Hossam Ahmed Salah

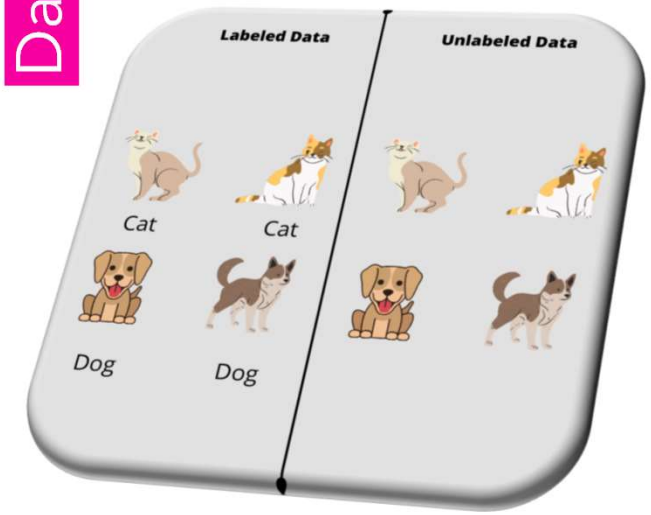
K-Nearest Neighbors(KNN)

supervised learner

Machine Learning Tasks

Data can be !

- Labeled (Supervised)
- Not labeled (Unsupervised)



ML task

Supervised
- Classification
- Regression

Unsupervised
- Clustering
- Segmentation
- Dimension reduction

Reinforcement
- Decision process
- Reward system
- Recommendation system

Machine Learning Tasks

Data can be !

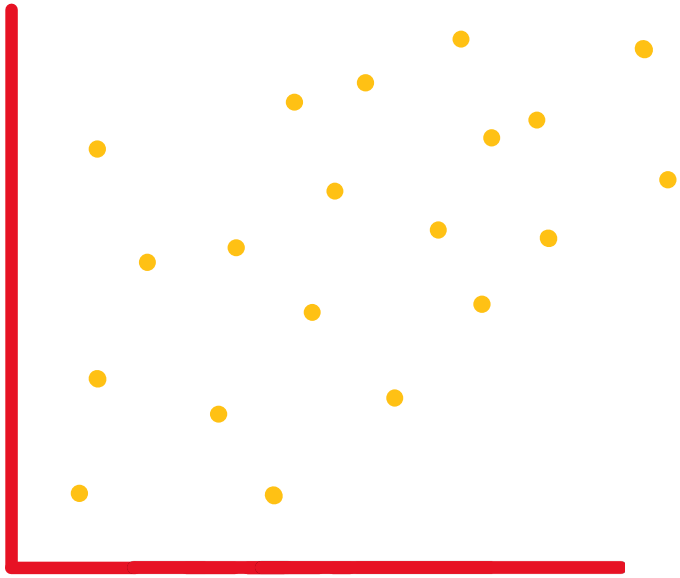
- Labeled (Supervised)
- Not labeled (Unsupervised)

Supervised

- Classification (sunny, cloudy,..)
- Regression (33°)

Variance

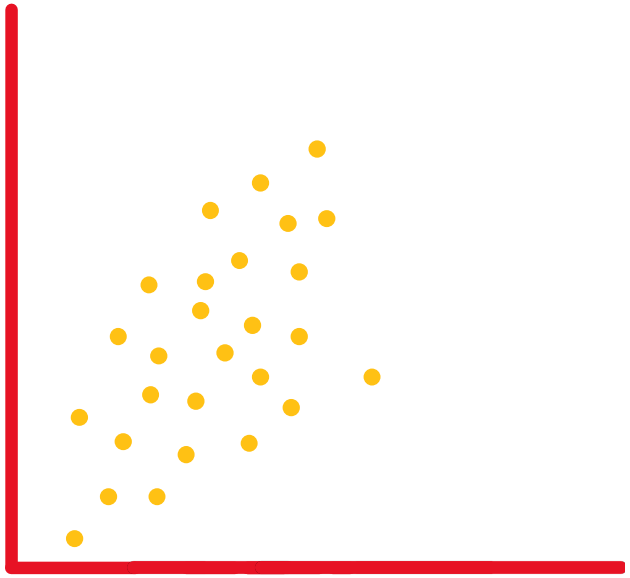
- Measurement of spread in the dataset
- مقياس لتباعد النقاط المختلفة من النقطة الوسط



High Variance

Variance

- Measurement of spread in the dataset
- مقياس لتباعد النقاط المختلفة من النقطة الوسط



Low Variance

Bias Error

In statistics, bias refers to the tendency of a statistical estimator or method to consistently overestimate or underestimate a population parameter

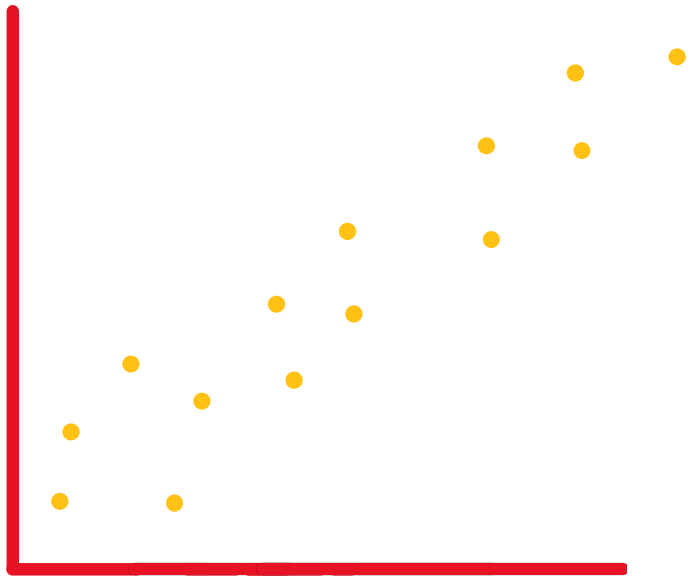
tendency which causes differences between results and facts

Bias in ML is a sort of mistake in which some aspects of a dataset are given more weight and/or representation than others

In ML bias inability to capture the underlying complexity of the data.

Overfitting and Underfitting

- Considering we have this data, and we want to train a 2 models



Overfitting and Underfitting

Models

- Linear

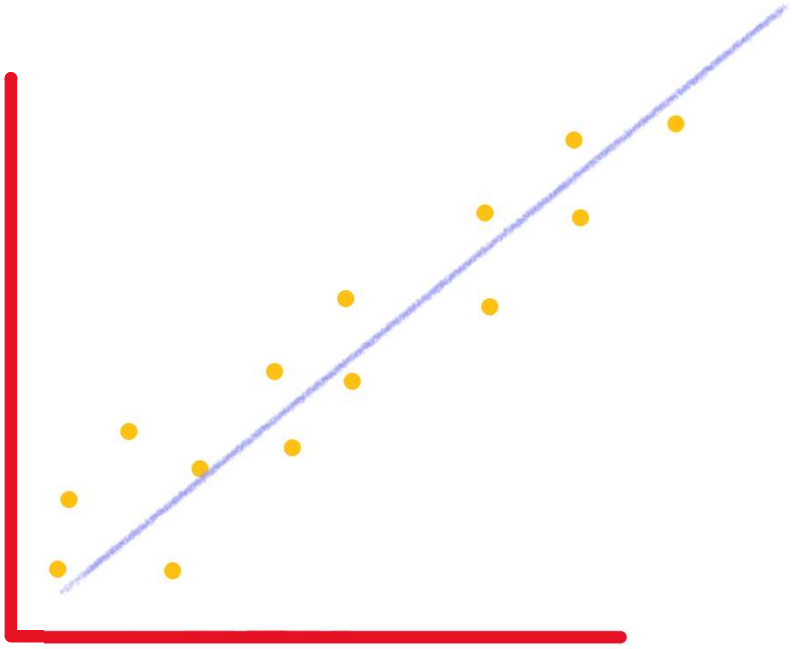
$$y = \beta_0 + \beta_1 x$$

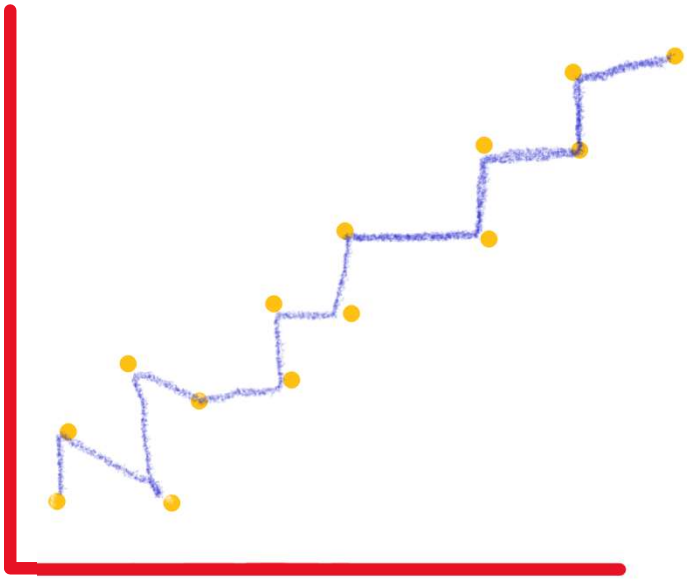
- Polynomial

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n$$

Overfitting and Underfitting

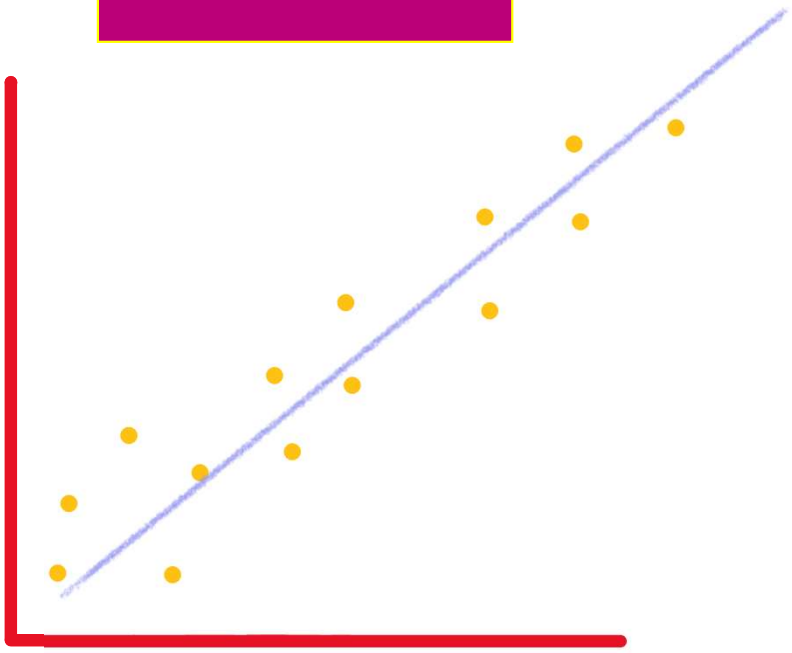
- Considering we have this data, and we want to train a 2 models



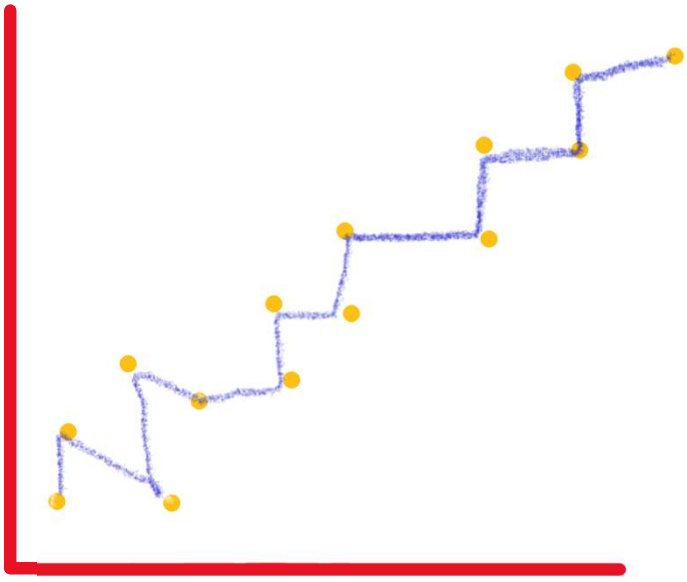


Overfitting and Underfitting

Only fit few points

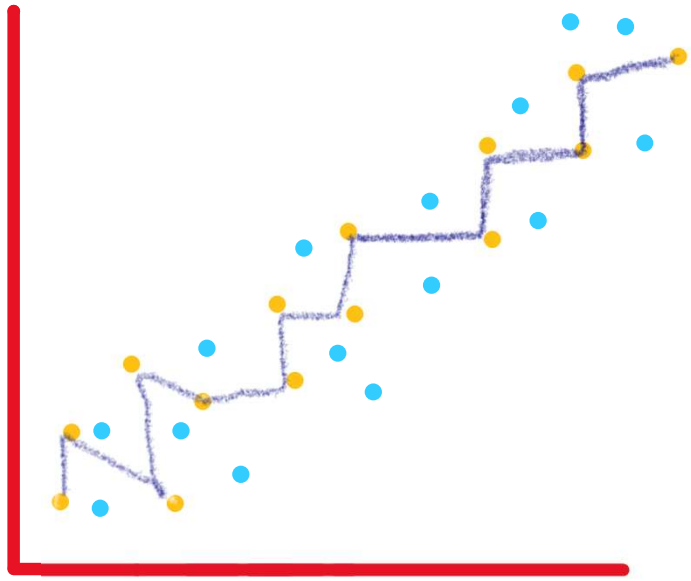


Perfect fit of the trained data 100%



Overfitting and Underfitting

- Unseen data in the training



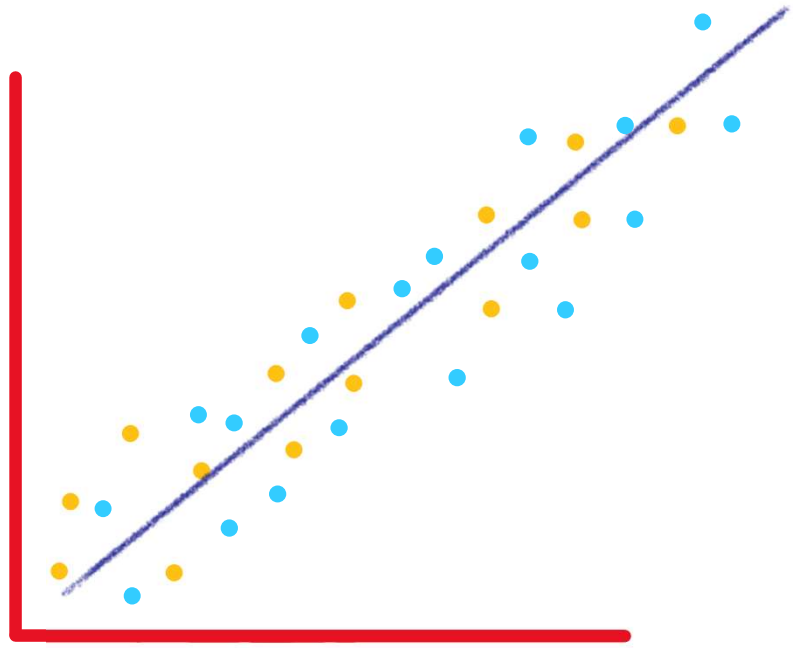
Your model is terrible on real data not near even

Seeking perfection in train set lead to **high variance** model

This called **overfitting**

Overfitting and Underfitting

- What about the linear Model



- Unseen data in the training

Terrible model but we the training error and testing error is not too far

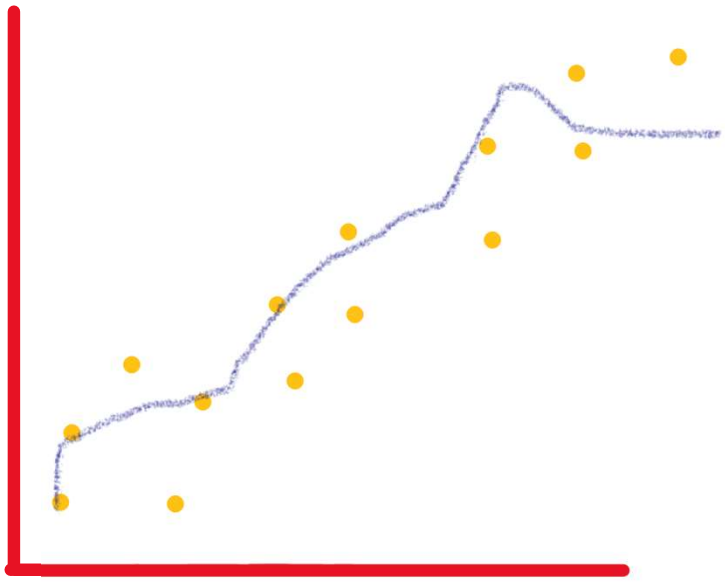
If average error is 20% for each point you can say that new points can be + or – 20%

This model has a **high bias** following the line doesn't care about the data

This model **underfit** the data too simple to model it

Overfitting and Underfitting

- We need a mode that generalize
  - we shall tolerant some bias (error) in the training set
  - That mean making this model has less variance
  - It's to simplify the model

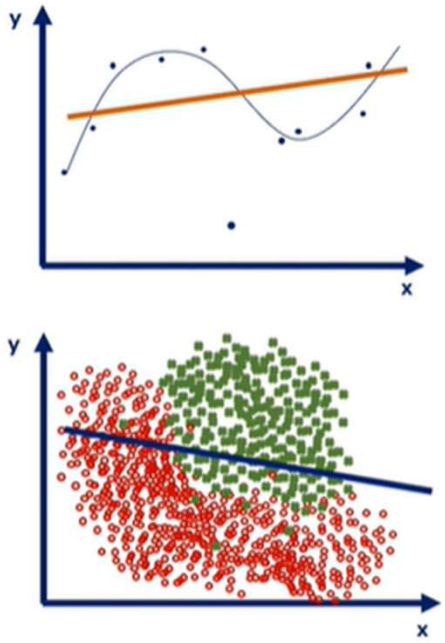


We have errors in the training set
But we can have less error when dealing with new input

A Best fit trying to achieve balance between Bias(error, simplicity) and Variance (complexity)

It's a tradeoff

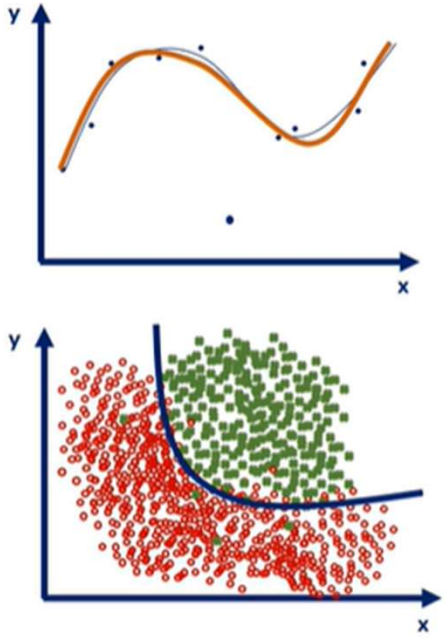
An **underfitted** model



Doesn't capture any logic

- High loss
- Low accuracy

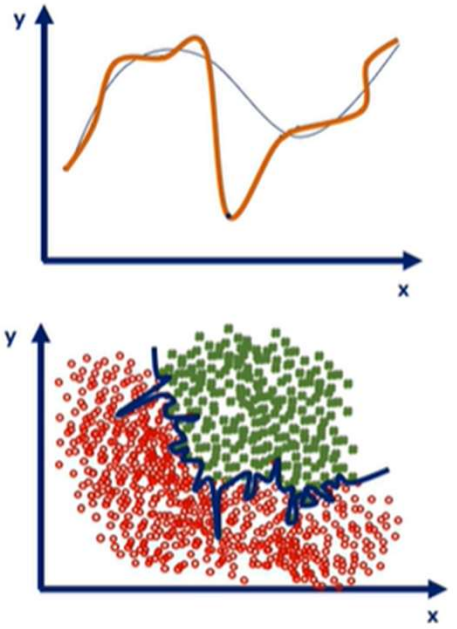
A **good** model



Captures the underlying logic of the dataset

- Low loss
- High accuracy

An **overfitted** model

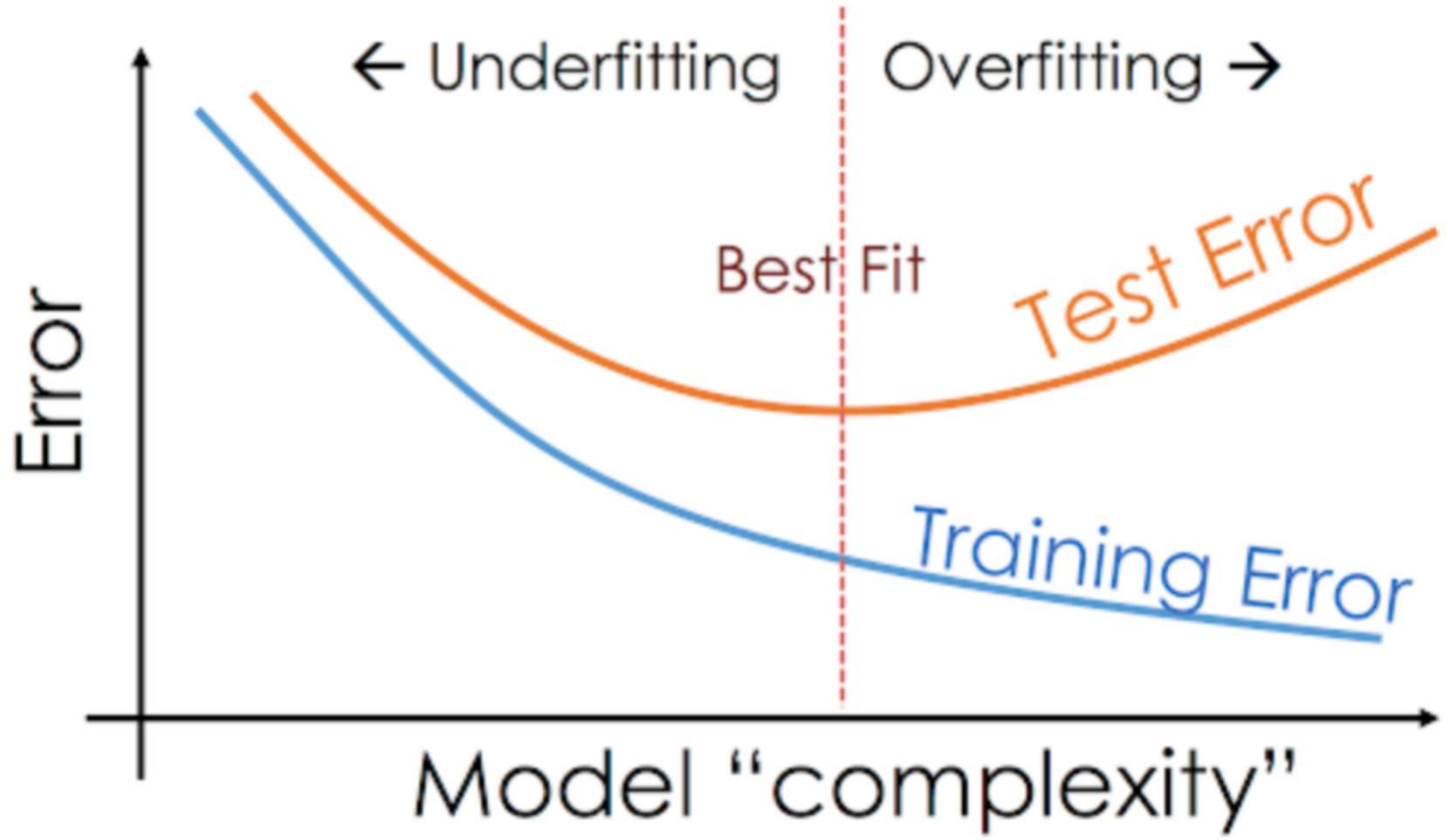


Captures all the noise, thus "missed the point"

- Low loss
- Low accuracy

Bias-variance tradeoff: The balance between underfitting and overfitting

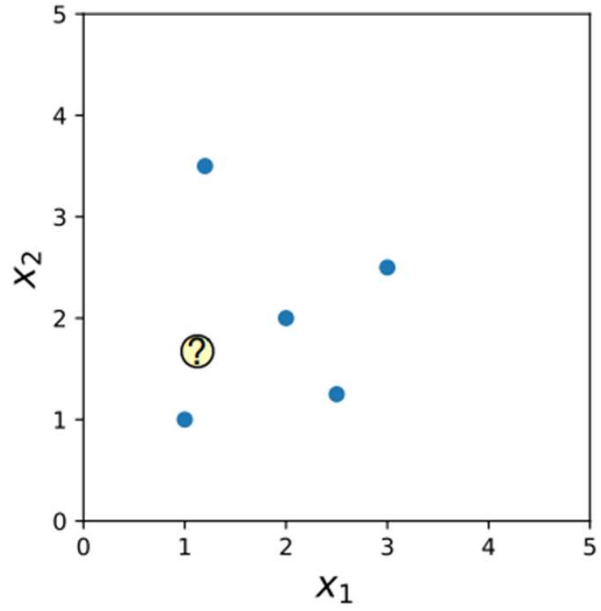
365 DataScience


← Underfitting
Overfitting →
Best Fit
Test Error
Training Error
Error
Model "complexity"

K-Nearest Neighbor (K-NN)

How KNN works ?

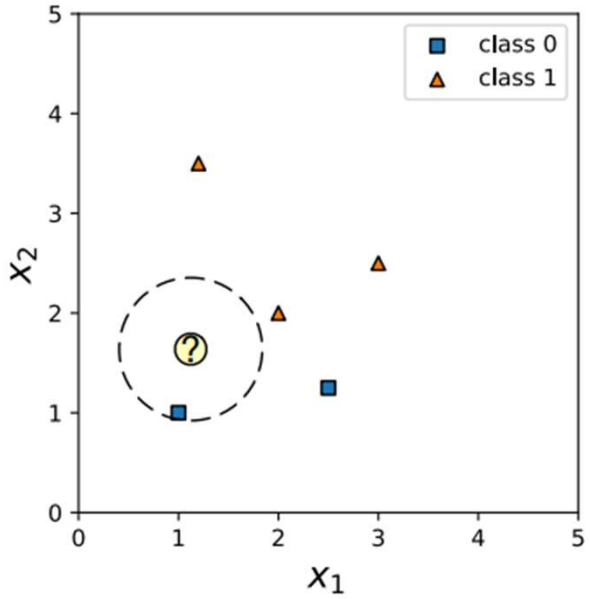
- **How to train 1-NN estimator ?(*Training*)**
  - Memorize the training data



K-Nearest Neighbor (K-NN)

How KNN works ?

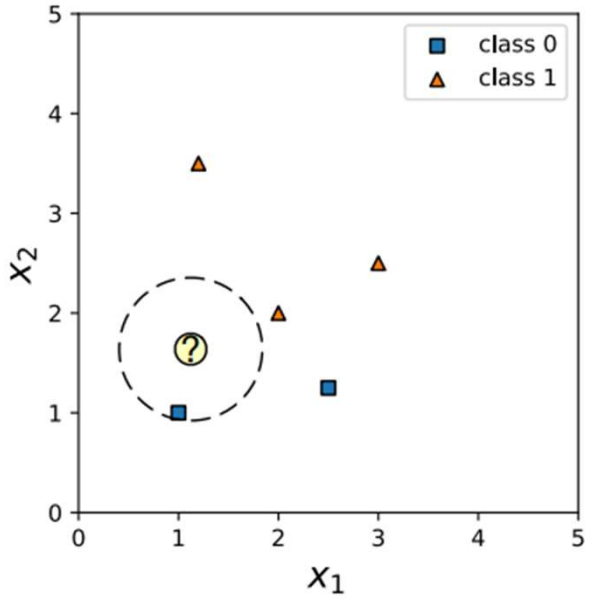
- **How 1-NN make predictions (*Predicting*)?**
  - If you want to predict x, find the nearest point in the training to the unseen x point
  - The nearest training point to our unseen point would be our prediction



K-Nearest Neighbor (K-NN)

How KNN works ?

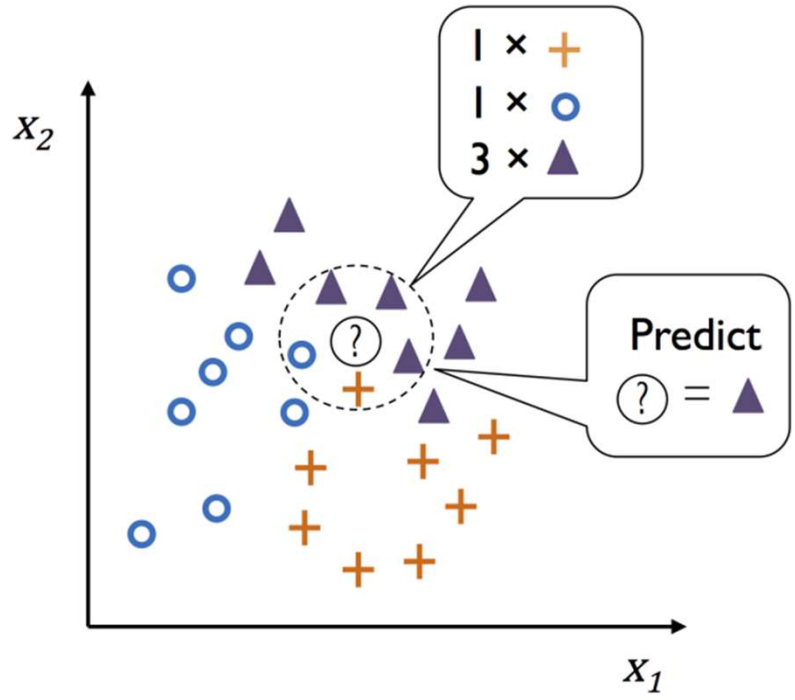
- **How 1-NN make predictions (*Predicting*)?**
  - If you want to predict x, find the nearest point in the training to the unseen x point
  - The nearest training point to our unseen point would be our prediction



K-Nearest Neighbor (K-NN)

How KNN works ?

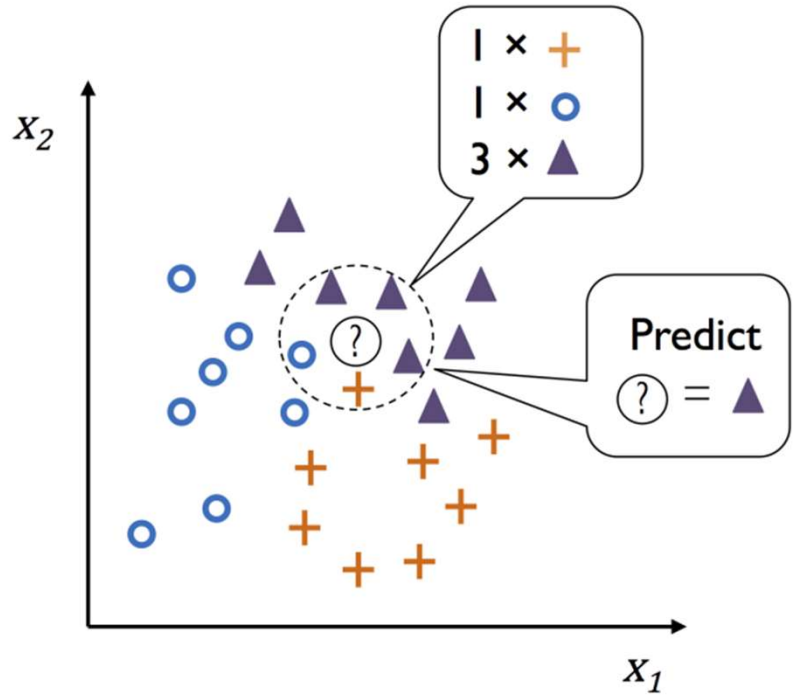
- **How k-NN make predictions (*Predicting*)?**
- **Query point (unseen point ?)**
- **K= 5 (asking 5 training points to vote)**
- **We have 3 classes**
- **See the k-nearest points from the unseen point and label it based on similarity of most points around it.**



K-Nearest Neighbor (K-NN)

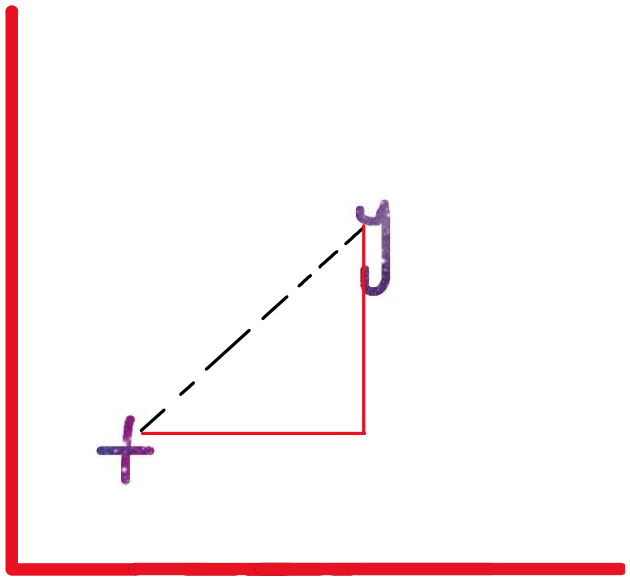
How KNN works ?

- **How k-NN make predictions (*Predicting*)?**
- **Query point (unseen point ?)**
- **K= 5 (asking 5 training points to vote)**
- **We have 3 classes**
- **See the k-nearest points from the unseen point and label it based on similarity of most points around it.**



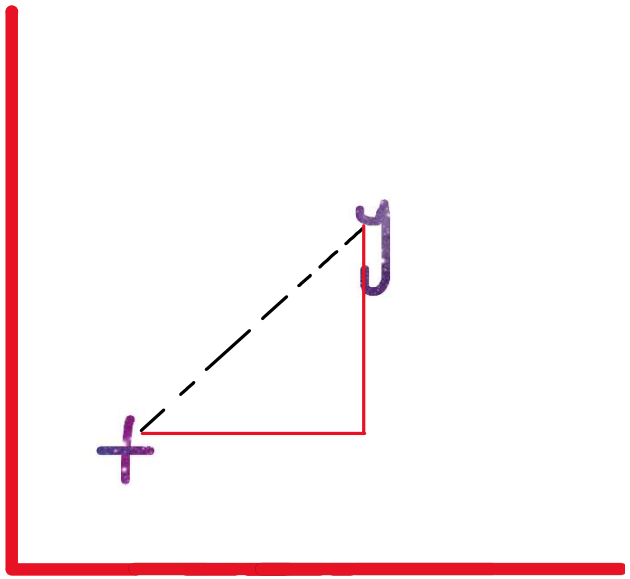
How to know the nearest point?

- Of course, using **distance**
- **How to calculate the distance?**



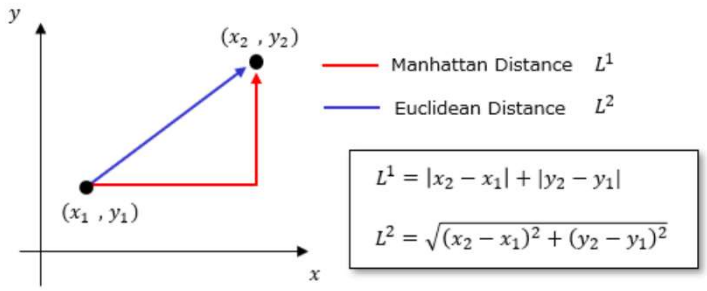
How to know the nearest point?

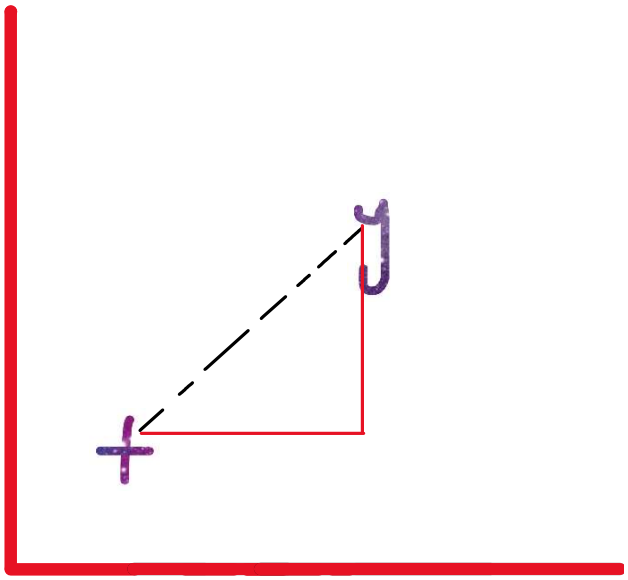
- Of course, using **distance**
- **How to calculate the distance?**
  - **Euclidian distance (L2)**



How to know the nearest point?

- Of course, using **distance**
- **How to calculate the distance?**
  - **Euclidian distance (L2)**
  - **Manhattan distance (L1)**

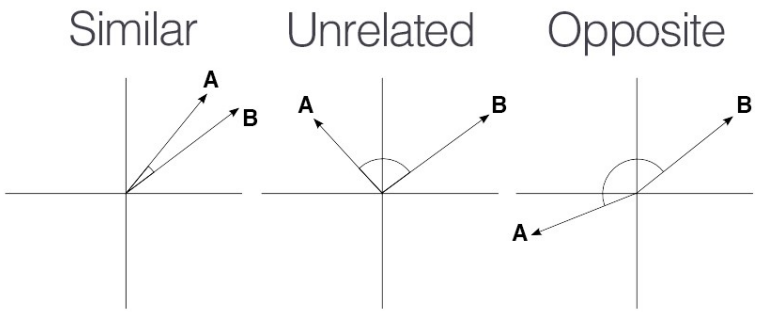


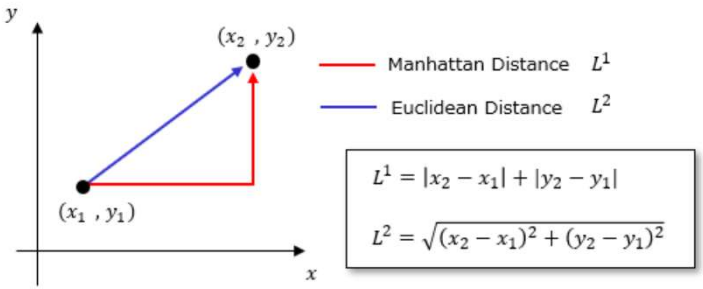


How to know the nearest point?

- Of course, using **distance**
- **How to calculate the distance?**
  - **Euclidian distance (L2)**
  - **Manhattan distance (L1)**
  - **Cosine similarity**

Distance Metric: A measure of how similar or dissimilar two data points are. Common distance metrics include Euclidean distance, Manhattan distance, and cosine similarity.





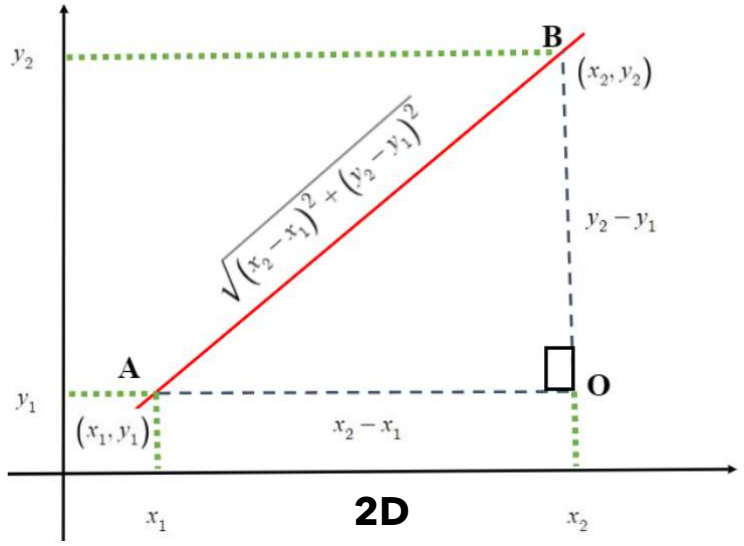
How to know the nearest point?

- Of course, using **distance**
- **How to calculate the distance?**
  - **Euclidian distance (L2)**

Distance Metric: A measure of how similar or dissimilar two data points are. Common distance metrics include Euclidean distance, Manhattan distance, and cosine similarity.

$$d(\mathbf{x}^{[a]}, \mathbf{x}^{[b]}) = \sqrt{\sum_{j=1}^{m} \left(x_j^{[a]} - x_j^{[b]}\right)^2},$$

vectors in m dimensions



How to know the nearest point?

- Other names of L2
- **How to calculate the distance?**
  - **Euclidian distance (L2)**

Sum of Squared Difference (SSD)

$$d_{\mathbf{SSD}} : (x, y) \mapsto \|x - y\|_2^2 = \langle x - y, x - y \rangle = \sum_{i=1}^{n} (x_i - y_i)^2$$

$$d(\mathbf{x}^{[a]}, \mathbf{x}^{[b]}) = \sqrt{\sum_{j=1}^{m} \left(x_j^{[a]} - x_j^{[b]} \right)^2},$$

Euclidean Distance

$$d_2 : (x, y) \mapsto \|x - y\|_2 = \sqrt{d_{\mathbf{SSD}}} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

vectors in m dimensions

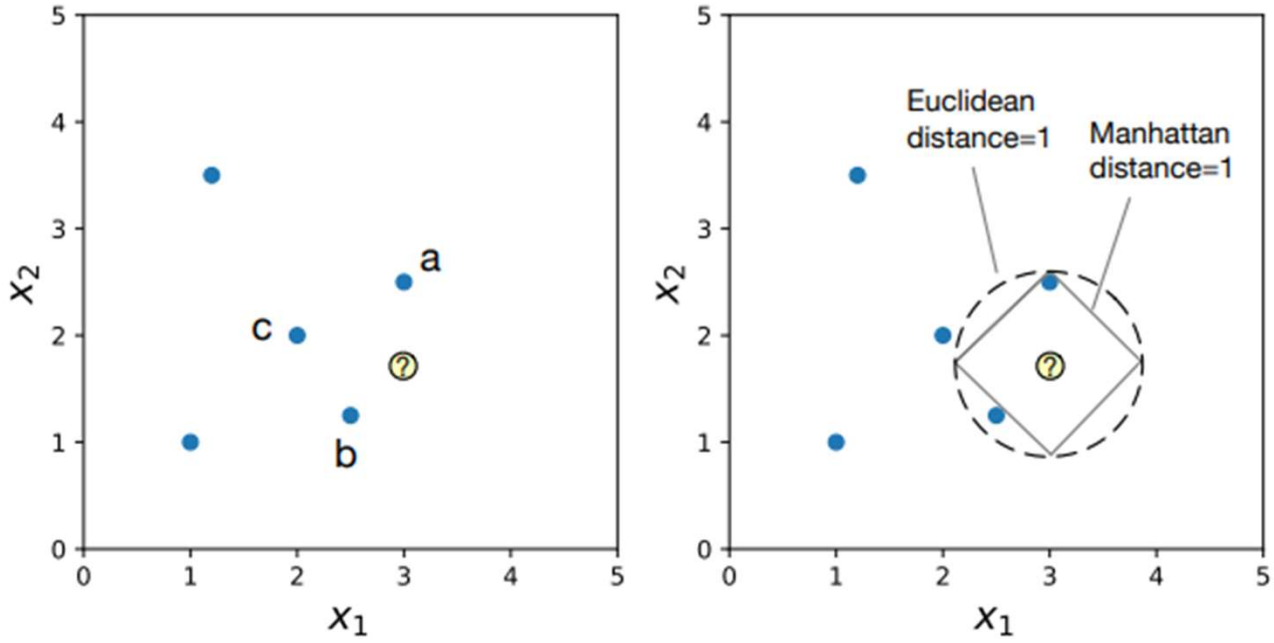
How to know the nearest point?

- Other names of L1
- **How to calculate the distance?**
  - **Manhattan distance (L1)**

Sum of Absolute Difference (SAD)

$$d_{\mathbf{SAD}} : (x, y) \mapsto \|x - y\|_1 = \sum_{i=1}^{n} |x_i - y_i|$$

How to know the nearest point?



Features of KNN

- **Non**parametric model
  - Linear Regression for example is a parametric model f(X-train; Weights)
  - KNN non-parametric as it the k-NN just memorize the train data f(X-train) in the training phase that is why it's also called a **Lazy learner**
- Instance-based (memory based)
  - NN stores the entire training dataset in memory. Each data point in the training dataset is an "instance," and this dataset is used for making predictions
  - It only memorize the training data no learning or generalization of the data.
  - Prediction on the Fly: When you want to make a prediction for a new data point, KNN looks for the k-nearest neighbors to the new point in the training dataset and assigns a class label based on the majority class among these neighbors (for classification) or computes an average value (for regression).

Advantages of K-NN

- **Easy to implement**: Given the algorithm's simplicity and accuracy, it is one of the first classifiers that a new data scientist will learn.
- **Adapts easily**: As new training samples are added, the algorithm adjusts to account for any new data since all training data is stored into memory.
- **Few hyperparameters**: KNN only requires a k value and a distance metric, which is low when compared to other machine learning algorithms.

Disadvantages of K-NN

- **Memory Usage: KNN is memory-intensive, especially with large datasets, as it needs to store the entire training dataset in memory.**
- **Curse of dimensionality: it doesn't perform well with high-dimensional data inputs.**
- **Prone to overfitting**
- **Sensitivity to Feature Scaling: KNN is sensitive to the scale of features. Features with larger ranges can dominate the distance calculations. Standardizing or normalizing features is often required.**

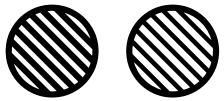
Disadvantages of K-NN

- **Computational Complexity: The main drawback of KNN is its computational intensity. It needs to compute distances between the query point and all training instances, which can be slow for large datasets**
- **Choosing the right value for "k" is critical. A small k can lead to noisy predictions, while a large k can lead to oversmoothed decisions. Selecting an optimal value for k can be challenging. (***odd k is preferred specially in binary classification to avoid ties in voting* **) we can use Elbow method also**

Disadvantages of K-NN

(*odd k is preferred specially in binary classification to avoid ties in voting* **) we can use Elbow method also**

$k = 2$

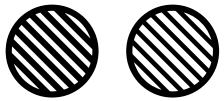


$k = 3$

Disadvantages of K-NN

(*odd k is preferred specially in binary classification to avoid ties in voting* **) we can use Elbow method also**

$k = 2$



$k = 3$

Tips to build a good KNN

- Choosing the value of k.
  - Odd k (classification)
  - Elbow method
- Scaling of the feature axes.
  - Scaling is important for distance and gradient based models
- Choice of distance measure.
  - Grid search

References

- https://numerics.mathdotnet.com/Distance
- https://en.wikipedia.org/wiki/Plurality_voting
- https://sebastianraschka.com/pdf/lecture-notes/stat479fs18/02_knn_notes.pdf
- https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
- https://github.com/hossamAhmedSalah/Machine_Learning_MSP [some reused slides and concepts]
- https://sebastianraschka.com/pdf/lecturenotes/stat479fs18/02_knn_notes.pdf
- https://pages.stat.wisc.edu/~sraschka/teaching/stat479-fs2018/
- https://github.com/rasbt/stat479-machine-learning-fs18/
- https://www.ibm.com/topics/knn