Overfitting in KNN

- When "k" is a **very small number- KNN can overfit**.
- It will classify just based on the **closest neighbors** instead of learning a good separating frontier between classes.

Underfitting in KNN

- If "k" is a **very big number -**

  KNN will **underfit**, in the limit.

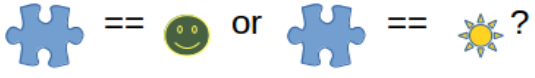
Dogs

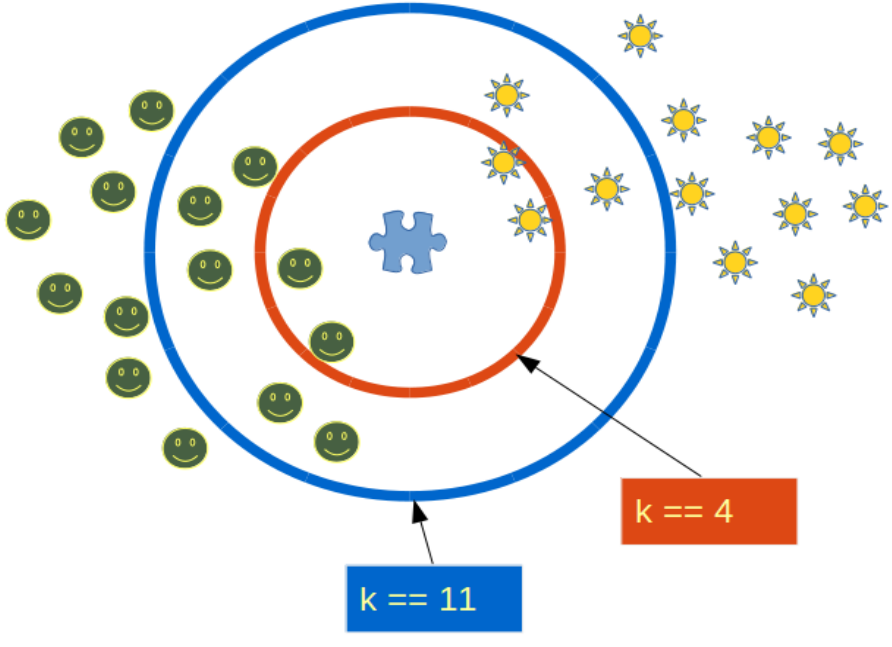
- If **k = n**, KNN will think **every point belongs** to the class **that has more samples**.

How do we choose the factor K ?

- **Sqrt(n),** where n is the total number of datapoints.
- **Odd value of k** is selected to **avoid any confusion** between two classes off data.
- In python, we will se how to use **cross validation** to choose the factor K(Later).

Challenge :



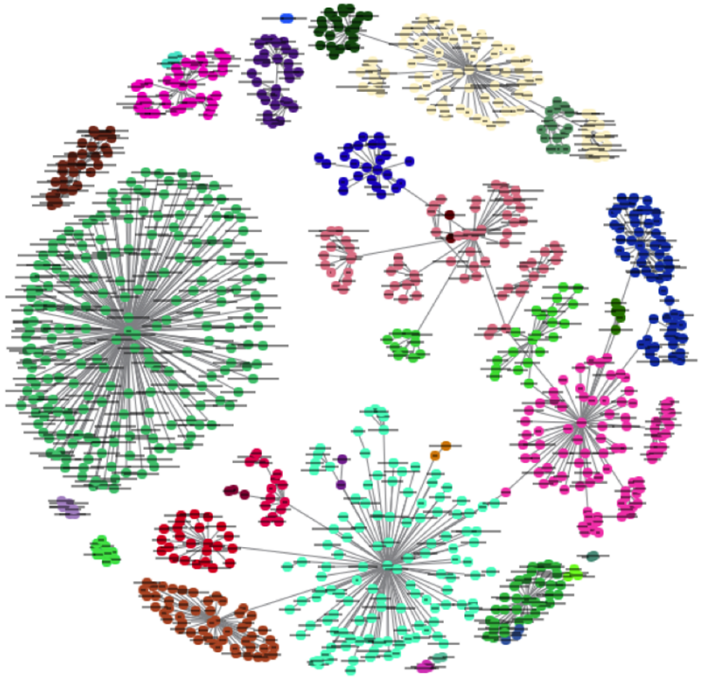



It’s a TIE

- Generally it's good to try an **odd number for k** to start out.
- This helps avoid situations where your **classifier "ties"** as a result of having the same number of votes for two different classes.
- This is particularly true if your dataset has **only two classes**.

KNN Regression

- For regression, value for the **new data point** will be **the average of the k neighbors.**



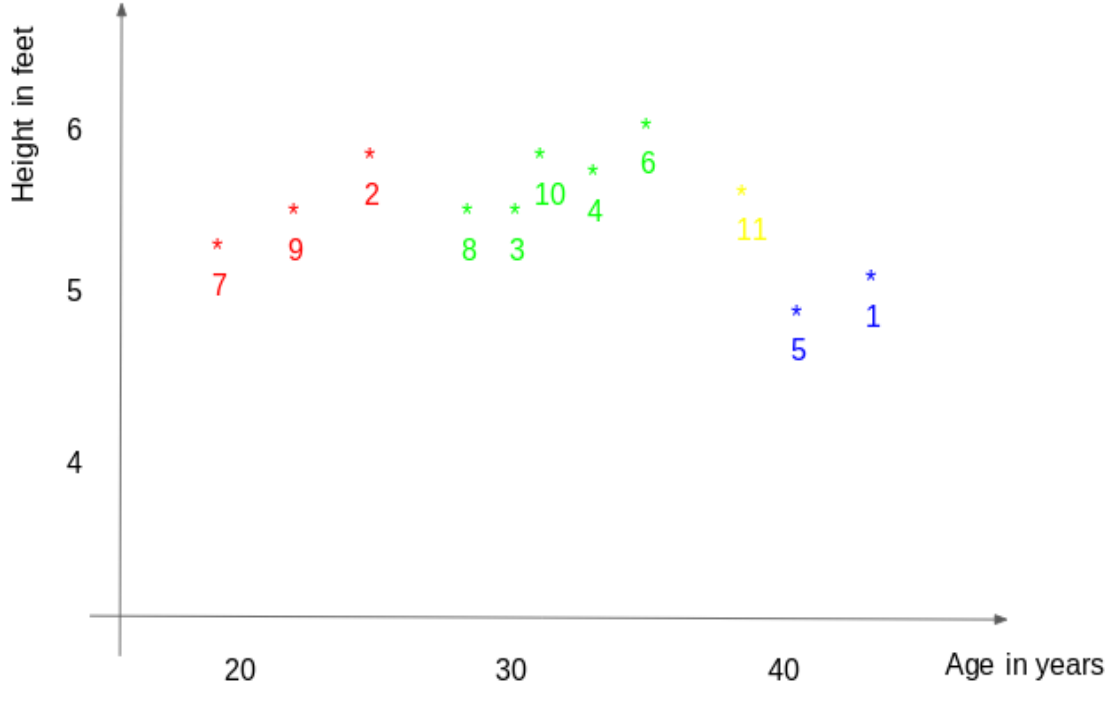
“You are average of the 5 people you spend the most time with”

Lets consider an example:

ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

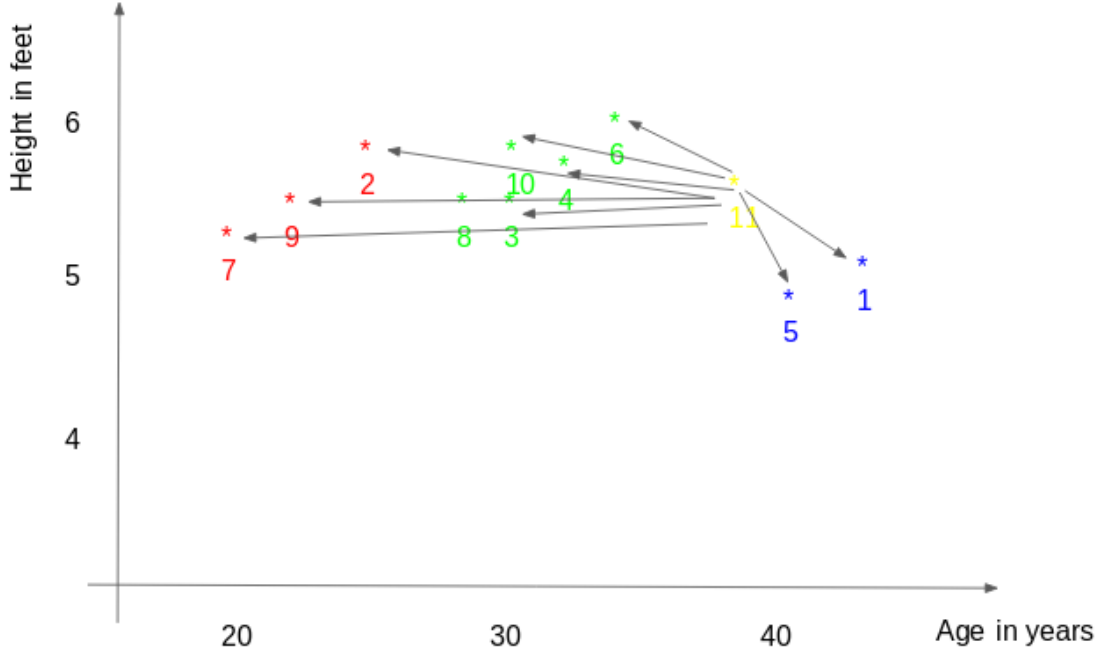
Below is the plot of height versus age from the table:

- Y-axis represents the **height of a person** (in feet)
- X-axis represents the **age** (in years).
- The points are numbered according to the **ID values**.
- The yellow point (ID 11) is our **test point**.



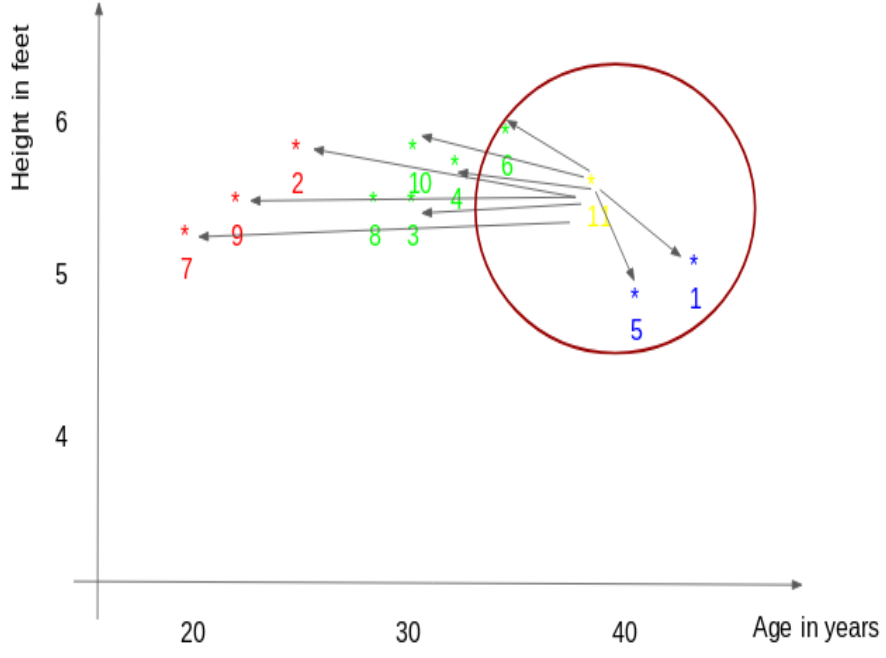
Step 1

- Distance between **the new point** and **each training point** is calculated.



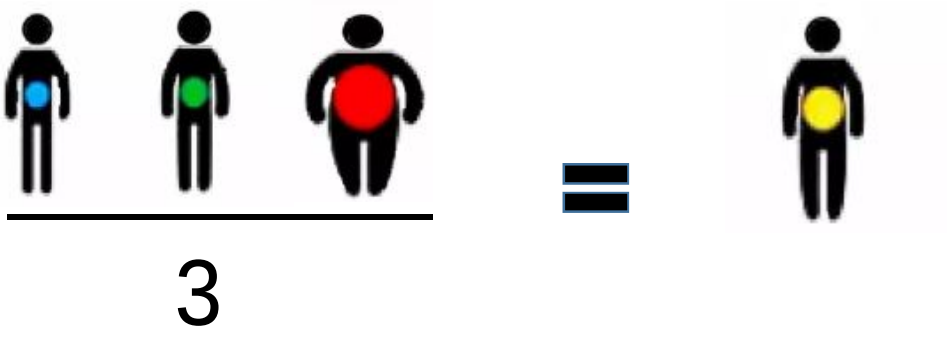
Step 2

- The **closest k data points** are selected (based on the distance).
- In this example, **points 1, 5, 6** will be selected if **value of k is 3.**



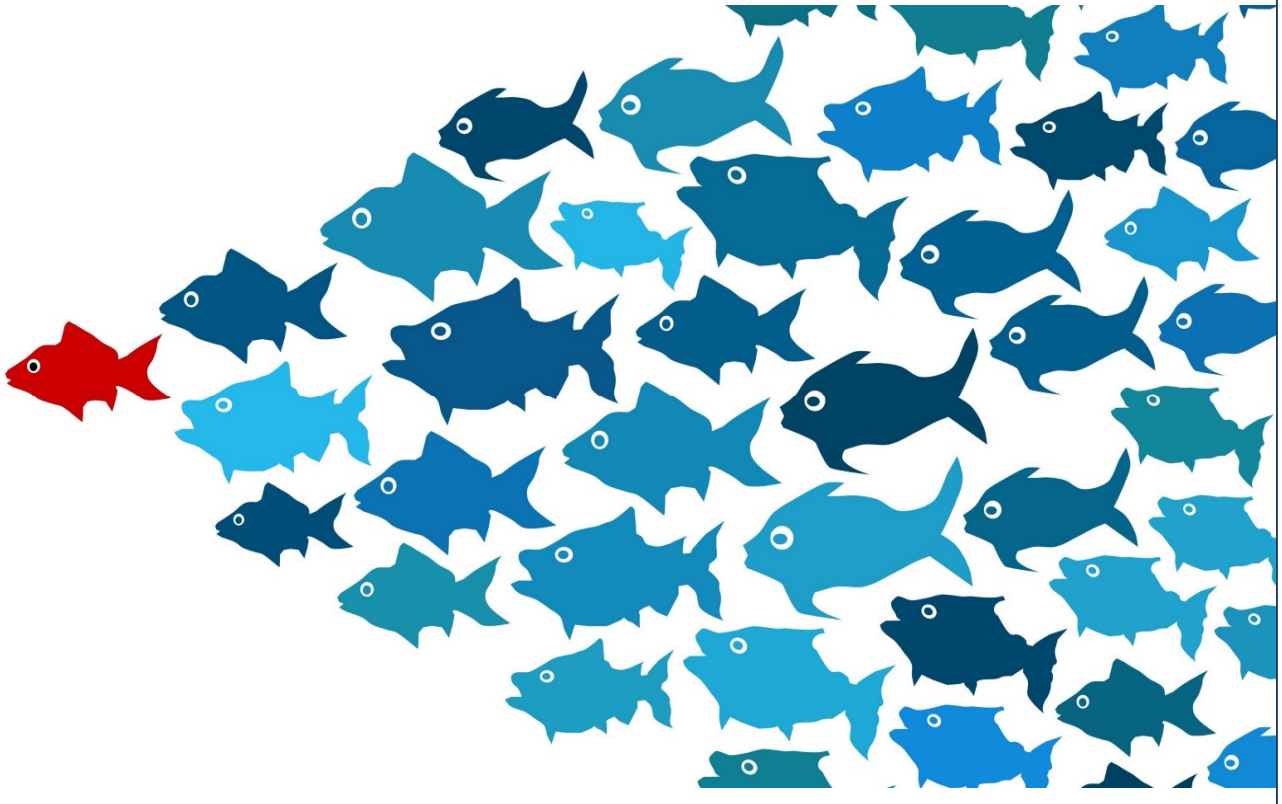
Step 3

- The **average of these data points** is the final prediction for the new point.
- Here, we have weight of ID11 = (**77+72+60)/3** = **69.66 kg**.



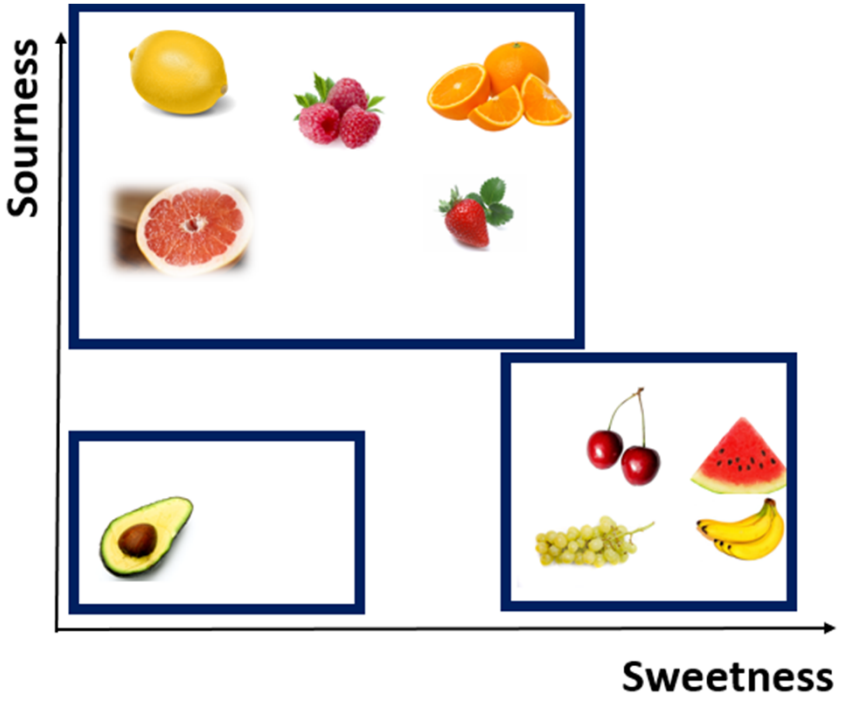
KNN Classification

- For classification, **count the number of data points** in each category among the k neighbors.
- **New data point** will belong to class that has the **most neighbors.**



Lets consider an example:

- Classifying fruits on the basis of **sweetness or sourness**



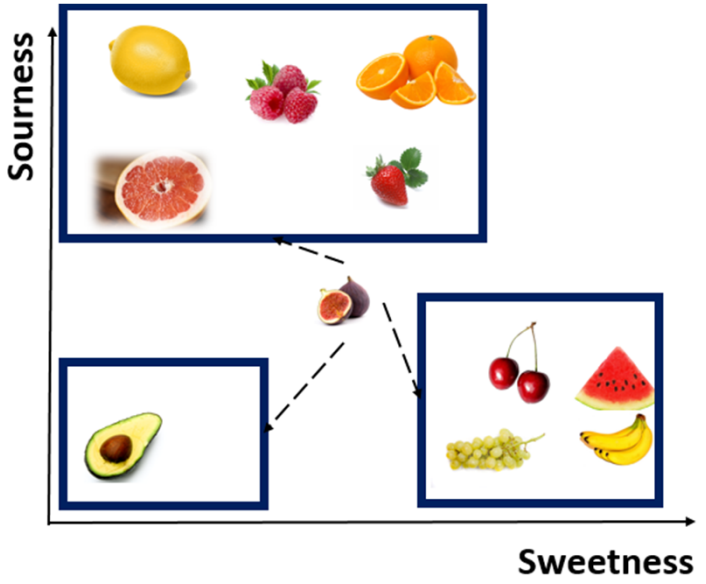
- We classified fruits on the basis of **closeness** .
- If a fruit has a high value for sourness then type is **Sour**
- If a fruit has a high value for sweetness then type is **Sweet**

Fruite	Sweetness	Sourness	Fruite Type
Lemon	1	9	Sour
Grapfruite	2	8	Sour
Orange	3	7	Sour
Rasberry	2	8	Sour
Cherry	6	4	Sweet
banana	9	1	Sweet
Grapes	8	2	Sweet
Watremelon	9	1	Sweet
Avacado	1	1	None
Strawberry	5	5	Sour

What if want to classify a new fruit ?

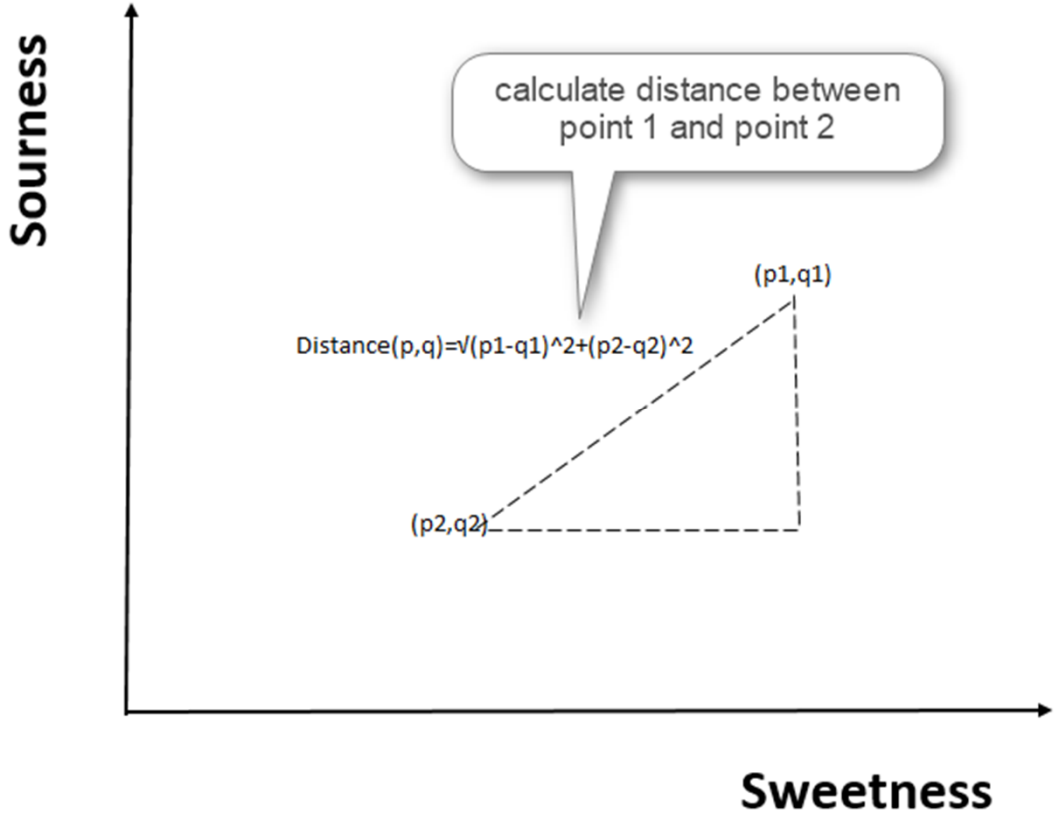
- In this case , we would find the **distance between this fruit with all the fruits.**

Fruite	Sweetnes Sourness	Fruite Typ
Fig	7	3 Sour



- **Euclidean distance** is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (y).

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



- First nearest neighbors are **Cherry and Grapes**.
- Second nearest neighbor is **Watermelon**.
- Third nearest neighbor is **Strawberry and banana**.
- Because all of these are Sweet fruits, we consider **new fruit** as a **sweet one.**

Fruite	Sweetnes	Sourness	Fruite Type	Distance to Fig
Lemon	1	9	Sour	sqrt((1-7)^2+(9-3)^2)=8.4
Grapfruite	2	8	Sour	sqrt((2-7)^2+(8-3)^2)=7.1
Orange	3	7	Sour	sqrt((3-7)^2+(7-3)^2)=5.6
Rasberry	2	8	Sour	sqrt((2-7)^2+(8-3)^2)=7.1
Cherry	6	4	Sweet	sqrt((6-7)^2+(4-3)^2)=1.41
banana	9	1	Sweet	sqrt((9-7)^2+(1-3)^2)=2.82
Grapes	8	2	Sweet	sqrt((8-7)^2+(2-3)^2)=1.41
Watremelon	9	1	Sweet	sqrt((9-7)^2+(1-3)^2)=2.44
Avacado	1	1	None	sqrt((1-7)^2+(1-3)^2)=6.3
Strawberry	5	5	Sour	sqrt((5-7)^2+(5-3)^2)=2.82