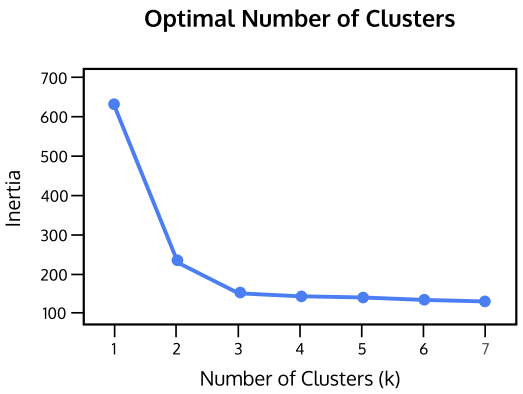
Unsupervised Learning Interview Question

K-Means: Inertia

Inertia measures how well a dataset was clustered by K-Means. It is calculated by measuring the distance between each data point and its centroid, squaring this distance, and summing these squares across one cluster.

A good model is one with low inertia AND a low number of clusters (K). However, this is a tradeoff because as K increases, inertia decreases.

To find the optimal K for a dataset, use the *Elbow method*; find the point where the decrease in inertia begins to slow. $K=3$ is the "elbow" of this graph.



Unsupervised Learning Basics

Patterns and structure can be found in unlabeled data using *unsupervised learning*, an important branch of machine learning. *Clustering* is the most popular unsupervised learning algorithm; it groups data points into clusters based on their similarity. Because most datasets in the world are unlabeled, unsupervised learning algorithms are very applicable.

Possible applications of clustering include:

- Search engines: grouping news topics and search results
- Market segmentation: grouping customers based on geography, demographics, and behaviors

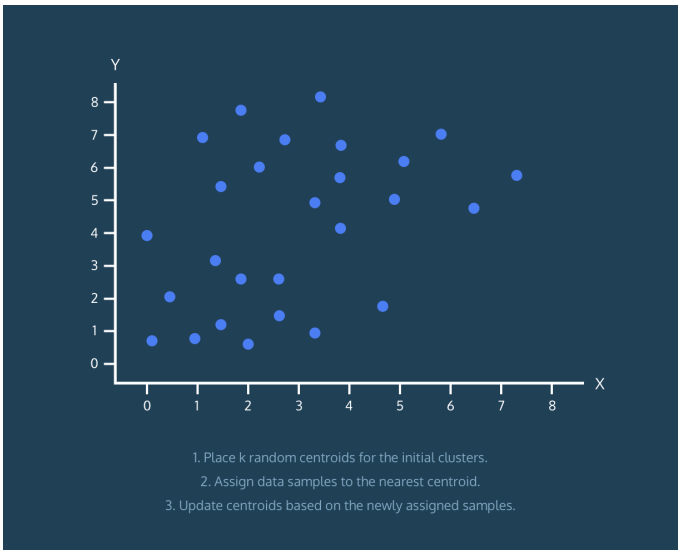
codecademy

K-Means Algorithm: Intro

K-Means is the most popular clustering algorithm. It uses an iterative technique to group unlabeled data into K clusters based on cluster centers (*centroids*). The data in each cluster are chosen such that their average distance to their respective centroid is *minimized*.

1. Randomly place K centroids for the initial clusters.
2. Assign each data point to their nearest centroid.
3. Update centroid locations based on the locations of the data points.

Repeat Steps 2 and 3 until points don't move between clusters and centroids stabilize.



K-Means Using Scikit-Learn

Scikit-Learn, or `sklearn`, is a machine learning library for Python that has a K-Means algorithm implementation that can be used instead of creating one from scratch.
To use it:

- Import the `KMeans()` method from the `sklearn.cluster` library to build a model with `n_clusters`
- Fit the model to the data samples using `.fit()`
- Predict the cluster that each data sample belongs to using `.predict()` and store these as `labels`

```
from sklearn.cluster import KMeans

model = KMeans(n_clusters=3)

model.fit(data_samples)

labels = model.predict(data_samples)
```

Inspecting Variable Types

One of the most important first steps when working with a dataset is to inspect the variable types, and identify relevant variables. An efficient method to use when inspecting variables is the `.head()` method which will return the first rows of a dataset.

```
print(df.head())
```

codecademy

One-Hot Encoding with Python

When working with nominal categorical variables in Python, it can be useful to use One-Hot Encoding, which is a technique that will effectively create binary variables for each of the nominal categories. This encodes the variable without creating an order among the categories. To one-hot encode a variable in a pandas dataframe, we can use the `.get_dummies()` .

```
df = pd.get_dummies(data = df, columns=
['column1', 'column2')
```

Print Share