

FUNDAMENTALS OF LEARNING

Dr. Debanjan Pathak

Least Square Method

What is **Least Square Method**

- The least-squares method is a statistical method that is practised to find a regression line or a best-fit line for the given pattern.
- The method of least squares is used in regression.
- In regression analysis, this method is said to be a standard approach for the approximation of sets of equations having more equations than the number of unknowns (overdetermined systems).
- It is used to approximate the solution by minimizing the sum of the squares of the residuals made in the results of each individual equation.
 - Residual: the difference between an observed value and the fitted value provided by a model
- The method of least squares assumes that the best fit curve of a given type is the curve that has the minimal sum of deviations, i.e., least square error from a given set of data.

Curve fitting and **Least Square Method**

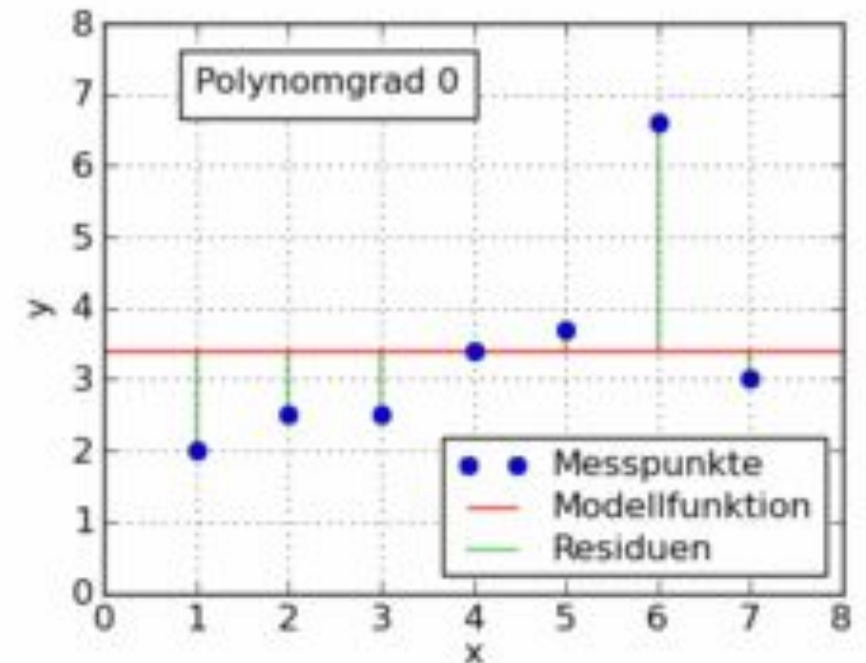
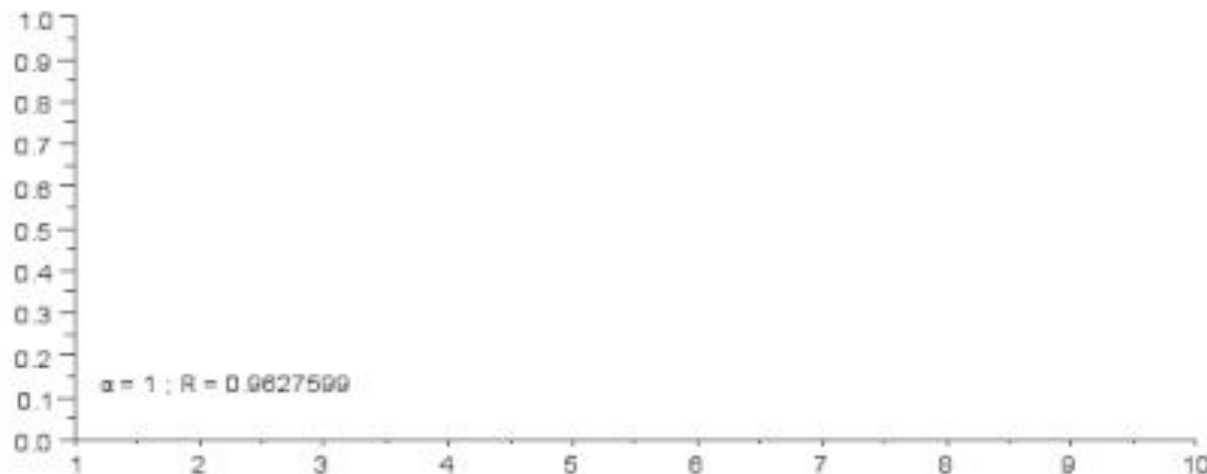
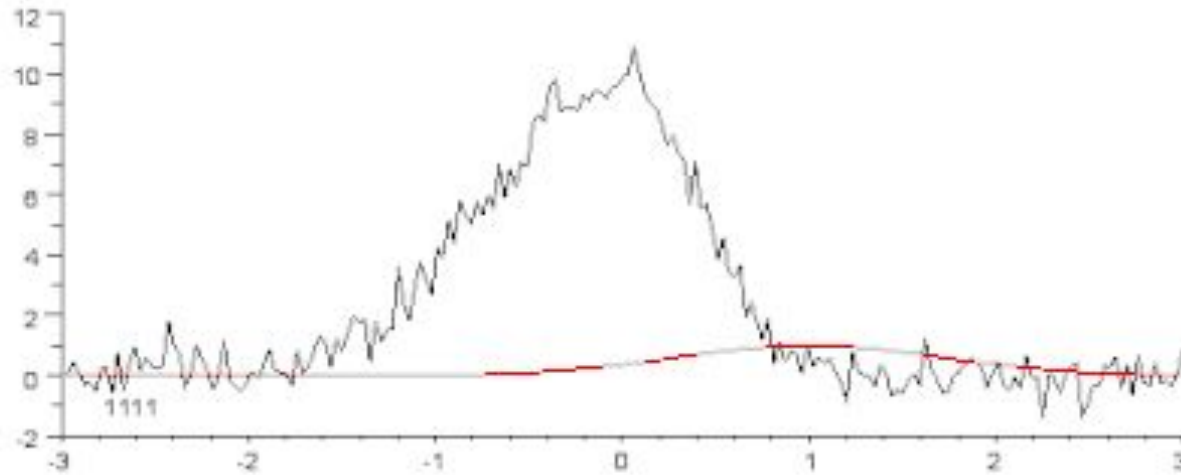
- Curve fitting is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints.
- Curve fitting can involve either interpolation, where an exact fit to the data is required, or smoothing, in which a "smooth" function is constructed that **approximately** fits the data.
- The least squares method is one way to compare the deviations.

Curve fitting and **Least Square Method**

Fitting of a noisy curve by an asymmetrical peak model, with an iterative process (Gauss–Newton algorithm with variable damping factor α).

Top: raw data and model.

Bottom: evolution of the normalised sum of the squares of the errors.



Best fitting and **Least Square Method**

- Suppose that the data points are $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ where x is the independent variable and y is the dependent variable. The fitting curve $f(x)$ has the deviation (error) e_i from each data point, as follows:

$$e_1 = y_1 - f(x_1),$$

$$e_2 = y_2 - f(x_2),$$

$$\vdots$$

$$e_n = y_n - f(x_n)$$

- According to the method of least squares, the best fitting curve has the property that $\sum_1^n e_i^2 = \sum_1^n [y_i - f(x_i)]^2$ is minimum.

Least square line $y=a+bx$ for the given data

- The method of least squares can be applied to determine the estimates of 'a' and 'b' in the simple linear regression equation using the given data by minimizing

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

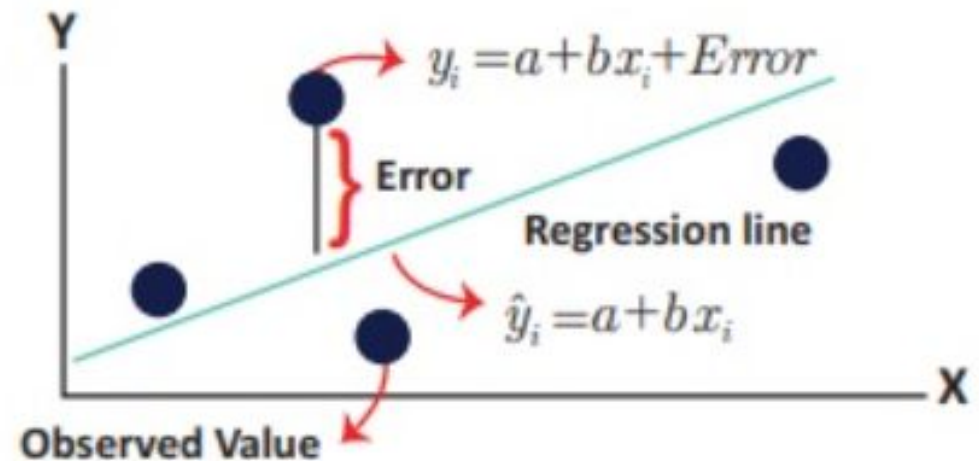
$$\text{i.e., } E(a,b) = \sum_{i=1}^n (y_i - a - bx_i)^2 .$$

- Differentiation of $E(a,b)$ with respect to 'a' and 'b' and equating them to zero constitute a set of two equations as described below:

$$\frac{\partial E(a,b)}{\partial a} = -2 \sum_{i=1}^n (y_i - a - bx_i) = 0$$

$$\frac{\partial E(a,b)}{\partial b} = -2 \sum_{i=1}^n x_i (y_i - a - bx_i) = 0$$

Simple Linear Regression Model



Least square line $y=a+bx$ for the given data

$$\frac{\partial E(a,b)}{\partial a} = -2 \sum_{i=1}^n (y_i - a - bx_i) = 0 \Rightarrow na + b \sum_{i=1}^n x_i = \sum_{i=1}^n y_i \Rightarrow na + b \sum x = \sum y$$

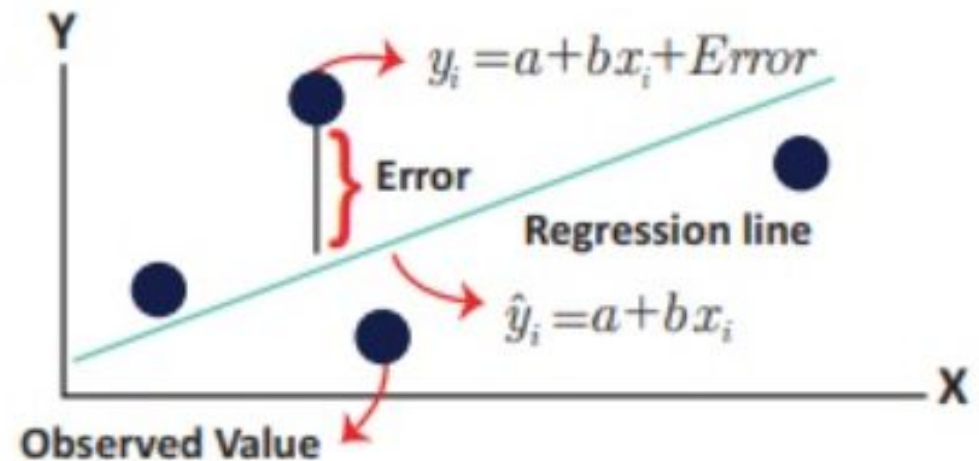
$$\frac{\partial E(a,b)}{\partial b} = -2 \sum_{i=1}^n x_i (y_i - a - bx_i) = 0 \Rightarrow a \sum_{i=1}^n x_i + b \sum_{i=1}^n x_i^2 = \sum_{i=1}^n x_i y_i \Rightarrow a \sum x + b \sum x^2 = \sum xy$$

- These equations are popularly known as normal equations. Solving these equations for 'a' and 'b' yield the estimates \hat{a} and \hat{b} .

$$\hat{a} = \frac{1}{n} \sum_{i=1}^n y_i - \hat{b} \frac{1}{n} \sum_{i=1}^n x_i$$

$$\hat{b} = \frac{\frac{1}{n} \sum_{i=1}^n x_i y_i - (\bar{x} \times \bar{y})}{\frac{1}{n} \sum_{i=1}^n x_i^2 - \bar{x}^2}$$

Simple Linear Regression Model



$$a_1 = \frac{n \sum_{i=1}^n x_i y_i - \sum_{i=1}^n x_i \sum_{i=1}^n y_i}{n \sum_{i=1}^n x_i^2 - \left(\sum_{i=1}^n x_i \right)^2}$$

$$a_0 = \frac{\sum_{i=1}^n y_i}{n} - a_1 \frac{\sum_{i=1}^n x_i}{n}$$

Example 1

- Find the least square line $y=a+bx$ for the data:

x	-2	-1	0	1	2
y	1	2	3	3	4

Solution:

x	y	x^2	xy
-2	1	4	-2
-1	2	1	-2
0	3	0	0
1	3	1	3
2	4	4	8
$\sum x = 0$	$\sum y = 13$	$\sum x^2 = 10$	$\sum xy = 7$

The normal equations are

$$na + b \sum x = \sum y$$

$$a \sum x + b \sum x^2 = \sum xy$$

Example 1

The normal equations are

$$na + b \sum x = \sum y$$

$$a \sum x + b \sum x^2 = \sum xy$$

Putting the values of n , $\sum x$, $\sum y$, $\sum x^2$, $\sum xy$ in the above equation, we get

$$5a + b \cdot 0 = 13 \Rightarrow 5a = 13$$

$$a \cdot 0 + b(10) = 7 \Rightarrow 10b = 7$$

From Eqs. (12.3) and (12.4) we get

$$a = \frac{13}{5} = 2.6, \quad b = \frac{7}{10} = 0.7$$

The required of least square line is

$$y = 2.6 + (0.7)x$$

Example 2

- Fit a straight line trend by the method of least square from the following data and find the trend values.

Year	1958	1959	1960	1961	1962
Sales (in lakhs of units)	65	95	80	115	105

- Solution:

Year	Sales	x	x^2	xy
1958	65	-2	4	-130
1959	95	-1	1	-95
1960	80	0	0	0
1961	115	1	1	115
1962	105	2	4	210
Total	$\sum y = 460$	$\sum x = 0$	$\sum x^2 = 10$	$\sum xy = 100$

Example 2

- Solution:

$$a = \frac{\sum y}{n} \Rightarrow a = \frac{460}{5} = 92$$

$$b = \frac{\sum xy}{\sum x^2} = \frac{100}{10} = 10$$

∴ The equation of the straight line trend is

$$y_c = a + bx \Rightarrow y_c = 92 + 10x$$

Year	Sales	Trend value y_c
1958	65	72
1959	95	82
1960	80	92
1961	115	102
1962	105	112

Example 3

- Fit a straight line trend by the method of least square from the following data and find the trend values.

Year	1950	1951	1952	1953	1954	1955	1956	1957
Value	346	411	392	512	626	640	611	796

- Solution:

Year	x	y	x ²	xy	Trend: $y=541.75+(59.60)x$
1950	-3.5	346	12.25	-1211.00	333.15
1951	-2.5	411	6.25	-1027.50	392.75
1952	-1.5	392	2.25	-588.00	452.35
1953	-0.5	512	0.25	-256.00	511.95
1954	0.5	626	0.25	313.00	571.95
1955	1.5	640	2.25	960.00	631.15
1956	2.5	611	6.25	1527.50	690.75
1957	3.5	796	12.25	2786.00	750.35
Total	0	4334	42.00	2504.00	

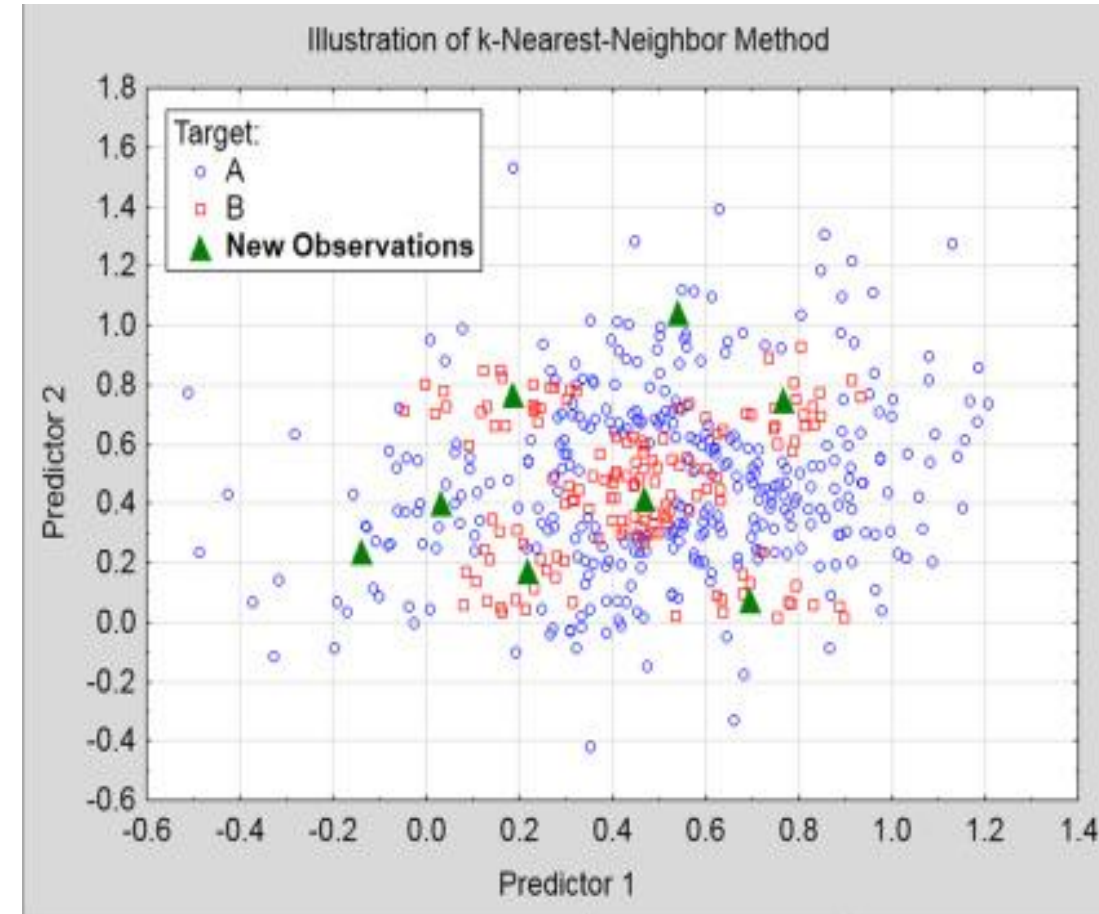
Nearest Neighbor Method

What is **Nearest Neighbor Method**

- "Nearest Neighbour" is merely "k-Nearest Neighbours" with $k=1$.
- The k-nearest neighbor (KNN) method is a good example of a “general approximator” that is entirely based on patterns in the data, without any specific “statistical model” that must be estimated.
- In fact, KNN methods do not rely on any “models” at all but instead simply use the existing data (or a sample of exemplars from the existing data) to “predict” new observations.
- Specifically, when predicting a new observation, the algorithm finds the most similar observations among the exemplars with respect to the available predictors and then makes the prediction that the new observation will have the same outcome (same predicted y , or predicted classification).

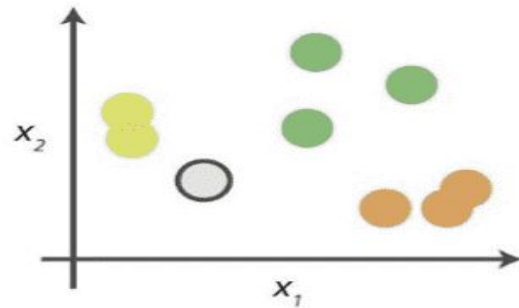
What is **Nearest Neighbor Method**

- For example, consider the classification problem illustrated in this figure.
- In this case there are two predictors—Predictor 1 and Predictor 2—and a number of exemplars or individual square (red) and round (blue) points that belong to category A or B, respectively.
- Now consider the problem of predicting new observations: triangles (green) to belong either to category A or B.



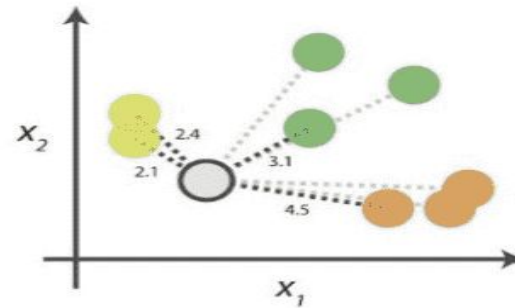
Steps in KNN Classification

0. Look at the data











Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances









Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point Distance			
	...	 2.1	→ 1st NN
	...	 2.4	→ 2nd NN
	...	 3.1	→ 3rd NN
	...	 4.5	→ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

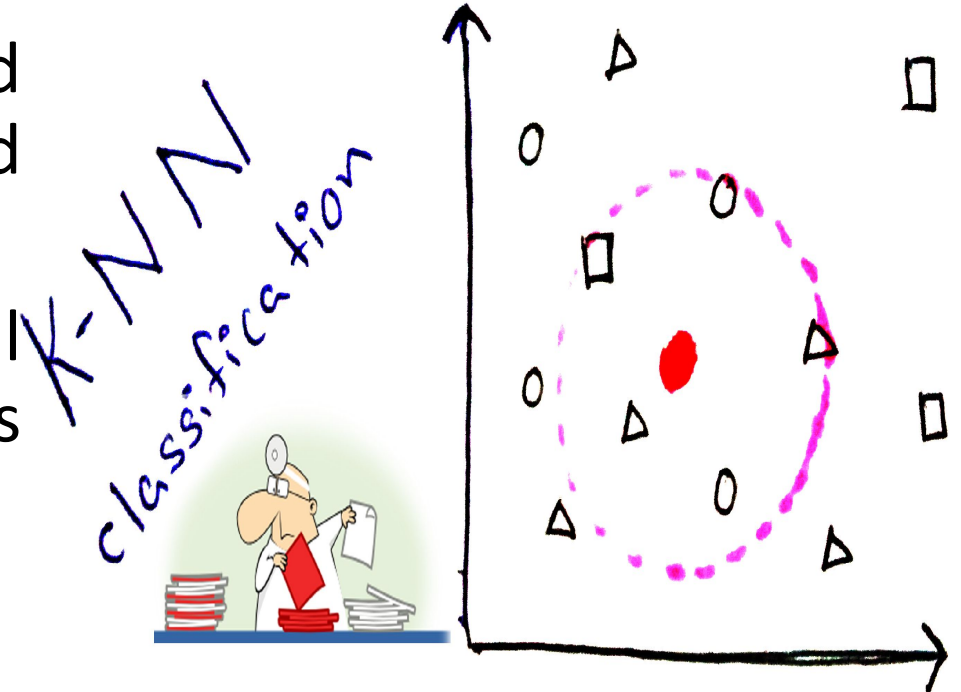
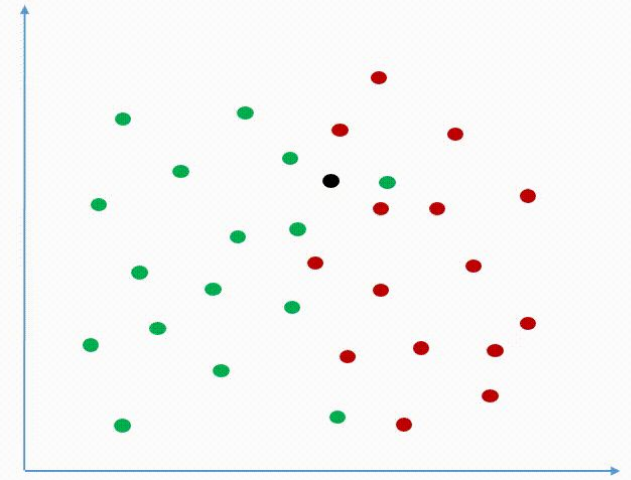
3. Vote on labels

Class	# of votes	
	2	→ Class  wins the vote! Point  is therefore predicted to be of class  .
	1	
	1	

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Application of KNN Classification

- K is the most important hyperparameter in the KNN algorithm.
- KNN makes predictions based on the K nearest instances in the training data.
- KNN can be used for both classification and regression; both supervised and unsupervised learning (K-Means clustering).
- As it has all the training data in memory, it will simply compare the new data with k of its nearest data points.

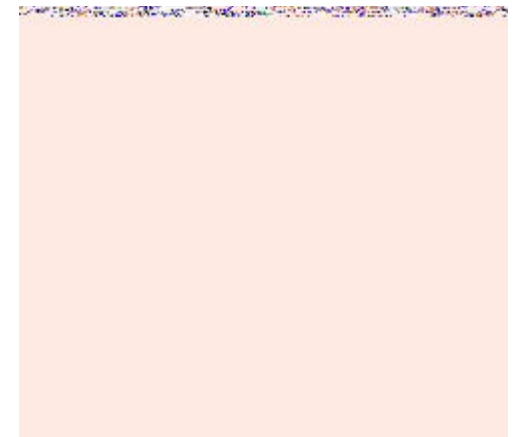
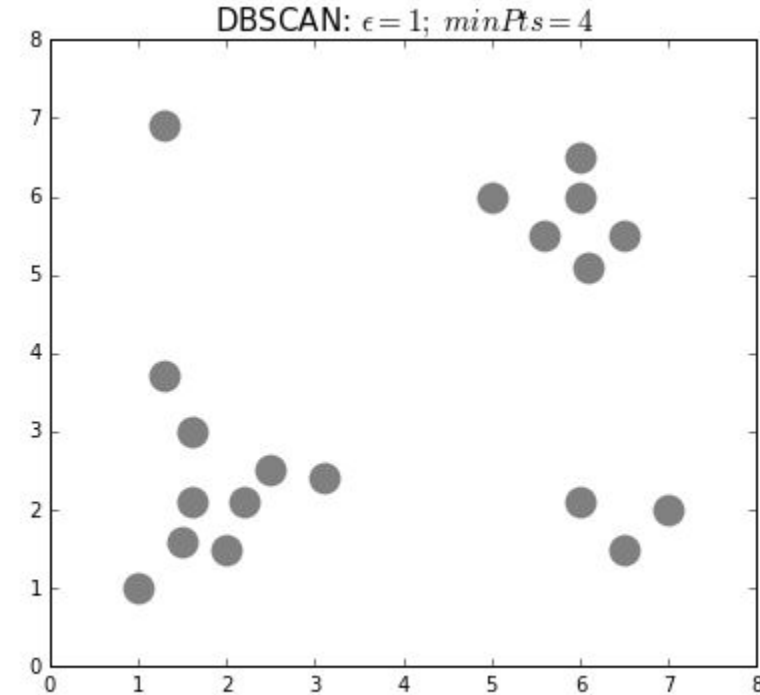


Application of KNN Classification

- The k-NN algorithm is a classification technique that does not make assumptions about the model between the class membership (y) and the features (x_1, x_2, \dots, x_n).
- This is a non-parametric technique as it does not involve estimation of parameters in an assumed model.
- Rather, this method pulls out information from **similarities** existing amongst the values of the patterns in the dataset.
- This model-free algorithm finds similar past patterns or instances from the training set with the use of a suitable distance measure and interpolates from them the correct output.
- In machine learning, these techniques are also known as instance-based or memory-based learning algorithms.

Differences between K-means and K-NN?

- **K-means** is a clustering algorithm that tries to partition a set of points into K sets (clusters) such that the points in each cluster tend to be near each other.
 - It is unsupervised because the points have no external classification.
- **K-nearest neighbors** is a classification (or regression) algorithm that in order to determine the classification of a point, combines the classification of the K nearest points.
 - It is supervised because you are trying to classify a point based on the known classification of other points.



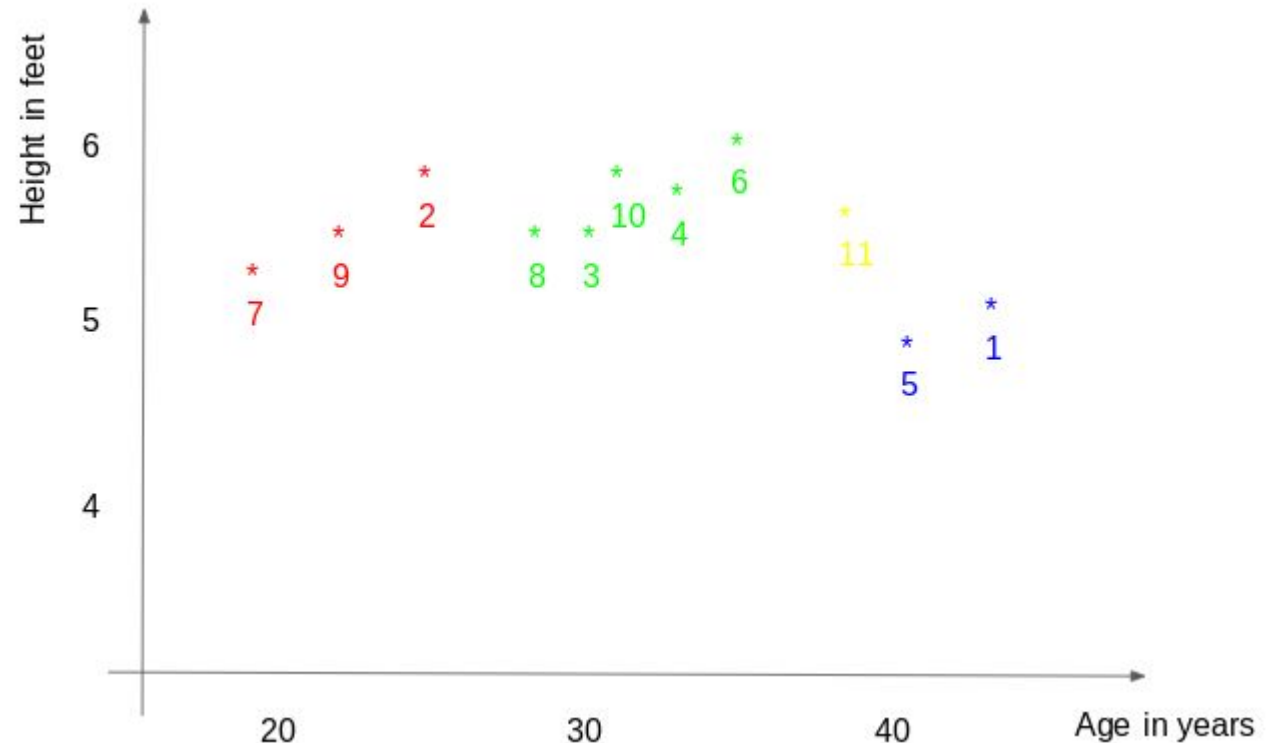
Example

- Predict the weight of the person with ID 11 based on his height and age.

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

Solution:

below is the plot of height versus age from the given table



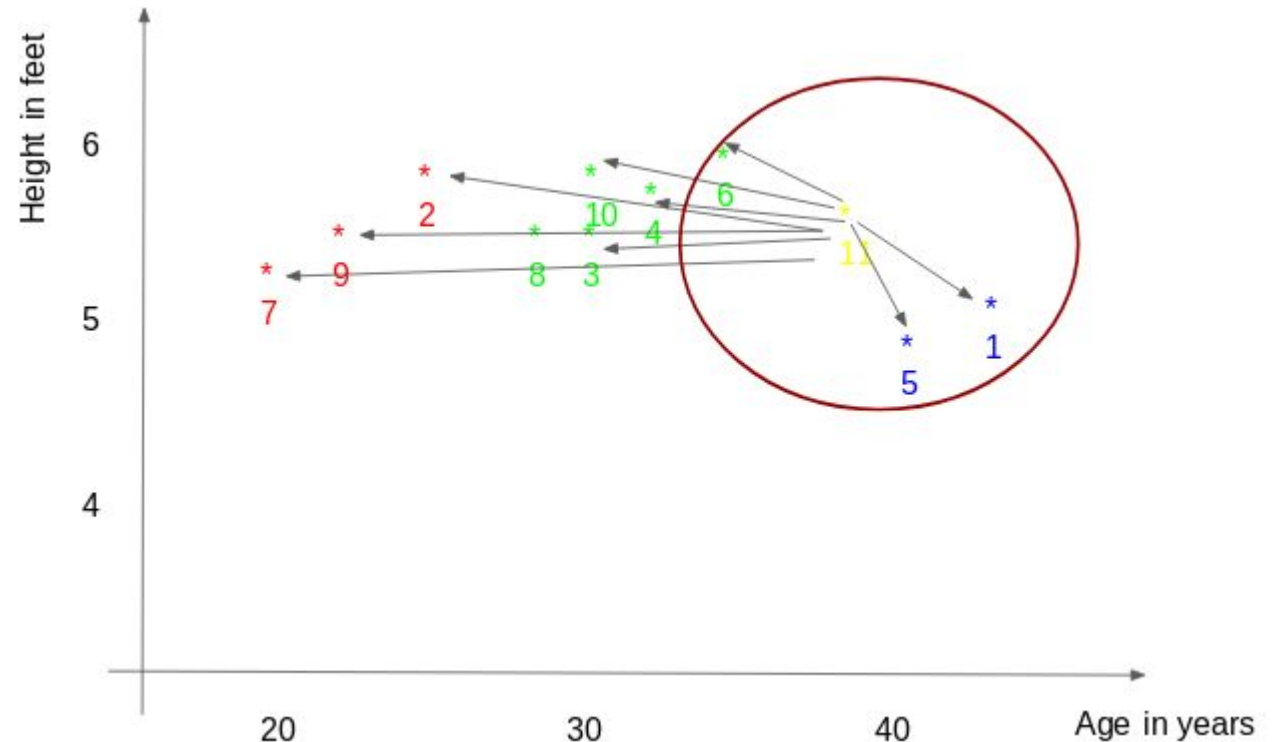
Example

- Predict the weight of the person with ID 11 based on his height and age.

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

Solution:

for a value $k = 3$, the closest points are ID1, ID5 and ID6.



Here, we get weight of ID11 = $(77+72+60)/3 = 69.66$ kg.

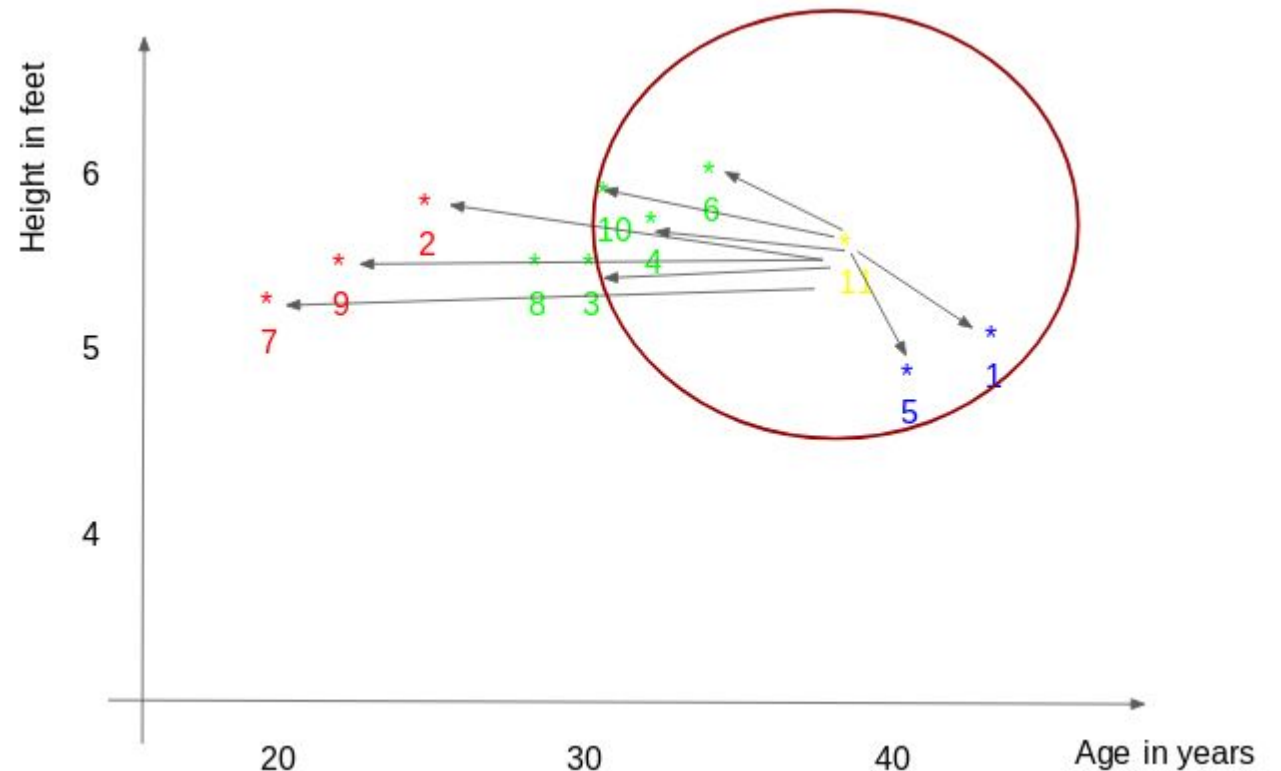
Example

- Predict the weight of the person with ID 11 based on his height and age.

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

Solution:

for a value $k = 5$, the closest points are ID1, ID4, ID5, ID6, ID10.



Here, we get weight of ID11 = $(77+59+72+60+58)/5 = 65.2$ kg.

Constraints

- Based on the k value, the final result tends to change; Then how can we figure out the optimum value of k ?
 - through finding training and validating/testing errors
- KNN works best when the dataset is not too large.
- The main reason being that the whole dataset is stored in memory. Therefore, when the number of samples increases, then KNN can become inefficient/slow.
- Another thing to notice is that KNN models are not really 'trained'; The dataset is just memorized.
- So, with large datasets, the main problem begins when finding the neighbors for a new data point, that is the testing phase.
- The most serious shortcoming of nearest-neighbor methods is that they are very sensitive to the presence of irrelevant parameters.

Distance Based Learning

What is **Distance Based Learning**

- As the name implies, distance-based learning/models work on the concept of distance.
- A few of the More popular machine learning algorithms that use distance measures at their core is
 - **K-Nearest Neighbors (KNN)**
 - **Learning Vector Quantization (LVQ)**: A type of neural network that lets you choose how many training instances to hang onto.
 - **Self-Organizing Map (SOM)**: A neural network-based dimensionality reduction algorithm.
 - **K-Means Clustering**
 - **kernel-based methods such as Support Vector Machine (SVM) algorithm**

Distance as measure of similarity

- Characterizing the similarity of the patterns in state space can be done through some form of metric (distance) measure: distance between two vectors is a measure of similarity between two corresponding patterns.
- Let us use d_{il} to depict a distance metric or dissimilarity measure, between patterns i and l . For pattern i , we have the vector of n measurements $(x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)})$, while for pattern l , we have the vector of n measurements $(x_1^{(l)}, x_2^{(l)}, \dots, x_n^{(l)})$.

Properties of Distances

- Distances can be defined in multiple ways, but in general the following properties are required:

Nonnegativity: $d_{il} \geq 0$

Self-proximity: $d_{ii} = 0$

Symmetry: $d_{il} = d_{li}$

Triangle Inequality: $d_{il} \leq d_{ik} + d_{kl}$

Types of Distances

- Euclidean distance
- Statistical distance
- Manhattan distance
- Minkowski metric
- Hamming distance

Euclidean distance

Euclidean distance: The most popular distance measure is the *Euclidean distance*, d_{il} , which between two patterns i and l , is defined by,

$$d_{il} = \sqrt{(x_1^{(i)} - x_1^{(l)})^2 + (x_2^{(i)} - x_2^{(l)})^2 + \dots + (x_n^{(i)} - x_n^{(l)})^2}$$

Despite Euclidean distance being the most commonly used similarity measure, there are three primary characteristics that need to be taken into consideration.

- (i) It depends highly on scale, and variables that possess bigger scales impact the total distance to a great extent. Thus, we first *normalize* continuous measurements and only then calculate the Euclidean distance. This transforms all measurements to the same scale.
- (ii) Euclidean distance entirely ignores the relationship between measurements. If there is a strong correlation between measurements, a *statistical distance* measure seems to be a better candidate.
- (iii) Euclidean distance is sensitive to outliers, and if the data comprises outliers, a preferred choice is the use of robust distances like *Manhattan distance*.

Normalized data

	x1	x2	x3		x1n	x2n	x3n
1	53	5	4646		1	0.833333333	0.388461538
2	53	5	11960		1	0.833333333	1
3	50	6	534		0.980392157	0.000820345	0.048090778
4	34	4	6688		0.666666667	0.000350079	0.602305476
5	45	4	6023	$x_{ij}=x_{ij}/\max(x_i)$	0.882352941	0.000350079	0.542417147
6	29	6	9014	(normalized)	0.568627451	0.000525118	0.811779539
7	33	6	6251		0.647058824	0.000525118	0.562950288
8	31	6	1093		0.607843137	0.000525118	0.098432997
9	21	5	6259		0.411764706	0.000437598	0.563670749
10	41	4	8715		0.803921569	0.000350079	0.784852305
11	51	5	10877		1	0.000437598	0.979556916
12	21	6	11104		0.6	0.000525118	1
13	35	6	10238		1	0.000525118	1
14	20	6	4985		1	0.000525118	1
	d12	7314				0.611538462	
	d23	11426.00044				1.264750375	
	d34	6154.021124				0.636849931	
	d45	665.0909712				0.223846333	
	d56	2991.043463				0.413497052	

Euclidean distance

- Example:
- `row1 = [10, 20, 15, 10, 5]`
- `row2 = [12, 24, 18, 8, 7]`
- `euclidean_distance(row1, row2) = 6.082762530298219`

Statistical distance

Statistical distance: This metric takes into consideration the correlation between measurements. With this metric, measurements extremely correlated with other measurements do not contribute as much as the uncorrelated or less correlated. The *statistical distance*, also referred to as the *Mahalanobis distance*, between patterns i and l is defined as,

$$d_{il} = \sqrt{(\mathbf{x}^{(i)} - \mathbf{x}^{(l)})^T \mathbf{\Sigma}^{-1} (\mathbf{x}^{(i)} - \mathbf{x}^{(l)})}$$

where $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(l)}$ are n -dimensional vectors of measurement values of patterns i and l , respectively, and $\mathbf{\Sigma}$ is the covariance matrix of these vectors.

Manhattan distance

Manhattan distance: This distance looks at the absolute differences rather than squared differences, and is defined by,

$$d_{il} = \sum_{j=1}^n |x_j^{(i)} - x_j^{(l)}|$$

- Example:
- row1 = [10, 20, 15, 10, 5]
- row2 = [12, 24, 18, 8, 7]
- `manhattan_distance(row1, row2) = 13`

Minkowski metric

Minkowski metric: One general class of metrics for n -dimensional patterns is the Minkowski metric (also referred to as the L_p norm):

$$L_p(\mathbf{x}^{(i)}, \mathbf{x}^{(l)}) = \left(\sum_{j=1}^n |x_j^{(i)} - x_j^{(l)}|^p \right)^{1/p}$$

where $p \geq 1$ is a selectable parameter. Setting $p = 2$ gives the familiar Euclidean distance (L_2 norm) and setting $p = 1$ gives the Manhattan distance (L_1 norm). With respect to Eqns given before

$$L_p(\mathbf{x}^{(i)}, \mathbf{x}^{(l)}) = \|\mathbf{x}^{(i)} - \mathbf{x}^{(l)}\|_p = \left(\sum_{j=1}^n |x_j^{(i)} - x_j^{(l)}|^p \right)^{1/p}$$

(Minkowski norm)

$$\|\mathbf{x}^{(i)} - \mathbf{x}^{(l)}\|_2 = [(x_1^{(i)} - x_1^{(l)})^2 + (x_2^{(i)} - x_2^{(l)})^2 + \dots + (x_n^{(i)} - x_n^{(l)})^2]^{1/2}$$

(Euclidean norm)

$$\|\mathbf{x}^{(i)} - \mathbf{x}^{(l)}\|_1 = \sum_{j=1}^n |x_j^{(i)} - x_j^{(l)}|$$

(Manhattan norm)

Hamming distance

- Hamming distance calculates the distance between two binary vectors, also referred to as binary strings or bitstrings

- **For a One-hot encoded string**

Hamming Distance = sum for i to N $\text{abs}(v1[i] - v2[i])$

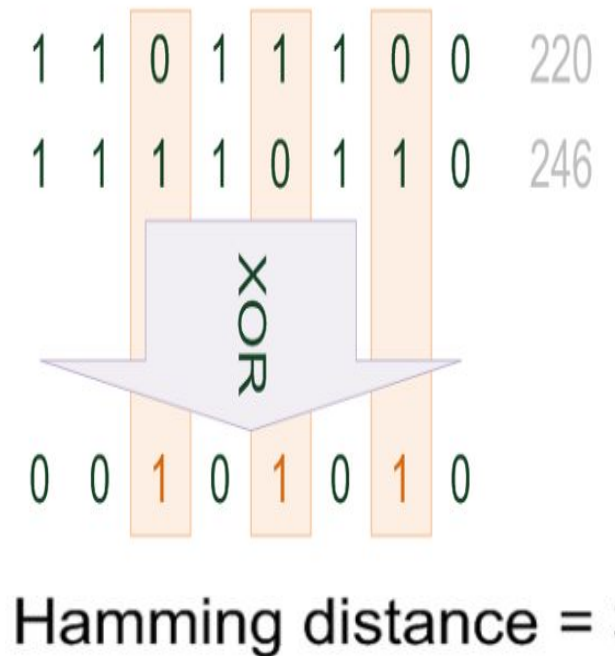
- **For bitstrings that may have many 1 bits**

Hamming Distance = $(\text{sum for i to N } \text{abs}(v1[i] - v2[i]))/N$

example: row1 = [0,0,0,0,0,1]

row2 = [0,0,0,0,1,0]

`hamming_distance(row1, row2)=0.3333`



Scope for Research (just an example)

<https://www.sciencedirect.com/science/article/pii/S1568494620307936>



Applied Soft Computing



Volume 98, January 2021, 106855



Standardized Variable Distances: A distance-based machine learning method

Abdullah Elen ^a , Emre Avcu ^b 

Show more 

+ Add to Mendeley  Share  Cite

<https://doi.org/10.1016/j.asoc.2020.106855>

[Get rights and content](#)

Highlights

- The Standardized Variable Distances (SVD) is a novel machine learning algorithm.
- The SVD is an improved version of the Minimum Distance Classifier (MDC) algorithm.