Machine Learning
ZG565

Dr. Sugata Ghosal
Sugata.ghosal@pilani.bits-pilani.ac.in

BITS Pilani
Pilani Campus

innovate achieve lead

BITS Pilani
Pilani Campus

Session 1
Date – 20th May 2023
Time – 2 PM to 4 PM

These slides are prepared by the instructor, with grateful acknowledgement of many others who made their course materials freely available online.

Session Content

- Objective of course
- Evaluation Plan
- What is Machine Learning?
- Application areas of Machine Learning
- Why Machine Learning is important?
- Design a Learning System
- Issues in Machine Learning

Objective of course

- Introduction to the basic concepts and techniques of Machine Learning
- Gain experience of doing independent study and research in the field of Machine Learning
- Develop skills of using recent machine learning software tools to evaluate learning algorithms and model selection for solving practical problems

What We'll Cover in this Course

- **Supervised learning algorithms**
  - Regression
  - Naïve Bayes
  - Logistic regression
  - Decision Tree and Random Forest
  - Support vector machines
- **Ensemble Techniques**
- **Unsupervised learning**
  - Clustering
- **Applications**

Books

Text books and Reference book(s)

T1	Tom M. Mitchell: Machine Learning, The McGraw-Hill Companies

R1	Christopher M. Bishop: Pattern Recognition & Machine Learning, Springer
R2	**P. Tan, et al.** Introduction to Data Mining**, Pearson**
R3	**C.J.C. BURGES: A Tutorial on Support Vector Machines for Pattern Recognition, Kluwer Academic Publishers, Boston.**

Evaluation Plan

Name	Type	Weight
3 Quiz, best 2 scores will be taken	Online	10%
Assignment-I	Take Home	10%
Assignment-II	Take Home	10%
Mid-Semester Test	Closed Book	30%
Comprehensive Exam	Open Book	40%

Please note there will be no change in submission dates for quiz and assignment

Lab Plan

Lab No.	Lab Objective
1	End to End Machine Learning
2	Linear Regression and Gradient Descent Algorithm
3	Logistic Regression Classifier
4	Decision Tree
5	Naïve Bayes Classifier
6	Random Forest

- **Labs not graded**
- **Most of the Lab recordings available at CSIS virtual labs**
- **Webinars will be conducted for lab sessions**
- **Labs will be conducted in Python**

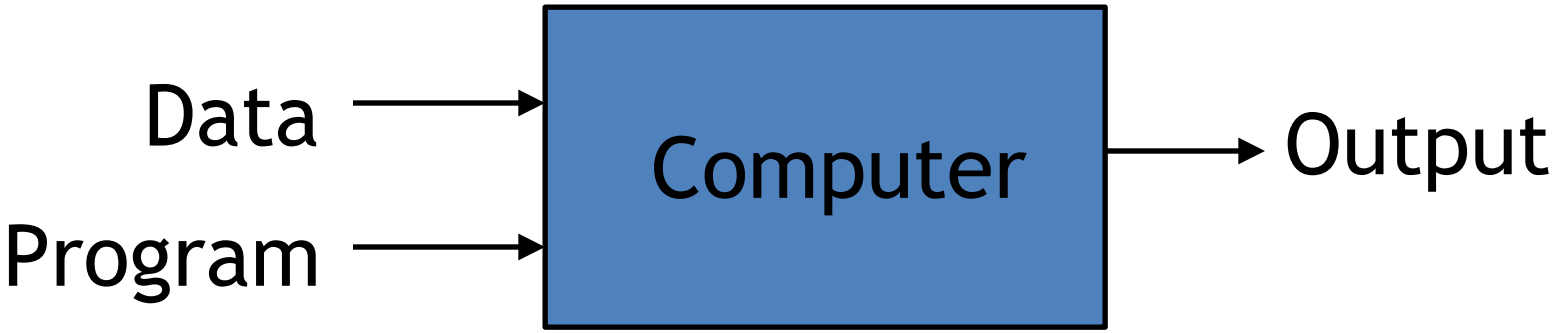
Machine Learning

- **Machine learning** is a scientific discipline that explores the construction and study of algorithms that can learn from data.
- Such algorithms operate by building a model based on inputs and using that to make predictions or decisions, rather than following only explicitly programmed instructions.

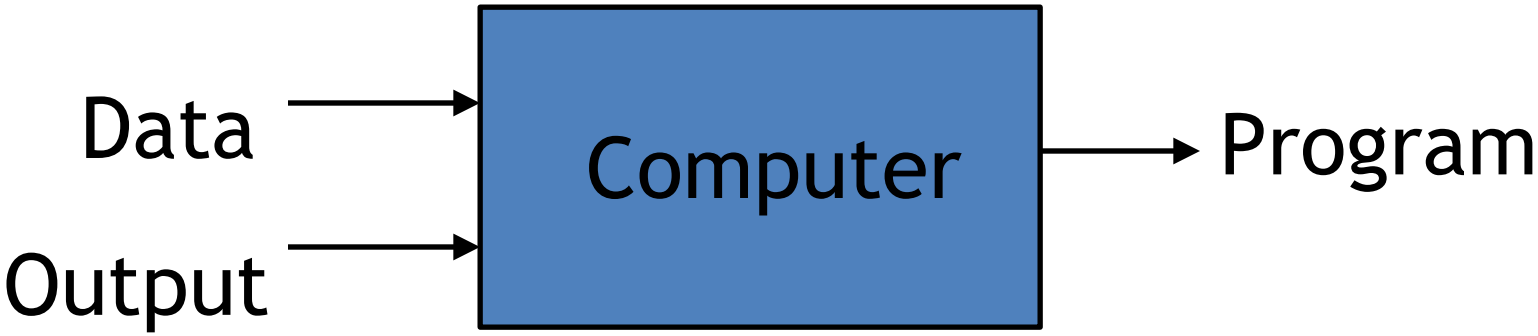
A Few Quotes

- “A breakthrough in machine learning would be worth ten Microsofts” (Bill Gates, Chairman, Microsoft)
- “Machine learning is the next Internet” (Tony Tether, Director, DARPA)
- “Web rankings today are mostly a matter of machine learning” (Prabhakar Raghavan, Dir. Research, Yahoo)
- “Machine learning is going to result in a real revolution” (Greg Papadopoulos, CTO, Sun)
- “Machine learning is today’s discontinuity” (Jerry Yang, CEO, Yahoo)

Traditional Programming



Machine Learning



Slide credit: Pedro Domingos

What is Machine Learning?

Definition by Tom Mitchell (1998):

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.” Example: playing checkers.

E = the experience of playing many games of checkers

T = the task of playing checkers.

P = the probability that the program will win the next game.

What is Machine Learning?

- To have a learning problem, we must identify
  - The class of tasks
  - The measure of performance to be improved
  - Source of experience

Example of Learning Problems

A Checker Learning Problem

- **Task T:** Playing Checkers
- **Performance Measure P:** Percent of games won against opponents
- **Training Experience E:** To be selected ==> Games Played against itself

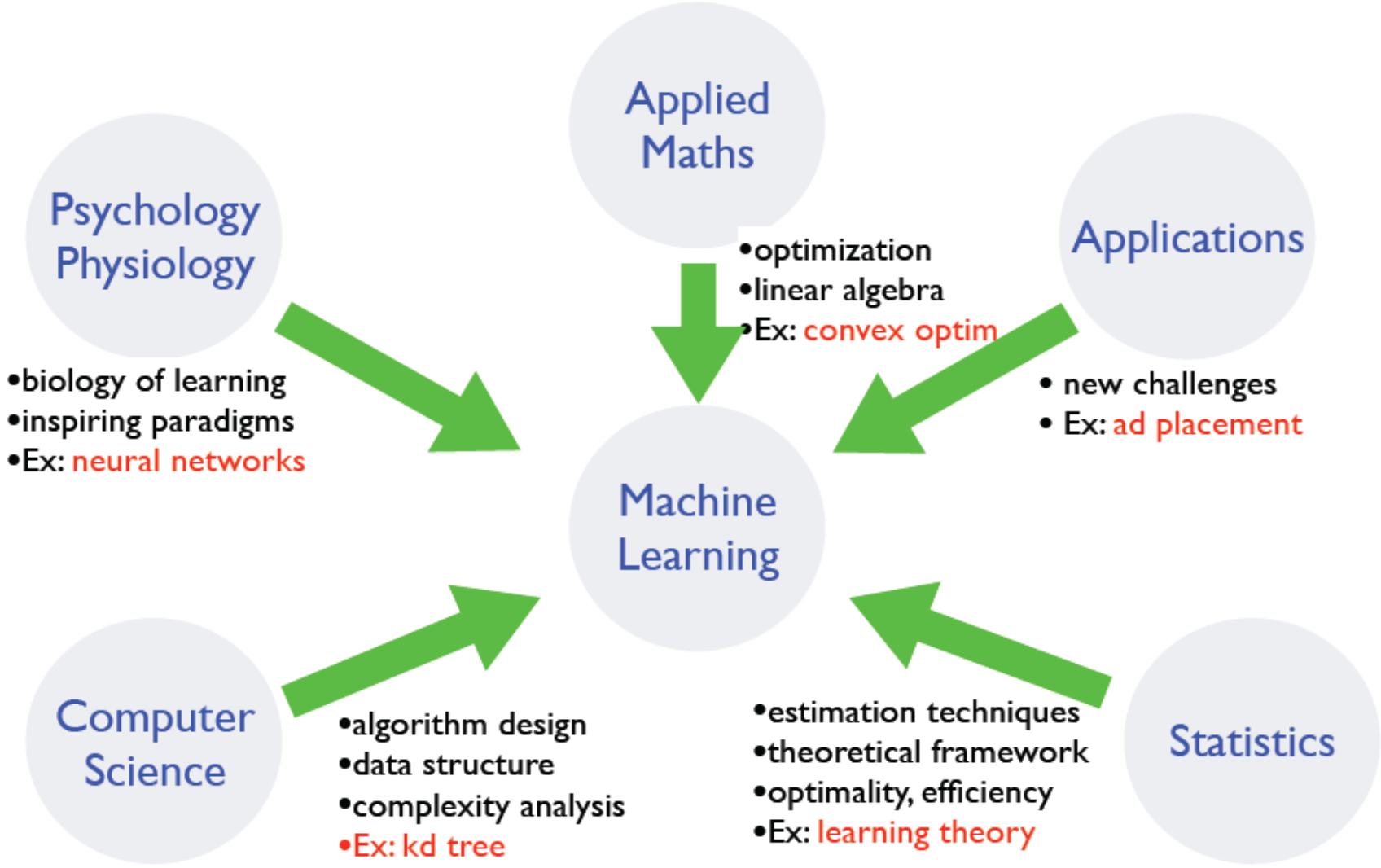
A handwriting recognition learning problem

- **Task T:** recognizing and classifying handwritten words within images
- **Performance measure P**: percent of words correctly classified
- **Training Experience E:** a database of handwritten words with given classifications

A robot driving learning problem

- **Task T:** driving on public four-lane highways using vision sensors
- **Performance measure P:** average distance travelled before an error (as judged by human)
- **Training experience E:** a sequence of images and steering commands recorded while observing a human driver

Where does ML fit in?



Slide credit: Dhruv Batra, Fei Sha

Why is Machine Learning Important?

- Some tasks cannot be defined well, except by examples.
- Relationships and correlations can be hidden within large amounts of data. Machine Learning may be able to find these relationships.
- Human designers often produce machines that do not work as well as desired in the environments in which they are used.

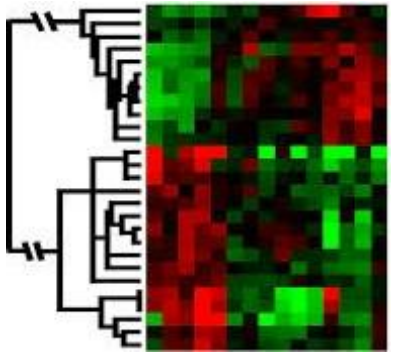
Why is Machine Learning Important ?

- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans (e.g., medical diagnostic).
- New knowledge about tasks is constantly being discovered by humans. It may be difficult to continuously re-design systems “by hand”.

When Do We Use Machine Learning?

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 Credit: Eric Eaton

Application Domains

- Web search
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Language Processing

Many more emerging…

Slide credit: Pedro Domingos

State of the Art Applications of Machine Learning

.

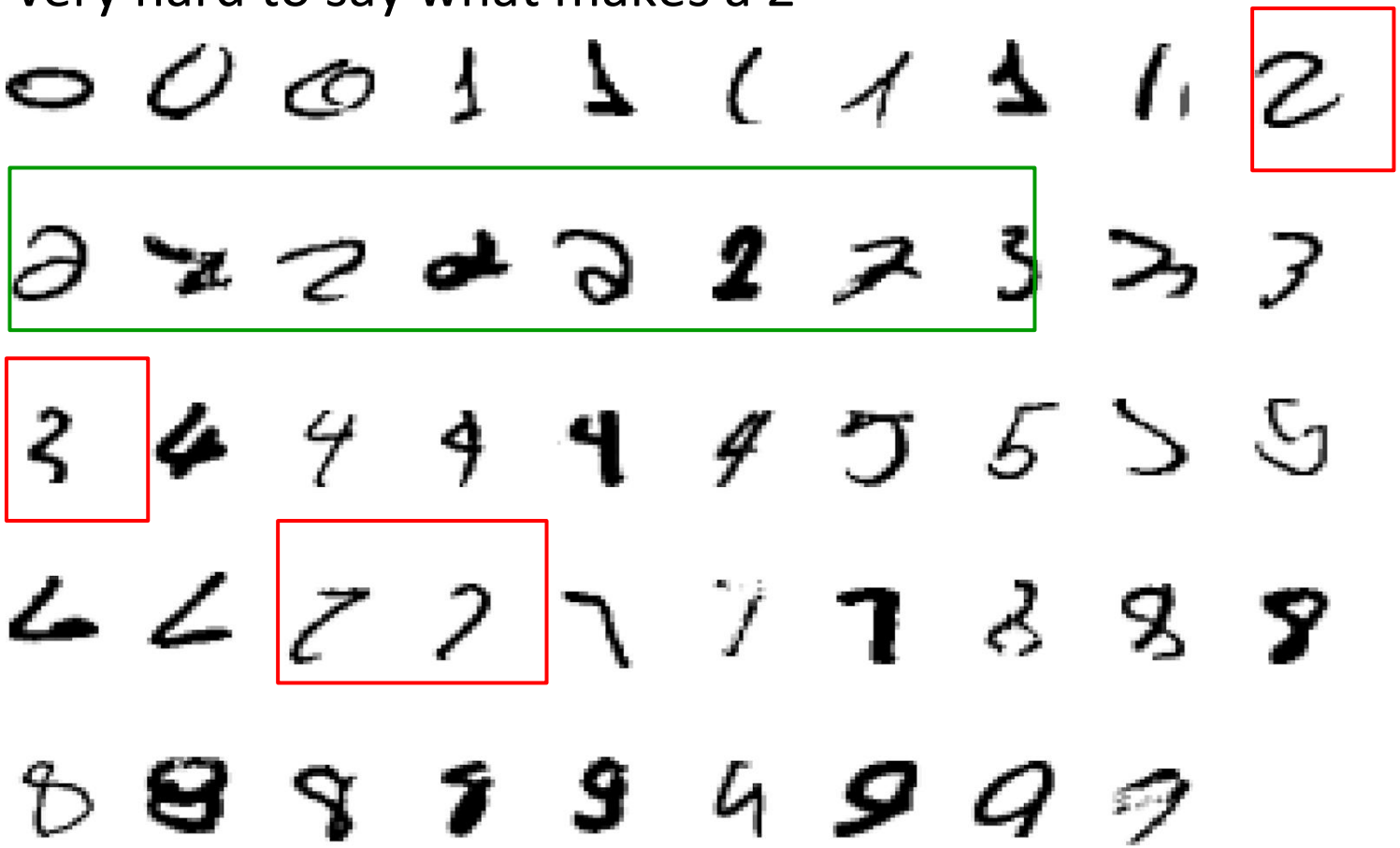
Application Types

- Medical diagnosis
- Credit card applications or transactions
- Fraud detection in e-commerce
- Worm detection in network packets
- Spam filtering in email
- Recommended articles in a newspaper
- Recommended books, movies, music, or jokes
- Financial investments
- DNA sequences
- Spoken words
- Handwritten letters
- Astronomical images

Slide credit: Ray Mooney

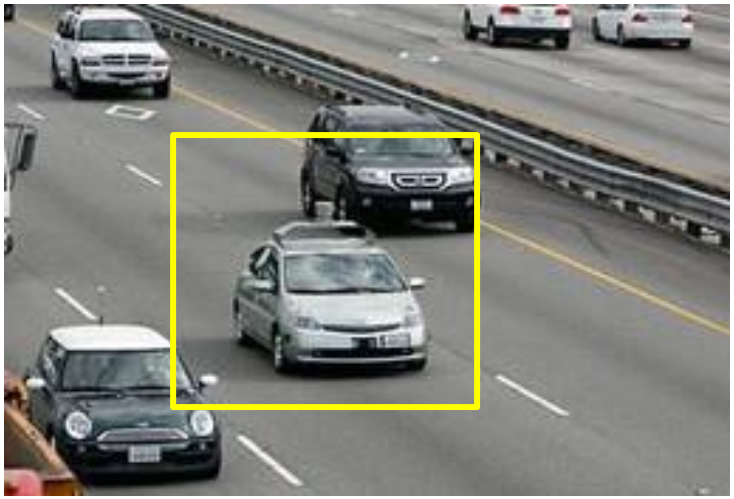
Pattern recognition

It is very hard to say what makes a 2



Slide credit: Geoffrey Hinton

Autonomous Cars

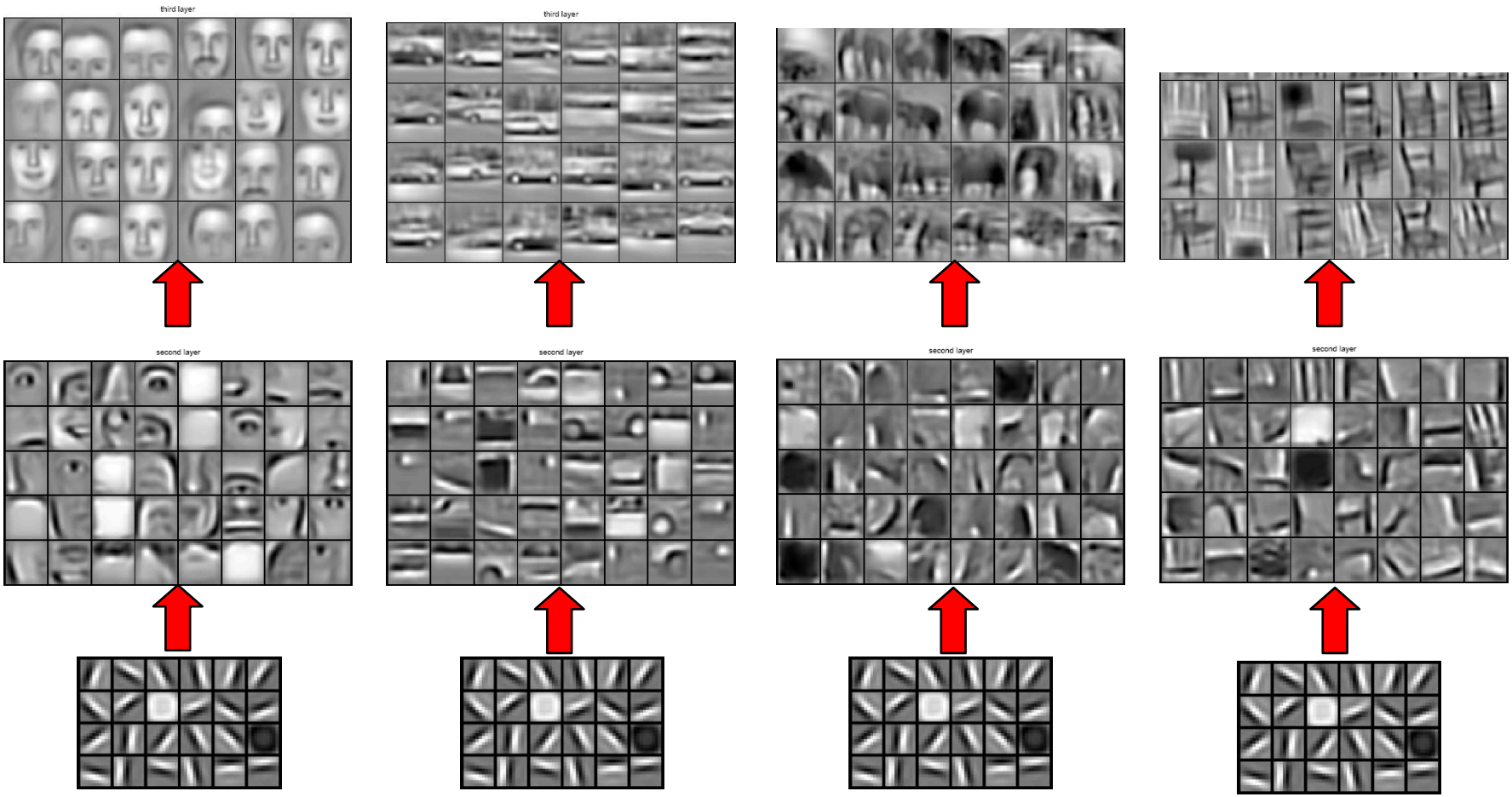


- Nevada made it legal for autonomous cars to drive on roads in June 2011
- As of 2013, four states (Nevada, Florida, California, and Michigan) have legalized autonomous cars

UPenn's Autonomous Car →

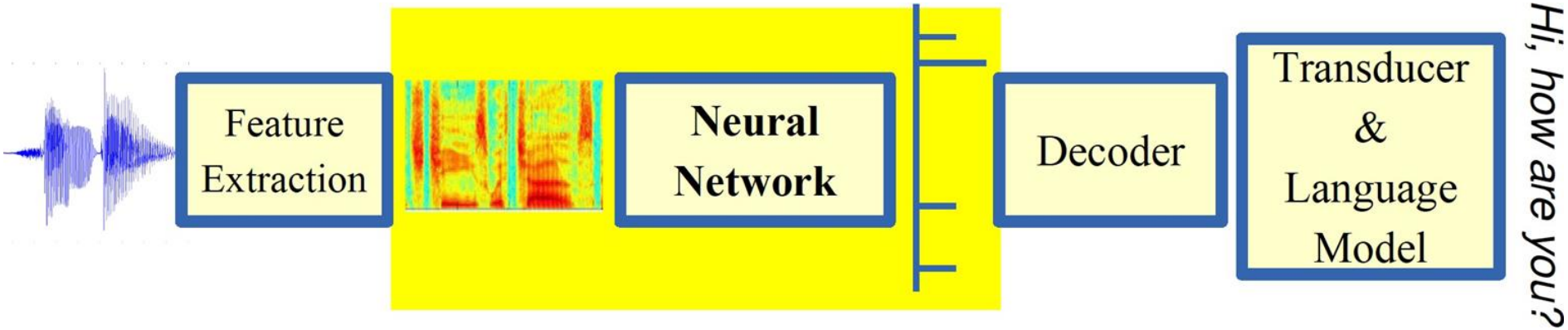
Slide credit: Eric Eaton

Learning of Object Parts

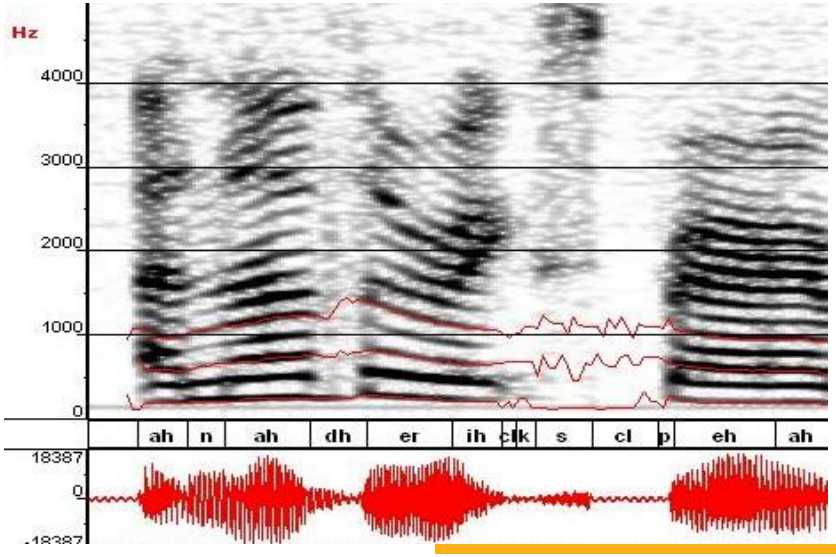


Slide credit: Andrew Ng

Automatic Speech Recognition



ML used to predict phoneme states from sound spectrogram Deep Learning Based Results



# Hidden Layers	1	2	4	8	10	12
Word Error Rate %	16.0	12.8	11.4	10.9	11.0	11.1

Baseline Gaussian Mixture Model based word error rate = 15.4%

[Zeiler et al. "On rectified linear units for speech recognition" ICASSP 2013]

Slide credit: Eric Eaton

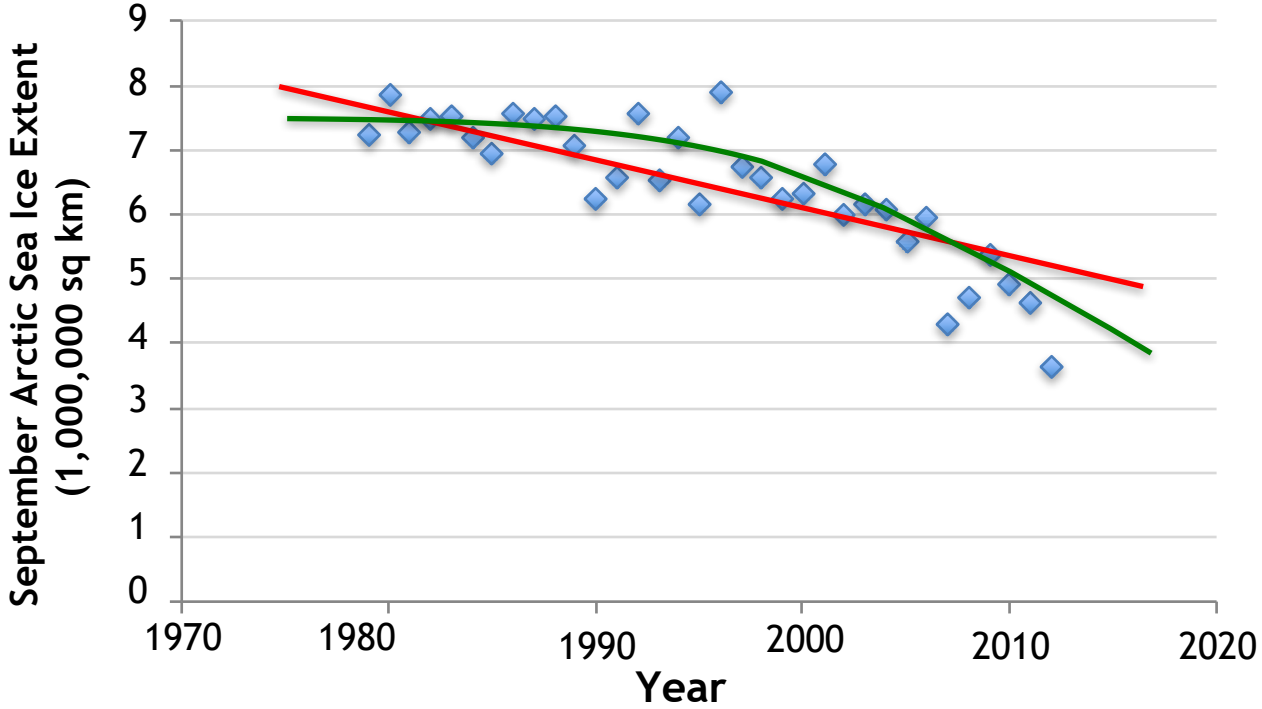
Robotics

Types of Learning

- **Supervised (inductive) learning**
  - Given: training data, desired outputs (labels)
- **Unsupervised learning**
  - Given: training data (without desired outputs)
- **Semi-supervised learning**
  - Given: training data + a few desired outputs
- **Reinforcement learning**
  - Given: rewards from sequence of actions

Supervised Learning: Regression

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



Data from G. Witt. Journal of Statistics Education, Volume 21, Number 1 (2013) Slide Credit: Eric Eaton

Regression

$$y = f(\mathbf{x})$$

output — prediction function — features

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Classification Example

- Apply a prediction function to a feature representation of the image to get the desired output:

f( ) = “apple”

f( ) = “tomato”

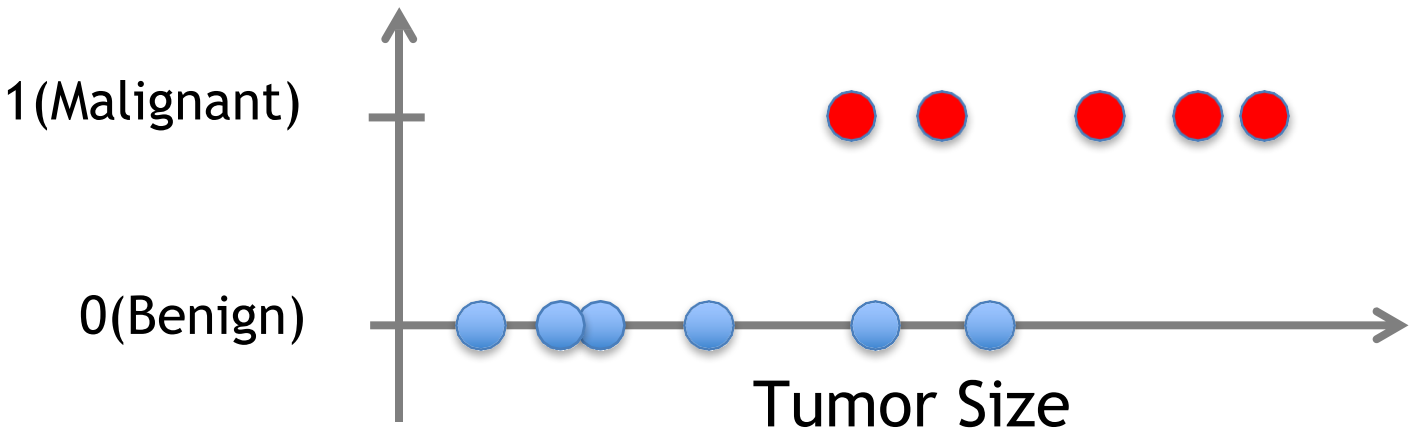
f( ) = “cow”

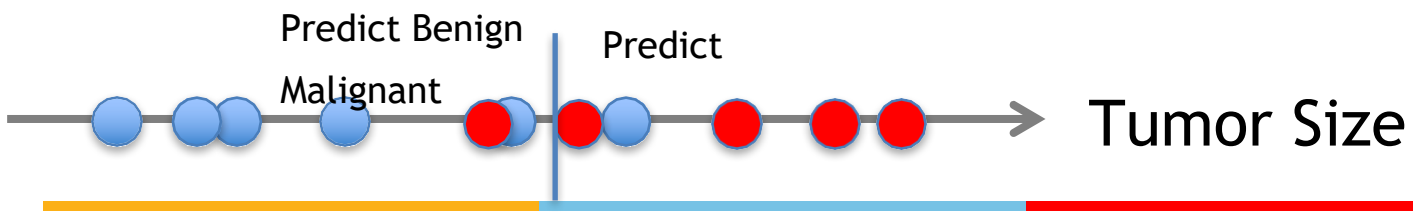
Slide credit: L. Lazebnik

Supervised Learning: Classification

- 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 categorical == classification

Breast Cancer (Malignant / Benign)

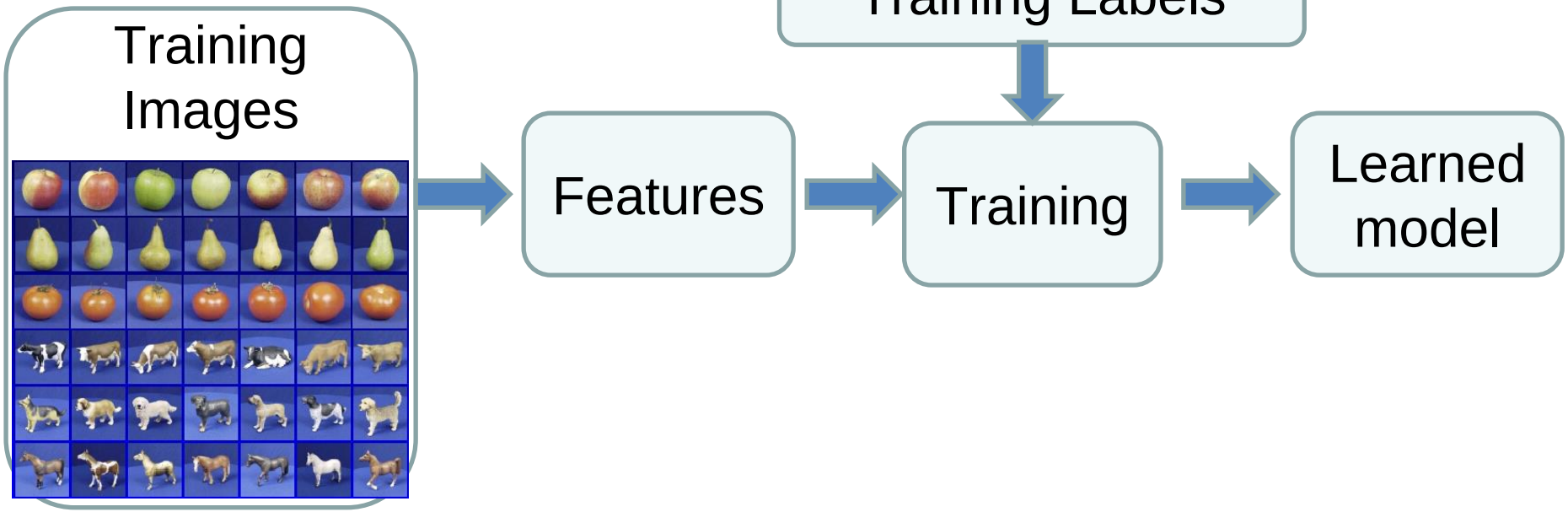




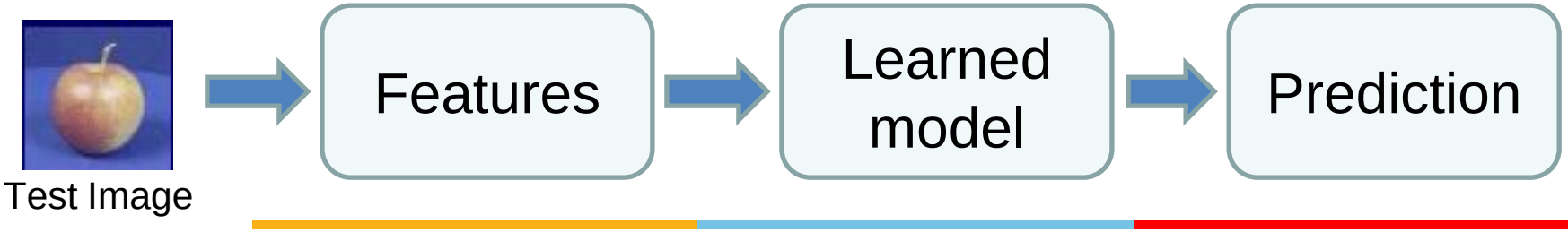
Slide Credit: Eric Eaton

Classification

Training



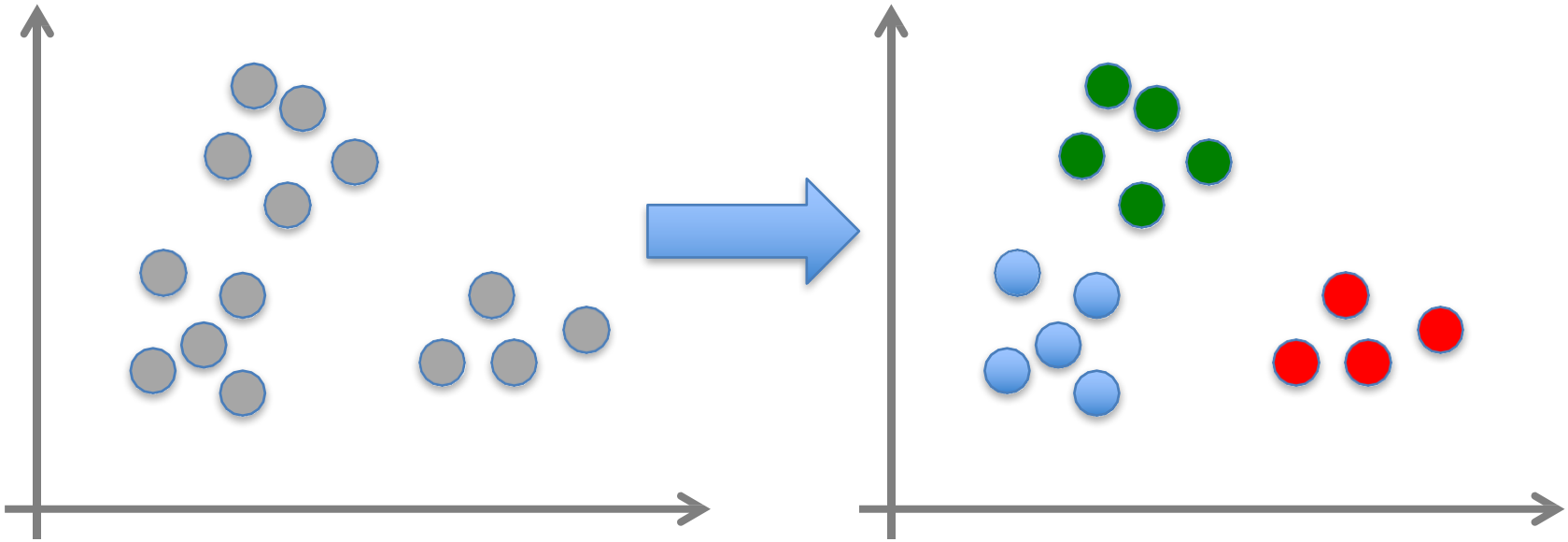
Testing



Slide credit: D. Hoiem and L. Lazebnik

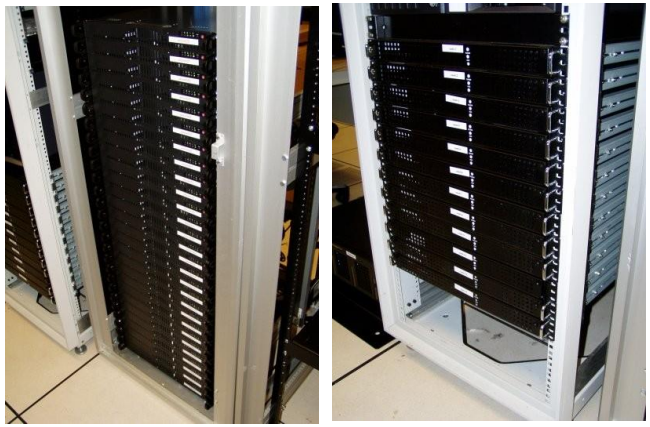
Unsupervised Learning

- Given $x_1, x_2, \ldots, x_n$ (without labels)
- Output hidden structure behind the x's
  - E.g., clustering

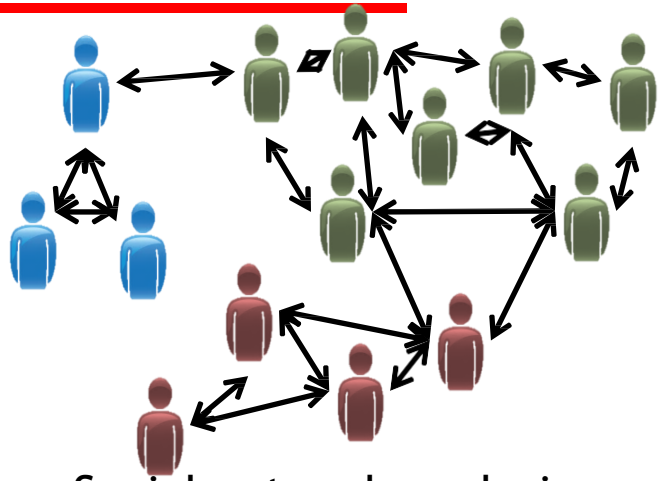


Slide Credit: Eric Eaton

Unsupervised Learning



Organize computing clusters



Social network analysis

Market segmentation

Astronomical data analysis

Slide credit: Andrew Ng

Reinforcement Learning

- No predefined data
- Semi-supervised learning model in machine learning,
- Allow an agent to take actions and interact with an environment so as to maximize the total rewards.
- Examples:
  - Resources management in computer clusters
  - Game playing
  - Robot in a maze

Slide Credit: Eric Eaton

Supervised, Unsupervised and Reinforcement Learning Comparison

Criteria	Supervised Learning	Unsupervised Learning	Reinforcement Learning
Definition	The machine learns by using labeled data	The machine is trained on unlabeled data without any guidance	An agent interacts with its environment by performing actions & learning from errors or rewards
Type of problems	Regression & classification	Association & clustering	Reward-based
Type of data	Labeled data	Unlabeled data	No predefined data
Training	External supervision	No supervision	No supervision
Approach	Maps the labeled inputs to the known outputs	Understands patterns & discovers the output	Follows the trial-and-error method

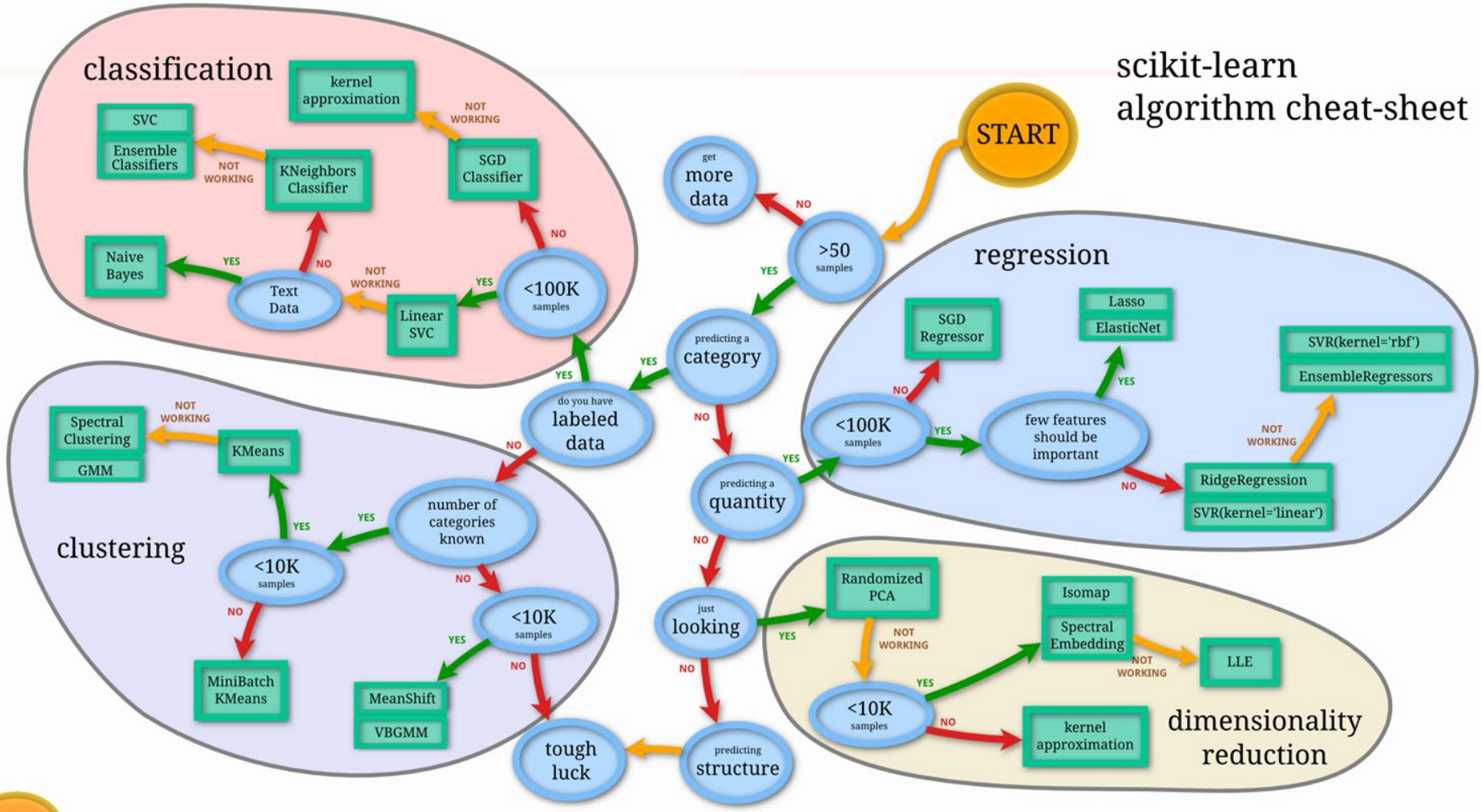
Open source ML programming tools

	Platform		Algorithms or Features
Scikit Learn	Linux, Mac OS, Windows	Python, C, C++	Classification, Regression, Clustering Preprocessing, Model Selection Dimensionality reduction.
PyTorch	Linux, Mac OS, Windows	Python, C++	Autograd Module, Optimization Module NN Module
TensorFlow	Linux, Mac OS, Windows	Python, C++	Provides a library for dataflow programming.
Weka	Linux, Mac OS, Windows	Java	Data preparation, Classification Regression, Clustering, Visualization Association rules mining

Open source ML programming tools

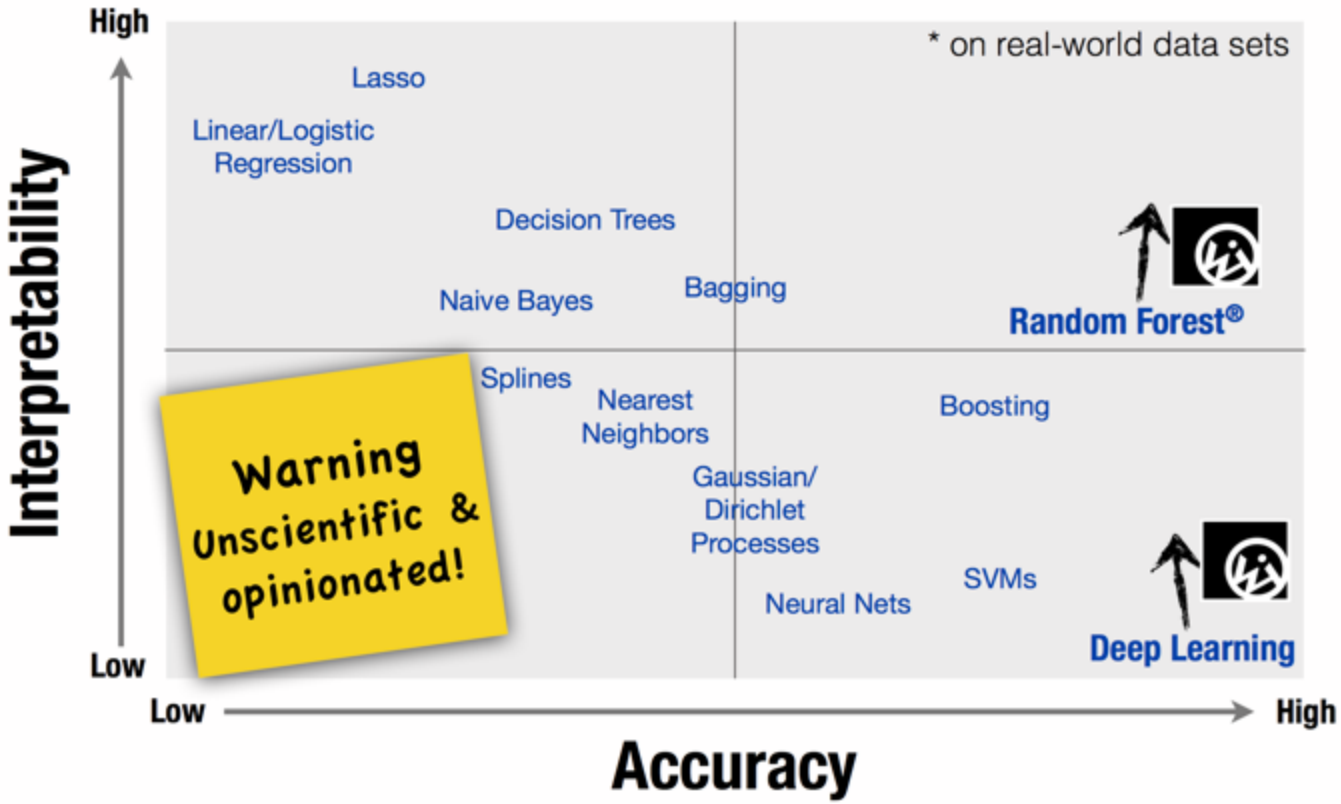
Colab	Cloud Service	-	Supports libraries of PyTorch, Keras, TensorFlow, and OpenCV
Apache Mahout	Cross-platform	Java Scala	Preprocessors,Regression Clustering, Recommenders Distributed Linear Algebra.
Accors.Net	Cross-platform	C#	Classification, Regression, Distribution Clustering, Hypothesis Tests & Kernel Methods, Image, Audio & Signal & Vision
Shogun	Windows Linux, UNIX Mac OS	C++	Regression, Classification, Clustering Support vector machines. Dimensionality reduction, Online learning etc.
Keras.io	Cross-platform	Python	API for neural networks

scikit-learn algorithm cheat-sheet



Back

ML Algorithmic Trade-Off



ML in a Nutshell

- Tens of thousands of machine learning algorithms
  - Hundreds new every year

- Every ML algorithm has three components
  - **Representation**
  - **Optimization**
  - **Evaluation**

Slide credit: Pedro Domingos

Evaluation

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- etc.

Slide credit: Pedro Domingos

Evaluating Performance

- If *y* is discrete:
  - Accuracy: # correctly classified / # all test examples
  - Precision/recall
    - True Positive, False Positive, True Negative, False Negative
    - **P**recision = TP / (TP + FP) = # predicted true pos / # predicted pos
    - **R**ecall = TP / (TP + FN) = # predicted true pos / # true pos
  - F-measure

    = 2PR / (P + R)

- Want evaluation metric to be in some range, e.g. [0 1]
  - 0 = worst possible classifier, 1 = best possible classifier

Evaluating Performance

- If *y* is continuous:
  - Sum-of-Squared-Differences (SSD) error between predicted and true *y*:

$$\mathbf{E} = \sum_{\mathbf{i}=1}^{\mathbf{n}} (\mathbf{f}(\mathbf{x_i}) - \mathbf{y_i})^2$$

Issues in Machine Learning

- What algorithms are available for learning a concept? How well do they perform?
- How much training data is sufficient to learn a concept with high confidence?
- When is it useful to use prior knowledge?
- Are some training examples more useful than others?
- What are the best tasks for a system to learn?
- What is the best way for a system to represent its knowledge?

Training vs Testing

- What do we want?
  - High accuracy on training data?
  - No, high accuracy on *unseen/new/test data!*
  - Why is this tricky?
- Training data
  - Features (x) and labels (y) used to learn mapping f
- Test data
  - Features used to make a prediction
  - Labels only used to see how well we've learned f!!!
- Validation data
  - Held-out set of the *training data*
  - Can use both features and labels to tune *parameters* of the model we're learning

Training vs. Test Distribution

- We generally assume that training and test examples are independently drawn from the same overall distribution of data
  - We call this “i.i.d” which stands for “independent and identically distributed”

Slide credit: Ray Mooney

ML in Practice

Loop

- Understand domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learn models
- Interpret results
- Consolidate and deploy discovered knowledge

Based on a slide by Pedro Domingos

Recommended Readings

- Tom M. Mitchell, Machine Learning, The McGraw-Hill Companies, Inc. International Edition 1997. **[Ch. 1]**
- **http://www.cs.princeton.edu/courses/archive/spr08/cos511/ [Web]**
- https://www.softwaretestinghelp.com/machine-learning-tools/

END
of
Session 1

Machine Learning
ZG565

Dr. Sugata Ghosal
Sugata.ghosal@pilani.bits-pilani.ac.in

BITS Pilani
Pilani Campus

innovate achieve lead

BITS Pilani
Pilani Campus

Session 2
Date – 27th May 2023
Time – 2 PM to 4 PM

These slides are prepared by the instructor, with grateful acknowledgement of many others who made their course materials freely available online.

Session Content

- Data - Types, Attributes
- Data Quality
- Data Preprocessing
- Performance Metrics
- Challenges of ML
- Model Evaluation, Selection

- Homework
  - Review Lab Capsule 1 from the “Machine Learning” Virtual Labs (accessible via elearn portal)

ML in a Nutshell

- Tens of thousands of machine learning algorithms
  - Hundreds new every year

- Every ML algorithm has three components
  - **Data Representation**
  - **Parameter Optimization**
  - **Model Evaluation, Selection**

Slide credit: Pedro Domingos

ML in Practice

Loop

- Understand domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc.
- Learn optimal parameter of the models
- Interpret results
- Consolidate and deploy discovered knowledge

Based on a slide by Pedro Domingos

Definition of Data

- Collection of ***data objects*** and their ***attributes***
- An ***attribute*** is a property or characteristic of an object
  - Examples: eye color of a person, temperature, etc.
  - *aka* variable, field, characteristic, dimension, or feature
- A collection of attributes describe an ***object***
  - Object is also known as record, point, case, sample, entity, or instance

Attributes

Objects

Tid	Refund	Marital Status	Taxable Income	Cheat
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

Attribute Values

- ***Attribute values*** are numbers or symbols assigned to an attribute for a particular object

- Distinction between attributes and attribute values
  - Same attribute can be mapped to different attribute values
    - Example: height can be measured in feet or meters
  - Different attributes can be mapped to the same set of values
    - Example: Attribute values for ID and age are integers
  - But properties of attribute can be different than the properties of the values used to represent the attribute

Types of Attributes

- There are different types of attributes
  - Nominal
    - Examples: ID numbers, zip codes
  - Ordinal
    - Examples: rankings (e.g., taste of potato chips on a scale from 1-10), grades, height {tall, medium, short}
  - Interval
    - Examples: calendar dates, temperatures in Celsius or Fahrenheit.
  - Ratio
    - Examples: temperature in Kelvin, length, counts, elapsed time (e.g., time to run a race)

Properties of Attribute Values

- The type of an attribute depends on which of the following properties/operations it possesses:
  - Distinctness: $= \neq$
  - Order: $< >$
  - Differences are meaningful : $+ -$
  - Ratios are meaningful $* /$

  - Nominal attribute: distinctness
  - Ordinal attribute: distinctness & order
  - Interval attribute: distinctness, order & meaningful differences
  - Ratio attribute: all 4 properties/operations

Difference Between Ratio and Interval

- Is it physically meaningful to say that a temperature of 10 ° is twice that of 5° on
  - the Celsius scale?
  - the Fahrenheit scale?
  - the Kelvin scale?

- Consider measuring the height above average
  - If Bill's height is three inches above average and Bob's height is six inches above average, then would we say that Bob is twice as tall as Bill?
  - Is this situation analogous to that of temperature?

	Attribute Type	Description	Examples	Operations
Categorical Qualitative	Nominal	Nominal attribute values only distinguish. (=, ≠)	zip codes, employee ID numbers, eye color, sex: {*male, female*}	mode, entropy, contingency correlation, $\chi 2$ test
	Ordinal	Ordinal attribute values also order objects. (<, >)	hardness of minerals, {*good, better, best*}, grades, street numbers	median, percentiles, rank correlation, run tests, sign tests
Numeric Quantitative	Interval	For interval attributes, differences between values are meaningful. (+, -)	calendar dates, temperature in Celsius or Fahrenheit	mean, standard deviation, Pearson's correlation, *t* and *F* tests
	Ratio	For ratio variables, both differences and ratios are meaningful. (*, /)	temperature in Kelvin, monetary quantities, counts, age, mass, length, current	geometric mean, harmonic mean, percent variation

This categorization of attributes is due to S. S. Stevens

	Attribute Type	Transformation	Comments
Categorical Qualitative	Nominal	Any permutation of values	If all employee ID numbers were reassigned, would it make any difference?
	Ordinal	An order preserving change of values, i.e., *new_value = f(old_value)* where *f* is a monotonic function	An attribute encompassing the notion of good, better best can be represented equally well by the values {1, 2, 3} or by { 0.5, 1, 10}.
Numeric Quantitative	Interval	*new_value = a * old_value + b* where a and b are constants	Thus, the Fahrenheit and Celsius temperature scales differ in terms of where their zero value is and the size of a unit (degree).
	Ratio	*new_value = a * old_value*	Length can be measured in meters or feet.

This categorization of attributes is due to S. S. Stevens

Discrete and Continuous Attributes

- Discrete Attribute
  - Has only a finite or countably infinite set of values
  - Examples: zip codes, counts, or the set of words in a collection of documents
  - Often represented as integer variables.
  - Note: binary attributes are a special case of discrete attributes
- Continuous Attribute
  - Has real numbers as attribute values
  - Examples: temperature, height, or weight.
  - Practically, real values can only be measured and represented using a finite number of digits.
  - Continuous attributes are typically represented as floating-point variables.

Asymmetric Attributes

- Only presence (a non-zero attribute value) is regarded as important
  - Words present in documents
  - Items present in customer transactions

- If we met a friend in the grocery store would we ever say the following?

  "I see our purchases are very similar since we didn't buy most of the same things."

Key Messages for Attribute Types

- The types of operations you choose should be "meaningful" for the type of data you have
  - Distinctness, order, meaningful intervals, and meaningful ratios are only four (among many possible) properties of data
  - The data type you see – often numbers or strings – may not capture all the properties or may suggest properties that are not present
  - Analysis may depend on these other properties of the data
    - Many statistical analyses depend only on the distribution
  - In the end, what is meaningful can be specific to domain

Important Characteristics of Data

- Dimensionality (number of attributes)
  - High dimensional data brings a number of challenges
- Sparsity
  - Only presence counts
- Resolution
  - Patterns depend on the scale
- Size
  - Type of analysis may depend on size of data

Types of data sets

- Record
  - Data Matrix
  - Document Data
  - Transaction Data
- Graph
  - World Wide Web
  - Molecular Structures
- Ordered
  - Spatial Data
  - Temporal Data
  - Sequential Data
  - Genetic Sequence Data

Record Data

Data that consists of a collection of records, each of which consists of a fixed set of attributes

Tid	Refund	Marital Status	Taxable Income	Cheat
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

Data Matrix

- If data objects have the same fixed set of numeric attributes, then the data objects can be thought of as points in a multi-dimensional space, where each dimension represents a distinct attribute

- Such a data set can be represented by an *m* by *n* matrix, where there are *m* rows, one for each object, and *n* columns, one for each attribute

Projection of x Load	Projection of y load	Distance	Load	Thickness
10.23	5.27	15.22	2.7	1.2
12.65	6.25	16.22	2.2	1.1

Document Data

- Each document becomes a 'term' vector
  - Each term is a component (attribute) of the vector
  - The value of each component is the number of times the corresponding term occurs in the document.

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

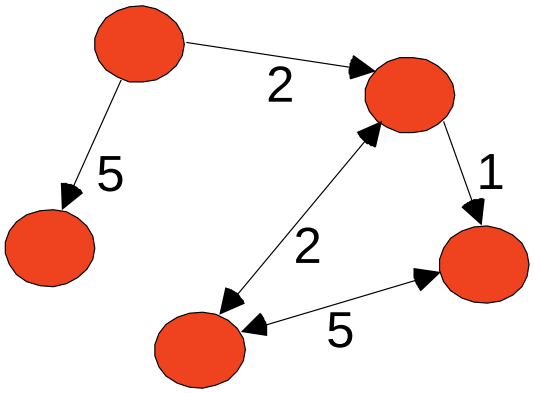
Transaction Data

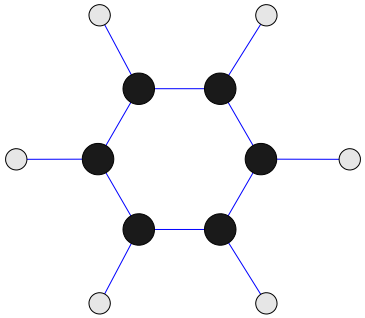
- A special type of data, where
  - Each transaction involves a set of items.
  - For example, consider a grocery store. The set of products purchased by a customer during one shopping trip constitute a transaction, while the individual products that were purchased are the items.
  - Can represent transaction data as record data

TID	*Items*
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

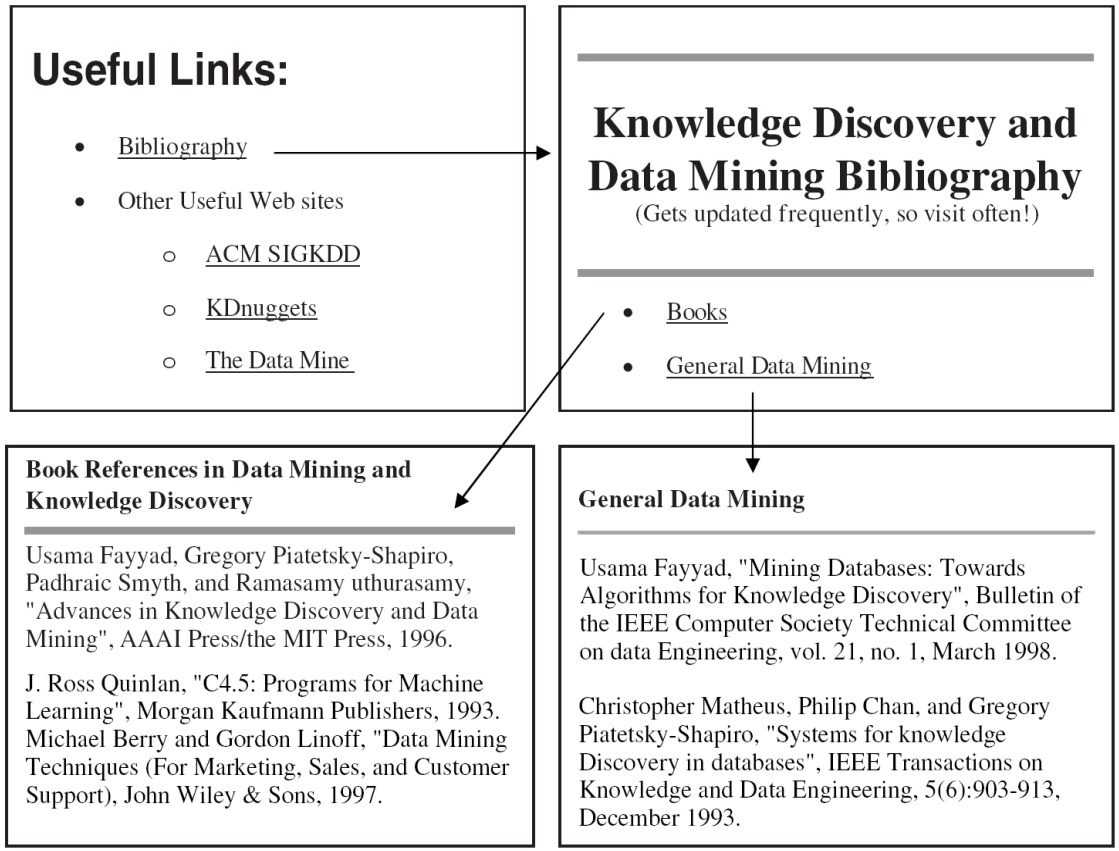
Graph Data

Examples: Generic graph, a molecule, and webpages





Benzene Molecule: C6H6



Ordered Data

Sequences of transactions

Items/Events

(A B) (D) (C E)
(B D) (C) (E)
(C D) (B) (A E)

An element of the sequence

Ordered Data

Genomic sequence data

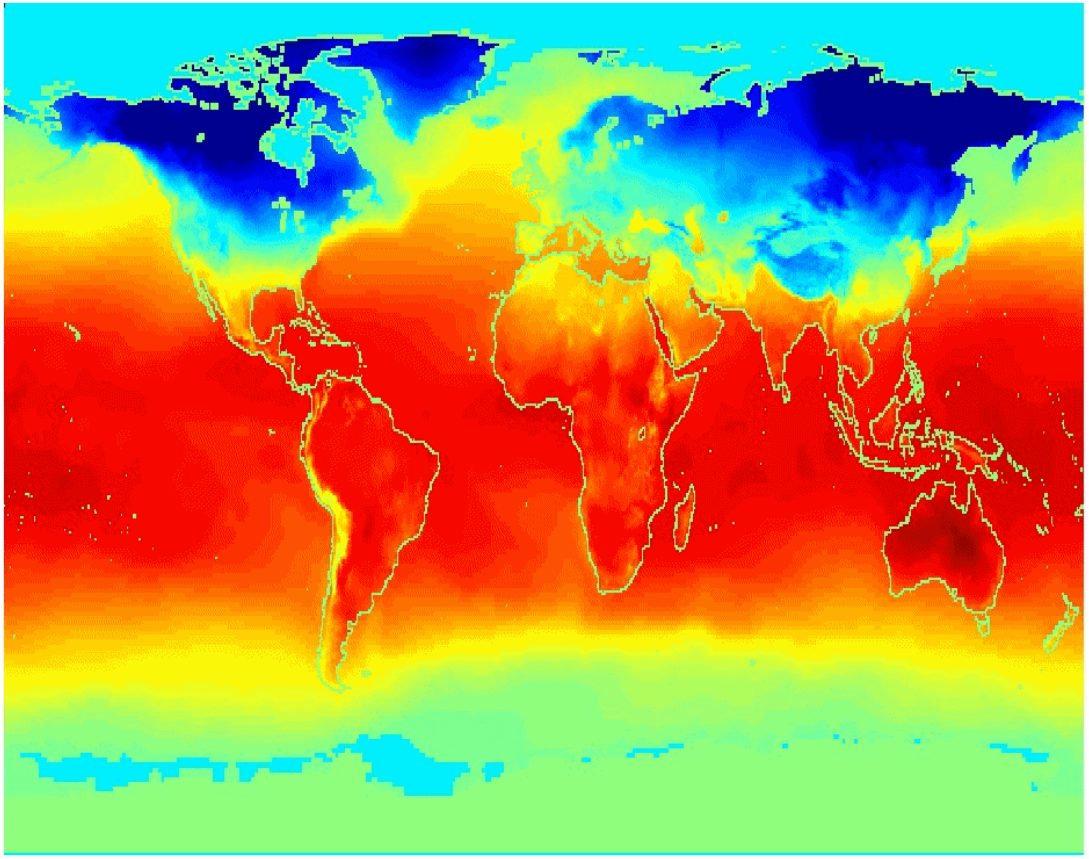
```
GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG
```

Ordered Data

Spatiotemporal Data

Jan

Average Monthly Temperature of land and ocean



Data Quality

- Poor data quality negatively affects many data processing efforts

- ML example: a classification model for detecting people who are loan risks is built using poor data
  - Some credit-worthy candidates are denied loans
  - More loans are given to individuals that default

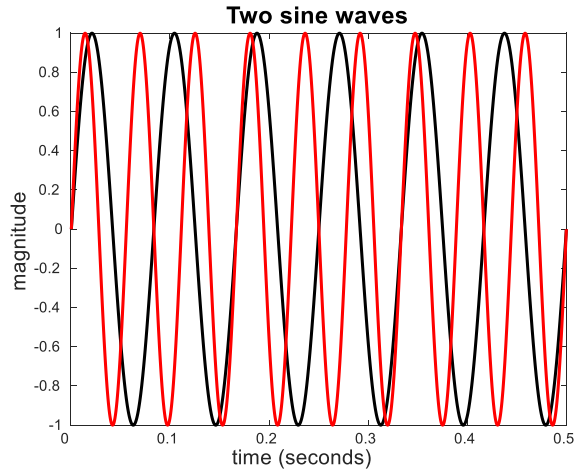
Data Quality …

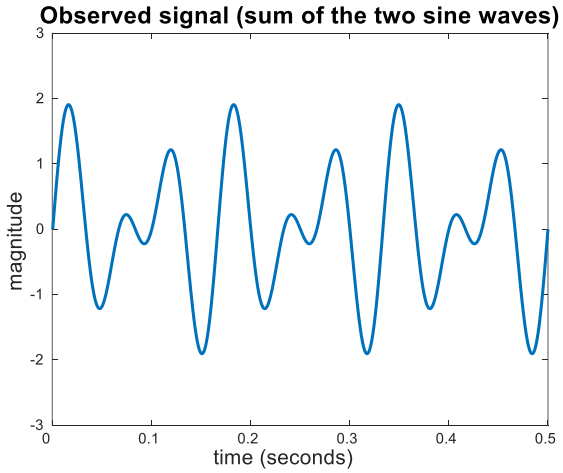
- What kinds of data quality problems?
- How can we detect problems with the data?
- What can we do about these problems?

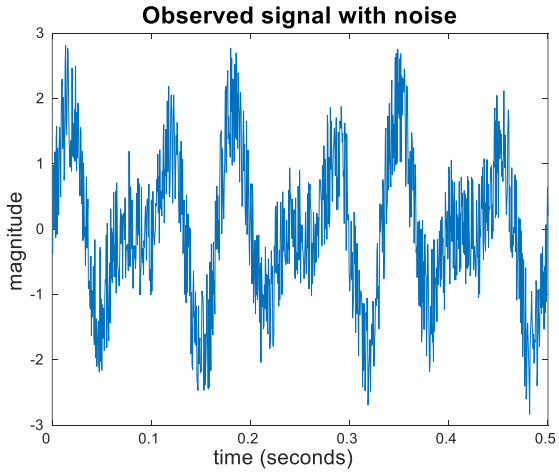
- Examples of data quality problems:
  - Noise and outliers
  - Wrong data
  - Fake data
  - Missing values
  - Duplicate data

Noise

- For objects, noise is an extraneous object
- For attributes, noise refers to modification of original values
  - Examples: distortion of a person's voice when talking on a poor phone and "snow" on television screen
  - The figures below show two sine waves of the same magnitude and different frequencies, the waves combined, and the two sine waves with random noise
    - The magnitude and shape of the original signal is distorted

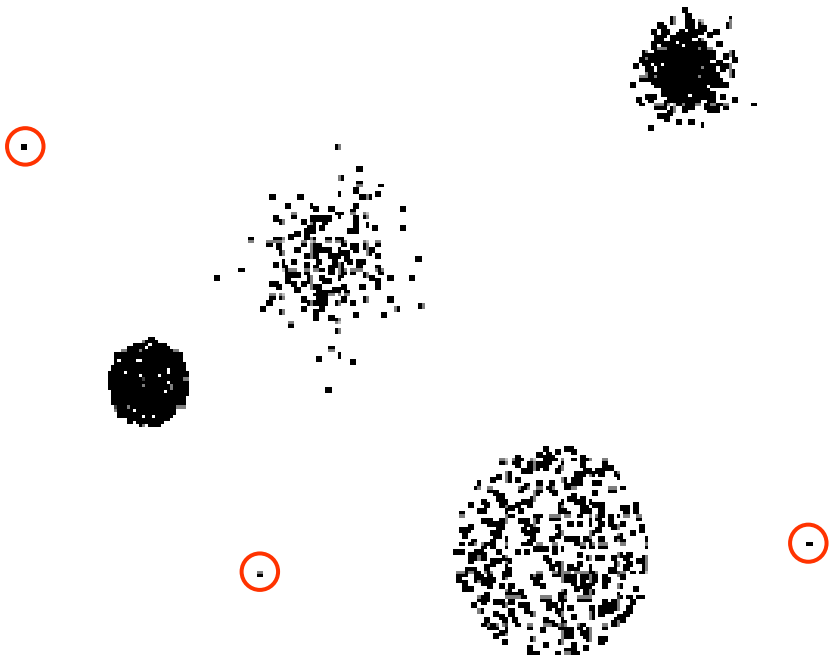






Outliers

- ***Outliers*** are data objects with characteristics that are considerably different than most of the other data objects in the data set
  - **Case 1:** Outliers are noise that interferes with data analysis
  - **Case 2:** Outliers are the goal of our analysis
    - Credit card fraud
    - Intrusion detection
- Causes?



Missing Values

- Reasons for missing values
  - Information is not collected
    (e.g., people decline to give their age and weight)
  - Attributes may not be applicable to all cases
    (e.g., annual income is not applicable to children)

- Handling missing values
  - Eliminate data objects or variables
  - Estimate missing values
    - Example: time series of temperature
    - Example: census results
  - Ignore the missing value during analysis

Duplicate Data

- Data set may include data objects that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources
- Examples:
  - Same person with multiple email addresses
- Data cleaning
  - Process of dealing with duplicate data issues

Data Preprocessing

- Aggregation
- Sampling
- Discretization and Binarization
- Attribute Transformation
- Dimensionality Reduction
- Feature subset selection
- Feature creation

Aggregation

- Combining two or more attributes (or objects) into a single attribute (or object)
- Purpose
  - Data reduction - reduce the number of attributes or objects
  - Change of scale
    - Cities aggregated into regions, states, countries, etc.
    - Days aggregated into weeks, months, or years
  - More "stable" data - aggregated data tends to have less variability

Table 2.4. Data set containing information about customer purchases.

Transaction ID	Item	Store Location	Date	Price	...
⋮	⋮	⋮	⋮	⋮	
101123	Watch	Chicago	09/06/04	$25.99	...
101123	Battery	Chicago	09/06/04	$5.99	...
101124	Shoes	Minneapolis	09/06/04	$75.00	...
⋮	⋮	⋮	⋮	⋮	

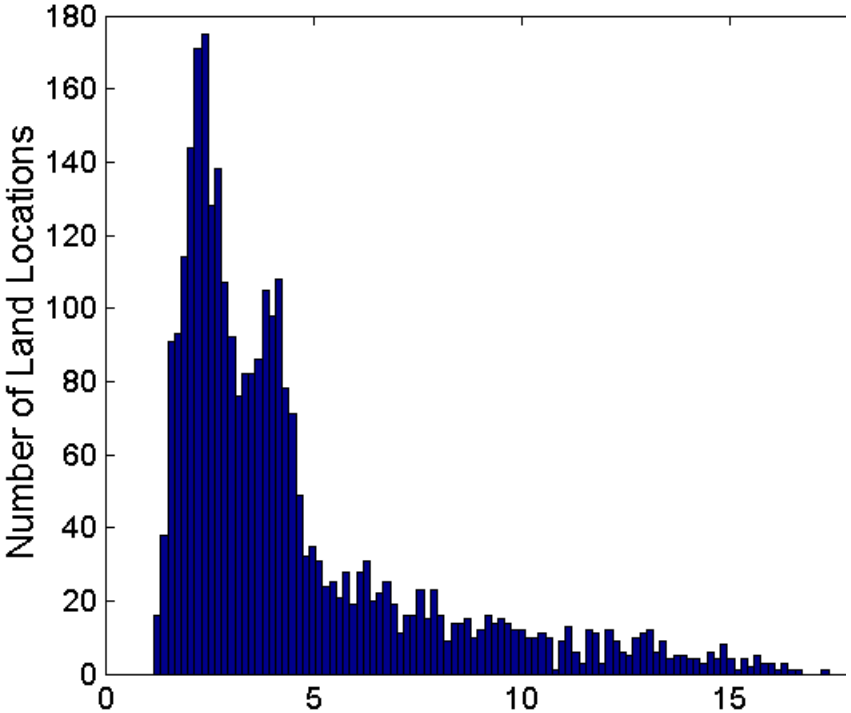
Example: Precipitation in Australia

- This example is based on precipitation in Australia from the period 1982 to 1993.

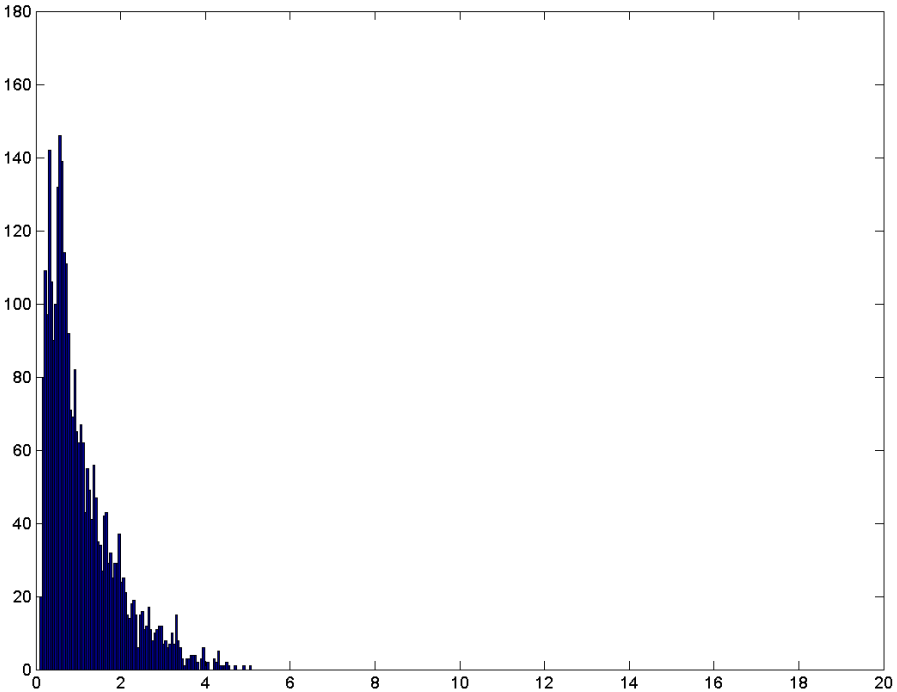
  The next slide shows
  - A histogram for the standard deviation of average monthly precipitation for 3,030 0.5◦ by 0.5◦ grid cells in Australia, and
  - A histogram for the standard deviation of the average yearly precipitation for the same locations.
- The average yearly precipitation has less variability than the average monthly precipitation.
- All precipitation measurements (and their standard deviations) are in centimeters.

Example: Precipitation in Australia (contd)

Variation of Precipitation in Australia



Standard Deviation of Average Monthly Precipitation



Standard Deviation of Average Yearly Precipitation

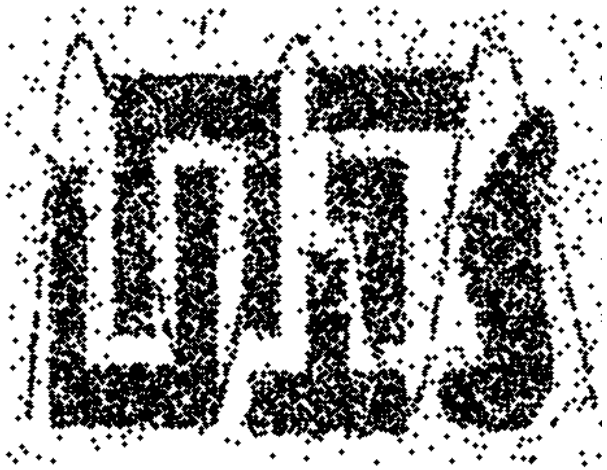
Sampling

- Sampling is the main technique employed for data reduction.
  - It is often used for both the preliminary investigation of the data and the final data analysis.
- Statisticians often sample because obtaining the entire set of data of interest is too expensive or time consuming.
- Sampling is typically used in data mining because processing the entire set of data of interest is too expensive or time consuming.

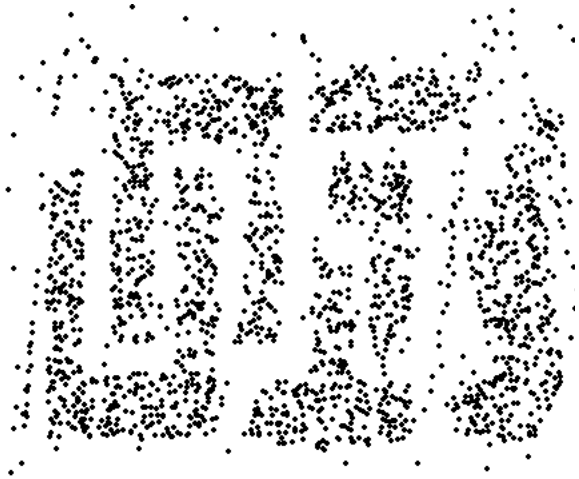
Sampling …

- The key principle for effective sampling is the following:
  - Using a sample will work almost as well as using the entire data set, if the sample is representative
  - A sample is representative if it has approximately the same properties (of interest) as the original set of data

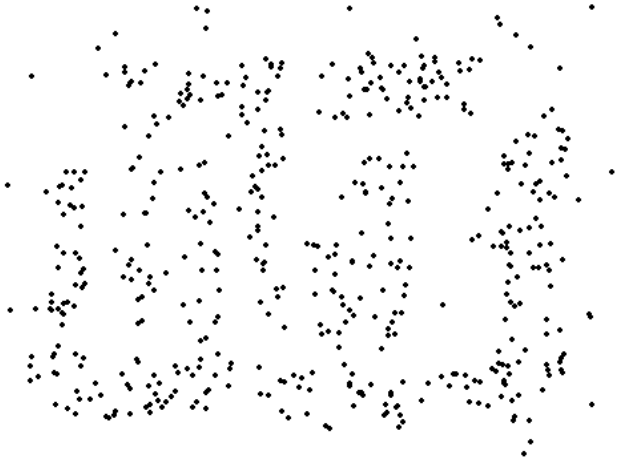
Sample Size



8000 points

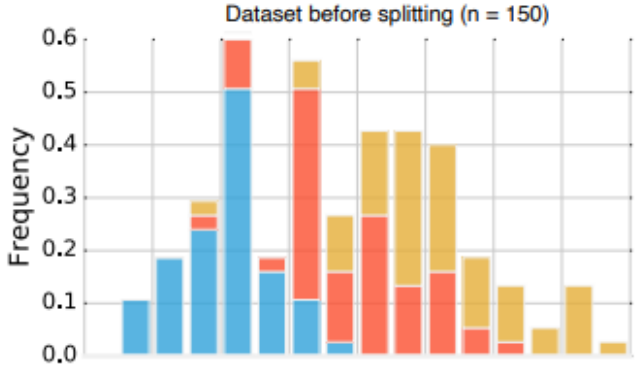


2000 Points



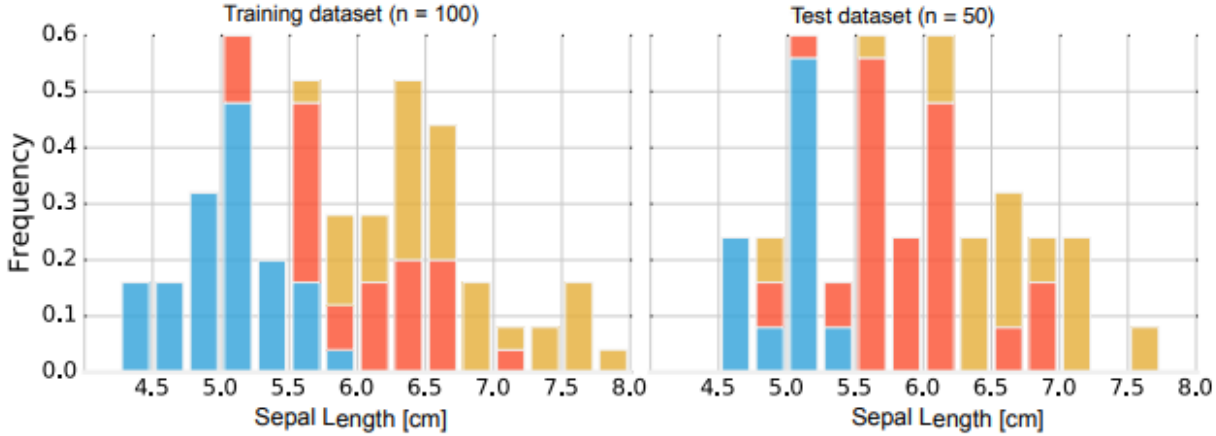
500 Points

Issues with Subsampling (Independence Violation)



IRIS Dataset of Flowers

50 Setosa, 50 Versicolor, and 50 Virginica



- **Random subsampling can assign 2/3 (100) to training set and 1/3 (50) to the test set**
- **Training set → 38 x Setosa, 28 x Versicolor, 34 x Virginica**
- **Test set → 12 x Setosa, 22 x Versicolor, 16 x Virginica**

Types of Sampling

- Simple Random Sampling
  - There is an equal probability of selecting any particular item
  - Sampling without replacement
    - As each item is selected, it is removed from the population
  - Sampling with replacement
    - Objects are not removed from the population as they are selected for the sample.
    - In sampling with replacement, the same object can be picked up more than once
- Stratified sampling
  - Split the data into several partitions; then draw random samples from each partition

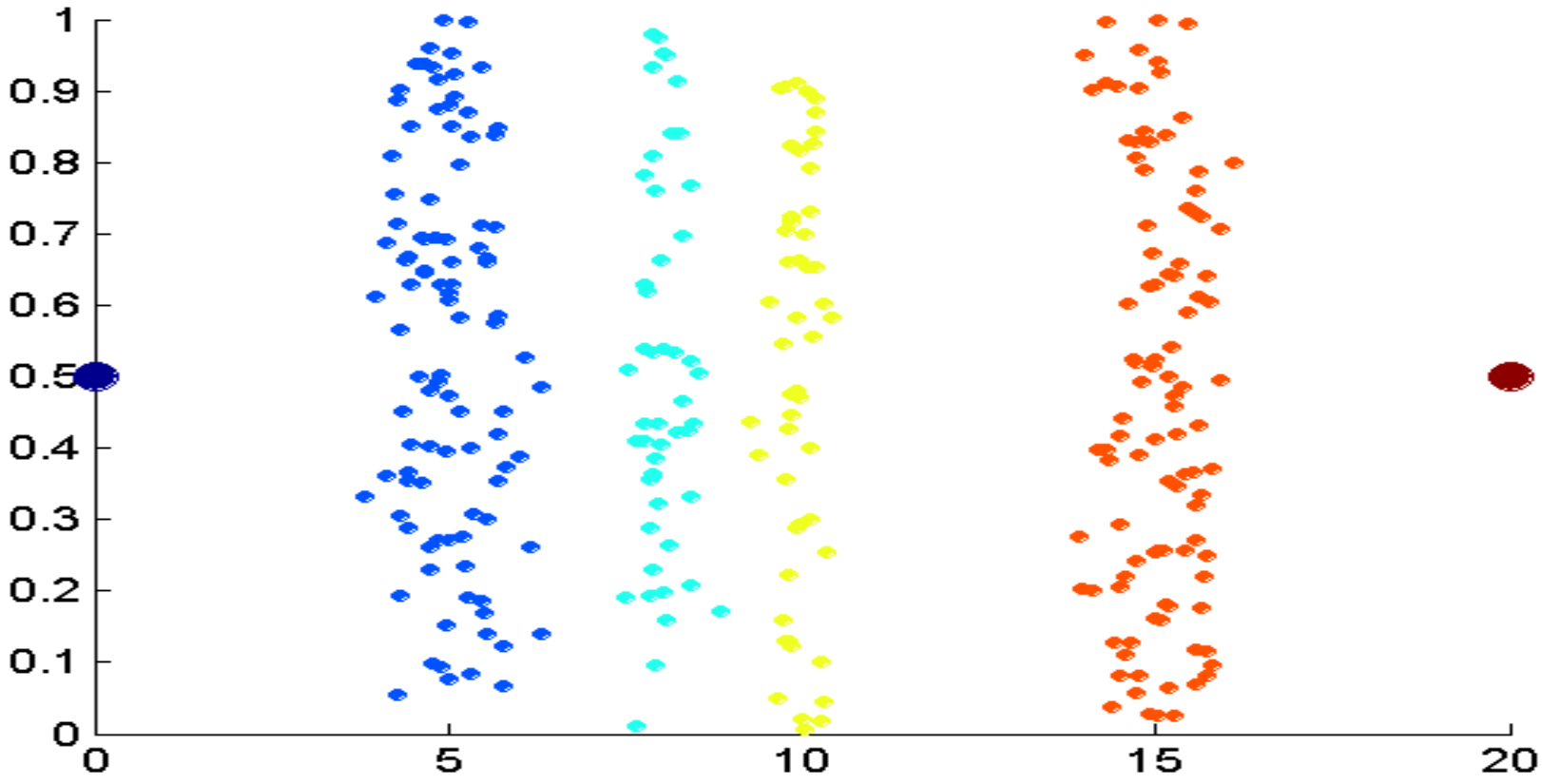
Building Classifiers with Imbalanced Training Set

- Modify the distribution of training data so that rare class is well-represented in training set
  - Undersample the majority class
  - Oversample the rare class

Discretization

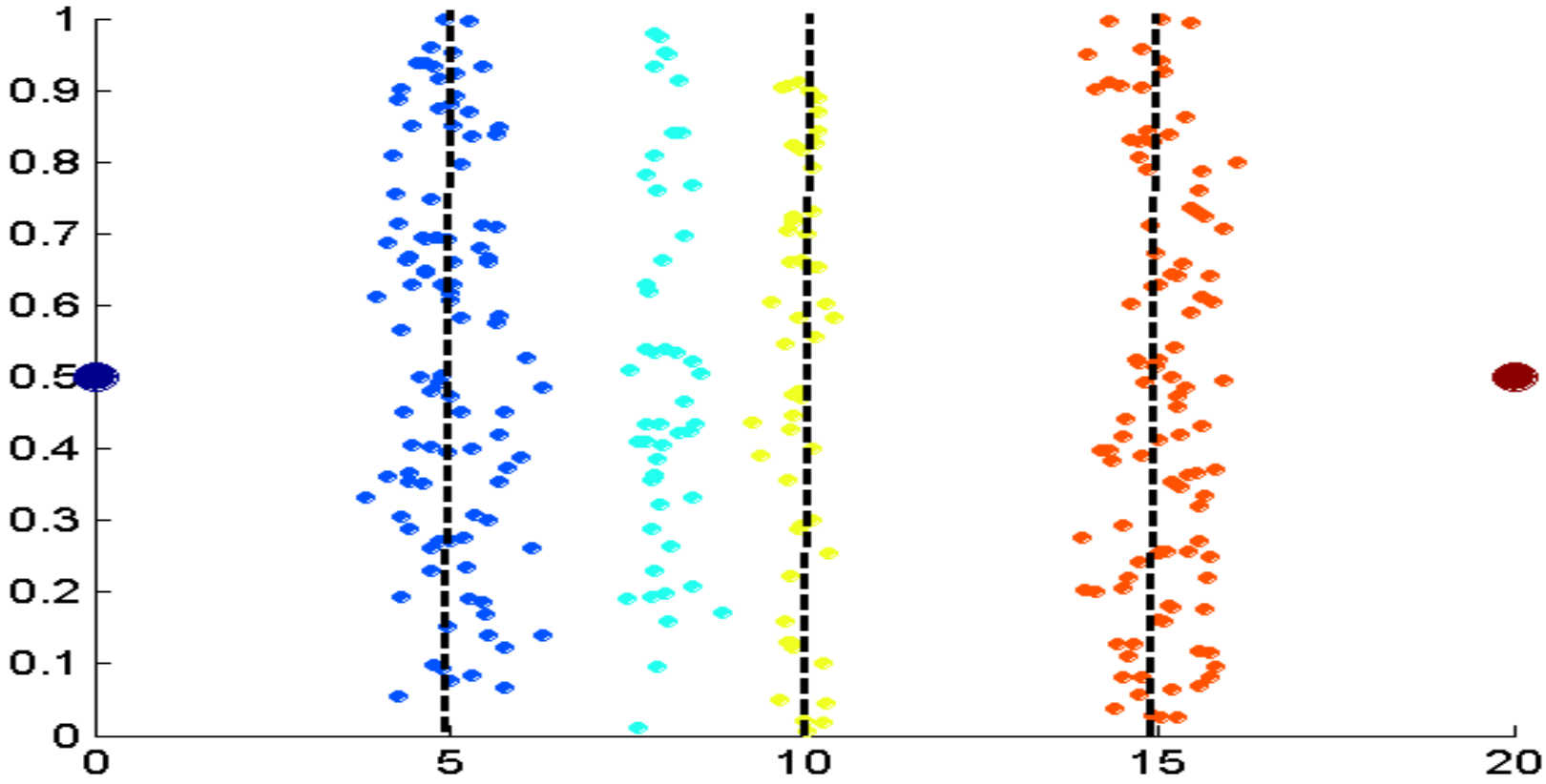
- Discretization is the process of converting a continuous attribute into an ordinal attribute
  - A potentially infinite number of values are mapped into a small number of categories
  - Discretization is used in both unsupervised and supervised settings

Unsupervised Discretization



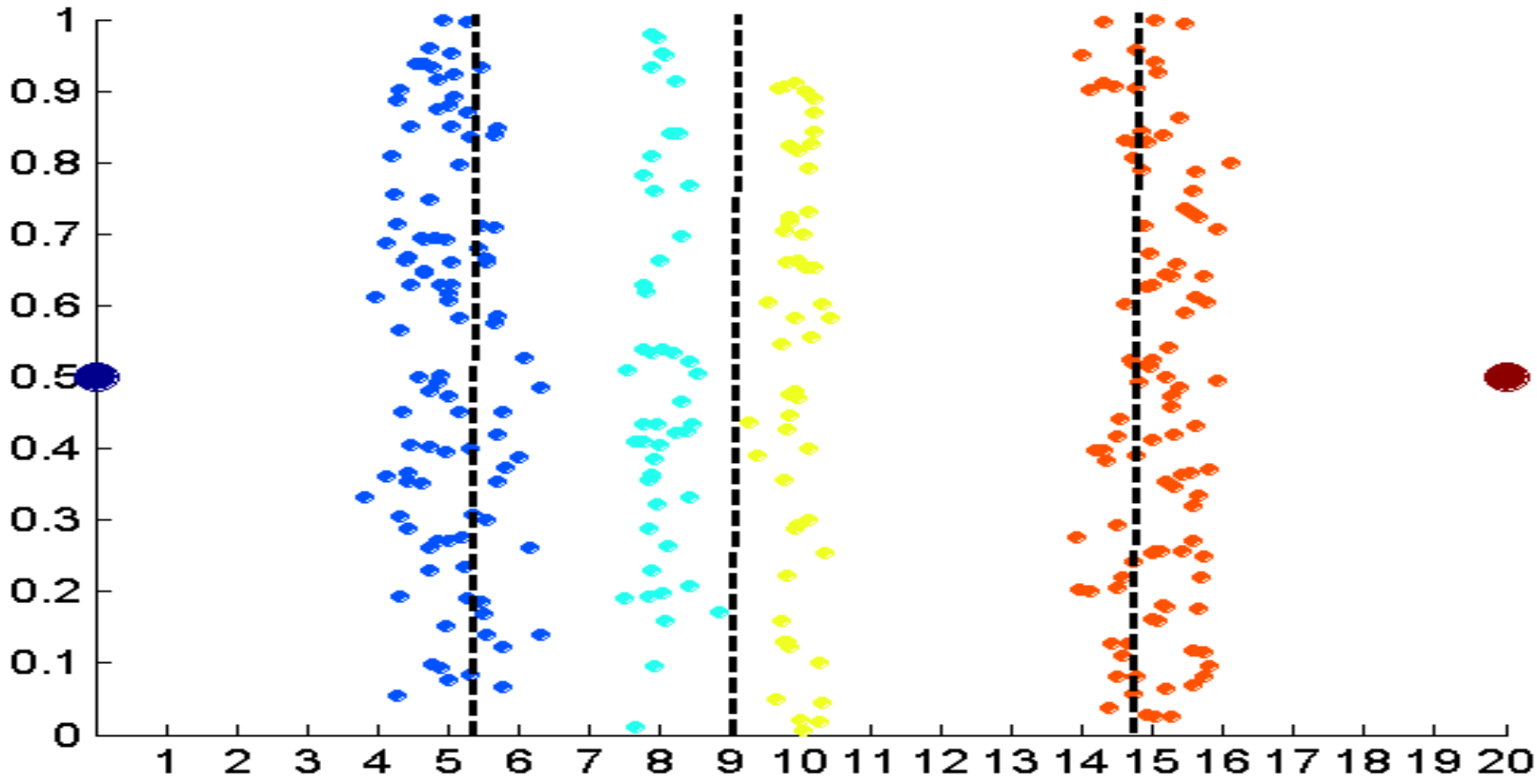
Data consists of four groups of points and two outliers. Data is one-dimensional, but a random y component is added to reduce overlap.

Unsupervised Discretization



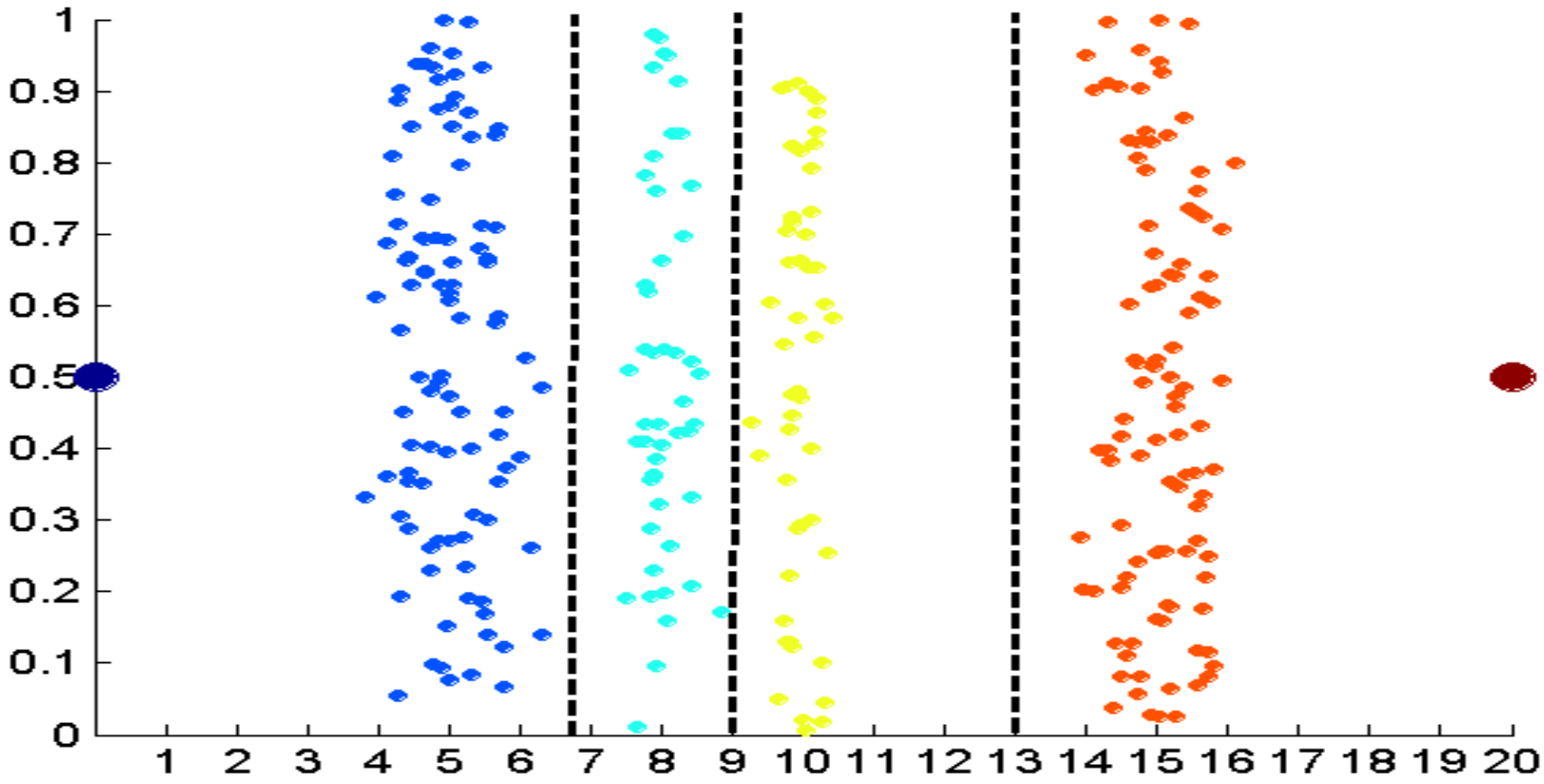
Equal interval width approach used to obtain 4 values.

Unsupervised Discretization



Equal frequency approach used to obtain 4 values.

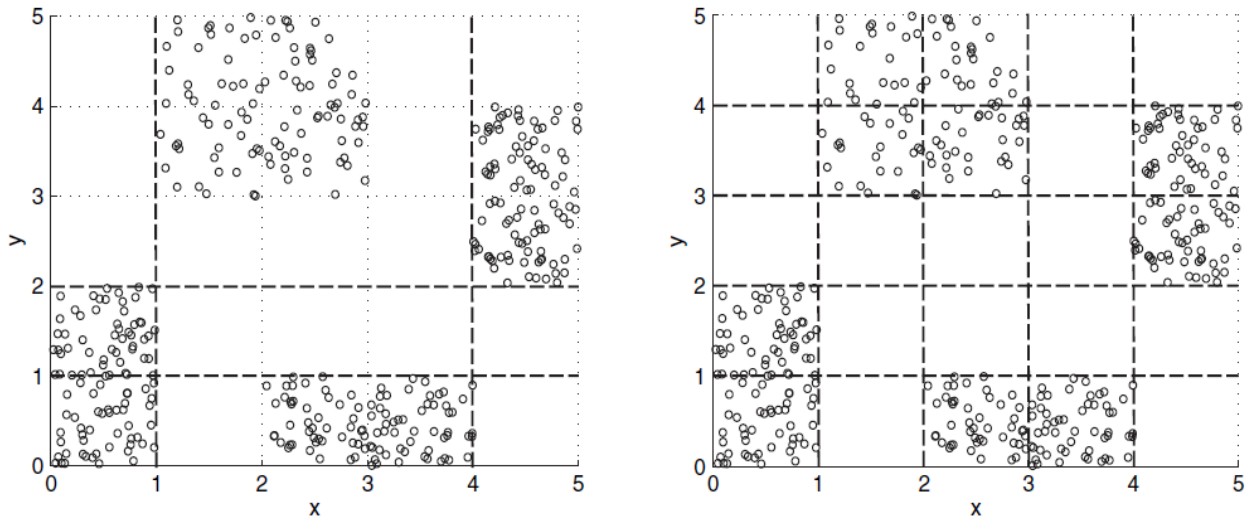
Unsupervised Discretization



K-means approach to obtain 4 values.

Discretization in Supervised Settings

- Many classification algorithms work best if both the independent and dependent variables have only a few values
- We give an illustration of the usefulness of discretization using the following example.



(a) Three intervals (b) Five intervals

Figure 2.14. Discretizing x and y attributes for four groups (classes) of points.

Binarization

- Binarization maps a continuous or categorical attribute into one or more binary variables

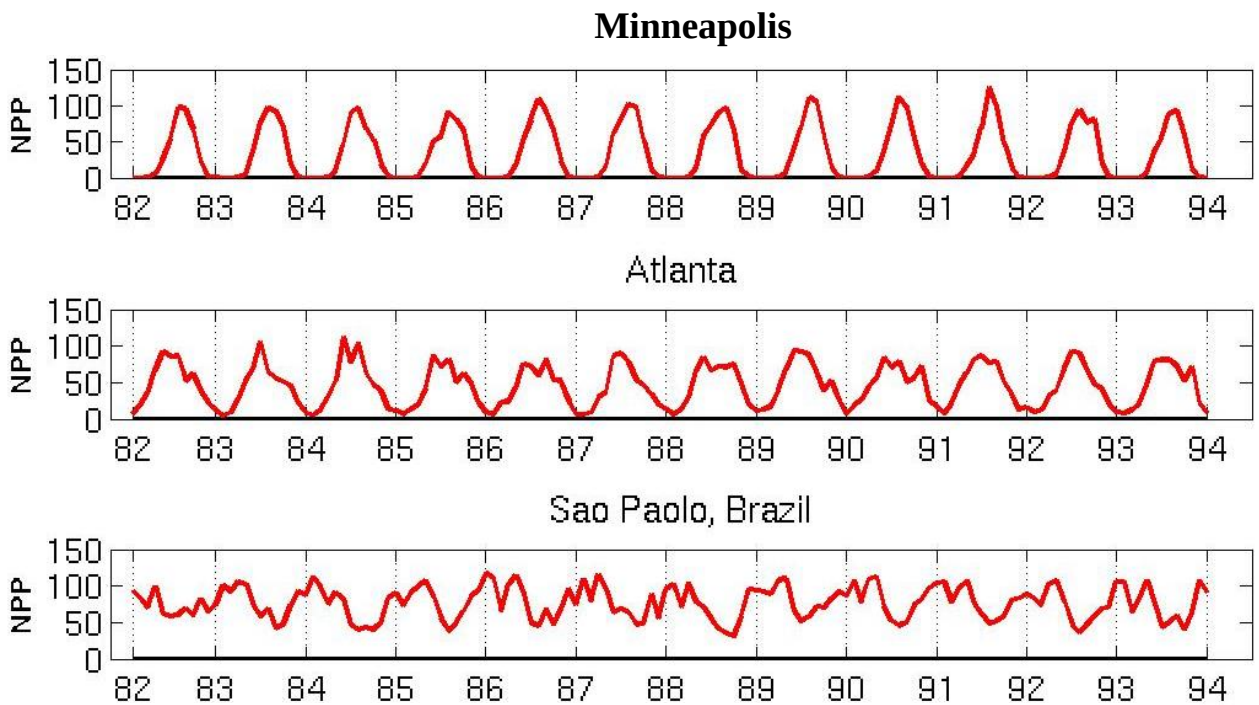
Table 2.6. Conversion of a categorical attribute to five asymmetric binary attributes.

Categorical Value	Integer Value	x_1	x_2	x_3	x_4	x_5
awful	0	1	0	0	0	0
poor	1	0	1	0	0	0
OK	2	0	0	1	0	0
good	3	0	0	0	1	0
great	4	0	0	0	0	1

Attribute Transformation

- An attribute transform is a function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value can be identified with one of the new values
  - Simple functions: x^k, $\log(x)$, e^x, $|x|$
  - Normalization
    - Refers to various techniques to adjust to differences among attributes in terms of frequency of occurrence, mean, variance, range
    - Take out unwanted, common signal, e.g., seasonality
  - In statistics, standardization refers to subtracting off the means and dividing by the standard deviation

Example: Sample Time Series of Plant Growth

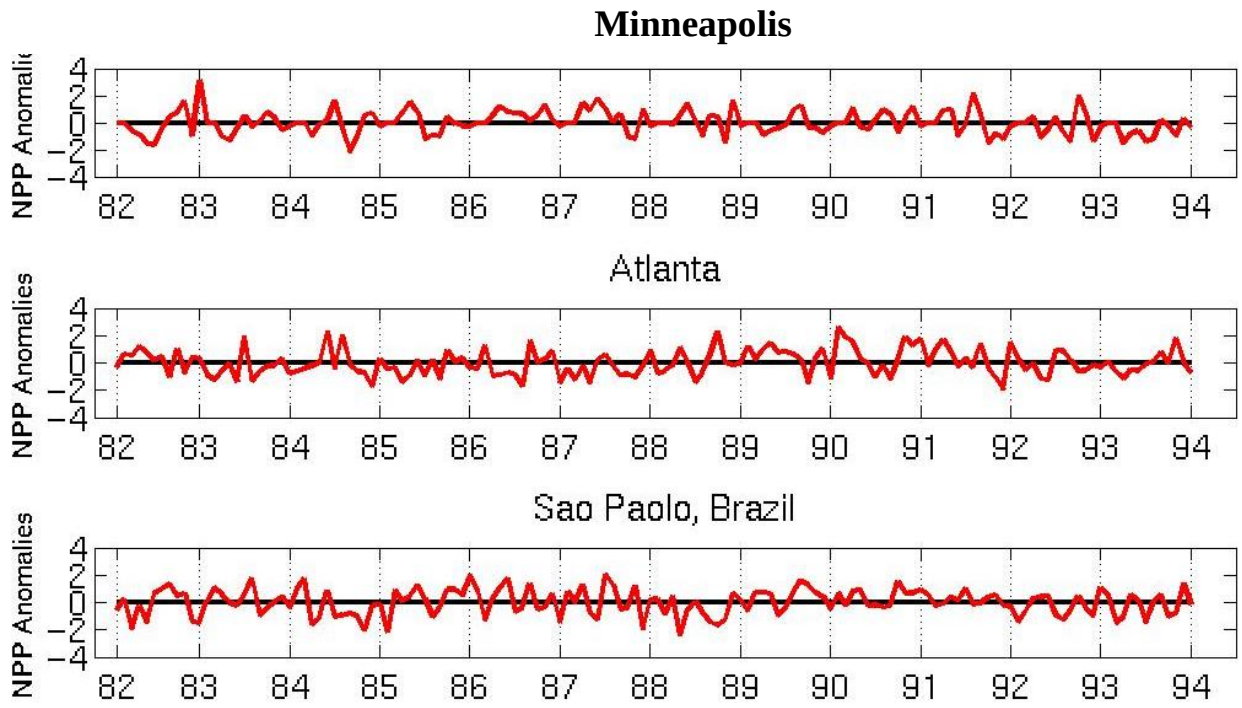


Net Primary Production (NPP) is a measure of plant growth used by ecosystem scientists.

Correlations between time series

	Minneapolis	Atlanta	Sao Paolo
Minneapolis	1.0000	0.7591	-0.7581
Atlanta	0.7591	1.0000	-0.5739
Sao Paolo	-0.7581	-0.5739	1.0000

Seasonality Accounts for Much Correlation



Normalized using monthly Z Score:

Subtract off monthly mean and divide by monthly standard deviation

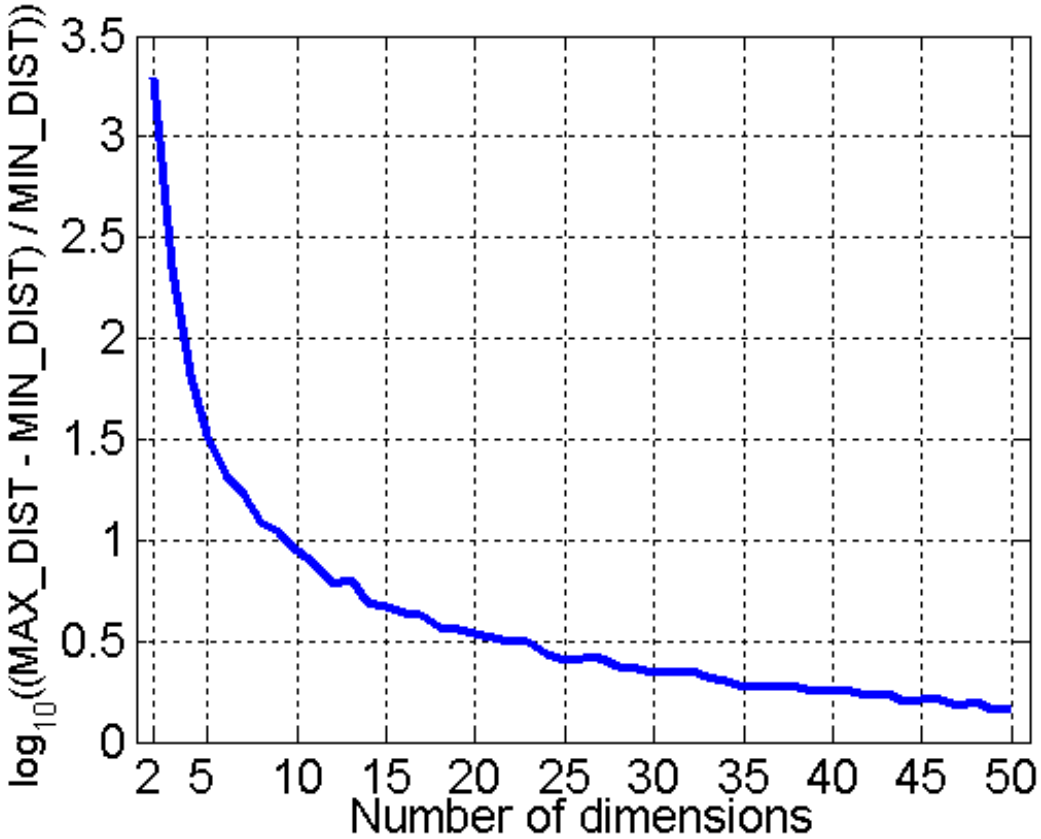
Correlations between time series

	Minneapolis	Atlanta	Sao Paolo
Minneapolis	1.0000	0.0492	0.0906
Atlanta	0.0492	1.0000	-0.0154
Sao Paolo	0.0906	-0.0154	1.0000

Curse of Dimensionality

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies

- Definitions of density and distance between points, which are critical for clustering and outlier detection, become less meaningful



- **Randomly generate 500 points**
- **Compute difference between max and min distance between any pair of points**

Dimensionality Reduction

- Purpose:
  - Avoid curse of dimensionality
  - Reduce amount of time and memory required by data mining algorithms
  - Allow data to be more easily visualized
  - May help to eliminate irrelevant features or reduce noise
- Techniques
  - Principal Components Analysis (PCA)
  - Singular Value Decomposition
  - Others: supervised and non-linear techniques

Feature Subset Selection

- Another way to reduce dimensionality of data
- Redundant features
  - Duplicate much or all of the information contained in one or more other attributes
  - Example: purchase price of a product and the amount of sales tax paid
- Irrelevant features
  - Contain no information that is useful for the data mining task at hand
  - Example: students' ID is often irrelevant to the task of predicting students' GPA
- Many techniques developed, especially for classification

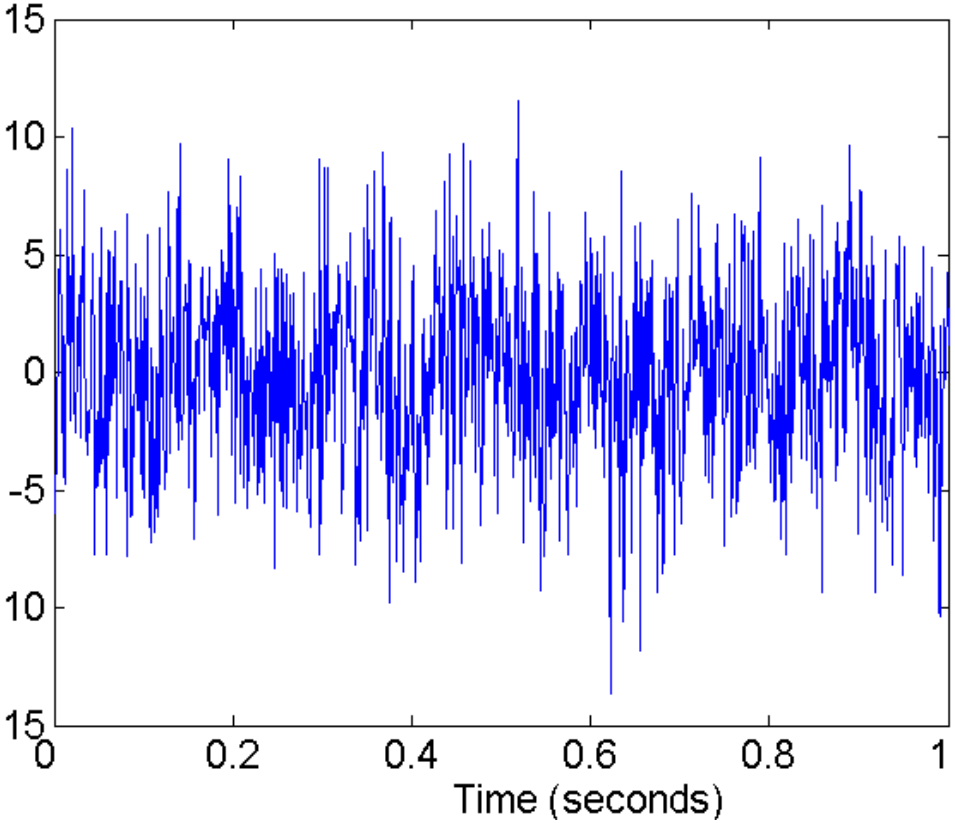
Feature Creation

- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes

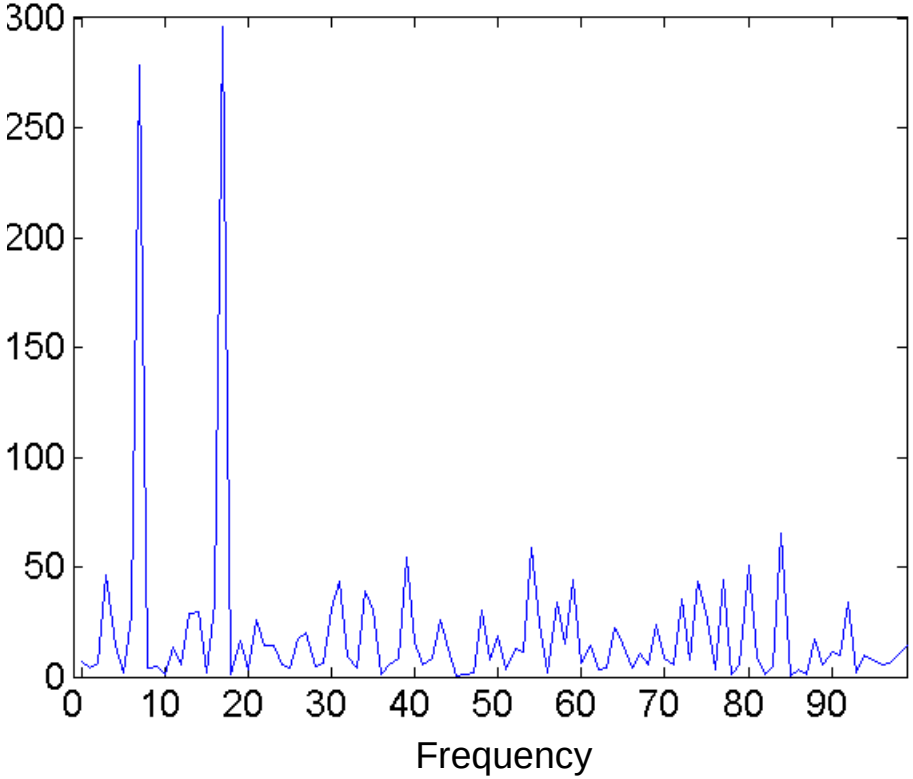
- Three general methodologies:
  - Feature extraction
    - Example: extracting edges from images
  - Feature construction
    - Example: dividing mass by volume to get density
  - Mapping data to new space
    - Example: Fourier and wavelet analysis

Mapping Data to a New Space

Fourier and wavelet transform



Two Sine Waves + Noise



Frequency

Evaluation Metrics: Confusion Matrix

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Evaluation Metrics: Accuracy

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

- Most widely-used metric:

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

Class Imbalance Problem

- 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
  - COVID-19 test results on a random sample

- **Key Challenge**:
  - Evaluation measures such as accuracy are not well-suited for imbalanced class

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 this trivial model does not detect any class YES example
  - Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	0	10
	Class=No	0	990

Which model is better?

A

		PREDICTED	
ACTUAL		Class=Yes	Class=No
	Class=Yes	0	10
	Class=No	0	990

Accuracy: 99%

B

		PREDICTED	
ACTUAL		Class=Yes	Class=No
	Class=Yes	10	0
	Class=No	500	490

Accuracy: 50%

Which model is better?

A

		PREDICTED	
ACTUAL		Class=Yes	Class=No
	Class=Yes	5	5
	Class=No	0	990

B

		PREDICTED	
ACTUAL		Class=Yes	Class=No
	Class=Yes	10	0
	Class=No	500	490

Alternative Measures

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

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

Alternative Measures

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

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

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

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

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

Alternative Measures

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

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

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

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

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

Which of these classifiers is better?

A

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	40	10
	Class=No	10	40

$$\text{Precision (p)} = 0.8$$
$$\text{Recall (r)} = 0.8$$
$$\text{F-measure (F)} = 0.8$$
$$\text{Accuracy} = 0.8$$

B

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	40	10
	Class=No	1000	4000

$$\text{Precision (p)} =\sim 0.04$$
$$\text{Recall (r)} = 0.8$$
$$\text{F-measure (F)} =\sim 0.08$$
$$\text{Accuracy} =\sim 0.8$$

Measures of Classification Performance

	PREDICTED CLASS		
ACTUAL CLASS		Yes	No
	Yes	TP	FN
	No	FP	TN

α is the probability that we reject the null hypothesis when it is true. This is a Type I error or a false positive (FP).

β is the probability that we accept the null hypothesis when it is false. This is a Type II error or a false negative (FN).

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

$$ErrorRate = 1 - accuracy$$

$$Precision = Positive\ Predictive\ Value = \frac{TP}{TP + FP}$$

$$Recall = Sensitivity = TP\ Rate = \frac{TP}{TP + FN}$$

$$Specificity = TN\ Rate = \frac{TN}{TN + FP}$$

$$FP\ Rate = \alpha = \frac{FP}{TN + FP} = 1 - specificity$$

$$FN\ Rate = \beta = \frac{FN}{FN + TP} = 1 - sensitivity$$

$$Power = sensitivity = 1 - \beta$$

Alternative Measures

A	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	10	40

$$\text{Precision (p)} = 0.8$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.2$$
$$\text{F-measure (F)} = 0.8$$
$$\text{Accuracy} = 0.8$$

$$\frac{\text{TPR}}{\text{FPR}} = 4$$

B	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	1000	4000

$$\text{Precision (p)} = 0.038$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.2$$
$$\text{F-measure (F)} = 0.07$$
$$\text{Accuracy} = 0.8$$

$$\frac{\text{TPR}}{\text{FPR}} = 4$$

Which of these classifiers is better?

A	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	10	40
	Class=No	10	40

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.2$$
$$\text{FPR} = 0.2$$
$$\text{F} - \text{measure} = 0.28$$

B	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	25	25
	Class=No	25	25

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.5$$
$$\text{FPR} = 0.5$$
$$\text{F} - \text{measure} = 0.5$$

C	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	40	10
	Class=No	40	10

$$\text{Precision (p)} = 0.5$$
$$\text{TPR} = \text{Recall (r)} = 0.8$$
$$\text{FPR} = 0.8$$
$$\text{F} - \text{measure} = 0.61$$

ROC (Receiver Operating Characteristic)

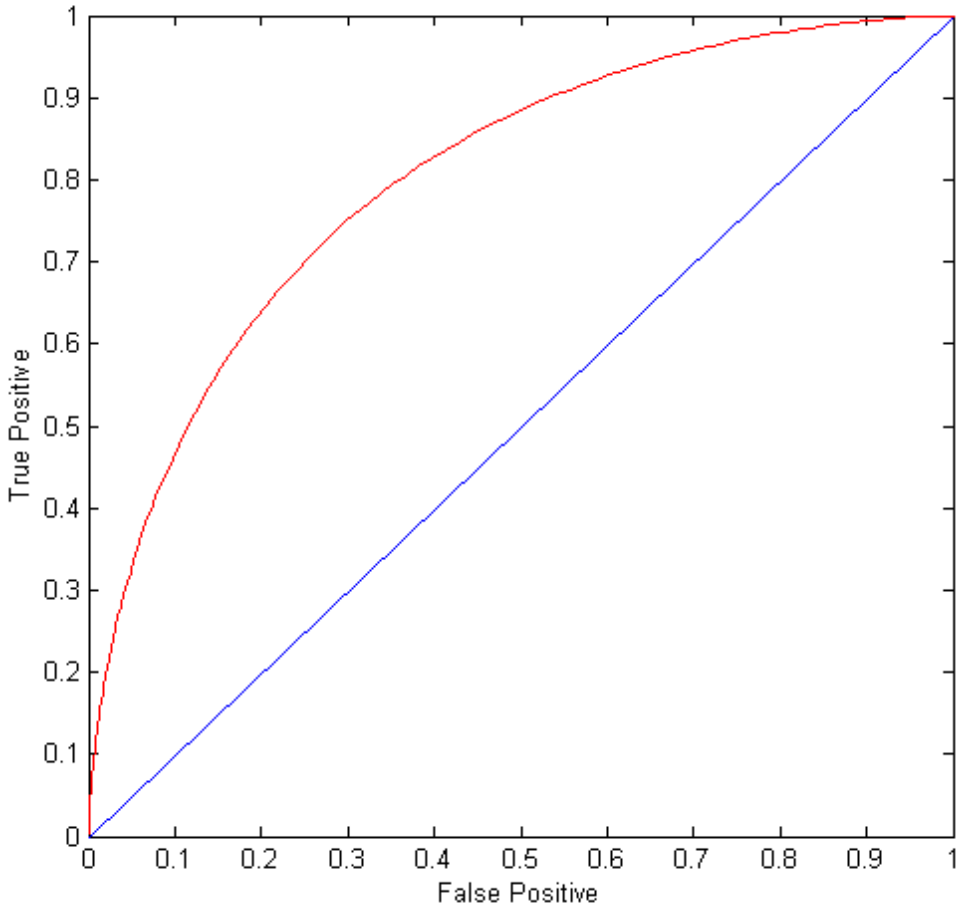
- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- ROC curve plots TPR against FPR
  - Performance of a model represented as a point in an ROC curve

ROC Curve

(TPR,FPR):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal

- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

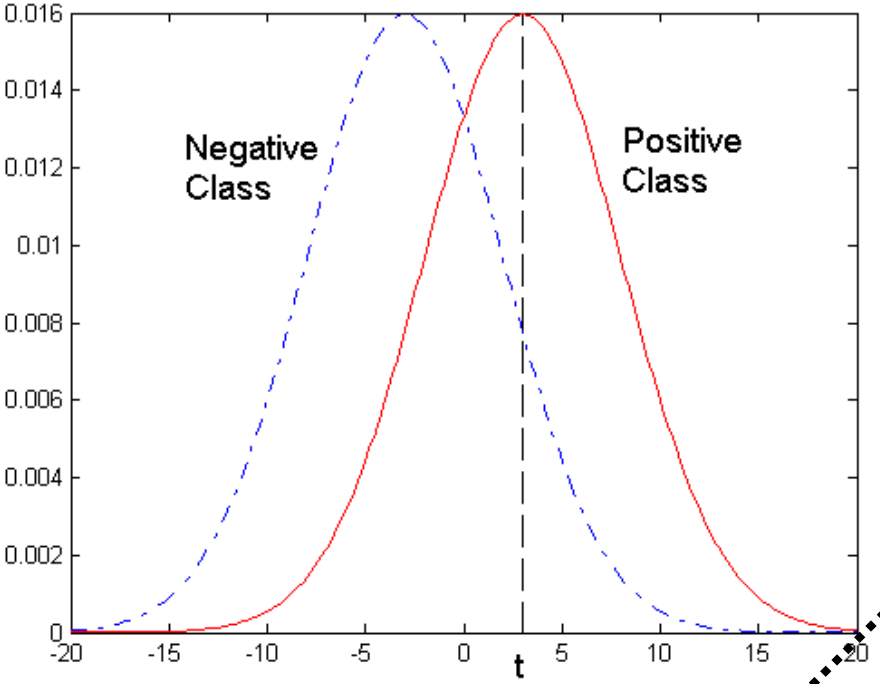


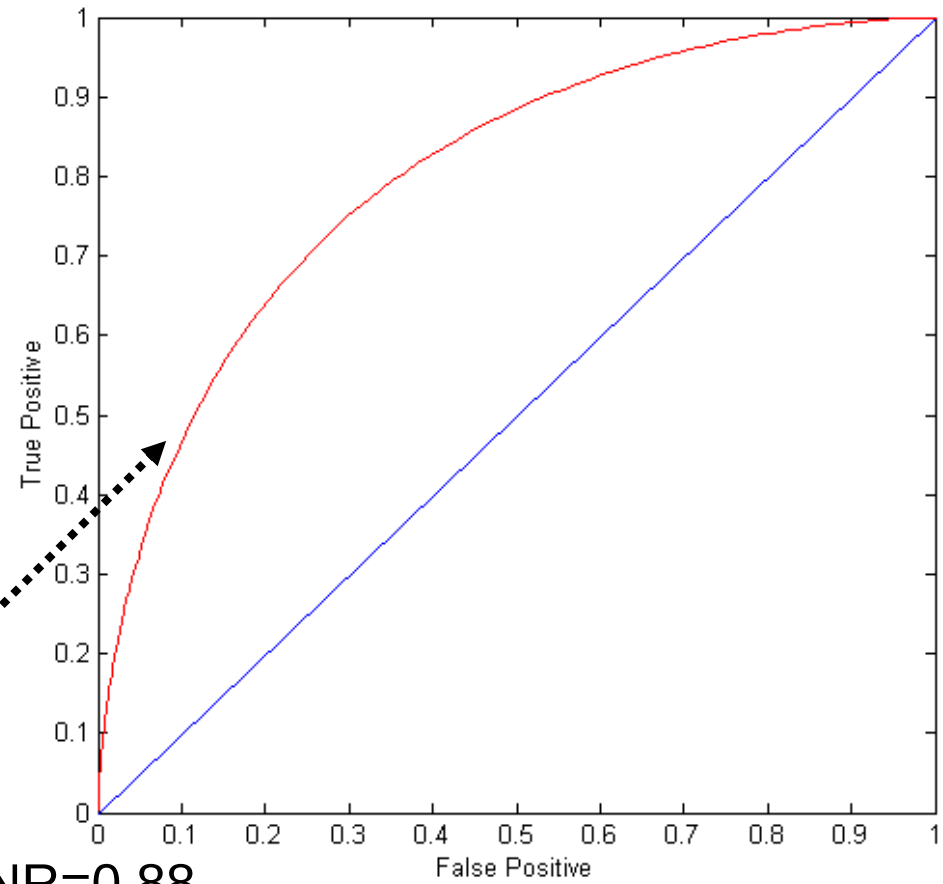
ROC (Receiver Operating Characteristic)

- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record
  - By using different thresholds on this value, we can create different variations of the classifier with TPR/FPR tradeoffs
- Many classifiers produce only discrete outputs (i.e., predicted class)
  - How to get continuous-valued outputs?
    - Decision trees, rule-based classifiers, neural networks, Bayesian classifiers, k-nearest neighbors, SVM

ROC Curve Example

- 1-dimensional data set containing 2 classes (positive and negative)
- Any points located at x > t is classified as positive





At threshold t:

TPR=0.5, FNR=0.5, **FPR=0.12**, TNR=0.88

How to Construct an ROC curve

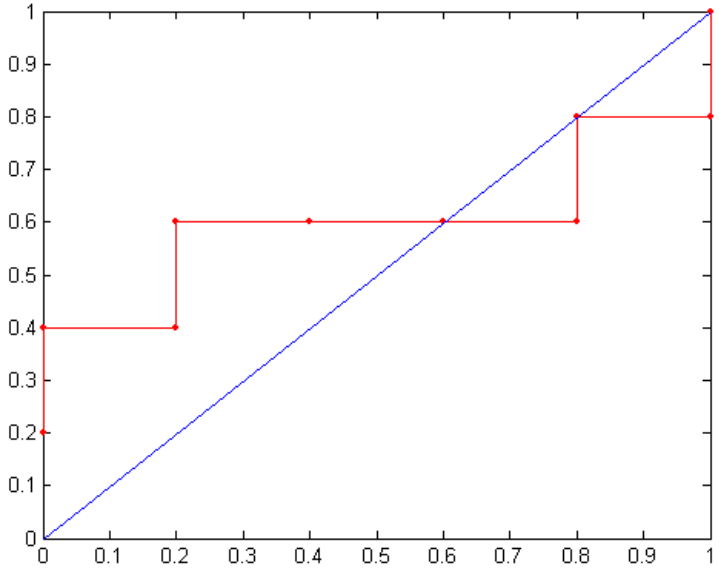
Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use a classifier that produces a continuous-valued score for each instance
  - The more likely it is for the instance to be in the + class, the higher the score
- Sort the instances in decreasing order according to the score
- Apply a threshold at each unique value of the score
- Count the number of TP, FP, TN, FN at each threshold
  - TPR = TP/(TP+FN)
  - FPR = FP/(FP + TN)

How to construct an ROC curve

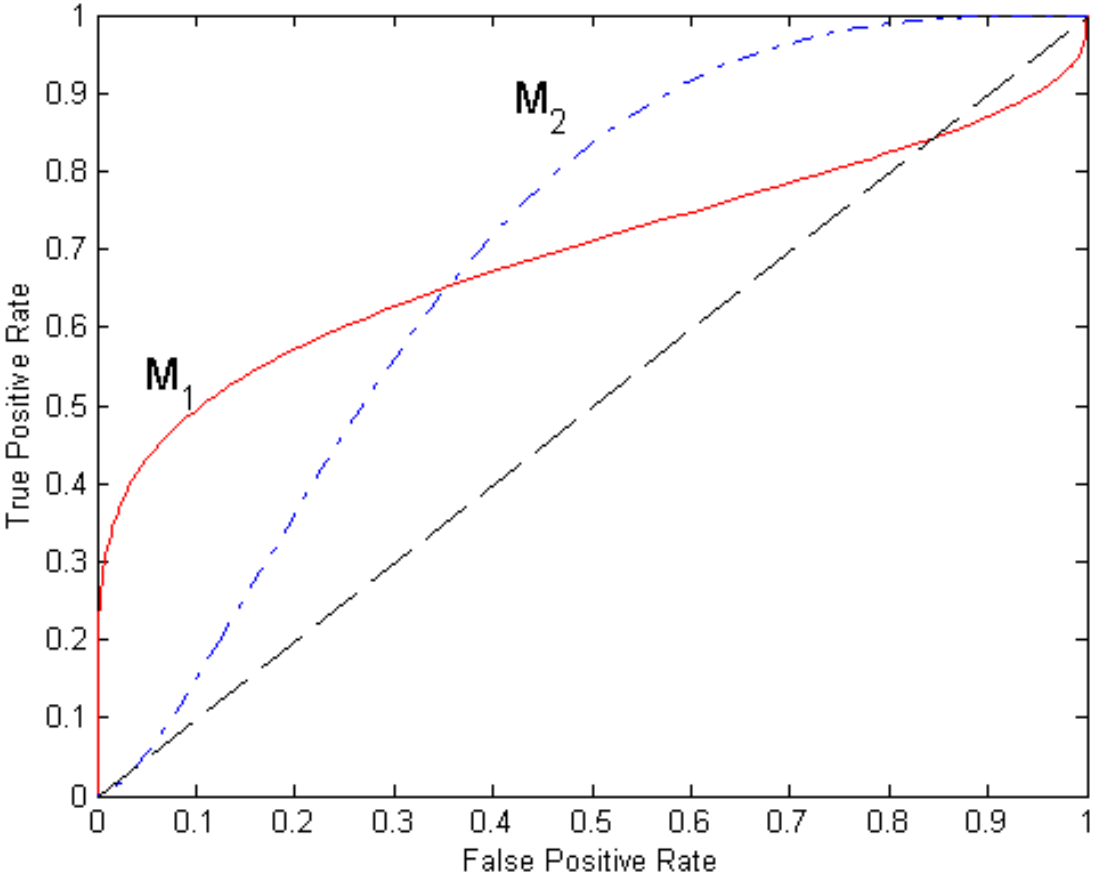
	Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=		0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
	TP	5	4	4	3	3	3	3	2	2	1	0
	FP	5	5	4	4	3	2	1	1	0	0	0
	TN	0	0	1	1	2	3	4	4	5	5	5
	FN	0	1	1	2	2	2	2	3	3	4	5
→	TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→	FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Using ROC for Model Comparison



- No model consistently outperforms the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve (AUC)
  - Ideal:
- Area = 1
  - Random guess:
- Area = 0.5

Dealing with Imbalanced Classes - Summary

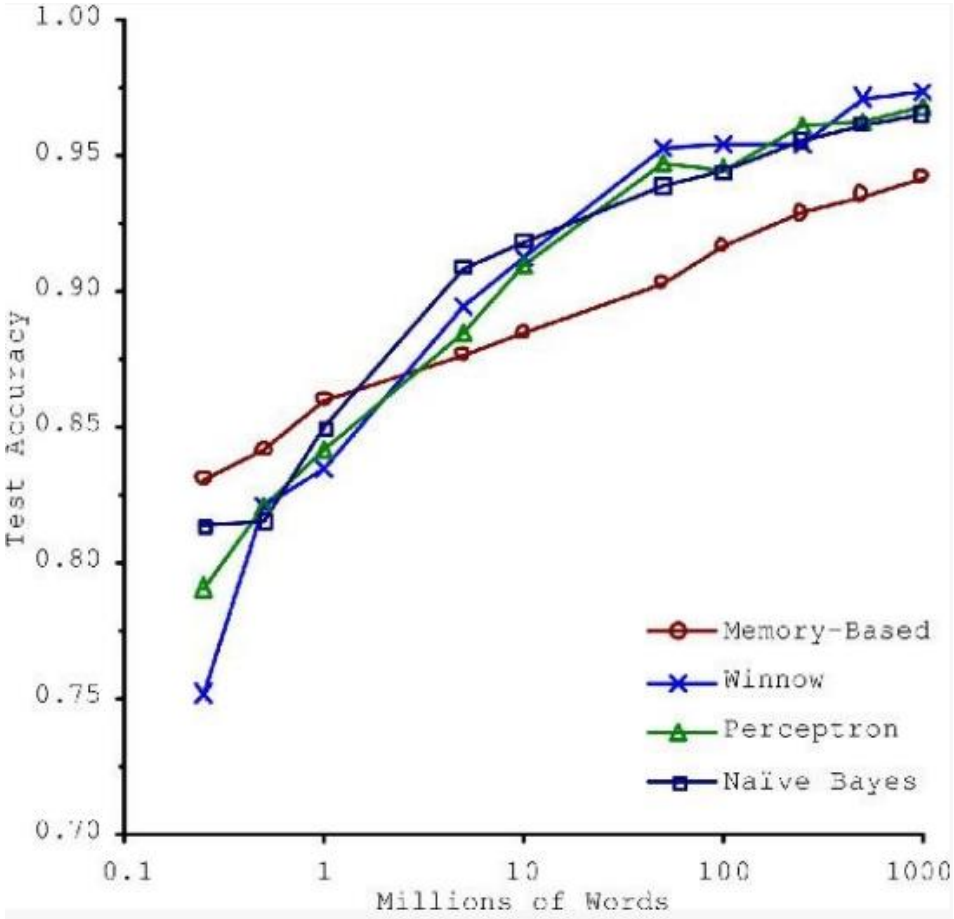
- Many measures exists, but none of them may be ideal in all situations
  - Random classifiers can have high value for many of these measures
  - TPR/FPR provides important information but may not be sufficient by itself in many practical scenarios
  - Given two classifiers, sometimes you can tell that one of them is strictly better than the other
    - C1 is strictly better than C2 if C1 has strictly better TPR and FPR relative to C2 (or same TPR and better FPR, and vice versa)
  - Even if C1 is strictly better than C2, C1's F-value can be worse than C2's if they are evaluated on data sets with different imbalances
  - Classifier C1 can be better or worse than C2 depending on the scenario at hand (class imbalance, importance of TP vs FP, cost/time tradeoffs)

Challenges of Machine Learning

- Training Data
  - Insufficient
  - Non representative
- Model Selection
  - Overfitting
  - Underfitting
- Validation and Testing

Insufficient Training Data

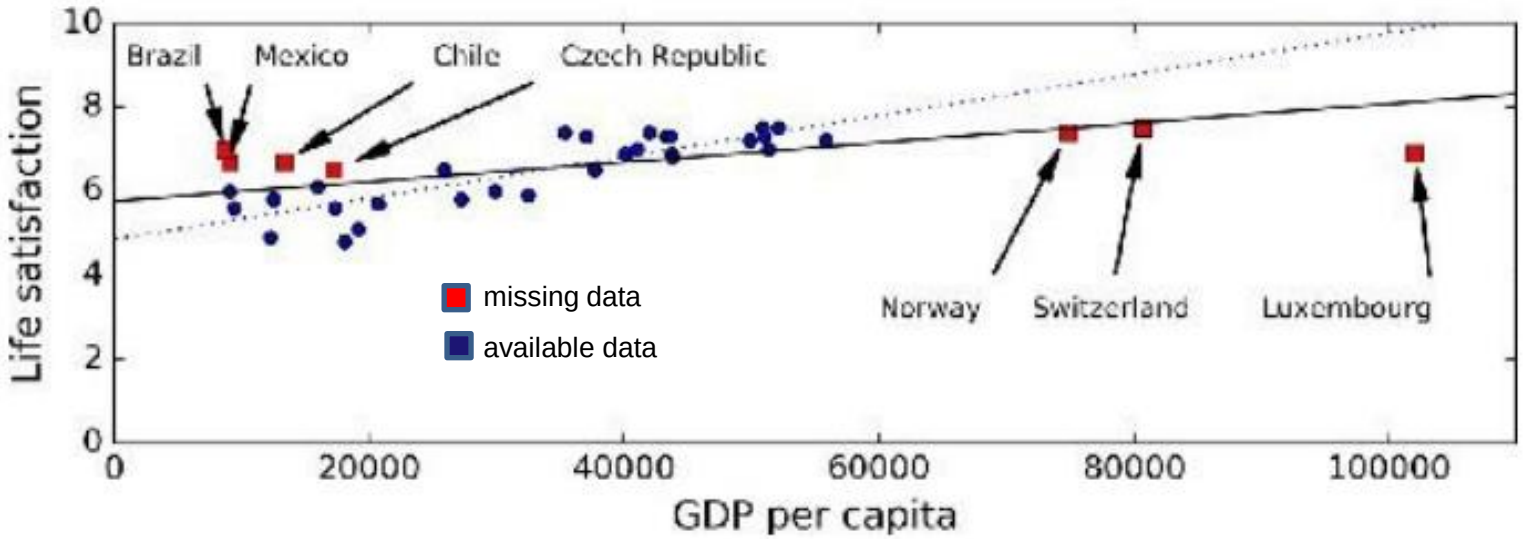
Consider trade-off Between Algorithm development & training data capture



Non-representative Training Data

Training Data be representative of the new cases we want to generalize

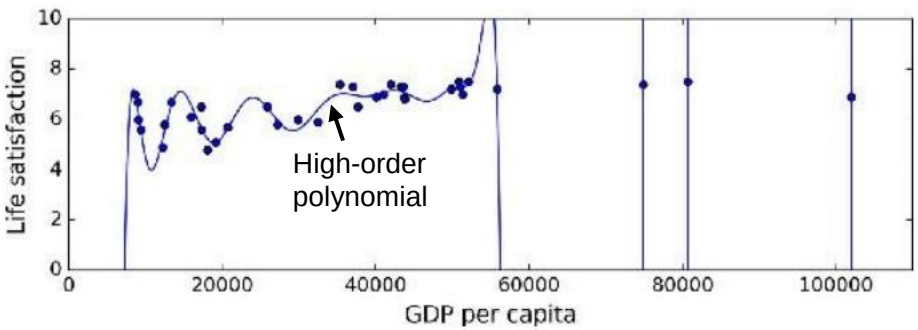
- Small sample size leads to sampling noise
  - Missing data over emphasizes the role of wealth on happiness
- If sampling process is flawed, even large sample size can lead to sampling bias

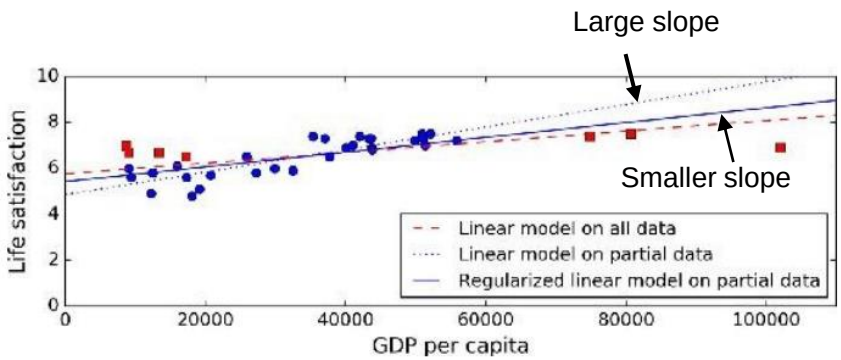


Model Selection

Overfitting or Underfitting

- Overfitting leads to high performance in training set but performs poorly on new data
  - e.g., a high-degree polynomial life satisfaction model that strongly overfits the training data
  - Small training set or sampling noise can lead to model following the noise than the underlying pattern in the dataset
  - Solution: Regularization to constrain the values of parameters





- Underfitting when the model is too simple to learn the underlying structure in the data
  - Select a more powerful model, with more parameters
  - Feed better features to the learning algorithm
  - Reduce regularization

Choice of Hyperparameters

Modern ML models often use a lot of model parameters

- Known as *hyperparameters*
- Model performance depends on choice of parameters
- Each parameter can assume a number of values
  - Real numbers or *categories*
- Exponential number of hyperparameter combinations possible
- Best model correspond to best cross validation performance over the set of hyperparameter combinations
- Expensive to perform
- Some empirical frameworks available for hyperparameter optimization
  - Grid search
  - Random search
  - Bayesian

Evaluating Predictive Performance of a Model

- Want to estimate the generalization performance, the predictive performance of our model on future (unseen) data.

- Want to increase the predictive performance by tweaking the learning algorithm and selecting the best performing model from a given hypothesis space.

- Want to identify the ML algorithm that is best-suited for the problem at hand; thus, we want to compare different algorithms, selecting the best-performing one as well as the best performing model from the algorithm's hypothesis space.

Evaluation and Validation

Performance of ML algorithms is statistical / predictive

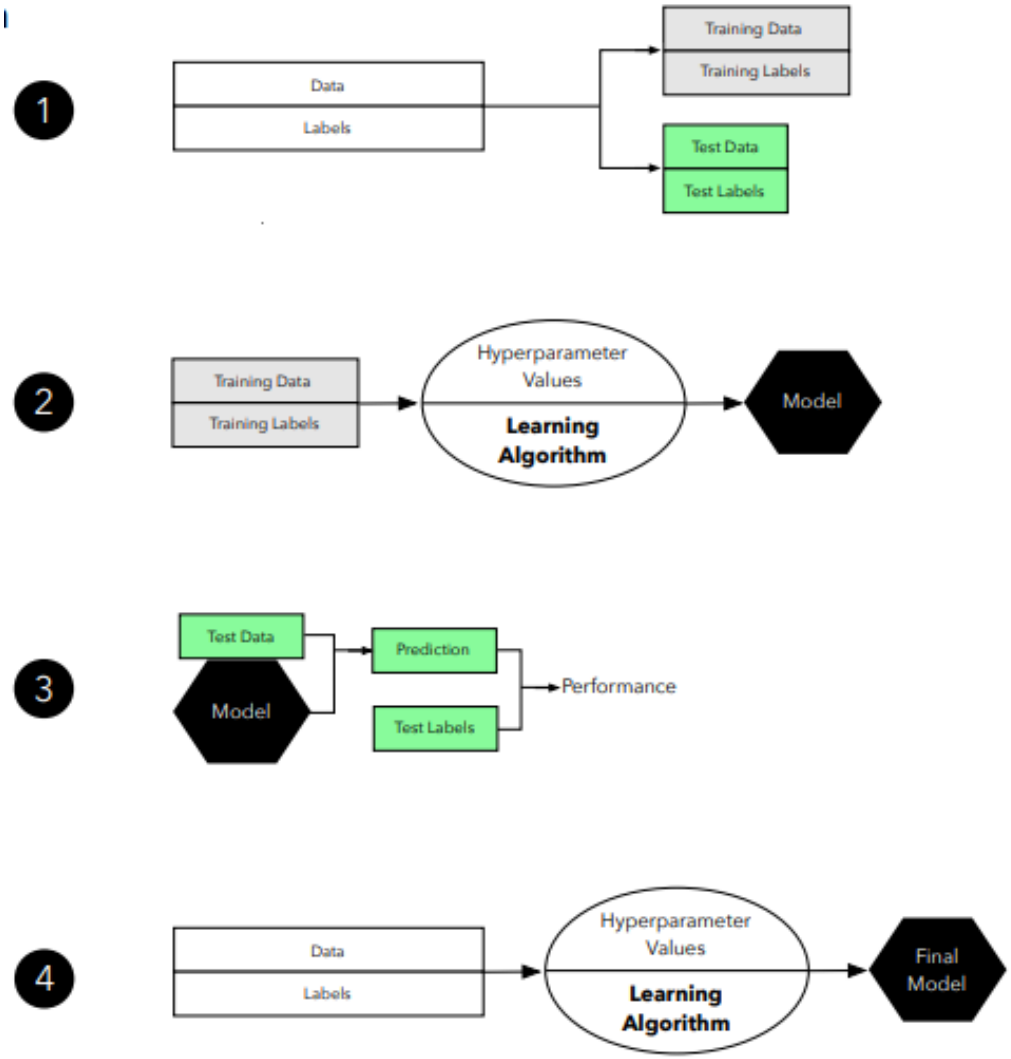
- Good ML algorithms need to work well on test data
  - But test data is often not accessible to the provider of the algorithm
- Common assumption is training data is representative of test data
- Randomly chosen subset of the training data is held out as validation set, *aka* dev set
- Once ML model is trained, its performance is evaluated on validation data
  - Expectation is ML model working well on validation set will work well on unknown test data
- Typically 20-30% of the data is randomly held out as validation data

Cross Validation

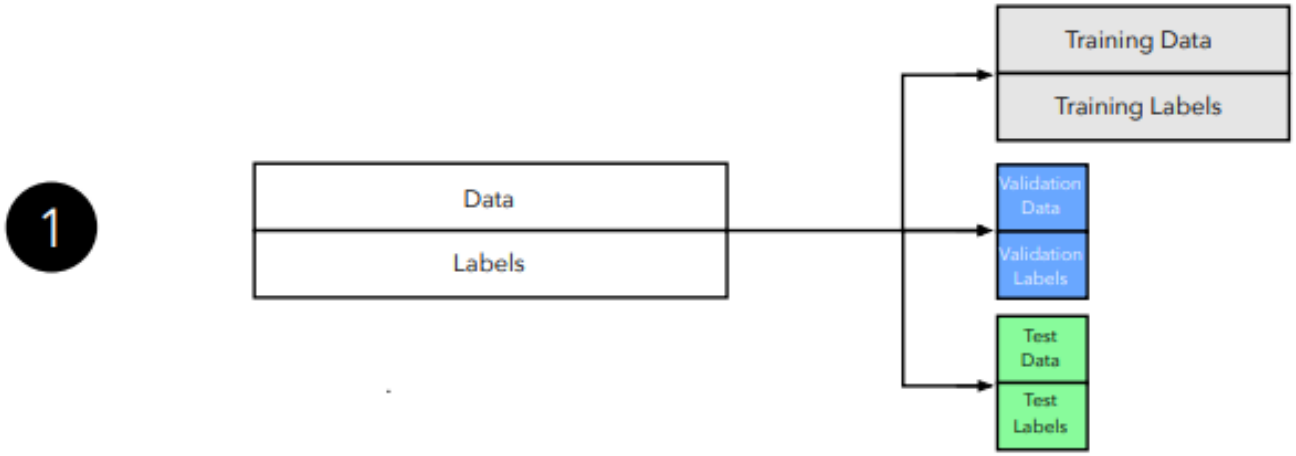
K-fold validation is often performed

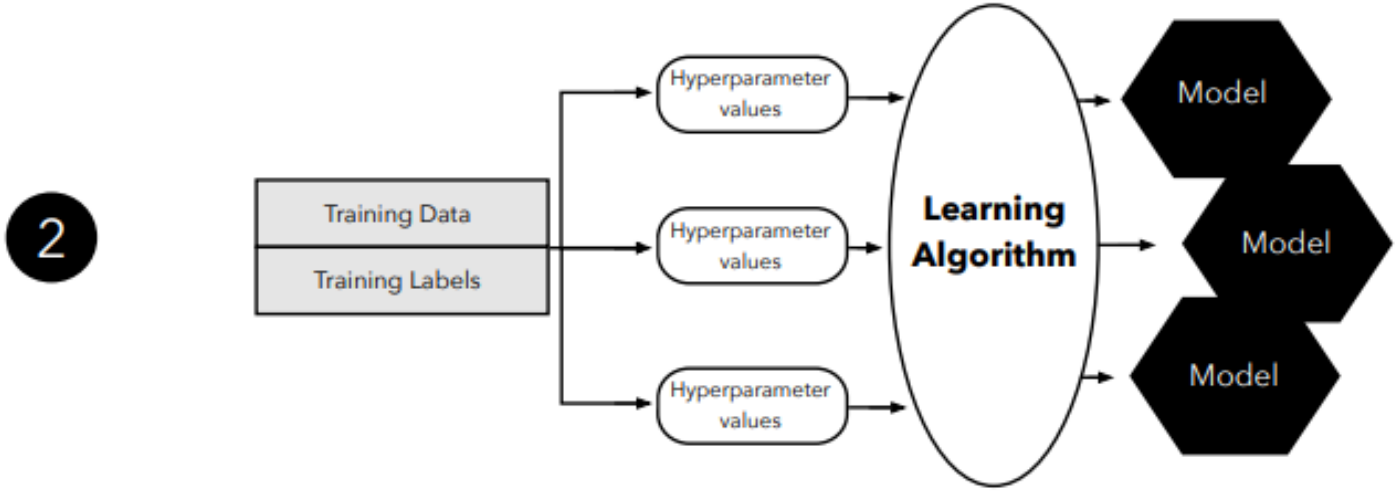
- To reduce the bias of validation set selection process
- Often K is chosen as 10
  - *aka* 10 fold cross validation
- 10 fold cross validation involves
  - randomly selecting the validation set 10 times
  - model generation with 10 resulting training set
  - Evaluate the performance of each on that validation set
  - averaging the performance over the validation sets

Holdout for Model Evaluation



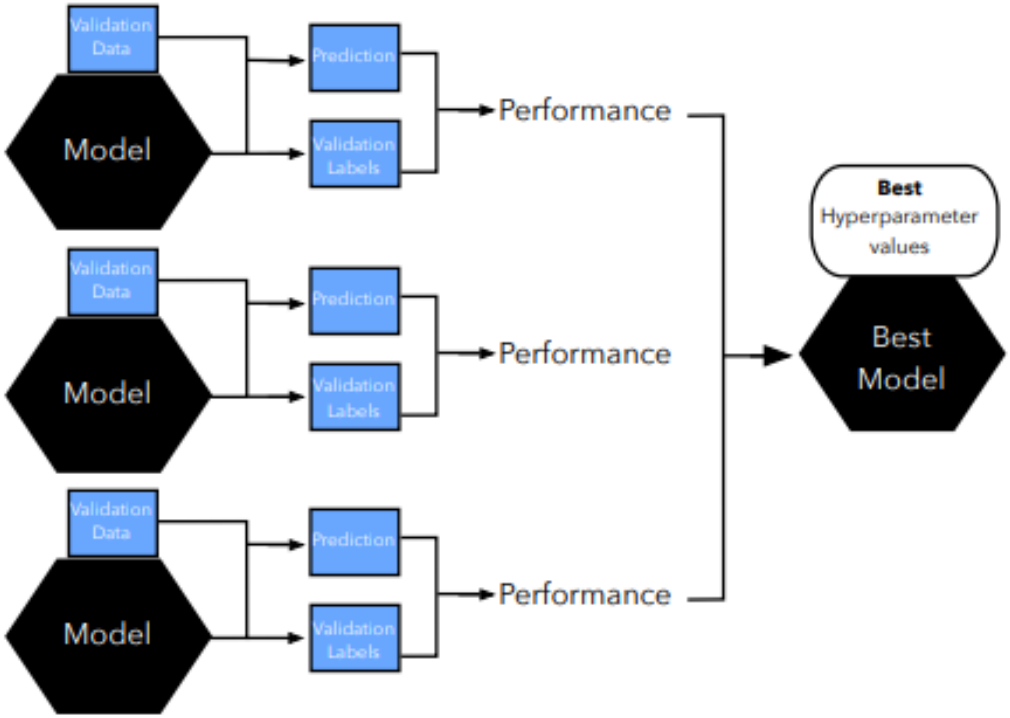
Holdout for Model Selection (1)



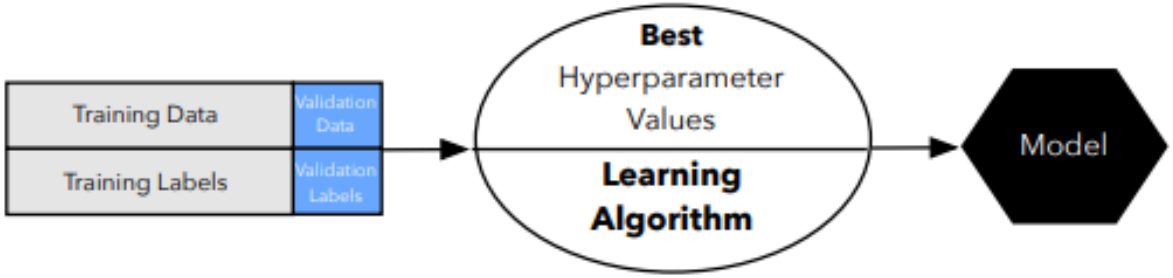


Holdout for Model Selection (2)

3

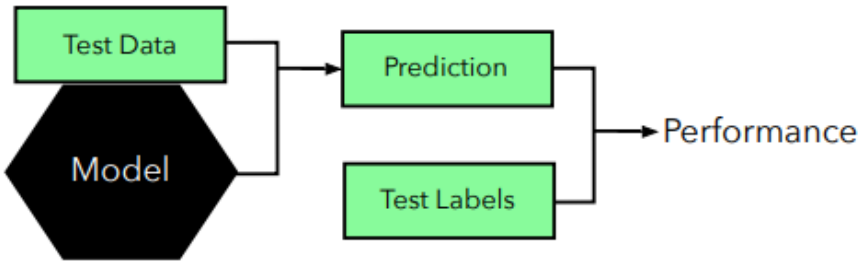


4

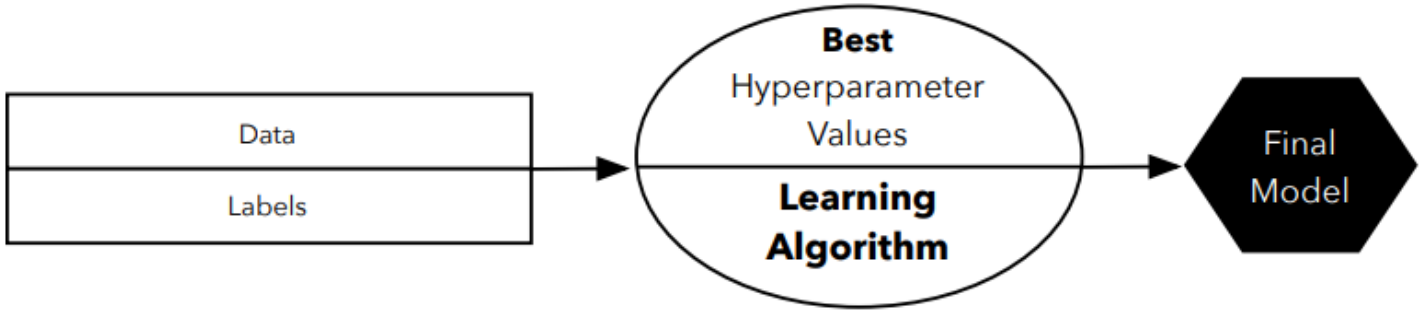


Holdout for Model Selection (3)

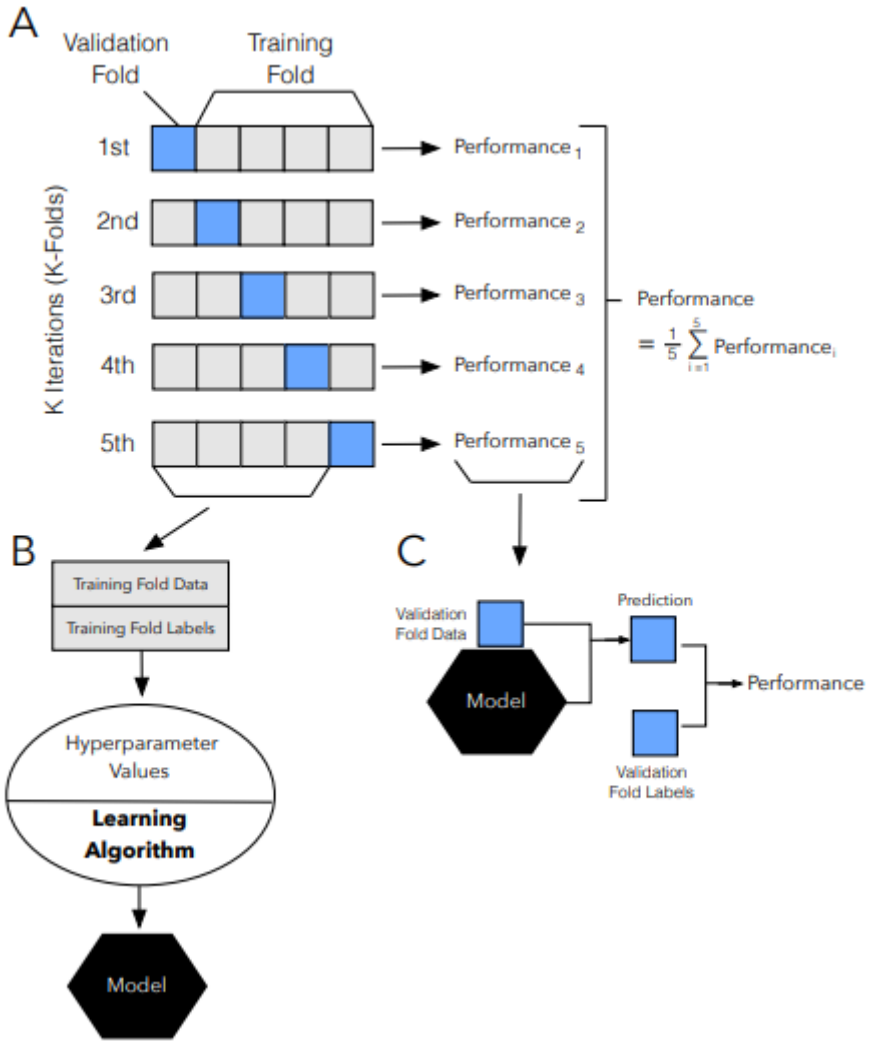
5



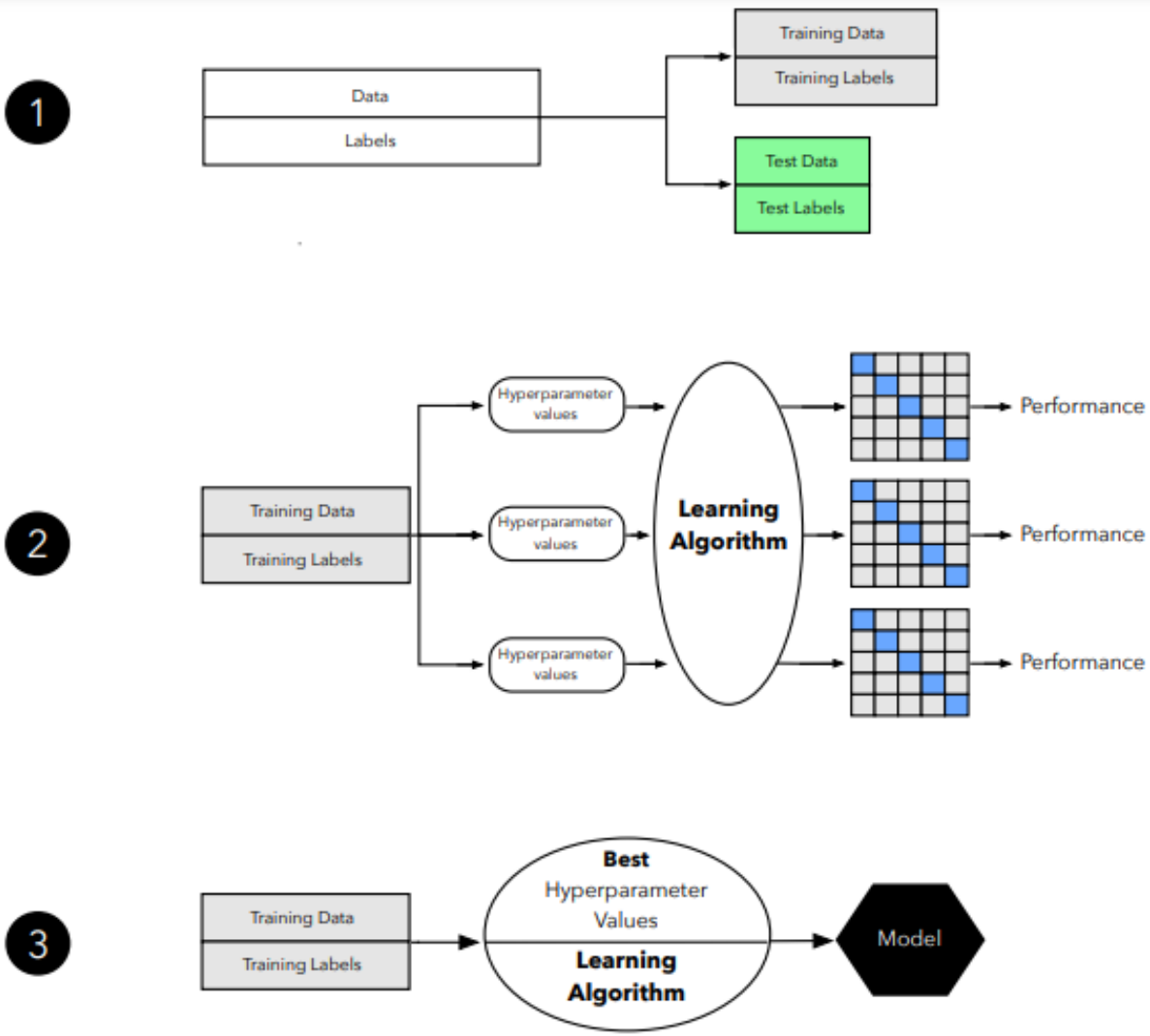
6



K Fold Cross Validation (CV) for Model Evaluation

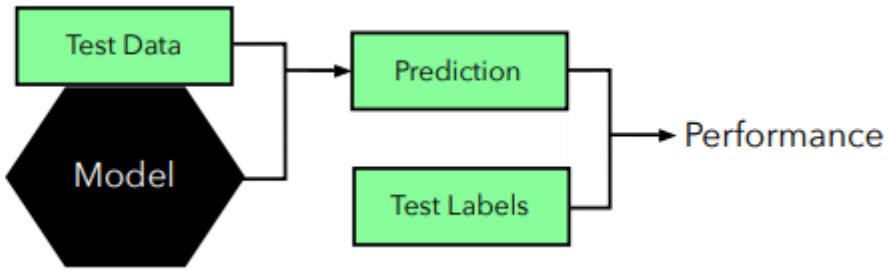


CV for Model Selection (1)

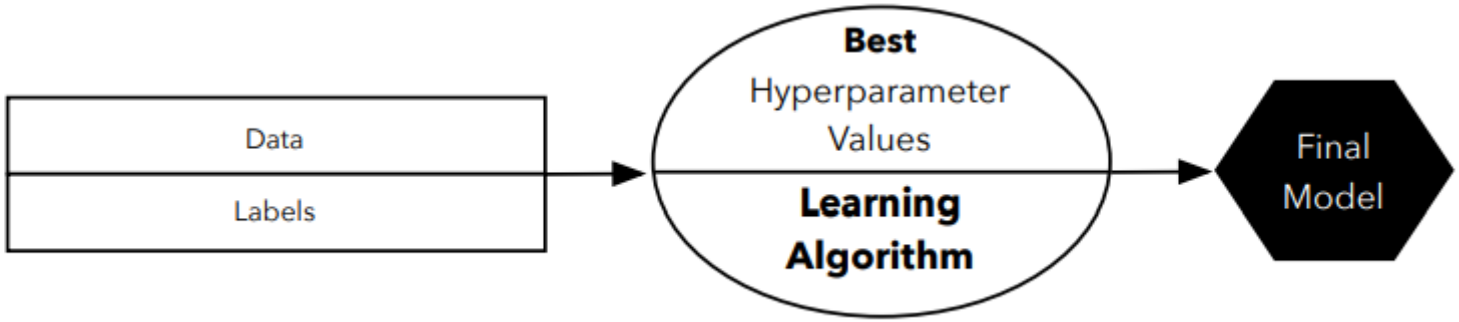


CV for Model Selection (2)

4



5



END
of
Session 2

Machine Learning

AIML CLZG565

M3 : Linear Models for Regression

Dr. Sugata Ghosal
sugata.ghosal@pilani.bits-pilani.ac.in

BITS Pilani
Pilani Campus

Linear Regression

Inductive Learning Hypothesis : Interpretation

- Target Concept
- Discrete : f(x) ∈ {Yes, No, Maybe} Classification
- Continuous : f(x) ∈ [20-100] Regression
- Probability Estimation : f(x) ∈ [0-1]

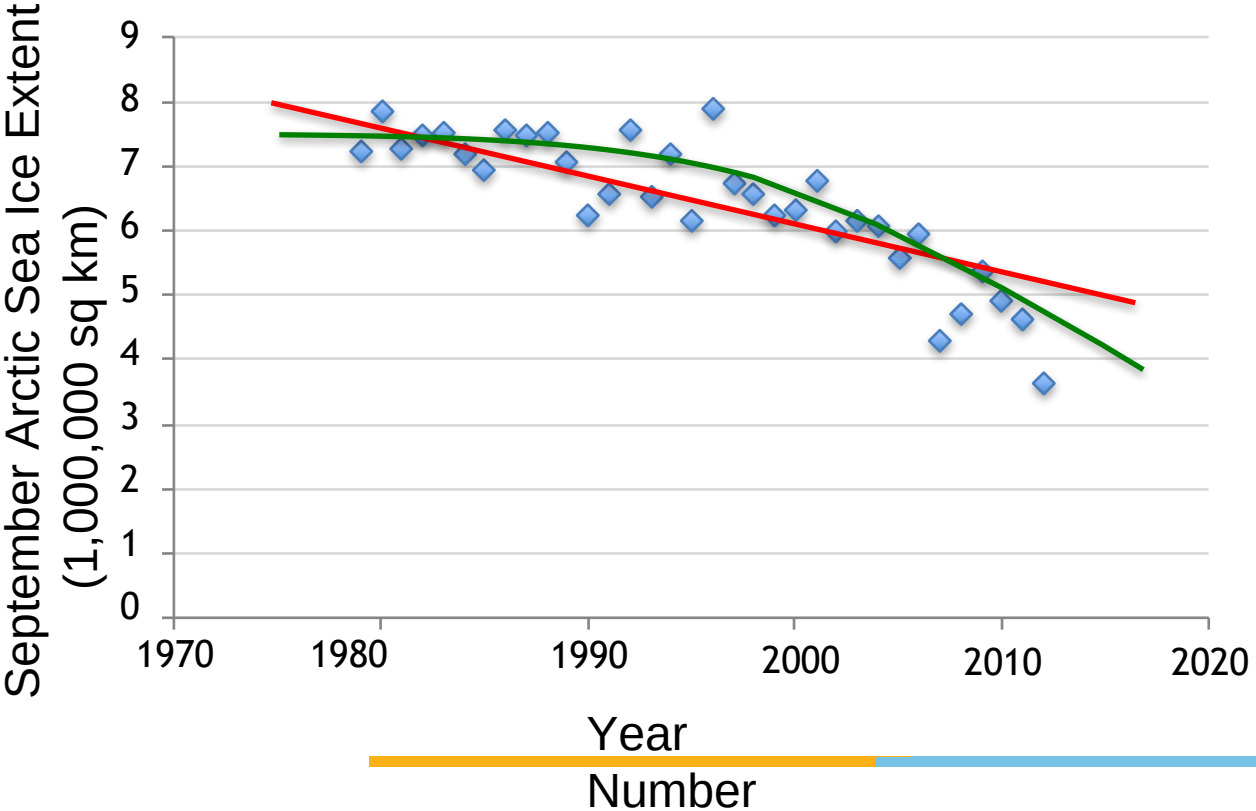
Sky	AirTemp	Altitude	Wind	Water	Forecast	Humidity
Sunny	Warm	Normal	Strong	Warm	Same	60
Sunny	Warm	High	Strong	Warm	Same	75
Rainy	Cold	High	Strong	Warm	Change	70
Sunny	Warm	High	Strong	Cool	Change	45

Formal Representation: Interpretation

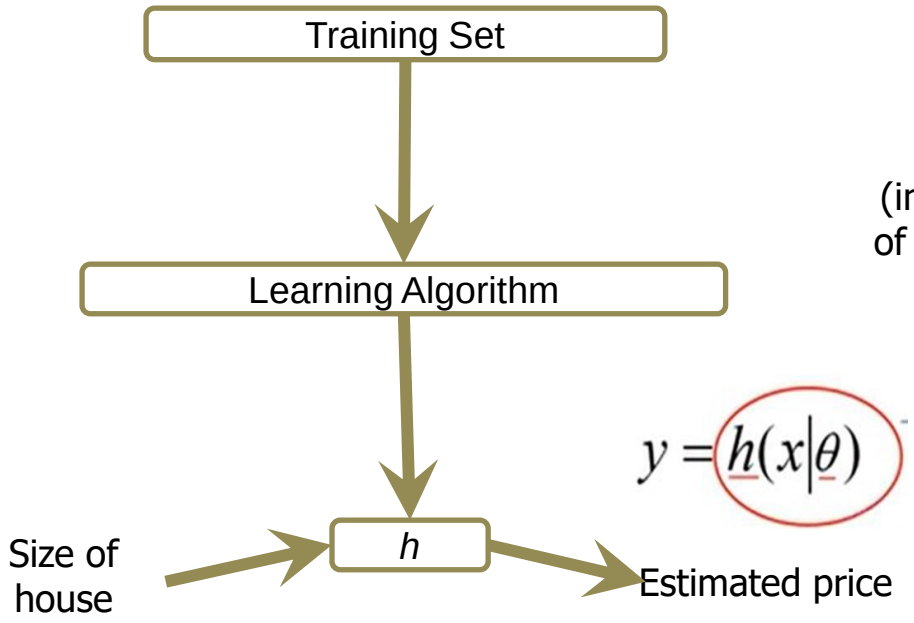
Supervised Learning: Regression

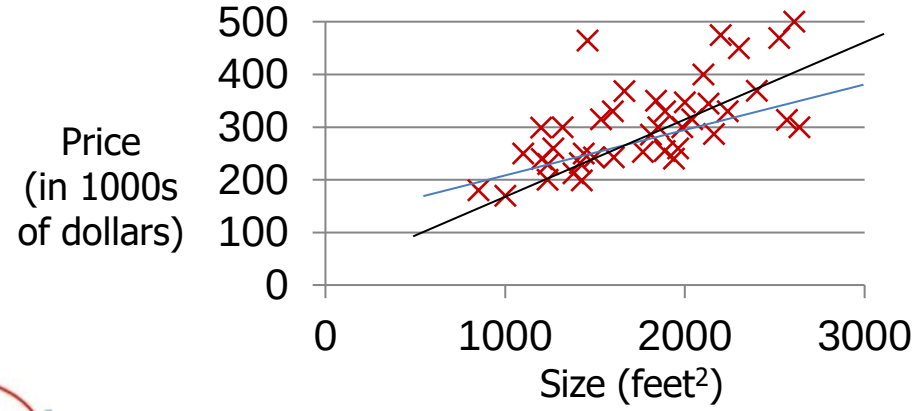
GOAL : Previously unseen records should be assigned a value as accurately as possible.

- 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



Machine Learning Process : Interpretation





Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

θ_i's : Parameters

h(.) : Model

How to choose θ_i's ?

Types of Regression Models

simple regression model

(Education) x ⟶ y (Income)

multiple regression model

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

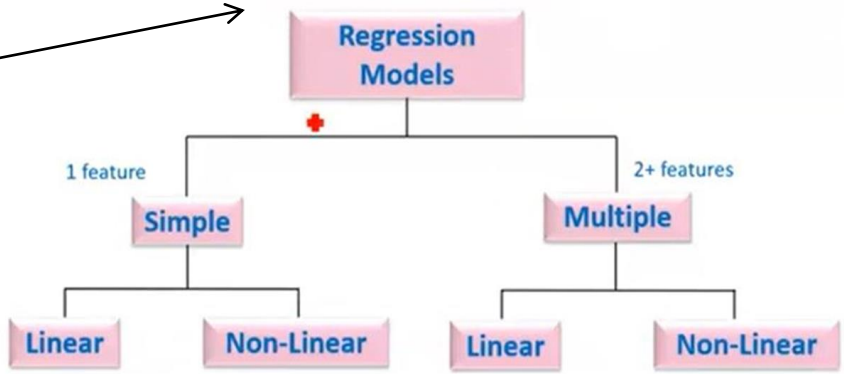
(Education) x_1

(Soft Skills) x_2

(Experience) x_3

(Age) x_4

y (Income)



- Hypothesis:

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \sum_{j=0}^{d} \theta_j x_j$$

Assume $x_0 = 1$

Learning the Model Parameters

Closed Form Solution Approach

Gradient Descent Approach

How to choose θ_i's ?

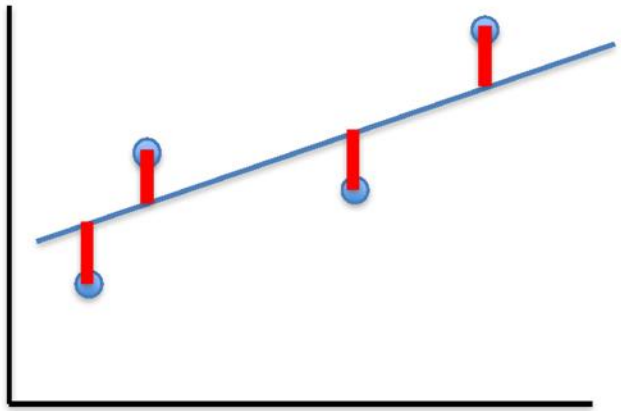
Linear Regression : Least Squares

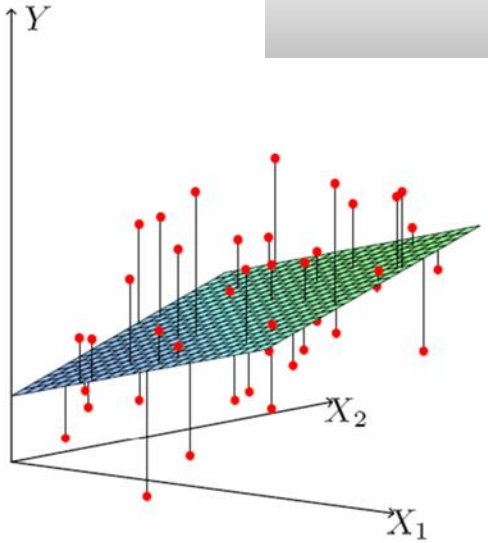
- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Hypothesis $h_\theta(x) = \theta_0 + \theta_1 x$
- Parameters θ_0 and θ_1
- Cost Function $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^{(i)})^2$
- Goal $\underset{\theta_0,\theta_1}{\text{minimize}}\ J(\theta_0,\theta_1)$





Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

Source Credit : Based on example by Andrew Ng

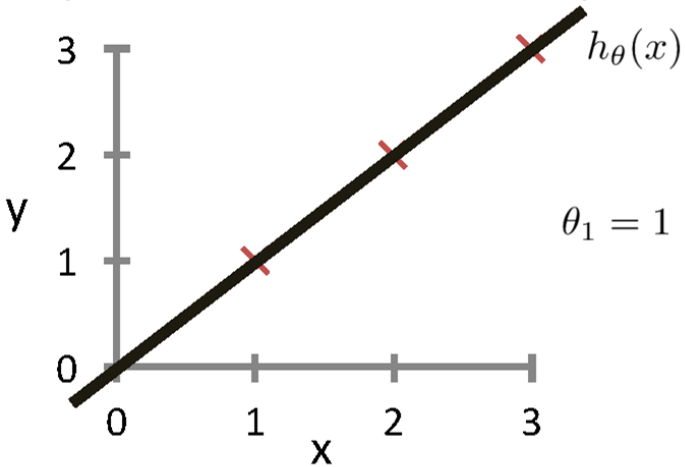
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

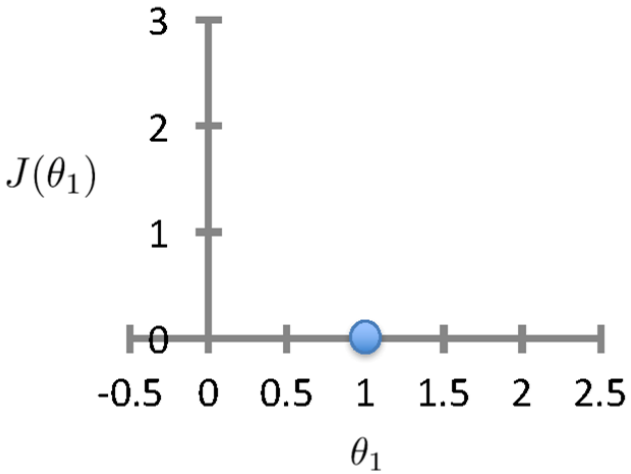
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



Source Credit : Based on example by Andrew Ng

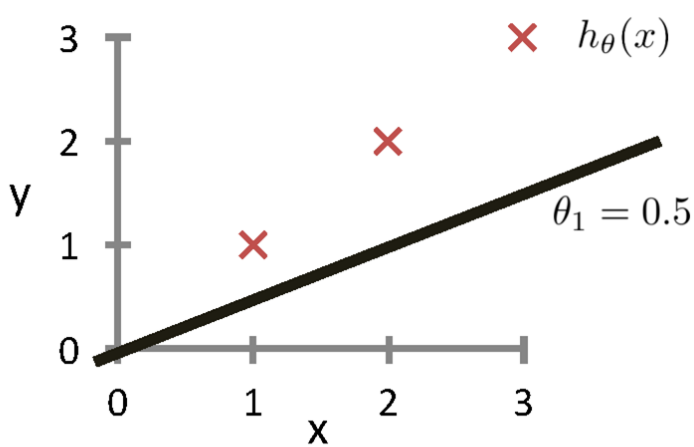
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

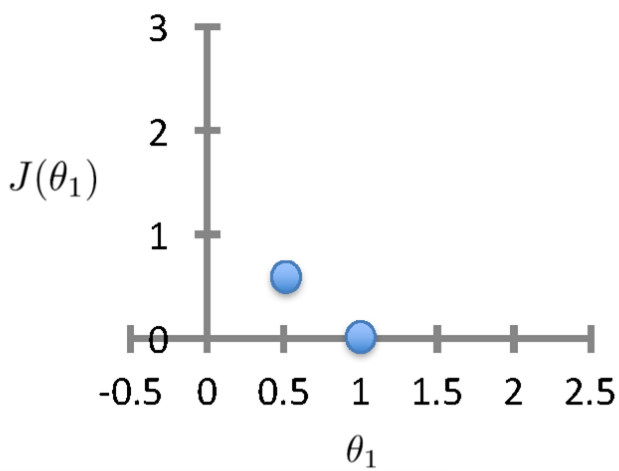
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



Based on example by Andrew Ng

$$J([0, 0.5]) = \frac{1}{2 \times 3}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right] \approx 0.58$$

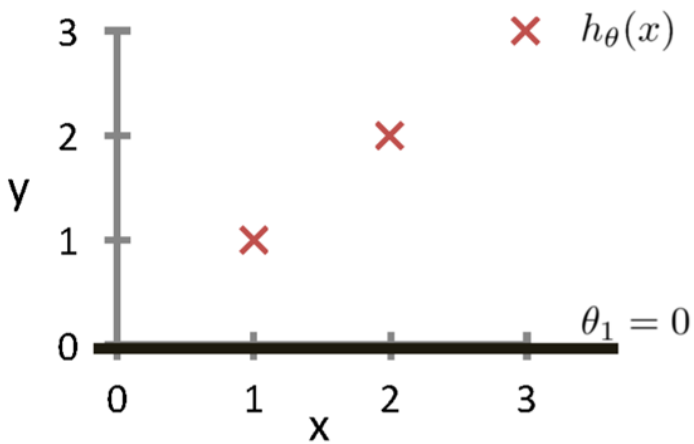
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

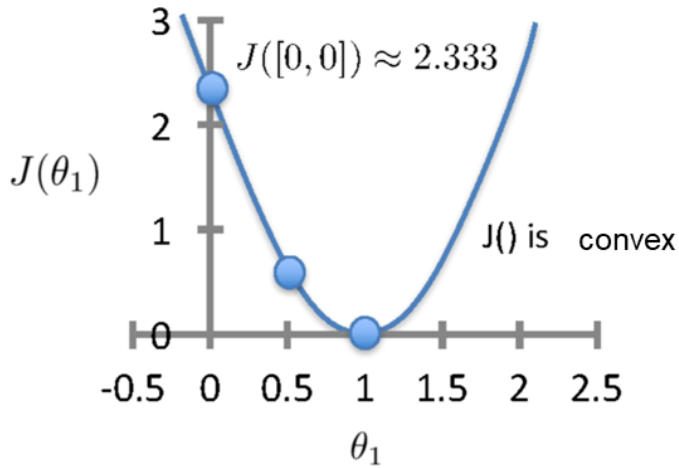
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

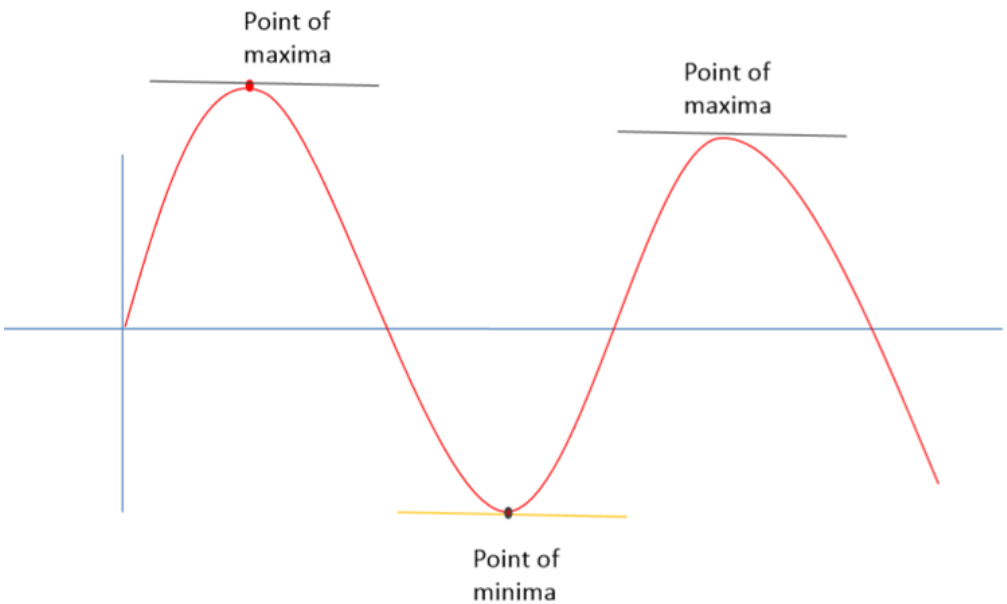
(function of the parameter θ_1)



Source Credit : Based on example by Andrew Ng

Maxima and Minima

- For maxima and minima $m=dy/dx=\tan 0^{0}=0$
- $dy/dx = 0$ means tangent is parralel to X –axis.



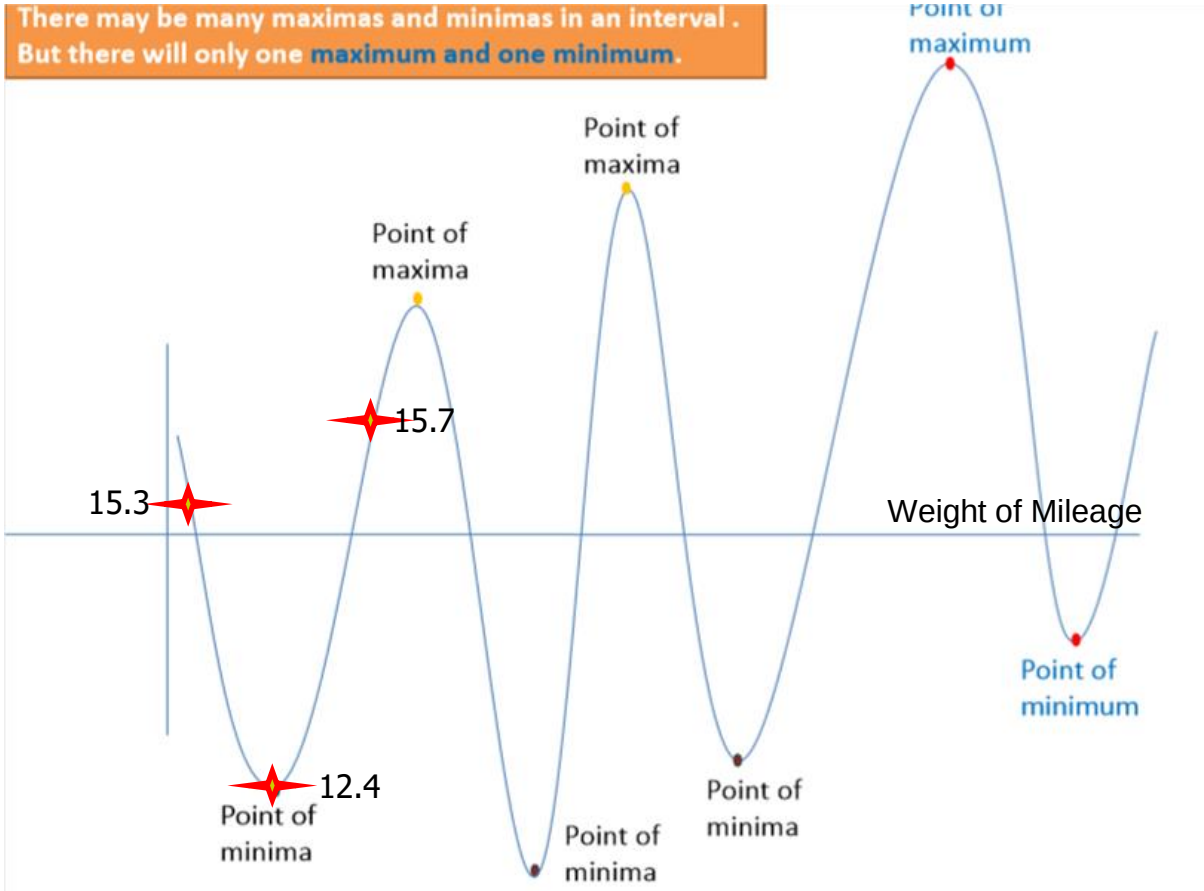
Notion of Maxima and Minima of a function

In Machine Learning

CarPrice = 1 + 0.05Mileage

CarPrice = 1 + 0.25Mileage

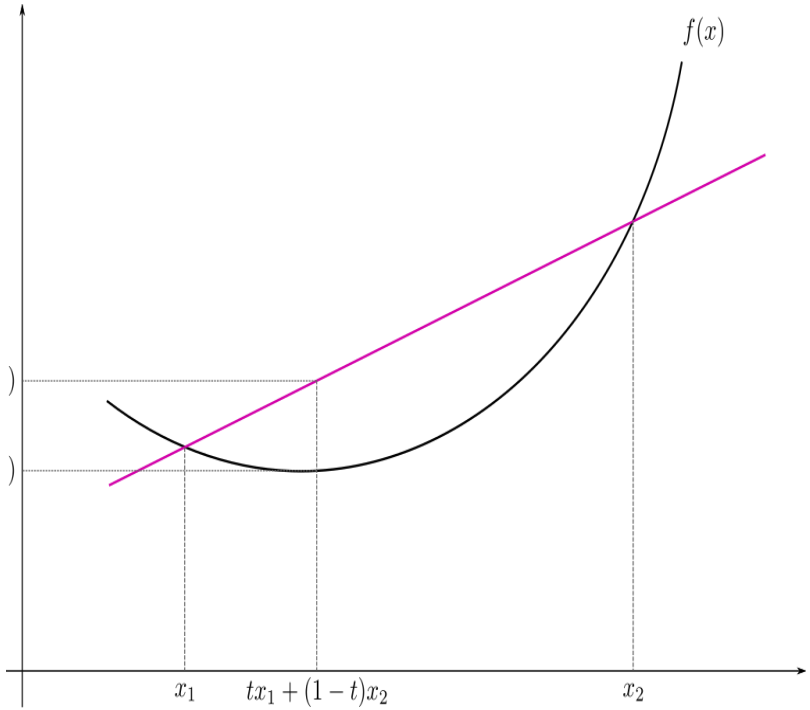
CarPrice = 1 + 0.75Mileage

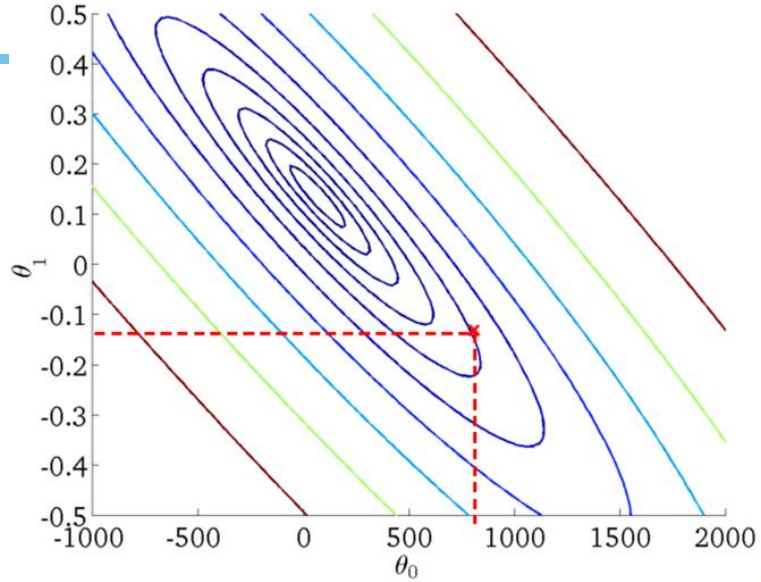


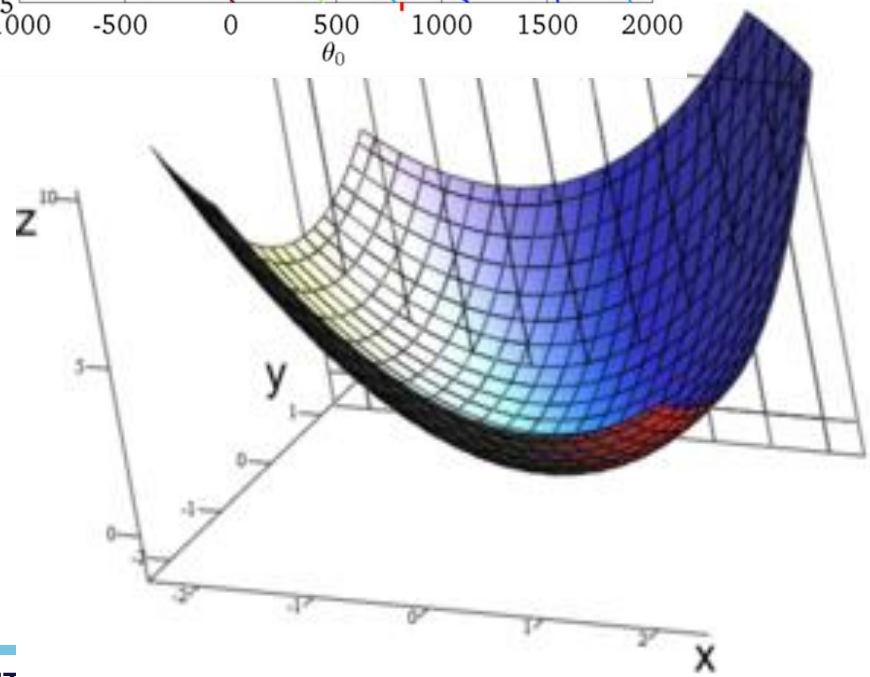
Mileage (in kmpl)	*Car Price (in cr)*
9.8	10.48
9.12	1.75
9.5	6.95
10	2.51

Sum of absolute Error / Training Error

Convex Function : Multivariate







Real-valued function defined on an *n*-dimensional interval is called **convex** if the line segment between any two points on the graph of the function lies above or on the graph

Closed Form Solution

Vectorization

X No.of.Years of Experience (in Years)	Y Salary Of the Employee (in Lakhs)
1	2
2	3
3	4
4	5
5	6

- Benefits of vectorization
  - More compact equations
  - Faster code (using optimized matrix libraries)
- Consider our model:

$$h(\boldsymbol{x}) = \sum_{j=0}^{d} \theta_j x_j$$

- Let

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{x}^\intercal = \begin{bmatrix} 1 & x_1 & \dots & x_d \end{bmatrix}$$

- Can write the model in vectorized form as $h(\boldsymbol{x}) = \boldsymbol{\theta}^\intercal \boldsymbol{x}$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \quad x^T = \begin{bmatrix} 1 & No.of.Years\ Experience \end{bmatrix}$$

Vectorization

X No.of.Years of Experience (in Years)	Y Salary Of the Employee (in Lakhs)
1	2
2	3
3	4
4	5
5	6

- Consider our model for n instances:

$$h\left(\boldsymbol{x}^{(i)}\right) = \sum_{j=0}^{d} \theta_j x_j^{(i)}$$

- Let

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(i)} & \dots & x_d^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & \dots & x_d^{(n)} \end{bmatrix}$$

$\mathbb{R}^{(d+1)\times 1}$ $\qquad$ $\mathbb{R}^{n\times(d+1)}$

Let:

$$\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

- Can write the model in vectorized form as $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{X}\boldsymbol{\theta}$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \quad X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix} \quad y = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

Vectorization

X No.of.Years of Experience (in Years)	Y Salary Of the Employee (in Lakhs)
1	2
2	3
3	4
4	5
5	6

- For the linear regression cost function:

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

$$= \frac{1}{2n}\sum_{i=1}^{n}\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}^{(i)} - y^{(i)}\right)^2$$

$$= \frac{1}{2n}\underbrace{(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^{\mathsf{T}}}_{\mathbb{R}^{1\times n}}\underbrace{(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})}_{\mathbb{R}^{n\times 1}}$$

$\mathbb{R}^{n\times(d+1)}$

$\mathbb{R}^{(d+1)\times 1}$

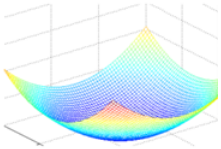
Let:

$$\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

$$\theta^T = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix} \quad X = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \quad y = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

Vectorization

- Instead of using GD, solve for optimal $\boldsymbol{\theta}$ analytically
  - Notice that the solution is when $\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = 0$



- Derivation:

$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{2n} (\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^\top (\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})$$

1 x 1

$$\propto \boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{\theta} - \boxed{\boldsymbol{y}^\top \boldsymbol{X}\boldsymbol{\theta}} - \boxed{\boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{y}} + \boldsymbol{y}^\top \boldsymbol{y}$$

$$\propto \boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{y} + \boldsymbol{y}^\top \boldsymbol{y}$$

Take derivative and set equal to 0, then solve for $\boldsymbol{\theta}$:

$$\frac{\partial}{\partial \boldsymbol{\theta}} (\boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \boldsymbol{X}^\top \boldsymbol{y} + \cancel{\boldsymbol{y}^\top \boldsymbol{y}}) = 0$$

$$(\boldsymbol{X}^\top \boldsymbol{X})\boldsymbol{\theta} - \boldsymbol{X}^\top \boldsymbol{y} = 0$$

$$(\boldsymbol{X}^\top \boldsymbol{X})\boldsymbol{\theta} = \boldsymbol{X}^\top \boldsymbol{y}$$

Closed Form Solution:	$\boldsymbol{\theta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$

Vectorization

- Can obtain $\boldsymbol{\theta}$ by simply plugging $\boldsymbol{X}$ and $\boldsymbol{y}$ into

$$\boldsymbol{\theta} = (\boldsymbol{X}^\intercal \boldsymbol{X})^{-1} \boldsymbol{X}^\intercal \boldsymbol{y}$$

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(i)} & \dots & x_d^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & \dots & x_d^{(n)} \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix}$$

- If $\boldsymbol{X}^\intercal \boldsymbol{X}$ is not invertible (i.e., singular), may need to:
  - Use pseudo-inverse instead of the inverse
    - In python, `numpy.linalg.pinv(a)`
  - Remove redundant (not linearly independent) features
  - Remove extra features to ensure that $d \leq n$

Fit the Linear Regression Model :

Using Closed Form

X No.of.Years of Experience (in Years)	Y Salary Of the Employee (in Lakhs)
1	2
2	3
3	4
4	5
5	6

$$\left(X^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} * X \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix} \right)^{-1} * X^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} * y \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

$$\left(\begin{bmatrix} 5 & 15 \\ 15 & 55 \end{bmatrix} \right)^{-1} * X^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} * y \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

$$\left(\begin{bmatrix} 1.1 & -0.3 \\ -0.3 & 0.1 \end{bmatrix} \right) * X^T \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} * y \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

$$\begin{bmatrix} 0.8 & 0.5 & 0.2 & -0.1 & -0.4 \\ -0.2 & -0.1 & 0 & 0.1 & -0.2 \end{bmatrix} * y \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

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

Can obtain $\boldsymbol{\theta}$ by simply plugging $\boldsymbol{X}$ and $\boldsymbol{y}$ into

$$\boldsymbol{\theta} = (\boldsymbol{X}^\intercal \boldsymbol{X})^{-1} \boldsymbol{X}^\intercal \boldsymbol{y}$$

$$\theta^T = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix} \quad X = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} \quad y = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix}$$

Gradient Descent Approach

BITS Pilani

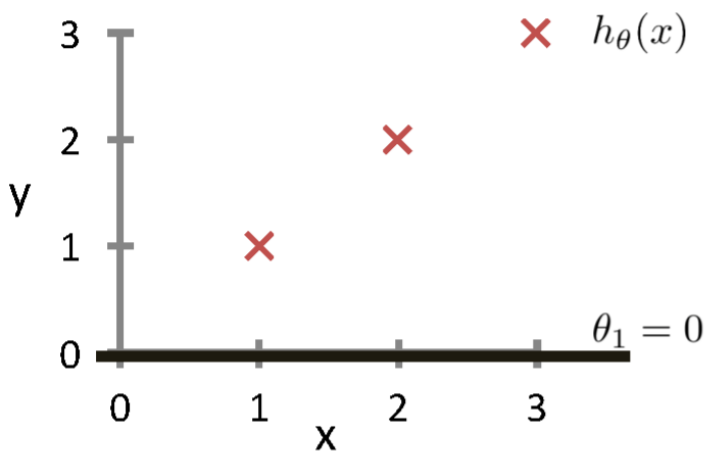
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

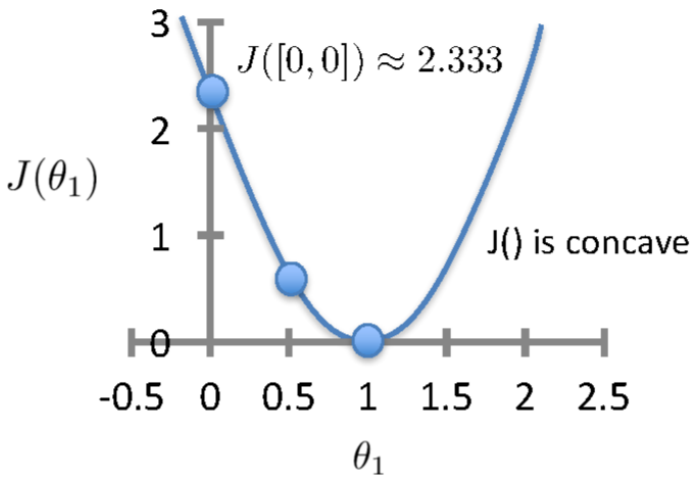
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



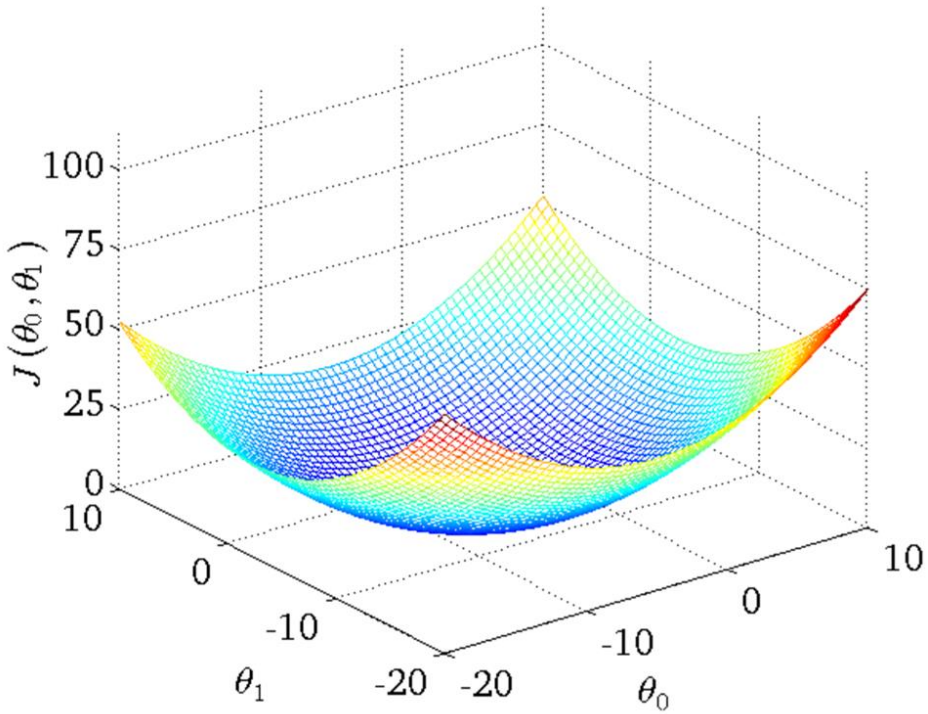
$J(\theta_1)$

(function of the parameter θ_1)



Source Credit : Based on example by Andrew Ng

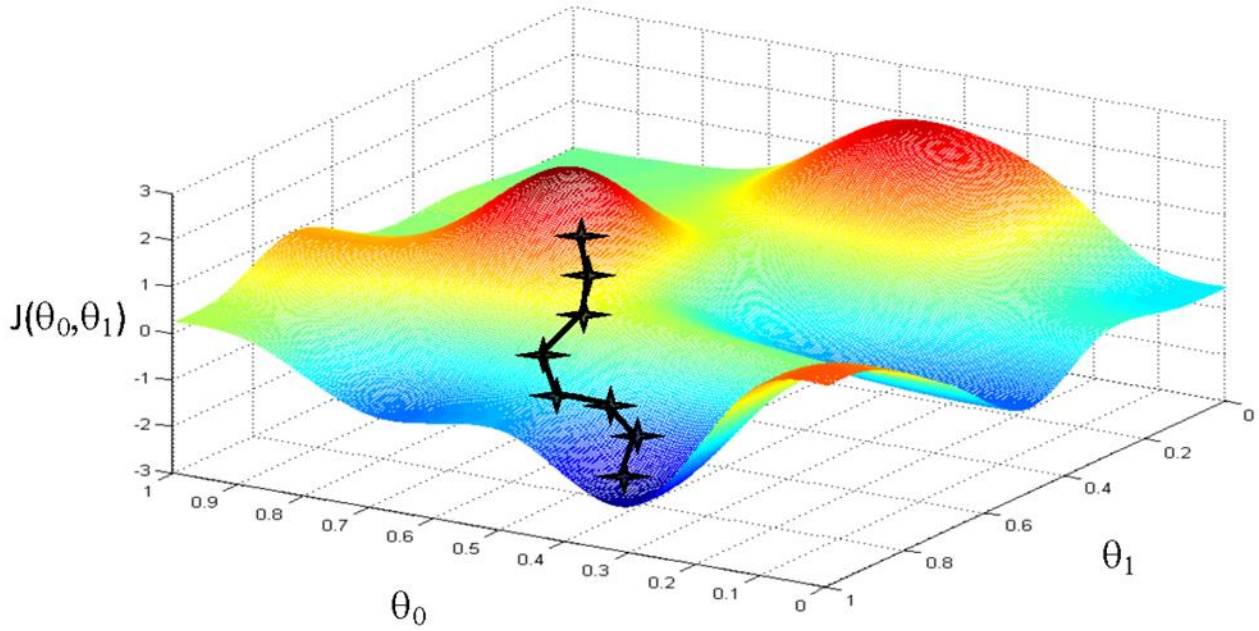
Intuition Behind Cost Function



Source Credit : Based on example by Andrew Ng

Basic Search Procedure

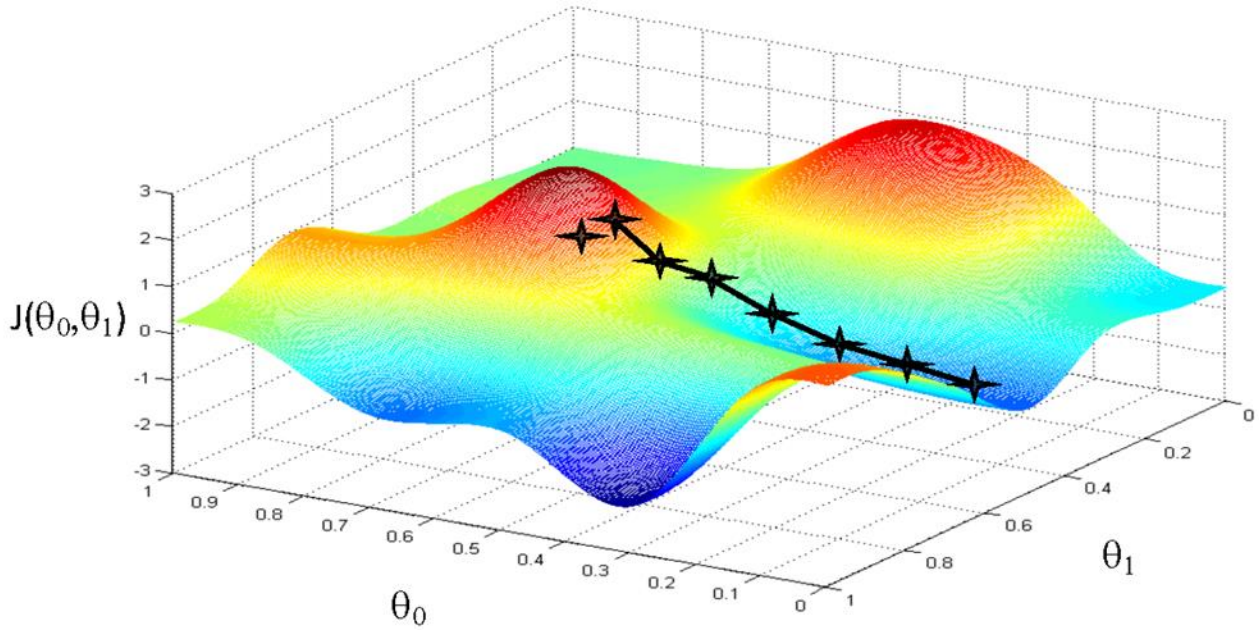
- Choose initial value for $\boldsymbol{\theta}$
- Until we reach a minimum:
  - Choose a new value for $\boldsymbol{\theta}$ to reduce $J(\boldsymbol{\theta})$



Source Credit : Figure by Andrew Ng

Basic Search Procedure

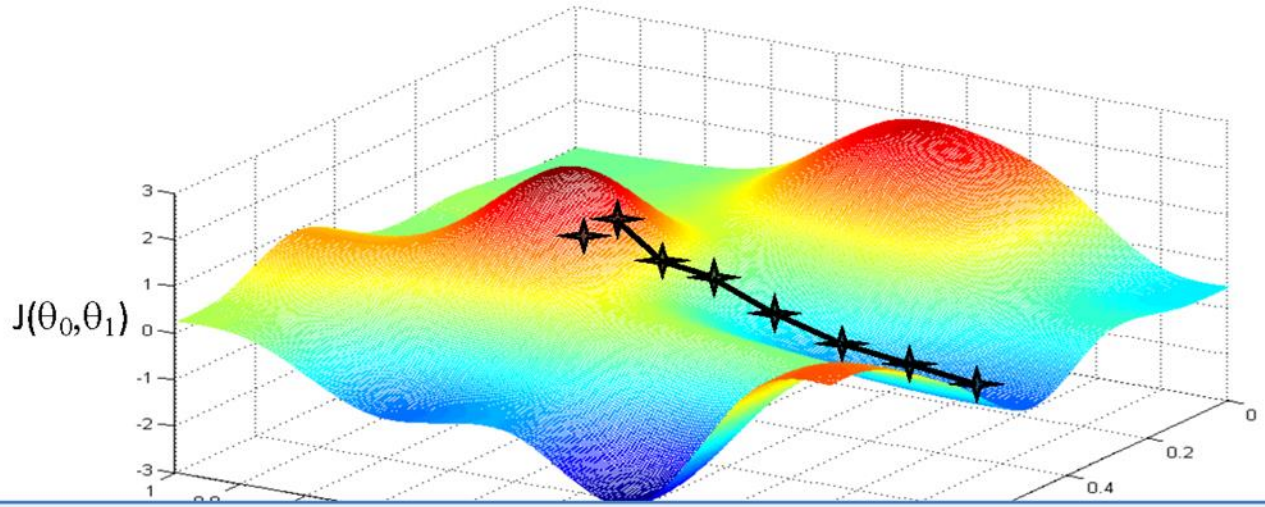
- Choose initial value for $\boldsymbol{\theta}$
- Until we reach a minimum:
  - Choose a new value for $\boldsymbol{\theta}$ to reduce $J(\boldsymbol{\theta})$



Source Credit : Figure by Andrew Ng

Basic Search Procedure

- Choose initial value for $\boldsymbol{\theta}$
- Until we reach a minimum:
  - Choose a new value for $\boldsymbol{\theta}$ to reduce $J(\boldsymbol{\theta})$



Since the least squares objective function is convex (concave), we don't need to worry about local minima

Source Credit : Figure by Andrew Ng

Gradient Descent

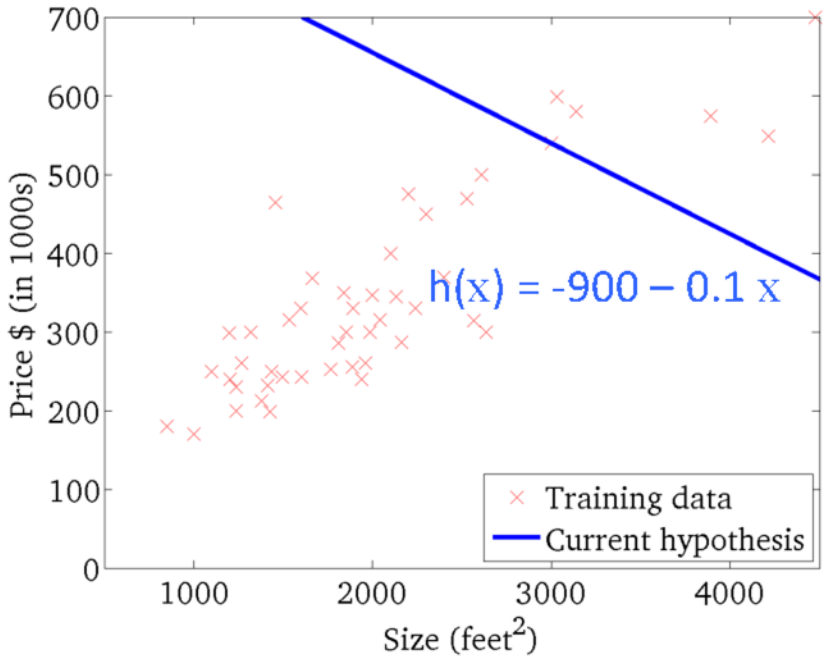
Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

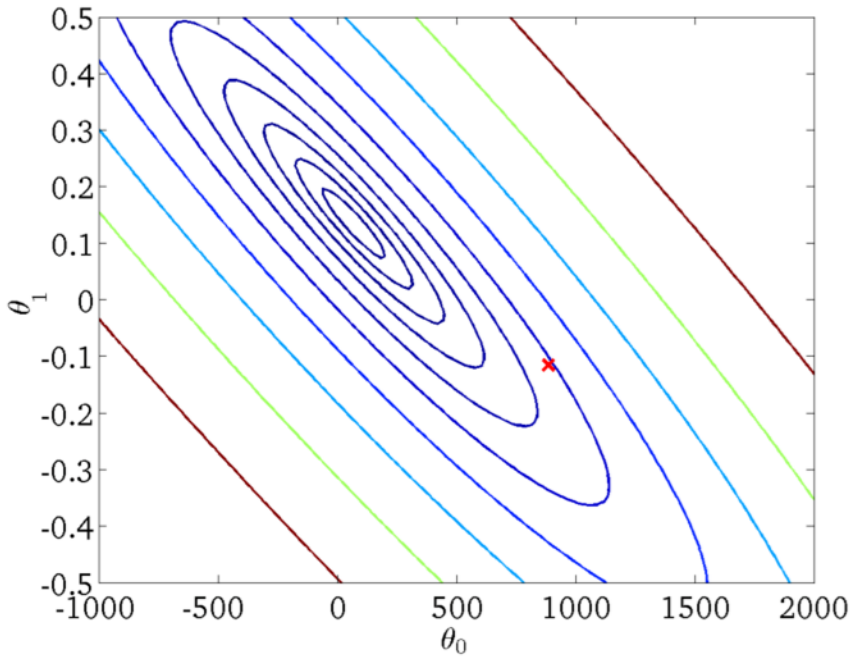
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



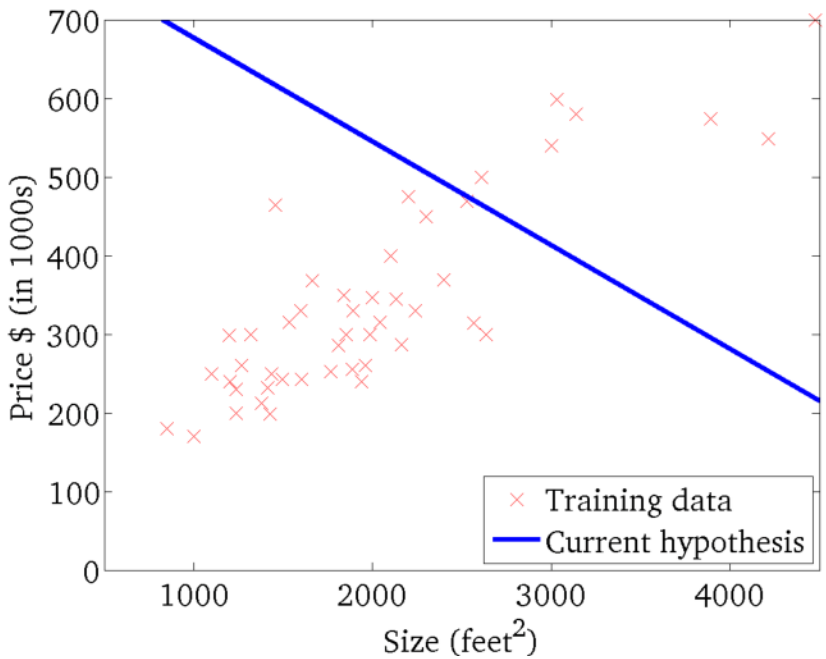


Source Credit : Slide by Andrew Ng

Gradient Descent

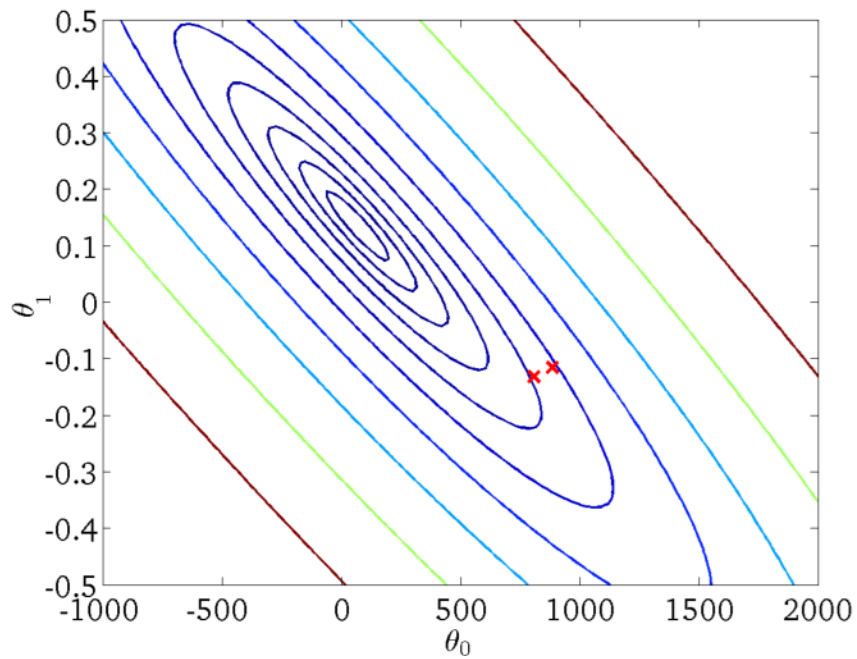
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

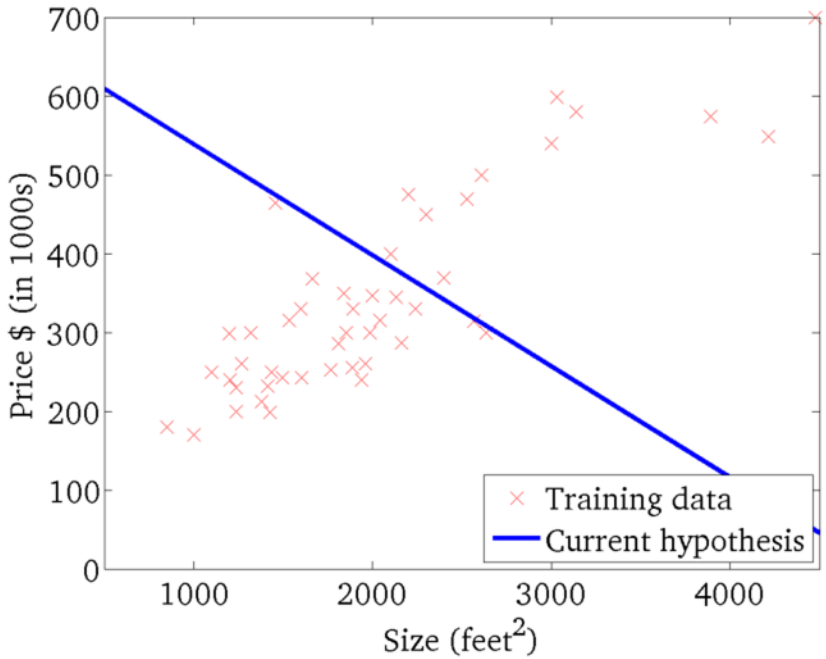


Source Credit : Slide by Andrew Ng

Gradient Descent

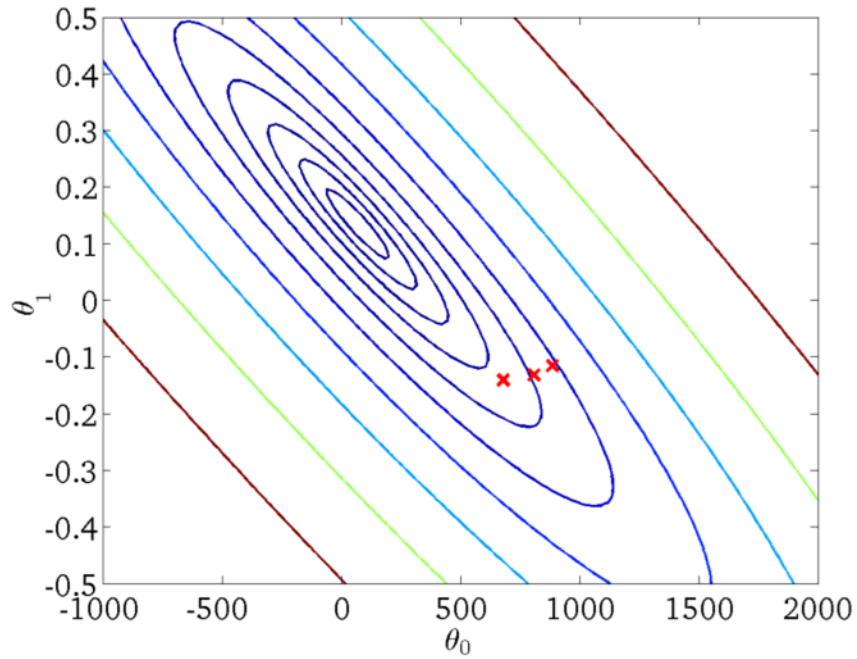
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

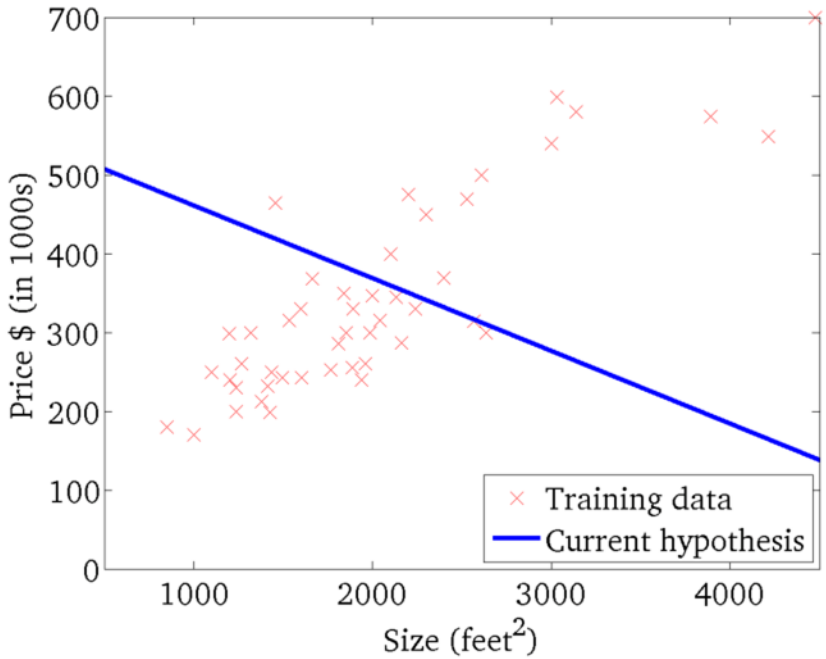


Source Credit : Slide by Andrew Ng

Gradient Descent

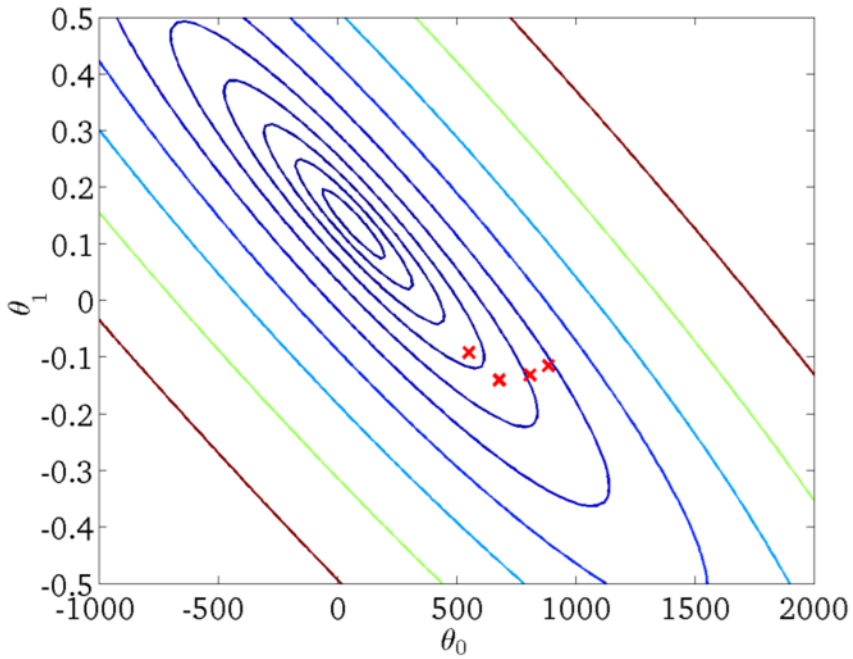
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

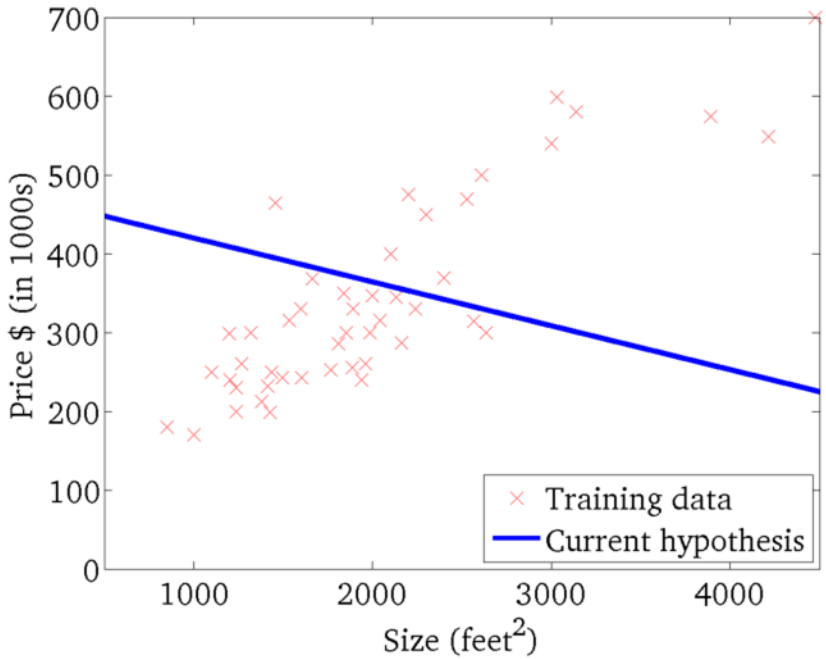


Source Credit : Slide by Andrew Ng

Gradient Descent

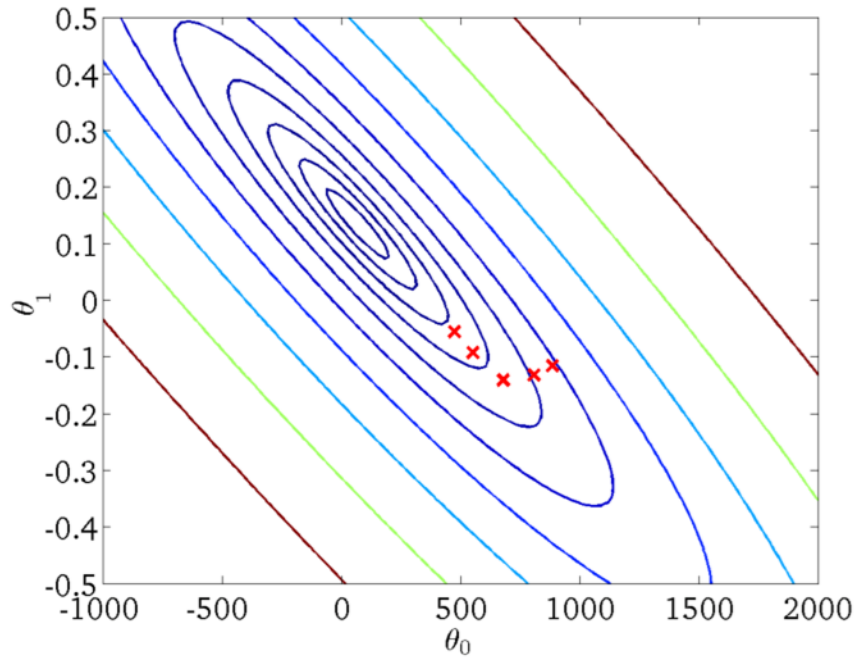
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

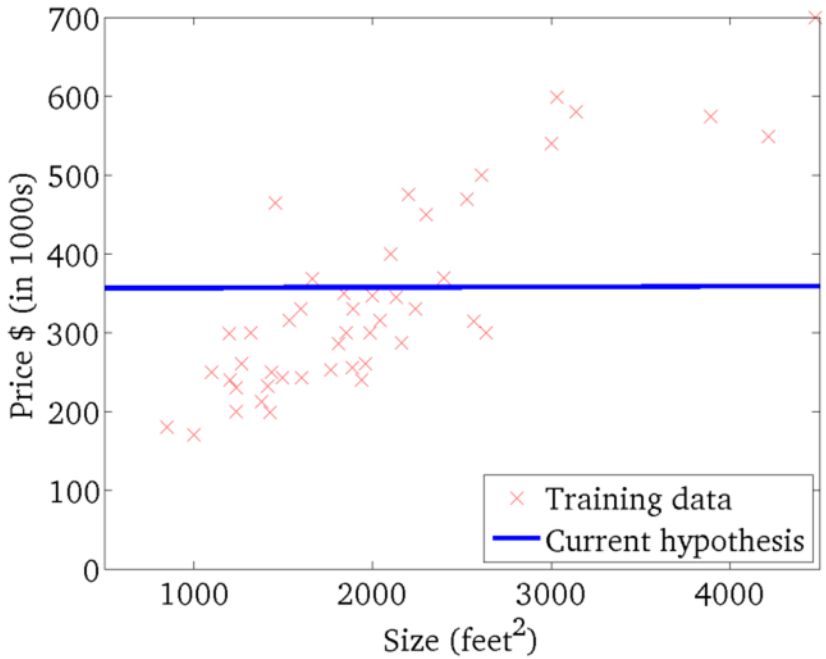


Source Credit : Slide by Andrew Ng

Gradient Descent

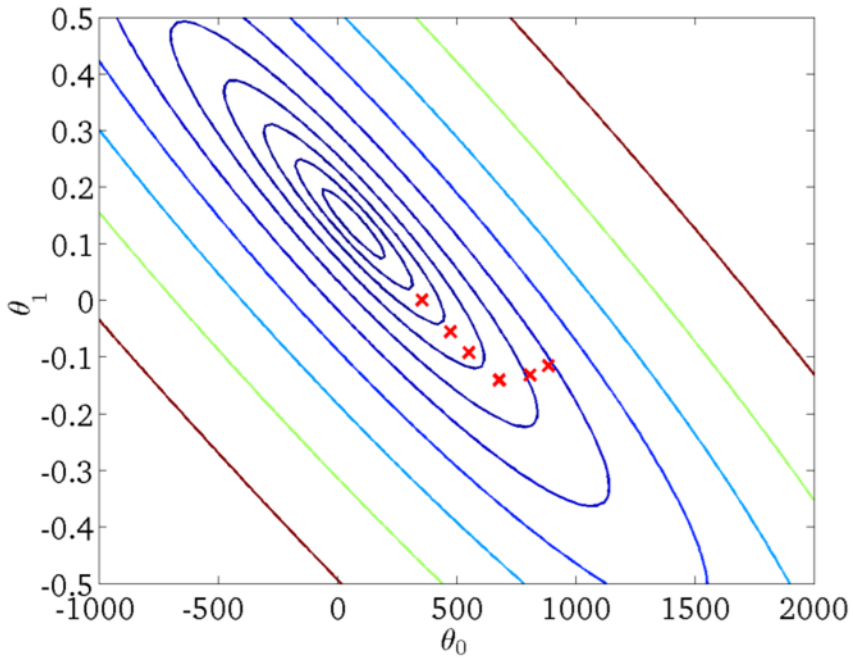
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

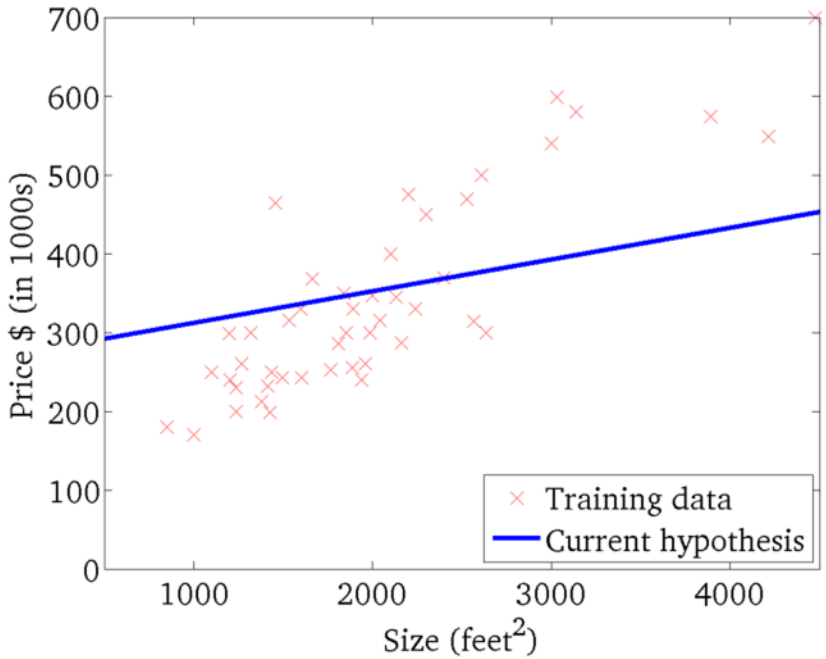


Source Credit : Slide by Andrew Ng

Gradient Descent

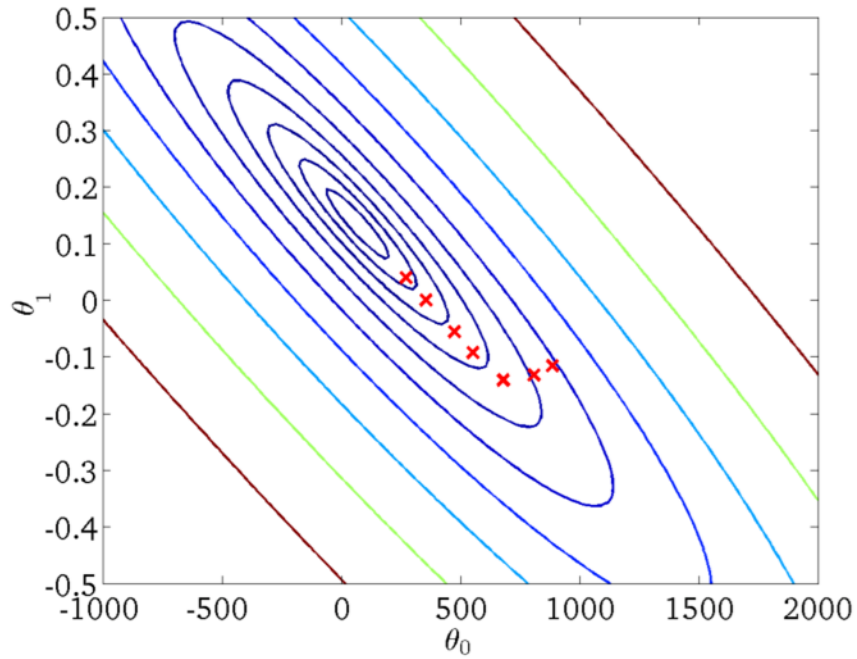
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

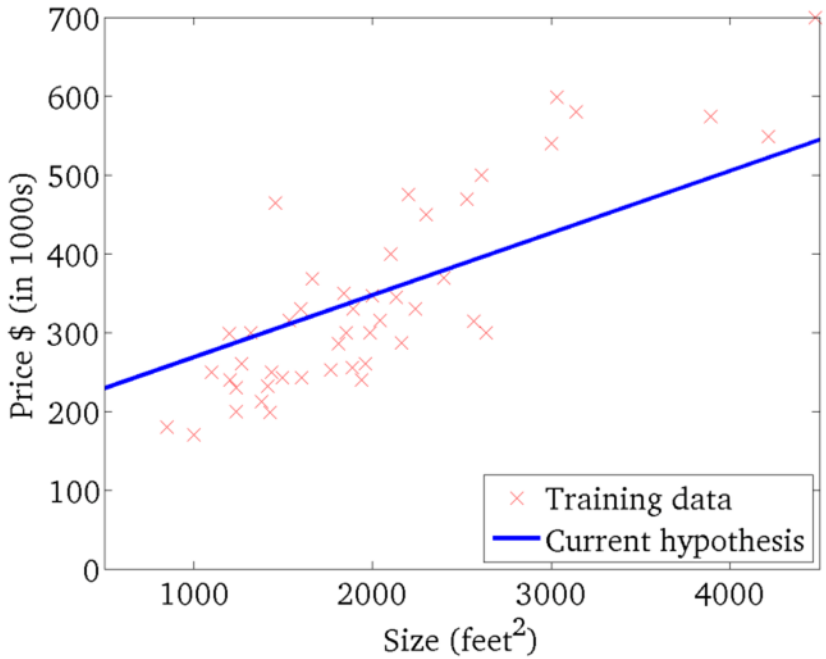


Source Credit : Slide by Andrew Ng

Gradient Descent

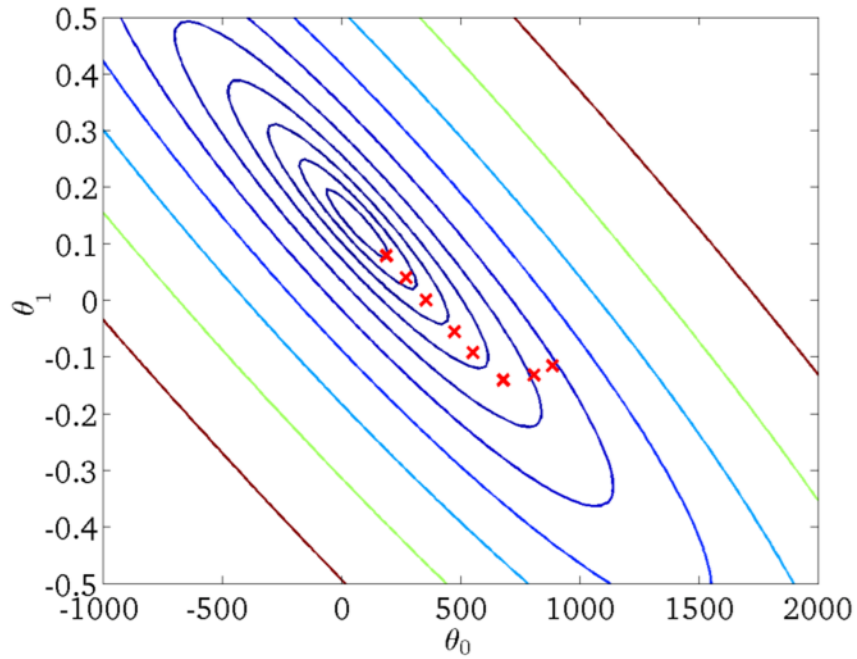
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

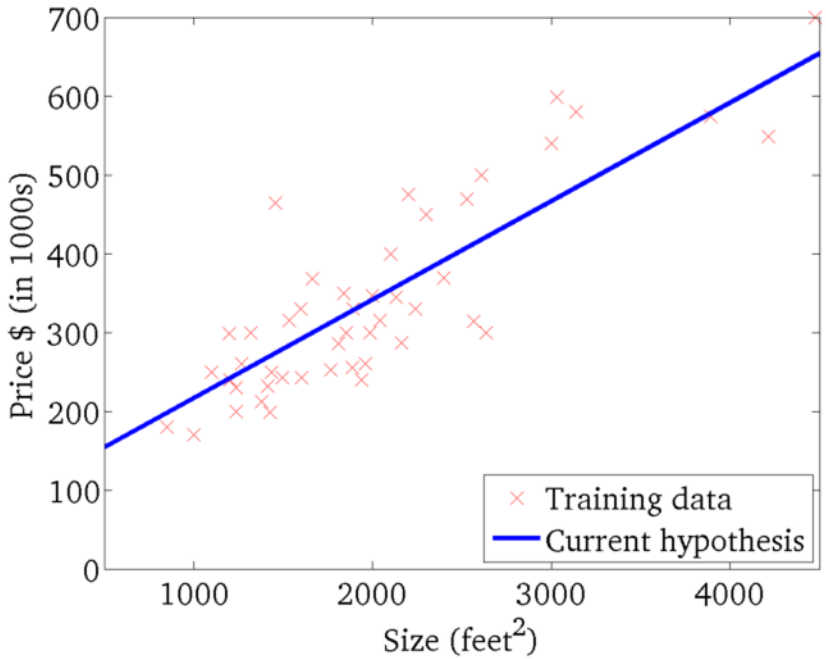


Source Credit : Slide by Andrew Ng

Gradient Descent

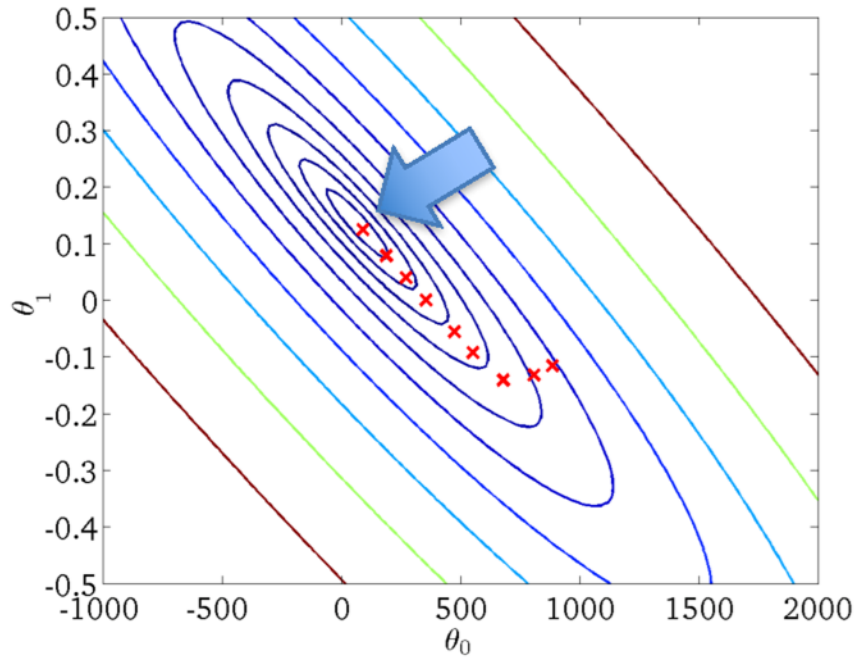
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Source Credit : Slide by Andrew Ng

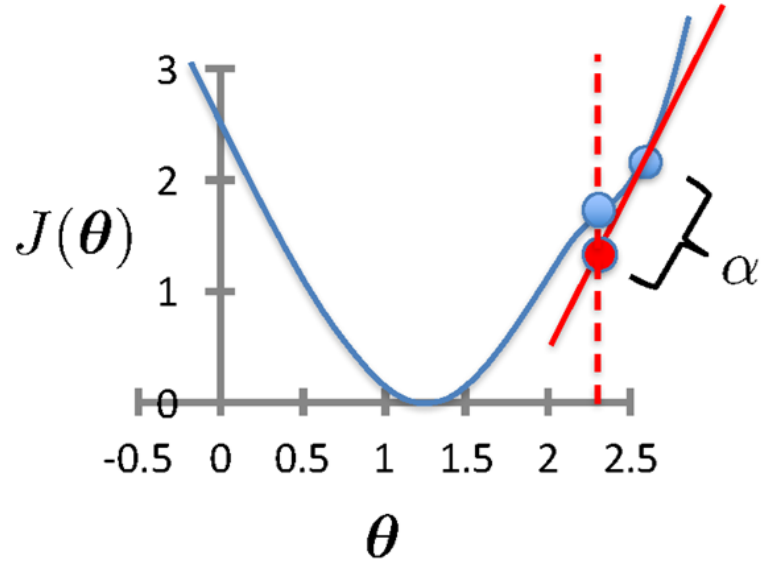
Gradient Descent

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update
for j = 0 ... d

learning rate (small)
e.g., α = 0.05



Gradient Descent

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update
for j = 0 ... d

For Linear Regression:

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) &= \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2 \\
&= \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right)^2 \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) \times \frac{\partial}{\partial \theta_j} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) x_j^{(i)}
\end{aligned}$$

Gradient Descent for Linear Regression

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

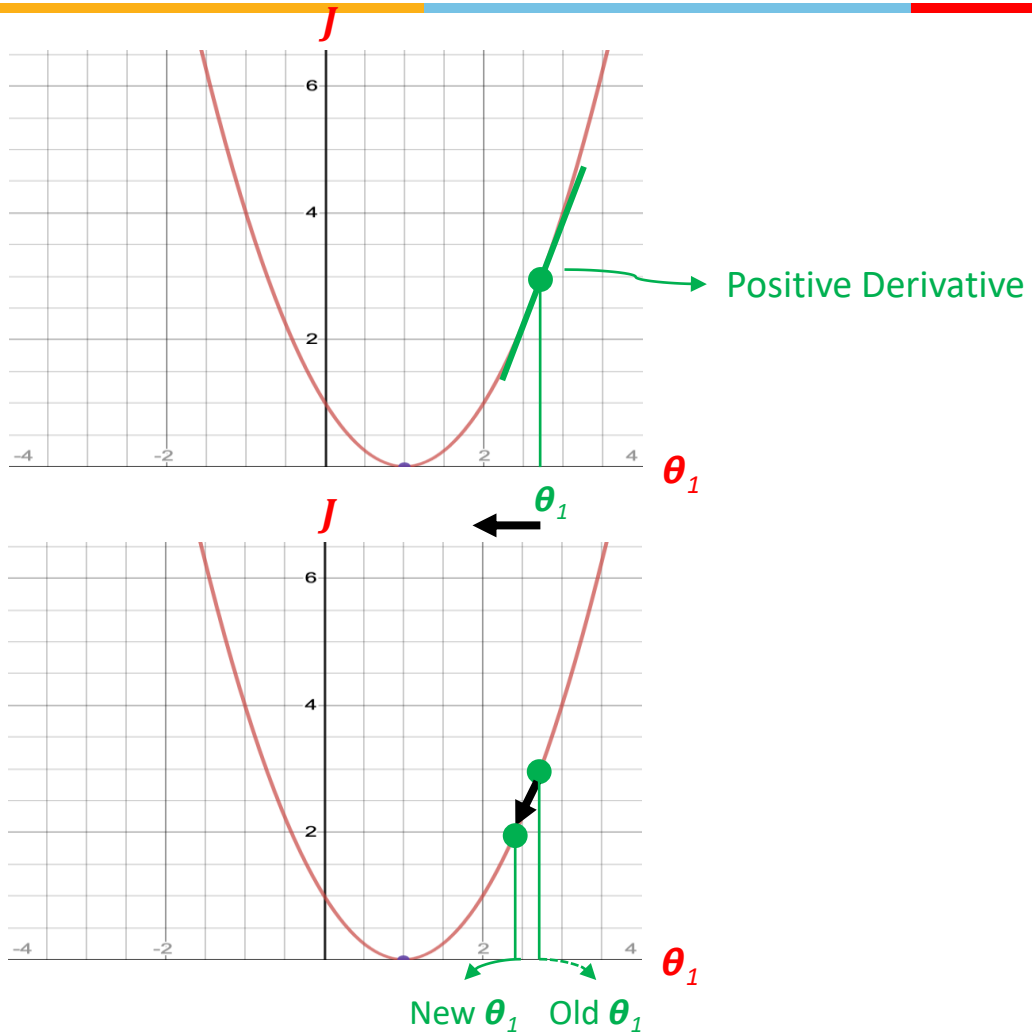
$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

simultaneous update for j = 0 ... d

- To achieve simultaneous update
  - At the start of each GD iteration, compute $h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right)$
  - Use this stored value in the update step loop
- Assume convergence when $\|\boldsymbol{\theta}_{new} - \boldsymbol{\theta}_{old}\|_2 < \epsilon$

L_2 norm: $\|\boldsymbol{v}\|_2 = \sqrt{\sum_i v_i^2} = \sqrt{v_1^2 + v_2^2 + \ldots + v_{|v|}^2}$

Guarantee of Convergence



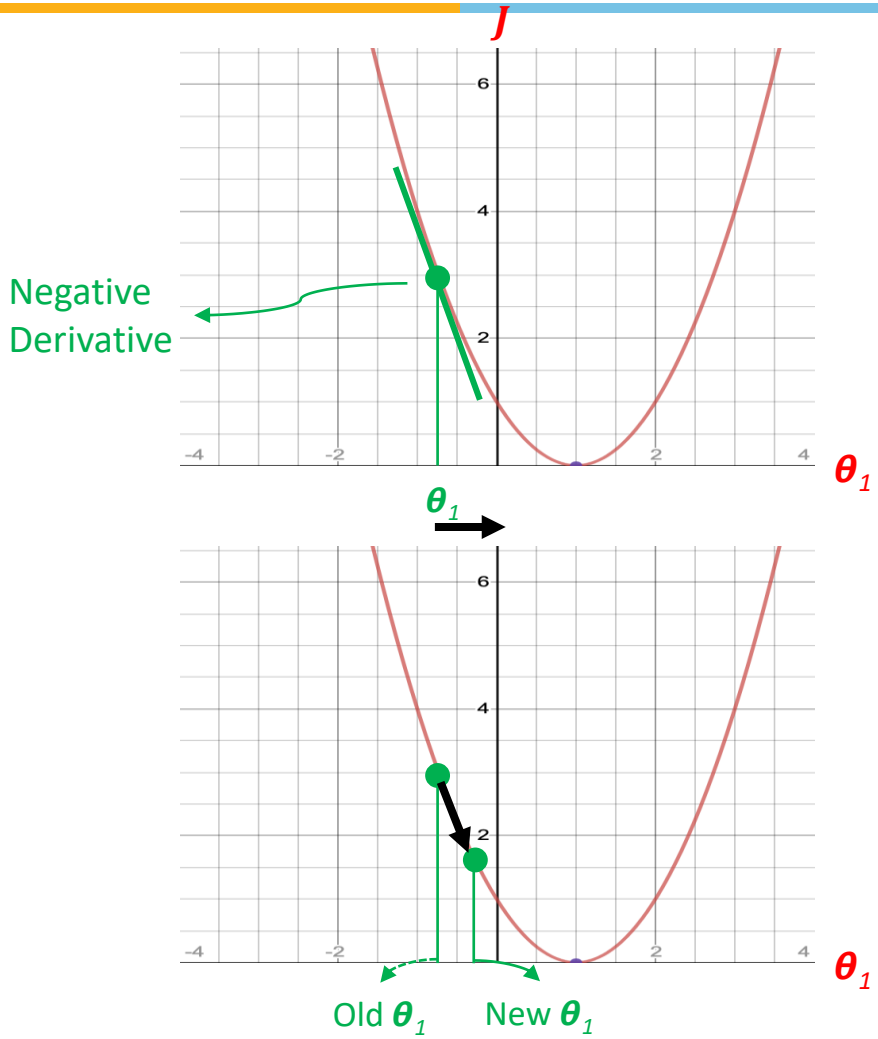
$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J}(\boldsymbol{\theta_1})}{\boldsymbol{d}\, \theta_j}$$

$$= \theta_1 - \alpha\, (Positive\ Number)$$

Decrease θ_1 by a certain value

Source Credit: Prof. Mohammad Hammoud

Guarantee of Convergence



$$\theta_1 = \theta_1 - \alpha \frac{\boldsymbol{d}\, \boldsymbol{J}(\boldsymbol{\theta_1})}{\boldsymbol{d}\, \theta_j}$$

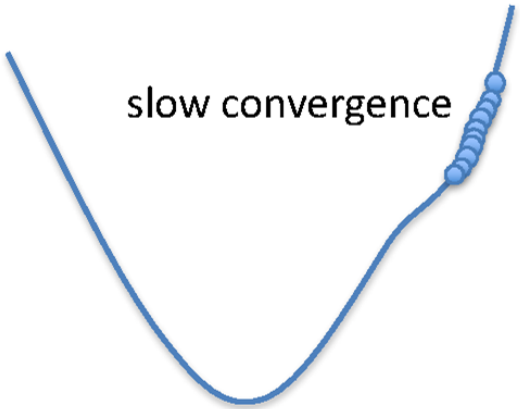
$$= \theta_1 - \alpha\, (Negative\ Number)$$

Increase θ_1 by a certain value

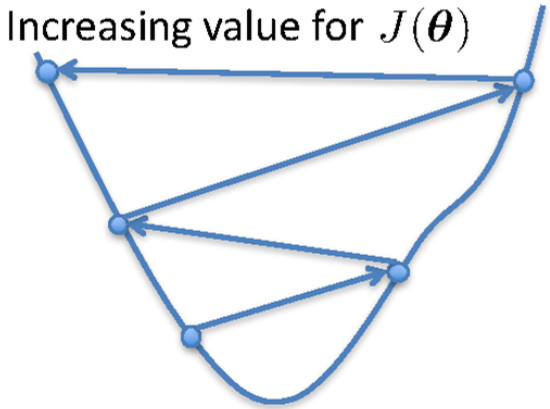
Source Credit: Prof. Mohammad Hammoud

Choosing Learning Rate

α too small



α too large



- May overshoot the minimum
- May fail to converge
- May even diverge

To see if gradient descent is working, print out $J(\theta)$ each iteration

- The value should decrease at each iteration
- If it doesn't, adjust α

Gradient Descent algorithm : Effect of Feature Scaling

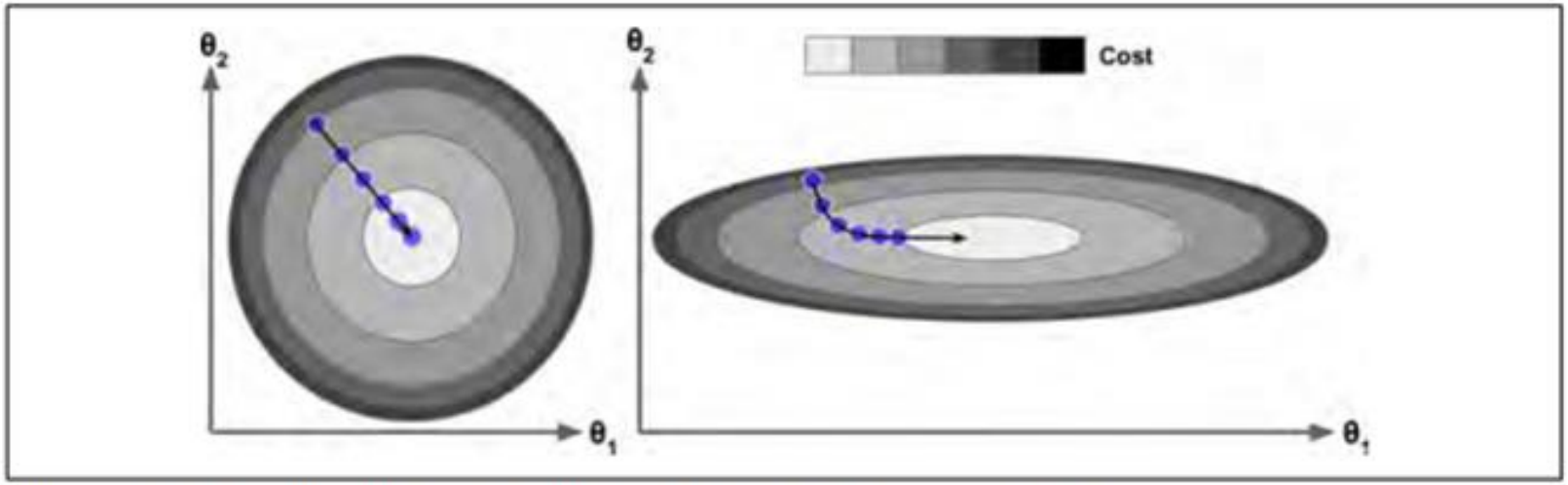


Figure 4-7. Gradient Descent with and without feature scaling

Gradient Descent algorithm : Effect of Learning Rate

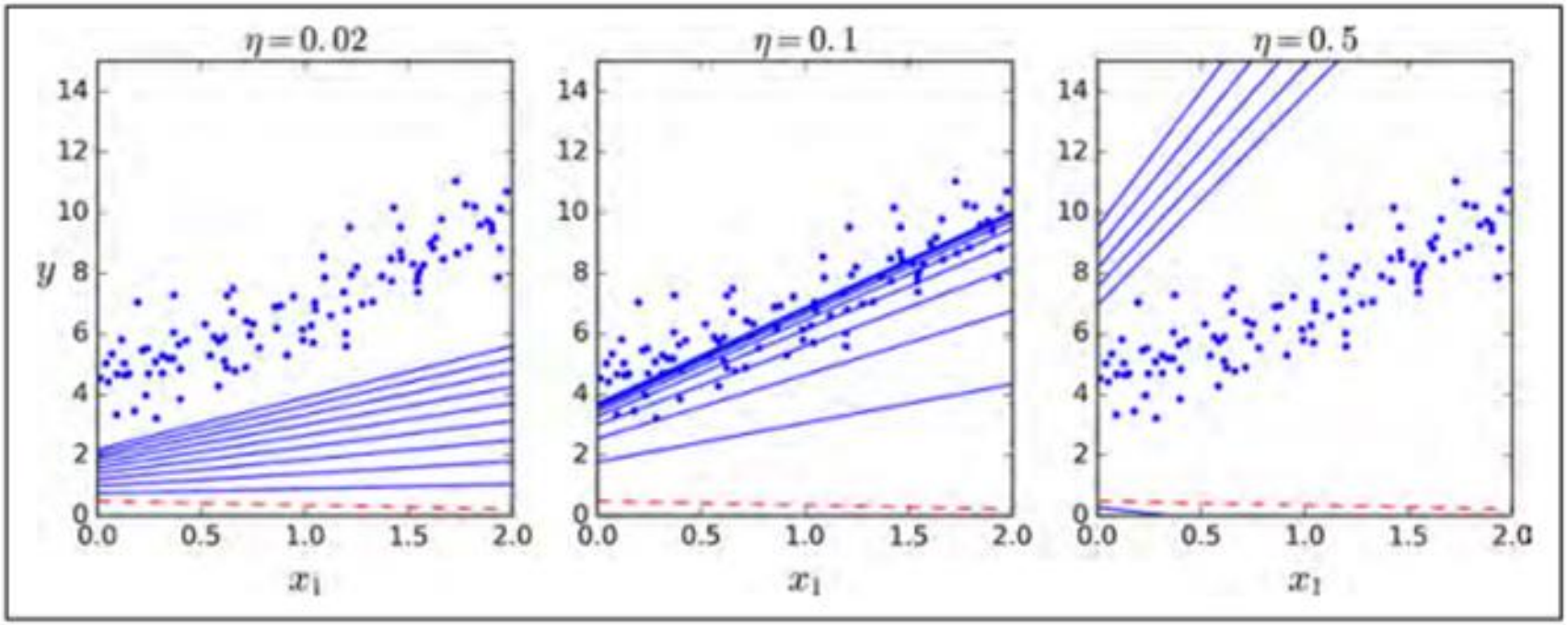


Figure 4-8. Gradient Descent with various learning rates

Gradient Descent: Variants

- **Batch** gradient descent refers to calculating the derivative from all training data before calculating an update.
- **Minibatch** refers to calculating derivative of mini groups of training data before calculating an update.
- **Stochastic** gradient descent refers to calculating the derivative from each training data instance and calculating the update immediately

repeat until convergence {

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_0 := \theta_0 - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})}$$

$$\theta_1 := \theta_1 - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}}$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

}

Update θ_0 and θ_1 simultaneously

$$temp0 := \theta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})$$

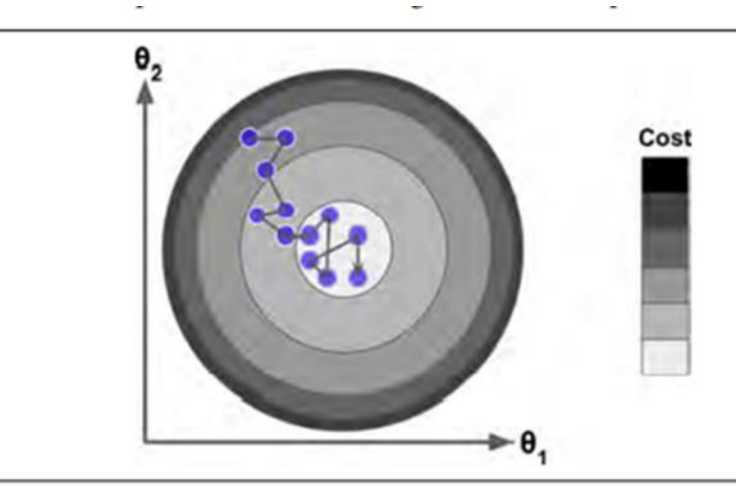
$$temp1 := \theta_1 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})\, x^{(i)}$$

$$\theta_0 := temp0$$

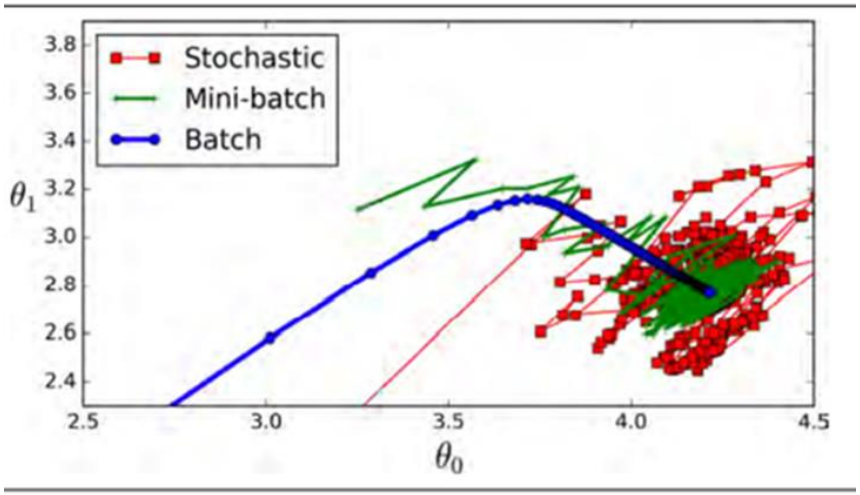
$$\theta_1 := temp1$$

Gradient Descent: Variants

- **Batch** gradient descent refers to calculating the derivative from all training data before calculating an update.
- **Minibatch** refers to calculating derivative of mini groups of training data before calculating an update.
- **Stochastic** gradient descent refers to calculating the derivative from each training data instance and calculating the update immediately



Stochastic Gradient Descent



1. Gradient Descent paths in parameter space

Fit a linear Regression Line :

Gradient Descent

Steps :

(Assuming : 'n' no. of instances and two predictors {x1, x2} and linear regression)

1. Identification of the equations y = w0+w1X1+W2X2
2. Cost function & derivative
    1. W0` = w0 – 1/n * learning rate * (sum (w0+w1X1+W2X2 - y))
    2. W1` = w1 – 1/n * learning rate * (sum (w0+w1X1+W2X2 - y) * x1)
    3. W2` = w2 – 1/n * learning rate * (sum (w0+w1X1+W2X2 - y)*x2)
3. Apply the equations

Fit a linear Regression Line :

Gradient Descent

Fit a linear regression. Show only the first iteration of Gradient descent algorithm using learning rate of **0.02** for the following data , if the Relative Risk of Coronary Heart Disease is believed to be only linearly dependent on BMI as well as Diastolic Pressure. Assume the intercept of the regression model as **5** and the slope of independent variables as **-0.03 (negative)**.

Patient	Systolic Pressure mm Hg	Diastolic Pressure mm Hg	BMI	Waist Circumference Threshold cm	RR-CHD (Relative Risk of Coronary Heart Disease)
1	140	80	35	100	1.81
2	120	80	25	80	1.22
3	130	100	30	60	1.71

Steps :

1. Identification of the equations y = w0+w1X1+W2X2
2. Cost function & derivative
   1. W0` = w0 – 1/3 * learning rate * (sum (w0+w1X1+W2X2 - y))
   2. W1` = w1 – 1/3 * learning rate * (sum (w0+w1X1+W2X2 - y) * x1)
   3. W2` = w2 – 1/3 * learning rate * (sum (w0+w1X1+W2X2 - y)*x2)
3. Apply the equations

Fit a linear Regression Line :

Gradient Descent

Fit a linear regression. Show only the first iteration of Gradient descent algorithm using learning rate of **0.02** for the following data , if the Relative Risk of Coronary Heart Disease is believed to be only linearly dependent on BMI as well as Diastolic Pressure. Assume the intercept of the regression model as **5** and the slope of independent variables as **-0.03 (negative)**.

Patient	Systolic Pressure mm Hg	Diastolic Pressure mm Hg	BMI	Waist Circumference Threshold cm	RR-CHD (Relative Risk of Coronary Heart Disease)
1	140	80	35	100	1.81
2	120	80	25	80	1.22
3	130	100	30	60	1.71

Steps :

1. Identification of the equations: RR-CHD = 5 – 0.03 * BMI – 0.03 * DiastolicPressure
2. Cost function & derivative
   1. W0` = w0 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD))
      = 5 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD))
   2. W1` = w1 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * BMI)
      = -0.03 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * BMI)
   3. W2` = w2 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * DiastolicPresure)
      = -0.03 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * DiastolicPresure)
3. Apply the equations : Answer at the end of first iteration:
   W0 = 5.0016 , W1 = 0.0476, w2= 0.179

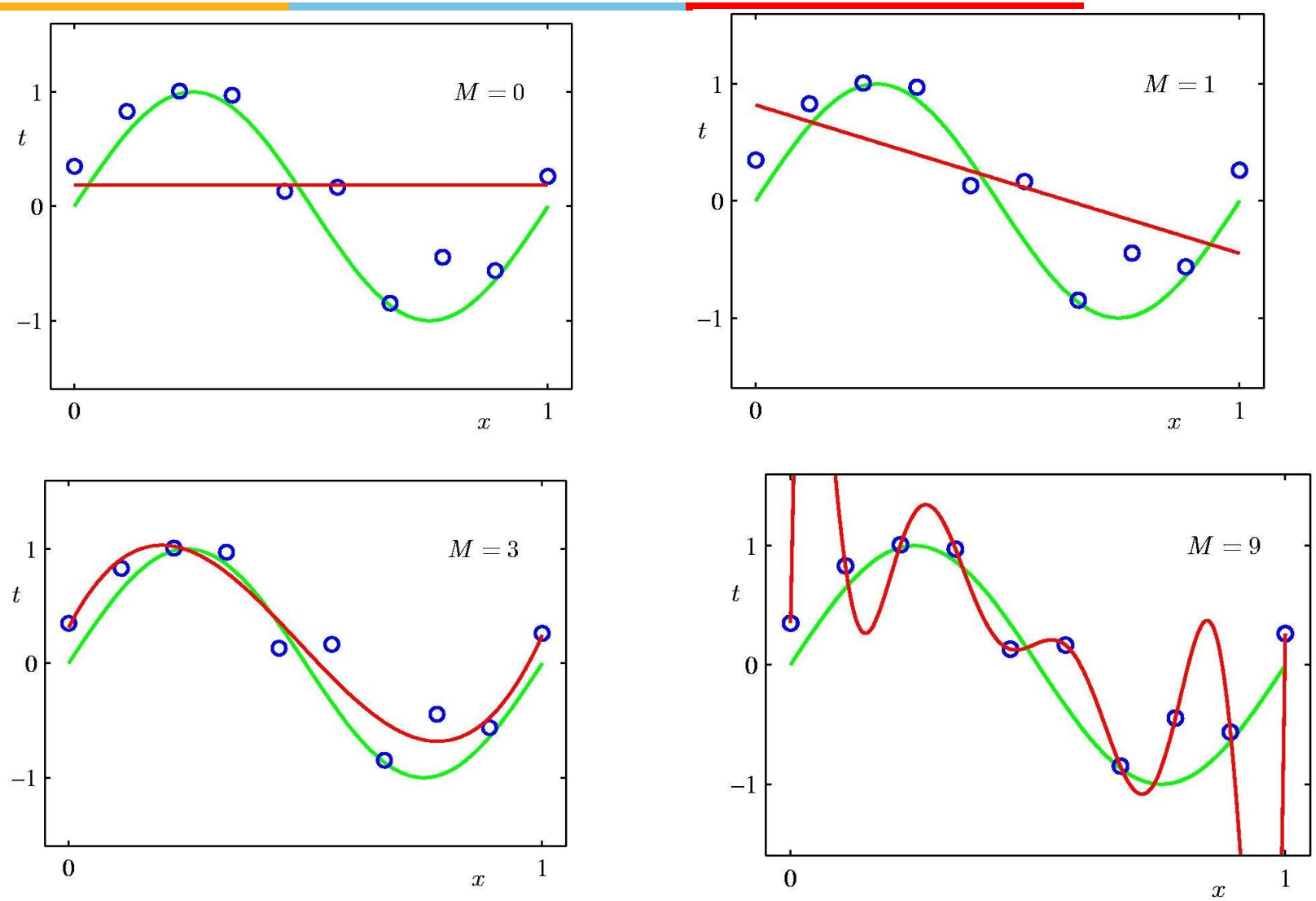
Closed Form Solution Vs. Gradient Descent

Gradient Descent	Closed Form Solution
• Requires multiple iterations • Need to choose α • Works well when n is large • Can support incremental learning	• Non-iterative • No need for α • Slow if n is large – Computing $(X^T X)^{-1}$ is roughly $O(n^3)$

Linear Basis Models

BITS Pilani

Polynomial Regression



Linear Basis Function Models

- The inputs **X** for linear regression can be:
  - Original quantitative inputs
  - Transformation of quantitative inputs
    - e.g. log, exp, square root, square, etc.
  - Polynomial transformation
    - example: $y = \beta_0 + \beta_1 \cdot x + \beta_2 \cdot x^2 + \beta_3 \cdot x^3$
  - Basis expansions
  - Dummy coding of categorical inputs
  - Interactions between variables
    - example: $x_3 = x_1 \cdot x_2$

This allows use of linear regression techniques to fit non-linear datasets.

X No.of.Years of Experience (in Years)	X^2	Y Salary Of the Employee (in Lakhs)
1	1	2
2	4	3
3	9	4
4	16	5
5	25	6

X1 = Graduate	X2 = PostGraduate	X3 = Others	Y Salary Of the Employee
0	0	1	2
1	0	0	3
0	0	1	4
0	1	0	5
1	0	0	6

Linear Basis Function Models

X No.of.Years of Experience (in Years)	X^2	Y Salary Of the Employee (in Lakhs)
1	1	2
2	4	3
3	9	4
4	16	5
5	25	6

Example: an M-th order polynomial function of one dimensional feature x:

$$y(x, \mathbf{w}) = w_0 + \sum_{j=1}^{M} w_j x^j$$

where x^j = j-th power of x

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

$\Phi_j(x)$ are known as *basis functions*. Typically, $\Phi_0(x) = 1$, so that w_0 acts as a bias.

In the simplest case, we use linear basis functions : $\Phi_d(x) = x_d$.

They are called linear models because this function is linear in **w**.

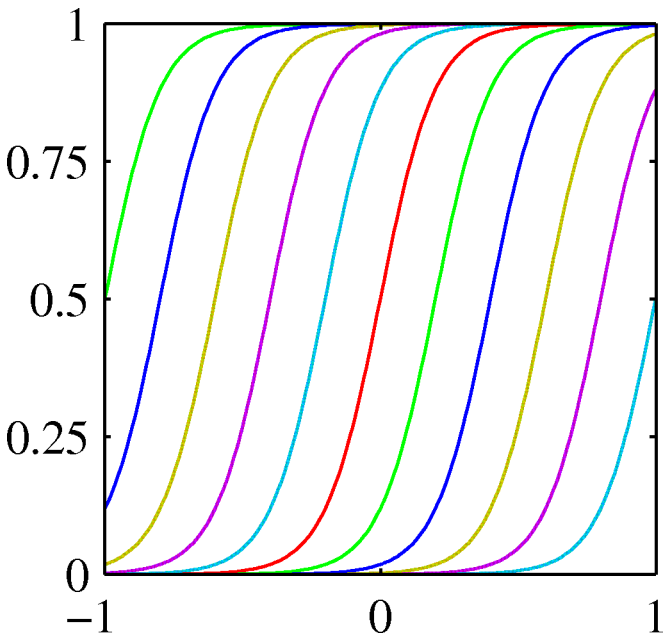
Linear Basis Function Models - Examples

Sigmoidal basis functions:

where $$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

Also these are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (slope).

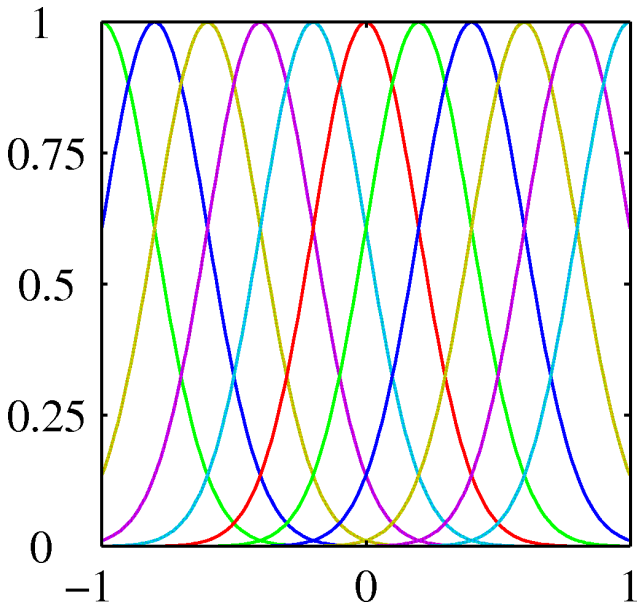


Linear Basis Function Models - Examples

Gaussian basis functions:

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

These are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (width).

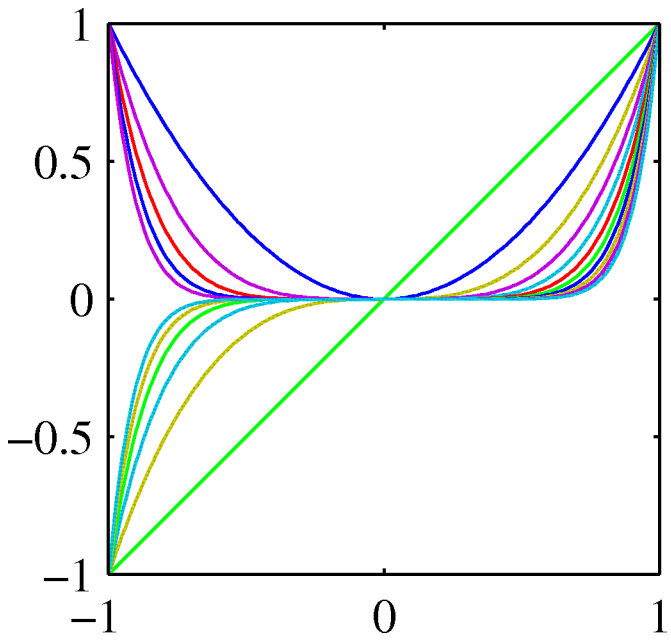


Linear Basis Function Models - Examples

Polynomial basis functions:

$$\phi_j(x) = x^j.$$

These are global; a small change in x affect all basis functions.

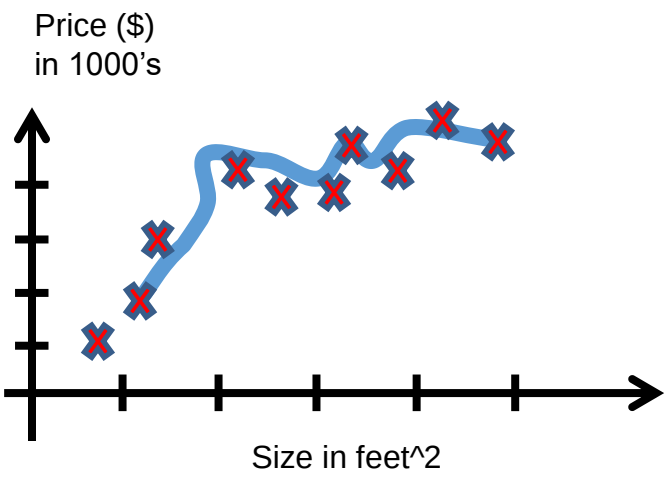


Notion of Bias - Variance

BITS Pilani

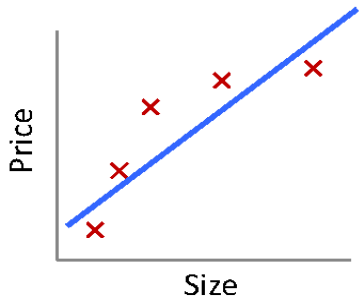
Addressing overfitting

- x_1 = size of house
- x_2 = no. of bedrooms
- x_3 = no. of floors
- x_4 = age of house
- x_5 = average income in neighborhood
- x_6 = kitchen size
- $\vdots$
- x_{100}



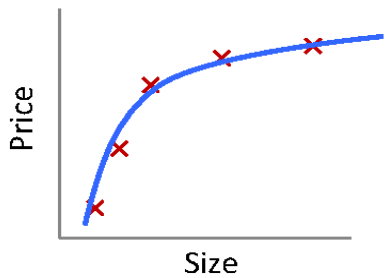
Slide credit: Andrew Ng

Quality of Fit



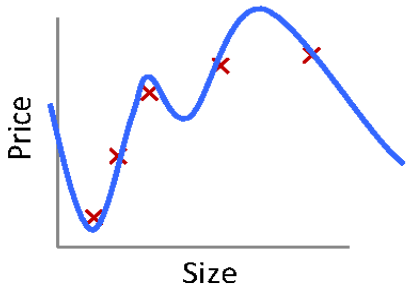
$\theta_0 + \theta_1 x$

Underfitting
(high bias)



$\theta_0 + \theta_1 x + \theta_2 x^2$

Correct fit



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Overfitting
(high variance)

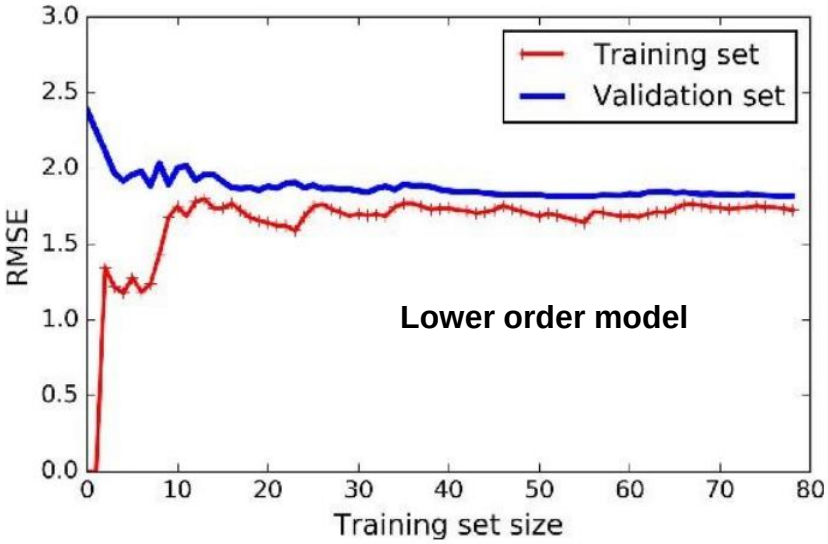
Overfitting:

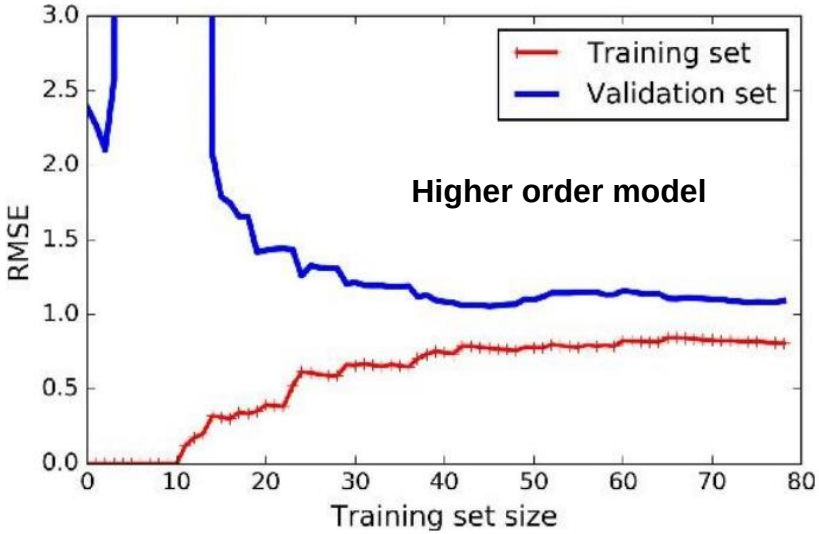
- The learned hypothesis may fit the training set very well ($J(\boldsymbol{\theta}) \approx 0$)
- ...but fails to generalize to new examples

Based on example by Andrew Ng

Effect of Training Size on Over fitting

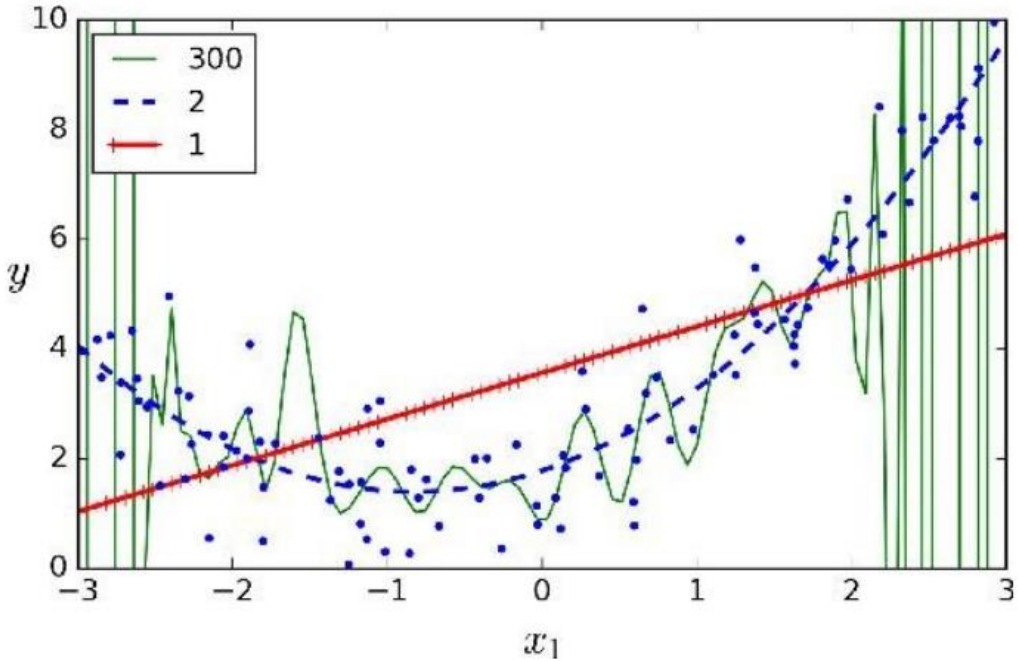
- Size of training dataset needs to be large to prevent when higher order model is used.





Polynomial Fitting can lead to Over fitting

- Underlying target function is quadratic
- Linear model results in under fitting with large bias
- Polynomial of order 300 results in a large variance



Regularization

BITS Pilani

Regularization

- A method for automatically controlling the complexity of the learned hypothesis
- **Idea**: penalize for large values of θ_j
  - Can incorporate into the cost function
  - Works well when we have a lot of features, each that contributes a bit to predicting the label
- Can also address overfitting by eliminating features (either manually or via model selection)

Regularization

- Linear regression objective function

$$J(\boldsymbol{\theta}) = \underbrace{\frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2}_{\text{model fit to data}} + \underbrace{\frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2}_{\text{regularization}}$$

  - λ is the regularization parameter ($\lambda \geq 0$)
  - No regularization on θ_0!

Understanding Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Note that $\sum_{j=1}^{d}\theta_j^2 = \|\boldsymbol{\theta}_{1:d}\|_2^2$
  - This is the magnitude of the feature coefficient vector!

- We can also think of this as:

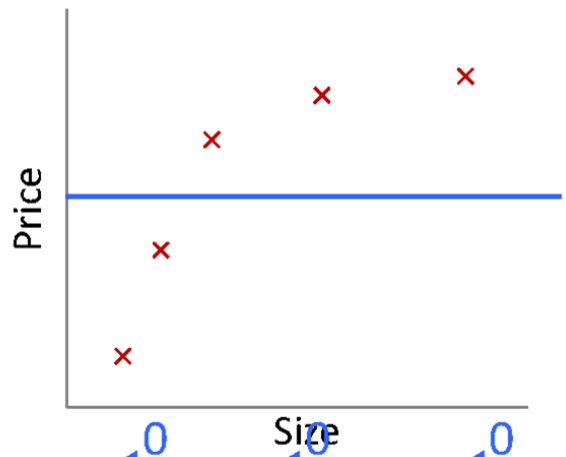
$$\sum_{j=1}^{d}(\theta_j - 0)^2 = \|\boldsymbol{\theta}_{1:d} - \vec{\mathbf{0}}\|_2^2$$

  - L_2 regularization pulls coefficients toward 0

Understanding Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- What happens if we set λ to be huge (e.g., 10^{10})?



$$\theta_0 + \cancelto{0}{\theta_1} x + \cancelto{0}{\theta_2} x^2 + \cancelto{0}{\theta_3} x^3 + \cancelto{0}{\theta_4} x^4$$

Based on example by Andrew Ng

Regularized Linear Regression

- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Gradient update:

$$\frac{\partial}{\partial\theta_0}J(\theta) \qquad \theta_0 \leftarrow \theta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$$\frac{\partial}{\partial\theta_j}J(\theta) \qquad \theta_j \leftarrow \theta_j - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} - \lambda\theta_j$$

- We can rewrite the gradient step as:

$$\theta_j \leftarrow \theta_j\left(1 - \alpha\lambda\right) - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

Thank you !

Welcome!!

Machine Learning

AIML CZG565

Linear Models for Classification

BITS Pilani
Pilani Campus

Dr. Sugata Ghosal
sugata.ghosal@pilani.bits-pilani.ac.in

Agenda

- Discriminant Functions
- Probabilistic Generative Classifiers
- Probabilistic Discriminative Classifiers
- Logistic Regression
- Applications : Text classification model, Image classification

Decision Theory
&
Classification Models

Inductive Learning Hypothesis : Interpretation

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No, Maybe} Classification
- Continuous : f(x) ∈ [20-100] Regression
- Probability Estimation : f(x) ∈ [0-1]

discriminative classifier

f(x)

Sky	AirTemp	Humidity	Wind	Water	Forecast	*EnjoySport?*
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Decision Theory

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No} ie., **t** ∈ {0, 1} Binary Classification
- Continuous : f(x) ∈ [20-100]
- Probability Estimation : f(x) ∈ [0-1]

Sky	AirTemp	Humidity	Wind	Water	Forecast	*EnjoySport?*
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Decision Theory :

The decision problem: given x, predict t according to a probabilistic model p(x, t)

- Target Concept : **t**
- Discrete : f(x) ∈ {Yes, No} ie., **t** ∈ {0, 1}
- Continuous : f(x) ∈ [20-100]
- Probability Estimation : f(x) ∈ [0-1]

$p(x, C_k)$ is the (central!) inference problem

	Sky	AirTemp	Humidity	Wind	Water	Forecast	*P(EnjoySport =Yes)*
X =<	Sunny ,	Warm ,	Normal ,	Strong ,	Warm ,	Same > →	0.95 = P(C_1 | X)
	Sunny	Warm	High	Strong	Warm	Same	0.7
	Rainy	Cold	High	Strong	Warm	Change	0.5
	Sunny	Warm	High	Strong	Cool	Change	0.6

Classification Problem: Stages

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}.$$

$$= \frac{p(x, C_k)}{p(x)} = \frac{p(x, C_k)}{\sum_{k=1}^{2} p(x, C_k)}$$

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	Yes
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes

Training Set

Learning algorithm

Induction/ Inference step

Learn Model for $p(x,C_k)$

Model

Apply Model to find optimal t

Deduction/ Decision Step

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Rainy	Cold	High	Strong	Change	?
Sunny	Warm	High	Strong	Change	?
Rainy	Warm	Normal	Breeze	Same	?

Test Set

Decision Region

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	Yes
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes

Training Set

Induction/ Inference step

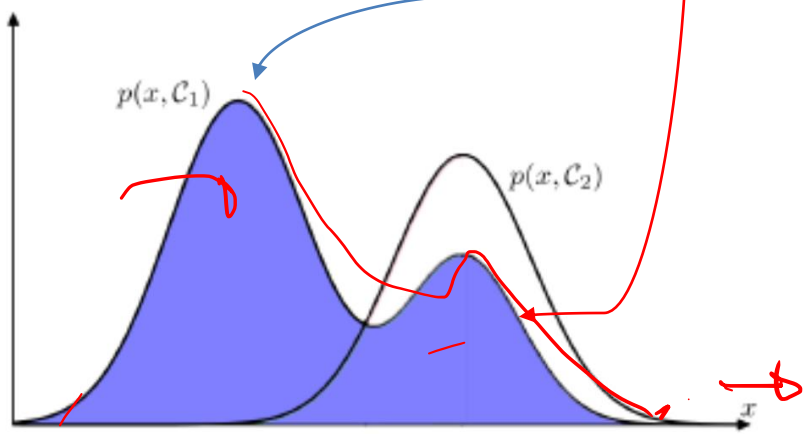
Learning algorithm

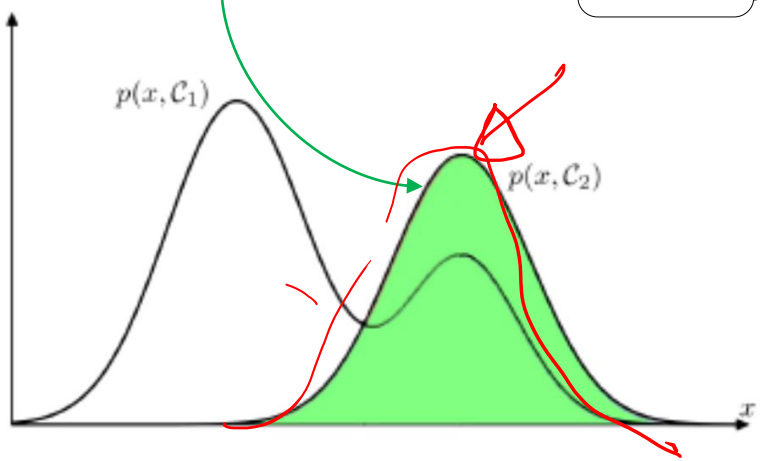
Learn Model for $p(x,C_k)$

Model divides the input space into regions $\mathbf{R}_k$ called **decision regions**, one for each class, such that all points in R_k are assigned to class C_k

A mistake occurs when an input vector belonging to class C_1 is assigned to class C_2

Model





Misclassification Rate

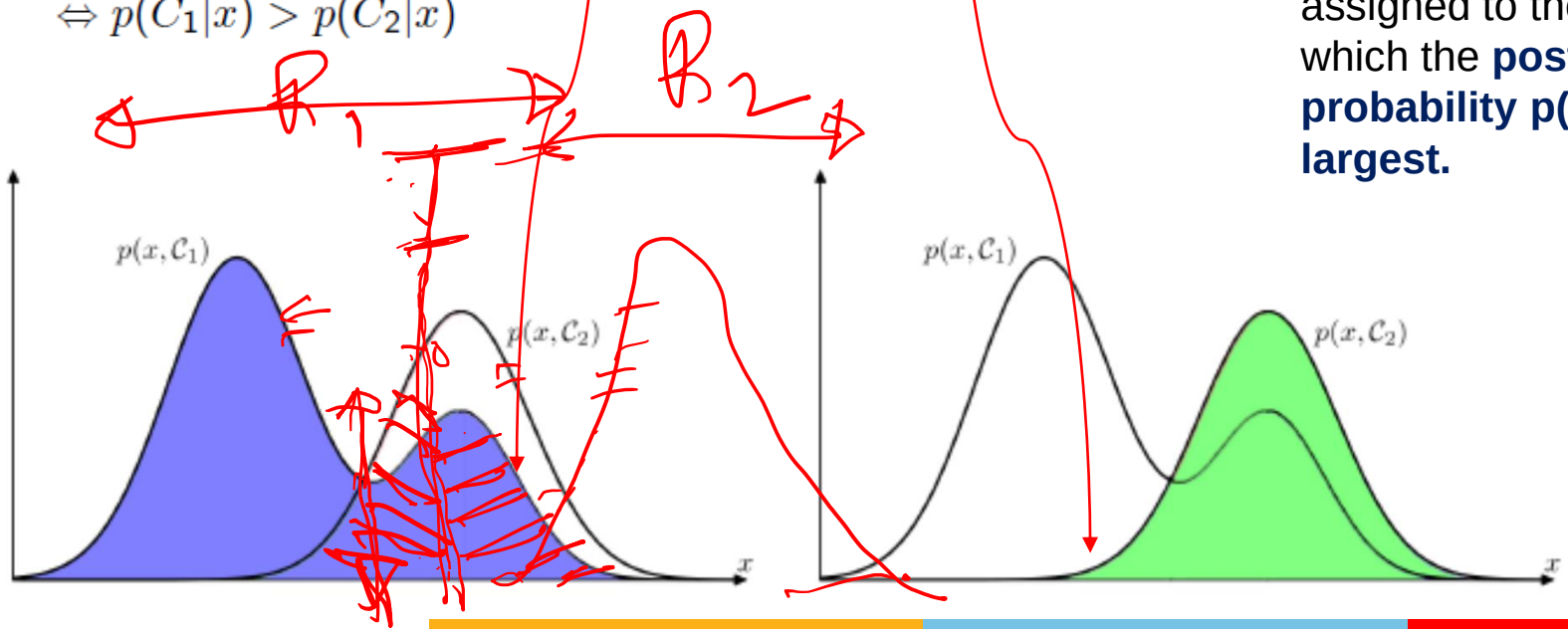
$$p(C_k|x) = \frac{p(x, C_k)}{p(x)}$$

$$\begin{aligned} p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2)\, \mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1)\, \mathrm{d}\mathbf{x}. \end{aligned}$$

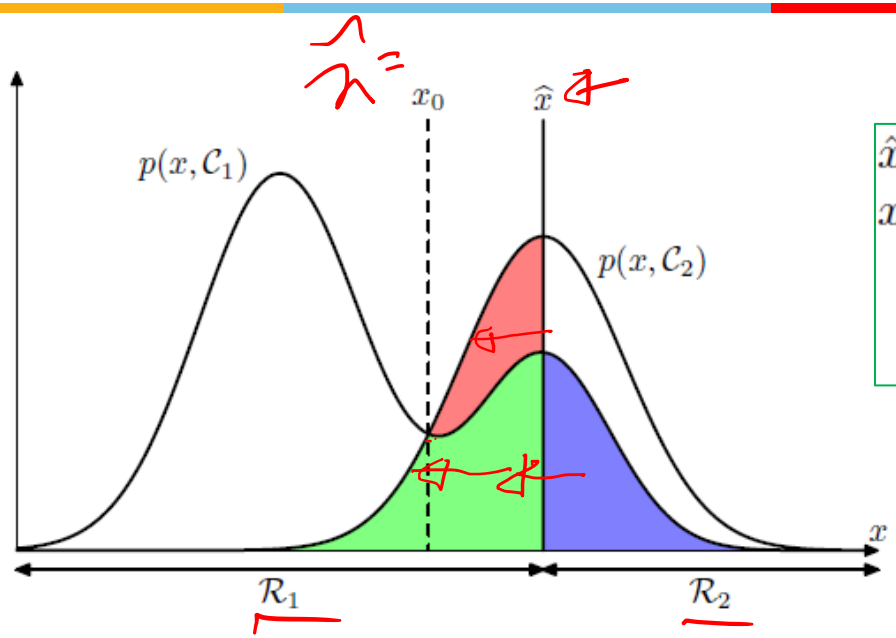
$$\begin{aligned} & p(x, C_1) > p(x, C_2) \\ & \Leftrightarrow p(C_1|x)p(x) > p(C_2|x)p(x) \\ & \Leftrightarrow p(C_1|x) > p(C_2|x) \end{aligned}$$

To minimize p(mistake), each **x** is assigned to whichever class has the smaller value of the integrand

The minimum probability of making a mistake is obtained if each value of **x** is assigned to the class for which the **posterior probability $p(C_k|x)$ is largest.**



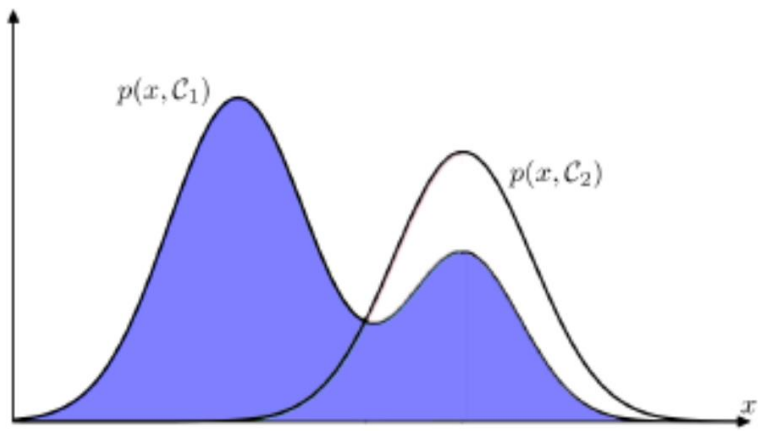
Decision Theory - Summary

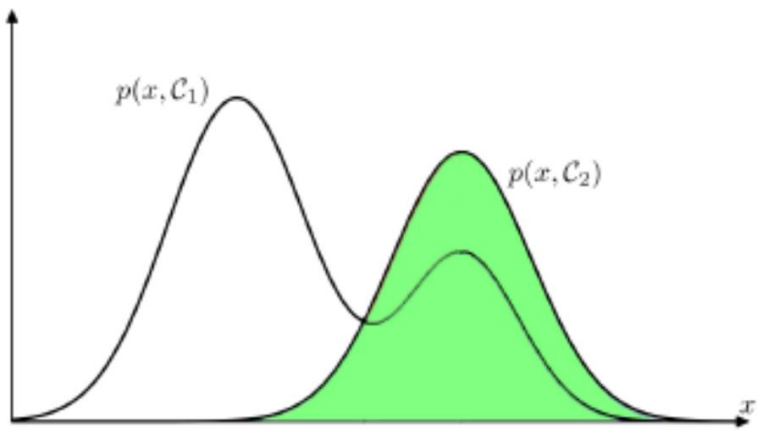


$\hat{x}$: decision boundary.

x_0: optimal decision boundary

$$x_0 : \arg\min_{\mathcal{R}_1}\{p\,(\text{mistake})\}$$





Linear Models for Classification

Types of Classification

Inductive Learning Hypothesis : Interpretation

- Target Concept
- Discrete : f(x) ∈ {Yes, No, Maybe} Classification
- Continuous : f(x) ∈ [20-100] Regression
- Probability Estimation : f(x) ∈ [0-1]

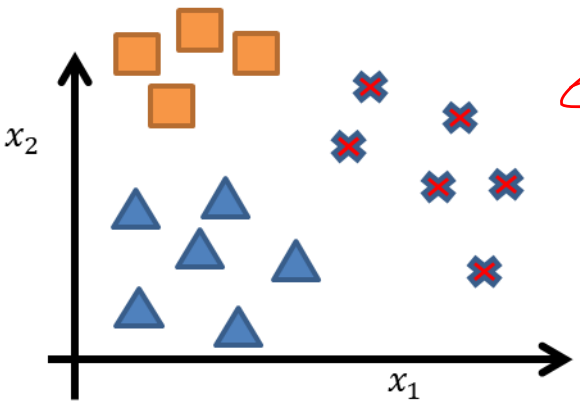
Sky	AirTemp	Altitude	Wind	Water	Forecast	Humidity
Sunny	Warm	Normal	Strong	Warm	Same	60
Sunny	Warm	High	Strong	Warm	Same	75
Rainy	Cold	High	Strong	Warm	Change	70
Sunny	Warm	High	Strong	Cool	Change	45

Types of Classification

Output Labels

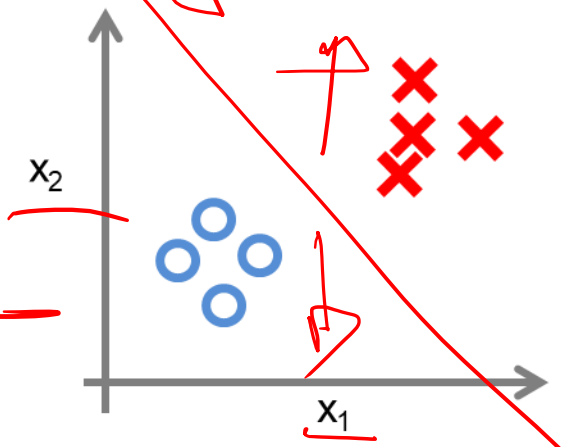
- Target Concept

Multi Class

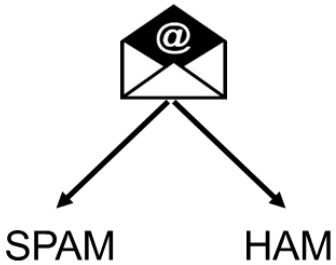


Enjoy Sports

Binary



Spam Classifier

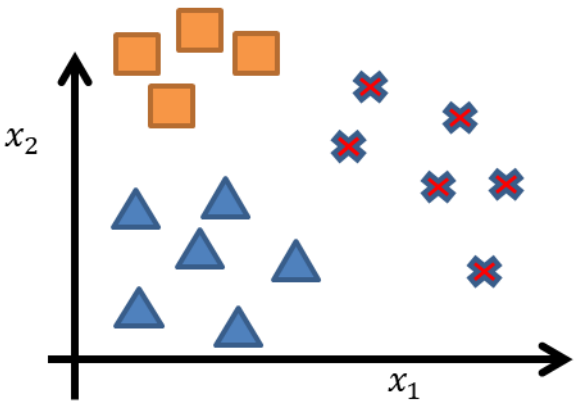


Types of Classification

Output Labels

- Target Concept

Multi Class

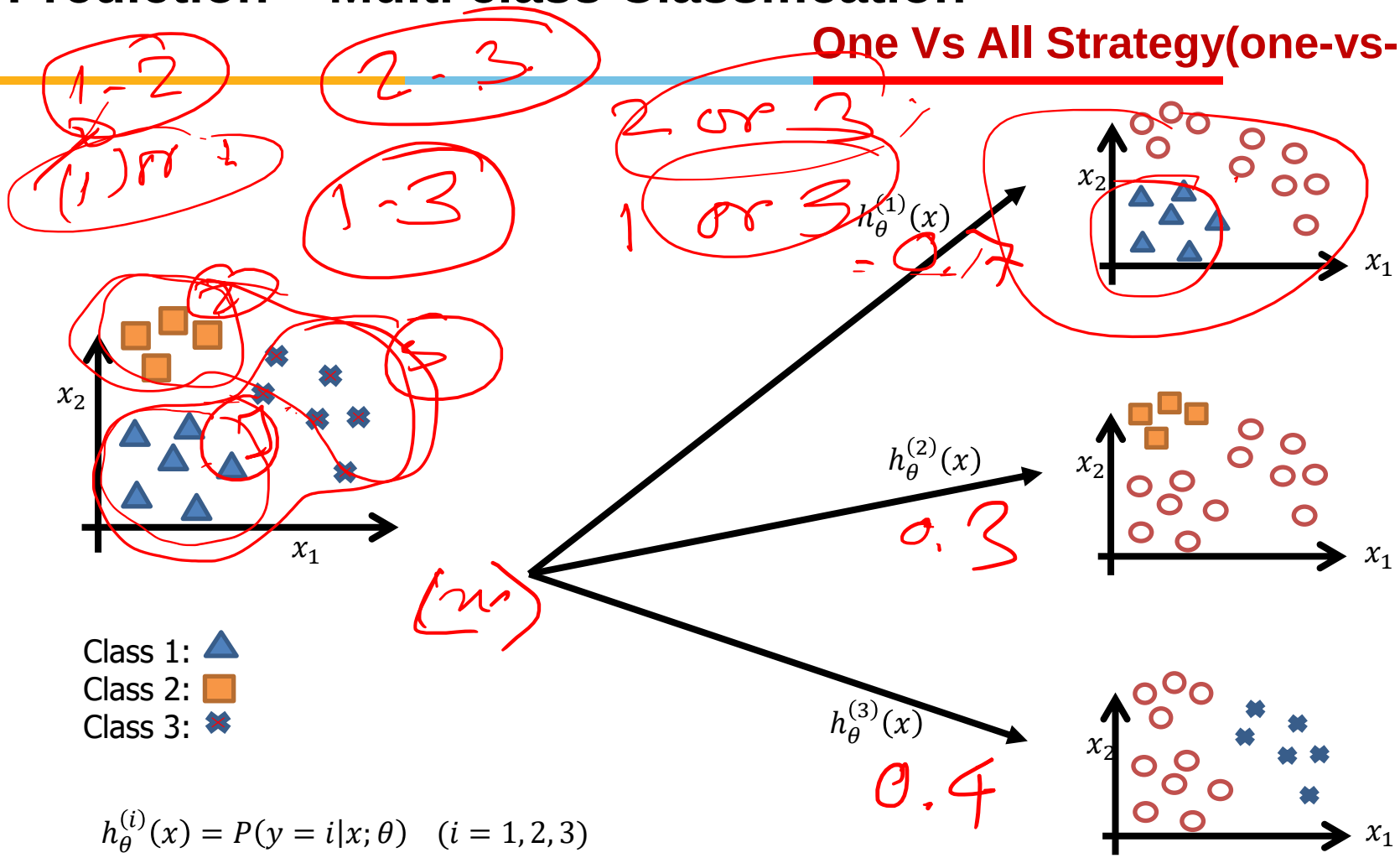


Enjoy Sports

YES NO MAYBE

Prediction – Multi class Classification

One Vs All Strategy(one-vs-rest)



Class 1: ▲
Class 2: ■
Class 3: ✖

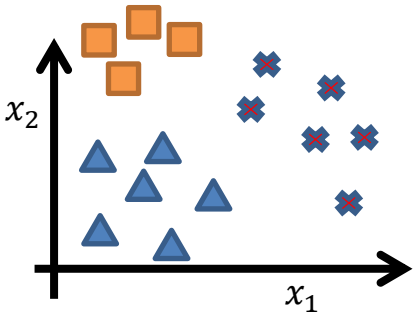
$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

$$\text{For input x Predict}: \ \max_i h_\theta^{(i)}(x)$$

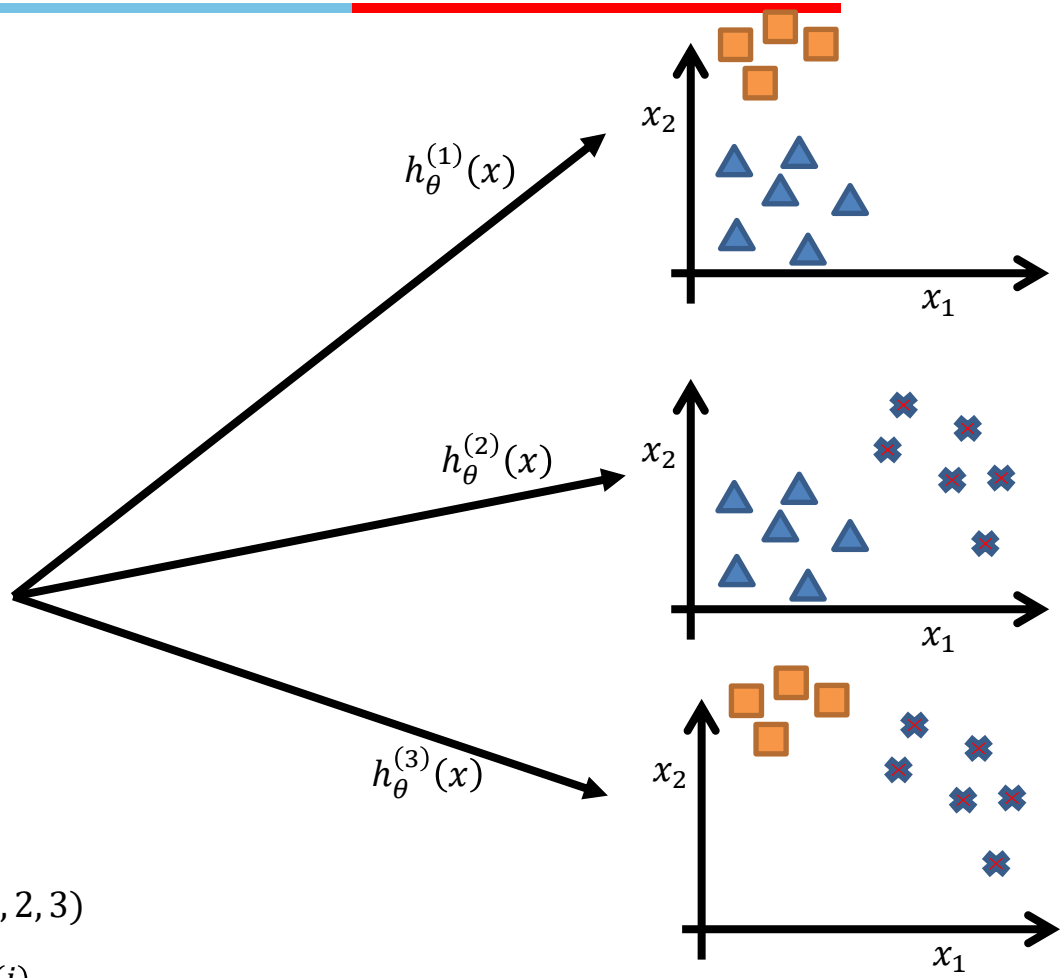
Note: Scikit-Learn detects when you try to use a binary classification algorithm for a multi-class classification task, and it automatically runs OvA (except for SVM classifiers for which it uses OvO)

Prediction – Multi class Classification

One Vs One Strategy



Class 1:
Class 2:
Class 3:



$$h_\theta^{(i)}(x) = P(y = i|x;\theta) \quad (i = 1, 2, 3)$$

$$\text{For input x Predict}_{\text{i}}: \ \max h_\theta^{(i)}(x)$$

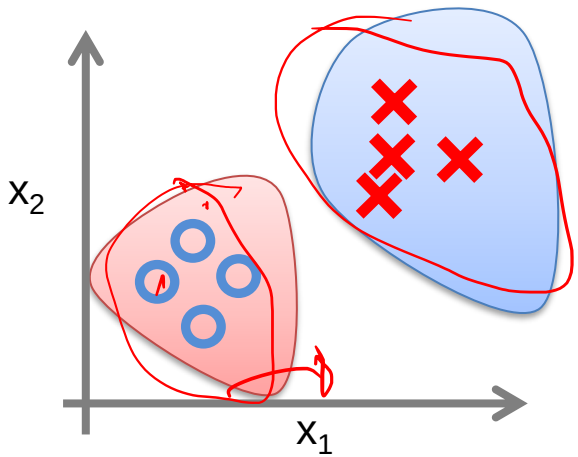
N × (N – 1) / 2 classifiers

Types of Classification

Decision Theory: Interpretation

Model Building

Generative



Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

$$P(Y \mid X_1 X_2 \ldots X_n) = \frac{P(X_1 X_2 \ldots X_d \mid Y) P(Y)}{P(X_1 X_2 \ldots X_d)}$$

Known as generative models, because by sampling from them it is possible to generate synthetic data points in the input space.

Eg., Gaussians, **Naïve Bayes**, Mixtures of multinomials **, Mixtures of Gaussians**, Bayesian networks

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

Types of Classification

$f(x) > 0.5$ for all points

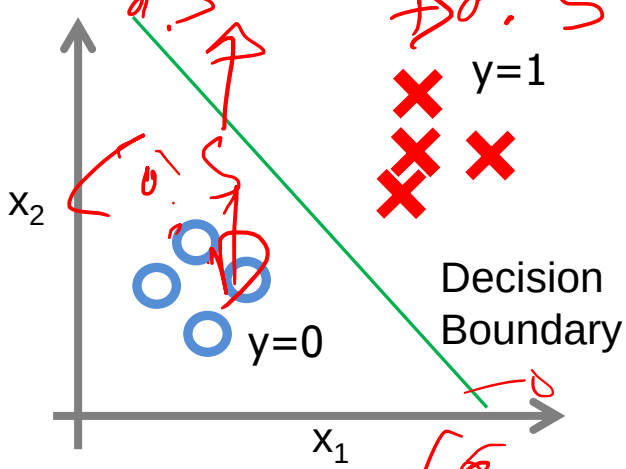
< 0.5 for 0

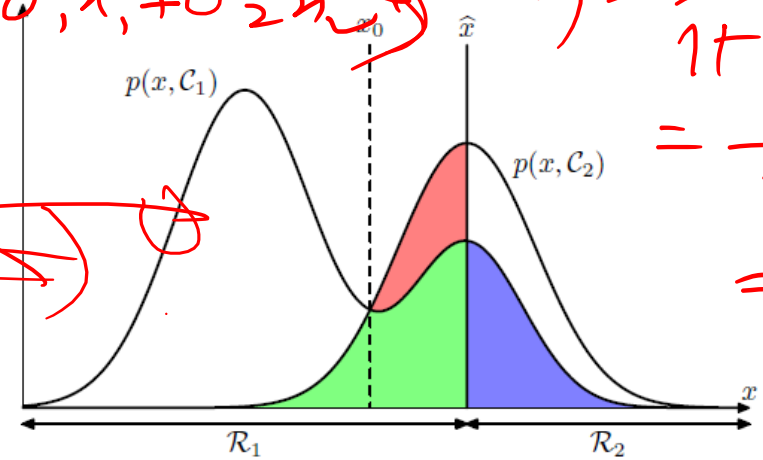
Decision Theory: Interpretation

Model Building

$$f(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1+\theta_2 x_2+\dots)}} = \frac{1}{1+e^{-0}} = \frac{1}{1+1} = 0.5$$

Discriminative





$P(c \mid x)$

Posterior Probability

$$\theta_0 + \sum_i \theta_i x_i \geq 0$$

$$\theta_0 + \sum_i \theta_i x_i < 0$$

$(\theta_0 + \theta_1 x_1 + \theta_2 x_2 \dots)$

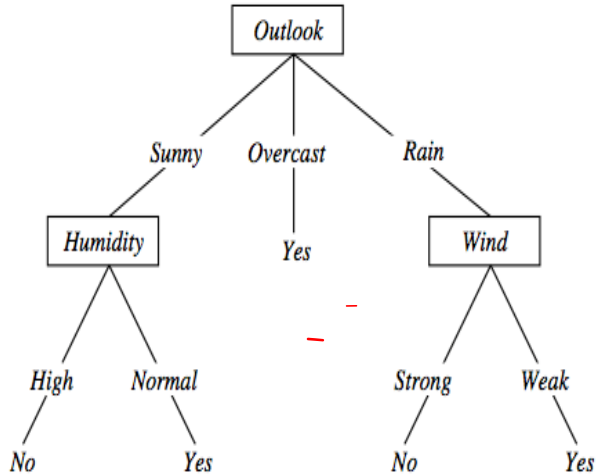
Logistic regression, SVMs , tree based classifiers (e.g. decision tree) Traditional neural networks, Nearest neighbor

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

Types of Classification

Decision Theory: Interpretation

Model Building



$IF\ OUTLOOK = Overcast\ THEN\ PLAY = Yes$

ELSE

$IF\ OUTLOOK = Rain\ AND\ WIND = Strong$

$THEN\ PLAY = No$

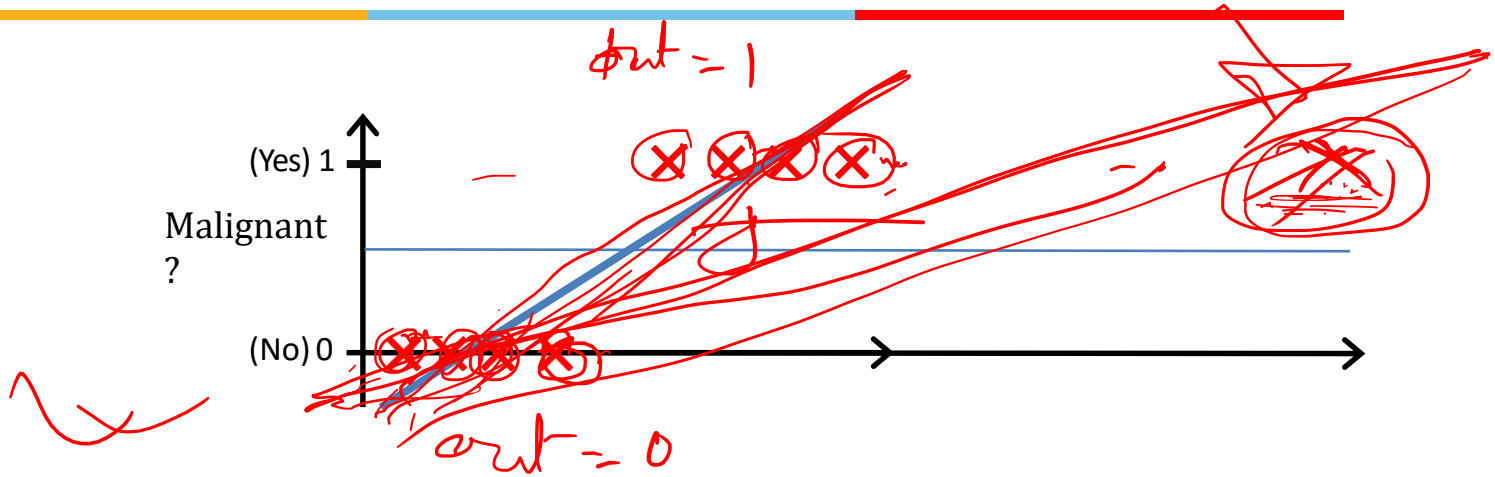
Logistic regression, SVMs , tree based classifiers (e.g. decision tree) Traditional neural networks, Nearest neighbor

Sky	AirTemp	Humidity	Wind	Forecast	Enjoy Sport?
Sunny	Warm	Normal	Strong	Same	Yes
Sunny	Warm	High	Strong	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	Normal	Breeze	Same	Yes
Sunny	Hot	Normal	Breeze	Same	No
Rainy	Cold	High	Strong	Change	No
Sunny	Warm	High	Strong	Change	Yes
Rainy	Warm	Normal	Breeze	Same	Yes

Logistic Regression

BITS Pilani

Logistic Regression vs Least Squares Regression



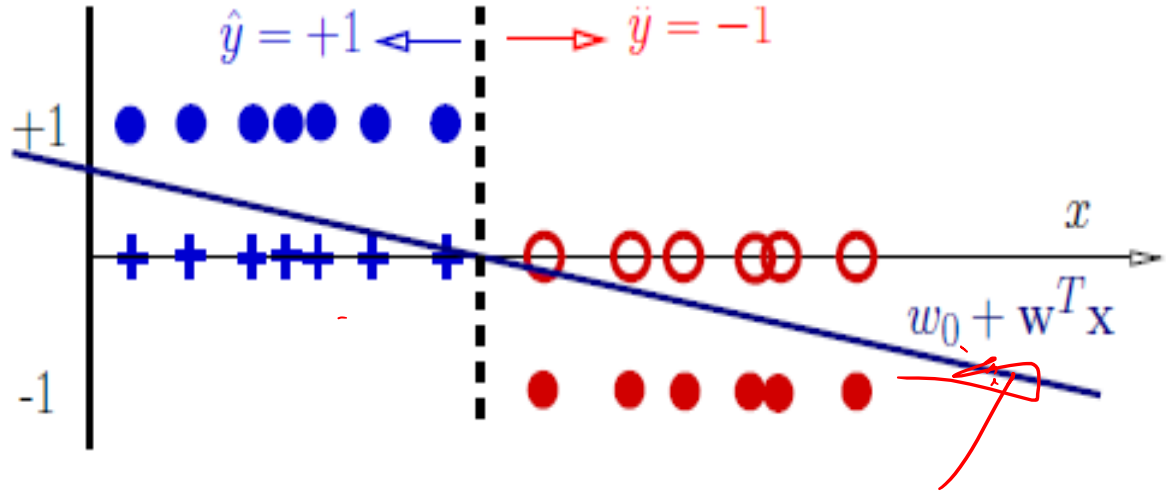
- Tumor Size
- Can we solve the problem using linear regression? E.g., fit a straight line and define a threshold at 0.5
- Threshold classifier output $h_\theta(x)$ at 0.5:

A Discriminant function f (x) directly map input to class labels
In two-class problem, f (.) is binary valued

If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Decision Rules



- Classifier:

  $f(\mathbf{x}, \mathbf{w}) = w_o + \mathbf{w}^T\mathbf{x}$ (linear discriminant function)

- Decision rule is

$$y = \begin{cases} 1 & \text{if } f(\mathbf{x}, \mathbf{w}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

$$f(x) = \frac{1}{1+e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

0.5

- Mathematically

  $y = \text{sign}(w_0 + \mathbf{w}^T\mathbf{x})$

- This specifies a linear classifier: it has a linear boundary (hyperplane)

  $w_0 + \mathbf{w}^T\mathbf{x} = 0$

A discriminant is a function that takes an input vector x and assigns it to one of K classes, denoted C_k.

Learning model parameters

- Training set:

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$$

- m examples

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \qquad x_0 = 1, y \in \{0, 1\}$$

θ

$\theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots$

$h_\theta(x^{(m)}) \approx y^{(m)}$

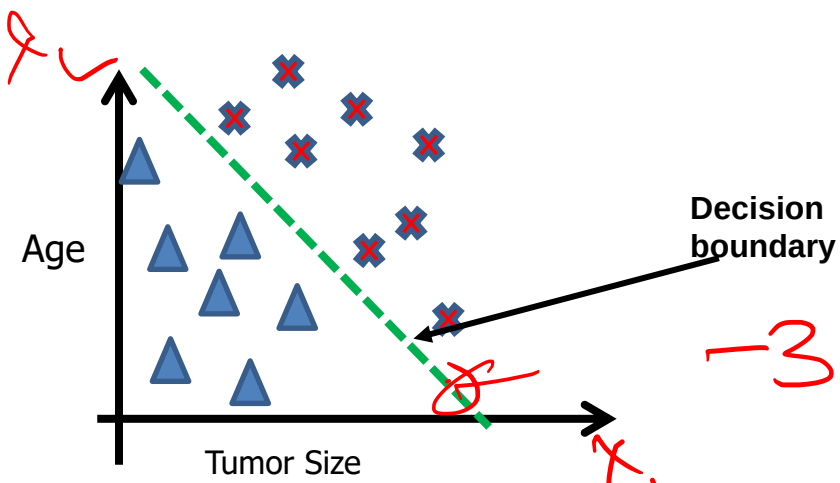
$\approx y^{(m)}$

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

- How to choose parameters (feature weights) ?

Logistic Regression

- At decision boundary output of logistic regression is 0.5
- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
  - e.g., $\theta_0 = -3$, $\theta_1 = 1$, $\theta_2 = 1$

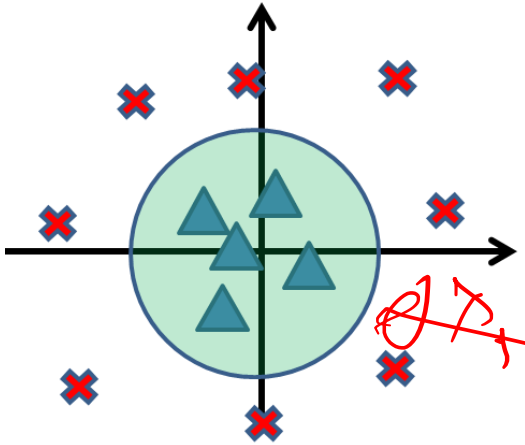


$-3 + \text{TumorSize} + \text{Age} = 0$

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

Slide credit: Andrew Ng

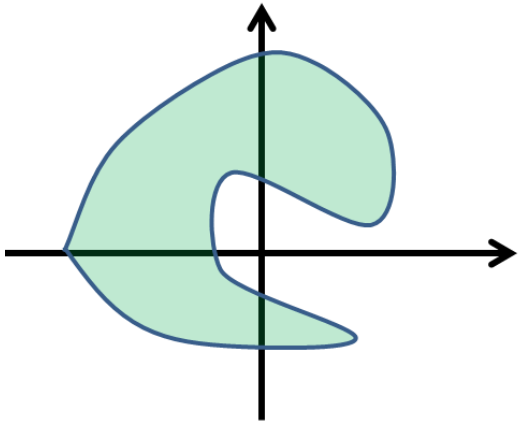
Logistic Regression



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

E.g., $\theta_0 = -1, \theta_1 = 0, \theta_2 = 0, \theta_3 = 1, \theta_4 = 1$

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



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

Slide credit: Andrew Ng

Logistic Regression

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

- m examples
- n features

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \quad x_0 = 1, y \in \{0, 1\}$$

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

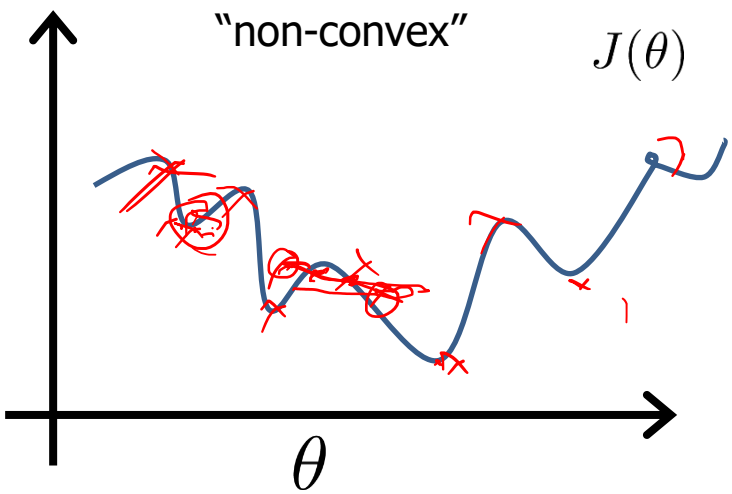
- How to choose parameters (feature weights)? θ

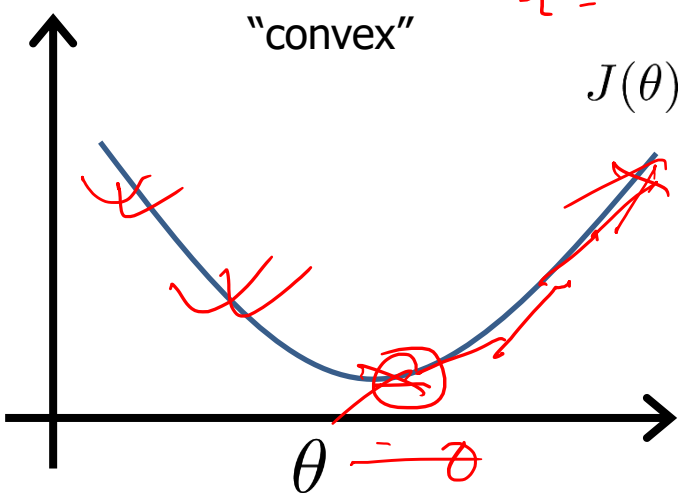
Logistic Regression

$$y_i = \frac{1}{1+e^{-\theta_0-\theta_1 x_1}}$$

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$
- How to choose parameters (feature weights)? $\boldsymbol{\theta}$

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





$\theta_0 = 0$

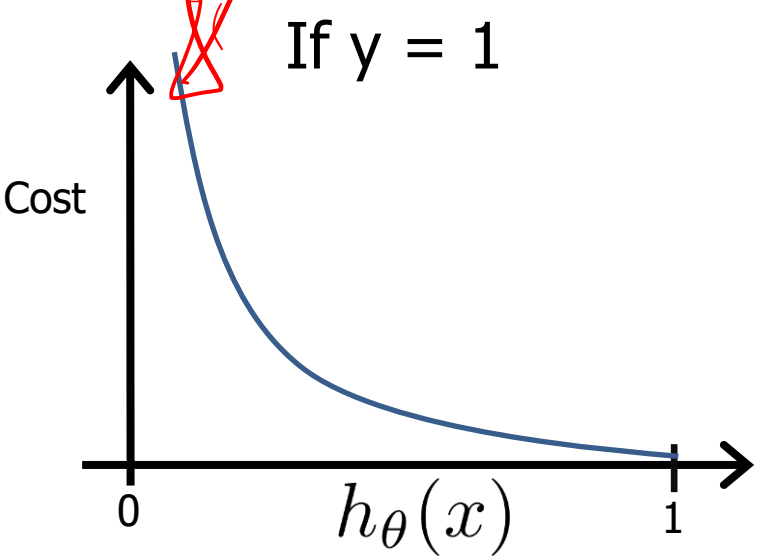
$\theta_1 \to 0$

Log-loss

Logistic regression cost function (cross entropy)

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

$-\log(h_\theta(x))$



Cost $= 0$ if $y = 1, h_\theta(x) = 1$

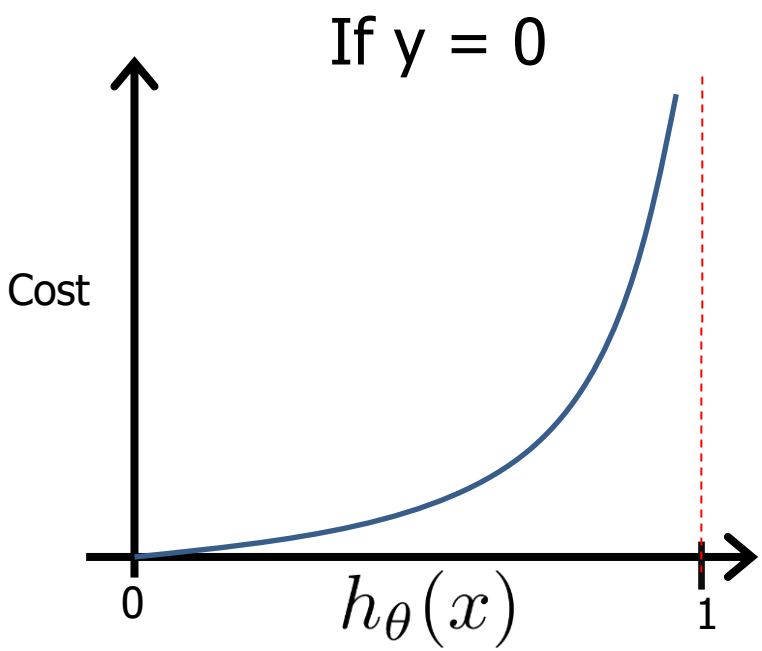
But as $h_\theta(x) \to 0$

$Cost \to \infty$

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

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



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

Cost function

average loss fn. value over training set

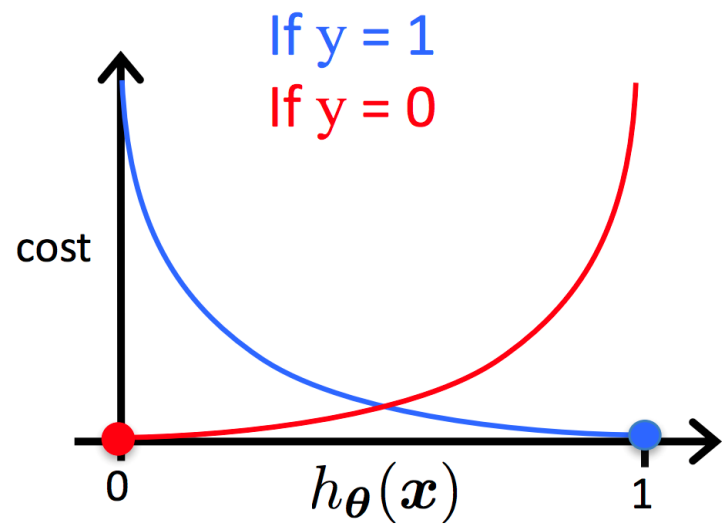
$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

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

To fit parameters θ : Apply Gradient Descent Algorithm $\min_\theta J(\theta)$

To make a prediction given new x :

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



Gradient Descent Algorithm

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

Goal: $\min_{\theta} J(\theta)$

Repeat

{

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

$$\boxed{\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m}\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x_j^{(i)}}$$

target

Gradient Descent Algorithm

$h_\theta(x)$

$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots$

Linear Regression

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

}

$$\boxed{h_\theta(x) = \theta^\top x}$$

Logistic Regression

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

}

$$h_\theta(x) = \frac{1}{1 + e^{-\theta_0 - \theta_1 x_1 - \theta_2 x_2}}$$

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

Slide credit: Andrew Ng

Example: Sentiment Analysis

NLP

$x_2=2$ $x_3=1$

It's hokey. There are virtually no surprises , and the writing is second-rate.
So why was it so enjoyable ? For one thing , the cast is
great . Another nice touch is the music . I was overcome with the urge to get off
the couch and start dancing . It sucked me in , and it'll do the same to you .

$x_1=3$ $x_5=0$ $x_6=4.19$ $x_4=3$

Var	Definition	Value in Fig. 5.2
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(66) = 4.19$

Sentiment Features

$[x_1 \; x_2 \cdots x_6]^T$

Classifying sentiment using logistic regression

θ

Suppose $w = [2.5, -5.0, -1.2, 0.5, 2.0, 0.7]$

$b = 0.1 \rightarrow w_0 = \theta_0$

$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_6 x_6$

$$
\begin{aligned}
p(+|x) = P(Y=1|x) &= \sigma(w \cdot x + b) \\
&= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\
&= \sigma(.833) \\
&= 0.70
\end{aligned}
$$

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

$$
\begin{aligned}
p(-|x) = P(Y=0|x) &= 1 - \sigma(w \cdot x + b) \\
&= 0.30
\end{aligned}
$$

$$\sigma(x) = \sigma(0.833) = \frac{1}{1+e^{-0.833}}$$

$0.7 \approx$

Logistic Regression – Fit a Model

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

$$\theta_0 := \theta_0 - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)(1)$$

$$\theta_{CGPA} := \theta_{CGPA} - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_{CGPA}^{(i)}$$

$$\theta_{IQ} := \theta_{IQ} - 0.3\frac{1}{6}\sum_{i=1}^{6}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_{IQ}^{(i)}$$

Hyper parameters:
Learning Rate = 0.3
Initial Weights = (0.5, 0.5,0.5)
Regularization Constant = 0

$\theta^{T}X$ = 0.5 +0.5 CGPA + 0.5 IQ

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

$$h_\theta(x) = \frac{1}{1 + e^{-(0.5 + 0.5\,CGPA + 0.5\,IQ)}}$$

Approx. : New weights
θ_0 = 0.4
$\theta_{1=CGPA}$ =-0.4
$\theta_{2=IQ} = -0.6$

Logistic Regression – Inference & Interpretation

CGPA	IQ	IQ	Job Offered
5.5	6.7	100	1
5	7	105	0
8	6	90	1
9	7	105	1
6	8	120	0
7.5	7.3	110	0

Assume : 0.4+0.3CGPA-0.45IQ

Predict the Job offered for a candidate : (5, 6)
h(x) = 0.31
Y-Predicted = 0 / No

Note :
The exponential function of the regression coefficient ($e^{w\text{-}cpga}$) is the odds ratio associated with a one-unit increase in the cgpa.
+ The odd of being offered with job increase by a factor of 1.35 for every unit increase in the CGPA [np.exp(model.params)]

Logistic regression (Classification)

- **Model**

$$h_\theta(x) = P(Y = 1|X_1, X_2, \cdots, X_n) = \frac{1}{1+e^{-\theta^\top x}}$$

- **Cost function**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})) \qquad \mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

- **Learning**

Gradient descent: Repeat $\{\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})\, x_j^{(i)}\}$

- **Inference**

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

Note:

- σ(t) < 0.5 when t < 0, and σ(t) ≥ 0.5 when t ≥ 0, so a Logistic model predicts 1 if xTθ is positive, and 0 if it is negative
- logit(p) = log(p / (1 - p)), is the inverse of the logistic function. Indeed, if you compute the logit of the estimated probability p, you will find that the result is t. The logit is also called the log-odds

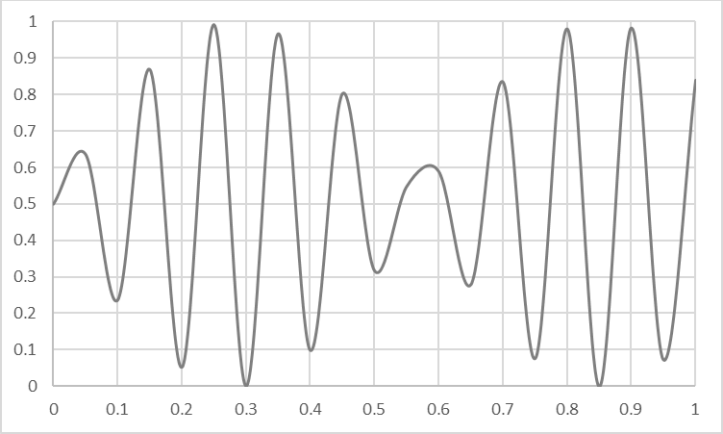
Overfitting vs Underfitting

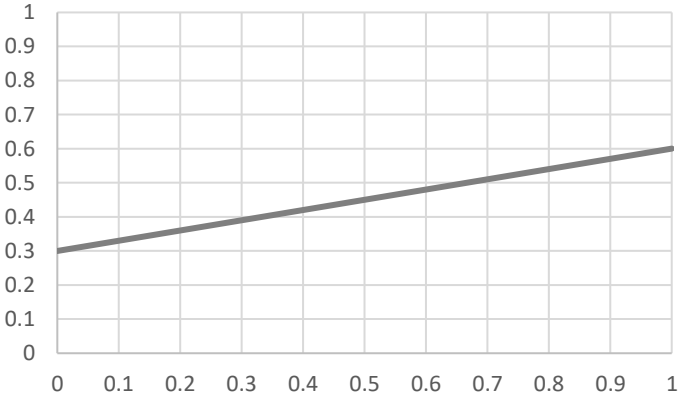
Overfitting

- Fitting the data too well
  - Features are noisy / uncorrelated to concept

Underfitting

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





Regularization

Note : This topic is already covered in the module 3 . Refresher & few more points added here

Regularization

- A method for automatically controlling the complexity of the learned hypothesis
- **Idea**: penalize for large values of θ_j
  - Can incorporate into the cost function
  - Works well when we have a lot of features, each that contributes a bit to predicting the label
- Can also address overfitting by eliminating features (either manually or via model selection)

Ways to Control Overfitting

- Regularization

$$Loss(S) = \sum_{i}^{n} Loss(\hat{y_i}, y_i) \quad + \alpha \sum_{j}^{\#\,Weights} |\theta_j|$$

Note:
The hyperparameter controlling the regularization strength of a Scikit-Learn LogisticRegression model is not alpha (as in other linear models), but its inverse: C. The higher the value of C, the less the model is regularized.

Regularization

- Linear regression objective function

$$J(\boldsymbol{\theta}) = \underbrace{\frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2}_{\text{model fit to data}} + \underbrace{\frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2}_{\text{regularization}}$$

  - λ is the regularization parameter ($\lambda \geq 0$)
  - No regularization on θ_0!

Understanding Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Note that $\sum_{j=1}^{d}\theta_j^2 = \|\boldsymbol{\theta}_{1:d}\|_2^2$
  - This is the magnitude of the feature coefficient vector!

- We can also think of this as:

$$\sum_{j=1}^{d}(\theta_j - 0)^2 = \|\boldsymbol{\theta}_{1:d} - \vec{\mathbf{0}}\|_2^2$$

  - L_2 regularization pulls coefficients toward 0

Regularization

Ridge Regression / Tikhonov regularization

- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{d}\theta_j^2$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Gradient update:

$$\frac{\partial}{\partial\theta_0}J(\theta) \quad \theta_0 \leftarrow \theta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$$\frac{\partial}{\partial\theta_j}J(\theta) \quad \theta_j \leftarrow \theta_j - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} \underbrace{- \lambda\theta_j}_{\text{regularization}}$$

$$\theta_j \leftarrow \theta_j\left(1 - \alpha\lambda\right) - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

Regularization

Lasso Regression (Least Absolute Shrinkage and Selection Operator Regression)

- Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{d}|\theta_j|$$

- Fit by solving $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Gradient update:

$\frac{\partial}{\partial\theta_0}J(\theta)$

$$\theta_0 \leftarrow \theta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)$$

$\frac{\partial}{\partial\theta_j}J(\theta)$

$$\theta_j = \theta_j - \frac{\alpha}{n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} - \alpha\lambda\,\mathrm{sign}(\theta_j)$$

regularization

$$\text{where} \quad \mathrm{sign}(\theta_i) = \begin{cases} -1 & \text{if } \theta_i < 0 \\ 0 & \text{if } \theta_i = 0 \\ +1 & \text{if } \theta_i > 0 \end{cases}$$

Thank you !

Required Reading for completed session :

T1 - Chapter # 6 (Tom M. Mitchell, Machine Learning)

R1 – Chapter # 3,#4 (Christopher M. Bhisop, Pattern Recognition & Machine Learning) & Refresh your MFDS course basics

Next Session Plan :

Module 5 – Decision Tree Classifier

Machine Learning ZG565

Dr. Sugata Ghosal

sugata.ghosal@pilani.bits-pilani.ac.in

BITS Pilani

Pilani Campus

innovate achieve lead

BITS Pilani
Pilani Campus

Lecture No. – 7 | Decision Tree

Date – 01/07/2023

Time: 2 PM – 4 PM

Session Content

- Decision Tree
- Handling overfitting
- Continuous values
- Missing Values

Decision trees

- Decision Trees is one of the most widely used and practical methods of classification
- Method for approximating discrete-valued functions
- Learned functions are represented as decision trees (or if-then-else rules)
- Expressive hypotheses space

Decision Tree

- Advantages:

– Inexpensive to construct

– Extremely fast at classifying unknown records

– Easy to interpret for small-sized trees

– Can easily handle redundant or irrelevant attributes (unless the attributes are interacting)

- Disadvantages:

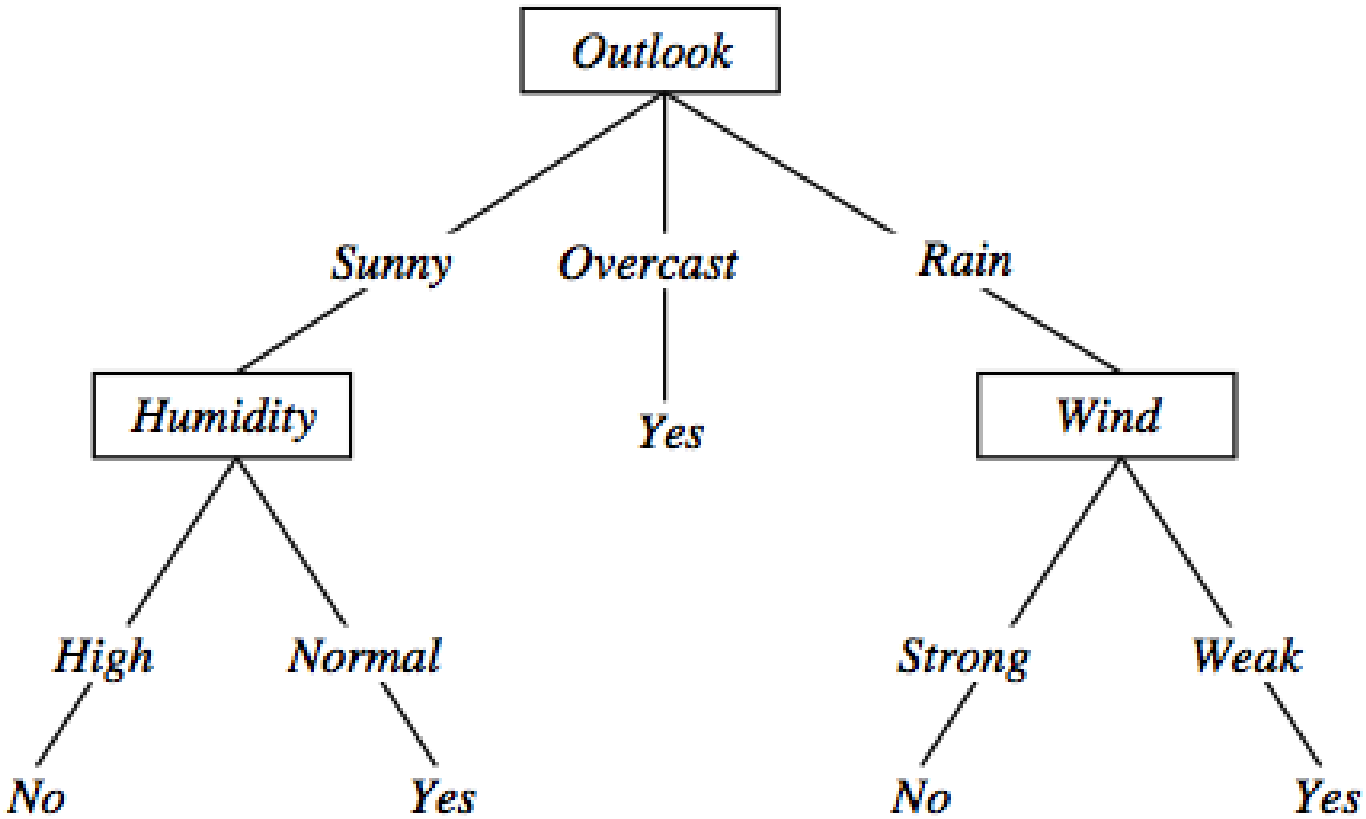
– Space of possible decision trees is exponentially large.

Greedy approaches are often unable to find the best tree.

– Does not take into account interactions between attributes

– Each decision boundary involves only a single attribute

Decision tree representation (PlayTennis)

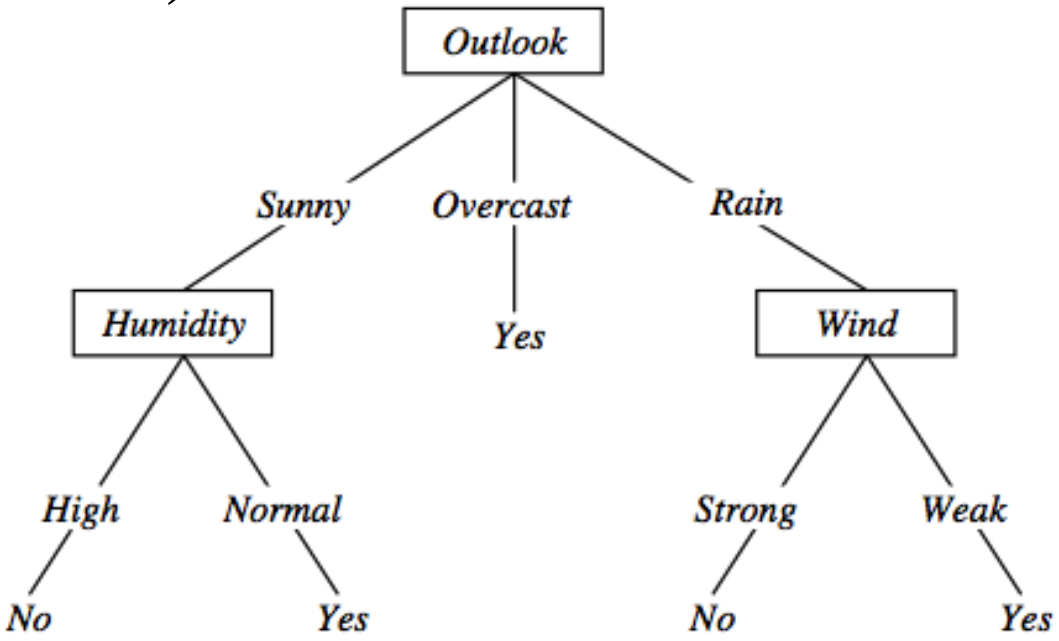


$\langle Outlook=Sunny, Temp=Hot, Humidity=High, Wind=Strong \rangle$ No

Decision trees expressivity

- Decision trees represent a disjunction of conjunctions on constraints on the value of attributes:

$$(Outlook = Sunny \wedge Humidity = Normal) \vee$$
$$(Outlook = Overcast) \vee$$
$$(Outlook = Rain \wedge Wind = Weak)$$



Measure of Information

- The amount of information (surprise element) conveyed by a message is inversely proportional to its probability of occurrence. That is

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

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

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

Entropy

- Entropy of discrete random variable $X=\{x_1, x_2 \ldots x_n\}$

$$\mathrm{H}(X) = \mathrm{E}[\mathrm{I}(X)] = \mathrm{E}[-\log(\mathrm{P}(X))].$$

; since: $\log_2(1/P(event)) = -\log_2 P(event)$

- As uncertainty increases, entropy increases
- Entropy across all values

$$\mathrm{H}(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i)$$

Entropy in general

- Entropy measures the amount of information in a random variable

$$H(X) = -p_+ \log_2 p_+ - p_- \log_2 p_- \qquad X = \{+, -\}$$

for binary classification [two-valued random variable]

$$H(X) = -\sum_{i=1}^{c} p_i \log_2 p_i = \sum_{i=1}^{c} p_i \log_2 1/p_i \qquad X = \{i, \ldots, c\}$$

for classification in *c* classes

Entropy in binary classification

- Entropy measures the *impurity* of a collection of examples. It depends from the distribution of the random variable *p.*
  - S is a collection of training examples
  - p_+ the proportion of positive examples in S
  - p_- the proportion of negative examples in S

$$Entropy\,(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_- \quad [0 \log_2 0 = 0]$$

$$Entropy\,([14+, 0-]) = -14/14 \log_2 (14/14) - 0 \log_2 (0) = 0$$

$$Entropy\,([9+, 5-]) = -9/14 \log_2 (9/14) - 5/14 \log_2 (5/14) = 0.94$$

$$Entropy\,([7+, 7-]) = -7/14 \log_2 (7/14) - 7/14 \log_2 (7/14) =$$

$$= 1/2 + 1/2 = 1 \qquad [\log_2 1/2 = -1]$$

Note: the log of a number < 1 is negative, $0 \le p \le 1$, $0 \le entropy \le 1$

- https://www.easycalculation.com/log-base2-calculator.php

Information gain as entropy reduction

- *Information gain* is the *expected* reduction in entropy caused by partitioning the examples on an attribute.
- The higher the information gain the more effective the attribute in classifying training data.
- Expected reduction in entropy knowing *A*

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|Sv|}{|S|} Entropy(S_v)$$

$Values(A)$ possible values for A
S_v subset of S for which A has value v

Example

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

Example: Information gain

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

- Let
  - $Values(Wind) = \{Weak, Strong\}$
  - $S = [9+, 5-]$
  - $S_{Weak} = [6+, 2-]$
  - $S_{Strong} = [3+, 3-]$
- Information gain due to knowing *Wind*:

$$Gain(S, Wind) = Entropy(S) - 8/14\ Entropy(S_{Weak}) - 6/14\ Entropy(S_{Strong})$$

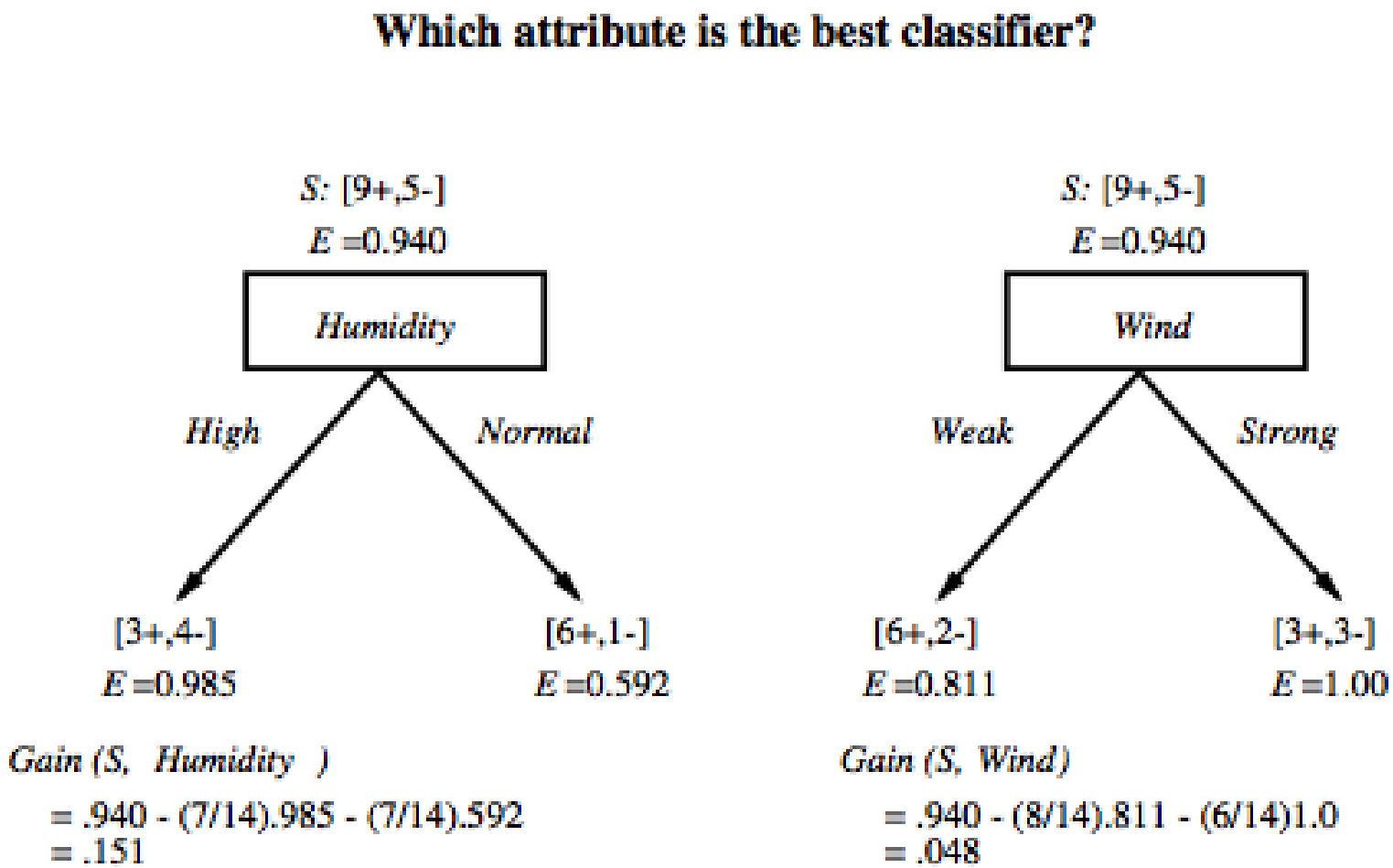
$$= 0.94 - 8/14 \times 0.811 - 6/14 \times 1.00$$

$$= 0.048$$

Example

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

Which attribute is the best classifier?

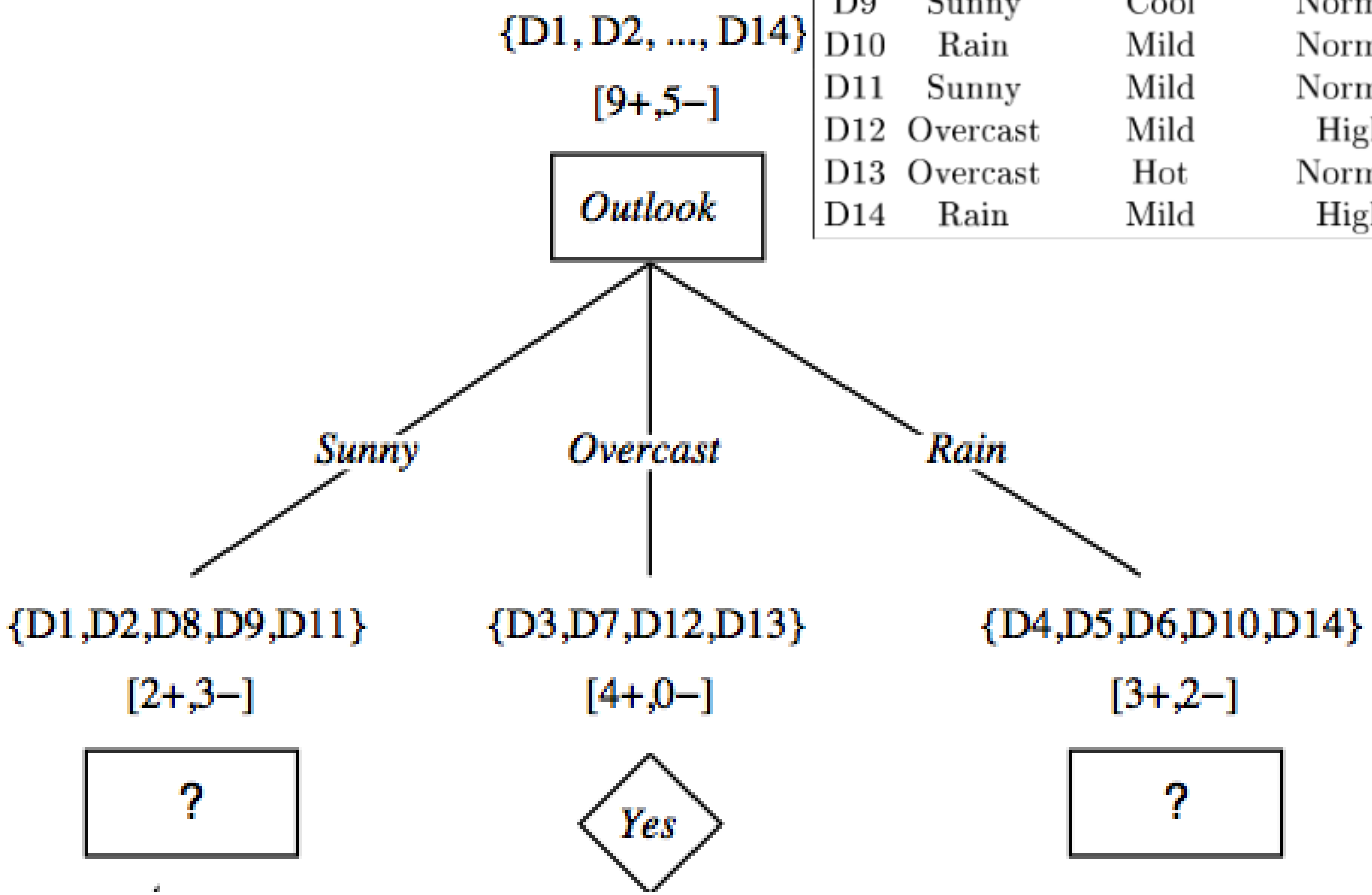


First step: which attribute to test at the root?

- Which attribute should be tested at the root?
  - $Gain(S, Outlook) = 0.246$
  - $Gain(S, Humidity) = 0.151$
  - $Gain(S, Wind) = 0.084$
  - $Gain(S, Temperature) = 0.029$
- *Outlook* provides the best prediction for the target
- Lets grow the tree:
  - add to the tree a successor for each possible value of *Outlook*
  - partition the training samples according to the value of *Outlook*

After first step

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



Second step

- Working on *Outlook=Sunny* node:

  $Gain(S_{Sunny}, Humidity) = 0.970 - 3/5 \times 0.0 - 2/5 \times 0.0 = 0.970$

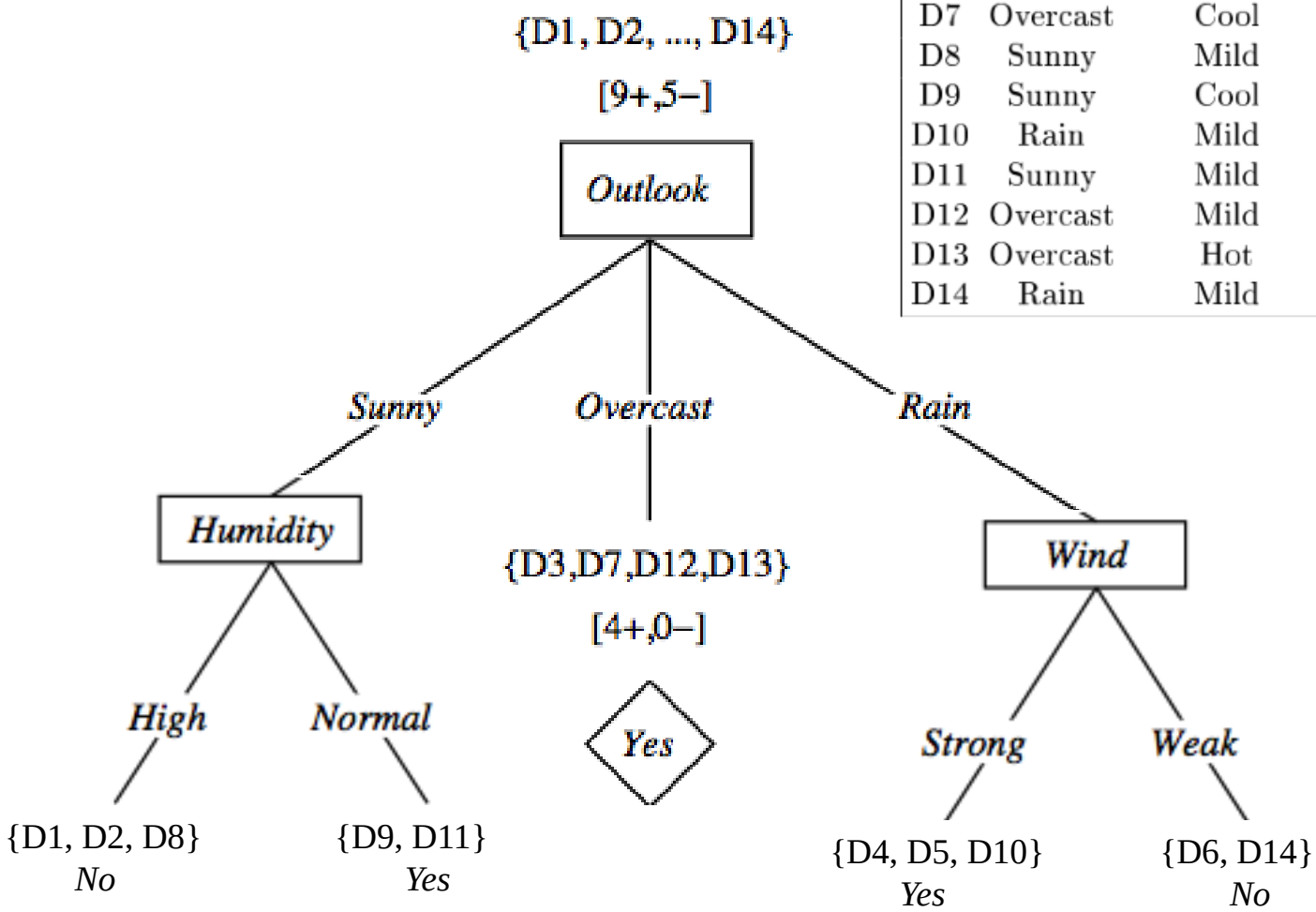
  $Gain(S_{Sunny}, Wind) = 0.970 - 2/5 \times 1.0 - 3.5 \times 0.918 = 0\ .019$

  $Gain(S_{Sunny}, Temp.) = 0.970 - 2/5 \times 0.0 - 2/5 \times 1.0 - 1/5 \times 0.0 = 0.570$

- *Humidity* provides the best prediction for the target
- Lets grow the tree:
  - add to the tree a successor for each possible value of *Humidity*
  - partition the training samples according to the value of *Humidity*

Second and third steps

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



ID3: algorithm

ID3(*X, T, Attrs*) *X*: training examples:
T: target attribute (e.g. *PlayTennis*),
Attrs: other attributes, initially all attributes

Create *Root* node
If all X's are +, *return Root* with class +
If all X's are –, *return Root* with class –
If Attrs is empty *return Root* with class most common value of *T* in *X*
else
  $A \leftarrow$ best attribute; decision attribute for Root $\leftarrow A$
  For each possible value v_i of *A:*
  - add a new branch below *Root,* for test $A = v_i$
  - $X_i \leftarrow$ subset of *X* with $A = v_i$
  - *If* X_i is empty *then* add a new leaf with class the most common value of *T* in *X*
    else add the subtree generated by ID3(X_i, *T, Attrs* – {*A*})
return Root

Prefer shorter hypotheses: Occam's razor

- Why prefer shorter hypotheses?
- Arguments in favor:
  - There are fewer short hypotheses than long ones
  - If a short hypothesis fits data unlikely to be a coincidence
  - Elegance and aesthetics
- Arguments against:
  - Not every short hypothesis is a reasonable one.
- Occam's razor says that when presented with competing hypotheses that make the **same** predictions, one should select the solution which is simple"

Issues in decision trees learning

- Overfitting
  - Reduced error pruning
  - Rule post-pruning
- Extensions
  - Continuous valued attributes
  - Handling training examples with missing attribute values

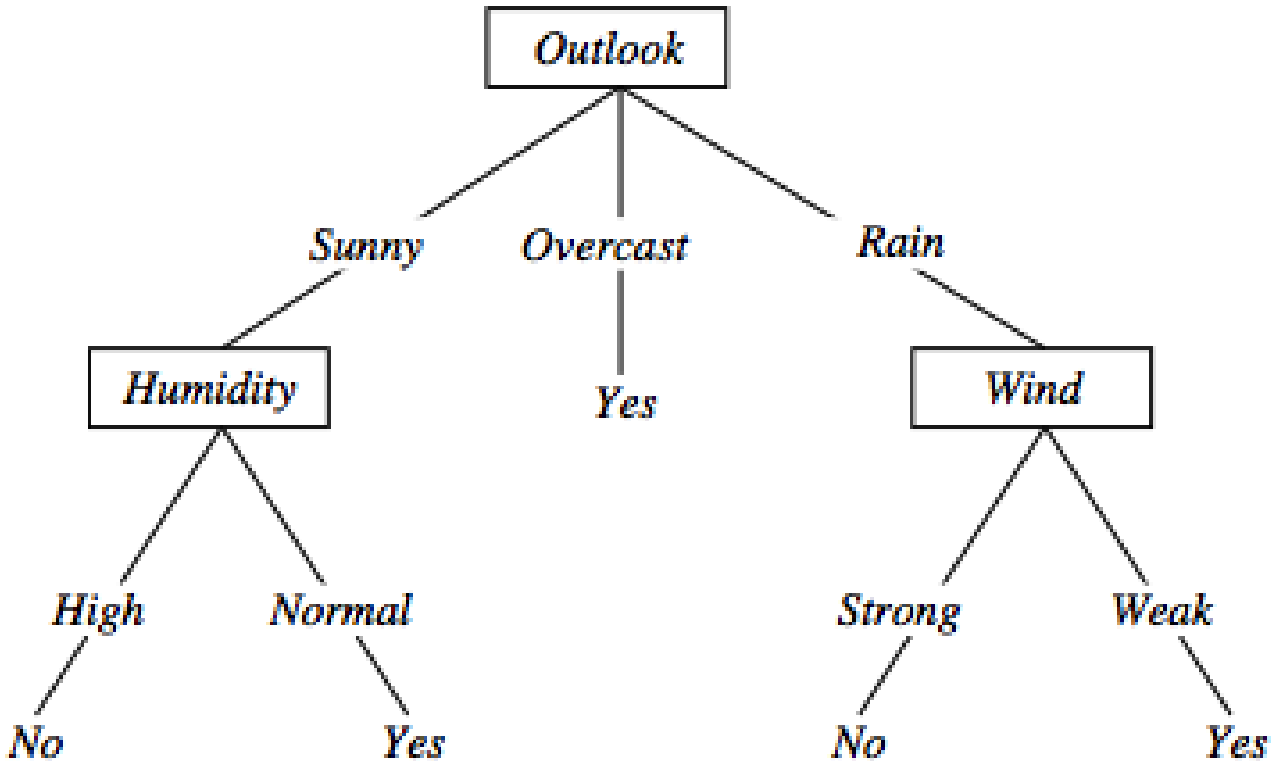
Overfitting: definition

- **overfitting** is "the production of an analysis that corresponds too closely or exactly to a particular set of data
- Building trees that “adapt too much” to the training examples may lead to “overfitting”.
- May therefore fail to fit additional data or predict future observations reliably
- **overfitted model** is a statistical model that contains more parameters than can be justified by the data

Example

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
D15	Sunny	Hot	Normal	Strong	No

Overfitting in decision trees



$\langle Outlook{=}Sunny, Temp{=}Hot, Humidity{=}Normal, Wind{=}Strong, PlayTennis{=}No \rangle$

New noisy example causes splitting of second leaf node.

Avoid overfitting in Decision Trees

- Two strategies:
  1. Stop growing the tree earlier the tree, before perfect classification
  2. Allow the tree to *overfit* the data, and then *post-prune* the tree
- Training and validation set
  - split the training in two parts (training and validation) and use validation to assess the utility of *post-pruning*
    - *Reduced error pruning*
    - *Rule post pruning*

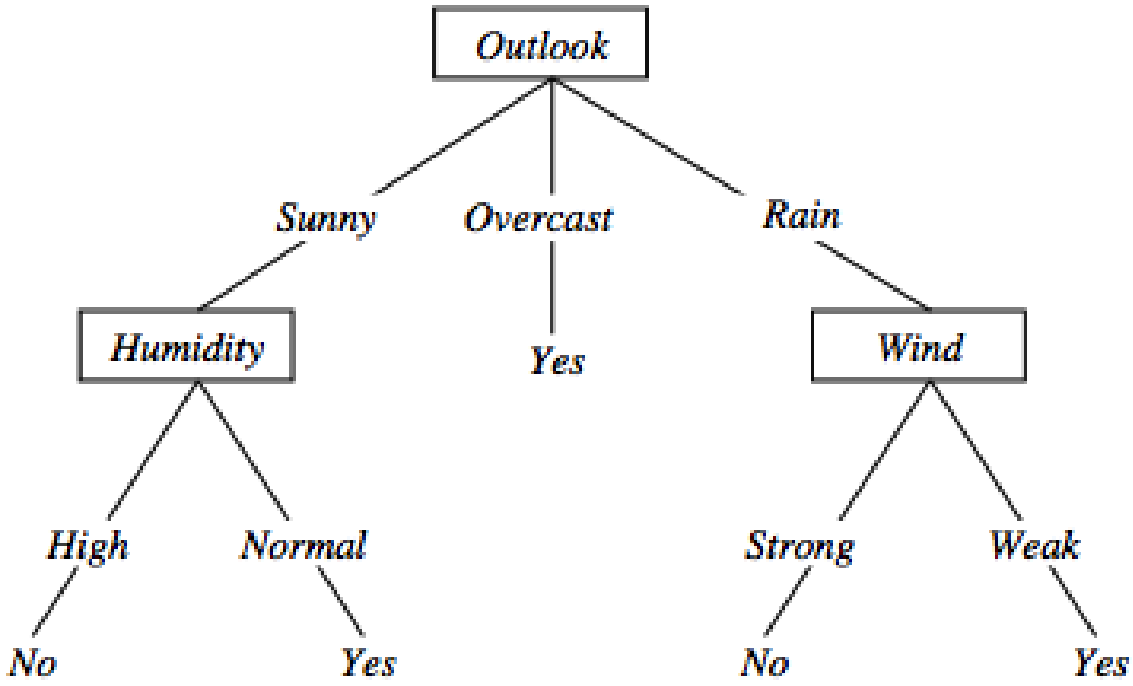
Reduced-error pruning

- Each node is a candidate for pruning
- *Pruning* consists in removing a subtree rooted in a node: the node becomes a leaf and is assigned the most common classification
- Nodes are removed only if the resulting tree performs no worse on the validation set.
- Nodes are pruned iteratively: at each iteration the node whose removal most increases accuracy on the validation set is pruned.
- Pruning stops when no pruning increases accuracy

Rule post-pruning

1. Create the decision tree from the training set
2. Convert the tree into an equivalent set of rules
   - Each path corresponds to a rule
   - Each node along a path corresponds to a pre-condition
   - Each leaf classification to the post-condition
3. Prune (generalize) each rule by removing those preconditions whose removal improves accuracy …
   - … over validation set
4. Sort the rules in estimated order of accuracy, and consider them in sequence when classifying new instances

Converting to rules



$$(Outlook=Sunny)\wedge(Humidity=High) \Rightarrow (PlayTennis=No)$$

Rule Post-Pruning

- Convert tree to rules (one for each path from root to a leaf)
- For each antecedent in a rule, remove it if error rate on validation set does not decrease
- Sort final rule set by accuracy

```
Outlook=sunny ^ humidity=high -> No
Outlook=sunny ^ humidity=normal -> Yes
Outlook=overcast -> Yes
Outlook=rain ^ wind=strong -> No
Outlook=rain ^ wind=weak -> Yes
```

Compare first rule to:

```
Outlook=sunny->No
Humidity=high->No
```

Calculate accuracy of 3 rules based on validation set and pick best version.

Why converting to rules?

- Each distinct path produces a different rule: a condition removal may be based on a local (contextual) criterion. Node pruning is global and affects all the rules
- Provides flexibility of not removing entire node
- In rule form, tests are not ordered and there is no book-keeping involved when conditions (nodes) are removed
- Converting to rules improves readability for humans

Dealing with continuous-valued attributes

- Given a continuous-valued attribute A, dynamically create a new attribute A_c

$$A_c = \text{True } \textit{if } A < c, \text{ False } \textit{otherwise}$$

- How to determine threshold value c ?
- Example. *Temperature* in the *PlayTennis* example
  - Sort the examples according to *Temperature*

Temperature	40	48	|	60	72	80	|	90
PlayTennis	No	No	*54*	Yes	Yes	Yes	*85*	No

  - Determine candidate thresholds by averaging consecutive values where there is a change in classification: (48+60)/2=54 and (80+90)/2=85

Problems with information gain

- Natural bias of information gain: it favors attributes with many possible values.
- Consider the attribute *Date* in the *PlayTennis* example.
  - *Date* would have the highest information gain since it perfectly separates the training data.
  - It would be selected at the root resulting in a very broad tree
  - Very good on the training, this tree would perform poorly in predicting unknown instances. Overfitting.
- The problem is that the partition is too specific, too many small classes are generated.
- We need to look at alternative measures …

An alternative measure: gain ratio

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- S_i are the sets obtained by partitioning on value i of A
- *SplitInformation* measures the entropy of S with respect to the values of A. The more uniformly dispersed the data the higher it is.

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

- *GainRatio* penalizes attributes that split examples in many small classes such as *Date*. Let $|S|=n$, *Date* splits examples in n classes
  - $SplitInformation(S, Date) = -[(1/n \log_2 1/n) + \ldots + (1/n \log_2 1/n)] = -\log_2 1/n = \log_2 n$
- Compare with A, which splits data in two even classes:
  - $SplitInformation(S, A) = -[(1/2 \log_2 1/2) + (1/2 \log_2 1/2)] = -[-1/2 - 1/2] = 1$

Handling missing values training data

- How to cope with the problem that the value of some attribute may be missing?
- The strategy: use other examples to guess attribute
  1. Assign the value that is most common among the training examples at the node
  2. Assign a probability to each value, based on frequencies, and assign values to missing attribute, according to this probability distribution

Applications

Suited for following classification problems:

- Applications whose Instances are represented by attribute-value pairs.
- The target function has discrete output values
- Disjunctive descriptions may be required
- The training data may contain missing attribute values

Real world applications

- Biomedical applications
- Manufacturing
- Banking sector
- Make-Buy decisions

Good References

Decision Tree

- https://www.youtube.com/watch?v=eKD5gxPPeY0&list=PLBv09BD7ez_4temBw7vLA19p3tdQH6FYO&index=1

Overfitting

- https://www.youtube.com/watch?time_continue=1&v=t56Nid85Thg
- https://www.youtube.com/watch?v=y6SpA2Wuyt8

Random Forest

- https://www.stat.berkeley.edu/~breiman/RandomForests/