6103 Final Individual Report

Group 4
Jingya Gao

1. Introduction

Elo, one of the largest payment brands in Brazil, has built partnerships with merchants in order to offer promotions or discounts to cardholders. However, the questions are: Do these promotions work for either the consumer or the merchant? Do customers enjoy their experience? Do merchants see repeat business? Personalization is key.

In this project, we aggregate merchant.csv with the new_merchant_transactions.csv and historical_transactions.csv tables and then aggregate the concatenated table to the main train table. New features are built by successive grouping on card_id, in order to recover some information. We then developed six algorithms, including linear regression, decision tree, random forest, K-nearest neighbors (KNN), naive Bayes, K-means clustering, agglomerative nesting(AGNES), density-based spatial clustering of applications with noise (DBSCAN) and support vector machine (SVM) to predict the target: customer loyalty, in order to identify and serve the most relevant opportunities to individuals. Our goal is to improve customers' lives and help Elo reduce unwanted campaigns, to create the right experience for customers.

2. Individual Work

(1) K-nearest neighbors (KNN)

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data).

(2) K-Means Clustering

K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.

(3) Agglomerative clustering

Agglomerative clustering: It's also known as AGNES (Agglomerative Nesting). It works in a bottom-up manner. That is, each object is initially considered as a single-element cluster (leaf). At each step of the algorithm, the two clusters that are the most similar are combined into a new bigger cluster (nodes). This procedure is iterated until all points are member of just one single big cluster (root) (see figure below). The result is a tree which can be plotted as a dendrogram.

(4) Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a very popular density based data clustering algorithm commonly used in data mining and machine learning. DBSCAN clusters the data points to separate the areas of high density with areas of low density. It also marks data points as outliers that are in the low density regions. The clusters formed can be of varying shapes based on the density of data points.

(5) Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra, which is out of the scope of this introduction to SVM.

A powerful insight is that the linear SVM can be rephrased using the inner product of any two given observations, rather than the observations themselves. The inner product between two vectors is the sum of the multiplication of each pair of input values

(6) PyQt

PyQt is a set of Python v2 and v3 bindings for The Qt Company's Qt application framework and runs on all platforms supported by Qt including Windows, OS X, Linux, iOS and Android. PyQt5 supports Qt v5. PyQt4 supports Qt v4 and will build against Qt v5. The bindings are implemented as a set of Python modules and contain over 1,000 classes.

PyQt4 and Qt v4 are no longer supported and no new releases will be made. PyQt5 and Qt v5 are strongly recommended for all new development.

3. Models

1) K-nearest neighbors (KNN)
    - Import key packages:

```
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
```

    - Split the data:

```
X_train, X_test, y_train, y_test =
train_test_split(X1, y1, test_size=0.3, random_state=100, stratify=y1)
```

    - Standardize data:

```
stdsc = StandardScaler()
```

    - Use KNN classifier:

```
clf = KNeighborsClassifier(n_neighbors=K)
```

    - Plot confusion matrix and calculate the accuracy with different K

2) K-Means Clustering
    - Import key packages:
      from sklearn.cluster import KMeans
    - Choose the X and Y:
      X = data[["purchase_amount_count_std", "auth_purchase_month_std"]]

y = data['new_target']

- Use K-means classifier:
  estimator = KMeans(n_clusters=3)
- Set predict_label and plot the result:
  x0 = X[label_pred == 0]
  x1 = X[label_pred == 1]
  x2 = X[label_pred == 2]
  plt.scatter(x0.iloc[:, 0], x0.iloc[:, 1], c="red", marker='o', label='label0')
  plt.scatter(x1.iloc[:, 0], x1.iloc[:, 1], c="green", marker='*', label='label1')
  plt.scatter(x2.iloc[:, 0], x2.iloc[:, 1], c="blue", marker='+', label='label2')

3) AGNES
   - Import key packages:
     from sklearn.cluster import AgglomerativeClustering
   - Choose X and Y:
     X = data[["month_lag_std", "auth_purchase_month_std"]]
     y = data['new_target']
   - Use AGNES classifier:
     clustering = AgglomerativeClustering(linkage='ward', n_clusters=3)
   - Confusion matrix:
     print(confusion_matrix(data.y, clustering.labels_))
   - Set predict_label and plot the result:
     x0 = X[label_pred == 0]
     x1 = X[label_pred == 1]
     x2 = X[label_pred == 2]
     plt.scatter(x0.iloc[:, 0], x0.iloc[:, 1], c="red", marker='o', label='label0')
     plt.scatter(x1.iloc[:, 0], x1.iloc[:, 1], c="green", marker='*', label='label1')
     plt.scatter(x2.iloc[:, 0], x2.iloc[:, 1], c="blue", marker='+', label='label2')
4) DBSCAN
   - Import key packages:
     from sklearn.cluster import DBSCAN
   - Choose X and Y：
     X = data[["purchase_amount_count_std", "auth_purchase_month_std"]]
     y = data['new_target']
   - Use DBSCAN classifier:
     dbscan = DBSCAN(eps=0.4, min_samples=9)
   - Set predict_label and plot the result:
     x0 = X[label_pred == 0]
     x1 = X[label_pred == -1]
     x2 = X[label_pred == 1]
     plt.scatter(x0.iloc[:, 0], x0.iloc[:, 1], c="red", marker='o', label='label0')
     plt.scatter(x1.iloc[:, 0], x1.iloc[:, 1], c="green", marker='*', label='label-1')
     plt.scatter(x2.iloc[:, 0], x2.iloc[:, 1], c="blue", marker='+', label='label1')
5) Support Vector Machine (SVM)
   - Import key packages:

   from sklearn.svm import SVC

from sklearn.model_selection import train_test_split

- Choose X and Y:

```
X = data[["month_lag_std", "auth_purchase_month_std"]]
y = data['new_target']
```

- Split the data:

```
X_train,X_val,y_train,y_val =
train_test_split(X,y,test_size=0.3, random_state=13)
```

- Use SVC classifier:

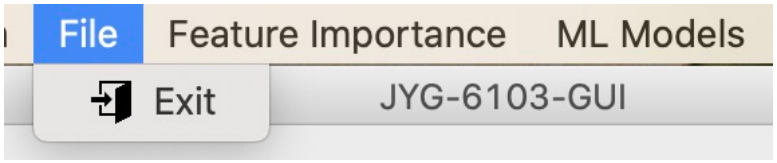
```
clf = SVC(C=6,kernel='rbf')
```

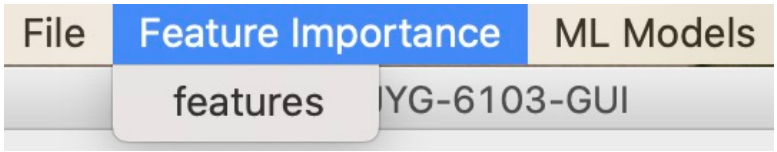
- Plot the result and show the accuracy
- Calculate the accuracy with samples

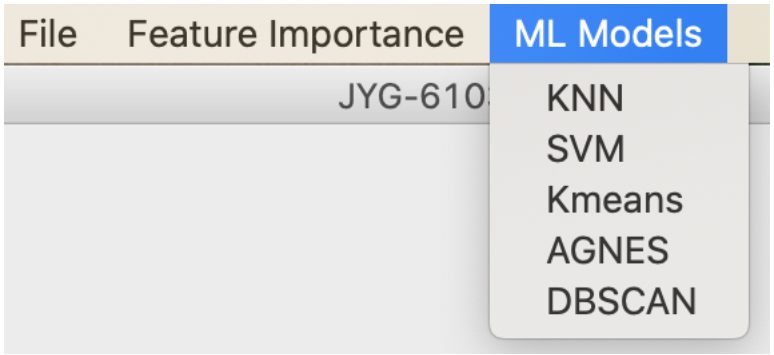
6) PyQt

- Set the menu bar (File; Feature Importance; ML Models):

- Link each method class to the menu:



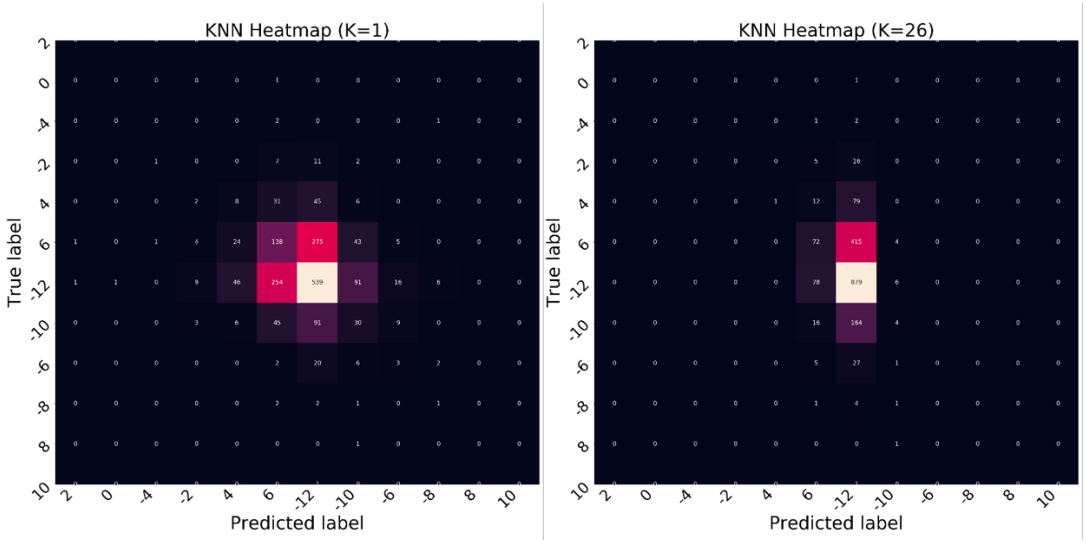




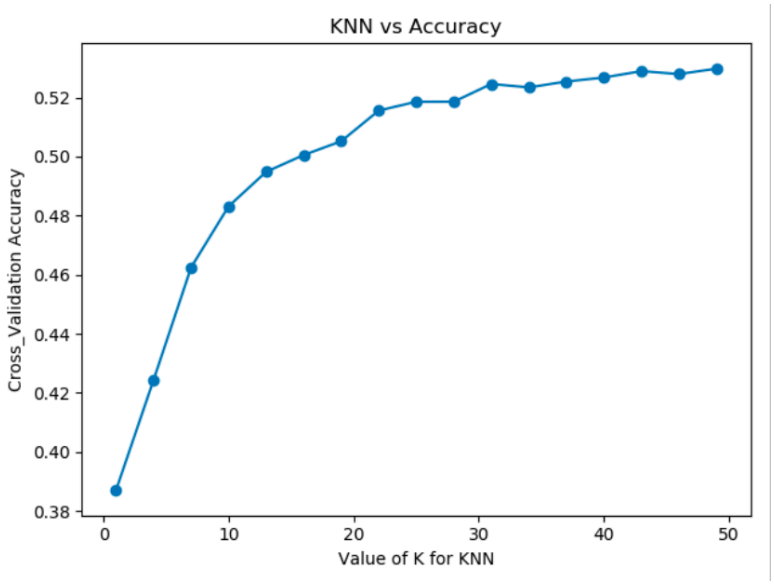
- Test the result of all methods

4. Results

(1) K-nearest neighbors (KNN)

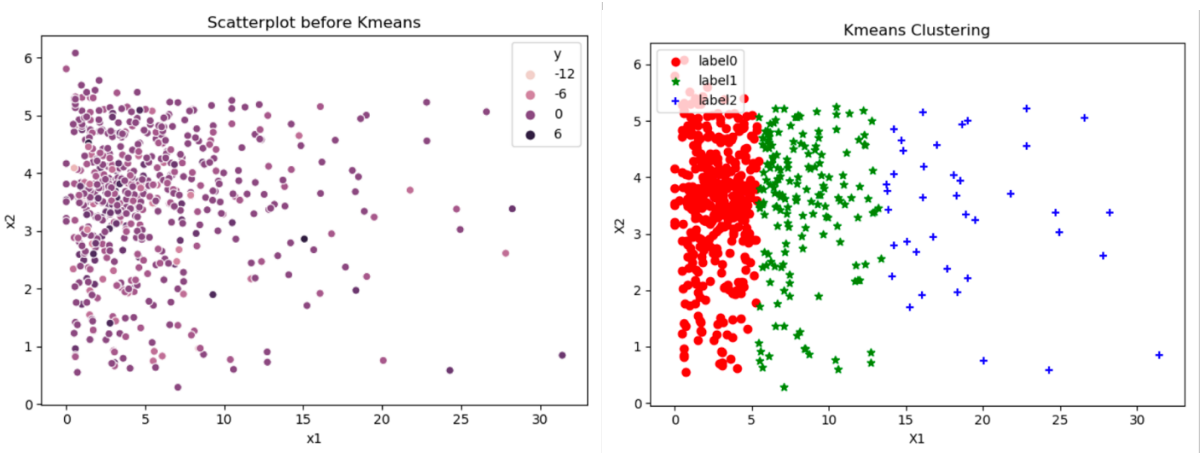


Accuracy : 40.01% (K=1); 53.20% (K=26)



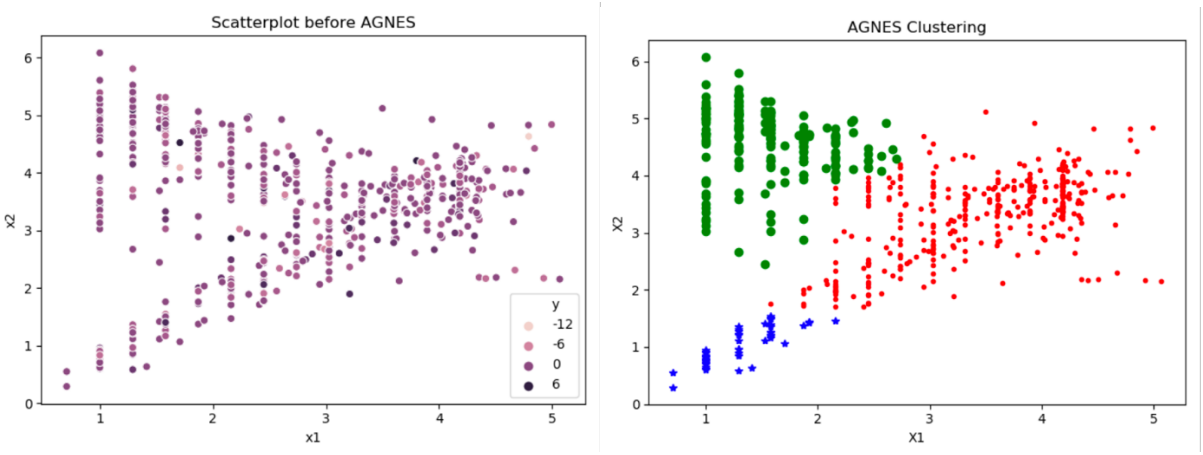
The chart above shows the relationship between K value and accuracy in K-nearest neighbors (KNN). K starts from 1, and the distance between each point is 3.

(2) K-Means Clustering



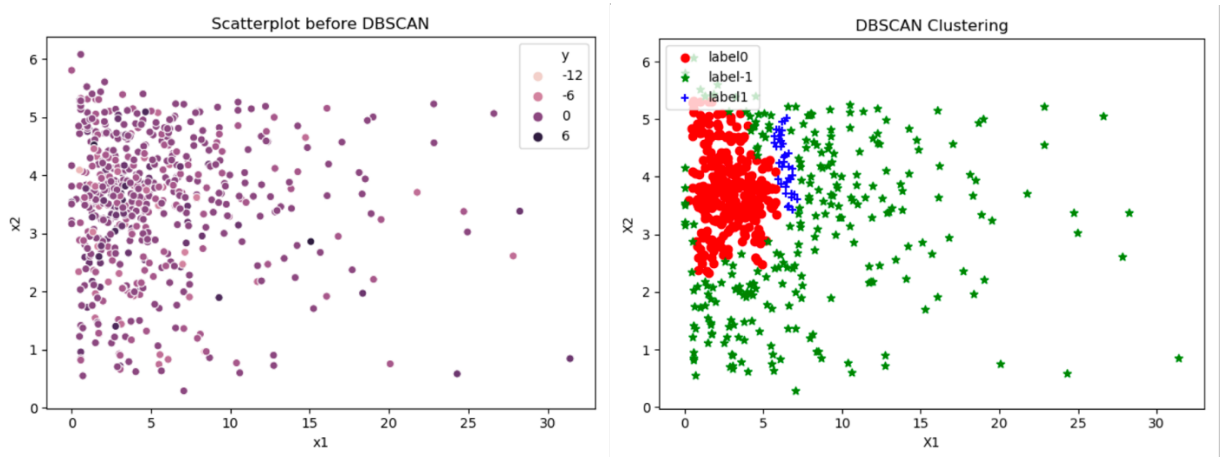
When features are selected as "purchase_amount_count_std" and "auth_purchase_month_std", scatter plots of raw data and result data after K-Means Clustering.

(3) AGNES



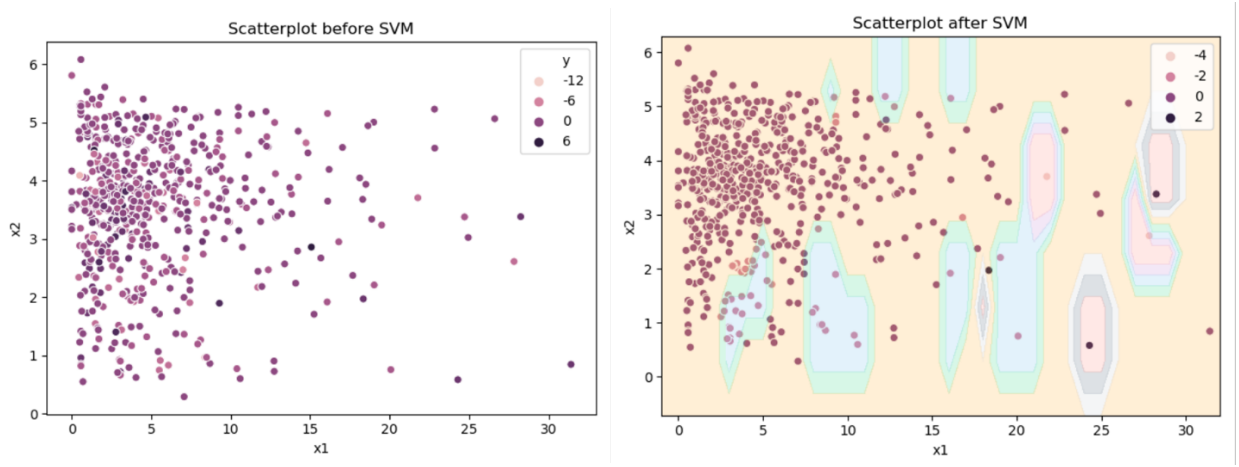
When features are selected as "month_lag_std" and "auth_purchase_month_std", scatter plots of raw data and result data after AGNES.

(4) DBSCAN

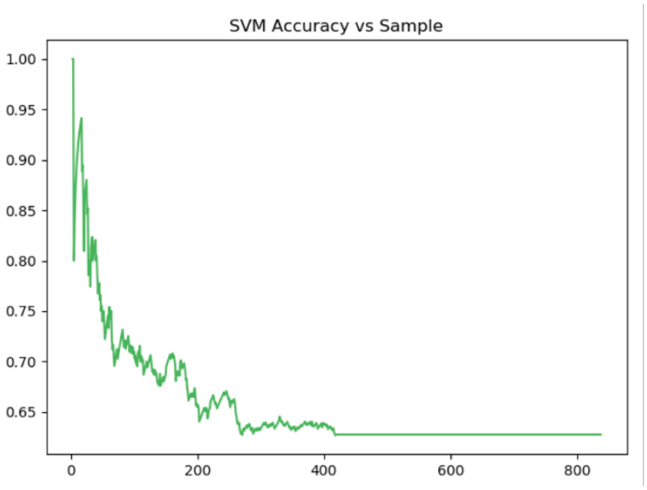


When features are selected as "purchase_amount_count_std" and "auth_purchase_month_std", scatter plots of raw data and result data after DBSCAN.

(5) SVM

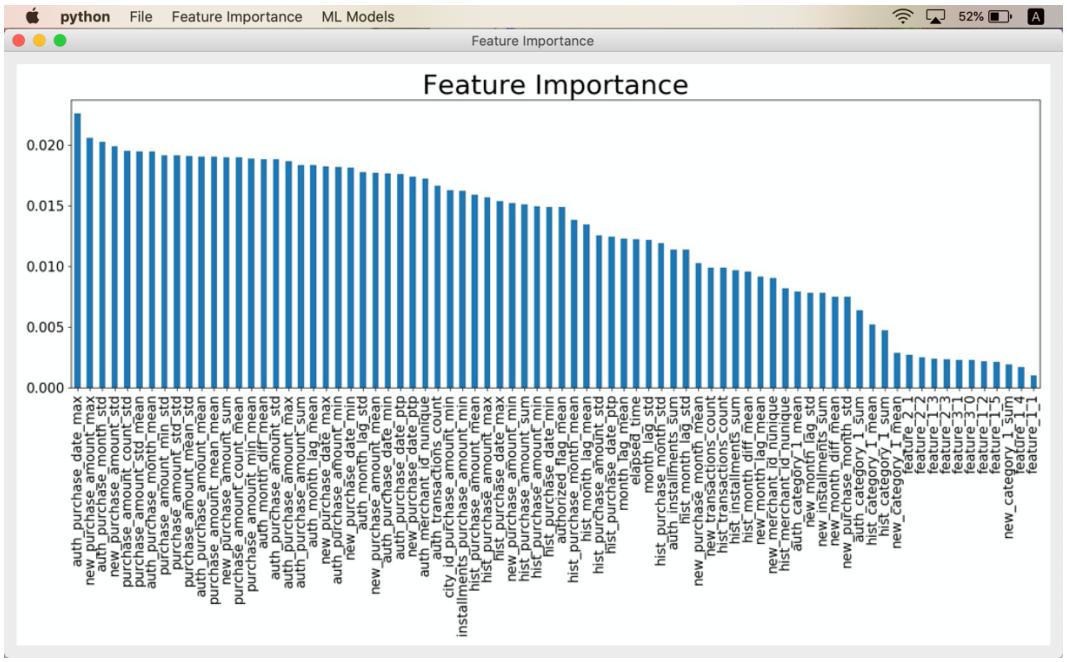


When features are selected as "purchase_amount_count_std" and "auth_purchase_month_std", scatter plots of raw data and result data after SVM.

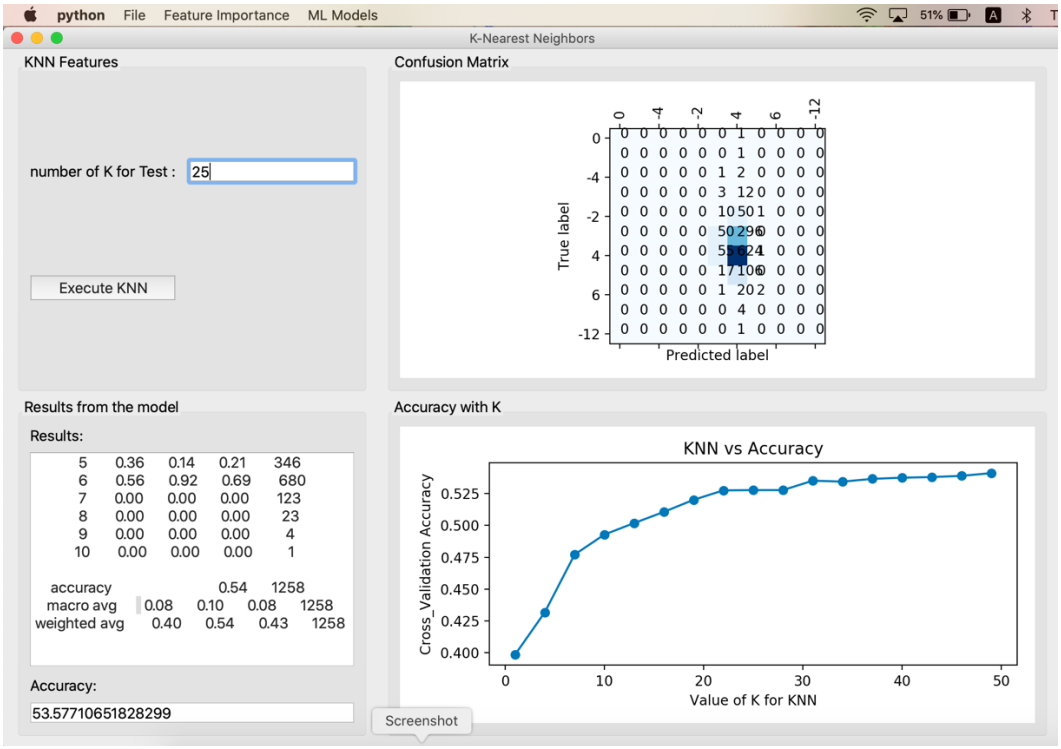


The chart shows the relationship between sample size and accuracy in Support Vector Machine (SVM). It can be seen from the curve that as the sample size increases, the accuracy of the SVM gradually stabilizes, and finally remains at about 62.5%.

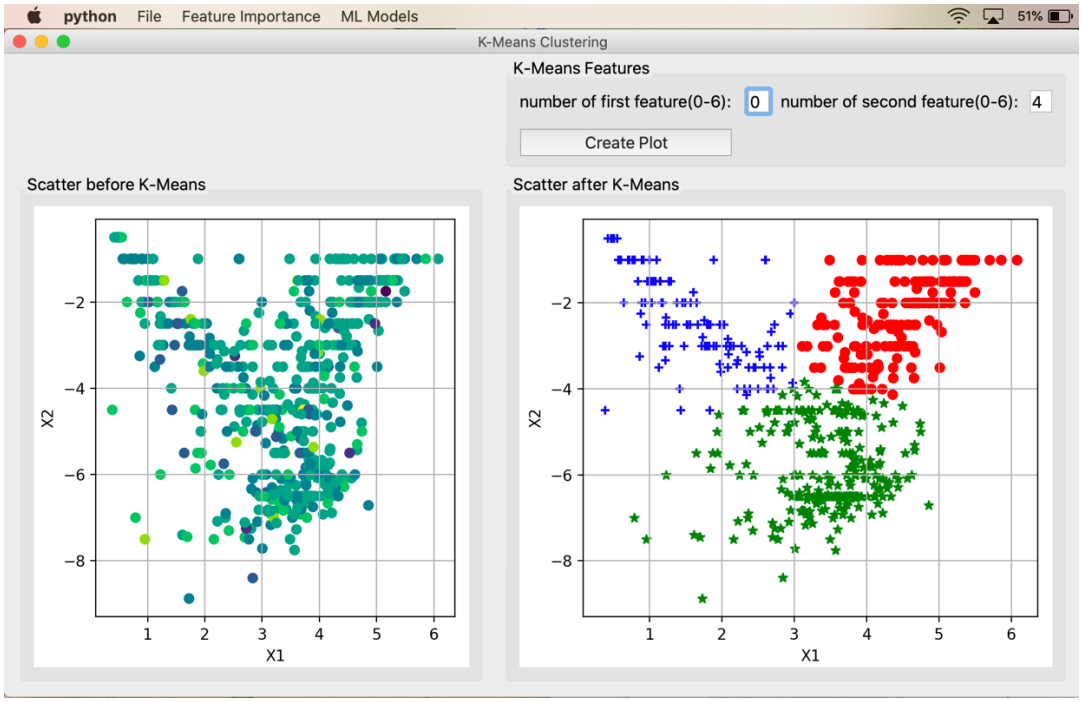
(6) PyQt



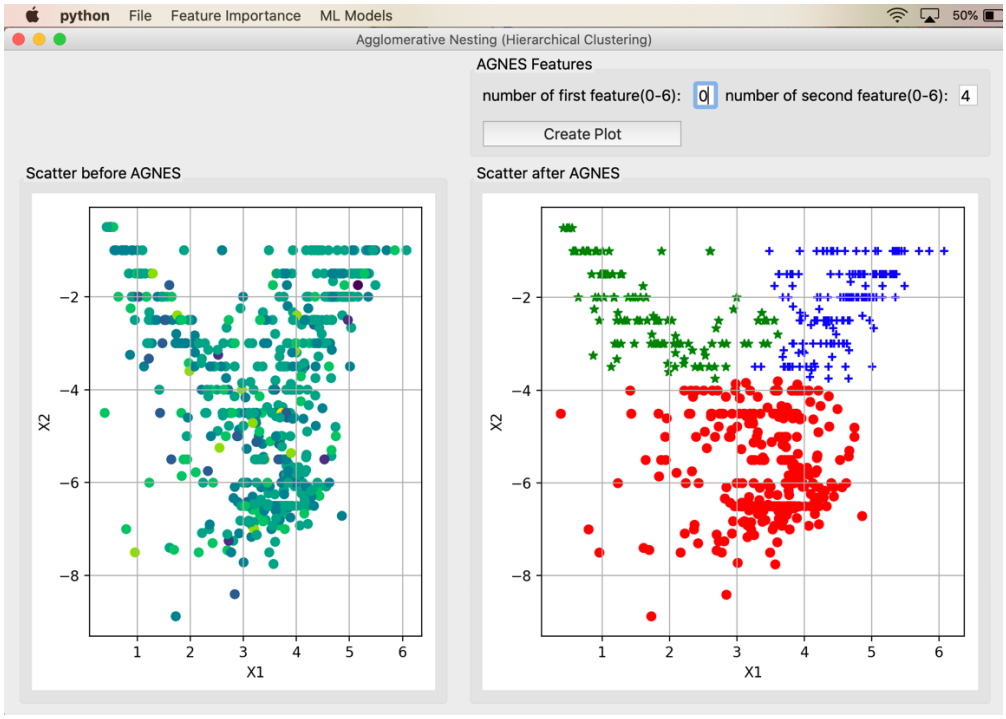
Result of “features” in “Feature Importance”.



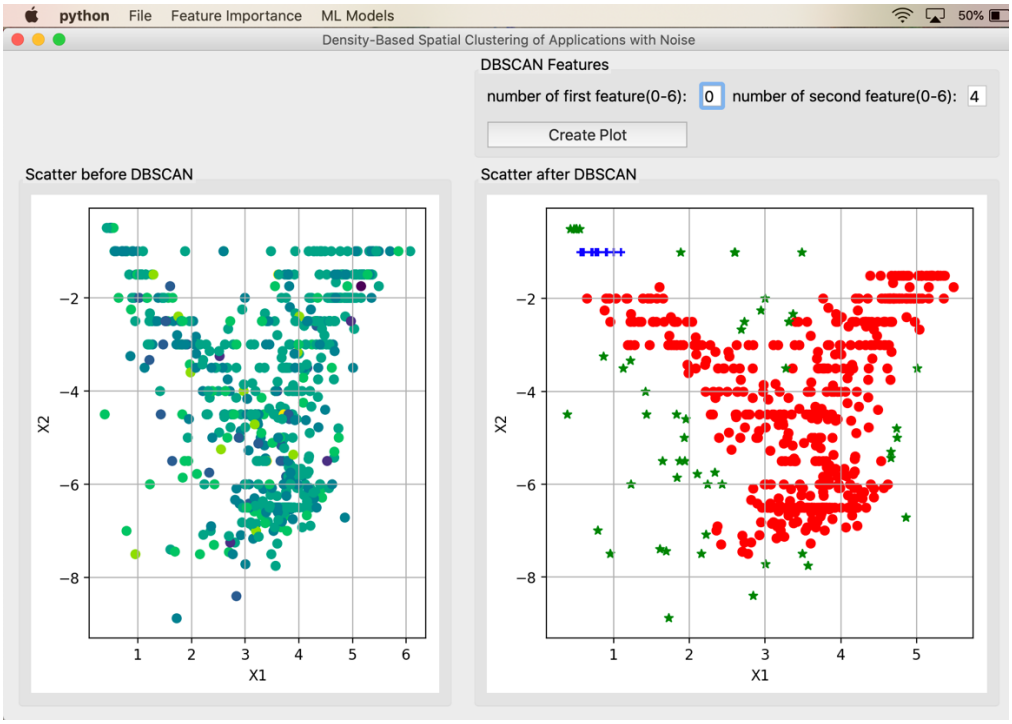
Result of “KNN” in “ML Models” when K=25.



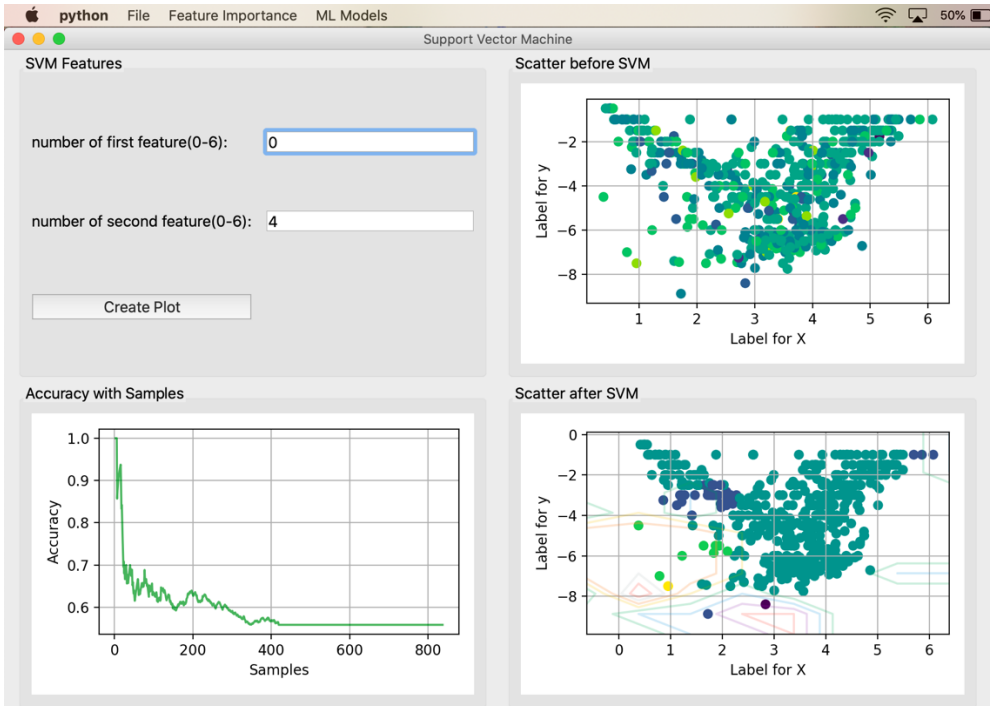
Result of “Kmeans” in “ML Models” when num[0] and num[4] features are chosen.



Result of “AGNES” in “ML Models” when num[0] and num[4] features are chosen.



Result of “DBSCAN” in “ML Models” when num[0] and num[4] features are chosen.



Result of "SVM" in "ML Models" when num[0] and num[4] features are chosen.

5. Summary and Conclusion

From the report results, we can see that the five models: K-nearest neighbors (KNN), k-means clustering, agglomerative nesting (AGNES), and density-based spatial clustering of applications with noise (DBSCAN) and support vector machine (SVM) can all run successfully.

Among them, it can be found in the use of the KNN model that when the target distance is set to 2, as the value of K increases, the accuracy of the model continues to improve, because the value of K changes to divide more data. Into a category. Similarly, we can increase the accuracy by increasing the spacing value, but this will reduce the accuracy of predicting some special values. For example, when the spacing is 5, only the probability of [0, 5] can be roughly predicted. Data such as 2.5 cannot be accurately predicted.

When the variables are selected the same, the results of the two methods of K-means and AGNES are roughly the same, but the results of DBSCAN and the former are somewhat different.

K-means is a simple and efficient way for large data sets, with low time and space complexity. The most important thing is that the results are easy to be locally optimal when the data set is large; K values need to be set in advance, which is sensitive to the selection of the first K points; very sensitive to noise and outliers; only used for numerical data; cannot solve non- Convex data.

DBSCAN is not sensitive to noise and it can find clusters of any shape. However, the results of clustering have a great relationship with parameters. DBSCAN uses fixed parameters to identify clusters, but when the degree of sparseness of the clusters is different, the same judgment criteria

may destroy the natural structure of the clusters, that is, the thinner clusters Will be divided into multiple classes or denser and closer classes will be merged into a cluster.

It can be seen from the graph of the SVM test that as the number of samples continues to increase, the accuracy of the model will gradually decrease, and will stabilize when the number of samples reaches a certain number.

Since SVM uses the quadratic programming to solve the support vector, solving the quadratic programming will involve the calculation of a matrix of order m (m is the number of samples). When the number of m is large, the storage and calculation of the matrix will consume a lot of machines. Memory and operation time. In view of the above problems, in the future research, the following methods can be used to improve the model: J.Platt's SMO algorithm, T. Joachims 'SVM, C.J.C.Burges' PCGC, Zhang Xuegong's CSVM, and O.L. Mangasarian's SOR algorithm.

6. Code Calculation

JYG-GUI: 23.7%

JYG-KNN: 20.2%

JYG-Kmeans: 17.3%

JYG-AGNES: 15.8%

JYG-DBSCAN: 11.5%

JYG-SVM: 21.1%

JYG-feature_importance: 9.2%

7. References

GARCIA, S. A. L. V. A. D. O. R. (2016). Data Preprocessing In Data Mining. Place of publication not identified: SPRINGER INTERNATIONAL PU.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). An introduction to statistical learning: with applications in R. New York: Springer.

Hastie, T., Friedman, J., & Tisbshirani, R. (2017). The Elements of statistical learning: data mining, inference, and prediction. New York: Springer.

Elo Merchant Category Recommendation Help understand customer loyalty. Elo. (March, 2019). Retrieved from https://www.kaggle.com/c/elo-merchant-category-recommendation

8. Github Link

https://github.com/xinyuyao22/Final-Project-Group4/tree/master/JingyaGao-individual-project