Machine learning

Prepared by : Dr. Hanaa Bayomi
Updated By: Prof Abeer ElKorany

Lecture 6: KNN

Content

1. Eager Vs Lazy learning
2. What is KNN
3. How to choose K
4. Error Rate
5. Label continuous output
6. Cross Validation

Background

- The classification algorithms presented before are **eager learners**
  - Construct a model before receiving new tuples to classify
  - Learned models are ready and eager to classify previously unseen tuples
- **Lazy learners**
  - The learner waits till the last minute before doing any model construction
  - In order to classify a given test tuple
    - Store training tuples
    - Wait for test tuples
    - Perform generalization based on similarity between test and the stored training tuples

Eager vs lazy learner

Eager learner	Lazy learner
The distinction between easy learners and lazy learners is based on **when the algorithm abstracts from the data**.	
1. When it receive data set it starts classifying (learning) 2. Then it does not wait for test data to learn 3. So it takes long time learning and less time classifying data	1. Just store Data set **without** learning from it 2. Start classifying data when it receive **Test data** 3. So it takes less time learning and more time classifying data
Do lot of work on training data Ex. Linear Regression, Decision tree	Do less work on training data Ex. K. Nearest Neighbors

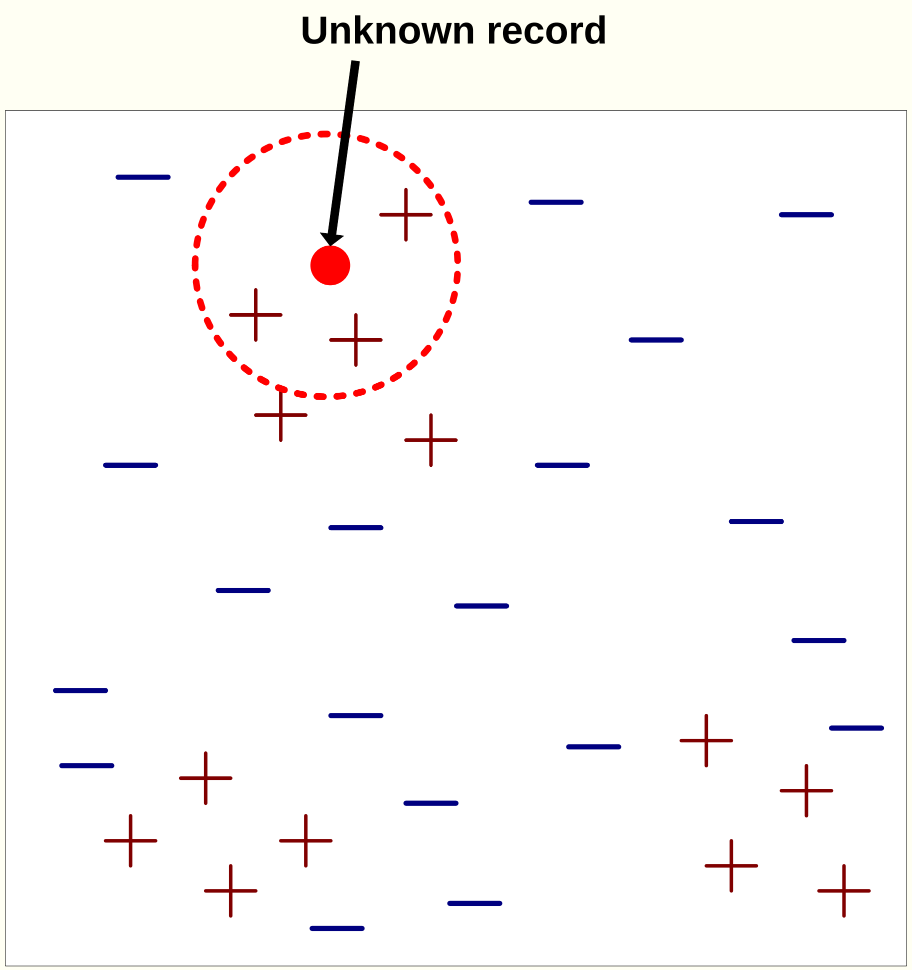
What is KNN

1. A very simple classification and regression algorithm
   a. in case of classification, new data point get classified in a particular class
   b. in case of regression, new data point get labeled based on the AVR(Average or Weighted value) Value of KNN
2. It is a lazy learner because it doesn't learn much from the training data (most of learning happens from a live data)
3. It is a supervised learning algorithm
4. Default method is Euclidean distance
5. Non- parametric method used for classification

Basic k-Nearest Neighbor Classification

- Given training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
- Define a distance metric between points in input space $D(x_1,x_i)$
  - E.g., Eucledian distance, Weighted Eucledian, Mahalanobis distance, TFIDF, etc.
- Training method:
  - Save the training examples
- At prediction time:
  - Find the k training examples $(x_1,y_1),\ldots(x_k,y_k)$ that are **closest** to the test example x given the distance $D(x_1,x_i)$
  - Predict the most frequent class among those y_i's.

Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

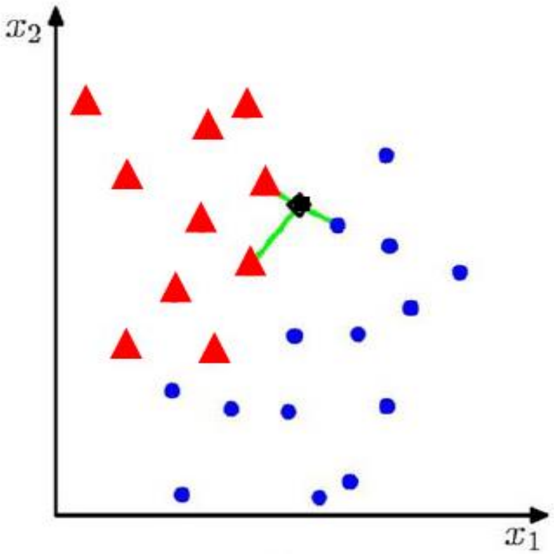
K-NN algorithm

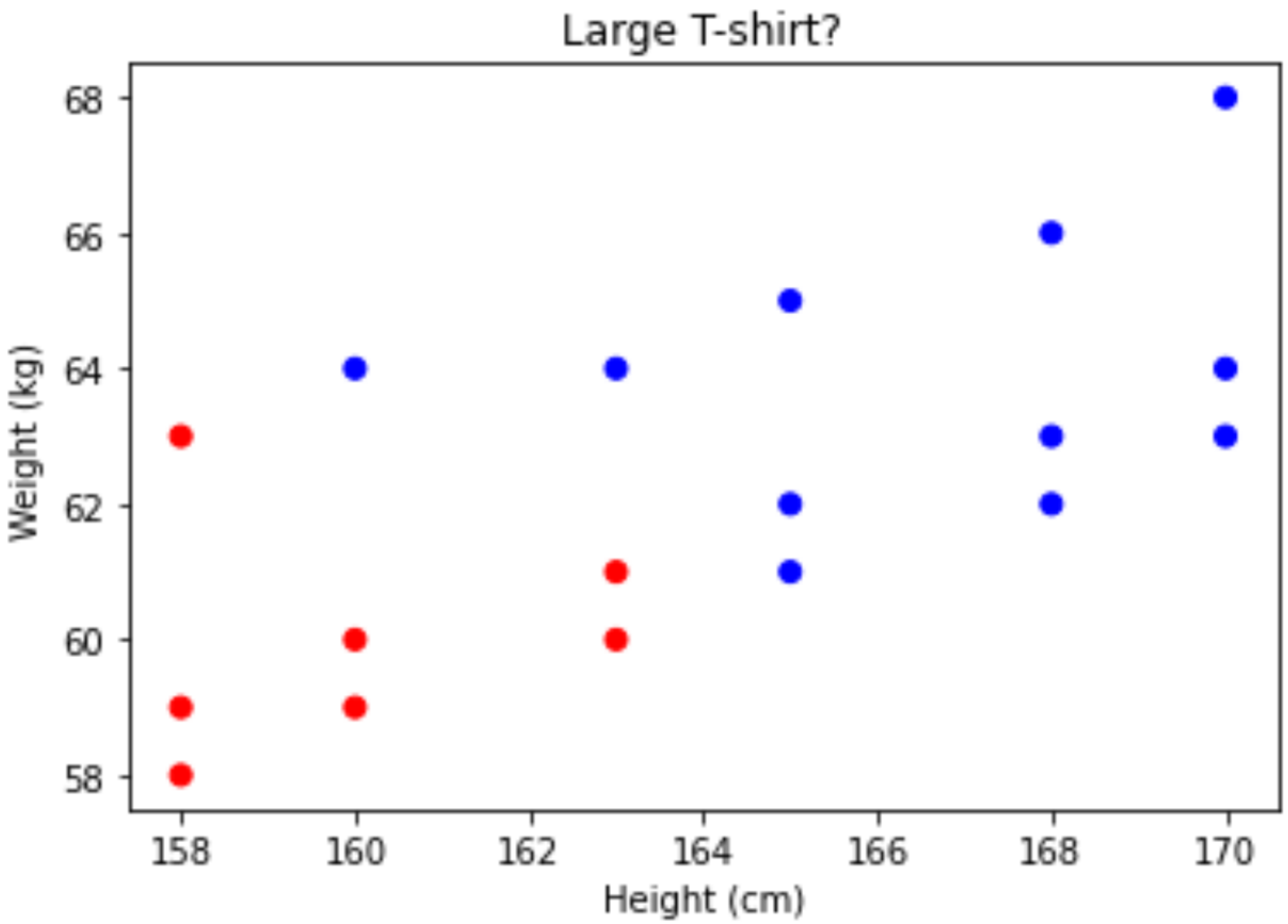
- 1 NN
  - Predict the same value/class as the nearest instance in the training set
- k NN
  - find the k closest training points (small $\|x_i - x_0\|$ according to some metric, for ex. euclidean, manhattan, etc.)
  - predicted class: majority vote

Distance functions

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

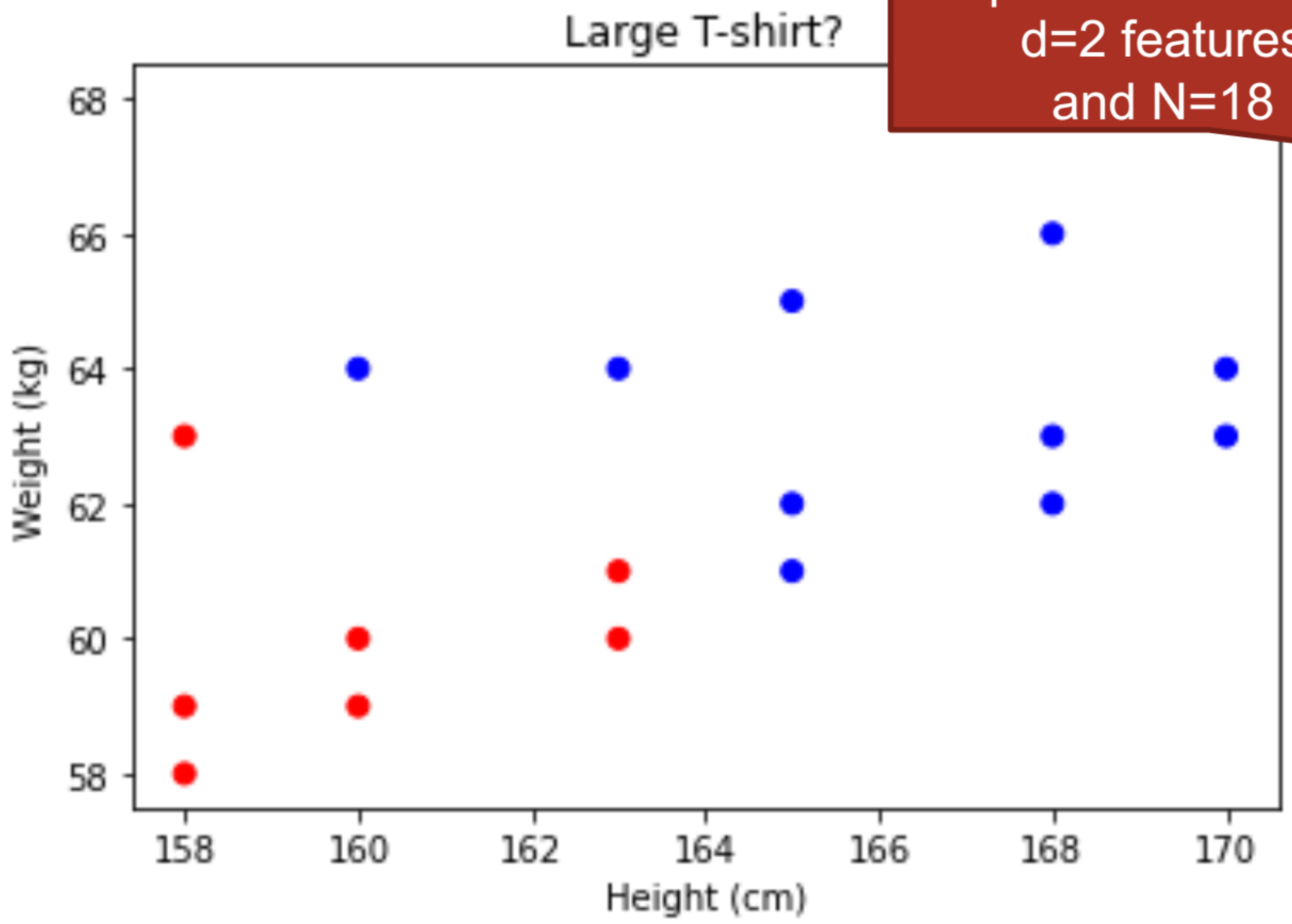
Manhattan $$\sum_{i=1}^{k}|x_i - y_i|$$





Height (cm)	Weight (kg)	Large (vs Medium) t-shirt?
158	58	F
158	59	F
158	63	F
160	59	F
160	60	F
163	60	F
163	61	F
160	64	T
163	64	T
165	61	T
165	62	T
165	65	T
168	62	T
168	63	T
168	66	T
170	63	T
170	64	T
170	68	T

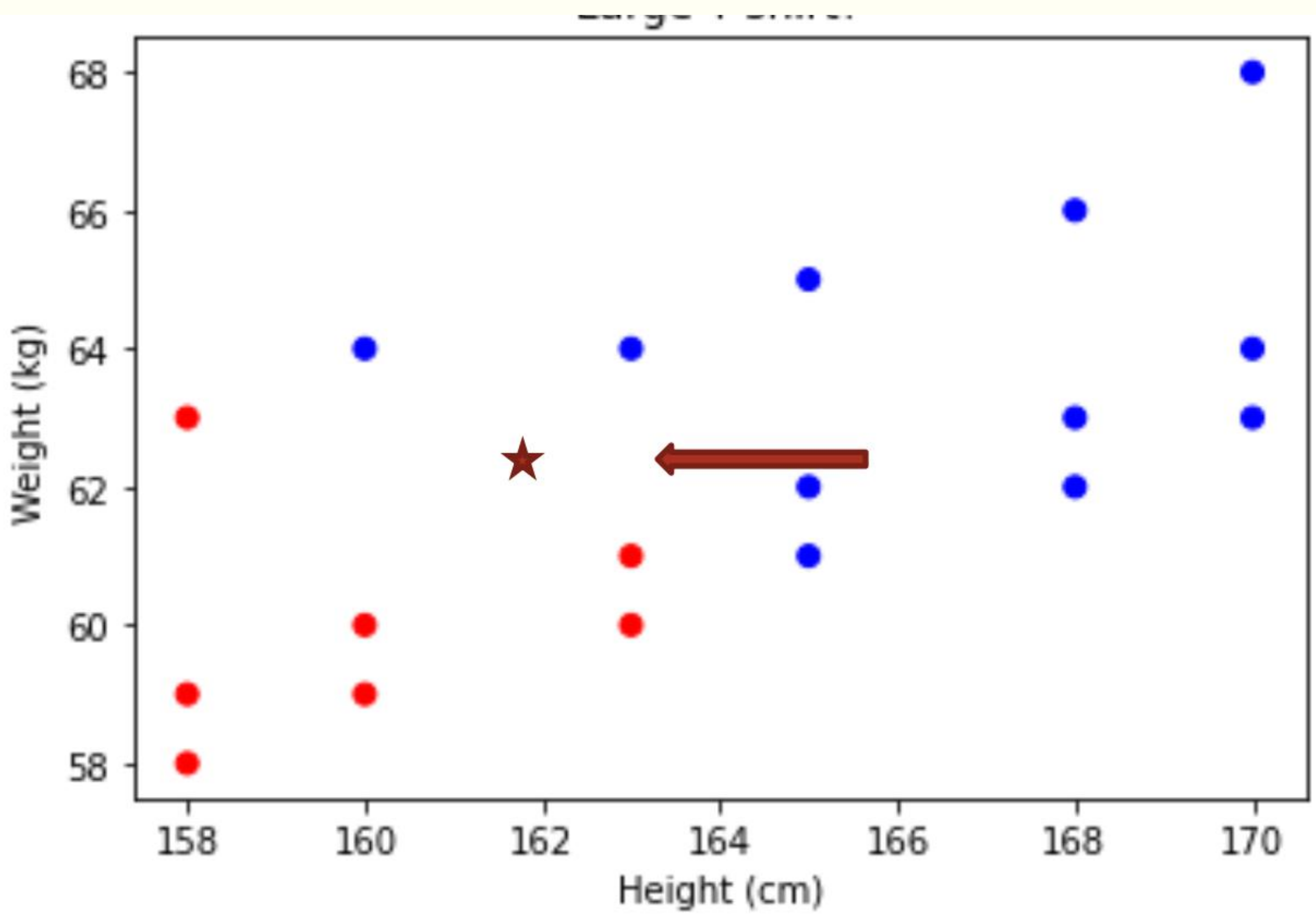
Based on data from https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html



Input matrix $\boldsymbol{X}$ with d=2 features and N=18

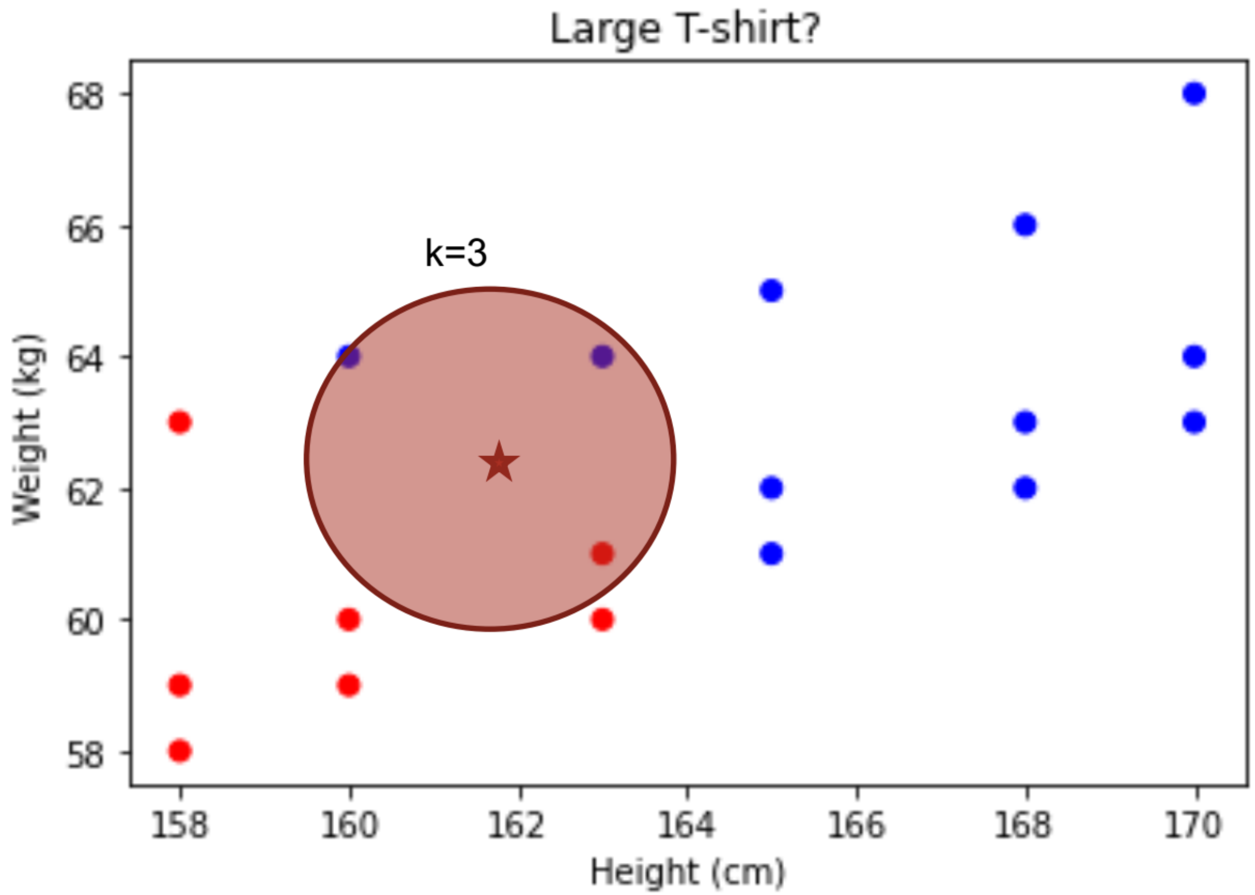
Height (cm)	Weight (kg)	Large (vs Medium) t-shirt?
158	58	F
158	59	F
158	63	F
160	59	F
160	60	F
163	60	F
163	61	F
160	64	T
163	64	T
165	61	T
165	62	T
165	65	T
168	62	T
168	63	T
168	66	T
170	63	T
170	64	T
170	68	T

Based on data from https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html



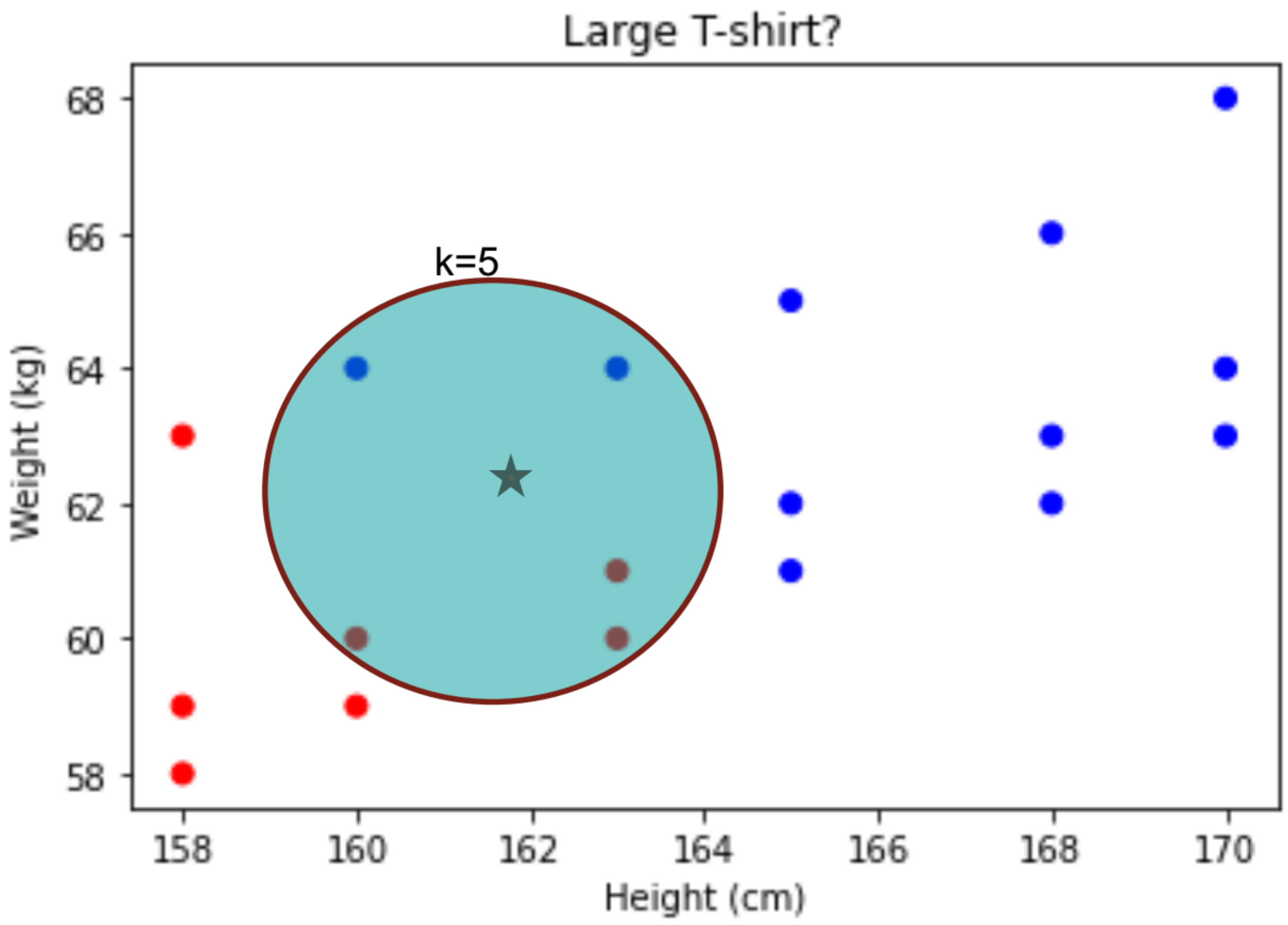
158	58	F
158	59	F
158	63	F
160	59	F
160	60	F
163	60	F
163	61	F
160	64	T
163	64	T
165	61	T
165	62	T
165	65	T
168	62	T
168	63	T
168	66	T
170	63	T
170	64	T
170	68	T

Based on data from https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html



Height (cm)	Weight (kg)	Large (vs Medium) t-shirt?
158	58	F
158	59	F
158	63	F
160	59	F
160	60	F
163	60	F
163	61	F
160	64	T
163	64	T
165	61	T
165	62	T
165	65	T
168	62	T
168	63	T
168	66	T
170	63	T
170	64	T
170	68	T

Based on data from https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html



Based on data from https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html

Height (cm)	Weight (kg)	Large (vs Medium) t-shirt?
158	58	F
158	59	F
158	63	F
160	59	F
160	60	F
163	60	F
163	61	F
160	64	T
163	64	T
165	61	T
165	62	T
165	65	T
168	62	T
168	63	T
168	66	T
170	63	T
170	64	T
170	68	T

Distance Measures

- **Numeric features:**
  - Euclidean, Manhattan, L^n-norm:

    $$L^n(\mathbf{x}_1, \mathbf{x}_2) = \sqrt[n]{\sum_{i=1}^{\#\text{dim}} |\mathbf{x}_{1,i} - \mathbf{x}_{2,i}|^n}$$
  - Normalized by: range, std. deviation

- **Symbolic features:**
  - Hamming/overlap
  - Value difference measure (VDM):

    $$\delta(val_i, val_j) = \sum_{h=1}^{\#\text{classes}} |P(c_h|val_i) - P(c_h|val_j)|^n$$

- **In general:** arbitrary, encode knowledge

Example

Consider a dataset with two continuous features, "Age" and "Income," and a binary target variable "Loan Approval" (0 or 1).

Consider a new instance :

age = 32 and income = 65000.

Calculate the approval status using KNN with k = 3.

Age	Income	Loan Approval
25	50000	0
30	60000	1
35	75000	1
40	80000	0

1. Calculate Euclidean distance:
   1. Distance from (32, 65000) to (fist) = sqrt($(32-25)^2 + (65000-50000)^2$) = 15346.09
   2. Distance from (32, 65000) to (second) = sqrt($(32-30)^2 + (65000-60000)^2$) = **7071.07**
   3. Distance from (32, 65000) to (third) = sqrt($(32-35)^2 + (65000-75000)^2$) = **10000.00**
   4. Distance from (32, 65000) to (fourth) = sqrt($(32-40)^2 + (65000-80000)^2$) = **15345.08**

2. Select 3 nearest neighbors **7071.07, 10000.00, 15345.08**

3. Determine the majority class:

2 instances of loan approval (class 1)

1 instance of loan rejection (class 0).

The predicted loan approval status for the new data point is 1 (Approved).

Example

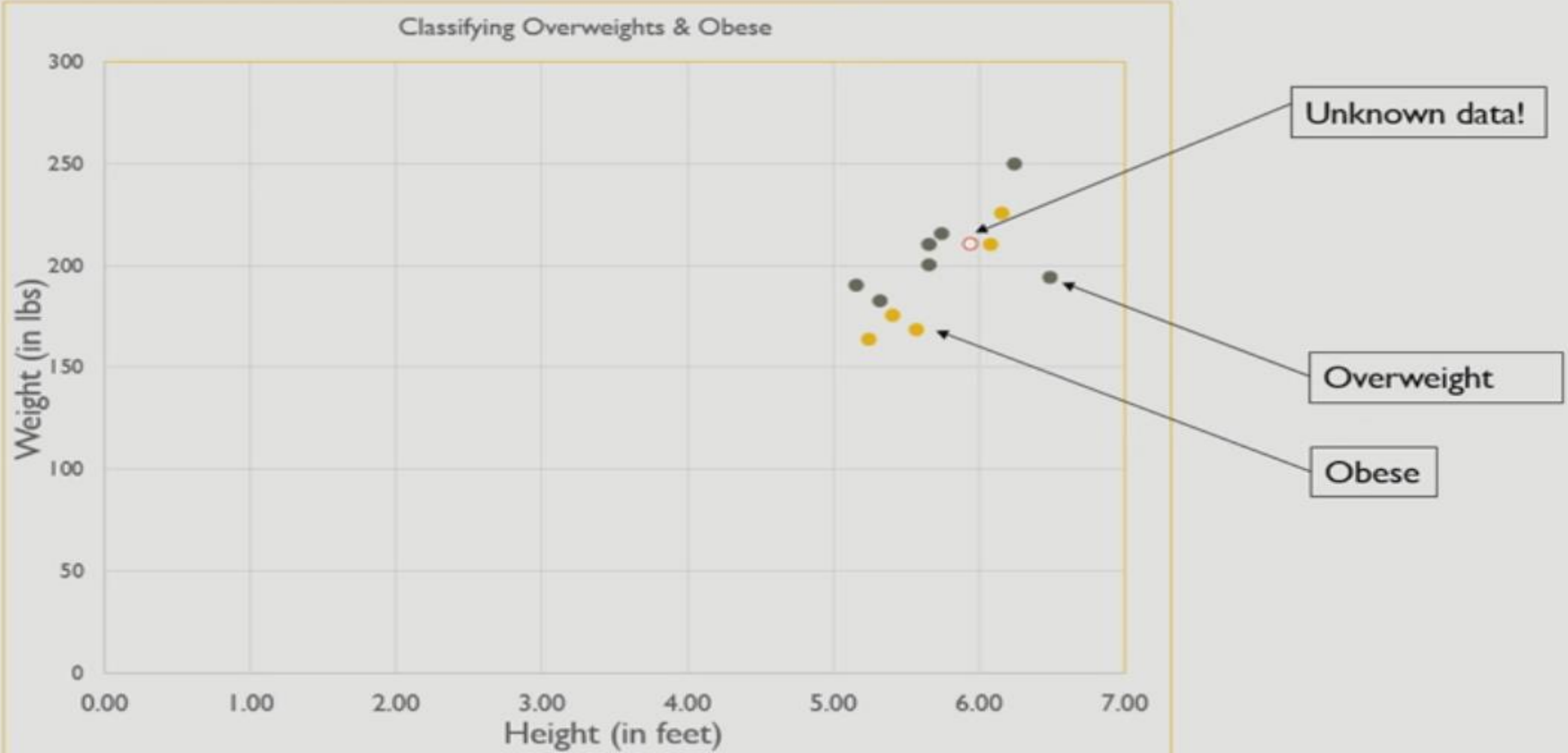
Height (feet)	Weight (pound)	Obesity
5.33	182	Obese
5.17	190	Obese
6.50	193	Overweight
5.67	210	Obese
6.17	225	Overweight
5.58	168	Overweight
5.75	215	Obese
6.25	249	Obese
6.08	210	Overweight
5.25	163	Overweight
5.42	175	Overweight
5.67	200	Obese

6 obese
&
6 overweight

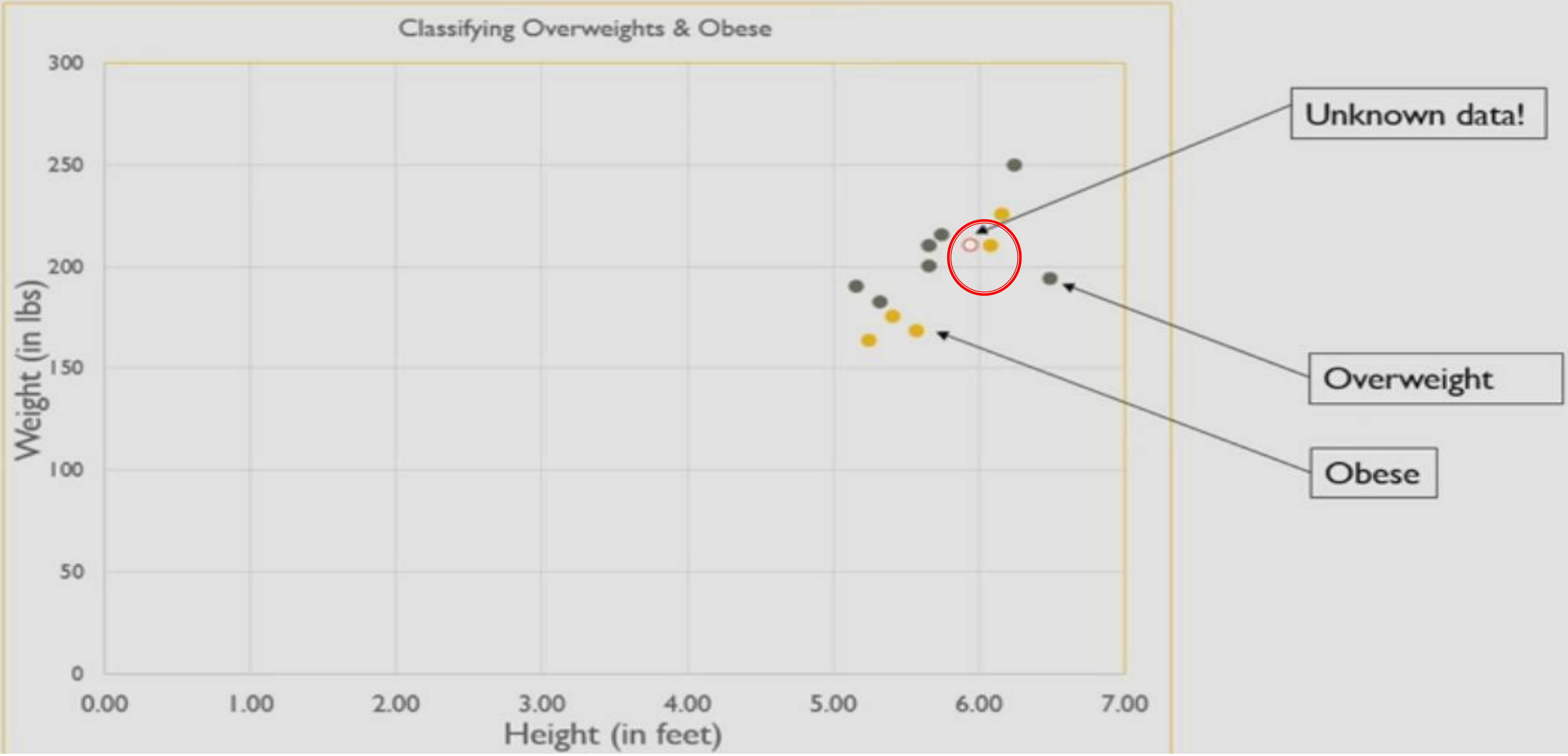
New Data

5.95	210	?

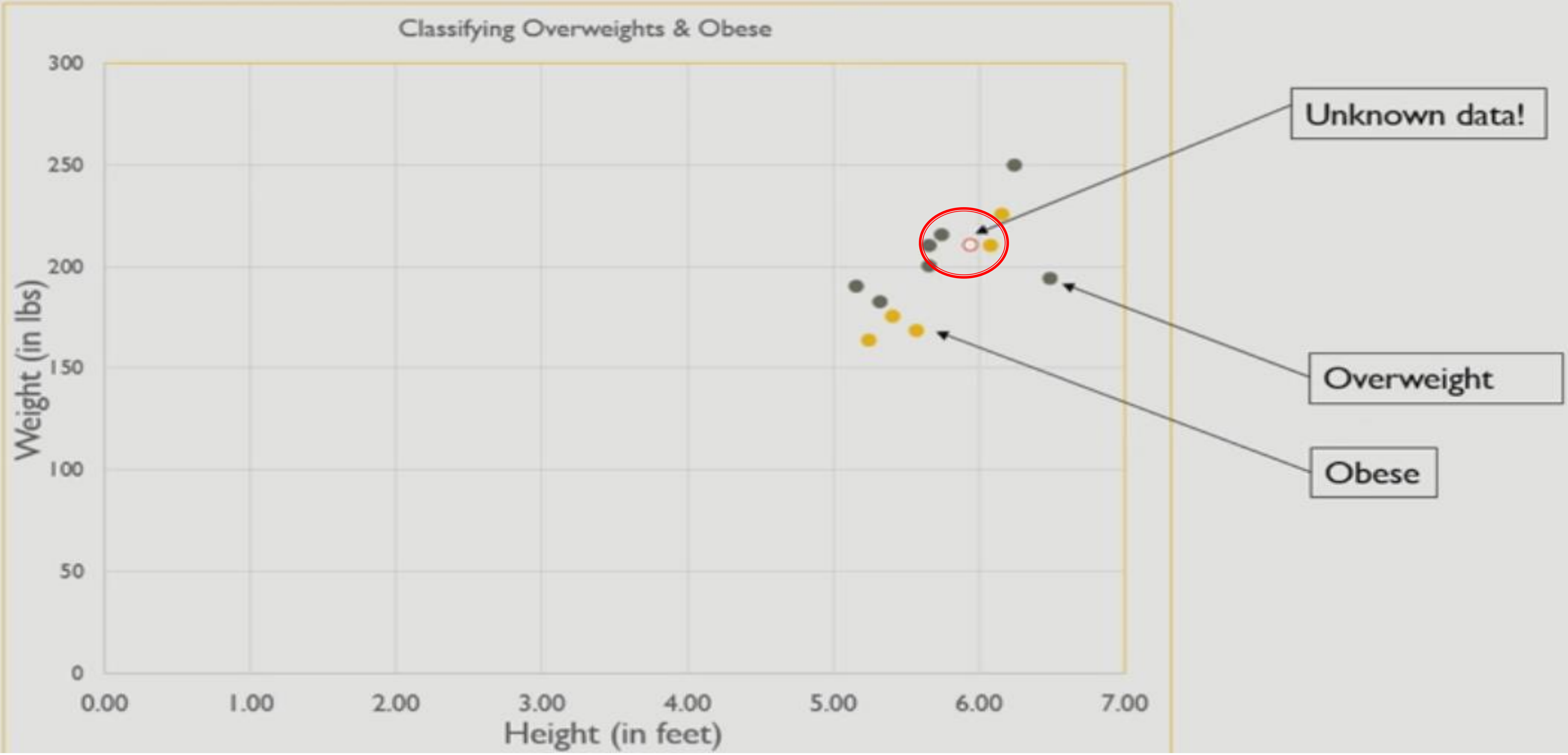
VISUALIZING THE DATA



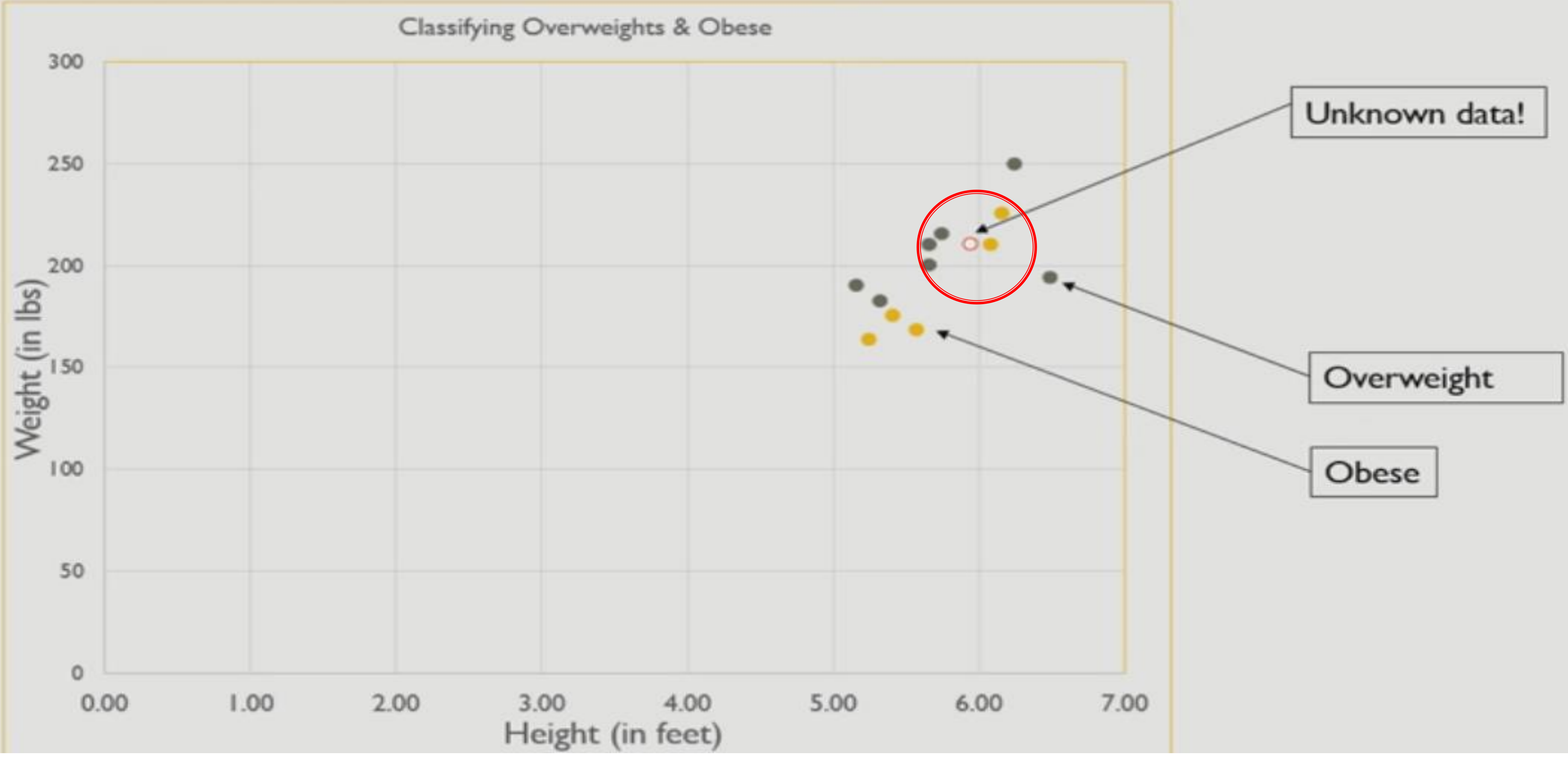
VISUALIZING THE DATA



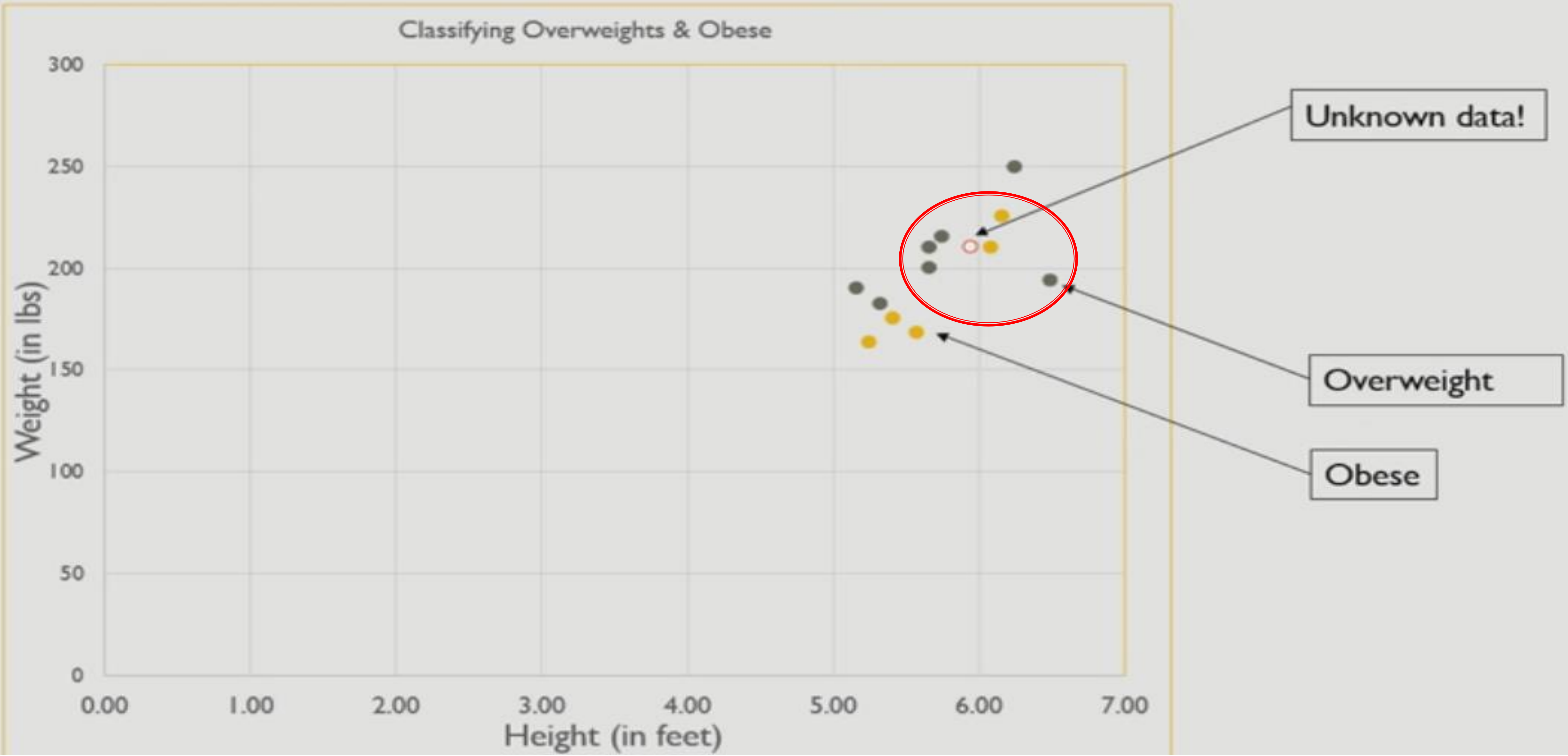
VISUALIZING THE DATA



VISUALIZING THE DATA

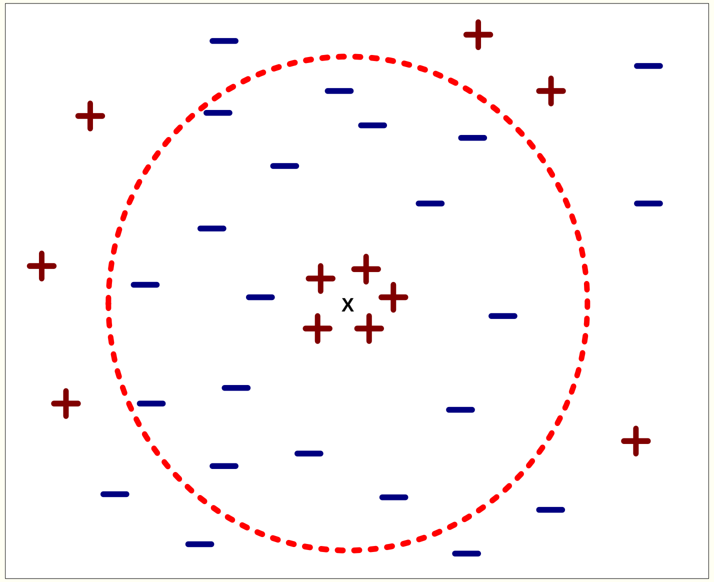


VISUALIZING THE DATA



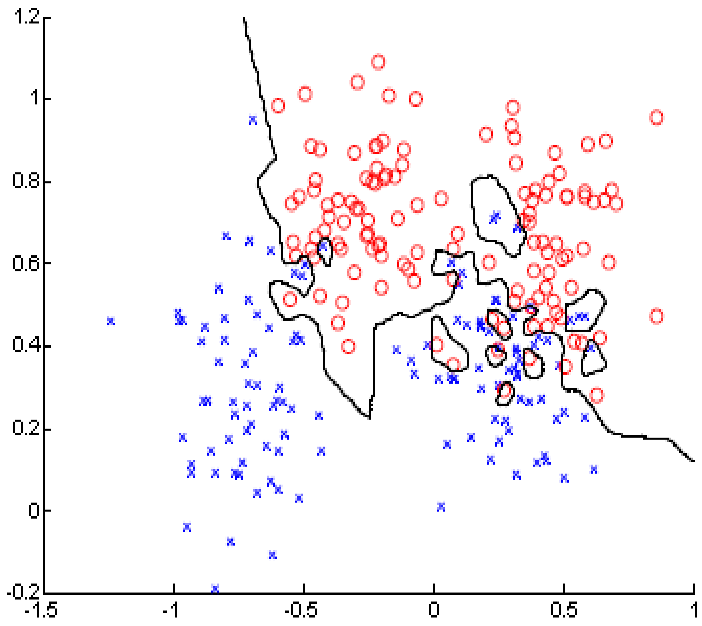
Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



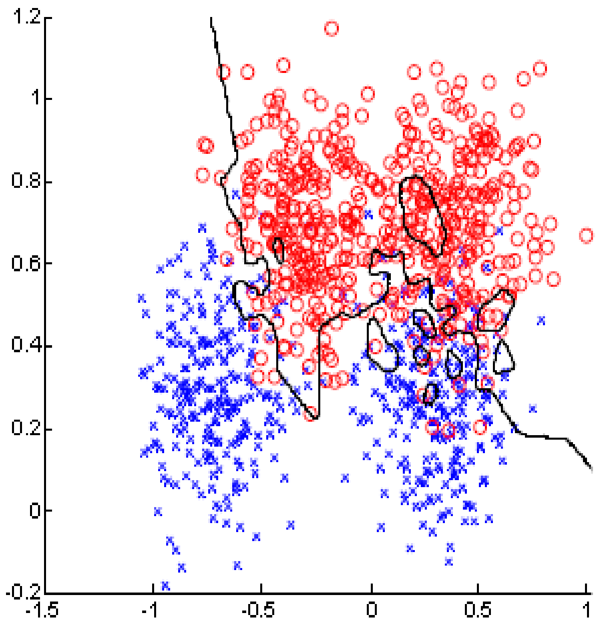
K = 1

Training data



error = 0.0

Testing data



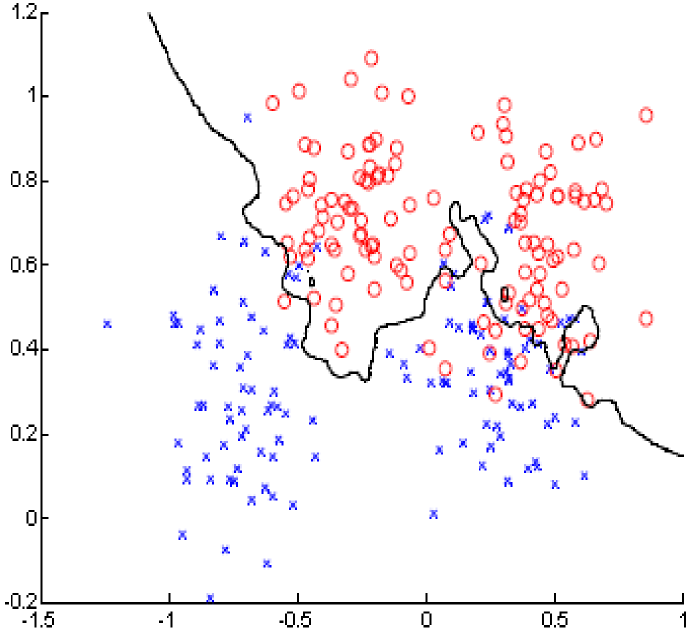
error = 0.15

K = 3


Training data
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-1.5
-1
-0.5
0
0.5
1
error = 0.0760
Testing data
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-1.5
-1
-0.5
0
0.5
1
error = 0.1340

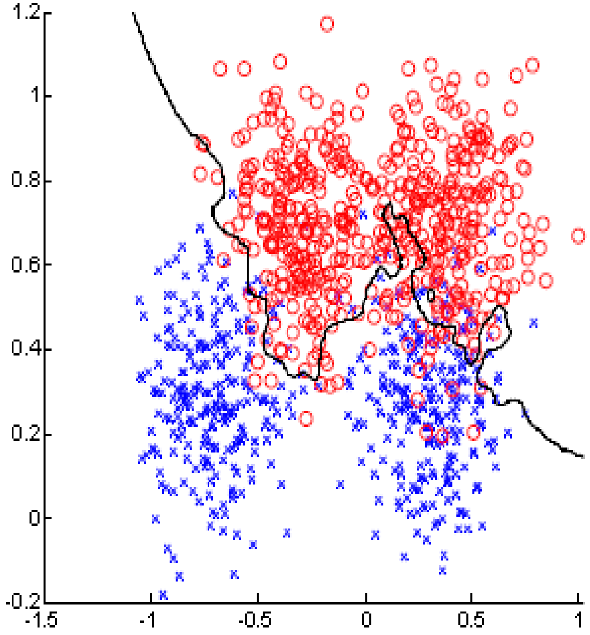
K = 7

Training data



error = 0.1320

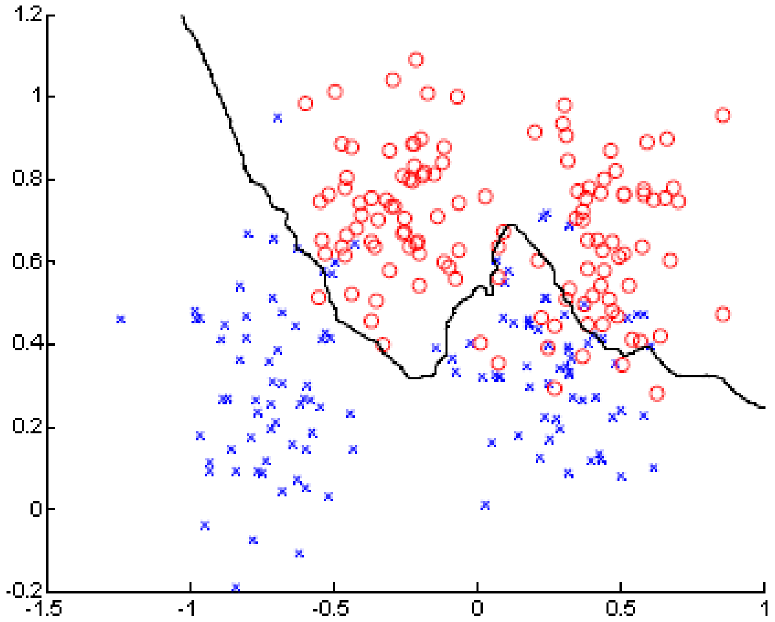
Testing data



error = 0.1110

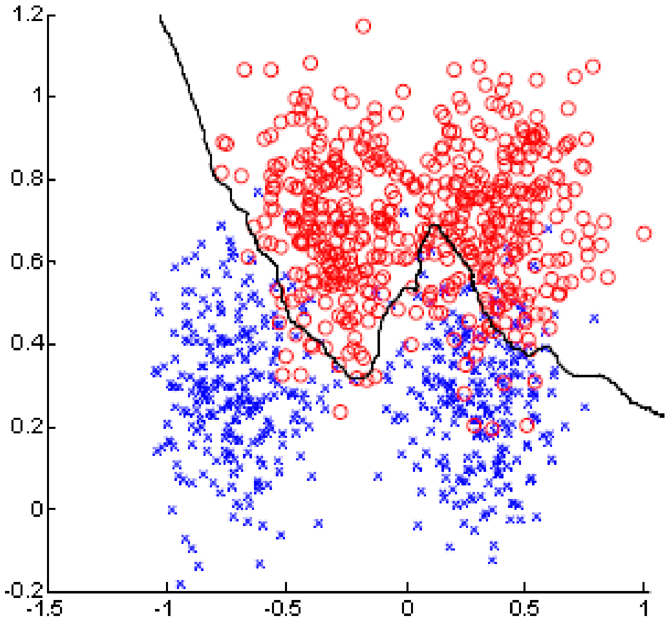
K = 21

Training data



error = 0.1120

Testing data



error = 0.0920

How to calculate Error Rate

Consider the following example

Sample	Sepal Length	Sepal Width	Label
1	5.1	3.5	Setosa
2	4.9	3.0	Setosa
3	6.2	2.9	Versicolor
4	5.5	2.4	Versicolor
5	5.7	3.6	Virginica

- It is required to use the following test sample with sepal length 5.8 and sepal width 3.2.
- Set k=3.
- Actual Predicted Value is “Versicolor”

How to calculate Error Rate (cont.)

1.Calculate distances: Calculate the distances between the test sample and all the training samples using a distance metric such as Euclidean distance.

1. Distance to Sample 1: sqrt($(5.8 - 5.1)^2 + (3.2 - 3.5)^2$)= 0.707
2. Distance to Sample 2: sqrt($(5.8 - 4.9)^2 + (3.2 - 3.0)^2$) = 0.949
3. Distance to Sample 3: sqrt($(5.8 - 6.2)^2 + (3.2 - 2.9)^2$) = 0.316
4. Distance to Sample 4: sqrt($(5.8 - 5.5)^2 + (3.2 - 2.4)^2$) = 0.806
5. Distance to Sample 5: sqrt($(5.8 - 5.7)^2 + (3.2 - 3.6)^2$) = 0.447

2. Find the k nearest neighbors: Select the k training samples with the closest distances to the test sample.

1. First nearest neighbor: Sample 3 (Versicolor) with a distance of 0.316
2. Second nearest neighbor: Sample 5 (Virginica) with a distance of 0.447
3. Third nearest neighbor: Sample 1 (Setosa) with a distance of 0.707

3. Determine the predicted label: Based on the majority class among the k nearest neighbors, the predicted label for the test sample is **Versicolor**.

4. Calculate the error rate: To calculate the error rate, compare the predicted label to the actual label of the test sample.

1. Actual label: Versicolor
2. Predicted label: Versicolor
3. The error rate is 0%.

Example

RID	Income($000's)	lot Size (000's sq.ft)	class: Owners =1 Non-Owners=2
1	60	18.4	1
2	85.5	16.8	1
3	64.8	21.6	1
4	61.5	20.8	1
5	87	23.6	1
6	110.1	19.2	1
7	108	17.6	1
8	82.8	22.4	1
9	69	20	1
10	93	20.8	1
11	51	22	1
12	81	20	2
13	75	19.6	2
14	52.8	20.8	2
15	64.8	17.2	2
16	43.2	20.4	2
17	84	17.6	2
18	49.2	17.6	2
19	59.4	16	2
20	66	18.4	2
21	47.4	16.4	2
22	33	18.8	2
23	51	14	2
24	63	14.8	2

mower

We randomly divide the data into

18 training cases

6 test cases:
tuples 6,7,12,14,19, 20

Use training cases to classify test cases and compute error rates

Example from J. Gamper

Choosing k

- If we choose **k=1** we will classify in a way that is very sensitive to the local characteristics of our data

- If we choose a **large value of k** we average over a large number of data points and average out the variability due to the noise associated with data points

- If we choose **k=18** we would simply predict the most frequent class in the data set in all cases
  - → Very stable but completely ignores the information in the independent variables

k	**1**	**3**	**5**	**7**	**9**	**11**	**13**	**18**
Misclassification error %	33	33	33	33	33	17	17	50

  - → We would choose k=11 (or possibly 13) in this case

Slide from J. Gamper

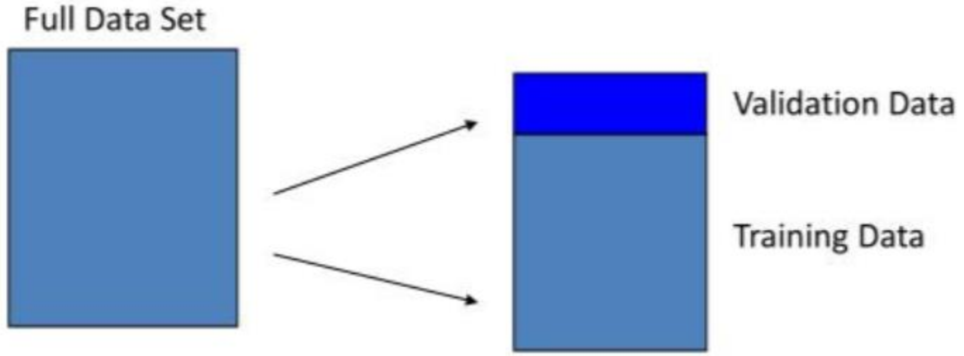
How to choose K

1. Choose an **odd K value** for the two classes
2. K must **not** be a ***multiple of the number of the classes***
3. If K too small , then the nearest neighbor classifier may be susceptible to **over fitting because of noise in training data**
4. If K too big , then the nearest neighbor classifier may mis-classify the test instance because its list of nearest neighbor may include data points that are located far away from the neighbor
5. Usually a value between 5-10 is taken as reasonable value of K.
6. Choose (learn) K by cross-validation
   - Split training data into training and validation
   - Hold out validation data and measure error on this

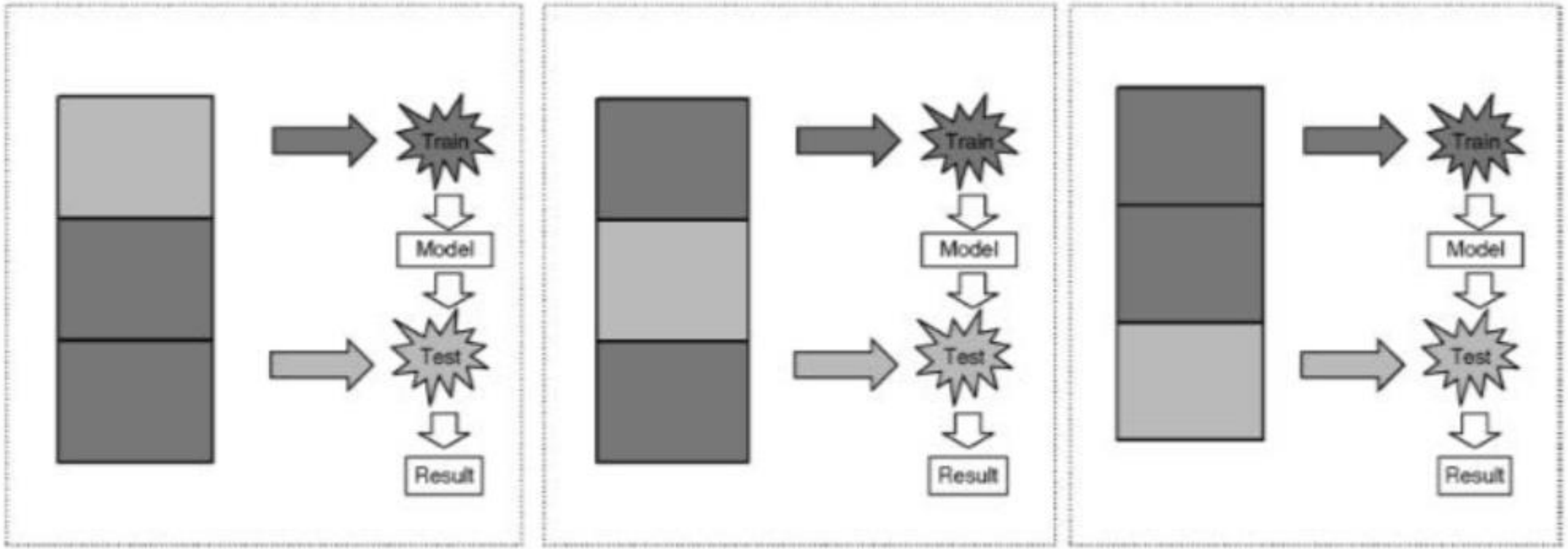
Cross-validation

- *K-fold cross-validation* avoids overlapping test sets
  - First step: split data into *k* subsets of equal size
  - Second step: use each subset in turn for testing, the remainder for training
  - This means the learning algorithm is applied to *k* different training sets
- Often the subsets are stratified before the cross-validation is performed to yield stratified *k*-fold cross-validation
- The error estimates are averaged to yield an overall error estimate; also, standard deviation is often computed
- Alternatively, predictions and actual target values from the *k* folds are pooled to compute one estimate
  - Does not yield an estimate of standard deviation

Disjoint Validation Data Sets



k-fold Cross Validation



More on cross-validation

- Standard method for evaluation: stratified ten-fold cross-validation
- Why ten?
  - Extensive experiments have shown that this is the best choice to get an accurate estimate
  - There is also some theoretical evidence for this
- Stratification reduces the estimate's variance
- Even better: repeated stratified cross-validation
  - E.g., ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

What about Distances between Non-numeric Data? Consider Strings…

Hamming distance (number of characters that are different)

ABCDE vs AGDDF → 3

Edit distance (number of character inserts/replacements/deletes to go from one to the other)

ROBOT vs BOT → 2

Jaccard distance between sets $\frac{|A \cap B|}{|A \cup B|}$

How to Handle continuous output

- There are two approaches that could be used to label test cases in case of continuous output:
  - Calculate average of the labels of the k nearest neighbors.
  - Assign weights to the neighbors based on their distances and calculate a weighted average.

- Average: Sum up the labels of the k nearest neighbors and divide it by k to obtain the average label. This average value will be the predicted label for the test case.

- Weighted average: Assign weights to the neighbors based on their distances.
  - Closer neighbors have higher weights, indicating their higher influence on the prediction.
  - The weights are inversely proportional to the distances.
  - The weighted average will be the predicted label for the test case.

How to calculate label in Regression

Suppose we have a dataset of housing prices with two features: square footage (independent variable) and price (dependent variable).

It is required to predict the price for a new house with a square footage of **1100.** We'll use KNN regression with **k=3** to label this test case.

Square Footage (Feature)	Price (Label)
1000	200,000
1500	250,000
1200	230,000
1800	300,000
900	180,000

How to calculate label in Regression

1.Calculate distances: We need to calculate the distances between the test case (1100 square footage) and all the training examples.

1. Distance to the first training example (1000 square footage): |1100 - 1000| = 100
2. Distance to the second training example (1500 square footage): |1100 - 1500| = 400
3. Distance to the third training example (1200 square footage): |1100 - 1200| = 100
4. Distance to the fourth training example (1800 square footage): |1100 - 1800| = 700
5. Distance to the fifth training example (900 square footage): |1100 - 900| = 200

2- As K=3, select the three training examples with the closest distances to the test case:

- First nearest neighbor: 1000 square footage (distance: 100)
- Second nearest neighbor: 1200 square footage (distance: 100)
- Third nearest neighbor: 900 square footage (distance: 200)

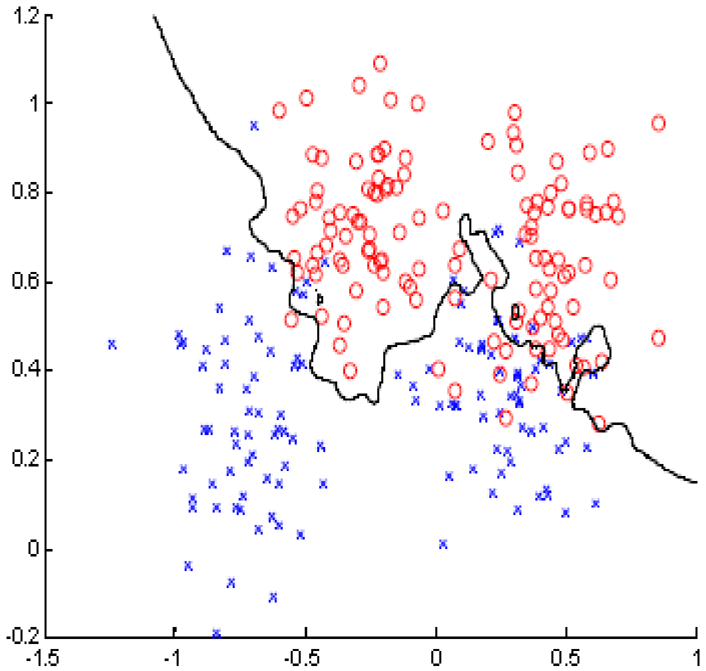
3- Calculate the predicted label:

Using average: Sum up the prices of the three nearest neighbors and divide it by 3 to obtain the average price.

Average price = (200,000 + 230,000 + 180,000) / 3 = 203,333.33

Advantages:

- K-NN is a simple but effective classification procedure
- Applies to multi-class classification
- Decision surfaces are non-linear
- Quality of predictions automatically improves with more training data
- Only a single parameter, K; easily tuned by cross-validation



Disadvantage

- - affected by local structure
- - sensitive to noise, irrelevant features
- - computationally expensive O(nd)
- - large memory requirements