# CLASSIFICATION

Dr. Srikanth Allamsetty

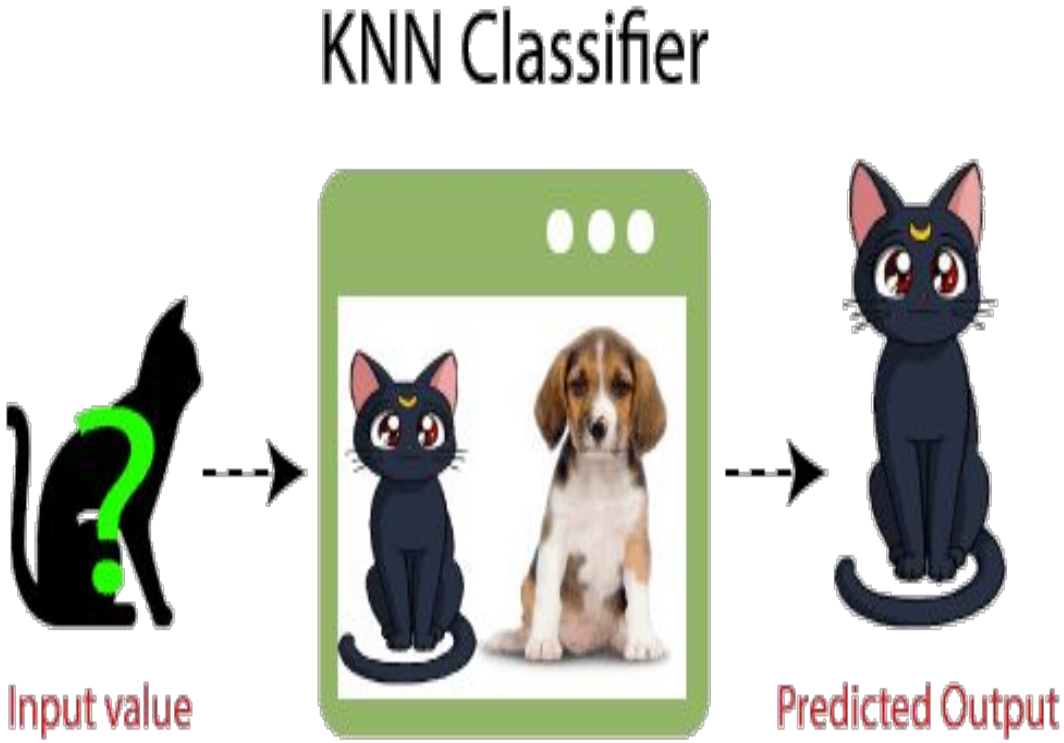
# Nearest Neighbor Learning

# Basics

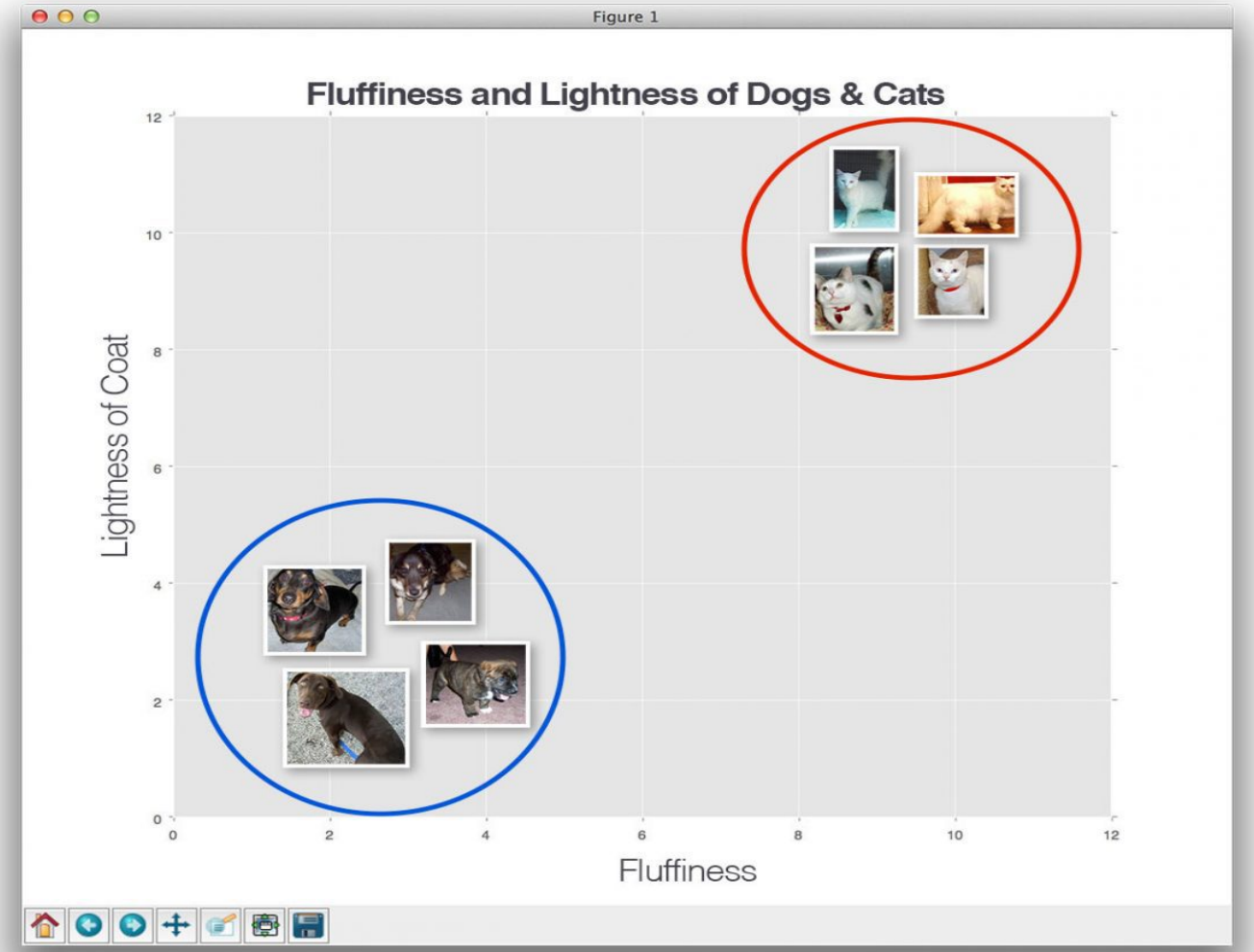
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- It is also 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.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- For any given problem, a small value of k will lead to a large variance in predictions; Alternatively, setting k to a large value may lead to a large model bias.

# Using it for pattern recognition

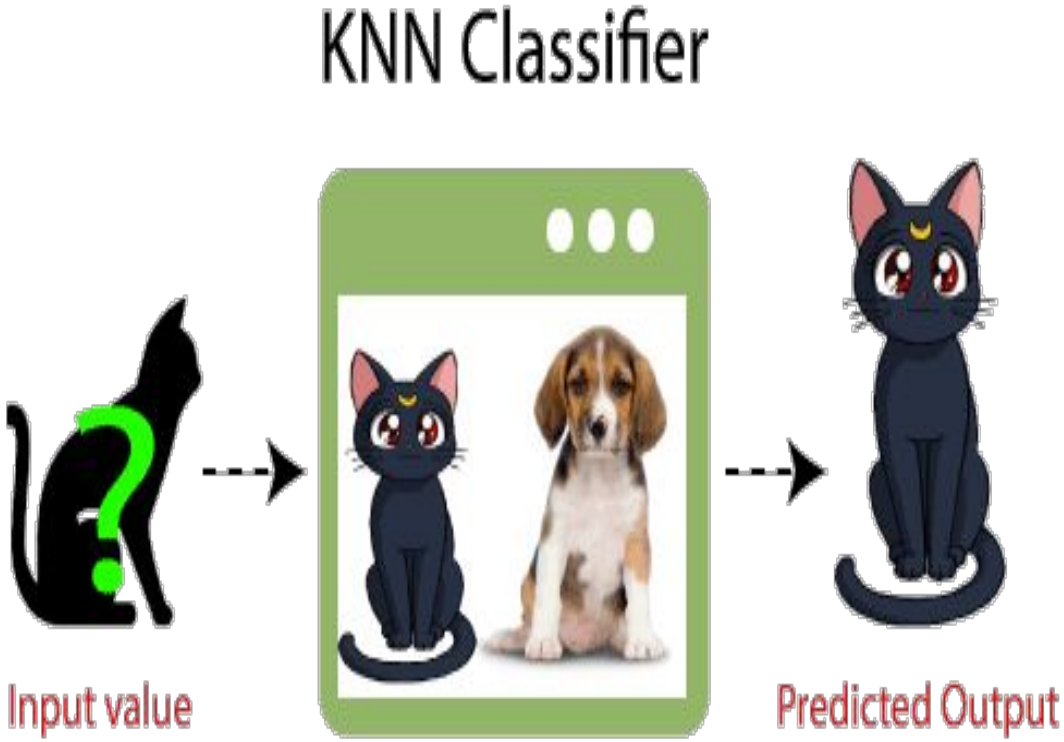
- Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog.
- So for this identification, we can use the KNN algorithm, as it works on a similarity measure.
- Our KNN model will find the similar features of the new data set to the cats and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



# Using it for pattern recognition




(Just one way to classify)



# Using it for estimating the density functions

- There are several types of non-parametric methods of interest in pattern recognition.
- It consists of procedures for estimating the density functions $p(x|y_q)$ from sample patterns (without the assumption that forms of underlying densities are known; and with the assumption that attributes are statistically independent).
- In probability theory, a probability density function is a function whose value at any given sample (or point) in the sample space (the set of possible values) can be interpreted as providing a relative likelihood that the value of the random variable would be close to that sample.

# Estimation of probability density function

- Assume that the sample $\{x^{(i)}\}$; i = 1,…, N, is drawn independently from some unknown probability density $p(.)$.
  - $\hat{p}(.)$ is our estimate of $p(.)$.
  - We start with the univariate case where $x(i)$ are scalars $(n = 1)$, and later generalize to the multidimensional case.
- For the non-parametric estimation, the oldest and the most popular method is the histogram, where the input space is divided into equal-sized intervals, named bins.
- The corresponding probability is approximated by the frequency ratio:

$$\hat{P}(x) = \frac{\#\{x^{(i)} \text{ in the same bin as } x\}}{N}$$

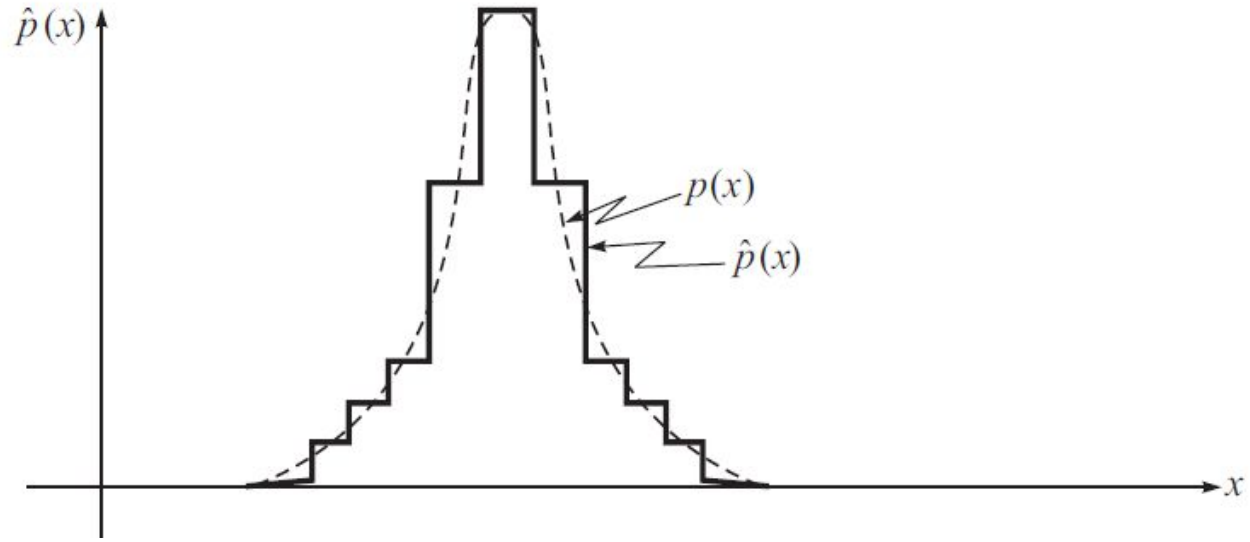
**where #{.} denotes the number of training instances $x^{(i)}$ that lie in the same bin as the test instance x.**

# Estimation of probability density function

- This approximation converges to the true probability as N approaches infinity.
- The corresponding probability density function value is assumed constant throughout

$$\hat{p}(x) \equiv \hat{p}(\bar{x}) = \left[ \frac{\#\{x^{(i)} \text{ in the same bin as } x\}}{N} \right] \frac{1}{h}$$

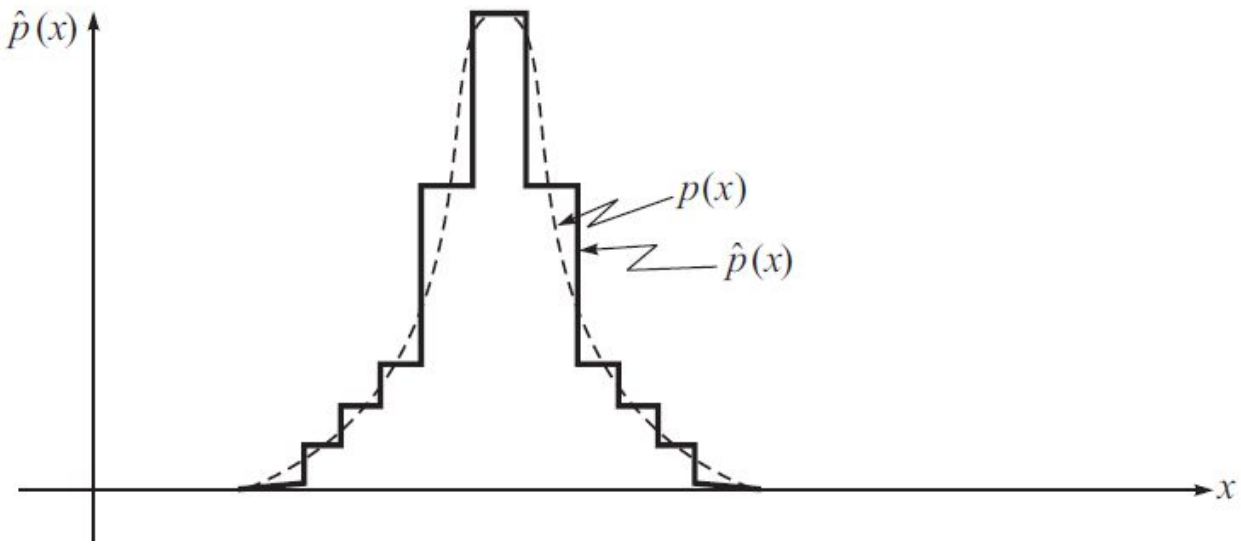
  - where $\bar{x}$ is the midpoint of the bin $\left( |x - \bar{x}| \leq \frac{h}{2} \right)$
  - and h is the width of the bin.



# Estimation of probability density function

- In constructing the histogram, we have to choose both an origin and a bin width h.
- With small bins, the estimate is spiky, and with larger bins, the estimate is smoother.
- There are discontinuities at the boundaries.

- For a large enough number of samples, the smaller the h, the better the accuracy of the resulting estimate.
- Usually, a large N is necessary for acceptable performance.



# Estimation of probability density function

- The nearest-neighbor class of estimators adopts the reasonable amount of smoothing to the local density of the data.
- The degree of smoothing is controlled by k (the number of neighbors taken into account), which is much smaller than N (the sample size).
- For each $x$ $(n = 1)$, we define $d_1(x) \le d_2(x) \le \cdots \le d_N(x)$ distances from $x$ to the points in the sample.
- The k-nearest neighbor (k-NN) density estimate is $\hat{p}(x) = \dfrac{k}{2N\, d_k(x)}$
- instead of fixing h and checking how many samples fall in the bin, we fix k, the number of observations to fall in the bin, and compute the bin size.
- Where density is high, bins are small, and where density is low, bins are larger.

# Estimation of probability density function

- Generalizing to multivariable data, $\hat{p}(\mathbf{x}) = \frac{k}{NV_k(\mathbf{x})}$
- where $V_k(x)$ is the volume of the n-dimensional hyperspace centered at $x$ with radius $r = ||x - x(k)||$; $x(k)$ is the $k^{\text{th}}$ nearest observation to $x$ among the neighbors.
- When used for classification, we need the class-conditional densities, $p(x|y_q)$ of the feature vector distributions.
- The estimator of the class-conditional density is given as, $\hat{p}(\mathbf{x}|y_q) = \frac{k_q}{N_q V_k(\mathbf{x})}$
- where $k_q$ is the number of neighbors out of $k$ nearest that belong to class $y_q$; and $N_q$ is the number of labeled instances belonging to class $y_q$.

# Estimating a posteriori probabilities using KNN

A reasonable estimate for $P(y_q|\mathbf{x})$ is

($y_q$ is the class for sample $x$)

$$\hat{P}(y_q|\mathbf{x}) = \frac{\hat{p}(\mathbf{x}|y_q)\,\hat{P}(y_q)}{\hat{p}(\mathbf{x})}$$

$$\hat{p}(\mathbf{x}|y_q) = \frac{k_q}{N_q V_k(\mathbf{x})}$$

$$\hat{P}(y_q) = N_q/N$$

$$\hat{p}(\mathbf{x}) = \frac{k}{N V_k(\mathbf{x})}$$

Therefore,

$$\hat{P}(y_q|\mathbf{x}) = \left(\frac{k_q}{N_q V_k(\mathbf{x})} \times \frac{N_q}{N}\right) \Big/ \left(\frac{k}{N V_k(\mathbf{x})}\right)$$

$$= k_q/k$$

**Thus, the estimate of the a posteriori probability that $y_q$ is the class for sample $x$ is just the fraction of the samples within the cell $V_k(x)$ that are labeled $y_q$.**

# KNN Classification

# Computer Vision: Car detection





Cars






Not a car

Testing:

What is this?

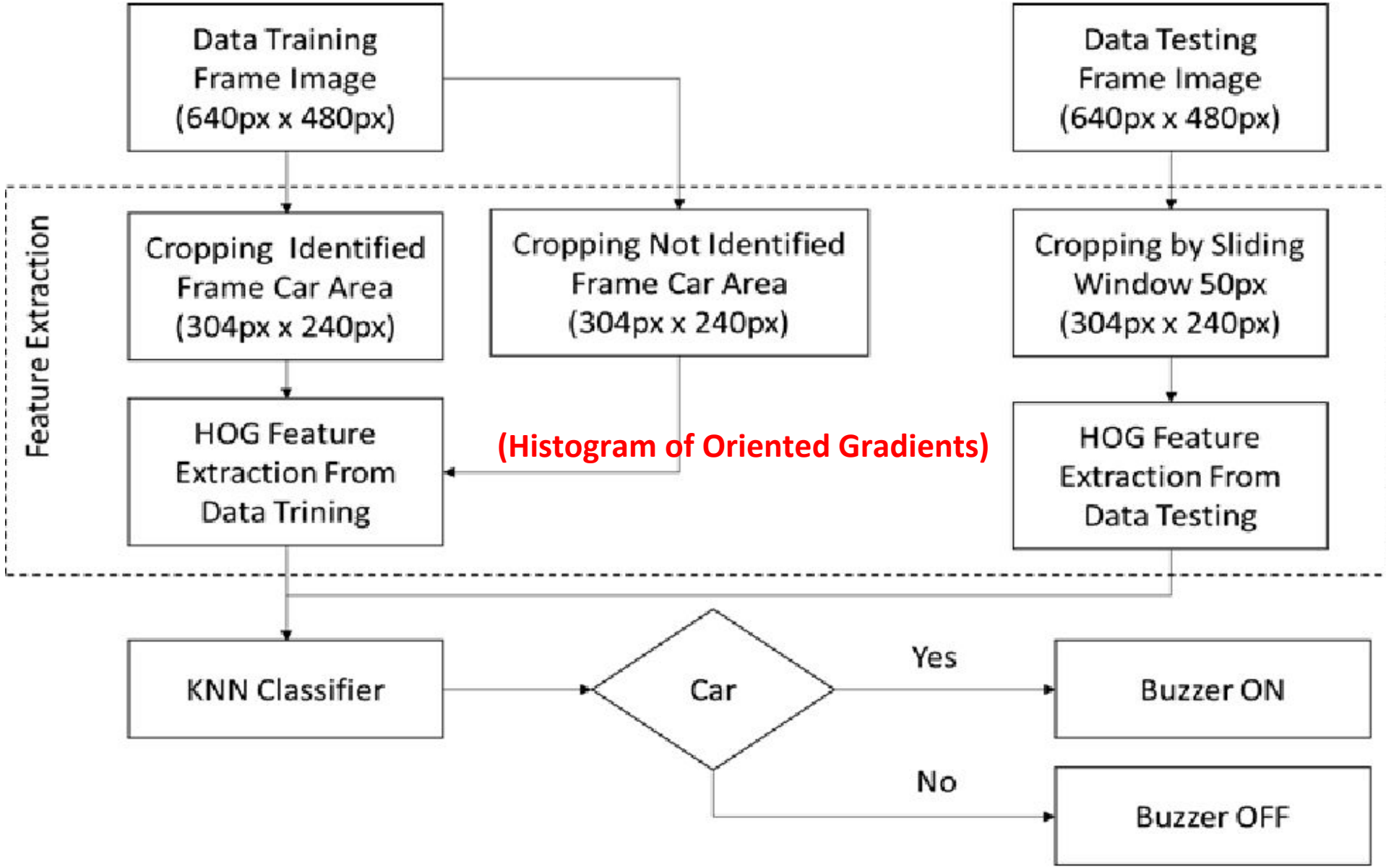
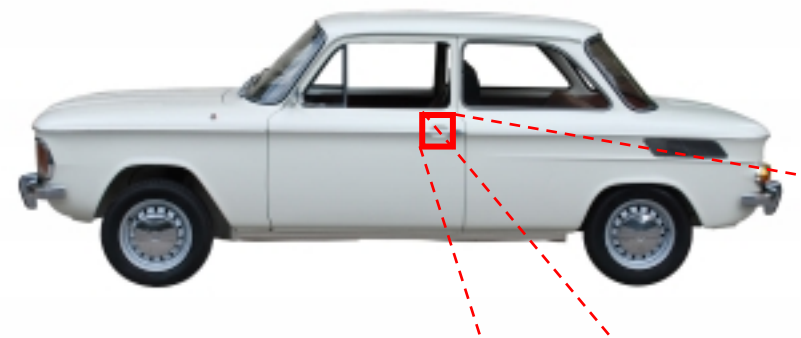



Figure 1 Block System

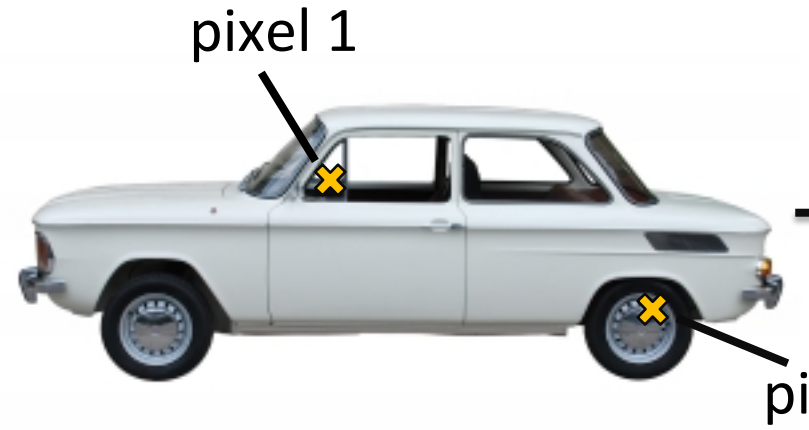
# What is this?

You see this:

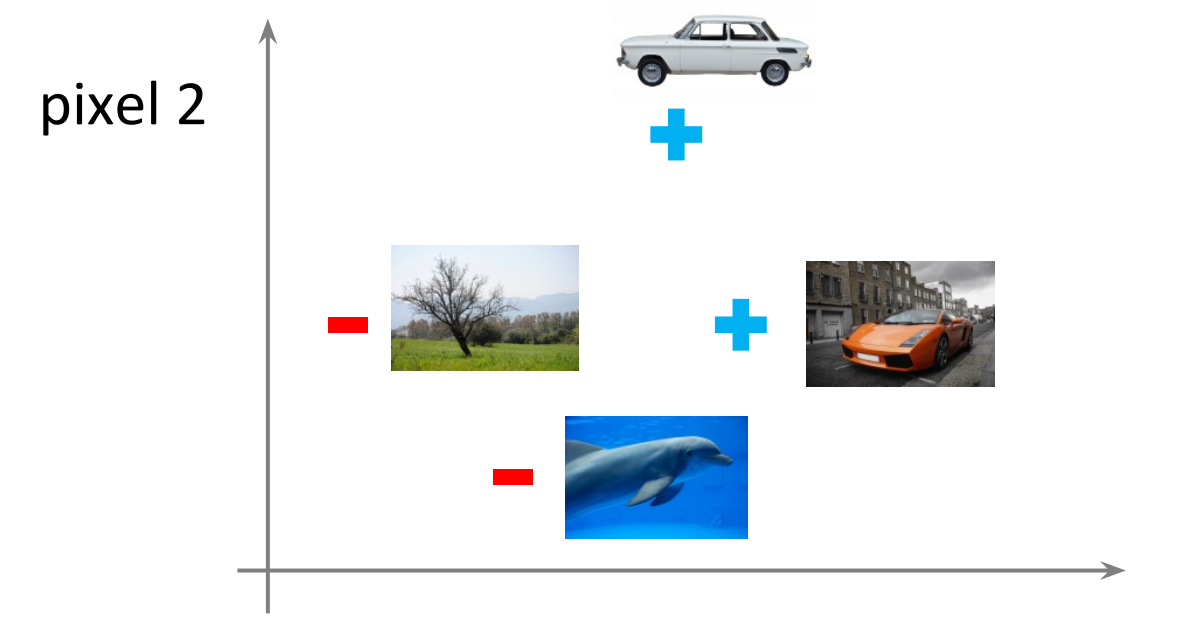


But the camera sees this:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 194 | 210 | 201 | 212 | 199 | 213 | 215 | 195 | 178 | 158 | 182 | 209 |
| 180 | 189 | 190 | 221 | 209 | 205 | 191 | 167 | 147 | 115 | 129 | 163 |
| 114 | 126 | 140 | 188 | 176 | 165 | 152 | 140 | 170 | 106 | 78 | 88 |
| 87 | 103 | 115 | 154 | 143 | 142 | 149 | 153 | 173 | 101 | 57 | 57 |
| 102 | 112 | 106 | 131 | 122 | 138 | 152 | 147 | 128 | 84 | 58 | 66 |
| 94 | 95 | 79 | 104 | 105 | 124 | 129 | 113 | 107 | 87 | 69 | 67 |
| 68 | 71 | 69 | 98 | 89 | 92 | 98 | 95 | 89 | 88 | 76 | 67 |
| 41 | 56 | 68 | 99 | 63 | 45 | 60 | 82 | 58 | 76 | 75 | 65 |
| 20 | 43 | 69 | 75 | 56 | 41 | 51 | 73 | 55 | 70 | 63 | 44 |
| 50 | 50 | 57 | 69 | 75 | 75 | 73 | 74 | 53 | 68 | 59 | 37 |
| 72 | 59 | 53 | 66 | 84 | 92 | 84 | 74 | 57 | 72 | 63 | 42 |
| 67 | 61 | 58 | 65 | 75 | 78 | 76 | 73 | 59 | 75 | 69 | 50 |

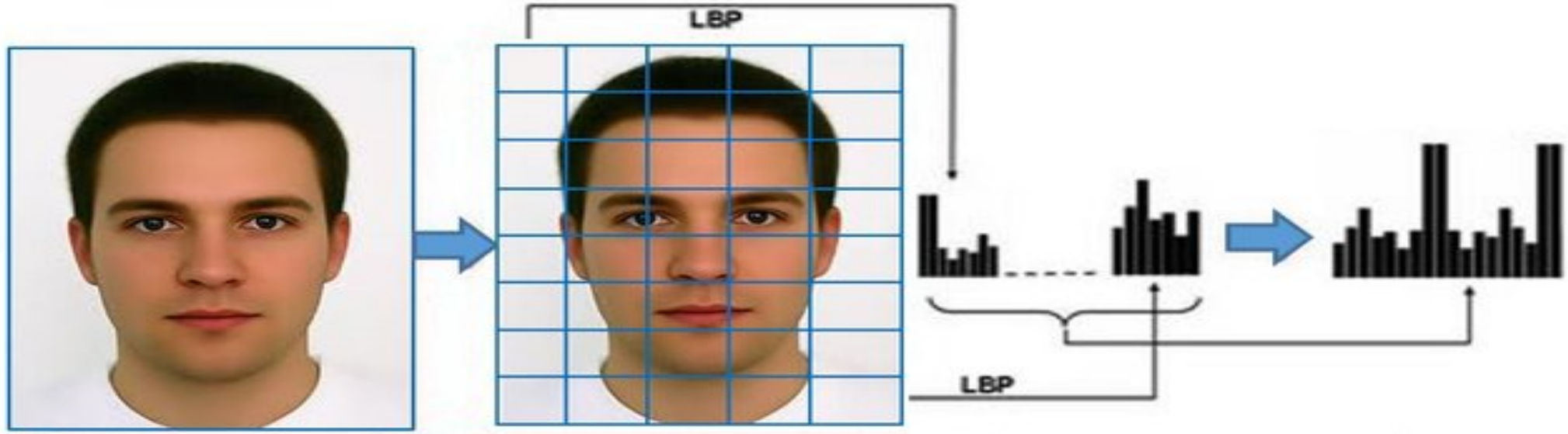



Learning Algorithm

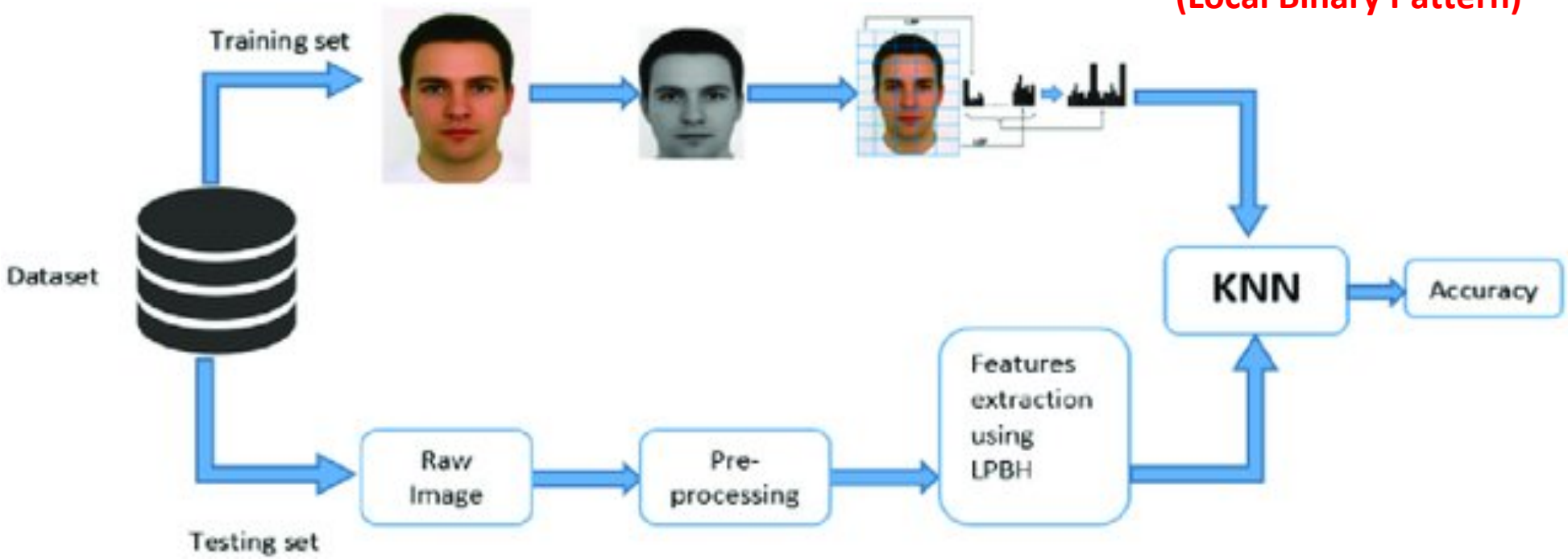



+ Cars

- “Non”-Cars




(Local Binary Pattern)




To find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram (based on distance).

# How does it work?

- We first determine the k points that are closest neighbors of x with the help of a specific distance metric.
- The categorization of x is then given by the class label found in most of the k neighbors.
- All neighbors have equal vote, and the class with the most number of votes among the k neighbors is selected.
- Ties are randomly broken or weighted vote is taken.
- k is usually taken an odd number to reduce ties: confusion exists usually between two neighboring classes.
- Note that k = 1 is usually not sufficient for determining the class of x due to noise and outliers in the data.
- A set of nearest neighbors is needed to accurately decide the class.

# Distance function?

- The key issue of k-NN algorithm is the distance/similarity function, which is selected on the basis of applications and nature of the data.
- The cautious selection of an appropriate distance function is a crucial step in the use of k-NN.
- A validation set can be made use of to pick the appropriate distance function for a given dataset by applying all candidates to gauge which one gives better results.

# Value of k?

- Another question is the number of neighbors to select.
- Again, investigation of varying numbers of neighbors with the help of the validation set can facilitate establishing the optimal number, as the number is dependent on the data distribution and relies heavily on the problem being solved.
- Selecting the appropriate training set is probably the most important step in k-NN process.
- The training set should include enough examples of all probable categories, that is, it should be a balanced training set having more or less the same number of instances for all classes.
- Generally speaking, the size of the training set should have at least thousands, if not hundreds of thousands or millions, of training examples.

# Value of k?

- The data is partitioned into training, validation, and test sets.
- Validation set is used to compare error rates for various values of k/various similarity measures, and test set is used to evaluate the performance.
- In the k-NN procedure, all neighbors have equal vote.
- In an alternative method: the Parzon-window approach, also called Probabilistic Neural Network (PNN), the weight of the vote is given by a kernel function, typically giving more weight to closer instances.
- Low k-value is sensitive to outliers and a higher K-value is more resilient to outliers as it considers more voters to decide prediction.

# Value of k?

- Cross-validation is a smart way to find out the optimal K value.
- It estimates the validation error rate by holding out a subset of the training set from the model building process.
- Cross-validation (let's say 10 fold validation) involves randomly dividing the training set into 10 groups, or folds, of approximately equal size.
- 90% data is used to train the model and remaining 10% to validate it.
- The misclassification rate is then computed on the 10% validation data.
- This procedure repeats 10 times.
- Different group of observations are treated as a validation set each of the 10 times.
- It results to 10 estimates of the validation error which are then averaged out.

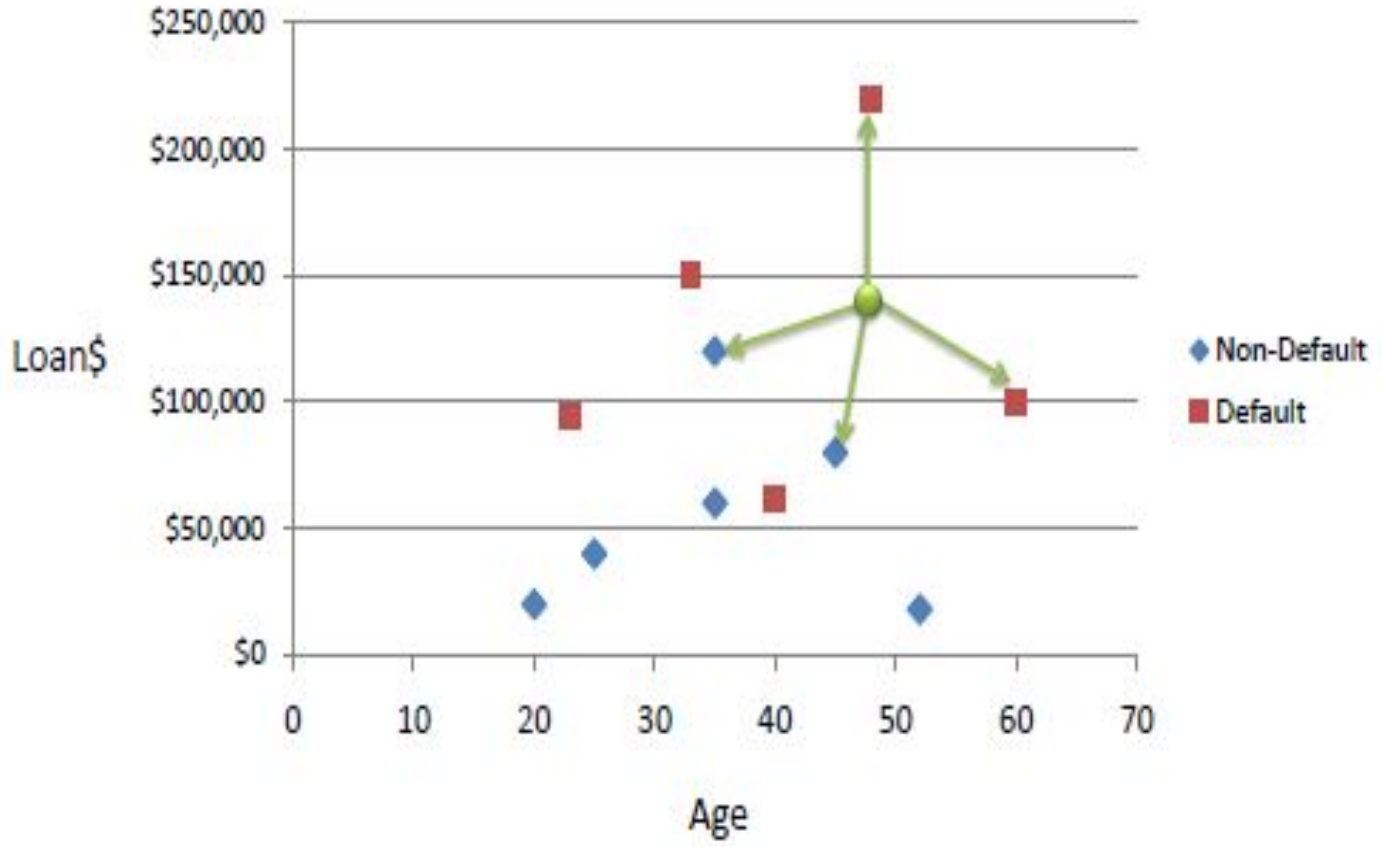
# Pros and Cons of KNN

| Pros | Cons |
| --- | --- |
| Easy to understand | Memory Intensive / Computationally expensive |
| No assumptions about data | Sensitive to scale of data |
| Can be applied to both classification and regression | Not work well on rare event (skewed) target variable |
| Works easily on multi-class problems | Struggle when high number of independent variables |

# Numerical Discussion

# Example 1:

- Consider the following data concerning credit default. Age and Loan are two numerical variables (predictors) and Default is the target.



# Example 1:

- We can now use the training set to classify an unknown case (Age=48 and Loan=$142,000) using Euclidean distance.
- If K=1 then the nearest neighbor is the last case in the training set with Default=Y.
- D = Sqrt[(48-33)^2 + (142000-150000)^2] = 8000.01 >> Default=Y
- With K=3, there are two “Default=Y” and one “Default=N” out of three closest neighbors. The prediction for the unknown case is again Default=Y.

| 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

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

# Example 1:

- One major drawback in calculating distance measures directly from the training set is in the case where variables have different measurement scales or there is a mixture of numerical and categorical variables.
- For example, if one variable is based on annual income in dollars, and the other is based on age in years then income will have a much higher influence on the distance calculated.
- One solution is to standardize the training set as shown here.

**Standardized Distance**

| 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

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

**Using the standardized distance on the same training set, the unknown case returned a different neighbor which is not a good sign of robustness.**

# Example 2:

- Suppose we have height, weight and T-shirt size of some customers and we need to predict the T-shirt size of a new customer given only height and weight information we have.
- Data including height, weight and T-shirt size information is shown here
- **Step 1 : Calculate Similarity based on distance function**
- There are many distance functions but Euclidean is the most commonly used measure.
- It is mainly used when data is continuous. Manhattan distance is also very common for continuous variables.

| Height (in cms) | Weight (in kgs) | T Shirt Size |
|---|---|---|
| 158 | 58 | M |
| 158 | 59 | M |
| 158 | 63 | M |
| 160 | 59 | M |
| 160 | 60 | M |
| 163 | 60 | M |
| 163 | 61 | M |
| 160 | 64 | L |
| 163 | 64 | L |
| 165 | 61 | L |
| 165 | 62 | L |
| 165 | 65 | L |
| 168 | 62 | L |
| 168 | 63 | L |
| 168 | 66 | L |
| 170 | 63 | L |
| 170 | 64 | L |
| 170 | 68 | L |

# Example 2:

- The idea to use distance measure is to find the distance (similarity) between new sample and training cases and then finds the k-closest customers to new customer in terms of height and weight.
- New customer named 'Monica' has height 161cm and weight 61kg.
- Euclidean distance between first observation and new observation (monica) is as follows - d=SQRT((161-158)^2+(61-58)^2)
- Similarly, we will calculate distance of all the training cases with new case and calculates the rank in terms of distance.
- The smallest distance value will be ranked 1 and considered as nearest neighbor.

Euclidean :

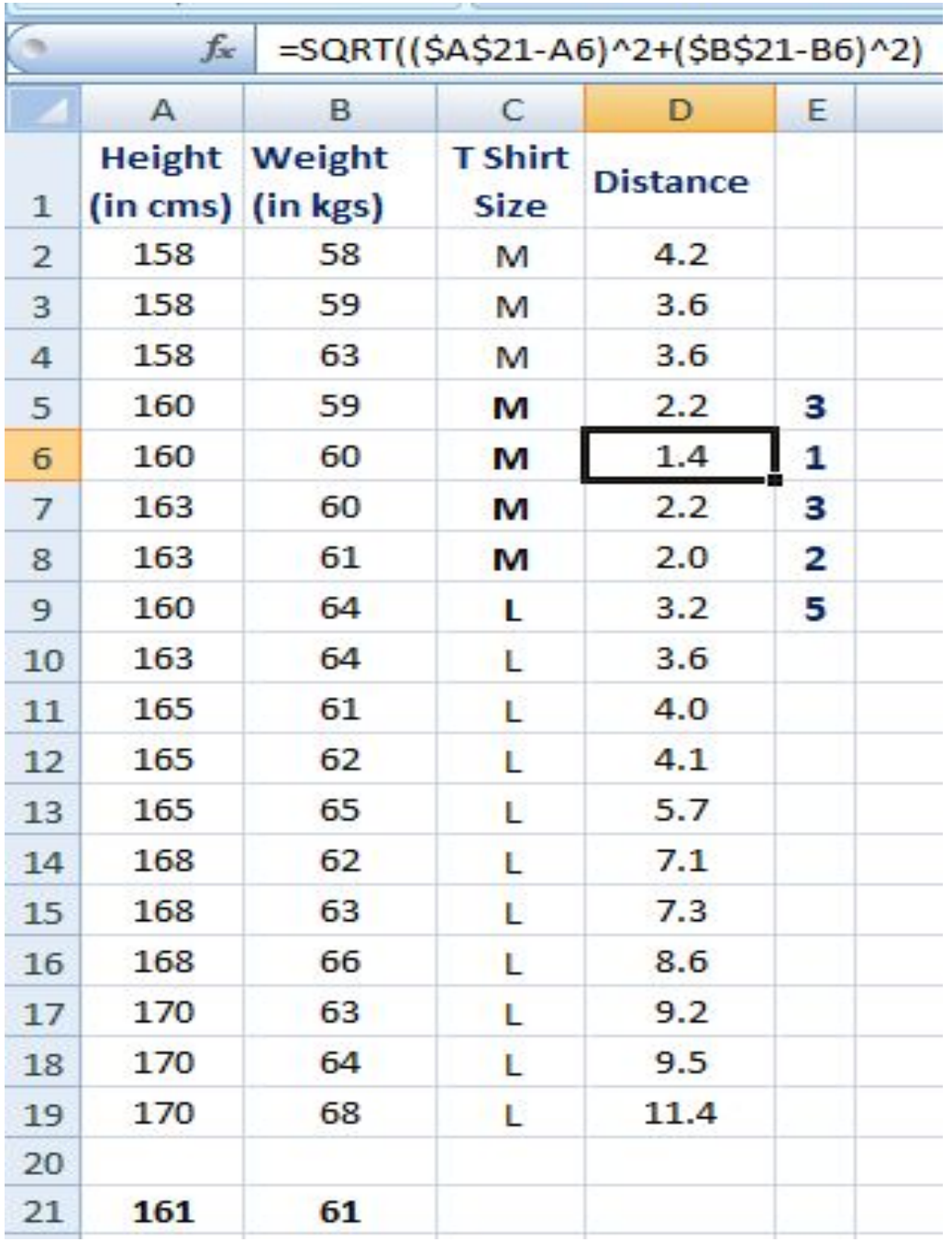
$$d(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}$$

Manhattan / city - block :

$$d(x,y) = \sum_{i=1}^{m}|x_i - y_i|$$

# Example 2:

- **Step 2 : Find K-Nearest Neighbors**
- Let k be 5.
- Then the algorithm searches for the 5 customers closest to Monica, i.e. most similar to Monica in terms of attributes, and see what categories those 5 customers were in.
- If 4 of them had ‘Medium T shirt sizes’ and 1 had ‘Large T shirt size’ then your best guess for Monica is ‘Medium T shirt.
- See the calculation shown in the figure.

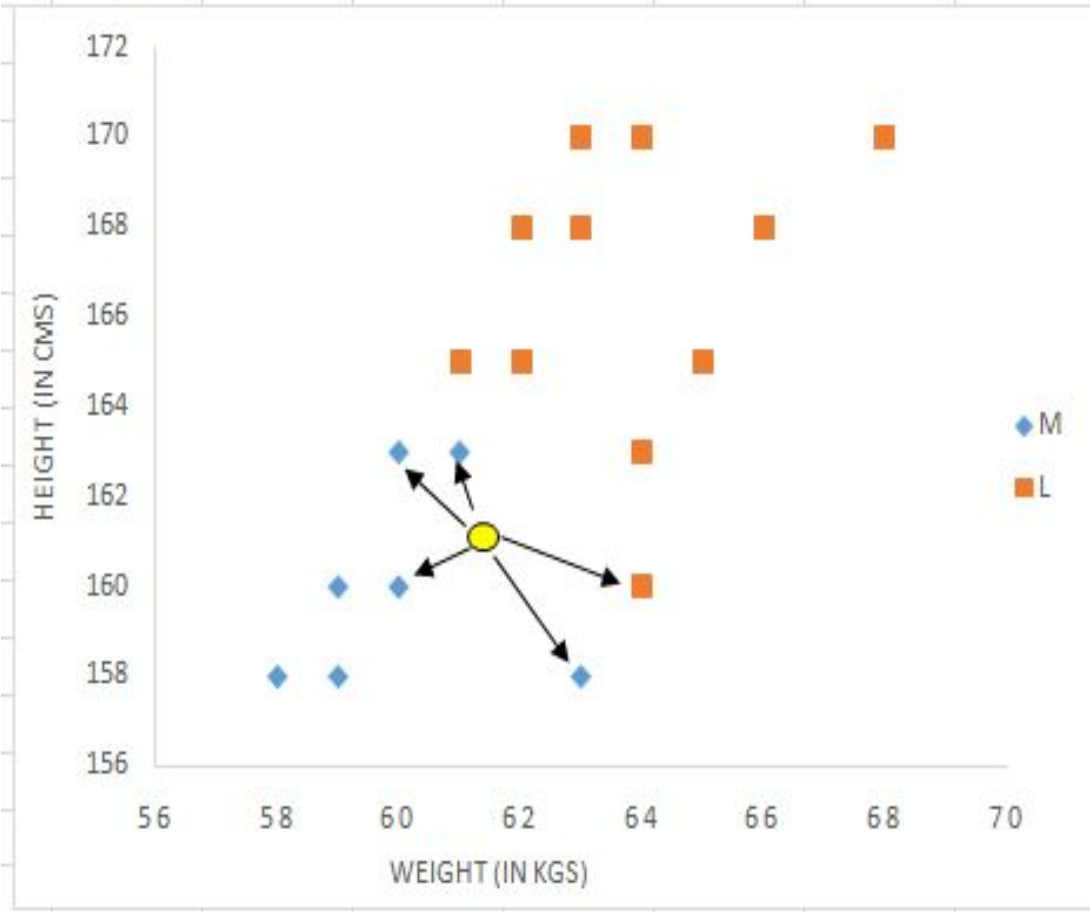


=SQRT(($A$21-A6)^2+($B$21-B6)^2)

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Height (in cms) | Weight (in kgs) | T Shirt Size | Distance | |
| 2 | 158 | 58 | M | 4.2 | |
| 3 | 158 | 59 | M | 3.6 | |
| 4 | 158 | 63 | M | 3.6 | |
| 5 | 160 | 59 | **M** | 2.2 | **3** |
| 6 | 160 | 60 | **M** | 1.4 | **1** |
| 7 | 163 | 60 | **M** | 2.2 | **3** |
| 8 | 163 | 61 | **M** | 2.0 | **2** |
| 9 | 160 | 64 | **L** | 3.2 | **5** |
| 10 | 163 | 64 | L | 3.6 | |
| 11 | 165 | 61 | L | 4.0 | |
| 12 | 165 | 62 | L | 4.1 | |
| 13 | 165 | 65 | L | 5.7 | |
| 14 | 168 | 62 | L | 7.1 | |
| 15 | 168 | 63 | L | 7.3 | |
| 16 | 168 | 66 | L | 8.6 | |
| 17 | 170 | 63 | L | 9.2 | |
| 18 | 170 | 64 | L | 9.5 | |
| 19 | 170 | 68 | L | 11.4 | |
| 20 | | | | | |
| 21 | **161** | **61** | | | |

# Example 2:

- In the graph, binary dependent variable (T-shirt size) is displayed in blue and orange color.
- 'Medium T-shirt size' is in blue color and 'Large T-shirt size' in orange color.
- New customer information is exhibited in yellow circle.
- Four blue highlighted data points and one orange highlighted data point are close to yellow circle.
- So the prediction for the new case is blue highlighted data point which is Medium T-shirt size.



# Example 2:

- **Standardization**
- In order to make the values comparable we need to standardize them which can be done by any of the following methods :

$$Xs = \frac{X - mean}{s.d.}$$

$$Xs = \frac{X - mean}{max - min}$$

$$Xs = \frac{X - min}{\max - min}$$

- After standardization, 5th closest value got changed as height was dominating earlier before standardization. Hence, it is important to standardize predictors before running K-nearest neighbor algorithm.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | Height (in cms) | Weight (in kgs) | T Shirt Size | Distance | |
| 2 | -1.39 | -1.64 | M | 1.3 | |
| 3 | -1.39 | -1.27 | M | 1.0 | |
| 4 | -1.39 | 0.25 | M | 1.0 | |
| 5 | -0.92 | -1.27 | M | 0.8 | 4 |
| 6 | -0.92 | -0.89 | M | 0.4 | 1 |
| 7 | -0.23 | -0.89 | M | 0.6 | 3 |
| 8 | -0.23 | -0.51 | M | 0.5 | 2 |
| 9 | -0.92 | 0.63 | L | 1.2 | |
| 10 | -0.23 | 0.63 | L | 1.2 | |
| 11 | 0.23 | -0.51 | L | 0.9 | 5 |
| 12 | 0.23 | -0.13 | L | 1.0 | |
| 13 | 0.23 | 1.01 | L | 1.8 | |
| 14 | 0.92 | -0.13 | L | 1.7 | |
| 15 | 0.92 | 0.25 | L | 1.8 | |
| 16 | 0.92 | 1.39 | L | 2.5 | |
| 17 | 1.39 | 0.25 | L | 2.2 | |
| 18 | 1.39 | 0.63 | L | 2.4 | |
| 19 | 1.39 | 2.15 | L | 3.4 | |
| 20 | | | | | |
| 21 | -0.7 | -0.5 | | | |

# Assessing Classification Accuracy

- Misclassification Error
- Metric for assessing the accuracy of classification algorithms is: *number of samples misclassified by the model* $h(\mathbf{w}, \mathbf{x})$.
- For binary classification problems,

$$y^{(i)} \epsilon\, [0,1], \text{ and } h(\mathbf{w}, \mathbf{x}) = \hat{y}^{(i)} \epsilon\, [0,1]; i = 1, \ldots, N$$

- For 0% error, $\left(y^{(i)} - \hat{y}^{(i)}\right) = 0$ for all data points

$$Misclassification\ error = \frac{\text{Number of data points for which} \left(y^{(i)} - \hat{y}^{(i)}\right) \neq 0}{N}$$