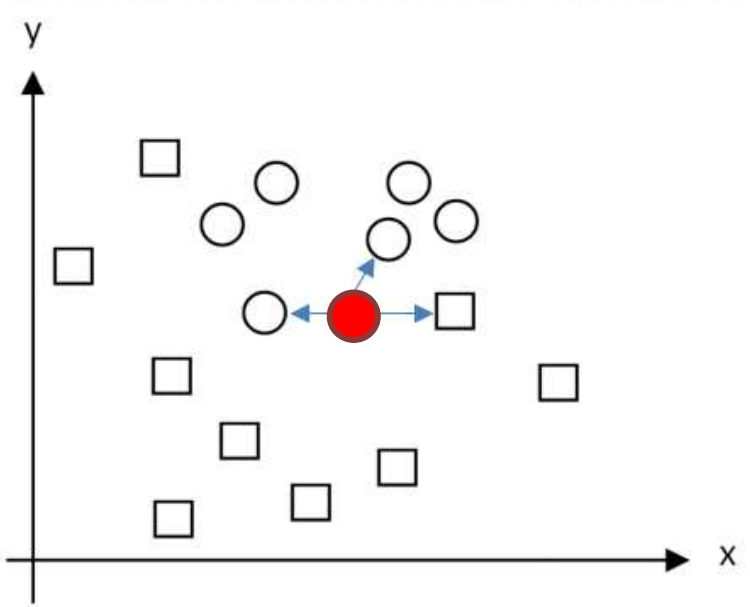
K-Nearest Neighbors (KNN)

- The k-nearest neighbor algorithm (k-NN) is a non-parametric supervised learning method that was introduced in 1951.
- It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data.
  - Unlike Naïve Bayes
- When there is little or no prior knowledge about the distribution of the data, KNN classification should be used
- KNN applications:
  - Recommendation system
  - Face recognition

K-Nearest Neighbors (KNN)

- The training samples are vectors in a multidimensional feature space.
  - Each sample is labeled with a class label
- We place the test (unknown/unlabeled) data in multidimensional feature space.
- Test data will be classified based on the k nearest neighbor in the multidimensional feature space.
  - k is defined by the user
- To calculate distance metric for continuous values, we use Euclidean Distance.



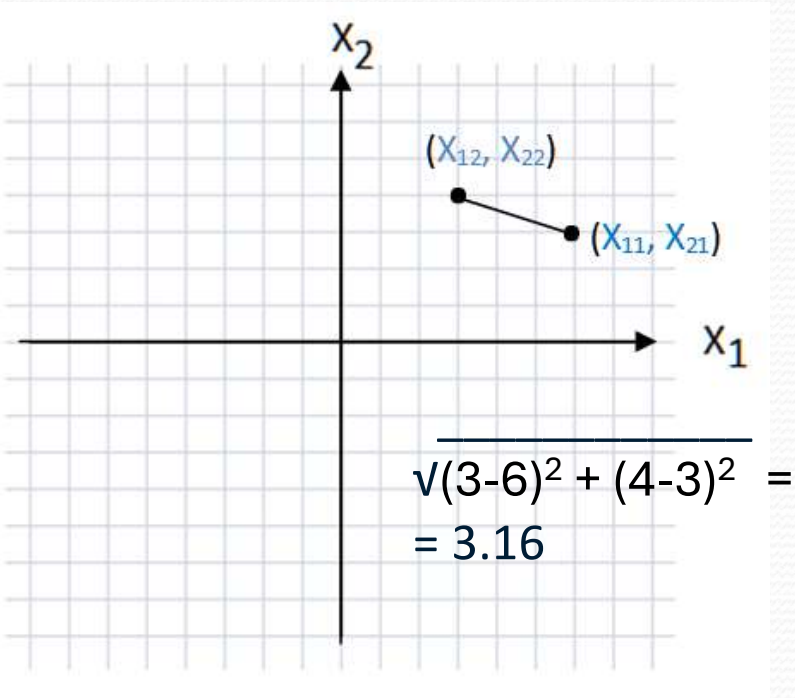
K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN)

To calculate the distance between two data points $X_1 = (x_{11}, x_{12}, \dots, x_{1n})$ and $X_2 = (x_{21}, x_{22}, \dots, x_{2n})$, we use Euclidian Distance:

$$d(X1, X2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

the square root of the sum of the squares of the differences between the attribute values (coordinates) of the points

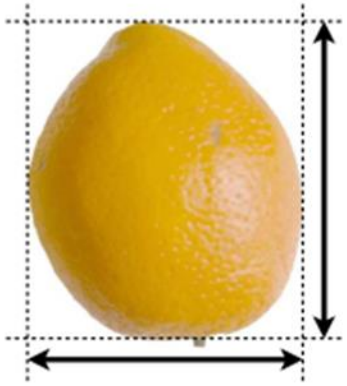


The Fruit Dataset

A bucket of fruits

The fruit dataset was created by Dr. Iain Murray at the University of Edinburgh. He bought a few dozen oranges, lemons and apples, and recorded their features in a table.

4 classes: {1:apple, 2:mandarin, 3:orange, 4:lemon}



The fruit dataset (a table)
Each row contains the information of a fruit sample/instance

fruit label	fruit_name	subtype	mass (g)	width (cm)	height (cm)	color_score
1	apple	granny_smith	192	8.4	7.3	0.55
4	lemon	spanish_belsan	194	7.2	10.3	0.70

In this table: what is input x? what is output y?

http://usapple.org/the-industry/apple-varieties/

In total, there are 59 fruit samples (i.e. 59 rows) in the table

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93

X[0], or X[0,:]

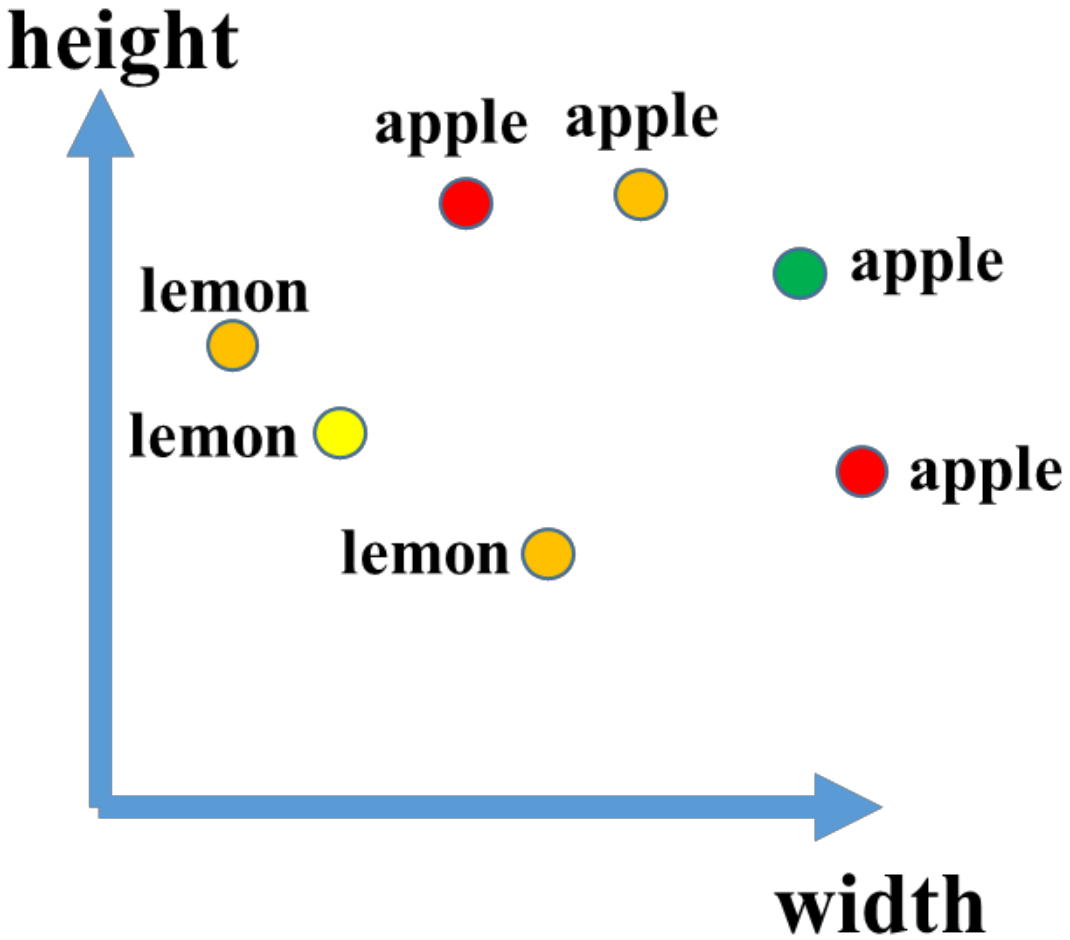
X[1], or X[1,:]

4 classes: {1:apple, 2:mandarin, 3:orange, 4:lemon}

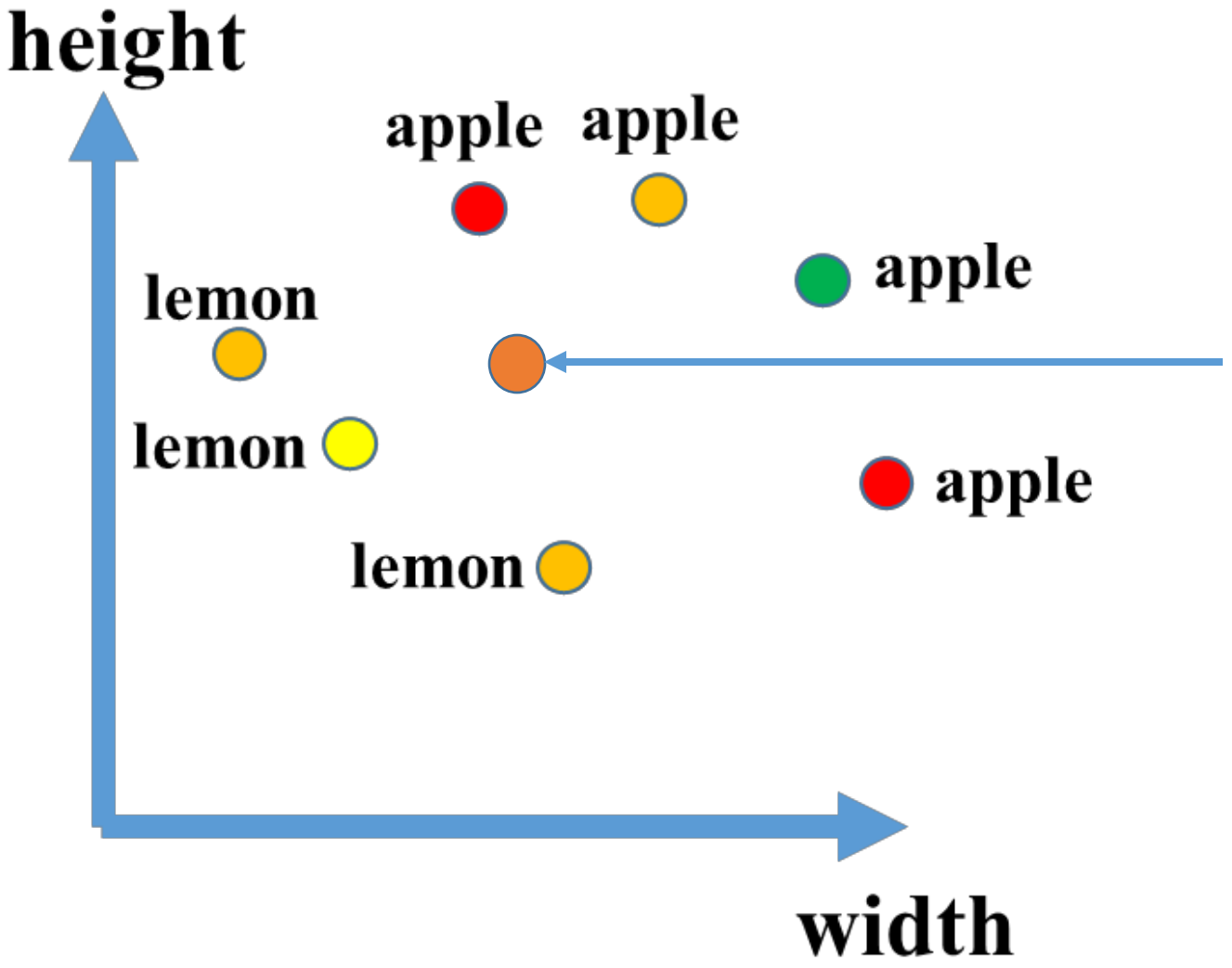
KNN classifier (K-Nearest Neighbor)

- A KNN classifier. The user needs to:

  (1) choose the value of K and (2) choose a distance measure



7 samples in the training set

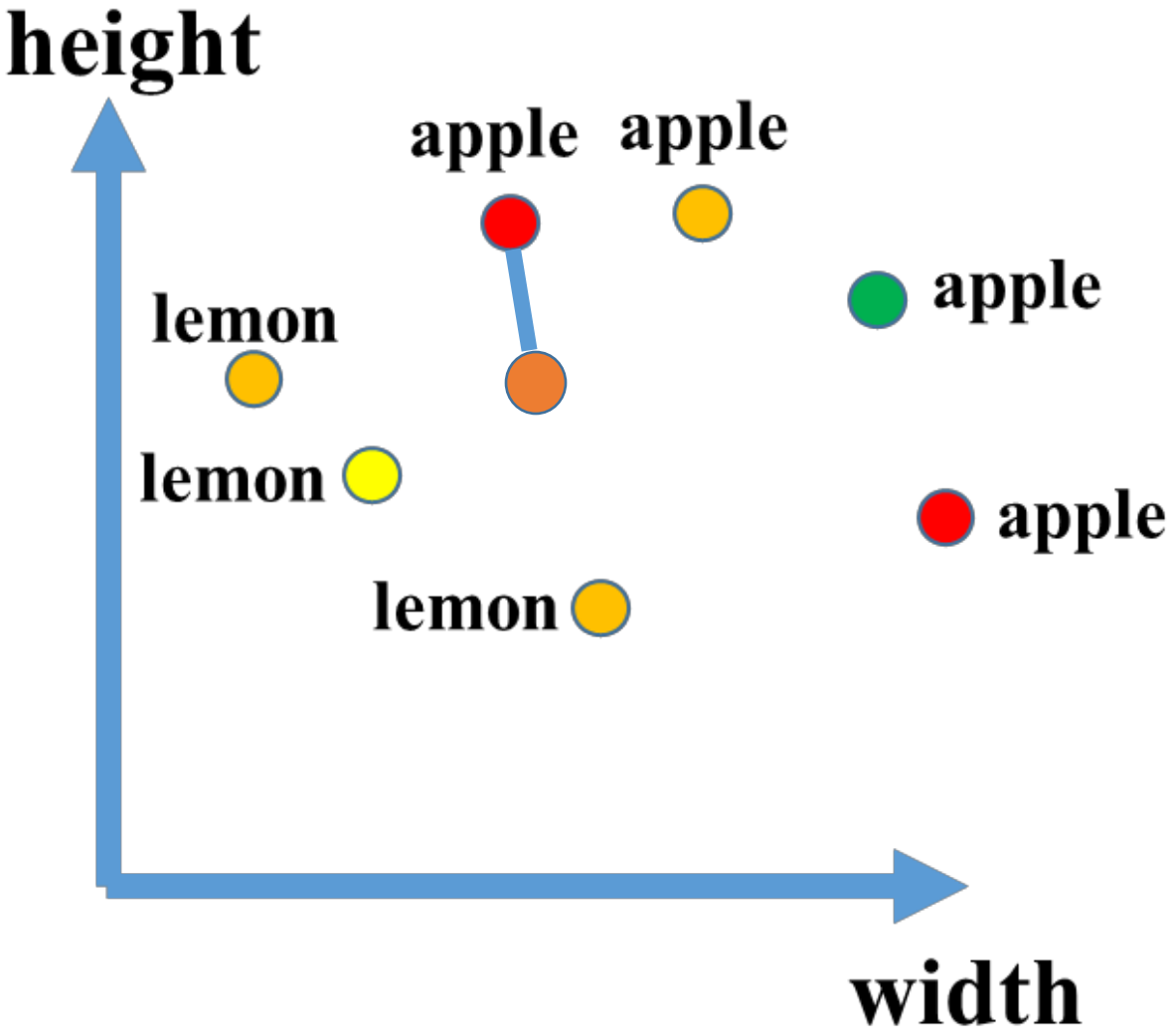


a new sample, class label is unknown

7 samples in training set

Let's set K=**1** and use Euclidean distance measure

Task: Find the nearest neighbor in the training set (by comparing distances)



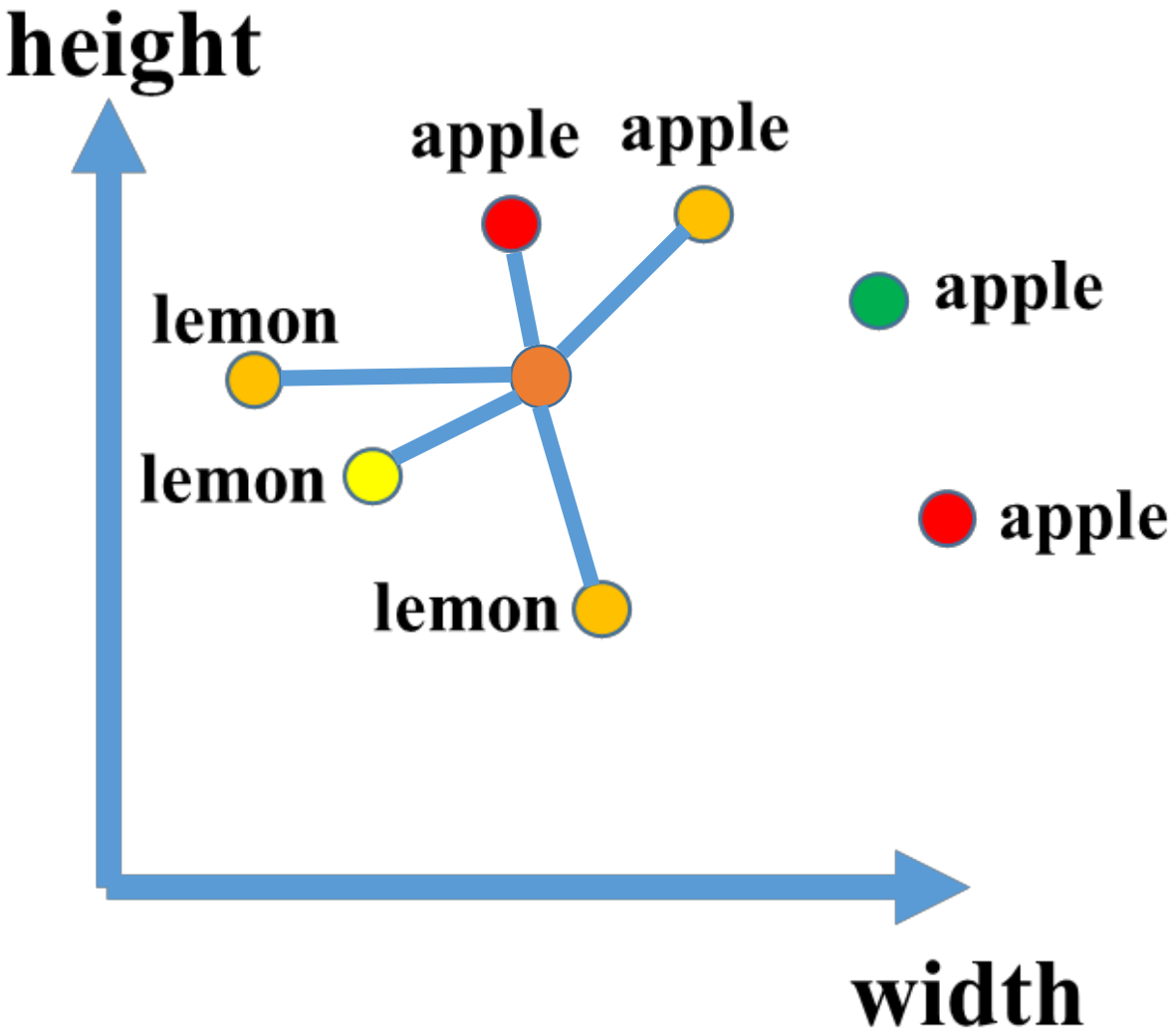
7 samples in training set

the nearest neighbor in the training set is an apple, therefore the KNN classifier will classify the input as an apple

● is classified as an apple because its nearest neighbor is an apple

Let's set **K=5** and use Euclidean distance measure

Task: Find the **5** nearest neighbors in the training set



7 samples in training set

Among the **5** nearest neighbors in the training set, there are 3 lemons and 2 apples, therefore, based on **majority vote**, the KNN classifier will classify the input as a lemon

● is classified as a lemon because the majority of its *K* nearest neighbors are lemons

K-Nearest Neighbor (KNN)

- For the best k value, you may need to do multiple attempts
- You can increase the k value by 1 each time
- The k value when you have the minimum error rate will be your most optimal k value
- Usually when the data size increases, the k value increases as well
- Since the KNN algorithm uses distance metrics, it would be affected by noise

K-Nearest Neighbor (KNN)

The KNN Algorithm:

1. Initialize K to your chosen number of neighbors
2. For each example in the data
   - Calculate the distance between the query example and the current example from the data.
   - Add the distance and the index of the example to an ordered collection
3. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances
4. Pick the first K entries from the sorted collection
5. Get the labels of the selected K entries
6. return the mode of the K labels

K-Nearest Neighbor (KNN)

- **Example 1:**
  - Assume we have the following data with X1 (Acid Durability) & X2 (Strength) attributes, and C is the class value for Paper Quality (0: low and 1: high)
  - Using KNN, find the class for the unlabeled data (X1=8, X2=4)
  - Choose k=3.

X1	X2	C
2	4	0
3	6	1
3	4	1
4	10	0
5	8	0
6	3	1
7	9	1
9	7	0
11	7	0
10	2	0
8	4	?

K-Nearest Neighbor (KNN)

- Step 1: k =3
- Step 2 : For each existing data point, calculate the distance between (8,4) and the data point

$$d(X1, X2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

X1	X2	C
2	4	0
3	6	1
3	4	1
4	10	0
5	8	0
6	3	1
7	9	1
9	7	0
11	7	0
10	2	0

K-Nearest Neighbor (KNN)

- Step 2

$$d(X1, X2) = \sqrt{(2-8)^2 + (4-4)^2} = 6$$
$$d(X1, X2) = \sqrt{(3-8)^2 + (6-4)^2} = 5.39$$
$$d(X1, X2) = \sqrt{(3-8)^2 + (4-4)^2} = 5$$
$$d(X1, X2) = \sqrt{(4-8)^2 + (10-4)^2} = 7.21$$
$$d(X1, X2) = \sqrt{(5-8)^2 + (8-4)^2} = 5$$
$$d(X1, X2) = \sqrt{(6-8)^2 + (3-4)^2} = 2.24$$
$$d(X1, X2) = \sqrt{(7-8)^2 + (9-4)^2} = 5.10$$
$$d(X1, X2) = \sqrt{(9-8)^2 + (7-4)^2} = 3.16$$
$$d(X1, X2) = \sqrt{(11-8)^2 + (7-4)^2} = 4.24$$
$$d(X1, X2) = \sqrt{(10-8)^2 + (2-4)^2} = 2.83$$

X1	X2	C
2	4	0
3	6	1
3	4	1
4	10	0
5	8	0
6	3	1
7	9	1
9	7	0
11	7	0
10	2	0

K-Nearest Neighbor (KNN)

- Step 2

X1	X2	distance	C
2	4	6	0
3	6	5.39	1
3	4	5	1
4	10	7.21	0
5	8	5	0
6	3	2.24	1
7	9	5.1	1
9	7	3.16	0
11	7	4.24	0
10	2	2.83	0

K-Nearest Neighbor (KNN)

- Step 3: Sort the distances in ascending order
- Step 4: Pick the k entries

X1	X2	distance	C
6	3	2.24	1
10	2	2.83	0
9	7	3.16	0
11	7	4.24	0
3	4	5	1
5	8	5	0
7	9	5.1	1
3	6	5.39	1
2	4	6	0
4	10	7.21	0

K-Nearest Neighbor (KNN)

Step 5: Get the labels of the selected K entries

- Two C = 0
- One C = 1

Step 6: return the mode of the K labels

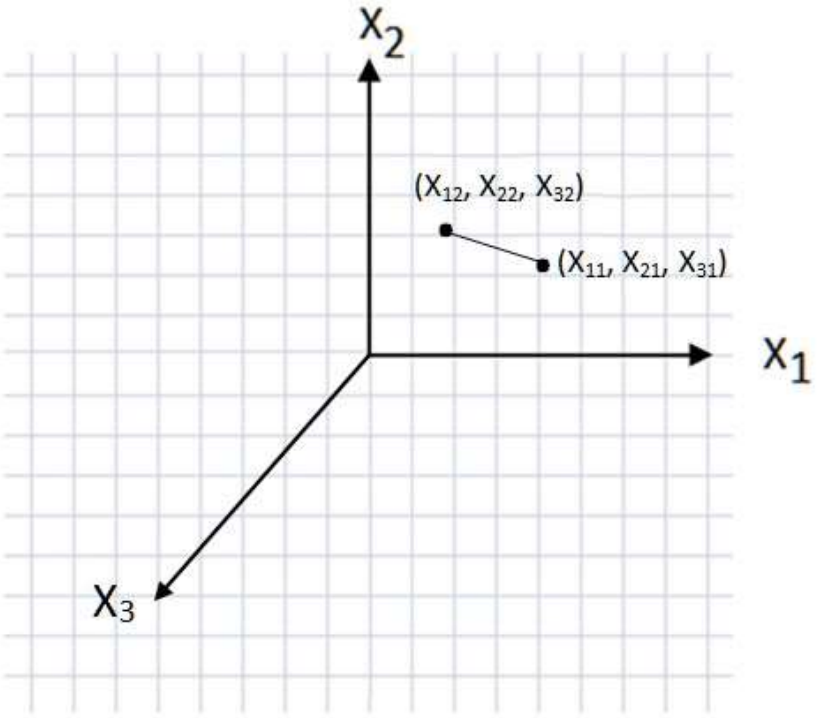
- For (8,4), C = 0

X1	X2	distance	C
6	3	2.24	1
10	2	2.83	0
9	7	3.16	0
11	7	4.24	0
3	4	5	1
5	8	5	0
7	9	5.1	1
3	6	5.39	1
2	4	6	0
4	10	7.21	0

K-Nearest Neighbor (KNN)

Example 2: Loan Application

Sample	Income	Age	# of Credit Cards	Decision
1	10K	60	3	No
2	27K	22	3	No
3	16K	38	2	No
4	96K	40	2	Yes
5	90K	40	3	Yes
6	67,5K	60	2	Yes
7	52K	20	4	No
8	90K	42	2	Yes
	37.5K	45	3	?



Pick k=3

K-Nearest Neighbor (KNN)

Sample	Income	Age	# of Credit Cards	Decision
1	10K	60	3	No
2	27K	22	3	No
3	16K	38	2	No
4	96K	40	2	Yes
5	90K	40	3	Yes
6	67,5K	60	2	Yes
7	52K	20	4	No
8	90K	42	2	Yes
	37.5K	45	3	?

$$d(X1, X2) = \sqrt{(10 - 37{,}5)^2 + (60 - 45)^2 + (3 - 3)^2} = 31{,}32$$

$$d(X1, X2) = \sqrt{(27 - 37{,}5)^2 + (22 - 45)^2 + (3 - 3)^2} = 25{,}28$$

$$d(X1, X2) = \sqrt{(16 - 37{,}5)^2 + (38 - 45)^2 + (2 - 3)^2} = 22{,}63$$

$$d(X1, X2) = \sqrt{(96 - 37{,}5)^2 + (40 - 45)^2 + (2 - 3)^2} = 58{,}72$$

$$d(X1, X2) = \sqrt{(90 - 37{,}5)^2 + (40 - 45)^2 + (3 - 3)^2} = 52{,}32$$

$$d(X1, X2) = \sqrt{(67{,}5 - 37{,}5)^2 + (60 - 45)^2 + (2 - 3)^2} = 33{,}55$$

$$d(X1, X2) = \sqrt{(52 - 37{,}5)^2 + (20 - 45)^2 + (4 - 3)^2} = 28{,}91$$

$$d(X1, X2) = \sqrt{(90 - 37{,}5)^2 + (42 - 45)^2 + (2 - 3)^2} = 52{,}59$$

K-Nearest Neighbor (KNN)

Sample	Income	Age	# of Credit Cards	Decision	Distance
3	10K	38	2	No	22.63
2	27K	22	3	No	25.28
7	16K	20	4	No	28.92
1	96K	60	3	Yes	31.32
6	90K	60	2	Yes	33.55
8	67.5K	42	2	Yes	52.59
5	52K	40	3	No	52.74
4	90K	40	2	Yes	58.72
	37,5K	45	3	No	

KNN can be used for classification and regression

- **For classification**, the output from a KNN classifier is a discrete value (class label), which is done by majority vote
- **For regression**, the output from a KNN regressor is a continues value (target value)
- **For regression**, the average target value of the *K*-nearest neighbors will be the predicted target value of the input $\boldsymbol{x}$

Assume K=3 and training samples $\boldsymbol{x_1, x_2, x_3}$ are the (K=3) nearest neighbors of $\boldsymbol{x}$, the target values are $\boldsymbol{y_1, y_2, y_3}$

Then, the predicted target value $\tilde{y}$ of $\boldsymbol{x}$ is $\boldsymbol{(y_1 + y_2 + y_3)/3}$

K-Nearest Neighbor (KNN)

- **KNN Algorithm**
  - **Strength:**
    - Very simple to understand and implement.
    - The decision boundaries can be of arbitrary shapes.
    - KNN classifier can be updated at a very little cost.
  - **Weakness:**
    - K-NN is computationally expensive.
    - It is a lazy learner (uses all the training data at the runtime, and therefore, it is slow).
    - Curse of dimensionality: distance can be dominated by irrelevant attributes