for a patient?
- Model interpretability is necessary to verify that what the model is doing is in line with what you expect
- As a data scientist you are often concerned with fine-tuning models to obtain optimal performance
- Interpretability of data and machine learning models is one of those aspects that is critical in the practical ‘usefulness’ of a data science pipeline and it ensures that the model is aligned with the problem you want to solve



Why it Interpretability is important?

The higher the **interpretability** of a **machine learning** model, the easier it is for someone to comprehend why certain decisions or predictions have been made.

A model is better **interpretable** than another model if its decisions are easier for a human to comprehend than decisions from the other model.

- Identify and mitigate bias - Fairness (gender bias)
- Accounting for the context of the problem
- Improving generalization and performance
- Ethical and legal reasons. (Health domain)

https://towardsdatascience.com/interpretability-in-machine-learning-70c30694a05f

Interpretability of Naïve Bayes Classifier

- Interpretability of NB classifier comes from its conditional independence assumption

- We can see exactly how much each parameter is impacting on the final result

Thank You!

In our next session:

Naïve Bayes Classifier- Implementation

Some references for more examples

- Movie Review: https://www.youtube.com/watch?time_continue=16&v=EGKeC2S44Rs

- Spam mails by Prof. Andrew Ng:

https://www.youtube.com/watch?v=z5UQyCESW64

https://www.youtube.com/watch?v=NFd0ZQk5bR4

- NLP by Prof. Dan Jurafsky, Stanford: https://www.youtube.com/watch?v=Fmu65a0v6Sw

- https://towardsdatascience.com/machine-learning-workflow-on-diabetes-data-part-01-573864fcc6b8
- https://www.youtube.com/watch?time_continue=509&v=r1in0YNetG8&feature=emb_logo

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Naïve Bayes – Issues, Applications

Dr. Chetana Gavankar

In this segment

Naïve Bayes – Advantages, Interpretability

- Issues with Naïve Bayes Classifier
- Laplace Smoothing
- Naïve Bayes Classifier : Generative model
- Advantages of Naïve Bayes Classifier

Issues with Naïve Bayes Classifier

Consider the table with Tid = 6 deleted

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Given X = (Refund = Yes, Divorced, 120K)

$P(X \mid No) = 2/6 \times 0 \times 0.0083 = 0$

$P(X \mid Yes) = 0 \times 1/3 \times 1.2 \times 10^{-9} = 0$

Naïve Bayes Classifier:

P(Refund = Yes | No) = 2/6
P(Refund = No | No) = 4/6
P(Refund = Yes | Yes) = 0
P(Refund = No | Yes) = 1
P(Marital Status = Single | No) = 2/6
P(Marital Status = Divorced | No) = 0
P(Marital Status = Married | No) = 4/6
P(Marital Status = Single | Yes) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
P(Marital Status = Married | Yes) = 0/3
For Taxable Income:
If class = No: sample mean = 91
sample variance = 685
If class = No: sample mean = 90
sample variance = 25

Naïve Bayes will not be able to classify X as Yes or No!

Slide adopted from "Introduction to Data mining" Vipin Kumar

Issues with Naïve Bayes Classifier

- If one of the conditional probabilities is zero, then the entire expression becomes zero
- Need to use other estimates of conditional probabilities than simple fractions
- Probability estimation:

$$\text{Original} : P(A_i \mid C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace} : P(A_i \mid C) = \frac{N_{ic} + 1}{N_c + \mathrm{T}}$$

$$\text{m-estimate} : P(A_i \mid C) = \frac{N_{ic} + mp}{N_c + m}$$

T: number of unique words across all documents

p: prior probability of the class

m: parameter

N_c: number of instances in the class
N_{ic}: number of instances having attribute value A_i in class c

Slide adopted from "Introduction to Data mining" Vipin Kumar

Case Study – Text Classification

- Naïve Bayes is commonly used for text classification
- For a document with k terms $d = (t_1, \dots, t_k)$

Fraction of documents in c

$$P(c|d) = P(c)P(d|c) = P(c)\prod_{t_i \in d} P(t_i|c)$$

- $P(t_i|c)$= Fraction of terms from all documents in c that are t_i.

Number of times t_i appears in some document in c

$$P(t_i|c) = \frac{N_{ic} + 1}{N_c + T}$$

Laplace Smoothing

Total number of terms in all documents in c

Number of unique words (vocabulary size)

- Easy to implement and works relatively well

Case Study – Text Classification

Text	Tag
“A great game”	Sports
“The election was over”	Not sports
“Very clean match”	Sports
“A clean but forgettable game”	Sports
“It was a close election”	Not sports

Which tag does the sentence *A very close game* belong to?
P(sports| *A very close game*)

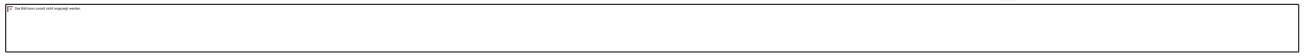
Bag of words i.e. word frequencies without considering order

Using Bayes Theorem:

P(sports| *A very close game*)

$$= \frac{\mathbf{P(}\textit{A very close game}\mathbf{|\ sports)\ P(sports)}}{\mathbf{P(}\textit{A very close game}\mathbf{)}}$$

We assume that every word in a sentence is **independent** of the other ones



“close” doesn’t appear in sentences of sports tag, So P(close | sports) = 0, which makes product 0

Laplace smoothing

- Laplace smoothing: we add 1 or in general constant k to every count so it's never zero.
- To balance this, we add the number of possible words to the divisor, so the division will never be greater than 1
- In our case, the possible words are ['a', 'great', 'very', 'over', 'it', 'but', 'game', 'election', 'clean', 'close', 'the', 'was', 'forgettable', 'match'].

Apply Laplace Smoothing

Word	P(word | Sports)	P(word | Not Sports)
a	2+1 / 11+14	1+1 / 9+14
very	1+1 / 11+14	0+1 / 9+14
close	0+1 / 11+14	1+1 / 9+14
game	2+1 / 11+14	0+1 / 9+14

$$P(a|Sports) \times P(very|Sports) \times P(close|Sports) \times P(game|Sports) \times P(Sports)$$
$$= 2.76 \times 10^{-5}$$
$$= 0.0000276$$

$$P(a|Not\ Sports) \times P(very|Not\ Sports) \times P(close|Not\ Sports) \times P(game|Not\ Sports) \times P(Not\ Sports)$$
$$= 0.572 \times 10^{-5}$$
$$= 0.00000572$$

Naïve Bayes Classifier: Generative Model

Naïve Bayes Classifier is a Generative Model

- Naïve Bayes classifier uses likelihood and prior probability to calculate conditional probability of the class
- Likelihood is based on joint probability, which is the core principle of probabilistic generative model
- Naïve Bayes simplifies the calculation of likelihood by the assumption of conditional independence among input parameters
- Each parameter's likelihood is determined using joint probability of the input parameter and the output label

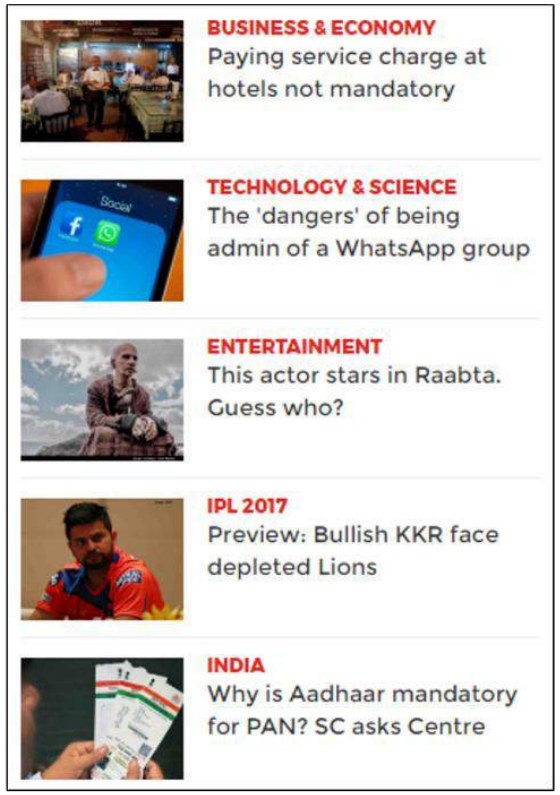
Naïve Bayes – Advantages

Advantages of Naïve Bayes Classifier

- Algorithm is simple to implement and fast
- If conditional independence holds, it will converge quickly than other methods
- Even in cases where conditional independence doesn’t hold, its results are quite acceptable
- Needs less training data (due to conditional independence assumption)
- Highly scalable, scales linearly with the number of predictors and data points
- Can be used for both binary and multi-class classification problems
- Handles continuous and discrete data
- Not sensitive to irrelevant features
- Doesn’t overfit the data due to small model size
- Handles missing values well (we can drop the entire feature and still not impact the result if the missing values are significant in population)

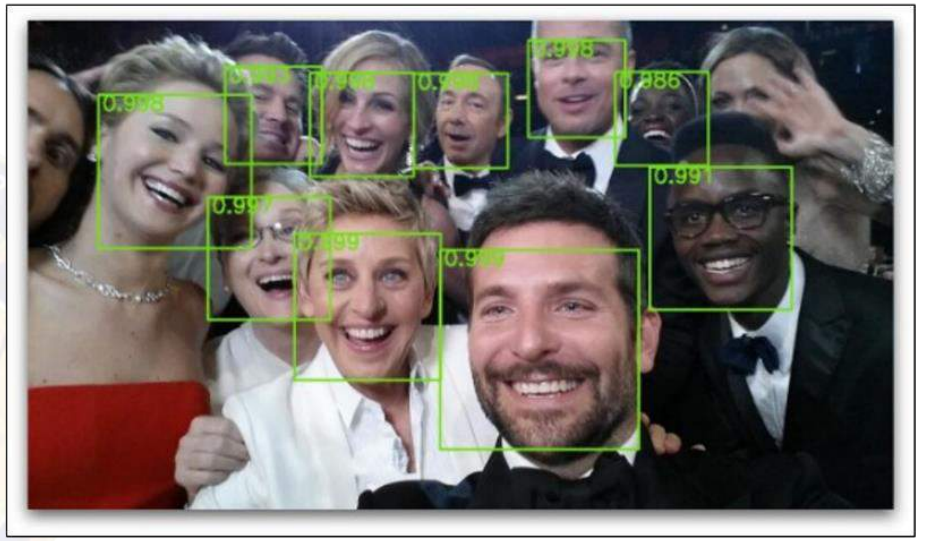
Naïve Bayes Classifier Applications

Categorizing News

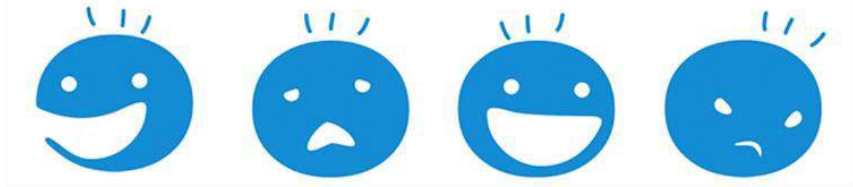


Email Spam Detection

Face Recognition



Sentiment Analysis



Thank You!

In our next session:

Naïve Bayes Classifier- Interpretability

Some references for more examples

- Movie Review: https://www.youtube.com/watch?time_continue=16&v=EGKeC2S44Rs

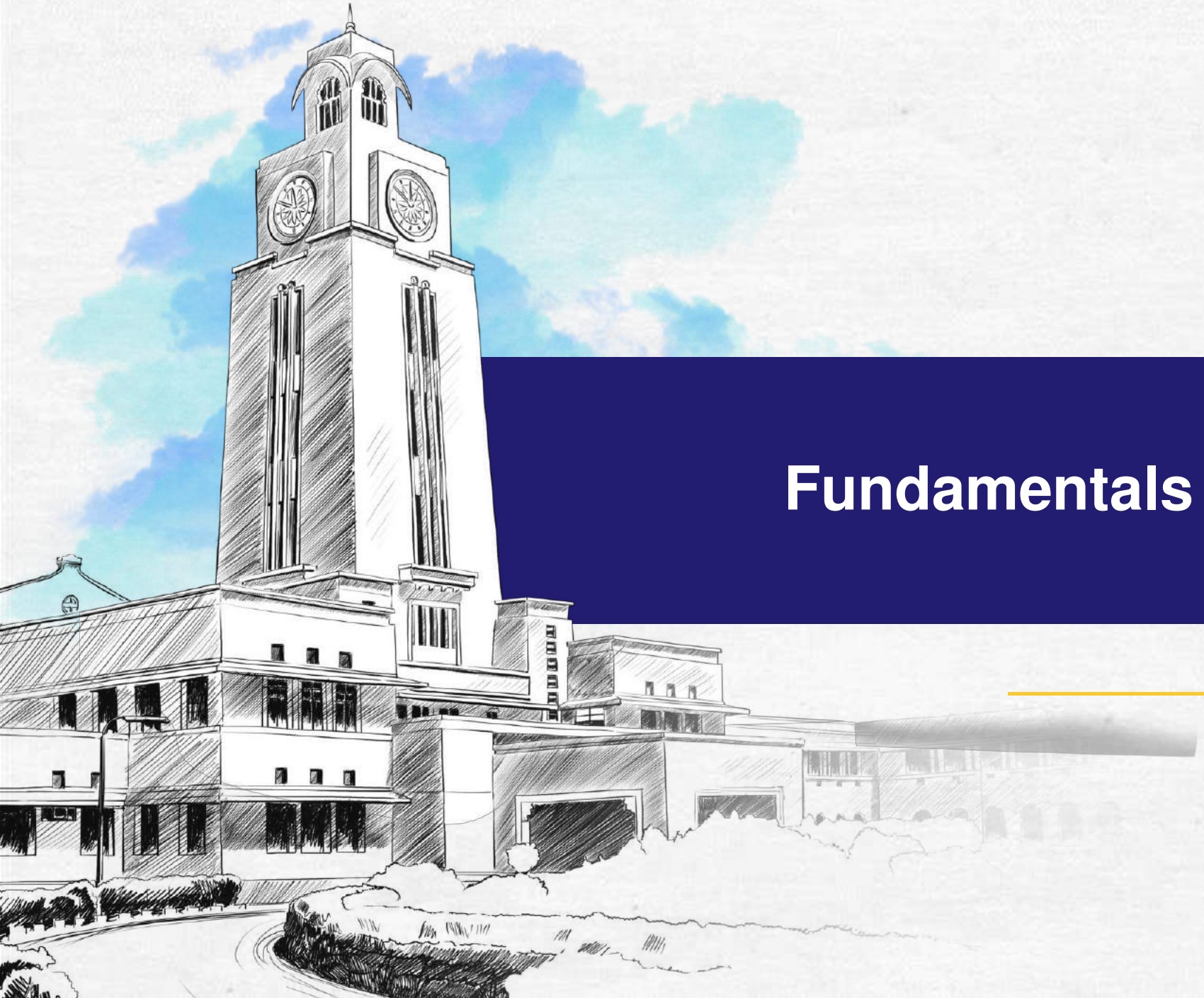
- Spam mails by Prof. Andrew Ng:

https://www.youtube.com/watch?v=z5UQyCESW64

https://www.youtube.com/watch?v=NFd0ZQk5bR4

- NLP by Prof. Dan Jurafsky, Stanford: https://www.youtube.com/watch?v=Fmu65a0v6Sw

- https://towardsdatascience.com/machine-learning-workflow-on-diabetes-data-part-01-573864fcc6b8
- https://www.youtube.com/watch?time_continue=509&v=r1in0YNetG8&feature=emb_logo



BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Fundamentals - Sigmoid Function

Dr. Chetana Gavankar

In this segment

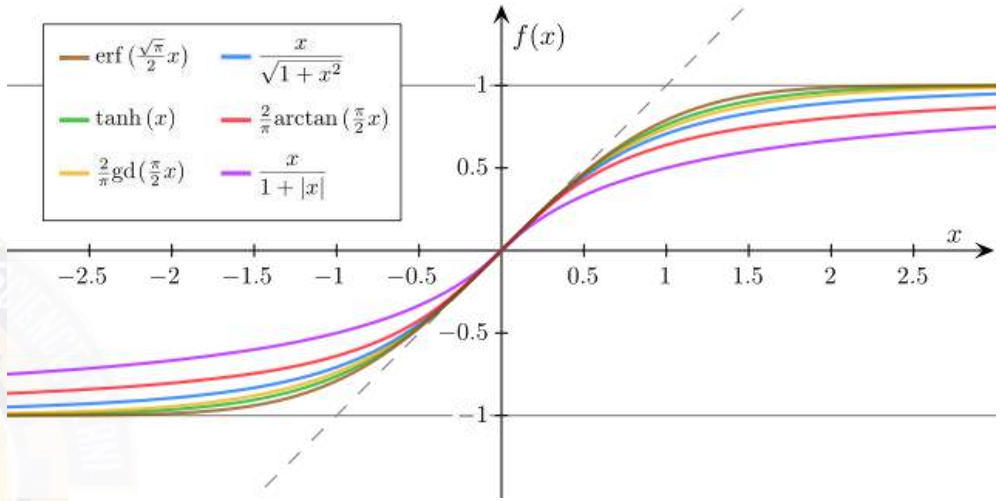
Fundamentals – Sigmoid Function

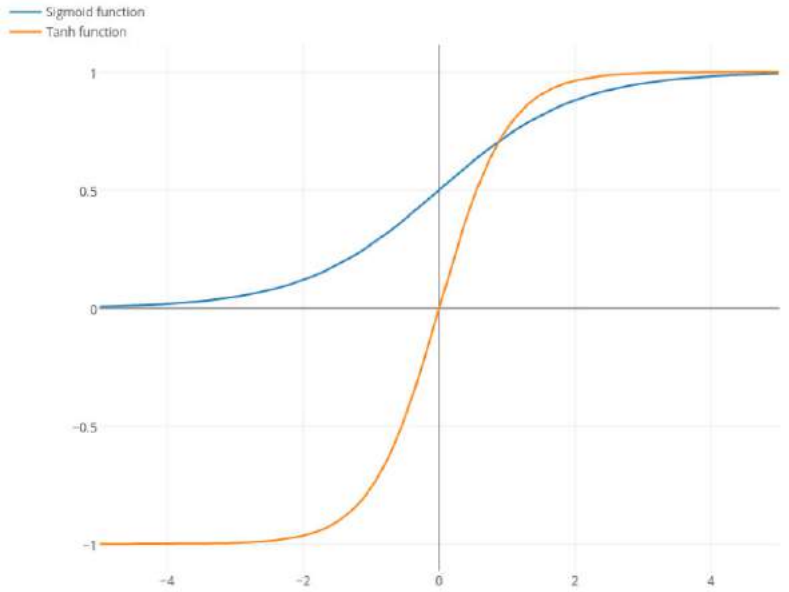
- Sigmoid Function – Overview
- Sigmoid Function – Significance
- Sigmoid Function – Derivative

Fundamentals – Sigmoid Function

Sigmoid Function – Overview

- Any function that takes the shape of ‘S’ is called a sigmoid function
- Maps input horizon to a bounded, symmetrical output range
- Examples:
  - Logistic Function
  - Hyperbolic Tangent
  - Arctangent function
  - Gudermannian function
  - Error function
  - Generalized Logistic function
  - …

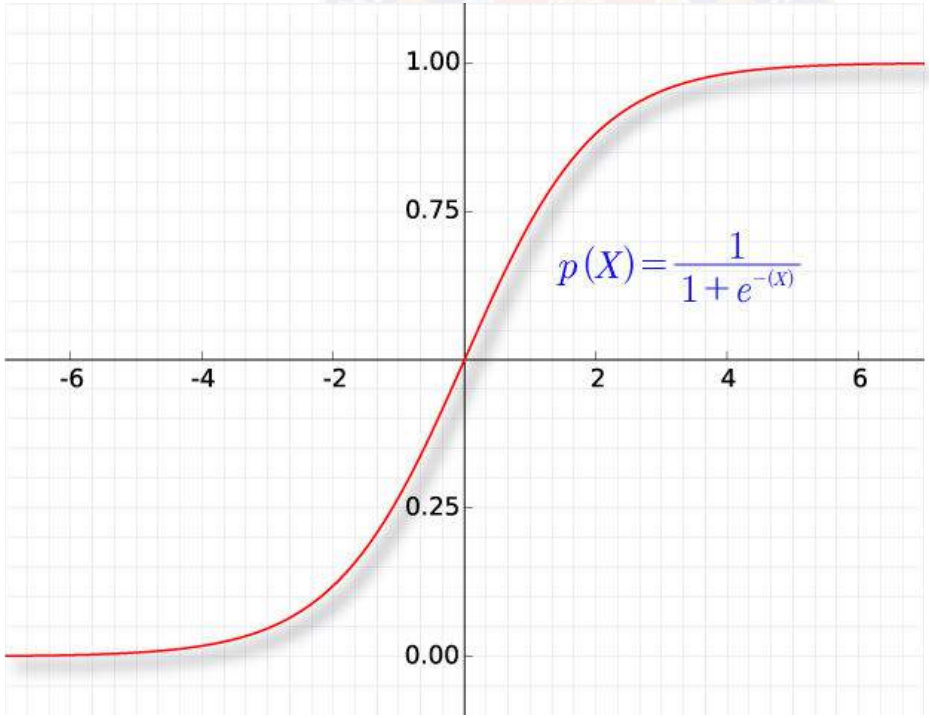




Fundamentals – Sigmoid Function

Sigmoid Function – Significance

- Known as ‘squashing function’
- Logistic function (a variant of sigmoid) bounds the output between 0 and 1
- Conditional probability of a random variable can be expressed as a sigmoid (logistic) function



Derivative of Sigmoid Function

- Maximum likelihood to determine the parameters of the logistic regression model.
- To do this, we shall make use of the derivative of the logistic sigmoid function
- Use any algorithm like the gradient descent algorithm to minimize cost function by using derivative

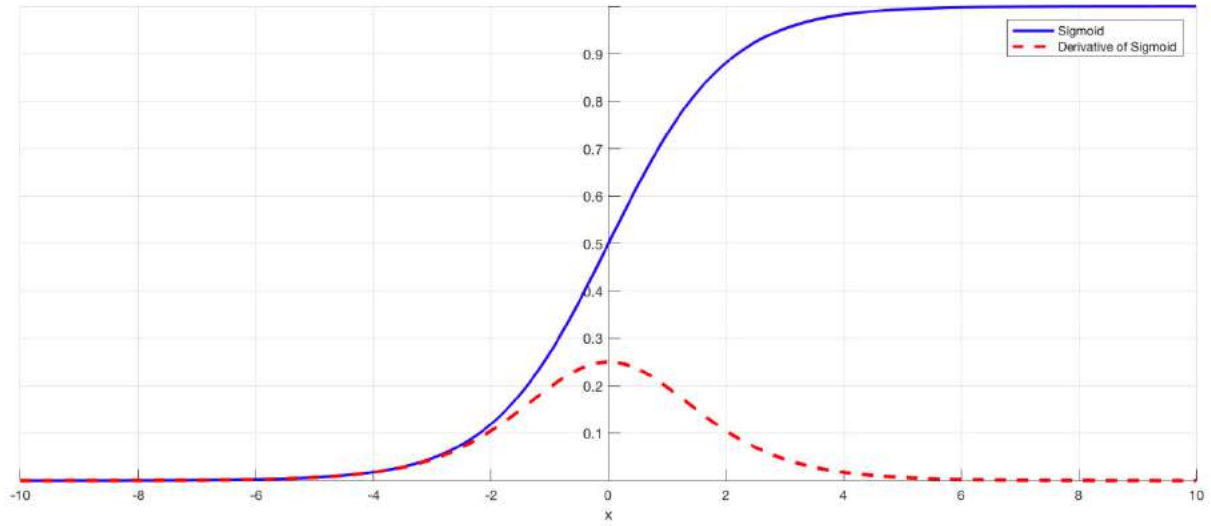
https://towardsdatascience.com/derivative-of-the-sigmoid-function-536880cf918e

Fundamentals – Sigmoid Function

Derivative of Sigmoid Function

$$\sigma(x) = \frac{1}{1+e^{-x}}.$$

$$\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+e^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right) \\
&= \sigma(x) \cdot (1-\sigma(x))
\end{aligned}$$



Derivative of sigmoid function is significant in the sense that it represents a bell shaped probability distribution function of the data

Fundamentals – Sigmoid Function

Logistic Function for Classification - Overview

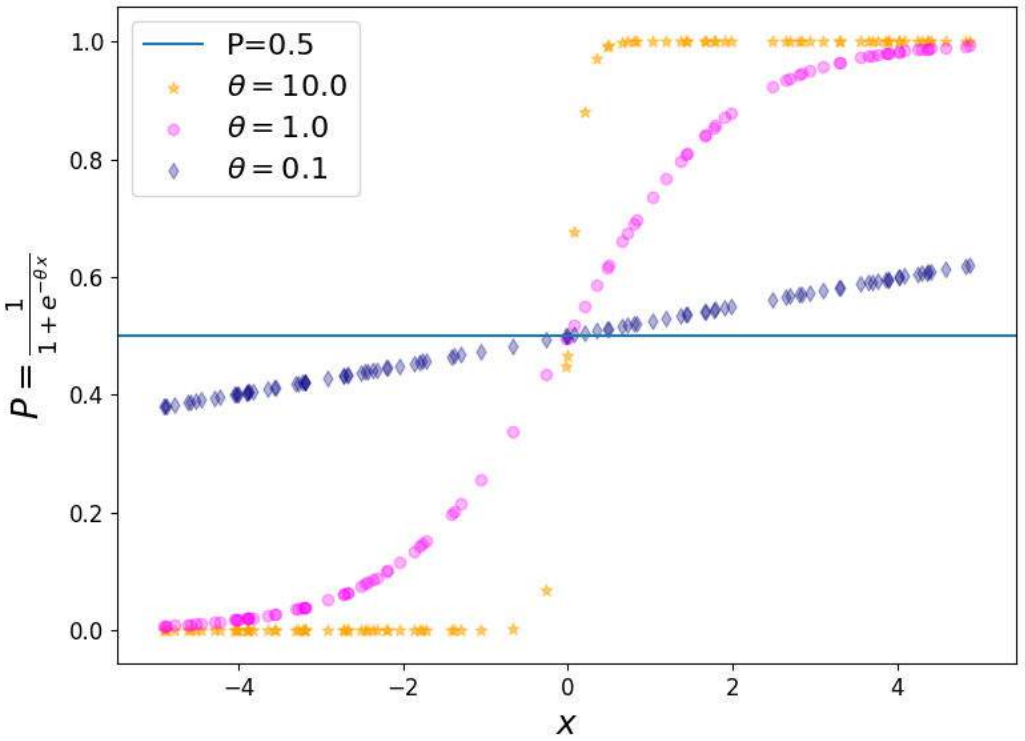
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$P(Y = 1 \mid \mathbf{X} = \mathbf{x}) = \sigma(z) \text{ where } z = \theta_0 + \sum_{i=1}^{m} \theta_i x_i$$

$$\theta^T \mathbf{x} = \sum_{i=1}^{m} \theta_i x_i = \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_m x_m$$

$$P(Y = 1 \mid \mathbf{X} = \mathbf{x}) = \sigma(\theta^T \mathbf{x})$$

$$P(Y = 0 \mid \mathbf{X} = \mathbf{x}) = 1 - \sigma(\theta^T \mathbf{x})$$



Thank You!

In our next session:

Foundations - Maximum Likelihood Estimation

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Maximum Likelihood Estimation

Dr. Chetana Gavankar

In this segment

Foundations - Maximum Likelihood Estimation

- Maximum Likelihood Estimation – Overview
- Maximum Likelihood Estimation – Example

MLE and MAP

- Both Maximum Likelihood Estimation (MLE) and Maximum A Posterior (MAP) are used to estimate parameters for a distribution.
- MLE is a special case of Max-A-Posteriori discussed in Naïve Bayes

Recap:

- Bayes rule:

$$P(Y = y_k | X_1, \cdots, X_n) = \frac{P(Y = y_k) P(X_1, \cdots, X_n | Y = y_k)}{\sum_j P(Y = y_j) P(X_1, \cdots, X_n | Y = y_j)}$$

- Assume conditional independence among X_i's:

$$P(Y = y_k | X_1, \cdots, X_n) = \frac{P(Y = y_k) \Pi_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \Pi_i P(X_i | Y = y_j)}$$

- Pick the most probable (MAP) Y

$$\hat{Y} \leftarrow \underset{y_k}{\operatorname{argmax}} P(Y = y_k) \Pi_i P(X_i | Y = y_k)$$

MLE is a special case of MAP, when prior follows a uniform distribution.

Foundations - Maximum Likelihood Estimation

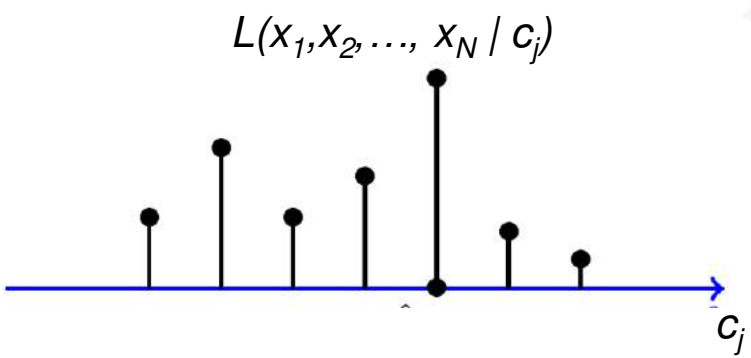
Maximum Likelihood Estimation - Overview

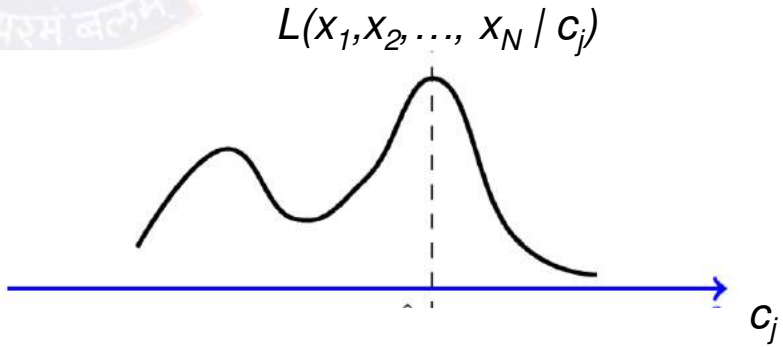
- For a case of N-input parameters $X = \{x_1, x_2, \ldots, x_N\}$ and output variable of K classes $C = \{c_1, c_2, \ldots, c_K\}$, likelihood is $P(x_1, x_2, \ldots, x_N \mid c_j)$
- If the input parameters are discrete, the likelihood function is defined as the probability mass function of likelihood,

  i.e. $L(x_1, x_2, \ldots, x_N \mid c_j) = P(x_1, x_2, \ldots, x_N \mid c_j)$
- For continuous random variables, the same becomes probability density function

  i.e. $L(x_1, x_2, \ldots, x_N \mid c_j) = f(x_1, x_2, \ldots, x_N \mid c_j)$
- Maximum likelihood estimate (MLE) of C, shown by as C_{ML} is a value of C_j that maximizes the likelihood function $L(x_1, x_2, \ldots, x_N \mid c_j)$

  i.e. $C_{ML} = \text{argmax}_{C_j} L(x_1, x_2, \ldots, x_N \mid c_j)$





Foundations - Maximum Likelihood Estimation

Maximum Likelihood Estimation - Example

A bag has 3 balls (red or blue). If 4 random pick and replacement events results in the following results – blue, red, blue, blue; what is the number of blue balls that gives max probability for the observed data ?

$$P_{X_i}(x) = \begin{cases} \frac{\theta}{3} & \text{for } x = 1 \\ 1 - \frac{\theta}{3} & \text{for } x = 0 \end{cases}$$

$$P_{X_1X_2X_3X_4}(1,0,1,1) = \frac{\theta}{3} \cdot \left(1 - \frac{\theta}{3}\right) \cdot \frac{\theta}{3} \cdot \frac{\theta}{3}$$
$$= \left(\frac{\theta}{3}\right)^3 \left(1 - \frac{\theta}{3}\right).$$

θ	$P_{X_1X_2X_3X_4}(1,0,1,1;\theta)$
0	0
1	0.0247
2	0.0988
3	0

Thank You!

In our next session:

Cross entropy error function for Logistic Regression

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Cross Entropy Error function for LR

Dr. Chetana Gavankar

Associate Professor, BITS Pilani
Chetana.gavankar@pilani.bits-pilani.ac.in

In this segment

Logistic Regression - Cross entropy error function

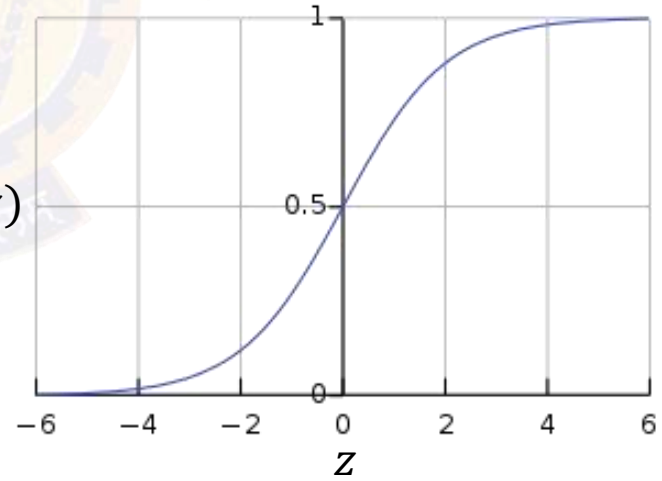
- Logistic Regression Function
- Cross entropy Error function
- Optimal Solution

Logistic Regression Function

- Want $0 \leq h_\theta(x) \leq 1$
- $h_\theta(x) = g(\theta^\top x)$,

  where $g(z) = \frac{1}{1+e^{-z}}$
- Sigmoid function
- Logistic function

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}}$$



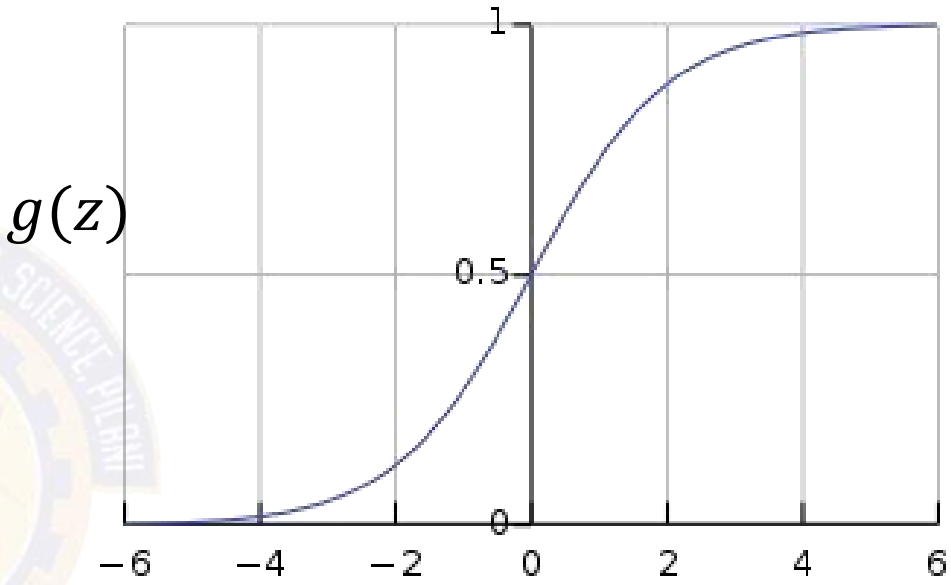
Slide credit: Andrew Ng

Logistic regression Classifier

$$h_\theta(x) = g(\theta^\top x)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$z = \theta^\top x$$



Predict “y = 1” if $h_\theta(x) \geq 0.5$ and $z = \theta^\top x \geq 0$

Predict “y = 0” if $h_\theta(x) < 0.5$ and $z = \theta^\top x < 0$

Slide credit: Andrew Ng

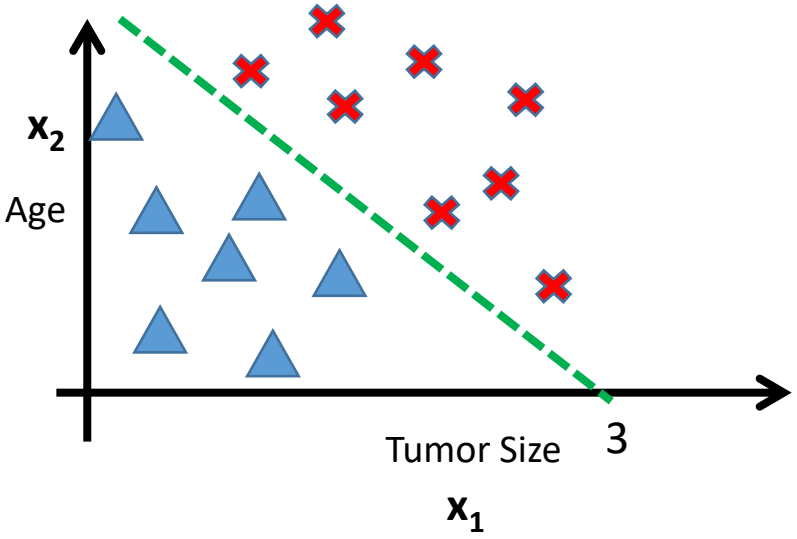
Decision boundary

- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

E.g., $\theta_0 = -3,\ \theta_1 = 1,\ \theta_2 = 1$

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

i.e $x_1 + x_2 \geq 3$



$h_\theta(x) = g(\theta^\top x)$

$z = \theta^\top x$

$g(z) = \dfrac{1}{1 + e^{-z}}$

Slide credit: Andrew Ng

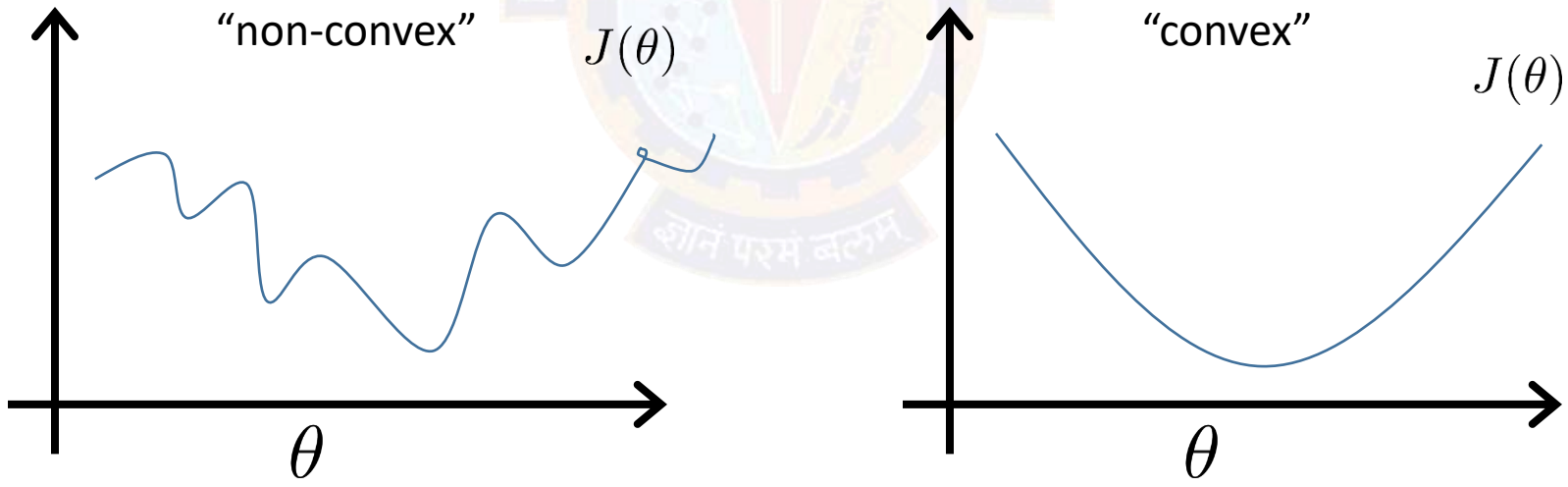
Logistic Regression Error Function

- Our prediction function is non-linear (due to sigmoid transform)
- Squaring this prediction as we do in MSE results in a non-convex function with many local minima.
- If our cost function has many local minimums, gradient descent may not find the optimal global minimum.
- So instead of Mean Squared Error, we use a error/ cost function called Cross-Entropy, also known as Log Loss.

MSE Cost Function

Linear regression: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$

$$\text{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$



Entropy

- It rained heavily in Shillong yesterday
- There was a heavy rainfall in Rajasthan last night.
- The amount of information conveyed by a message is inversely proportional to its probability of occurrence. That is

$$I_k \alpha \frac{1}{p_k}$$

Entropy

- The mathematical operator satisfies above properties is the logarithmic operator. Therefore,

$$I_k = \log_r \frac{1}{p_k} \; units$$

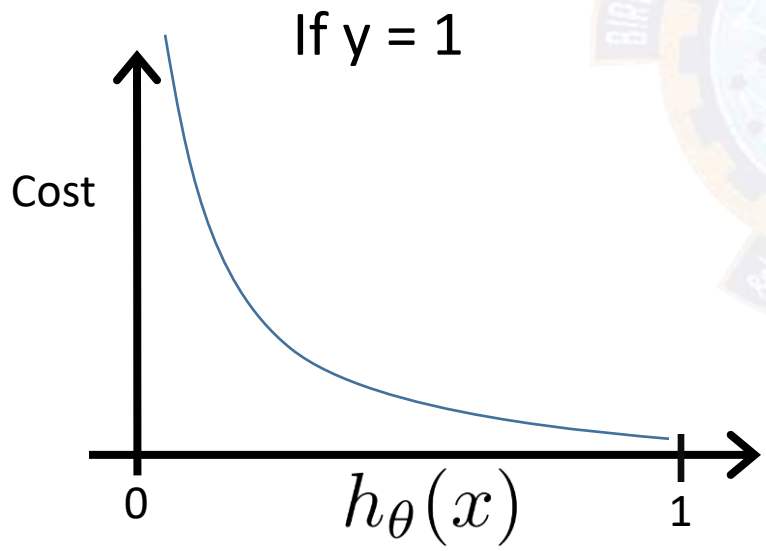
Cross Entropy

- Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1.
- Cross-entropy loss increases as the predicted probability diverges from the actual label.
- So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value.
- A perfect model would have a log loss of 0.

- Cross-entropy loss can be divided into two separate cost functions:
  one for y=1 and
  one for y=0.

Logistic Regression Cost Function (cross entropy)

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



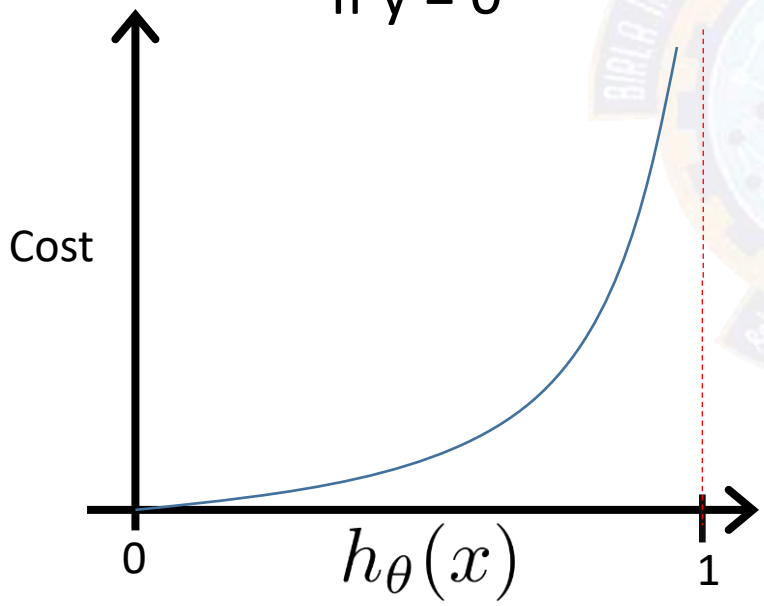
$\text{Cost} = 0$ if $y = 1, h_\theta(x) = 1$
But as $h_\theta(x) \to 0$
$Cost \to \infty$

Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x; \theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.

Logistic Regression Cost Function (cross entropy)

$$\text{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

If y = 0



Cost=0; If y=0 and $h_\Theta(x)=0$

Logistic Regression Cost Function (cross entropy)

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

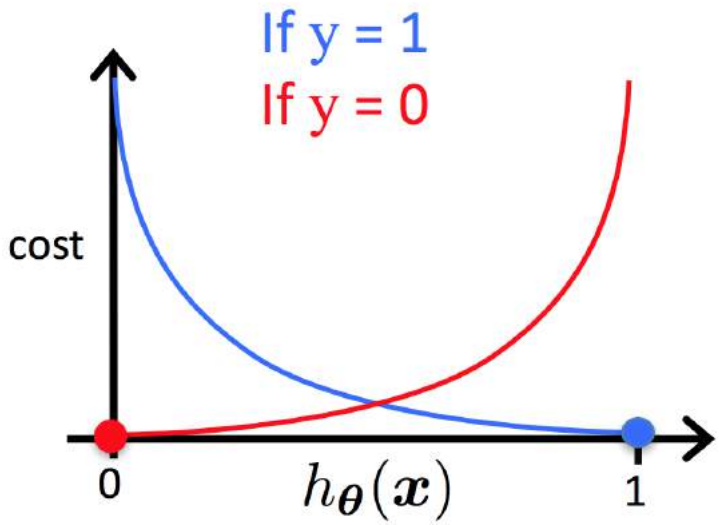
$$= -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

To fit parameters θ : Apply Gradient Descent Algorithm

$$\min_\theta J(\theta)$$

To make a prediction given new x

Output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$



Thank You!

In our next session:

Logistic Regression is Probabilistic Discriminative

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Logistic Regression - Probabilistic Discriminative

Dr. Chetana Gavankar

Associate Professor, BITS Pilani
Chetana.gavankar@pilani.bits-pilani.ac.in

In this segment

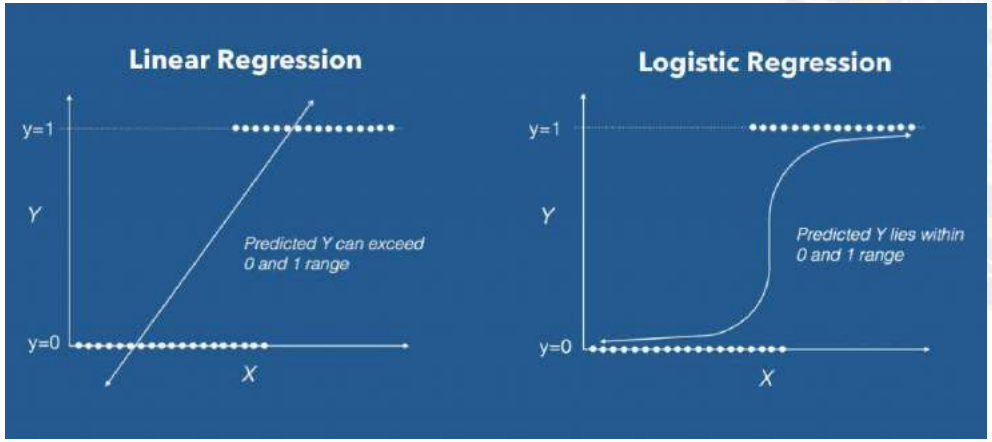
Logistic Regression - Probabilistic Discriminative

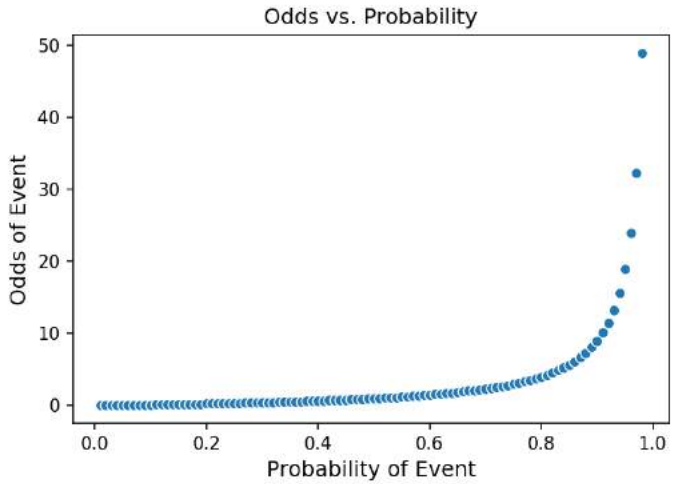
- Logistic Regression - Log of Odds
- Logistic Regression – Illustration
- Probabilistic discriminative versus Probabilistic Generative Models
- Logistic Regression v/s Naïve Bayes Classifier

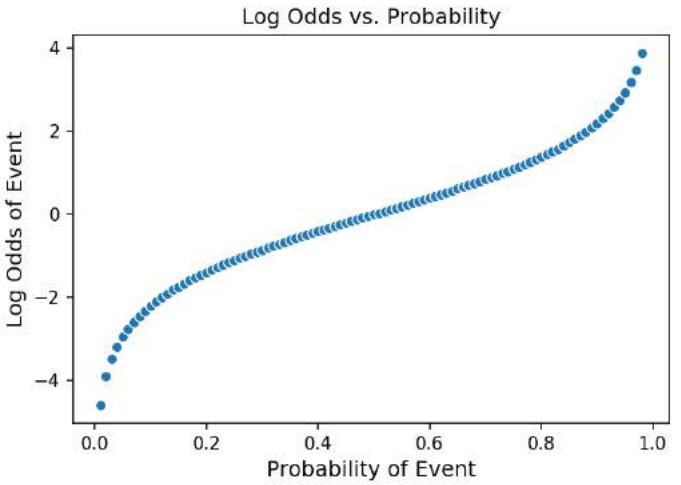
Logistic Regression is Probabilistic Discriminative

Logistic Regression – where it works

- Linear discriminants have limitations on some data sets which are not linearly separable
- Logistic Regression can be used to approximate such datasets, using log of odds







Logistic Regression is Probabilistic Discriminative

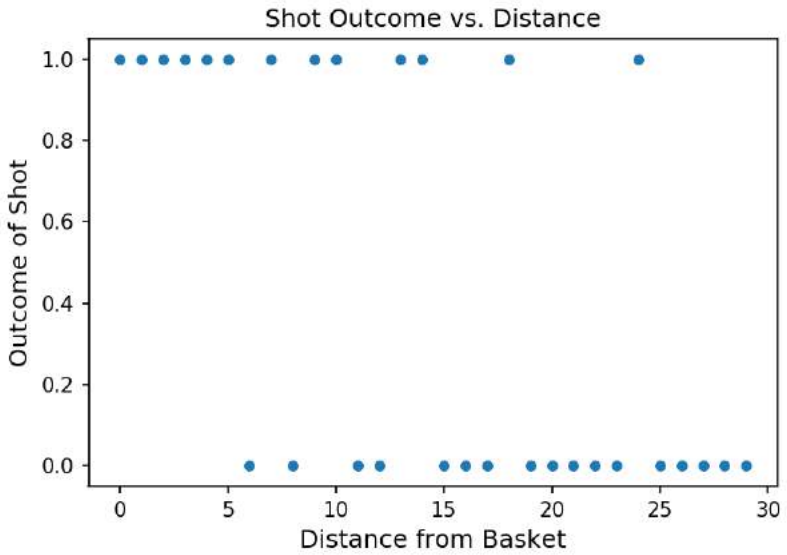
Logistic Regression – Log of Odds

- Log of Odds – $O = \frac{p(X)}{1-p(X)}$
- Use log of odds z such that – $$Z_i = ln\left(\frac{P_i}{1-P_i}\right) = \alpha + \beta_1 x_1 + .. + \beta_n x_n$$
- Solving for P_i, we get $$P_i = E(y = 1|x_i) = \frac{e^z}{1+e^z} = \frac{e^{\alpha+\beta_i x_i}}{1+e^{\alpha+\beta_i x_i}}$$
- Which is sigmoid on z (log of odds)
- Coefficients are determined using Maximum likelihood estimator, while keeping cost function(cross entropy error function) minimum

Logistic Regression is Probabilistic Discriminative

Logistic Regression – Illustration

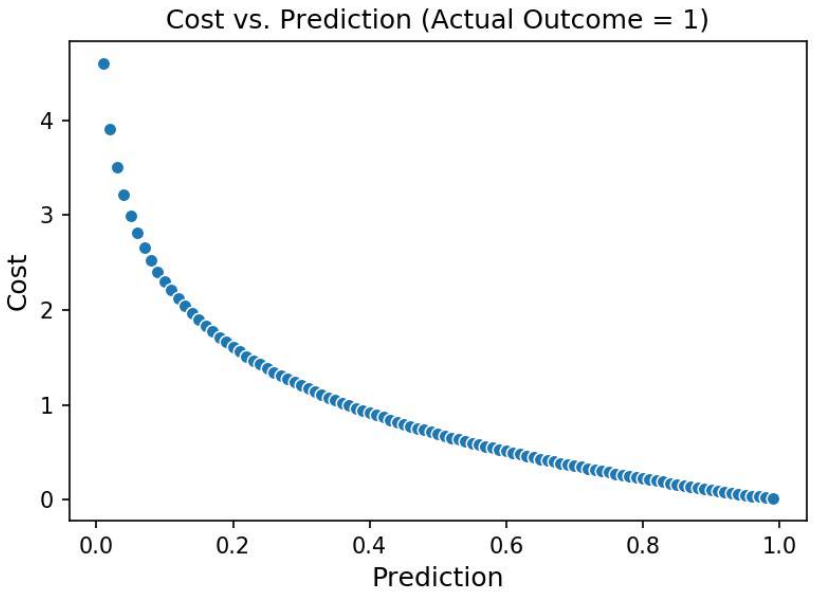
- Basket Ball – calculate the probability of shooting ball into a basket based on distance from basket
- Input parameter (X) – distance from basket
- Outcome (Y) – Yes or No
- Data distribution is not linearly separable
- The parameter for sigmoid, ***Z = B0 + B1*X***
  - i.e. Z = B0 + B1*distance_from_basket
  - Also Z = log(odds_of_making_shot)
- Probability of making shot (Y) = 1 / [1 + e^(-Z)]

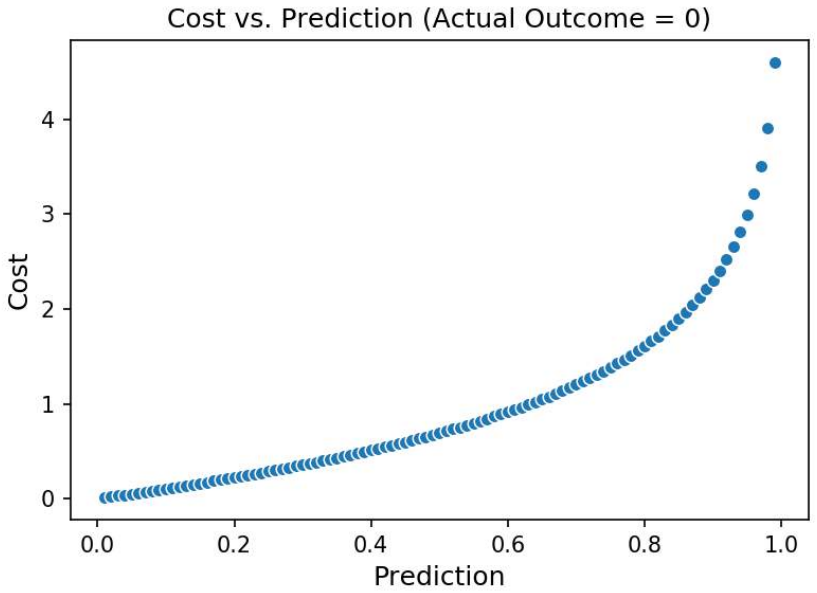


Logistic Regression is Probabilistic Discriminative

Logistic Regression – Illustration

- Cost function calculation
  - If Actual Outcome = 1, then Cost = -log(pred_prob)
  - Else if Actual Outcome = 0, then Cost = -log(1-pred_prob)
- Compute total cost

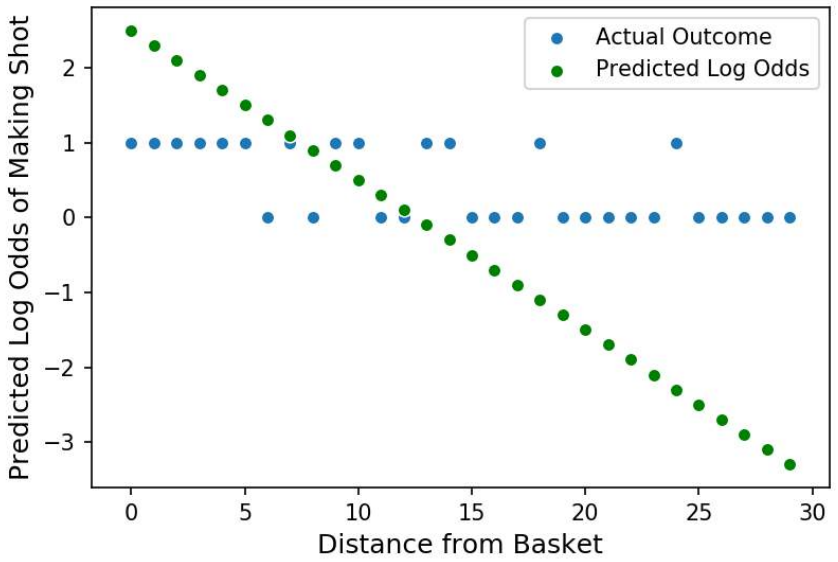


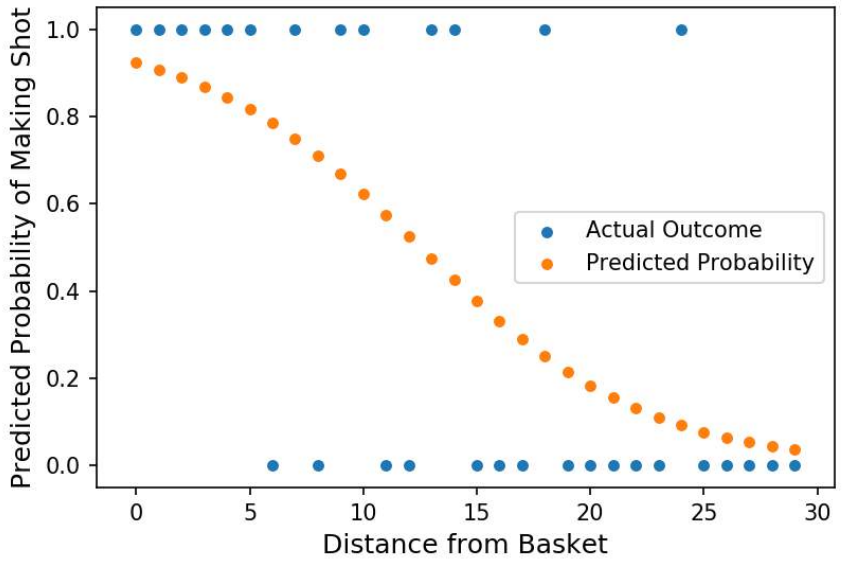


Logistic Regression is Probabilistic Discriminative

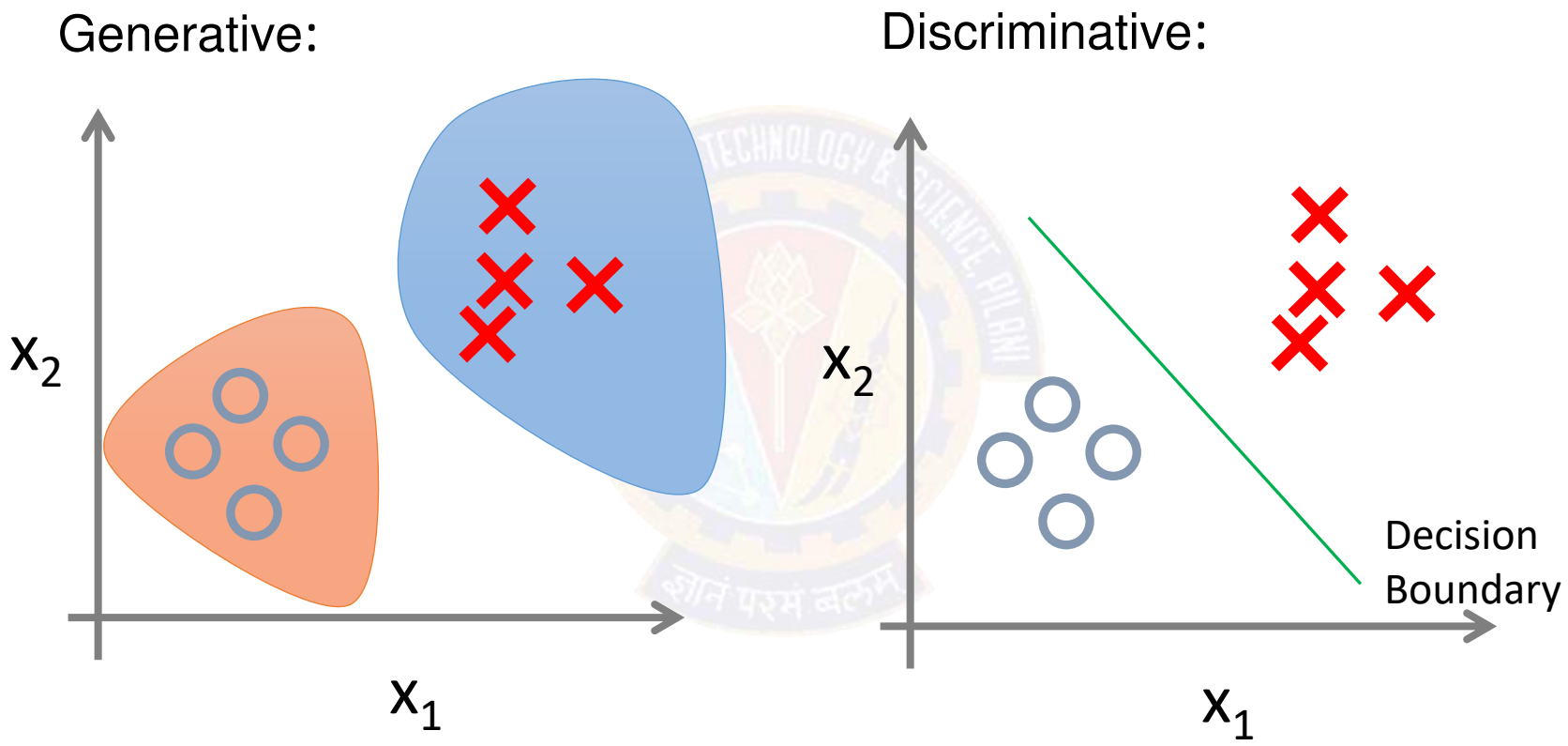
Logistic Regression – Illustration

- Using optimization techniques (linear regression), find B0 and B1 that minimize total cost
- Calculate Z in each case and penalize based on cost function





Probabilistic Generative v/s Discriminative Model



Probabilistic Generative v/s Discriminative Model

Generative	Discriminative
Ex: Naïve Bayes	Ex: Logistic Regression
Estimate $P(Y)$ and $P(X\|Y)$	Finds class label directly $P(Y\|X)$
Prediction $\hat{y} = \text{argmax}_y \, P(Y = y)P(X = x\|Y = y)$	Prediction $\hat{y} = P(Y = y\|X = x)$

Naïve Bayes versus Logistic Regression

- Naïve Bayes are Generative Models which Logistic Regression are Discriminative Models
- Naïve Bayes easy to construct
- Naïve Bayes better on smaller datasets
- Naive Bayes also assumes that the features are conditionally independent. Real data sets are never perfectly independent
- When the training size reaches infinity, logistic regression performs better than the generative model Naive Bayes.
  - Optional reading by Ng and Jordan has proofs and experiments
- Logistic regression allows arbitrary features

Thank You!

In our next session:

Logistic Regression – Python Implementation

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Logistic Regression – Python Implementation

Dr. Chetana Gavankar

In this segment

Logistic Regression – Python Implementation

- Tech Stack
- Implementation
  - Load Data
  - Pre-tune input parameters
  - Build the model
  - Evaluate the model

Logistic Regression – Python Implementation

Tech Stack

- Base language - Python 2.x
- Logistic Regression routines -Sklearn
- Data framing - Pandas
- Math routines - Numpy
- Graphs – Matplotlib, Seaborn

Logistic Regression – Python Implementation

Python Implementation

- Load the data

In [2]:

```
diabetes = pd.read_csv('../input/diabetes.csv')
diabetes.head()
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Logistic Regression – Python Implementation

Python Implementation

- Analyze the data

In [5]:
```
diabetes.describe()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Ou
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	76
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.3
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.4
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.0
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.0
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.0
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.0
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.0

Logistic Regression – Python Implementation

Python Implementation

- Model parameter tuning with Cross Validation libraries

In [19]:

```
from sklearn.model_selection import GridSearchCV

c_values = list(np.arange(1, 100))

param_grid = [
    {
        'C': c_values,
        'penalty': ['l1'],
        'solver': ['liblinear'],
        'multi_class': ['ovr'],
        'random_state': [42]
    },
    {
        'C': c_values,
        'penalty': ['l2'],
        'solver': ['liblinear', 'newton-cg', 'lbfgs'],
        'multi_class': ['ovr'],
        'random_state': [42]
    }
]

grid = GridSearchCV(
    LogisticRegression(),
    param_grid,
    cv=strat_k_fold,
    scoring='f1'
)
grid.fit(X, y)

# Best LogisticRegression parameters
print(grid.best_params_)
# Best score for LogisticRegression with best parameters
print('Best score: {:.2f}%'.format(grid.best_score_ * 100))
```

```
{'C': 1, 'multi_class': 'ovr', 'penalty': 'l2', 'random_state': 42, 'solver': 'newton-cg'}
Best score: 62.25%
```

Logistic Regression – Python Implementation

Python Implementation

- Train the model

In [20]:

```
log_reg = LogisticRegression(
    # Parameters chosen based on GridSearchCV result
    C=1,
    multi_class='ovr',
    penalty='l2',
    solver='newton-cg',
    random_state=42
)
log_reg.fit(X_train, y_train)

log_reg_predict = log_reg.predict(X_test)
log_reg_predict_proba = log_reg.predict_proba(X_test)[:, 1]
```

Logistic Regression – Python Implementation

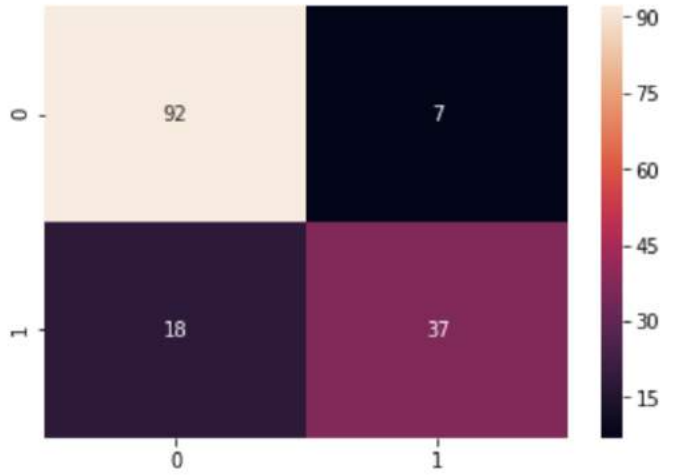
In [21]:

```
print('Accuracy: {:.2f}%'.format(accuracy_score(y_test, log_reg_predict) * 100))
print('AUC: {:.2f}%'.format(roc_auc_score(y_test, log_reg_predict_proba) * 100))
print('Classification report:\n\n', classification_report(y_test, log_reg_predict))
print('Training set score: {:.2f}%'.format(log_reg.score(X_train, y_train) * 100))
print('Testing set score: {:.2f}%'.format(log_reg.score(X_test, y_test) * 100))
```

```
Accuracy: 83.77%
AUC: 85.38%
Classification report:

             precision    recall  f1-score   support

          0       0.84      0.93      0.88        99
          1       0.84      0.67      0.75        55

avg / total       0.84      0.84      0.83       154

Training set score: 76.22%
Testing set score: 83.77%
```

```
outcome_labels = sorted(diabetes.Outcome.unique())

sns.heatmap(
    confusion_matrix(y_test, log_reg_predict),
    annot=True,
    xticklabels=outcome_labels,
    yticklabels=outcome_labels
)
```

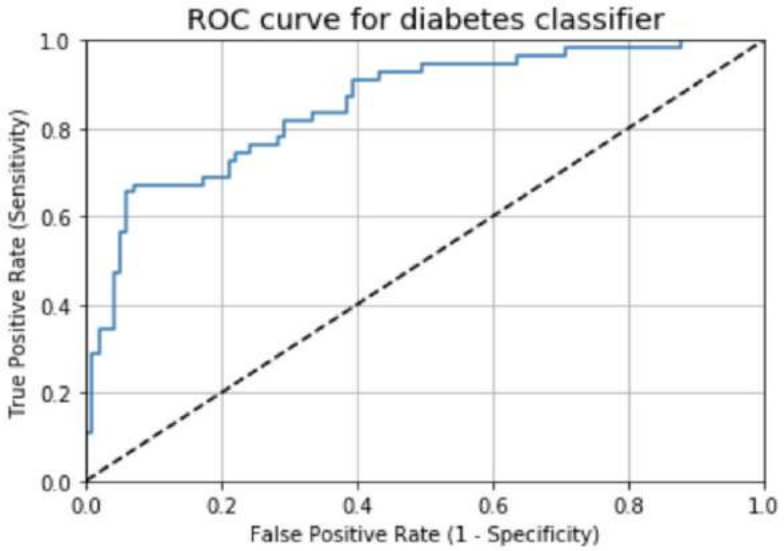
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fefd4e37d30>
```



Logistic Regression – Python Implementation

In [23]:

```
fpr, tpr, thresholds = roc_curve(y_test, log_reg_predict_proba)

plt.plot([0,1],[0,1],'k--')
plt.plot(fpr, tpr)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.rcParams['font.size'] = 12
plt.title('ROC curve for diabetes classifier')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.grid(True)
```



Thank You!

In our next session:

Logistic Regression –Overfitting, Interpretability, Applications

BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Logistic Regression – Overfitting, Interpretability and Applications

Dr. Chetana Gavankar

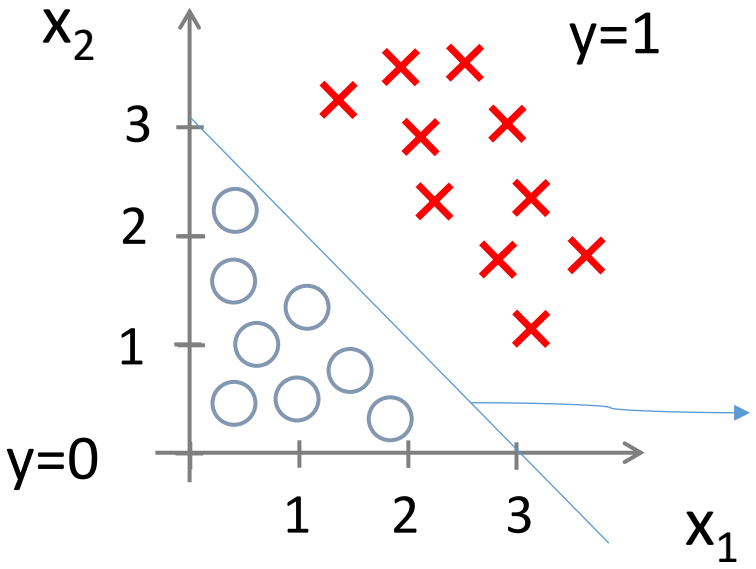
Associate Professor, BITS Pilani
Chetana.gavankar@pilani.bits-pilani.ac.in

In this segment

Logistic Regression – Miscellaneous Topics

- Logistic Regression – Decision Boundary
- Overfitting - Countermeasures
- Interpretability of Logistic Regression

Decision Boundary



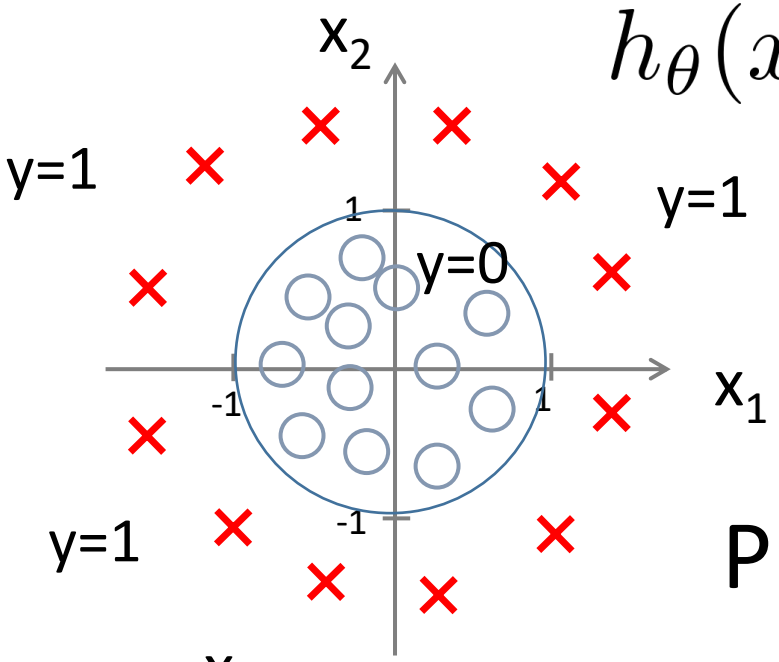
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

Decision Boundary

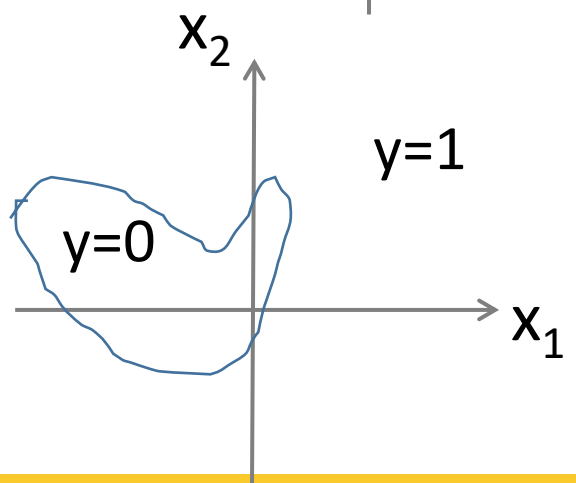
Non-linear decision boundaries



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

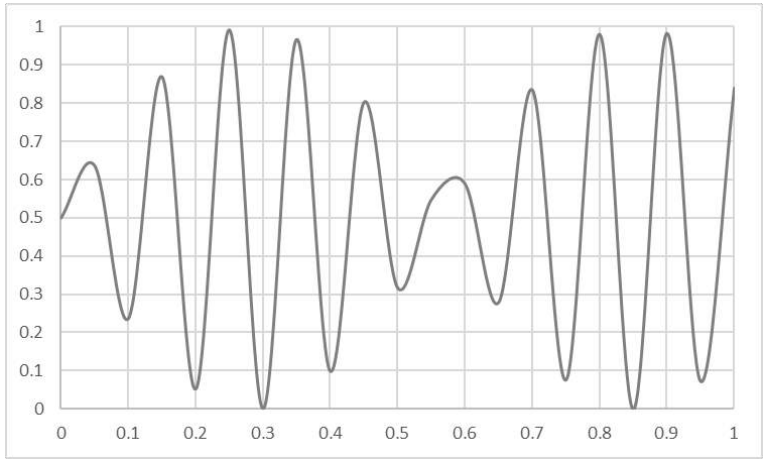


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \ldots)$$

Overfitting vs Underfitting

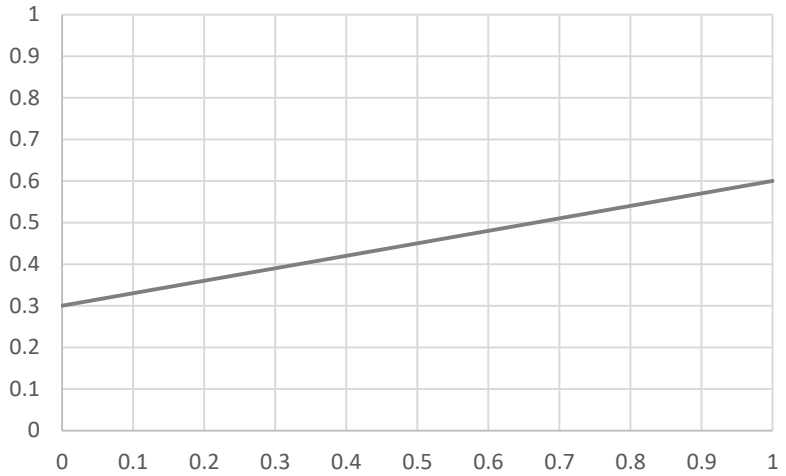
Overfitting

- Fitting the data too well
  - Features are noisy / uncorrelated to concept
  - Modeling process very sensitive (powerful)



Underfitting

- Learning too little of the true concept
  - Features don't capture concept
  - Too much bias in model



Handling Missing Values

- Replace missing values with column averages (i.e. replace missing values in feature 1 with the average for feature 1).
- Replace missing values with column medians.
- Remove records that are missing features.
- Use a machine learning technique that uses classification trees, e.g. Decision tree

Interpretability of Logistic Regression

$h_\theta(x)$ = estimated probability that y = 1 on input x

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

and $h_\theta(x) = 0.7$

Tell patient that 70% chance of tumor being malignant

Probability that y = 1, given x, parameterized by θ

$$P(y = 0|x; \theta) + P(y = 1|x; \theta) = 1$$
$$P(y = 0|x; \theta) = 1 - P(y = 1|x; \theta)$$

Interpretability of Logistic Regression

- Outcome in logistic regression is a probability between 0 and 1.
- Big advantage over models that can only provide the final classification. Knowing that an instance has a 99% probability for a class compared to 51% makes a big difference.

$$\frac{P(y=1)}{1-P(y=1)} = odds = exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p\right)$$

- Adding 1 to one of the feature values

$$\frac{odds_{x_j+1}}{odds} = \frac{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j(x_j+1) + \ldots + \beta_p x_p\right)}{exp\left(\beta_0 + \beta_1 x_1 + \ldots + \beta_j x_j + \ldots + \beta_p x_p\right)}$$

- Using $\frac{exp(a)}{exp(b)} = exp(a-b)$

$$\frac{odds_{x_j+1}}{odds} = exp\left(\beta_j(x_j+1) - \beta_j x_j\right) = exp\left(\beta_j\right)$$

- Change in x_j by one unit increases the log odds ratio by the value of the corresponding weight.

Logistic Regression Applications

- **Credit Card Fraud** : Predicting if a given credit card transaction is fraud or not
- **Health** : Predicting if a given mass of tissue is benign or malignant
- **Marketing** : Predicting if a given user will buy an insurance product or not
- **Banking** : Predicting if a customer will default on a loan.

References

- http://www.cs.cmu.edu/~tom/NewChapters.html
- http://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf
- https://medium.com/@sangha_deb/naive-bayes-vs-logistic-regression-a319b07a5d4c
- https://www.youtube.com/watch?v=-la3q9d7AKQ

Interpretability

- https://christophm.github.io/interpretable-ml-book/logistic.html

Thank You!

In our next session:

Decision Tree