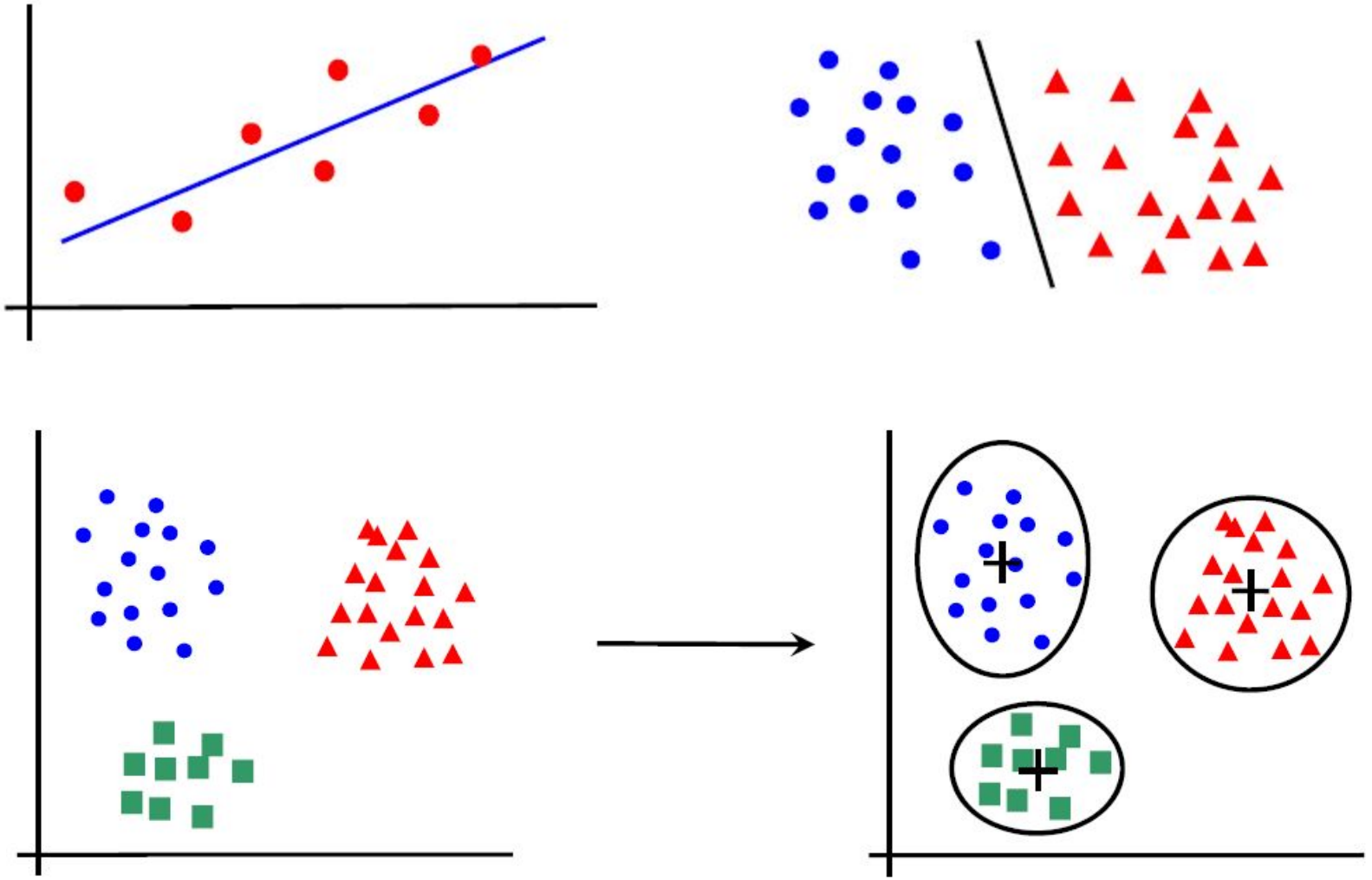
Machine Learning

Introduction

Indian Institute of Information Technology
Sri City, Chittoor

Welcome to Machine Learning Class



Today's Agenda

- Course plan
  - Pre-requisite
  - Topics
  - Textbooks/References
  - Evaluation components
  - Honor code
- Introduction to machine learning
  - What is ML?
  - When do we use ML?
  - Applications
  - Relation with AI and DL
  - Relation with other fields
  - Different machine learning paradigms

Pre-requisite

- Probability
  - Distribution, random variable, expectation, conditional probability, variance, density
- Linear algebra
  - Linear transformation, Rank, Positive definite matrix
  - Eigenvalue & Eigenvector, Diagonalization
- Calculus (Optimization – Minima, maxima)
- Basic programming
  - Python (First Priority)
  - Matlab/C/C++ (Second Priority)

Topics

- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
  - Clustering
- Semi-supervised Learning
- Reinforcement Learning

Textbooks/References

1. "Pattern Classification" by R. O. Duda, P. E. Hart and D. G. Stork.
2. "An Introduction to Statistical Learning" by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani.
3. "Pattern Recognition and Machine Learning" by Christopher M. Bishop.
4. "Introduction to Machine Learning" by Ethem Alpaydin.
5. "Pattern Recognition: An Algorithmic Approach" by M. Narasimha Murty, V. Susheela Devi.
6. "Machine learning" by Tom Mitchell.

Evaluation Components

- Mid-Exam(s): 20%
- End-Exam: 30%
- Assignments: 20%
- Scheduled Quiz: 10%
- Project: 20%

This is tentative and will be finalized in Class committee meeting.

Honor Code

Do's

- Write down the code independently
- Submit the assignment within the deadline
- Read the books/references for detail description of the topics

Don'ts

- copy, refer to, or look at any official or unofficial previous years' solutions in preparing the answers

Introduction to ML

- What is ML?
- Terminologies used in ML
- When do we use ML?
- Applications
- Relation with AI and DL
- Relation with other fields
- Different machine learning paradigms

What is ML?

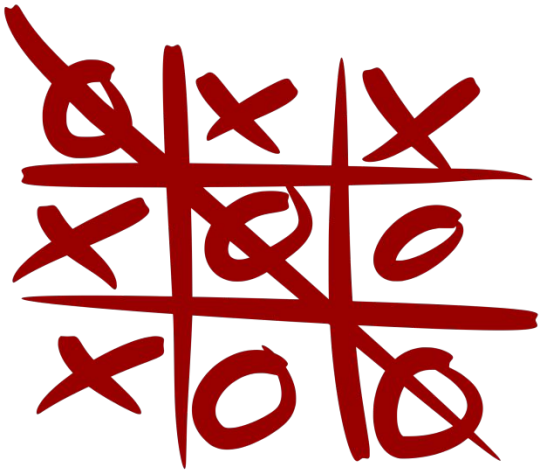
- Machine learning (ML) is the study of computer algorithms that improve automatically through experience.

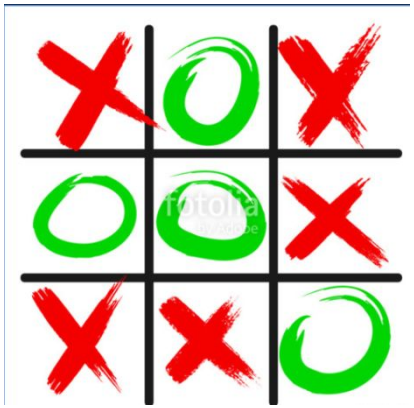
- Machine-learning algorithms use statistics to find patterns in massive amounts of data.

- Traditionally, software engineering combined human created rules with data to create answers to a problem. Instead, machine learning uses data and answers to discover the rules behind a problem – **F. Chollet, Deep Learning with Python**

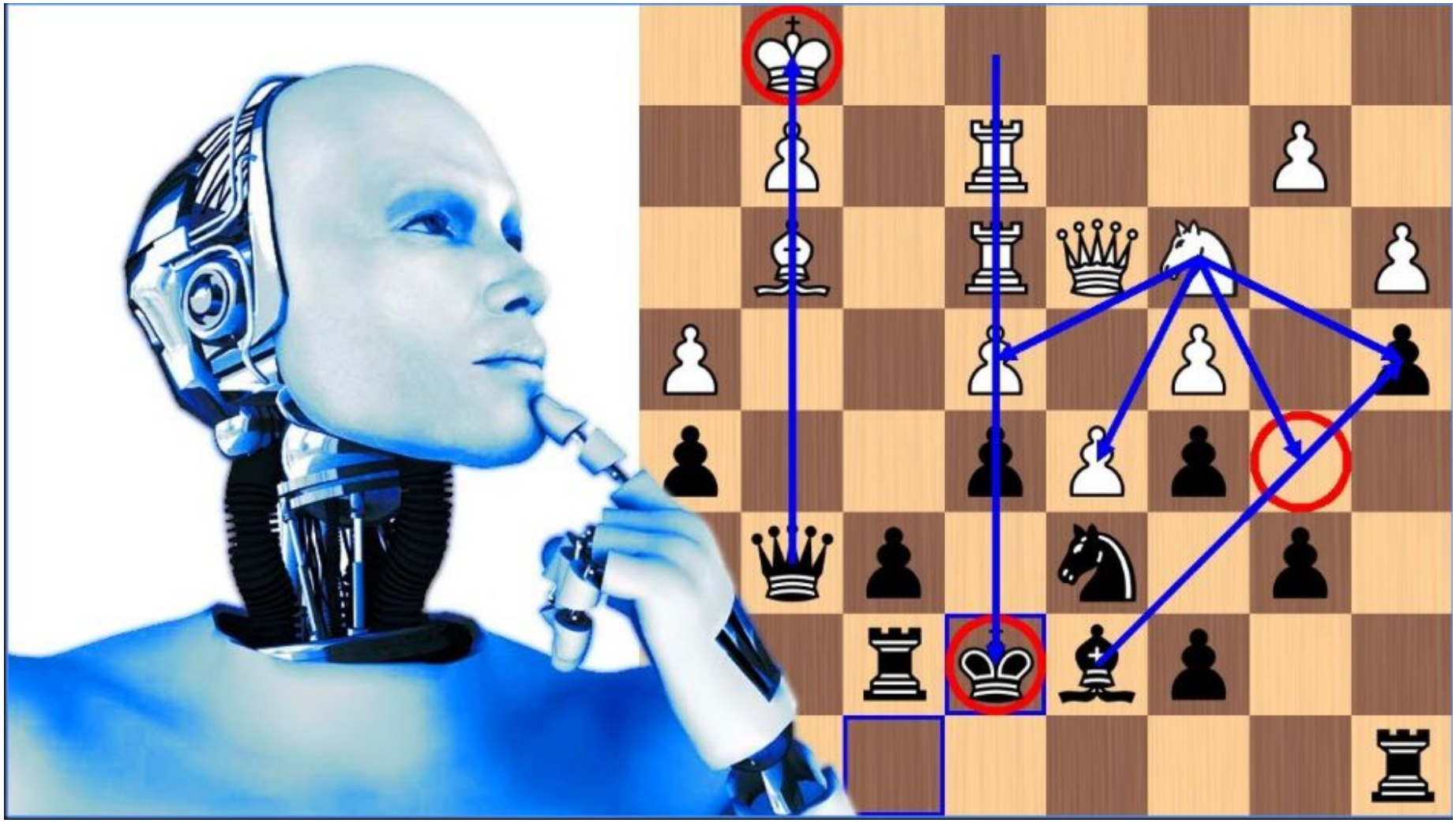
Early AI – Rule based

- Tic-tac-toe

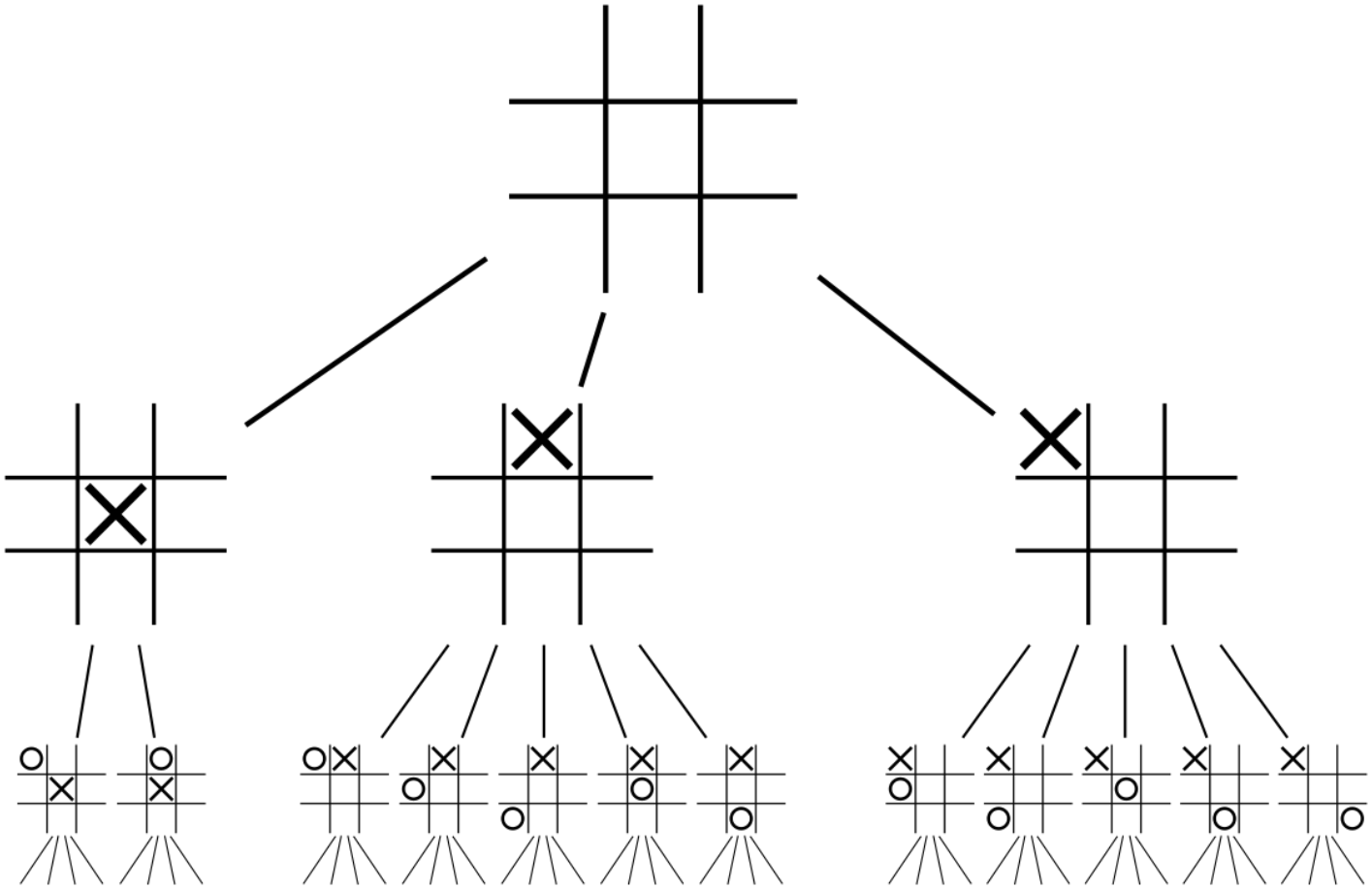




Chess --



Tic-tac-toe – how it is done?

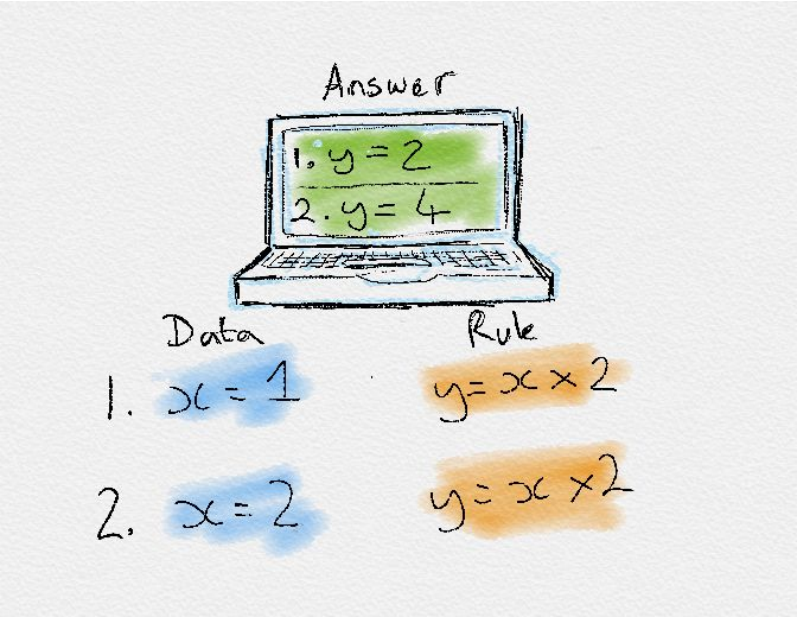


Deductive Vs Inductive

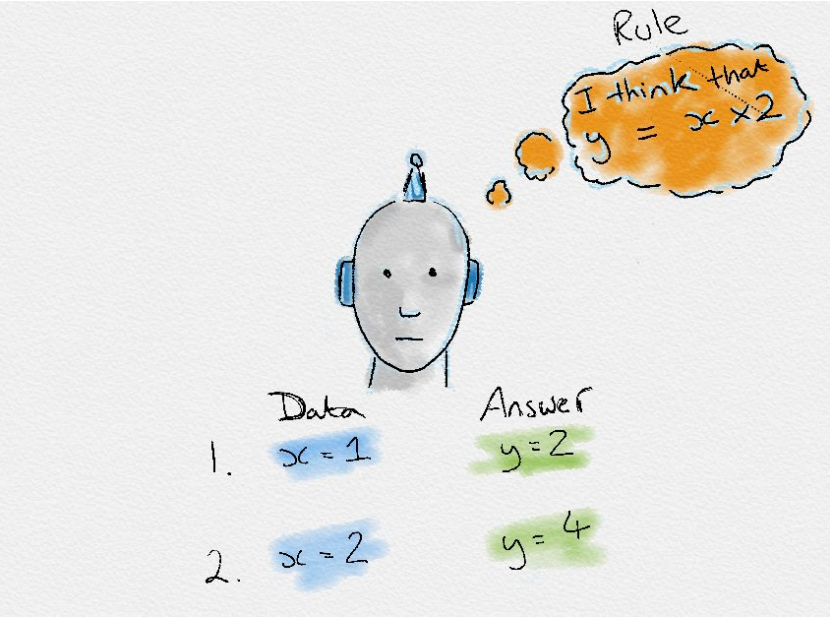
- Giving rules can work for game playing like problems.
- But, you cannot give all rules for a self-driving car.
  - This attempt failed.
- So, inductive means, we will not give explicit rules, but give examples saying which is good which is bad, so on.
  - Machine will learn rules from the data.

Deductive Vs Inductive

Deductive Reasoning	Inductive Learning



Traditional Programming



Machine Learning

Credits: Machine Learning | An Introduction | by Gavin Edwards | Towards Data Science

Machine Learning

- Learning from data

ALGORITHMS

DATA

Terminologies used in ML

- ML systems learn how to make inference from the input data samples to produce useful predictions on un-seen (test) data.
- Input data:
  - labelled examples: A labelled example includes feature(s) and the label. {features, label}: (x, y)For e.g.:

Features:	Label
Normal RBC, Normal HgB	Healthy
Low RBC, Low HgB	Anaemic

  - unlabelled examples: An unlabelled example contains features but not the label. {features, ?}: (x, ?)
    - For e.g.: (Normal RBC, Low HgB)

Features: House type	Price
4BHK,	40,000
4BHK,	15,000
2BHK,	25,000
2BHK,	8,000

Terminologies used in ML

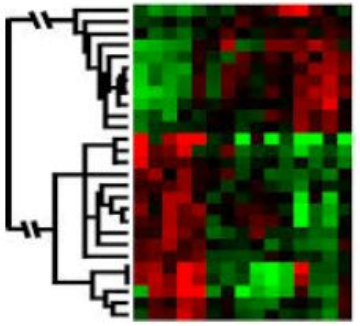
- Machine Learning Model:
  - A ML model defines the relationship between the features and label.
    - For e.g.: An anaemia diagnostic model might associate certain features strongly with “anaemic“ or “healthy”, and predict the labels based on the association rules it inferred.
  - Two Phases of ML model development
    - **Training** means creating or **learning** the model.
    - **Testing/Inference** means applying the trained model to unlabelled examples in order to infer the label.

When do we use ML?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Humans can't explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)



Learning isn't always useful:

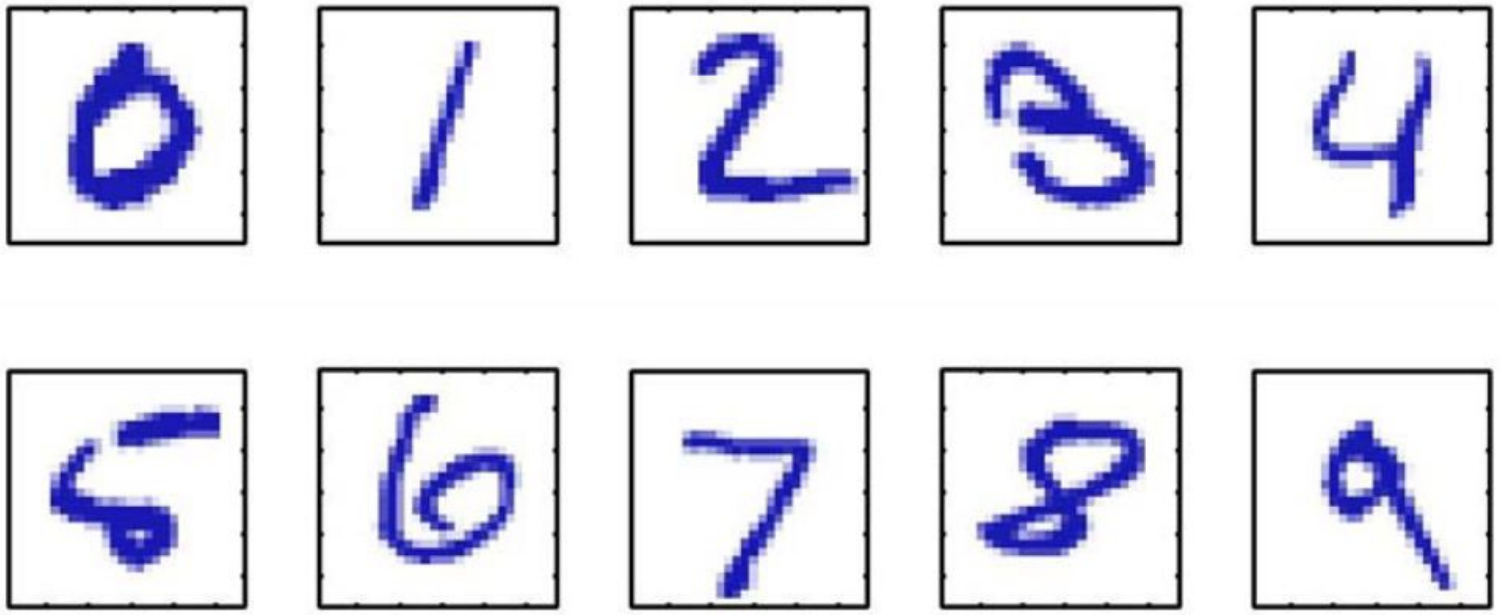
- There is no need to "learn" to calculate payroll

Slide adapted from **E Alpaydin**

Applications

- Hand-written digit recognition
- Speech recognition
- Face detection
- Object classification
- Email spam detection
- Computational biology
- Autonomous cars
- Computer-aided diagnosis

Hand-written Digit Recognition



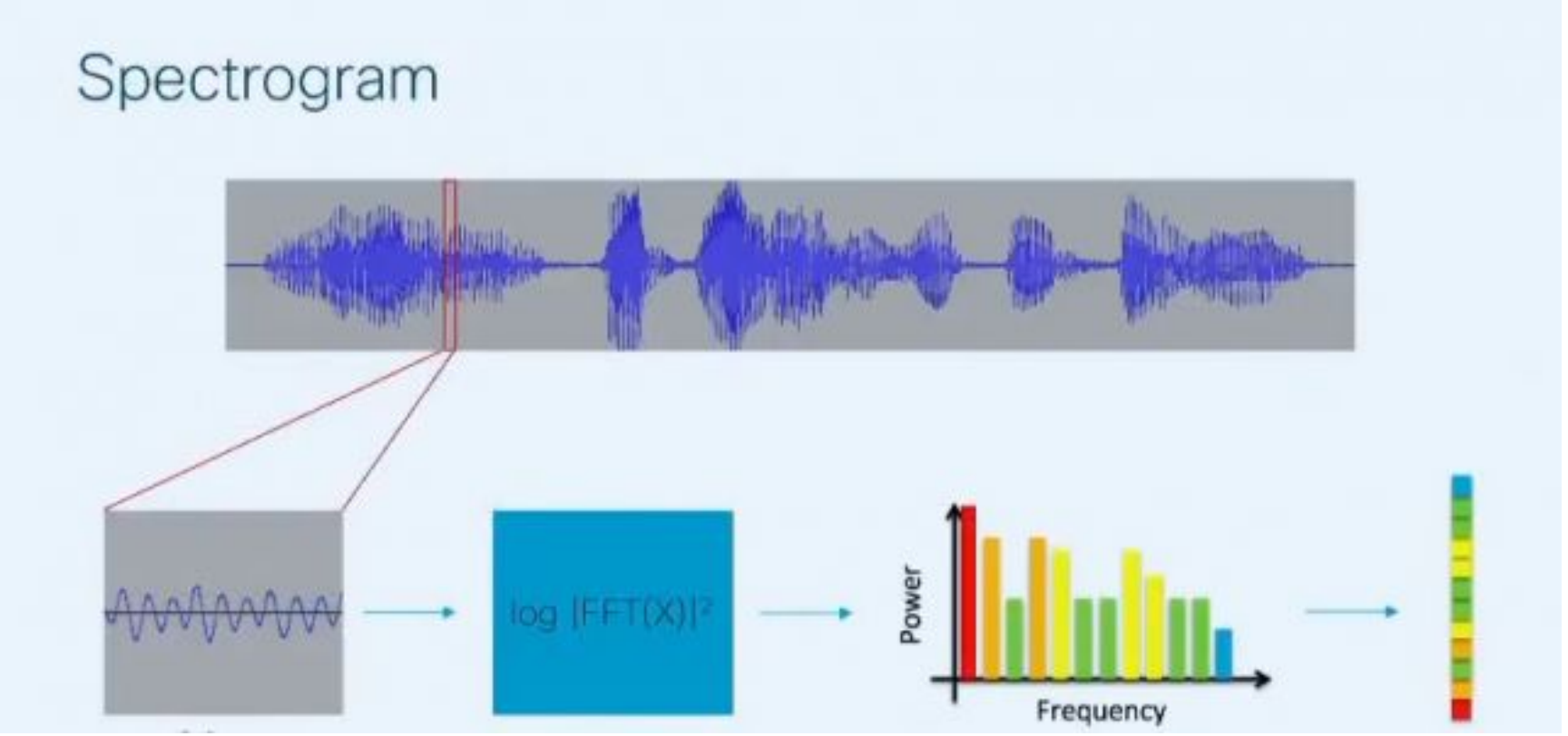
Images are 28 x 28 pixels

Represent input image as a vector $\mathbf{x} \in \mathbb{R}^{784}$

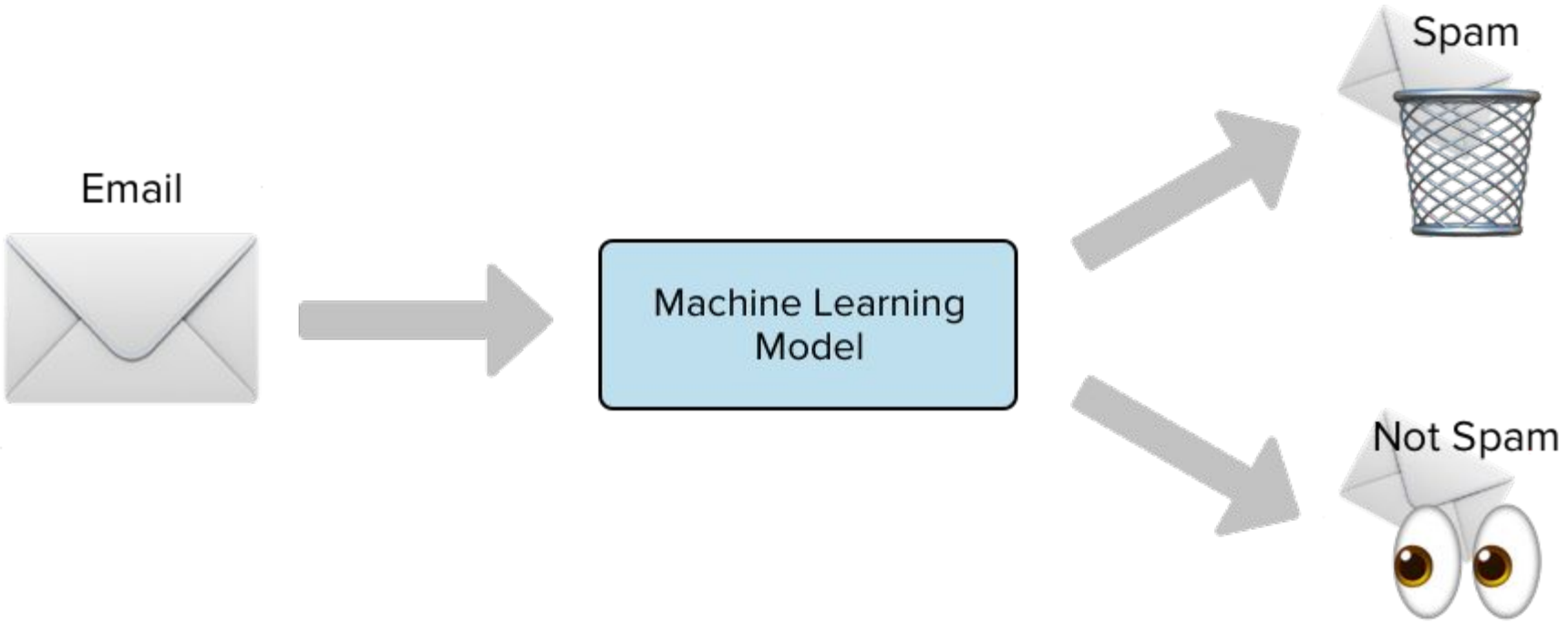
Learn a classifier $f(\mathbf{x})$ such that,

$$f : \mathbf{x} \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

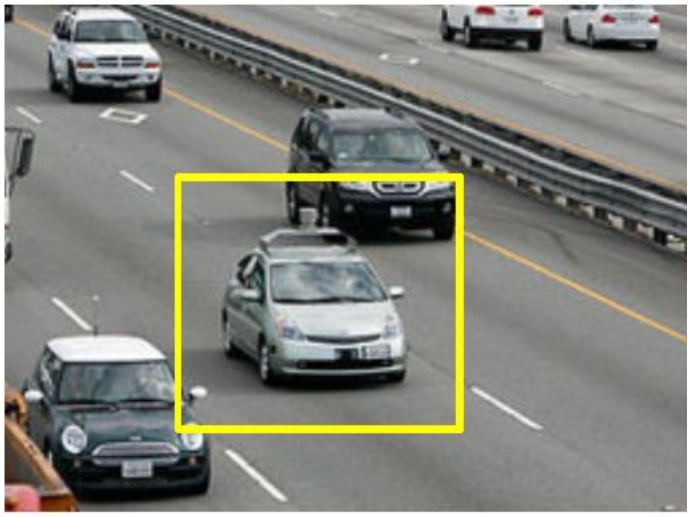
Speech Recognition



Email Spam Detection



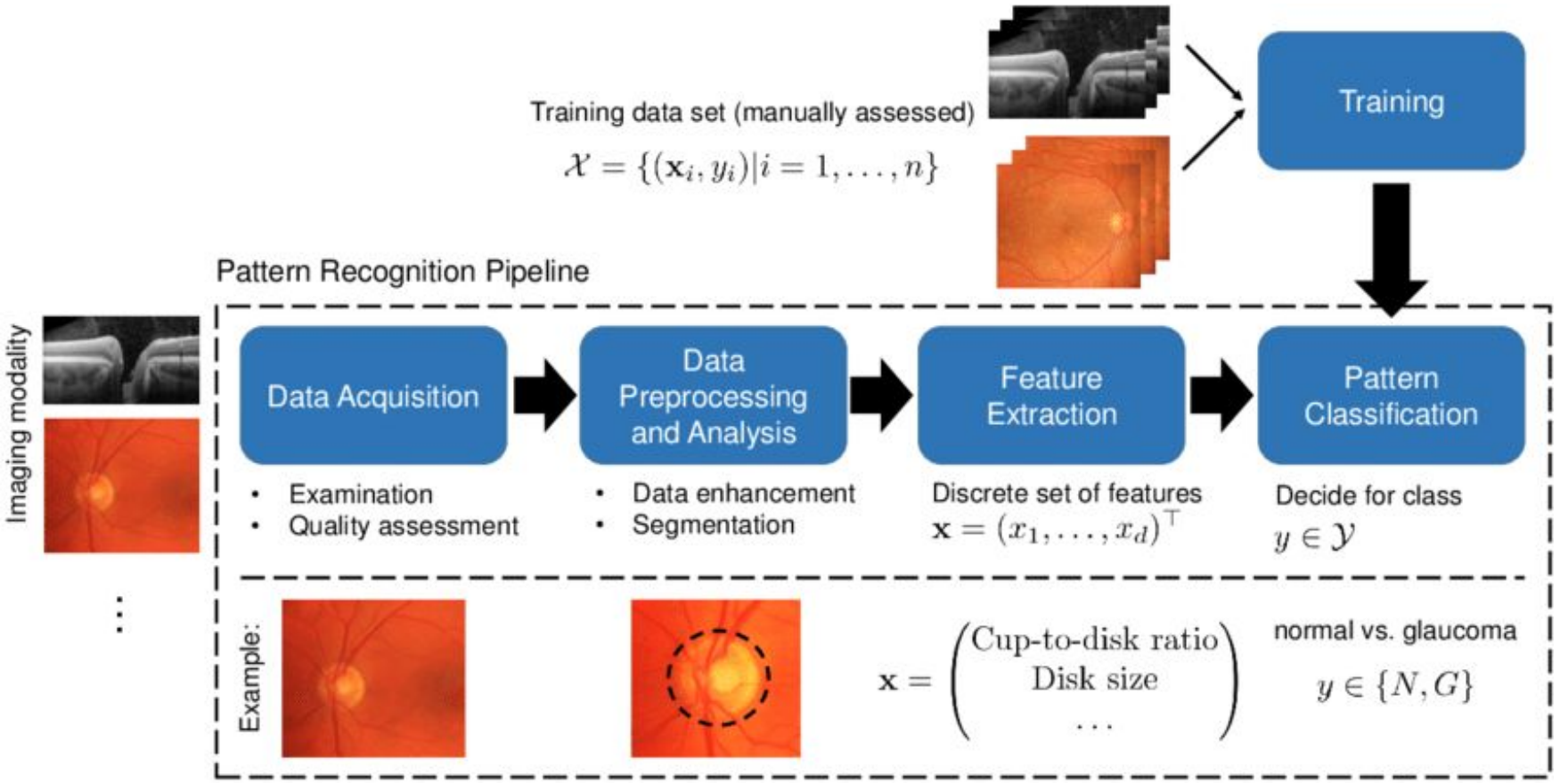
Autonomous Cars



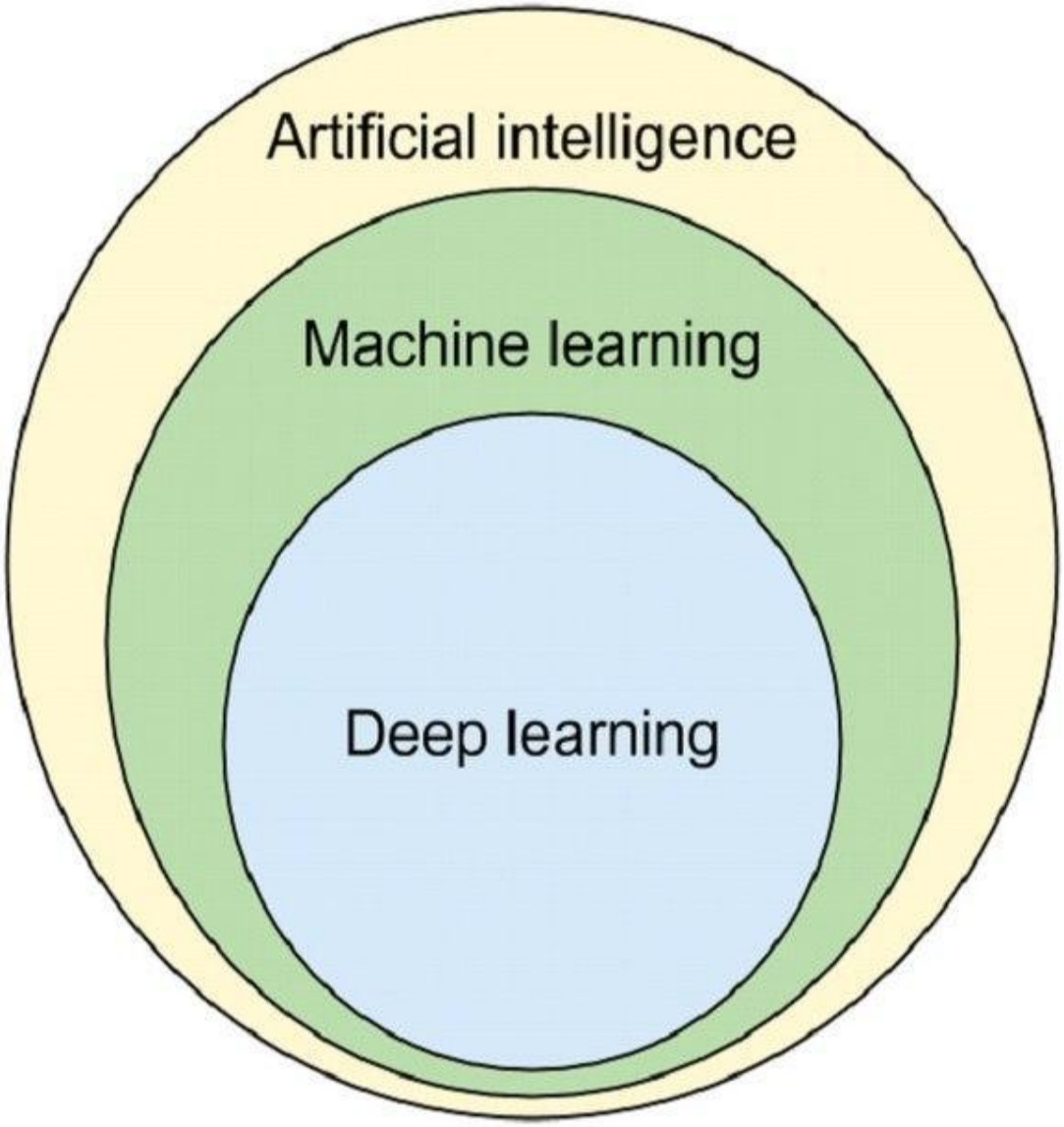
- Nevada made it legal for autonomous cars to drive on roads in June 2011
- As of 2013, four states (Nevada, Florida, California, and Michigan) have legalized autonomous cars

Penn's Autonomous Car →
(Ben Franklin Racing Team)

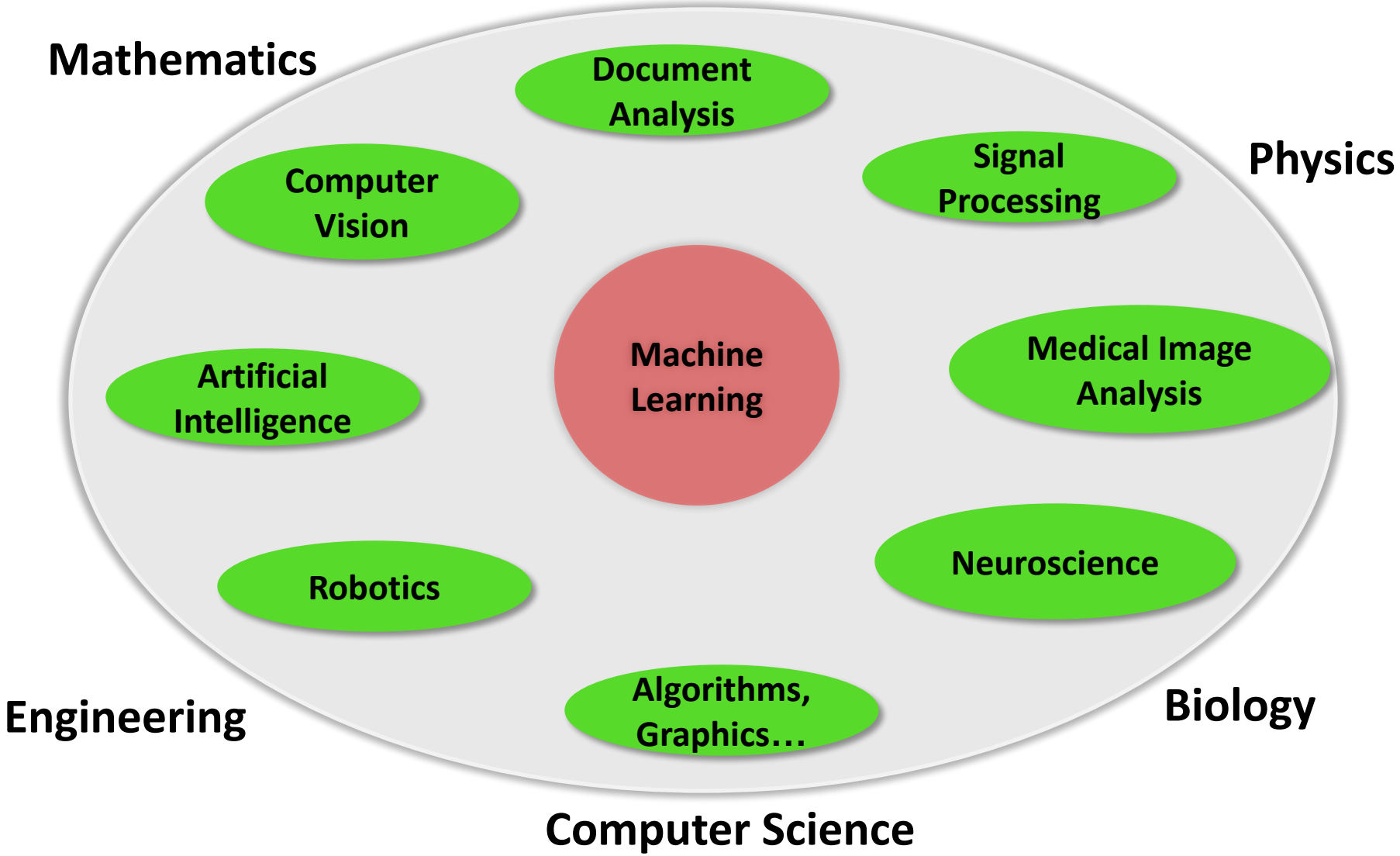
Computer-aided Diagnosis



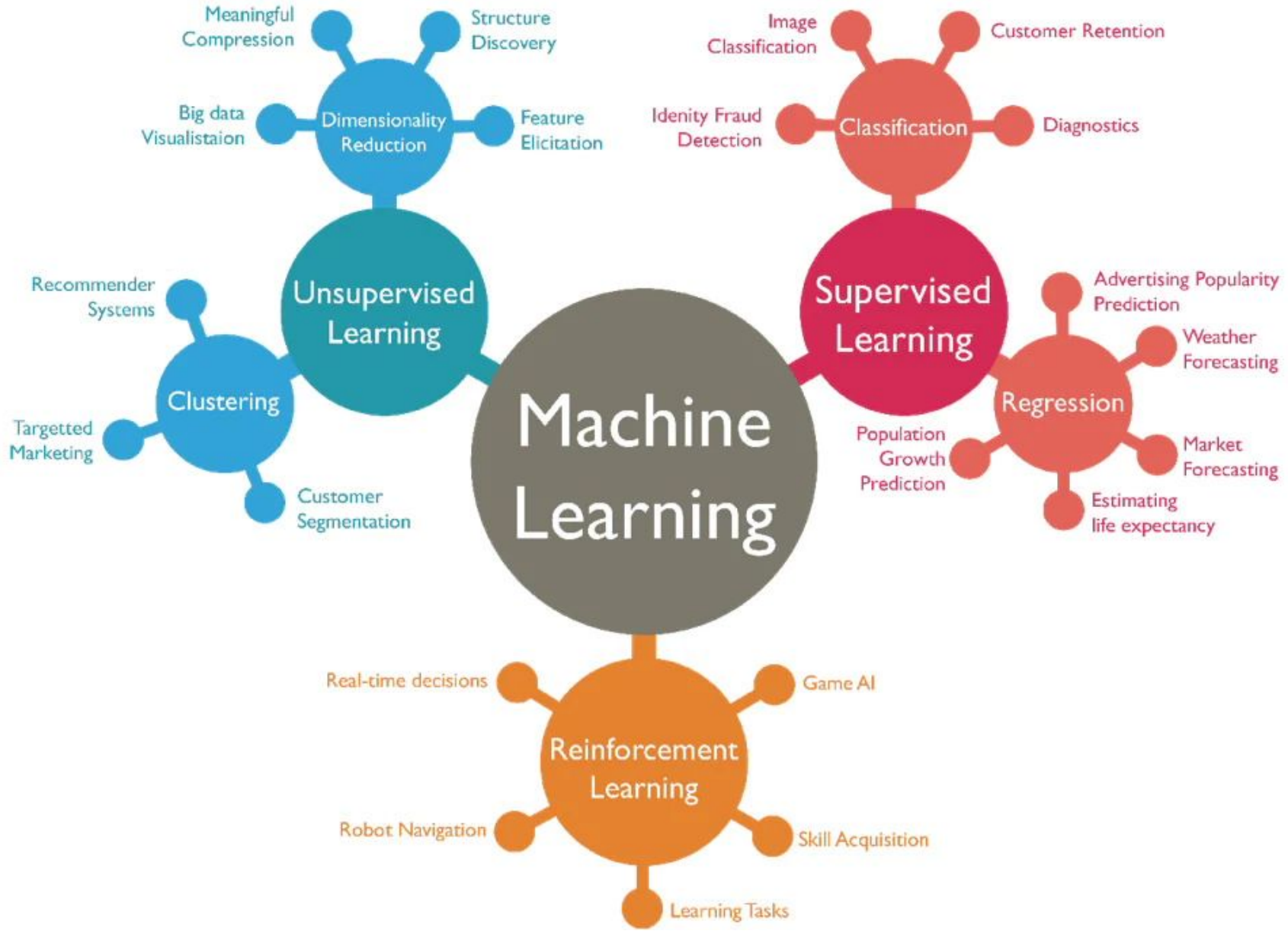
Relation with AI and DL



Relation with Other Fields



Different Machine Learning Paradigms



Credits: J. Chugh, "Types of machine learning and top 10 algorithms everyone should know," 2018

Supervised Learning

- To learn an unknown target function ***f***
- Input: a training set of labeled examples (x_j, y_j) where $y_j = f(x_j)$
- E.g., x_j is an image, $f(x_j)$ is the label "giraffe"
- Output: hypothesis *h* that is "close" to *f*, i.e., predicts well on unseen examples ("test set")
- Many possible hypothesis families for *h* – Linear models, logistic regression, neural networks, decision trees, examples (nearestneighbors) etc.

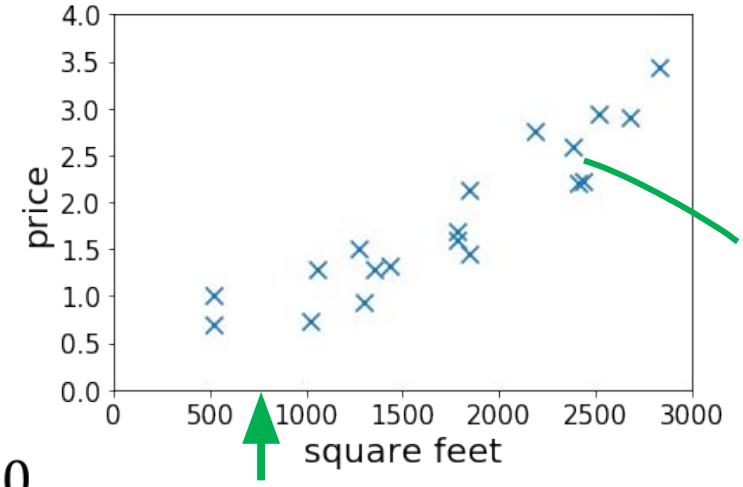
Slide Adapted from Stuart Russell

Housing Price Prediction

- Given: a dataset that contains n samples

$$\left(x^{(1)}, y^{(1)}\right), \ldots \left(x^{(n)}, y^{(n)}\right)$$

- Task: If a residence has x square feet, predict its price?



15th sample $(x^{(15)}, y^{(15)})$

$x = 800$

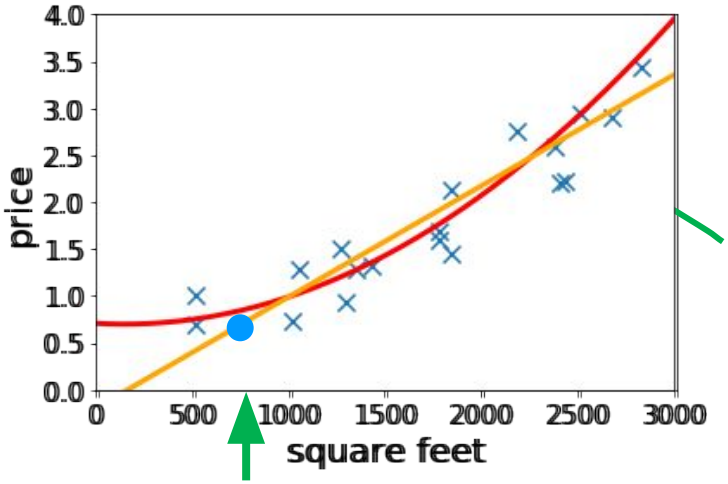
$y = ?$

Housing Price Prediction

- Given: a dataset that contains n samples

$$(x^{(1)}, y^{(1)}), \ldots (x^{(n)}, y^{(n)})$$

- Task: If a residence has x square feet, predict its price?



15th sample $(x^{(15)}, y^{(15)})$

$x = 800$

$y = ?$

- Solution: fitting linear/quadratic functions to the dataset.

High-dimensional Features

- $x \in \mathbb{R}^d$ for large d
- E.g.,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_d \end{bmatrix}$$

--- living size
--- lot size
--- # floors
--- condition
--- zip code
⋮

→ y --- price

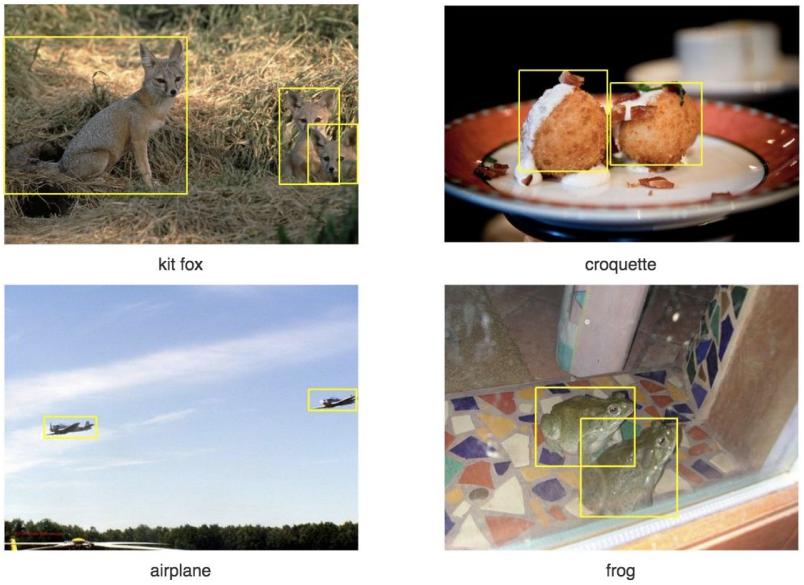
Supervised Learning in CV

- Image Classification
  - x = raw pixels of the image, y = the main object

ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Supervised Learning in CV

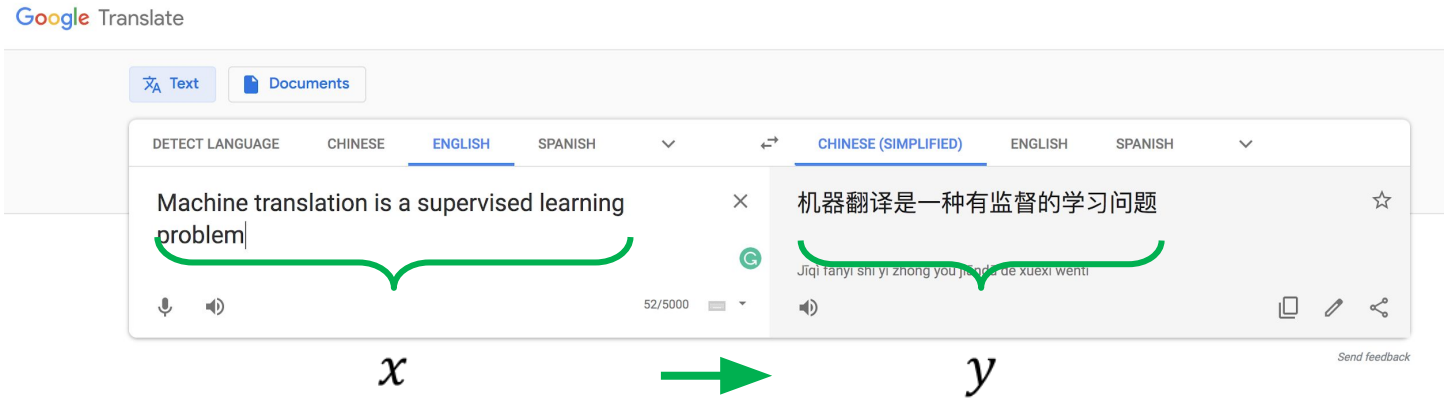
- Object localization and detection
  - x = raw pixels of the image, y = the bounding boxes



ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Supervised Learning in Natural Language Processing

- Machine translation



- Note: This course only covers the basic and fundamental techniques of supervised learning (which are not enough for solving hard vision or NLP problems.)

Supervised Learning

Advantage

- Well established field.
  - has good quantified methods (often with theoretical bounds)

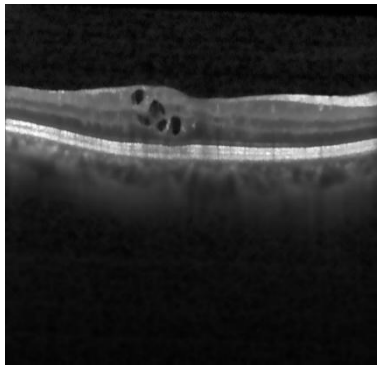
- You can easily test and debug your learning machine.
  - Since the labelled data is available you can easily inspect its output and find out what errors it's making on what type of input data.

Supervised Learning

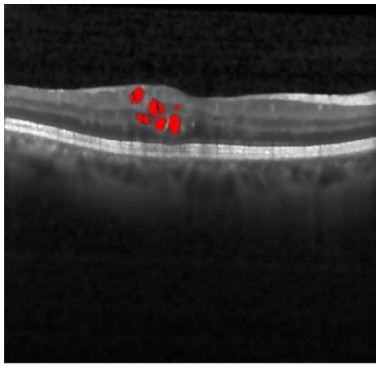
Disadvantage

- Collecting and labelling data is expensive and time-consuming.

  Example: Speech Recognition, Medical Image Analysis, etc.



Original Retinal Scan



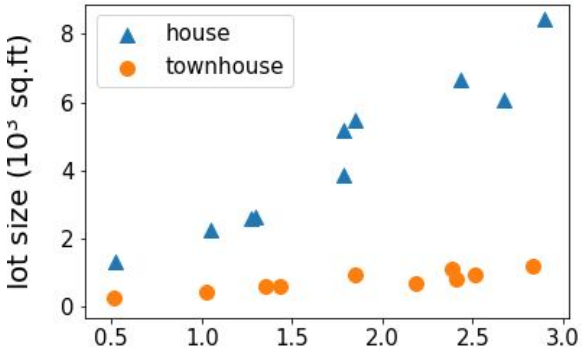
Intra-retinal cysts annotated scan

- Errors in your training data might confuse your algorithm and lower its accuracy. Garbage-in -> Garbage-out

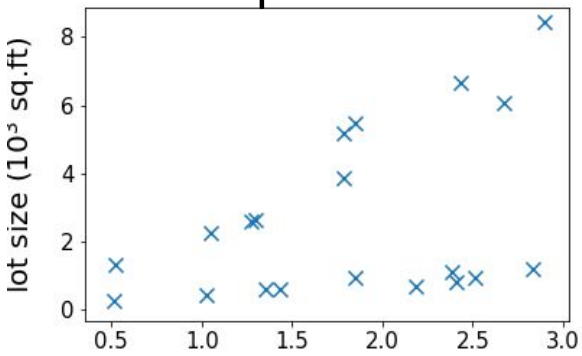
Unsupervised Learning

- Dataset contains no labels: $x^{(1)}, \ldots x^{(n)}$
- Goal (vaguely-posed): to find interesting structures in the data

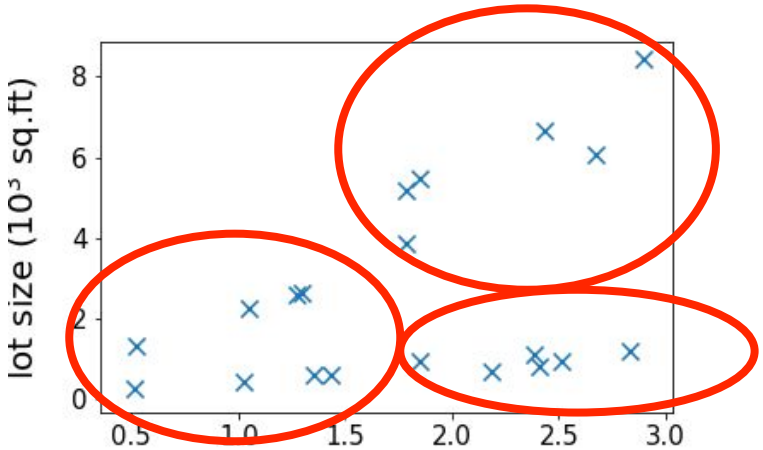
supervised



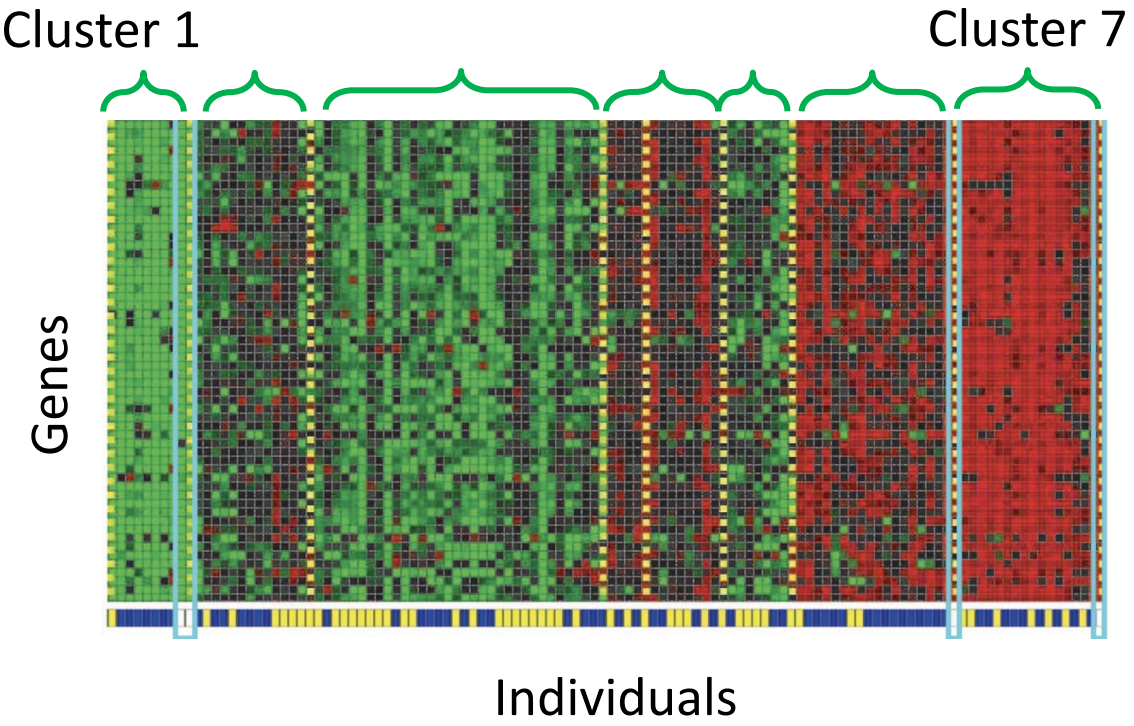
unsupervised



Clustering



Clustering Genes



Identifying Regulatory Mechanisms using Individual Variation Reveals Key Role for Chromatin Modification. [Su-In Lee, Dana Pe'er, Aimee M. Dudley, George M. Church and Daphne Koller. ’06]

Need for Unsupervised Learning

- Annotating large datasets is very costly and time consuming. Example: Speech Recognition, Medical Image Analysis, etc.

- There may be cases where we don't know how many/what classes is the data divided into.
  Example: Data Mining, Sentimental Analysis.

- We may want to use clustering to gain some insight into the structure of the data before designing a classifier.

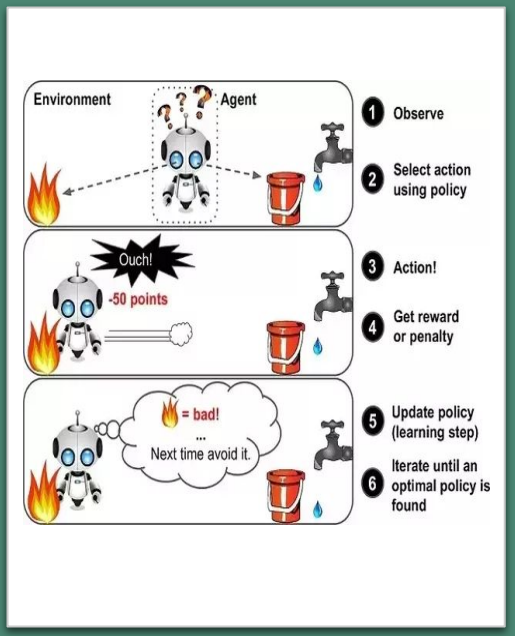
Disadvantages of Unsupervised Learning

- Unsupervised Learning is harder as compared to Supervised Learning. Since, quantifying the inference made is difficult, since the ground-truth is missing.

- How do we know if results are meaningful since it has unlabelled data?
  - External evaluation- Expert analysis.
  - Internal evaluation- Objective function.

Reinforcement Learning

A reinforcement learning algorithm, or agent, learns by interacting with its environment. The agent receives rewards by performing correctly and penalties for performing incorrectly.



Reinforcement learning example: Fire fighting agent

Need for Reinforcement Learning

- Reinforcement learning can be used to solve very complex problems that cannot be solved by conventional techniques.
- In the absence of a training dataset, it is bound to learn from its experience.
- Reinforcement learning models can outperform humans in many tasks and learning process is similar to human learning.
- DeepMind's AlphaGo program, a reinforcement learning model, beat the world champion *Lee Sedol* at the game of *Go* in March 2016.

Disadvantages of Reinforcement Learning

- Reinforcement learning needs a lot of data and a lot of computation. It is data-hungry.
  - So for solving video games and puzzles it performs well.

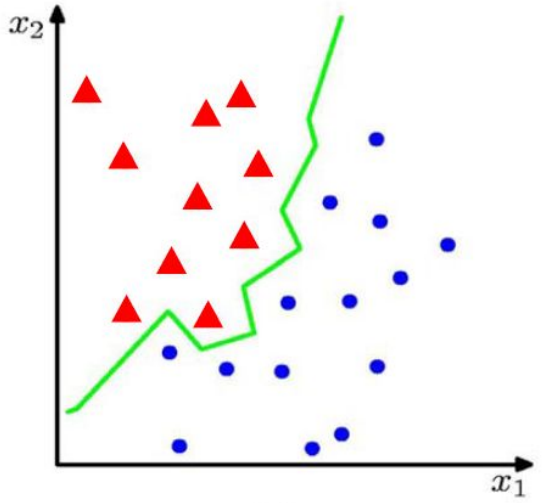
- Reinforcement learning assumes the world is Markovian, which it is not.
  - The Markovian model describes a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.

What is classification problem?

- Let there are two classes of objects.
  - Class 1: Set of dog pictures
  - Class 2: Set of cat pictures

- Problem is
  - Given a picture, you should say whether it is cat or dog.
  - For a human being it is easy…, but for a machine it is a non-trivial problem.

What is classification problem?



- Suppose we are given a training set of N observations

$$(x_1, \ldots, x_N) \text{ and } (y_1, \ldots, y_N), x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$$

- Classification problem is to estimate f(x) from this data such that

$$f(x_i) = y_i$$

Classification: Supervised Learning

Training Phase

We have shown a set of dog pictures and a set of cat pictures to a child.

Classification: Supervised Learning

Testing Phase

What is Learning?

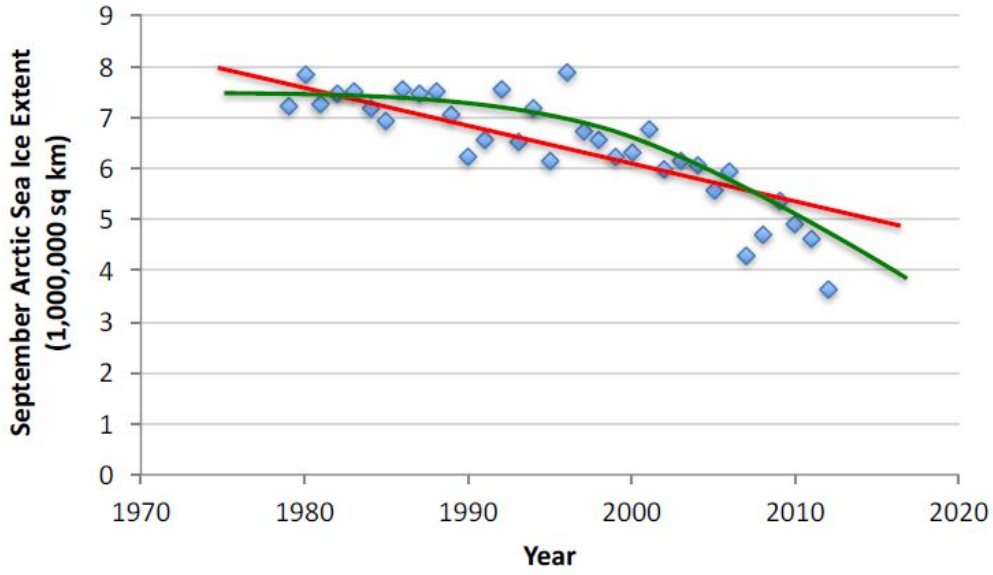
- Child has learnt what is it that is common among dogs … and, what is it that is common among cats… also, what are the distinguishing features/attributes.

- Child has learnt the pattern (regularity) behind all dogs and the pattern behind all cats.

- Child then recognized a test image as having a particular pattern that is unique to dogs.

What is Regression Problem?

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is real-valued == regression



Popular ML algorithms

Classification

- Linear Classifiers
- Support Vector Machines
- Decision Trees
- K-Nearest Neighbor
- Random Forest

Regression

- Linear Regression
- Logistic Regression
- Polynomial Regression

Resources: Journals

- Journal of Machine Learning Research www.jmlr.org
- Machine Learning
- IEEE Transactions on Neural Networks
- IEEE Transactions on Pattern Analysis and Machine Intelligence
- Annals of Statistics
- Journal of the American Statistical Association

Resources: Conferences

- International Conference on Machine learning (ICML)
- European Conference on Machine Learning (ECML)
- Neural Information Processing Systems (NIPS)
- Computational Learning
- International Joint Conference on Artificial Intelligence (IJCAI)
- ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)
- IEEE Int. Conf. on Data Mining (ICDM)

Thank You:
Question?

Machine Learning

Introduction to ML Model Evaluation

IIIT Sri City

This week's Agenda

- Recap to classification?
- How to evaluate the classification model?
- Evaluation Metrics for a Classification model
- Recap to regression problem?
- How to evaluate the regression model?
- Evaluation Metrics for a regression model
- Dataset: Train, Validation and Test sets
- Train, test and validation split
- Data Sampling Methods
- Overfit and Underfit

Introduction to parametric and non-parametric machine learning models.

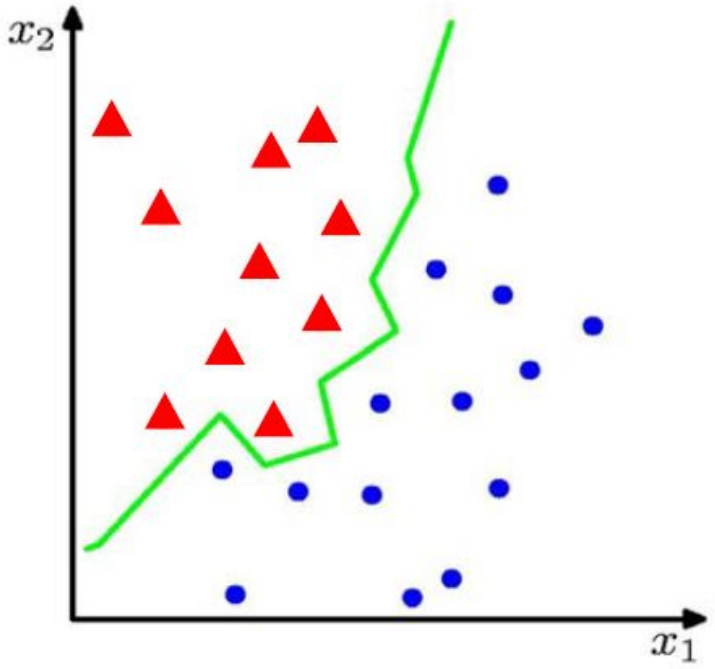
- *A learning model that summarizes data with a set of parameters of fixed size (independent of the number of training examples) is called a parametric model.*
  - *No matter how much data you throw at a parametric model, it won't change its mind about how many parameters it needs.*

- Some more examples of parametric machine learning algorithms include:
  - Logistic Regression
  - Linear Discriminant Analysis
  - Perceptron

- *Algorithms that do not make strong assumptions about the form of the mapping function are called nonparametric machine learning algorithms.*
  - *Nonparametric methods are good when you have a lot of data and no prior knowledge, and when you don't want to worry too much about choosing just the right features.*

- Some more examples of popular nonparametric machine learning algorithms are:
  - k-Nearest Neighbors
  - Decision Trees like CART and C4.5
  - Support Vector Machines

Recap: What is classification problem?



- Suppose we are given a training set of N observations

$$(x_1, \ldots, x_N) \text{ and } (y_1, \ldots, y_N), x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$$

- Classification problem is to estimate f(x) from this data such that

$$f(x_i) = y_i$$

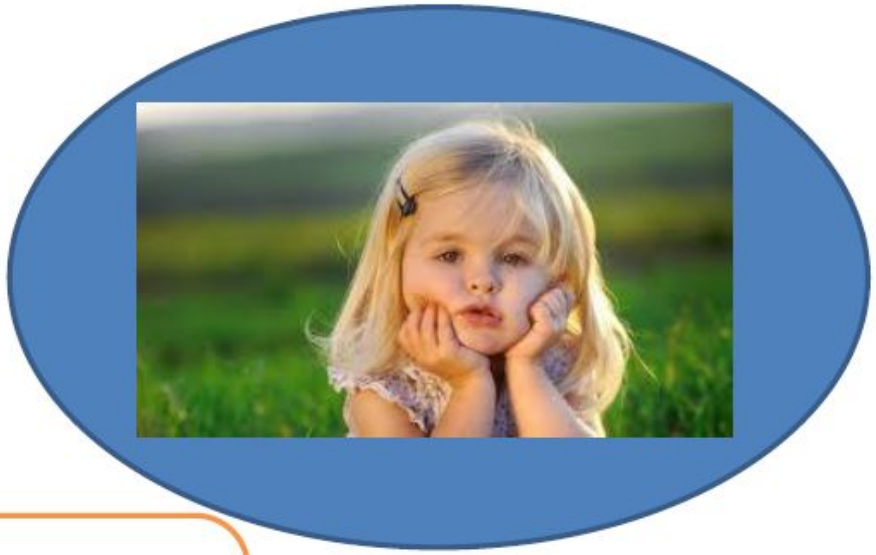
Classification: Supervised Learning

Training Phase

We have shown a set of dog pictures and a set of cat pictures to a child.

Classification: Supervised Learning

Testing Phase



DOG

This picture as it is may not be in the training set

Child has done more than just remembering

Evaluating a classifier

Classification Accuracy and Classification Error

Let the test set size be *n*.

Classification Accuracy (CA):

$$CA = \frac{\text{Number of correct predictions}}{N}$$

$$Error = \frac{\text{Number of wrong predictions}}{N}$$

CA and Error

In the previous formula we assumed that the test examples are all equally likely.

In other words, the Probability of drawing any of the test example is the same.

However, this need not be the case always.

Recap: Probability

Probability theory is built on 3 axioms.

The terms one should be clear:

1. Experiment
2. Sample space
3. Event
4. Probability

Kolmogorov Axioms of Probability

To the probabilities of outcomes/events of an experiment must obey the axioms:

- **Axiom 1**: For any event A, $\Pr(A) \geq 0$
- **Axiom 2**: $\Pr(\Omega) = 1$
- **Axiom 3**: For a collection of *mutually exclusive* events, $A_1, A_2, \ldots, A_n$

$$\Pr(A_1 \cup A_2 \ldots \cup A_n) = \sum_{i=1}^{n} \Pr(A_i)$$

- Everything else in probability theory can be deduced starting with these axioms

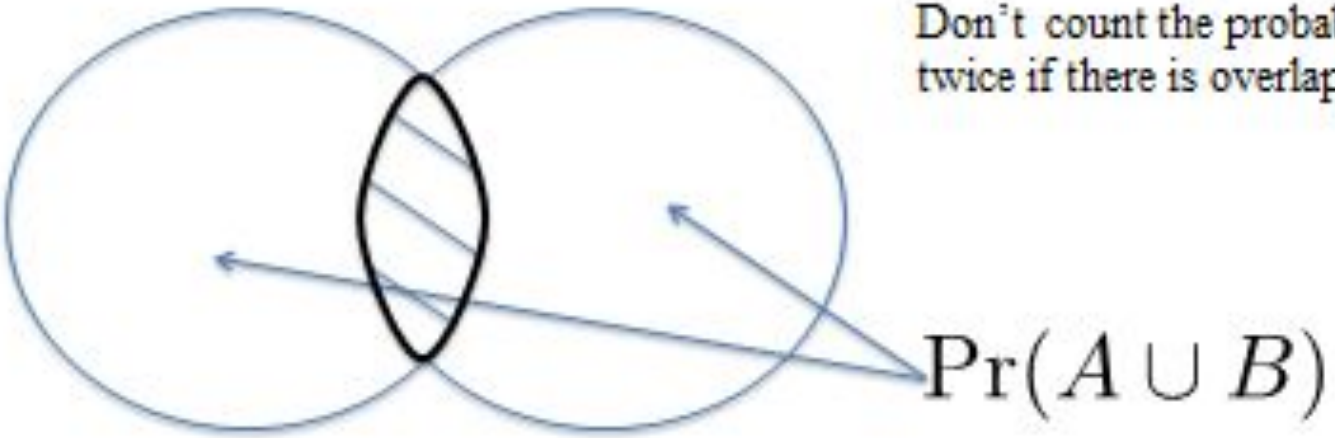
Handy Consequences of Kolmogorov Axioms

- Important consequences:
  - Probability of a union of non-disjoint events

$$\Pr(A \cup B) = \Pr(A) + \Pr(B) - \Pr(A \cap B)$$

In words: The probability of A *or* B is the probability of A *plus* the probability of B *minus* the probability of A *and* B

Don't count the probabilities of A and B twice if there is overlap between the events



$\Pr(A \cup B)$

Loss function

Let X be the given example with y being the target label (ground-truth).

Let the classifier predicted y'

Loss of this prediction, P(error|X) and P(error) :

$$\mathcal{L}(x) = \begin{cases} 0, & \text{if } y = y' \\ 1, & \text{otherwise} \end{cases}$$

$$P(\text{error}|x) = \mathcal{L}(x)$$

{This assumes that the feat. space is a discrete one}

$$P(\text{error}) = E[P(\text{error}|x)]$$

$$= \sum_{\forall x} \mathcal{L}(x)\, P(x)$$

If the space is continuous,

$$p(\text{error}|x) = L(x)$$

$$P(\text{error}) = E\left[p(\text{error}|x)\right]$$

$$= \int L(x)\, p(x)\, dx$$

{The integration is over the feat. space}

Example (discrete case)

Data point	Ground truth	Predicted label	Probability
X1	Cat	Cat	1/8
X2	Cat	Dog	1/4
X3	Dog	Dog	1/4
X4	Dog	Cat	3/8

P(error) = Average Error rate = 0(⅛) +1(¼)+0(¼) +1(⅜) = ⅝.

Note: The entire space is covered by these four examples.

Eg 2

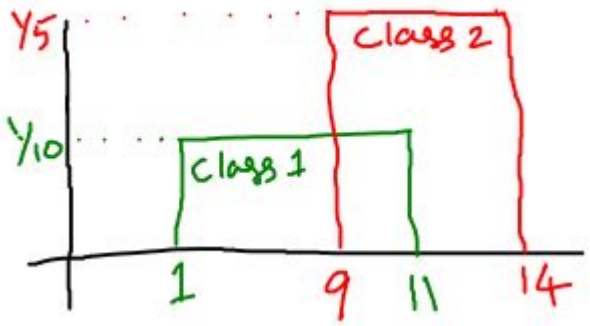
Assume we are working with 1D, two class problem. Let the class labels are 1 and 2.

Let the classifier is a simple rule: if the feature value (x) is less-than 10 the class assigned is 1, else the class assigned is 2.

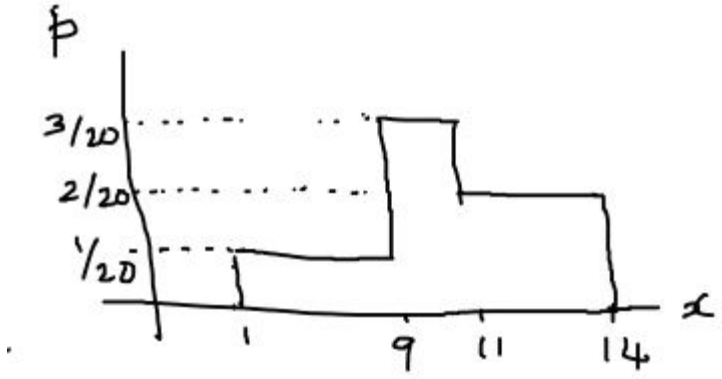
$$x < 10 \Rightarrow \text{class} = 1$$
$$x \geq 10 \Rightarrow \text{class} = 2$$

Eg 2 Continued

Assume that the data is drawn from the mixture of the two class-conditionals (shown below) where each class is equally likely.

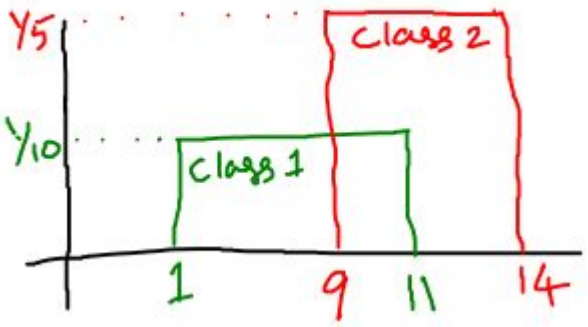


Class-conditional distributions



Distribution from which x is drawn

Eg 2



The classifier is not making any mistake for $x < 9$ and for $x > 11$. Only when x is in $[9, 11]$ one has to worry about the mistakes made.

Eg 2

[9, 10] error: Let X is an arbitrary element drawn in [9,10]

P(err | X) = P(y=class2|X)

We use the notation P(class2|X) instead of P(y=class2|X)

Eg 2

[9, 10] error: Let X is an arbitrary element drawn in [9,10]

$$P(err \mid X) = P(class2 \mid X)$$

$$P(class2 \mid X) = \frac{p(X, class2)}{p(X)}$$

$$= \frac{p(X \mid class2)\, P(class2)}{p(X)}$$

$$p(X) = p(X|class1)P(class1) + p(X|class2)P(class2)$$

So, for X in [9,10],

$$P(class2|X) = \frac{\frac{1}{5} \cdot \frac{1}{2}}{\frac{1}{5} \cdot \frac{1}{2} + (1/10).\frac{1}{2}} = \frac{2}{3}$$

[10,11] error: Let X is an arbitrary element drawn in [10,11]

$$P(\text{err} \mid X) = P(\text{class1} \mid X) = \frac{1}{3}$$

P(error)

$$P(\text{err}) = \int P(\text{err}|x)\, p(x)\, dx$$

$$= \int_{1}^{9} P(\text{class 2}|x)\, p(x)\, dx$$

$$+ \int_{9}^{10} P(\text{class 2}|x)\, p(x)\, dx$$

$$+ \int_{10}^{11} P(\text{class 1}|x)\, p(x)\, dx$$

$$+ \int_{11}^{14} P(\text{class 1}|x)\, p(x)\, dx$$

$$P(\text{error}) = 0 + \tfrac{1}{5}.\tfrac{1}{2} + 1/10.(\tfrac{1}{2}) + 0 = 3/20$$

Eg2 Contd

By shifting the threshold of the classifier can we reduce the error rate of the classifier?

If so, what would be the best possible threshold?

This actually points to the Bayes Classifier.

Actual error-rate

The actual error-rate calculation is theoretical and is possible when the distributions (Probability structure) of the given problem is known.

In reality we will not be having this.

Test set

Test set is an independently drawn set from the same distribution as the training set.

Since we are given with an example set, normal practice is to divide this into **the training set** and **the test set**.

Note that this division has to be random and in practice often disjoint division is followed.

Multiple test sets are possible

Since multiple training and test set divisions are possible, often we measure the accuracy or error as the average one over multiple divisions.

This can give us variance or standard deviation of the computed accuracy or error.

Later we will see about **bias and variance** of the learning.

Other evaluation metrics (especially for binary classification)

Total number of Dog Images (10), Cat (10), N=20

Confusion Matrix: Normally this matrix is of size *K*K* where *K* is number of classes.

	N=20	Predicted	
		Dog (+)	Cat (-)
Actual	Dog (+)	7	3
	Cat (-)	4	6

	N=20	Predicted	
		Dog (+)	Cat (-)
Actual	Dog (+)	TP	FN
	Cat (-)	FP	TN

True Positives: 7 (Dogs images were classified as Dog)

False Positives (*Type 1 Error*): 4 (Cats images were classified as Dog)

True Negatives: 6 (Cats images were classified as cat)

False Negatives (Type 2 Error): 3 (Dogs images were classified as cat)

Total (N) =TP+FP+FN+TN=20

Evaluation Metrics for a classification model

- **Accuracy**: Accuracy is number of correct predictions out of total records.

  Accuracy=(*TP*+*TN*)/Total=13/20=65%.

- Misclassification rate or error rate:

  Error rate=(*FP*+*FN*)/Total=7/20=35%.

Accuracy Paradox:

Consider, Total number of Dog Images (19), Cat (1), N=20

	N=20	Predicted	
		Dog (+)	Cat (-)
Actual	Dog (+)	19	0
	Cat (-)	1	0

	N=20	Predicted	
		Dog (+)	Cat (-)
Actual	Dog (+)	TP	FN
	Cat (-)	FP	TN

Accuracy= (TP+TN)/Total=(19+0)/20=99%

Evaluation Metrics for a Classification model

- Precision (positive predicted value): Is the fraction of documents retrieved that are relevant.
  - It is the number of positive predictions divided by the total number of positive class values predicted.

    $Precision = TP/(TP+FP) = 19/(19+1) = 19/20 = 99\%$.
  - Your classifier said something is positive, how much precise this decision is?
- Recall (sensitivity or true positive rate): It is the fraction of relevant documents that are retrieved.
  - It is the number of positive predictions divided by the number of positive class values in the test data.

    $Recall = TP/(TP+FN) = 19/(19+0) = 100\%$.
  - Fractions of positives that are actually said to be positive by your classifier.

PRECISION, RECALL ARE USED IN INFORMATION RETRIEVAL SYSTEMS. These can be used to evaluate a binary classification.

F-1 Measure

- F-1 Measure: A balanced measure between precision and recall.

  F-1 Measure=2*(Precision * Recall)/(Precision + Recall)

  F-1=2*(0.99*1.0)/(0.99+1.0)=2*(0.99)/(1.99)=1.98/1.99=0.99

Evaluation Metrics for a Classification model

- Specificity (True Negative Rate): How often the ML model predicts negative samples correctly out of total negative samples.

  Specificity=TN/(TN+FP)=0/(0+1)=0%

  -Model has 0% effective in predicting negative samples as negative.

- False positive rate (FPR): How often the ML model predict the negative samples (cat) as positive (dog)?

  FPR=FP/(TN+FP)=1/(0+1)=100%

  Note: It can also be calculated as FPR=1-Specificity=1-0=100%

  ML model has 100% FPR, which means that every time model will classifies every Negative (Cat) sample as Positive (Dog).

- *Sensitivity and Specificity are most important in medical diagnosis.*

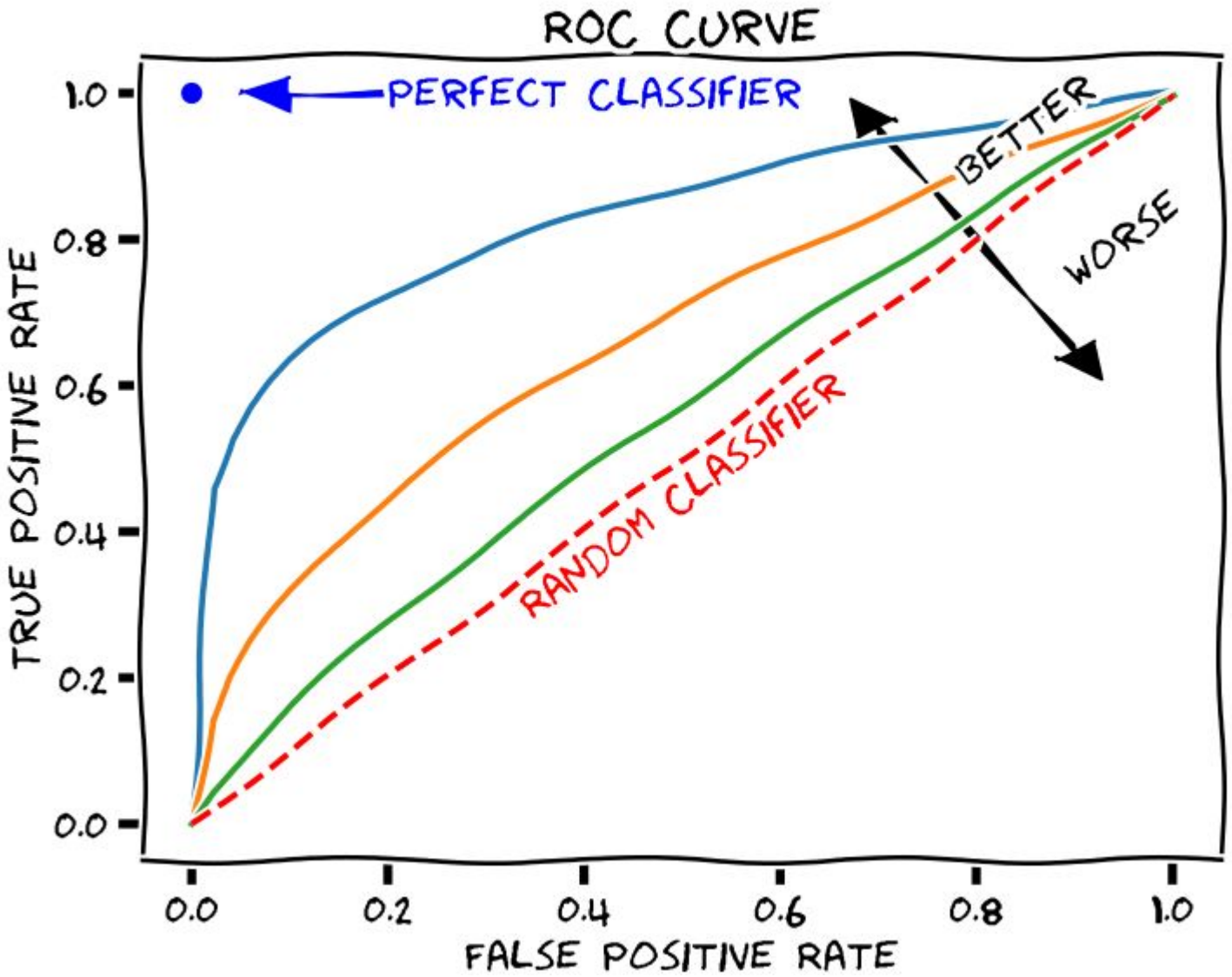
ROC Curve

- A receiver operating characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

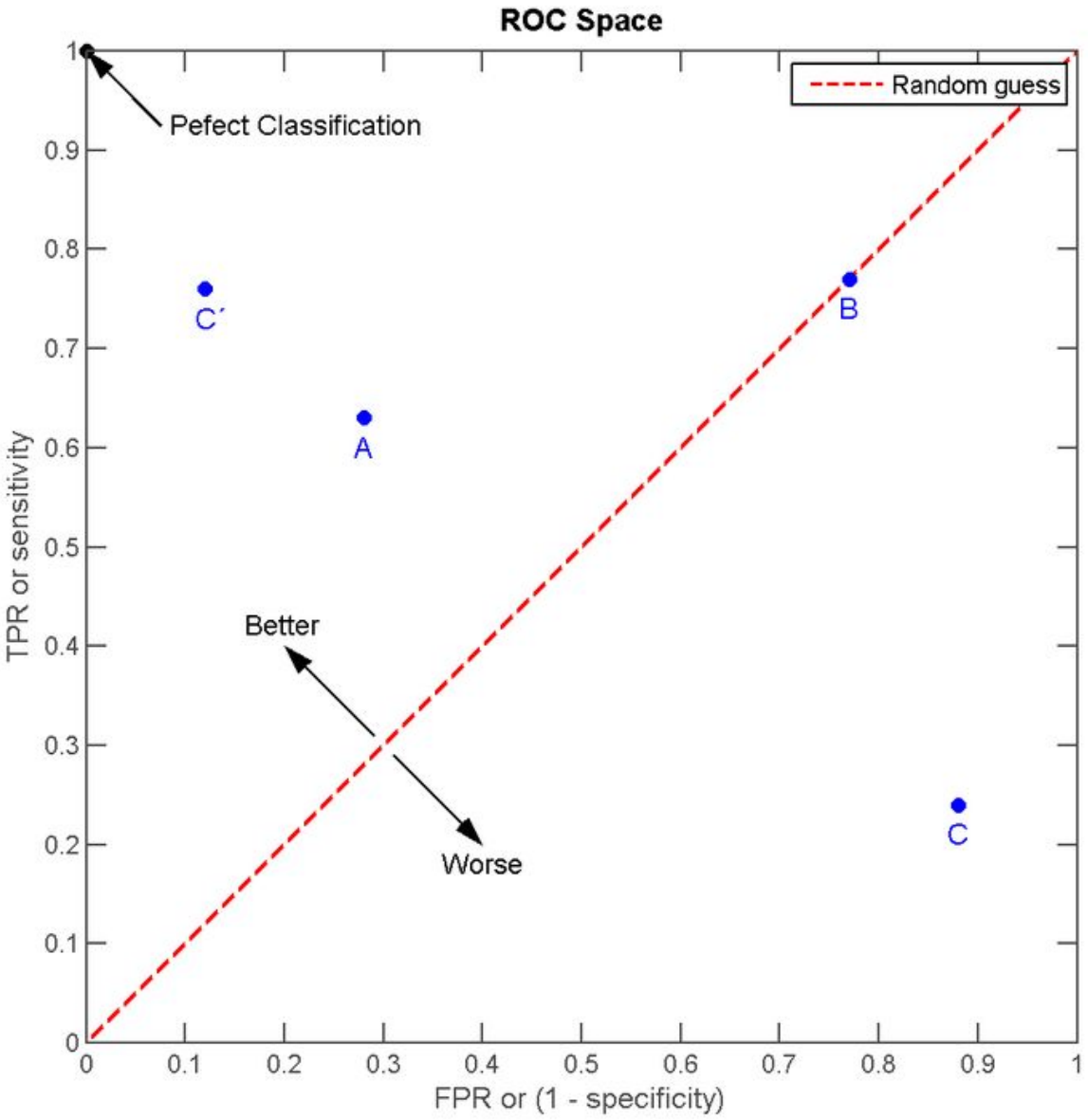
- The method was originally developed for operators of military radar receivers, which is why it is so named.

- The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

ROC Curve


ROC CURVE
PERFECT CLASSIFIER
BETTER
WORSE
RANDOM CLASSIFIER
TRUE POSITIVE RATE
FALSE POSITIVE RATE
0.0
0.2
0.4
0.6
0.8
1.0

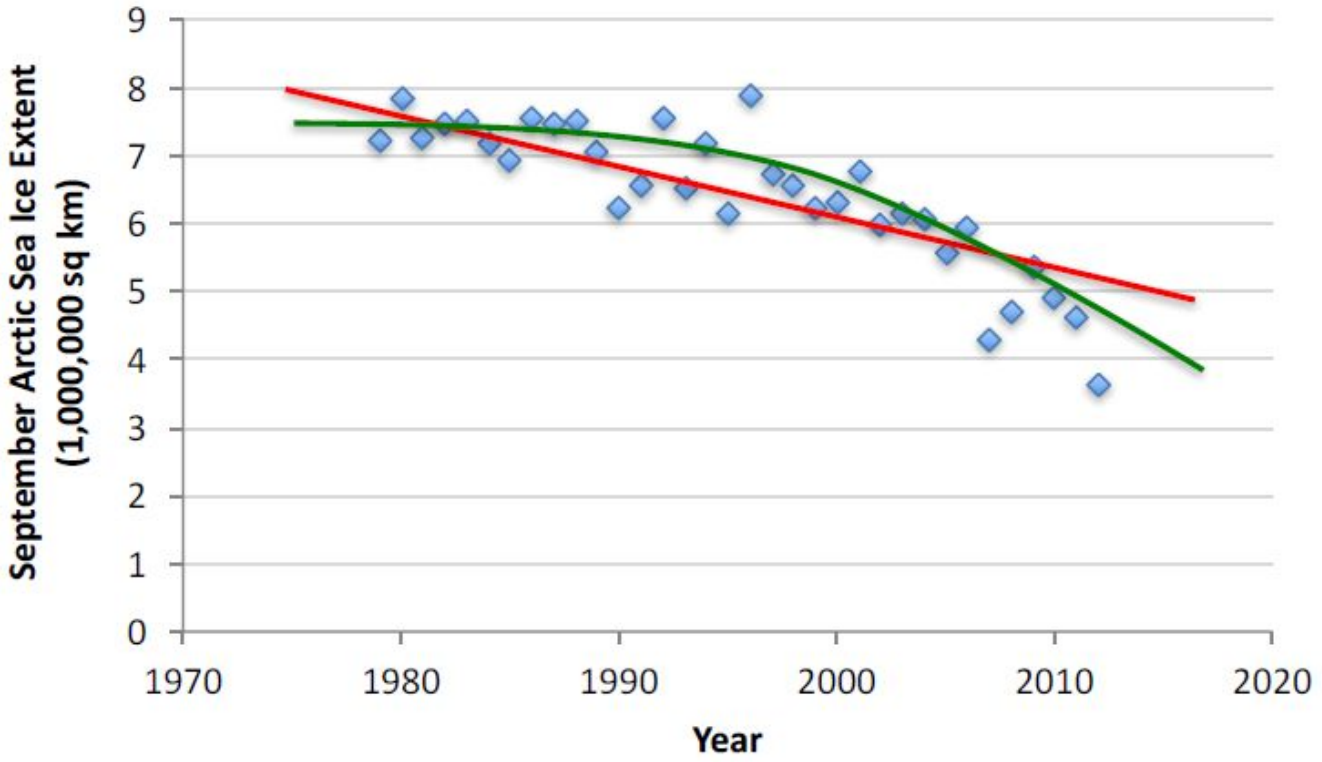
ROC Curve


ROC Space
Random guess
Pefect Classification
C´
A
B
C
Better
Worse
TPR or sensitivity
FPR or (1 - specificity)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Regression

What is the Regression Problem?

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is real-valued == regression

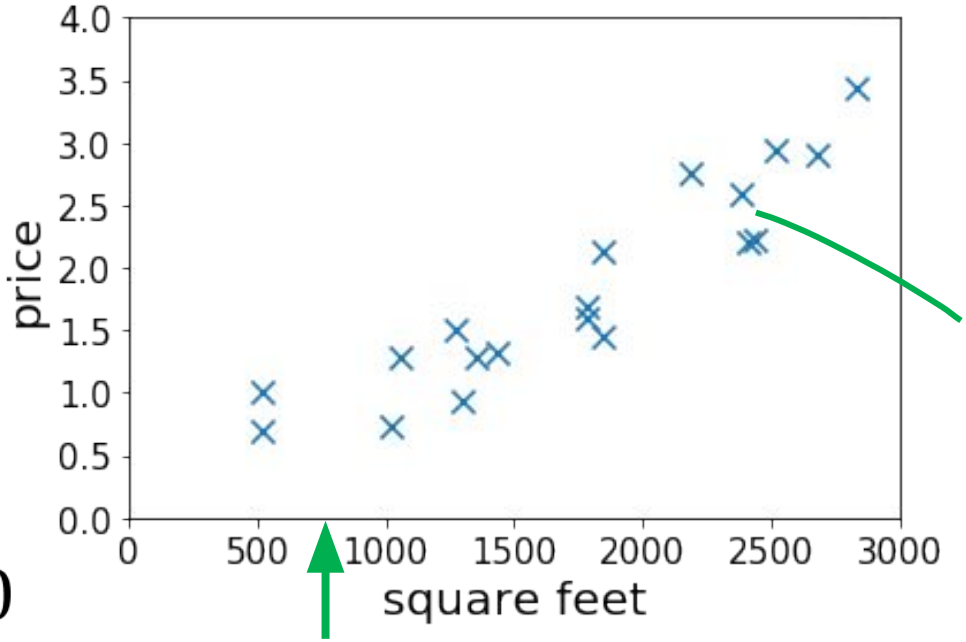


Housing Price Prediction

- Given: a dataset that contains n samples

$$\left(x^{(1)}, y^{(1)}\right), \ldots \left(x^{(n)}, y^{(n)}\right)$$

- Task: If a residence has x square feet, predict its price?



15th sample $(x^{(15)}, y^{(15)})$

$x = 800$

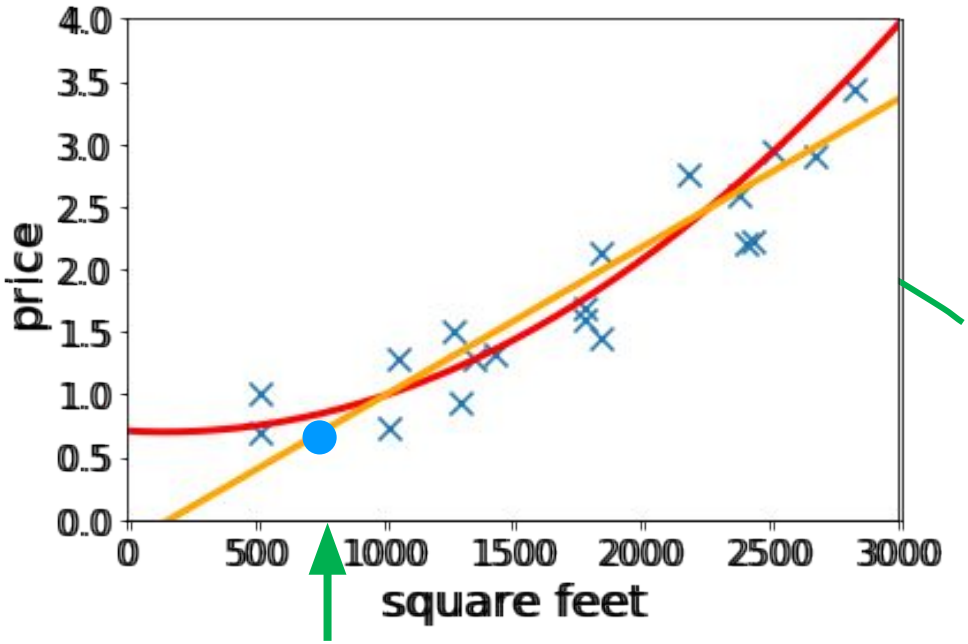
$y = ?$

Housing Price Prediction

- Given: a dataset that contains n samples

$$\left(x^{(1)}, y^{(1)}\right), \ldots \left(x^{(n)}, y^{(n)}\right)$$

- Task: If a residence has x square feet, predict its price?



15th sample $(x^{(15)}, y^{(15)})$

$x = 800$

$y = ?$

- Solution: fitting linear/quadratic functions to the dataset.

How to evaluate the regression model?

- What is the accuracy of your model prediction?
- Compared to classification (where you have discrete set of labels as prediction) predicting the accuracy in regression (where you have continuous real value number) is slightly difficult.!
- It might be impossible for your ML model to predict the exact value.
- So to calculate the accuracy, you can compared the predicted value as how close it is against the real value.

Evaluation Metrics for a Regression model

- *There are three main metrics used for evaluating the regression models:*
  - *R Square/Square of the Correlation Coefficient*
  - *Mean Square Error(MSE)/Root Mean Square Error(RMSE)*
  - *Mean Absolute Error(MAE)*

Evaluating a regression model: R square value

- *R Square/coefficient of determination:* It measures the proportion of the variance in the dependent variable that is predictable from the independent variable(s).

- *R* Square value is between 0 to 1 and bigger value indicates a better fit between prediction and actual value

$$R^2 = 1 - \frac{SS_{Regression}}{SS_{Total}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Where y_i is the original value, $\hat{y}_i$ is the predicted value and $\bar{y}$ is the mean of original values.

Evaluating a regression model: Mean Squared Error (MSE)

- MSE is calculated by the sum of square of prediction error which is real output minus predicted output and then divide by the number of data points.
- It gives you an absolute number on how much your predicted results deviate from the actual number.
- Root Mean Square Error(RMSE) is the square root of MSE.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Where y_i is the original value and $\hat{y}_i$ is the predicted value.

Evaluating a regression model: Mean Absolute Error (MAE)

- Mean Absolute Error(MAE) is similar to Mean Square Error(MSE).
- Unlike MSE where we take the sum of square of errors, MAE computes the sum of absolute value of error.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

Where y_i is the original value and $\hat{y}_i$ is the predicted value.

- Note: MSE gives larger penalisation to big prediction error by square it while MAE treats all errors the same.

Dataset: Train, Validation and Test sets

- **Train set:** The model is initially fit on a training dataset,[3] which is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model.

- In practice, the training dataset often consists of pairs of an input vector (or scalar) and the corresponding output vector (or scalar), where the answer key is commonly denoted as the target (or label).

Dataset: Train, Validation and Test sets

- **Validation set:** Successively, the fitted model is used to predict the responses for the observations in a second dataset called the validation dataset.

- The validation dataset provides an unbiased evaluation of a model fit on the training dataset while tuning the model's hyperparameters e.g. the number of hidden units (layers and layer widths) in a neural network.

- The validation dataset functions as a hybrid: it is training data used for testing, but neither as part of the low-level training nor as part of the final testing.

Dataset: Train, Validation and Test sets

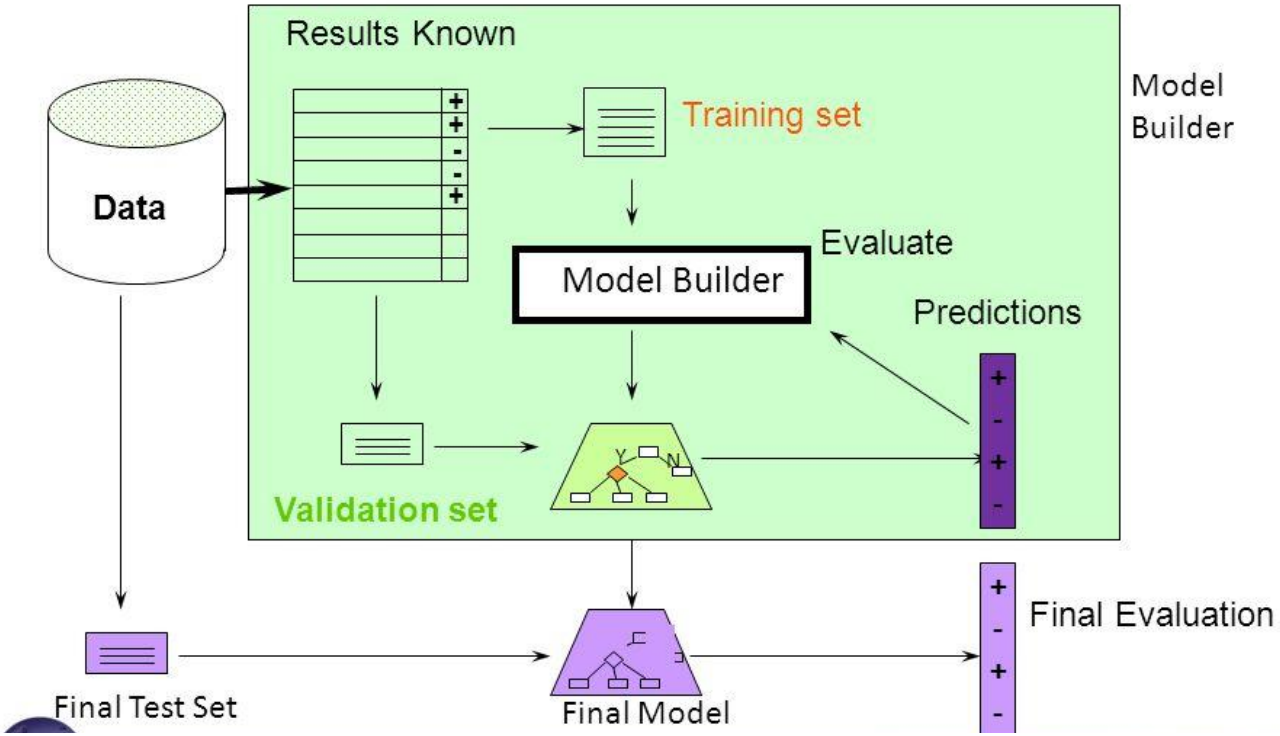
- **Test set:** Finally, the test dataset is a dataset used to provide an unbiased evaluation of a final model fit on the training dataset.

- The test dataset is typically used to assess the final model that is selected during the validation process.

- It is only used once a model is completely trained(using the train and validation sets).
  - Many a times the validation set is used as the test set, but it is not good practice.

- The test set generally contains carefully sampled data that spans the various classes that the model would face, when used in the real world.

Train, test and validation split



Data Sampling Methods

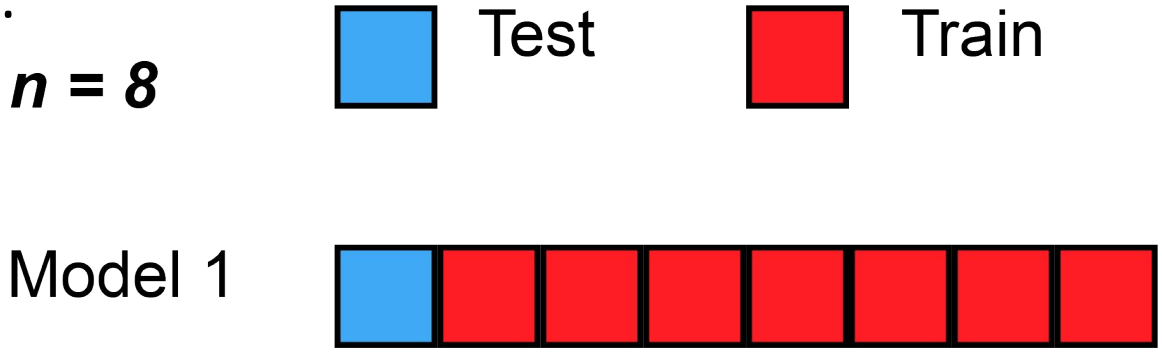
- **Hold out method**: In the holdout method, we randomly assign data points to two sets d0 and d1, usually called the training set and the test set, respectively.

- The size of each of the sets is arbitrary although typically the test set is smaller than the training set.

Data Sampling Methods

- **Cross Validation Method**: Cross-validation involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (Validation set).

- To reduce variability, in most methods multiple rounds of cross-validation are performed using different partitions, and the validation results are combined (e.g. averaged) over the rounds to give an estimate of the model's predictive performance.



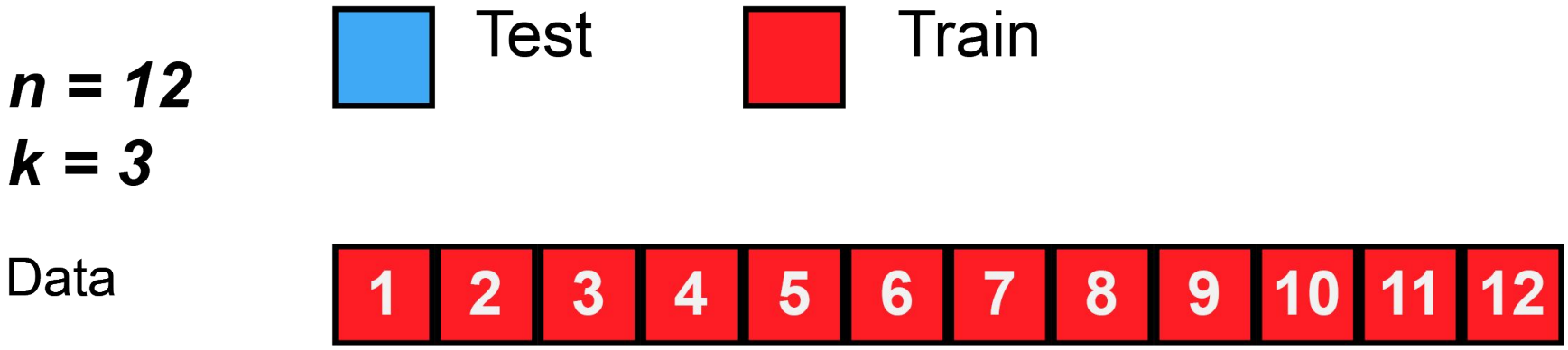
Data Sampling Methods

k-fold Cross validation: This procedure has a single parameter called “k” that refers to the number of groups that a given data sample is to be split into.

The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    1. Take the group as a hold out or test data set
    2. Take the remaining groups as a training data set
    3. Fit a model on the training set and evaluate it on the test set
    4. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores

k-fold Cross validation:



Overfit vs Underfit

- **Overfitting:** refers to a model that models the training data too well.

- Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data.

- In overfitting, ML model learns the inference rules based on the noise or random fluctuations in the training data.

Overfit vs Underfit

- **Underfitting:** refers to a model that can neither model the training data nor generalize to new data.

- An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

- Underfitting is often not discussed as it is easy to detect given a good performance metric. The remedy is to move on and try alternate machine learning algorithms.

Thank You:
Question?

Machine Learning

Non-parametric Algorithms: k-NN Classifier and Parzen Window

Indian Institute of Information Technology
Sri City, Chittoor

Agenda

- Parzen Window
- Defining R_n (region in the feature-space)
- Two different approaches - fixed volume vs. fixed number of samples in a variable volume
- Example 3D hypercube
- The window function and estimation
- Critical parameters of the Parzen-window technique: window width and kernel
- Selecting Window
- Selecting Kernel

Introduction

Probability density estimation is one of the oldest problems in statistics and machine learning.

There are two approaches, viz.,

1. Parametric, and
2. Non-parametric.

Parametric Density Estimation

- We assume a parametric class (the form) from which the data is drawn. For eg., Gaussian distribution.
- Then we try to estimate the parameters of the Gaussian distribution ie., mean and covariance matrix.

- For doing the parameter estimation we use the training examples.

We study about this Later.

Non-parametric Methods

No assumption is made about the form of the distribution.

Depends totally on the data set.

Non-parametric Methods

1. Parzen Window based
2. Nearest Neighbors based

Parzen window

- The Parzen-window method (also known as Parzen-Rosenblatt window method) is a widely used non-parametric approach to estimate probability density p(x), for a specific point x.
- Notation: The estimate of p(x) when we use dataset of size n is denoted by $p_n(x)$.
- It doesn't require any knowledge or assumption about the underlying distribution.
- A popular application of the Parzen-window technique is to estimate the class-conditional densities (or also often called 'likelihoods').
- Likelihoods, $p(x \mid \omega_i)$ in a supervised pattern classification problem from the training dataset (where p(x) refers to the probability density that the sample x belongs to the particular class ω_i).

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

Where would this method be useful?

- Imagine that we are about to design a Bayes classifier for solving a statistical pattern classification task using Bayes' rule:

$$P(\omega_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid w_i) \cdot P(\omega_i)}{p(\boldsymbol{x})}$$

$$\Rightarrow posterior\ probability = \frac{likelihood \cdot prior\ probability}{evidence}$$

- If the parameters of the the class-conditional densities (also called likelihoods) are known, it is pretty easy to design the classifier.
- Imagine we are about to design a classifier for a pattern classification task where the parameters of the underlying sample distribution are not known.
- Therefore, we wouldn't need the knowledge about the whole range of the distribution; it would be sufficient to know the probability of the particular point, which we want to classify, in order to make the decision.

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

Parzen Window

- In parzen window we are going to see how we can estimate this probability from the training sample.

- However, the only problem of this approach would be that we would seldom have exact values - if we consider the histogram of the frequencies for a arbitrary training dataset.

- Therefore, we define a certain region (i.e., the Parzen-window) around the particular value to make the estimate.

[1] *Parzen, Emanuel*. On Estimation of a Probability Density Function and Mode. The Annals of Mathematical Statistics 33 (1962), no. 3, 1065 1076.
[2] *Rosenblatt, Murray*. Remarks on Some Nonparametric Estimates of a Density Function. The Annals of Mathematical Statistics 27 (1956), no. 3, 832–837.

The most fundamental technique

- The probability P that a vector x will fall in a region R is given by

$$P = \int_{\mathcal{R}} p(\mathbf{x}')\, d\mathbf{x}'.$$

- $p(x) \approx \frac{\left(\int_R\ p(x')dx'\right)}{V}$ where V is the volume of the region R

Defining the Region R_n

- The basis of this approach is to count how many data-points fall within a specified region R_n (or "window"). Our intuition tells us, that (based on the observation), the probability that one sample falls into this region is:

$$P = \frac{\#\ of\ samples\ in\ R}{Total\ samples}$$

- The probability of observing k points out of n in a Region R_n : we consider **a binomial distribution:**

$$P_k = \binom{n}{k} \cdot P^k \cdot (1-P)^{n-k}$$

- In a binomial distribution, the probability peaks sharply at the mean

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

Defining the Region Rn

- Let p(x) be the probability density at x. Let over the small region R it is uniformly distributed.

$$P = \int_R p(x')\,dx' = p(x) \cdot V$$

where V is the volume of the region R, and if we rearrange those terms, so that we arrive at the following equation:

$$\frac{k}{n} = p(x) \cdot V$$

$$\Rightarrow p(x) = \frac{k/n}{V}$$

- This simple equation above (i.e, the "probability estimate") lets us calculate the probability density of a point x by counting how many points k fall in a defined region (or volume).

Credits: Sebastian Raschka

To estimate the density at $\mathbf{x}$, we form a sequence of regions $\mathcal{R}_1, \mathcal{R}_2, ...$, containing $\mathbf{x}$ — the first region to be used with one sample, the second with two, and so on. Let V_n be the volume of $\mathcal{R}_n$, k_n be the number of samples falling in $\mathcal{R}_n$, and $p_n(\mathbf{x})$ be the nth estimate for $p(\mathbf{x})$:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n}.$$

If $p_n(\mathbf{x})$ is to converge to $p(\mathbf{x})$, three conditions appear to be required:

- $\lim_{n\to\infty} V_n = 0$
- $\lim_{n\to\infty} k_n = \infty$
- $\lim_{n\to\infty} k_n/n = 0.$

Theoretically it can be shown that

- V_n can be reduced as n increases: Like $V_n = 1/\sqrt{n}$. Starting with $V_1 = 1$.
- Or, k_n can be increased as n increases: Like $k_n = \sqrt{n}$. Here the volume of the region is increased to fit the k_n points exactly.

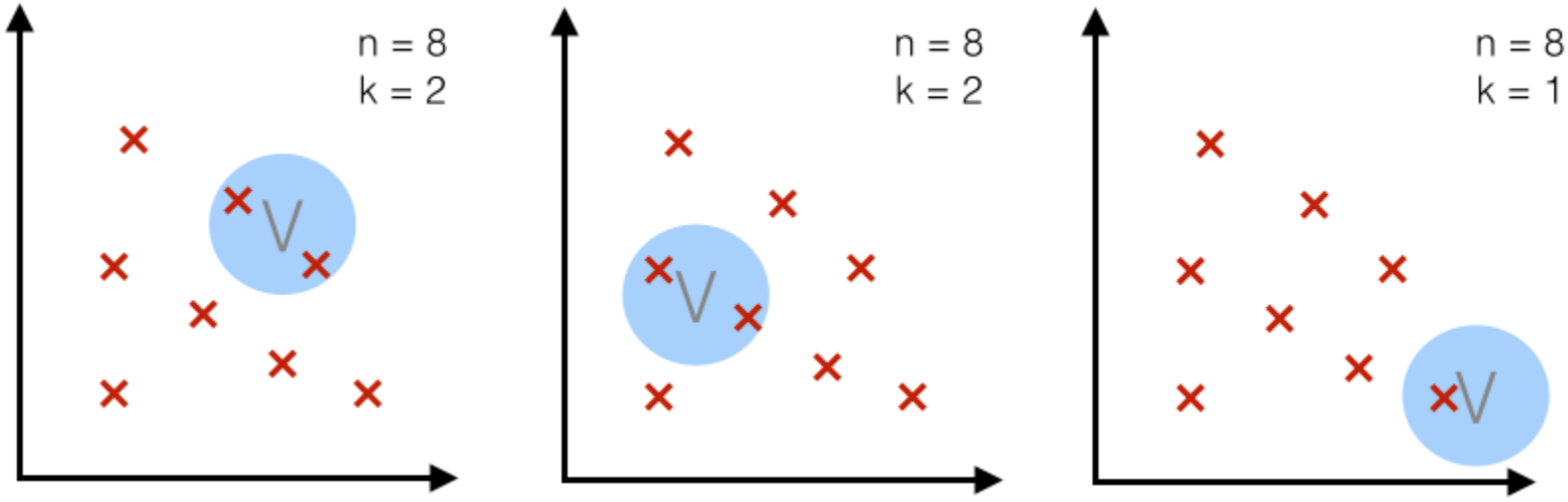
- Both these approaches are shown to converge to the true density *asymptotically*.

In Practice: Two approaches followed are-

Two different approaches - fixed volume vs. fixed number of samples in a variable volume

Case 1 - fixed volume:

- For a particular number n (= number of total points), we use volume V of a fixed size and observe how many points k fall into the region.

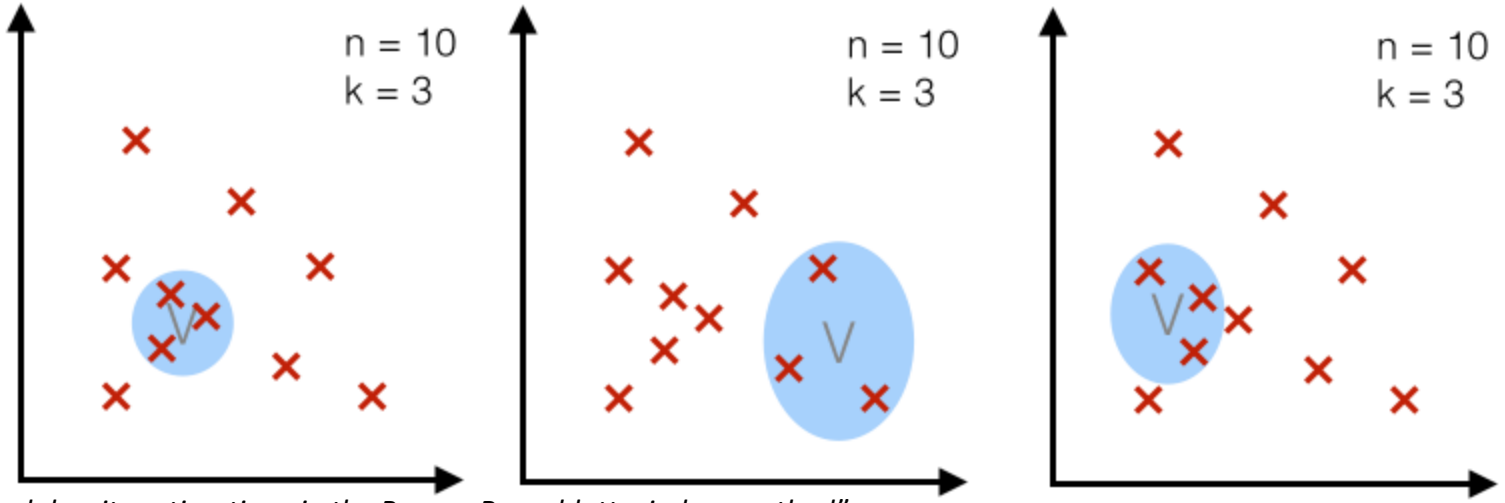


Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

Two different approaches - fixed volume vs. fixed number of samples in a variable volume

Case 2 - fixed *k*:

- For a particular number n (= number of total points), we use a fixed number k (number of points that fall inside the region or volume) and adjust the volume accordingly..



Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

- The second approach, fixed k, is nothing but the k-Nearest Neighbor Classifier.
- We will see about this, in detail later.

Example 3D-hypercubes

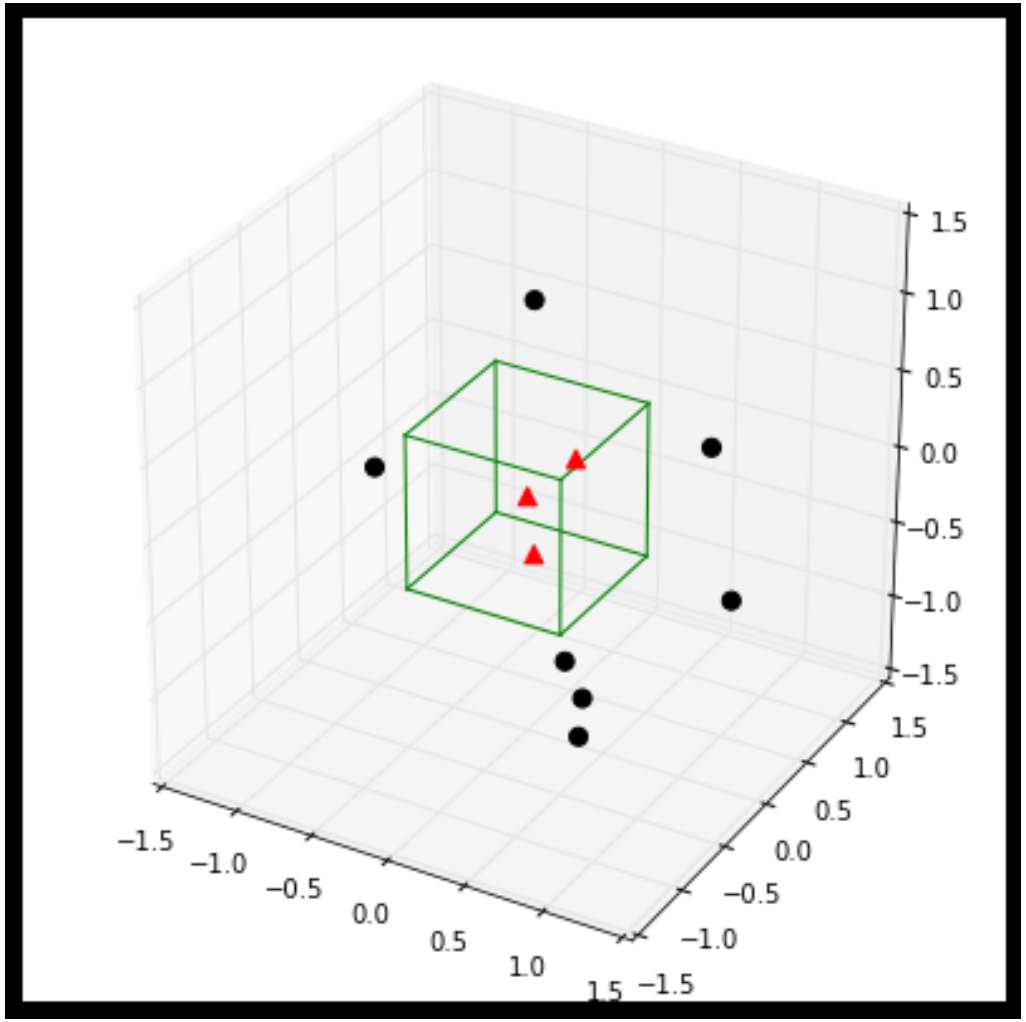
- To illustrate this with an example and a set of equations, let us assume this region R_n is a hypercube.

- The volume of this hypercube is defined by $V_n = h_n^d$, where h_n is the length of the hypercube, and d is the number of dimensions.

- For an 2D-hypercube with length 1, for example, this would be $V_1 = 1^2$ and for a 3D hypercube $V_1 = 1^3$, respectively.

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

Example: A typical 3-dimensional unit hypercube ($h_1 = 1$) representing the region R_1, and 10 sample points, where 3 of them lie within the hypercube (red triangles), and the other 7 outside (black dots).



Credits: "Kernel density estimation via the Parzen-Rosenblatt window method"
By Sebastian Raschka

- Each point falling within the window (hyper-cube) contributes to the density.
- Points falling outside will not.
- We can formalize this in to a window function.

The window function

- Once we visualized the region R_1 like above, it is easy and intuitive to count how many samples fall within this region, and how many lie outside.
- To approach this problem more mathematically, we would use the following equation to count the samples k_n within this hypercube, where φ is our so-called window function.

$$\phi(\boldsymbol{u}) = \begin{cases} 1 & |u_j| \leq 1/2\,; \qquad j = 1, \ldots, d \\ 0 & otherwise \end{cases}$$

for a hypercube of unit length 1 centered at the coordinate system's origin.

- If we extend on this concept, we can define a more general equation that applies to hypercubes of any length h_n that are centered at *x*:

$$k_n = \sum_{i=1}^{n} \phi\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h_n} \right)$$

$$\text{where } \boldsymbol{u} = \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h_n} \right)$$

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method" By Sebastian Raschka

Parzen-window estimation

- we can now formulate the Parzen-window estimation with a hypercube kernel as follows:

$$p_n(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h^d} \phi \left[\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h_n} \right]$$

where

$$h^d = V_n \quad \text{and} \quad \phi \left[\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h_n} \right] = k$$

- And applying this to our unit-hypercube example above, for which 3 out of 10 samples fall inside the hypercube (into region *R*), we can calculate the probability *p(**x**)* that ***x*** samples fall within region *R* as follows:

$$\boldsymbol{x} = \frac{k/n}{h^d} = \frac{3/10}{1^3} = \frac{3}{10} = 0.3$$

Credits: "Kernel density estimation via the Parzen-Rosenblatt window method" By Sebastian Raschka

An important observation

- A point falling slightly outside the window will not contribute to the density.
- This is incorrect.
- Also, intuitively, very near point to x should contribute more to the density than far point (even though both are within the window).

An important observation

- Each point in the training set should contribute to density.
  - Near contributes more than far.
- This gives smooth estimates
- This avoids selecting the window width problem.
- Empty window problem can be avoided.

An important Observation

- For p(x), let near-by points contribute more, and far-away points less.
- For example one can use a Gaussian function to do this.

An important Observation

- Assume that a Gaussian N(0,1) is kept at each of the data points. For example, let x_i be the data point.
- Let the dataset size is n.
- Then contribution of x_i to $p(x)$ will be
-

$$\varphi(x - x_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-x_i)^2}$$

- If x is a multivariate data point, say with d dimensions, then

$$\phi(x - x_i) = \frac{1}{(2\pi)^{d/2}} \exp\left[-\frac{1}{2} \|x - x_i\|^2 \right]$$

The Gaussian Kernel

- The estimation of $p(x)$ when we have n data points is:

$$p_n(x) = \frac{1}{n}\sum_{i=1}^{n}\phi(x - x_i) = \frac{1}{n \cdot (2\pi)^{d/2}}\sum_{i=1}^{n}\exp\left[-\frac{1}{2}\|x - x_i\|^2\right]$$

- This approach is called "Parzen-window density estimation using the Gaussian Kernel"

Note, division by the volume (of the hypercube !) V is not appearing here. Since the contribution of each data point to density is itself density.

- Classification example

In classifiers based on Parzen-window estimation:

- We estimate the densities for each category and classify a test point by the label corresponding to the maximum posterior

- The decision region for a Parzen-window classifier depends upon the choice of window function as illustrated in the following figure.

In the next Lecture we will see…

In Practice, K-NNC is the most used classifer (which is nothing but a non-parametric density estimation based method only!!)

Thank You:
Question?

K-Nearest Neighbor Classifier

A non-parametric density estimation based classifier

Introduction

- K-Nearest Neighbor Classifier (k-NNC) is a simple classifier.
- It does not have a design phase, except for the value of k and the distance measure.
- Many people think that this is a weak classifier
  - That is, not a good one.
- In contrary, this is a strong classifier with well established asymptotic bounds.

- Given x to be classified –
- Let the set of classes be $\Omega = \{\omega_1, \dots, \omega_c\}$.
- We find $P(\omega_i|x)$ for all classes, i.e., $i = 1,2, \dots, c$.
- Note that $P(\omega_i|x)$ is the Posterior Probability that x belong to the class ω_i
- The classifier's decision is the class-label ω_{max} where $P(\omega_{max}|x) \geq P(\omega_i|x)$ for all i.

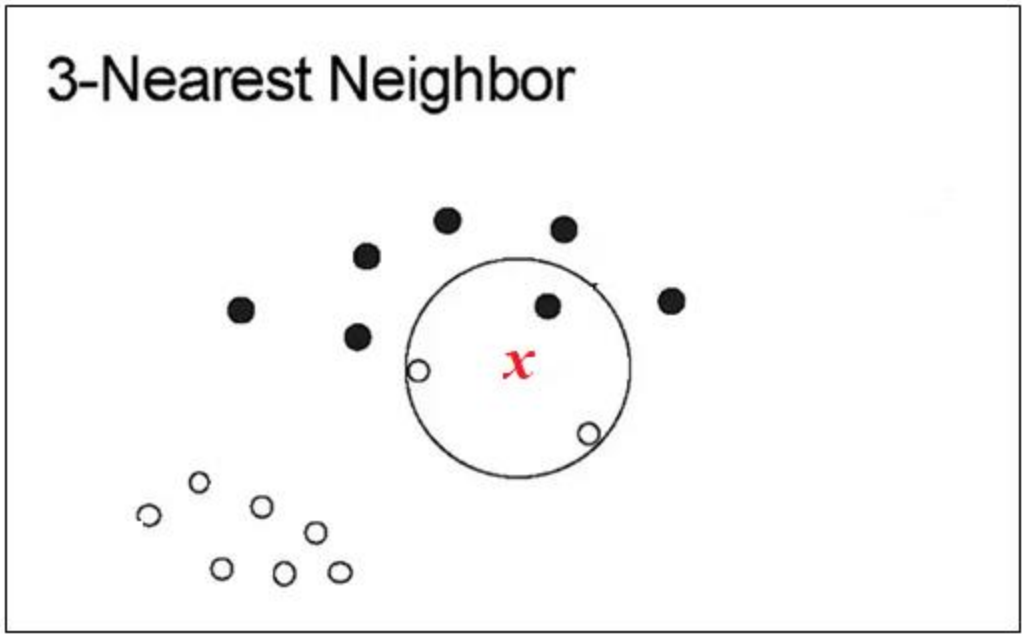
Posterior Probability

- We first find class-conditional densities, in order to find the Posterior Probability.
- To estimate the class-conditional $p(x|\omega_i)$ we do:

Class-conditional

- To estimate the class-conditional $p(x|\omega_i)$ we do:
  - For the given positive integer k , draw a hyper-sphere at x such that exactly k points are within the sphere.
  - count the number of points from class ω_i falling in the sphere. Let this be k_i
  - Let the number of training examples in class ω_i be n_i
  - Let the total number of training examples be n

- Then $p(x|\omega_i) = \dfrac{k_i}{n_i \cdot V}$
- Here, V is the volume of the hyper-sphere.



In the example,

$$p(x|Black) = \frac{1}{7 \cdot V}$$

$$p(x|White) = \frac{2}{9 \cdot V}$$

Apriori Probabilities

- We do: $P(\omega_i) = \frac{n_i}{n}$
- So in the example $P(Black) = \frac{7}{16}$, and $P(White) = \frac{9}{16}$.

Posterior Probability

- $P(\omega_i|x) = \frac{p(x|\omega_i)P(\omega_i)}{p(x)} = \frac{\frac{k_i}{n_i V}\cdot\frac{n_i}{n}}{p(x)}$

$$= \frac{k_i}{Vnp(x)}$$

- Decision : $argmax_\omega\{P(\omega_1|x), \dots, P(\omega_c|x)\}$
- Decision : $argmax_\omega\{\frac{k_1}{Vnp(x)}, \dots, \frac{k_c}{Vnp(x)}\}$

- Decision : $argmax_{\omega}\{\frac{k_1}{Vnp(x)}, \ldots, \frac{k_c}{Vnp(x)}\}$
- Decision : $argmax_{\omega}\{k_1, \ldots, k_c\}$

k-NNC is an approximation of Bayes classifier!

Mathematically it can be shown that when $n \to \infty$, $k \to \infty$ and $k/n \to 0$, k-NNC is exactly the Bayes classifier.

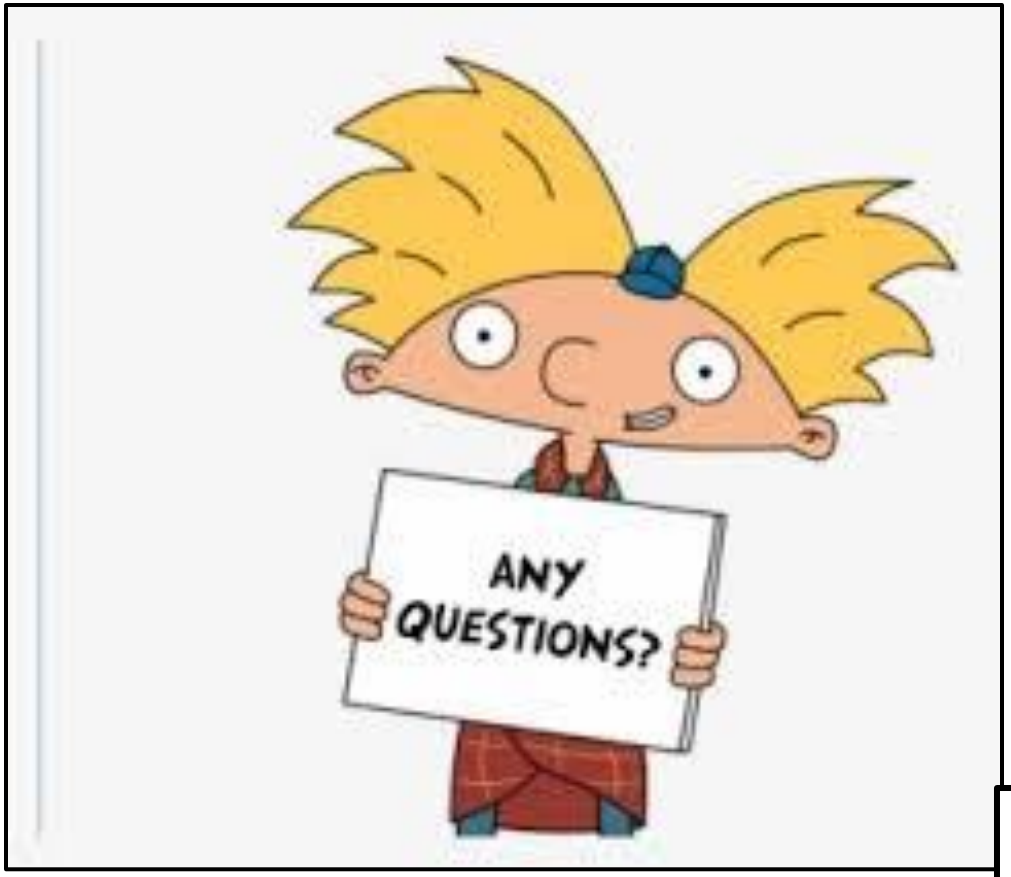
When $k = 1$, k-NNC is simply called the *nearest neighbor classifier (NNC)*.

- Asymptotically, it can be shown that the error of the NNC is less than twice the error of the Bayes Classifier.

- Cross-validation can be used to find the value of k.

Problems with k-NNC

- Test time.
  - Scanning of the entire training set is required.
- Test space
  - Entire training set has to be stored.
- Both time and space complexities of k-NNC (assuming that the k value is a small constant): $O(n)$.
- This is quiet heavy. For Neural networks this is $O(1)$.


ANY
QUESTIONS?

Bayesian Decision Theory

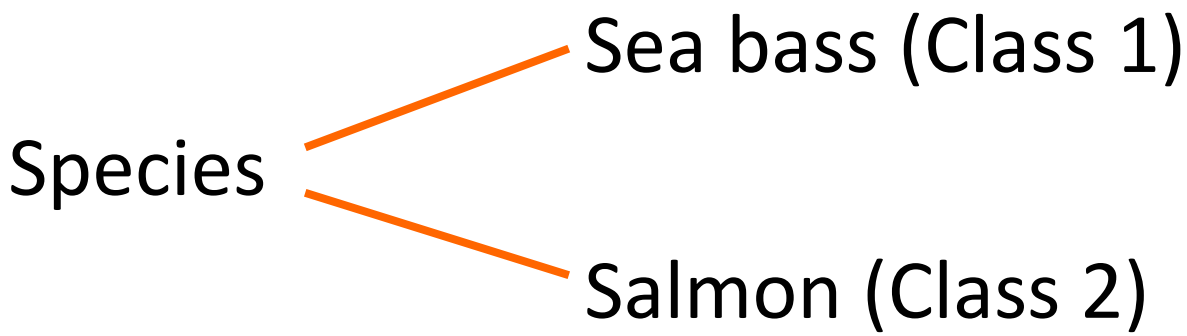
Primary source of reference: *Pattern Classification* – Duda and Hart

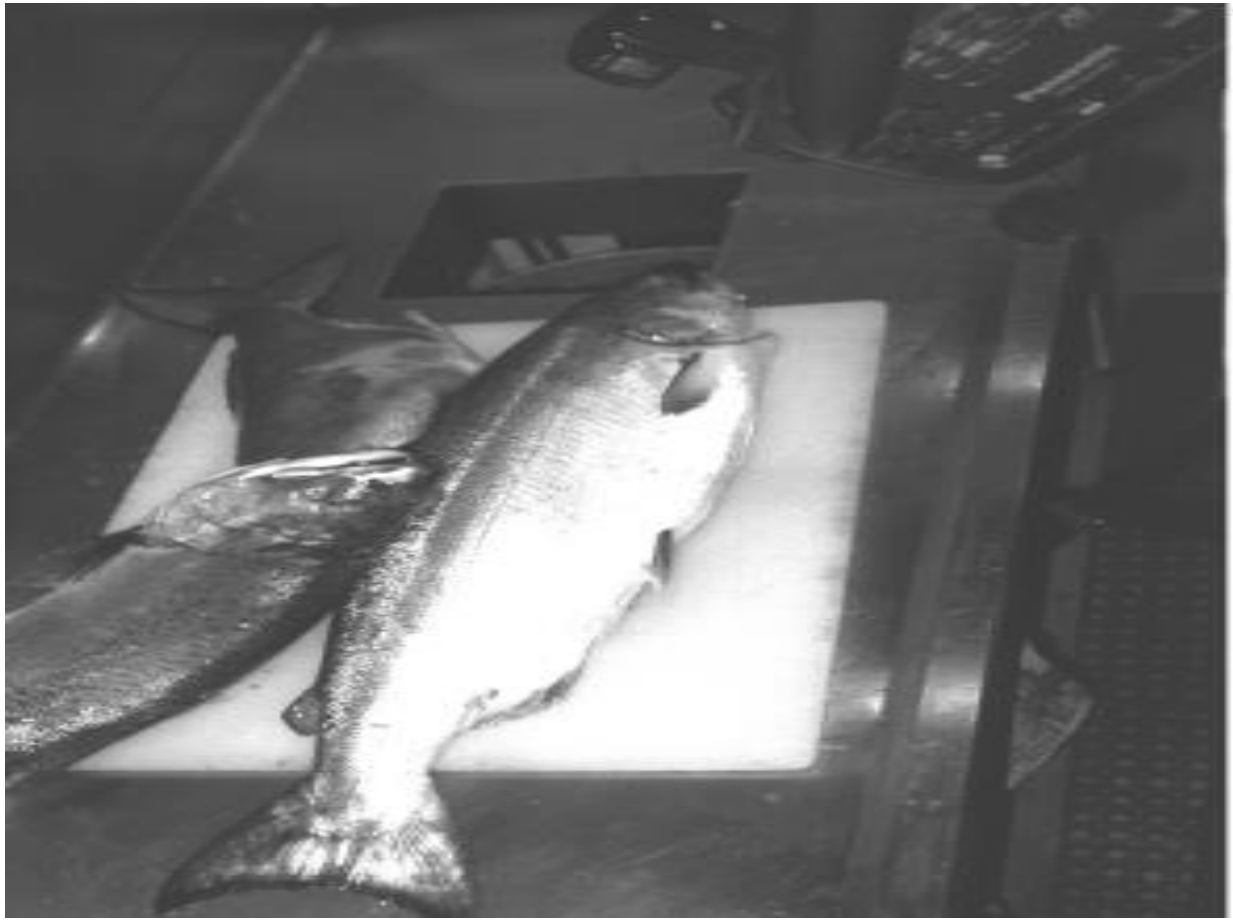
Introduction

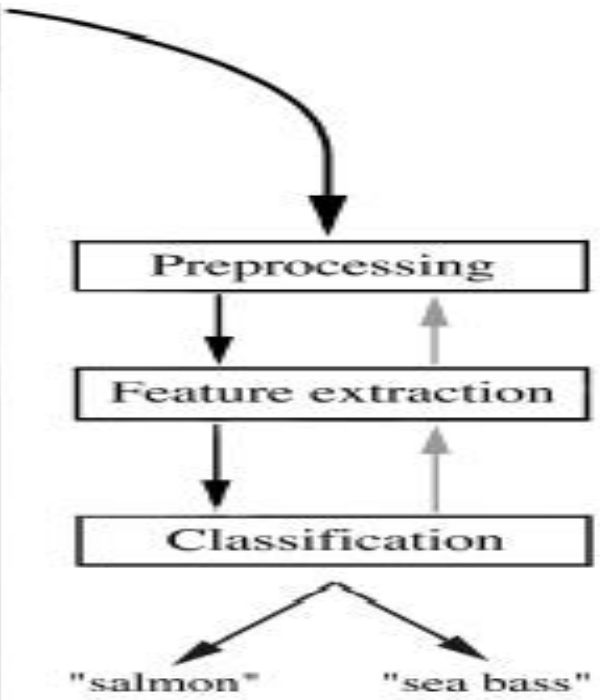
Bayesian Decision Theory–Continuous Features

An Example

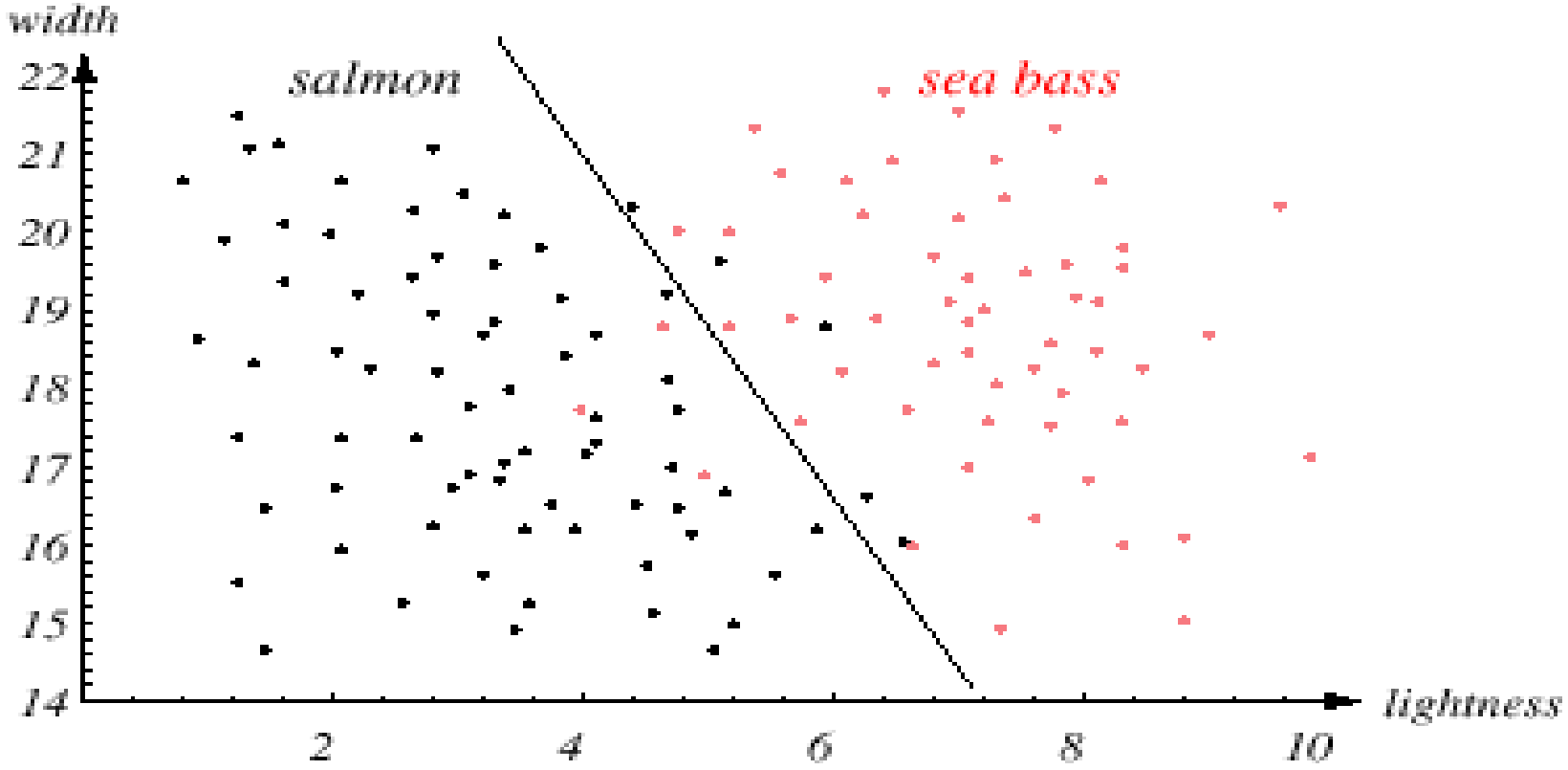
- "Sorting incoming Fish on a conveyor according to species using optical sensing"






Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

- Problem Analysis
  - Set up a camera and take some sample images to extract features like
    - Length of the fish
    - Lightness (based on the gray level)
    - Width of the fish


width
salmon
sea bass
lightness
22
21
20
19
18
17
16
15
14
2
4
6
8
10

Introduction

- The sea bass/salmon example (a two class problem)

- For example if we randomly catch 100 fishes and out of this if 75 are *sea bass* and 25 are *salmon*.
- Let the rule, in this case is: For any fish say its class is *sea bass*.
- What is the error rate of this rule?
- This information which is independent of feature values is called apriori knowledge.

- Let the two classes are ω_1 and ω_2
  - $P(\omega_1) + P(\omega_2) = 1$
  - State of nature (class) is a random variable
  - If $P(\omega_1) = P(\omega_2)$, we say it is of uniform priors
    - The catch of salmon and sea bass is equi-probable

- Decision rule with only the prior information
  - Decide ω_1 if $P(\omega_1) > P(\omega_2)$, otherwise decide ω_2
- *This is not a good classifier.*
- *We should take feature values into account !*
- *If x is the pattern we want to classify, then use the rule:*

If $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ *then assign class* ω_1
Else assign class ω_2

- $P(\omega_1 \mid x)$ *is called posteriori probability of class* ω_1 *given that the pattern is x.*

Bayes rule

- From data it might be possible for us to estimate $p(x \mid \omega_j)$, *where i = 1 or 2.* These are called class-conditional distributions.
- Also it is easy to find apriori probabilities $P(\omega_1)$ and $P(\omega_2)$. How this can be done?
- Bayes rule combines apriori probability with class conditional distributions to find posteriori probabilities.

Bayes Rule

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

This is Bayes Rule

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, **53:370-418**

$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j) \,.\, P(\omega_j)}{p(x)}$$

- Where in case of two categories

$$p(x) = \sum_{j=1}^{j=2} p(x \mid \omega_j) P(\omega_j)$$

- Posterior = $\frac{\text{Likelihood . Prior}}{\text{Evidence}}$

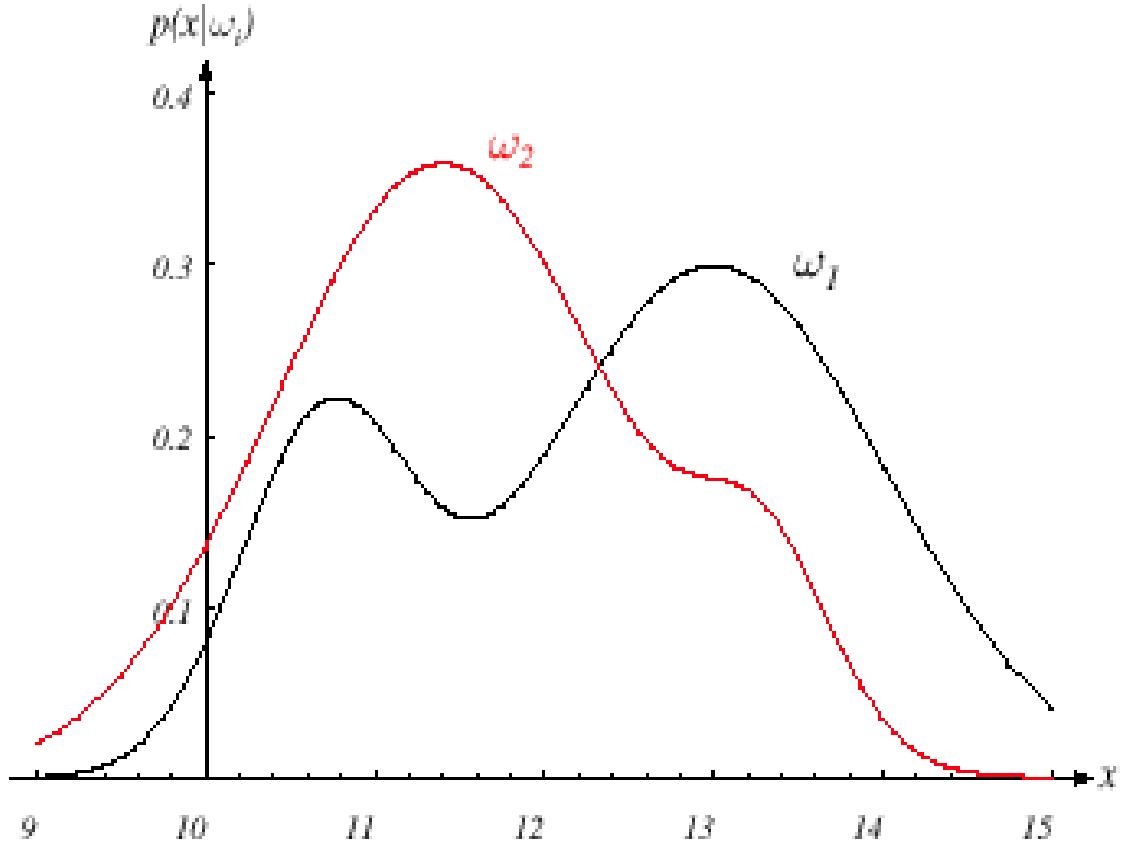


FIGURE 2.1. Hypothetical class-conditional probability density functions show the probability density of measuring a particular feature value x given the pattern is in category ω_j. If x represents the lightness of a fish, the two curves might describe the difference in lightness of populations of two types of fish. Density functions are normalized, and thus the area under each curve is 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

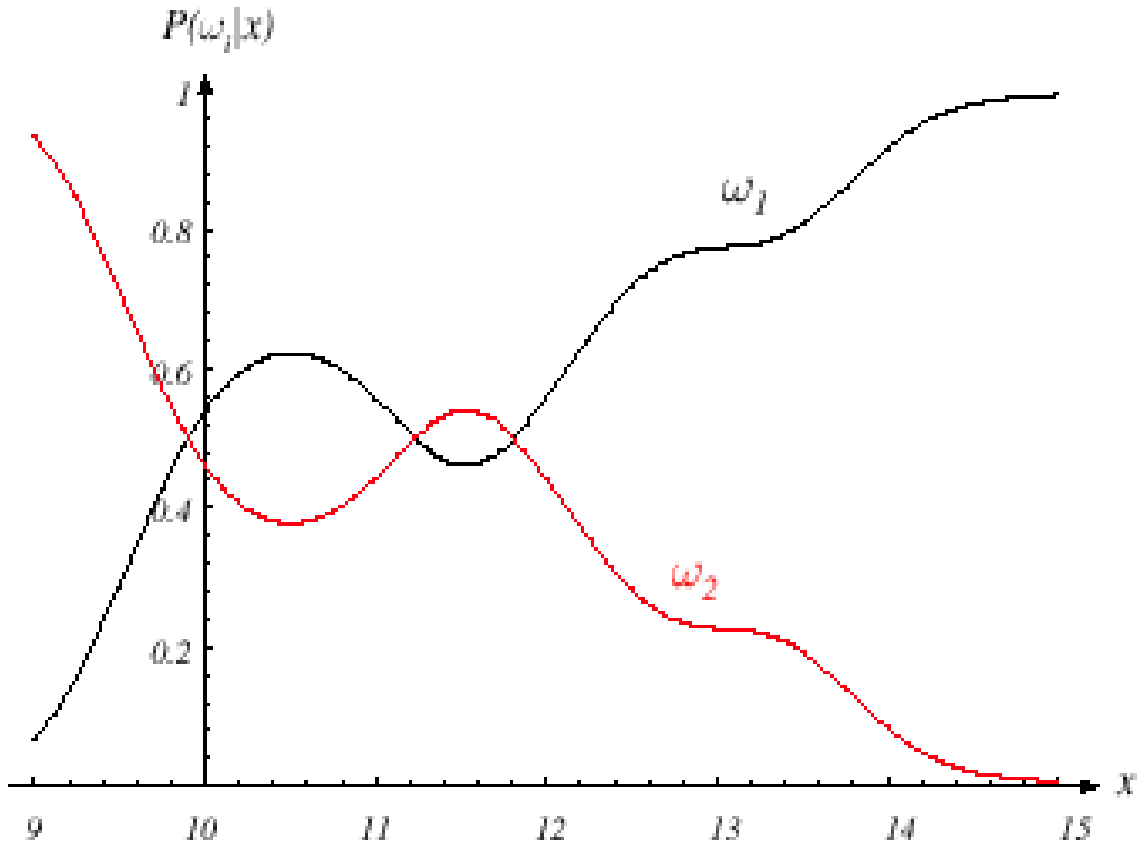


FIGURE 2.2. Posterior probabilities for the particular priors $P(\omega_1) = 2/3$ and $P(\omega_2) = 1/3$ for the class-conditional probability densities shown in Fig. 2.1. Thus in this case, given that a pattern is measured to have feature value $x = 14$, the probability it is in category ω_2 is roughly 0.08, and that it is in ω_1 is 0.92. At every x, the posteriors sum to 1.0. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

- Decision given the posterior probabilities

  X is an observation for which:

  if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ ⇒ True state of nature = ω_1
  if $P(\omega_1 \mid x) < P(\omega_2 \mid x)$ ⇒ True state of nature = ω_2

  Therefore:

  whenever we observe a particular x, the probability of error is :

  $P(error \mid x) = P(\omega_1 \mid x)$ *if we decide* ω_2
  $P(error \mid x) = P(\omega_2 \mid x)$ *if we decide* ω_1

- Minimizing the probability of error

- Decide ω_1 *if* $P(\omega_1 \mid x) > P(\omega_2 \mid x)$;
  otherwise decide ω_2

  Therefore:

$$P(\text{error} \mid x) = \min [P(\omega_1 \mid x), P(\omega_2 \mid x)]$$

(error of Bayes decision)

Average error rate

Average probability of error, P(error) is :

$$\int P(error \mid x)\, p(x) dx$$

This is the expected value of $P(error|x)$ *w.r.t.* x ,

i.e., $E_x[P(error \mid x)]$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,$ $P(white \mid \omega_2) = 0.6,\quad P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,$ $P(white \mid \omega_2) = 0.6,\quad P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,$ $P(white \mid \omega_2) = 0.6,\quad P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,\ P(white \mid \omega_2) = 0.6,\ P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.75}{0.3} = 0.5$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.25}{0.3} = 0.5$$

1. Consider a one dimensional two class problem. The feature used is color of fish. Color can be either white or dark $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25,\ P(white \mid \omega_1) = 0.2,\ P(white \mid \omega_2) = 0.6,\ P(dark \mid \omega_1) = 0.8,\ P(dark \mid \omega_2) = 0.4$ Find $P(error)$ of the Bayes Classifier.

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.75 + 0.6 * 0.25 = 0.3$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.75 + 0.4 * 0.25 = 0.7$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.75}{0.3} = 0.5$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.25}{0.3} = 0.5$$

$$P(\omega_1|dark) = \frac{P(dark|\omega_1)P(\omega_1)}{P(dark)} = \frac{0.8 * 0.75}{0.7} = \frac{6}{7}$$

$$P(\omega_2|dark) = \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)} = \frac{0.4 * 0.25}{0.7} = \frac{1}{7}$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

$$P(error) = 0.5 * 0.3 + \frac{1}{7} * 0.7 = 0.25$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

$$P(error) = 0.5 * 0.3 + \frac{1}{7} * 0.7 = 0.25$$

- But, what is the error, if we use only apriori probabilities?

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

$$P(error) = \big(P(white|\omega_2) + P(dark|\omega_2)\big)P(\omega_2)$$

$$P(error) = P(\omega_2) = 0.25$$

Since, $P(\omega_1) = 0.75,\ P(\omega_2) = 0.25$, every pattern is assigned to ω_1, So the error,

$$P(error) = P(\omega_2|white)P(white) + P(\omega_2|dark)P(dark)$$

$$P(error) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)}P(white) + \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)}P(dark)$$

$$P(error) = \big(P(white|\omega_2) + P(dark|\omega_2)\big)P(\omega_2)$$

$$P(error) = P(\omega_2) = 0.25$$

- Same error? Where is the advantage?!

Consider $P(\omega_1) = 0.5,\ P(\omega_2) = 0.5$

$$P(white) = P(white|\omega_1)P(\omega_1) + P(white|\omega_2)P(\omega_2)$$

$$P(white) = 0.2 * 0.5 + 0.6 * 0.5 = 0.4$$

$$P(dark) = P(dark|\omega_1)P(\omega_1) + P(dark|\omega_2)P(\omega_2)$$

$$P(dark) = 0.8 * 0.5 + 0.4 * 0.5 = 0.6$$

$$P(\omega_1|white) = \frac{P(white|\omega_1)P(\omega_1)}{P(white)} = \frac{0.2 * 0.5}{0.4} = 0.25$$

$$P(\omega_2|white) = \frac{P(white|\omega_2)P(\omega_2)}{P(white)} = \frac{0.6 * 0.5}{0.4} = 0.75$$

$$P(\omega_1|dark) = \frac{P(dark|\omega_1)P(\omega_1)}{P(dark)} = \frac{0.8 * 0.5}{0.6} = \frac{2}{3}$$

$$P(\omega_2|dark) = \frac{P(dark|\omega_2)P(\omega_2)}{P(dark)} = \frac{0.4 * 0.5}{0.6} = \frac{1}{3}$$

$$P(error) = P(error|white)P(white) + P(error|dark)P(dark)$$

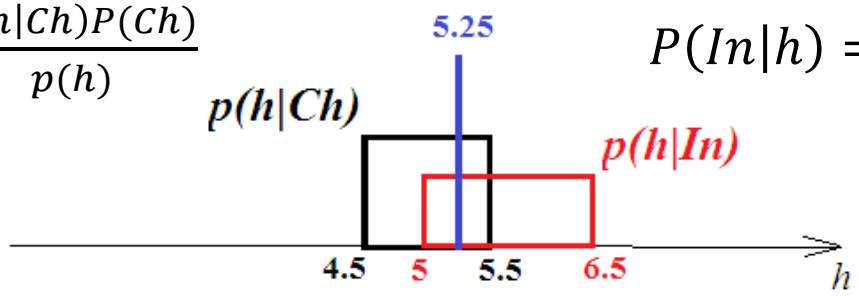
$$P(error) = 0.25 * 0.4 + \frac{1}{3} * 0.6 = 0.3$$

- But, P(error) based on apriori probabilities only is 0.5.
- Error based on the Bayes classifier is the lower bound.
  - Any classifier's error is greater than or equal to this.
- One can prove this!

2. We know that 30% of people in Sri City are from China. Remaining are Indians. We know that Chinese height is a uniformly distributed random variable with parameters 4.5 and 5.5. We also know that Indians height is also uniformly distributed with parameters 5 and 6.5. By measuring his/her height we want to classify the person as "Chinese" or "Indian" (We know that the person is living in Sri City). We follow the rule "if height is 5.25 or below classify the person as Chinese, otherwise classify the person as Indian". There are two types of mistakes in this classification, (1) Chinese being classified as Indians, (2) Indians being classified as Chinese. Find the probability of making each of these two types of mistakes.

$$P(Ch) = 0.3 \qquad P(In) = 0.7$$

$$P(Ch|h) = \frac{p(h|Ch)P(Ch)}{p(h)} \qquad P(In|h) = \frac{p(h|In)P(In)}{p(h)}$$



$$p(h) = p(h|Ch)P(Ch) + p(h|In)P(In)$$

$$= \begin{cases} 0.3, & \text{for } h \text{ in } [4.5, 5] \\ 1 * 0.3 + \frac{2}{3} * 0.7 = 0.767, & \text{for } h \text{ in } [5, 5.5] \\ \frac{2}{3} * 0.7 = 0.467, & \text{for } h \text{ in } [5.5, 6.5] \end{cases}$$

For $h \in [4.5,\ 5]$

$P(Ch \mid h) = (1 * 0.3)/0.3 = 1$

$P(In|h) = 0$

For $h \in [5,\ 5.5]$

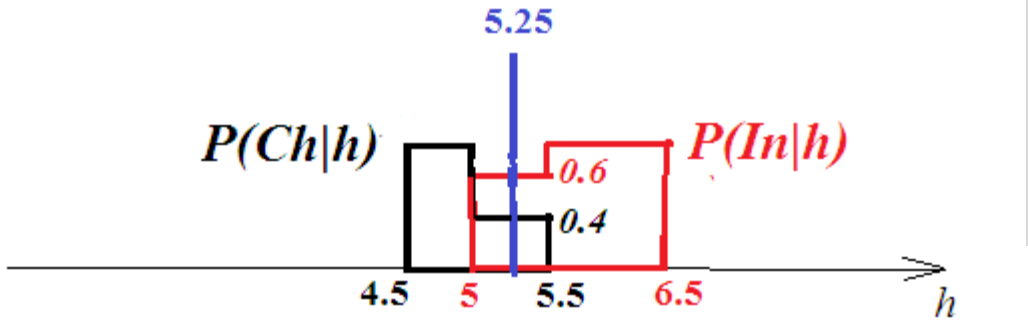
$P(Ch \mid h) = (1 * 0.3)/0.767 = 0.39$

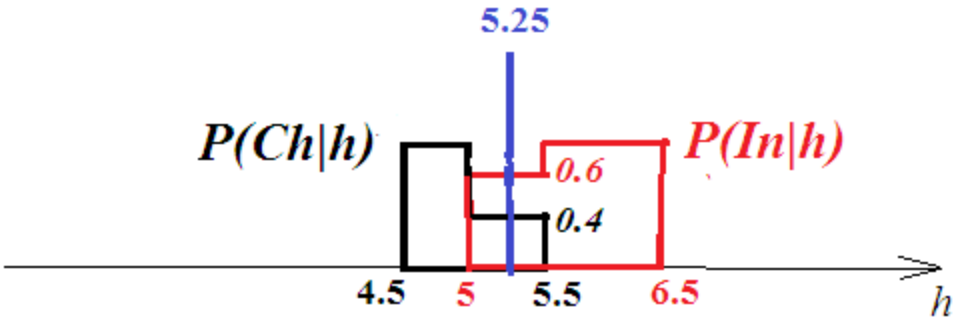
$P(In \mid h) = 0.67 * 0.7/0.767 = 0.61$

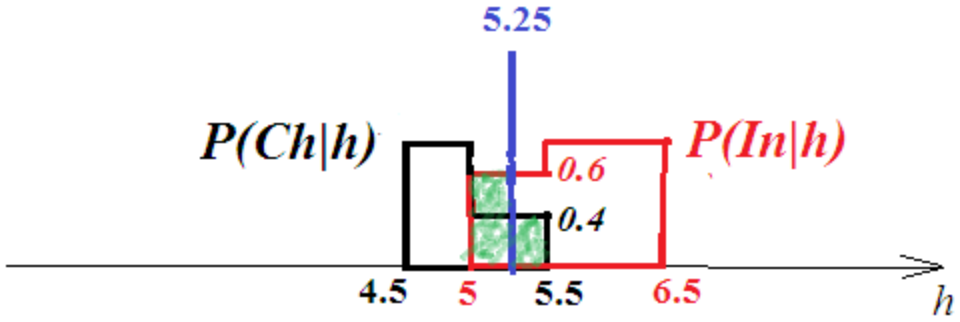
For $h \in [5,\ 6.5]$

$P(Ch \mid h) = 0$

$P(In|h) = 1$







Error is due to this region

$$P(\text{error}) = \int_{5}^{5.25} 0.609\, p(h)dh + \int_{5.25}^{5.5} 0.391\, p(h)dh$$

$$= 0.117 + 0.075 = 0.192$$

- But, this is not the Bayes classifier.
- What does the Bayes classifier do?
- What is the error of the Bayes classifier?

Bayesian Decision Theory – Continuous Features

- Generalization of the preceding ideas
  - Use of more than one feature
  - Use more than two states (classes) of nature
  - Allowing actions (decisions) other than just classification.
  - Introduce a *loss function* which is more general than the probability of error.

- Allowing actions other than classification primarily allows the possibility of rejection
- Refusing to make a decision in close or bad cases!
- The loss function states how costly each action taken is

Problem setting

What is given to us:

Loss function : $\lambda(\alpha_i \mid \omega_j)$ is the loss of taking action α_i when the state of nature is ω_j

Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature

(or “categories” or “classes”)

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

For example : α_1 is the action *ring the alarm*

α_2 is the action *shutdown the system*

Let $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature

(or "categories" or "classes")

Let $A = \{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

For example : α_1 is the action *ring the alarm*

α_2 is the action *shutdown the system*

Objective: Given a pattern x, find the action to take.

That is, to find a function $\alpha(x)$ which maps x to action.

How to find the best action?

Let $R(\alpha_i \mid x)$ is the risk of taking action α_i when the given pattern is x (this is called conditional risk).

We can take the action α_k provided $R(\alpha_k \mid x)$ is minimum in $\{ R(\alpha_1 \mid x), R(\alpha_2 \mid x), \ldots, R(\alpha_a \mid x) \}$

How to relate $R(\alpha_i \mid x)$ with $\lambda(\alpha_i \mid \omega_j)$ values.

$$R(\alpha_i \mid x) = \sum_{j=1}^{j=c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x)$$

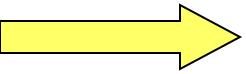
α(x) is given, how good is this?

- Overall risk of this rule can be found

$$R = \int R(\alpha(\mathbf{x})|\mathbf{x})p(\mathbf{x})\ d\mathbf{x},$$

- The smaller this quantity, better the decision rule.

Select the action α_i for which $R(\alpha_i \mid x)$ is minimum



R is minimum and R in this case is called the

Bayes risk = best performance that can be achieved!

- Two-category classification

α_1 : deciding ω_1

α_2 : deciding ω_2

$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$

loss incurred for deciding ω_i when the true state of nature is ω_j

Conditional risk:

$$R(\alpha_1 \mid x) = \lambda_{11}P(\omega_1 \mid x) + \lambda_{12}P(\omega_2 \mid x)$$
$$R(\alpha_2 \mid x) = \lambda_{21}P(\omega_1 \mid x) + \lambda_{22}P(\omega_2 \mid x)$$

Our rule is the following:

if $R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$

action α_1: "decide ω_1" is taken

This results in the equivalent rule :

decide ω_1 if:

$$(\lambda_{21} - \lambda_{11})\, p(x \mid \omega_1)\, P(\omega_1) > (\lambda_{12} - \lambda_{22})\, p(x \mid \omega_2)\, P(\omega_2)$$

and decide ω_2 otherwise

An Example

- Let the two actions are:

α_1 ⟹ ***x is criminal***

α_2 ⟹ ***x is innocent***

Let $\lambda(\alpha_1 \mid \omega_1) = 0;$ $\lambda(\alpha_1 \mid \omega_2) = 10;$

$\lambda(\alpha_2 \mid \omega_1) = 1;$ $\lambda(\alpha_2 \mid \omega_2) = 0;$

Assume equal priors, and

Let $p(x \mid \omega_1) = 0.8,$ $p(x \mid \omega_2) = 0.6$

What action you will take?

ω_1 is class of criminals,

ω_2 is class of innocents.

Example: Contd…

- $P(\omega_j \mid x) = \dfrac{p(x \mid \omega_j) \,.\, P(\omega_j)}{p(x)}$

- $P(\omega_1 \mid x) = \dfrac{0.8(0.5)}{0.8(0.5) + 0.6(0.5)} = 8/14 = 0.57$

- $P(\omega_2 \mid x) = 0.43$

- *As per plain Bayes rule we declare the person to be a criminal.*

Example: Contd …

- $R(\alpha_1 \mid x) = \lambda(\alpha_1 \mid \omega_1) P(\omega_1 \mid x) + \lambda(\alpha_1 \mid \omega_2) P(\omega_2 \mid x)$
  $= 0 (0.57) + 10 (0.43)$
  $= 4.3$

- $R(\alpha_2 \mid x) = \lambda(\alpha_2 \mid \omega_1) P(\omega_1 \mid x) + \lambda(\alpha_2 \mid \omega_2) P(\omega_2 \mid x)$
  $= 1 (0.57) + 0 (0.43)$
  $= 0.57$

Action taken: x is innocent

Likelihood ratio:

The preceding rule is equivalent to the following rule:

$$if\ \frac{p(x|\omega_1)}{p(x|\omega_2)} > \frac{\lambda_{12}-\lambda_{22}}{\lambda_{21}-\lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Then take action α_1 (decide ω_1)

Otherwise take action α_2 (decide ω_2)

Example 1

- Let $\Omega = \{\omega_1, \omega_2, \omega_3\}$, $A = \{\alpha_1, \alpha_2, \alpha_3\}$

  $P(\omega_1 | x) = \mathbf{0.1}$, $P(\omega_2 | x) = \mathbf{0.4}$, $P(\omega_3 | x) = \mathbf{0.5}$

  Loss function is

Loss	ω_1	ω_2	ω_3
α_1	0	1	2
α_2	1	0	2
α_3	3	10	0

Find $R(\alpha_k | x)$ for k = 1, 2, 3.

Find what is the best action?

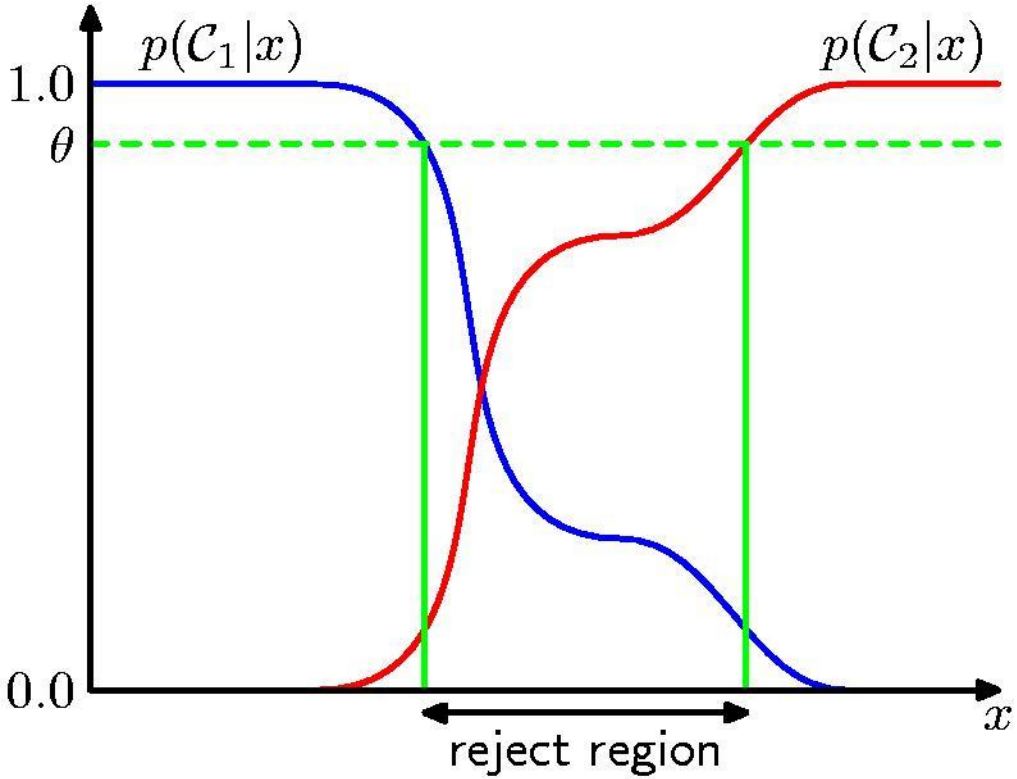
Reject option (I do not want to classify)

Actions are : Assign a class label or reject

Sometimes, when misclassification is costly, we can reject to classify it.

May be some other expert can look into it and can take appropriate action.

Reject Option



- Read Duda and Hart book and try to solve some problems related to this.

Naïve Bayes Classification

Introduction

- The Bayes Classifier requires probability structure of the problem to be known.
- Density estimation (using non-parametric or parametric methods) is one way to handle the problem.
- There are several problems ….

Problems with density estimation

- Large datasets are needed.
- Numeric valued features are required.

- In practice these two may not be satisfied.

How to overcome the problem

- One has to work with the given data set.
- So, Probability estimations needs to be done using the given data only.
- Often marginal probabilities can be better estimated than the joint probabilities.
- Also, marginal probabilities are easy to compute.

Play-tennis data

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

- P(<sunny,cool,high,false>|N) = 0
- But, P(sunny|N) = 3/5, P(cool|N) = 1/5,
  P(high|N) = 4/5, P(false|N) = 2/5.

- P(<sunny, cool, high, false>|N) = 0
- This may be because of the smaller dataset.
- If we increase the dataset size, this may become a positive number.

- This problem is often referred to as “the curse of dimensionality”.

Assumption

- Make the assumption that for a given class, features are independent of each other.
- In practice, this assumption holds very often.

- Then P(<sunny,cool,high,false>|N) = P(sunny|N) . P(cool|N) . P(high|N) . P(false|N) = 3/5 . 1/5 . 4/5 . 2/5 = 24/625.

Naïve Bayesian Classification

- Naïve assumption: for a given class, features are independent of each other

$$P(<x_1,\ldots,x_k>|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$$

- $P(x_i|C)$ is estimated as the relative freq of samples having value x_i as i-th attribute in class C
- It often makes the problem a feasible and easy one to solve.

Play-tennis example: estimating $P(x_i|C)$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14
P(n) = 5/14

outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample X = <rain, hot, high, false>

- P(X|p)·P(p) =

  P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) = 3/9·2/9·3/9·6/9·9/14 = 0.010582

- P(X|n)·P(n) =

  P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) = 2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

With Continuous features

- In order to use the Naïve Bayes classifier, the features has to be discretized appropriately (otherwise what happens?)

With Continuous features

- In order to use the Naïve Bayes classifier, the features has to be discretized appropriately (otherwise what happens?)
- Height = 4.234 will not occur anywhere in that column; but 4.213, 4.285 may be occuring. If you discretize (eg., rounding) then fequency ratio's are meaningful.
- Clustering of feature values of a feature may be done to achieve a better discretization.