THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

AMA 2233

Data Analytics and Visualization

Yancheng Yuan

Department of Data Science and Artificial Intelligence

Recap of last lecture

Maximize Log Likelihood Function

Log Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, the log likelihood function is given by

$$l(w) = \log(L(w)) = mlog\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{1}{2\sigma^2}||y - Xw||^2$$

Linear regression: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, we find the best model parameter w by maximizing the log likelihood function, or equivalently, minimize $-l(w)$ (modify the constant):

$$w^* = argmin_w \; -l(w) = argmin_w \frac{1}{m}||y - Xw||^2$$

Binary Classification

Binary classification: We call it a binary classification problem if the variable y only takes values 0 or 1.

Model settings: For a given variable x, we assume the probability that y takes the value 1 is

$$P(y = 1 \mid x; w) = h_w(x) = \frac{1}{1 + e^{-w^T x}}$$

Remark: The probability that y takes the value 0 is

$$P(y = 0 \mid x; w) = 1 - P(y = 1 \mid x; w) = \frac{e^{-w^T x}}{1 + e^{-w^T x}}$$

Likelihood Function

Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, the likelihood function is given by

$$L(w) = \prod_{i=1}^{m} h_w(x^{(i)})^{y^{(i)}} \left(1 - h_w(x^{(i)})\right)^{1-y^{(i)}}$$

Logistic regression: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, we find the best model parameter w by maximizing the log likelihood function,

$$w^* = argmax_w \sum_{i=1}^{m} y^{(i)} \log\left(h_w(x^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h_w(x^{(i)})\right)$$

Gradient of $l(w)$

$$\frac{\partial}{\partial w_j} \mathrm{y}^{(i)} \log(h_w(x^{(i)})) = y^{(i)} x_j^{(i)} (1 - h_w(x^{(i)}))$$

$$\frac{\partial}{\partial w_j} (1 - \mathrm{y}^{(i)}) \log(1 - h_w(x^{(i)})) = -(1 - y^{(i)}) x_j^{(i)} h_w(x^{(i)})$$

$$\frac{\partial}{\partial w_j} l(w) = \sum_{i=1}^{m} \left(y^{(i)} - h_w(x^{(i)}) \right) x_j^{(i)}$$

$$\nabla l(w) = \left(\frac{\partial}{\partial w_0} l(w), \ldots, \frac{\partial}{\partial w_n} l(w) \right)^T$$

Gradient Ascent Algorithm for Logistic Regression

Start from some $w^0 \in \mathbb{R}^{n+1}$, gradient descent (GD) algorithm updates as:

$$w^{k+1} = w^k + \alpha_k \nabla\, l(w^k),$$

until

$$||\nabla\, l\left(w^{k+1}\right)|| \leq \varepsilon,$$

for some tolerance $\varepsilon > 0$.

Key points:

1. Compute $\boldsymbol{\nabla\, l\left(w^k\right)}$.
2. Choose step size $\alpha_k > 0$ satisfying

$$\boldsymbol{l\left(w^{k+1}\right) > l(w^k).}$$

Agenda



1. Multi-Class classification
2. Clustering
3. K-means
4. Details of the algorithm
5. Some Potential Challenges
6. Applications

Multi-Class Classification

Multi-Class Classification

Multi-Class classification: We consider the classification problem where the variable y takes values in $\{1, 2, \dots, k\}$.

Model settings: For a given variable x, we assume the probability that y takes the value j is

$$P(y = j \mid x; w_j) = h_{w_j}(x) = \frac{e^{w_j^T x}}{\sum_{l=1}^{k} e^{w_l^T x}}, j = 1, 2, \dots, k.$$

Useful Notation

Indicator function: Let A be a set, we define the indicator of the set A as

$$1_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

With the indicator function, we denote the label $\bar{y}$ equivalently as a k-dimensional vector:

$$y = (1_{\{1\}}(\bar{y}), 1_{\{2\}}(\bar{y}), \dots, 1_{\{k\}}(\bar{y})).$$

Likelihood Function

Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, where $y^{(i)}$ is a k-dimensional vector, the likelihood function is given by

$$L(W) = \prod_{i=1}^{m}(\prod_{j=1}^{k}(h_{w_j}(x^{(i)})^{y_j^{(i)}}))$$

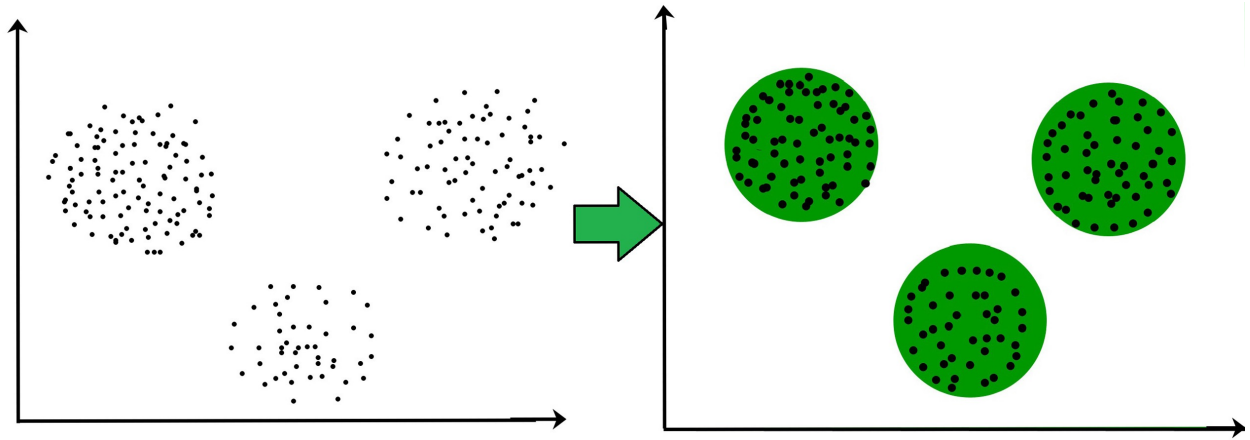
Softmax regression: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, we find the best model parameter w by maximizing the log likelihood function,

$$w^* = argmax_W \sum_{i=1}^{m}\sum_{j=1}^{k} y_j^{(i)} \log(h_{w_j}(x^{(i)}))$$

Clustering

Clustering

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more **similar** (in some sense) to each other than to those in **other groups** (clusters).



Clustering in Euclidean Space

Clustering in Euclidean space: Giving n data points $\{x_1, x_2, \dots, x_n\} \subseteq R^d$, our target is to partite the data points into K clusters based on reasonable similarity measure.

K-means model: Giving n data points $\{x_1, x_2, \dots, x_n\} \subseteq R^d$, K-means model solves the following

$$\min_{S_1 \cup \cdots \cup S_K} \sum_{l=1}^{K} \sum_{i \in S_l} ||x_i - \frac{1}{|S_l|} \sum_{j \in S_l} x_j||^2$$

How to find the solution? Lloyd's algorithm

Lioyd's algorithm

Input: n data points $\mathrm{X} = \{x_1, x_2, \dots, x_n\} \subseteq R^d$, number of clusters K.

Initialization: Randomly initialize K centroid $\mu_1^0, \dots, \mu_K^0$.

for $\boldsymbol{t = 0, 1, 2, \cdots}$

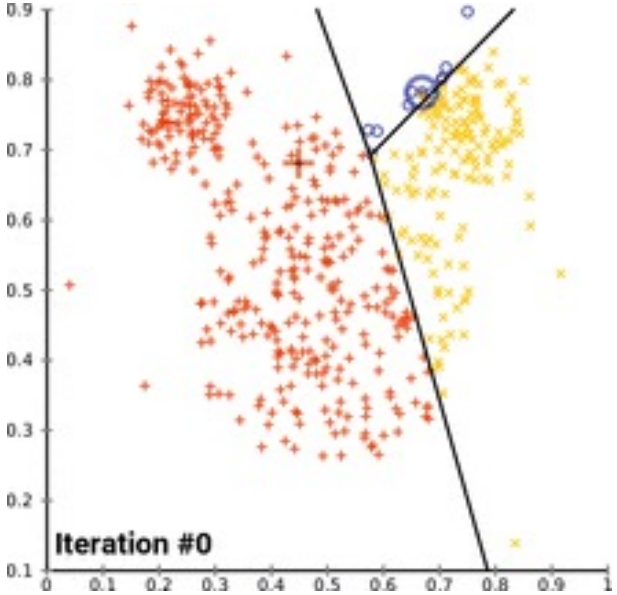
Step 1: Update Cluster Assignment:

$$S_l^t = \{i \in [n] \mid ||x_i - \mu_l^t||^2 \leq ||x_i - \mu_j^t||^2, \forall\, j, 1 \leq j \leq K\}.$$

Step 2: Update Centroids:

$$\mu_l^{t+1} = \frac{1}{|S_l^t|} \sum_{i \in S_l^t} x_i$$

An simple visualization



Advantages of K-means

1. Simple and intuitive.

2. Gives exact K clusters.

3. Highly scalable.

Computational Complexity Analysis (in terms of additions and multiplications):
Step 1: $O(knd)$
Step 2: $O(nd)$

Potential Issues for K-means

Issue 1: Need prior knowledge about K. This is not practical in many real applications.

A potential solution: Try different K and observe the clustering results.

Issue 2: Since the K-means model is nonconvex and it is NP hard to find the global solution, we usually get a suboptimal solution to the K-means model.

Issue 3: K-means method is sensitive to the initialization.

A potential solution: Run K-means algorithm with a good initialization.

Potential Issues for K-means





Potential Issues for K-means





Idea for a good Initialization

Key Idea: Initialize the centroids to spread as far as possible but close the given collection of data points.

How to realize it?

Randomly sampling the points from the given collection of data with probability proportional to the distance to the existing centroids.

K-Means++: Run K-means with a better initialization

K-means++ algorithm

Input: n data points $\mathrm{X} = \{x_1, x_2, \dots, x_n\} \subseteq R^d$, number of clusters K.

Initialization of centroids:

Step 1: Choose an initial centroid $\mathbf{C} = \{\boldsymbol{\mu}_\mathbf{1}^\mathbf{0}\}$ uniformly at random from **X**.

Step 2: Choose the next centroid $\boldsymbol{\mu}_\mathbf{i}^\mathbf{0} \in \boldsymbol{X}$ with probability

$$P(x_i) = \frac{D_C(x_i)^2}{\sum_{x_j \in X} D_C\left(x_j\right)^2}$$

and set $\mathbf{C} = \mathbf{C} \cup \{\boldsymbol{\mu}_\mathbf{1}^\mathbf{0}\}$.

Step 3: Repeat **Step 2 until** $|\mathbf{C}| = \mathbf{K}$.

Run K-means with the initialized centroids $\mathbf{C}$.

Applications

Color Quantization of a Color Image

Color Quantization: Performs a pixel-wise Vector Quantization (VQ) of an image of the summer palace (China), reducing the number of colors required to show the image from 96,615 unique colors to 64, while preserving the overall appearance quality.

Original image (96,615 colors)



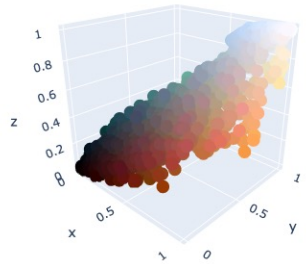
Quantized image (64 colors, K-Means)



More Results on Color Quantization

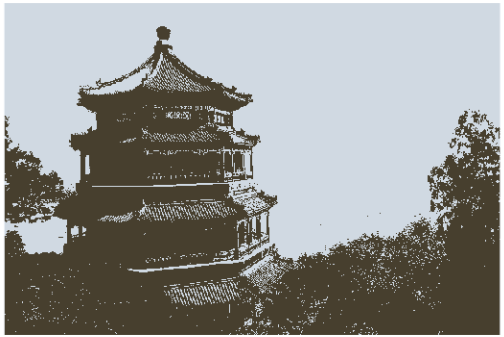
Original image (96,615 colors)

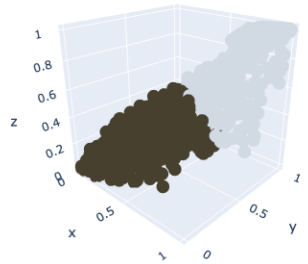




K=2 Play Pause K=4 K=8 K=16 K=32 K=64 K=128 Origin

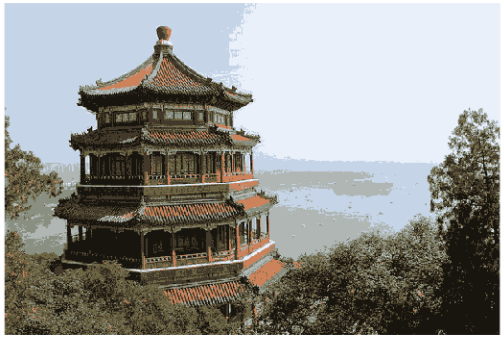
Quantized image (2 colors, K-Means)

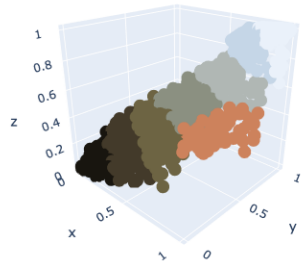




K=2 Play Pause K=4 K=8 K=16 K=32 K=64 K=128 Origin

Quantized image (8 colors, K-Means)



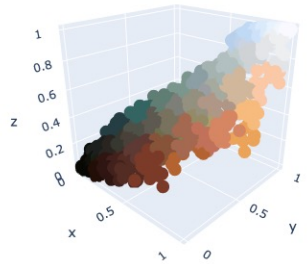


K=2 K=4 K=8 K=16 K=32 K=64 K=128 Origin

Play Pause

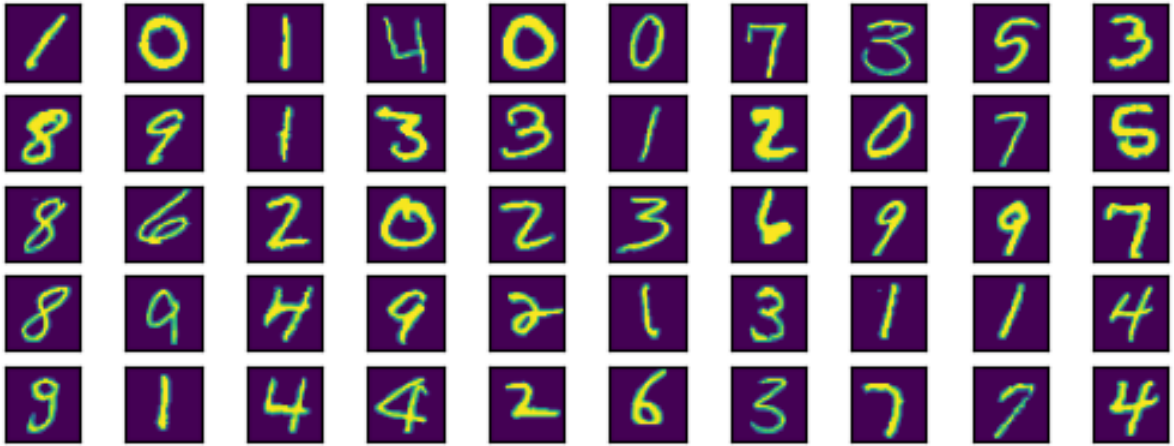
Quantized image (64 colors, K-Means)



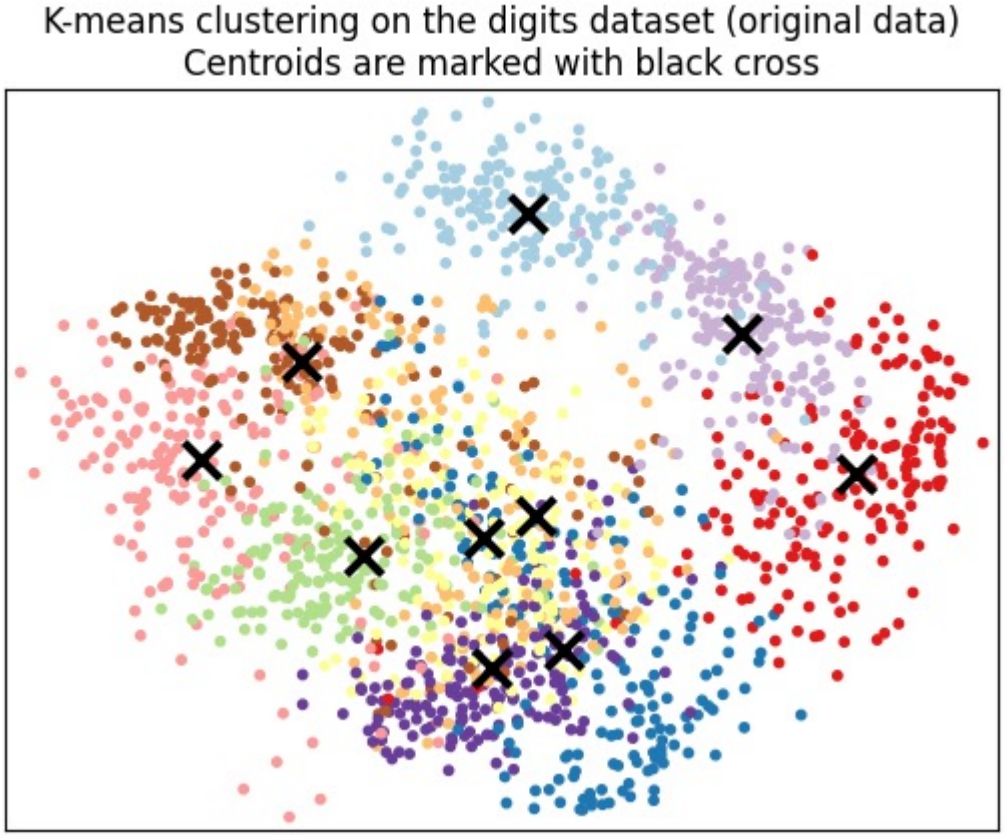


K=2 Play Pause K=4 K=8 K=16 K=32 K=64 K=128 Origin

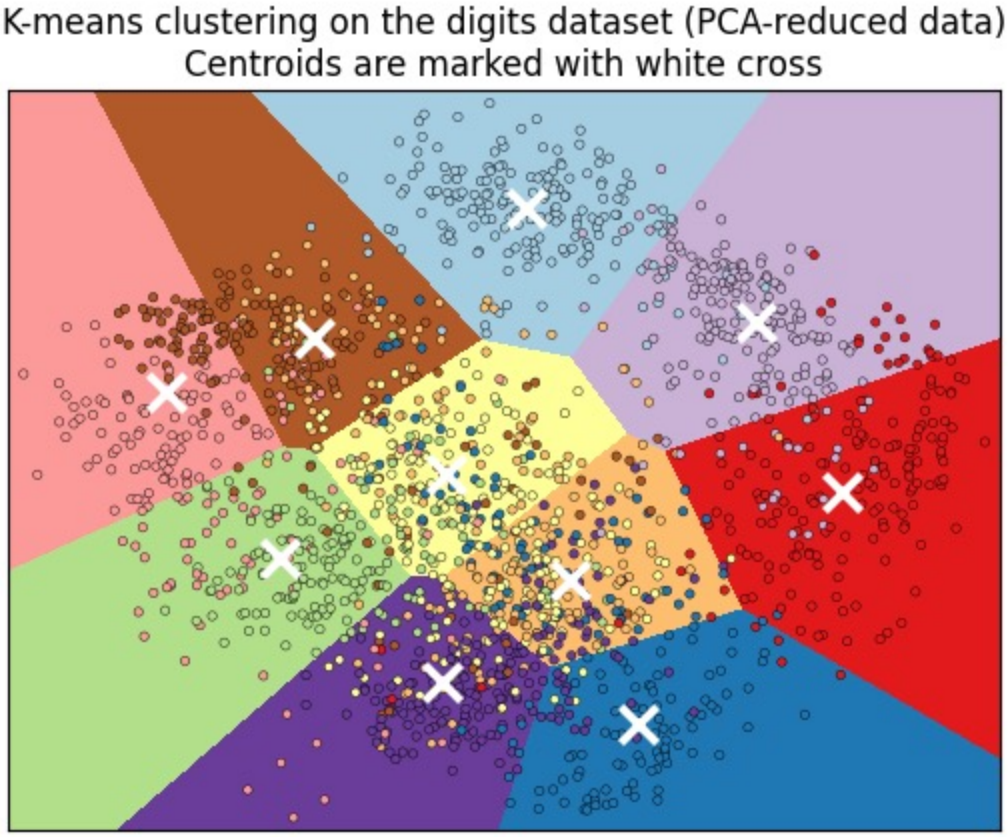
Clustering the MNIST Data



Kmeans on the original data with K = 10



Kmeans on the data after PCA with K = 10





More on K-means can be found at: https://en.wikipedia.org/wiki/K-means_clustering