SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

Hi guys,
Welcome to the K-Nearest-Neighbors lecture!

'K-Nearest-Neighbors' (also known as KNN) is another very important Machine Learning model to solve classification problems. In this lecture, we are going to discuss the basic working principle behind KNN.

✅ *Optional Readings and References:*
sklearn's Official Documentation - 1.6 Nearest Neighbours
Introduction to Statistical Learning - Chapter 4 (4.6.5)

General message: *Key concepts along with significant commentary / text is provided in the slides, so that they serve as a reference for the respective theory lecture. However, the suggested readings are recommended to explore more on the topic under discussion! Good luck!*

Dr. Junaid S. Qazi
PhD

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

- KNN is widely used for classification and is one of the simplest among all the Machine Learning algorithms.
- The principle behind the method is to find a pre-defined number of training samples, closest in distance to the test point, and predict the label from these.

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

- KNN is widely used for classification and is one of the simplest among all the Machine Learning algorithms.
- The principle behind the method is to find a pre-defined number of training samples, closest in distance to the test point, and predict the label from these.

In simple words, KNN stores entire dataset as a training algorithm and categorizes the test datapoint using stored information of its nearest neighbors, k is the number of neighbors.

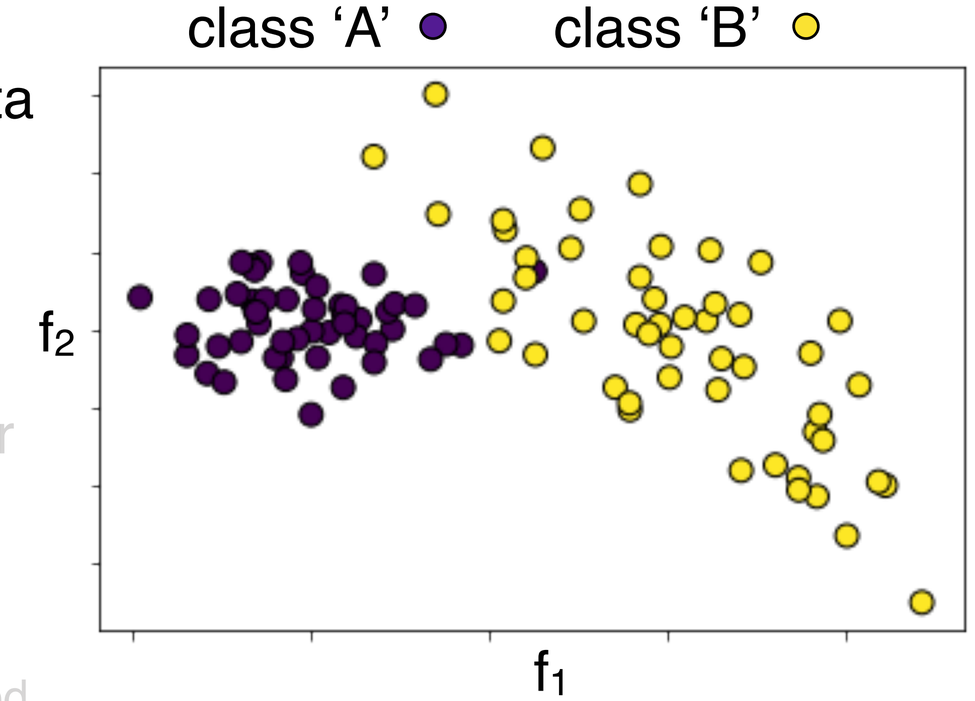
Let’s learn with some example data

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

Let's consider:

- We have a dataset with two classes, 'A' & 'B' and the entire data along with its features, f_1, f_2 are stored as a training algorithm.
- We want to predict the class for red, green & blue test points.
- Based on the association with their neighboring points, its straight forward to predict the class for red and green:
  - The nearest neighboring datapoint for red are in 'B', so KNN will predict class B for red
  - The nearest neighboring datapoint for green are in 'A', so KNN will predict class A for green



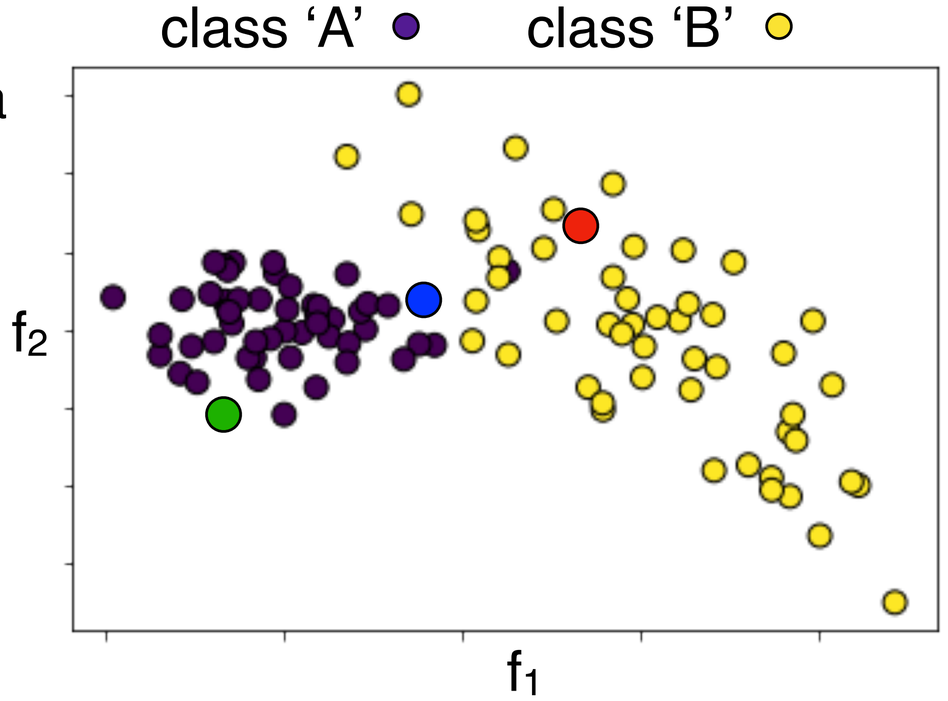
Code used to generate the plot is available at: http://scikit-learn.org/stable/auto_examples/datasets/plot_random_dataset.html

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

Let's consider:

- We have a dataset with two classes, 'A' & 'B' and the entire data along with its features, f_1, f_2 are stored as a training algorithm.
- We want to predict the class for red, green & blue test points.
- Based on the association with their neighboring points, its straight forward to predict the class for red and green:
  - The nearest neighboring datapoint for red are in 'B', so KNN will predict class B for red
  - The nearest neighboring datapoint for green are in 'A', so KNN will predict class A for green



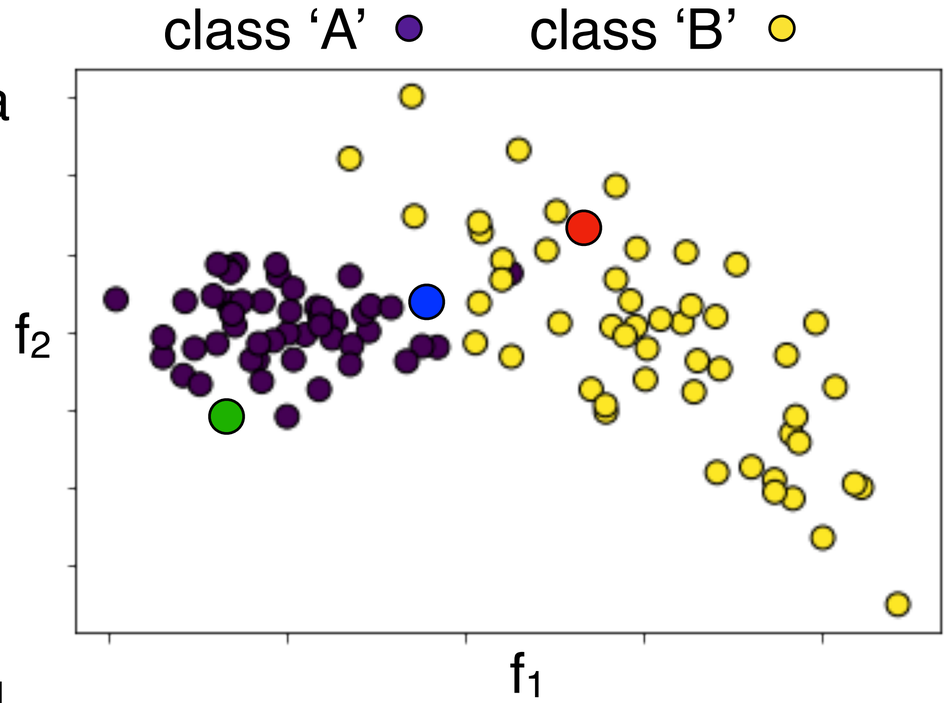
Code used to generate the plot is available at: http://scikit-learn.org/stable/auto_examples/datasets/plot_random_dataset.html

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

Let's consider:

- We have a dataset with two classes, 'A' & 'B' and the entire data along with its features, f_1, f_2 are stored as a training algorithm.
- We want to predict the class for red, green & blue test points.
- Based on the association with their neighboring points, its straight forward to predict the class for red and green:
  - The nearest neighboring datapoint for red are in 'B', so KNN will predict class B for red
  - The nearest neighboring datapoint for green are in 'A', so KNN will predict class A for green



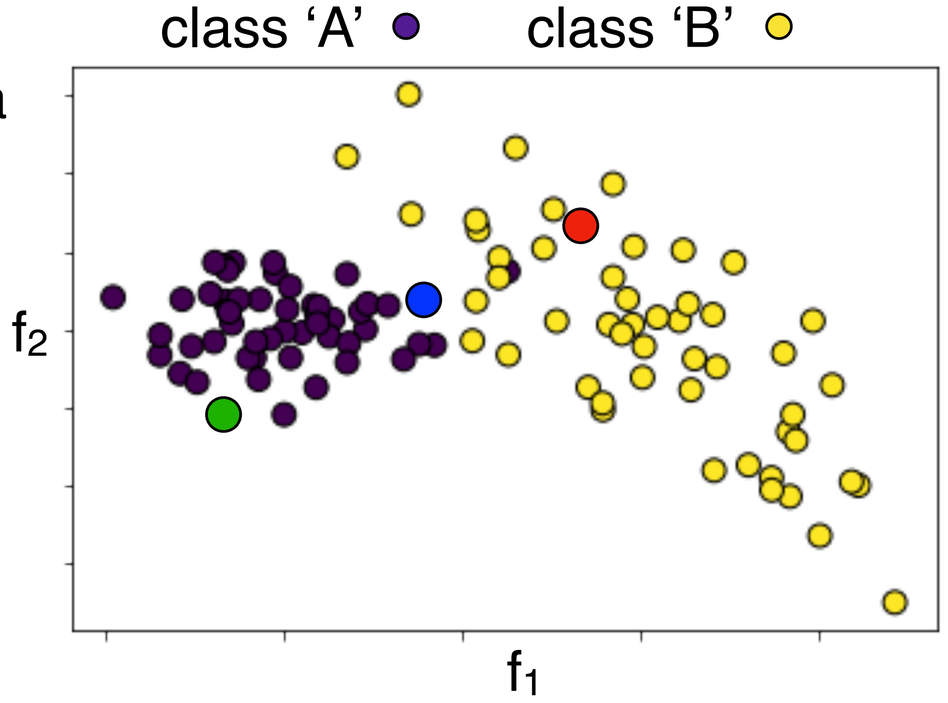
Code used to generate the plot is available at: http://scikit-learn.org/stable/auto_examples/datasets/plot_random_dataset.html

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

Let's consider:

- We have a dataset with two classes, 'A' & 'B' and the entire data along with its features, f_1, f_2 are stored as a training algorithm.
- We want to predict the class for red, green & blue test points.
- Based on the association with their neighboring points, its straight forward to predict the class for red and green:
  - The nearest neighboring datapoint for red are in 'B', so KNN will predict class B for red
  - The nearest neighboring datapoint for green are in 'A', so KNN will predict class A for green



To predict the class for blue datapoint, let's discuss KNN with another simpler example with fewer data-points.

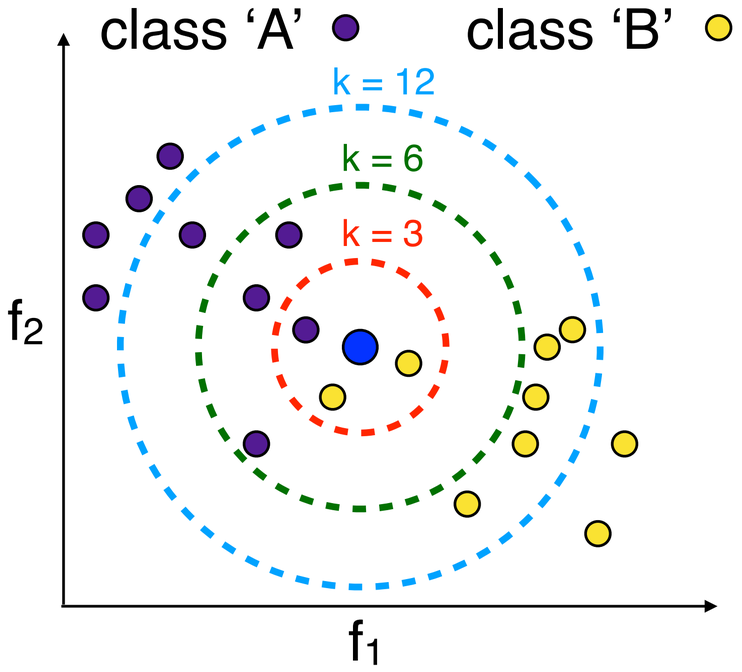
Code used to generate the plot is available at: http://scikit-learn.org/stable/auto_examples/datasets/plot_random_dataset.html

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

We already know, the training algorithm is actually storing entire dataset in KNN.

For predictions, KNN simply looks at the k points in the training set that are nearest to the test input, counts how many members of each class are in the set, and return the class with highest frequency as a prediction. So, the class with the most votes in taken as the prediction.



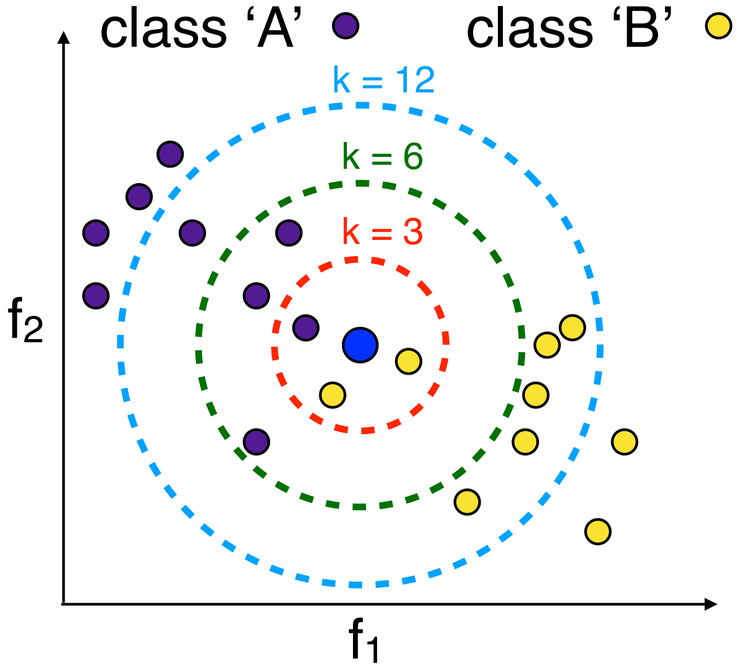
SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

We already know, the training algorithm is actually storing entire dataset in KNN.

For predictions, KNN simply looks at the k points in the training set that are nearest to the test input, counts how many members of each class are in the set, and return the class with highest frequency as a prediction. So, the class with the most votes in taken as the prediction.

In other words, the algorithm calculate the distance from the test point to all data-points in the train dataset, sort the points by increasing distance from the test input into a distance matrix and predict the majority class for k closest points.



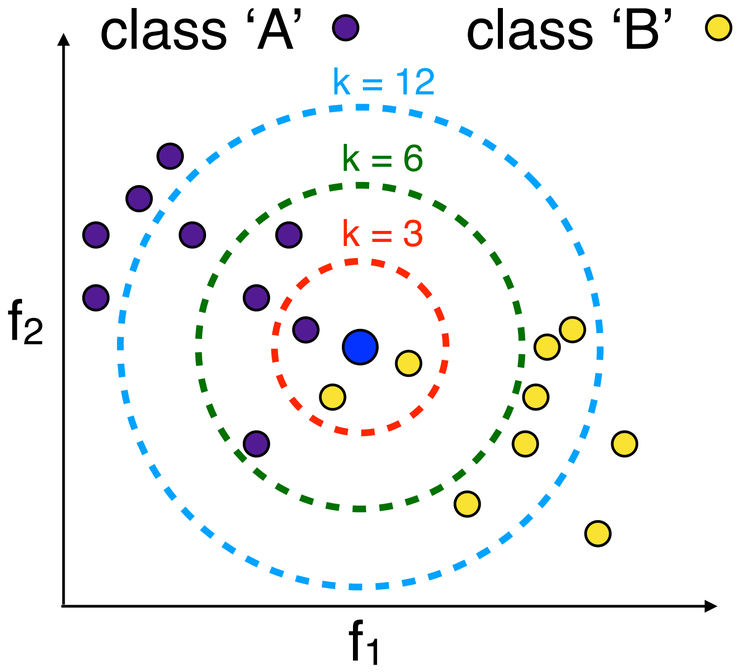
SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

We already know, the training algorithm is actually storing entire dataset in KNN.

For predictions, KNN simply looks at the k points in the training set that are nearest to the test input, counts how many members of each class are in the set, and return the class with highest frequency as a prediction. So, the class with the most votes in taken as the prediction.

In other words, the algorithm calculate the distance from the test point to all data-points in the train dataset, sort the points by increasing distance from the test input into a distance matrix and predict the majority class for k closest points.



By applying the above principle in the diagram, for:

- **k = 3 —> Predicted class B for test/blue datapoint**
- **k = 6 —> Predicted class A for test/blue datapoint**
- **k = 12 —> Predicted class B for test/blue datapoint**

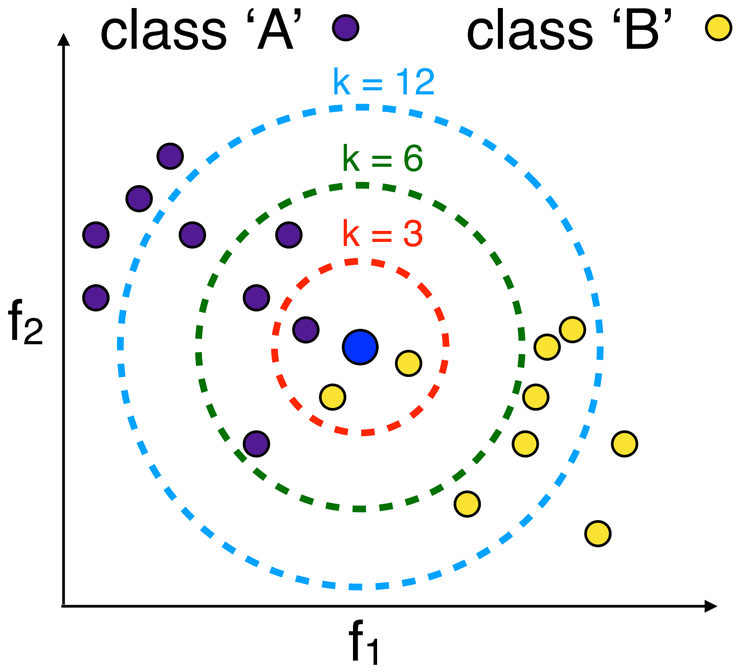
SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

We already know, the training algorithm is actually storing entire dataset in KNN.

For predictions, KNN simply looks at the k points in the training set that are nearest to the test input, counts how many members of each class are in the set, and return the class with highest frequency as a prediction. So, the class with the most votes in taken as the prediction.

In other words, the algorithm calculate the distance from the test point to all data-points in the train dataset, sort the points by increasing distance from the test input into a distance matrix and predict the majority class for k closest points.



By applying the above principle in the diagram, for:

- **k = 3 —> Predicted class B for test/blue datapoint**
- **k = 6 —> Predicted class A for test/blue datapoint**
- **k = 12 —> Predicted class B for test/blue datapoint**

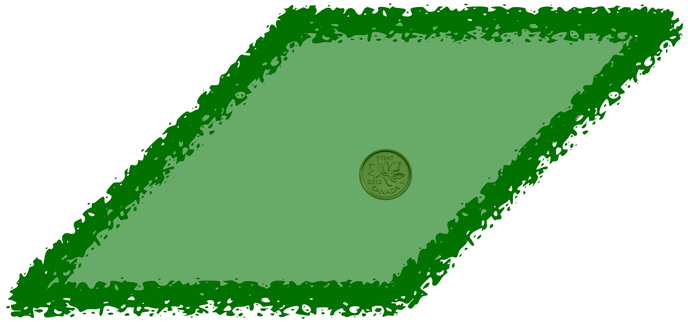
Did you notice, the value of ‘k’ will affect the predictions?
Selection of k is very important and we will learn how to select k for our dataset in the practice section.

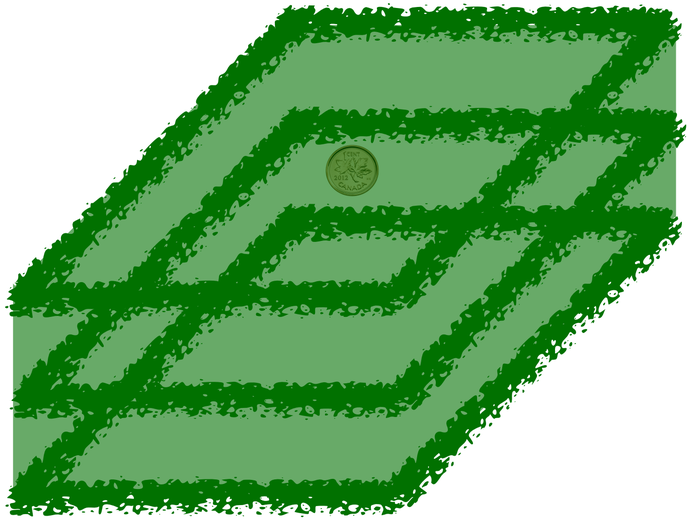
SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

KNN is simple and works quite well with small number of input variables (features or dimensions), provided it is given a good distance metric and has enough labeled training data. However, the main problem with KNN is that it does not work well with high dimensional inputs.The poor performance in high dimensional settings is due to the **curse of dimensionality**.

Straight line 100 yards*
(1 dimension / feature)





Cube , 100 yards* each side
(3 dimensions / features)

* *Scale, 100 yards, is used for understanding only, all the dimension, especially in cube are not properly scaled.*

SCIENCE ACADEMY
Excellence in Teaching & Learning

K-Nearest Neighbors - KNN

KNN is simple and works quite well with small number of input variables (features or dimensions), provided it is given a good distance metric and has enough labeled training data. However, the main problem with KNN is that it does not work well with high dimensional inputs.The poor performance in high dimensional settings is due to the **curse of dimensionality**.

When the number of features or dimensions is large, there tends to be a deterioration in the performance of KNN and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This **phenomenon is known as the curse of dimensionality**.

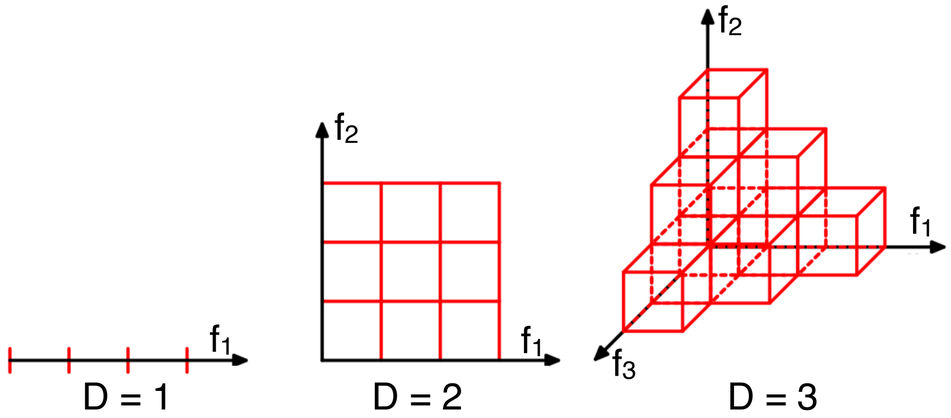
SCIENCE ACADEMY
Excellence in Teaching & Learning

Curse of dimensionally

Let's try to understand **curse of dimensionality** with some Illustrations:

Illustration* of the curse of dimensionality, showing how the number of regions of a regular grid grows exponentially with the dimensionality D of the space.

(For clarity, only a subset of the cubical regions are shown for D = 3)



(source page 36, Fig 1.21 from Pattern Recognition and Machine Learning by Christopher M. Bishop)

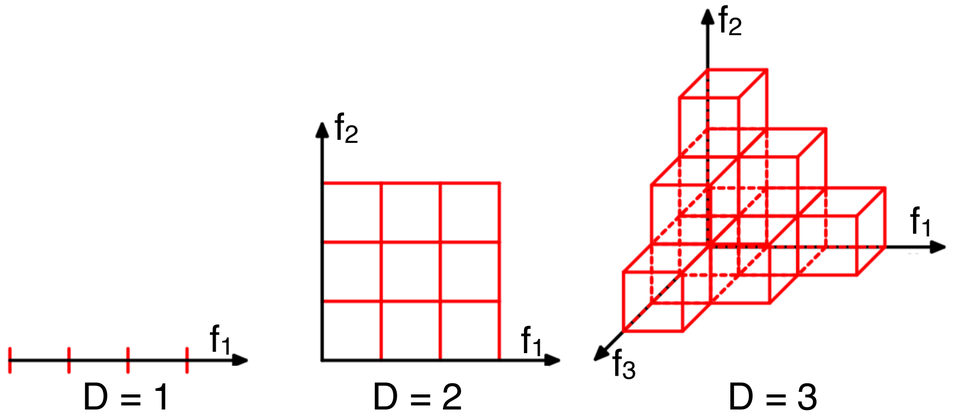
SCIENCE ACADEMY
Excellence in Teaching & Learning

Curse of dimensionally

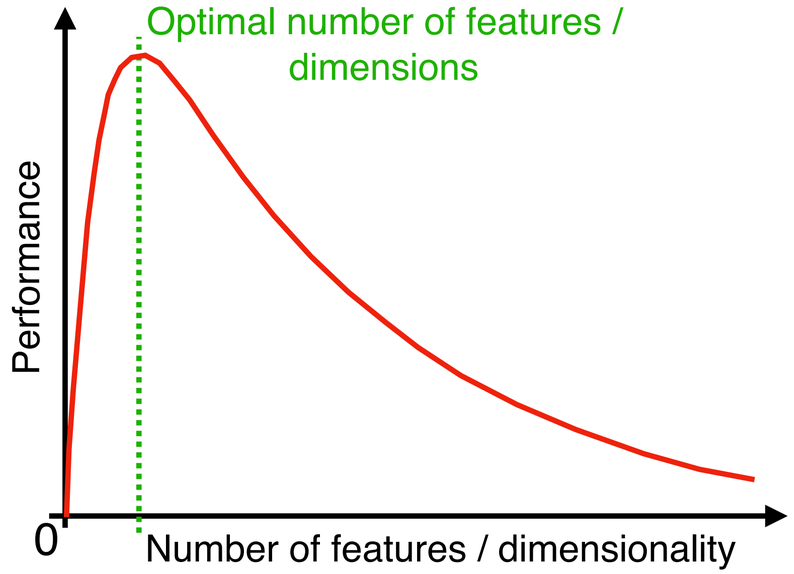
Let's try to understand **curse of dimensionality** with some Illustrations:

Illustration* of the curse of dimensionality, showing how the number of regions of a regular grid grows exponentially with the dimensionality D of the space.

(For clarity, only a subset of the cubical regions are shown for D = 3)

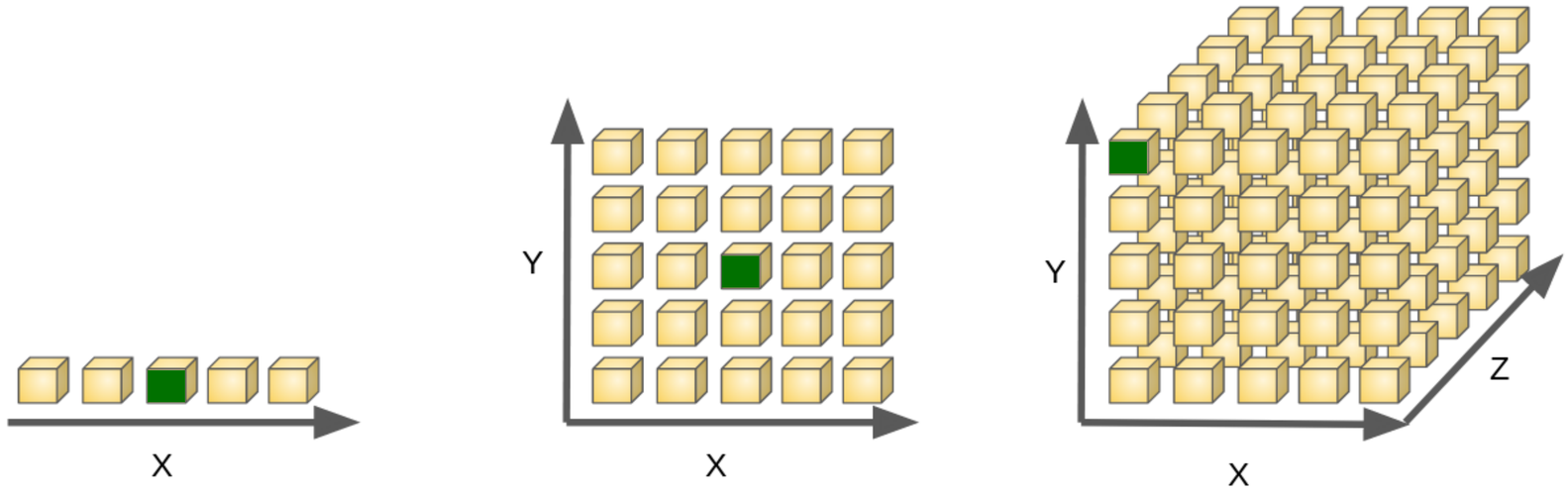


As the dimensionality increases, the classifier's performance increases until the optimal number of features is reached. Further increasing the dimensionality without increasing the number of training samples results in a decrease in classifier performance.



(source page 36, Fig 1.21 from Pattern Recognition and Machine Learning by Christopher M. Bishop)

SCIENCE ACADEMY
Excellence in Teaching & Learning



More the dimensions would be, harder it would be to search the treasure box, “ ” because more boxes would be empty!

This picture above may help to understand curse of dimensionality!

Ok, let’s move on and learn by doing!