Introduction to Data Science

22KDL

Lab04 - KMeans Clustering

Deadline: **23h59 - 19/05/2024**
TA contact info: hduc.lee@gmail.com

Objective: The objective of this assignment is to deepen your understanding of the K-Means clustering algorithm by implementing it from scratch using object-oriented programming (OOP) principles. You will create a custom KMeans class with methods such as fit, transform, and predict, following the API standards similar to Sklearn. Through this assignment, you will gain insights into the inner workings of the algorithm and its practical applications.

Dataset:
https://drive.google.com/file/d/1IqSv-q8bE3Fa5n93ZB7-Jza0DffC3TK4/view?usp=sharing

Description:
Statistics for a large number of US Colleges from the 1995 issue of US News and World Report, with 777 observations on the following 18 variables.

1. Private: A factor with levels No and Yes indicating private or public university
2. Apps: Number of applications received
3. Accept: Number of applications accepted
4. Enroll: Number of new students enrolled
5. Top10perc: Pct. new students from the top 10% of H.S. class
6. Top25perc: Pct. new students from the top 25% of H.S. class
7. F.Undergrad: Number of full-time undergraduates
8. P.Undergrad: Number of part-time undergraduates
9. Outstate: Out-of-state tuition
10. Room.Board: Room and board costs
11. Books: Estimated book costs
12. Personal: Estimated personal spending
13. PhD: Pct. of faculty with Ph.D.'s
14. Terminal: Pct. of faculty with a terminal degree
15. S.F.Ratio: Student/faculty ratio
16. perc.alumni: Pct. alumni who donate

Requirements: Complete the following tasks with the provided dataset.
Data Preprocessing:

- Perform necessary data preprocessing steps such as handling missing values, scaling numerical features, and encoding categorical variables if applicable.

Implement the KMeans clustering algorithm **from scratch** using **Numpy,** not using Sklearn (just for the Testing and Validation step):

- Define a Python class named KMeans such as
  - **Parameters**:
    - the number of clusters (K)
    - convergence tolerance
    - maximum number of iterations.
  - **Attributes**:
    - *cluster_centers_*: Coordinates of cluster centers. If the algorithm stops before fully converging (see tol and max_iter), these will not be consistent with labels_.
    - *labels_*: Labels of each point
    - *n_iter_*: Number of iterations run
    - *inertia_*: Sum of squared distances of samples to their closest cluster center
  - **Methods:**
    - **fit:** method to train the K-Means model on the input data.
    - **transform:** method to transform input data to a cluster-distance space.
    - **predict:** method to predict cluster labels for new data points.
    - **fit_transform:** method to compute clustering and transform X to cluster-distance space. (Equivalent to **fit(X).transform(X)**, but more efficiently implemented).
    - **fit_predict:** method to compute cluster centers and predict cluster index for each sample. (Convenience method; equivalent to calling fit(X) followed by predict(X)).

Testing and Validation:

- Test your custom KMeans class implementation on the provided dataset.
- Validate the correctness of your implementation by comparing results with sklearn's KMeans implementation.
- Experiment with different values of K (number of clusters) and choose an optimal value based on appropriate evaluation metrics (e.g., silhouette score, elbow method).
- Compare the results of your K-means and the Sklearn K-means with the *Private* attribute in the dataset (ground truth).

Analysis and Interpretation:

- Visualize the clusters using appropriate plots (e.g., scatter plots) to gain insights into the data distribution and cluster separability.

Encouragement for Experimentation:

- Encourage collaboration and discussion (BUT NOT **Plagiarism**) among students to share insights and learn from each other's approaches.

- Encourage students to leverage platforms like Google Colab/ Kaggle Notebook to experiment with the implementation. Google Colab provides free access to resources (CPU/GPU), facilitating faster experimentation.
- Encourage students to seek help from teaching assistants during lab sessions if they encounter difficulties.

Plagiarism Warning:

- Students are strictly prohibited from copying or reproducing the solution code from their peers. Each submission must be the individual work of the student. **Any instances of plagiarism or copying will result in a grade of 0 points for the assignment.**

Submission Guidelines:

- Jupyter Notebook containing:
  - Python code
  - Analysis
  - Visualizations
  - Report summarizing your findings and insights.
- Please send me your work before the due date.
- You can download the jupyter-notebook file (*.ipynb) by the following steps:
  - *File -> Download -> Download .ipynb*
- Name your notebook by the following pattern (same for Google Colab notebook's title): DS2024_Lab<LabID>_<StudentID>_<StudentName>.ipynb.
  - Example: DS2024_Lab01_21280075_NguyenVanA.ipynb
- The code results have to be printed out in the notebook.
- Include comments explaining key parts of the code if possible.
- Submit the notebook at: https://forms.gle/xTXYqxxTffTJYwMBA

There is **NO** acceptance for **cheating** or **copying**.