Online Student Training for "Artificial Intelligence & Machine Learning"

(4th Feb, 2021 – 17th Mar, 2021)

KNN Classifier

Faculty Trainer

Naw Varsha Pipada

Department of Computer Science & Engineering

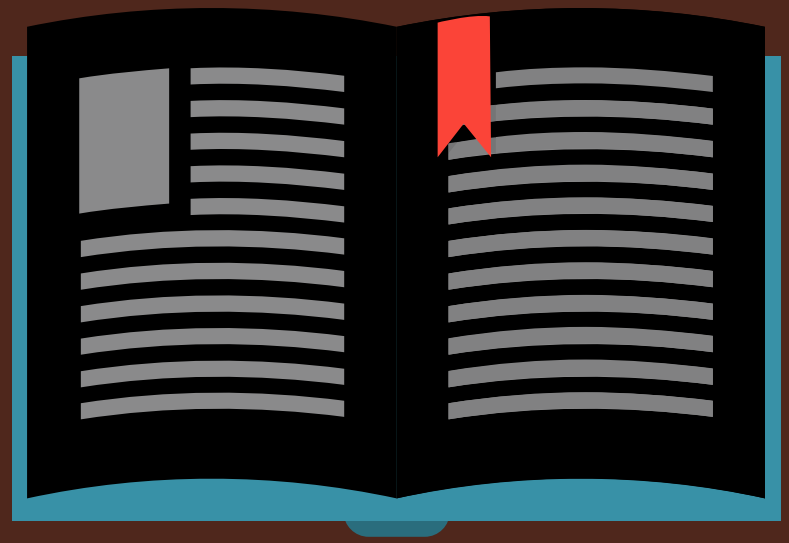
Engineering College Bikaner

Contents

Introduction
Introduction to KNN

How it Works
Basic steps for KNN working

Working Example
An example to understand the concept



Points to Think of
Some issues related to KNN

Pros of KNN
To understand the advantages of KNN

Cons of KNN
To understand the limitations of KNN

KNN Classification

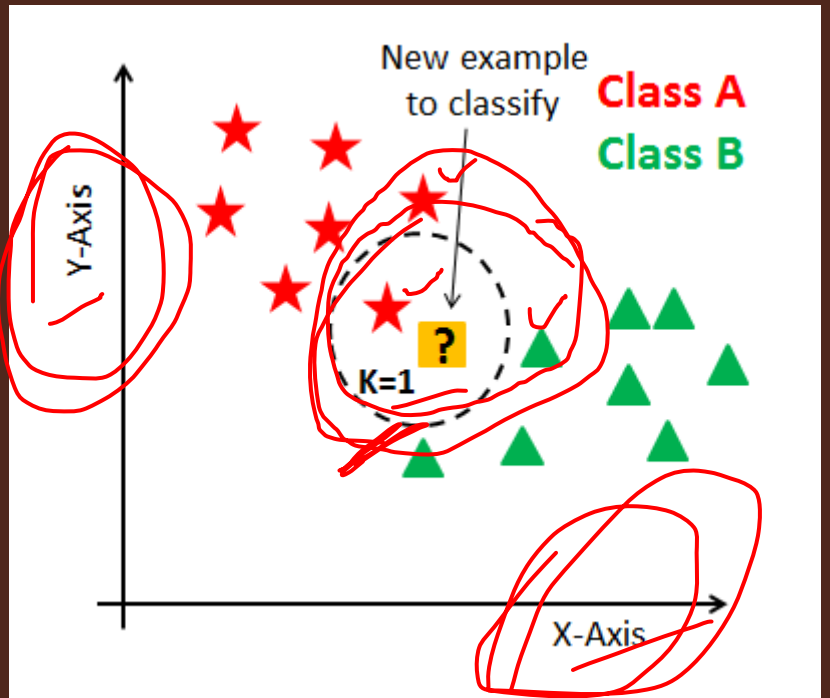
- K Nearest Neighbor(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms.
- KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition.
- KNN algorithm used for both classification and regression problems

KNN Classification

- KNN is a non-parametric and lazy learning algorithm.
- Non-parametric
  - means there is no assumption for underlying data distribution.
  - the model structure determined from the dataset.
  - This will be very helpful in practice where most of the real world datasets do not follow mathematical theoretical assumptions.
- Lazy algorithm
  - means it does not need any training data points for model generation.
  - All training data used in the testing phase.
  - This makes training faster and testing phase slower and costlier.

How it works

- K is the number of nearest neighbors.
  - The number of neighbors is the core deciding factor.
  - K is generally an odd number if the number of classes is 2.
  - When K=1, then the algorithm is known as the nearest neighbor algorithm.

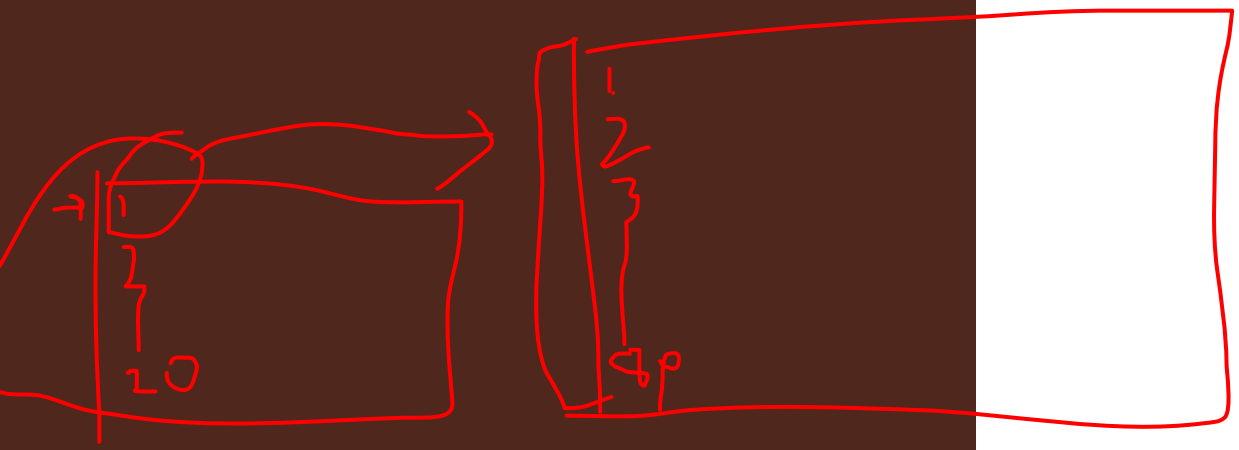


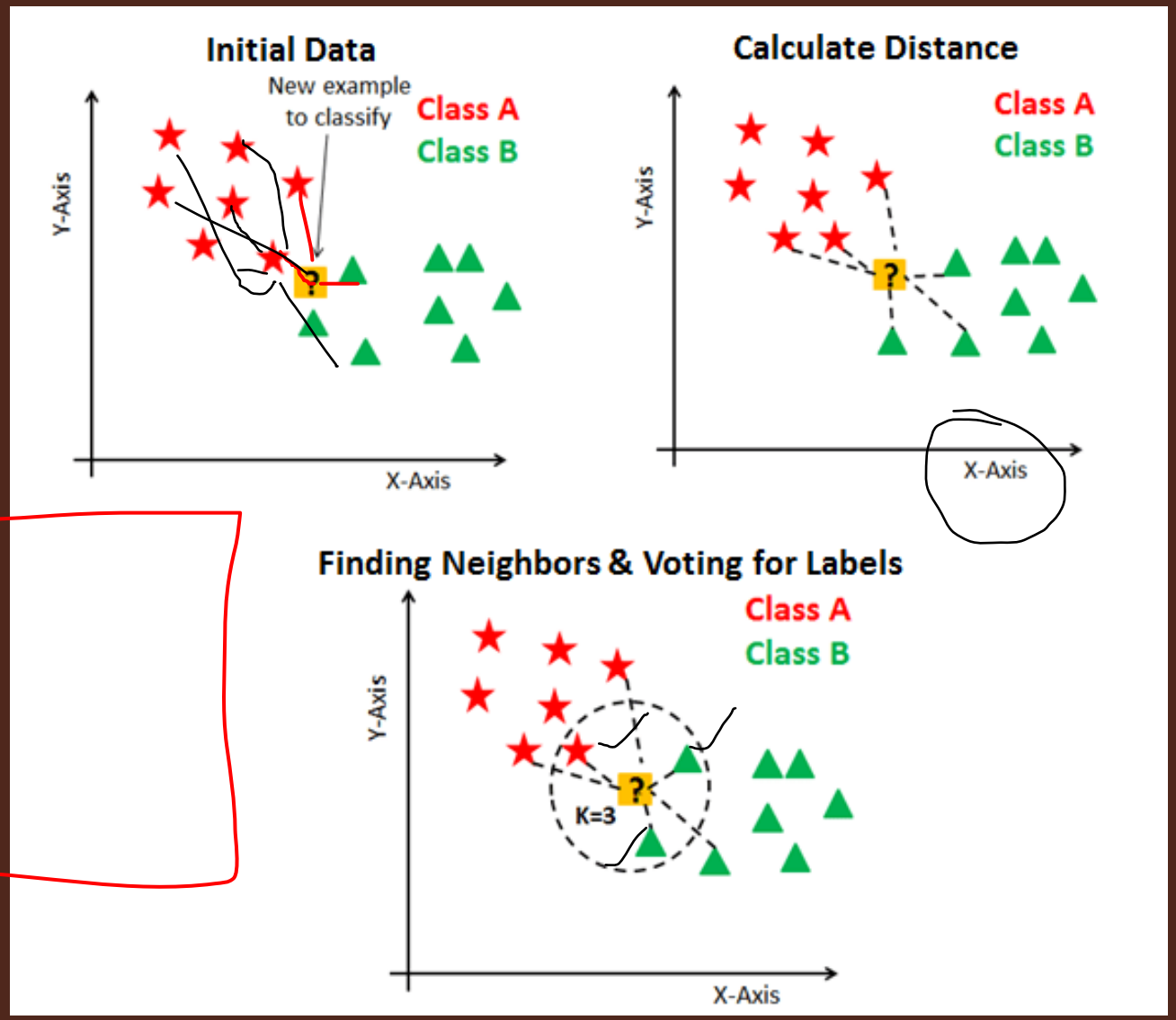
How it works

- Suppose P1 is the point, for which label needs to predict.
- First, you find the k closest point to P1 and then classify points by majority vote of its k neighbors.
- Each object votes for their class and the class with the most votes is taken as the prediction.

Basic Steps

- Calculate distance
- Find closest neighbors
- Vote for labels





Distance Calculation

- Euclidean distance
- Manhattan distance
- Minkowski distance

Euclidean

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan

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

Minkowski

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

- Hamming distance
  - Generally, used for categorical features

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Basic KNN Pseudocode

1. Load the data
2. Initialize the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
   a. Calculate the distance between test data and each row of training data.
   b. Sort the calculated distances in ascending order based on distance values
   c. Get the labels(target) of the top K entries
   d. If regression, return the mean of the K labels
   e. If classification, return the mode of the K labels

Working Example

S.No	X1	X2	Y
1	7	7	Bad
2	7	4	Bad
3	3	4	Good
4	1	4	Good

1. Calculate Distance

S.No	X1	X2	Squared Euclidean Distance
1	7	7	$(7-3)^2+(7-7)^2=16$
2	7	4	$(7-3)^2+(4-7)^2=25$
3	3	4	$(3-3)^2+(4-7)^2=9$
4	1	4	$(1-3)^2+(4-7)^2=13$

2. Rank acc. to distance

S.No	X1	X2	Y
3	3	4	Good
4	1	4	Good
1	7	7	Bad
2	7	4	Bad

3. Return mode of K labels

Testing Instance=>
(X1,X2)=(3,7)

Let K=3

Out of top K rows, i.e., 3 rows, the majority class is 'Good'
Hence the test instance is classified as 'Good'

Points to think about

How to choose value of K

Limitations of KNN

Pros of KNN

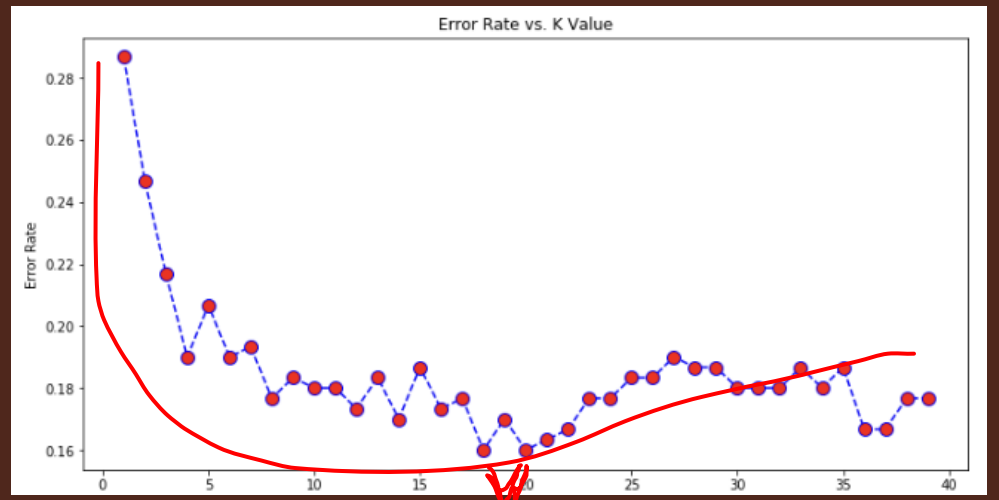
To find solution to limitations

How to choose K: The hyperparameter

- According to research, no optimal number of neighbors suits all kind of data sets. Each dataset has it's own requirements.
- Large k:
  - less sensitive to noise (particularly class noise)
  - better probability estimates for discrete classes
  - larger training sets allow larger values of k
- Small k:
  - captures fine structure of problem space better
  - may be necessary with small training sets
- Balance must be struck between large and small k
- Generally, an odd value of K is chosen for even number of classes.
- You can also check by generating the model on different values of k and check their performance

How to choose K : Elbow Method

- Run KNN classifier on the dataset for a range of values of *k* (say, *k* from 1 to 40)
- For each value of *k* calculate the error rate.
- Then, plot a line chart of the error rate for each value of *k*.
- If the line chart looks like an arm, then the "elbow" on the arm is the value of *k* that is the best or the value of k with least error rate.



KNN algorithm : Pros

- Quick calculation time – works good with small feature set
- Simple algorithm – to interpret
- Versatile – useful for regression and classification
- High accuracy – you do not need to compare with better-supervised learning models

KNN algorithm : Cons

- Accuracy depends on the quality of the data
- With large data, the prediction stage might be slow
- Sensitive to the scale of the data and irrelevant features
- Require high memory – need to store all of the training data
- Given that it stores all of the training, it can be computationally expensive

KNN algorithm : Cons (Sensitive to Scale of data)

- In the given dataset, income will have a much higher influence on the distance calculated
- Methods to improve: Normalization and Min-max scaling

Age	Loan	Default	Distance	
25	$40,000	N	102000	
35	$60,000	N	82000	
45	$80,000	N	62000	
20	$20,000	N	122000	
35	$120,000	N	22000	2
52	$18,000	N	124000	
23	$95,000	Y	47000	
40	$62,000	Y	80000	
60	$100,000	Y	42000	3
48	$220,000	Y	78000	
33	$150,000	Y	8000	1
48	$142,000	?		

Euclidean Distance

$$D = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

Use Min-Max Scaling

Age	Loan	Default	Distance
0.125	0.11	N	0.7652
0.375	0.21	N	0.5200
0.625	0.31	N	0.3160
0	0.01	N	0.9245
0.375	0.50	N	0.3428
0.8	0.00	N	0.6220
0.075	0.38	Y	0.6669
0.5	0.22	Y	0.4437
1	0.41	Y	0.3650
0.7	1.00	Y	0.3861
0.325	0.65	Y	0.3771
0.7	0.61	?	

Standardized Variable

$$X_s = \frac{X - Min}{Max - Min}$$

KNN algorithm : Cons (Irrelevant features or noise)

- Similar to scaling problem, there might be the influence of irrelevant features.
- Feature selection techniques are used before applying KNN.
- Can use weighted distance,
  - large weights => attribute is more important
  - small weights => attribute is less important
  - zero weights => attribute doesn't matter

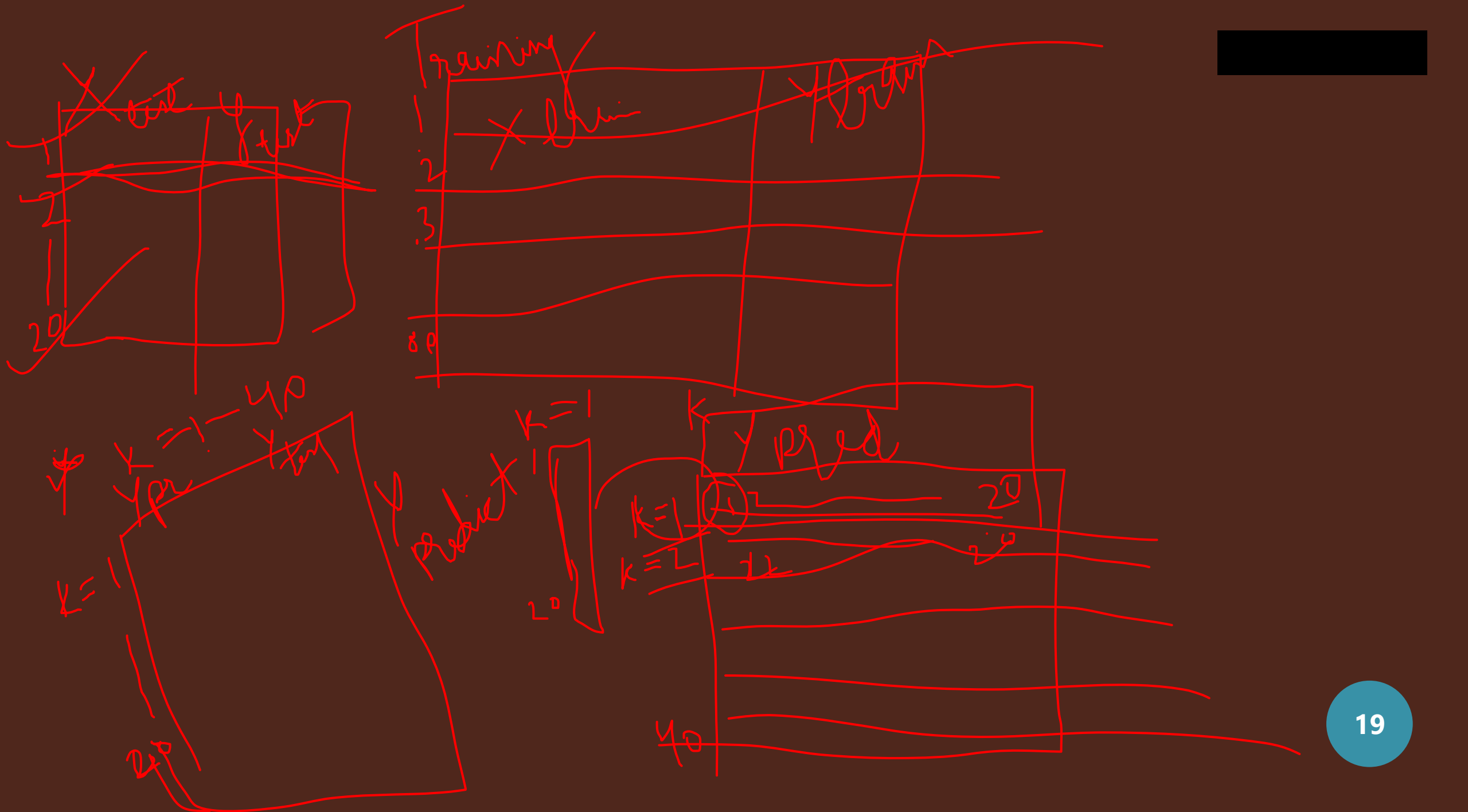
$$D(c1,c2) = \sqrt{\sum_{i=1}^{N} w_i \left(attr_i(c1) - attr_i(c2)\right)^2}$$

KNN algorithm : Cons (Curse of Dimensionality)

- As number of dimensions increases, distance between points becomes larger and more uniform

$$D(c1,c2)=\sqrt{\sum_{i=1}^{relevant}\left(attr_i(c1)-attr_i(c2)\right)^2+\sum_{j=1}^{irrelevant}\left(attr_j(c1)-attr_j(c2)\right)^2}$$

- if number of relevant attributes is fixed, increasing the number of less relevant attributes may swamp distance
- when more irrelevant than relevant dimensions, distance becomes less reliable
- Possible Solutions
  - Feature Extraction
  - PCA to reduce dimensions
  - Large K
  - Weighted KNN – using Kernel Functions
  - More complex distance function


Training
1
2
3
80
20
K=1
40
20
K=1
K=2
20
40

Thank You For Your Time