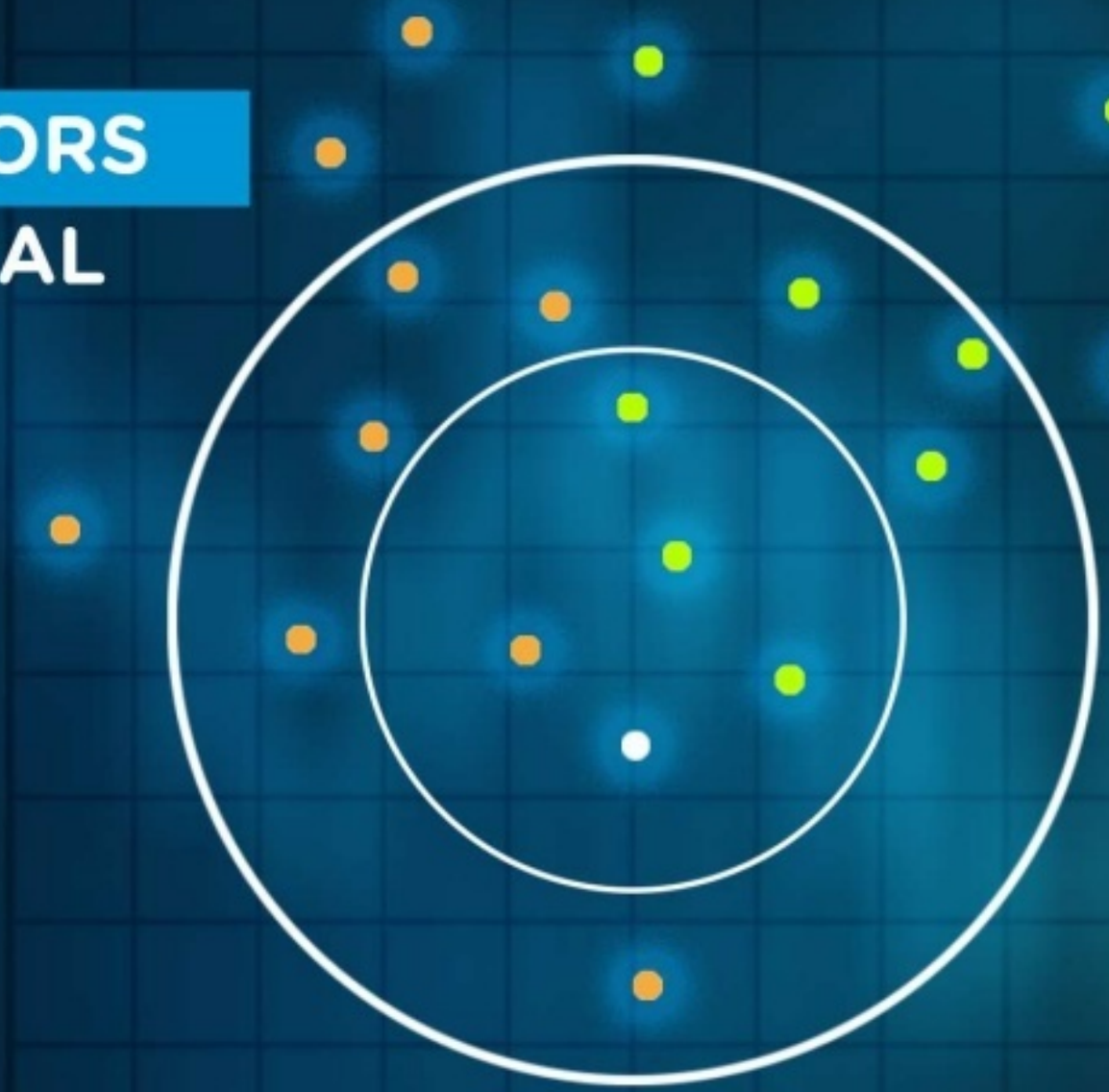
K-NEAREST NEIGHBORS
ALGORITHM TUTORIAL



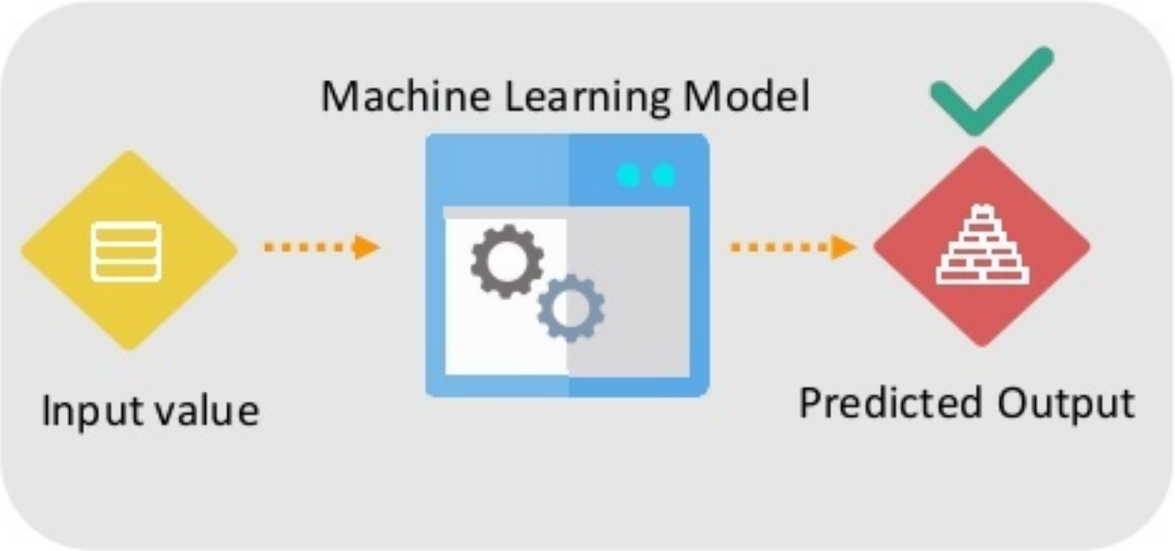
simplilearn

What's in it for you?

- Why do we need KNN?
- What is KNN?
- How do we choose the factor 'K'?
- When do we use KNN?
- How does KNN Algorithm work?
- Use Case: Predict whether a person will have diabetes or not

©Simplilearn. All rights reserved.

Why KNN?



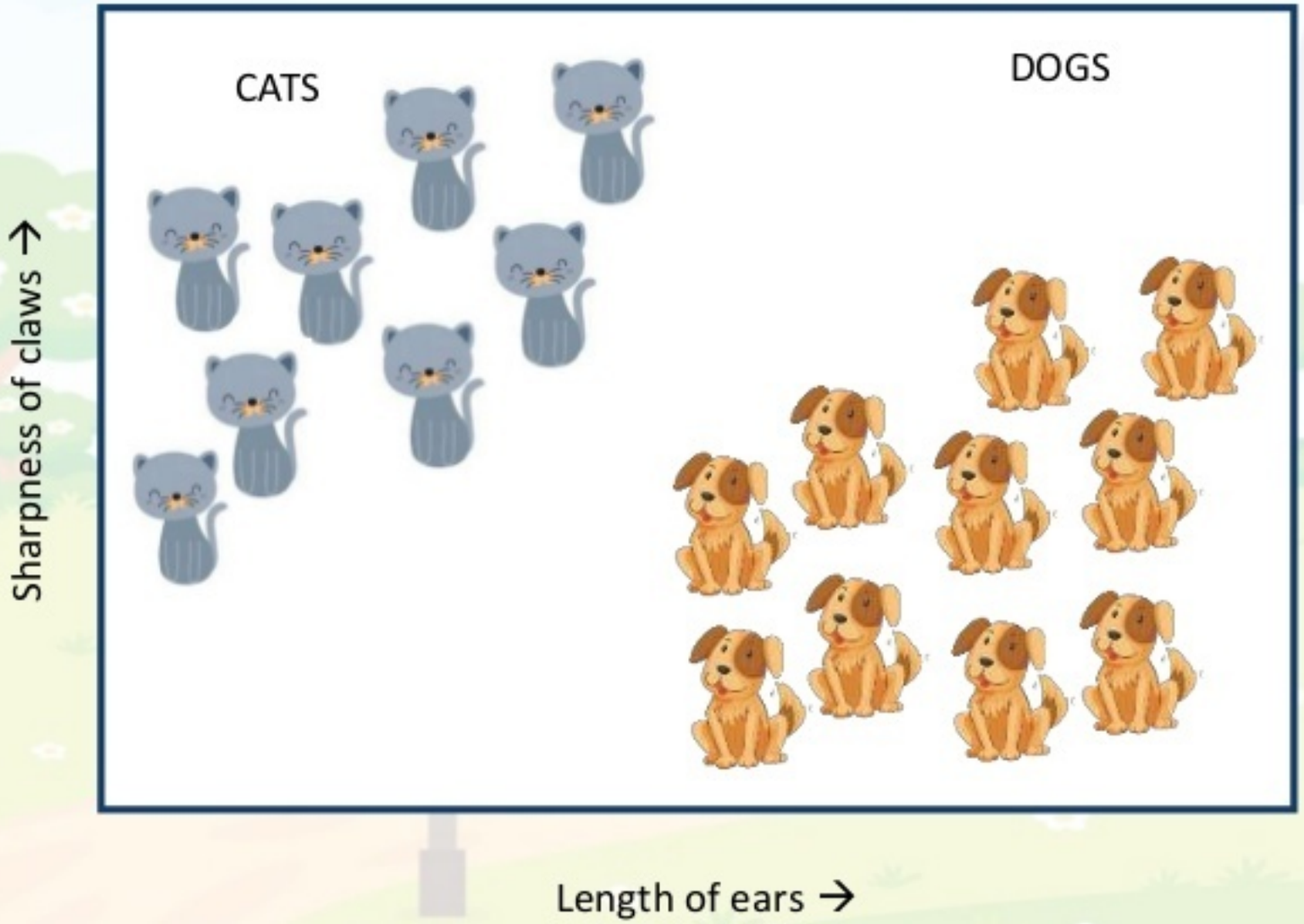
©Simplilearn. All rights reserved.

Is that a dog?

No dear, you can differentiate between a cat and a dog based on their characteristics

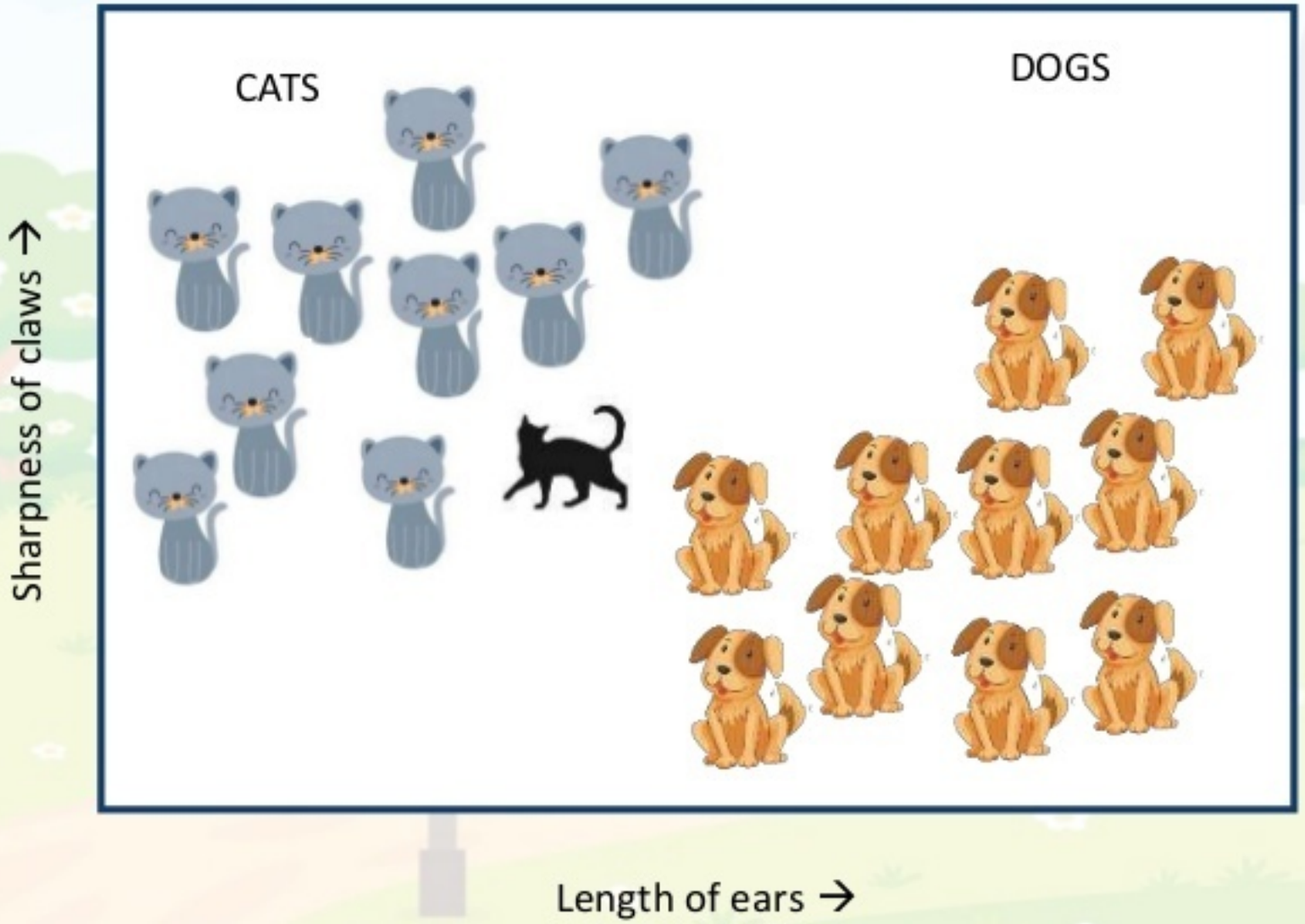
No dear, you can differentiate between a cat and a dog based on their characteristics

CATS	DOGS
Sharp Claws, uses to climb	Dull Claws
Smaller length of ears	Bigger length of ears
Meows and purrs	Barks
Doesn't love to play around	Loves to run around



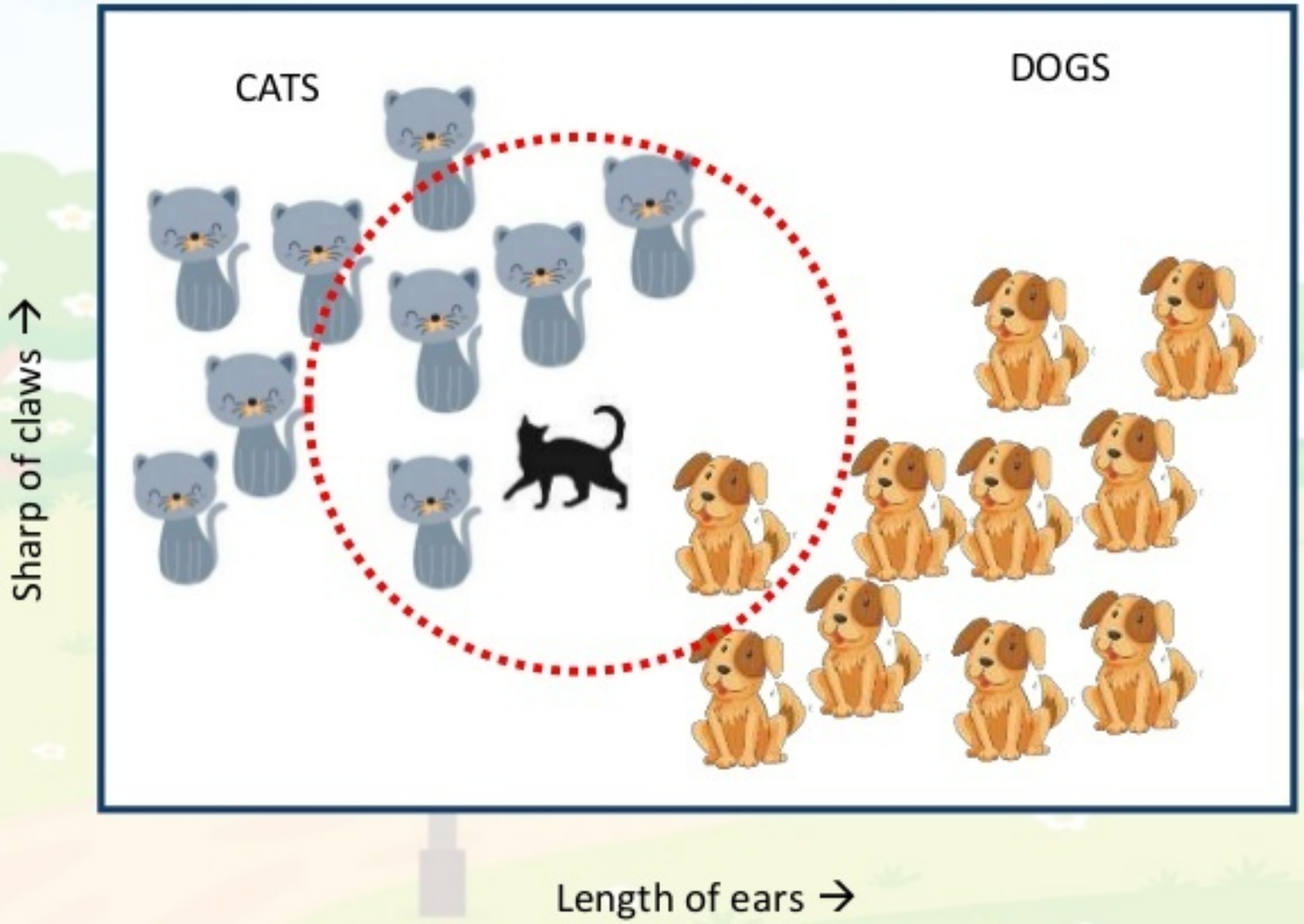
No dear, you can differentiate between a cat and a dog based on their characteristics

Now tell me if it is a cat or a dog?

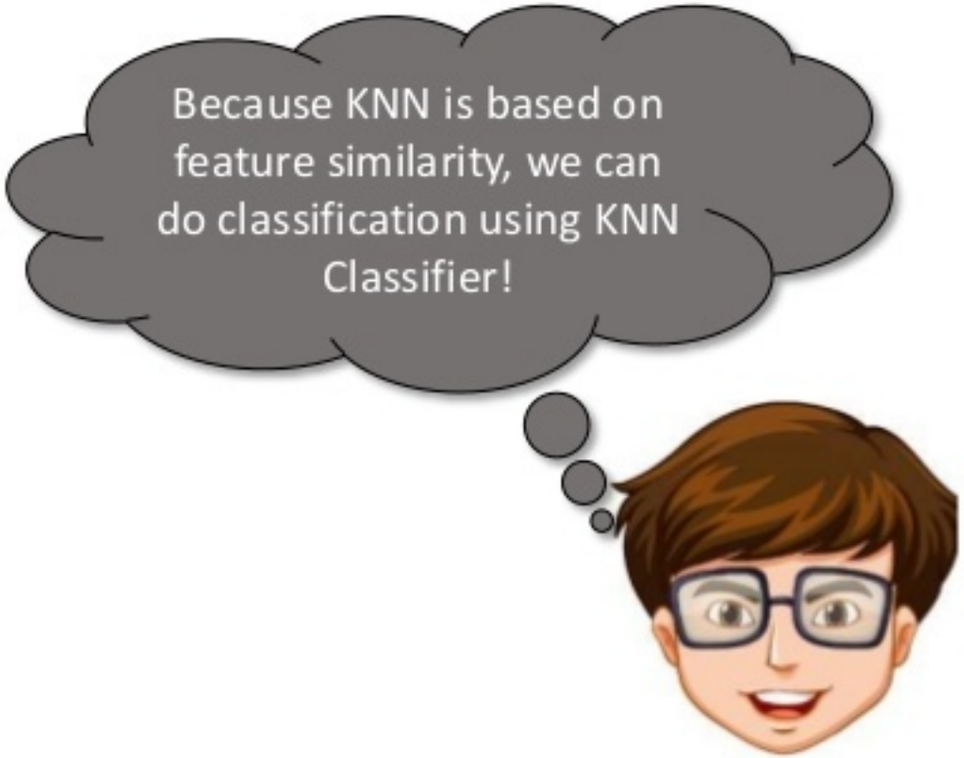


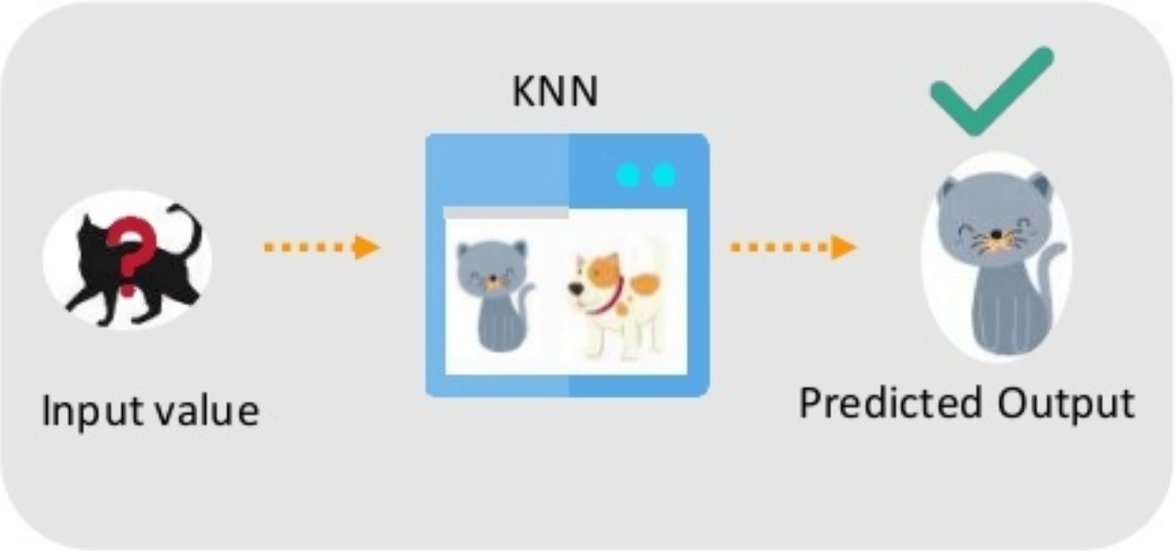
Now tell me if it's a cat or a dog?

It’s features are more like cats, it must be a cat!


CATS
DOGS
Sharp of claws →
Length of ears →

Why KNN?





©Simplilearn. All rights reserved.

What is KNN?

©Simplilearn. All rights reserved.

What is KNN Algorithm?

KNN – K Nearest Neighbors, is one of the simplest **Supervised** Machine Learning algorithm mostly used for

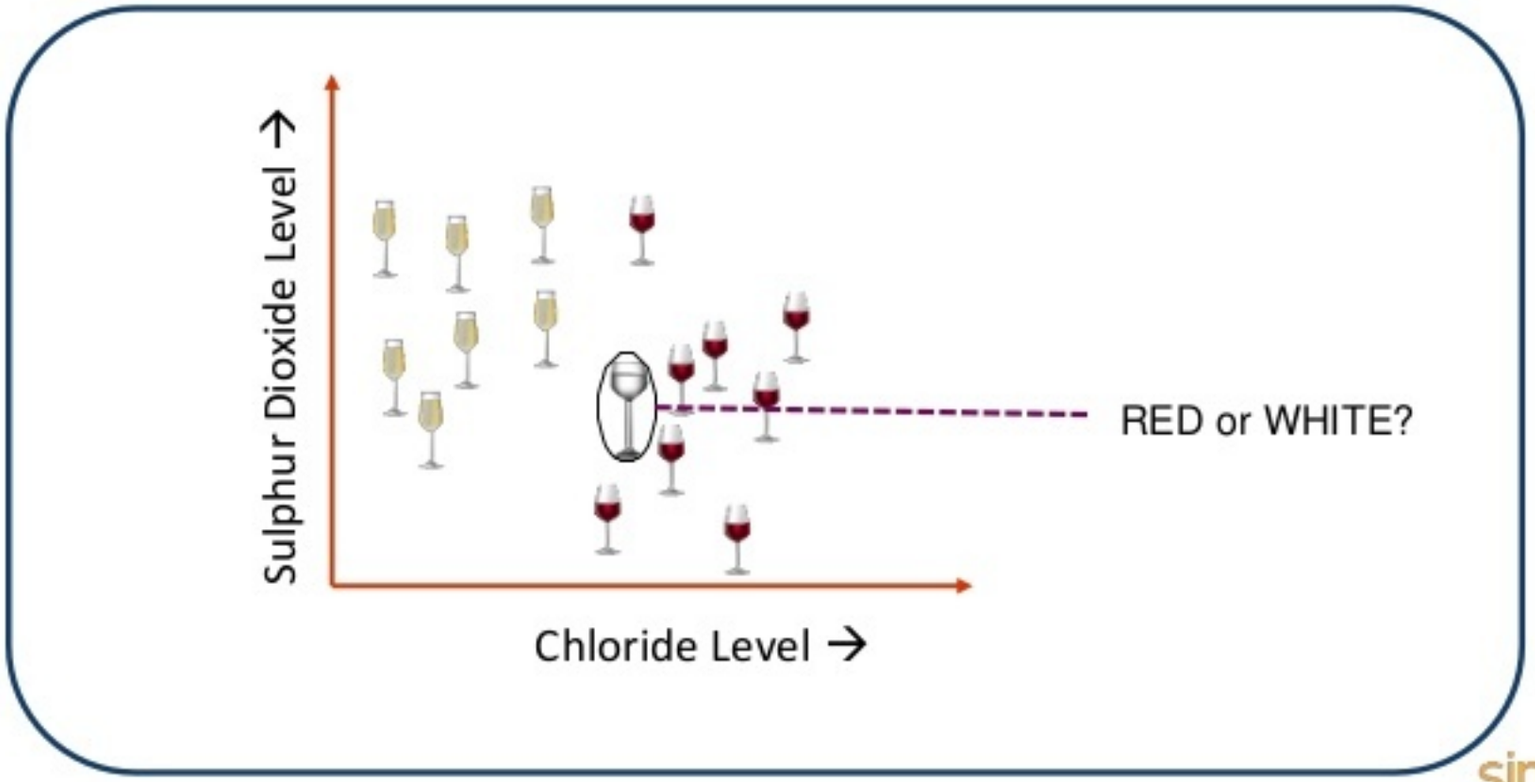
Classification

It classifies a data point based on how its neighbors are classified

©Simplilearn. All rights reserved.

What is KNN Algorithm?

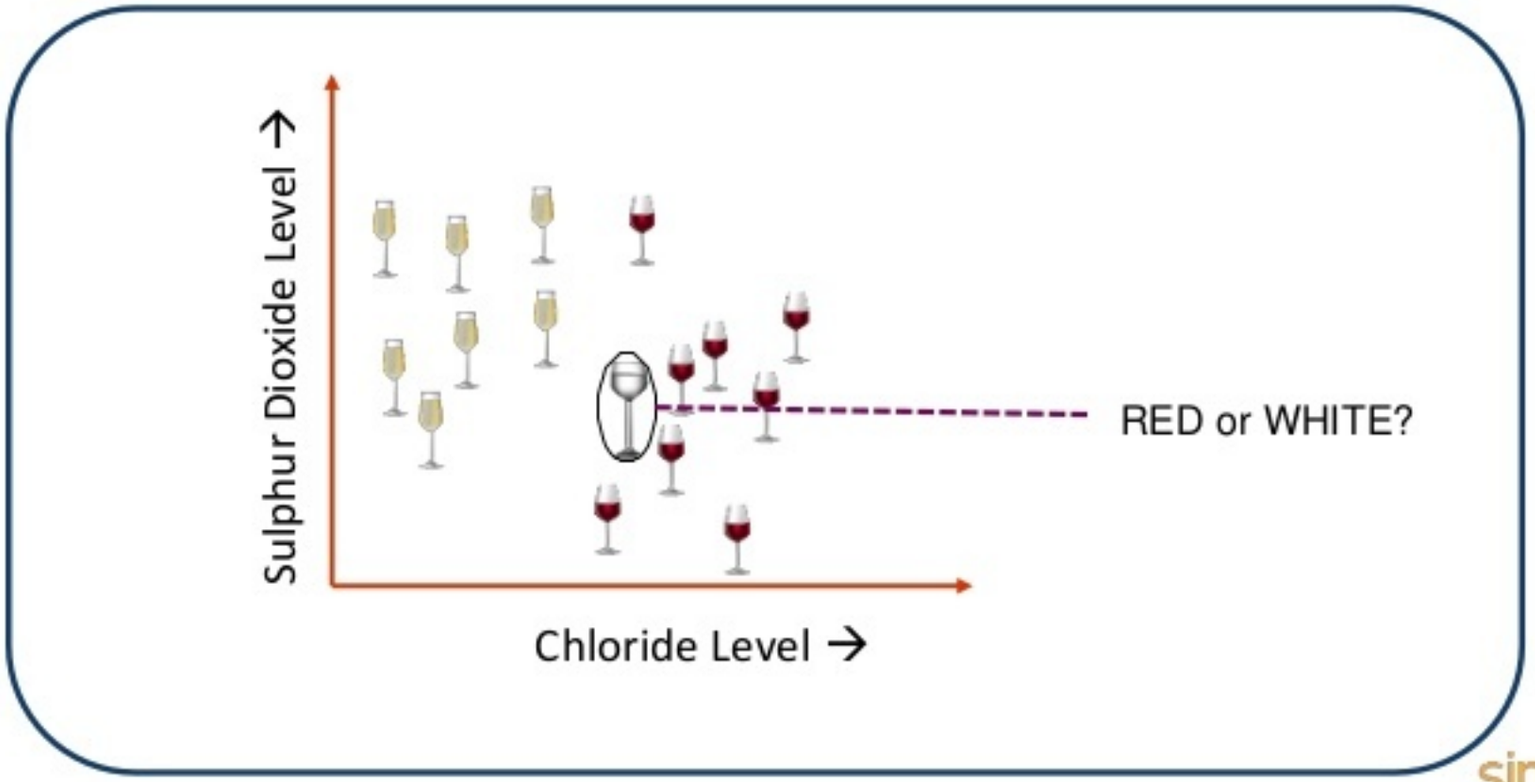
KNN stores all available cases and classifies new cases based on a similarity measure



©Simplilearn. All rights reserved.

What is KNN Algorithm?

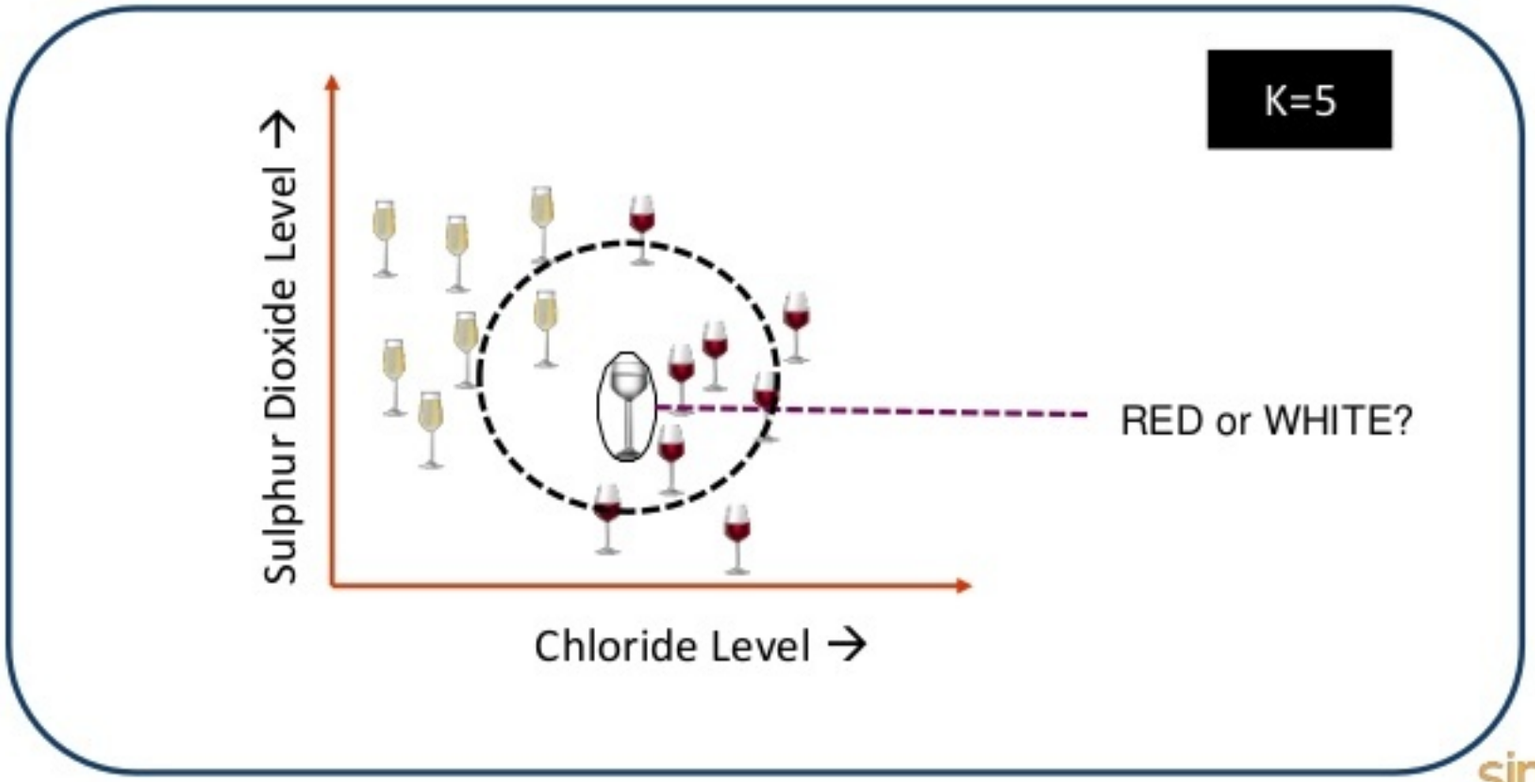
But, what is K?



©Simplilearn. All rights reserved.

What is KNN Algorithm?

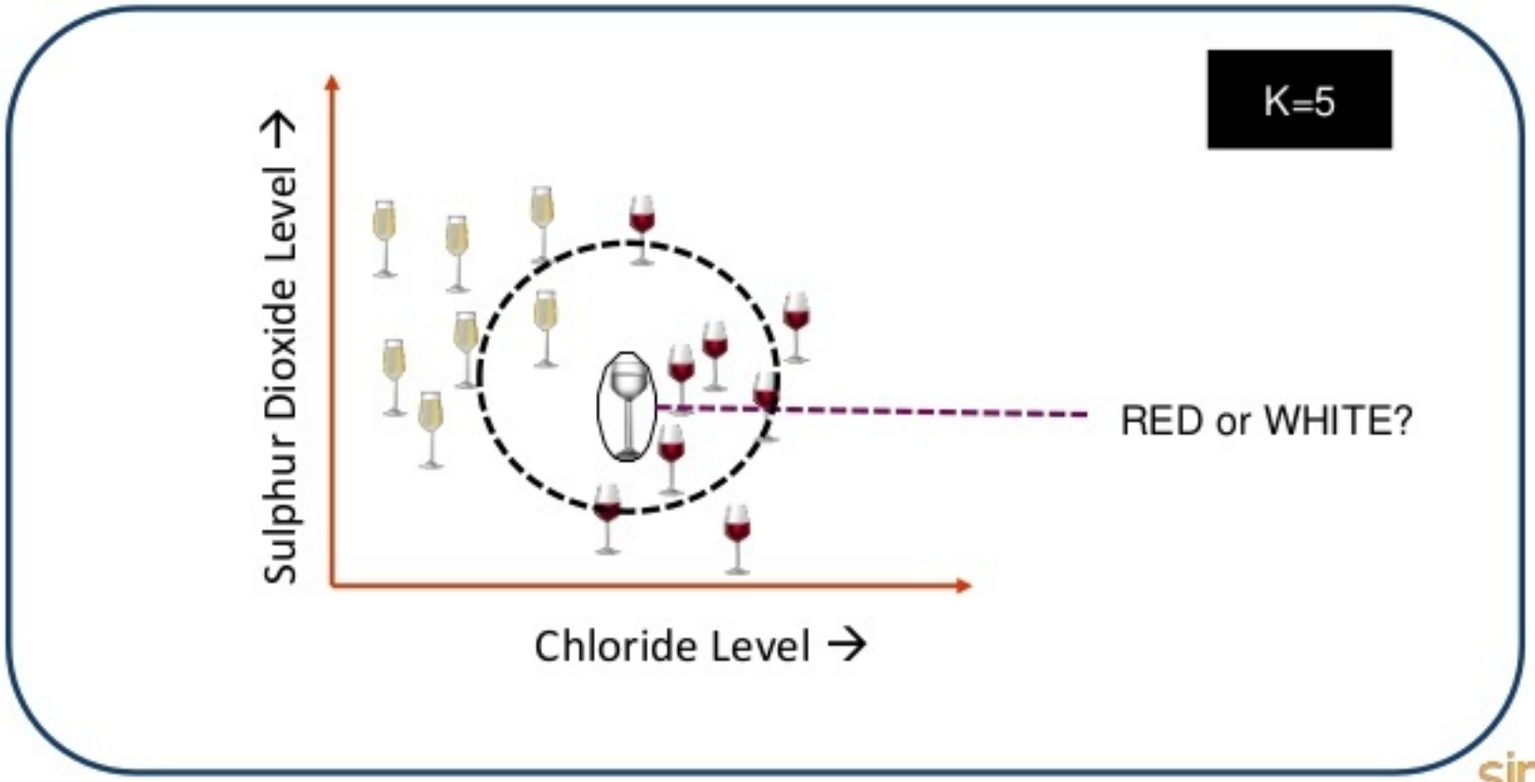
k in **KNN** is a parameter that refers to the number of nearest neighbors to include in the majority voting process



©Simplilearn. All rights reserved.

What is KNN Algorithm?

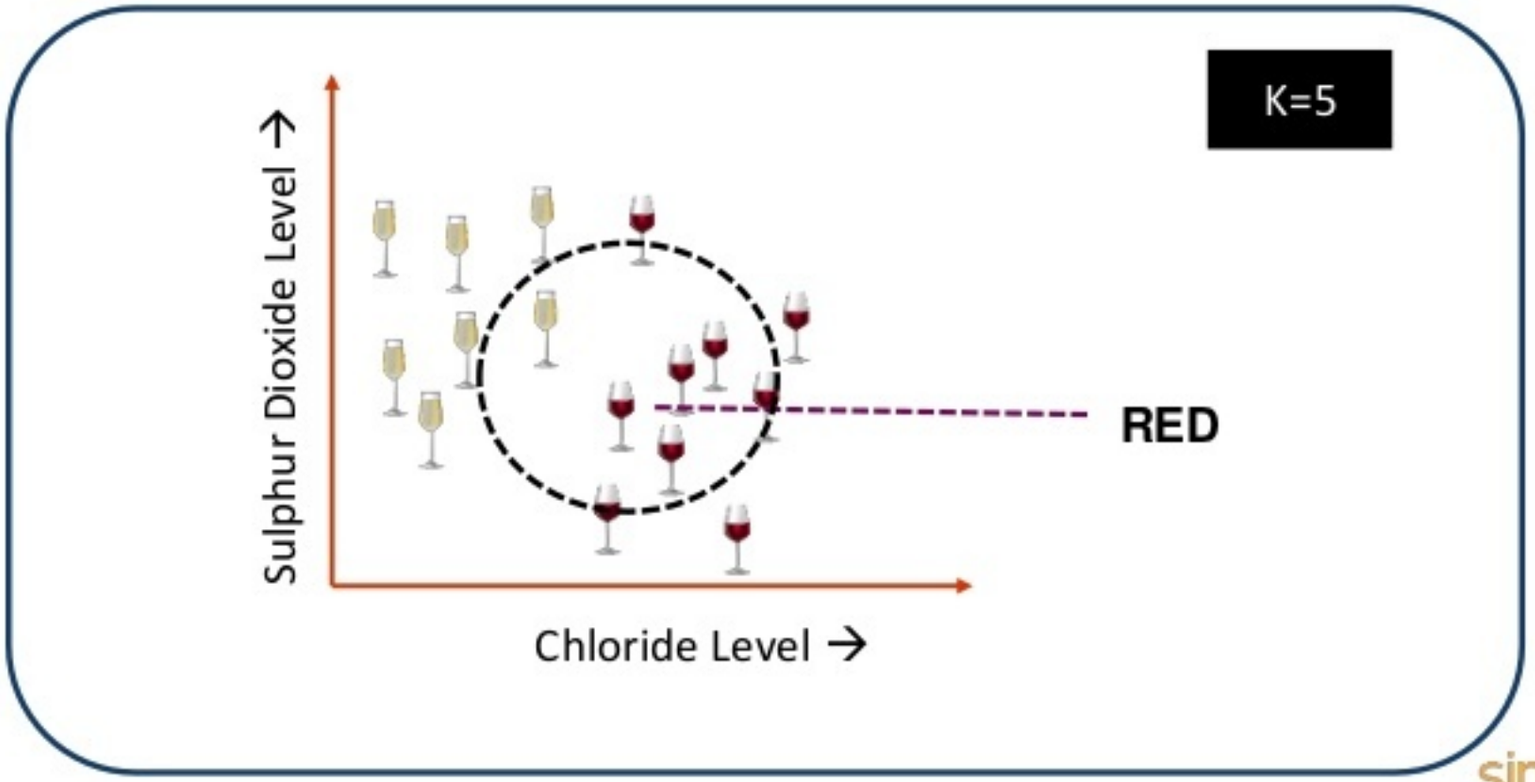
A data point is classified by majority votes from its 5 nearest neighbors



©Simplilearn. All rights reserved.

What is KNN Algorithm?

Here, the unknown point would be classified as red, since 4 out of 5 neighbors are red



©Simplilearn. All rights reserved.

How do we choose ‘k’?

©Simplilearn. All rights reserved.

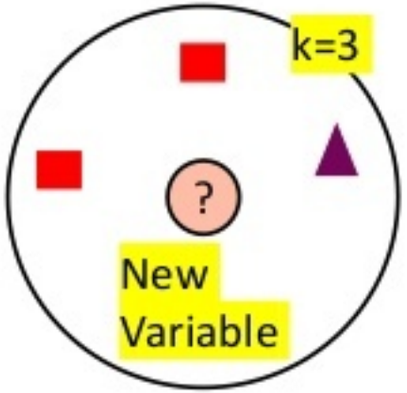
How do we choose the factor ‘k’?

KNN Algorithm is based on **feature similarity**: Choosing the right value of *k* is a process called parameter tuning, and is important for better accuracy

©Simplilearn. All rights reserved.

How do we choose the factor ‘k’?

KNN Algorithm is based on **feature similarity**: Choosing the right value of *k* is a process called parameter tuning, and is important for better accuracy

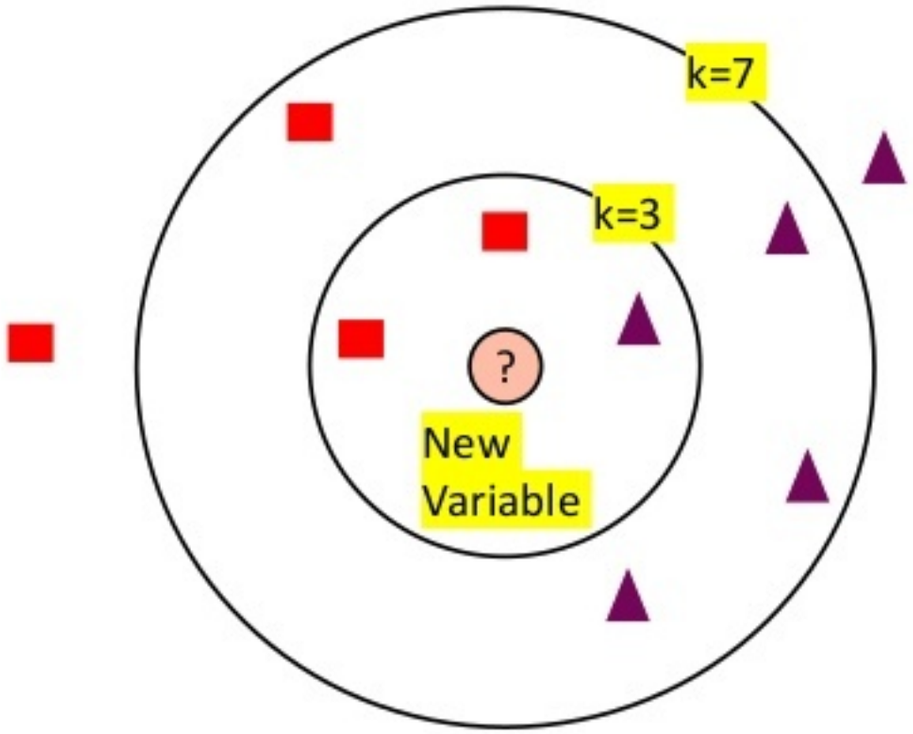


So at k=3, we can classify ‘?’ as

©Simplilearn. All rights reserved.

How do we choose the factor ‘k’?

KNN Algorithm is based on **feature similarity**: Choosing the right value of *k* is a process called parameter tuning, and is important for better accuracy



But at k=7, we classify ‘?’ as

©Simplilearn. All rights reserved.

How do we choose the factor ‘k’?

KNN Algorithm is based on **feature similarity**: Choosing the right value of k is a process called parameter tuning, and i

The class of unknown data point was ■ at k=3 but changed at k=7, so which k should we choose?

So at k=3, we can classify ‘?’ as

©Simplilearn. All rights reserved.

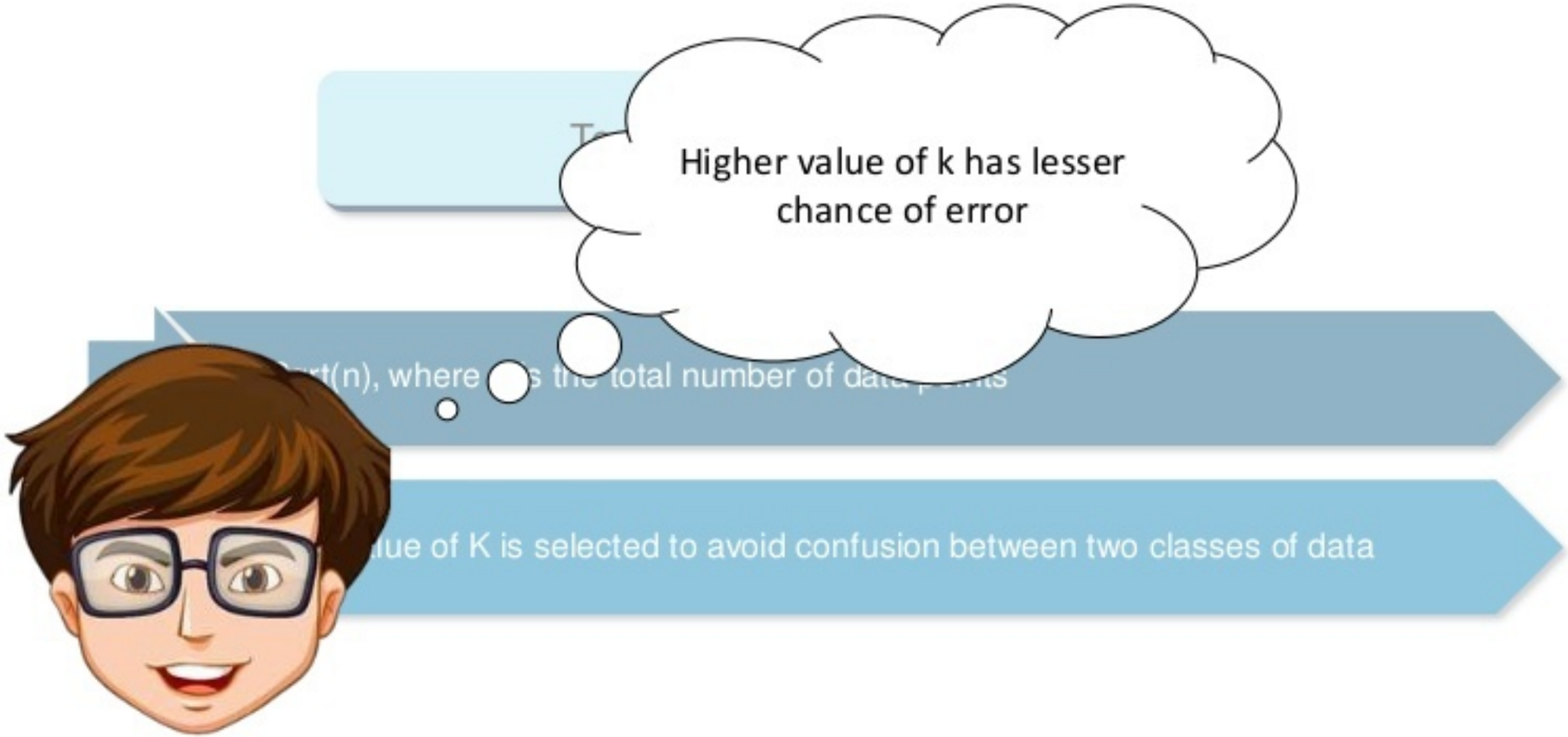
How do we choose the factor ‘k’?

To choose a value of k:

- Sqrt(n), where n is the total number of data points
- Odd value of K is selected to avoid confusion between two classes of data

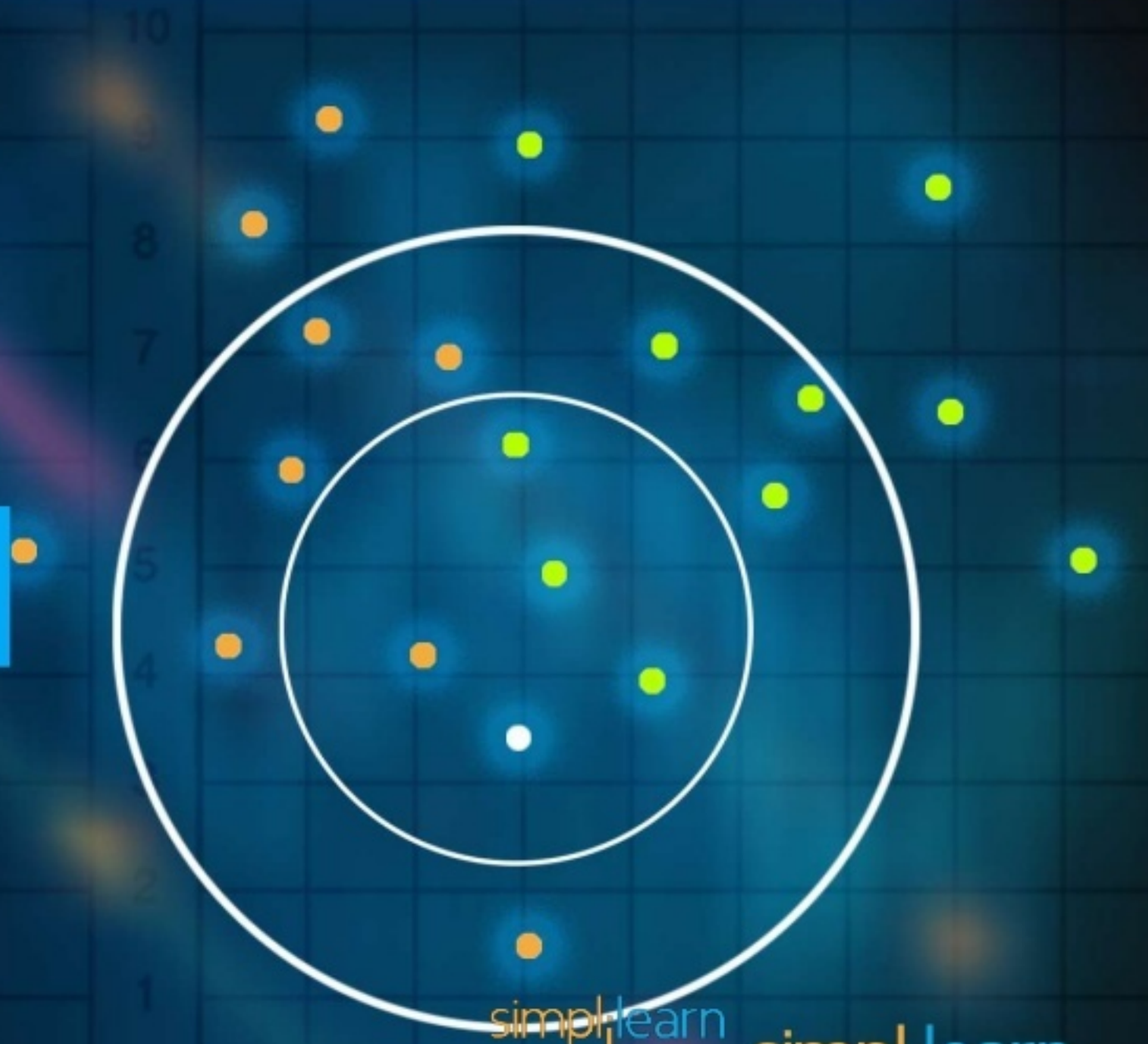
©Simplilearn. All rights reserved.

How do we choose the factor ‘k’?



©Simplilearn. All rights reserved.

When do we use KNN?



©Simplilearn. All rights reserved.

When do we use KNN Algorithm?

We can use KNN when

Data is labeled

Dog

©Simplilearn. All rights reserved.

When do we use KNN Algorithm?

We can use KNN when

Data is labeled

Dog

Data is noise free

Weight(x2)	Height(y2)	Class
51	167	Underweight
62	182	one-fourty
69	176	23
64	173	hello kitty
65	172	Normal

Noise

©Simplilearn. All rights reserved.

When do we use KNN Algorithm?

We can use KNN when

Data is labeled

Dog

Dataset is small

Because KNN is a 'lazy learner' i.e. doesn't learn a discriminative function from the training set

Data is noise free

Weight(x2)	Height(y2)	Class
51	167	Underweight
62	182	one-fourty
69	176	23
64	173	hello kitty
65	172	Normal

Noise

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Consider a dataset having two variables: height (cm) & weight (kg) and each point is classified as Normal or Underweight

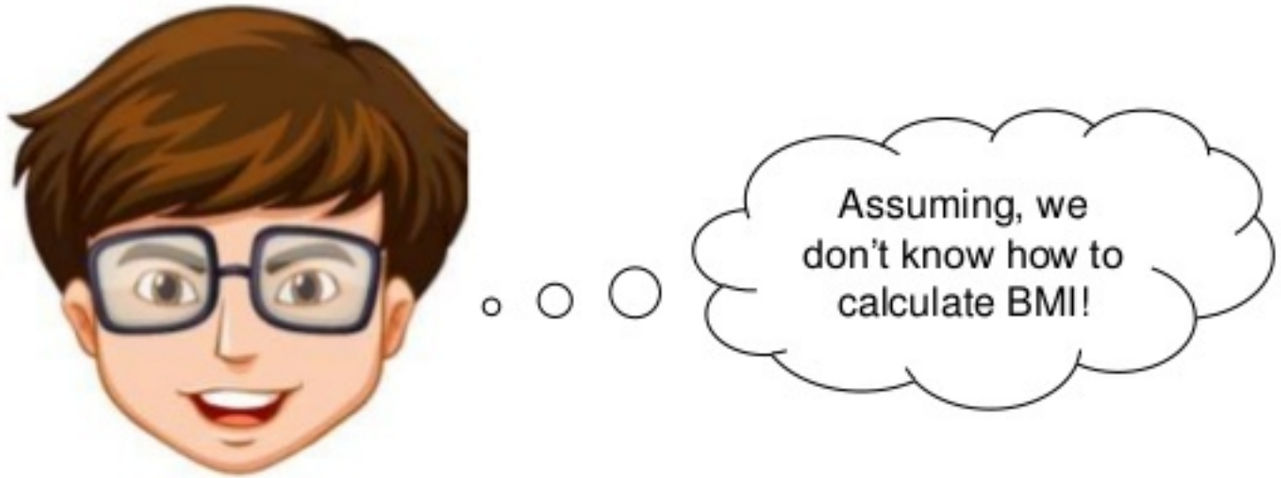
Weight(x2)	Height(y2)	Class
51	167	Underweight
62	182	Normal
69	176	Normal
64	173	Normal
65	172	Normal
56	174	Underweight
58	169	Normal
57	173	Normal
55	170	Normal

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

On the basis of the given data we have to classify the below set as Normal or Underweight using KNN

57 kg	**170 cm**	**?**



©Simplilearn. All rights reserved.

How does KNN Algorithm work?

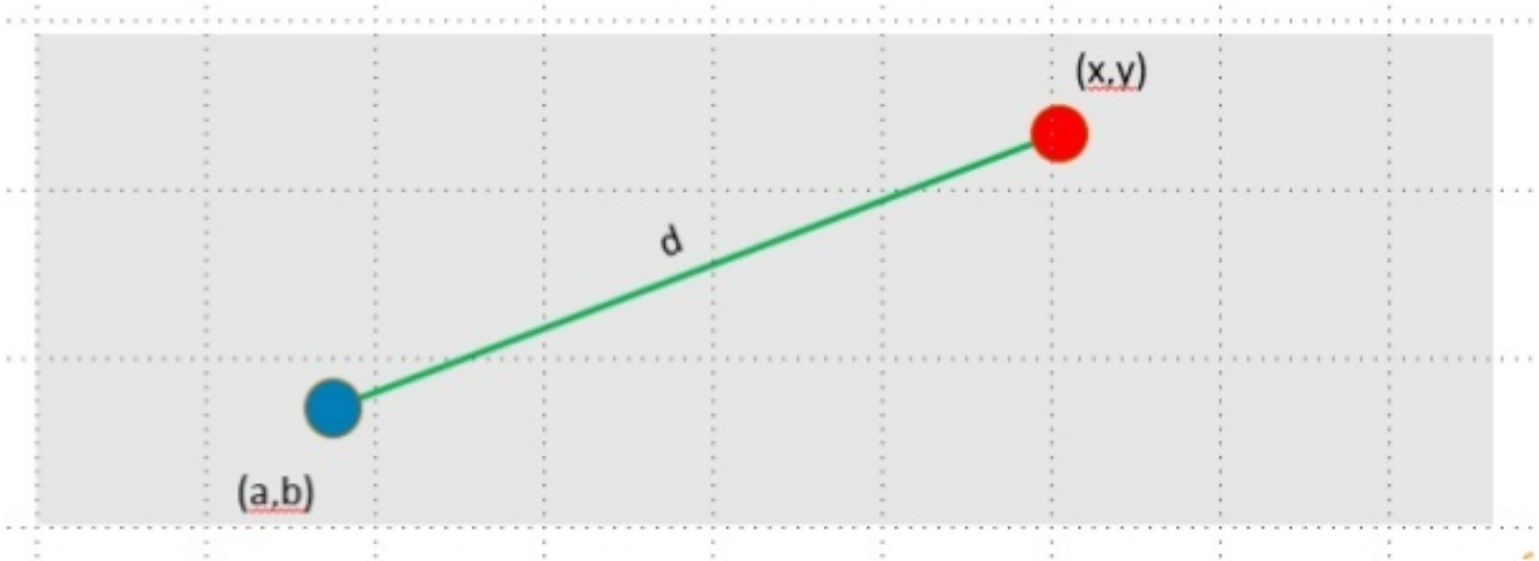
To find the nearest neighbors, we will calculate Euclidean distance

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

According to the **Euclidean distance** formula, the **distance** between two points in the plane with coordinates (x, y) and (a, b) is given by:

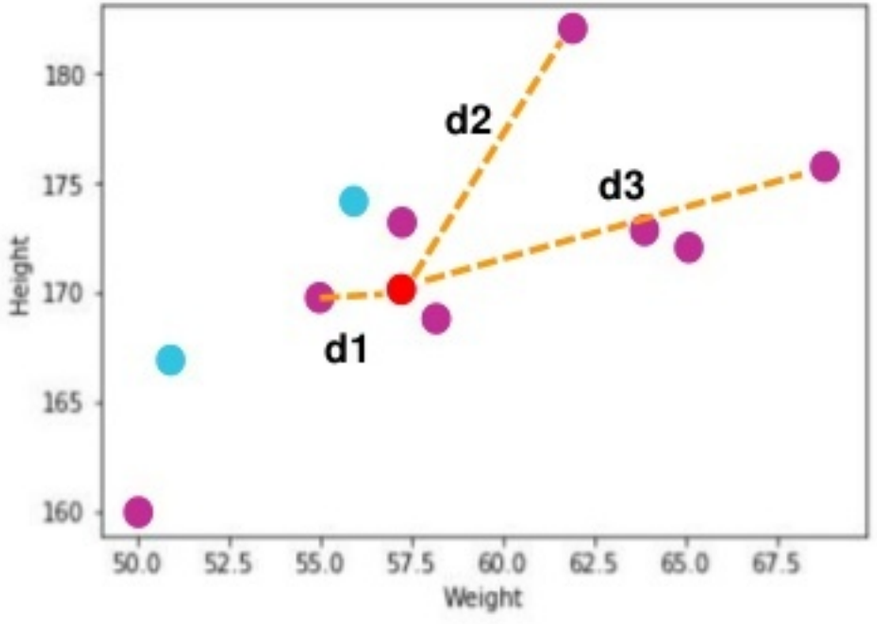
$$\text{dist}(d) = \sqrt{(x - a)^2 + (y - b)^2}$$



©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Let's calculate it to understand clearly:



Unknown data point

dist(**d1**)= $\sqrt{(170-167)^2 + (57-51)^2} \sim= 6.7$

dist(**d2**)= $\sqrt{(170-182)^2 + (57-62)^2} \sim= 13$

dist(**d3**)= $\sqrt{(170-176)^2 + (57-69)^2} \sim= 13.4$

Similarly, we will calculate Euclidean distance of unknown data point from all the points in the dataset

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Hence, we have calculated the Euclidean distance of unknown data point from all the points as shown:

Where (x1, y1) = (57, 170) whose class we have to classify

Weight(x2)	Height(y2)	Class	Euclidean Distance
51	167	Underweight	6.7
62	182	Normal	13
69	176	Normal	13.4
64	173	Normal	7.6
65	172	Normal	8.2
56	174	Underweight	4.1
58	169	Normal	1.4
57	173	Normal	3
55	170	Normal	2

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Now, lets calculate the nearest neighbor at k=3

Weight(x2)	Height(y2)	Class	Euclidean Distance
51	167	Underweight	6.7
62	182	Normal	13
69	176	Normal	13.4
64	173	Normal	7.6
65	172	Normal	8.2
56	174	Underweight	4.1
58	169	Normal	1.4
57	173	Normal	3
55	170	Normal	2

k = 3

57 kg	**170 cm**	**?**

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Now, lets calculate the nearest neighbor at k=3

We have n=10,
And sqrt(10)=3.1
Hence, we have taken k=3

Wei			n Distance
			.7
			.4
			7.6
65			
56	174	derweight	4.1
58	169	Normal	1.4
57	173	Normal	3
55	170	Normal	2

57 kg	170 cm	?

©Simplilearn. All rights reserved.

How does KNN Algorithm work?

Class	Euclidean Distance
Underweight	6.7
Normal	13
Normal	13.4
Normal	7.6
Normal	8.2
Underweight	4.1
Normal	1.4
Normal	3
Normal	2

k = 3

So, majority neighbors are pointing towards '*Normal*'

Hence, as per KNN algorithm the class of (57, 170) should be 'Normal'

©Simplilearn. All rights reserved.

Recap of KNN

Recap of KNN

- A positive integer k is specified, along with a new sample
- We select the k entries in our database which are closest to the new sample
- We find the most common classification of these entries
- This is the classification we give to the new sample

©Simplilearn. All rights reserved.

USE CASE: Predict Diabetes

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Objective: Predict whether a person will be diagnosed with diabetes or not

> We have a dataset of 768 people who were or were not diagnosed with diabetes

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Import the required Scikit-learn libraries as shown:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Load the dataset and have a look:

```
dataset = pd.read_csv('../Downloads/diabetes.csv')
```

```
dataset.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Values of columns like ‘Glucose’, BloodPressure’ cannot be accepted as zeroes because it will affect the outcome

We can replace such values with the mean of the respective column:

```
# Replace zeroes
zero_not_accepted = ['Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'Insulin']
```

```
for column in zero_not_accepted:
    dataset[column] = dataset[column].replace(0, np.NaN)
    mean = int(dataset[column].mean(skipna=True))
    dataset[column] = dataset[column].replace(np.NaN, mean)
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Before proceeding further, let's split the dataset into train and test:

```
# split dataset
X = dataset.iloc[:, 0:8]
y = dataset.iloc[:, 8]
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0, test_size=0.2)
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Rule of thumb: Any algorithm that computes distance or assumes normality, **scale your features!**

Feature Scaling:

```
# Feature scaling
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

N_neighbors here is 'K'
p is the power parameter to define the metric used, which is 'Euclidean' in our case

Then define the model using KNeighborsClassifier and fit the train data in the model

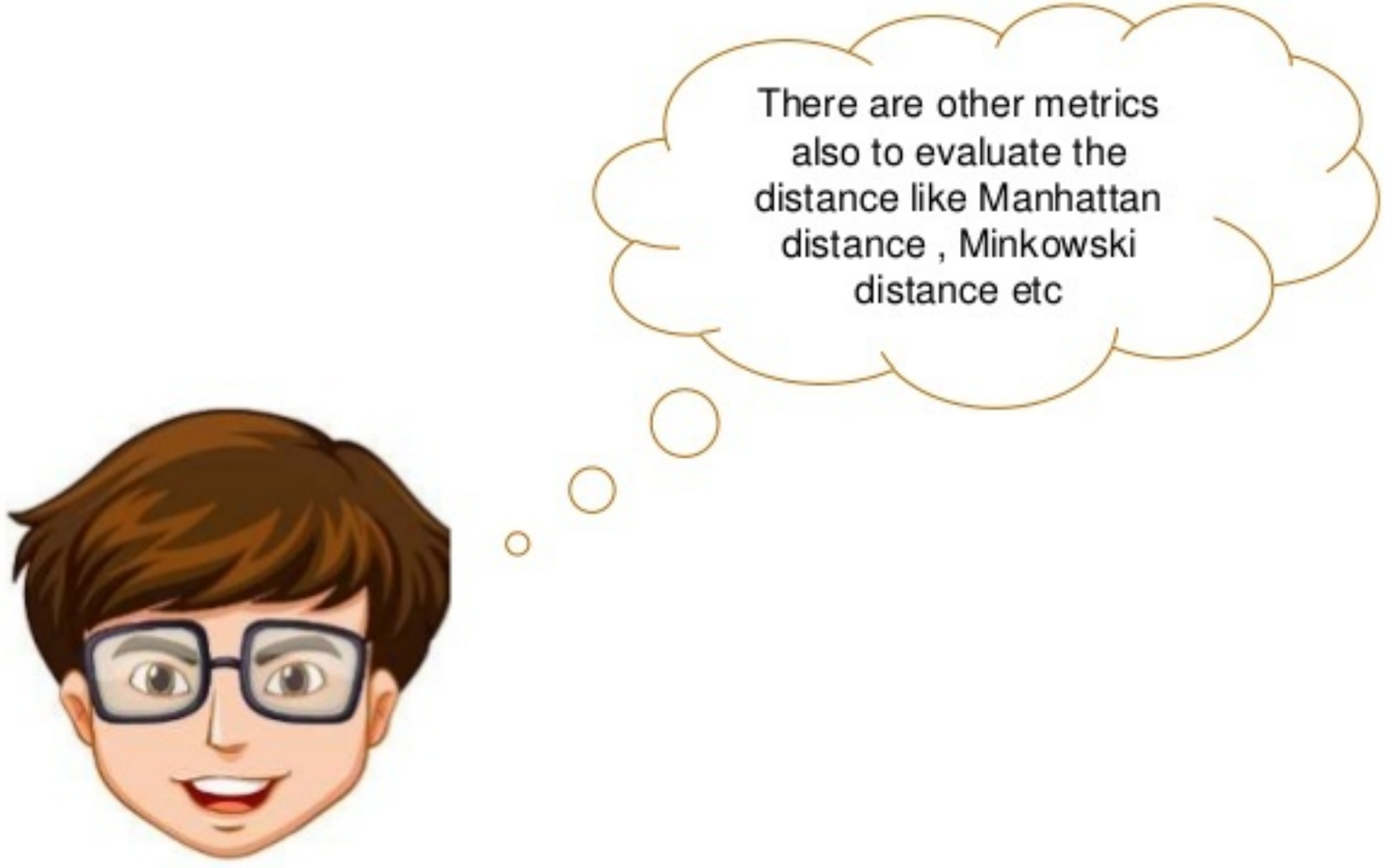
```
# Define the model: Init K-NN
classifier = KNeighborsClassifier(n_neighbors=11, p=2, metric='euclidean')
```

```
# Fit Model
classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
           metric_params=None, n_jobs=1, n_neighbors=11, p=2,
           weights='uniform')
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes



©Simplilearn. All rights reserved.

KNN - Predict diabetes

Let's predict the test results:

```
# Predict the test set results
y_pred = classifier.predict(X_test)
```

```
y_pred
```

```
array([1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      dtype=int64)
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

It's important to evaluate the model, let's use confusion matrix to do that:

```
# Evaluate Model
cm = confusion_matrix(y_test, y_pred)
print (cm)
print(f1_score(y_test, y_pred))
```

```
[[94 13]
 [15 32]]
0.6956521739130436
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

Calculate accuracy of the model:

```
print(accuracy_score(y_test, y_pred))
```

```
0.8181818181818182
```

©Simplilearn. All rights reserved.

KNN - Predict diabetes

©Simplilearn. All rights reserved.

KNN - Predict diabetes

```
print(accuracy_score(y_test, y_pred))
```

```
0.8181818181818182
```

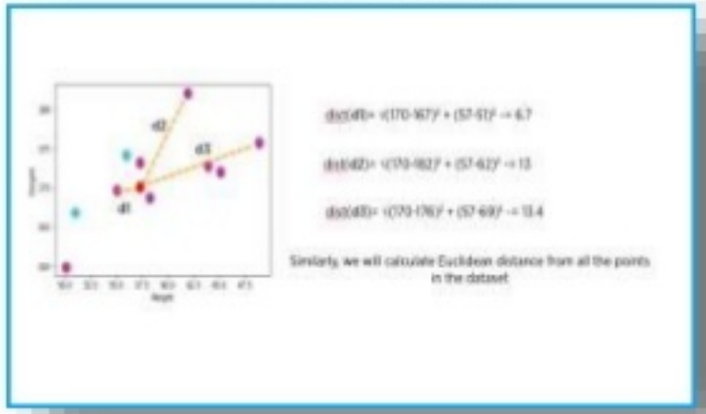
And accuracy of 80% tells us that it is a pretty fair fit in the model!

©Simplilearn. All rights reserved.

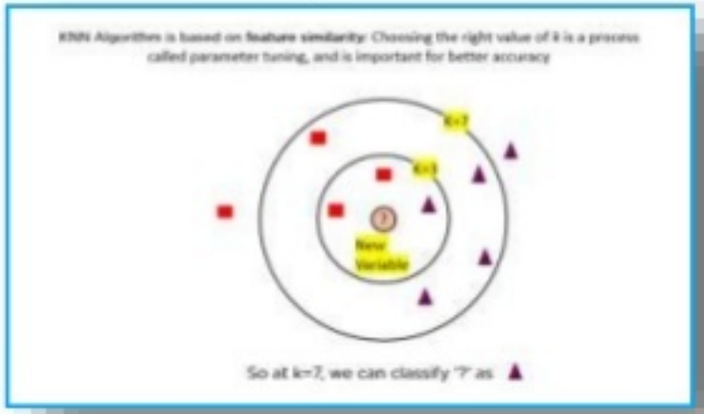
Summary

Why we need knn?

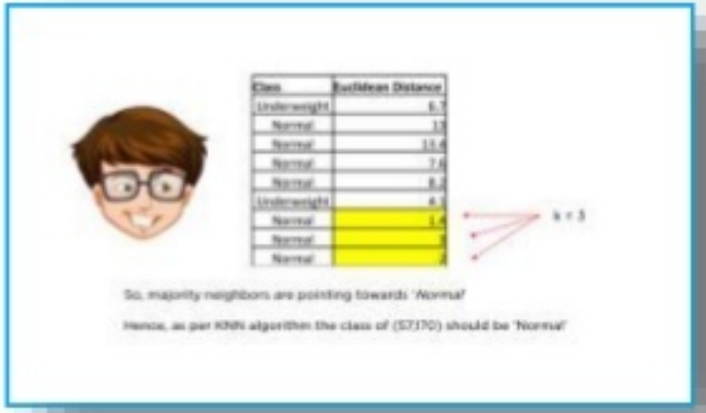
Eucledian distance



Choosing the value of k



How KNN works?



Knn classifier for diabetes prediction

©Simplilearn. All rights reserved.

THANK YOU
For more information, visit
www.simplilearn.com
simplilearn