Machine learning Algorithm 8/8/21

K - Nearest Neighbor.

K-NN Theory.

- Easy to implement supervised machine learning algorithm that can be used to solve both classification & regression problem.

Breaking it down

- Supervised Machine learning Algorithm is one that depends on → labeled input data
  → to learn a function that produces an ~~appropiate~~ appropriate output when given new unlabeled data.

→ classification problem
  ↳ have discrete value as its output.
  ↳ Not a mathematical operation
  ↳ have predictor (a set of predictor) & a label.

→ A regression problem

↳ has a real number & its output.

↳ have one or more independent variable or a dependant variable.

→ An unsupervised Machine learning.

↳ make use of input data without any labels.

↳ Tries to learn the basic structure of the data to give us more insight into the data.

→ KNN Captures the idea of similarity (distance, proximity or closeness)

→ K-NN algorithm stores all the available data & classifies a new data point based on similarity.

→ New datas can easily be classified into well suite category

→ K-NN is a non-parametric algorithm, it does not make any assumption on underlying data.

→ It's called lazy learner algorithm
  ↳ It does not learn from training set, instead it stores the dataset & at a time of classification, it perform an action on dataset.

→ K-NN algorithm at the training phase.
  ↳ stores the dataset
  ↳ When it gets new data, it classfies that data into a category that is much similar to new data.

→ Types of distance Metrics.
  ↳ Minkowski Distance
  ↳ Manhattan Distance
  ↳ Euclidean Distance * Most popular.
  ↳ Cosine Distance
  ↳ Chebyshev Distance.

Distance Metrics Concepts

Minkowski Distance

- Intended for real-valued vector space.
- Calculate distance in a normed vector space.
  - ↳ Spaces where distances can be represented as a vector that has a length & the lengths cannot be negative.

Conditions

- ↳ Non-negativity : $d(x,y) >= 0$
- ↳ Identity : $d(x,y) = 0$ if & only if $x == y$.
- ↳ Symmetry : $d(x,y) = d(y,x)$
- ↳ Triangle Inequality : $d(x,y) + d(y,z) >= d(x,z)$

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

We manipulate this formula with different p values.

- $p=1$ ↳ Manhattan Distance
- $p=2$ ↳ Euclidean Distance.

Manhattan Distance

- Known As Taxicab / city block distance.
- Distance between two points is the sum of the absolute differences of their Cartesian coordinates.

from $\left(\sum_{i=1}^{n} |x_i - y_i|^1\right)^{1/p}$

$$d = \sum_{i=1}^{n} |x_i - y_i|$$

eg/ red (4, 4) & green (1, 1).

$d = |4-1| + |4-1| = 6$ ✱..

→ This distance is prefered in case of high dimensionality.

Euclidean Distance

- It measure of the true straight line distance between two points of Euclidean Space.

from eq: $\left(\sum_{i=1}^{n} |x_i - y_i|^2\right)^{1/2}$

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

if red $(4,4)$ & green $(1,1)$
$x_1\ y_1$ $x_2\ y_2$

$$d(x,y) = \sqrt{(4-1)^2 + (4-1)^2}$$

$$= 4.24 \#$$

Cosine distance

- Calculate similarity between two vectors
- Measure by the cosine of the angle between two vectors pointing in the same direction.

- This distance gives us a new perspective to a business problem to find some hidden information.

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|}$$

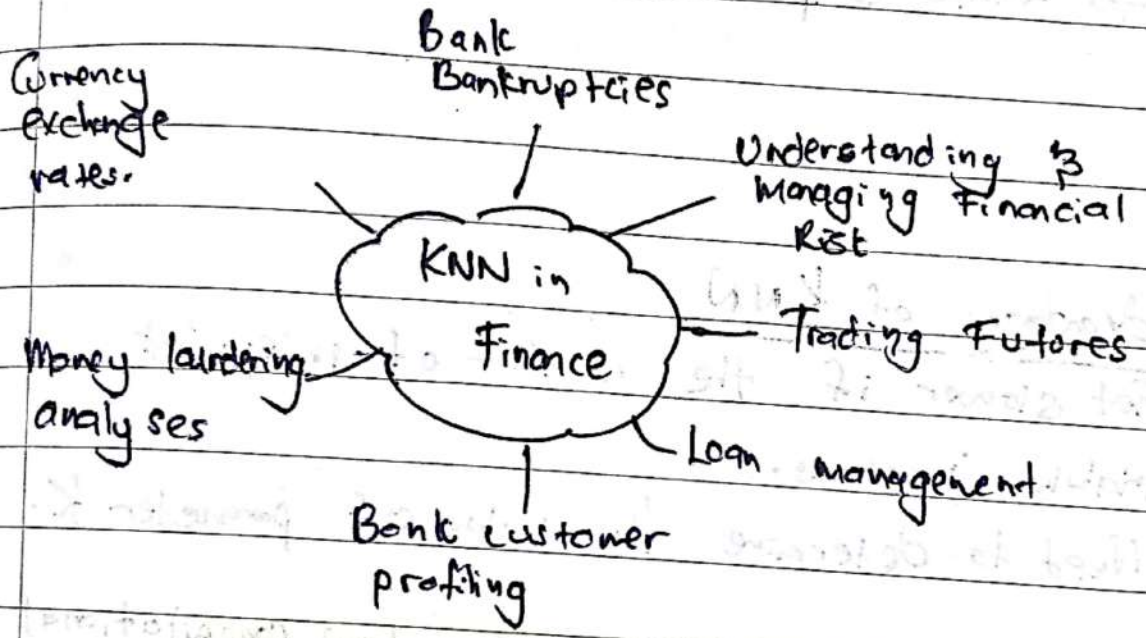
We will get 2 values $(0 \text{ to } 1)$, 0 = 100% similar
1 = 0% similar

The KNN Algorithm

1. Load the data.
2. Initialize K to your chosen number of neighbours.
3. For each example in the data.
   - 3.1 Calculate the distance between the query example to an ordered collection.
   - 3.2 Add the distance & the index of the example to an ordered collection.
4. Sort the ordered collection of distances & indices from smallest to largest (in ascending order) by distances.
5. Pick the first K entries from the sorted collection.
6. Get the labels of the selected K entries.
7. If regression, return the mean of K labels.
8. If classification, return the mode of the K labels.

Application of KNN

- Climate forecasting
- Estamating soil water parameter.
- Stock market forecasting.

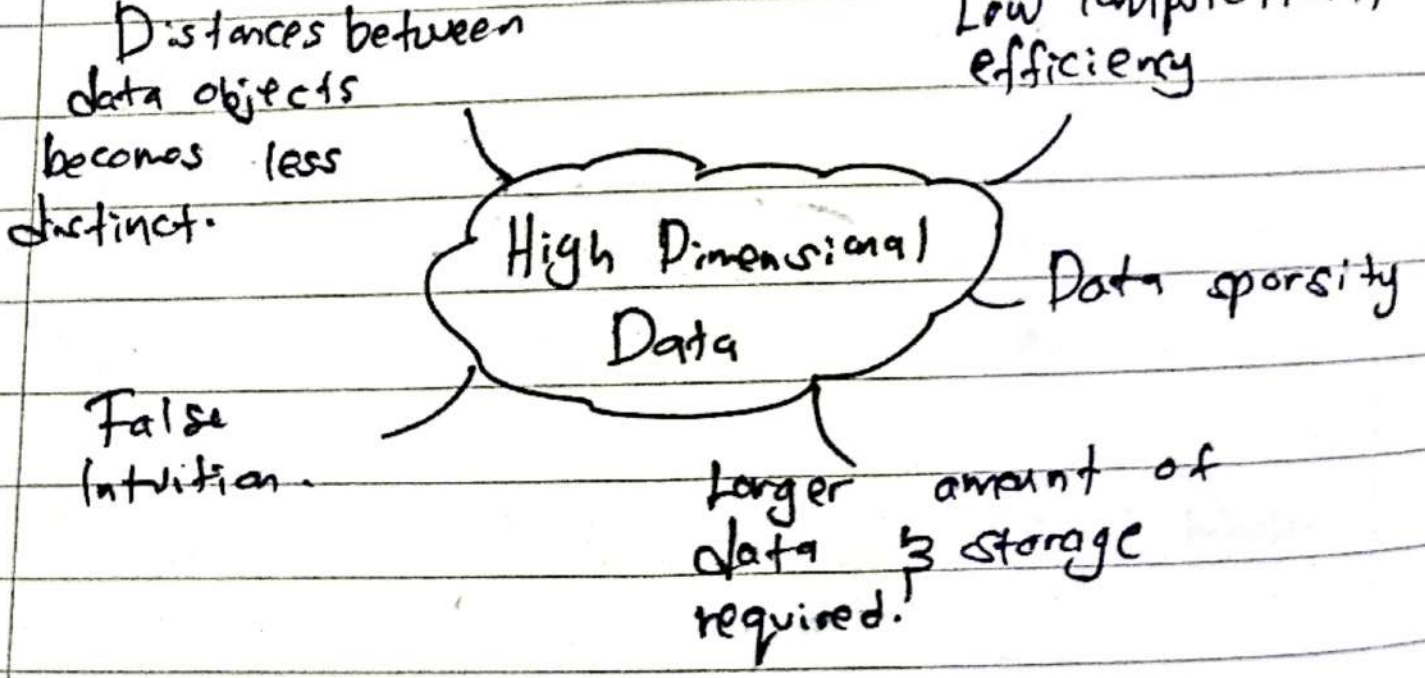


Advantages of KNN

→ Robust to noisy training data
→ Effective if training data is large.
→ No training phase.
→ Learns complex models easily

Disadvantages of KNN

- Get slower if the number of independent variables increases.
- Need to determine the value of parameter K.



Choosing the right value of K

- Change the value of K & run KNN algorithm several times to achieve :-
  - K that reduces the number of errors
  - Maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before.

* → As we ~~increase~~ decrease the value of K to 1, our ~~our~~ predictions become less stable.

→ If we increase the value of K, our prediction becomes more stable. * Famous value, K = 5

→ At one point the number of errors will increase, this ~~shows~~ shows the limit of ~~incre~~ K increase.

→ We usually make K = Odd number
To make a tie breaker

Standardizing the variable

- Using sklearn.preprocessing.standardscaler()
- fit_transform() = Training data.
- transform() = Test data.

~ Data standardization is the process of rescaling the attributes so that they have mean as 0 & variance as 1.

~ Formula :-

$$x_{scaled} = \frac{x - mean}{sd}$$

* - fit_transform() - Scale the training data & learn the ~~de~~scaling parameter of the data.

- Fit method is calculating the mean & variance of each features present in our data.

- Transform method is transforming all the features using the respective mean & variance.

transform( ) - We can use the same mean & variance as it is calculated from the training data.

Why we doesn't use fit in our test data ?

~ It will compute a new mean & variance that is new scale for each feature.

~ Will ~~learn~~ let our model learn about our test data too.