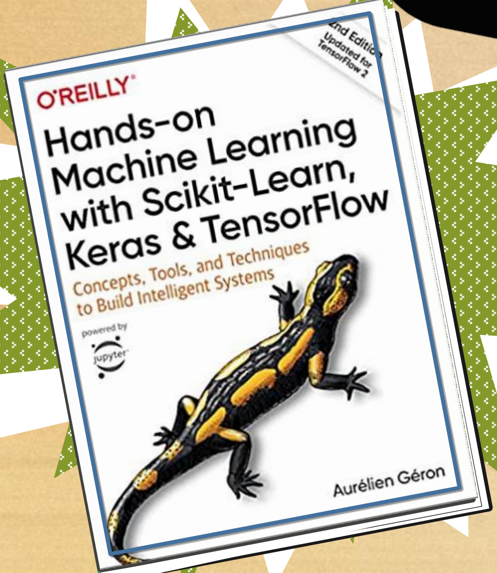
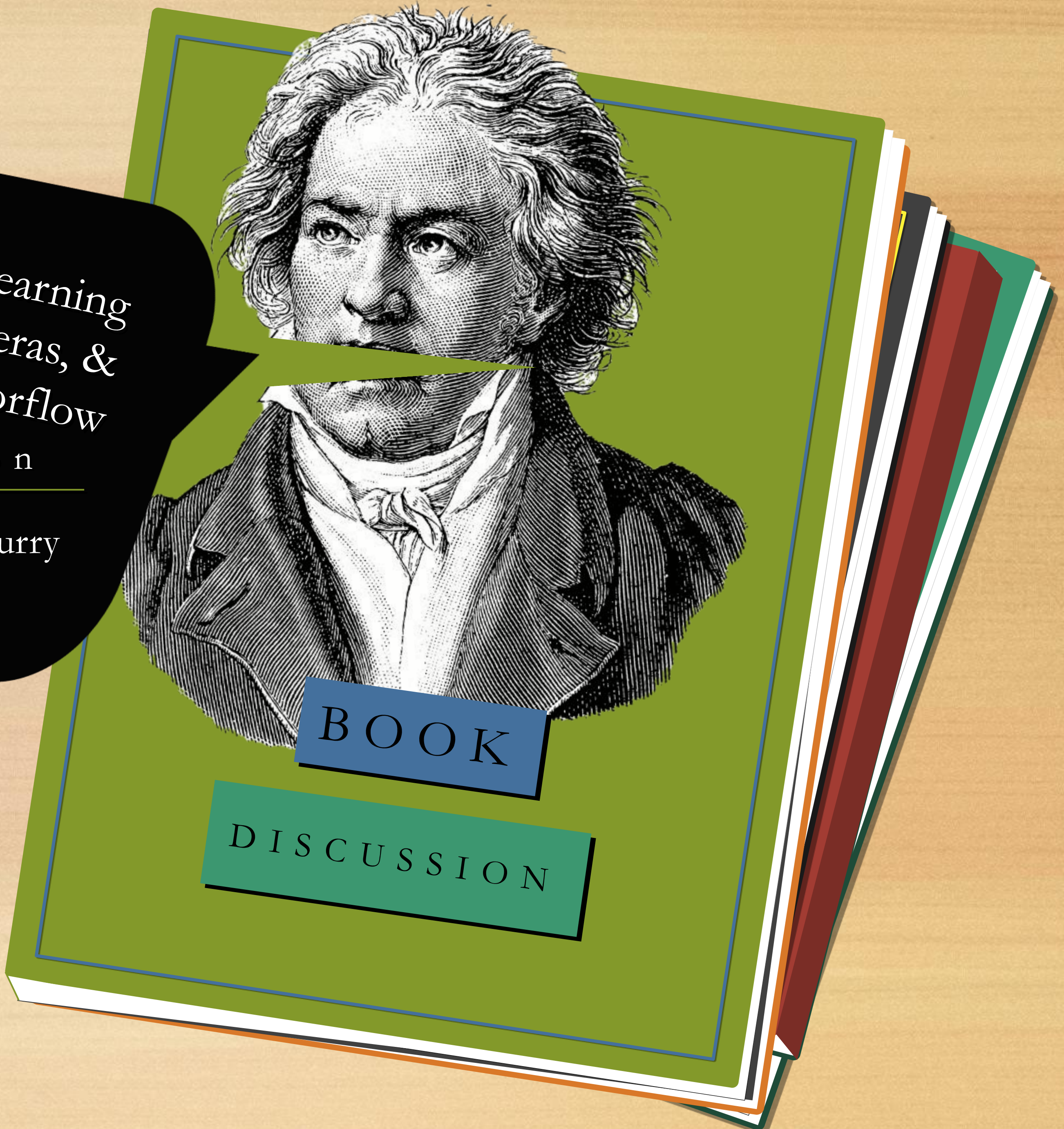
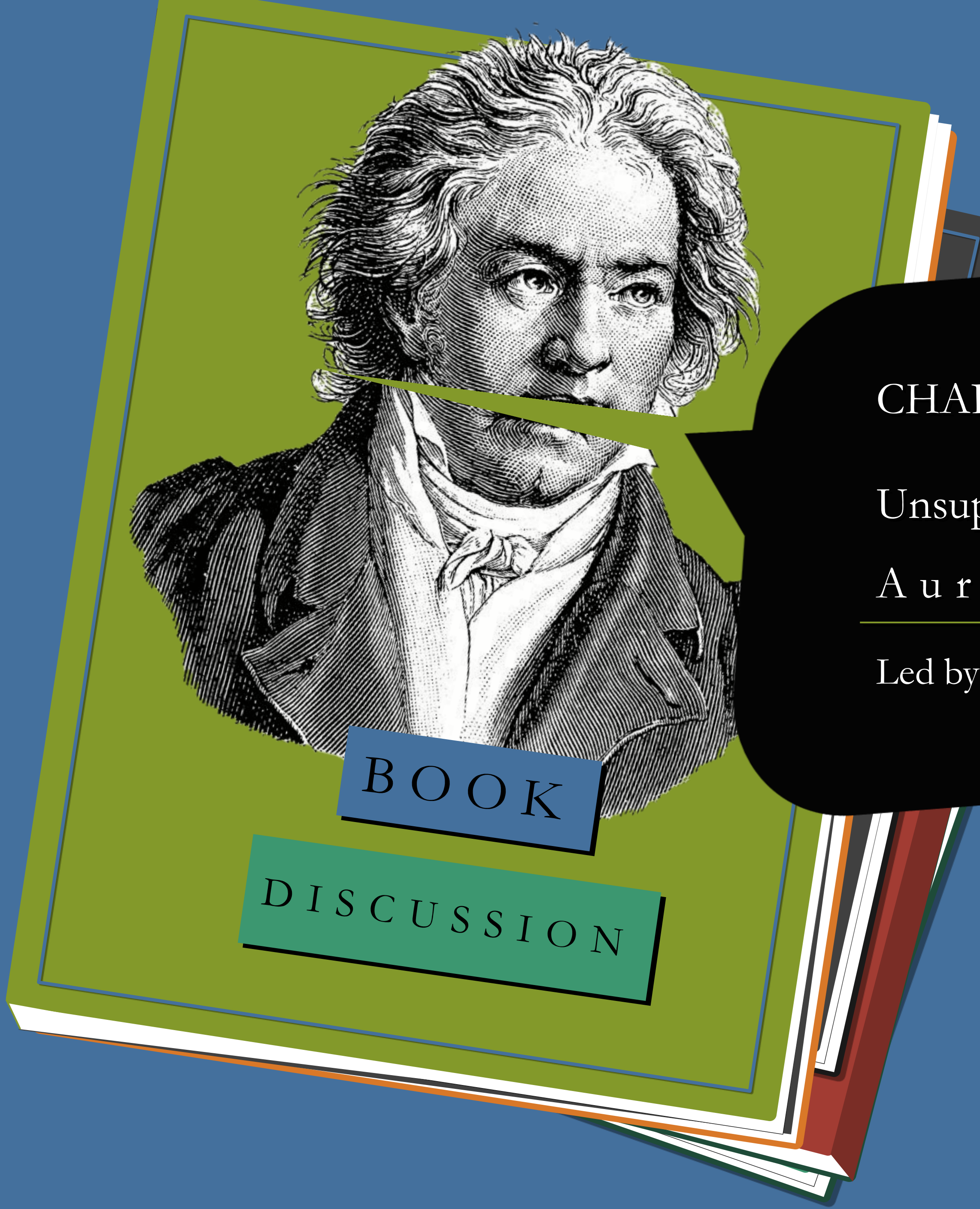
Hands-on Machine Learning with Scitkit-Learn, Keras, & Tensorflow

Aurélien Géron

Discussion led by: Kalika Kay Curry







CHAPTER 9:

Unsupervised Learning

Aurélien Géron

Led by: Kalika Kay Curry

Unsupervised Learning

Modeling Unlabeled Data

Why Unsupervised Learning

- Unsupervised Learning – High Potential
- Works to cloister/compare unlabeled instances (*manufacturing example*).
- Dimensionality Reduction is the most common form of unsupervised learning method.

Types of Unsupervised Learning

- Clustering, objects are grouped together
- Anomaly Detection – learn what's normal to find what's abnormal.
- Density estimation – Probability density function (PDF) is estimated of the random process that generated the dataset. Used for anomaly detection, analysis, and visualization.

Part I: Clustering

Focused on clustering unsupervised learning models

What is Clustering

- There is no real definition of what a cluster is; must consider context.
- Assigns instances to groups, similar to supervised classification models, but without labels.
- There are different ways to cluster
  - random points as a centroid
  - run around in a circle
  - bottom up or top down approach (hierarchy).

Why Cluster

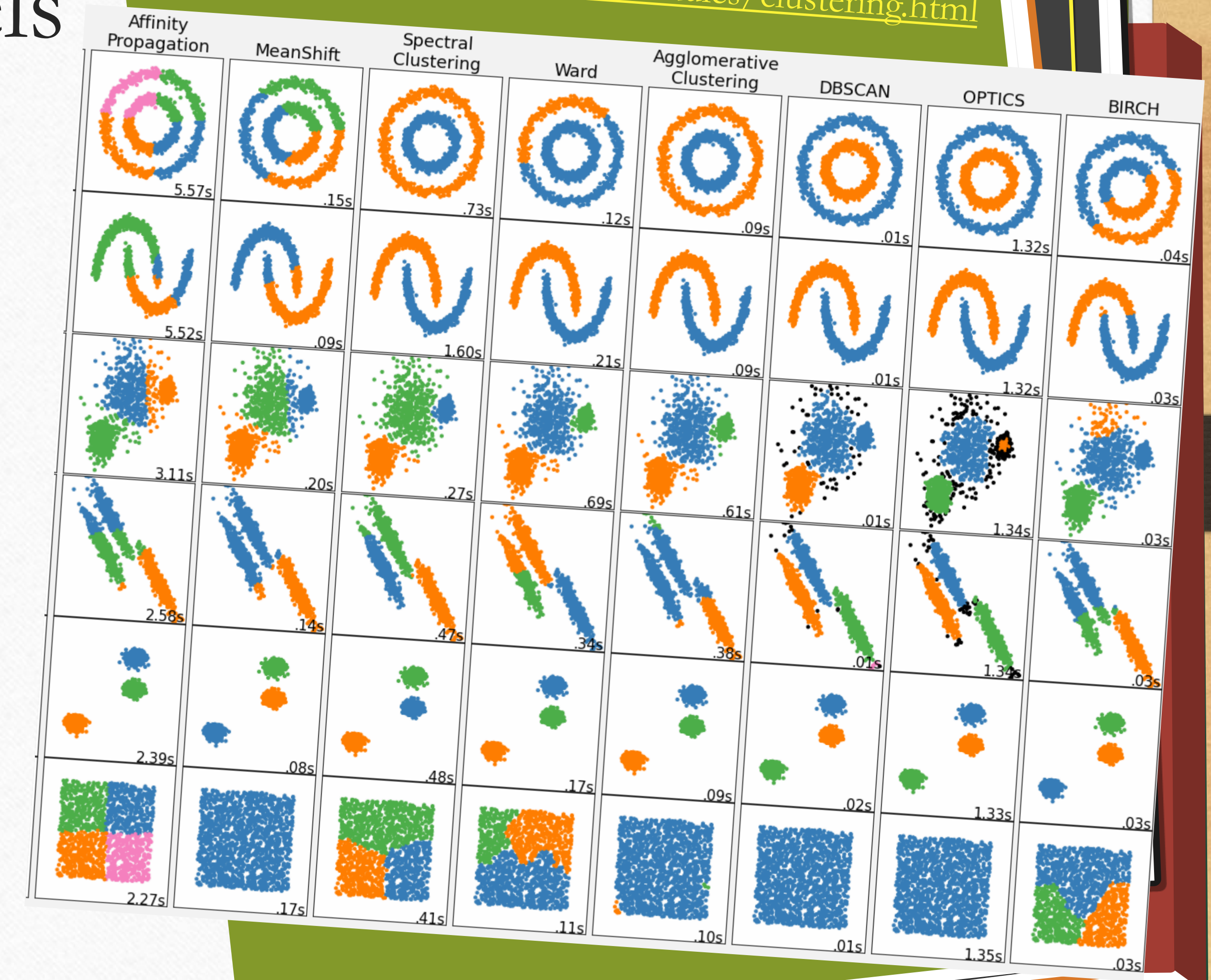
- Dimensionality reduction (Preprocessing)
  - Include cluster in the pipeline.
  - GridSearchCV for best k value
- Semi supervised learning
  - Train a dataset from labeled clusters
- Segment an image
  - Color segmentation

- Customer segmentation
- Data analysis
- Anomaly detection (outlier detection)
- Search engines

Types of Clustering Models

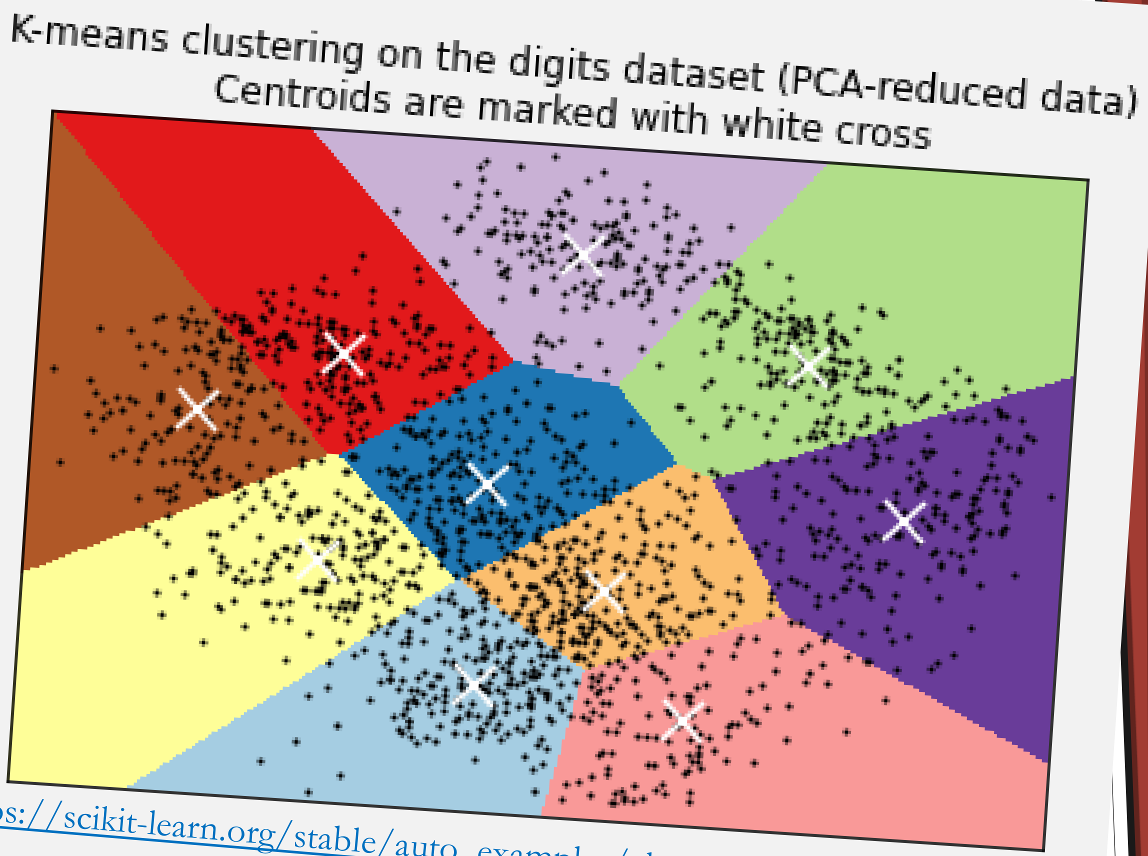
- K-Means
- DBScan
- Agglomerative – Hierarchical, matrix required for large datasets.
- BIRCH – Hierarchical, designed for large datasets.
- Mean-Shift – not suited for large datasets, similar to dbscan
- Affinity Propagation – not suited for large datasets. Uses a voting mechanism.
- Spectral – combo-unsupervised learning method. Combines an embedded dimensionality reduction with another unsupervised learning. Used for complex data structures and to cut graphs.

https://scikit-learn.org/stable/modules/clustering.html



K-Means History

- Best described with a Voronoi diagram (right).
- AKA Lloyd-Forgy
  - Developed in 1957 by Lloyd (copyright)
  - Developed in 1965 by Forgy
- In 2006, Arthur and Vassilvitskii provided introduced a Kmeans++, faster way of identifying centroids and is the default.
- There are a couple of other varieties such as Mini-batch K-Means; is faster, but has more inertia.
- Requires the n_clusters parameter (k)



https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html

K-Means Usage

- Reminder: Scale the data
- Cluster amount
  - Inertia elbow chart
  - Silhouette line graph
  - Silhouette diagram
- Limited Solution
  - We’re all a little limited.
  - Ladybug example from image segmentation.
  - Does not perform well with
    - Non spherical shapes (whatever that means)
    - Different densities
    - Varying sizes

- Inertia identifies the best solution (best centroid location).
  - Mean squared distance between each instance and its closest centroid
- Getting lucky: Centroid initialization methods *(Risk Mitigation)*
  - Random centroid initializations can produce suboptimal solutions as a convergence occurs at random.
  - Kmeans++ Algorithm
- n_init has a default setting and determines how many times the centroids should be selected for the optimal solution.
- If you know the initialization points for the clusters, you can set the init parameter manually. with an n_init parameter set to one.

Types of Clustering

```
#function from chapter 5 (no plot)
def gaussian_rbf(x, landmark, gamma):
    return np.exp(-gamma * np.linalg.norm(x - landmark, axis=1)**2)
```

```
x1_example = X1D[3, 0]
for landmark in (-2, 1):
    k = gaussian_rbf(np.array([[x1_example]]), np.array([[landmark]]), gamma)
    print("Phi({}, {}) = {}".format(x1_example, landmark, k))
```

```
Phi(-1.0, -2) = [0.74081822]
Phi(-1.0, 1) = [0.30119421]
```

Conversely to a distance transformation we can use a similarity function.

Hard Clustering and Soft Clustering

- Hard Clustering
  - is one instance is assigned to a cluster
- Soft Clustering
  - assigns each instance a score per cluster.
  - Can be the distance from the centroid, which can be found using the transform method; which represents the Euclidian distance of an instance form each centroid.
  - Can be the a similarity/affinity score such as the Gaussian Radial Basis Function. (chapter 5)
  - The transform method can be used to gather the distance between each instance to each centroid. It is in fact, the Euclidian distance.

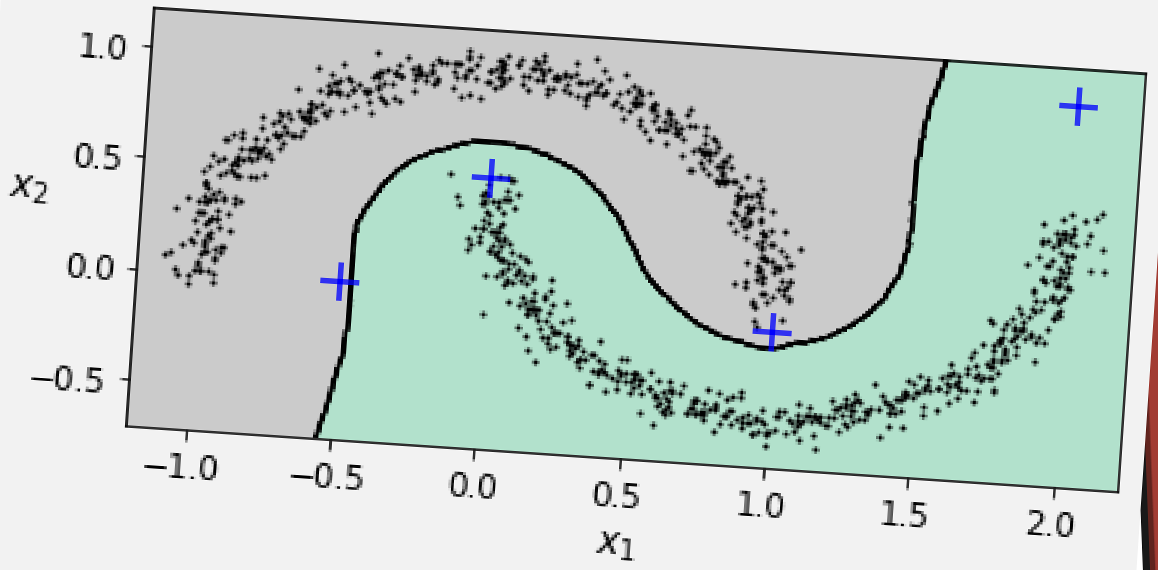
- Note: the similarity/affinity function is not provided as an example in the github solutions/notebook provided to the author.
- The affinity function states that the number of features will increase, drastically with an extremely large training set. What does this mean for dimensionality reduction; when we're working with clustering? Can we expect to see that the similarity/affinity results would be overkill or over extensive?

DBScan

https://scikit-learn.org/stable/modules/clustering.html#dbscan

- Continuous regions of high density.
- Epsilon neighborhood a count of the number of instances that are within a small distance from it.
- Min_samples: if it has at least these number of samples, it's a core instances.
- If you're in the same neighborhood; you belong to the same cluster. This can include many core instances, so if there is a long running instance; you're all one.
- If it's not a core instance and is not in the neighborhood, it's an outlier, represented with a -1.
- No predict method; a classifier is better at predicting the cluster the data may belong to (*knn example to the right*)

https://github.com/alexhegit/handson-ml2/blob/master/09_unsupervised_learning.ipynb



Part II: Gaussian Mixture Modeling (GMM)

Focused on Gaussian Mixture Modeling (GMM).

Thank You

Kalika Kay Curry

Kalika_kay@hotmail.com