Department of Computer Engineering
University of Peradeniya

CO 544 Machine Learning and Data Mining

Lab 05

08^{th} of June 2020

Objectives: To provide students hands-on experience in clustering and Association Rule Learning using Python. At the end of the lab, students should be able to,

- Use the K-Means algorithm for a given dataset.
- Analyze the output of K-Means algorithm and interpret the results.
- Use the Apriori algorithm to discover association rules.
- Analyze the output of Apriori algorithm and interpret the results.

1. Clustering

1.1 Importing required modules.

```
from sklearn.cluster import KMeans
from sklearn.datasets.samples_generator import make_blobs   #to generate sample datasets
```

1.2 creating a sample dataset with 4 clusters

```
X, y = make_blobs(n_samples=400, centers=4, cluster_std=0.90, random_state=1)
```

1.3 Determining the optimum value of **k** using **Elbow method**.

```
wcss = []    #within cluster sum of squares
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

1.4 Applying K-Means algorithm.

```
kmeans = KMeans(n_clusters=4, random_state=0)  #from Elbow method we identified n_clusters=4
closest_cluster_index = kmeans.fit_predict(X)
cluster_centers=kmeans.cluster_centers_
```

1.5 Visualizing

```
plt.scatter(X[:,0], X[:,1],c='green')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s=200,c='black',marker='*')
plt.show()
```

Excercise 01

(a) Import 'iris' dataset from the standard Scikit-learn datasets. The dataset is intended to use for classification. Make the dataset into an unlabeled dataset by removing the class attribute.

(b) Use Elbow method to identify the best **k** which minimizes the Within-Cluster-Sum-of-Squared(inertia).

(c) Fit the k-Means algorithm with the k value identified from part(b).

(d) Explain the results of following code snippet.

```
kmeans.cluster_centers_
```

; where kmeans is the KMeans object

(e) Visualize the data points and cluster centers in a 3D plot where first three variables of the dataset are the axes.

2. Association Rule Learning

1.1 Install Apriori.

```
pip install mlxtend  # by installing mlxtend library
```

1.2 Importing Required modules.

```
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
from mlxtend.preprocessing import TransactionEncoder
```

1.3 Input data

```
dataset = [['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
           ['Milk', 'Unicorn', 'Corn', 'Kidney Beans', 'Yogurt'],
           ['Corn', 'Onion', 'Onion', 'Kidney Beans', 'Ice cream', 'Eggs']]
```

1.4 Creating the dataframe of frequent itemsets.

```
te = TransactionEncoder()
te_ary = te1.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te1.columns_)
```

1.5 Applying Apriori algorithm and finding Association rules.

```
sup = apriori(df, min_support=0.002, use_colnames=True)  #itemsets which minimum support is greate
rules= association_rules(freq, metric="lift", min_threshold=1)//
```

$$Lift(\{X\} \rightarrow \{Y\}) = \frac{(Transactions\ containing\ both\ X\ and\ Y)/(Transactions\ containing\ X)}{Fraction\ of\ transactions\ containing\ Y}$$

Note:

Excercise 02

(a) Import the given dataset 'groceries.csv'

(b) Explore the dataset and create the frequent item dataset.

(c) Apply Apriori algorithm and identify the itemsets where the support is greater than 10%.

(d) Find the set of Association rules using the metric 'lift'

(e) Select any Association rule from the set and describe the selected rule.

(f) How many rules are there when the 'lift' is greater than 4 and the 'confidence' is greater than 0.8.

3. Submission

Submit two Python files with source codes for each excercise. You can answer the questions as comments in the Python file. Put both files in a folder, compress it and name it as **e15xxxlab5** where **xxx** is your registration number.