514-Assignment-3

Muhammad Somaan

May 2024

1 Association Rule Mining

I have applied Association Rule Mining using Apriori algorithm with minimum support and confidence of 75

1.1 Frequent Itemsets

In total, 511 frequent itemsets were identified. The first 9 frequent itemsets identified is as follows.
{{'G'}, {'C'}, {'I'}, {'E'}, {'B'}, {'F'}, {'H'}, {'J'}, {'D'}}
From this, we can infer that the users do not look at segments A, K, L, M, N, O, R. This shows us that people usually don't look at the bottom of the page nor look at the search box. The last itemset is as follow.
{{'H', 'C', 'I', 'E', 'G', 'J', 'B', 'D', 'F'}}
This shows us that the users often look at the segments B to J together.

1.2 Support and Confidence

The support and confidence value are chosen high enough to find interesting rules and frequent itemsets.
Support is the percentage of how likely 2 items are to be seen together. Changing support value to a low value will make segements that are looked together by chance appear in our frequent itemsets, which will not be interesting. This increases the amount of patterns found, but many may be uninteresting.
Confidence is the percentage of how likely is another item to be found when a previous itemset is present in a transaction. Changing the confidence to a lower value, will keep the patterns that has a low chance of appearing together, so the amount of uninteresting rules will increase.

1.3 Code

The code utilizes the apriori library from here[1]. First of all the data in read from the file and then it is cleaned to seperate the transactions and remove any duplicates and their timings. After this the apriori algorithm is applied to find the frequent itemsets which are then printed in the end.

2 DBSCAN

DBSCAN is applied on the webpage data, by taking the X and Y point of the webpage that the people looked at.

2.1 Working

The data for each user is first input and then combined together in one dataframe of pandas. Since we are only interested in the location that each user looked at, we only take the MappedFixationPointX and MappedFixationPointY, and the rest of the data is ignored.

2.2 Code

Firstly all the files in the folder are opened and data from them is read and appended to a single dataframe, taking care that the file is not empty, since P-38.txt is an empty file.

```
for file in txt_files:
    if os.stat(file).st_size == 0:
        print("Empty file:", file)
        continue
    print('Location:', file)
    print('File Name:', file.split("\\")[-1])
    data = pd.read_csv(file, sep="\t")
    all_data_frames.append(data)
print("Total files:", len(all_data_frames))
complete_data = pd.concat(all_data_frames)

X_train = complete_data[["MappedFixationPointX", "MappedFixationPointY"]]
X = np.array(X_train)
print(X)
```

Figure 1: Data Input from Files

Following this an array for eps_values and min_samples is created for possible values for the scan, and then the result is plotted on the webpage page.

```
eps_values = [10, 12.5, 15, 20, 25, 30]
min_samples = [5, 6, 7, 8, 9, 10, 11, 12, 15, 18, 20, 25]
```

Figure 2: Possible eps and min sample values

2.3 Results

The best possible EPS value and min sample values found are 30 and 25, accordingly. Mapping the clusters onto the image provided clearly shows us the distinction between different segments on the webpage, that the users looked at.

Figure 3: DBSCAN Result

It can clearly be seen that segment E is properly clustered, but the interesting thing to note is that segment B is divided into multiple smaller clusters; these clusters form on top of the different text or button sections. This shows us that even though users used segment B, different users used different parts of the B segment. Another interesting observation is that almost none of the users tried to click on the store section of the segment.

3 Acknowledgements

References

[1] Chonyy. (n.d.). GitHub - chonyy/apriori_python: Python implementation of Apriori algorithm, new and simple! GitHub. `https://github.com/chonyy/apriori_python/tree/master`