Unit-4

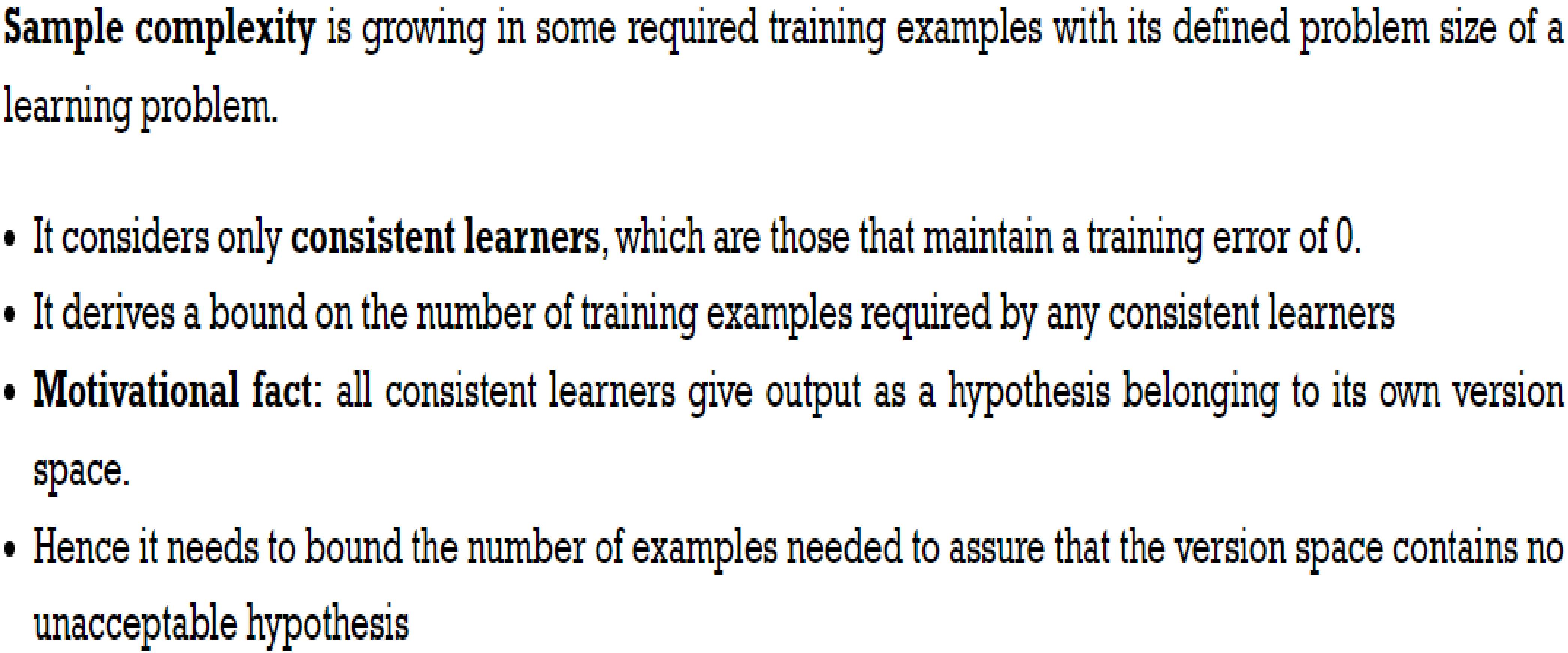
Computational Learning Theory

- Computational learning theory, or *CoLT* for short, is a field of study concerned with the use of formal mathematical methods applied to learning systems.
- Computational learning theory may be thought of as an extension or sibling of statistical learning theory, or *SLT* for short, that uses formal methods to quantify learning algorithms.
- **Computational Learning Theory** (*CoLT*): Formal study of learning tasks.
- **Statistical Learning Theory** (*SLT*): Formal study of learning algorithms.

- Computational learning theory is essentially a sub-field of artificial intelligence (AI) that focuses on studying the design and analysis of machine learning algorithms.
- **How important is computational learning theory?**
- Computational learning theory provides a formal framework in which it is possible to precisely formulate and address questions regarding the performance of different learning algorithms. Thus, careful comparisons of both the predictive power and the computational efficiency of competing learning algorithms can be made. Three key aspects that must be formalized are:

1. The way in which the learner interacts with its environment,
2. The definition of success in completing the learning task,
3. A formal definition of efficiency of both data usage (sample complexity) and processing time (time complexity).

Sample Complexity for Finite Hypothesis spaces



Sample complexity is growing in some required training examples with its defined problem size of a learning problem.

- It considers only **consistent learners**, which are those that maintain a training error of 0.
- It derives a bound on the number of training examples required by any consistent learners
- **Motivational fact**: all consistent learners give output as a hypothesis belonging to its own version space.
- Hence it needs to bound the number of examples needed to assure that the version space contains no unacceptable hypothesis

Sample Complexity for Infinite Hypothesis spaces

Motivation: classification is to distinguish subsets.

- A set of instances S and a hypothesis space H.

- Can H distinguish all subsets of S?

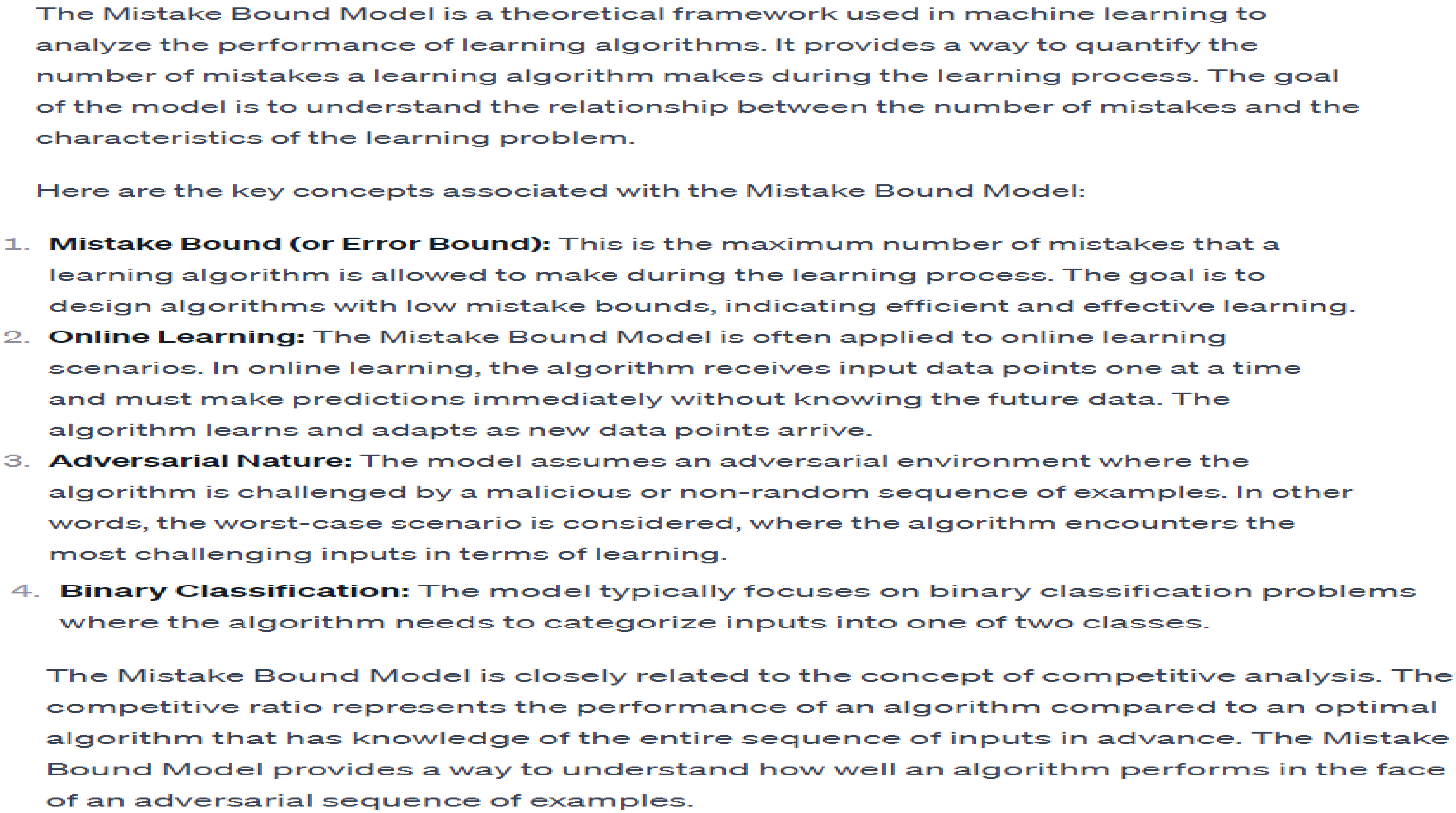
- The number of concepts over S: 2 S

- (Def) S is shattered by H if all the concepts over S are included in H.

- I.e., for any bi-partition ($S1$, $S2$) of S, there exists one h in H, such that $h(s) = 0$ for every $s \in S1$ and $h(s) = 1$ for each $s \in S2$.

- Corollary: $H \geq 2$ S if H can shatter S.

The Mistake Bound Model of Learning

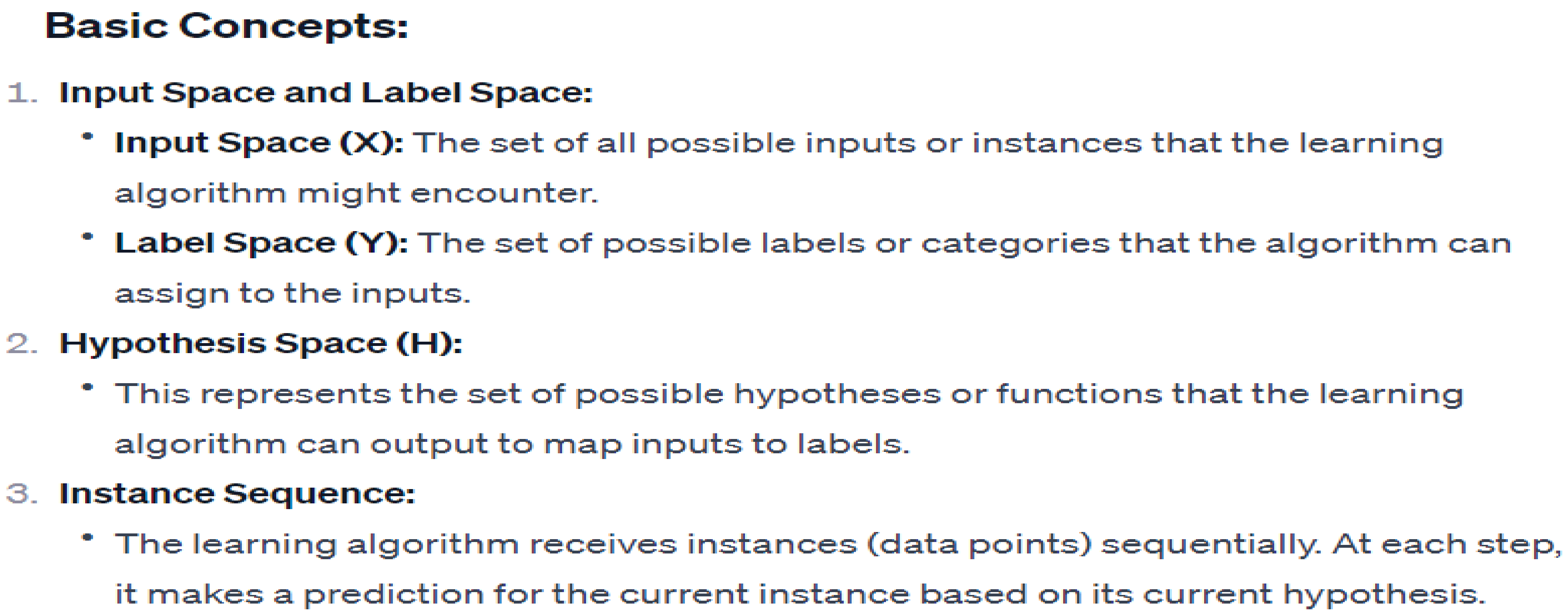


The Mistake Bound Model is a theoretical framework used in machine learning to analyze the performance of learning algorithms. It provides a way to quantify the number of mistakes a learning algorithm makes during the learning process. The goal of the model is to understand the relationship between the number of mistakes and the characteristics of the learning problem.

Here are the key concepts associated with the Mistake Bound Model:

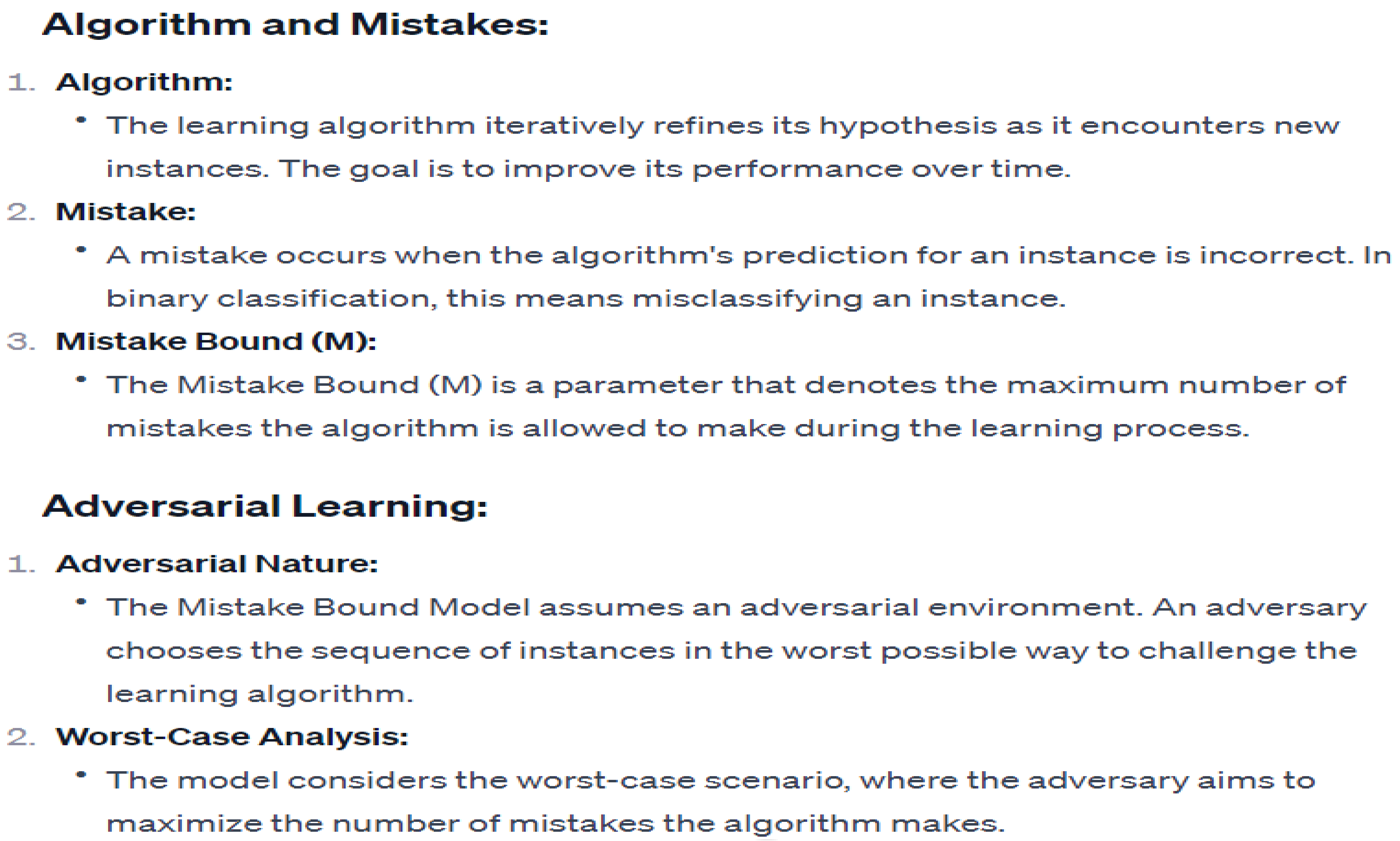
1. **Mistake Bound (or Error Bound):** This is the maximum number of mistakes that a learning algorithm is allowed to make during the learning process. The goal is to design algorithms with low mistake bounds, indicating efficient and effective learning.
2. **Online Learning:** The Mistake Bound Model is often applied to online learning scenarios. In online learning, the algorithm receives input data points one at a time and must make predictions immediately without knowing the future data. The algorithm learns and adapts as new data points arrive.
3. **Adversarial Nature:** The model assumes an adversarial environment where the algorithm is challenged by a malicious or non-random sequence of examples. In other words, the worst-case scenario is considered, where the algorithm encounters the most challenging inputs in terms of learning.
4. **Binary Classification:** The model typically focuses on binary classification problems where the algorithm needs to categorize inputs into one of two classes.

The Mistake Bound Model is closely related to the concept of competitive analysis. The competitive ratio represents the performance of an algorithm compared to an optimal algorithm that has knowledge of the entire sequence of inputs in advance. The Mistake Bound Model provides a way to understand how well an algorithm performs in the face of an adversarial sequence of examples.



Basic Concepts:

1. **Input Space and Label Space:**
   - **Input Space (X):** The set of all possible inputs or instances that the learning algorithm might encounter.
   - **Label Space (Y):** The set of possible labels or categories that the algorithm can assign to the inputs.
2. **Hypothesis Space (H):**
   - This represents the set of possible hypotheses or functions that the learning algorithm can output to map inputs to labels.
3. **Instance Sequence:**
   - The learning algorithm receives instances (data points) sequentially. At each step, it makes a prediction for the current instance based on its current hypothesis.



Algorithm and Mistakes:

1. **Algorithm:**
   - The learning algorithm iteratively refines its hypothesis as it encounters new instances. The goal is to improve its performance over time.
2. **Mistake:**
   - A mistake occurs when the algorithm's prediction for an instance is incorrect. In binary classification, this means misclassifying an instance.
3. **Mistake Bound (M):**
   - The Mistake Bound (M) is a parameter that denotes the maximum number of mistakes the algorithm is allowed to make during the learning process.

Adversarial Learning:

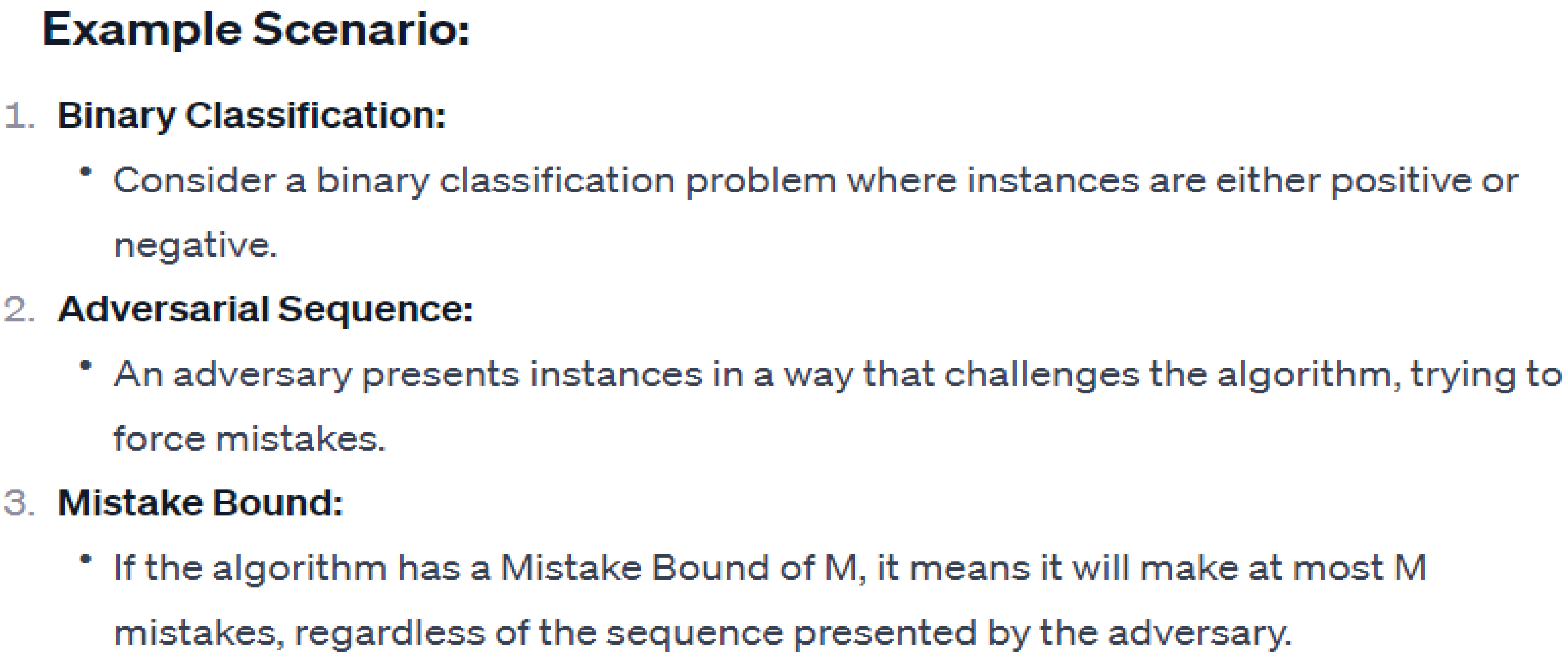
1. **Adversarial Nature:**
   - The Mistake Bound Model assumes an adversarial environment. An adversary chooses the sequence of instances in the worst possible way to challenge the learning algorithm.
2. **Worst-Case Analysis:**
   - The model considers the worst-case scenario, where the adversary aims to maximize the number of mistakes the algorithm makes.

Learning Process:

1. **Online Learning:**
   - The algorithm processes instances one by one, updating its hypothesis after each instance. It must make predictions without knowledge of future instances.
2. **Adaptation:**
   - The algorithm adapts to the sequence of instances, attempting to minimize mistakes over time.

Theoretical Guarantees:

1. **Competitive Analysis:**
   - The Mistake Bound Model often employs competitive analysis to compare the performance of the learning algorithm against an optimal algorithm with knowledge of the entire instance sequence.
2. **Performance Guarantees:**
   - The model provides theoretical guarantees on the algorithm's performance, stating that it will make at most M mistakes over the sequence of instances.



Example Scenario:

1. **Binary Classification:**
   - Consider a binary classification problem where instances are either positive or negative.
2. **Adversarial Sequence:**
   - An adversary presents instances in a way that challenges the algorithm, trying to force mistakes.
3. **Mistake Bound:**
   - If the algorithm has a Mistake Bound of M, it means it will make at most M mistakes, regardless of the sequence presented by the adversary.

Instance-based learning

- The Machine Learning systems which are categorized as instance-based learning are the systems that learn the training examples by heart and then generalizes to new instances based on some similarity measure. **It is called instance-based because it builds the hypotheses from the training instances.**
- It is also known as **memory-based learning** or **lazy-learning** (because they delay processing until a new instance must be classified).
- The time complexity of this algorithm depends upon the size of training data. Each time whenever a new query is encountered, its previously stores data is examined. And assign to a target function value for the new instance.

- For example, **If we were to create a spam filter with an instance-based learning algorithm, instead of just flagging emails that are already marked as spam emails, our spam filter would be programmed to also flag emails that are very similar to them.** This requires a measure of resemblance between two emails. A similarity measure between two emails could be the same sender or the repetitive use of the same keywords or something else.
- **Advantages:**

1. Instead of estimating for the entire instance set, local approximations can be made to the target function.
2. This algorithm can adapt to new data easily.

- **Disadvantages:**

1. Classification costs are high
2. Large amount of memory required to store the data, and each query involves starting the identification of a local model from scratch.

- Some of the instance-based learning algorithms are :

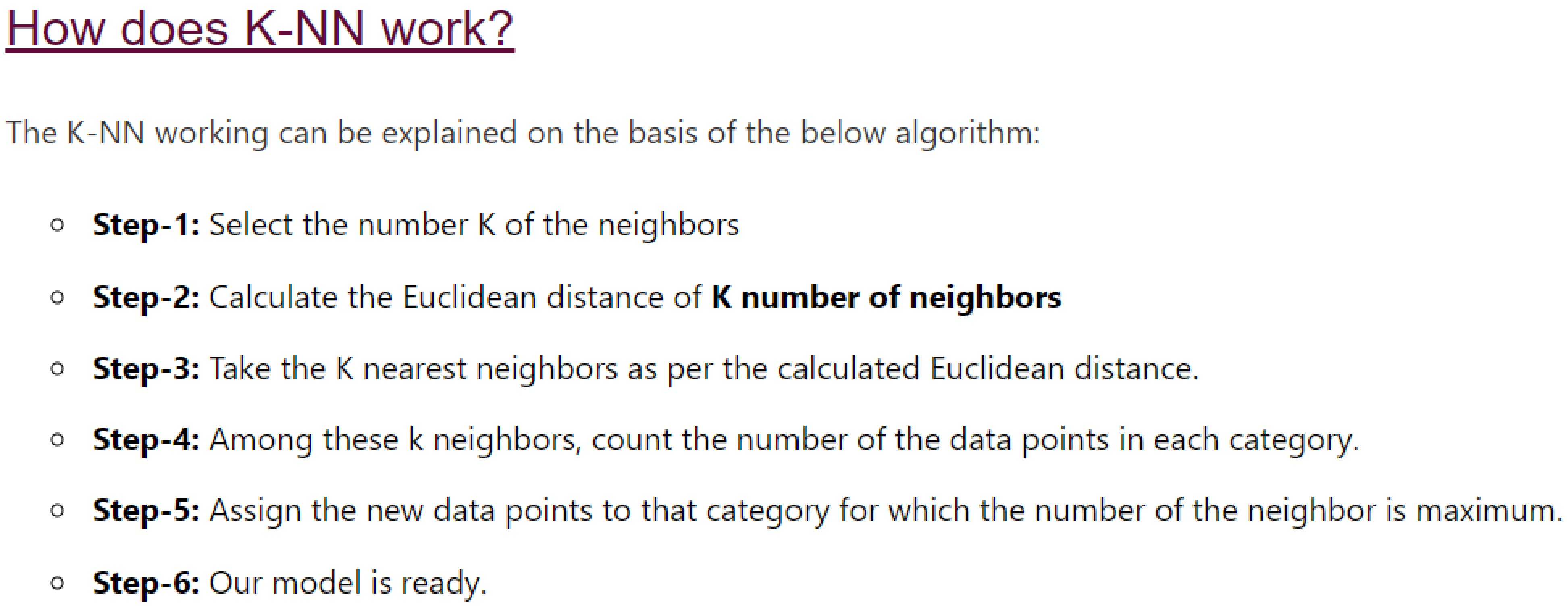
1. K Nearest Neighbor (KNN)
2. Self-Organizing Map (SOM)
3. Learning Vector Quantization (LVQ)
4. Locally Weighted Learning (LWL)
5. Case-Based Reasoning

K-Nearest Neighbor(KNN) Algorithm

- KNN algorithms based on Supervised Learning technique.
- KNN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- KNN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- KNN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- KNN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

- **Example:** Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

KNN Classifier
Input value
Predicted Output

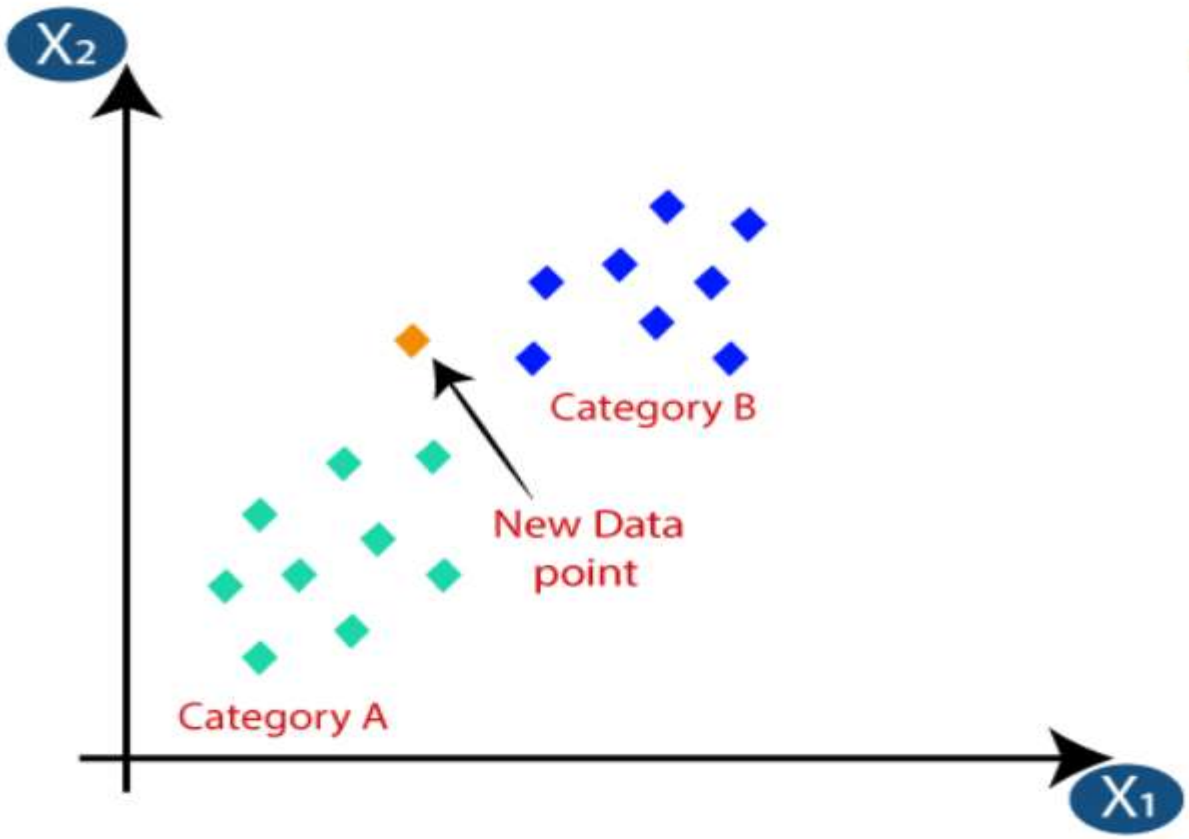


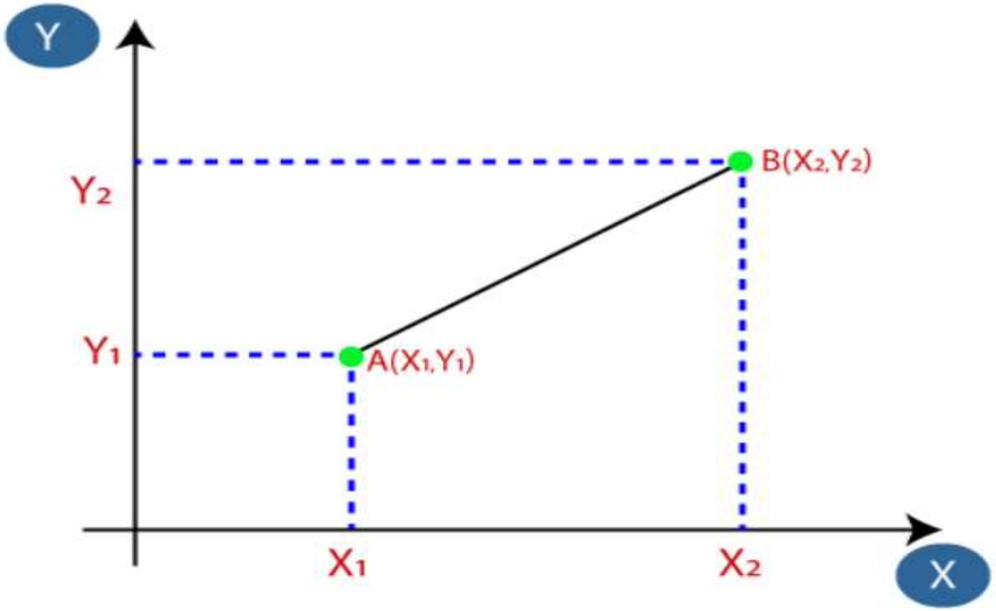
How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:

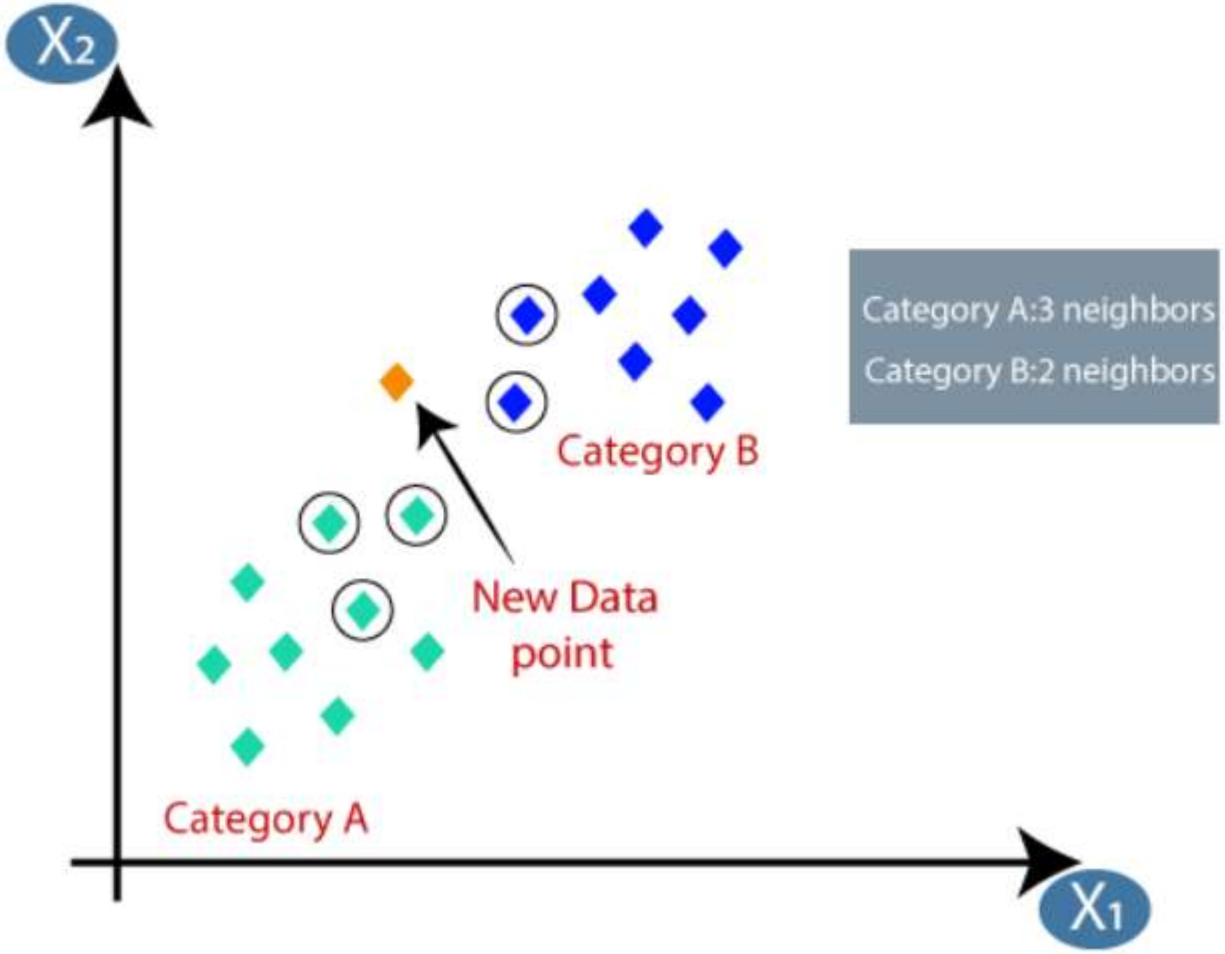




Euclidean Distance between A_1 and B_2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting the value of K in the K-NN algorithm:

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
- Large values for K are good, but it may find some difficulties.

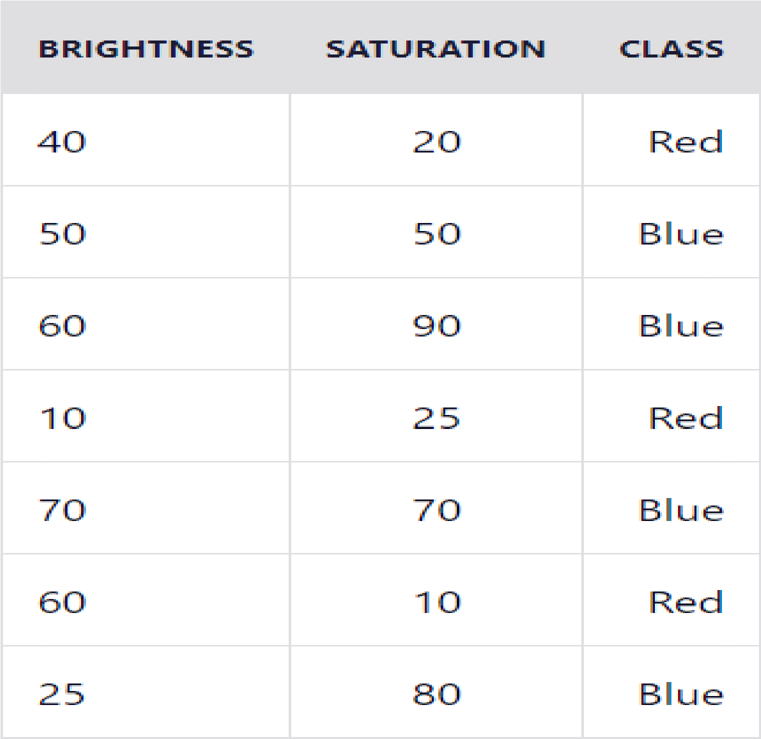
Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

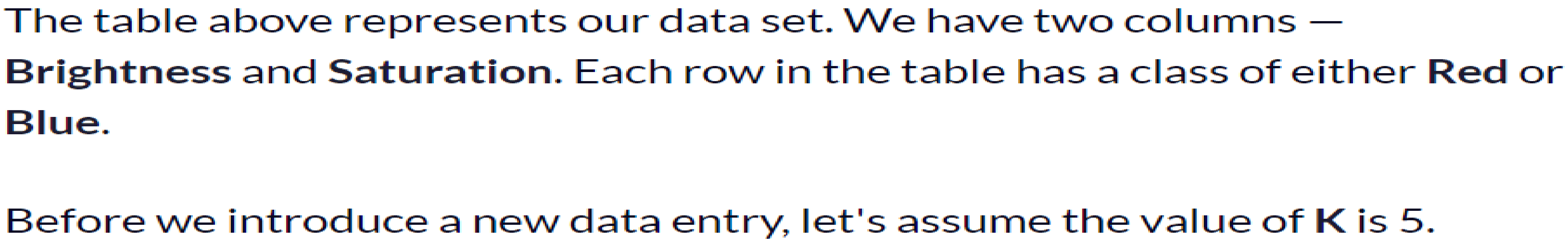
Disadvantages of KNN Algorithm:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

Example:



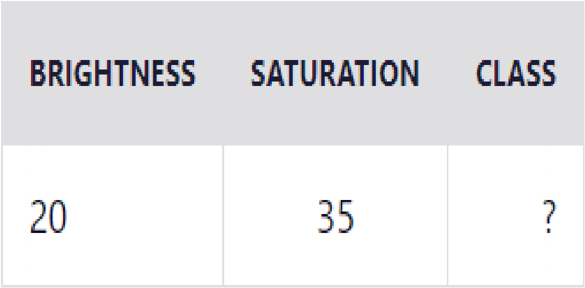
BRIGHTNESS	SATURATION	CLASS
40	20	Red
50	50	Blue
60	90	Blue
10	25	Red
70	70	Blue
60	10	Red
25	80	Blue



The table above represents our data set. We have two columns — **Brightness** and **Saturation**. Each row in the table has a class of either **Red** or **Blue**.

Before we introduce a new data entry, let's assume the value of **K** is 5.

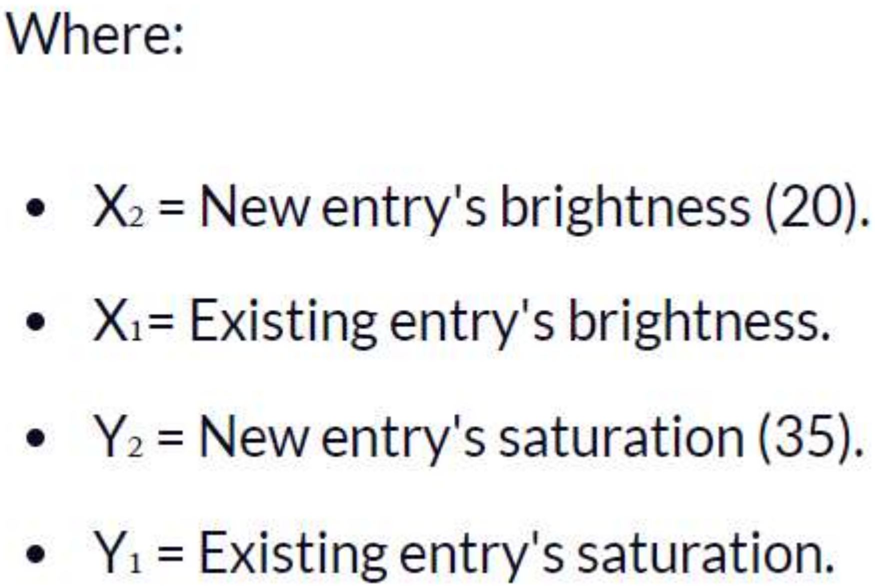
Here's the new data entry:



BRIGHTNESS	SATURATION	CLASS
20	35	?

We have a new entry but it doesn't have a class yet. To know its class, we have to calculate the distance from the new entry to other entries in the data set using the Euclidean distance formula.

Here's the formula: $\sqrt{}(X_2-X_1)^2+(Y_2-Y_1)^2$

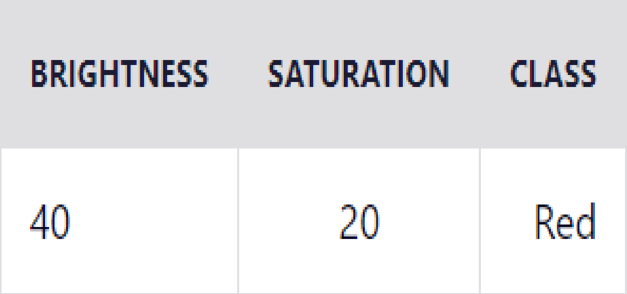


Where:

- X_2 = New entry's brightness (20).
- X_1= Existing entry's brightness.
- Y_2 = New entry's saturation (35).
- Y_1 = Existing entry's saturation.

Distance #1

For the first row, d1:



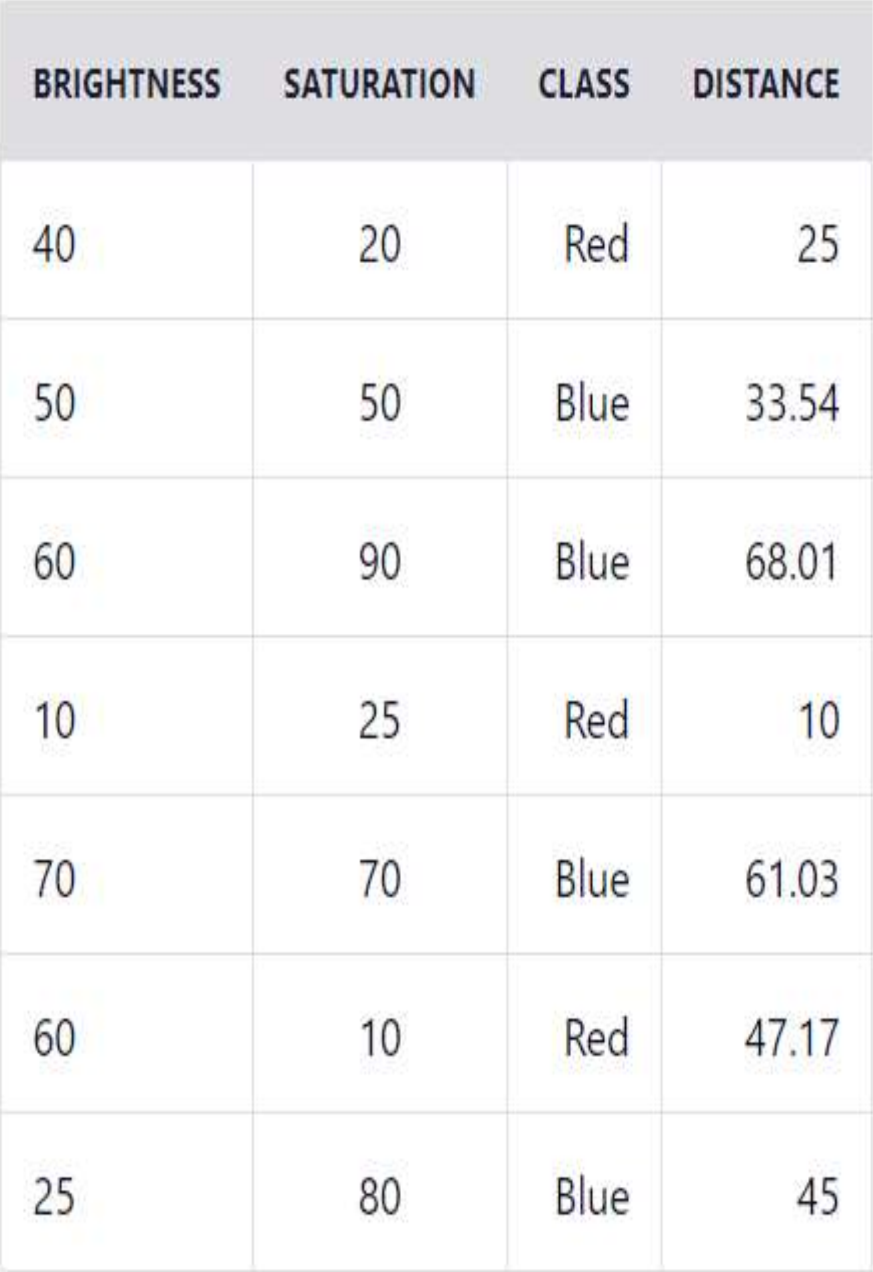
BRIGHTNESS	SATURATION	CLASS
40	20	Red

$$d1 = \sqrt{(20 - 40)^2 + (35 - 20)^2}$$

$$= \sqrt{400 + 225}$$

$$= \sqrt{625}$$

$$= 25$$



BRIGHTNESS	SATURATION	CLASS	DISTANCE
40	20	Red	25
50	50	Blue	33.54
60	90	Blue	68.01
10	25	Red	10
70	70	Blue	61.03
60	10	Red	47.17
25	80	Blue	45

Since we chose 5 as the value of **K**, we'll only consider the first five rows. That is:

BRIGHTNESS	SATURATION	CLASS	DISTANCE
10	25	Red	10
40	20	Red	25
50	50	Blue	33.54
25	80	Blue	45
60	10	Red	47.17

As you can see above, the majority class within the 5 nearest neighbors to the new entry is **Red**. Therefore, we'll classify the new entry as **Red**.

Regression

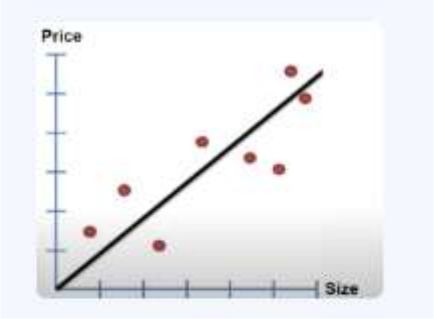
- Regression is defined as a statistical method that helps us to analyze and understand the relationship between two or more variables of interest.
- The process that is adapted to perform regression analysis helps to understand which factors are important, which factors can be ignored, and how they are influencing each other.
- **For the regression analysis is be a successful method two variables used:**

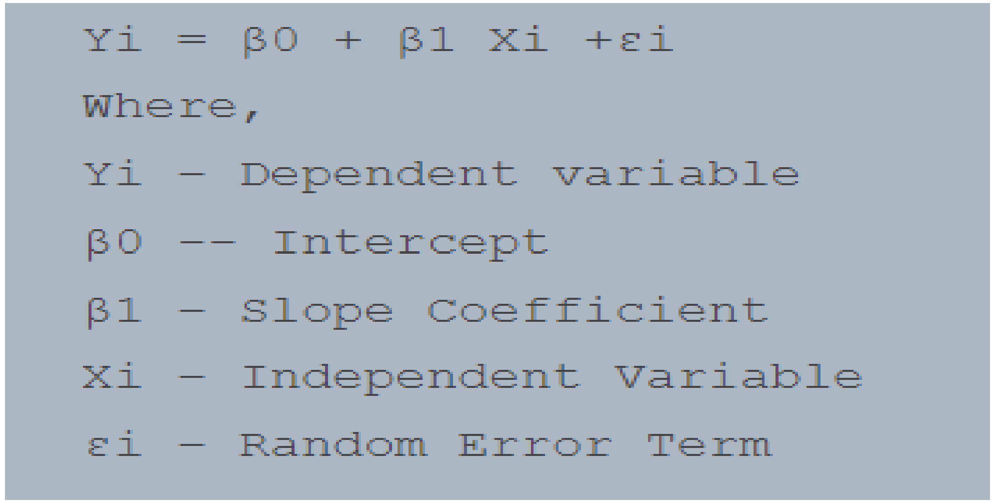
1. Dependent Variable: This is the variable that we are trying to understand or forecast.

2. Independent Variable: These are factors that influence the analysis or target variable and provide us with information regarding the relationship of the variables with the target variable.

Linear Regression

- The simplest of all regression types is Linear Regression which tries to establish relationships between Independent and Dependent variables.
- Linear Regression is a predictive model used for finding the linear relationship between a dependent variable and one or more independent variables.
- This regression works well when we are having linearly separable data.

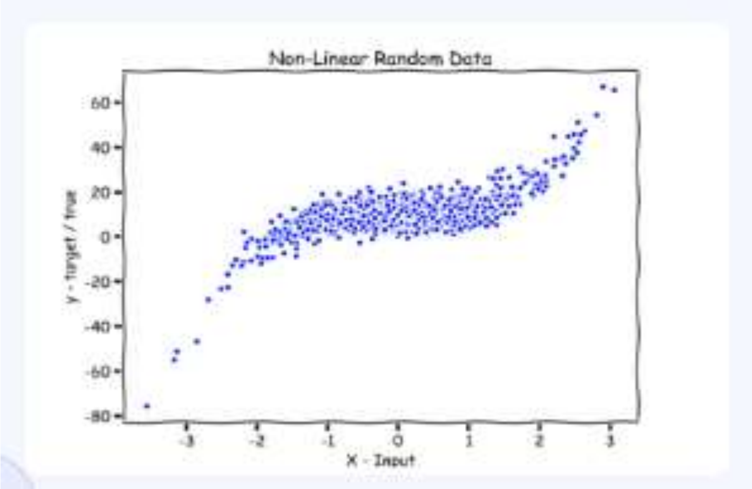




```
Yi = β0 + β1 Xi +εi
Where,
Yi – Dependent variable
β0 -- Intercept
β1 – Slope Coefficient
Xi – Independent Variable
εi – Random Error Term
```

Locally Weighted Linear Regression (LWLR)

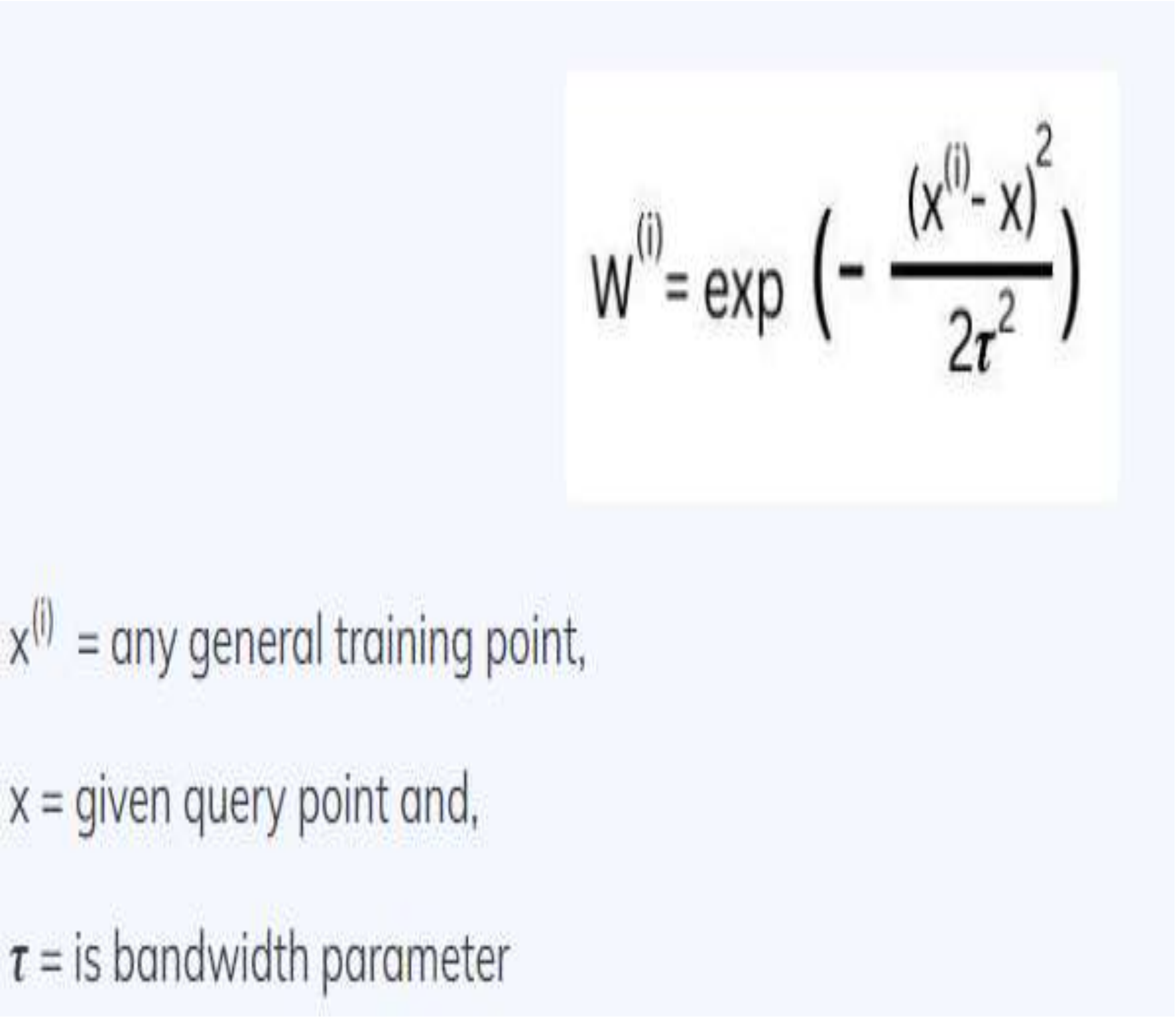
- It is a non-parametric algorithm, unlike a typical linear regression algorithm which is a parametric algorithm. A parametric algorithm is an algorithm that doesn’t need to retain the training data when we need to make predictions.
- This regression works well when we are having non linearly separable data.



Important Points:

- LWLR is a non-parametric regression technique that fits a linear regression model to a dataset by giving more weight to nearby data points.
- LWLR fits a separate linear regression model for each query point based on the weights assigned to the training data points.
- The weights assigned to each training data point are inversely proportional to their distance from the query point.
- Training data points that are closer to the query point will have a higher weight and contribute more to the linear regression model.
- LWLR is useful when a global linear model does not well-capture the relationship between the input and output variables. The goal is to capture local patterns in the data.

- **Assigning weights to a line- known as Kernel smoothing**
- A weight function which is given as:
- The bandwidth parameter controls how quickly the weight should fall with the distance of the point with the training point. In simpler terms, it controls the width of how varied the data is.



$$W^{(i)} = \exp\left(- \frac{(x^{(i)} - x)^2}{2\tau^2}\right)$$

$x^{(i)}$ = any general training point,

x = given query point and,

τ = is bandwidth parameter

- **Advantages of Locally Weighted Linear Regression**
- It is a simple algorithm.
- It can give excellent results when we have non-linear data points, and features are less.
- **Disadvantages:**
- Need to evaluate whole dataset every time a new data comes.
- Computation cost is high.
- Memory requirement is more.

Radial Basis Function

- **Radial Basis Kernel** is a kernel function that is used in machine learning to find a non-linear classifier or regression line.
- **Kernel Function:**
- **Kernel Function is used to transform n-dimensional input to m-dimensional input, where m is much higher than n then find the dot product in higher dimensional efficiently**.
- The main idea to use kernel is: A linear classifier or regression curve in higher dimensions becomes a Non-linear classifier or regression curve in lower dimensions.

Case Based Reasoning (CBR) Classifier

- **Case-Based Reasoning classifiers (CBR)** use a database of problem solutions to solve new problems. It stores the tuples or cases for problem-solving as complex symbolic descriptions.
- When a new case arises to classify, a Case-based Reasoner(CBR) will first check if an identical training case exists. If one is found, then the accompanying solution to that case is returned.
- If no identical case is found, then the CBR will search for training cases having components that are similar to those of the new case. Conceptually, these training cases may be considered as neighbors of the new case. The CBR tries to combine the solutions of the neighboring training cases to propose a solution for the new case.

- **Applications of CBR includes:**
- Problem resolution for customer service help desks, where cases describe product-related diagnostic problems.
- It is also applied to areas such as engineering and law, where cases are either technical designs or legal rulings, respectively.
- Medical educations, where patient case histories and treatments are used to help diagnose and treat new patients.
- **Challenges with CBR**
- Finding a good similarity metric (eg for matching subgraphs) and suitable methods for combining solutions.
- Selecting salient features for indexing training cases and the development of efficient indexing techniques.