Project Report
DSAA2011 Machine Learning

Hua XU
HKUST(GZ)
hxu401@connect.hkust-gz.edu.cn

Jianhao RUAN
HKUST(GZ)
jruan189@connect.hkust-gz.edu.cn

Leyi WU
HKUST(GZ)
lwu398@connect.hkust-gz.edu.cn

1 Introduction

2 Mandatory tasks

The whole project can be decomposed into several stages, including data preprocessing, data visualization, clustering analysis, prediction and model selection.

2.1 Data preprocessing

- ✓the applied methods;
- observed patterns of dataset or insights

The Adult Income datasets is a binary classification dataset with 14 usable variables, six of which are integer types while the rest eight variables are categorical. In this project, we are using different strategy to process the data.

- As for the categorical variables, we firstly use the most frequent value to impute the missing values in the data, then we use one-hot encoding to convert the values into one-hot vectors. This step is implemented in Python with `OneHotEncoder` in scikit-learn.
- As for the numeric variables, we use median values of the variables to replace the missing values, and for 12455 we firstly normalize the data to 0 mean and 1 variance with `StandardScaler` in scikit-learn.

After imputing missing values and converting all features to numeric ones, we then get a wide dataframe with 123456 features, making the computation costs high and taking much time in training and testing. For efficiency and visualization purposes, we then use t-SNE to reduce the dimensionality of the original data and get both the 2D and 3D embedding for future usages.

justify why we should use t-sne in 3D: comparing the time and performance

put a table here to show the performance of t-SNE embedding

2.2 Data visualization

- t-SNE projection plot
- discussion of observed patterns or insights

explicitly show the mis assigned labels and try to make the 3D graph more readable

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

With 2D and 3D t-SNE embedding, we would like to visualize the dataset and explore more information within the dataset. For the 2D visualization, the first thing we notice is that samples from two different classes are not very distinguishable as they fall in the same clusters with different labels. As for samples with income less than 50K, they are spread over almost all the figure while the samples with income greater than 50K tend to cluster on only one side of the figure, which indicates that the class $\leq 50K$ is a more general, diverse class with samples from different distribution while for the class $> 50K$.

If we take a closer look at the labels inside, we may see that there are different sub clusters.

Moreover, this also indicates that most of the features in the original dataset may not be useful in terms of separating the samples apart and these features make the samples close to each other in the original feature space, and we may need to find the important ones.

Moreover, this also indicates that if we just directly use the t-SNE embedding to perform future tasks, we may fail since we cannot separate the samples based on the embedding good enough.

2.3 Clustering analysis

- description of the two chosen algorithms and why they are selected
- evaluation and interpretation of clustering results
- visualization of the clusters
- Comparison of algorithm performances

try to make all the cluseters scrollable

show the misclassified clusters

Methods

Based on the t-SNE visualization, we have already observed the clustering characteristic within the data and we can perform clustering analysis to gain more insights. Here we are using K-means and agglomerative hierarchical clustering.

K-means K-means is a classic clustering algorithm, which iteratively calculates the distance between sample points and the centroids of the cluster. The algorithm can be divided into **assignment step** and **update step**, as the assignment step assigns each data point x_i from the dataset X to the closest cluster centroid μ_i using the formula $r_i = \underset{k\in\{1\dots K\}}{\arg\min} \|x_i - \mu_k\|$. The centroids will then be updated during the update step, as each of the centroid of the cluster will be updated with the formula $\mu_k = \frac{\sum_{i:r_i=k} x_i}{\sum_{i:r_i=k} 1}$. In this method, the choose of the initial centroids is very important and therefore here we uses the **k-means++** initialization, which choose the first center randomly from the data points and for each remaining point, compute its squared distances to the nearest existing center. After that we choose another centroid with probability proportional to the squared distance. Then we repeat the process until K centers are chosen. This can reduce sensitivity to poor initialization, theoretically guarantees $O(\log K)$-competitive solutions. We implement this method in Python with `KMeans` in scikit-learn.

Agglomerative clustering Agglomerative clustering is a bottom-up approach which starts with each data point as a single cluster and merges the closest pairs of clusters iteratively. It will continue the process until one cluster remains. In this method, the way to measure the distance between the clusters will affect the performance of clustering methods, and we may have for different ways: 1) single linkage, which uses minimum pairwise distance; 2) complete linkage, which uses maximum pairwise distance; 3) average linkage, which uses average of all pairwise distances; 4) wardd's method, which aims at minimizing within-cluster sum of squares measured by the costs of merging cluster A and B: $\Delta(A,B) = \frac{2n_A n_b}{n_A+n_B}\|\mu_A - \mu_B\|^2$ where n_A, n_B means the number of elements in the corresponding clusters. We implement this method in Python with `AgglomerativeClustering` in scikit-learn.

For these two methods, k-means is the most classic clustering methods which can act as a baseline and since it is fast, it can quickly bring us more insights on the clustering analysis. And since we

have already see sub-clusters in the t-SNE analysis, we may try to use hierarchical clustering for more precisely capturing the locality of small sub-clusters.

Results

The clustering results is visualized in Fig and we have used some methods to evaluate the performance of the clustering.

add a table here for performance comparison

2.4 Prediction: training and testing

- description of the chosen classification target, model classes and why they were selected
- description of training process
- visualization of the results (decision boundary)
- evaluation process (confusion matrices)
- interpretation of prediction results on all the sets
- comparison of algorithm performances

logistic regression and decision tree classifier

Methods

Based on the t-SNE visualization we may see that it is very hard to find a continuous line or surface to separate the samples from the two different classes apart. Therefore, as for prediction, we would like to use decision tree based models and logistic regressions, as the decision tree based models may create discrete decision boundaries while logistic regression may bring us a curly decision boundary, making the classification precise.

Logistic regression Logistic regressions 123456.

Decision trees Decision trees 123456.

Results

After fitting the model with the train data, we then get the trained model.

2.5 Evaluation and choice of prediction model

3 Conclusion

References

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.

Credit

- **Leyi WU**:
- **Hua XU**:
- **Jianhao RUAN**:
- **AI**: