Module 3

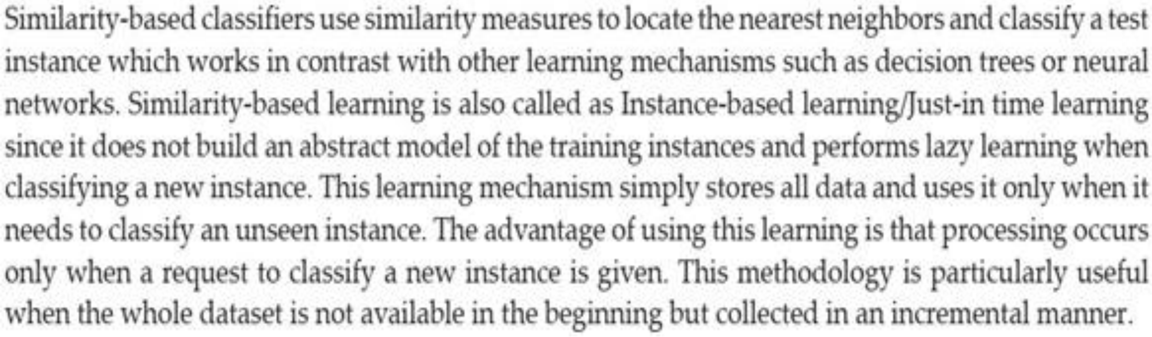
Topics: *Basics of Learning theory, Similarity Based Learning, Regression Analysis*

Textbook 2: Chapter 3 - 3.1 to 3.4, Chapter 4, chapter 5.1 to 5.4

Chapter 4

Similarity Based Learning

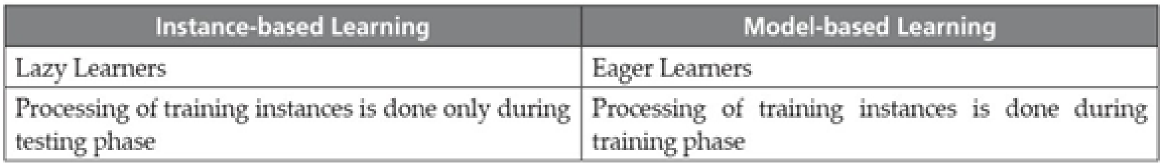
4.1 Similarity or Instance-based Learning

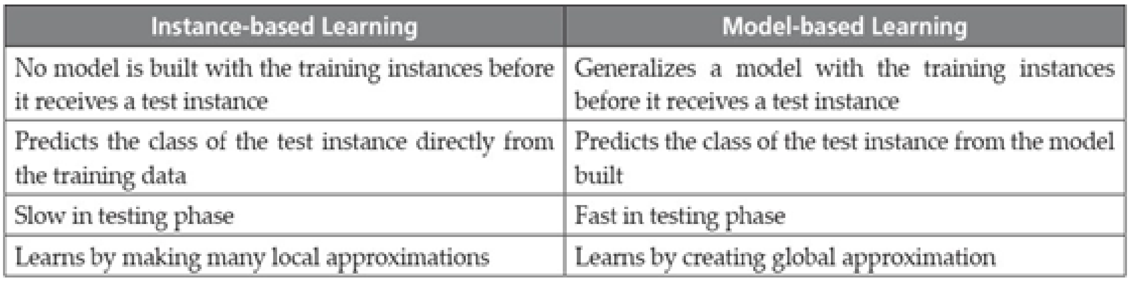


Similarity-based classifiers use similarity measures to locate the nearest neighbors and classify a test instance which works in contrast with other learning mechanisms such as decision trees or neural networks. Similarity-based learning is also called as Instance-based learning/Just-in time learning since it does not build an abstract model of the training instances and performs lazy learning when classifying a new instance. This learning mechanism simply stores all data and uses it only when it needs to classify an unseen instance. The advantage of using this learning is that processing occurs only when a request to classify a new instance is given. This methodology is particularly useful when the whole dataset is not available in the beginning but collected in an incremental manner.

4.1.1 Difference between Instance-and Model-based Learning

Table 4.1: Differences between Instance-based Learning and Model-based Learning





Instance-based Learning	Model-based Learning
Lazy Learners	Eager Learners
Processing of training instances is done only during testing phase	Processing of training instances is done during training phase
No model is built with the training instances before it receives a test instance	Generalizes a model with the training instances before it receives a test instance
Predicts the class of the test instance directly from the training data	Predicts the class of the test instance from the model built
Slow in testing phase	Fast in testing phase
Learns by making many local approximations	Learns by creating global approximation

Some examples of Instance-based Learning algorithms are:

a) KNN

b) Variants of KNN

c) Locally weighted regression

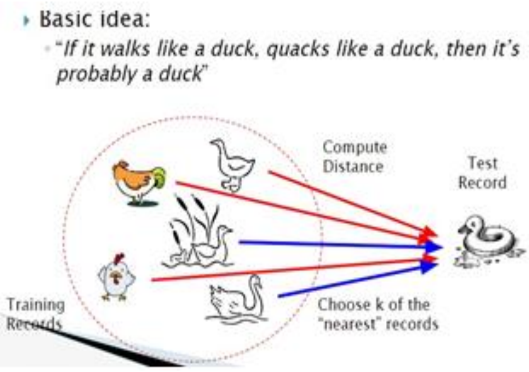
d) Learning vector quantization

e) Self-organizing maps

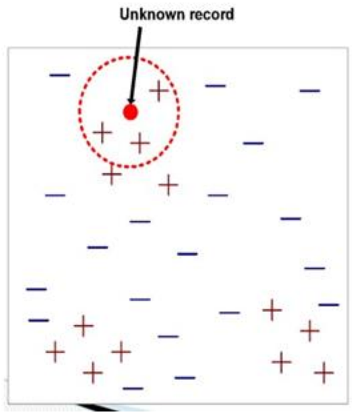
f) RBF networks

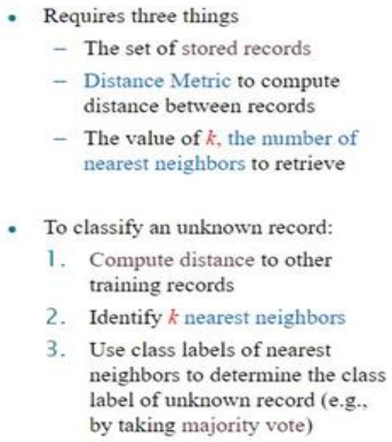
4.2 Nearest-Neighbor Learning

- A powerful classification algorithm used in pattern recognition.
- K nearest neighbors stores all available cases and classifies new cases based on a similarity measure (e.g distance function).
- One of the top data mining algorithms used today.
- A non-parametric lazy learning algorithm (An Instance based Learning method).
- Used for both classification and regression problems.



- Basic idea:
  - *"If it walks like a duck, quacks like a duck, then it's probably a duck"*





- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve
- To classify an unknown record:
  1. Compute distance to other training records
  2. Identify *k* nearest neighbors
  3. Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

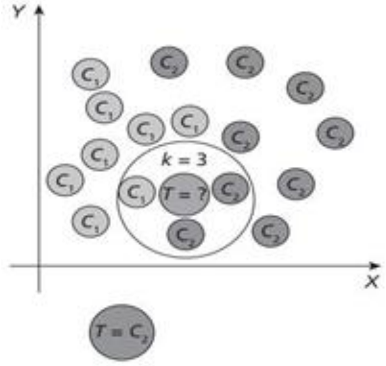
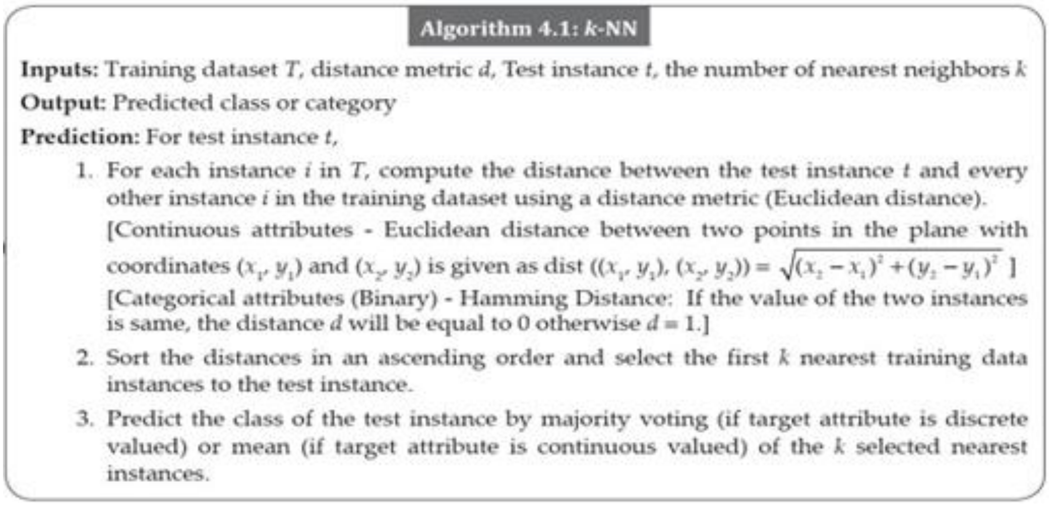


Figure 4.1: Visual Representation of *k*-Nearest Neighbor Learning

In the above diagram 4.1, 2 classes of objects called C1 and C2. When given a test instance T,the category of this test instance is determined by looking at the class ofk=3 nearest neighbors. Thus, the class of this test instance T ispredicted as C2.



Algorithm 4.1: *k*-NN

Inputs: Training dataset *T*, distance metric *d*, Test instance *t*, the number of nearest neighbors *k*

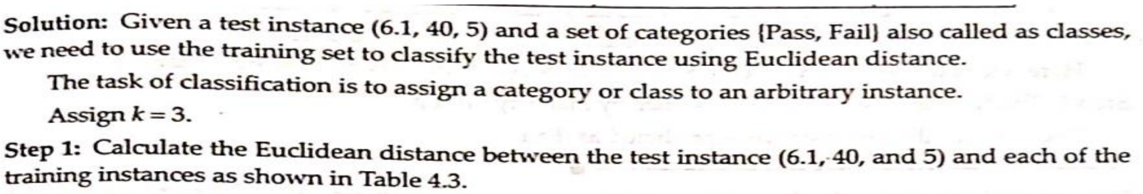
Output: Predicted class or category

Prediction: For test instance *t*,

1. For each instance *i* in *T*, compute the distance between the test instance *t* and every other instance *i* in the training dataset using a distance metric (Euclidean distance).
   [Continuous attributes - Euclidean distance between two points in the plane with coordinates (x_1, y_1) and (x_2, y_2) is given as dist $((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$]
   [Categorical attributes (Binary) - Hamming Distance: If the value of the two instances is same, the distance *d* will be equal to 0 otherwise $d = 1$.]
2. Sort the distances in an ascending order and select the first *k* nearest training data instances to the test instance.
3. Predict the class of the test instance by majority voting (if target attribute is discrete valued) or mean (if target attribute is continuous valued) of the *k* selected nearest instances.

Consider the student performance training, dataset of 8 data instances shown in Table 4.2 which describes the performance of individual students in a course and their CGPA obtained in the previous semesters. The independent attributes are CGPA, Assessment and Project. The target variable is 'Result' which is a discrete valued variable that takes two values 'Pass' or 'Fail. Based on the performance of a student, classify whether a student will pass or fail in that course.

Sl No.	CGPA	Assessment	Project Submitted	Result
1	9.2	85	8	Pass
2	8	80	7	Pass
3	8.5	81	8	Pass
4	6	45	5	Fail
5	6.5	50	4	Fail
6	8.2	72	7	Pass
7	5.8	38	5	Fail
8	8.9	91	9	Pass



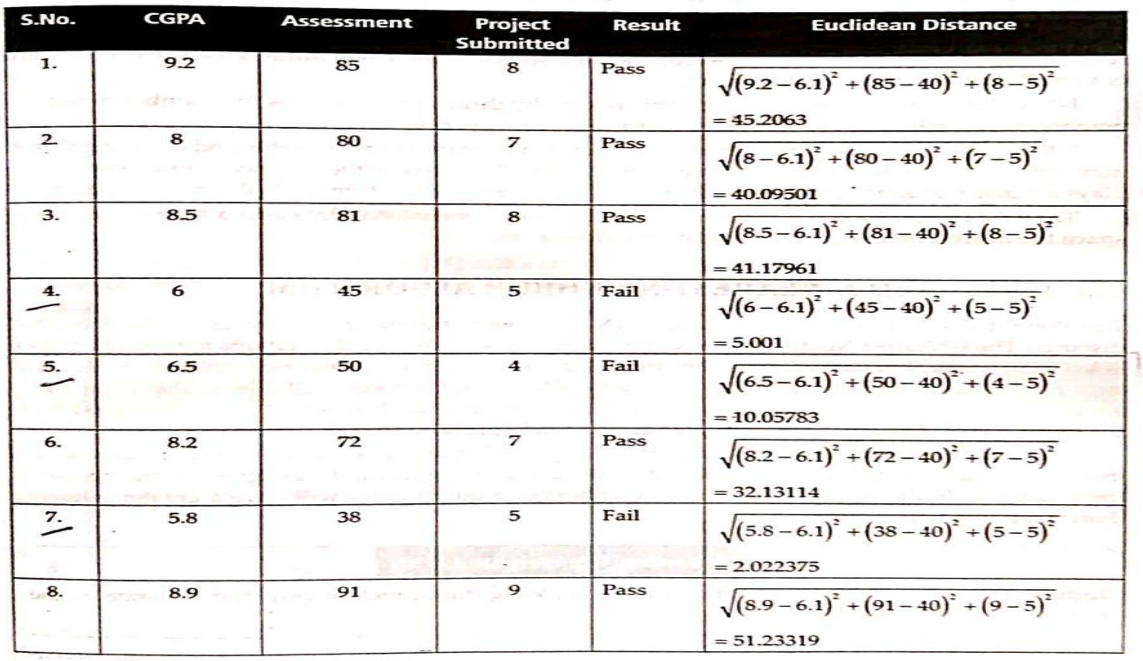
Solution: Given a test instance (6.1, 40, 5) and a set of categories {Pass, Fail} also called as classes, we need to use the training set to classify the test instance using Euclidean distance.

The task of classification is to assign a category or class to an arbitrary instance.

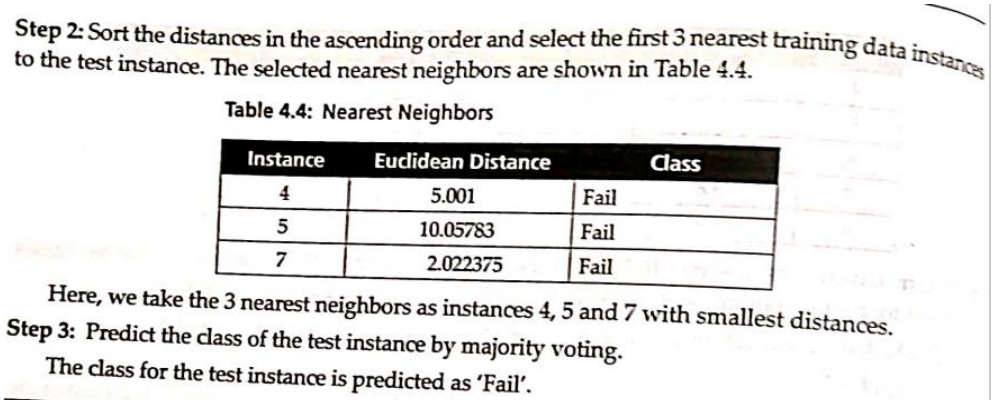
Assign $k = 3$.

Step 1: Calculate the Euclidean distance between the test instance (6.1, 40, and 5) and each of the training instances as shown in Table 4.3.

Table 4.3: Euclidean Distance



S.No.	CGPA	Assessment	Project Submitted	Result	Euclidean Distance
1.	9.2	85	8	Pass	$\sqrt{(9.2-6.1)^2+(85-40)^2+(8-5)^2}$ $= 45.2063$
2.	8	80	7	Pass	$\sqrt{(8-6.1)^2+(80-40)^2+(7-5)^2}$ $= 40.09501$
3.	8.5	81	8	Pass	$\sqrt{(8.5-6.1)^2+(81-40)^2+(8-5)^2}$ $= 41.17961$
4.	6	45	5	Fail	$\sqrt{(6-6.1)^2+(45-40)^2+(5-5)^2}$ $= 5.001$
5.	6.5	50	4	Fail	$\sqrt{(6.5-6.1)^2+(50-40)^2+(4-5)^2}$ $= 10.05783$
6.	8.2	72	7	Pass	$\sqrt{(8.2-6.1)^2+(72-40)^2+(7-5)^2}$ $= 32.13114$
7.	5.8	38	5	Fail	$\sqrt{(5.8-6.1)^2+(38-40)^2+(5-5)^2}$ $= 2.022375$
8.	8.9	91	9	Pass	$\sqrt{(8.9-6.1)^2+(91-40)^2+(9-5)^2}$ $= 51.23319$



Step 2: Sort the distances in the ascending order and select the first 3 nearest training data instances to the test instance. The selected nearest neighbors are shown in Table 4.4.

Table 4.4: Nearest Neighbors

Instance	Euclidean Distance	Class
4	5.001	Fail
5	10.05783	Fail
7	2.022375	Fail

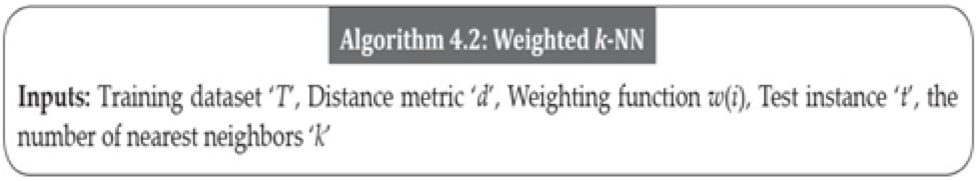
Here, we take the 3 nearest neighbors as instances 4, 5 and 7 with smallest distances.

Step 3: Predict the class of the test instance by majority voting.

The class for the test instance is predicted as 'Fail'.

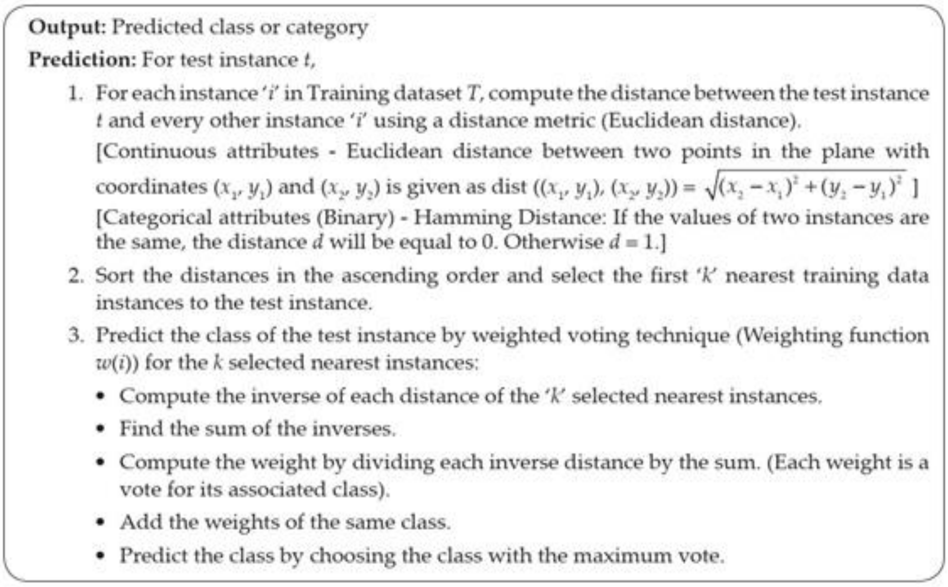
4.3 Weighted K-Nearest Neighbor Algorithm

The weighted KNN is an extension of k-NN.It chooses the neighbors by using the weighted distance. In weighted kNN, the nearest k points are given a weight using a function called as the kernel function. The intuition behind weighted kNN, is to give more weight to the points which are nearby and less weight to the points which are farther away.



Algorithm 4.2: Weighted *k*-NN

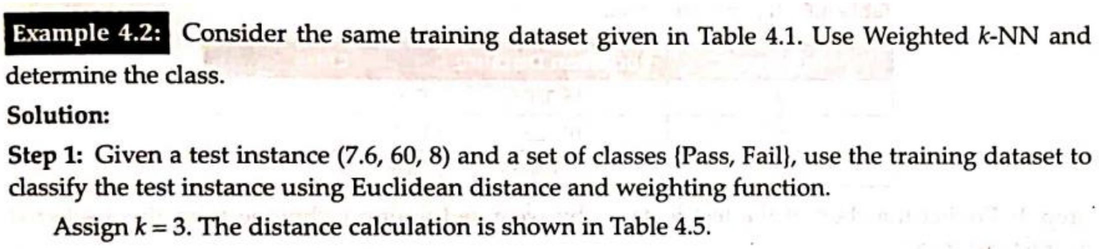
Inputs: Training dataset '*T*', Distance metric '*d*', Weighting function $w(i)$, Test instance '*t*', the number of nearest neighbors '*k*'



Output: Predicted class or category

Prediction: For test instance *t*,

1. For each instance '*i*' in Training dataset *T*, compute the distance between the test instance *t* and every other instance '*i*' using a distance metric (Euclidean distance).
   [Continuous attributes - Euclidean distance between two points in the plane with coordinates (x_1, y_1) and (x_2, y_2) is given as dist $((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$]
   [Categorical attributes (Binary) - Hamming Distance: If the values of two instances are the same, the distance *d* will be equal to 0. Otherwise *d* = 1.]
2. Sort the distances in the ascending order and select the first '*k*' nearest training data instances to the test instance.
3. Predict the class of the test instance by weighted voting technique (Weighting function *w*(*i*)) for the *k* selected nearest instances:
   - Compute the inverse of each distance of the '*k*' selected nearest instances.
   - Find the sum of the inverses.
   - Compute the weight by dividing each inverse distance by the sum. (Each weight is a vote for its associated class).
   - Add the weights of the same class.
   - Predict the class by choosing the class with the maximum vote.



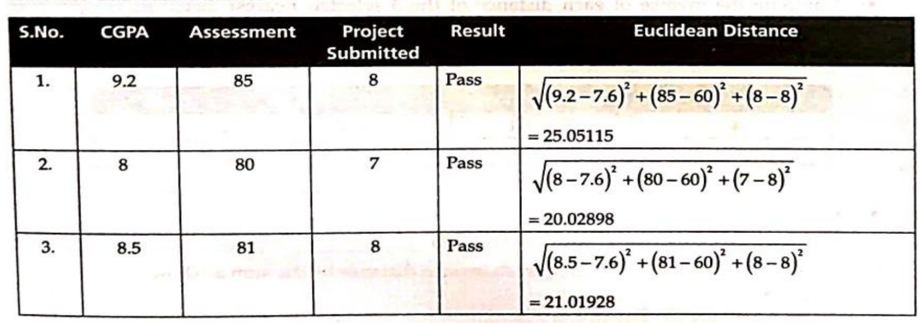
Example 4.2: Consider the same training dataset given in Table 4.1. Use Weighted *k*-NN and determine the class.

Solution:

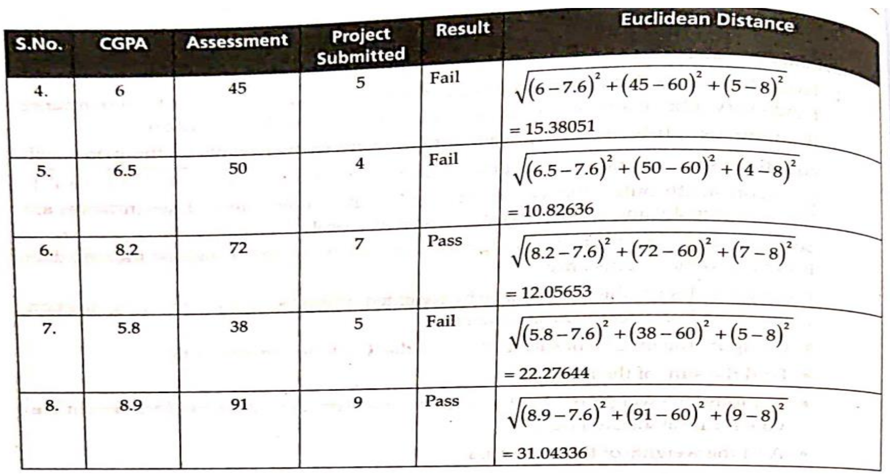
Step 1: Given a test instance (7.6, 60, 8) and a set of classes {Pass, Fail}, use the training dataset to classify the test instance using Euclidean distance and weighting function.

Assign *k* = 3. The distance calculation is shown in Table 4.5.

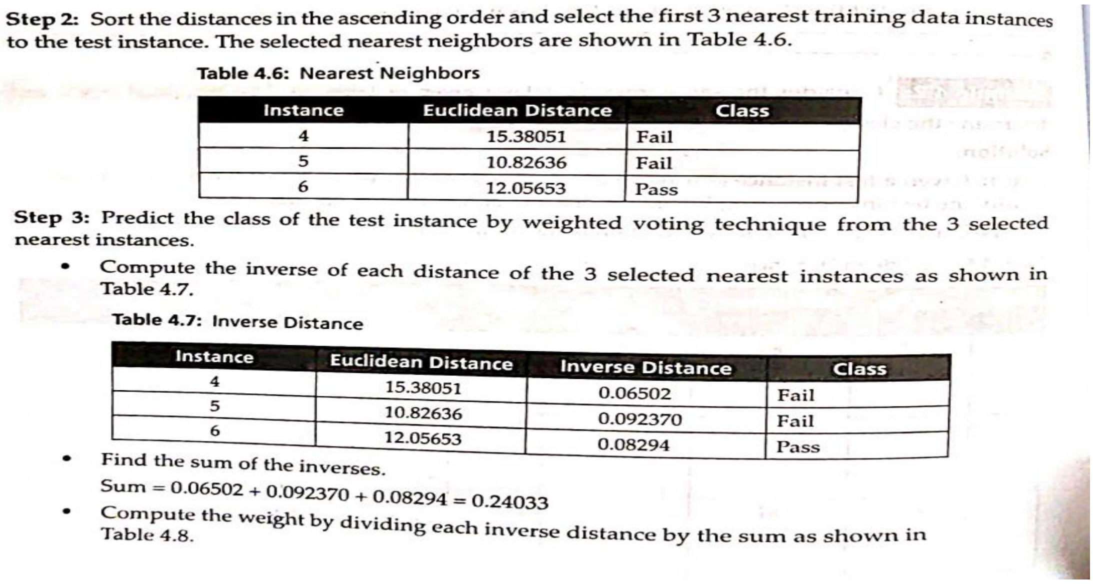
Table 4.5: Euclidean Distance



S.No.	CGPA	Assessment	Project Submitted	Result	Euclidean Distance
1.	9.2	85	8	Pass	$\sqrt{(9.2-7.6)^2+(85-60)^2+(8-8)^2}$ = 25.05115
2.	8	80	7	Pass	$\sqrt{(8-7.6)^2+(80-60)^2+(7-8)^2}$ = 20.02898
3.	8.5	81	8	Pass	$\sqrt{(8.5-7.6)^2+(81-60)^2+(8-8)^2}$ = 21.01928



S.No.	CGPA	Assessment	Project Submitted	Result	Euclidean Distance
4.	6	45	5	Fail	$\sqrt{(6-7.6)^2+(45-60)^2+(5-8)^2}$ = 15.38051
5.	6.5	50	4	Fail	$\sqrt{(6.5-7.6)^2+(50-60)^2+(4-8)^2}$ = 10.82636
6.	8.2	72	7	Pass	$\sqrt{(8.2-7.6)^2+(72-60)^2+(7-8)^2}$ = 12.05653
7.	5.8	38	5	Fail	$\sqrt{(5.8-7.6)^2+(38-60)^2+(5-8)^2}$ = 22.27644
8.	8.9	91	9	Pass	$\sqrt{(8.9-7.6)^2+(91-60)^2+(9-8)^2}$ = 31.04336



Step 2: Sort the distances in the ascending order and select the first 3 nearest training data instances to the test instance. The selected nearest neighbors are shown in Table 4.6.

Table 4.6: Nearest Neighbors

Instance	Euclidean Distance	Class
4	15.38051	Fail
5	10.82636	Fail
6	12.05653	Pass

Step 3: Predict the class of the test instance by weighted voting technique from the 3 selected nearest instances.

- Compute the inverse of each distance of the 3 selected nearest instances as shown in Table 4.7.

Table 4.7: Inverse Distance

Instance	Euclidean Distance	Inverse Distance	Class
4	15.38051	0.06502	Fail
5	10.82636	0.092370	Fail
6	12.05653	0.08294	Pass

- Find the sum of the inverses.
  Sum = 0.06502 + 0.092370 + 0.08294 = 0.24033
- Compute the weight by dividing each inverse distance by the sum as shown in Table 4.8.

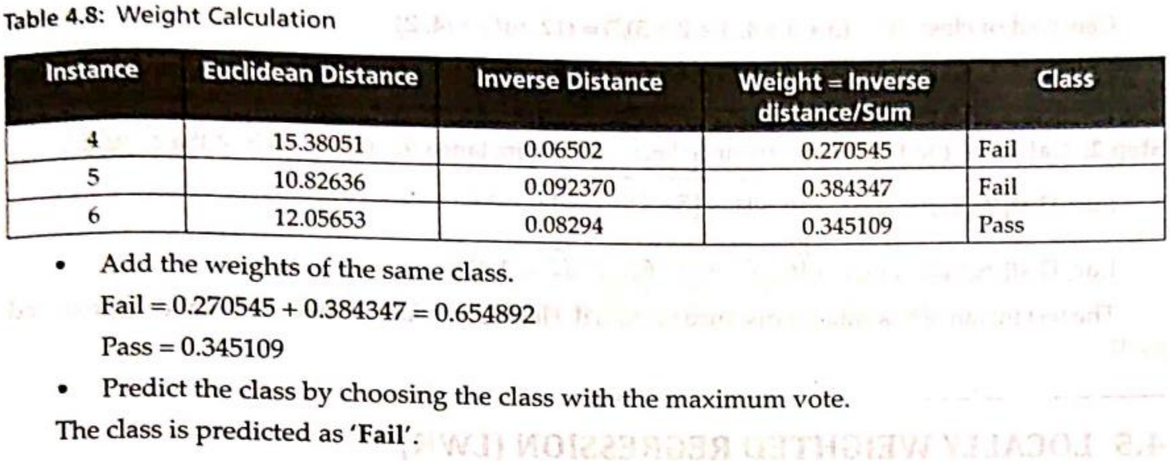


Table 4.8: Weight Calculation

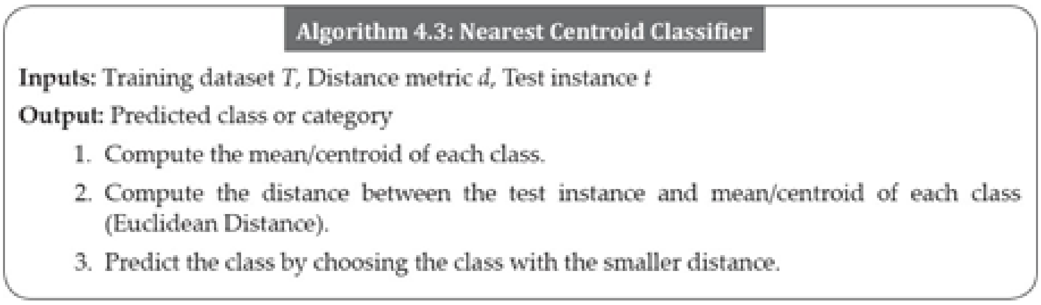
Instance	Euclidean Distance	Inverse Distance	Weight = Inverse distance/Sum	Class
4	15.38051	0.06502	0.270545	Fail
5	10.82636	0.092370	0.384347	Fail
6	12.05653	0.08294	0.345109	Pass

- Add the weights of the same class.
  Fail = 0.270545 + 0.384347 = 0.654892
  Pass = 0.345109
- Predict the class by choosing the class with the maximum vote.

The class is predicted as **'Fail'**.

4.4 Nearest Centroid Neighbor

The Nearest Centroids algorithm assumes that the centroids in the input feature space are different for each target label. The training data is split into groups by class label, then the centroid for each group of data is calculated. Each centroid is simply the mean value of each of the input variables, so it is also called as Mean Difference classifier. If there are two classes,then two centroids or points are calculated; three classes give three centroids, and so on.

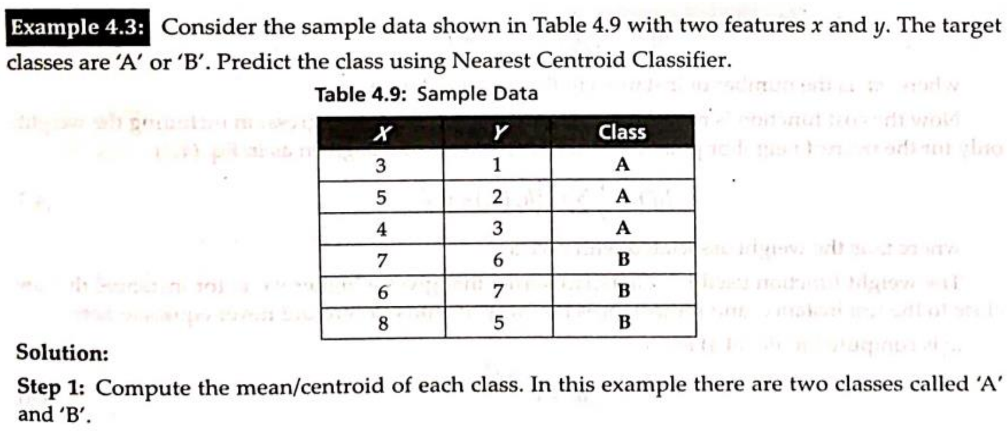


Algorithm 4.3: Nearest Centroid Classifier

Inputs: Training dataset *T*, Distance metric *d*, Test instance *t*

Output: Predicted class or category

1. Compute the mean/centroid of each class.
2. Compute the distance between the test instance and mean/centroid of each class (Euclidean Distance).
3. Predict the class by choosing the class with the smaller distance.



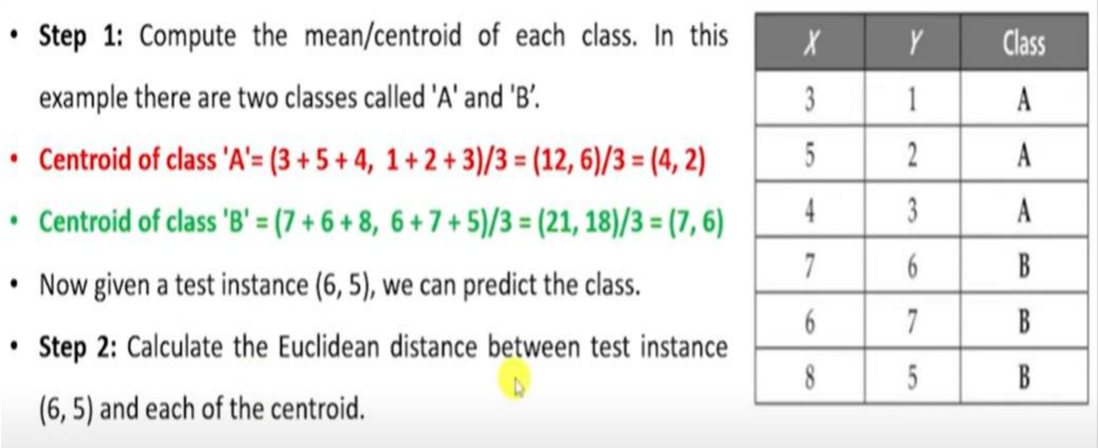
Example 4.3: Consider the sample data shown in Table 4.9 with two features x and y. The target classes are 'A' or 'B'. Predict the class using Nearest Centroid Classifier.

Table 4.9: Sample Data

X	Y	Class
3	1	A
5	2	A
4	3	A
7	6	B
6	7	B
8	5	B

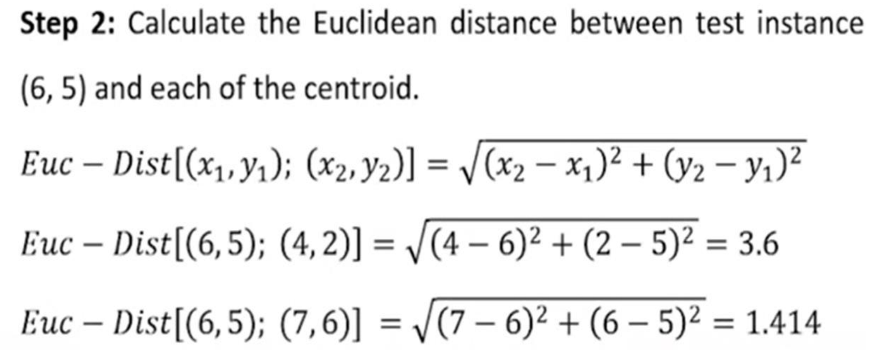
Solution:

Step 1: Compute the mean/centroid of each class. In this example there are two classes called 'A' and 'B'.



- **Step 1:** Compute the mean/centroid of each class. In this example there are two classes called 'A' and 'B'.
- Centroid of class 'A'= (3 + 5 + 4, 1 + 2 + 3)/3 = (12, 6)/3 = (4, 2)
- Centroid of class 'B' = (7 + 6 + 8, 6 + 7 + 5)/3 = (21, 18)/3 = (7, 6)
- Now given a test instance (6, 5), we can predict the class.
- **Step 2:** Calculate the Euclidean distance between test instance (6, 5) and each of the centroid.

X	Y	Class
3	1	A
5	2	A
4	3	A
7	6	B
6	7	B
8	5	B



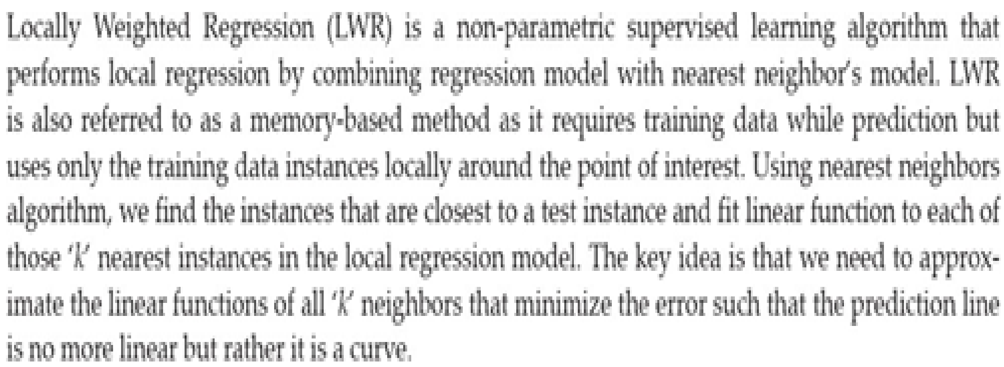
Step 2: Calculate the Euclidean distance between test instance (6, 5) and each of the centroid.

$$Euc - Dist[(x_1, y_1);\ (x_2, y_2)] = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

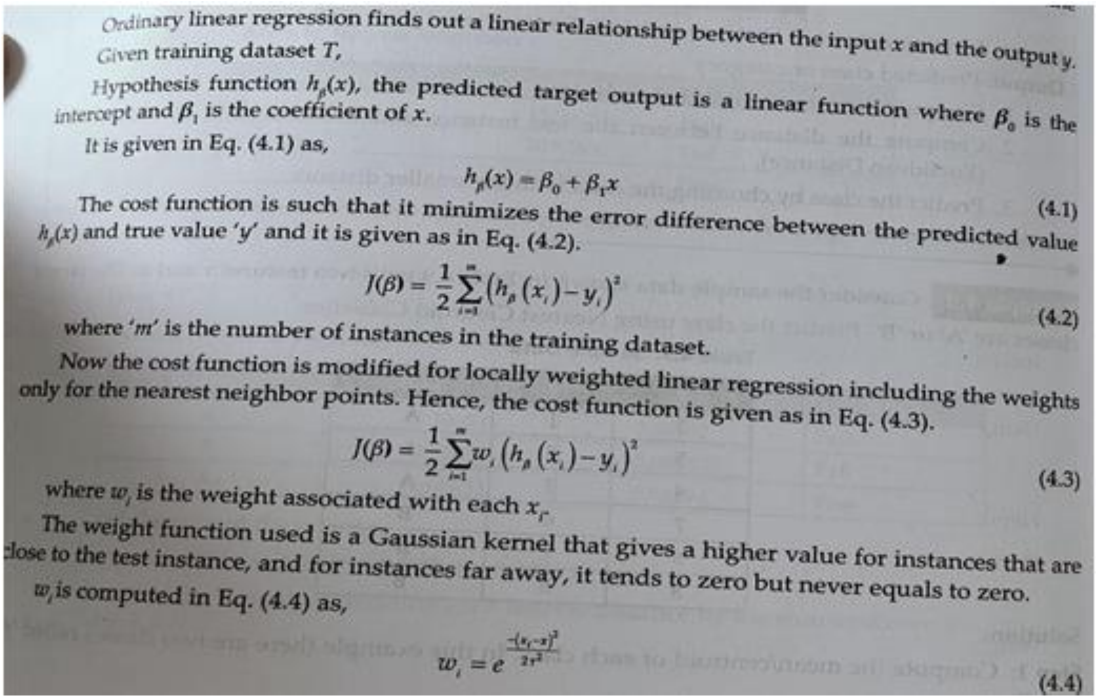
$$Euc - Dist[(6,5);\ (4,2)] = \sqrt{(4-6)^2 + (2-5)^2} = 3.6$$

$$Euc - Dist[(6,5);\ (7,6)] = \sqrt{(7-6)^2 + (6-5)^2} = 1.414$$

4.5 Locally Weighted Regression (LWR)



Locally Weighted Regression (LWR) is a non-parametric supervised learning algorithm that performs local regression by combining regression model with nearest neighbor's model. LWR is also referred to as a memory-based method as it requires training data while prediction but uses only the training data instances locally around the point of interest. Using nearest neighbors algorithm, we find the instances that are closest to a test instance and fit linear function to each of those 'K' nearest instances in the local regression model. The key idea is that we need to approximate the linear functions of all 'K' neighbors that minimize the error such that the prediction line is no more linear but rather it is a curve.



Ordinary linear regression finds out a linear relationship between the input x and the output y.

Given training dataset T,

Hypothesis function $h_\beta(x)$, the predicted target output is a linear function where β_0 is the intercept and β_1 is the coefficient of x.

It is given in Eq. (4.1) as,

$$h_\beta(x) = \beta_0 + \beta_1 x \tag{4.1}$$

The cost function is such that it minimizes the error difference between the predicted value $h_\beta(x)$ and true value 'y' and it is given as in Eq. (4.2).

$$J(\beta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\beta(x_i) - y_i\right)^2 \tag{4.2}$$

where 'm' is the number of instances in the training dataset.

Now the cost function is modified for locally weighted linear regression including the weights only for the nearest neighbor points. Hence, the cost function is given as in Eq. (4.3).

$$J(\beta) = \frac{1}{2}\sum_{i=1}^{m} w_i \left(h_\beta(x_i) - y_i\right)^2 \tag{4.3}$$

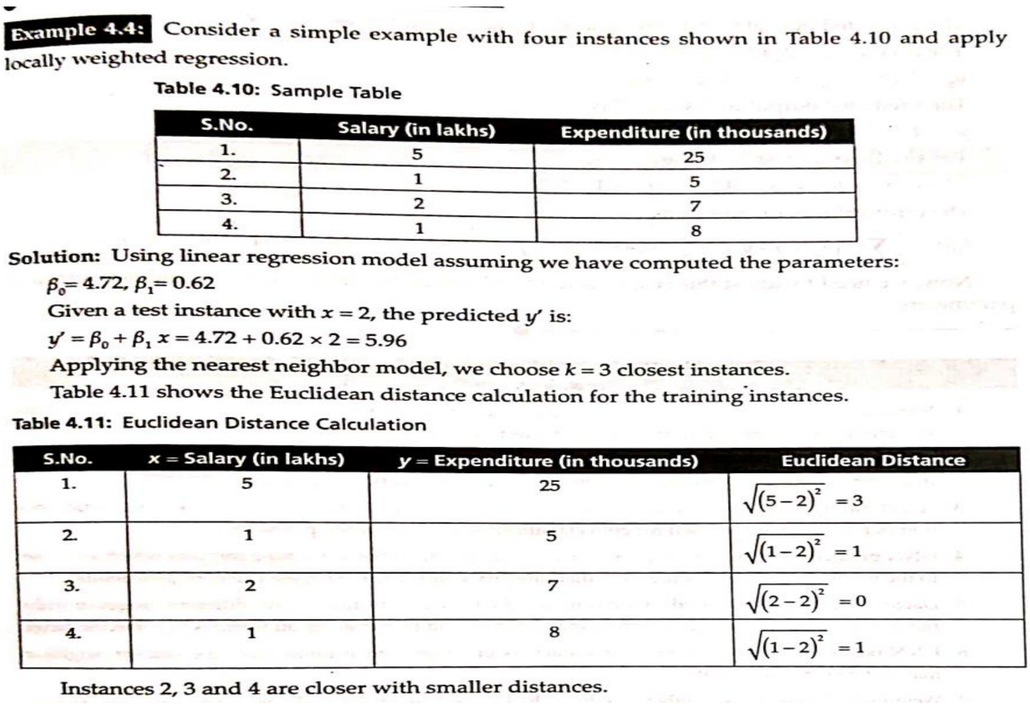
where w_i is the weight associated with each x_i.

The weight function used is a Gaussian kernel that gives a higher value for instances that are close to the test instance, and for instances far away, it tends to zero but never equals to zero.

w_i is computed in Eq. (4.4) as,

$$w_i = e^{\frac{-(x_i - x)^2}{2r^2}} \tag{4.4}$$

Where, r is called the bandwidth parameter and controls the rate at which wi reduces to zero with distance from xi.



Example 4.4: Consider a simple example with four instances shown in Table 4.10 and apply locally weighted regression.

Table 4.10: Sample Table

S.No.	Salary (in lakhs)	Expenditure (in thousands)
1.	5	25
2.	1	5
3.	2	7
4.	1	8

Solution: Using linear regression model assuming we have computed the parameters:

$\beta_0 = 4.72$, $\beta_1 = 0.62$

Given a test instance with $x = 2$, the predicted y' is:

$y' = \beta_0 + \beta_1 x = 4.72 + 0.62 \times 2 = 5.96$

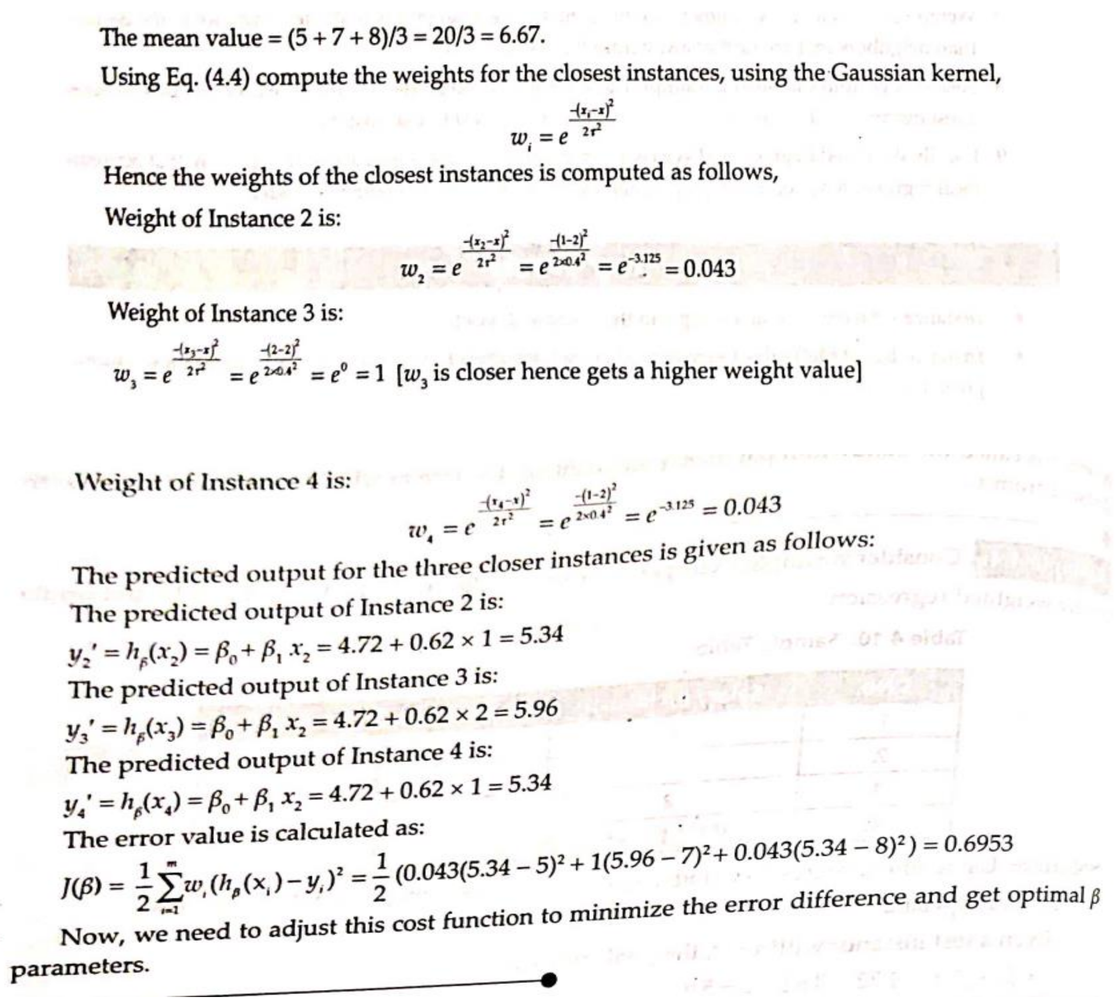
Applying the nearest neighbor model, we choose $k = 3$ closest instances.

Table 4.11 shows the Euclidean distance calculation for the training instances.

Table 4.11: Euclidean Distance Calculation

S.No.	x = Salary (in lakhs)	y = Expenditure (in thousands)	Euclidean Distance
1.	5	25	$\sqrt{(5-2)^2} = 3$
2.	1	5	$\sqrt{(1-2)^2} = 1$
3.	2	7	$\sqrt{(2-2)^2} = 0$
4.	1	8	$\sqrt{(1-2)^2} = 1$

Instances 2, 3 and 4 are closer with smaller distances.



The mean value = (5 + 7 + 8)/3 = 20/3 = 6.67.

Using Eq. (4.4) compute the weights for the closest instances, using the Gaussian kernel,

$$w_i = e^{\frac{-(x_i - x)^2}{2\tau^2}}$$

Hence the weights of the closest instances is computed as follows,

Weight of Instance 2 is:

$$w_2 = e^{\frac{-(x_2 - x)^2}{2\tau^2}} = e^{\frac{-(1-2)^2}{2\times 0.4^2}} = e^{-3.125} = 0.043$$

Weight of Instance 3 is:

$$w_3 = e^{\frac{-(x_3 - x)^2}{2\tau^2}} = e^{\frac{-(2-2)^2}{2\times 0.4^2}} = e^0 = 1$$ [w_3 is closer hence gets a higher weight value]

Weight of Instance 4 is:

$$w_4 = e^{\frac{-(x_4 - x)^2}{2\tau^2}} = e^{\frac{-(1-2)^2}{2\times 0.4^2}} = e^{-3.125} = 0.043$$

The predicted output for the three closer instances is given as follows:

The predicted output of Instance 2 is:

$y_2' = h_\beta(x_2) = \beta_0 + \beta_1 x_2 = 4.72 + 0.62 \times 1 = 5.34$

The predicted output of Instance 3 is:

$y_3' = h_\beta(x_3) = \beta_0 + \beta_1 x_2 = 4.72 + 0.62 \times 2 = 5.96$

The predicted output of Instance 4 is:

$y_4' = h_\beta(x_4) = \beta_0 + \beta_1 x_2 = 4.72 + 0.62 \times 1 = 5.34$

The error value is calculated as:

$$J(\beta) = \frac{1}{2}\sum_{i=1}^{m} w_i (h_\beta(x_i) - y_i)^2 = \frac{1}{2}(0.043(5.34 - 5)^2 + 1(5.96 - 7)^2 + 0.043(5.34 - 8)^2) = 0.6953$$

Now, we need to adjust this cost function to minimize the error difference and get optimal β parameters.

CHAPTER 5
REGRESSION ANALYSIS

5.1 Introduction to Regression

Regression analysis is a fundamental concept that consists of a set of machine learning methods that predict a continuous outcome variable (y) based on the value of one or multiple predictor variables (x). **OR**

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables.

Regression is a supervised learning technique which helps in finding the correlation between variables.

It is mainly used for prediction, forecasting, time series modelling, and determining the causal- effect relationship between variables.

Regression shows a line or curve that passes through all the datapoints on target-predictor graph in such a way that the vertical distance between the datapoints and the regression line is minimum." The distance between datapoints and line tells whether a model has captured a strong relationship or not.

- Function of regression analysis is given by: Y=f(x)

  Here, y is called dependent variable and x is called independent variable.

Applications of Regression Analysis

1) Sales of a goods or services
2) Value of bonds in portfolio management
3) Premium on insurance companies
4) Yield of crop in agriculture
5) Prices of real estate

5.2 INTRODUCTION TO LINEARITY, CORRELATION AND CAUSATION

A correlation is the statistical summary of the relationship between two sets of variables. It is a core part of data exploratory analysis, and is a critical aspect of numerous advanced machine learning techniques. Correlation between two variables can be found using a scatter plot.

There are different types of correlation:

- **Positive Correlation**: Two variables are said to be positively correlated when their values move in the same direction. For example, in the image below, as the value for X increases, sodoes the value for Y at a constant rate.
- **Negative Correlation**: Finally, variables X and Y will be negatively correlated when their values change in opposite directions, so here as the value for X increases, the value for Y decreases at a constant rate.
- **Neutral Correlation**: No relationship in the change of variables X and Y. In this case, the values are completely random and do not show any sign of correlation, as shown in the following image:

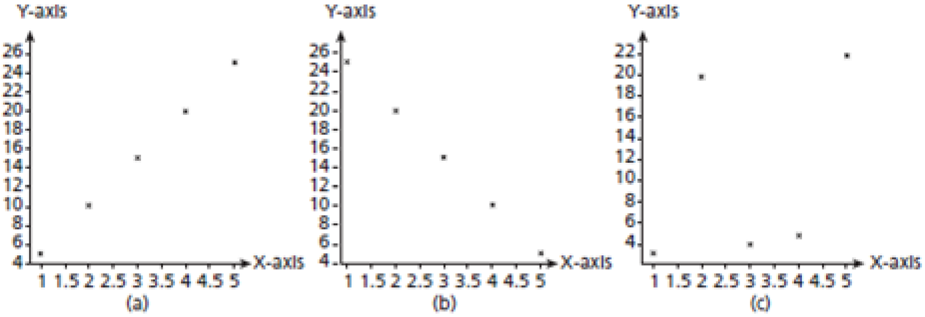


Figure 5.1: Examples of (a) Positive Correlation (b) Negative Correlation (c) Random Points with No Correlation

Causation

Causation is about relationship between two variables as x causes y. This is called x implies b.Regression is different from causation. Causation indicates that one event is the result of the occurrence of the other event; i.e. there is a causal relationship between the two events.

Linear and Non-Linear Relationships

- The relationship between input features (variables) and the output (target) variable is fundamental. These concepts have significant implications for the choice of algorithms, model complexity, and predictive performance.
- Linear relationship creates a straight line when plotted on a graph, a Non-Linear relationship does not create a straight line but instead creates a curve.
- Example:

  Linear-the relationship between the hours spent studying and the grades obtained in a class.

  Non-Linear- GPS Signal

- **Linearity:**

  Linear Relationship: A linear relationship between variables means that a change in one variable is associated with a proportional change in another variable. Mathematically, it can be represented as y = a * x + b, where y is the output, x is the input, and a and b are constants.

  Linear Models: Goal is to find the best-fitting line (plane in higher dimensions) to the data points. Linear models are interpretable and work well when the relationship between variablesis close to being linear.

  Limitations: Linear models may perform poorly when the relationship between variables is non-linear. In such cases, they may underfit the data, meaning they are too simple to capture the underlying patterns.

- **Non-Linearity:**

  Non-Linear Relationship: A non-linear relationship implies that the change in one variable isnot proportional to the change in another variable. Non-linear relationships can take various forms, such as quadratic, exponential, logarithmic, or arbitrary shapes.

  Non-Linear Models: Machine learning models like decision trees, random forests, support vector machines with non-linear kernels, and neural networks can capture non-linear relationships. These models are more flexible and can fit complex data patterns.

  Benefits: Non-linear models can perform well when the underlying relationships in the data are complex or when interactions between variables are non-linear. They have the capacity to capture intricate patterns.

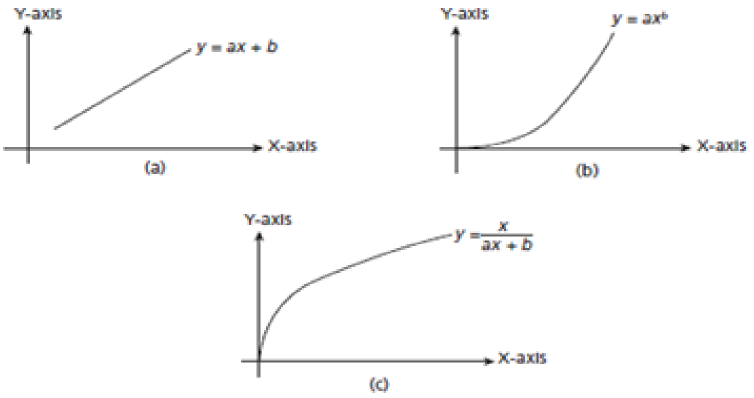
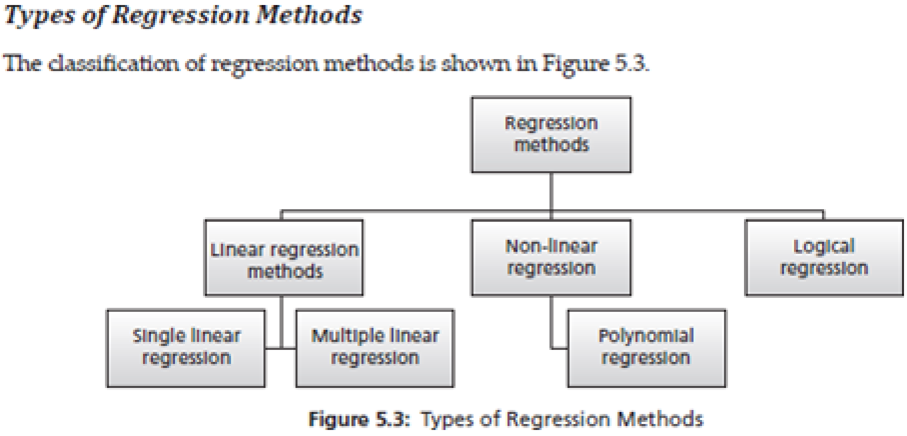


Figure 5.2: (a) Example of Linear Relationship of the Form $y = ax + b$ (b) Example of a Non-linear Relationship of the Form $y = ax^b$ (c) Examples of a Non-linear Relationship $y = \frac{x}{ax + b}$

Types of Regression



Types of Regression Methods

The classification of regression methods is shown in Figure 5.3.

Figure 5.3: Types of Regression Methods

Linear Regression:

Single Independent Variable: Linear regression, also known as simple linear regression, is used when there is a single independent variable (predictor) and one dependent variable (target).

Purpose: Linear regression is used to establish a linear relationship between two variables and make predictions based on this relationship. It's suitable for simple scenarios where there's onlyone predictor.

Multiple Regression:

Multiple Independent Variables: Multiple regression, as the name suggests, is used when thereare two or more independent variables (predictors) and one dependent variable (target).

Purpose: Multiple regression allows you to model the relationship between the dependent variable and multiple predictors simultaneously. It is used when there are multiple factors that may influence the target variable, and you want to understand their combined effect and makepredictions based on all these factors.

Polynomial Regression:

Polynomial regression is an extension of multiple regression used when the relationship between the independent and dependent variables is non-linear.

Logistic Regression:

Logistic regression is used when the dependent variable is binary (0 or 1). It models the probability of the dependent variable belonging to a particular class.

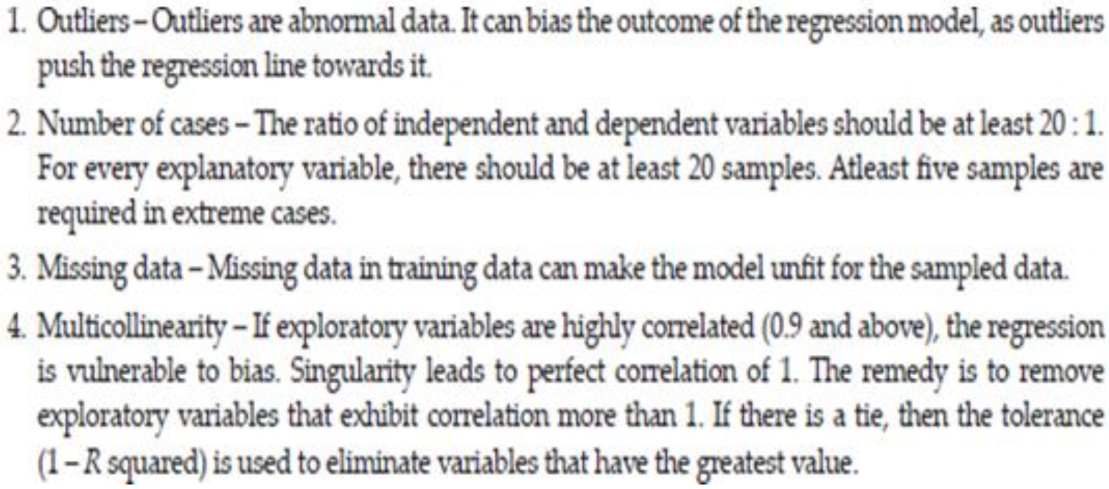
Lasso Regression (L1 Regularization):

Lasso regression is used for feature selection and regularization. It penalizes the absolutevalues of the coefficients, which encourages sparsity in the model.

Ridge Regression (L2 Regularization):

Ridge regression is used for regularization to prevent overfitting in multiple regression. Itpenalizes the square of the coefficients.

Limitations of Regression

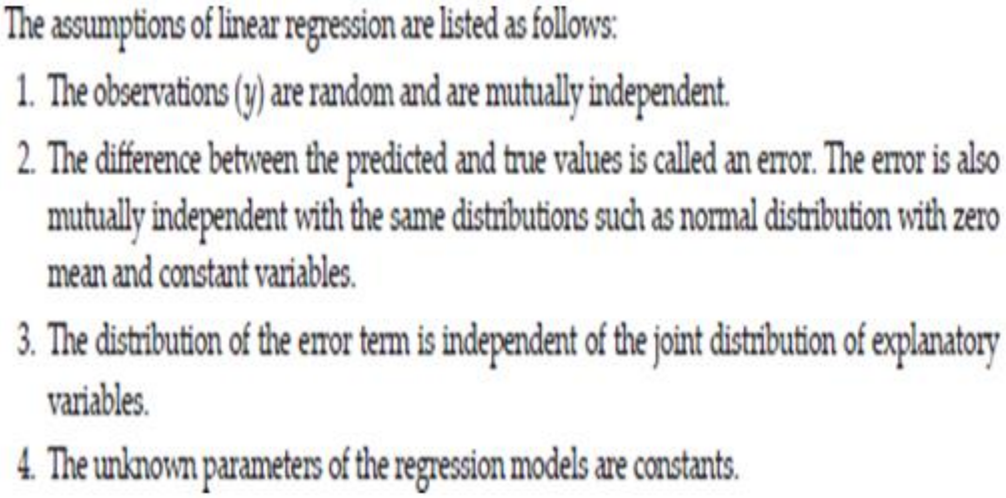


1. Outliers – Outliers are abnormal data. It can bias the outcome of the regression model, as outliers push the regression line towards it.
2. Number of cases – The ratio of independent and dependent variables should be at least 20 : 1. For every explanatory variable, there should be at least 20 samples. Atleast five samples are required in extreme cases.
3. Missing data – Missing data in training data can make the model unfit for the sampled data.
4. Multicollinearity – If exploratory variables are highly correlated (0.9 and above), the regression is vulnerable to bias. Singularity leads to perfect correlation of 1. The remedy is to remove exploratory variables that exhibit correlation more than 1. If there is a tie, then the tolerance (1 – *R* squared) is used to eliminate variables that have the greatest value.

5.3 INTRODUCTION TO LINEAR REGRESSION

Linear regression model can be created by fitting a line among the scattered data points. The line is of the form:

y= a0+a1*x+e



The assumptions of linear regression are listed as follows:

1. The observations (*y*) are random and are mutually independent.
2. The difference between the predicted and true values is called an error. The error is also mutually independent with the same distributions such as normal distribution with zero mean and constant variables.
3. The distribution of the error term is independent of the joint distribution of explanatory variables.
4. The unknown parameters of the regression models are constants.

Ordinary Least Square Approach

The ordinary least squares (OLS) algorithm is a method for estimating the parameters of a linear regression model. Aim: To find the values of the linear regression model's parameters (i.e., the coefficients) that minimize the sum of the squared residuals.

In mathematical terms, this can be written as: **Minimize** ∑(**yi** – **ŷ**i)^**2**

where **y**i is the actual value, **ŷ**i is the predicted value.

A linear regression model used for determining the value of the response variable, **ŷ,** can berepresented as the following equation.

y = b0 + b1x1 + b2x2 + … + bnxn + e

where: y - is the dependent variable, b0 is the intercept, e isthe error term

b1, b2, …, bn are the coefficients of the independentvariables x1, x2, …, xn

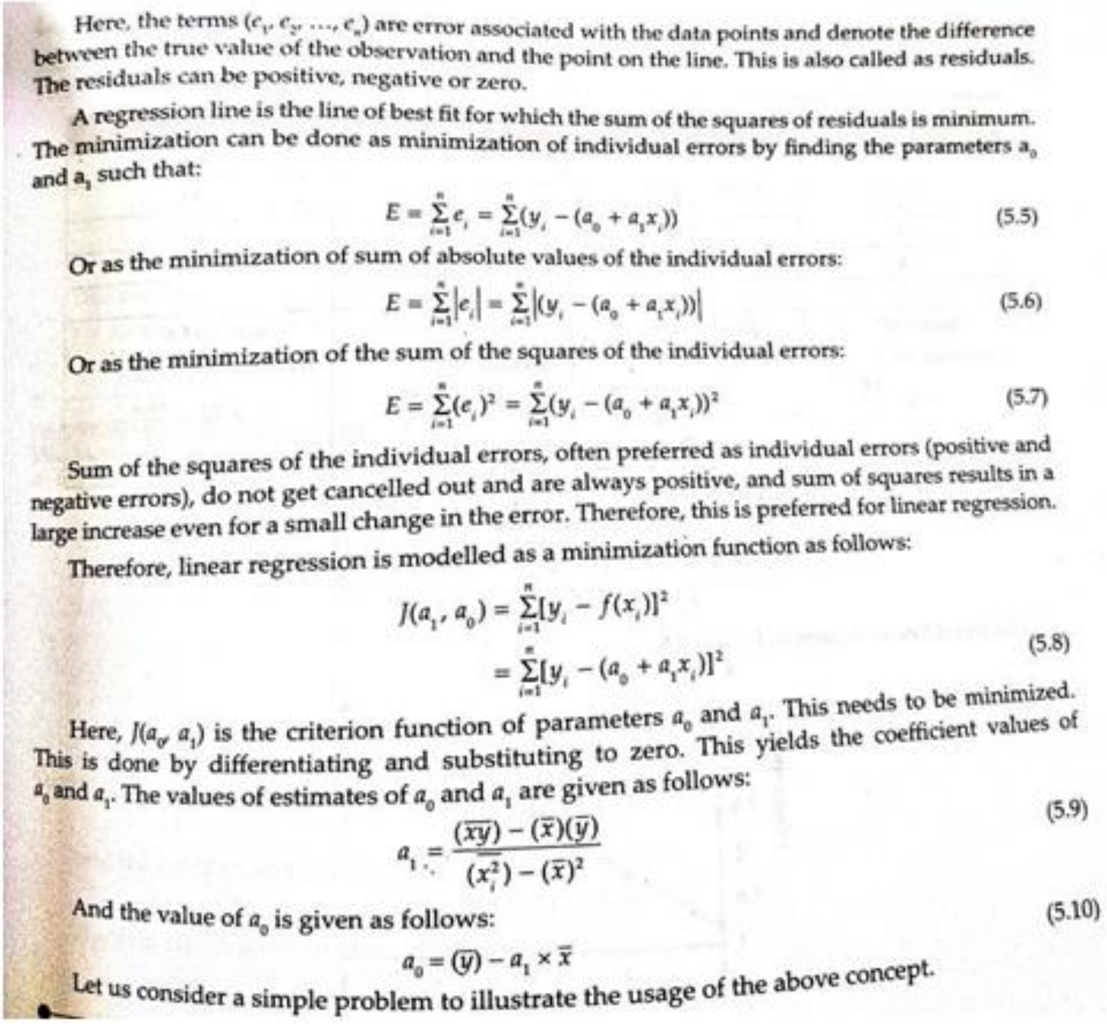
The coefficients b1, b2, …, bn can also be called the **coefficientsof determination**. The goal of the OLS method can be used to estimate the unknown parameters (**b1, b2, …, bn**) by minimizingthe sum of squared residuals (RSS). The sum of squared residualsis also termed the sum of squared error (SSE).

This method is also known as the **least-squares method** for regression or linear regression.Mathematically the line of equations for points are:

y1=(a0+a1x1)+e1

y2=(a0+a1x2)+e2 ……. yn=(a0+a1xn)+en.

In general ei=yi - (a0+a1x1)



Here, the terms ($e_1, e_2, \ldots, e_n$) are error associated with the data points and denote the difference between the true value of the observation and the point on the line. This is also called as residuals. The residuals can be positive, negative or zero.

A regression line is the line of best fit for which the sum of the squares of residuals is minimum. The minimization can be done as minimization of individual errors by finding the parameters a_0 and a_1 such that:

$$E = \sum_{i=1}^{n} e_i = \sum_{i=1}^{n}(y_i - (a_0 + a_1 x_i)) \quad (5.5)$$

Or as the minimization of sum of absolute values of the individual errors:

$$E = \sum_{i=1}^{n} |e_i| = \sum_{i=1}^{n} |(y_i - (a_0 + a_1 x_i))| \quad (5.6)$$

Or as the minimization of the sum of the squares of the individual errors:

$$E = \sum_{i=1}^{n} (e_i)^2 = \sum_{i=1}^{n} (y_i - (a_0 + a_1 x_i))^2 \quad (5.7)$$

Sum of the squares of the individual errors, often preferred as individual errors (positive and negative errors), do not get cancelled out and are always positive, and sum of squares results in a large increase even for a small change in the error. Therefore, this is preferred for linear regression.

Therefore, linear regression is modelled as a minimization function as follows:

$$\begin{aligned} J(a_1, a_0) &= \sum_{i=1}^{n} [y_i - f(x_i)]^2 \\ &= \sum_{i=1}^{n} [y_i - (a_0 + a_1 x_i)]^2 \end{aligned} \quad (5.8)$$

Here, $J(a_0, a_1)$ is the criterion function of parameters a_0 and a_1. This needs to be minimized. This is done by differentiating and substituting to zero. This yields the coefficient values of a_0 and a_1. The values of estimates of a_0 and a_1 are given as follows:

$$a_1 = \frac{(\overline{xy}) - (\overline{x})(\overline{y})}{(\overline{x_i^2}) - (\overline{x})^2} \quad (5.9)$$

And the value of a_0 is given as follows: (5.10)

$$a_0 = (\overline{y}) - a_1 \times \overline{x}$$

Let us consider a simple problem to illustrate the usage of the above concept.

Linear Regression Example

Example 5.1: Let us consider an example where the five weeks' sales data (in Thousands) is given as shown below in Table 5.1. Apply linear regression technique to predict the 7th and 9th month sales.

Table 5.1: Sample Data

x_i (Week)	y_i (Sales in Thousands)
1	1.2
2	1.8
3	2.6
4	3.2
5	3.8

Table 5.2: Computation Table

x_i	y_i	$(x_i)^2$	$x_i \times y_i$
1	1.2	1	1.2
2	1.8	4	3.6
3	2.6	9	7.8
4	3.2	16	12.8
5	3.8	25	19
Sum = 15 Average of (x_i) $= \bar{x} = \frac{15}{5}$ $= 3$	Sum = 12.6 Average of (y_i) $= \bar{y} = \frac{12.6}{5}$ $= 2.52$	Sum = 55 Average of (x_i^2) $= \overline{x_i^2} = \frac{55}{5}$ $= 11$	Sum = 44.4 Average of $(x_i \times y_i)$ $= \overline{xy} = \frac{44.4}{5}$ $= 8.88$

Let us compute the slope and intercept now using Eq. (5.9) as:

$$a_1 = \frac{8.88 - 3(2.52)}{11 - 3^2} = 0.66$$

$$a_0 = 2.52 - 0.66 \times 3 = 0.54$$

The fitted line is shown in Figure 5.5.

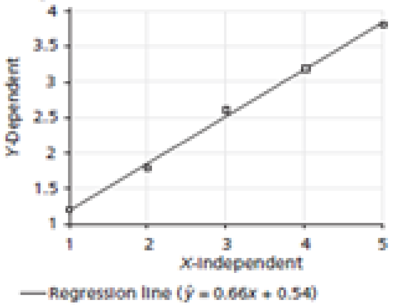
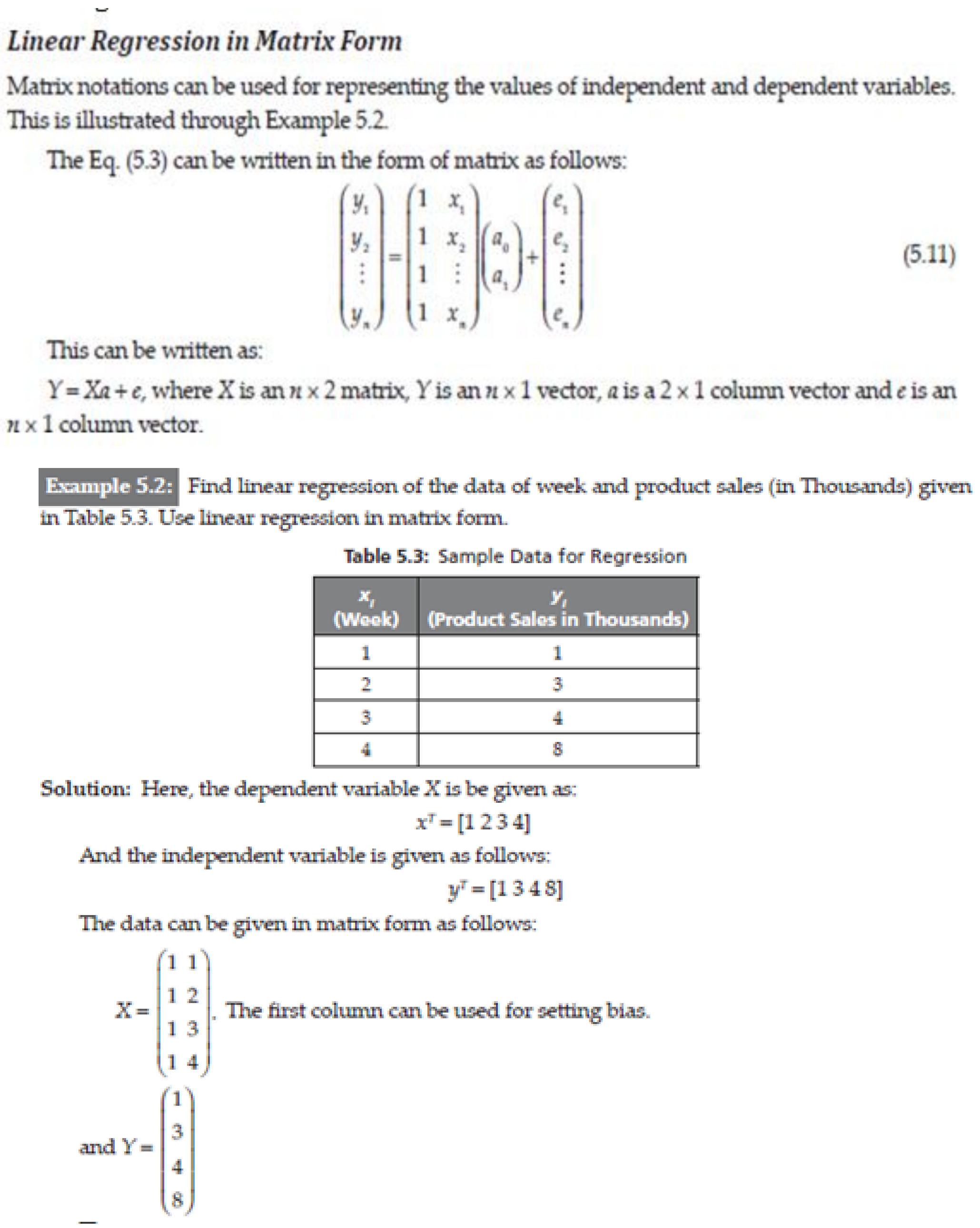


Figure 5.5: Linear Regression Model Constructed

Let us model the relationship as $y = a_0 + a_1 \times x$. Therefore, the fitted line for the above data is: $y = 0.54 + 0.66 \times x$.

The predicted 7th week sale would be (when $x = 7$), $y = 0.54 + 0.66 \times 7 = 5.16$ and the 12th month, $y = 0.54 + 0.66 \times 12 = 8.46$. All sales are in thousands.

Linear Regression in Matrix Form



Linear Regression in Matrix Form

Matrix notations can be used for representing the values of independent and dependent variables. This is illustrated through Example 5.2.

The Eq. (5.3) can be written in the form of matrix as follows:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & \vdots \\ 1 & x_n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix} \tag{5.11}$$

This can be written as:

$Y = Xa + e$, where X is an $n \times 2$ matrix, Y is an $n \times 1$ vector, a is a 2×1 column vector and e is an $n \times 1$ column vector.

Example 5.2: Find linear regression of the data of week and product sales (in Thousands) given in Table 5.3. Use linear regression in matrix form.

Table 5.3: Sample Data for Regression

x_i (Week)	y_i (Product Sales in Thousands)
1	1
2	3
3	4
4	8

Solution: Here, the dependent variable X is be given as:

$$x^T = [1\ 2\ 3\ 4]$$

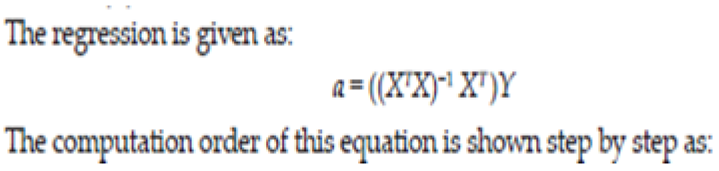
And the independent variable is given as follows:

$$y^T = [1\ 3\ 4\ 8]$$

The data can be given in matrix form as follows:

$X = \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix}$. The first column can be used for setting bias.

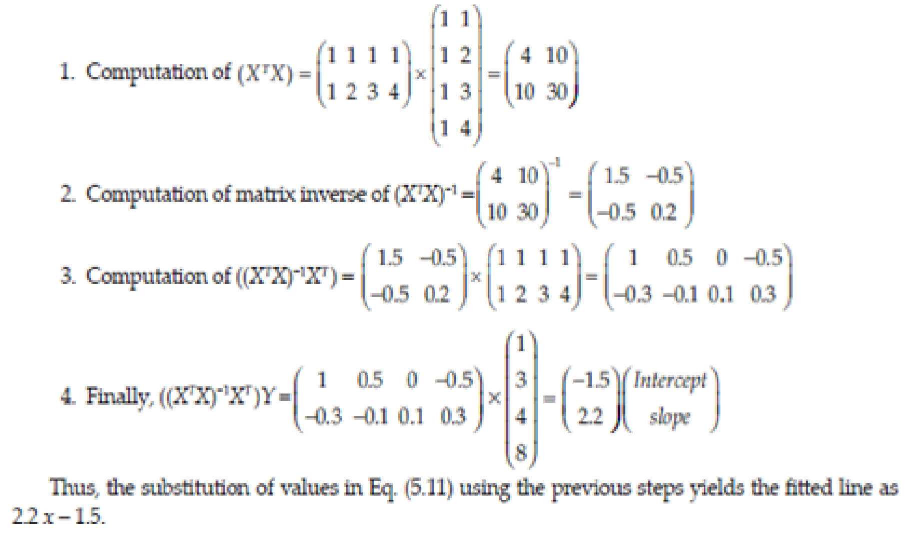
and $Y = \begin{pmatrix} 1 \\ 3 \\ 4 \\ 8 \end{pmatrix}$



The regression is given as:

$$a = ((X^T X)^{-1} X^T)Y$$

The computation order of this equation is shown step by step as:



1. Computation of $(X^TX) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{pmatrix} \times \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix} = \begin{pmatrix} 4 & 10 \\ 10 & 30 \end{pmatrix}$

2. Computation of matrix inverse of $(X^TX)^{-1} = \begin{pmatrix} 4 & 10 \\ 10 & 30 \end{pmatrix}^{-1} = \begin{pmatrix} 1.5 & -0.5 \\ -0.5 & 0.2 \end{pmatrix}$

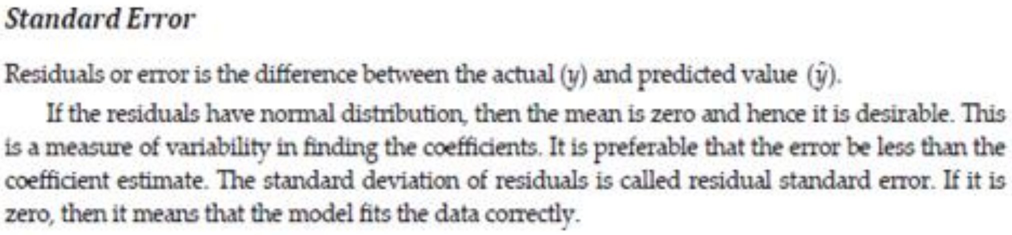
3. Computation of $((X^TX)^{-1}X^T) = \begin{pmatrix} 1.5 & -0.5 \\ -0.5 & 0.2 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0.5 & 0 & -0.5 \\ -0.3 & -0.1 & 0.1 & 0.3 \end{pmatrix}$

4. Finally, $((X^TX)^{-1}X^T)Y = \begin{pmatrix} 1 & 0.5 & 0 & -0.5 \\ -0.3 & -0.1 & 0.1 & 0.3 \end{pmatrix} \times \begin{pmatrix} 1 \\ 3 \\ 4 \\ 8 \end{pmatrix} = \begin{pmatrix} -1.5 \\ 2.2 \end{pmatrix} \begin{pmatrix} Intercept \\ slope \end{pmatrix}$

Thus, the substitution of values in Eq. (5.11) using the previous steps yields the fitted line as $2.2x - 1.5$.

5.4 VALIDATION OF REGRESSION METHODS

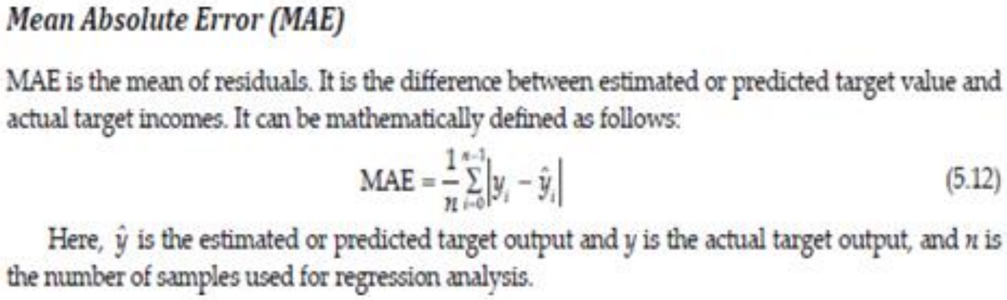
The regression should be evaluated using some metrics for checking the correctness. Thefollowing metrics are used to validate the results of regression.



Standard Error

Residuals or error is the difference between the actual (y) and predicted value $(\hat{y})$.

If the residuals have normal distribution, then the mean is zero and hence it is desirable. This is a measure of variability in finding the coefficients. It is preferable that the error be less than the coefficient estimate. The standard deviation of residuals is called residual standard error. If it is zero, then it means that the model fits the data correctly.

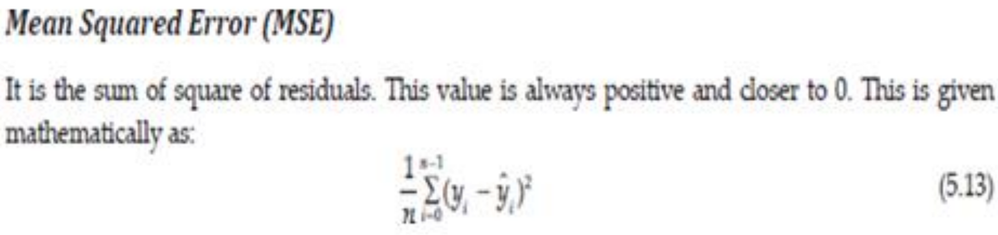


Mean Absolute Error (MAE)

MAE is the mean of residuals. It is the difference between estimated or predicted target value and actual target incomes. It can be mathematically defined as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=0}^{n-1}\left|y_i - \hat{y}_i\right| \tag{5.12}$$

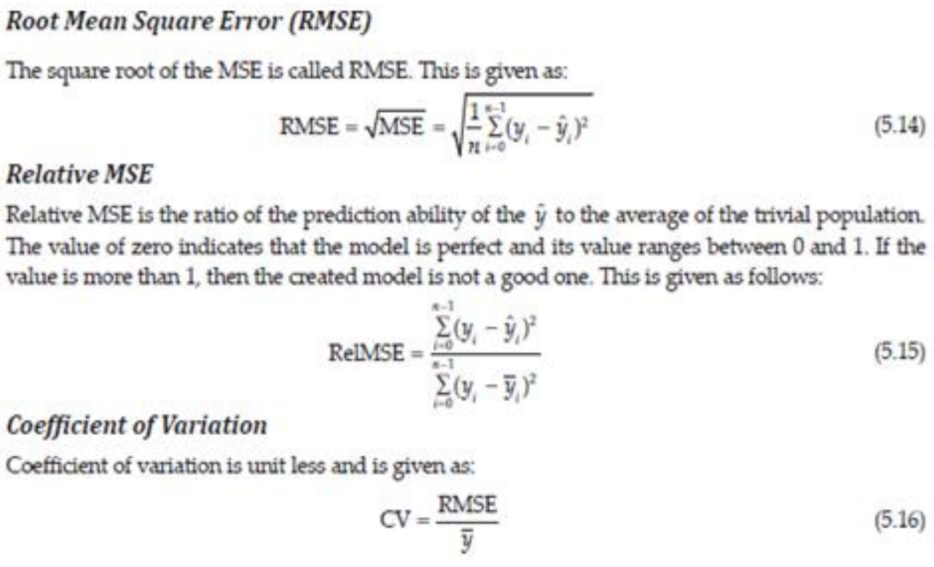
Here, $\hat{y}$ is the estimated or predicted target output and y is the actual target output, and n is the number of samples used for regression analysis.



Mean Squared Error (MSE)

It is the sum of square of residuals. This value is always positive and closer to 0. This is given mathematically as:

$$\frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2 \tag{5.13}$$



Root Mean Square Error (RMSE)

The square root of the MSE is called RMSE. This is given as:

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2} \tag{5.14}$$

Relative MSE

Relative MSE is the ratio of the prediction ability of the $\hat{y}$ to the average of the trivial population. The value of zero indicates that the model is perfect and its value ranges between 0 and 1. If the value is more than 1, then the created model is not a good one. This is given as follows:

$$\text{RelMSE} = \frac{\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y}_i)^2} \tag{5.15}$$

Coefficient of Variation

Coefficient of variation is unit less and is given as:

$$\text{CV} = \frac{\text{RMSE}}{\bar{y}} \tag{5.16}$$

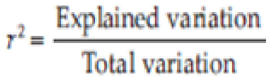
Coefficient of Determination

The coefficient of determination (R^2 or r-squared) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable.

The sum of the squares of the differences between the y-value of the data pair and the average of y is called total variation. Thus, the following variation can be defined as,

The explained variation is given by, **=∑(Ŷi – mean(Yi))2** The unexplained variation is given by, **=∑(Yi - Ŷi)2**

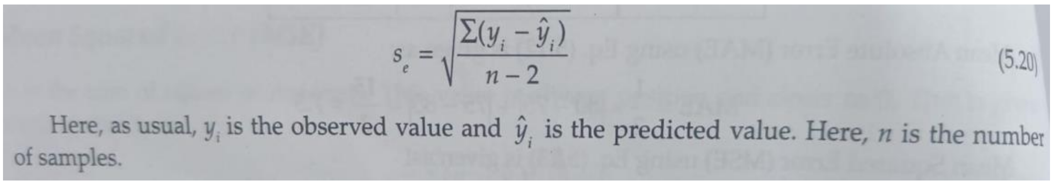
Thus, the total variation is equal to the explained variation and the unexplained variation. The coefficient of determination r2 is the ratio of the explained and unexplained variations.



$$r^2 = \frac{\text{Explained variation}}{\text{Total variation}}$$

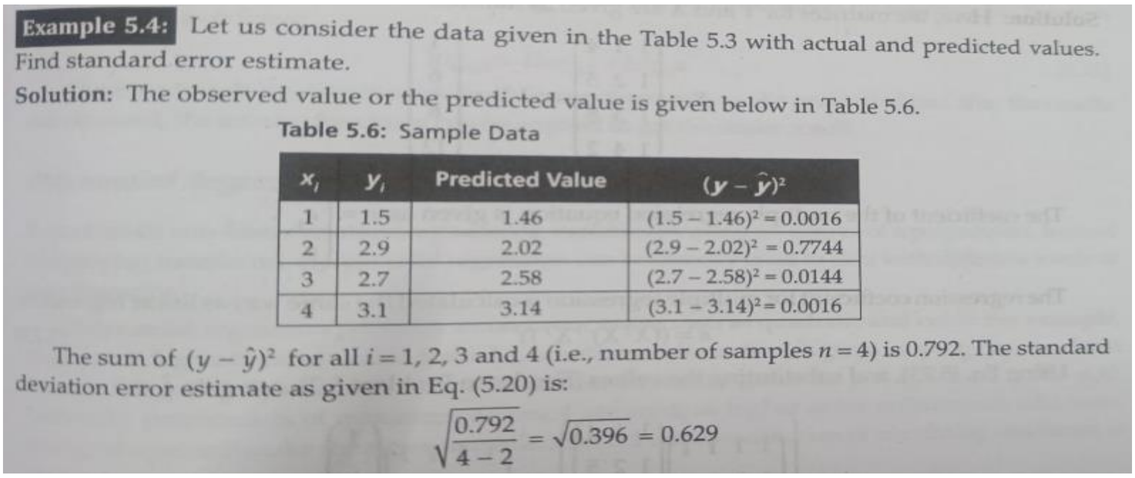
Standard Error Estimate:

Standard Error Estimate is another useful measure of regression. It is standard deviation of the observed values to the predicted values. This is given as:



$$s_e = \sqrt{\frac{\Sigma(y_i - \hat{y}_i)}{n-2}} \tag{5.20}$$

Here, as usual, y_i is the observed value and $\hat{y}_i$ is the predicted value. Here, n is the number of samples.



Example 5.4: Let us consider the data given in the Table 5.3 with actual and predicted values. Find standard error estimate.

Solution: The observed value or the predicted value is given below in Table 5.6.

Table 5.6: Sample Data

x_i	y_i	Predicted Value	$(y-\hat{y})^2$
1	1.5	1.46	$(1.5-1.46)^2 = 0.0016$
2	2.9	2.02	$(2.9-2.02)^2 = 0.7744$
3	2.7	2.58	$(2.7-2.58)^2 = 0.0144$
4	3.1	3.14	$(3.1-3.14)^2 = 0.0016$

The sum of $(y-\hat{y})^2$ for all $i = 1, 2, 3$ and 4 (i.e., number of samples $n = 4$) is 0.792. The standard deviation error estimate as given in Eq. (5.20) is:

$$\sqrt{\frac{0.792}{4-2}} = \sqrt{0.396} = 0.629$$