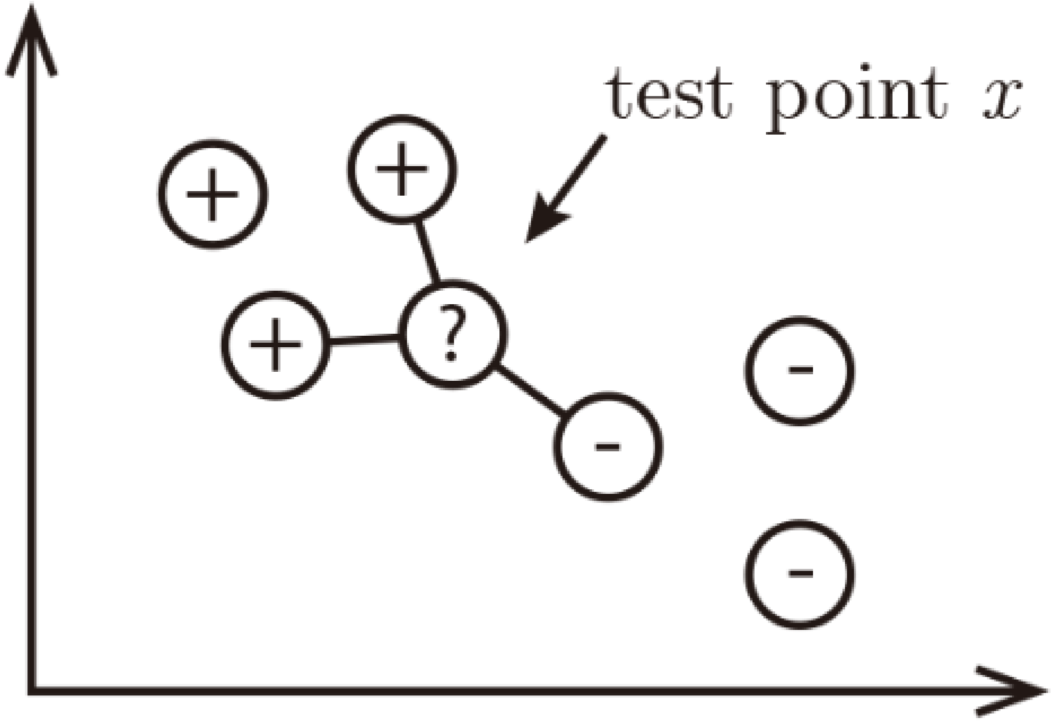
Hardware for Machine Learning

Lecture 1:
KNN (k-Nearest Neighbors)

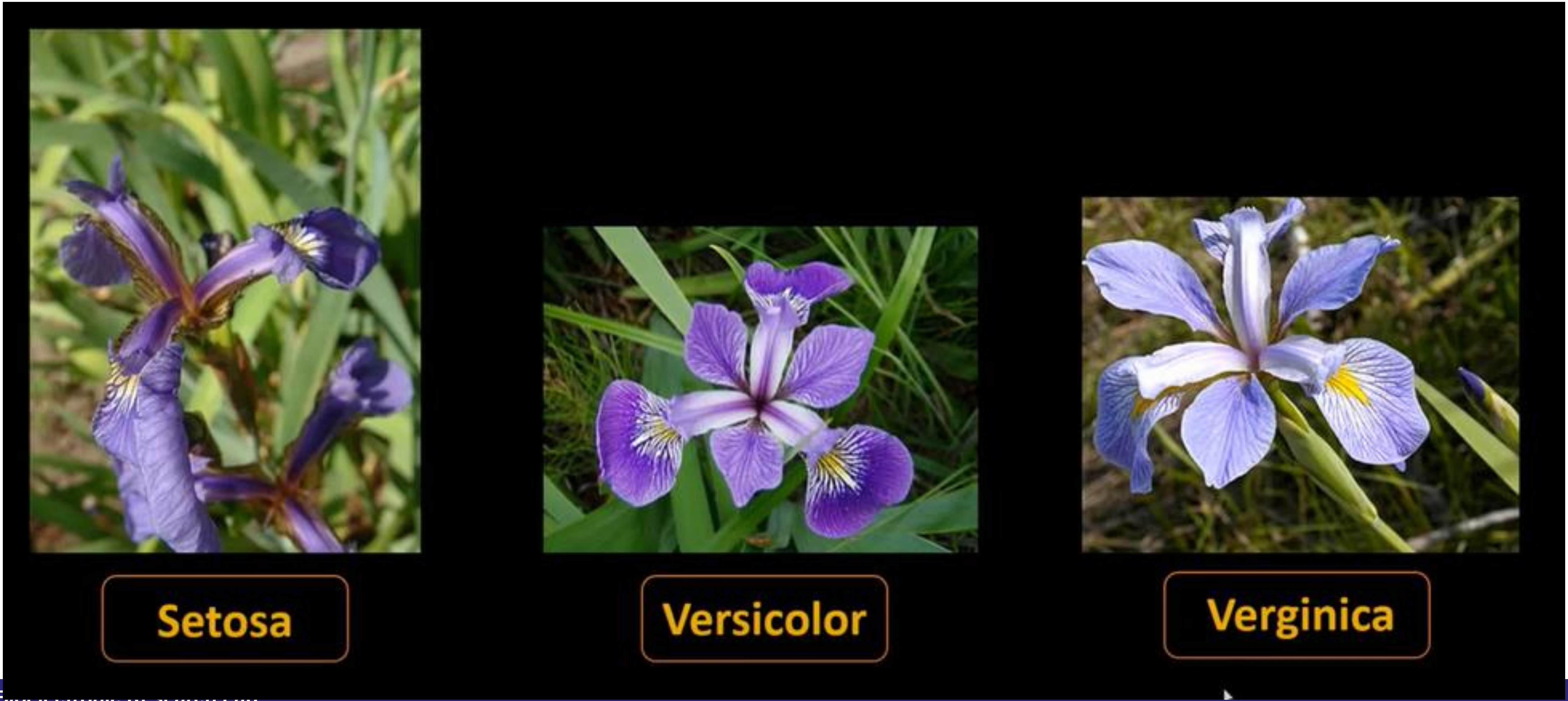
Hao Jiang
School of Engineering
San Francisco State University

What is KNN?

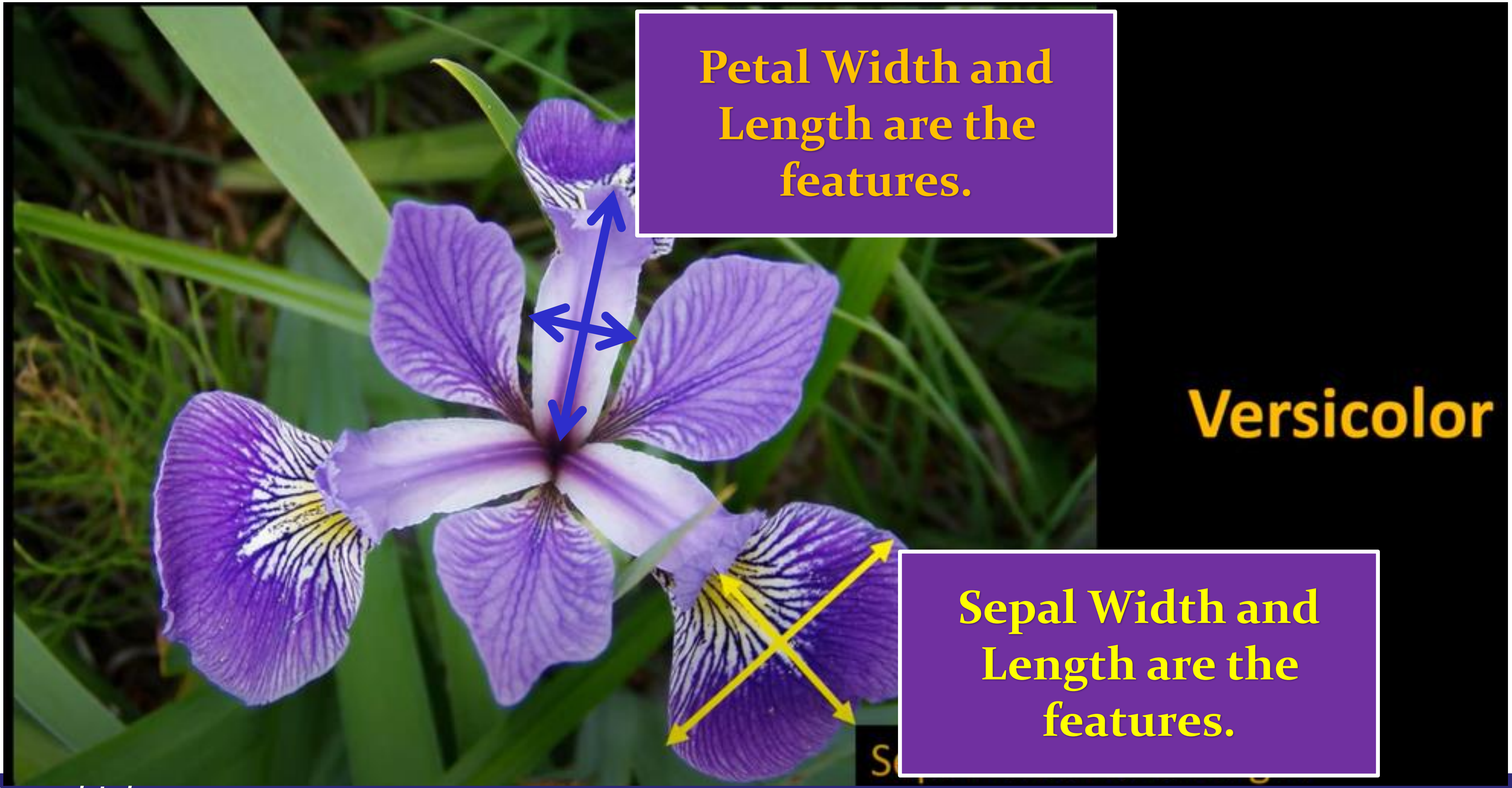
- KNN stores all the available cases with the ground truth label
- KNN classifies a new case based on a similarity measure
  - The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.



Example: Identify 3 Iris Flowers



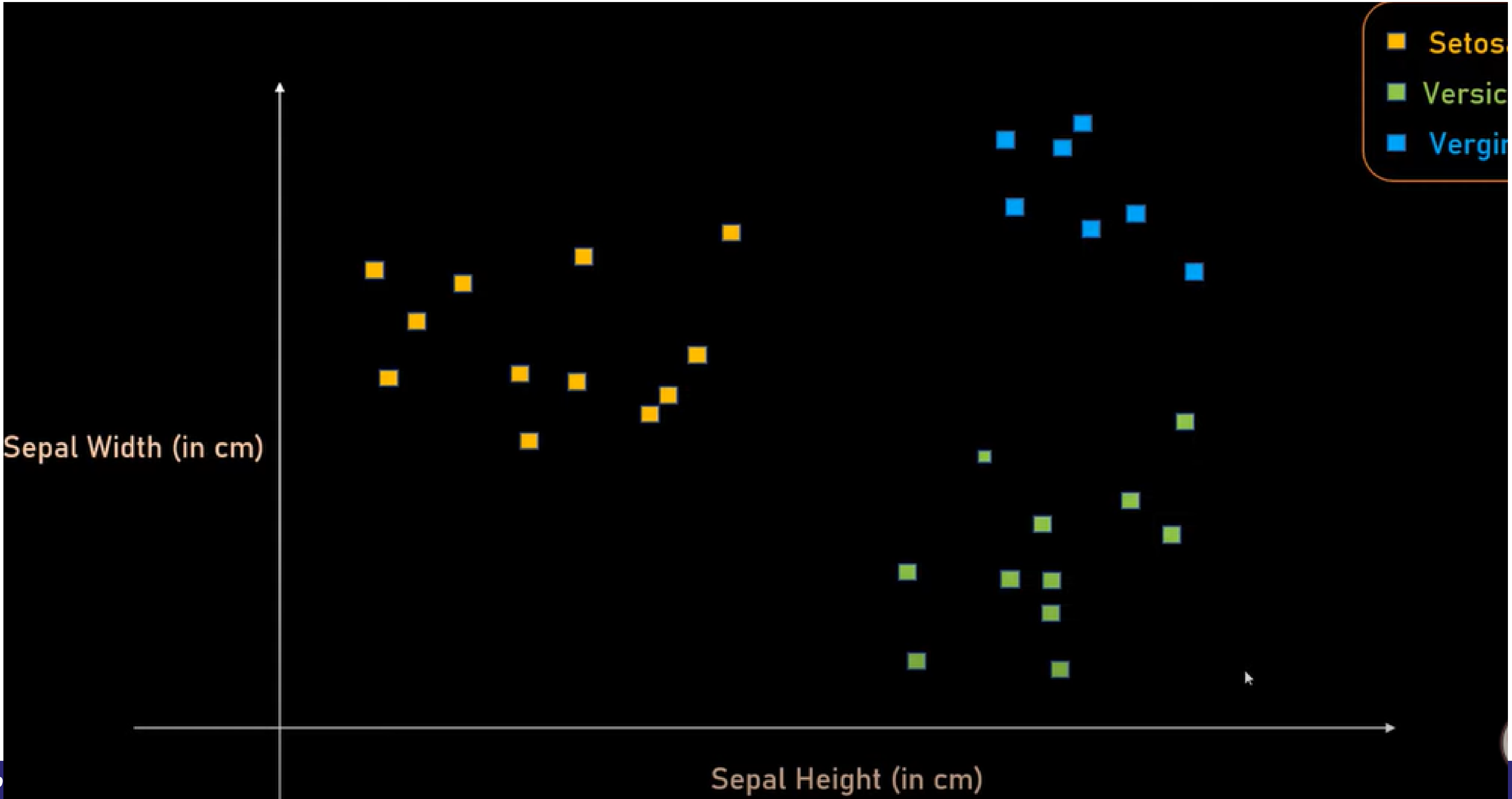
Establish the Data: Features



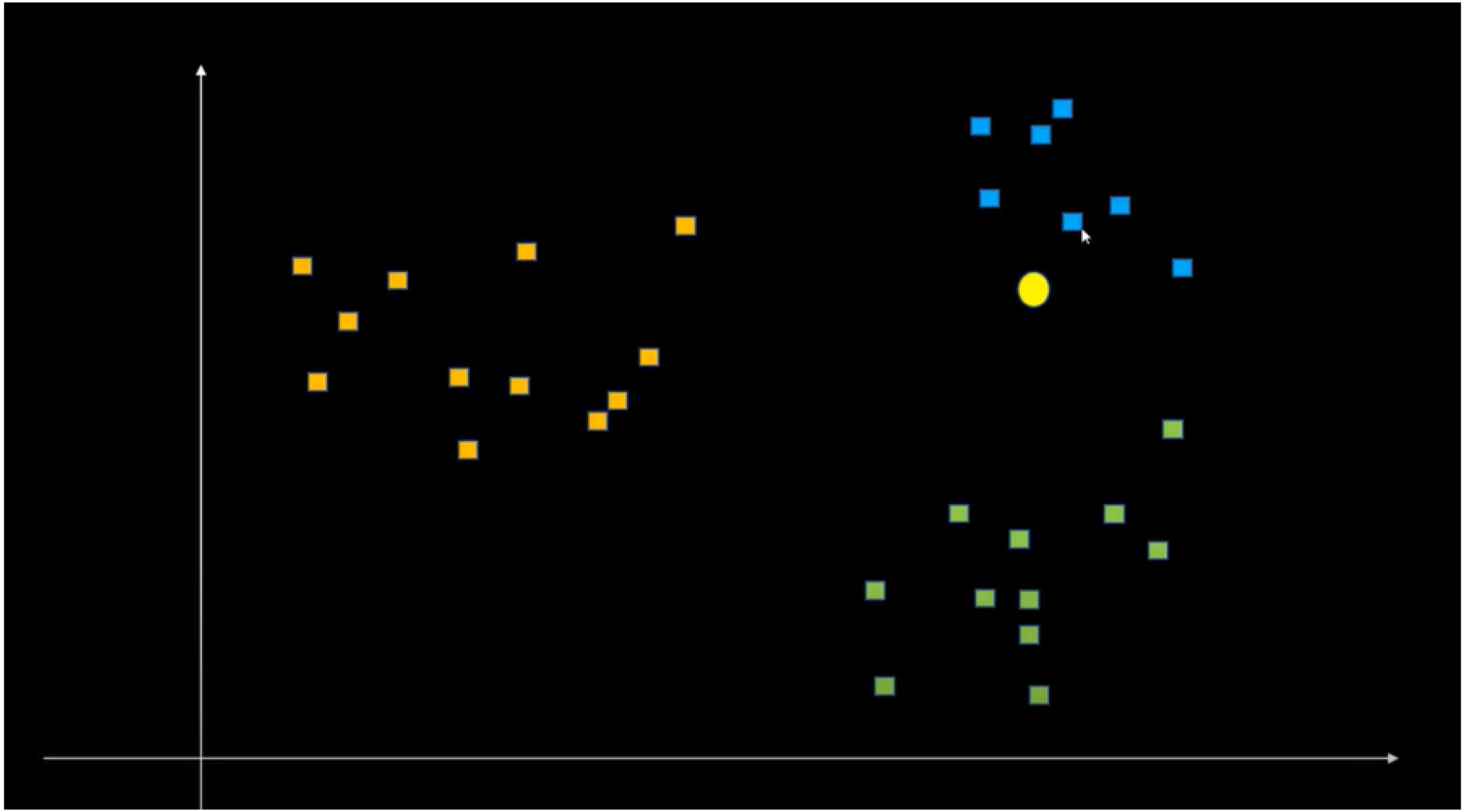
Petal Width and Length are the features.

Sepal Width and Length are the features.

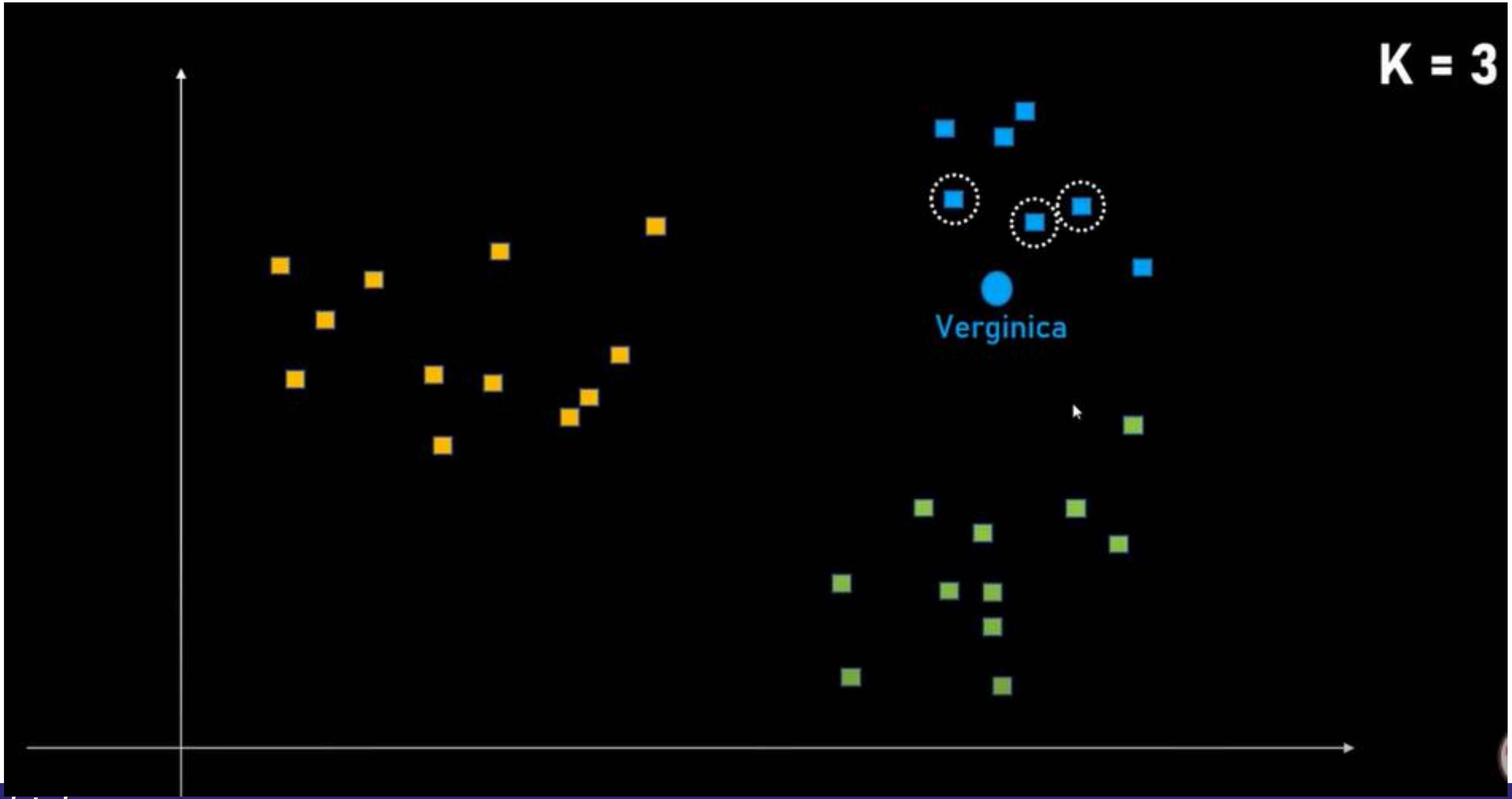
Add Label to Data



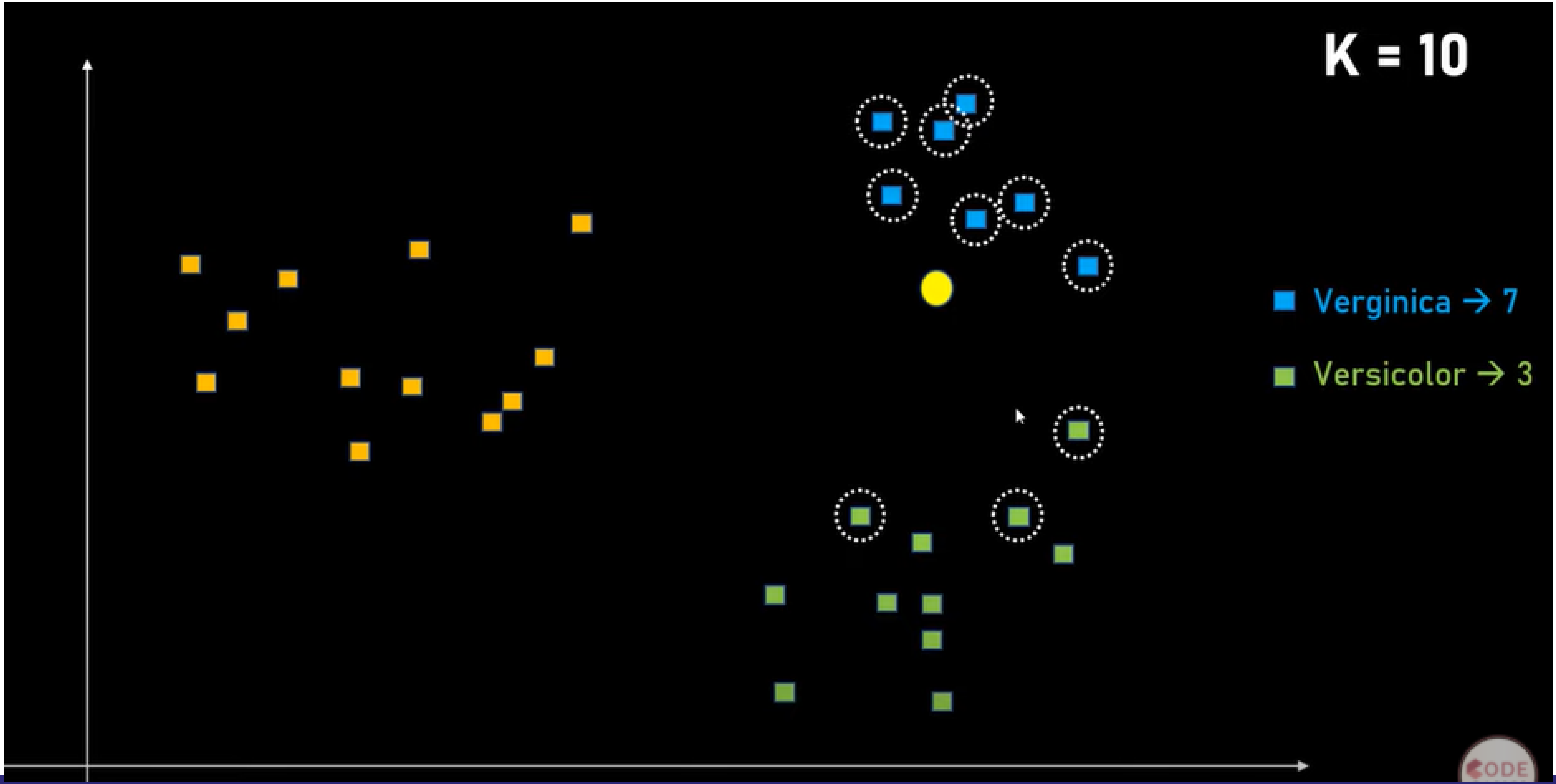
An unknown data point



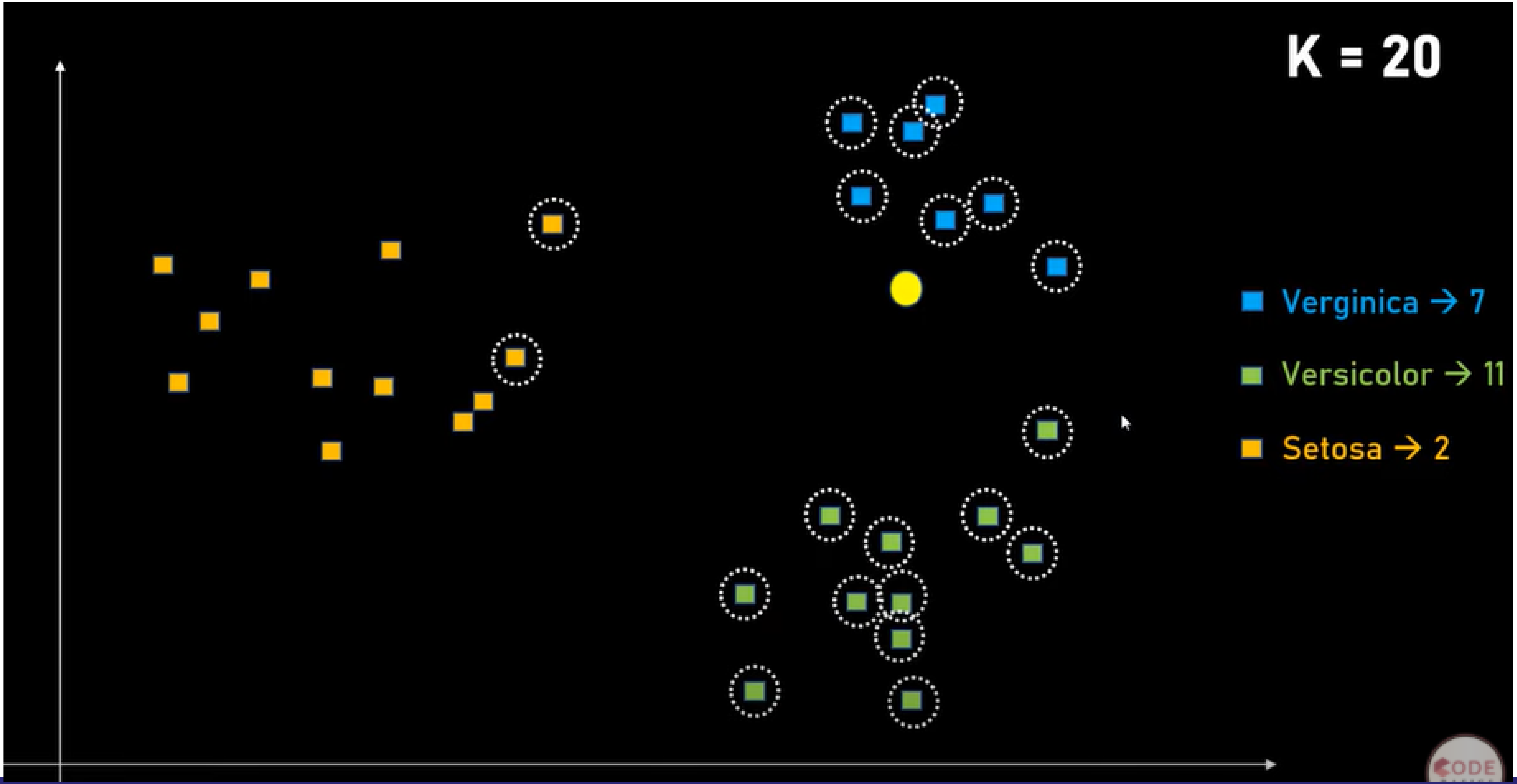
K: how many data points to be considered



K=10



K=20



The KNN Algorithm

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

K-stands for?

- The k-nearest neighbors algorithm uses a very simple approach to perform classification. When tested with a new example, it looks through the training data and finds the K training examples that are closest to the new example. It then assigns the most common class label.

- K is therefore just the number of neighbors "voting" on the test example's class.

- If K=1, then test examples are given the same label as the closest example in the training set. If K=3, the labels of the three closest classes are checked and the most common (i.e., occurring at least twice) label is assigned, and so on for larger K.

Choosing the right value for K

- As we decrease the value of K to 1, our predictions become less stable.
  - Just think for a minute, imagine K=1 and we have a query point surrounded by several reds and one green (I’m thinking about the top left corner of the colored plot above), but the green is the single nearest neighbor. Reasonably, we would think the query point is most likely red, but because K=1, KNN incorrectly predicts that the query point is green.
- Inversely, as we increase the value of K, our predictions become more stable due to majority voting / averaging, and thus, more likely to make more accurate predictions (up to a certain point). Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far.
- In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.

Summary (1)

- The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

- A supervised machine learning algorithm (as opposed to an unsupervised machine learning algorithm) is one that relies on labeled input data to learn a function that produces an appropriate output when given new unlabeled data.

- Supervised machine learning algorithms are used to solve classification or regression problems. A classification problem has a discrete value as its output. For example, “likes pineapple on pizza” and “does not like pineapple on pizza” are discrete. There is no middle ground.

Summary (2)

- Advantages
  - The algorithm is simple and easy to implement.
  - There’s no need to build a model, tune several parameters, or make additional assumptions. ***No training is needed!***
  - The algorithm is versatile. It can be used for any classification.

- Disadvantages
  - The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

Numerical Method for KNN

- Python code “distance_3D”

- Excel file “KNN_method”