FourthBrain

MLE Program, Cohort 11 (MLE11)

Week 8: Unstructured Data, Clustering, Dim. Reduction, Semi-supervised learning, Zero-shot learning

Last Week!

Concepts

- AutoML Libraries
- Neural Networks Basics
- Convolutional Neural Network
- Recurrent Neural Networks
- Graph Neural Networks
- Generative Adversarial Networks

Hands on

- Leveraging deep learning for tabular data

© 2022 FourthBrain

🤖 This Week!

Concepts

- Dealing with Unstructured Data
- Clustering
- Dimensionality Reduction
- Label propagation/label spreading
- Co-training algorithms
- Zero-shot learning

Hands on

- Predicting customer responses and metadata tagging using data visualization with Tensorboard
- Optional Midterm Project Assignment

© 2022 FourthBrain

What questions do you have?

Structured vs Unstructured Data

Types of Big Data

- Structured Data
- Semi Structured Data
- Unstructured Data

Structured Data

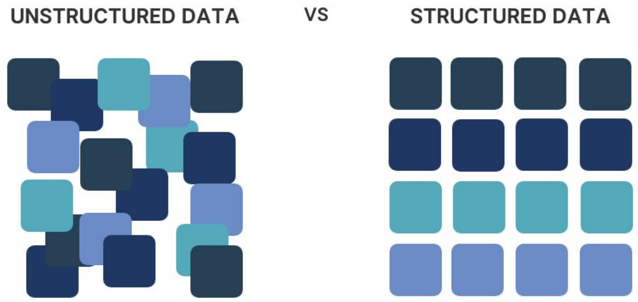
- Also called “Relational Data”
- Data in which elements are addressable for effective analysis
- Organized into a formatted repository (i.e. SQL Database)
- Stored in the repository as tables with rows and columns
- Can have relational keys
- Can easily be mapped into pre-designed fields
- This type of data is the most processed in the development and is considered one of the simplest ways to manage information

Semi-Structured Data

- Information that is not located in a relational database
- It has some organizational properties that make it easier to analyze
- Needs processing to store it in a relational database
- Some semi-structured data is very hard to be stored in relational databases
- Example: XML Data

Unstructured Data

- Data not organized in a predefined manner
- Does not have a predefined data model
- Is not a good fit for a mainstream relational Database
- Mostly used in a variety of IT, Business Intelligence, and Analytics applications
- Example: Word, PDF, Text, Media logs

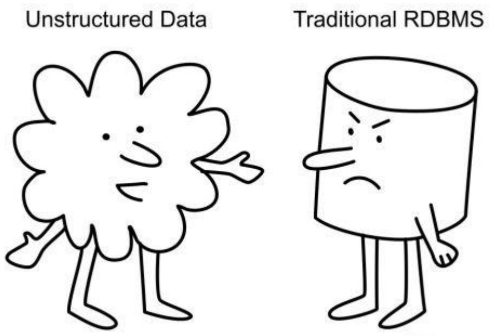


Types of Unstructured Data

- Business Documents
- Emails
- Social Media
- Customer Feedback
- Webpages
- Open Ended Survey Responses
- Images
- Audios
- Videos

Unstructured Data - Challenges

- Absence of unique identifiers
- Different nomenclature
- Different file formats
- Linguistic barrier
- Missing Data
- No Data Architecture
- Different data formats (i.e. dates)



"I'm sorry, I'm just not that into you..."

Unstructured Data

- Until recently, it was so hard to analyze unstructured data due to the hundreds of human hours required to wade through it by hand
- Solution?

Advancements in AI tools!

- Nowadays, it is possible for machines to sort unstructured data automatically
- Save a huge amount of time
- Allow teams to make data-based decisions based on powerful customer insights

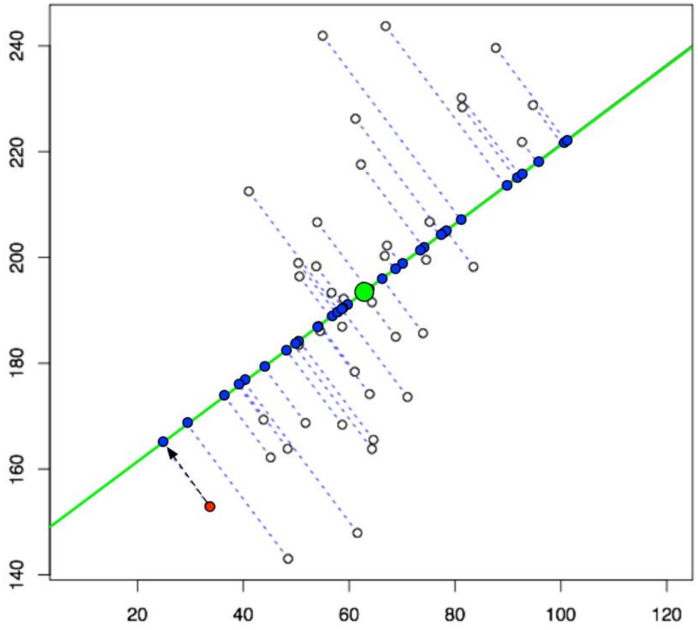
Dimensionality Reduction

Dimensionality Reduction

- Refers to techniques that reduce the number of input variables in a dataset
- This is a very important step especially if we have unstructured data due to the high volume
- More input features often make a predictive modeling task more challenging to model (Generally referred to as the curse of dimensionality)
- Dimensionality reduction is also used for data visualization

Dimensionality Reduction

Example Visualization from 2D to 1D



Benefits?

Dimensionality Reduction - Benefits

- Less training time
- Less computational resources
- Increase in performance of the ML algorithms
- Avoids overfitting
- Useful for data visualization
- Takes care of multicollinearity
- Useful for factor analysis
- Removes noise in the data
- Used for image compression
- Used to transform non-linear data into a linearly-separable form

Principal Component Analysis

- PCA – Principal Component Analysis – is a technique commonly used for reducing the dimensionality of data

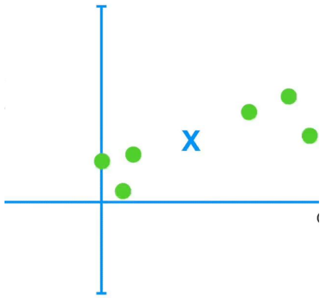
- It preserves as much as possible of the information contained in the original data

- PCA achieves its goal by projecting data onto a lower-dimensional subspace that retains most of the variance among the data points

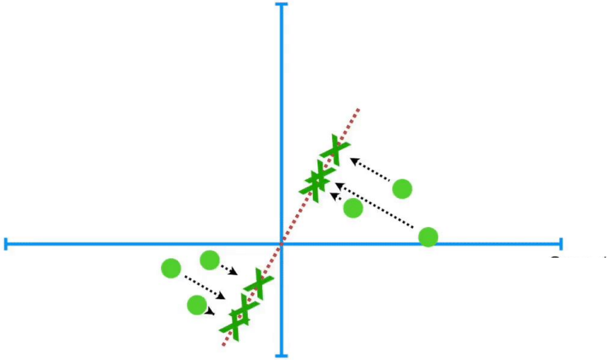
- “SK-Learn” module in python could be used to apply PCA decomposition
- A good article on PCA.

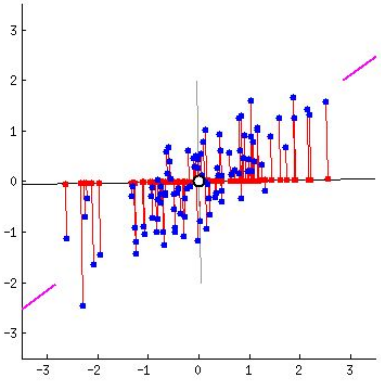
How PCA constructs the principal components

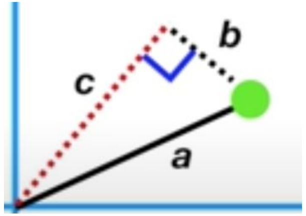
- Principal components are constructed in such a manner that the first principal component accounts for the **largest possible variance** in the data set.



$d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$ = sum of squared distances = SS(distances)







PCA Steps

- Standardize the data
- Calculate the Covariance matrix
- Calculate the Eigenvectors and Eigenvalues of the Covariance matrix
- Reduce the Dimensionality

PCA – Step 1

- Standardize the data
  - Data forming the dataset has often different units and different means
  - This can cause issues such as producing extremely large numbers during calculation
  - To make the process more efficient, center the data at mean zero and make it unit-free
  - You can achieve this by subtracting the current mean from the data and dividing by the standard deviation
  - This preserves correlations but ensures the total variance is equal to 1
- Another approach could be scaling where you scale the data to 0 and 1 (i.e. MinMaxScaler from sklearn library)

PCA – Step 2

- Calculate the Covariance matrix
  - PCA attempts to capture the most of the information in a dataset by identifying the principal components that maximize the variance between observations.
  - The covariance matrix is a symmetric matrix with rows and columns equal to the number of dimensions in the data
  - Covariance matrix tells us how the features or variables diverge from each other by calculating the covariance between the pairwise means

$$\begin{bmatrix} cov(x_1, x_1) & \dots & cov(x_1, x_n) \\ \dots & & \\ cov(x_n, x_1) & \dots & cov(x_n, x_n) \end{bmatrix}$$

PCA – Step 3

- Calculate the Eigenvectors and Eigenvalues of the Covariance matrix
  - Eigenvectors are linearly independent vectors that do not change direction when a matrix transformation is applied
  - Eigenvalues are scalars that indicate the magnitude of the Eigenvector
  - The eigenvectors of the covariance matrix point in the direction of the largest variance
  - The larger the eigenvalue is, the more of the variance is explained
  - In other terms, eigenvector with the largest eigenvalue corresponds to the first principal component, The one with the second largest eigenvalue corresponds to the second principal component, etc.

PCA – Step 4

- Reduce Dimensionality
  - Principal components are efficient feature combinations that ensure that the information explained does not overlap between features.
  - Eliminating information redundancy helps in reducing dimensionality.
  - And since the variance declines with every new principal component, we can reduce dimensionality by eliminating the least important principal components.

- Usually, PCA is performed using a software tool that will execute all these steps automatically for you. (i.e. sklean.decomposition.PCA)

Dimensionality Reduction Applications

- Image and video compression. Also, using Robust PCA for videos and images
- Comp Vision anomalies in objects
- Vector Databases
- Removing multicollinearity
- Complex medical data and medical imaging
- NLP

Clustering

Clustering

- Potential solution to overcome all the problems of unstructured data
- An unsupervised clustering approach enables the business to programmatically bin this data
- The data bins are programmatically generated based on the algorithm’s understanding of the data.
- Why binning?
    - Binning would help tone down the volume of the data
    - Binning would help understand the broader spectrum effortlessly
    - An example would be to only understand the top keywords of the K clusters

Clustering Example: Text Process

- Cleaning and Preprocessing
- Stemming and Lemmatization
- Feature Extraction
- Dimensionality Reduction
- Clustering

Group discussion

5 min
(3-4 per room)

What pre-processing steps of textual data can you think of ?

Designate *one person to share* from your breakout room

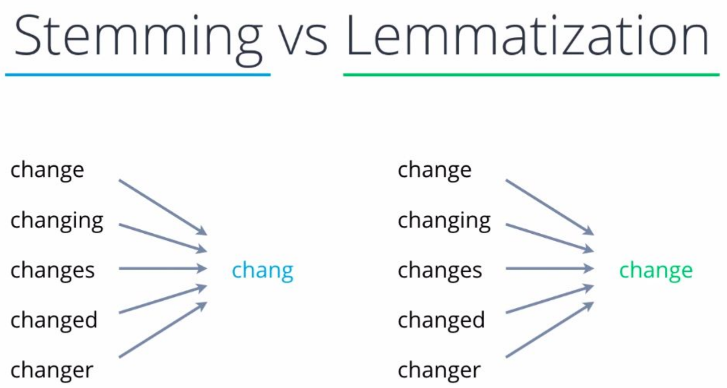
© 2022 FourthBrain

Pre-Processing (Text Clustering Example)

- Cleaning
  - Remove irrelevant items, such as HTML tags
- Normalization
  - Case Normalization and Removing punctuation
- Tokenization
  - Split text into words or **tokens** (symbol)
- Stop Word Removal
  - Removing words that are too common
- Part of Speech Tagging
  - Grouping words together
- Named Entity Recognition
  - Noun phrases that refer to some specific object, person, or place.

Stemming / Lemmatization

- Difference between Stemming and Lemmatization is that
  - The final output in Lemmatization: meaningful word
  - The final output in Stemming: could be an unmeaningful word
- Usually Lemmatization is done before Stemming
- Need of a dictionary
  - Stemming: No
  - Lemmatization: Yes



Feature Extraction – Example in Text

- Extract and produce feature representations that are appropriate for the type of NLP task you are trying to accomplish and the type of model you are planning to use

- The output is mostly a blueprint vector representing the word / sentence

- Example approaches:
  - Bag of words
  - TF-IDF

Bag of Words

- Treat each document (a piece of raw text) as an unordered set of words
- A set of documents is known as a **corpus**
- Collect all unique words in the corpus to form the **vocabulary**
- Let these words form the vector element positions
- Count the number of occurrences of each word in each document
- **Downside:** Treats every word as being equally important

Bag of Words

	littl	hous	prairi	mari	lamb	silenc	twinkl	star
"Little House on the Prairie"	1	1	1	0	0	0	0	0
"Mary had a Little Lamb"	1	0	0	1	1	0	0	0
"The Silence of the Lambs"	0	0	0	0	1	1	0	0
"Twinkle Twinkle Little Star"	1	0	0	0	0	0	2	1

TF-IDF

- TF-IDF: Term Frequency, Inverse Document Frequency
- Some words could be common in the corpus, like cost in a financial document
- TF-IDF assigns weights to words, that signify their relevance in documents
- It is compensated by counting the number of documents in which each word occurs (document frequency)
- Divide the term frequency by the document frequency
- Result gives a metric that is proportional to the frequency of occurrence of a term

$$TF = \frac{Number\ of\ times\ a\ word\ "X"\ appears\ in\ a\ Document}{Number\ of\ words\ present\ in\ a\ Document}$$

$$IDF = log\left(\frac{Number\ of\ Documents\ present\ in\ a\ Corpus}{Number\ of\ Documents\ where\ word\ "X"\ has\ appeared}\right)$$

$$TF\ IDF\ =\ TF * IDF$$

Feature extractions for other domains

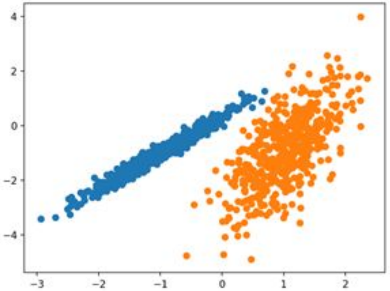
- Features in an image could be represented as a base start as each pixel of the image, and then transformed to be binned together. For example a 30x30 image would have 900 pixels/features

- A video could be split into different frames (pictures) and then the features could be extracted from these images as mentioned above

- Features could be extracted from audios by studying the frequencies and signals generated by each type of the output we get.

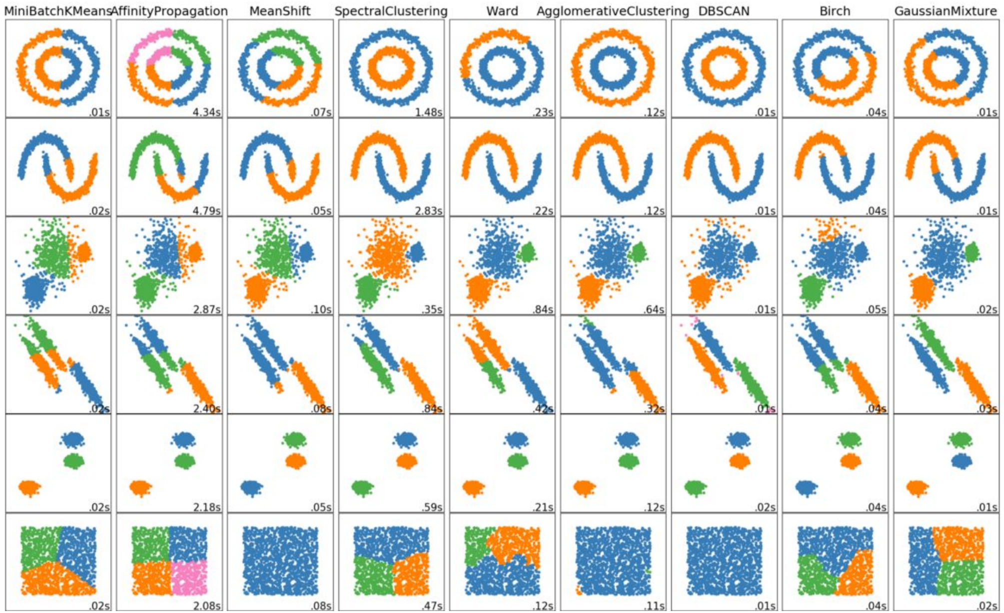
Clustering

- The method of identifying similar groups of data in a dataset
- Entities in each group are comparatively more similar to entities of that group than those of other groups
- Clustering techniques are widely used to solve unsupervised learning problems
- Each clustering technique is distinguished by a unique way of choosing how to cluster the data



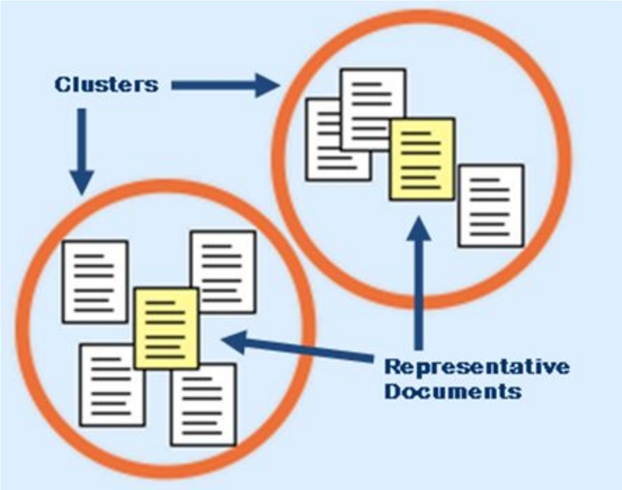
Some Clustering Algorithms

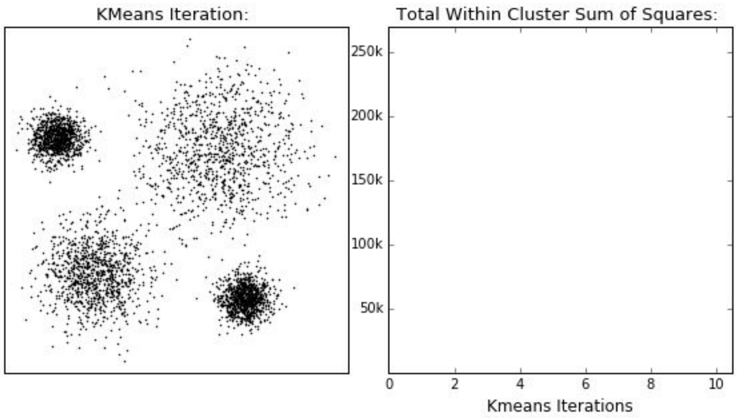
- Affinity Propagation
- Agglomerative Clustering
- BIRCH
- DBSCAN
- K-means
- Mini-Batch K-Means
- Mean Shift
- OPTICS
- Spectral Clustering
- Mixture of Gaussians
- WARD



Data Clustering – K-Means (Text Example)

- After finishing the TF-IDF transformation, the documents are put through a K-Means clustering algorithm
- This computes the Euclidean Distances amongst these documents and clusters nearby documents together

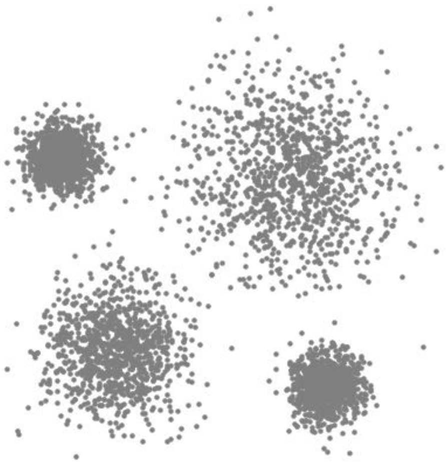




Mean-Shift clustering

- Mean shift clustering is a sliding-window-based algorithm that attempts to find dense areas of data points

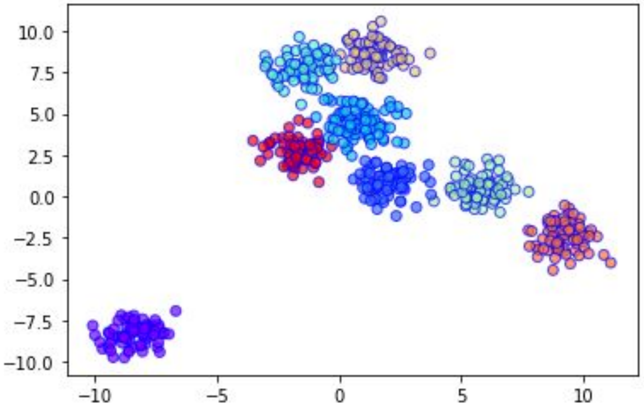
- Centroid-based algorithm



BIRCH Clustering

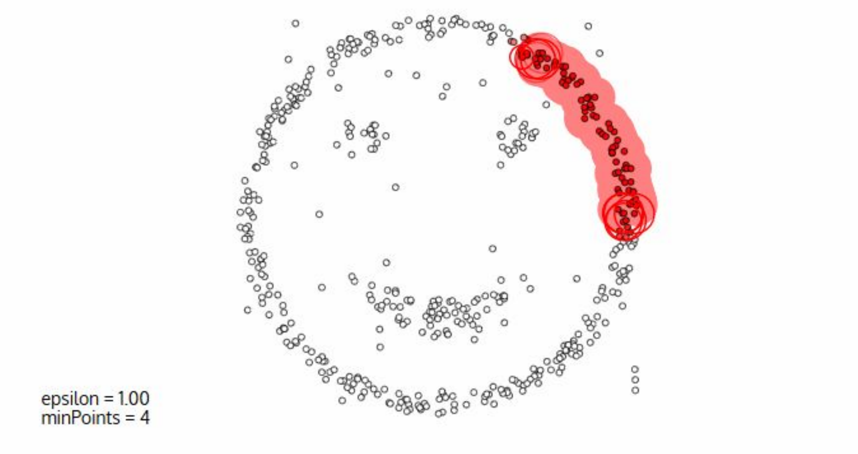
- Balanced Iterative Reducing and Clustering using Hierarchies
- It is a clustering algorithm that can cluster large datasets
- It first generates a small and compact summary of the large dataset
- It then retains as much information as possible
- Lastly, the smaller summary is clustered instead of clustering the larger dataset

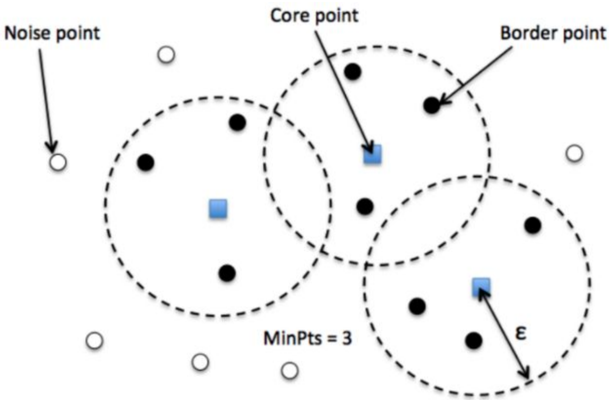
model = Birch(branching_factor = 50, n_clusters = None, threshold = 1.5)



DBSCAN Clustering

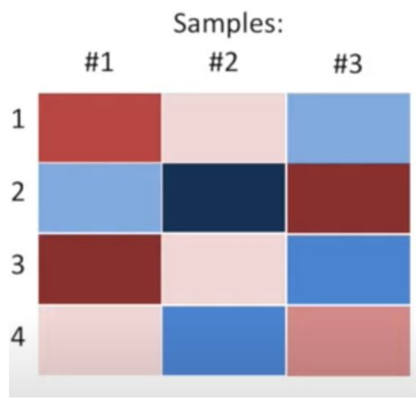
- Density Based Spatial Clustering of Applications with Noise
- Partitioning methods (K-means) and hierarchical clustering works best for finding spherical-shaped clusters (convex clusters)
- DBSCAN fixed this issue and is able to handle real life data clustering that have arbitrary shapes and may contain noise

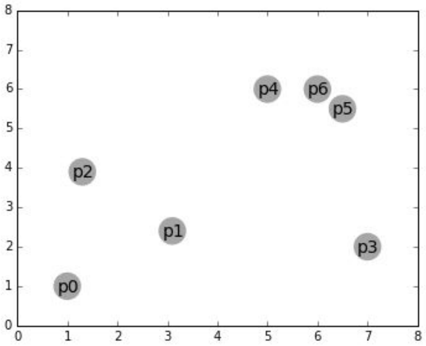


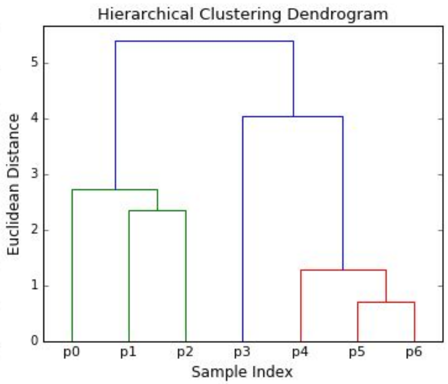


Hierarchical Clustering

- starts by treating each observation as a separate cluster. Then, it repeatedly executes the following two steps:
- (1) identify the two clusters that are closest together
- (2) merge the two most similar clusters.



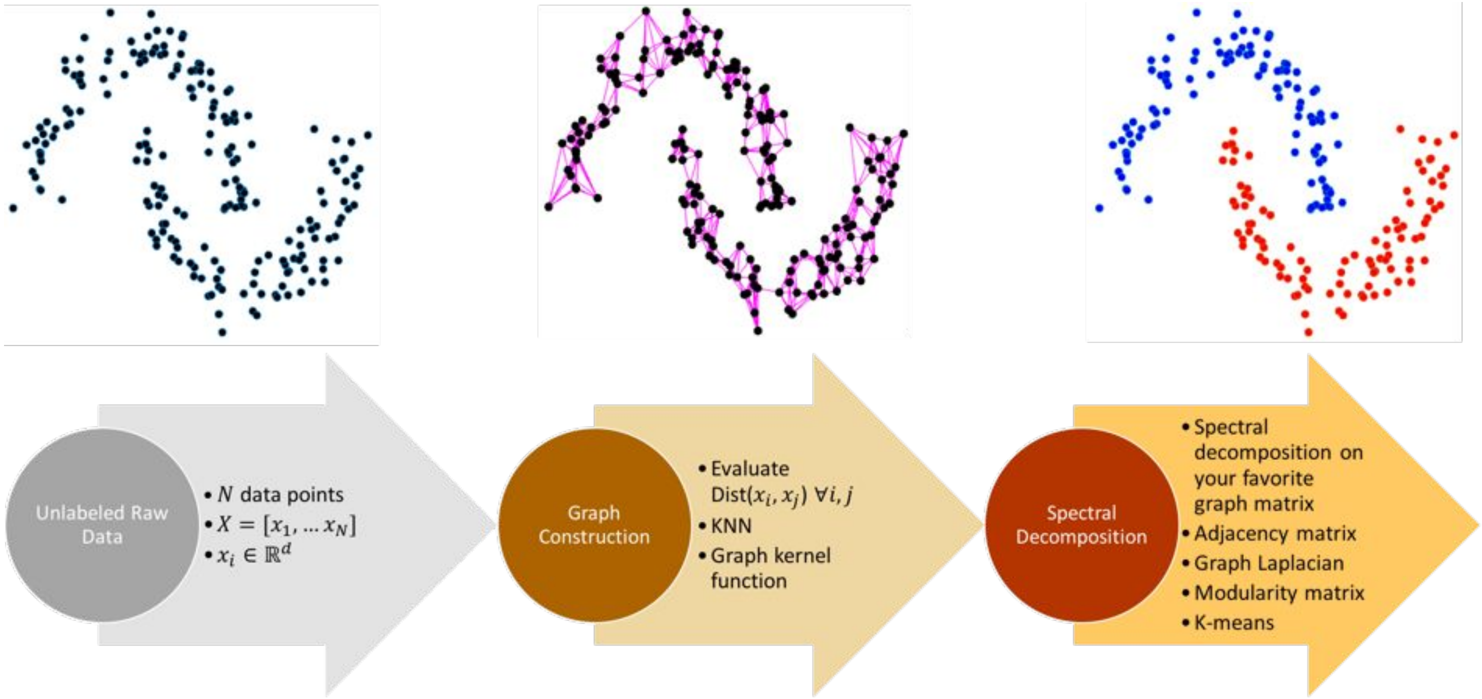




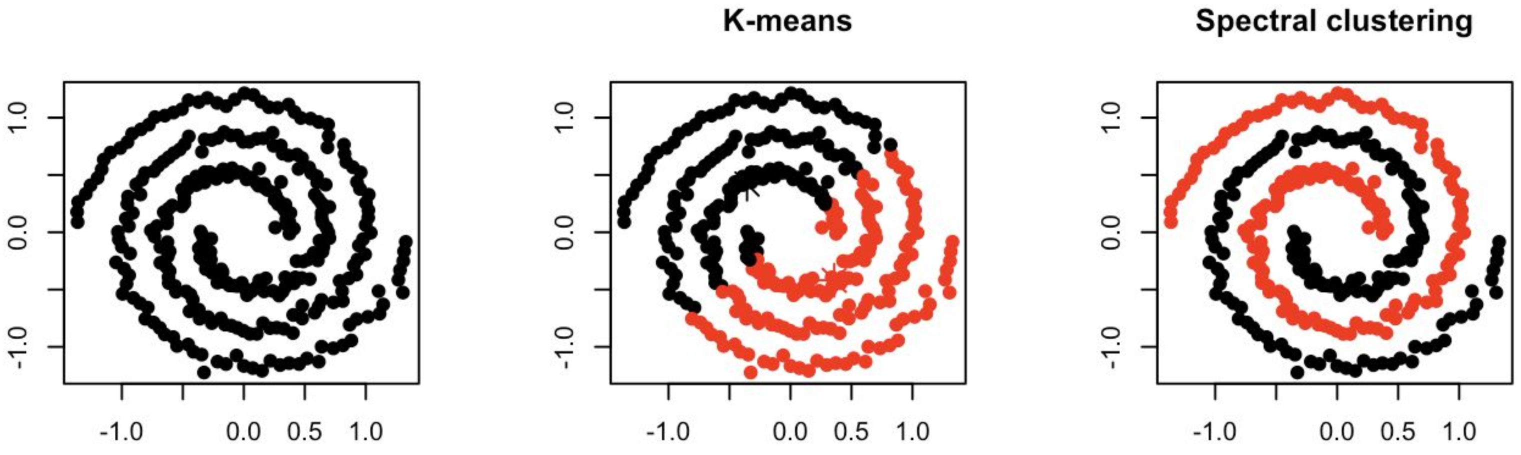
Spectral Clustering

- Make no assumption of the shapes of the cluster (i.e. non convex shape clusters)
- Spectral Clustering is a growing clustering algorithm that has performed better than many traditional clustering algorithms
- It transforms the clustering problem into a graph-partitioning problem
- It treats each data point as a graph-node
- A typical implementation consists of 3 fundamental steps
    - Building the Similarity Graph
    - Projecting the data onto a lower Dimensional Space (eigen vectors of the graph Laplacian)
    - Clustering the data

Spectral Clustering



K-Means vs Spectral Clustering Visualization



Semi-supervised Learning

Label Propagation
and
Label Spreading

Label Propagation Algorithm

- Label Propagation Algorithm (LPA) is a fast algorithm for finding communities in a graph
- It detects these communities using network structure alone
- It does not require a pre-defined objective function or prior information about the communities
- It works by propagating labels throughout the network and forming communities based on this process of label propagation

Label Propagation

- Intuition behind the algorithm:
    - A single label can quickly become dominant in a densely connected group of nodes
    - But this label will have trouble crossing a sparsely connected region
- Labels will get trapped inside a densely connected group of nodes, and those nodes that end up with the same label when the algorithm finishes can be considered as part of the same community

Label Propagation – How it works?

- Every node is initialized with a unique community label (identifier)
- These labels propagate through the network
- At every iteration of propagation, each node updates its label to the one that the maximum number of its neighbors belongs to.
- Ties are broken arbitrarily but deterministically
- LPA reaches convergence when each node has the majority label of its neighbors
- LPA stops if either convergence happens or the user-defined maximum number of iterations is achieved

Label Propagation - Results

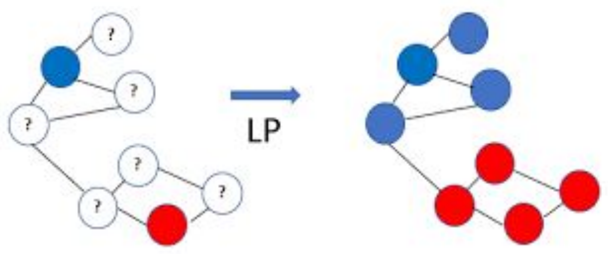
- As labels propagate, densely connected groups of nodes quickly reach a consensus on a unique label.
- At the end of the propagation, only a few labels will remain (while most of the others will have disappeared)
- Nodes that have the same community label at convergence are said to belong to the same community

- Nodes can be assigned preliminary labels to narrow down the range of solutions generated – This means that it can be used as a

Label Propagation - Semi-supervised Learning

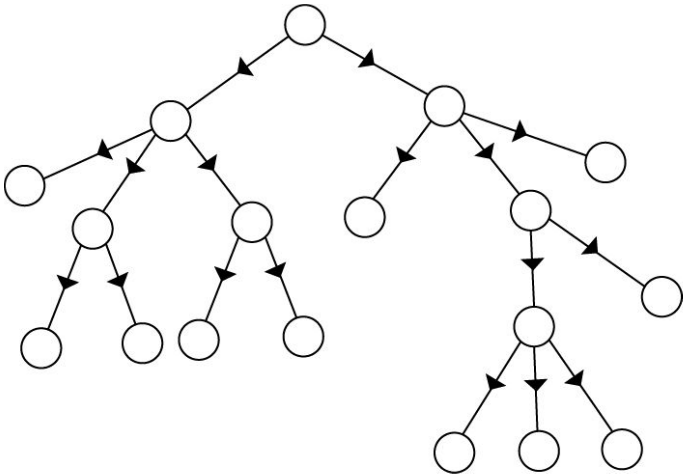
Can label Propagation work as a semi-supervised machine learning algorithm?

- Short answer is yes.
- Nodes can be assigned preliminary labels to narrow down the range of solutions generated
- This means that it can be used as a semi-supervised way of finding communities where we hand-pick some initial communities



Label Propagation

- Works well with:
  - Directed Graphs
  - Undirected Graphs
  - Homogeneous Graphs
  - Weighted Graphs



- Does not work well with:
  - Heterogeneous graphs

Label Spreading Algorithm

- Label Spreading algorithm approach is very similar to the Label Propagation algorithm for semi-supervised learning

- This algorithm was introduced by Dengyong Zhou, et al. in their 2003 paper titled "Learning With Local And Global Consistency."

- The label spreading is inspired by a technique from experimental psychology called spreading activation networks.

- Points in the dataset are connected in a graph based on their relative distances in the input space.

- The weight matrix of the graph is normalized symmetrically, much like spectral clustering.

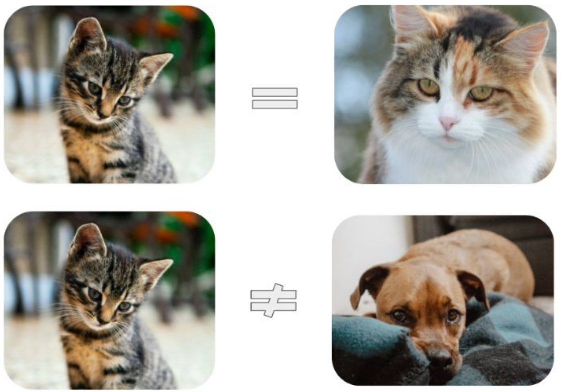
Zero-shot Learning

Zero-Shot Learning (ZLS) - Motivation

- The motivation behind zero-shot learning is that the model should learn how to classify classes it hasn’t seen before
- As an example, think of creating an algorithm that can classify all the animal species in the world
- It would be really hard to have a dataset that contain the 1,899,587 animal species labels!

Zero-Shot Learning – Contrastive Learning

- Contrastive Learning is a technique used to learn the general features of a dataset **without labels** by teaching the model which data points are similar or different
- With contrastive learning, the model performance can be significantly improved even when only a fraction of the dataset is labelled



Zero-Shot Learning – How it works?

- Contrastive Language-Image Pretraining (CLIP) proposed by OpenAI is a Zero-Shot Learning approach that performed well in a zero-shot setting

- The goal is to learn how to classify images without any explicit labels

- The intuition behind CLIP is to use the text related to the image to decide what is in the image

- CLIP has 2 stages:
  - Training Stage (learning stage)
  - Inference Stage (predictions stage)

Zero-Shot Learning – How it works?

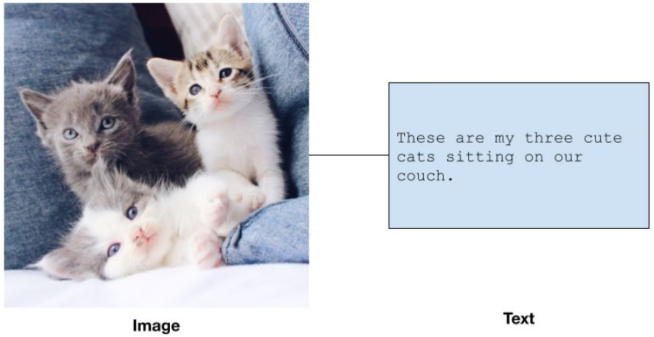
Image

These are my three cute cats sitting on our couch.

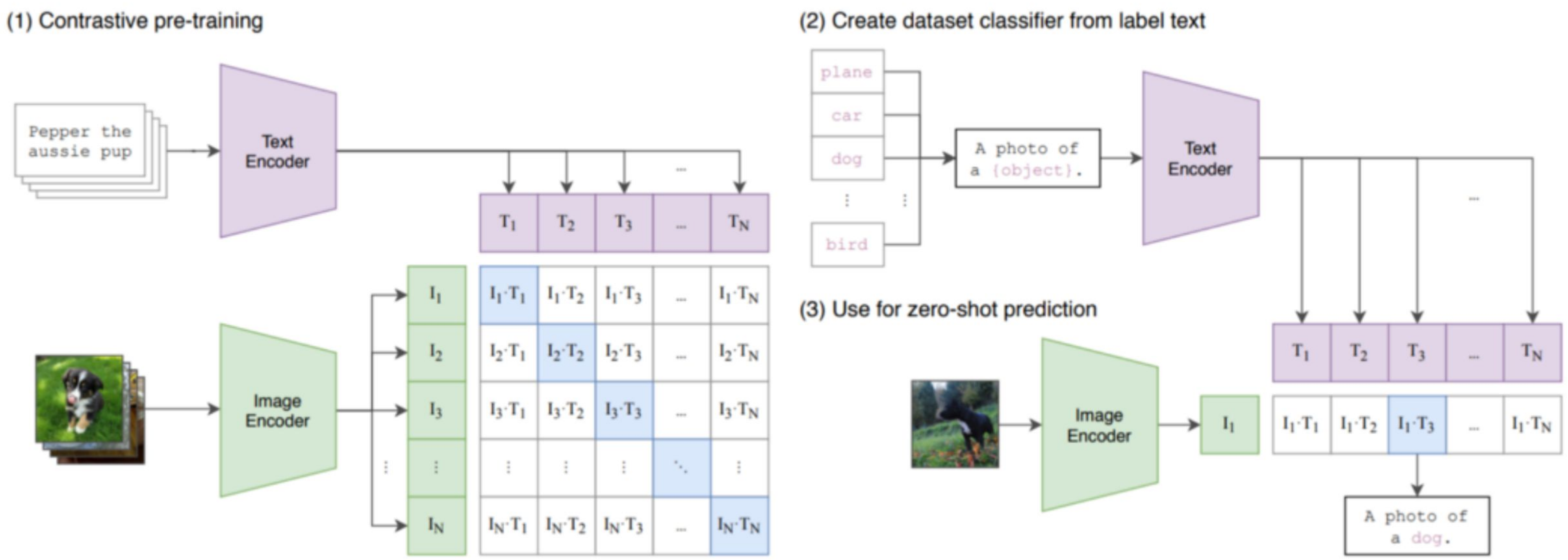
Text

Zero-Shot Learning – How it works?

- As a human that never saw a cat before, you can read this text and probably decipher that the three things in the image are “cats”
- In a similar pattern, by seeing millions of image-text pairings of different objects, the model will be able to understand how certain phrases and words correspond to certain patterns in the images
- Once the model has this understanding, it can use the accumulated knowledge to extrapolation the other classes.



Zero-Shot Learning – CLIP Approach

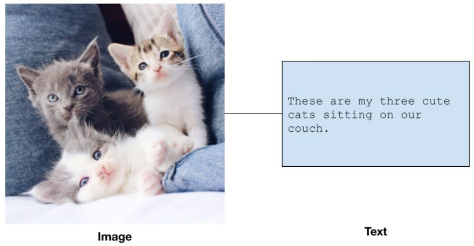


Zero-Shot Learning – Encoding

- Encodings are lower-dimensional representations of data
- For example, encodings for an image or a text should represent the most important, and distinguishable information of that image or text
- Note that encodings of similar objects are similar, while encodings of different objects are different.

	King	Queen	Princess	Boy
Royal	0,99	0,99	0,99	0,01
Male	0,99	0,02	0,01	0,98
Female	0,02	0,99	0,99	0,01
Age	0,7	0,6	0,1	0,2

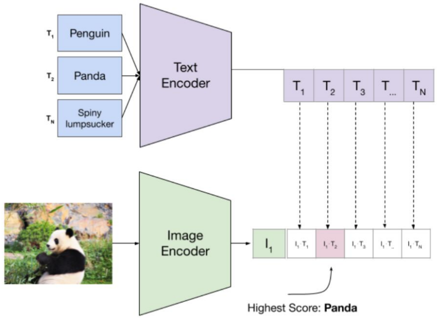
Zero-Shot Learning



- The idea behind this technique is to encode the image and text to make them as close as possible to each other
- Then the algorithm groups them into newly created classes never seen before!
- However, an important factor to mention here is that the model is learning how to create good labels for us from the images.
- This means that the output encoding of the images is our model output and should be matched with the text encoding as expected output

Zero-Shot Learning – Inference

- Once the model is trained on enough image-text pairings, it can be used for inference
- The inference stage of Zero-Shot Learning is set up as a typical classification task by first obtaining a list of all possible labels and then choosing the highest probability / most similar one
- Each label will then be encoded by the pretrained text encoder



What questions do you have?

Feedback on Lecture and Concepts?

See you on Thursday!