Data Science and Visualization (DSV, F23)

8. Clustering (II)

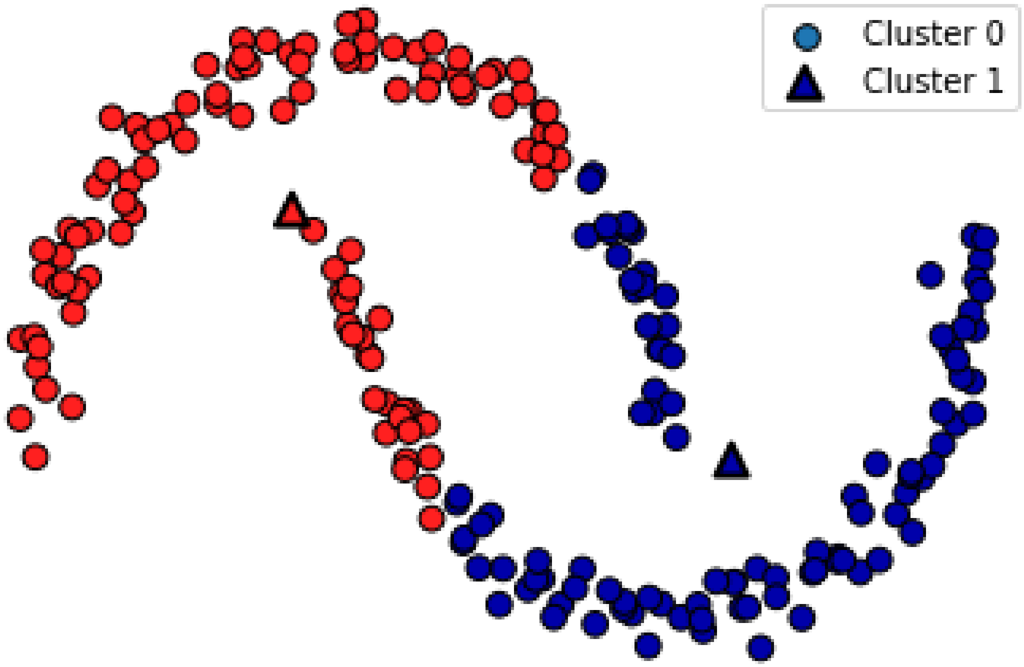
Hua Lu
https://luhua.ruc.dk; luhua@ruc.dk
PLIS, IMT, RUC

Agenda

- DBSCAN
- Evaluation of clustering
- Feature engineering

Failure of k-Means

- We've seen this example
- How can we obtain the right clustering for such a case?

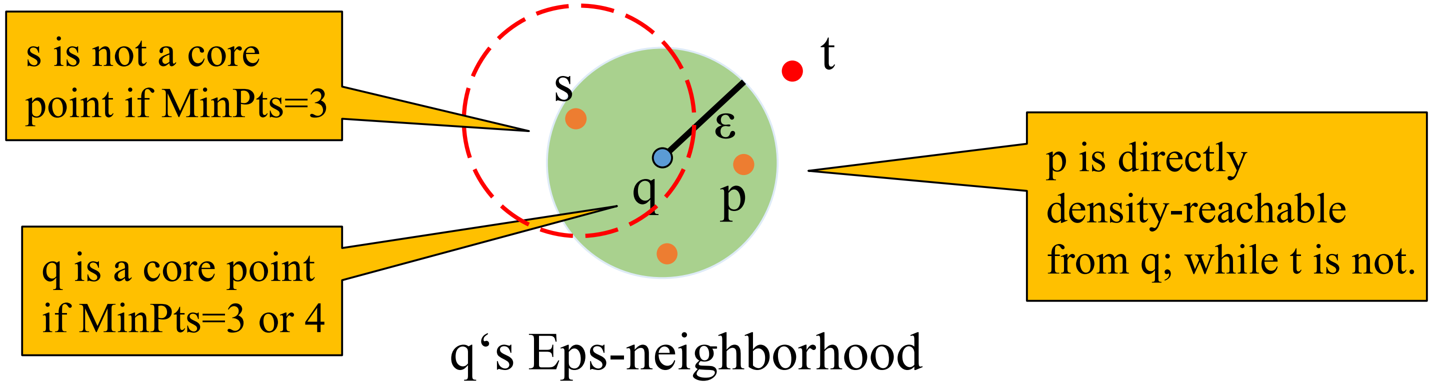


DBSCAN

- Density Based Spatial Clustering of Applications with Noise
- Outliers will not effect creation of clusters.
- Algorithm parameters (hyperparameters)
  - **MinPts** – minimum number of points in a cluster
    - Size of a cluster (number of points)
    - **min_samples** in sklearn.cluster.DBSCAN
  - **Eps** – for each point in a cluster there must be another point in it less than this distance away.
    - Distance between points
    - **eps** in sklearn.cluster.DBSCAN

DBSCAN Concepts (1)

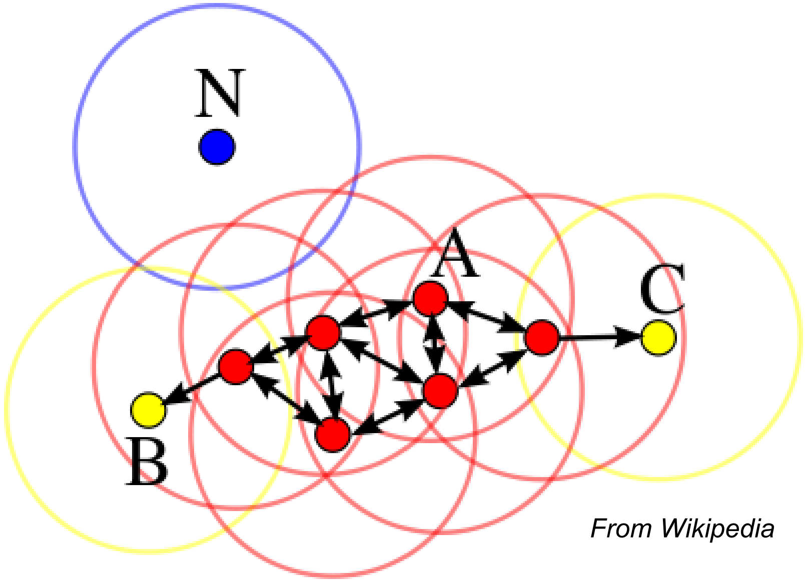
- **Eps-neighborhood**
  - Covers all points within Eps distance of a point.
- **Core point**
  - Whose Eps-neighborhood is dense enough (with at least MinPts points)
- **Directly density-reachable**
  - A point *p* is directly density-reachable from another point *q* if the distance is small (≤ Eps) and *q* is a core point.



DBSCAN Concepts (2)

- **Density-reachable***:* A point *p* is density-reachable from another point *q* if there is a *path* from *q* to *p* and the path consists of only core points.
  - I.e., if there is a chain of points $p_1=q$, p_2, …, $p_n=p$ such that p_{i+1} is directly density-reachable from p_i. More specifically,
    1. p_1, …, p_{n-1} are core points;
    2. the distance between each pair ≤ Eps;
    3. *p* may not be a core point.
  - Density-reachable is *not* symmetric.
    - A is not density-reachable from B or C as they are not core.

Assume MinPts=3.
- Red points are core points.
- Points B and C are *density-reachable* from A.
- Point B is not density-reachable from C; and vice versa.

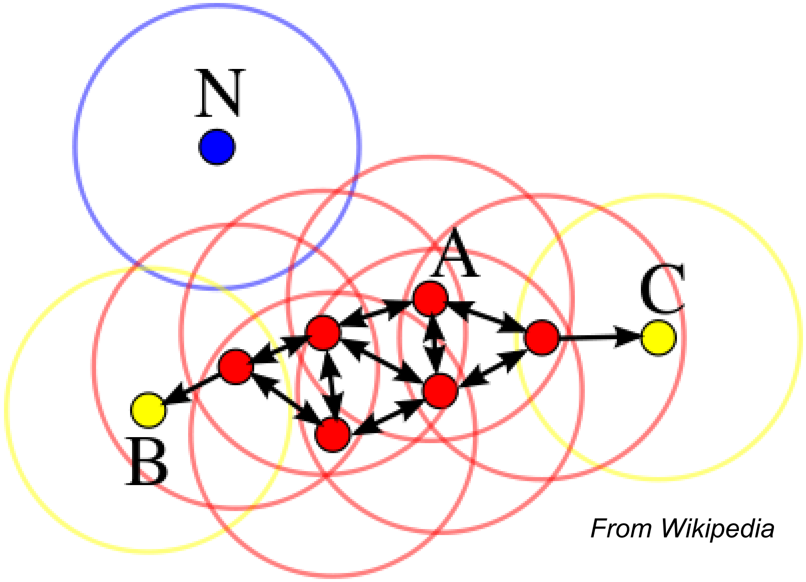


From Wikipedia

DBSCAN Concepts (3)

- **Density-connected***:* two points *p* and *q* are density-connected if there is a point *o* such that both *p* and *q* are density-reachable from *o*.
  - B and C are density-connected (via A).
  - Density-connected is symmetric.

- Clusters in DBSCAN
  - A cluster contains at least MinPts points
  - Density-connected points go to the same cluster
    - E.g., all red points plus B and C
- Outliers in DBSCAN
  - Those points not in any cluster



From Wikipedia

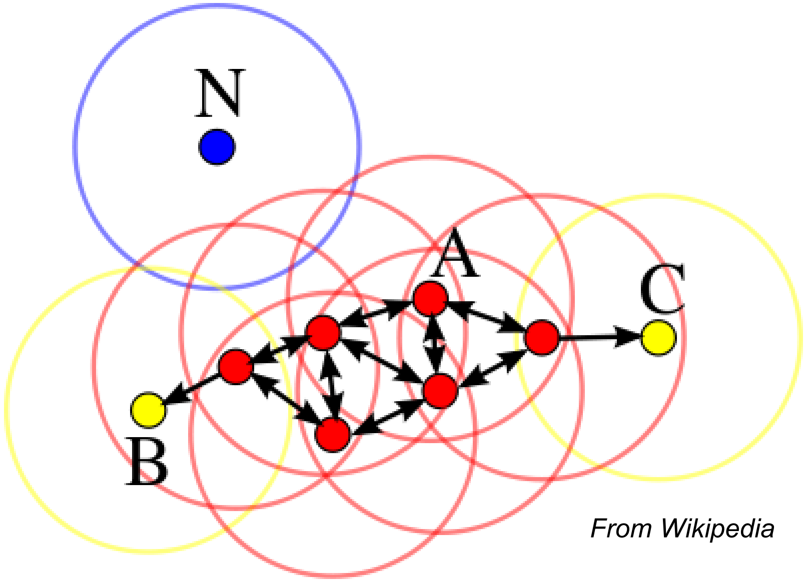
DBSCAN Algorithm

```
DBSCAN(D, eps, MinPts)
   C = 0
   for each unvisited point P in dataset D
      mark P as visited
      NeighborPts = regionQuery(P, eps)
      if sizeof(NeighborPts) < MinPts
         mark P as NOISE
      else
         C = next cluster
         expandCluster(P, NeighborPts, C, eps, MinPts)

expandCluster(P, NeighborPts, C, eps, MinPts)
   add P to cluster C
   for each point P' in NeighborPts
      if P' is not visited
         mark P' as visited
         NeighborPts' = regionQuery(P', eps)
         if sizeof(NeighborPts') >= MinPts
            NeighborPts = NeighborPts joined with NeighborPts'
      if P' is not yet member of any cluster
         add P' to cluster C

regionQuery(P, eps)
   return all points within P's eps-neighborhood
```

From Wikipedia

DBSCAN Properties

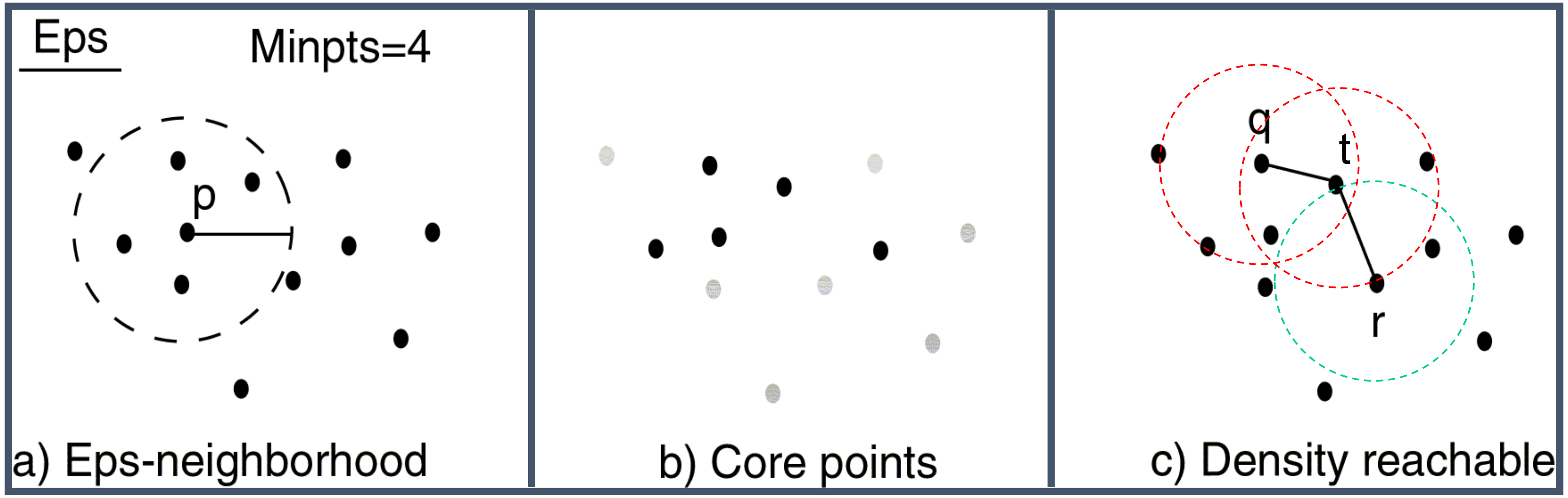
- A cluster satisfies two properties:
  - All points within a cluster are mutually density-connected.
  - If a point *p* is density-connected to any point of a cluster, *p* belongs to the same cluster as well.
- In this example, point N is not included in any cluster. It is a *noise point*, neither a core point nor density-reachable.



From Wikipedia

Another DBSCAN Example

- Point r is not a core point but it is in the Eps-neighborhood of core point t
- Point r is density reachable from q, not vice versa.



Example in Jupyter Notebook

- Datasets
  - make_blobs
  - make_moons
- We need to notice
  - Effect of eps and min_samples
  - Effect of noises (outliers) in the data

- Lecture8_DBSCAN.ipynb
  - from sklearn.cluster import DBSCAN

Agenda

- DBSCAN
- Evaluation of clustering
- Feature engineering

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters
  - high *intra-cluster* similarity: **cohesive** within clusters
  - low *inter-cluster* similarity: **distinctive** between clusters
- The quality of a clustering method depends on
  - the similarity measure used by the method
  - its implementation (e.g., hyperparameters), and
  - its ability to discover *some* or *all* of the hidden patterns

Evaluation of Clustering in Scikit-Learn

- If clustering groundtruth is available
  - Compare the clustering result with the groundtruth by measuring a score
    - Adjusted Rand Index (**ARI**): adjusted_rand_score(groundtruth, clustering_result)
    - Normalized Mutual Information (**NMI**): normalized_mutual_info_score(groundtruth, clustering_result)
- Otherwise
  - **Silhouette score**
  - silhouette_score(X, clustering_results) computes the *compactness* of a cluster
- All scores are in sklearn.metrics.cluster
  - The higher a score is, the better the clustering result.

Rand Index (William M. Rand 1971)

Advanced

- A set $S = \{o_1, \ldots, o_n\}$. Two partitions: $X = \{X_1, \ldots, X_r\}$ and $Y = \{Y_1, \ldots, Y_r\}$
  - a: #pairs of elements in S that are in the same X_i and in the same Y_j
  - b: #pairs of elements in S that are in different X_is and in different Y_js
  - c: #pairs of elements in S that are in the same X_i but in different Y_js
  - d: #pairs of elements in S that are in different X_is but in the same Y_j
- Rand Index $\boldsymbol{R} = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}}$, where $\binom{n}{2} = \frac{n(n-1)}{2}$ (binomial coefficient)
  - A value between 0 and 1.
  - 0: the two clusterings do not agree on any pair of points.
  - 1: the two clusterings are exactly the same.
- Example
  - Dataset: {A, B, C, D, E}
  - Method 1 Clusters: {{A, B, C}, {D, E}} , Method 2 Clusters: {{A, B}, {C, D}, {E}}
  - a=1: {A, B}; b=5: {A, D}, {A, E}, {B, D}, {B, E}, {C, E}; a+b+c+d=$\binom{5}{2}$=10
  - R = (1+5)/10 = 0.6

Adjusted Rand Index

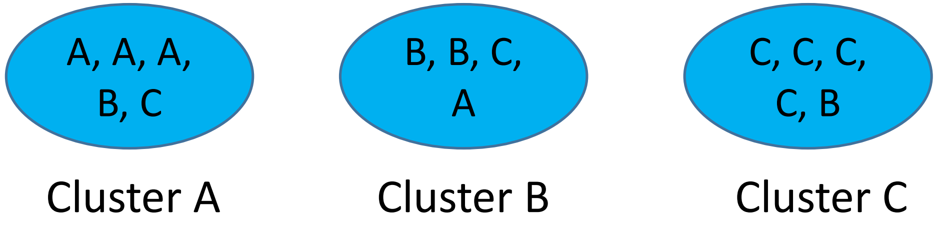
- A set $S = \{o_1, \ldots, o_n\}$. Two partitions: $X = \{X_1, \ldots, X_r\}$ and $Y = \{Y_1, \ldots, Y_r\}$
- **The contingency table**: $n_{ij} = |X_i \cap Y_j|$
  - Each entry denotes the number of objects in *common* between X_i and Y_j

$X \backslash Y$	Y_1	Y_2	$\cdots$	Y_s	sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
sums	b_1	b_2	$\cdots$	b_s	

- Adjusted Rand Index $$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

Purity Score

- If we have the groundtruth for clustering, the best case would be that each 'predicted' cluster contains only objects from the same groundtruth cluster.
  - For each cluster, we 'label' it with the most frequent 'groundtruth' cluster 'label'.

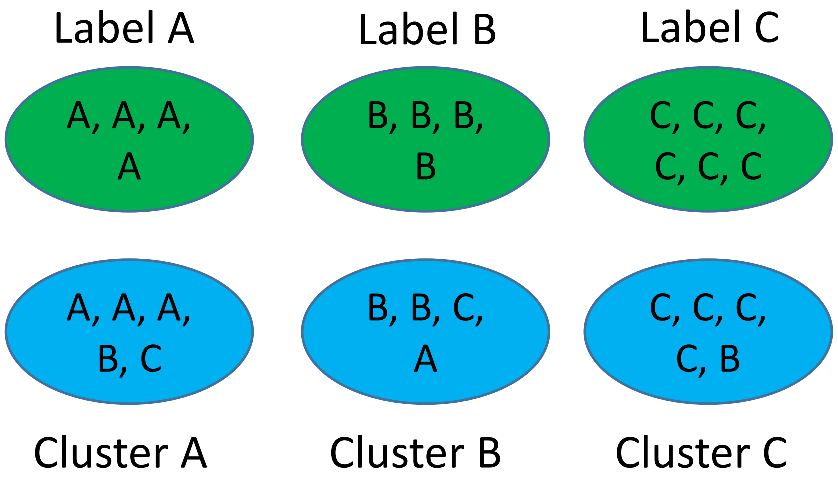


  - **Purity Score** is the average number of 'correct' cluster labels cross all clusters.
  - In this example, Purity = (3+2+4)/(5+4+5) = 9/14 = 0.642
- However, if we put each object in its own singleton cluster, we will always get Purity maximized to 1! Therefore, we need to take into account the number of clusters as well.

Normalized Mutual Information

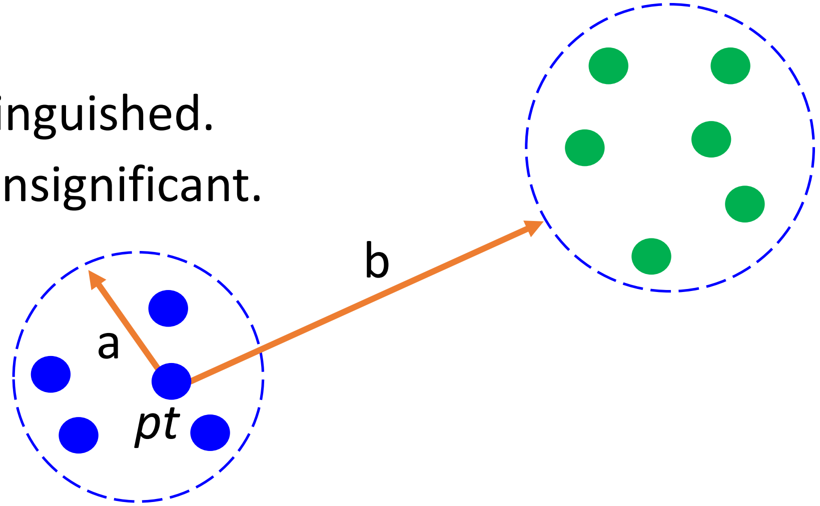
Advanced

- Consider the groundtruth as Y, and the clustering result as C
- Mutual Information tells how Y and C, as two splits, *agree* with each other
  - how much information they share about each other, or how can you know about one of them if you know the other one
  - I(Y; C) = entropy(Y) – entropy(Y|C)
- Normalized Mutual Information
  - NMI(Y, C) = 2*I(Y; C) / (entropy(Y)+entropy(C))
- Entropy is a measure that quantifies uncertainty.
  - Entropy(S) = $-\Sigma p_i * \log_2(p_i)$
  - entropy(Y|C):
    - conditional entropy of labels given the clustering result C
- For more details of entropy and NMI
  - https://course.ccs.neu.edu/cs6140sp15/7_locality_cluster/Assignment-6/NMI.pdf
  - https://towardsdatascience.com/evaluation-metrics-for-clustering-models-5dde821dd6cd



Silhouette Score

- Silhouette score for one point *pt*
  - s(*pt*) = (b - a) / max(a, b)
  - a: the average distance between *pt* and all others in the same cluster (**cohesive**)
  - b: the smallest average distance between *pt* and all points in any other cluster (**distinctive**)
- Silhouette score for a clustering result *X*
  - s(*X*) = ($\bar{b}$ - $\bar{a}$) / max($\bar{a}$, $\bar{b}$)
    - $\bar{a}$, $\bar{b}$: Average a and b for all points in the dataset
  - 1: Clusters are well apart from each other and clearly distinguished.
  - 0: Clusters are indifferent. The distance between them is insignificant.
  - -1: Clusters are assigned in the wrong way.
- Used when groundtruth is *unavailable*



Example in Jupyter Notebook

- Data
  - Two moons
  - Shopping data
- NB:
  - Different scoring functions may fit different scenarios (data, clustering method)

- Lecture8_clustering_evaluation.ipynb

Applications of Clustering

- **Biology**: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- **Information retrieval**: document clustering
- **Land use**: Identification of areas of similar land use in an earth observation database
- **Marketing**: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- **City-planning**: Identifying groups of houses according to their house type, value, and geographical location
- **Earth-quake studies**: Observed earth quake epicenters should be clustered along continent faults
- **Climate**: understanding earth climate, find patterns of atmospheric and ocean
- **Economics**: market research

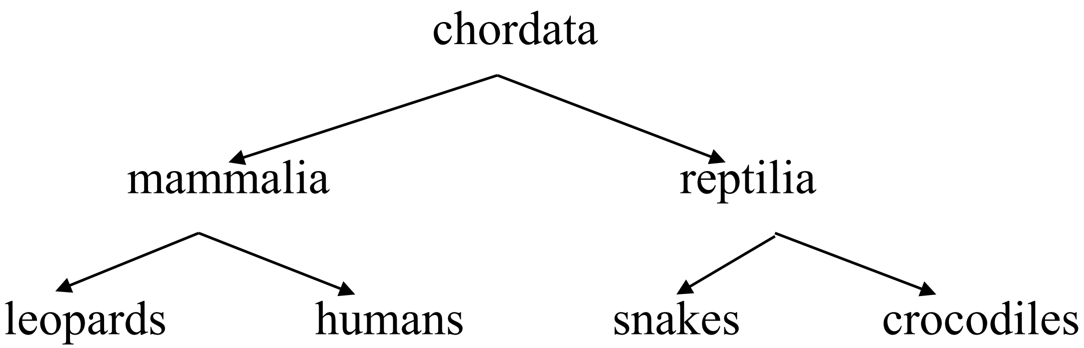
How to choose a clustering method?

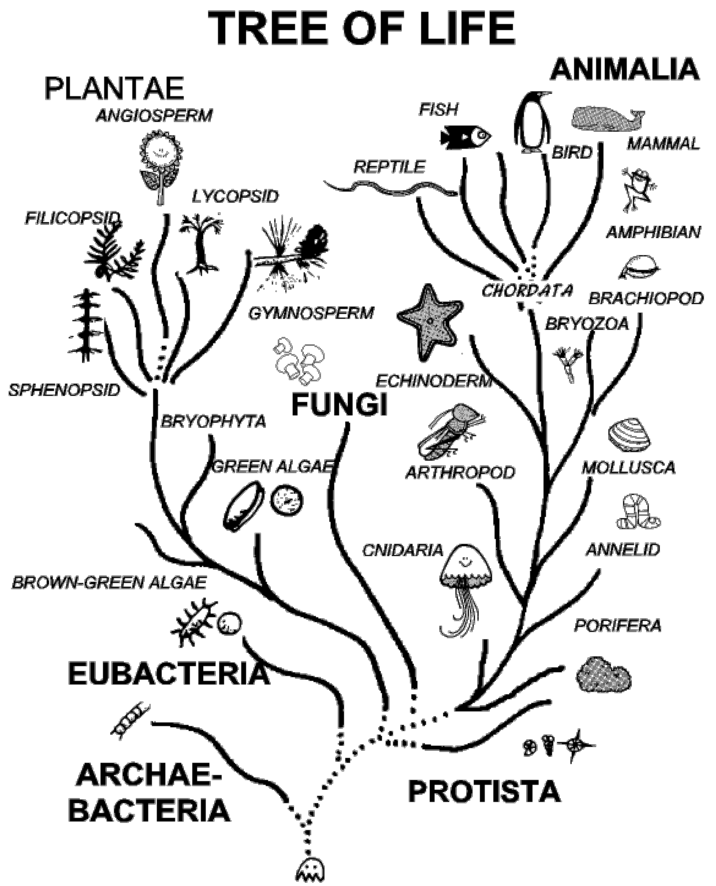
- K-means
  - Only applicable to continuous domains
  - Need to specify k
  - Unsuitable for non-convex shapes
- Agglomerative (hierarchical)
  - If your data is hierarchical
  - If you don’t know how many clusters you should have
- DBSCAN (density based)
  - If your data contains noise or your resulted cluster can be of arbitrary shapes
  - If you want to be able to isolate outliers
- Ask yourself: Which method fits your data best?

Taxonomy

Which clustering method to use for these datasets?

- Biology taxonomy

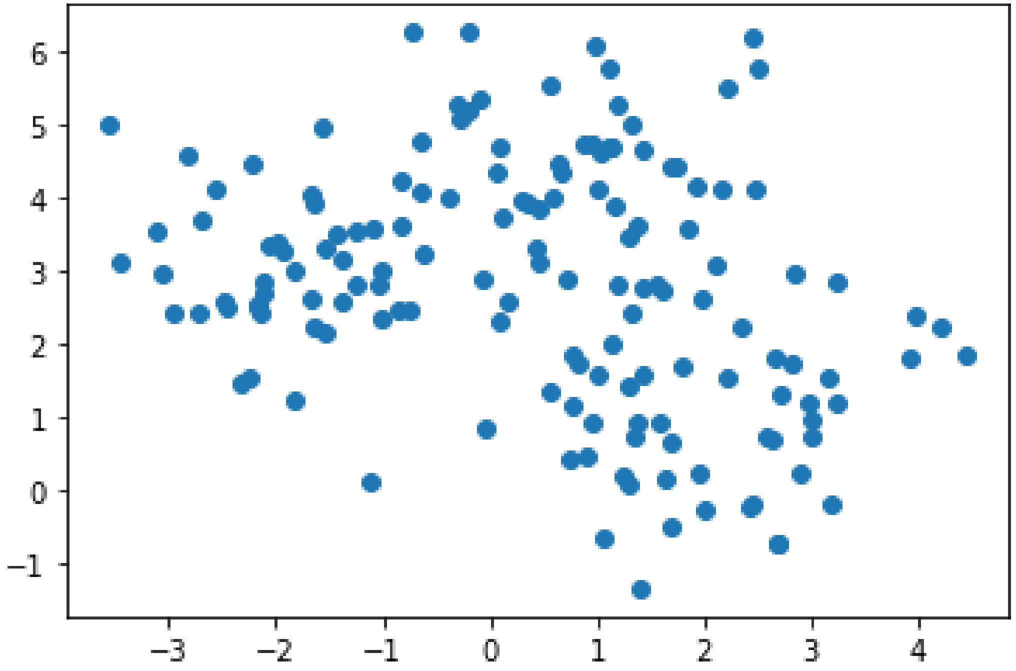




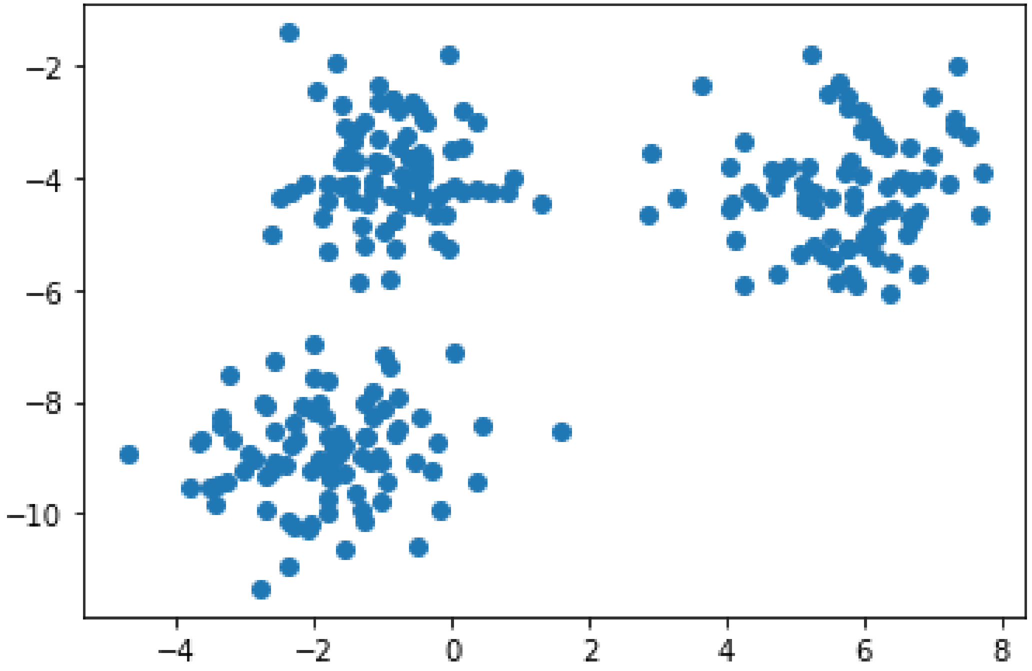
https://www.msnucleus.org/membership/html/k-6/lc/organ/6/lco6_3a.html

Random distributions

- A single random distribution



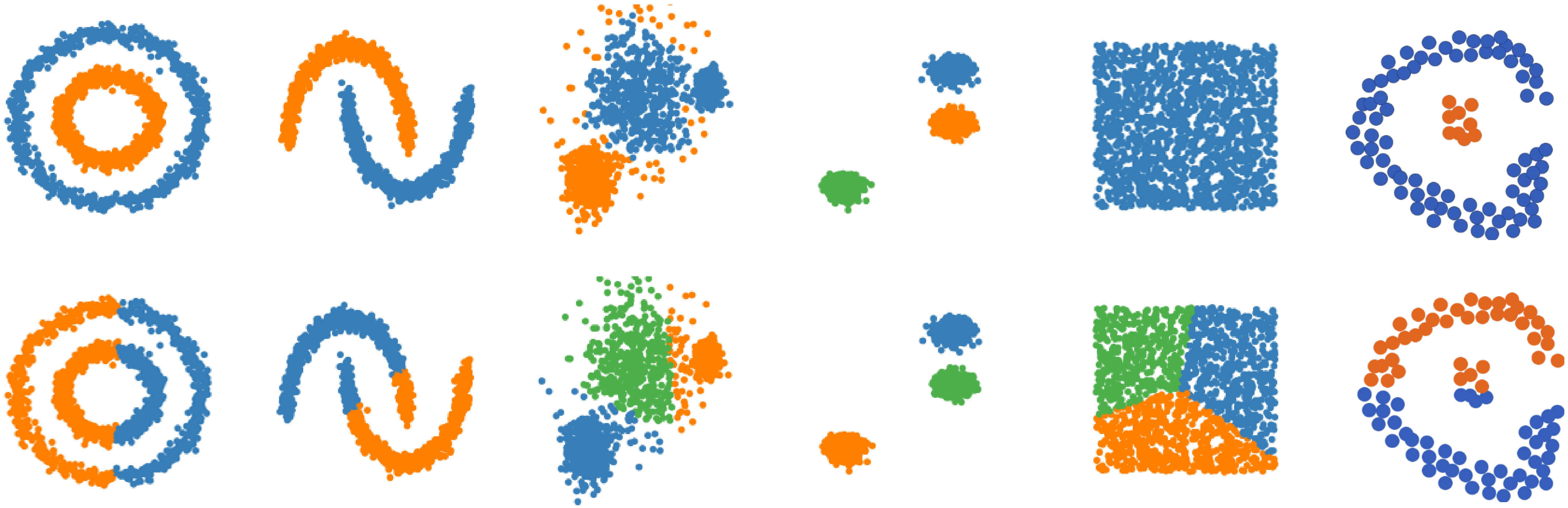
- Multiple distributions



Which clustering method to use for these datasets?

Special shapes: DBSCAN vs K-means

- Which is by which?



https://github.com/NSHipster/DBSCAN

Last column: https://towardsdatascience.com/understanding-dbscan-and-implementation-with-python-5de75a786f9f

Agenda

- DBSCAN
- Evaluation of clustering
- Feature engineering
  - One-hot-encoding
  - Binning
  - Automatic feature selection (advanced)

Feature Engineering

- Using the features/attributes in a dataset to create additional features that are (*hopefully*) better at representing the underlying structure of the data.
  - Sometimes a dataset contains only a limited number of features.
  - Some models work better if more features are provided.
- We can generate new features based on existing *numeric* ones:
  - **polynomials**: polynomial function of original features
    - Polynomial regression: x -> (1, x, x^2, x^3, …) -> linear regression
  - **univariate nonlinear functions**: e.g., sin, exp, log
    - x -> sin(x), 2^x, ln(x)…
  - **binning/discretization**: divide a range into a small number of bins
- We may also need special treatment for *categorical* dimensions.

Categorical or Numeric Values

- Categorical values

	Name	Gender	Department
0	Alex Adam	Male	IMT
1	Babara Brian	Female	ISE
2	Cindy Carlsen	Female	INM
3	David Dickens	Male	IKH

Most models don’t accept categorical values or handle them meaningfully.

- Converted to numeric values

	Name	Gender	Department
0	Alex Adam	**1**	1
1	Babara Brian	**0**	3
2	Cindy Carlsen	**0**	2
3	David Dickens	**1**	0

- Some models may internally carry out operations on numeric values, e.g., on Department 1+3+2+0/4=1.5.
- Some model may consider larger values are better than smaller ones, e.g., on Gender 1>0.
- How to avoid such meaningless cases?

One-Hot-Encoding

In pandas:
data_dummies = pandas.get_dummies(data)

- Aka one-out-of-N encoding
  - Each N-valued categorical domain is represented by N boolean features.
  - For each data point, only one of the N features in a converted domain is set to 1 (hot).

		Gender: N=2		Department: N=4			
	Name	**Gender_Male**	**Gender_Female**	**Dpt_IKH**	**Dpt_IMT**	**Dpt_INM**	**Dpt_ISE**
0	Alex Adam	1	0	0	1	0	0
1	Babara Brian	0	1	0	0	0	1
2	Cindy Carlsen	0	1	0	0	1	0
3	David Dickens	1	0	1	0	0	0

Training and Test Data for One-Hot-Encoding

- For each original feature, the **same** encoding schema should be used for training data and test data. (Encoding before splitting)
  - Number, sequence and semantics of features
- What will be wrong for the following?

Training data

	Name	Gender	Department
0	Alex Adam	Male	IMT
1	Babara Brian	Female	ISE
2	Cindy Carlsen	Female	INM

Dpt_IMT	Dpt_INM	Dpt_ISE

Test data

	Name	Gender	Department
0	Wendy Allen	Female	INM
1	Bob Brian	Male	IKH
2	Cindra Kim	Female	IMT

Dpt_IKH	Dpt_IMT	Dpt_INM

- We must ensure all categories appear in both training and test data

Discrete Numeric Values

- Many *discrete* numeric values in a given dataset do not mean continuous values or features, but they may be treated as continuous values by models.
  - E.g., Gender column in the table
    - 1 and 0 here are still ‘categorical values’
  - So get_dummies(.) on the *whole* dataset will ignore those numeric values.

	Name	Gender	Department
0	Alex Adam	1	IMT
1	Babara Brian	0	IKH
2	Cindy Carlsen	0	INM
3	David Dickens	1	ISE

- In one-hot-encoding, we can specify columns to transform:
  - data[‘Gender’] = data[‘Gender’].astype(str)   # Change the type first
    pd.get_dummies(data, columns=[‘Gender’, ‘Department’])

Example in Jupyter Notebook

- Data
  - aduts.csv
  - In the Moodle

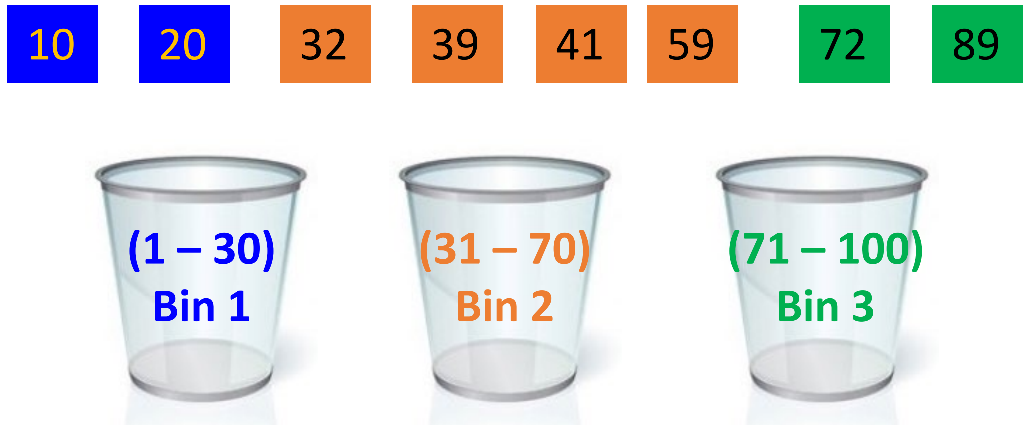
	age	workclass	education	gender	hours-per-week	occupation	income
0	39	State-gov	Bachelors	Male	40	Adm-clerical	<=50K
1	50	Self-emp-not-inc	Bachelors	Male	13	Exec-managerial	<=50K
2	38	Private	HS-grad	Male	40	Handlers-cleaners	<=50K
3	53	Private	11th	Male	40	Handlers-cleaners	<=50K
4	28	Private	Bachelors	Female	40	Prof-specialty	<=50K

- NB
  - Modelling requires numeric values
  - Whole dataset vs. selected columns

- Lecture8_onehotencoding.ipynb

Binning

- Dividing a *continuous* feature into distinct, *categorical* groups.
  - Fixed-Width Binning: pandas.cut(.)
  - Quantile Binning (Fixed-Frequency): pandas.qcut(.)
  - Binning with labels
- After binning, one-hot-encoding is often used



Example in Jupyter Notebook

- Data
  - the age_income_data
- Column for binning
  - age

- Lecture8_binning.ipynb

Benefits of Binning

- Data after binning
  - Fewer possible data values (a few categories instead of an infinite range)
  - More certain and stable values
  - Information become blurred and less precise
- Then why binning?
  - To avoid overfitting a model
  - To speed up model construction and training
  - To increase the **stability** and **robustness** of a model
    - A model is *stable* if its prediction does not change much for slight changes in training data.
    - A model is *robust* if it still makes reliable predictions for noise or adversarial data.
      - E.g., trained on age range (9, 89), predicting for new_age=1000
  - Data smoothing
    - E.g., using the mean of a category to replace outlier raw values in the category

Automatic Feature Selection

Advanced

- What if we need to choose from a set of features?
  - We certainly want to use features that result in good performance of data modelling.
- Automatic feature selection
  - Univariate Statistics
  - Model-based Selection
  - Iterative Selection

Univariate Statistics

Advanced

- Principle
  - Consider each feature *f* individually.
  - A significant relationship between *f* and the target?
  - Select those *f*s that are related with the highest confidence.
- A.k.a. univariate feature selection in scikit-learn
  - sklearn.feature_selection
  - Two classes: SelectPercentile and SelectKBest
    - First parameter: *score_func=<function f_classif>*
      - f_classif (for classification, default) or f_regression

Model-based Selection

Advanced

- Principle
  - Use a supervised learning model to judge the importance of each feature.
    - A different model than the final task can be used
  - Select only the most important features
  - All features are considered at once
- In scikit-learn
  - from sklearn.feature_selection import SelectFromModel
  - DT and DT-based models provide an attribute feature_importances_

```
selector = SelectFromModel(RandomForestClassifier(n_estimators=100, random_state=42), threshold="median")
selector.fit(X_train, y_train)
X_train_l1 = selector.transform(X_train) # Use only the features selected by the Model-based selection
LogisticRegression(max_iter=1000).fit(X_train_l1, y_train) # Do a regression using the selected features only
```

Iterative Selection

Advanced

- Principle
  - Multiple models, and multiple features incrementally (adding or elimination)
  - Recursive feature elimination (RFE)
    - Starts with all features to build a model, discards the least important features, and builds a new model with the remaining features.
    - Repeats until a pre-specified number of features remain
- In scikit-learn
  - from sklearn.feature_selection import RFE

```
selector = RFE(RandomForestClassifier(n_estimators=100, random_state=42), n_features_to_select=40)
selector.fit(X_train, y_train)
X_train_rfe = selector.transform(X_train) # Use only the features selected by the Model-based selection
LogisticRegression(max_iter=1000).fit(X_train_rfe, y_train) # Do a regression using the selected features only
```

Example in Jupyter Notebook

Advanced

- mpg data
  - (398, 9)
  - Mile per gallon: fuel economy about cars
  - A regression problem: to predict mpg values

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	usa	ford torino

- Lecture8_AFS.ipynb

jupyter DEMO

Automatic Feature Selection: Comparison

Advanced

	Pros	**Cons**
Univariate Statistics	• Simple to use • Works if too many (uninformative) features	• Consider features separately • Effect might be not very good
Model-based Selection	• Consider all features to capture interaction	• Bias of the used model
Iterative Selection	• Better features selected	• High computational cost

Notebooks and Data

- Lecture8_DBSCAN.ipynb
  - make_blobs
  - make_moons
- Lecture8_clustering_evaluation.ipynb
  - Two moons
  - shopping_data.csv
- Lecture8_onehotencoding.ipynb
  - aduts.csv
- Lecture8_binning.ipynb
  - Ch5_age_income_data.csv
- Lecture8_AFS.ipynb
  - Mile per gallon data (mpg.csv)

Summary

- Clustering
  - DBSCAN
  - Clustering evaluation
  - Clustering method choosing
- Feature engineering
  - One-hot-encoding
  - Data binning
  - Automatic feature selection
    - Univariate Statistics
    - Model-based Selection
    - Iterative Selection

References

- Mandatory reading
  - Muller and Guido: Introduction to Machine Learning with Python, O'Reilly, 2016
    - Chapter 3: Clustering: DBSCAN, Comparing and Evaluating Clustering Algorithms, Summary of Clustering Methods
    - Chapter 4: Categorical Variables, Binning, Automatic Feature Selection (optional)
- Further readings
  - DBSCAN documentation
    - https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
  - Automatic feature selection
    - https://lucashomil.github.io/datascience/blog-2.html
    - https://towardsdatascience.com/5-feature-selection-method-from-scikit-learn-you-should-know-ed4d116e4172

Exercises

1. Apply DBSCAN clustering to the bikes dataset (in Moodle) in Jupyter Notebook
   - Vary eps and min_samples
   - Visualize the DBSCAN clustering results with their Silhouette scores
2. Work on the diamonds dataset (in Moodle) in Jupyter Notebook
   1. Plot a histogram of the **price** column
   2. Apply fixed-width binning to the **price** column with 10 bins
   3. Apply quatile binning to the **price** column with 10 bins