

CSPE65: Machine Learning Techniques and Practices

Dr. R. Bala Krishnan

Asst. Prof.

Dept. of CSE

NIT, Tiruchirappalli – 620 015

E-Mail: balakrishnan@nitt.edu

Ph: 999 470 4853

Course Content

Course Code	:	CSPE65
Course Title	:	Machine Learning Techniques and Practices
Number of Credits	:	3-0-0-3
Pre-requisites (Course Code)	:	CSPC54
Course Type	:	PE

Course Objectives

- To understand the principles and concepts of machine learning
- To learn the clustering techniques and their utilization in machine learning
- To study the neural network systems for machine learning
- To understand reinforcement and deep learning
- To learn methodology and tools to apply machine learning algorithms to real data and evaluate their performance

Course Contents

UNIT I

Introduction and mathematical preliminaries - Vectors - Inner product - Outer product - Inverse of a matrix - Eigen analysis - Singular value decomposition - Probability distributions - Conditional probability distribution and Joint probability distribution - Bayes theorem - Types of Machine Learning - Supervised Learning - Classification models - Naïve Bayes Classifier - Decision trees - Entropy computation using GINI - Information Gain - Support Vector Machines - non-linear kernels - KNN model - MLP - CART - Ensemble Methods: Bagging - Boosting - Gradient boosting.

UNIT II

Linear models for regression - Maximum Likelihood Estimation (MLS) - least squares - regularized least squares - The Bias-Variance Decomposition - Bayesian Linear Regression - Linear models for classification - Discriminant functions - Fisher's linear discriminant - Probabilistic generative models - Probabilistic discriminative models - Bayesian logistic regression - Bayesian learning - maximum a posterior (MAP) estimation.

Course Content

UNIT III

Clustering - K-Means clustering - Hierarchical Clustering - Mixture of Gaussians - Expectation maximization for mixture models (EM) - Dimensionality Reduction - Principal Component Analysis (PCA) - Linear Discriminant Analysis (LDA).

UNIT IV

Graphical models - Markov random fields - Hidden Markov Models - Representation - learning - Decoding - Inference in graphical models - Monte Carlo models - Sampling.

UNIT V

Reinforcement Learning - Model based - Model Free - Q learning - Introduction to Deep learning.

Course Outcomes

Upon completion of this course, the students will be able to:

- Appreciate the underlying mathematical relationships within and across machine learning algorithms and the paradigms of supervised and un-supervised learning
- Have an understanding of the strengths and weaknesses of machine learning algorithms
- Appreciate machine learning challenges and suggest solutions for the same
- Design and implement various machine learning algorithms in a range of real-world applications
- Suggest supervised / unsupervised machine learning approaches for any application

Books

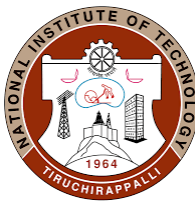
- **Text Books (TB)**

1. Tom Mitchell, *"Machine Learning"*, McGraw Hill, 1997
2. E. Alpaydin, *"Introduction to Machine Learning"*, Second Edition, Prentice-Hall of India, 2010

- **Reference Books (RB)**

1. Simon Haykin, *"Neural Networks and Learning Machines"*, Pearson, 2008
2. Shai Shalev-Shwartz, Shai Ben-David, *"Understanding Machine Learning from Theory to Algorithms"*, Cambridge University Press, 2014
3. R.O. Duda, P.E. Hart, D.G. Stork, *"Pattern Classification"*, Second Edition, Wiley-Interscience, 2000
4. T. Hastie, R. Tibshirani, J. Friedman, *"The Elements of Statistical Learning"*, Springer, 2011

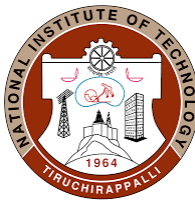
Books & Chapters



Unit	Book	Chapter
I		
II		
II_		
III		
IV		
V		

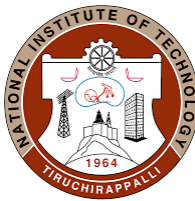
Evaluation Method

Sl. No.	Mode of Assessment	Week / Date	Duration	% Weightage
1.	Cycle Test 1	As per Academic Schedule (<i>Feb 3rd Week</i>)	1 hour	20
2.	Cycle Test 2	As per Academic Schedule (<i>Mar 4th Week</i>)	1 hour	20
3.	Assignment 1	Feb 4 th Week	-	15
4.	Assignment 2	Mar 5 th Week	-	15
5.	Compensation Assessment	As per Academic Schedule	1 hour	20
6.	Final Assessment	As per Academic Schedule (<i>May 5th Onwards</i>)	2 hour	30



Related Courses

- Data Science
- Artificial Intelligence
- Data Mining
- Big Data Mining
- Big Data Analytics



Time Table

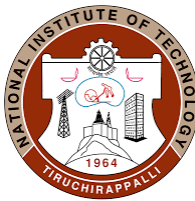
Programme: B.Tech.

Class: 3rd year

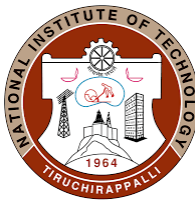
Section: A, B

PAC Chairperson: Dr. R. Leela Velusamy

DAY / TIME	1		2		3		4		5	6	7	8
	08:30 to 09:20	BREAK	09:30 to 10:20	BREAK	10:30 to 11:20	BREAK	11:30 to 12:20	LUNCH BREAK	01:30 to 02:20	02.30 to 03.20	03.30 to 04:20	04:20 to 05:10
Monday												
Tuesday			I									
Wednesday							I					
Thursday												
Friday	I											



Thank You



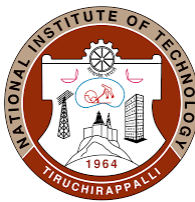
Introduction to Machine Learning

- Facilitate machines to learn-like humans
- Think- and act-like humans
- Applications
 - Automation
 - Decision-Making
 - ✓ Classify
 - ✓ Cluster
 - ✓ Predict Future

How do Humans Learn?

- Listen to Elders / Teachers / Neighbours
- Learn through Experience
- Read Books
- Watch Television
- Watch other's Activity and Learn

How do Humans Detect Objects?



**Human Visionary
System (*Eyes*)**

→ Properties (Height, Weight, Colour, Appearance)

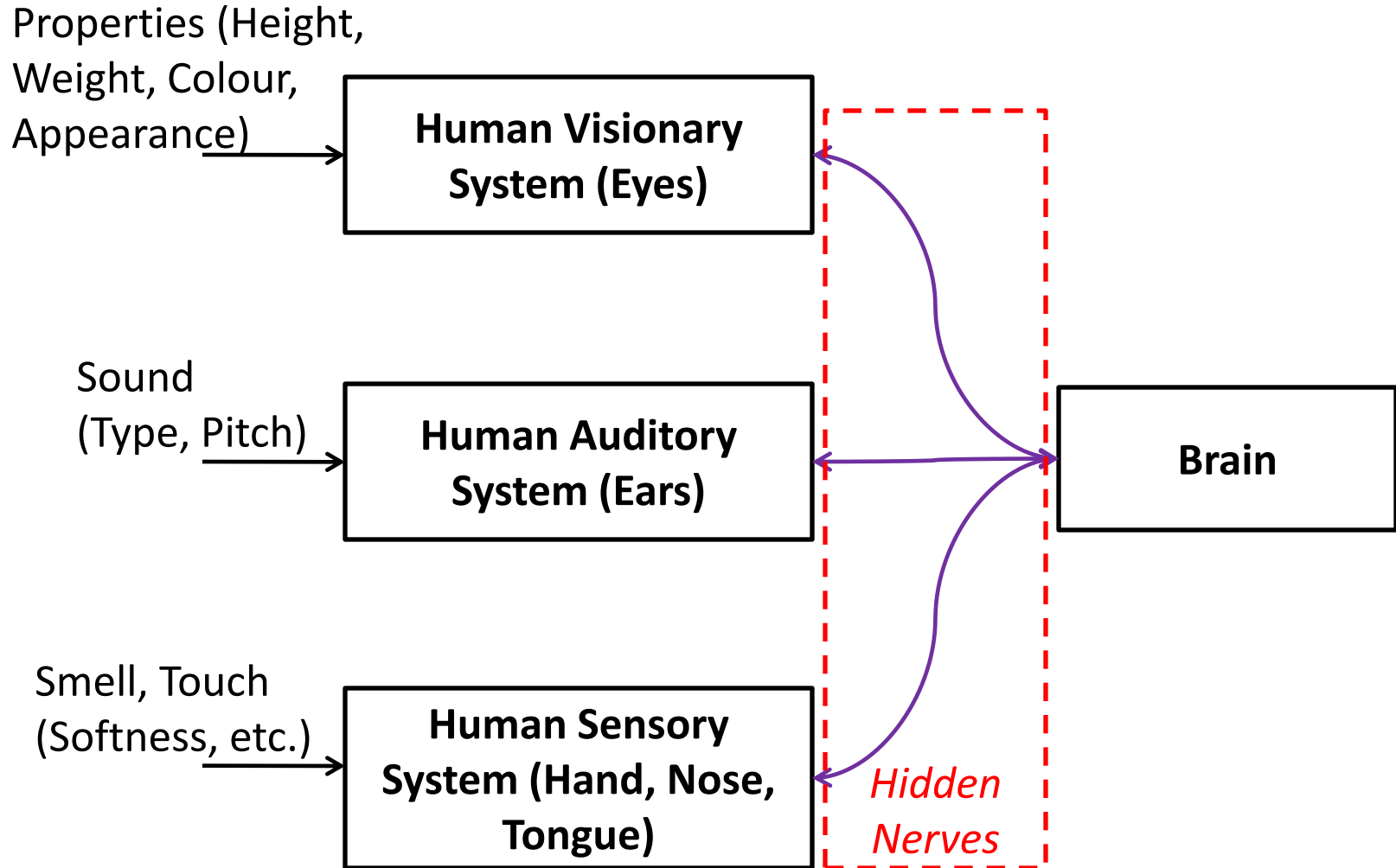
**Human Auditory
System (*Ears*)**

→ Sound (Type, Pitch)

**Human Sensory
System (*Hand, Nose,
Tongue*)**

→ Smell, Touch (Softness, etc.)

Detecting Methodology



Information Gathering



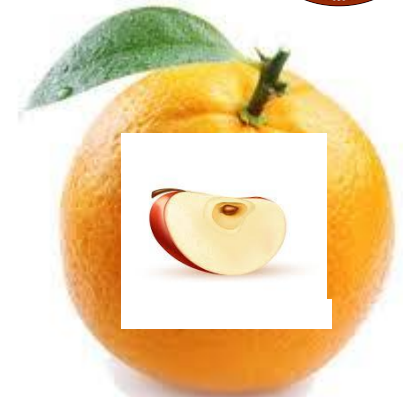
Apple
(Eye)



Orange
(Eye)



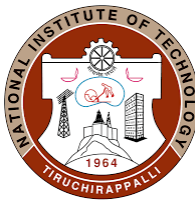
Smell + Taste
Orange
(Nose + Tongue)



Smell + Taste
Apple
(Nose + Tongue)

ID	Outer Colour	Inner Colour	Target
1	Red	White	Apple
2	Orange	Orange	Orange
3	Red	Orange	Orange
4	Red	Orange	Apple
5	Orange	White	Apple
6	Orange	White	Orange

Possible Problems in Dataset



ID	Outer Colour	Inner Colour	Target
1	Red	White	Apple
2	Orange	Orange	Orange
3	Red	Orange	Orange
4	Red	Orange	Orange
5	Red	Orange	Orange
6	Red	Orange	Orange
7	Red	Orange	Orange
8	Red	Orange	Apple
9	Orange	White	Apple
10	Orange	White	Orange

Terminologies Involved in Dataset

Features / Attributes

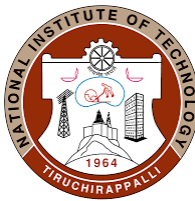
Sample	Features / Attributes			Target
	ID	Outer Colour	Inner Colour	
Training Dataset	1	Red	White	Apple
	2	Orange	Orange	Orange
	3	Red	Orange	Orange
	4	Red	Orange	Apple
Testing Dataset	5	Orange	White	Apple
	6	Orange	White	Orange

Types of

Learning

- Supervised
- Unsupervised
- Semi-Supervised

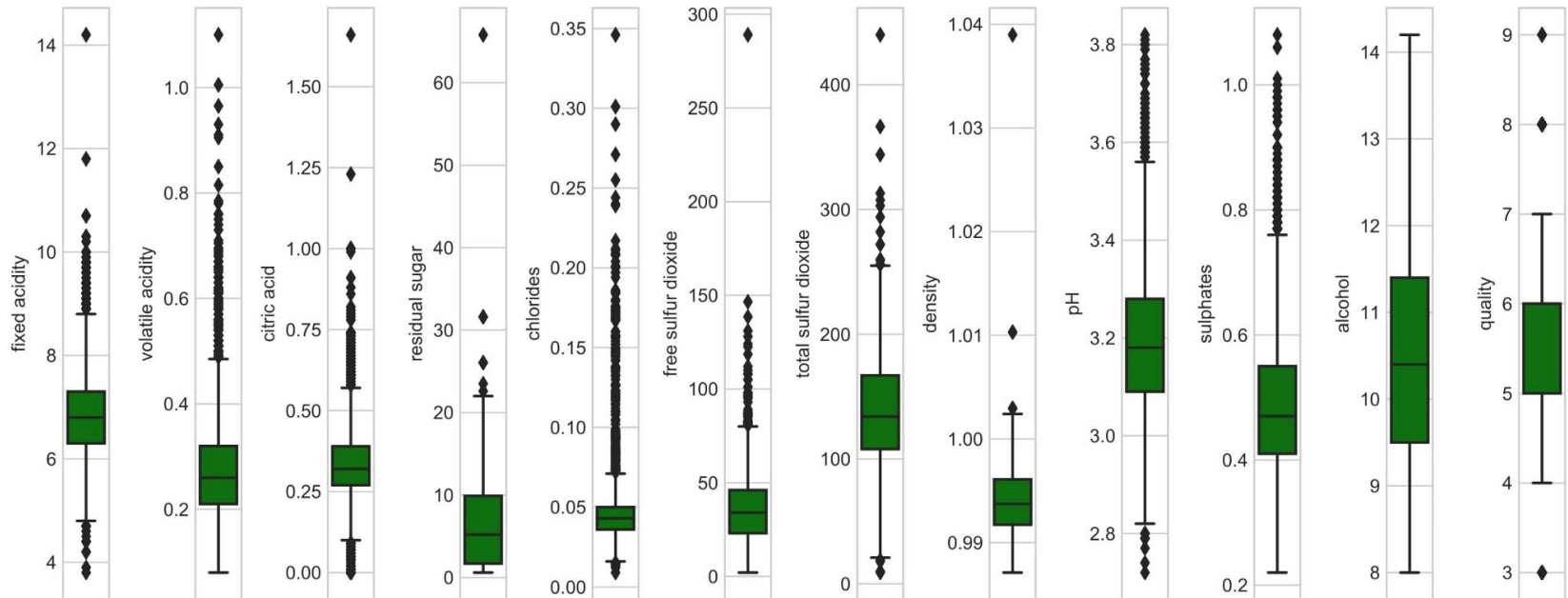
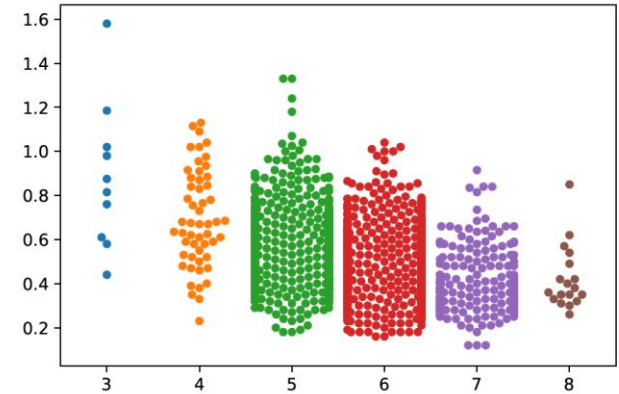
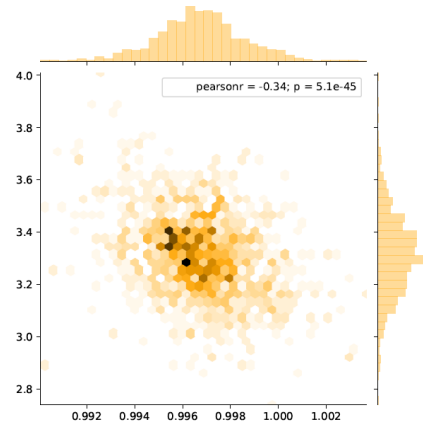
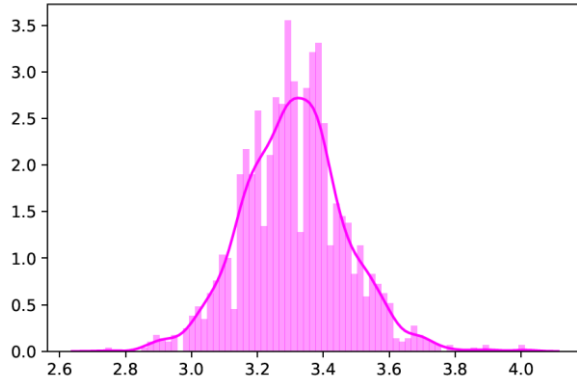
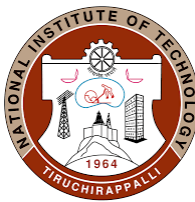
Data Preprocessing Steps



- Handling NULL / Missing Values
- Feature Selection (Pearson Correlation, Spearman Correlation, etc.)
- Feature Scaling or Normalization or Standardization (Zscore, MinMax, etc.)
- Feature Encoding (One Hot Encoding, Dummy Encoding, etc.)
- Feature Engineering
- Feature Binning (Bin by Mean, Bin by Median, etc.)
- Dimensionality Reduction (Principal Component Analysis)
- Class Imbalance (SMOTE Algorithm)

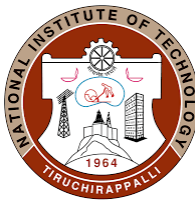
Exploratory Data Analytics

(Understand the Properties of Data)

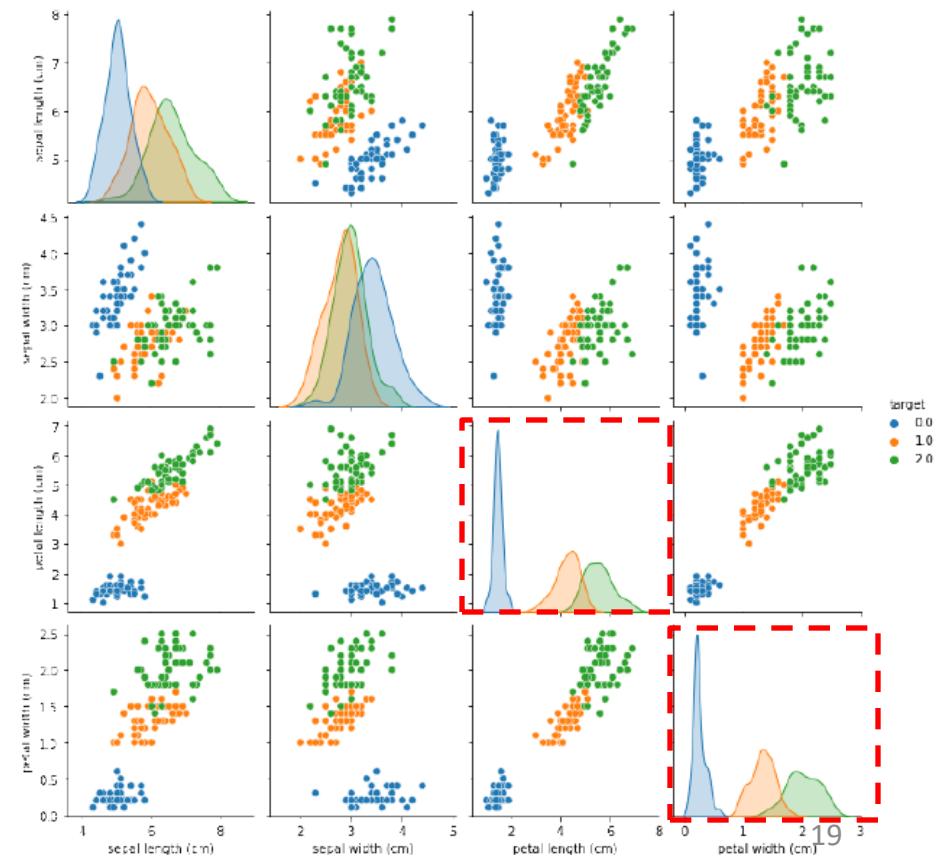
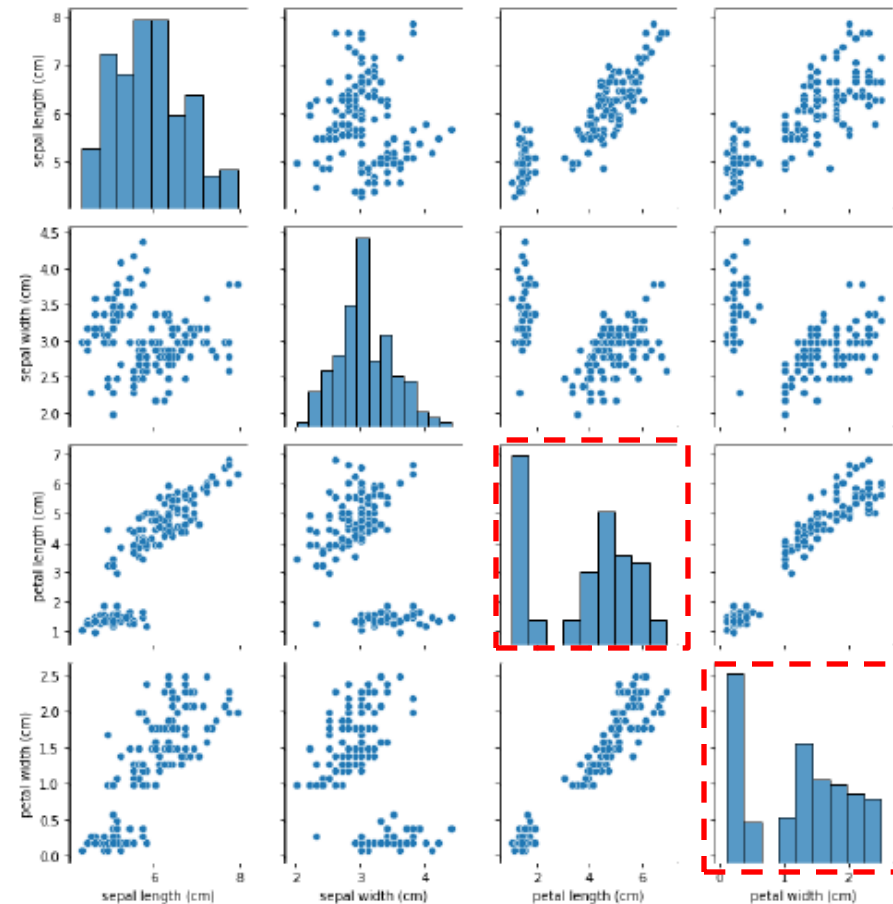


Exploratory Data Analytics

(Understand the Properties of Data)

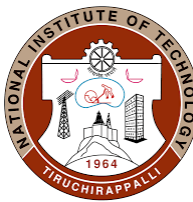


Sepal Length (in cm)	Sepal Width (in cm)	Petal Length (in cm)	Petal Width (in cm)	Target
5.5	6.5	3.5	2.5	Setosa
1.2	4.2	3.5	2.8	Versicolor
3.8	1.5	2.8	3	Virginica

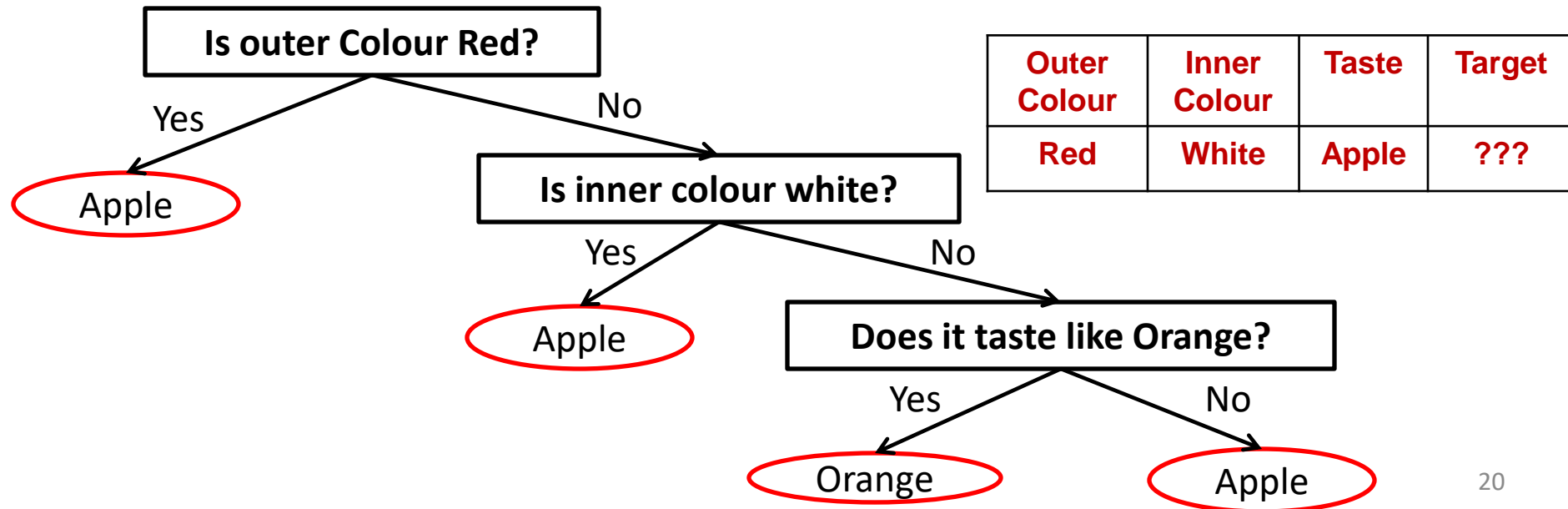


Classification – Decision Tree Algorithm

(Supervised Learning)

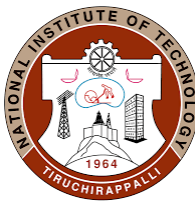


ID	Outer Colour	Inner Colour	Taste	Target
1	Red	White	Apple	Apple
2	Orange	Orange	Orange	Orange
3	Red	Orange	Orange	Orange
4	Red	Orange	Apple	Apple
5	Orange	White	Apple	Apple
6	Orange	White	Orange	Orange

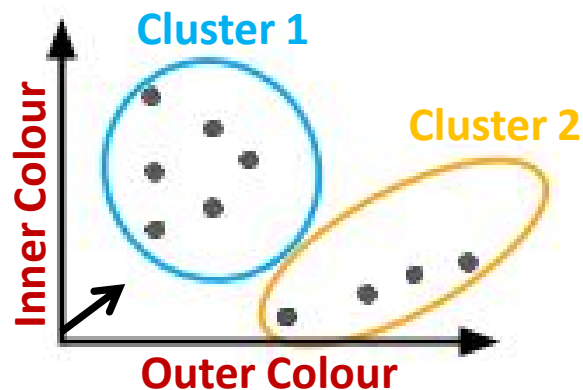


Clustering – K-Means Algorithm

(Unsupervised Learning)



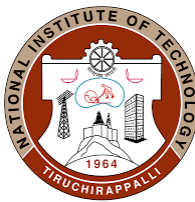
ID	Outer Colour	Inner Colour	Taste
1	Red	White	Apple
2	Orange	Orange	Orange
3	Red	Orange	Orange
4	Red	Orange	Apple
5	Orange	White	Apple
6	Orange	White	Orange



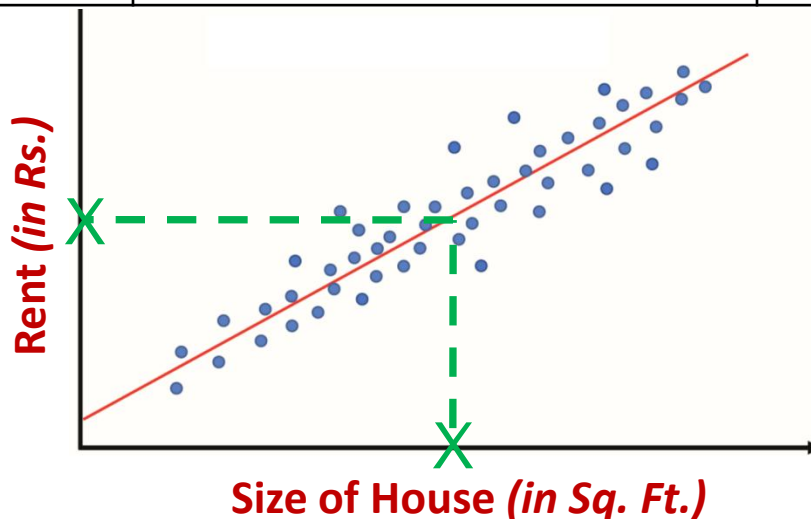
Outer Colour	Inner Colour	Taste	Cluster
Red	White	Apple	???

Regression – Linear Regression Algorithm

(Supervised Learning)



ID	Size of House (in Sq. Ft.)	No. of Rooms	Rent (in Rs.)
1	1200	5	12,000
2	1500	4	15,000
3	1600	3	10,000
4	1000	6	11,000
5	1300	4	11,500
6	1250	3	12,000



Size of House (in Sq. Ft.)	No. of Rooms	Rent (in Rs.)
1400	3	???

Evaluation Metrics – Confusion Matrix

- True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN)
- Classification system trained to distinguish between cats and dogs
- Sample of 13 animals → 8 cats and 5 dogs

Actual = [1,1,1,1,1,1,1, 0,0,0,0,0]

Prediction = [0,0,0,1,1,1,1, 0,0,0,1,1]

Objective:

Detect Cat (Positive Class)

	Actual Class		
		Cat	Dog
	Predicted Class		
	Cat	5	2
	Dog	3	3

	Actual Class		
		Cat	Non-cat
	Predicted Class		
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN

Evaluation Metrics – Accuracy

- **Accuracy:** Measure of the effectiveness of the algorithm

$$\text{Accuracy} = \frac{\text{No. of Correct Predictions}}{\text{Total No. of Predictions Made}}$$

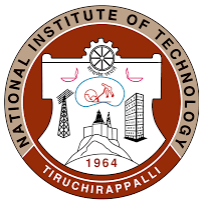
Precision, Recall, F1 Score

$$\text{Accuracy(ACC)} = \frac{\sum TP + \sum TN}{\sum TP + TN + FP + FN} = \frac{5 + 3}{5 + 3 + 2 + 3} = 61.5\%$$

	Actual Class		
		Cat	Dog
	Predicted Class		
	Cat	5	2
	Dog	3	3

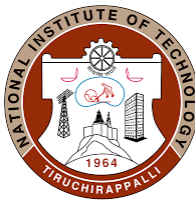
	Actual Class		
		Cat	Non-cat
	Predicted Class		
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN

- **Demerits:** Accuracy will yield misleading results if the data set is unbalanced
- **Eg:** If there were 95 cats and only 5 dogs in the data, a particular classifier might classify all the observations as cats
 - Overall accuracy would be 95%, but in more detail the classifier would have a 100% recognition rate (sensitivity) for the cat class but a 0% recognition rate for the dog class



Thank You

Other Problems (Data Preprocessing)



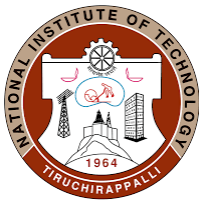
#	Cushion Color	Cushion Diameter (in cm)	Cushion Length (in cm)	Cushion Area (in mm)	Cap Color	Cap diameter (in cm)	Cap Length (in cm)	Tail Color	Tail Length (in cm)	Target
1	Black	0.8	3	2.4	Black	1	4	Black	3	Black
2	Black	0.8	3	2.4	Black	1	4	Black	3	Black
3	Blue	0.8	3	2.4	Blue		4	Blue	3	Blue
4	Blue	0.8%	3	2.4	Blue	1	4	Blue	3	Blue
5	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue
6	Black	0.8	3	2.4	Black	1	4.	Black	3	Black
7	Black	0.8	3	2.4	Black	NULL	4	Black	3	Black
8	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue
9	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue

Dataset Preparation Steps

- Handle all typos in dataset
- Reduce the number of features (Feature Selection)
- Scale the values in each feature (Feature Normalization or Standardization)
- Create new features from existing ones (Feature Engineering)
- Represent the dataset in a lesser dimensional space (Dimensionality Reduction)

Terminologies Learnt

- Data Preprocessing
- Feature Selection
- Feature Normalization or Standardization
- Feature Engineering
- Dimensionality Reduction
- Handle Class Imbalance
- Model Building
 - Training Phase
 - Testing Phase
- Prediction Phase



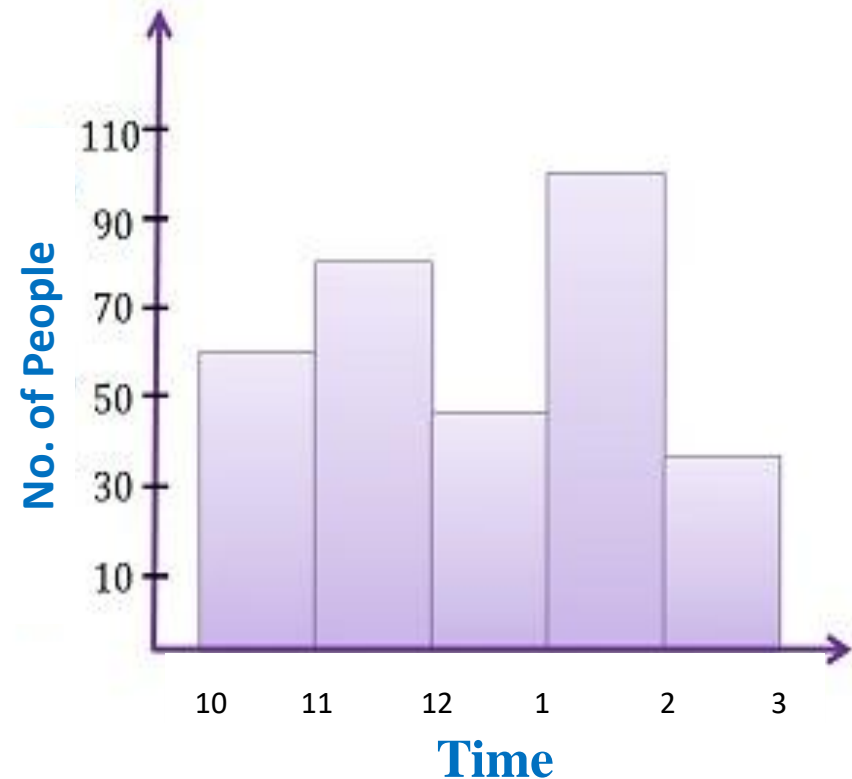
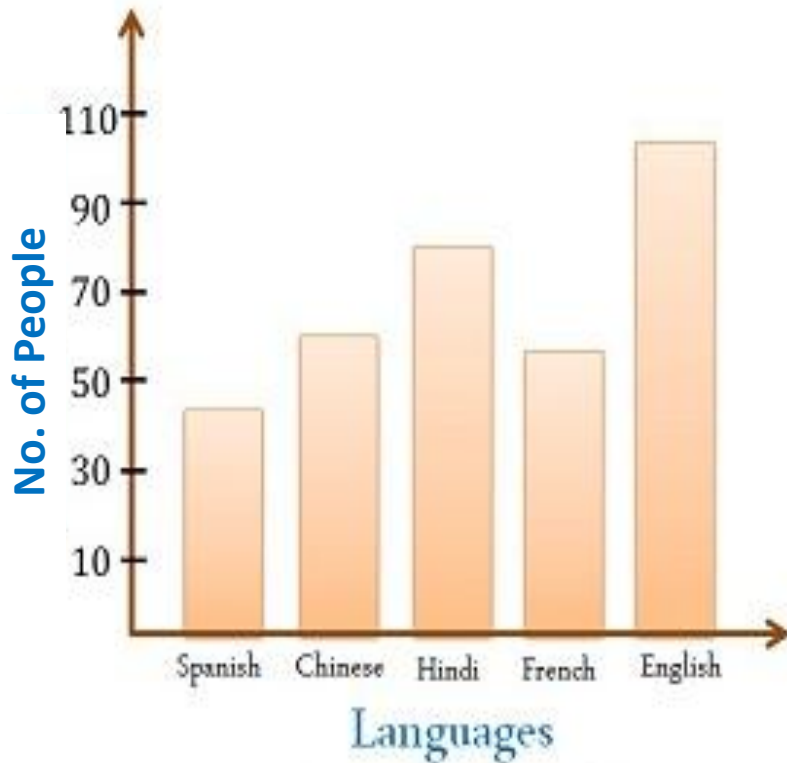
Data Visualization

Plots

- Pie Chart
- Bar Chart
- Histogram
- Box-and-Whisker Plot
- Joint Plot
- Pair Plot

Bar Chart vs Histogram

Visitors to Museum

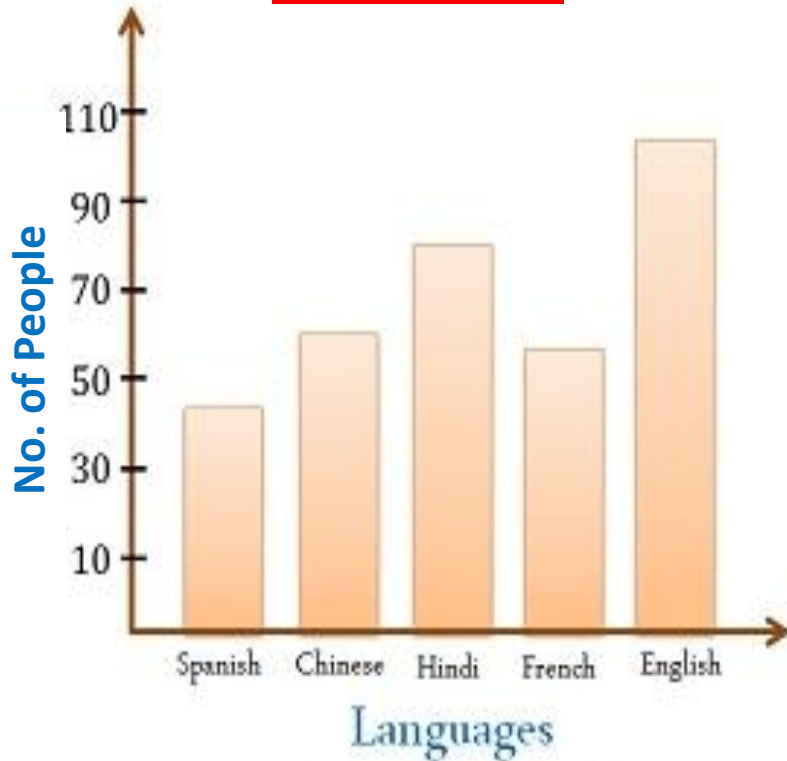


Sl. No.	Name	In Time	Out Time	Language
1.	Bala	10:00 AM	10:55 AM	English

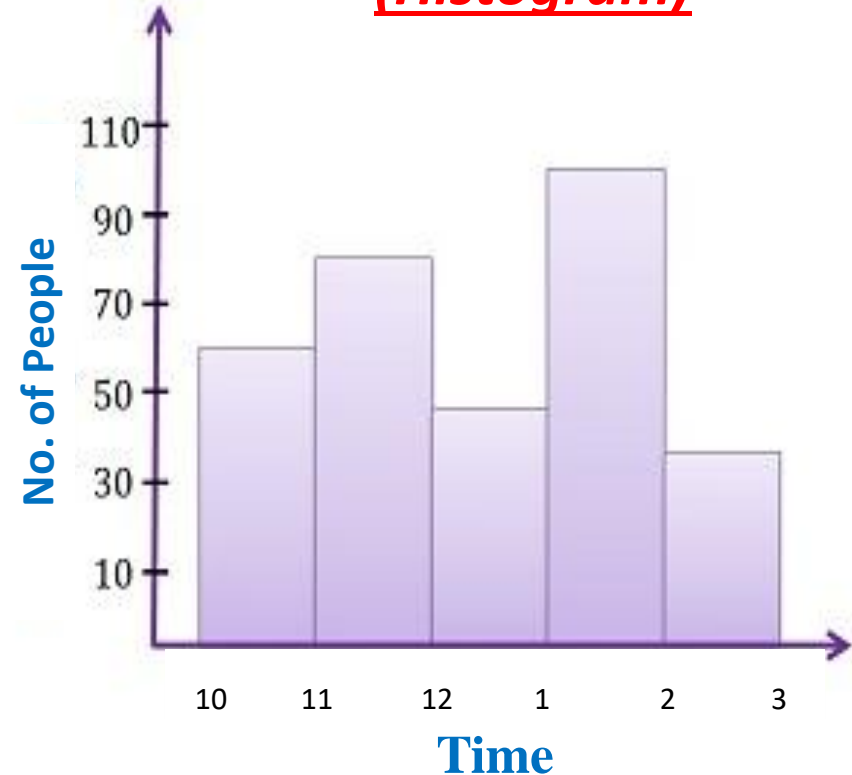
Bar Chart vs Histogram

Visitors to Museum

Discrete
(Bar Chart)



Continuous
(Histogram)



Sl. No.	Name	In Time	Out Time	Language
1.	Bala	10:00 AM	10:55 AM	English

Box-and-Whisker Plot

Values = 7, 3, 14, 9, 7, 8, 12

Ascending = 3, 7, 7, 8, 9, 12, 14

Q1

Q2

Q3

1. 1st Quartile = 7

2. 2nd Quartile = 8

3. 3rd Quartile = 12

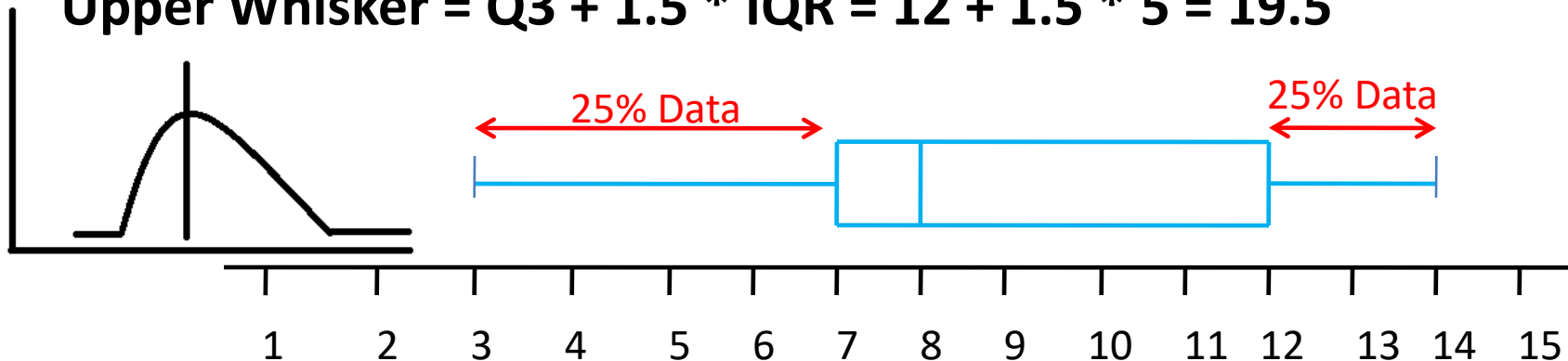
4. Smallest # = 3

5. Largest # = 14

Inter-Quartile Range (IQR) = $Q3 - Q1 = 12 - 7 = 5$

Lower Whisker = $Q1 - 1.5 * IQR = 7 - 1.5 * 5 = 0.5$

Upper Whisker = $Q3 + 1.5 * IQR = 12 + 1.5 * 5 = 19.5$



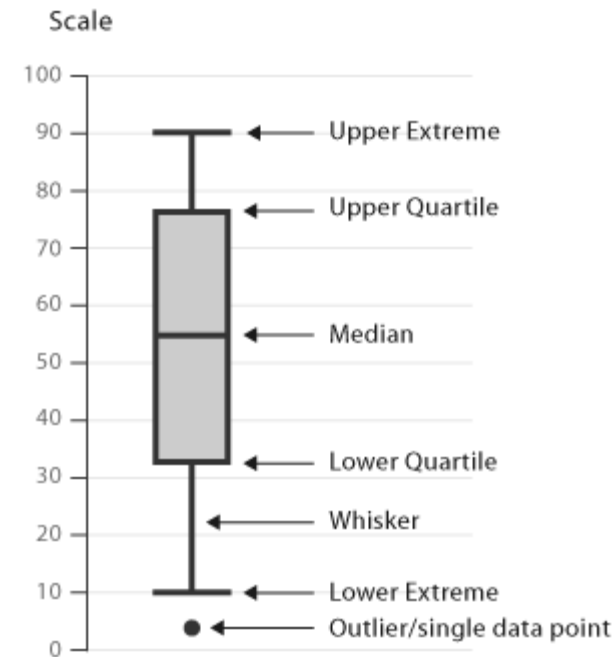
Positively-skewed or Right-skewed: Mean > Median;

Negatively-skewed or Left-skewed: Mean < Median

Key Observations from Box-and-Whisker Plot



- Distribution of the data → Range
- What the key values are, such as: the average, median, 25th percentile, etc.
- If there are any outliers and what their values are
- Is the data symmetrical
- How tightly is the data grouped
- If the data is skewed and if so, in what direction



https://datavizcatalogue.com/methods/box_plot.html

Box-and-Whisker Plot

Original Value = 2, 51, 51, 62, 53, 49, 54, 50, 43, 63, 60

Ascending Order = 2, 43, 49, 50, 51, 51, 53, 54, 60, 62, 63

↓
Q1

↓
Q2

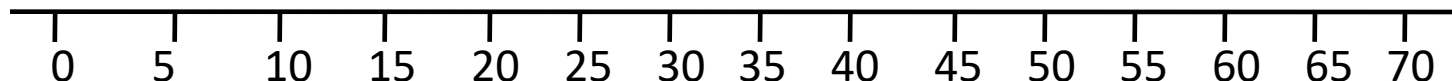
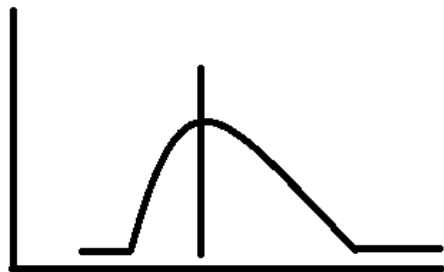
↓
Q3

$$\text{IQR} = 60 - 49 = 11$$

$$\text{Lower Whisker} = Q1 - 1.5 * \text{IQR} = 49 - 1.5 * 11 = 32.5$$

$$\text{Upper Whisker} = Q3 + 1.5 * \text{IQR} = 60 + 1.5 * 11 = 76.5$$

1. 1st Quartile = 49
2. 2nd Quartile = 51
3. 3rd Quartile = 60
4. Smallest # = 2 43
5. Largest # = 63



Box-and-Whisker Plot

Original Value = 2, 51, 52, 62, 53, 49, 54, 50, 43, 63, 60, 45

Ascending Order = 2, 43, 45, 49, 50, 51, 52, 53, 54, 60, 62, 63

$$Q1 = 94 / 2 \quad Q2 = 103 / 2 \quad Q3 = 114 / 2$$

$$IQR = 57 - 47 = 10$$

$$\text{Lower Whisker} = Q1 - 1.5 * IQR = 47 - 1.5 * 10 = 32$$

$$\text{Upper Whisker} = Q3 + 1.5 * IQR = 57 + 1.5 * 10 = 72$$

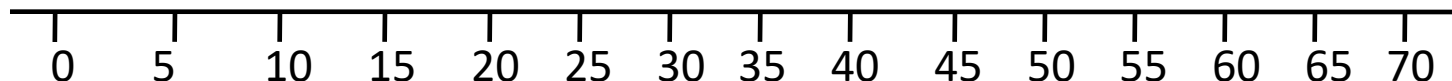
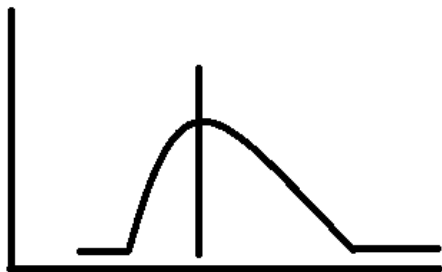
1. 1st Quartile = 47

2. 2nd Quartile = 51.5

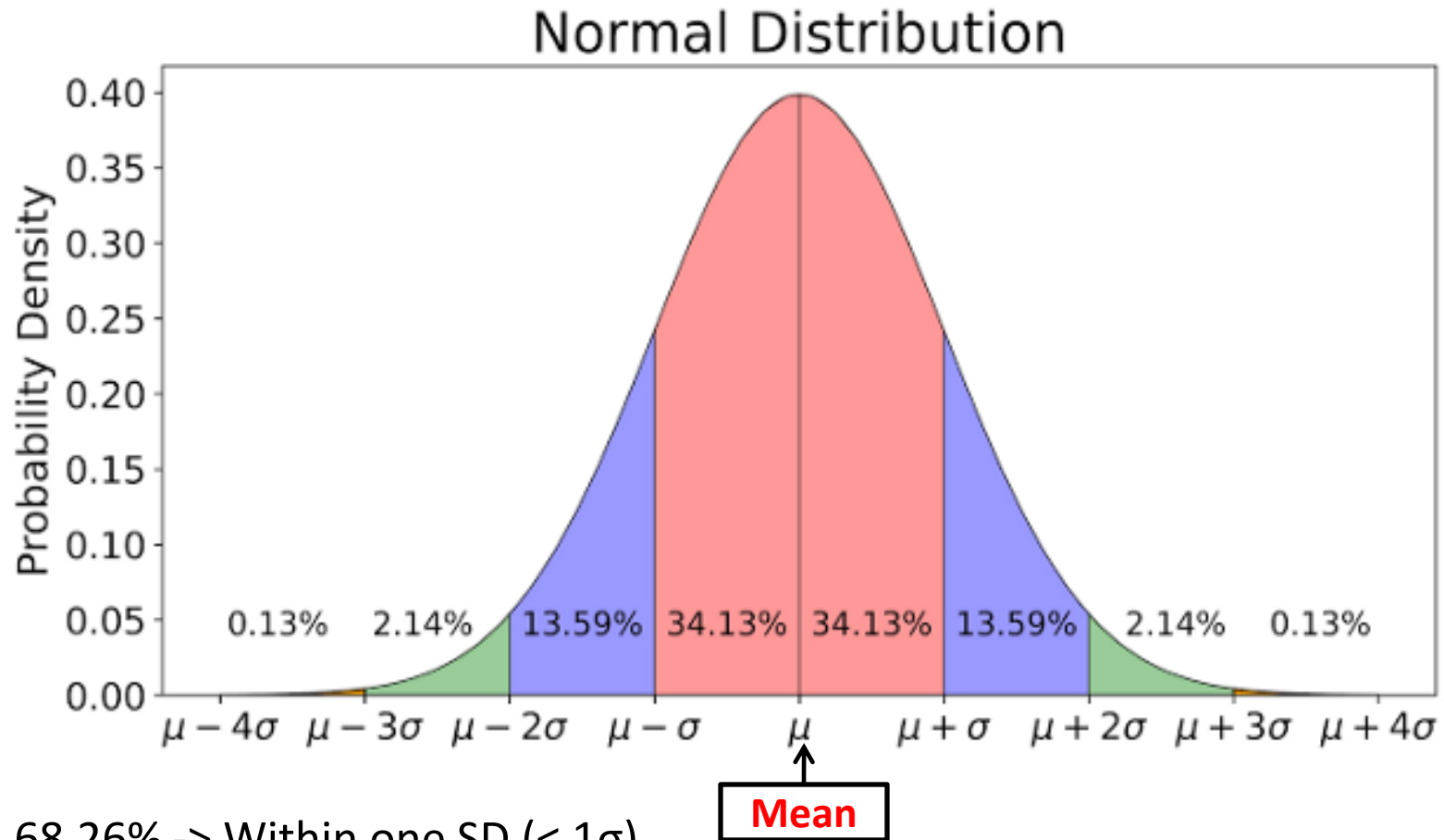
3. 3rd Quartile = 57

4. Smallest # = 2 43

5. Largest # = 63



Why **1.5** * IQR?



68.26% -> Within one SD ($< 1\sigma$)

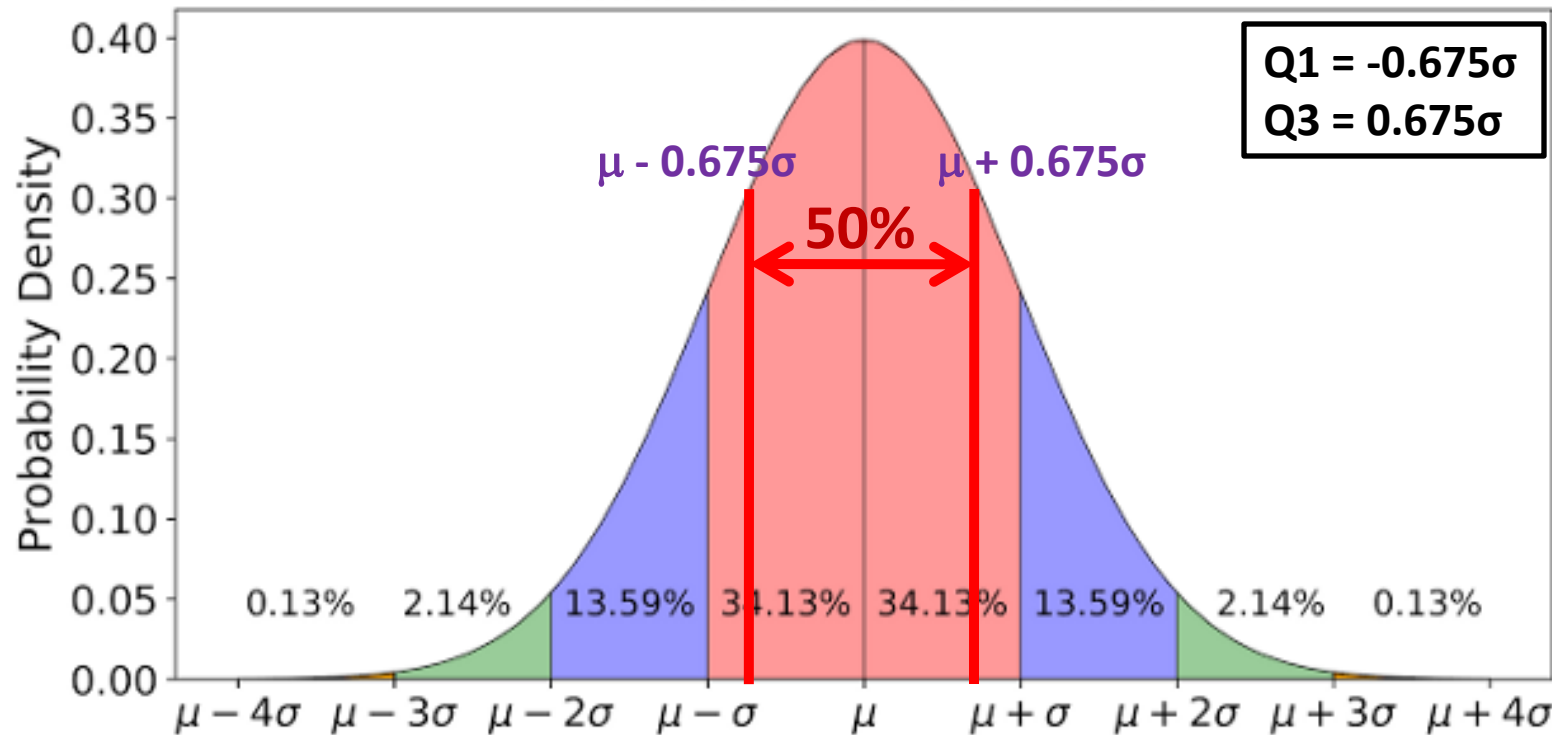
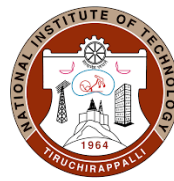
95.44% -> Within two SD ($< 2\sigma$)

99.72% -> Within three SD ($< 3\sigma$)

0.28% -> Outliers

Why **1.5** * IQR?

Normal Distribution



68.26% -> Within one SD ($< 1\sigma$)

95.44% -> Within two SD ($< 2\sigma$)

99.72% -> Within three SD ($< 3\sigma$)

0.28% -> Outliers

50% -> Within 0.675 SD ($< 0.675\sigma$)

- <https://towardsdatascience.com/why-1-5-in-iqr-method-of-outlier-detection-5d07fdc82097>
- <http://www.cs.uni.edu/~campbell/stat/normfact.html>
- <https://mathbitsnotebook.com/Algebra2/Statistics/STstandardNormalDistribution.html>

Why **1.5** * IQR?

Lower Whisker = $Q1 - ? * IQR$

Upper Whisker = $Q3 + ? * IQR$

Scale = 1

Lower Bound:

$$\begin{aligned} &= Q1 - \mathbf{1} * IQR \\ &= Q1 - 1 * (Q3 - Q1) \\ &= -0.675\sigma - 1 * (0.675 - [-0.675])\sigma \\ &= -0.675\sigma - 1 * 1.35\sigma \\ &= \mathbf{-2.025\sigma} \end{aligned}$$

Upper Bound:

$$\begin{aligned} &= Q3 + \mathbf{1} * IQR \\ &= Q3 + 1 * (Q3 - Q1) \\ &= 0.675\sigma + 1 * (0.675 - [-0.675])\sigma \\ &= 0.675\sigma + 1 * 1.35\sigma \\ &= \mathbf{2.025\sigma} \end{aligned}$$

Scale = 2

Lower Bound:

$$\begin{aligned} &= Q1 - \mathbf{2} * IQR \\ &= Q1 - 2 * (Q3 - Q1) \\ &= -0.675\sigma - 2 * (0.675 - [-0.675])\sigma \\ &= -0.675\sigma - 2 * 1.35\sigma \\ &= \mathbf{-3.375\sigma} \end{aligned}$$

Upper Bound:

$$\begin{aligned} &= Q3 + \mathbf{2} * IQR \\ &= Q3 + 2 * (Q3 - Q1) \\ &= 0.675\sigma + 2 * (0.675 - [-0.675])\sigma \\ &= 0.675\sigma + 2 * 1.35\sigma \\ &= \mathbf{3.375\sigma} \end{aligned}$$

Scale = 3

Lower Bound:

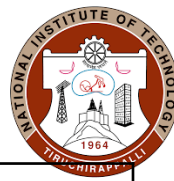
$$\begin{aligned} &= Q1 - \mathbf{1.5} * IQR \\ &= Q1 - 1.5 * (Q3 - Q1) \\ &= -0.675\sigma - 1.5 * (0.675 - [-0.675])\sigma \\ &= -0.675\sigma - 1.5 * 1.35\sigma \\ &= \mathbf{-2.7\sigma} \end{aligned}$$

Upper Bound:

$$\begin{aligned} &= Q3 + \mathbf{1.5} * IQR \\ &= Q3 + 1.5 * (Q3 - Q1) \\ &= 0.675\sigma + 1.5 * (0.675 - [-0.675])\sigma \\ &= 0.675\sigma + 1.5 * 1.35\sigma \\ &= \mathbf{2.7\sigma} \end{aligned}$$

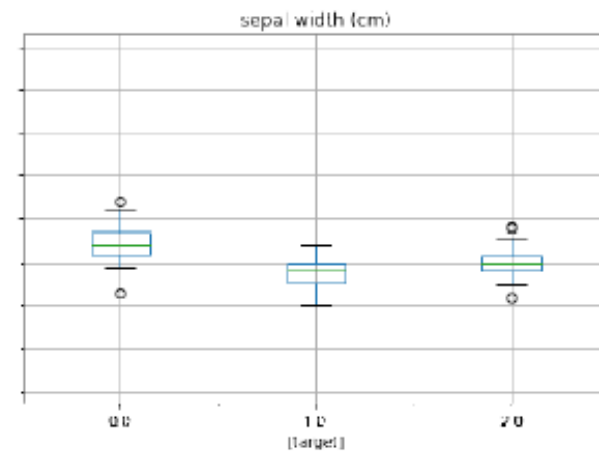
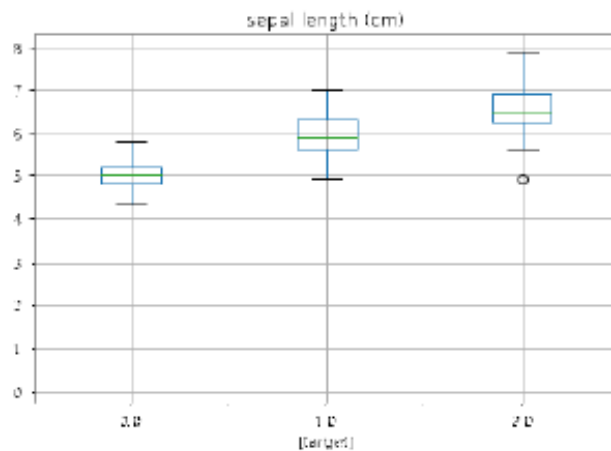
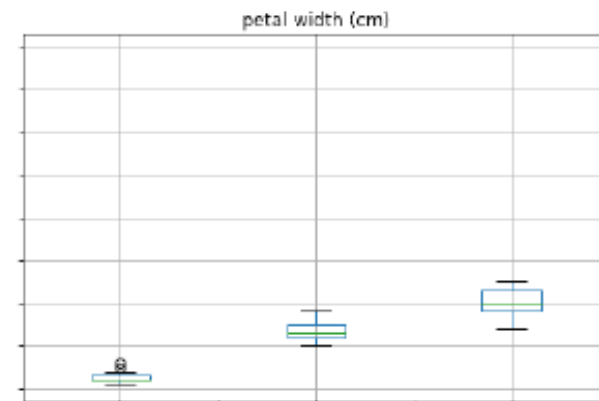
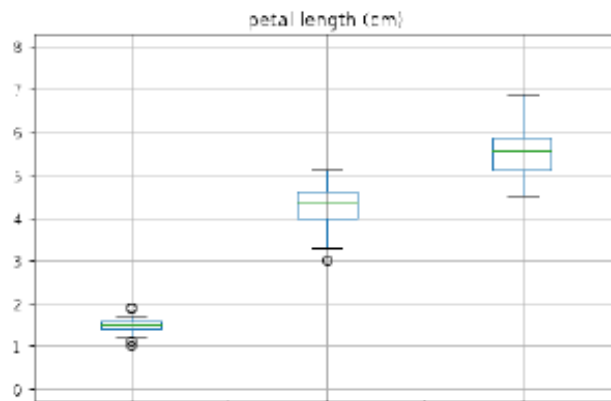
We know that, in Normal Distribution, 99.72% data lies within three SD ($< 3\sigma$)

Box-and-Whisker Plot



Sepal Length (cm)	Sepal Width (cm)	Petal Length (cm)	Petal Width (cm)	Target
5.5	6.5	3.5	2.5	Setosa
1.2	4.2	3.5	2.8	Versicolor
3.8	1.5	2.8	3	Virginica

Boxplot grouped by target

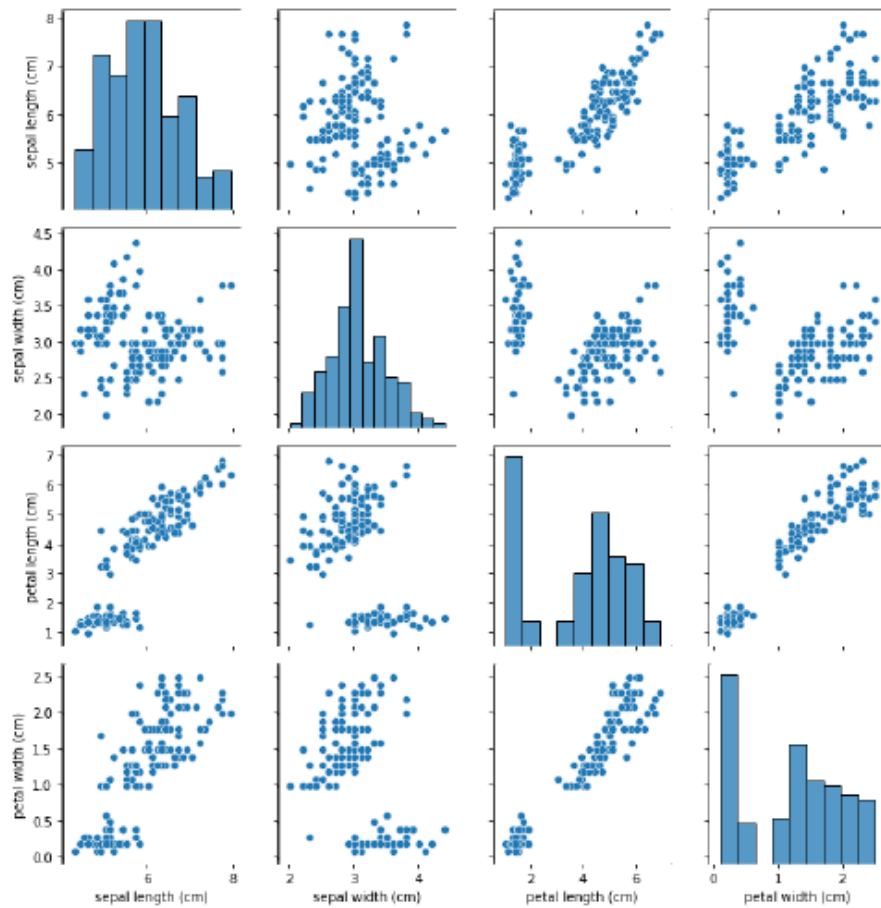


Pair Plot

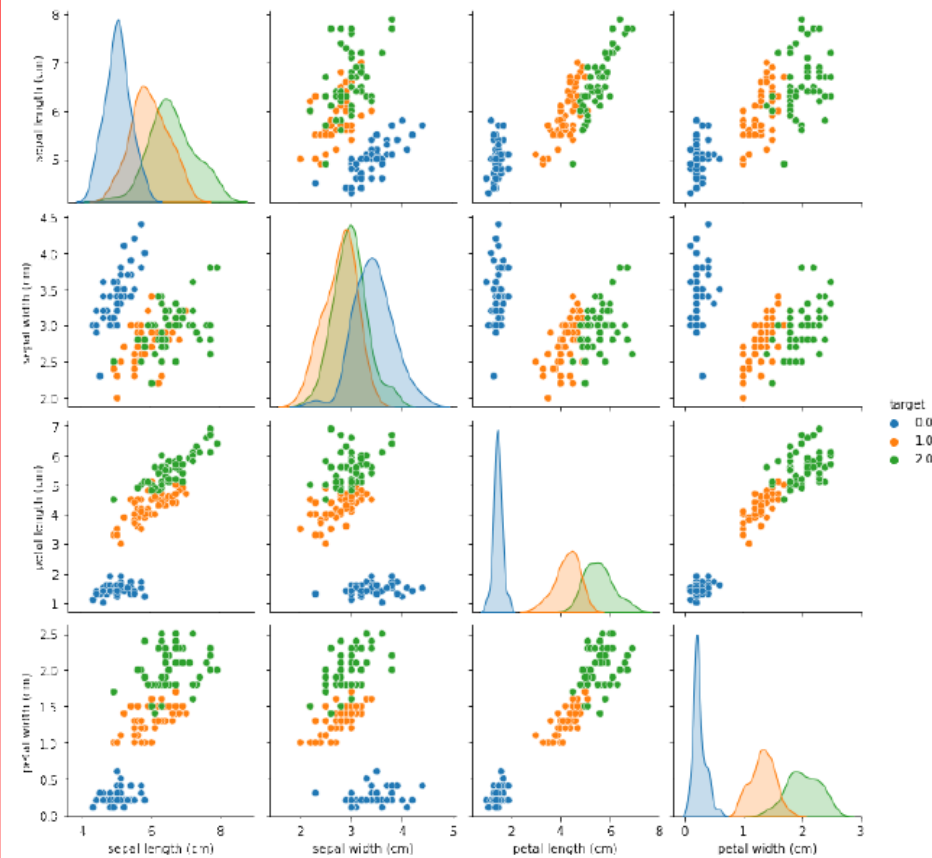


Sepal Length (cm)	Sepal Width (cm)	Petal Length (cm)	Petal Width (cm)	Target
5.5	6.5	3.5	2.5	Setosa
1.2	4.2	3.5	2.8	Versicolor
3.8	1.5	2.8	3	Virginica

```
plt.figure(figsize=(24,15))
sns.pairplot(df)
```



```
sns.pairplot(data, hue="target", palette="tab10")
```



Miscellaneous



- **Feature Binning:** Conversion of a continuous variable to categorical

Age	Bin Range	Player Type
35	[35 – 40)	Senior
38		Senior
40	[40 – 45)	Senior-most
43		Senior-most

Destructive Process

- **Feature Encoding:** Conversion of a categorical variable to numerical features

Player Type	Integer
Junior	0
Junior-most	1
Senior	2
Senior-most	3

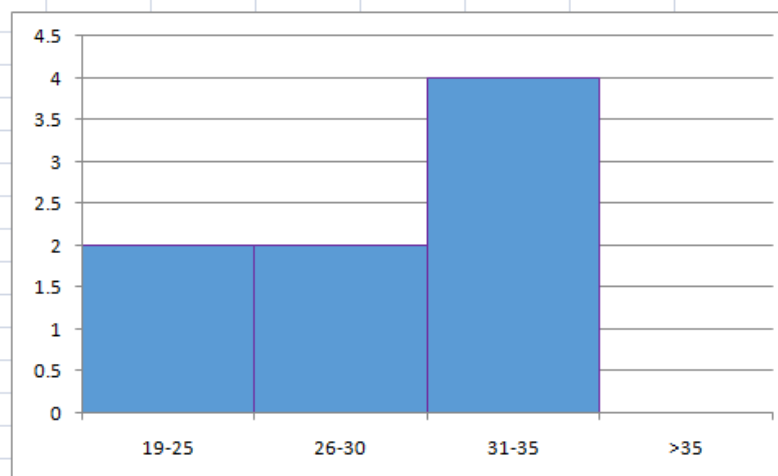
Feature Binning

Feature Binning

	Name	Age	Grade	Role	Rating	Country
0	Virat	31	Best	Batsman	871	IND
1	Rohit	33	Better	Batsman	855	IND
2	Babar	25	Good	Batsman	829	PAK
3	Boult	31	Best	Bowler	722	NZ
4	Bumrah	26	Better	Bowler	719	IND
5	Mujeeb	19	Good	Bowler	701	AFG
6	Nabi	35	Best	All-Rounder	301	AFG
7	Stokes	29	Better	All-Rounder	285	ENG

Bin 0 -> 19 – 25
 Bin 1 -> 26 – 30
 Bin 2 -> 31 – 35

Bins	Frequency
19-25	2
26-30	2
31-35	4
>35	0



Feature Binning

- Numeric feature in your dataset with a range of continuous numbers, with too many values to model
- In visualizing the dataset, the first indication of this is the feature has a large number of unique values
- Relationship between such a numeric feature and the class label is not linear
 - Feature value does not increase or decrease monotonically with the class label
- Consider binning the continuous values into groups using a technique known as quantization where each bin represents a different range of numeric feature
- Each categorical feature (bin) can then be modeled as having its own linear relationship with the class label

Feature Binning

- **Eg:** Say you know that the continuous numeric feature called mobile minutes used is not linearly correlated with the likelihood for a customer to churn
- You can bin the numeric feature mobile minutes used into a categorical feature that might be able to capture the relationship with the class label (churn) more accurately
- Optimum number of bins is dependent on characteristics of the variable and its relationship to the target and this is best determined through experimentation

Feature Binning

- During binning (or quantization), each value in a column is mapped to a bin by comparing its value against the values of bin edges
- **Eg:** If the value is 1.5 and the bin edges are **1**, 2, and **3**, the element would be mapped to bin number 2
- Value 0.5 would be mapped to bin number 1 (the underflow bin)
- Value 3.5 would be mapped to bin number 4 (the overflow bin)

Feature Binning

- Smoothing by Equal Frequency Bins
- Smoothing by Bin Means
- Smoothing by Bin Median
- Smoothing by Bin Boundaries

8, 16, 9, 15, 21, 21, 24, 30, 26, 27, 30, 30, 34

Step 1: Sort => 8, 9, 15, 16, 21, 21, 24, 26, 27, 30, 30, 34

Smoothing by Equal-Frequency Bins

8, 9, 15, 16, 21, 21, 24, 26, 27, 30, 30, 34

Bin 1: 8, 9, 15, 16

Bin 2: 21, 21, 24, 26

Bin 3: 27, 30, 30, 34

Bin Size = 4 (*No. of elements in each bin*)

Smoothing by Bin Mean

8, 9, 15, 16, 21, 21, 24, 26, 27, 30, 30, 34

Bin 1: 8, 9, 15, 16 => Mean = 12

Bin 2: 21, 21, 24, 26 => Mean = 23

Bin 3: 27, 30, 30, 34 => Mean = 30

Bin 1: 12, 12, 12, 12

Bin 2: 23, 23, 23, 23

Bin 3: 30, 30, 30, 30

Smoothing by Bin Boundaries

8, 9, 15, 16, 21, 21, 24, 26, 27, 30, 30, 34

Bin 1: 8, 9, 15, 16

Bin 2: 21, 21, 24, 26

Bin 3: 27, 30, 30, 34

Bin 1: 8, 8, 16, 16

Bin 2: 21, 21, 26, 26

Bin 3: 27, 27, 27, 34

Feature Binning

	Name	Age	Grade	Role	Rating	Country
0	Virat	31	Best	Batsman	871	IND
1	Rohit	33	Better	Batsman	855	IND
2	Babar	25	Good	Batsman	829	PAK
3	Boult	31	Best	Bowler	722	NZ
4	Bumrah	26	Better	Bowler	719	IND
5	Mujeeb	19	Good	Bowler	701	AFG
6	Nabi	35	Best	All-Rounder	301	AFG
7	Stokes	29	Better	All-Rounder	285	ENG

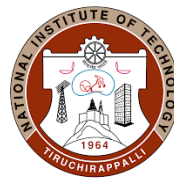
Bin 0 -> 19 – 25
 Bin 1 -> 26 – 30
 Bin 2 -> 31 – 35

```
bins = [19, 25, 30, 35]    =>  (18 to 25], (25 to 30], (30 to 35]
group_names = ['Young Player', 'Senior Player', 'Senior-most Player']
df['Player_Category'] = pd.cut(df['Age'], bins, labels=group_names)
```

- <https://towardsdatascience.com/feature-engineering-deep-dive-into-encoding-and-binning-techniques-5618d55a6b38>
- <https://pbpython.com/pandas-qcut-cut.html>

Feature Encoding

Categorical/Nominal vs Ordinal Variables



- Categorical or Nominal Variables
 - Has two or more categories, but there is no intrinsic ordering to the categories
 - **Eg:** A binary variable (such as yes/no question) is a categorical variable having two categories (yes or no) and there is no intrinsic ordering to the categories
- Ordinal Variables
 - Similar to a categorical variable
 - Difference between the two is that there is a clear ordering of the categories
 - **Eg:** A grade variable is an ordinal variable having 7 categories (F, E, D, C, B, A, S) and there is an intrinsic ordering to the categories

Label Encoding

- Transforms categorical variables into numerical variables by assigning a numerical value to each of the categories
- Label encoding can be used for Ordinal variables

[male, female]	[0, 1]
[blue, green, red, black]	[0, 1, 2, 3]
[10, 21], [22, 33], [34, 45], [46, 55]	[0, 1, 2, 3]

- <https://towardsdatascience.com/feature-engineering-deep-dive-into-encoding-and-binning-techniques-5618d55a6b38>
- <https://www.analyticsvidhya.com/blog/2020/08/types-of-categorical-data-encoding/#:~:text=Dummy%20Encoding,-Dummy%20coding%20scheme&text=This%20categorical%20data%20encoding%20method,over%20one%2Dhot%2Dencoding.>

Ordinal Encoding

- Ordinal encoding is an encoding technique to transform an original categorical variable to a numerical variable
- Ensures the ordinal nature of the variables is sustained

[male, female]	[0, 1]
[10, 21], [22, 33], [34, 45], [46, 55]	[0, 1, 2, 3]
[cold, warm, hot]	[0, 1, 2]
[poor, fair, good, very good, excellent]	[0, 1, 2, 3, 4]
[F, E, D, C, B, A, S]	[0, 1, 2, 3, 4, 5, 6]

Frequency Encoding

- Transforms an original categorical variable to a numerical variable
- Considers the frequency distribution of the data
- Can be useful for nominal features

	Column	Freq_Encoding
→	red	5
	green	3
→	red	5
	green	3
	blue	4
→	red	5
→	red	5
	blue	4
→	red	5
	blue	4
	blue	4
	green	3

Binary Encoding

- Transforms an original categorical variable to a numerical variable by encoding the categories as Integer and then converted into binary code
- Preferable for variables having a large number of categories
- For a 100 category variable, Label Encoding creating 100 labels each corresponding to a category, but instead binary encoding creating only 7 categories

Column	Label Enc	Binary enc1	Binary enc2	Binary enc3
red	1	0	0	1
green	2	0	1	0
red	1	0	0	1
green	2	0	1	0
blue	3	0	1	1
red	1	0	0	1
grey	4	1	0	0
blue	3	0	1	1
red	1	0	0	1
blue	3	0	1	1
blue	3	0	1	1
green	2	0	1	0
grey	4	1	0	0

Decimal	Binary	No. of Bits Required
0	--- 0	1
1	--- 1	1
2	-- 1 0	2
3	-- 1 1	2
4	- 1 0 0	3
5	- 1 0 1	3
6	- 1 1 0	3
7	- 1 1 1	3
8	1 0 0 0	4
9	1 0 0 1	4

One Hot Encoding

- One hot encoding technique splits the category each to a column
- Creates k different columns each for a category and replaces one column with 1 rest of the columns is 0

Column	red	green	blue
→ red	1	0	0
→ green	0	1	0
red	1	0	0
green	0	1	0
→ blue	0	0	1
red	1	0	0
red	1	0	0
blue	0	0	1
red	1	0	0
blue	0	0	1
blue	0	0	1
green	0	1	0

Dummy Encoding

- Similar to one-hot encoding
- In one-hot encoding, for N categories in a variable, it uses N binary variables
- Dummy encoding is a small improvement over one-hot-encoding
 - Encoding uses only N-1 features to represent N labels/categories

Column	Code
A	100
B	010
C	001

One- Hot Coding

Column	Code
A	10
B	01
C	00

Dummy Code

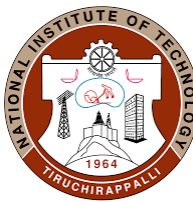
Drawbacks of One Hot and Dummy Encodings



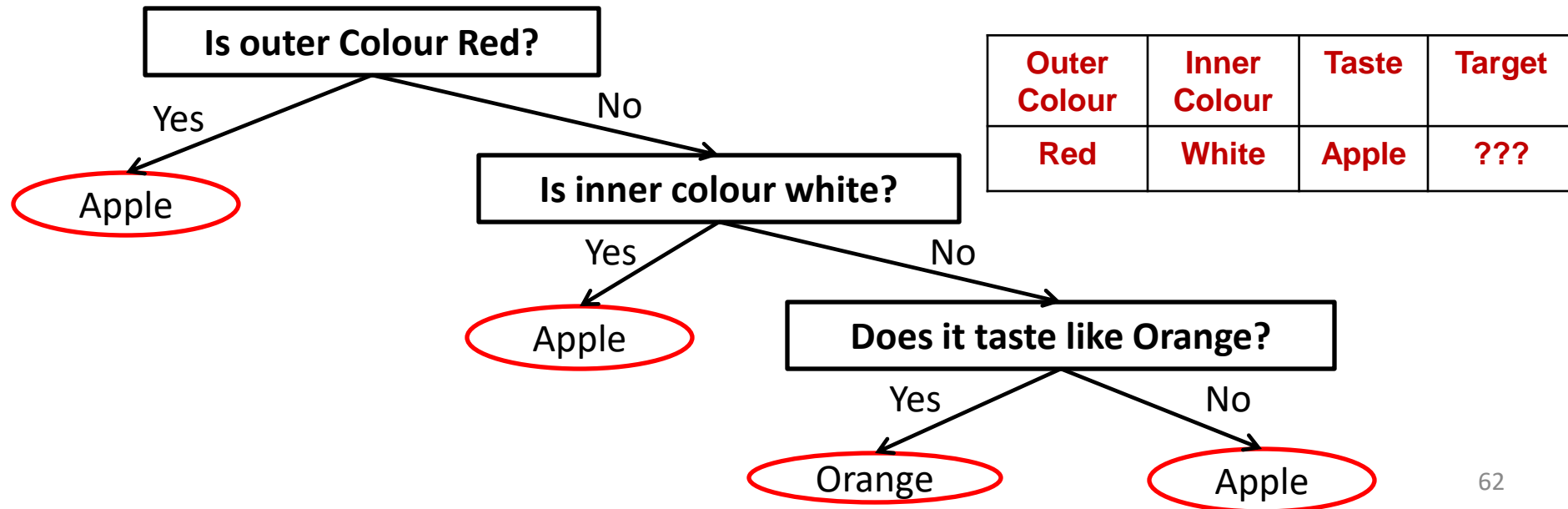
- These two encoding schemes introduce sparsity in the dataset
 - Several columns having 0s and a few of them having 1s
 - Creates multiple dummy features in the dataset without adding much information
- Might lead to a Dummy variable trap
 - Phenomenon where features are highly correlated -> Means using the other variables, we can easily predict the value of a variable
 - Due to the massive increase in the dataset, coding slows down the learning of the model along with deteriorating the overall performance that ultimately makes the model computationally expensive
- While using tree-based models these encodings are not an optimum choice

Classification – Decision Tree Algorithm

(Supervised Learning)



ID	Outer Colour	Inner Colour	Taste	Target
1	Red	White	Apple	Apple
2	Orange	Orange	Orange	Orange
3	Red	Orange	Orange	Orange
4	Red	Orange	Apple	Apple
5	Orange	White	Apple	Apple
6	Orange	White	Orange	Orange



Feature Engineering

Feature Engineering



- Process of transforming raw data into features that better represent the underlying problem to the machine learning algorithm

CITY 1 LAT.	CITY 1 LNG.	CITY 2 LAT.	CITY 2 LNG.	Driveable?	=>	DISTANCE (MI.)	Driveable?
123.24	46.71	121.33	47.34	Yes		14	Yes
123.24	56.91	121.33	55.23	Yes		28	Yes
123.24	46.71	121.33	55.34	No		705	No
123.24	46.71	130.99	47.34	No		2432	No

- Exceptionally difficult for a machine learning algorithm to learn the relationships between these four attributes and the class label
- Compute the distance between the source and destination and use it as a feature
- Results in improved model accuracy on unseen data

Feature Engineering



- Feature engineering is when you use your knowledge about the data to create fields that make a machine learning algorithm work better
- Engineered features that enhance the training provide information that better differentiate the patterns in the data
- Should strive to add a feature that provides additional information that is not clearly captured or easily apparent in the original or existing feature set

Feature Engineering

- Decompose Categorical Attributes
 - Item_Color -> Red, Blue, **Unknown**
 - Binary Feature -> Has_Color
 - Binary Features -> Is_red, Is_Blue, Is_Unknown
- Decompose Date and Time
 - 2014-09-20T20:45:40Z
 - Numerical Feature -> Hour_of_Day
- Reframe Numerical Quantities
 - Transform into a new unit or the decomposition into multiple components
 - Quantity like weight, distance, timing
 - A linear transform may be useful to regression and other scale dependent

Feature Selection

Feature Selection



Id	Cushion Color	Cushion Diameter (in cm)	Cushion Length (in cm)	Cushion Area	Cap Color	Cap diameter (in cm)	Cap Length (in cm)	Tail Color	Tail Length (in cm)	Target
1	Black	0.8	3	2.4	Black	1	4	Black	3	Black
2	Black	0.8	3	2.4	Black	1	4	Black	3	Black
3	Blue	0.8	3	2.4	Blue		4	Blue	3	Blue
4	Blue	0.8%	3	2.4	Blue	1	4	Blue	3	Blue
5	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue
6	Black	0.8	3	2.4	Black	1	4.	Black	3	Black
7	Black	0.8	3	2.4	Black	1	4	Black	3	Black
8	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue
9	Blue	0.8	3	2.4	Blue	1	4	Blue	3	Blue

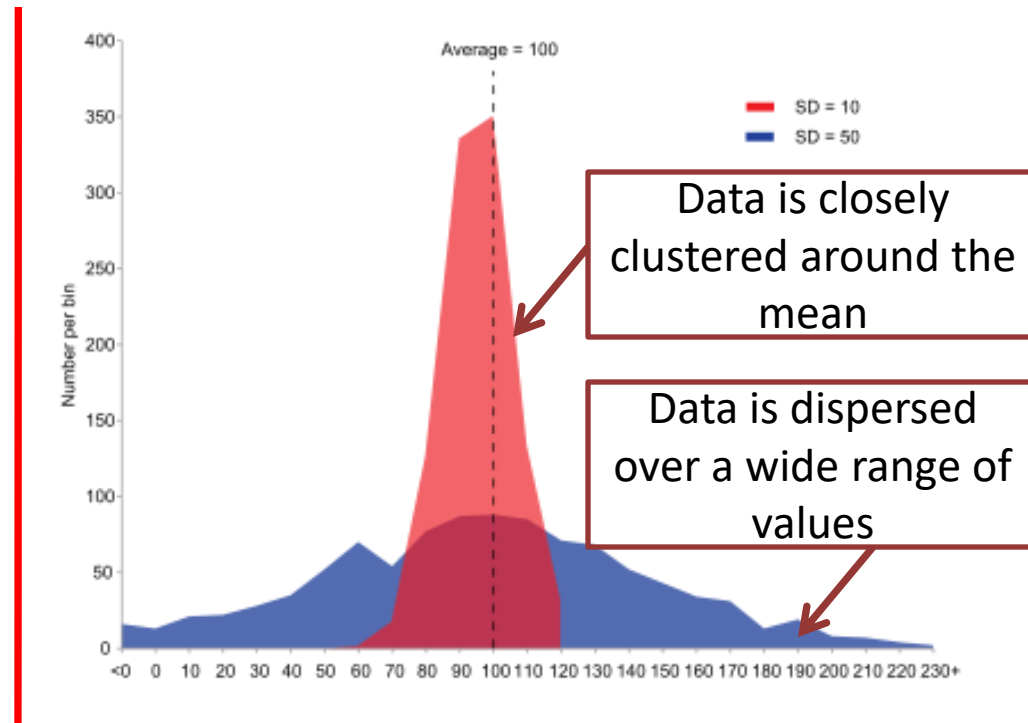
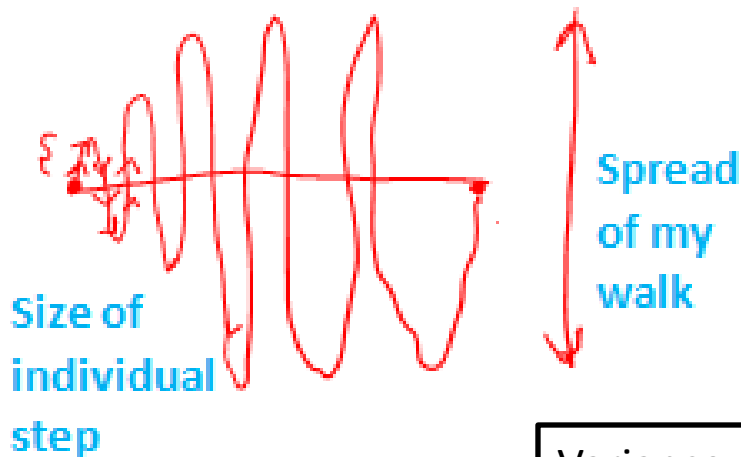
Variance & Standard Deviation



- Variance is the expectation of the squared deviation of a random variable from its mean
- Informally, it measures how far a set of numbers is spread out from their average value

$$\text{Var}(X) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

$$\text{SD } (\sigma) = \text{Sqrt}(\text{Var})$$



Variance -> How far a set of nos. is spread out from their avg
SD -> On an avg, how much a data differs from the mean

Feature Selection

- Adding irrelevant or distracting attributes to a dataset often confuses machine learning systems
- Feature selection is the process of determining the features with the highest information value to the model
- Extract a subset of original features in the data without changing them
- Increases model accuracy by eliminating irrelevant features -> Makes model training more efficient
- Two main approaches
 - ✓ Filtering Method
 - ✓ Wrapper Method

Feature Selection

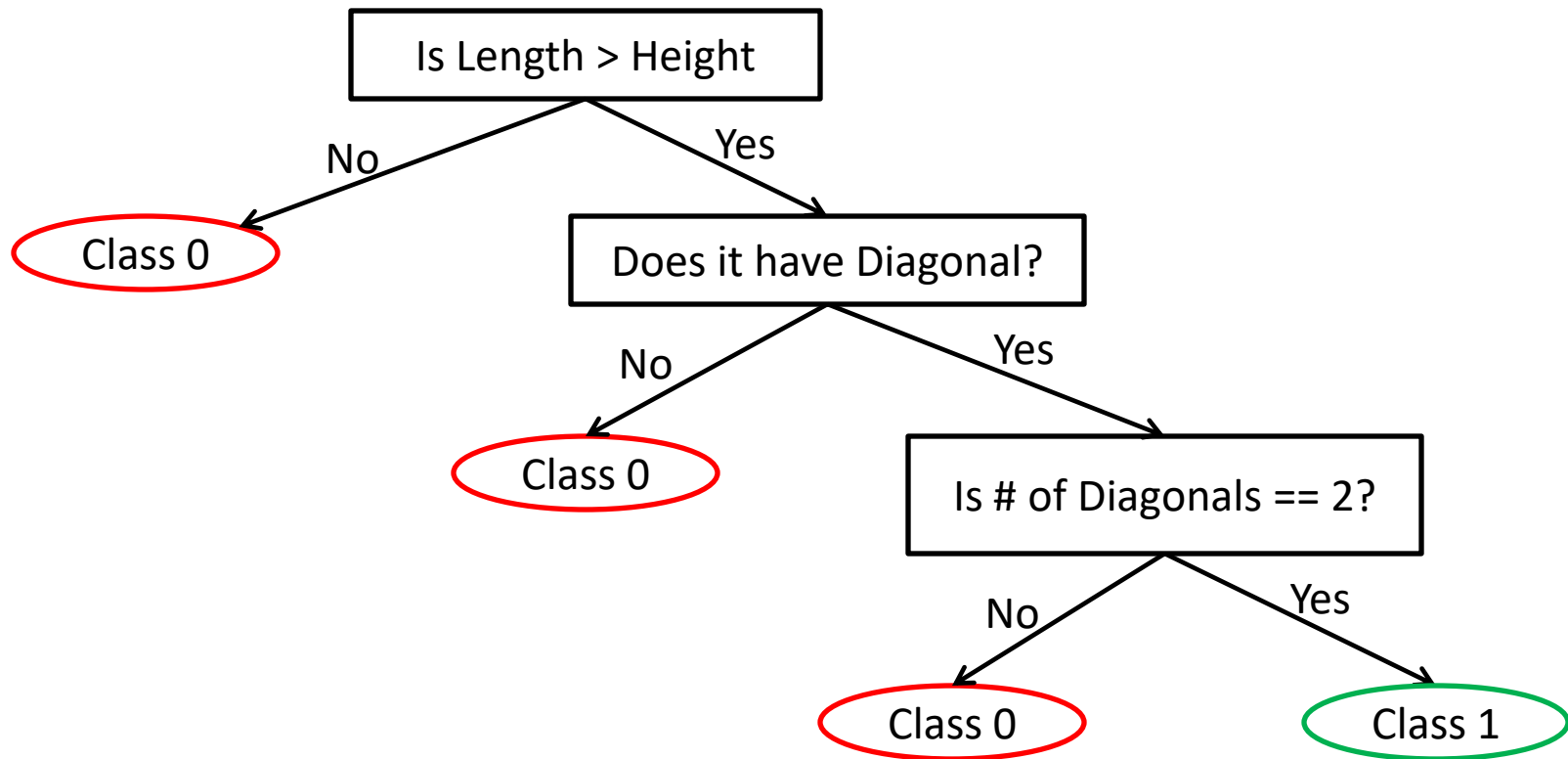
Sl. No.	Filtering Method	Wrapper Method
1.	Analyzes features using a test statistic and eliminate redundant or non-informative features Eg: Eliminate features that have little correlation to the class labels	Utilizes a classification model as part of feature selection Eg: Decision Tree -> Selects the most promising attribute to split on at each point and should never select irrelevant or unhelpful attributes
2.	Faster to compute since each feature only needs to be compared against its class label	Evaluate feature sets by constructing models and measuring it's performance
3.	Disadv: A feature with weak correlation to its class labels is eliminated. Some of these eliminated features, however, may have performed well when combined with other features	Disadv: Requires a large number of models to be trained and evaluated (a quantity that grows exponentially in the number of features)

Find an approach that works best, incorporate it into your data science workflow and run experiments to determine its effectiveness

Binary Classification – Decision Tree



Shape	Diag	# of Diag	Length (in cm)	Height (in cm)	Class
Rectangle	Yes	2	100	10	1
Rectangle	Yes	2	1000	100	1
Square	Yes	2	100	100	0
Triangle	No	0	100	1000	0
Circle	No	0	100	100	0
Diamond	Yes	2	100	100	0
Rectangle	Yes	2	100	5	1



Feature Selection vs Dimensionality Reduction

- While feature selection seeks to reduce the number of features in a dataset, it is not usually referred to by the term “dimensionality reduction”
- Dimensionality reduction methods create engineered features by transforming the original features

Filter-Based Feature Selection

- Provides multiple feature selection algorithms -> Type of predictive task and data types
- Evaluating the correlation between each feature and the target attribute, these methods apply a statistical measure to assign a score to each feature
- Features are then ranked by the score, which may be used to help set the threshold for keeping or eliminating a specific feature

Feature Selection Techniques

- Pearson Correlation
- Mutual Information
- Kendall Correlation
- Spearman Correlation
- Chi-Squared
- Fisher Score
- Count-Based

Covariance

- Indicates the direction of the linear relationship between variables

Sl. No.	Size (in sq. ft.)	Price (in Rs.)
1.	1200	20,000/-
2.	1800	30,000/-
3.	2400	40,000/-
4.	3000	50,000/-

$S \uparrow$ $P \uparrow$
 $S \uparrow$ $P \downarrow$
 $S \downarrow$ $P \uparrow$
 $S \downarrow$ $P \downarrow$

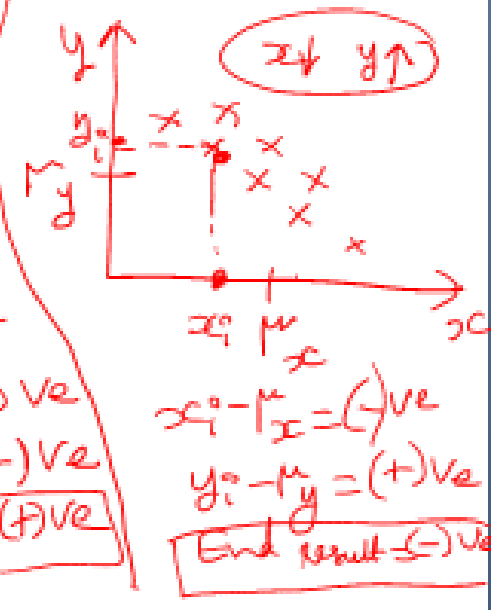
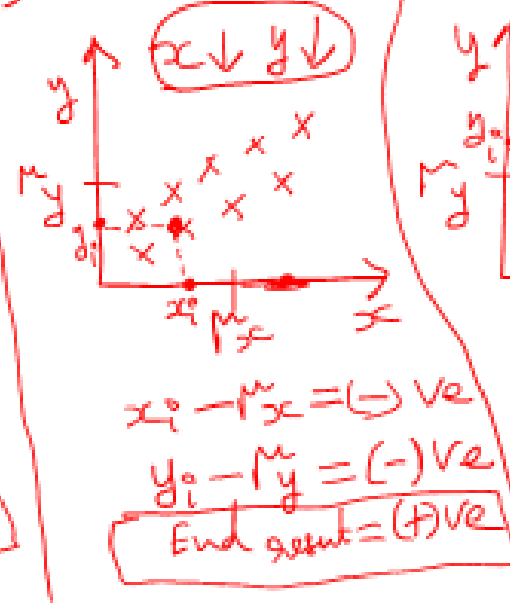
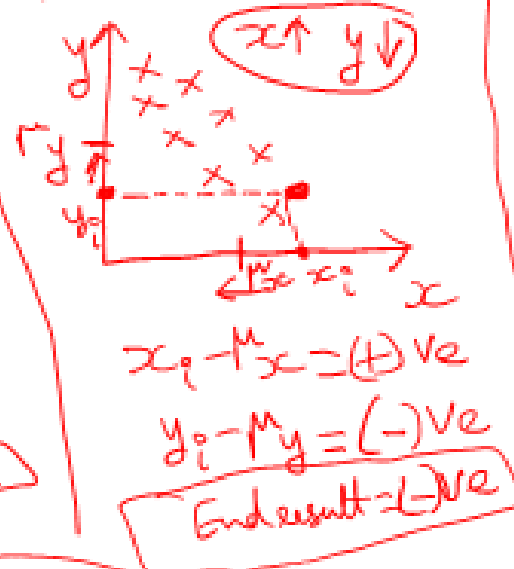
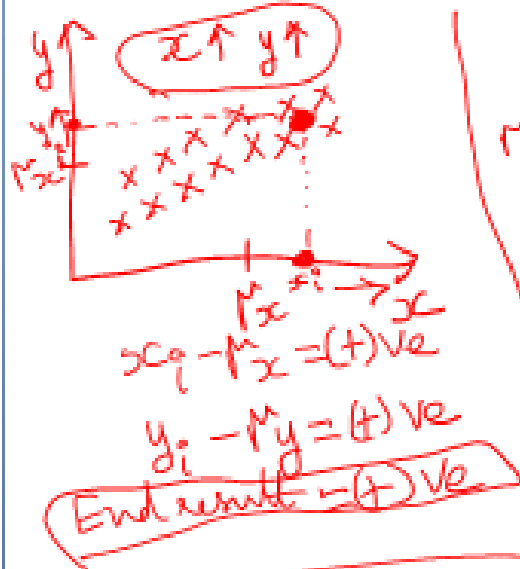
Quantify the relation between two variables with the help of covariance

$$\begin{aligned}
 \text{Cov}(\frac{\text{Size}}{x}, \frac{\text{Price}}{y}) &= \frac{1}{n} \sum_{i=1}^n \underbrace{(x_i - \mu_x)}_{\text{deviation of } x} * \underbrace{(y_i - \mu_y)}_{\text{deviation of } y} \\
 \text{Var}(x) &= \frac{1}{n} \sum_{i=1}^n (x_i - \mu_x)^2 = \frac{1}{n} \sum_{i=1}^n \underbrace{(x_i - \mu_x)}_{\text{deviation of } x} * \underbrace{(x_i - \mu_x)}_{\text{deviation of } x}
 \end{aligned}$$

$$\boxed{\text{Cov}(x, x) = \text{Var}(x)}$$

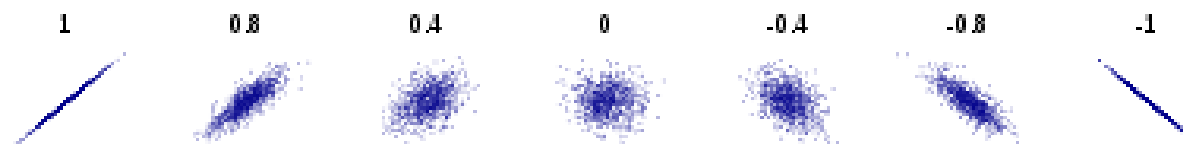
Covariance

$$\text{COV}(x, y) = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}) * (y_i - \bar{y})$$



Direction of Relationship

How much Positive?
How much Negative?



Pearson Correlation

- Pearson's correlation statistics, or Pearson's correlation coefficient or r value
- For any two variables, it returns a value that indicates the strength of the correlation

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

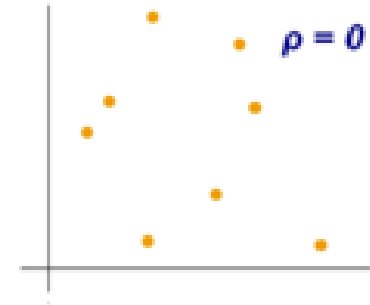
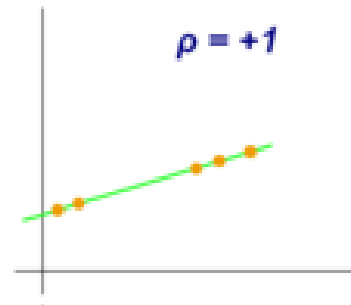
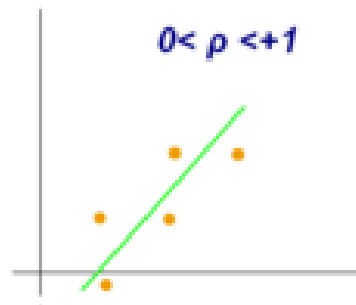
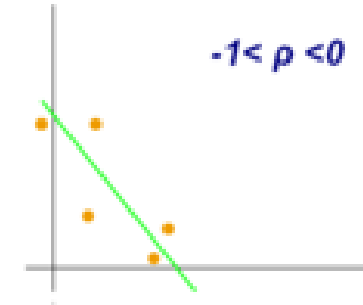
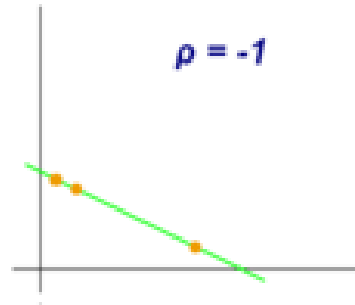
$$-1 \leq \rho \leq 1$$

- Computed by taking the covariance of two variables and dividing by the product of their standard deviations
- Coefficient is not affected by changes of scale in the two variables

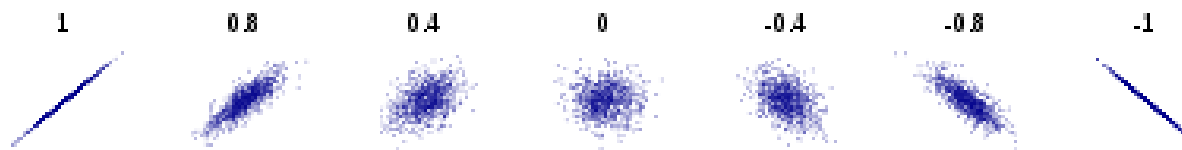
Transform X to $a + bX$ and transform Y to $c + dY$, where a , b , c , and d are constants with $b, d > 0$, without changing the correlation coefficient

Pearson Correlation

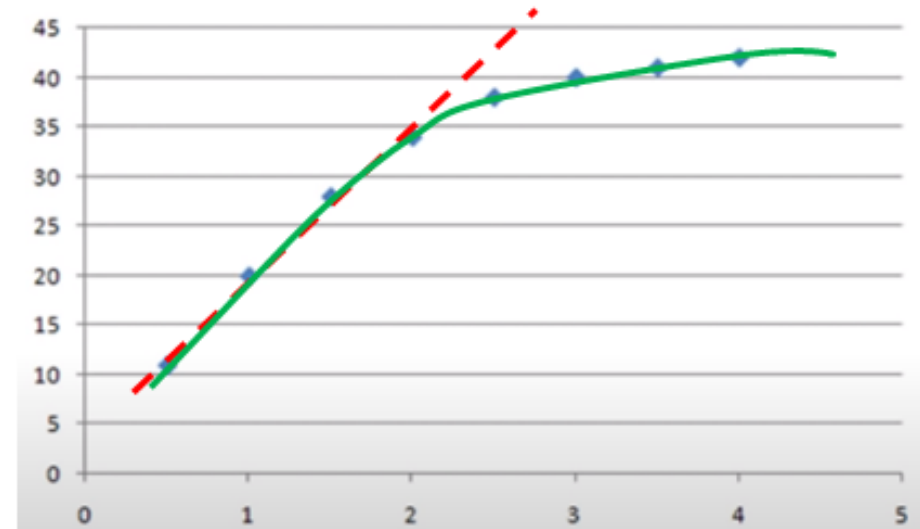
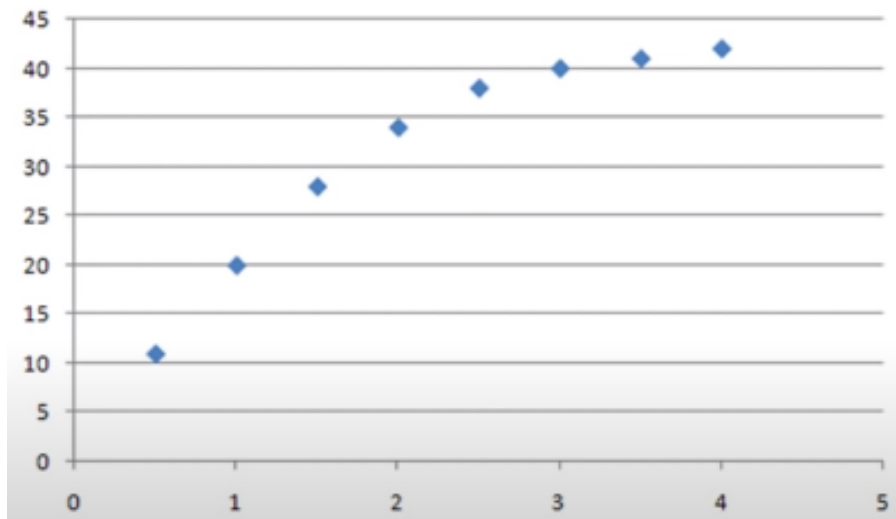
- Strength
- Direction of Relationship



x y
1 2 3



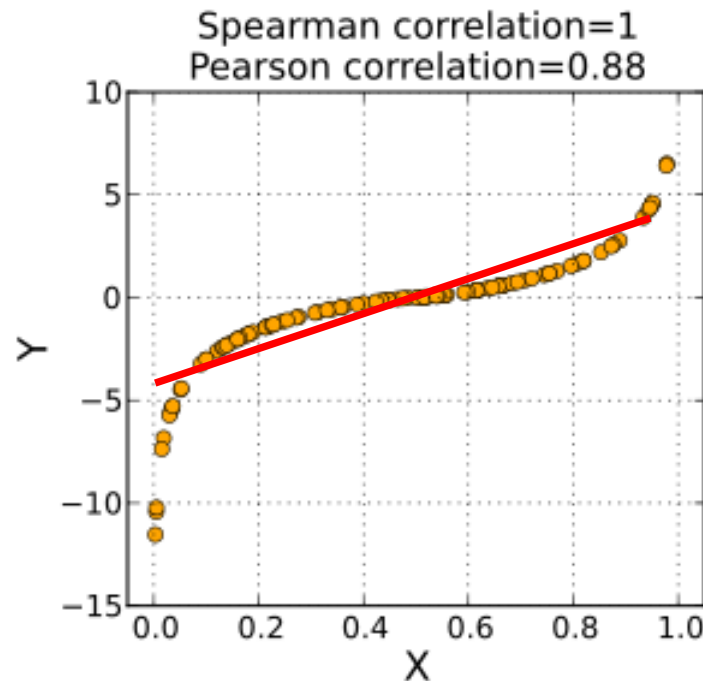
Spearman Rank Correlation Coefficient



$x \uparrow \quad y \uparrow$

Pearson Correlation Coefficient
value will be much closer to "1"

Spearman Rank Correlation Coefficient



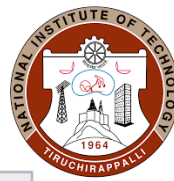
$x \uparrow \quad y \uparrow$

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

$$r_s = \rho_{rg_X, rg_Y} = \frac{\text{cov}(rg_X, rg_Y)}{\sigma_{rg_X} \sigma_{rg_Y}}$$

ρ denotes the usual [Pearson correlation coefficient](#), but applied to the rank variables
 $\text{cov}(rg_X, rg_Y)$ is the [covariance](#) of the rank variables,
 σ_{rg_X} and σ_{rg_Y} are the [standard deviations](#) of the rank variables.

Spearman Rank Correlation Coefficient



IQ, X_i	Hours of TV per week, Y_i
106	7
100	27
86	2
101	50
99	28
103	29
97	20
113	12
112	6
110	17

IQ, X_i	Hours of TV per week, Y_i	rank x_i	rank y_i	d_i	d_i^2
86	2	1	1	0	0
97	20	2	6	-4	16
99	28	3	8	-5	25
100	27	4	7	-3	9
101	50	5	10	-5	25
103	29	6	9	-3	9
106	7	7	3	4	16
110	17	8	5	3	9
112	6	9	2	7	49
113	12	10	4	6	36

When we should not use the second formula?

- If ties are present in the data set
- When ranks are normalized to [0, 1] ("relative ranks") – *[First formula is insensitive both to translation (+) and linear scaling (x)]*
- When data set is truncated

$$r_s = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \quad \rho = 1 - \frac{6 \times 194}{10(10^2 - 1)}$$

$$d_i = \text{rg}(X_i) - \text{rg}(Y_i) \quad \rho = -29/165 = -0.175757575$$

That the value is close to zero shows that the correlation between IQ and hours spent watching TV is very low, although the negative value suggests that the longer the time spent watching television the lower the IQ

Feature Scaling

Types of Feature Scaling

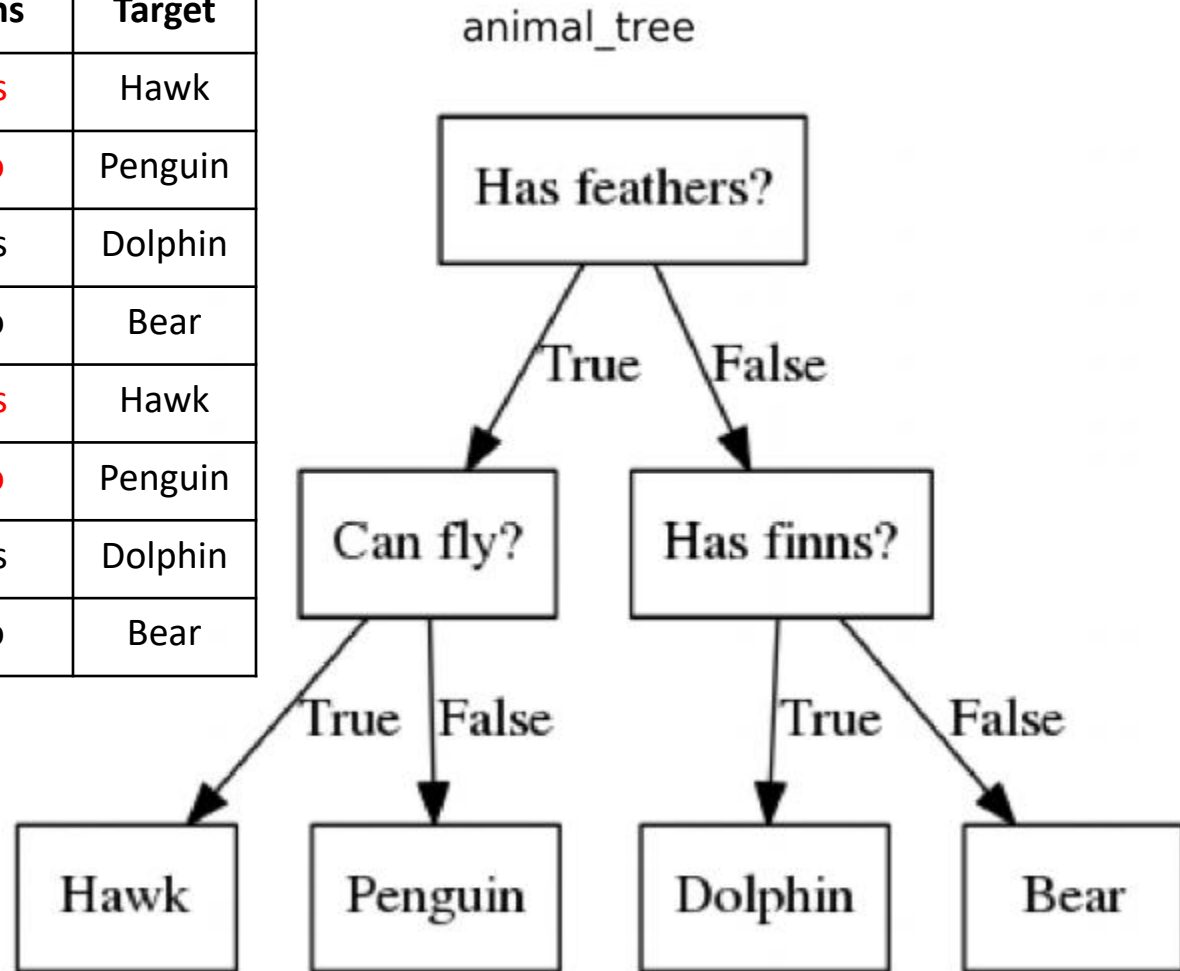
- Feature Standardization (Zscore)
- Feature Normalization

Feature Scaling

- Replaced missing values, removed outliers, pruned redundant rows
- Transform the values so that they are on a common scale, yet maintain their general distribution and ratios
- Why?
 - Different scales -> Millimeters, Centimeters, Meters
 - K-means -> Calculates distance between two points by the Euclidean distance
 - If one of the features has a broad range of values, the distance will be governed by this particular feature
 - Normalize the range of all features so that each feature contributes approximately proportionately to the final distance

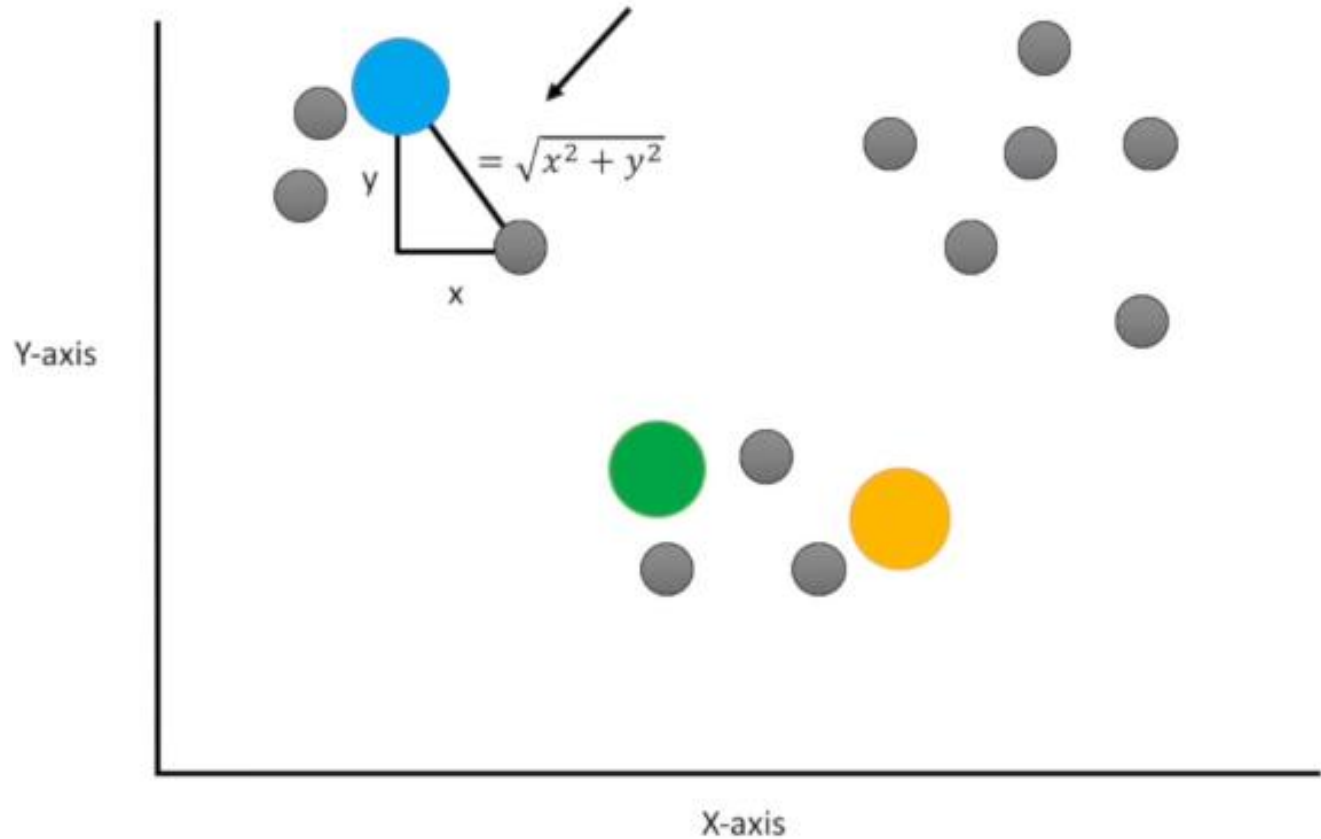
Decision Trees

Sl. No.	Feather	Fly	Finns	Target
1.	Yes	Yes	Yes	Hawk
2.	Yes	No	No	Penguin
3.	No	No	Yes	Dolphin
4.	No	Yes	No	Bear
5.	Yes	Yes	Yes	Hawk
6.	Yes	No	No	Penguin
7.	No	No	Yes	Dolphin
8.	No	Yes	No	Bear



K-Means Clustering

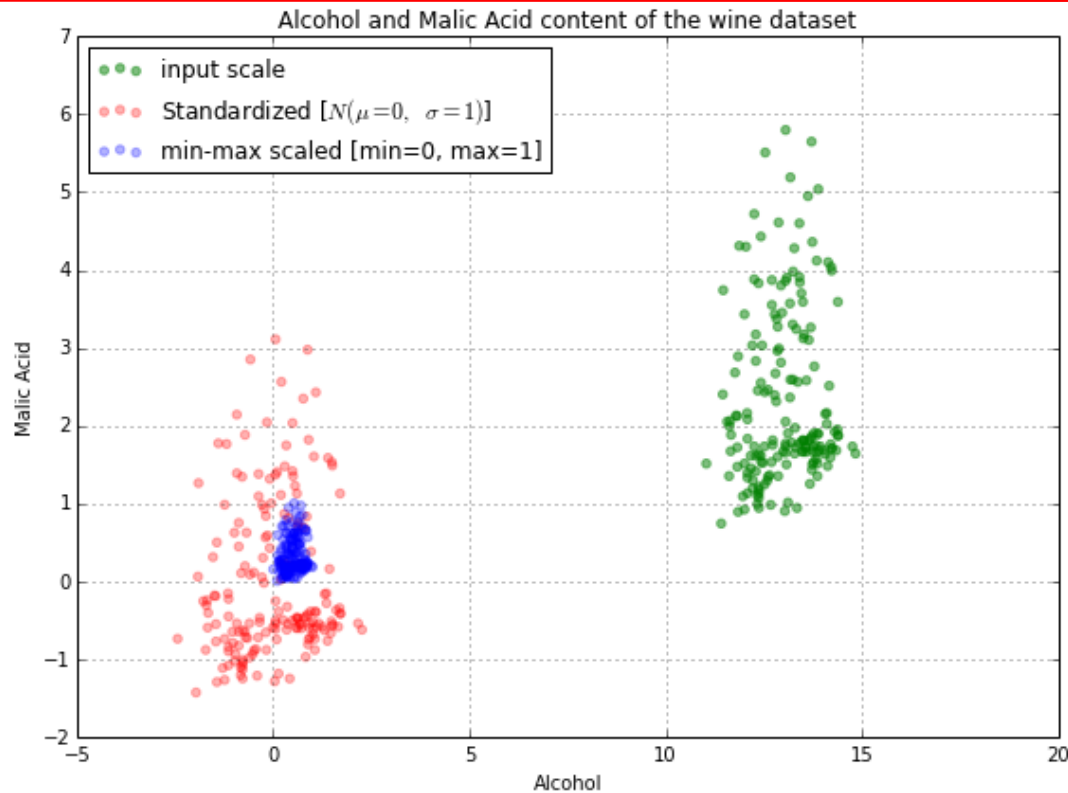
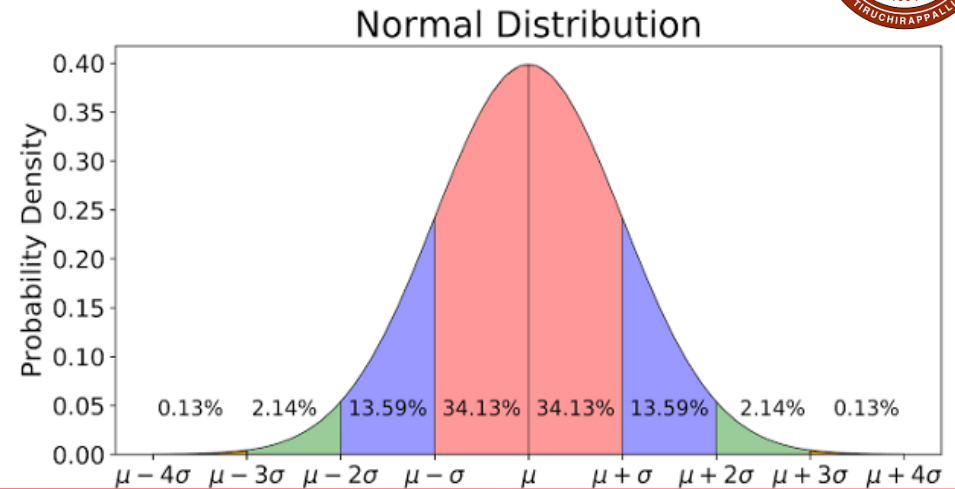
Rooms	Price
2	10,000
3	15,000
4	12,000
5	18,000



Why Feature Scaling?



68.26% -> Within one SD ($< 1\sigma$)
95.44% -> Within two SD ($< 2\sigma$)
99.72% -> Within three SD ($< 3\sigma$)
0.28% -> Outliers



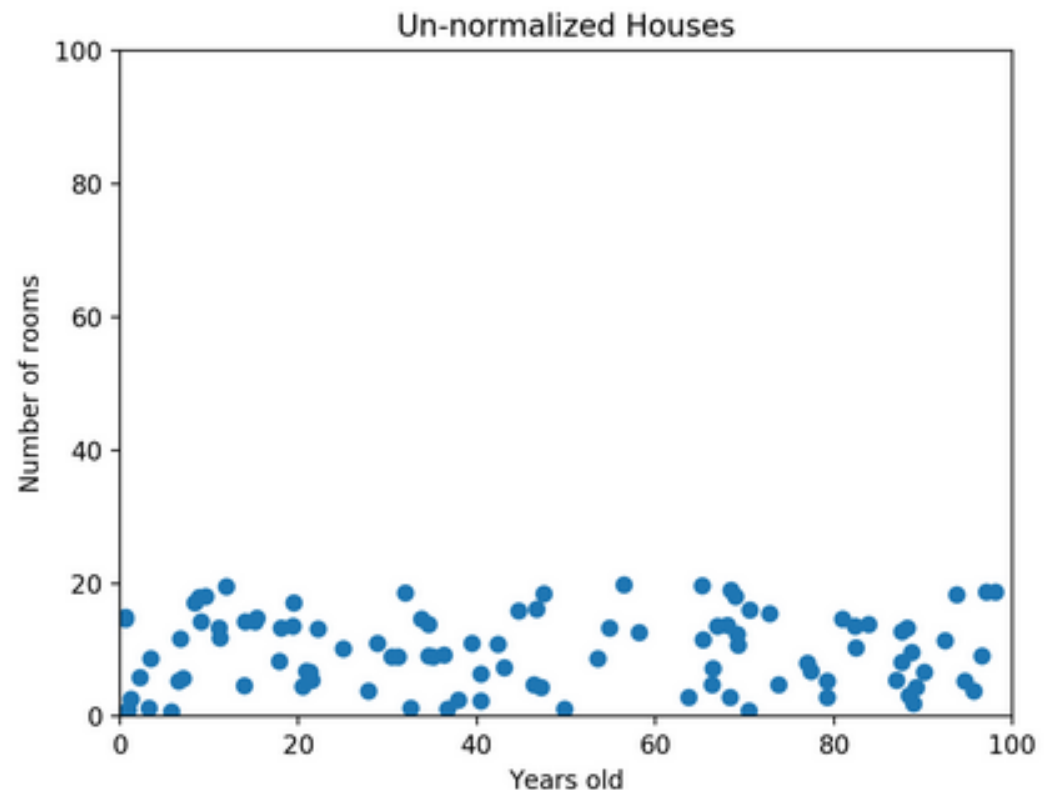
Feature Scaling Techniques

- **Zscore** -> Converts all values to a z-score
- **Min Max** -> Linearly rescales every feature to the **[0,1]** interval by shifting the values of each feature so that the minimal value is 0, and then dividing by the new maximal value (the difference between the original maximal and minimal values)
- **Logistic** -> Values are transformed using a log transform
- **Lognormal** -> Converts all values to a lognormal scale
- **Tanh Transformation** -> Converts all numeric features to values within a range of **0-1**, while preserving the overall distribution of values

Why Feature Scaling?

- Machine Learning algorithm should realize that there is a huge difference between a house with 2 rooms and a house with 20 rooms
- But right now, because two houses can be 100 years apart, the difference in the number of rooms contributes less to the overall difference
- **Make every data point have the same scale so each feature is equally important**

# of Rooms	Years
2	10
3	5
4	20

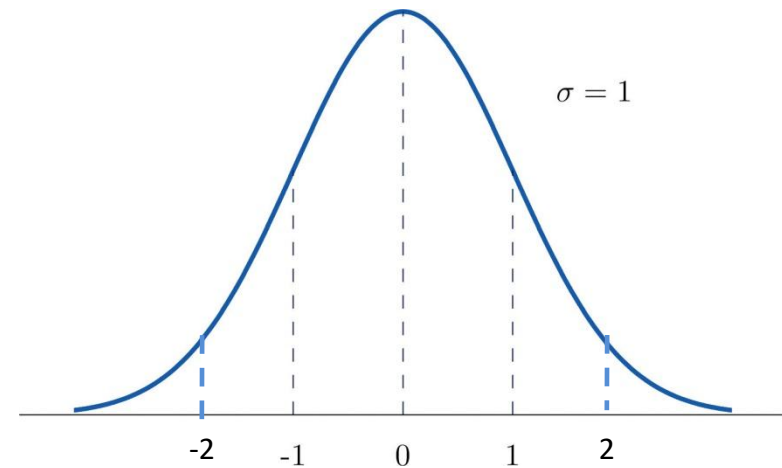
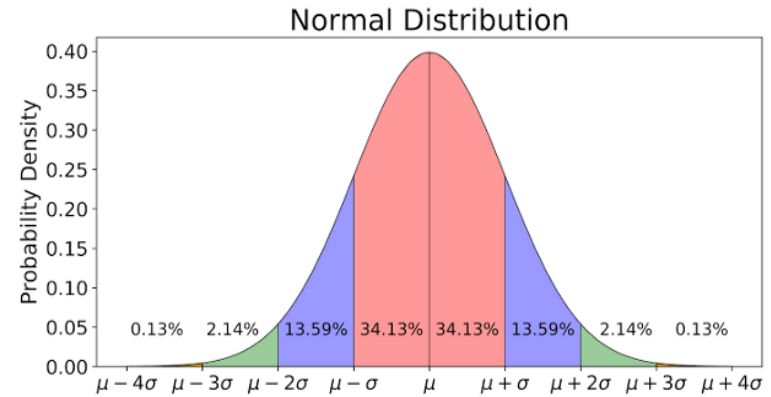


Zscore Standardization / Normalization

{1, 2, 3, 4, 5}

$$\mu = 3; \sigma = 1$$

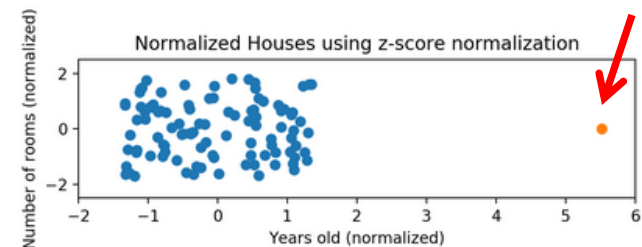
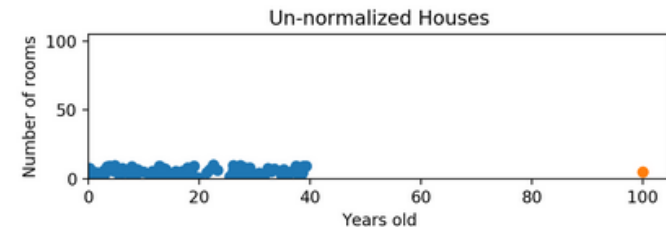
$$z = \frac{x - \mu}{\sigma}$$



Zscore Standardization / Normalization

- If a value is exactly equal to the mean, it will be normalized to 0
- If it is below the mean, it will be a negative number
- If it is above the mean it will be a positive number
- The size of those negative and positive numbers is determined by the standard deviation
- If the unnormalized data had a large standard deviation, the normalized values will be closer to 0

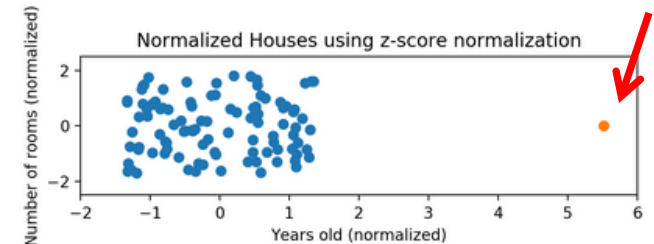
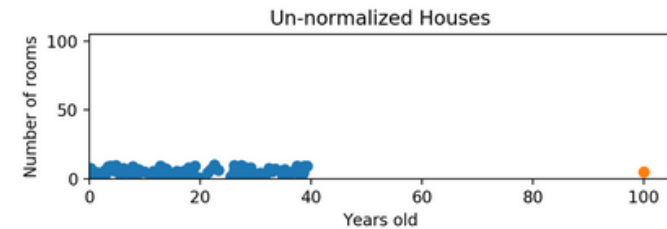
$$z = \frac{x - \mu}{\sigma}$$



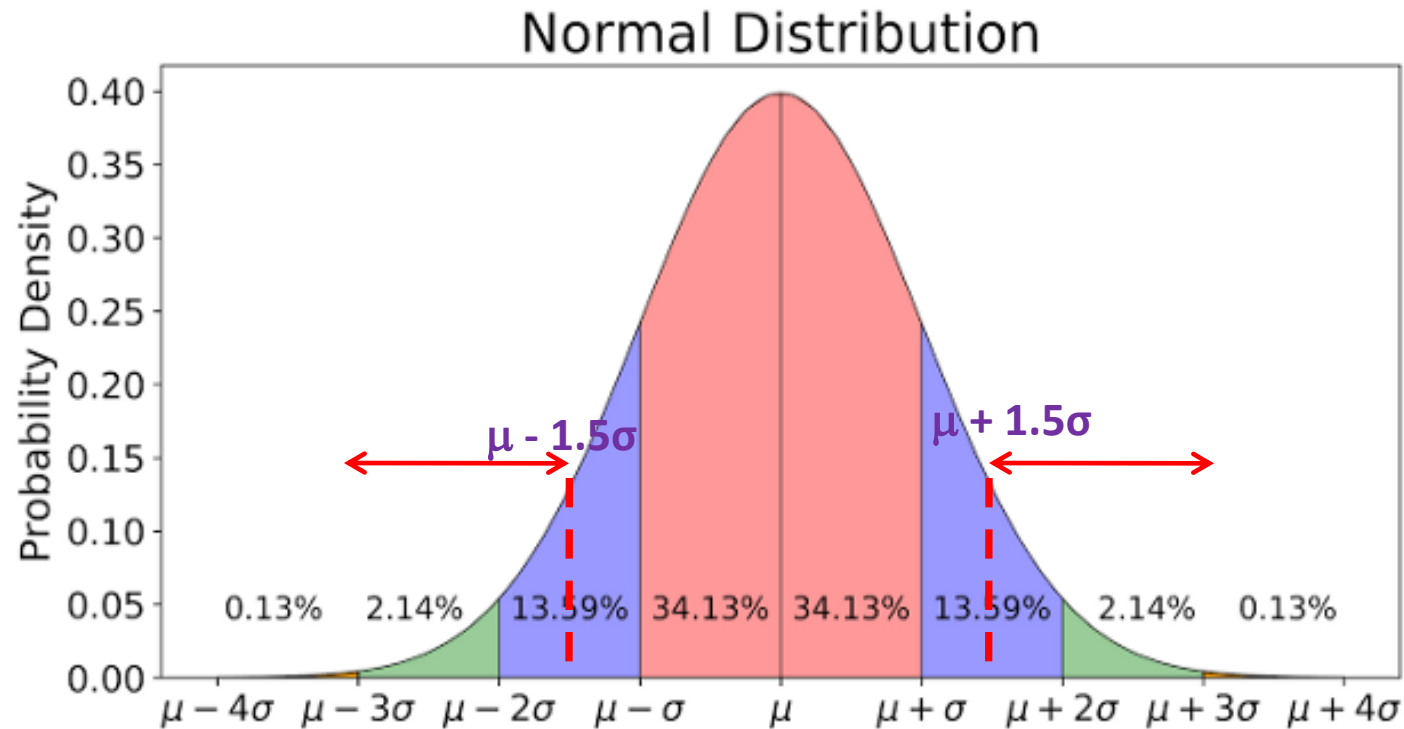
Variance -> How far a set of nos. is spread out from their avg
 SD -> On an avg, how much a data differs from the mean

Zscore Standardization / Normalization

- **Handles outlier**
- While the data still *looks* squished, notice that the points are now on roughly the same scale for both features — almost all points are between -2 and 2 on both the x-axis and y-axis
- Only potential downside is that the features aren't on the *exact* same scale
- x-axis now has a range from about -1.5 to 1.5 while the y-axis has a range from about -2 to 2



Zscore Standardization / Normalization



Zscore Standardization / Normalization

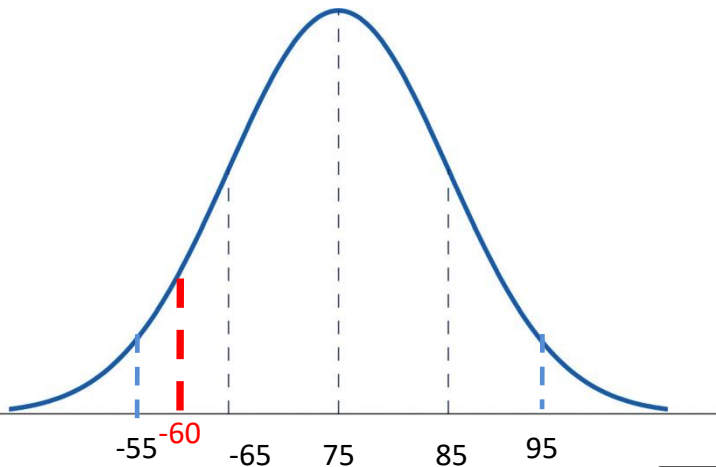


- $\mu = 75; \sigma = 10$
- $P(x > 60) = ?$

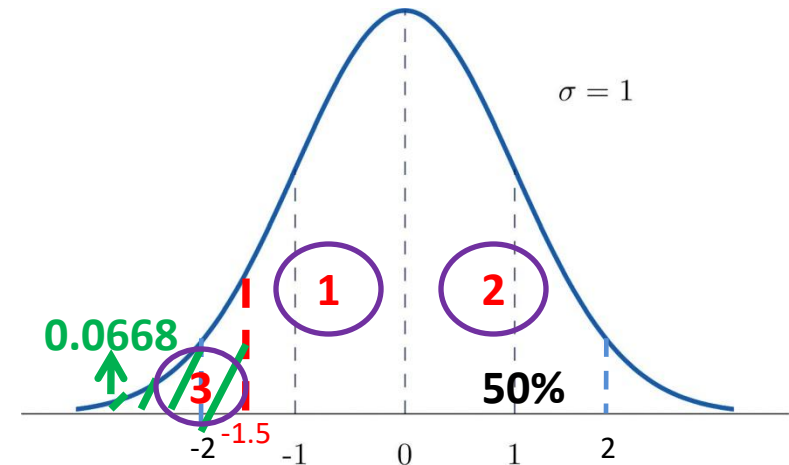
```
from sklearn.preprocessing import StandardScaler  
  
scaling=StandardScaler()  
  
scaling.fit_transform(df[['Alcohol','Malic']])
```

What is the probability that a new student will score > 60 (or) **What proportion of students who have scored > 60 or distribution of data above 60?**

$$z = \frac{x - \mu}{\sigma}$$



$$X + 6.68 + 50 = 100$$
$$X \approx 44\%$$



Probability of a new student scoring above 60 is $(44\% + 50\%) = 94\%$

Zscore Table

<i>z</i>	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
-2.0	.0228	.0222	.0217	.0212	.0207	.0202	.0197	.0192	.0188	.0183
-1.9	.0287	.0281	.0274	.0268	.0262	.0256	.0250	.0244	.0239	.0233
-1.8	.0359	.0351	.0344	.0336	.0329	.0322	.0314	.0307	.0301	.0294
-1.7	.0446	.0436	.0427	.0418	.0409	.0401	.0392	.0384	.0375	.0367
-1.6	.0548	.0537	.0526	.0516	.0505	.0495	.0485	.0475	.0465	.0455
-1.5	.0668	.0655	.0643	.0630	.0618	.0606	.0594	.0582	.0571	.0559
-1.4	.0808	.0793	.0778	.0764	.0749	.0735	.0721	.0708	.0694	.0681
-1.3	.0968	.0951	.0934	.0918	.0901	.0885	.0869	.0853	.0838	.0823
-1.2	.1151	.1131	.1112	.1093	.1075	.1056	.1038	.1020	.1003	.0985
-1.1	.1357	.1335	.1314	.1292	.1271	.1251	.1230	.1210	.1190	.1170
-1.0	.1587	.1562	.1539	.1515	.1492	.1469	.1446	.1423	.1401	.1379
-0.9	.1841	.1814	.1788	.1762	.1736	.1711	.1685	.1660	.1635	.1611
-0.8	.2119	.2090	.2061	.2033	.2005	.1977	.1949	.1922	.1894	.1867
-0.7	.2420	.2389	.2358	.2327	.2296	.2266	.2236	.2206	.2177	.2148
-0.6	.2743	.2709	.2676	.2643	.2611	.2578	.2546	.2514	.2483	.2451
-0.5	.3085	.3050	.3015	.2981	.2946	.2912	.2877	.2843	.2810	.2776
-0.4	.3446	.3409	.3372	.3336	.3300	.3264	.3228	.3192	.3156	.3121
-0.3	.3821	.3783	.3745	.3707	.3669	.3632	.3594	.3557	.3520	.3483
-0.2	.4207	.4168	.4129	.4090	.4052	.4013	.3974	.3936	.3897	.3859
-0.1	.4602	.4562	.4522	.4483	.4443	.4404	.4364	.4325	.4286	.4247
-0.0	.5000	.4960	.4920	.4880	.4840	.4801	.4761	.4721	.4681	.4641

Min Max Normalization

$$X_i^S = \frac{X_i - X_{min}}{X_{max} - X_{min}}$$

Weight Range = 160 to 200

Min = 160

Max = 200

Calculate for 180:

$$0 \leq X_i^S \leq 1$$

```
from sklearn.preprocessing import MinMaxScaler  
scaling=MinMaxScaler()  
scaling.fit_transform(df[['Alcohol','Malic']])
```

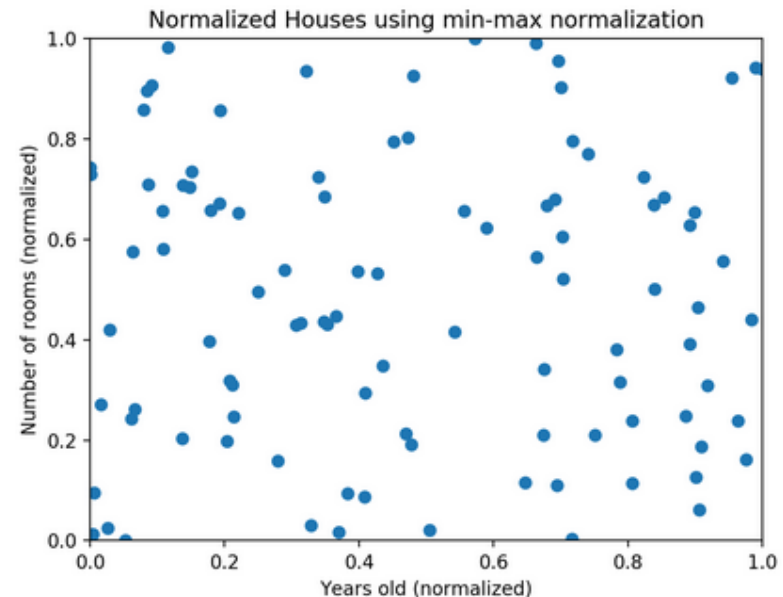
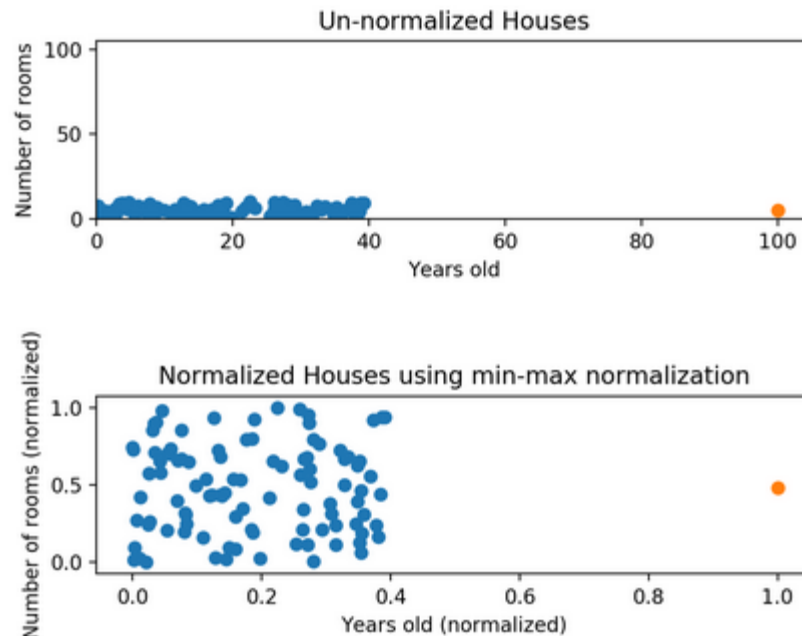
$$= (180 - 160) / 40 = 0.5$$

Min Max Normalization



- Minimum value gets transformed into a 0
- Maximum value gets transformed into a 1
- Other value gets transformed into a decimal between 0 and 1

$$X_i^S = \frac{X_i - X_{min}}{X_{max} - X_{min}}$$

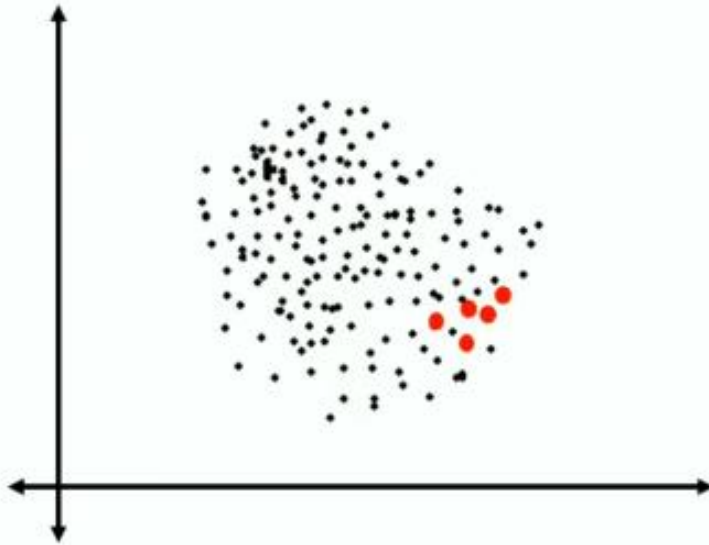
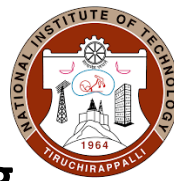


If we were to compare these points, then y-axis would dominate; the y-axis can differ by 1, but the x-axis can only differ by 0.4

Summary

- Min-max normalization: Guarantees all features will have the exact same scale but does not handle outliers well
- Z-score standardization / normalization: Handles outliers, but does not produce normalized data with the *exact* same scale
- Applications
 - Use when Euclidean distance or Gradient Descent is involved -> KNN, k-means, Regression, DL, CNN, ANL
 - No impact -> Decision Tree, Random Forest, Boosting, Bagging

Class Imbalance



Two Class Classification

No-Fraud → 99.5%

Fraud → 0.5%

Random Undersampling

Total Observations = 1,000

Fraudulent = 10 or 1%

Normal = 990 or 99%

Reduce normal to 90

Fraudulent = 10 or 10%

Leads to information loss

Random Oversampling

Total Observations = 1,000

Fraudulent = 10 or 1%

Normal = 990 or 99%

Increase fraudulent by 100

Fraudulent 110 or 10%

Leads to over sampling due to copying the same information

Use Case

- Credit card fraud
- Manufacturing defect
- Rare disease diagnosis
- Natural disaster

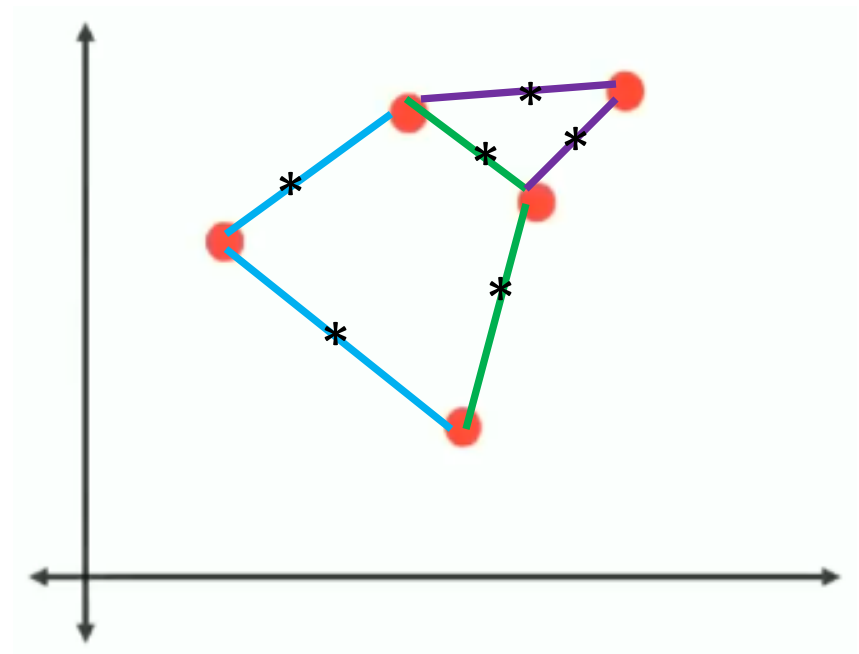
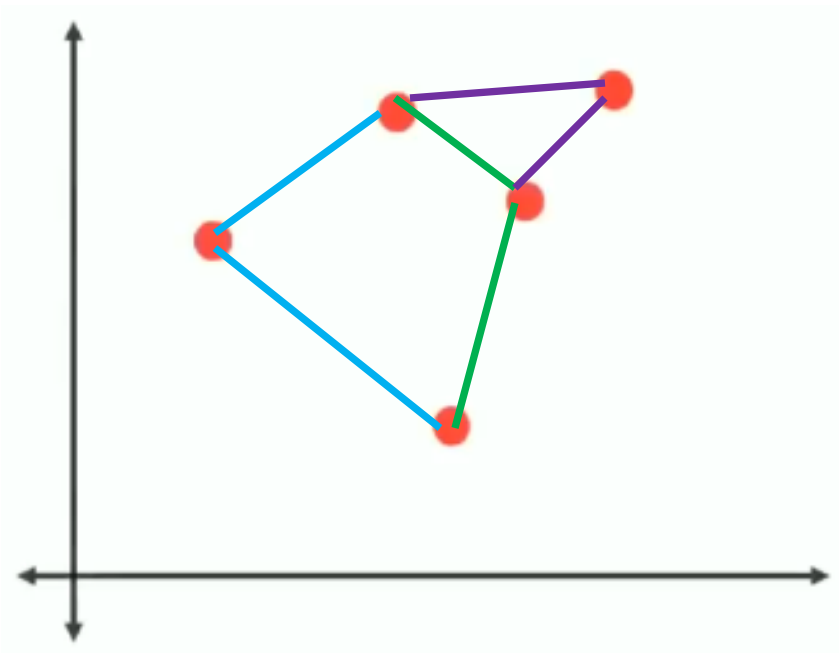
Class Imbalance

- Total number of a class of data (class 1) is far less than the total number of another class of data (class 2)
 - fraud detection, anomaly detection, medical diagnosis, oil spillage detection, facial recognition
- Works best when the number of instances of each classes are roughly equal
 - Biased -> Higher predictive accuracy over the majority class(es), but poorer predictive accuracy over the minority class
- SMOTE (Synthetic Minority Over-sampling Technique) is specifically designed for learning from imbalanced datasets

Class Imbalance - SMOTE

- Combination of oversampling and undersampling
- Majority class is first under-sampled -> Randomly removes samples from the majority class population until the minority class becomes some specified percentage
- Oversampling is then used to build up the minority class -> Not by simple replication, rather it constructs via an algorithm
- Creates synthetic instances of the minority class
- New instances are created by:
 - Taking samples of the feature space and its nearest neighbors
 - Combining features of the target case with features of its neighbors

SMOTE



SMOTE

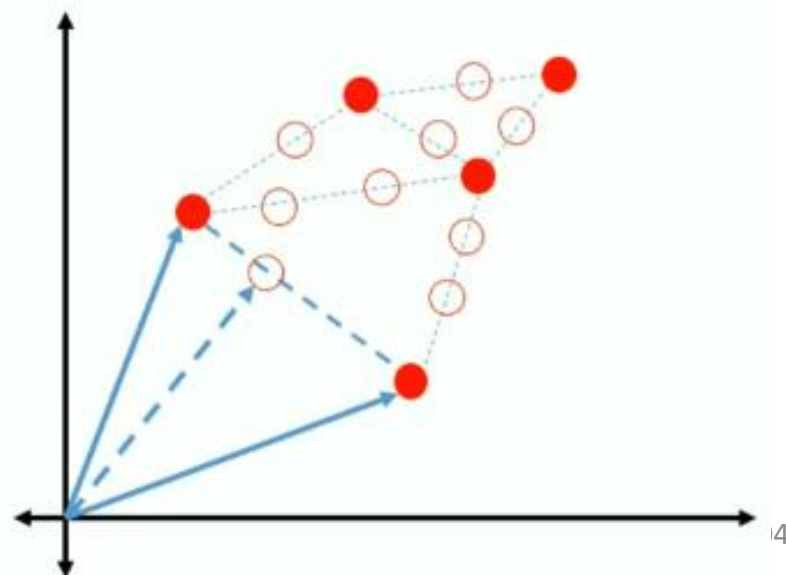
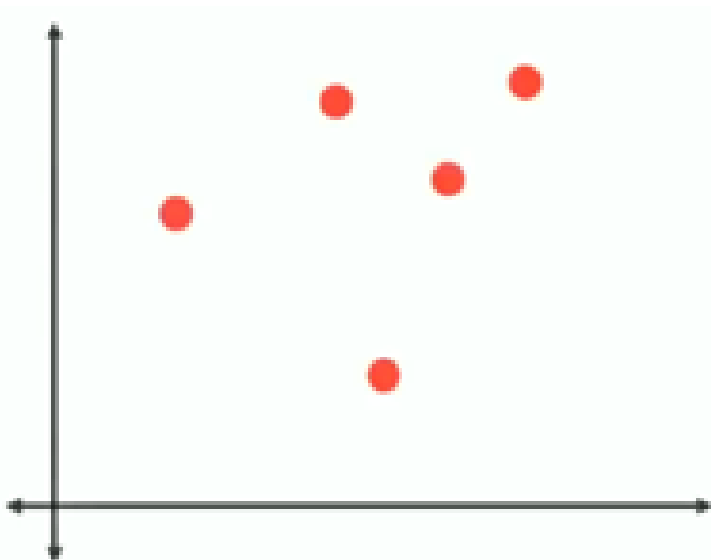


SMOTE Process

- Identify the feature vector and its nearest neighbour
- Take the **Linear distance** between the two
- Multiply the **distance** with a random number between 0 and 1
- Identify a new point on the line segment by adding the **result** to feature vector
- Repeat the process for identified feature vectors

A feature vector is a vector containing multiple elements about an object

$$\text{Synthetic Feature} = [(\text{Considered Feature} - \text{Nearest Neighbor}) * \text{Rand}(0,1)] + \text{Considered Feature}$$



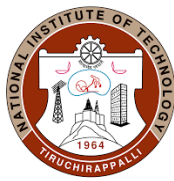
Curse of Dimensionality / Dimensionality Reduction

Curse of Dimensionality

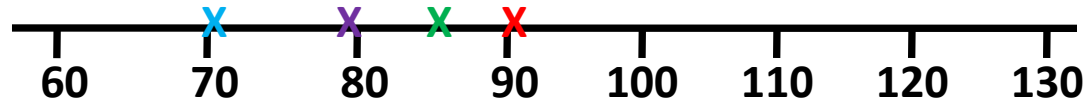
- One of the more important results in machine learning
- Number of data dimensions increases, it becomes more difficult to create a model that generalizes and applies well to data that are not in the training set
- Minimize the number of dimensions
 - Requires a combination of clever feature engineering and use of dimensionality reduction techniques
- Goal of dimensionality reduction -> Convert data into a lower dimensional representation, such that uninformative variance in the data is discarded
 - Lower dimensional data is more amenable to processing by machine learning algorithms

**Reduce the number of variables of a data set,
while preserving as much information as possible**

Principal Component Analysis

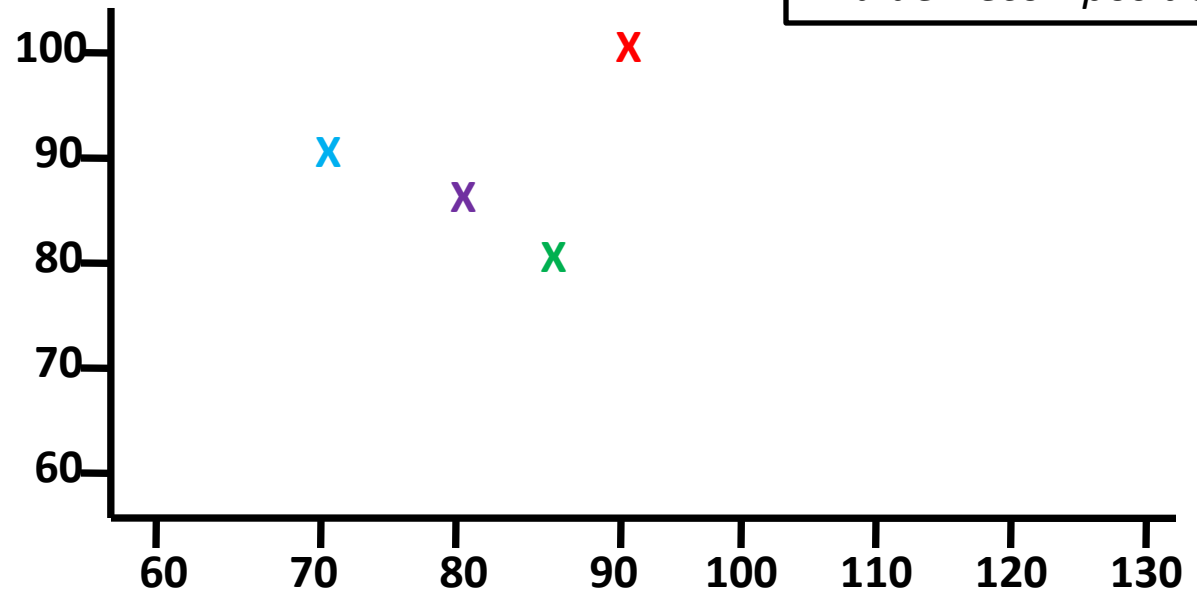


Roll No	English
1	90
2	85
3	80
4	70



PCA using Singular Value Decomposition

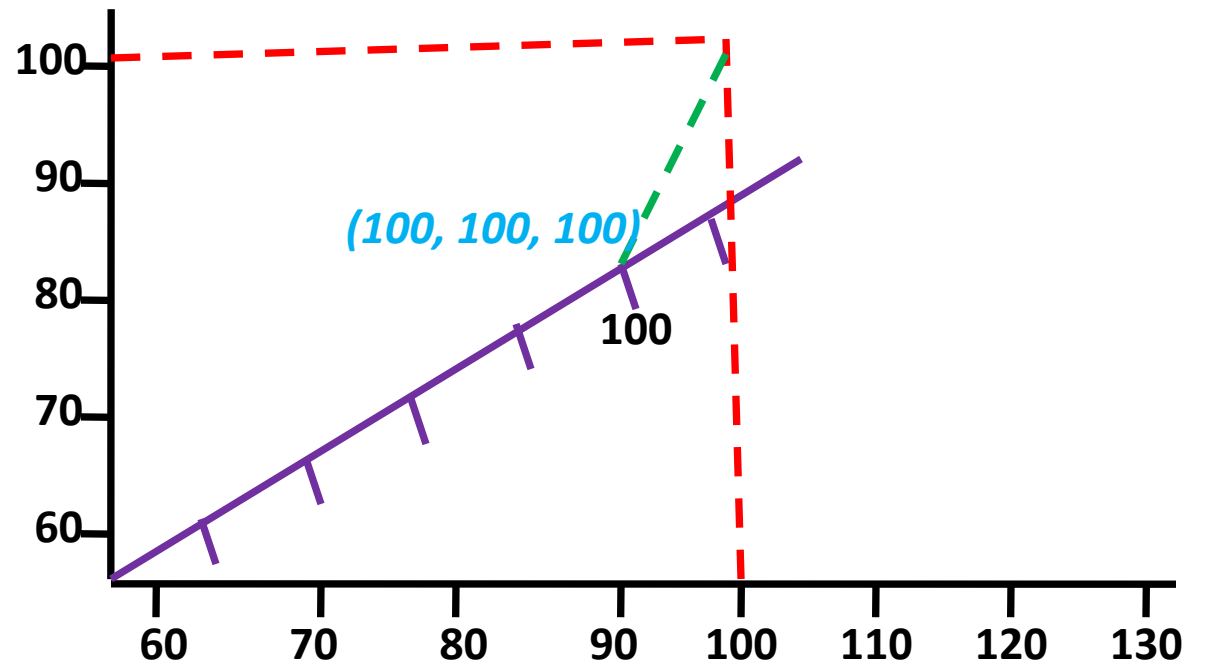
Roll No	English	Maths
1	90	100
2	85	80
3	80	85
4	70	90



Principal Component Analysis



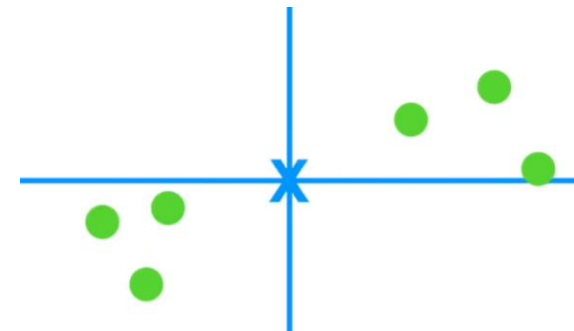
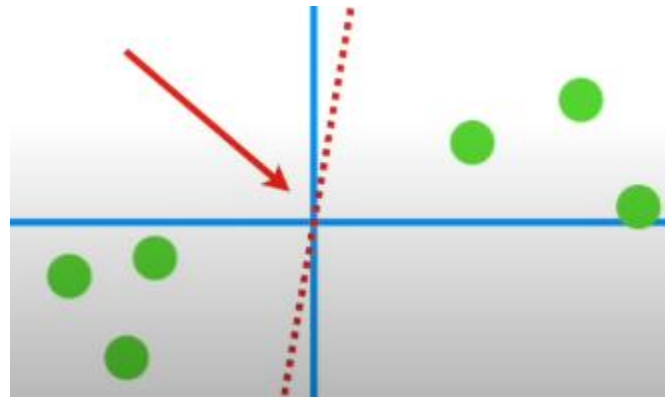
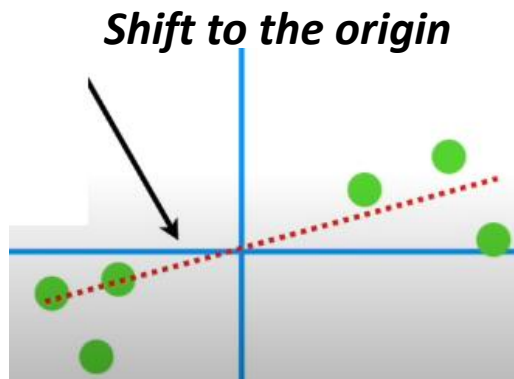
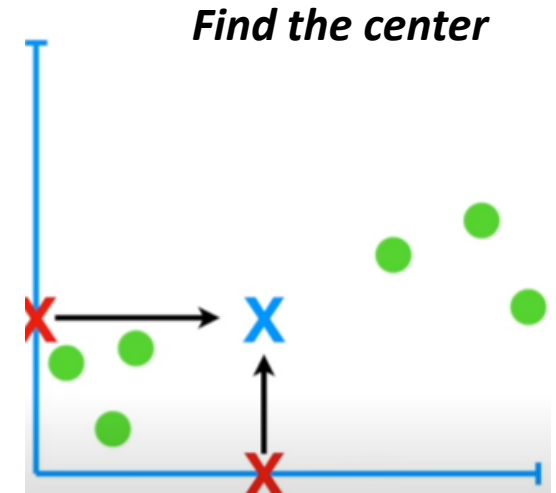
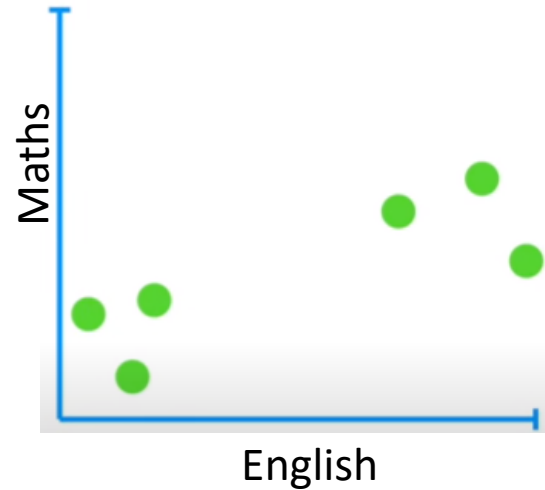
Roll No	English	Maths	Physics
1	100	100	100
2	85	80	90
3	80	85	100
4	70	90	85



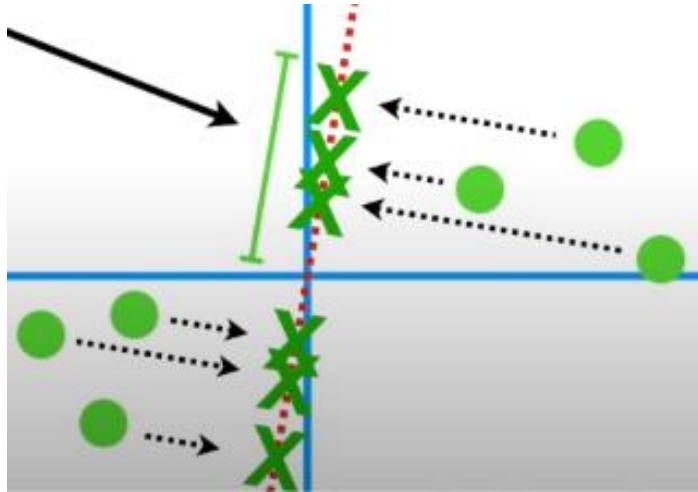
Principal Component Analysis



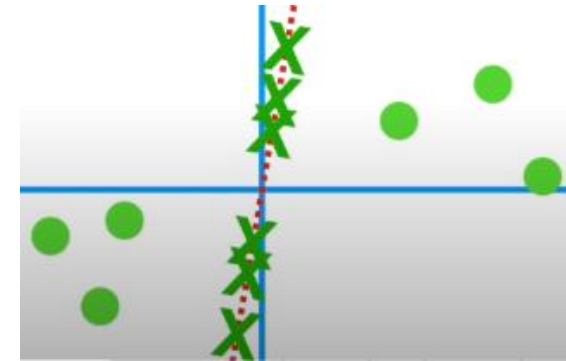
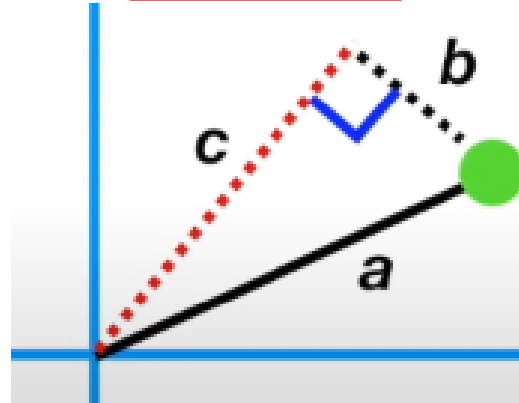
Roll No	English	Maths
1	90	100
2	85	80
3	80	85
4	70	90



How the Line is Chosen?



$$a^2 = b^2 + c^2$$



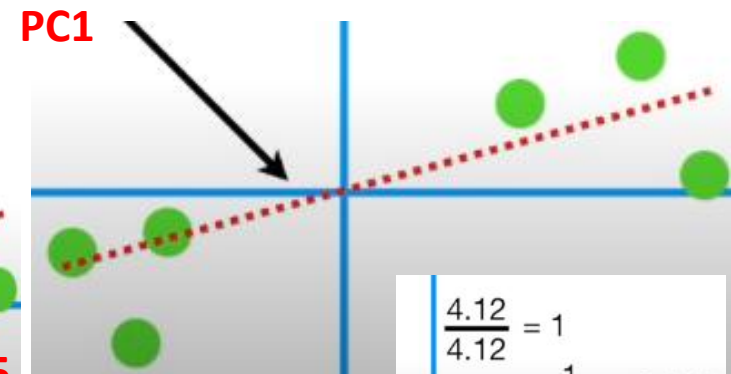
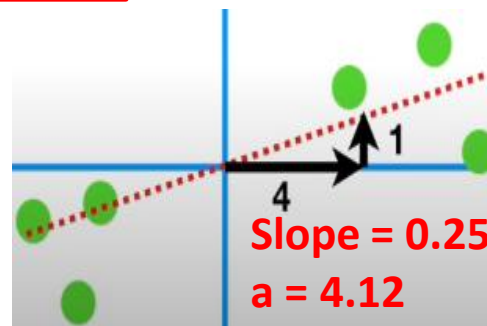
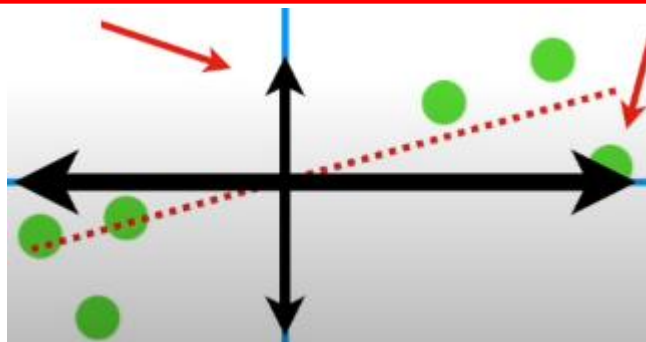
- Try to find the line that minimizes those distances

or

- Try to find the line that maximizes the distances from the projected points to the origin

$$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2 = \text{sum of squared distances} = \text{SS}(\text{distances})$$

Stop when the SS(distances) is very large

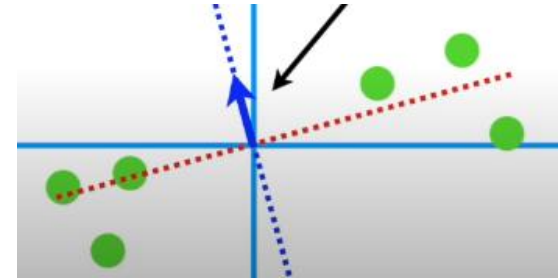
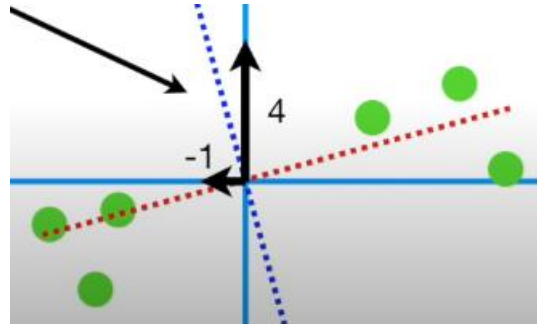
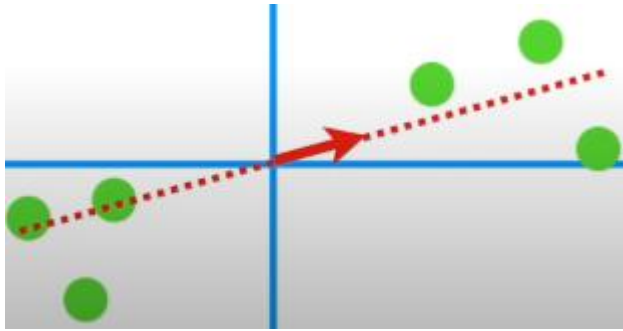


$$\frac{4.12}{4.12} = 1$$

$$\frac{1}{4.12} = 0.242$$

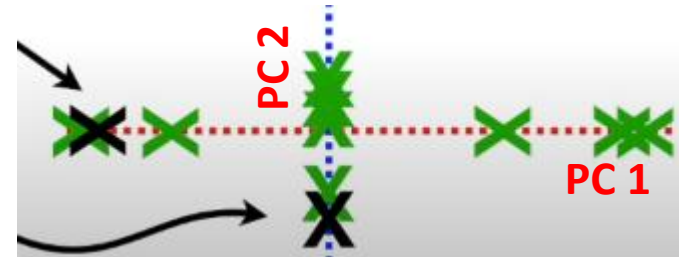
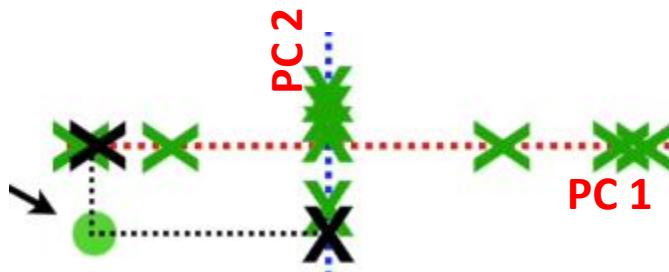
$$\frac{4}{4.12} = 0.97$$

Principal Component Analysis



- Unit vector consisting of 0.97 parts of x-axis and 0.242 parts of y-axis is called the **Singular Vector** or **Eigen Vector for PC1**
- 0.97 & 0.242 -> **Loading Scores**
- SS(distances for PC1) -> **Eigenvalue for PC1**
- Sqrt(Eigenvalue for PC1) -> **Singular Value for PC1**

- Unit vector consisting of -0.242 parts of x-axis and 0.97 parts of y-axis is called the **Singular Vector** or **Eigen Vector for PC1**
- -0.242 & 0.97 -> **Loading Scores**
- SS(distances for PC1) -> **Eigenvalue for PC1**



Principal Component Analysis

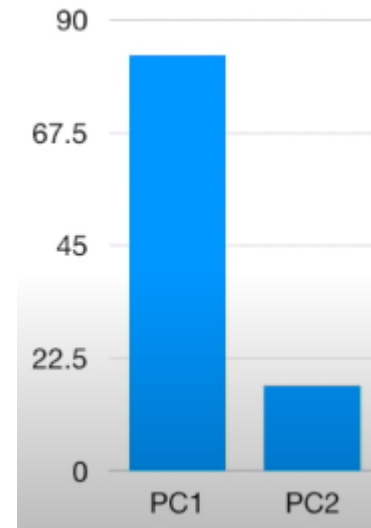
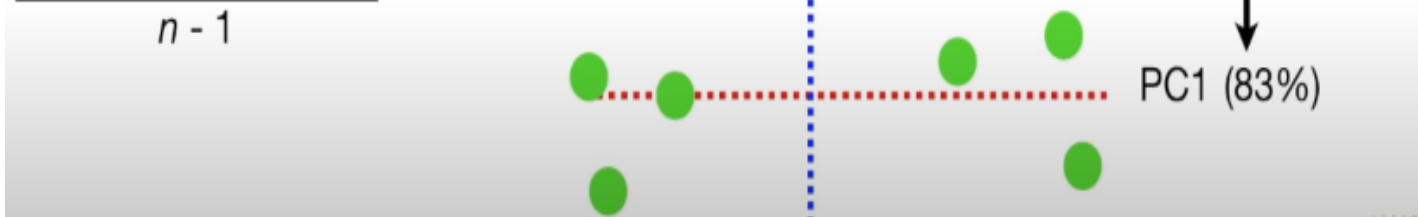
For the sake of the example, imagine that the Variation for **PC1 = 15**, and the variation for **PC2 = 3**.

That means that the total variation around both PCs is **15 + 3 = 18...**

$$\frac{SS(\text{distances for PC1})}{n - 1} = \text{Variation for PC1}$$

$$\frac{SS(\text{distances for PC2})}{n - 1} = \text{Variation for PC2}$$

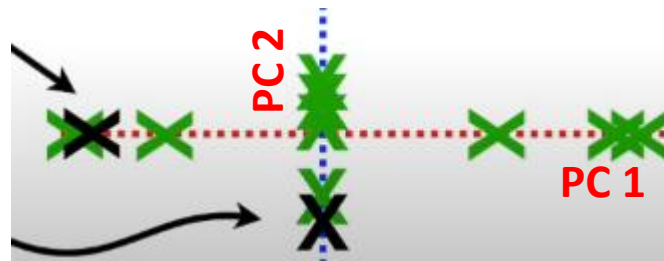
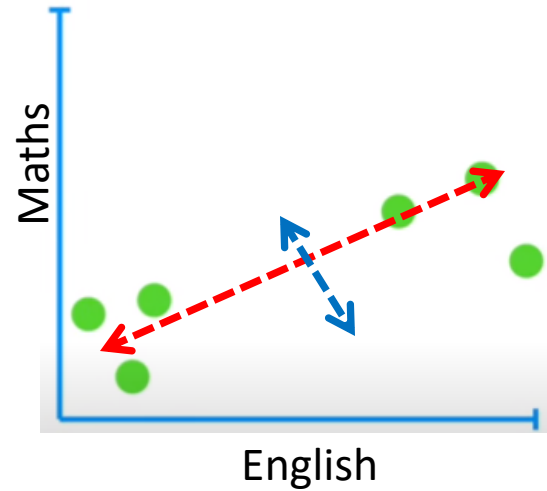
PC2 ...and that means PC1 accounts for **15 / 18 = 0.83 = 83%** of the total variation around the PCs.



Scree Plot -> Graphical representation of the percentages of variation that each PC accounts for

- <https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>
- <https://cmdlinetips.com/2018/03/pca-example-in-python-with-scikit-learn/>
- <https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html>

Principal Component Analysis



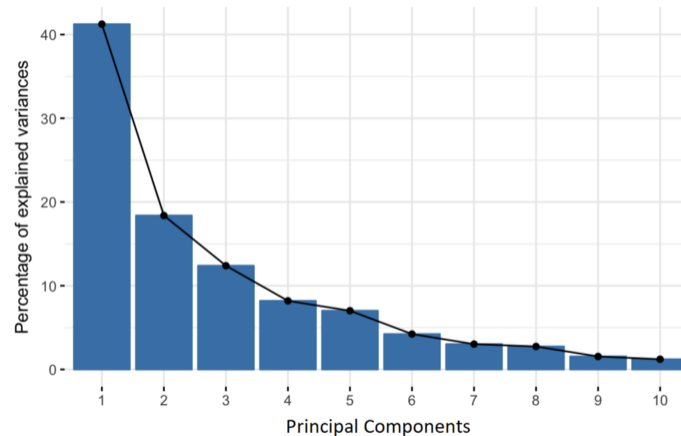
Principal Component Analysis



- Principal components are new variables that are constructed as linear combinations or mixtures of the initial variables

$$PC1 = 4X + 1 Y$$

- 10-dimensional data gives you 10 principal components, but PCA tries to put maximum possible information in the first component, then maximum remaining information in the second and so on



- Adv:
 - Allows to plot \geq 3D data in a 2D plot
 - Allows to identify which features are important

Principal Component Analysis

- Geometrically speaking, principal components represent the directions of the data that explain a maximal amount of variance, that is to say, the lines that capture most information of the data
- Relationship between variance and information here, is that, the larger the variance carried by a line, the larger the dispersion of the data points along it, and the larger the dispersion along a line, the more the information it has
- To put all this simply, just think of principal components as new axes that provide the best angle to see and evaluate the data, so that the differences between the observations are better visible

Principal Component Analysis



- Statistical procedure that uses a mathematical transformation to convert your training dataset into a set of values of linearly uncorrelated variables called principal components
- The number of principal components is less than or equal to the number of original attributes in your training set
- The transformation is defined in such a way that:
 - First principal component has the largest possible variance and accounts for as much of the variability in the data as possible, and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to (uncorrelated with) the preceding components

Principal Component Analysis

- Principal components are orthogonal because they are the eigenvectors of the covariance matrix, which is symmetric
- Thus, the original data is approximated by data that has many fewer dimensions and that summarizes well the original data
- PCA is sensitive to the relative scaling of the original variables, so your dataset must be normalized prior to PCA for good results
- You can view PCA as a data-mining technique
- High-dimensional data is replaced by its projection onto the most important axes
- These axes are the ones corresponding to the largest eigenvalues

Covariance Matrix

- Is a $p \times p$ symmetric matrix (where p is the number of dimensions)
- For 3 variables – x, y and z

PCA using Covariance Matrix

$$\begin{bmatrix} \text{Cov}(x, x) & \text{Cov}(x, y) & \text{Cov}(x, z) \\ \text{Cov}(y, x) & \text{Cov}(y, y) & \text{Cov}(y, z) \\ \text{Cov}(z, x) & \text{Cov}(z, y) & \text{Cov}(z, z) \end{bmatrix}$$

- If positive then: the two variables increase or decrease together (correlated)
- If negative then: One increases when the other decreases (Inversely correlated)

Eigen Vector and Eigen Value



$$v1 = \begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix} \quad \lambda_1 = 1.284028$$

$$v2 = \begin{bmatrix} -0.7351785 \\ 0.6778736 \end{bmatrix} \quad \lambda_2 = 0.04908323$$

**This is just an example and
not for the previous slide's
covariance matrix**

- If we rank the eigenvalues in descending order, we get $\lambda_1 > \lambda_2$, which means that the eigenvector that corresponds to the first principal component (PC1) is $v1$ and the one that corresponds to the second component (PC2) is $v2$
- After having the principal components, to compute the percentage of variance (information) accounted for by each component, we divide the eigenvalue of each component by the sum of eigenvalues
- If we apply this on the example above, we find that PC1 and PC2 carry respectively 96% and 4% of the variance of the data

Summary

- Data Preprocessing
 - Feature Encoding
 - Feature Engineering
 - Feature Binning
 - Feature Scaling
 - Feature Selection
 - Dimensionality Reduction
 - Data Visualization
 - Outlier Detection and Handling
 - Class Imbalance
- Cross Validation
 - Evaluation Metrics
 - ML Algorithms

Identifying Outliers

Using Built-in Functions:

- Describe
- IQR
- Skewness
- Zscore

Using Visualization Techniques:

- Box Plot
- Histogram
- Scatter Plot

Handling Outliers:

First, mark the outliers

- Quantile-based Flooring and Capping
- Trimming
- Rescale / Transform
- Replace with Median Value

Handling Outlier

- <https://www.pluralsight.com/guides/cleaning-up-data-from-outliers>
- <https://www.dezyre.com/recipes/deal-with-outliers-in-python>
- <https://medium.com/analytics-vidhya/outlier-treatment-9bbe87384d02>

Splitting the Dataset

Shape	Diag	# of Diag	Length (in cm)	Height (in cm)	Class
Rectangle	Yes	2	100	10	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	100	10	1
Rectangle	Yes	2	100	10	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	1000	100	1
Rectangle	Yes	2	1000	100	1
Square	Yes	2	100	100	0
Triangle	No	0	100	1000	0
Circle	No	0	100	100	0
Diamond	Yes	2	100	100	0
Rectangle	Yes	2	100	5	1

Training
Dataset

Testing
Dataset

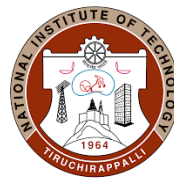
Introduction

- Machine learning models ought to be able to give accurate predictions in order to create real value for a given organization
- While training a model is a key step, how the model generalizes on unseen data is an equally important aspect that should be considered in every machine learning pipeline
- We need to know
 - Whether it actually works
 - If we can trust its predictions
 - Could the model be merely memorizing the data it is fed with
 - ✓ Unable to make good predictions on future samples, or samples that it hasn't seen before?
- Techniques used in evaluating how well a machine learning model generalizes to new, previously unseen data

Model Evaluation Techniques

- Methods for evaluating a model's performance are divided into 2 categories -> Holdout and Cross-validation
- Both methods use a test set (i.e data not seen by the model) to evaluate model performance
- It's not recommended to use the data we used to build the model to evaluate it
 - This is because our model will simply remember the whole training set and will therefore always predict the correct label for any point in the training set -> Overfitting

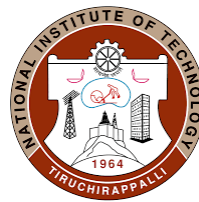
Holdout



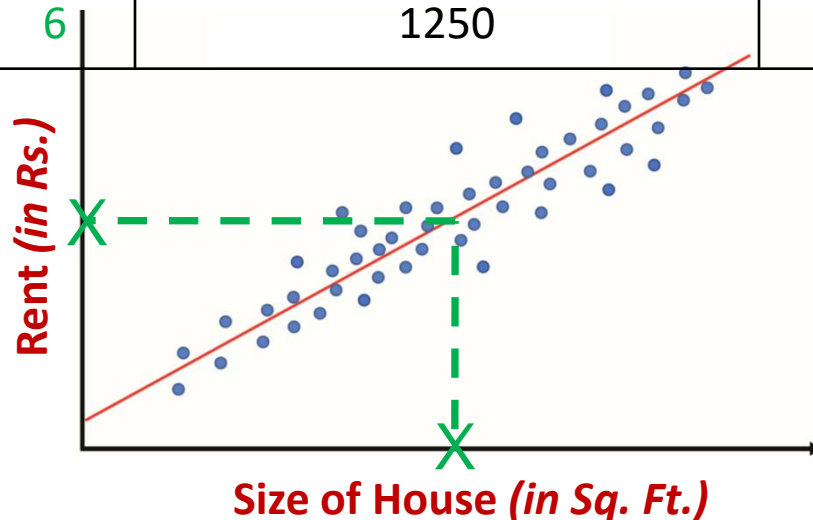
- Purpose of holdout evaluation is to test a model on different data than it was trained on
- This provides an unbiased estimate of learning performance
- Dataset is *randomly* divided into three subsets
- **Training Set** -> Subset of the dataset used to build predictive models
- **Validation Set** -> Subset of the dataset used to assess the performance of the model built in the training phase
 - Provides a test platform for fine-tuning a model's parameters and selecting the best performing model
- **Test Set** (Unseen Data) -> Subset of the dataset used to assess the likely future performance of a model
 - If a model fits to the training set much better than it fits the test set, overfitting is probably the cause
- Holdout approach is useful because of its speed, simplicity and flexibility
- This technique is often associated with high variability since differences in the training and test dataset can result in meaningful differences in the estimate of accuracy

Regression – Linear Regression Algorithm

(Supervised Learning)



ID	Size of House (in Sq. Ft.)	No. of Rooms	Rent (in Rs.)
1	1200	5	12,000
2	1500	4	15,000
3	1600	3	10,000
4	1000	6	11,000
5	1300	4	11,500
6	1250	3	12,000



Size of House (in Sq. Ft.)	No. of Rooms	Rent (in Rs.)
1400	3	???

Holdout

split data

```
data = ...
```

```
train, validation, test = split(data)
```

tune model hyperparameters

```
parameters = ...
```

```
for params in parameters:
```

```
    model = fit(train, params)
```

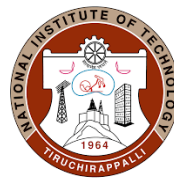
```
    skill = evaluate(model, validation)
```

evaluate final model for comparison with other models

```
model = fit(train)
```

```
skill = evaluate(model, test)
```


Cross-Validation



- Cross-validation is a technique that involves partitioning the original observation dataset into a training set, used to train the model, and an independent set used to evaluate the analysis
- We'd be “testing” our machine learning model in the “training” phase to check for overfitting and to get an idea about how our machine learning model will generalize to independent data (test data set)
- Can also be used to compare the performance of different machine learning models on the same data set and can also be helpful in selecting the values for a model's parameters that maximize the accuracy of the model — also known as parameter tuning
- Types of cross-validation
 - Leave-One-Out
 - K-fold
 - Stratified
 - Time-Series

Leave-One-Out Cross Validation



No. of Samples = 6

1 Sample	5 Samples				Iteration 1	
1 Sample	1 Sample	4 Samples			Iteration 2	
2 Samples		1 Sample	3 Samples		Iteration 3	
3 Samples			1 Sample	2 Samples	Iteration 4	
4 Samples				1 Sample	2 Samples	Iteration 5
5 Samples					1 sample	Iteration 6

Disadv:

- No. of iterations
- Low bias

- White -> Test Sample
- Blue -> Training Sample

K-fold Cross Validation

- Select, random value for “k”
- Let, $k = 5 \rightarrow$ Means 5 experiments
- Data is first partitioned into 5 parts of (approximately) equal size

200 Samples	800 Samples			Experiment 1 (Accuracy 1)
200 Samples	200 Samples	600 Samples		Experiment 2 (Accuracy 2)
400 Samples		200 Samples	400 Samples	Experiment 3 (Accuracy 3)
600 samples			200 Samples	Experiment 4 (Accuracy 4)
800 Samples				Experiment 5 (Accuracy 5)

- Error estimation is averaged over all k trials to get the total effectiveness of our model

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

- White \rightarrow Test Sample
- Blue \rightarrow Training Sample

Why random state?

CT 1 (Student scored B Grade):

Classroom



+

Tuition / Coaching Centre



CT 2 (Student scored S Grade):

Classroom



What do you observe?

Why random state?

CT 1 (Student scored B Grade):

Classroom



+

Tuition / Coaching Centre



CT 2 (Student scored S Grade):

Classroom



1. Student has performed better
2. Teacher is different in scenario 2
3. Students are different in scenario 2

Cannot figure out whether the tuition / coaching centre has played a role

Why random state?

CT 1 (Student scored B Grade):

Classroom



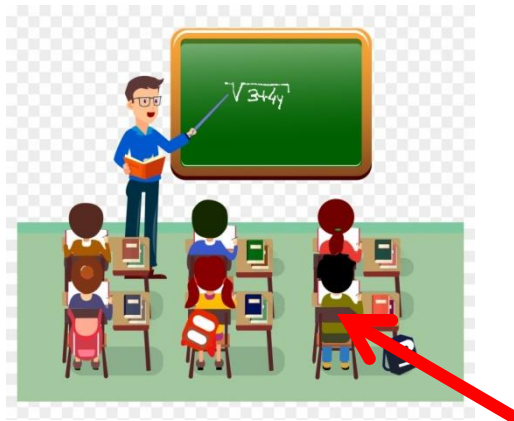
+

Tuition / Coaching Centre



CT 2 (Student scored S Grade):

Classroom



1. Teacher is same in both scenarios
2. Students are same in both scenarios
3. **Student has performed better →
Tuition / Coaching Centre has played
a role**

Stratified Cross Validation



200 Samples	800 Samples		
200 Samples	200 Samples	600 Samples	
400 Samples		200 Samples	400 Samples
600 samples			200 Samples
800 Samples			200 Samples

- 1000 Samples (600 -> Yes; 400 -> No)
- Makes sure that proper instances of each class is present in both Train and Test Dataset
- Eg: 70% + 30% or 60% + 40%

- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

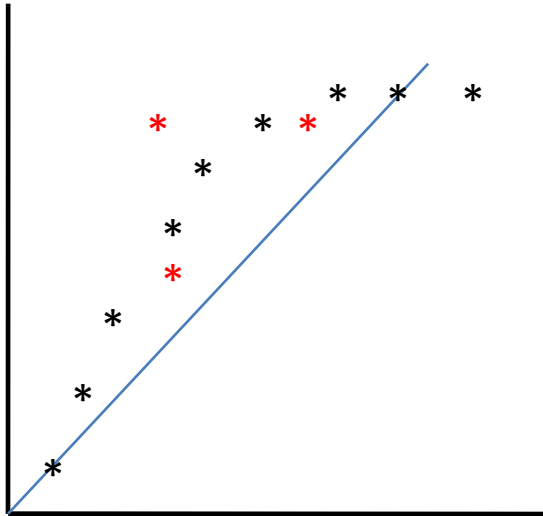
Time Series Cross Validation

Day	Stock 1	Stock 2	Stock 3	Stock 4
Day 1				
Day 2				
Day 3				
Day 4				
Day 5				

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
Day 3	Day 4	Day 5	Day 6	Day 7	Day 8
Day 4	Day 5	Day 6	Day 7	Day 8	Day 9
Day 5	Day 6	Day 7	Day 8	Day 9	Day 10

Bias vs Variance

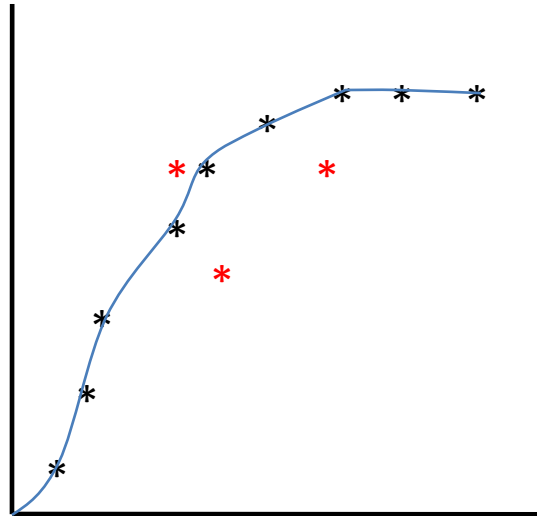
Under-Fitting



Training Dataset:
Error is high (**High Bias**)

New / Test Dataset:
Error rate is high (**High Variance**)

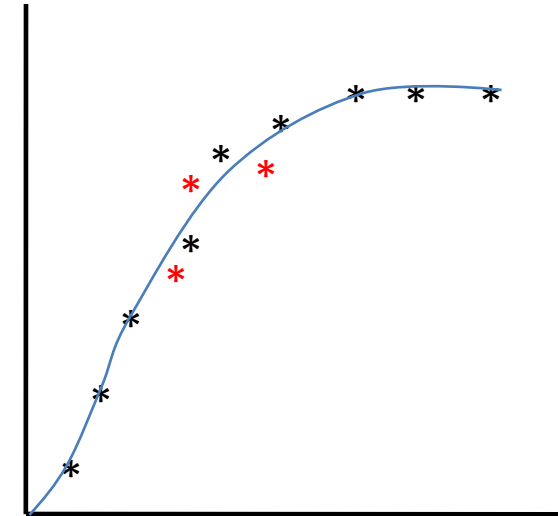
Over-Fitting



Training Dataset:
Error is low (**Low Bias**)

New / Test Dataset:
Error rate is high (**High Variance**)

Perfect-Fitting



Training Dataset:
Error is low (**Low Bias**)

New / Test Dataset:
Error rate is low (**Low Variance**)

Model Evaluation Metrics

- Model evaluation metrics are required to quantify model performance
- Choice of evaluation metrics depends on a given machine learning task (such as classification, regression, ranking, clustering, topic modeling, among others)
- Some metrics, such as precision-recall, are useful for multiple tasks
- Supervised learning tasks such as classification and regression constitutes a majority of machine learning applications
- We focus on metrics for these two supervised learning models

Classification Metrics

- Classification Accuracy
- Confusion matrix
- Logarithmic Loss
- Area under curve (AUC)
- F-Measure or F1 Score

Classification Accuracy

- Accuracy is a common evaluation metric for classification problems
- Number of correct predictions made as a ratio of all predictions made

$$\text{Accuracy} = \frac{\text{No. of Correct Predictions}}{\text{Total No. of Predictions Made}}$$

- Disadv: Accuracy will yield misleading results if the data set is unbalanced
- Eg: If there were 95 cats and only 5 dogs in the data, a particular classifier might classify all the observations as cats
 - Overall accuracy would be 95%, but in more detail the classifier would have a 100% recognition rate (sensitivity) for the cat class but a 0% recognition rate for the dog class

Confusion Matrix

- TP, TN, FP, FN
- Classification system trained to distinguish between cats and dogs
- Sample of 13 animals —> 8 cats and 5 dogs

-> Actual = [1,1,1,1,1,1,1,1, 0,0,0,0,0]

-> Prediction = [0,0,0,1,1,1,1,1, 0,0,0,1,1]

	Actual Class		
		Cat	Dog
	Predicted Class		
	Cat	5	2
	Dog	3	3

	Actual Class		
		Cat	Non-cat
	Predicted Class		
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN

Accuracy

- **Accuracy:** Measure of the effectiveness of the algorithm

$$Accuracy(ACC) = \frac{\sum TP + \sum TN}{\sum TP + TN + FP + FN} = \frac{5 + 3}{5 + 3 + 2 + 3} = 61.5\%$$

	Actual Class		
Predicted Class		Cat	Dog
	Cat	5	2
	Dog	3	3

	Actual Class		
Predicted Class		Cat	Non-cat
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN

Drawback of Accuracy

$$Accuracy(ACC) = \frac{\sum TP + \sum TN}{\sum TP + TN + FP + FN} = \frac{5 + 3}{5 + 3 + 2 + 3} = 61.5\%$$

	Actual Class	
Predicted Class	10 TP	15 FN
	25 FP	100 TN

- No. of Samples = 150
- Class 1 = 35 (*Aim to detect*)
- Class 0 = 115

- Accuracy = $(10 + 100) / (10 + 100 + 25 + 15) = 73.3\%$

	Actual Class	
Predicted Class	0 TP	25 FN
	0 FP	125 TN

	Actual Class	
Predicted Class	0 TP	35 FN
	0 FP	115 TN

- Accuracy = $(0 + 115) / (0 + 115 + 0 + 35) = 76.7\%$

Precision & Recall

- **Precision:** Ratio of correctly predicted positive values to the total predicted positive values

‘how much the model is right when it says it is right’

$$Precision = \frac{TP}{TP + FP} = \frac{5}{5 + 2} = 71.4\%$$

- **Recall/Sensitivity/True Positive Rate:** Percentage of total relevant results correctly classified by the algorithm

‘how much extra right ones, the model missed when it showed the right ones’

$$Recall = \frac{TP}{TP + FN} = \frac{5}{5 + 3} = 62.5\%$$

Precision and Recall are used when class imbalance is present and also the detection of positive classes is very important

	Actual Class		
		Cat	Dog
	Predicted Class		
	Cat	5	2
	Dog	3	3

	Actual Class		
		Cat	Non-cat
	Predicted Class		
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN ⁴

F1 Score

- **F1 Score:** Weighted harmonic mean of Precision and Recall
- Measure of a test's accuracy that considers both precision and recall of the test to compute the score

$$F1\ Score = 2 * \frac{Recall * Precision}{Recall + Precision} = 2 * \frac{0.714 * 0.625}{0.714 + 0.625} = 66.6\%$$

	Actual Class		
Predicted Class		Cat	Dog
	Cat	5	2
	Dog	3	3

	Actual Class		
Predicted Class		Cat	Non-cat
	Cat	5 TP	2 FP
	Non-cat	3 FN	3 TN

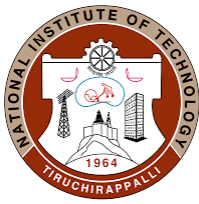
Specificity

- Percentage of negative instances out of the *total actual negative* instances

Like finding out how many healthy patients were not having cancer and were told they don't have cancer

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

$$\text{False Positive Rate} = (1 - \text{Specificity}) = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$



Thank You