Dimensional reduction and clustering

Szu-Chi Chung

Department of Applied Mathematics, National Sun Yat-sen University

How to select/extract important feature after cleaning?

- First encountering a cleaned dataset can sometimes feel overwhelming. You might be presented with hundreds or thousands of features without even a description to go by. Where do you even begin?
- *Feature selection* is the process of selecting a subset of relevant features for use in model construction. It is used for several reasons:
  - Simplification of models to make them easier to interpret by researchers/users
  - Shorter training times
  - To avoid the curse of dimensionality
- The central premise is that the data contains some features that are either *redundant* or *irrelevant*, and can thus be removed without incurring much loss
- It should be distinguished from *feature extraction*. The latter creates new features from functions of the original features

How to select/extract important feature after cleaning?

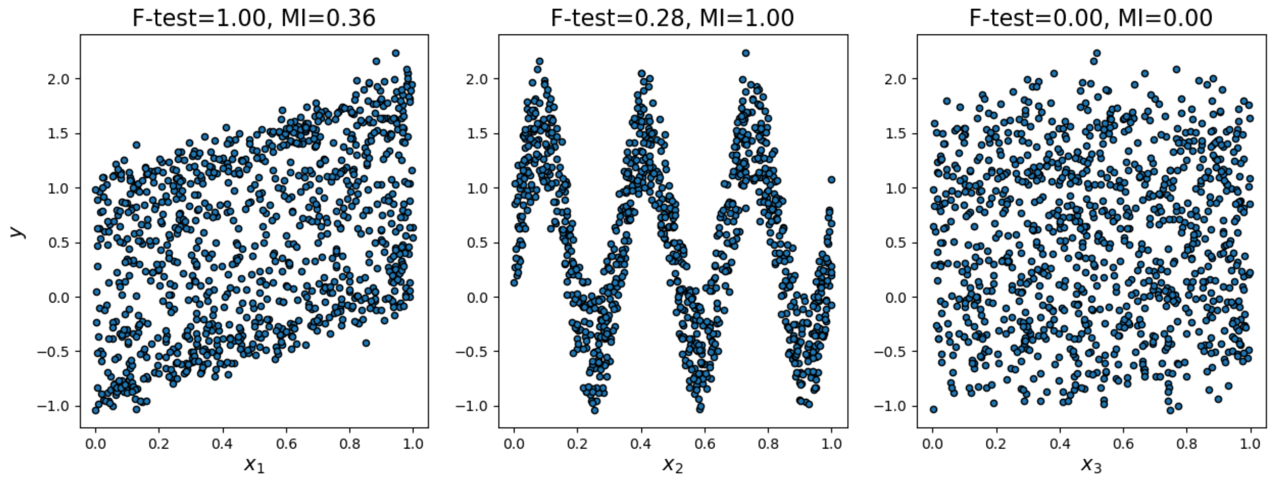
- Univariate and subset selection: We identify a subset of the p predictors that we believe to be related to the response. We then fit a model on the reduced set of variables
- Select from model using importance metrics: We fit a model involving all p predictors, but the estimated coefficients or the importance of features are used to rank the features
- Select from model shrinkage: We fit a model involving all p predictors, but the estimated coefficients are shrunken towards zero
- Dimension Reduction/Clustering: We project/cluster the p predictors into a M-dimensional subspace, where $M < p$. Then these M projections/clusters are used as predictors to fit

1. Univariate feature selection

- A great first step is to construct a ranking with a feature utility metric, a function measuring associations between a feature and the target
  - The simple baseline is to removes all features whose variance doesn't meet some threshold. For instance, zero-variance features (have the same value in all samples) can be removed
- Univariate feature selection works by selecting the best features based on univariate statistical tests
  - Select *K* best features: removes all but the highest scoring features
  - Select by percentile: removes all but a user-specified highest scoring percentage of features using common univariate statistical tests for each feature

Univariate feature selection

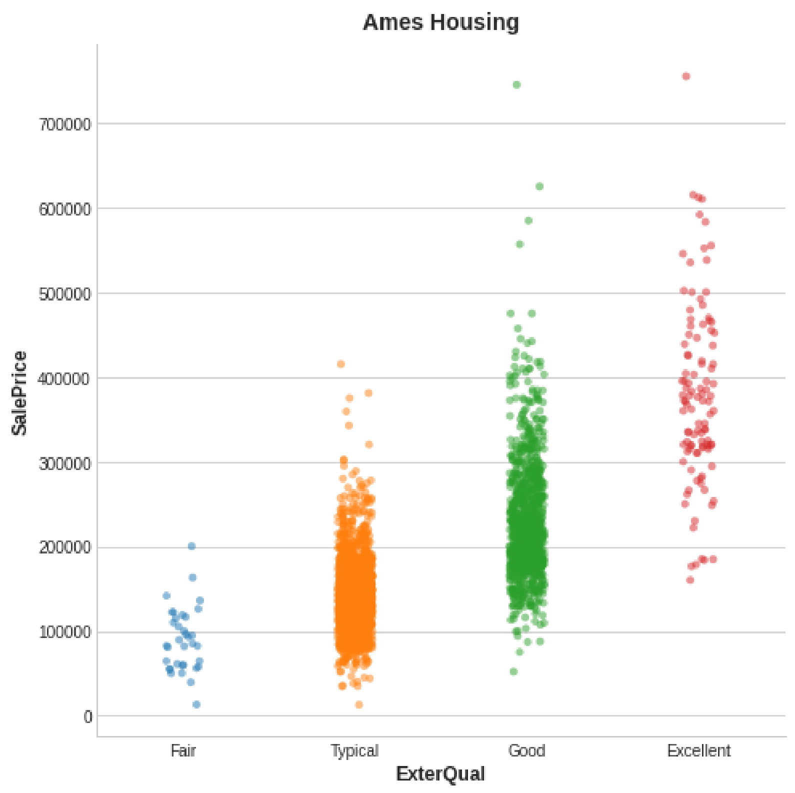
- The statistical test can return univariate scores or p-values:
  - For regression: F-test or mutual information
  - For classification: χ^2, F-test or mutual information
- The methods based on F-test estimate the degree of linear dependency between two random variables. On the other hand, mutual information methods can capture any kind of statistical dependency, but being nonparametric, they require more samples for accurate estimation



$$F = \left(\frac{n-p-1}{p}\right)\left(\frac{r^2}{1-r^2}\right)$$

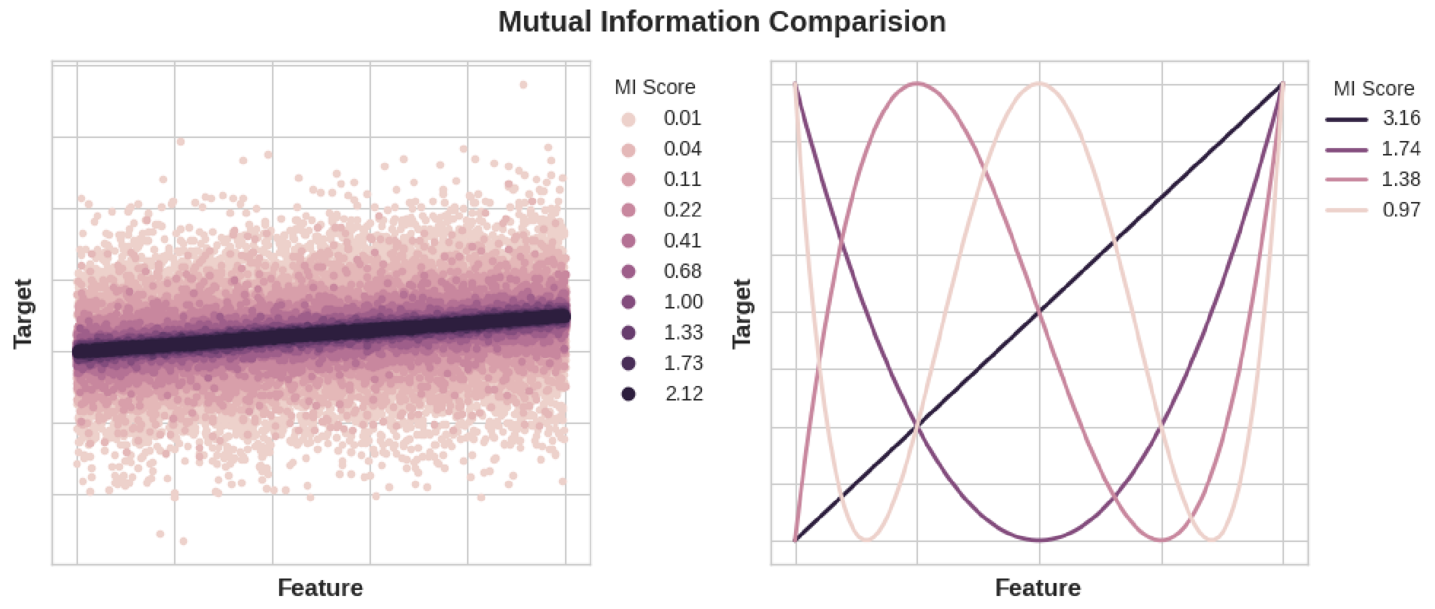
Univariate feature selection - Mutual information

- Mutual information (MI) is a lot like correlation in that it measures a relationship between two quantities
  - The advantage of MI is that it can detect any kind of relationship. MI describes relationships in terms of uncertainty. The MI between two quantities is a measure of the extent to which knowledge of one quantity reduces uncertainty about the other
  - We can see that knowing the value of *ExterQual* should make you more certain about the corresponding *SalePrice* - each category of *ExterQual* tends to concentrate *SalePrice* to within a certain range. The MI that *ExterQual* has with *SalePrice* is the **average reduction of uncertainty** in *SalePrice* taken over the four values of *ExterQual*



Univariate feature selection - Mutual information

- MI can help you to understand the relative potential of a feature as a predictor of the target, considered by itself. It's possible for a feature to be informative when interacting with other features, but not so informative all alone. **MI can't detect interactions between features**
- The actual usefulness of a feature depends on the model you use it with. A feature is only useful to the extent that its relationship with the target is *one your model can learn*. Just because a feature has a high MI score doesn't mean your model will be able to do anything with that information. You may need to transform the feature first to expose the association



Multivariate method - subset selection

Best subset and stepwise model selection procedures

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation
2. For $k = 1,2, \ldots p$:
   a) Fit all $\binom{p}{k}$ models that contain exactly k predictors
   b) Pick the best among these $\binom{p}{k}$ models, and call it M_k. Here best is defined as having the smallest RSS
3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated score, C_p (AIC), BIC, or adjusted R^2

Multivariate method - stepwise selection

- For computational reasons, best subset selection cannot be applied with large p
- Best subset selection may also suffer from statistical problems when p is large: larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data
    - Thus an enormous search space can lead to overfitting and high variance
- For both of these reasons, stepwise methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection
    - Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model
    - In particular, at each step the variable that gives the greatest additional improvement to the fit is added to the model

In detail

Forward Stepwise Selection

1. Let M_0 denote the null model, which contains no predictors
2. For $k = 0, \dots p - 1$:
   a) Consider all $p - k$ models that augment the predictors in M_k with one additional predictor
   b) Choose the best among these $p - k$ models, and call it M_{k+1}. Here best is defined as having the smallest RSS
3. Select a single best model from among $M_0, \dots, M_p$ using cross-validated prediction score, C_p (AIC), BIC, or adjusted R^2

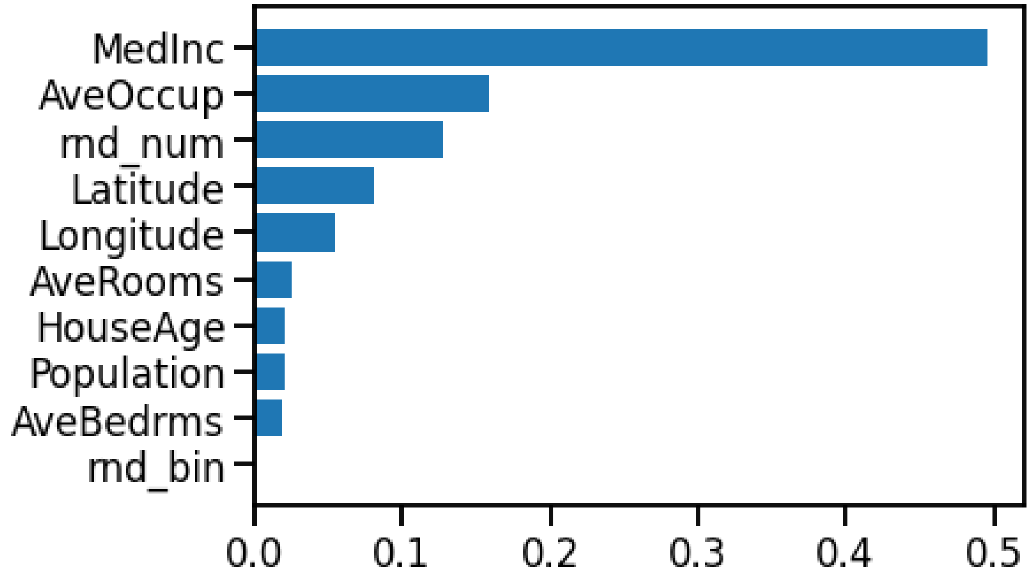
- Though forward stepwise selection considers $p(p + 1)/2 + 1$ models, it performs a guided search over model space, and so the effective model space considered contains substantially more than $p(p + 1)/2 + 1$ models

More on forward stepwise selection

- Computational advantage over best subset selection is clear
- It is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors
  - For instance, suppose that in a given data set with $p = 3$ predictors, the best possible one-variable model contains X_1, and the best possible two-variable model instead contains X_2 and X_3. Then forward stepwise selection will fail to select the best possible two-variable model, because M_1 will contain X_1, so M_2 must also contain X_1 together with one additional variable
- For high dimensional data with $p > n$, the forward selection can still be applied by considering only $M_1, \ldots, M_n$
  - In some model like linear regression, this will be a strength

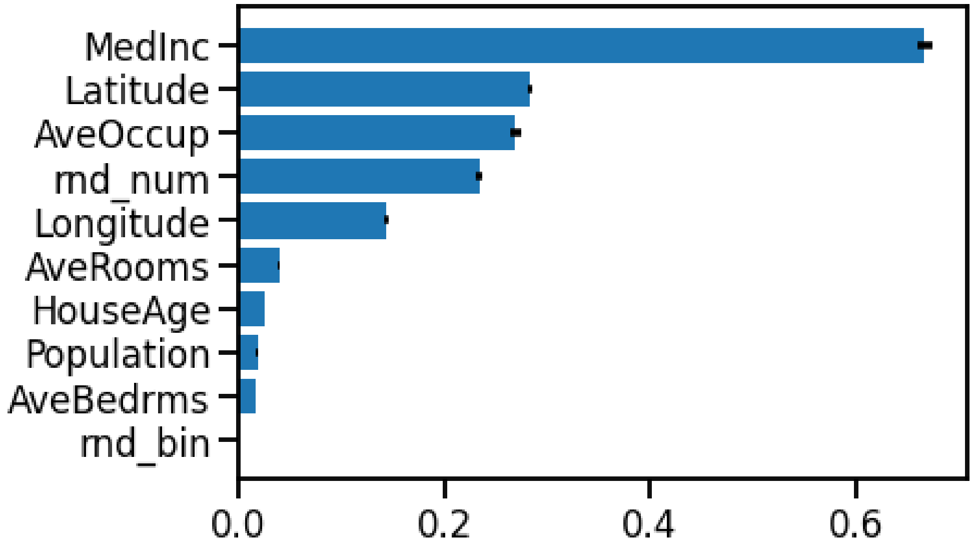
2. Select from model using importance metrics

- Any estimator that assigns importance to each feature through a specific attribute (such as feature importance)
  - The features are considered unimportant and removed if the corresponding importance of the feature values are below the provided *threshold* parameter
  - The importance of a feature is basically: how much this feature is used in each tree of the forest. Formally, it is computed as the (normalized) total reduction of the criterion brought by that feature
  - It will has a small bias toward high cardinality features (typically numerical features)



Select from model - Permutation importance

- A technique to evaluate the feature importance of any given fitted model. It basically shuffles a feature and sees how the model changes its prediction. Thus, the change in prediction will correspond to the feature importance
  - We will shuffle this specific feature, keeping the other feature as is, and run our same model (already fitted) to predict the outcome. *The decrease of the score shall indicate how the model had used this feature to predict the target*. The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled
  - On the contrary, if the feature is not used by the model, the score shall remain the same, thus the feature importance will be close to 0

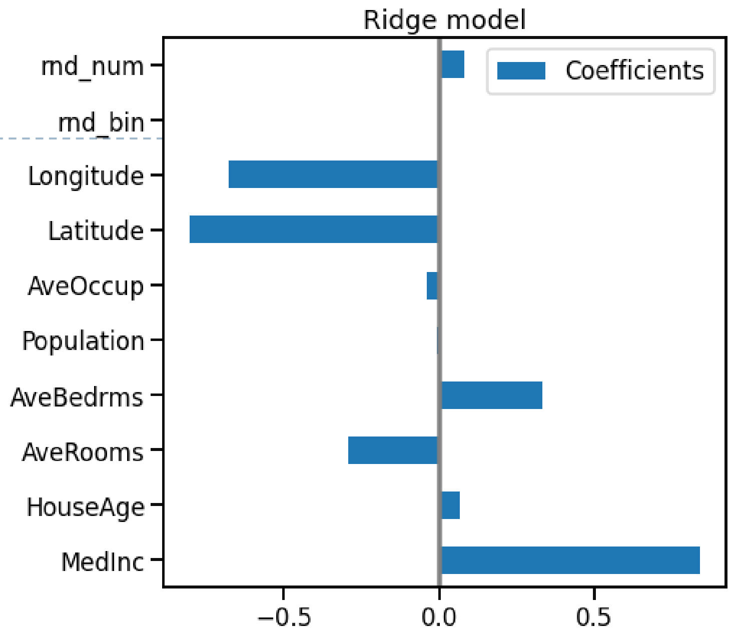


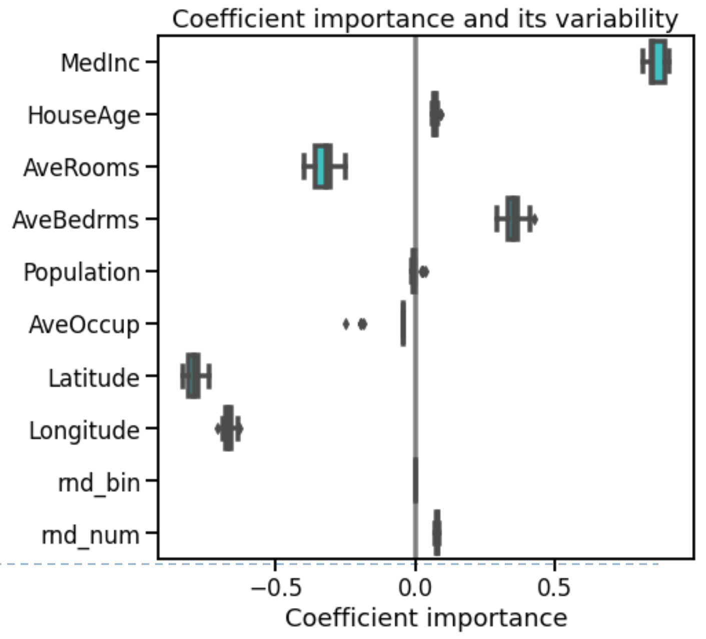
Short conclusions

- Tree-based models can give high importance to features that may not be predictive on unseen data when the model is overfitting. Permutation-based feature importance avoids this issue, since it can be computed on unseen data
  - Feature importance for trees are strongly biased and favor high cardinality features (typically numerical features) over low cardinality features such as binary features or categorical variables with a small number of possible categories
  - When two features are correlated and one of the features is permuted, the model will still have access to the feature through its correlated feature. This will result in a lower importance value for both features, where they might be important. One way to handle this is to cluster features that are correlated and only keep one feature from each cluster

Select from model shrinkage

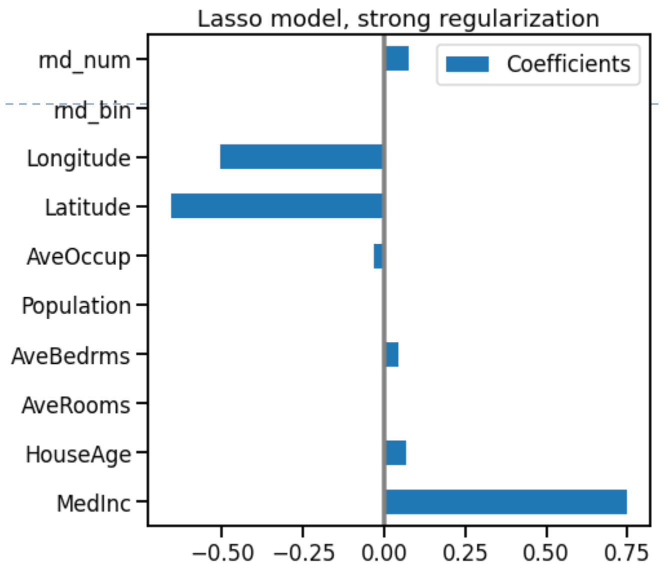
- In linear models, coefficients represent the relationship between the given feature X and the target y, assuming that all the other features remain constant
  - The coefficients of a linear model quantify the variation of a the output when the given feature is varied, keeping all other features constant
  - In practice, ridge regression are often used
    - Before any interpretation, we may need to scale each column
  - We can check the coefficient variability through cross-validation. If coefficients vary significantly when changing the input dataset their robustness is not guaranteed, and they should probably be interpreted with caution

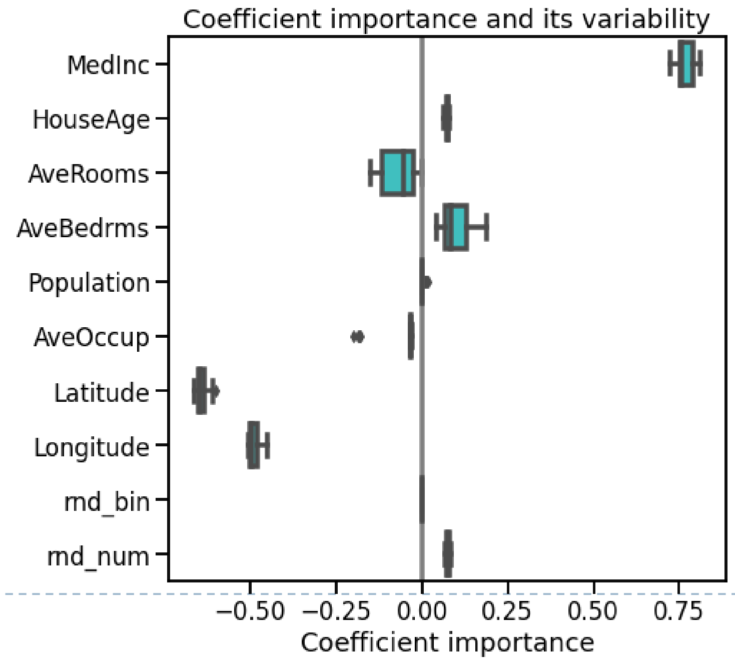




Select from model shrinkage

- Above methods do have one obvious disadvantage: unlike subset selection, which will generally select models that involve just a subset of the variables, they will include all p predictors in the final model
  - As with ridge regression, the lasso shrinks the coefficient estimates towards zero
  - However, in the case of the lasso, the l_1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large. Hence, much like best subset selection, the lasso performs variable selection
  - We say that the lasso yields sparse models - that is, models that involve only a subset of the variables



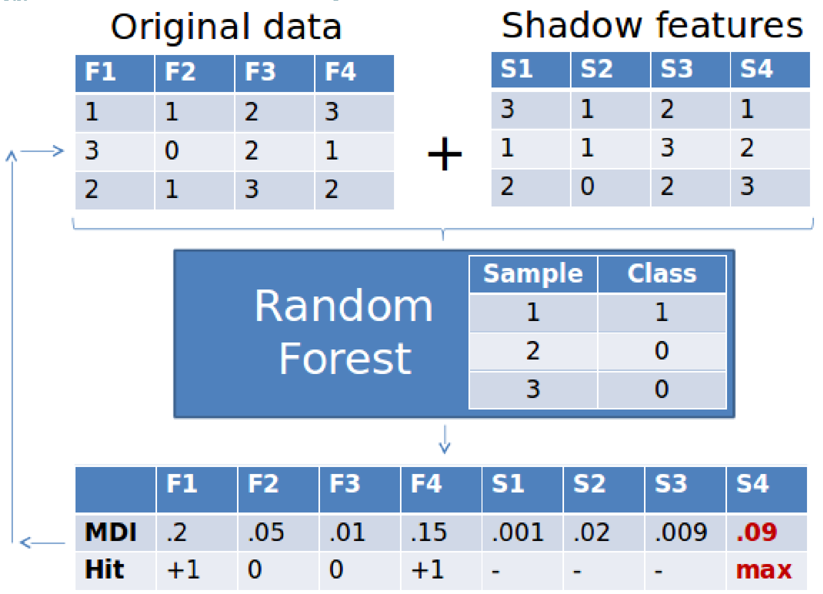


Boruta – using concept similar to permutation test

- Boruta tries to capture all the important, interesting features you might have in your dataset with respect to the target
  - This makes it suited for biomedical data analysis, where we regularly collect measurements of thousands of features and we have no clue where should we cut off the threshold
  - Boruta use the binomial test to examine the following hypothesis

H_0 : Feature importance of a feature is the same as the maximum of random features

H_a : Feature importance of a feature is larger than the maximum of random features



F1	F2	F3	F4
1	1	2	3
3	0	2	1
2	1	3	2

S1	S2	S3	S4
3	1	2	1
1	1	3	2
2	0	2	3

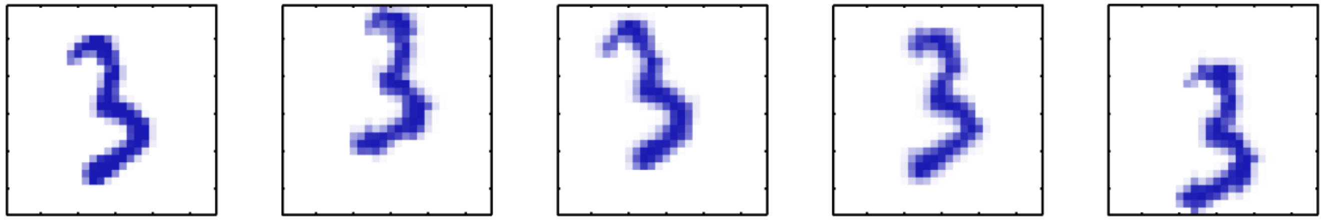
Sample	Class
1	1
2	0
3	0

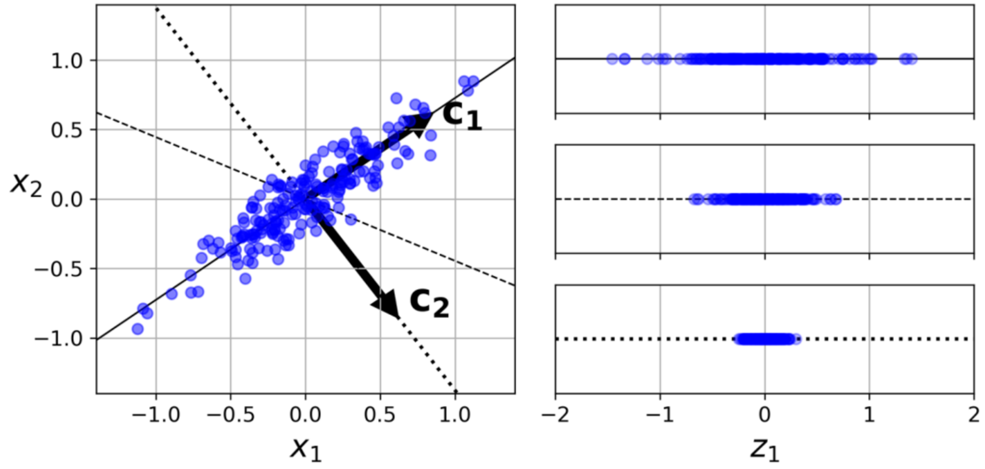
	F1	F2	F3	F4	S1	S2	S3	S4
MDI	.2	.05	.01	.15	.001	.02	.009	.09
Hit	+1	0	0	+1	-	-	-	max

	F1	F2	F3	F4	S1	S2	S3	S4
MDI	.2	.05	.01	.15	.001	.02	.009	.09
Hit	+1	0	0	+1	-	-	-	max
Hit	+1	+1	0	0	-	max	-	-
Hit	+1	0	0	+1	-	-	-	max

3. Dimensional reduction by linear projection

- In most real-world problems, training instances are not spread out uniformly across all dimensions. Many features are almost constant, while others are highly correlated As a result, all training instances actually lie within much lower-dimensional *subspace* of the high-dimensional space
  - Dimensionality reduction tries to preserve various measure or structure in high dimensional space like distance, topology, density etc



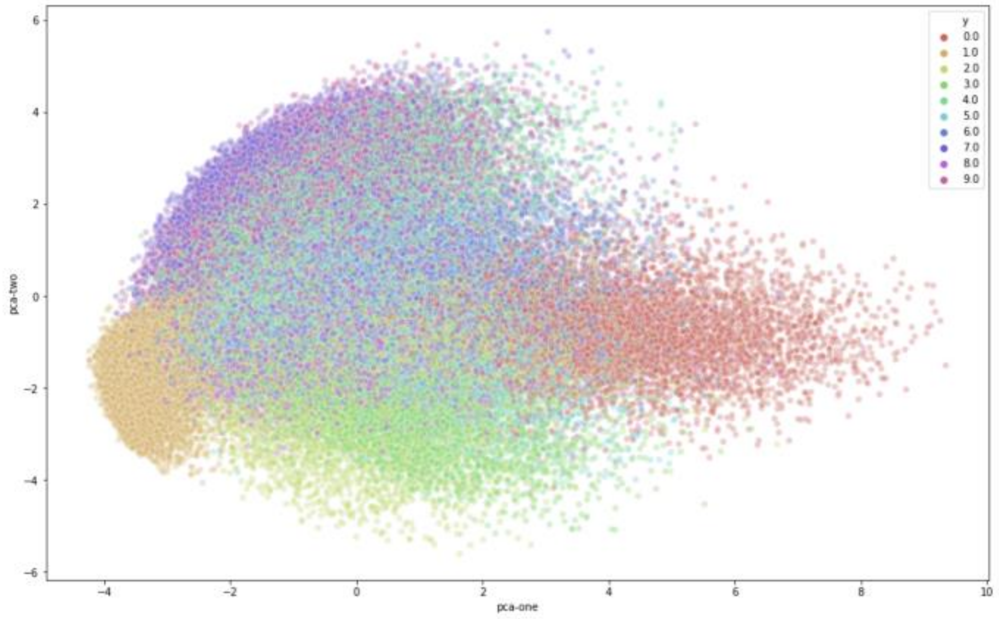


Principal Components Analysis (PCA)

- Suppose that we wish to visualize n observations with measurements on a set of p features, $X_1, X_2, \ldots, X_p$, as part of an exploratory data analysis
  - We could do this by examining two-dimensional scatterplots of the data, each of which contains the n observations' measurements on two of the features. However, there are $\binom{p}{2}$ such scatterplots
  - Most likely none of them will be informative since they each contain just a small fraction of the total information present in the data set
- Clearly, a better method is required to visualize the n observations when p is large. We would like to find a low-dimensional representation of the data that captures as much of the information as possible

Principal Components Analysis (PCA)

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance
  - The information is measured by the amount that the observations vary along each dimension
  - Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization



https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b

Interpretation of principal components

- The first principal component loading vector solves the optimization problem

$$\max_{\Phi_{11},\ldots,\Phi_{p1}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\Phi_{j1}x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}\Phi_{j1}^2 = 1$$

  - This problem can be solved via a singular-value decomposition of the data matrix *X* or the eigenvalue decomposition of the covariance matrix *X*
- The loading vector Φ_1 with elements $\Phi_{11}\Phi_{21}\ldots\Phi_{\mathrm{p}1}$ denfies a direction in feature space along which the data vary the most
- If we project the n data points $x_1, x_2, \ldots, x_n$ onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves
- The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance among all linear combinations that are uncorrelated with Z_1

Another interpretation of principal components

- The first M principal component score vectors and the first M principal component loading vectors provide the best M-dimensional approximation (in terms of Euclidean distance) to the ith observation x_{ij}

$$x_{ij} \approx \sum_{m=1}^{M} z_{im}\Phi_{jm}$$

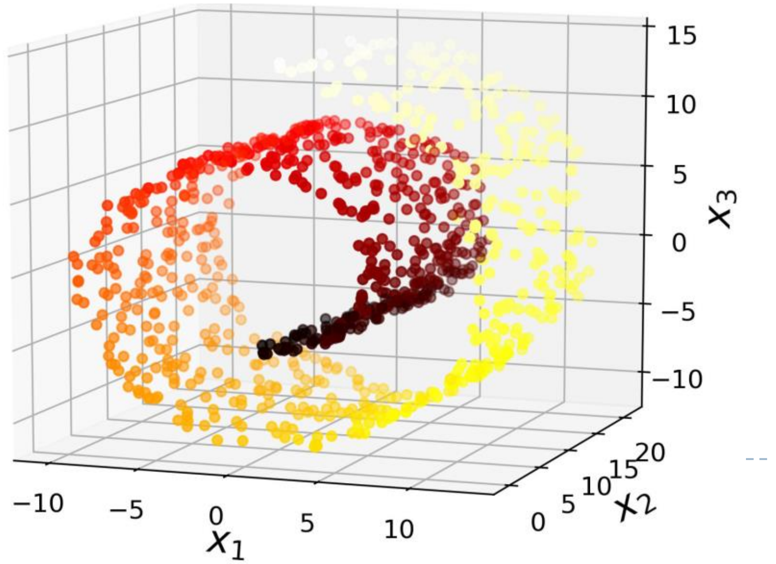
- Suppose the data matrix X is column-centered. Out of all approximations of the form $x_{ij} \approx \sum_{m=1}^{M} a_{im}b_{jm}$. We have

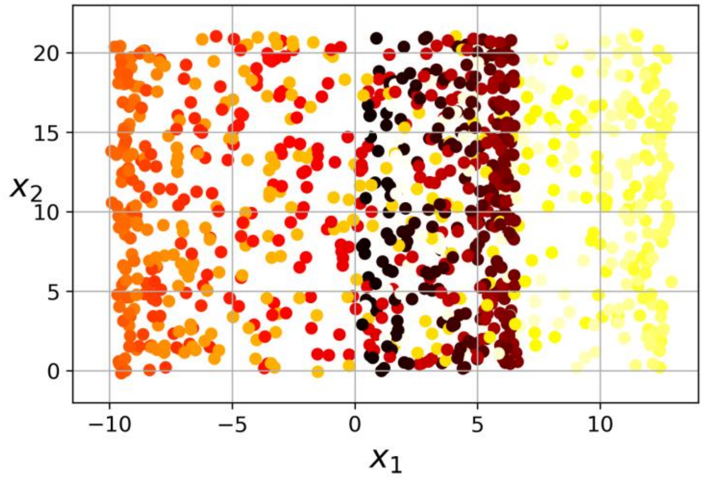
$$\min_{A \in R^{n\times M},\, B \in R^{p\times M}} \{\sum_{j=1}^{p}\sum_{i=1}^{n}(x_{ij} - \sum_{m=1}^{M} a_{im}b_{jm})^2\}$$

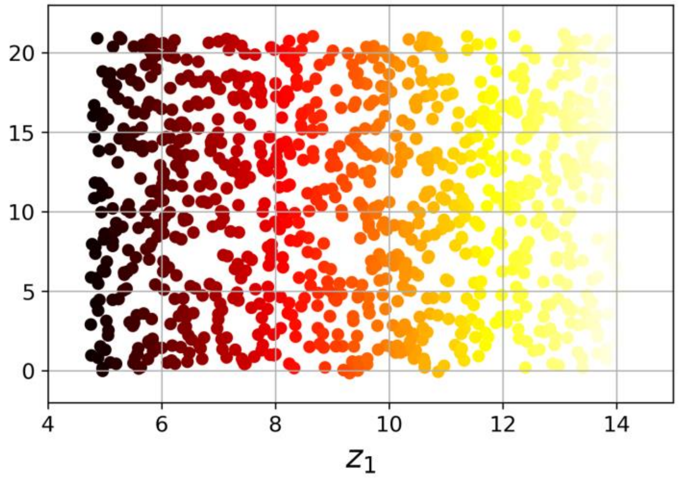
- It can be shown that for any value of M, the columns of the matrices $\hat{A}$ and $\hat{B}$ are in fact the first M principal components score and loading vectors

Manifold learning

- A M-dimensional manifold is a part of an p-dimensional space (where $M < p$) that *locally resembles* a M-dimensional hyperplane
  - In the case of the Swiss roll, $M = 2$ and $p = 3$: it locally resembles a 2D plane, but it is bent and twisted in the third dimension
  - Simply projecting onto a plane would squash different layers of the Swiss roll
- Many dimensionality reduction algorithms work by modeling the manifold on which the training instances lie; this is called manifold learning







t-SNE (t-distributed Stochastic Neighbor Embedding)

- t-SNE is popular manifold learning method that is specialized for visualization and EDA
  - It is a statistical method for visualizing high-dimensional data by giving each data point a location in a two or three-dimensional map
  - It converts affinities of data points to probabilities and tends to preserve topology that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points
  - It is particularly sensitive to local structure and can revealing the structure at many scales or data lie on several manifolds on a single map
  - However, the global structure is not explicitly preserved and we should use it with caution for downstream analysis

t-Stochastic Neighbor Embedding

Caveats

- t-SNE does not work well for general dimensionality problem where the embedded dimension is greater than 2D or 3D
  - O(Mn^2) computational complexity. But can be reduced to $O(Mn\log n)$ using Barnes-Hut t-SNE which only works for the target dimensionality is smaller than 3 and dense dataset. The flt-tsne or UMAP can be use to give linear scalability
- We can also add new data point into the embedding by parametric t-SNE
- There are more and more theoretical guarantee for t-SNE. See here and here
  - t-SNE is generally very good at capture local structure and best tool for visualization
  - Noise may appear to have some structure and cluster may not discover by t-SNE
  - Generally, we do not use t-SNE for clustering purpose
- UMAP on the other hand can be used as a preprocessing methods for clustering or other machine learning task

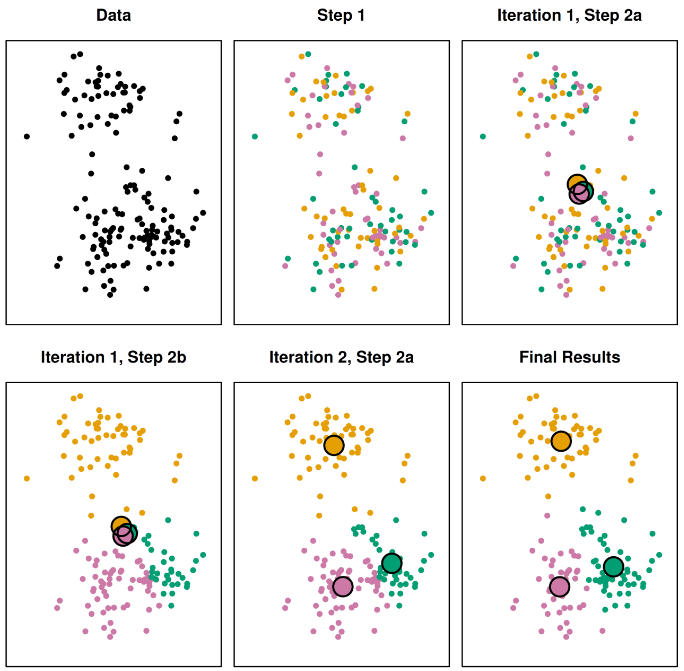
Cluster analysis

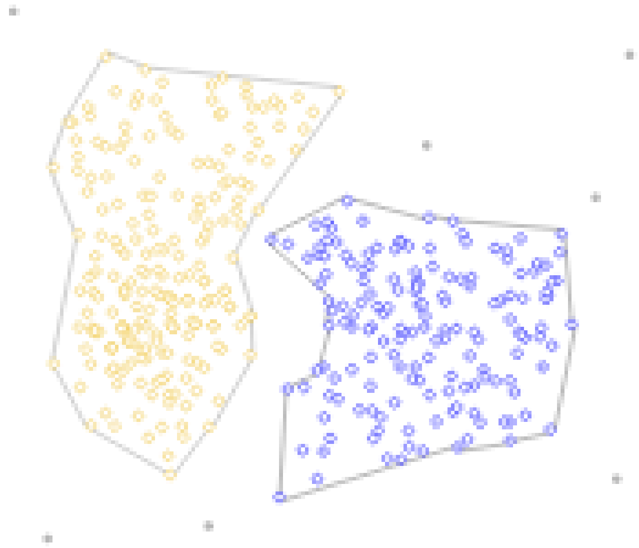
- Clustering refers to a very broad set of techniques for finding subgroups, or clusters, in a data set. We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other

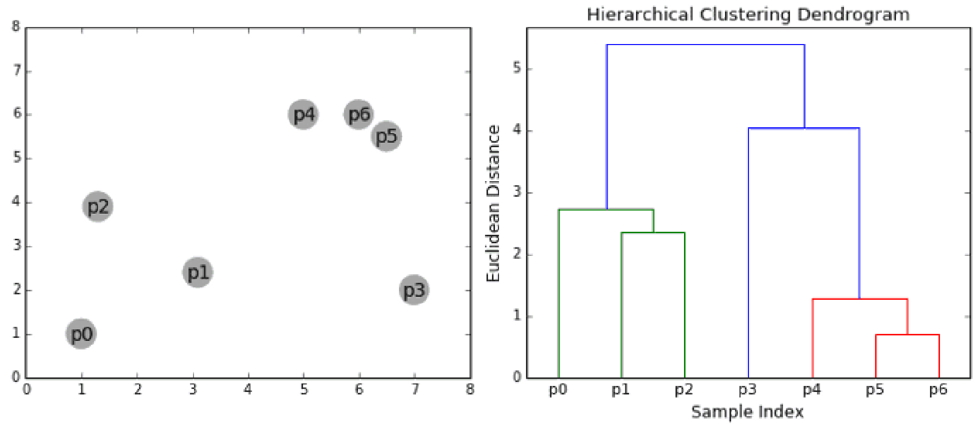
1. Once a dataset has been clustered, it is possible to measure each instance's *affinity* with each cluster which reduce to *k* dimensional space (*representation learning*)
2. We can cluster observations on the basis of the features in order to identify subgroups among the observations, or we can cluster features on the basis of the observations in order to discover subgroups among the features (*feature clustering*)
3. For semi-supervised learning: if you only have a few labels, you could perform clustering and propagate the labels to all the instances in the same cluster
4. For *anomaly detection*: any instance that has a low affinity to all the clusters is likely to be an anomaly

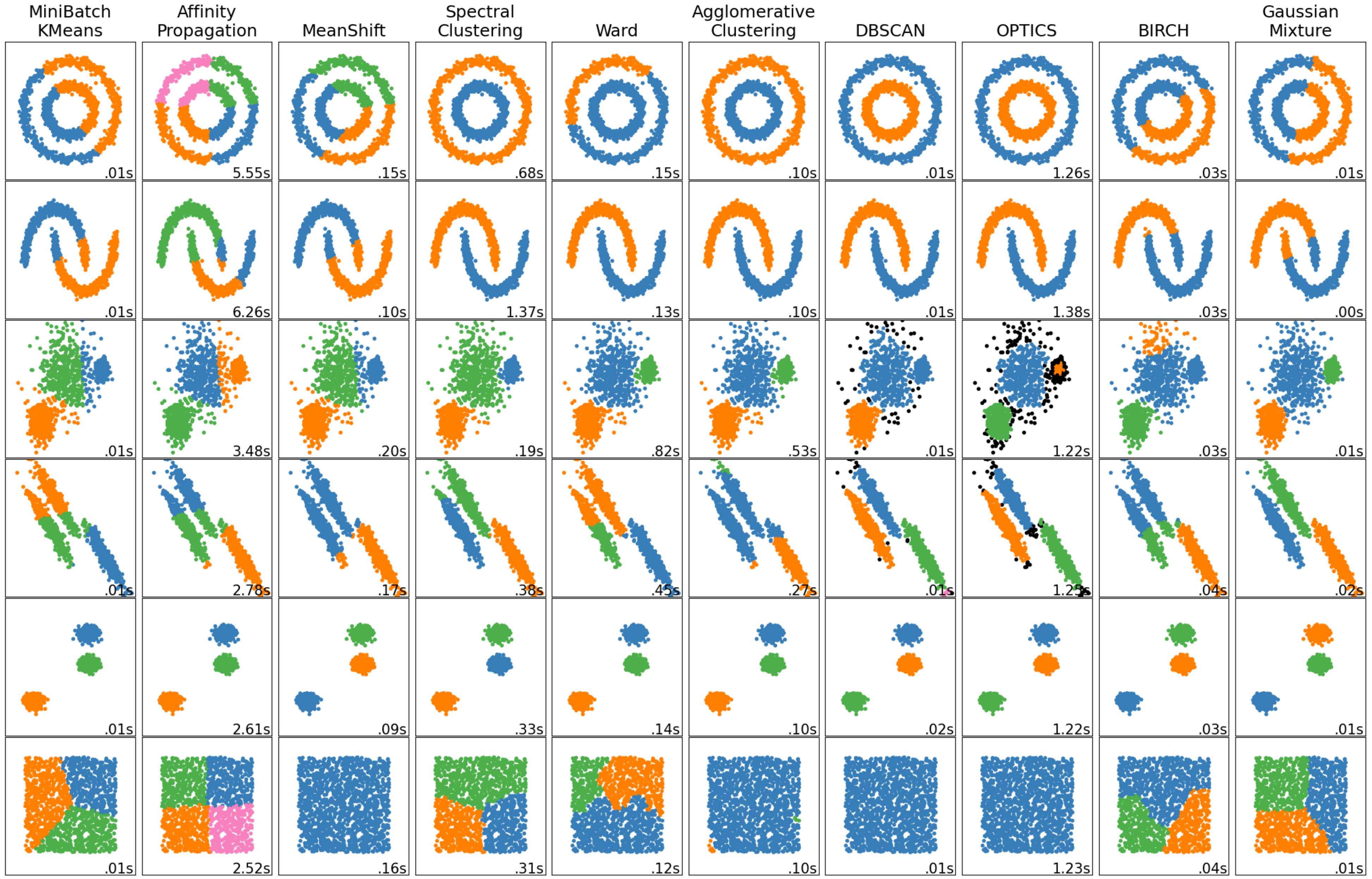
Taxonomy of clustering algorithms

- When choosing a clustering algorithm, you should consider whether the algorithm scales to your dataset. Datasets in machine learning can have millions of examples, but not all clustering algorithms scale efficiently
  - Many clustering algorithms work by computing the similarity between all pairs of examples





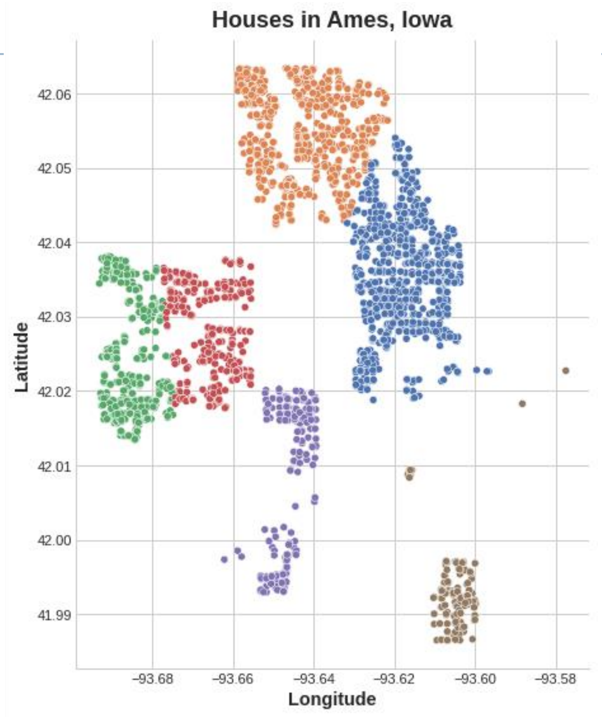




https://scikit-learn.org/stable/modules/clustering.html

Clustering as a preprocessing method

- Clustering can be an efficient approach to dimensionality reduction, in particular as a preprocessing step before a supervised learning algorithm
  - Cluster labels as a feature
    - The motivating idea for adding cluster labels is that the clusters will break up complicated relationships across features into simpler chunks. Our model can then just learn the simpler chunks one-by-one instead having to learn the complicated whole all at once
    - Distance to cluster centers can also be good features
    - Feature agglomeration by pooling function



Longitude	Latitude	Cluster	D_C1	...	D_C3
-93.619	42.054	3	1.25		5.24
-93.619	42.053	3	3.56		3.33
-93.638	42.060	1	5.21		0.89
-93.602	41.988	0	7.83		4.31

Clustering for semi-supervised learning

- Another use case for clustering is in semi-supervised learning, when we have plenty of unlabeled instances and very few labeled instances
  - Just like active learning, we can first cluster all the data points and ask for the label of the images that are most close to the cluster center
    - Using this images to train classifier may be much better than random instances
  - We can also propagated the labels to all the other instances in the same cluster. This is called label propagation

Short summary

- For unsupervised algorithm. To make the *affinity* concrete, we must define what it means for two or more observations to be similar or different
  - Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied
  - Supervised metrics is also possible, you can refer to similarity learning
  - Metric learning may also be helpful in this case
- Both clustering and PCA seek to simplify the data via a small number of summaries, but their mechanisms are different:
  - PCA looks for a low-dimensional representation of the observations that explains a good fraction of the variance
  - Clustering looks for homogeneous subgroups among the observations

Conclusions

- The principal advantage of selecting features within a data science pipeline is to reduce the time to train this pipeline and its time to predict, simplification of models to make them easier to interpret and to avoid the curse of dimensionality
  - Univariate and subset Selection
  - Select from model using importance or using shrinkage
  - Dimension Reduction/clustering
- Unsupervised learning is important for understanding the variation and grouping structure of a set of unlabeled data, and can be a useful pre-processor for supervised learning
  - It is intrinsically more difficult than supervised learning because there is no gold standard (like an outcome variable) and no single objective (like test set accuracy)

References

[1] Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition
Chapter 8,9

[2] https://inria.github.io/scikit-learn-mooc/python_scripts/dev_features_importance.html

[3] https://www.kaggle.com/learn/feature-engineering

[4] An Introduction to Statistical Learning, second edition

Appendix

Resources

- Feature selection
  - https://scikit-learn.org/stable/modules/feature_selection.html#
  - https://scikit-learn.org/stable/modules/permutation_importance.html
  - https://scikit-learn.org/stable/modules/unsupervised_reduction.html
- Boruta
  - https://danielhomola.com/feature%20selection/phd/borutapy-an-all-relevant-feature-selection-method/
- Clustering
  - https://developers.google.com/machine-learning/clustering
  - https://github.com/microsoft/ML-For-Beginners/blob/main/5-Clustering/1-Visualize/README.md#introduction-to-clustering

Resources

- Manifold learning
  - https://scikit-learn.org/stable/modules/manifold.html
- Spectral clustering
  - https://jlmelville.github.io/smallvis/spectral.html
- Mixture models
  - https://cs229.stanford.edu/syllabus.html
  - https://cs229.stanford.edu/notes2021fall/cs229-notes7b.pdf
  - Variational Bayesian Gaussian Mixture

Backward stepwise selection: details

Backward Stepwise Selection

1. Let M_p denote the full model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   a) Consider all k models that contain all but one of the predictors in M_k , for a total of $k - 1$ predictors
   b) Choose the best among these k models, and call it M_{k-1}. Here best is defined as having the smallest RSS
3. Select a single best model from among $M_0, \ldots, M_p$ using cross-validated score, C_p (AIC), BIC, or adjusted R^2

More on backward stepwise selection

- Like forward stepwise selection, the backward selection approach searches through only $1 + p(p+1)/2$ models, and so can be applied in settings where p is too large to apply best subset selection
- Like forward stepwise selection, backward stepwise selection is not guaranteed to yield the best model containing a subset of the p predictors
- Backward selection requires that the number of samples n is larger than the number of variables p (so that the full model can be fit). In contrast, forward stepwise can be used even when $n < p$, and so is the only viable subset method when p is very large

Collinearity

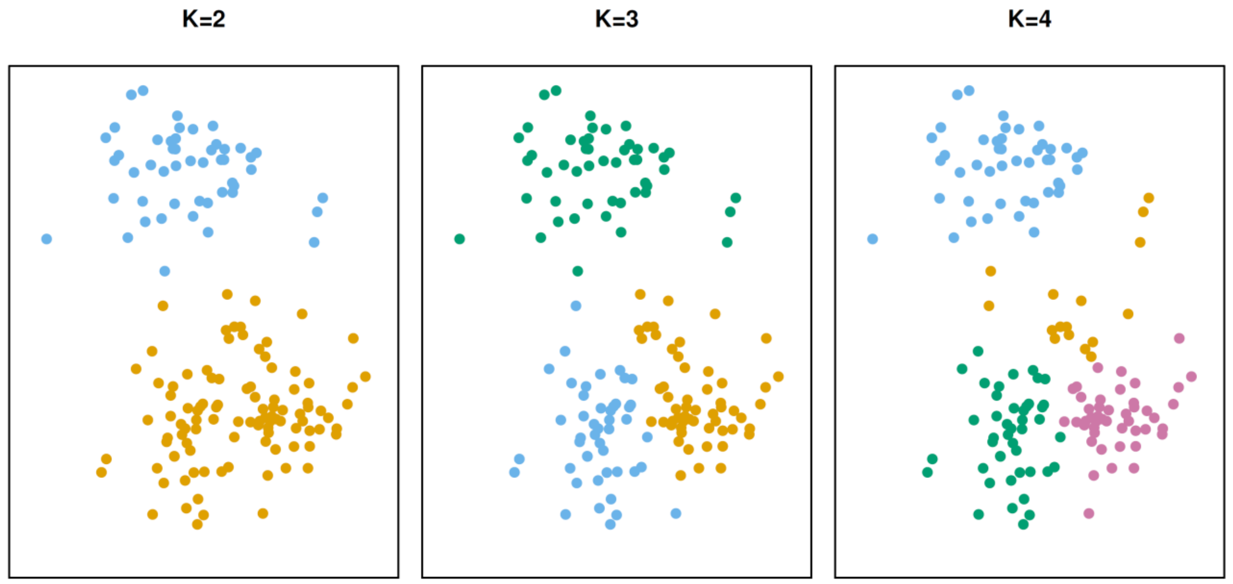
- Instead of inspecting the correlation matrix, a better way to assess multicollinearity collinearity is to compute the variance inflation factor (VIF)
  - The VIF for each variable can be computed using the formula

$$VIF\left(\hat{\beta}_j\right) = \frac{1}{1 - R^2_{X_j|X_{-j}}}$$

where $R^2_{X_j|X_{-j}}$ is the R^2 from a regression of X_j onto all of the other predictors

- When faced with the problem of collinearity
  1. The first is to drop one of the problematic variables from the regression. This can usually be done without much compromise to the regression fit
  2. The second solution is to combine the collinear variables together into a single predictor

Details of K-means clustering



- A simulated data set with 150 observations in 2-dimensional space. The color of each observation indicates the cluster to which it was assigned using the K-means clustering algorithm
- The cluster labels were not used in clustering; instead, they are the outputs of the clustering procedure

Details of K-means clustering

- Let $C_1, \dots, C_K$ denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:

1. $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \dots, n\}$. In other words, each observation belongs to at least one of the K clusters
2. $C_k \cap C_{k'} = 0$ for all $k \neq k'$. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster (In contrast to fuzzy clustering)

- For instance, if the ith observation is in the kth cluster, then $i \in C_k$

Details of K-means clustering: continued

- The idea behind K-means clustering is that a good clustering is one for which the within-cluster variation is as small as possible
- The within-cluster variation for cluster C_k is a measure $W(C_k)$ of the amount by which the observations within a cluster differ from each other
- Hence we want to solve the problem

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{k=1}^{K} \mathrm{W}(C_k) \right\}$$

- In words, this formula says that we want to partition the observations into K clusters such that the total within-cluster variation, summed over all K clusters, is as small as possible

How to define within-cluster variation?

- Typically we use Euclidean distance

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2,$$

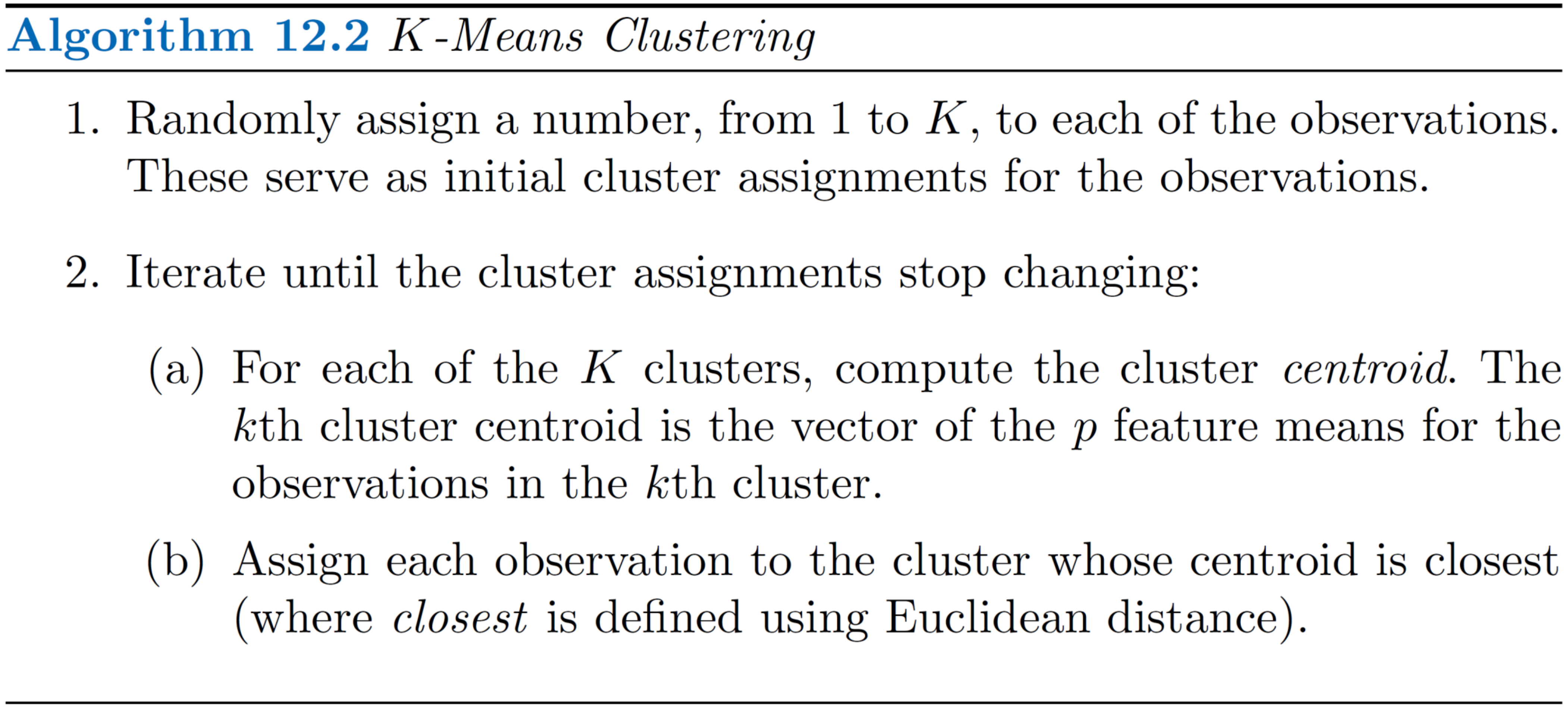
Where $|C_k|$ denotes the number of observations in the kth cluster

- Combining previous two equation gives the optimization problem which minimize the following objective function that defines K-means clustering,

$$\min_{C_1,\ldots,C_K} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$$

- There are K^n ways to partition n observation into K clusters

Details of K-means clustering



Algorithm 12.2 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

Properties of the Algorithm

- This algorithm is guaranteed to decrease the value of the objective function at each step. Note that

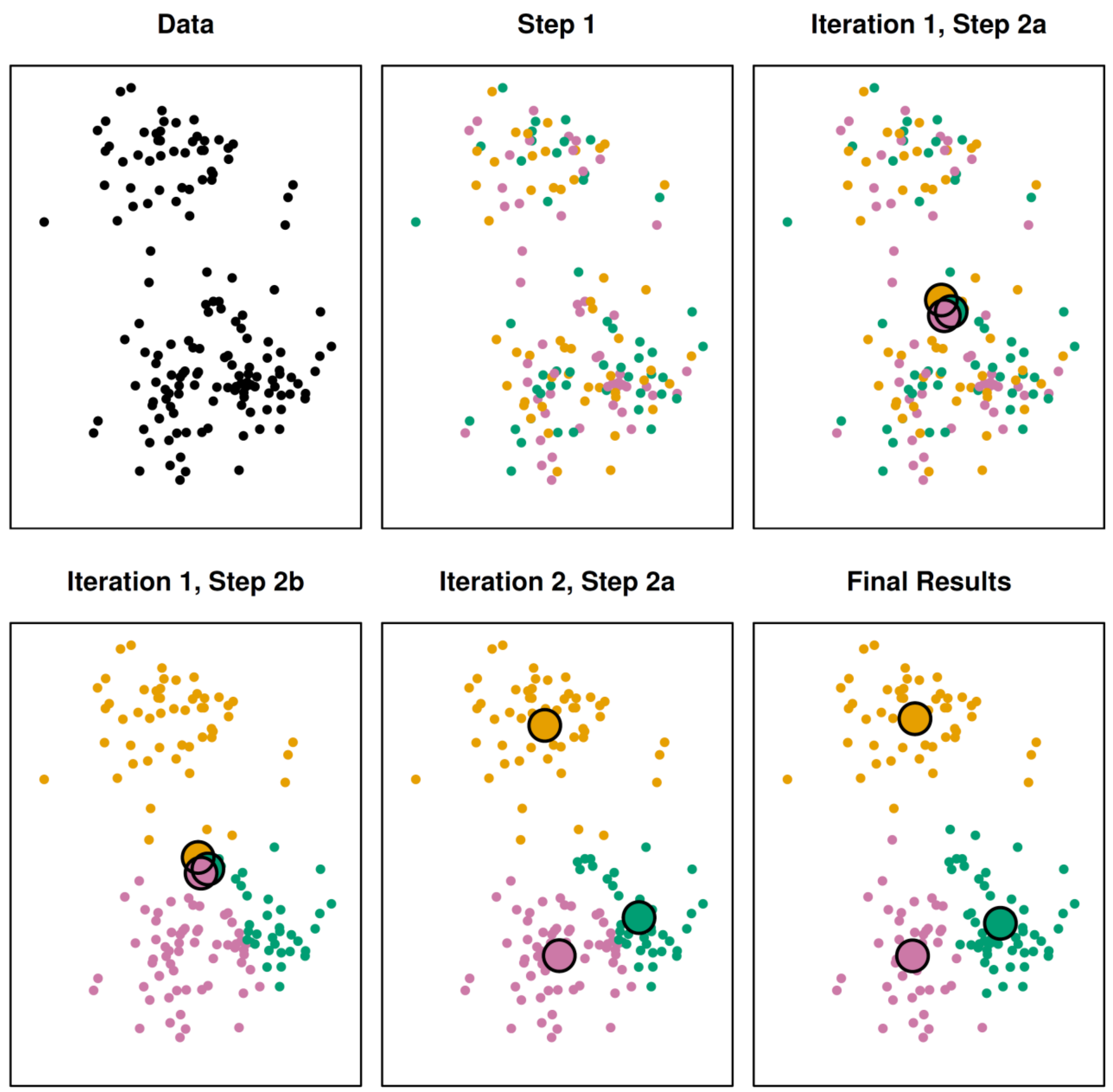
$$\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2 = 2\sum_{i\in C_k}\sum_{j=1}^{p}(x_{ij}-\bar{x}_{kj})^2$$

Where $\bar{x}_{kj} = \frac{1}{|C_k|}\sum_{i\in C_k} x_{ij}$ is the mean for feature j in cluster C_k and

$$\frac{1}{n}\sum_{i,j=1}^{n}(x_i - x_j)^2 = \frac{1}{n}\sum_{i,j=1}^{n}[(x_i-\bar{x})-(x_j-\bar{x})]^2 = \frac{2}{n}\sum_{i}^{n}(x_i-\bar{x})^2$$

- However it is not guaranteed to give the global minimum
- It is close related to Gaussian mixture model

Example

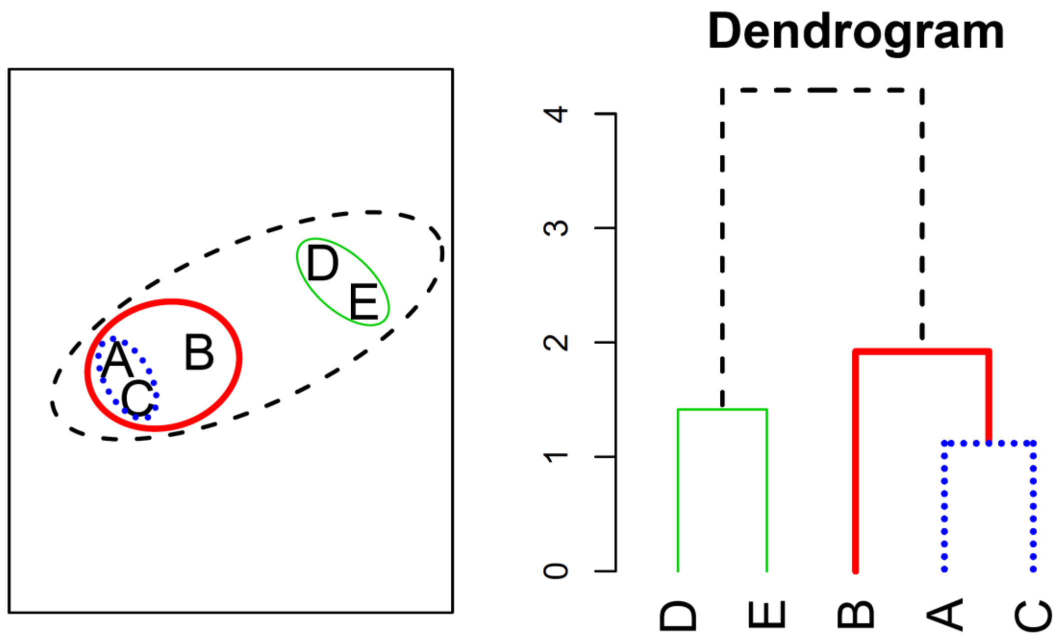

Data
Step 1
Iteration 1, Step 2a
Iteration 1, Step 2b
Iteration 2, Step 2a
Final Results

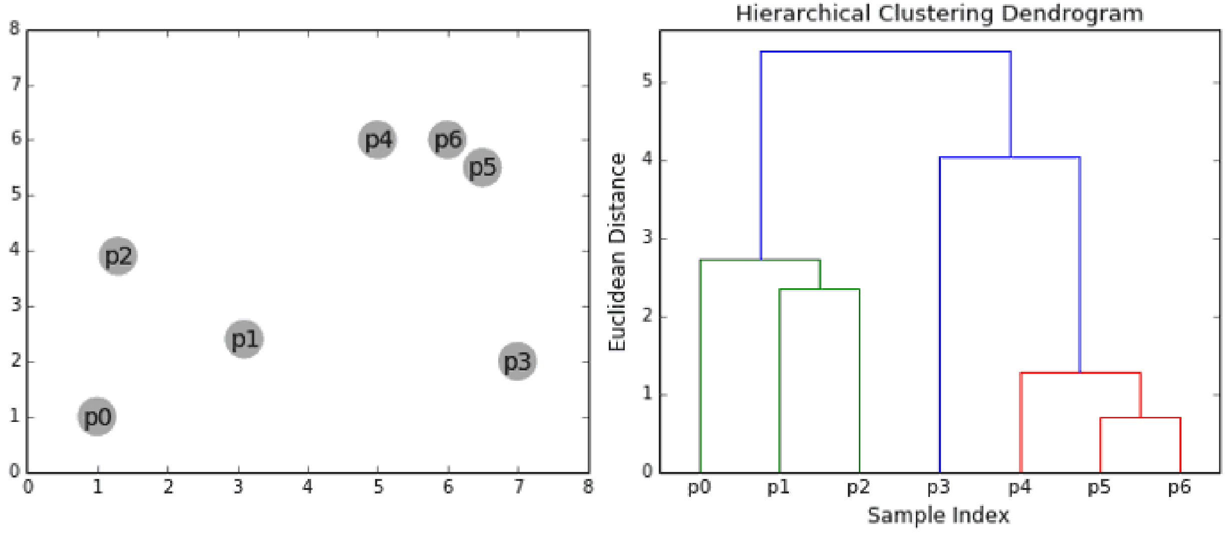
Hierarchical Clustering

- K-means clustering requires us to pre-specify the number of clusters K. This can be a disadvantage
- Hierarchical clustering is an alternative approach which does not require that we commit to a particular choice of K
- In this section, we describe bottom-up or agglomerative clustering (cf. divisive clustering). This is the most common type of hierarchical clustering, and refers to the fact that a dendrogram is built starting from the leaves and combining clusters up to the trunk

Agglomerative (Hierarchical or bottom up) Clustering

- Start with each point in it’s own cluster and then, for each cluster, use some criterion to choose another cluster to merge with. Do this repeatedly until you have only one cluster and you get a hierarchy, or binary tree, of clusters branching down to the last layer which has a leaf for each point in the dataset



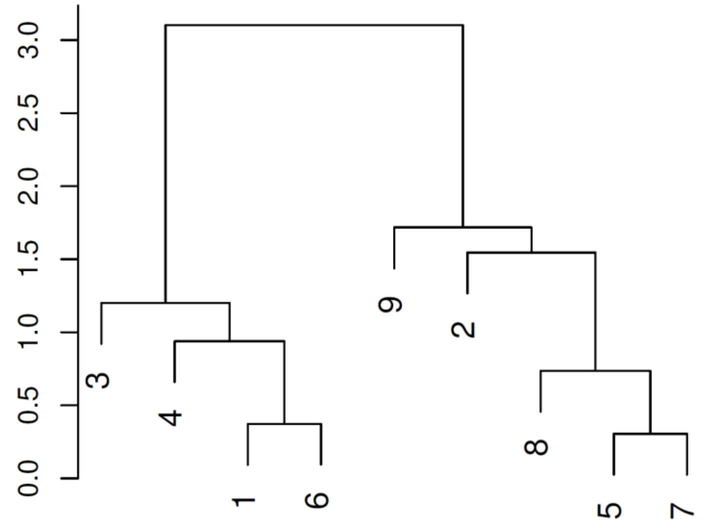


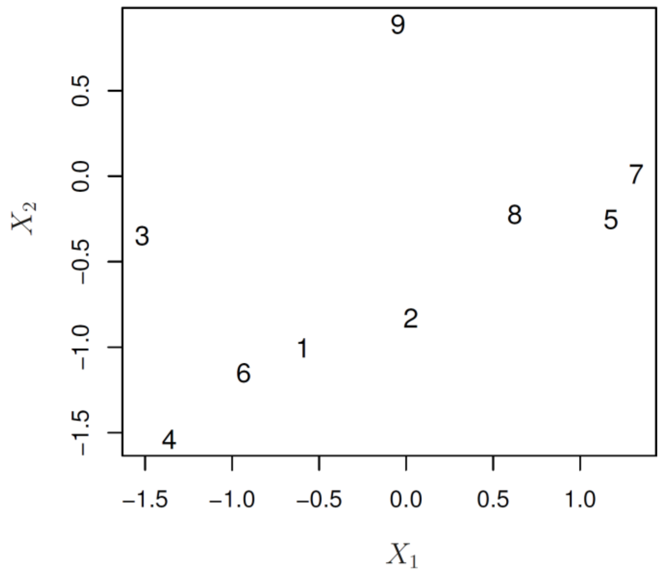
Assumption behind Hierarchical Clustering

- The term hierarchical refers to the fact that clusters obtained by cutting the dendrogram at a given height are necessarily nested within the clusters obtained by cutting the dendrogram at any greater height
- Suppose that our observations correspond to a group of men and women, evenly split among Americans, Japanese, and French. We can imagine a scenario in which the best division into two groups might split these people by gender, and the best division into three groups might split them by nationality
- In this case, the true clusters are not nested, in the sense that the best division into three groups does not result from taking the best division into two groups and splitting up one of those groups

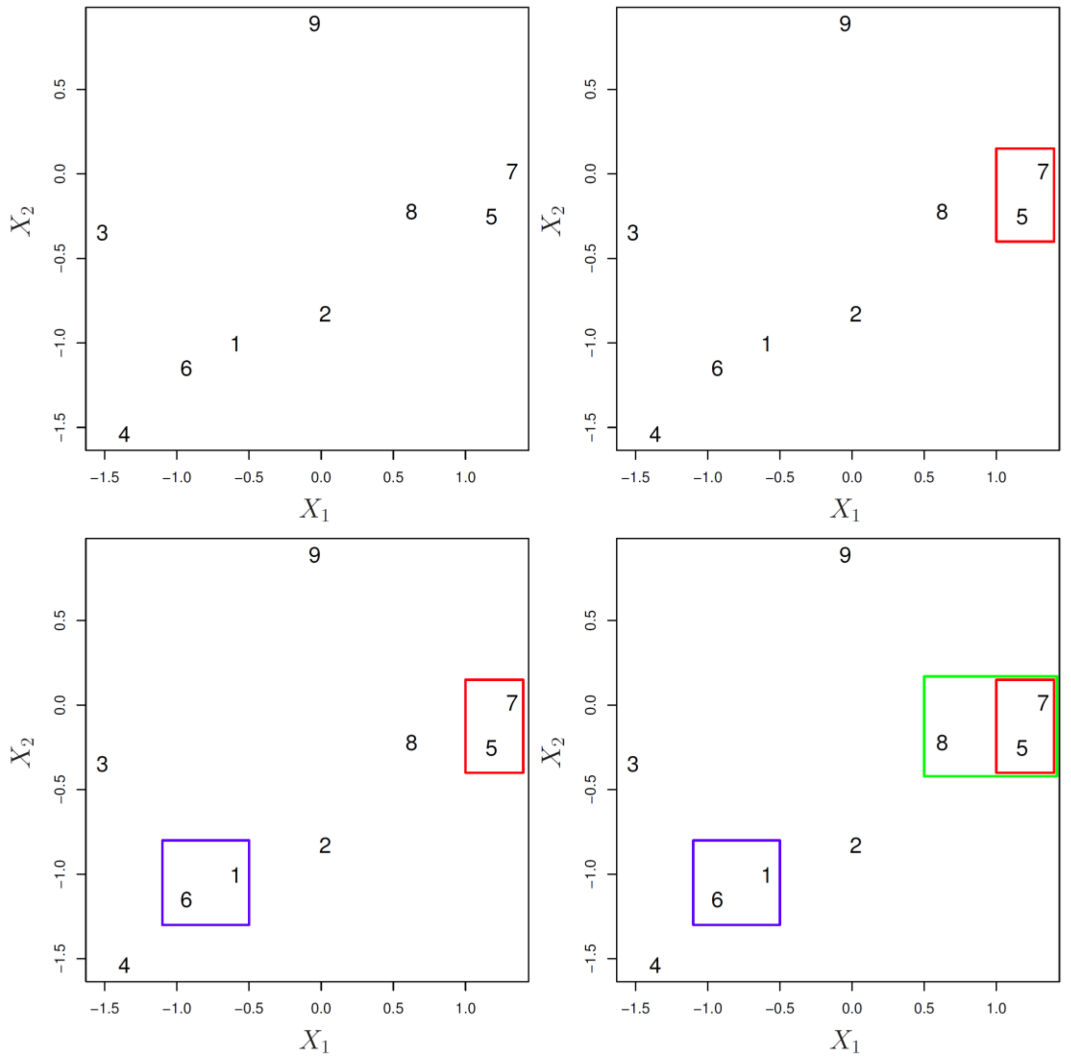
Another Example

- An illustration of how to properly interpret a dendrogram with nine observations in two-dimensional space. The raw data on the right was used to generate the dendrogram on the left
- Observations 5 and 7 are quite similar to each other, as are observations 1 and 6
- However, observation 9 is no more similar to observation 2 than it is to observations 8, 5, and 7, even though observations 9 and 2 are close together in terms of horizontal distance
- This is because observations 2, 8, 5; and 7 all fuse with observation 9 at the same height, approximately 1.8



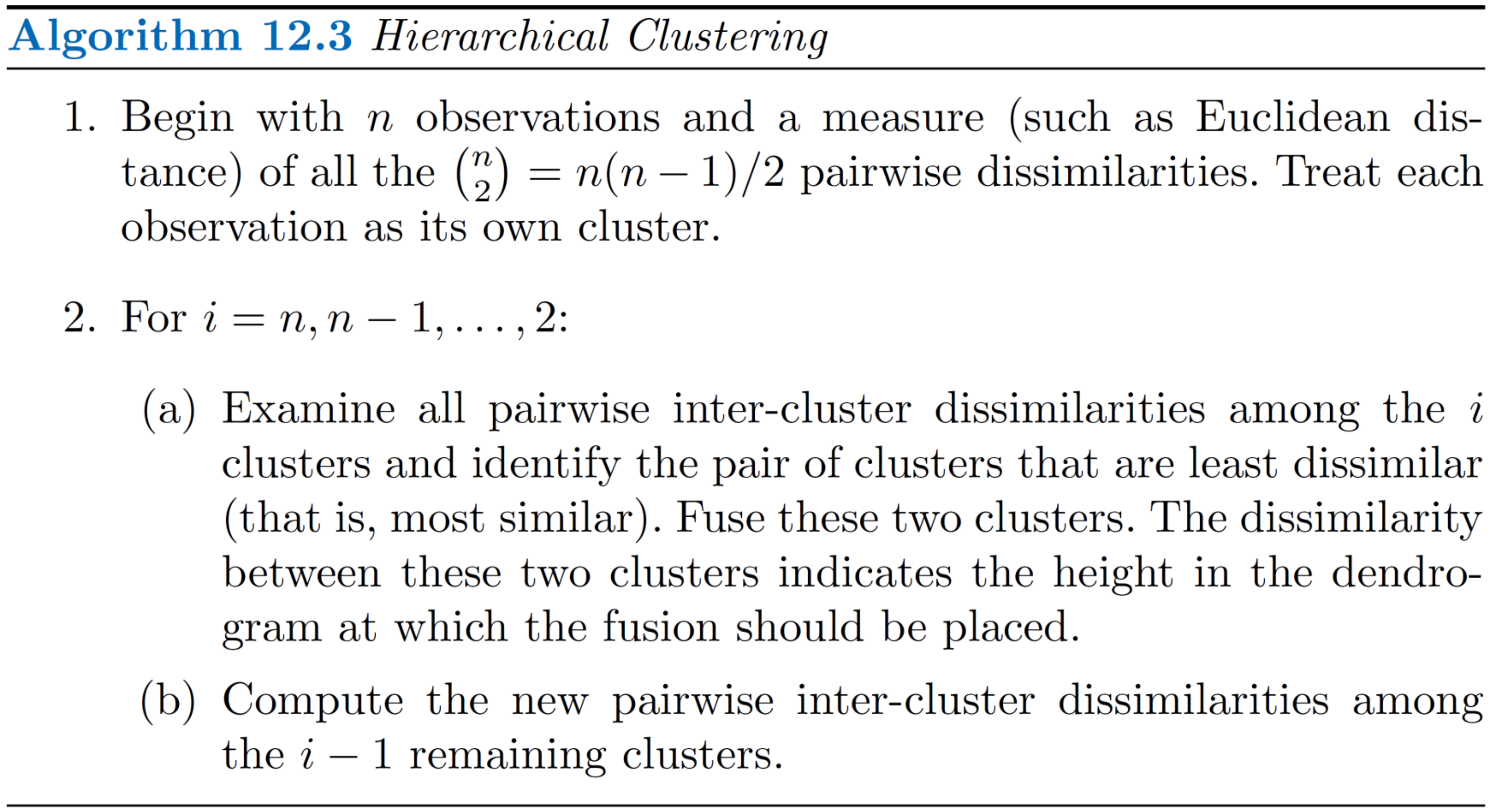


Merges in previous example



Hierarchical Clustering

- How did we determine that the cluster {5, 7} should be fused with the cluster {8}?



Algorithm 12.3 *Hierarchical Clustering*

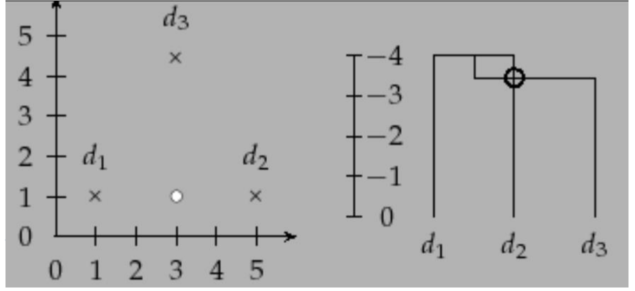
1. Begin with n observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

2. For $i = n, n-1, \ldots, 2$:

   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

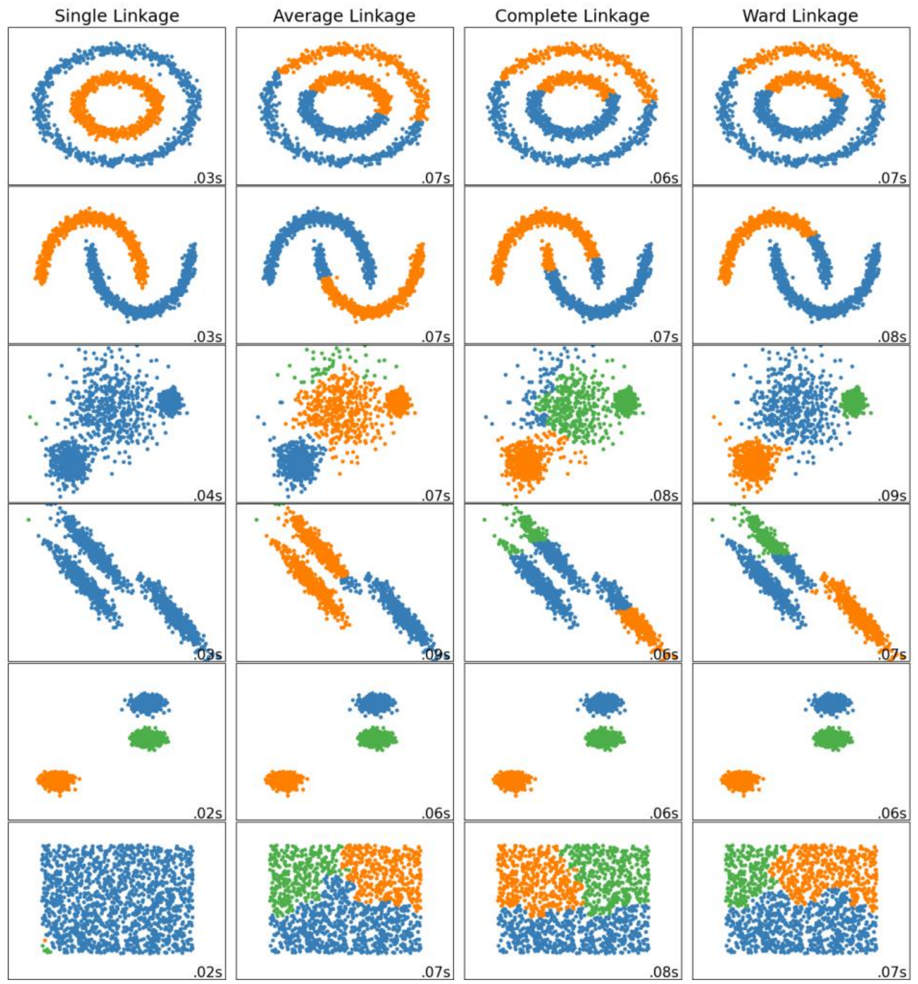
Types of Linkage



Linkage	Description
Complete	Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities
Single	Minimal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities
Average	Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities
Centroid	Dissimilarity between the centroid for cluster A (a mean vector of length p) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*
Ward	Minimizes the sum of squared differences within all clusters. It is a variance-minimizing approach and in this sense is similar to the k-means objective function

Types of Linkage

- Agglomerative cluster has a "*rich get richer*" behavior that leads to uneven cluster sizes
  - Single linkage seems like the worst strategy, and Ward gives the most regular sizes
- However, the affinity cannot be varied with Ward, thus for non Euclidean metrics, average linkage is a good alternative
- Single linkage, while not robust to noisy data, can be computed very efficiently. It can also perform well on non-globular data

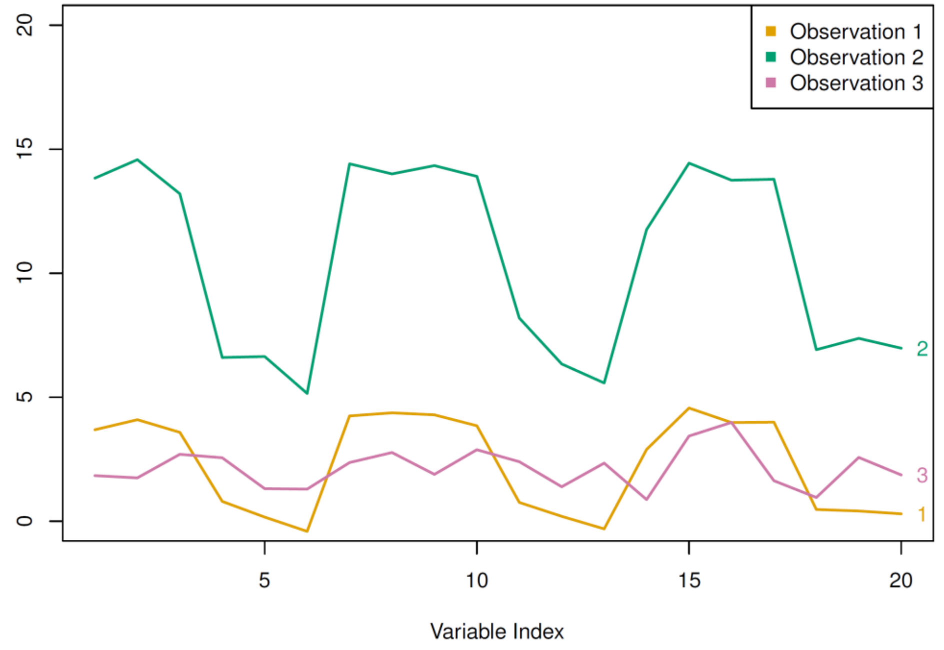


Good EDA clustering algorithm

1. **Don’t be wrong**: If you are doing EDA you are trying to learn and gain intuitions about your data. In that case it is far better to get no result at all than a result that is wrong. This means a good EDA clustering algorithm needs to conservative in it’s clustering
2. **Intuitive Parameters**: If you know little about your data it can be hard to determine what value or setting a parameter should have. This means parameters need to be intuitive enough that you can hopefully set them without having to know a lot about your data
3. **Stable Clusters**: If you run the algorithm twice with a different random initialization, you should expect to get roughly the same clusters back. If you are sampling your data, taking a different random sample shouldn’t change much. If you vary the clustering algorithm parameters you want the clustering to change in a somewhat stable predictable fashion
4. **Performance**: You can sub-sample, but ultimately you need a clustering algorithm that can scale to large data sizes. A clustering algorithm isn’t much use if you can only use it if you take such a small sub-sample that it is no longer representative of the data at large!

Choice of dissimilarity measure

- So far have used Euclidean distance
  - Supervised metrics is also possible
- An alternative is correlation-based distance which considers two observations to be similar if their features are highly correlated
  - Correlation-based distance focuses on the shapes of observation profiles rather than their magnitudes
  - More metrics



Choice of dissimilarity measure

- Consider an online retailer interested in clustering shoppers based on their past shopping histories
  - The data takes the form of a matrix where the rows are the shoppers and the columns are the items available for purchase; the elements of the data matrix indicate the number of times a given shopper has purchased a given item
  - If Euclidean distance is used, then shoppers who have bought very few items overall will be clustered together. This may not be desirable. On the other hand, if correlation-based distance is used, then shoppers with similar preferences will be clustered together