A DATA DRIVEN RECOMMENDER FRAMEWORK

By

SHUBHANK VARSHNEY

A thesis submitted to the

Graduate School – New Brunswick

Rutgers, The state University of New Jersey

In partial fulfillments of the requirements

For the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Prof. Ivan Marsic

and approved by

_____

_____

_____

New Brunswick, New jersey

October, 2017

ABSTRACT OF THE THESIS

A DATA DRIVEN RECOMMENDER FRAMEWORK

By SHUBHANK VARSHNEY

Thesis Director: Prof. Ivan Marsic

Process mining has been receiving a large amount of scrutiny by researches and industry personnel alike in the recent past. The process logs and traces left by business process can be a big source of information and knowledge about the behavioral aspects of the process. Process mining techniques help extract the knowledge from the traces and logs. This knowledge based data coupled with process mining can used to design recommendations for a user. Most of the existing recommender systems have not been developed based on process mining algorithms and hence provide a whole new scope for research and improvement in their development. Therefore, in this thesis we propose a novel data driven recommender model that can provide recommendations of sequential procedural steps, visualizations and diagnostics for a specific group of patients to provide a feasible and more accurate solution to the problem.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Process mining is a discipline of data science that provides comprehensive tools to obtain meaningful fact-based insights and thereby act as a means to improve the process. The main driver for this technology is the fact that more and more event logs are being recorded by business processes. These logs consist of activities with plethora of characteristics which are worth exploring. There exist many tools that record activity logs and provide recommendations based on those [1]. These tools often provide solutions that cater to small traces of data and with limited comprehensiveness. One of the major limitation that exists in these tools is their inability to use similar historic process performances and contextual information to determine the established procedures. Our work, in this document focuses on solving and eliminating these limitations and aims at providing a comprehensive and robust recommender system to the users. We aim to propose data-driven process analysis and recommender system that can not only provide the contemporary recommendations of process steps but also help with the retrospective analysis. It relies on using historical data to mine the associations between the recommended performance steps and contextual attributes.

Mining sequential patterns of interest has always been a task of challenge and interest to researchers. Health care institutions in industry and academia often need to study and analyze the steps taken in a procedure for a certain group of patients. The analysis can be challenging when the list of steps in a procedure are one too many. This calls for the need to provide a mechanism to the institutions for an easy identification and association of a patient to a set of procedures based on the information obtained from previous such patients. Our aim is to explore this problem in detail and provide a feasible and viable solution to the users. A detailed study and explanation about the procedure can be found in [1]. The current work is an extension of the previous work by adding closed continuous sequence mining algorithm. We have taken a data of 122 patients in collaboration with Children's National Medical Center, Washington D.C as a case study for this work.

1.1 THESIS GOALS

As discussed earlier, the current recommender models don't take into account the historical process performance and contextual information to determine the established procedures. The current work aims at addressing these drawbacks by 1) clustering the patients into groups based on similarity in their attributes/features and 2) the introduction of sequential pattern mining techniques and conducting tests to associate sequential procedures to a group of patients. A number of tests and analysis is conducted to prove the validity of our results in the work.

The framework has been developed in Python language by making use of its versatile and powerful set of libraries.

CHAPTER 2

RELATED WORK

A large amount of data regarding a patient's state of trauma and the steps taken to remedy that is available. This data is often a great source of knowledge which can be put to use in the creation of a robust model for a workflow procedure. Electronic Health Record(EHR) and Personal Health Record(PHR) systems provide a means to store such data and allows heath care institutions to share it electronically. However, such a system can be a real asset only when it can do more than just store data in a structured format. Patients can benefit more if they can get to make informed decisions based on the information obtained from the stored data.

A lot of research has been done in the recent past to develop such a framework. The motive behind most of it provide a retrospective analysis of the process. For example, Clark et.al [2] and Fitzgerald et.al [3] developed a computer aided system to recommend next steps. However, their system relies heavily on rules specified by experts and hence is prone to human bias. Our work on the other hand is automatic and data-driven and incorporates trace clustering and prototyping and mining significant sequence of steps for each cluster by the use of state of art Gap-BIDE [4] sequence mining algorithm.

The first sequential pattern mining algorithm was Generalized Sequential Pattern(GSP) [5] and was based on the famous Apriori property. Since then many sequential pattern mining algorithms have been proposed for the sake of performance improvement. The Clospan algorithm [6] was developed to mine closed sequential patterns. It mines a candidate set of closed patterns based on PrefixSpan algorithm [7] and keeps that candidate set for post pruning. The gap-BIDE algorithm [4] which doesn't need to keep a candidate set of closed sequences is also based on the framework of PrefixSpan algorithm. It prunes the candidate set as soon as it is found. There are several other algorithms that focus on mining gap based sequential patterns. The TEIRESIAS [8] algorithm mines genomic sequences with fixed wildcards in between. The MPPm [9] algorithm aims at mining frequent gap constrained sequential patterns in a single sequence. All these algorithms,

however, suffer from large space overhead when the dataset is large as they need to keep the position of every item in all of its appearances.

K-means [10] clustering is the simplest unsupervised learning algorithm that can very well solve the clustering problem. This algorithm, however, suffers badly when the number of clusters are not known in advance and when there is large amount of categorical data present. Hierarchical clustering algorithm, on the other hand has been commonly used for clustering process traces [11]. Its performance is stable even in case of categorical features. It doesn't require a predefined number of clusters and produces an intuitive dendograph. Unlike Hierarchical and K-means Clustering, the Affinity Propagation Clustering(APC) [12] algorithm doesn't require the user to input the number of clusters and relieves us of the task of selecting the number of clusters. We have implemented APC Algorithm and Hierarchical clustering along with K-means in our work.

CHAPTER 3

FRAMEWORK DESIGN OVERVIEW

Finding relevant information from large chunks of data and presenting it in understandable form for a user can be a challenging task. People seek for such information primarily on Internet and other web-based search engines. However, these media usually present a barrier in the following departments:

- Find the source and determining its source of relevance.
- Ease of understanding the information found from the source.
- Awareness of the context.

Our recommender framework is meant to mitigate these drawbacks and provide high quality contextual information. Unlike most of the recommender systems that propose one or two steps at a time, our model proposes all the steps at once. Although, it may not be feasible for the user to study and follow the list of steps all at once, the recommendation can be used at runtime to study the process compliance and detect any steps or procedures that may have been omitted. Figure 3.1 illustrates the framework and the steps involved in obtaining the recommendations. The framework has two stages: process analysis and then recommendation. Process analysis includes: clustering of traces, determining the cluster prototypes that represent the established procedure for each cluster, finding the correlation between the prototypes and the mined attributes, visualizing the information. Process recommendation is composed of predicting the cluster to which given trace belongs based on observed attributes of the context and displaying the prototype sequential patterns of the cluster as the recommended procedure.

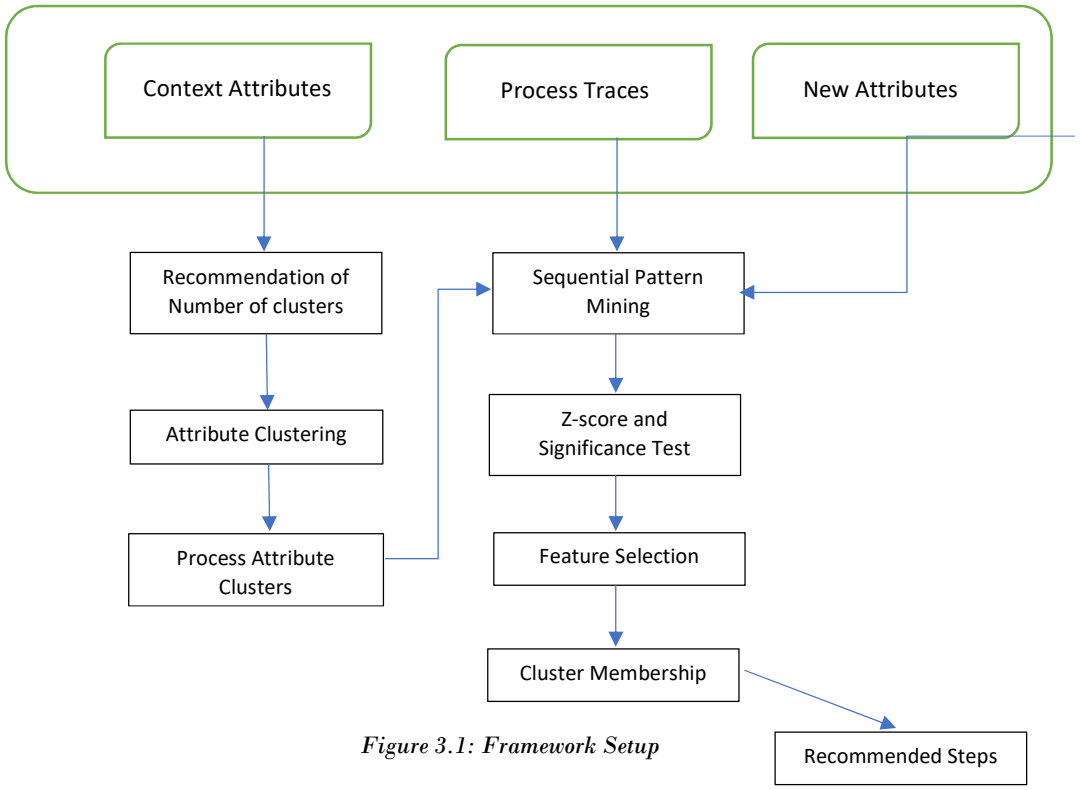


Figure 3.1: Framework Setup

As can be seen in Figure 3.2(a), we have captured the behavior of a trace with the help of activity type and the timestamp comprising the start and end time of the activity. So, an activity can be represented by its type(A_{type}) and the time stamps (t_{start} , t_{end}), where A_{type} is the type of activity and t_{start} , t_{end} are it's starting and ending times. A process case, as shown in figure 3.2(b) is composed of a unique case id(*id*), a trace T, which is a vector of performed activities and a vector x of context attributes.

Case ID	Activity	Start Times	Complete Time
160409	Activation	00:28.1	00:29.1
160409	Bair hugger preparation-EC	01:35.0	01:35.1
160409	Yankauer suction preparation-RO	01:53.0	01:58.1
160409	Oxygen Preparation-BC	02:00.1	02:56.0
160409	Oxygen Preparation-BC	05:28.0	05:35.1
160409	Oxygen Preparation-BC	10:46.0	11:29.0
160409	Pt arrival	10:47.0	10:48.0
160409	Pre-CNMC Tasks	10:47.0	10:48.0
160409	EMS c-collar-CS	10:47.0	17:34.1

Figure 3.2(a): Process traces

Case ID	xxxxx	yyyyy
Age	23	12
Sex	M	F
Response	Transfer	Stat
Injury_type	Blunt	Penetration
Critical	Yes	No
Weekend	Yes	No
Injury on Neck	Yes	No
Injury on Back	Yes	No

Figure 3.2(a): Process Context Attributes

The process trace cluster is the group of traces that have similar type and performance order of activities. A prototype trace is the trace that represents the most typical performance of that cluster. The prototype is basically the summary of information contained in the cluster and highlights the commonalities of the process traces. This can help in comparing and visualizing the differences between the clusters.

The features like significant sequence mining and cluster prototyping make this system one of its kind and increases its audience. All these features are obtained by putting to implementation the latest technology and algorithms. The whole system is data intensive and performance and accuracy is it's major aspect. Various visualizations have been obtained using the robust matplotlib library in python and feature extraction is obtained using the state of art gap-BIDE algorithm and n-grams model and using various clustering techniques.

CHAPTER 4

CLUSTERING PROCESS ATTRIBUTES

In this chapter, we answer the first research question: How can we group together the patients with similar behavior in the attribute log? The answer to this question is related directly to *clustering*. To find meaningful clusters, clustering of context attributes is performed so that the attributes within the clusters are highly correlated and interdependent on each other, whereas the ones outside the cluster are more independent and less correlated. Clustering the attributes helps reduce the search dimension of a data mining algorithm. This feature is of great help in application when there are a large number of attributes for each tuple. It is for this reason and to obtain meaningful groups of traces clustering is an important preprocessing step for our sequence mining algorithm. Fig. 4.1 shows an example of traces that form clusters.

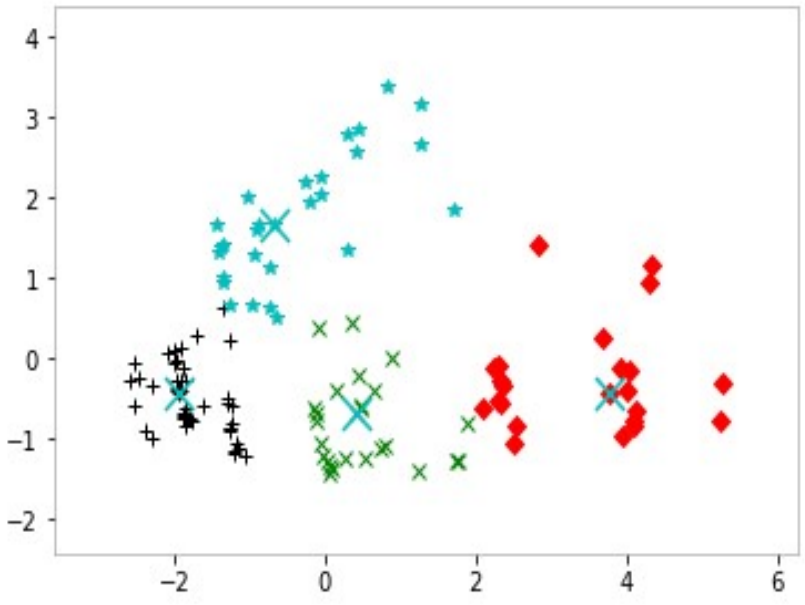


Figure 4.1: Example of clustering of traces

Exemplar based clustering and density based clustering are two broad classes of clustering algorithms. While exemplar based algorithms first selects the representative points(exemplars) from the dataset and then assign the remaining

objects to their nearest exemplar, density based algorithms like DBSCAN form groups of objects based on two parameters: objects within a specific radius from an object(eps) and minimum specified number of neighbors of the object within the radius(min_obj). These algorithms fail when the dataset is of varying density. Moreover, as pointed out, DBSCAN is sensitive to two parameters correct set of which is hard to determine. Fig 4.2, shows the performance of DBSCAN algorithm for our dataset. As can be seen from the figure, the algorithm doesn't give us the desired clean clustered data.

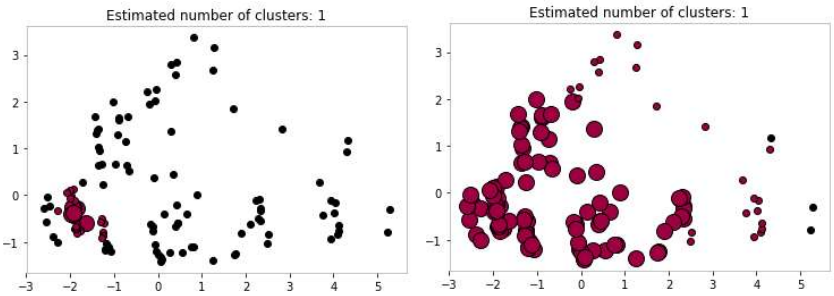


Figure 4.2: Performance of DBSCAN for(eps=0.5,min_obj=20) and for(eps=1.5,min_obj = 25)

Exemplar based clustering is an important is an important class of clustering algorithms. Classical clustering techniques like K-means and the recent ones like Affinity Propagation fall under this category. We used the following three clustering algorithms on our dataset: K-means, Hierarchical clustering, Affinity Propagation.

Finding the right number of clusters for an algorithm is a well-known problem. To overcome this problem, we performed silhouette coefficient analysis of K-means and Affinity Propagation clustering algorithms to find the right number of clusters to be used for our dataset while applying the clustering algorithm. To analyze the behavior of Affinity Propagation algorithm, we also observed the longest range for which the number of clusters remain stable.

4.1 ANALYSIS OF CLUSTERS

We perform silhouette analysis to examine the consistency within the clusters of data. The analysis provides an interpretation of how well an object lies within its cluster. In other word, it's a measure of how similar an object is to its own cluster

when compared to other clusters. The silhouette score ranges from -1 to 1, where a high value indicates that the object fits well within the cluster.

4.1.1 ANALYSIS OF K-MEANS CLUSTERING ALGORITHM

Selecting the right number of clusters is a prerequisite for K-means clustering algorithm. This is a well-known problem. We used Silhouette analysis to find the right number of clusters for our dataset. The results of our analysis are shown in table below:

No. of clusters(n_clusters)	Average Silhouette Score
2	0.535
3	0.486
4	0.527
5	0.530
6	0.545
7	0.483

Table 4.1 Results of Silhouette Analysis for K-means clustering

From the table above we can see that the **n_clusters** value of 3 and 7 are a bad pick owing to their low value of the score. For n_clusters = 2, the size of cluster 0 is large owing to the grouping of other clusters into one. This is shown in Fig. 4.3. Choosing n_clusters = 6 is also a bad choice due to the presence of cluster with below average silhouette score and wide fluctuations in the size of silhouette plots. Out of values 4 and 5 we chose the n_clusters value of 4 because in this case the silhouette plots are more or less of the same size and hence reflect a uniform clustering.

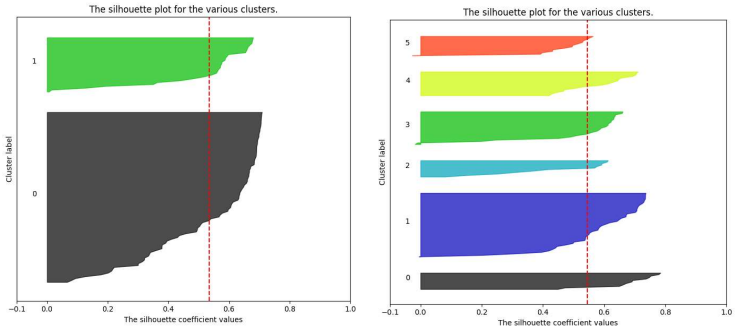


Figure 4.3: Silhouette Plot for n_clusters = 2 and n_clusters = 6

Thus, we chose the n_clusters = 4 for K-means clustering algorithm.

4.1.2 ANALYSIS OF AP CLUSTERING ALGORITHM

Unlike K-means or Hierarchical clustering algorithms, the Affinity Propagation(AP) algorithm doesn't require us to specify the number of clusters beforehand. Although, a similar parameter called preference(p) is required in AP clustering algorithm, its selection is more robust than that of number of clusters(k) in K-means. This is because *p* linearly controls the perception granularity. A plot of the negative value of preference is plotted for a suggested number of clusters for our dataset in Fig 4.4. We can see that the value of preference remains stable for the longest period when the number of clusters is 4.

We also conducted silhouette analysis for the number of clusters obtained above and obtained a value of 0.695, which is better than the value we obtained for K-means clustering algorithm using the same number of clusters for our dataset.

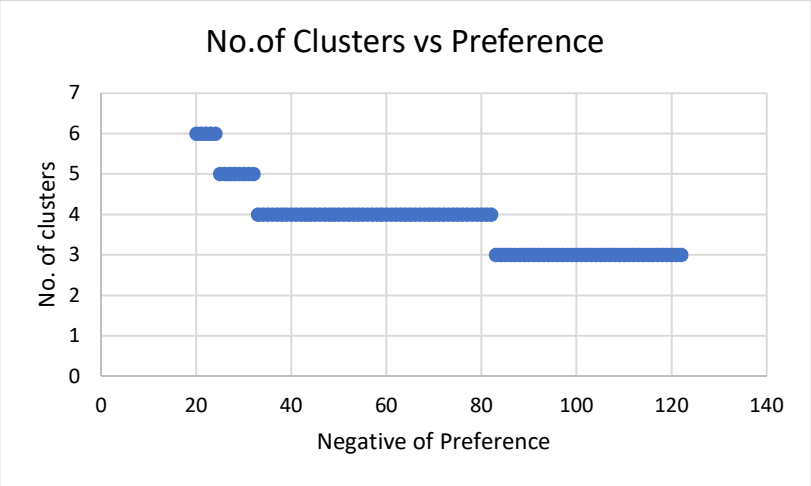


Figure 4.4: No. of clusters varying with preference for AP clustering algorithm

4.2 CONCLUSION

From our analysis of clustering algorithms on our dataset we concluded that both K-mean and Affinity Propagation perform well for our dataset. However, the AP clustering algorithm has a better silhouette score for the same number of clusters

than k-means algorithm. Hence, it's a better candidate to cluster the traces in our case.

Clustering Algorithm	Silhouette Score
K-means	0.527
Affinity Propagation	0.695

Table 4.2 Comparison of Silhouette scores for n_clusters = 4

CHAPTER-5

SEQUENTIAL PATTERN MINING

5.1 INTRODUCTION

In the previous chapter we explored the concept of clustering the data into meaningful groups and performed experiments to find the clustering algorithm that is best suited for our dataset. In this chapter, we explore in detail the second research question: **How do we mine the meaningful sequence of steps for each group?** The answer to this question is directly related to *sequential pattern mining* [13]. In our attempt to find accurate and specific significant recommendations for a particular group of patients under purview we have implemented and compared the results of sequential mining algorithms and analyzed them together to find the one best suited for our dataset.

5.2 SEQUENTIAL PATTERN MINING

Sequential Pattern Mining is an important data mining problem with broad applications. It discovers frequent subsequences as patterns in a database. It finds its use in the analysis of access patterns and the areas such as analysis of time related processes such as DNA sequences, weather patterns etc. Sequential pattern mining has emerged as a technology to pertain to discover such subsequences. This problem was first addressed by Agrawal and Srikant [14] and was defined as follows:

"Given *a database of sequences where each sequence consists of a list of transactions ordered by transaction time and each transaction is a set of items, sequential pattern mining is to discover all sequential patterns with a user specified minimum support, where the support of a pattern is the number of data sequences that contain the pattern.*"

Our problem statement and the dataset both qualify for the above problem and we have explored it using Gap-Bide and n-grams algorithms in our work.

5.3 gap-BIDE ALGORITHM

A lot of algorithms have been proposed to mine patterns from a log data. Some of the well-known algorithms are AprioriAll algorithm [15] [16] and GSP (Generalized Sequential Pattern) algorithm. These algorithms mine sequential patterns while maintaining a candidate set of already mined sequential patterns in the mining process. When the dataset is huge, these algorithms will generate a large number of candidate patterns. In other words, these algorithms need a lot of memory in the case of a large dataset. The BIDE algorithms [17] don't maintain a candidate set while mining the frequent sequential patterns and hence they need less space during the mining process. Also, the feature of gap helps in mining sequential patterns that are not continuous, thereby helping in mining more hidden relationships. Thus, the gap-BIDE algorithm just like its predecessors avoids the curse of candidate maintenance and test paradigm, prunes the search space more deeply and checks the pattern closure in an efficient way consuming lesser amount of space than other algorithms.

Table 5.1 shows the input sequence database(SDB) in our running example:

SEQUENCE ID	SEQUENCE
1	C A A B C
2	A B C B
3	C A B C
4	A B B C A

Table 5.1 Example Sequence Database

Fig.5.1 shows a lexographical sequence tree built from our example. Each node of the tree contains a sequence along with its support.

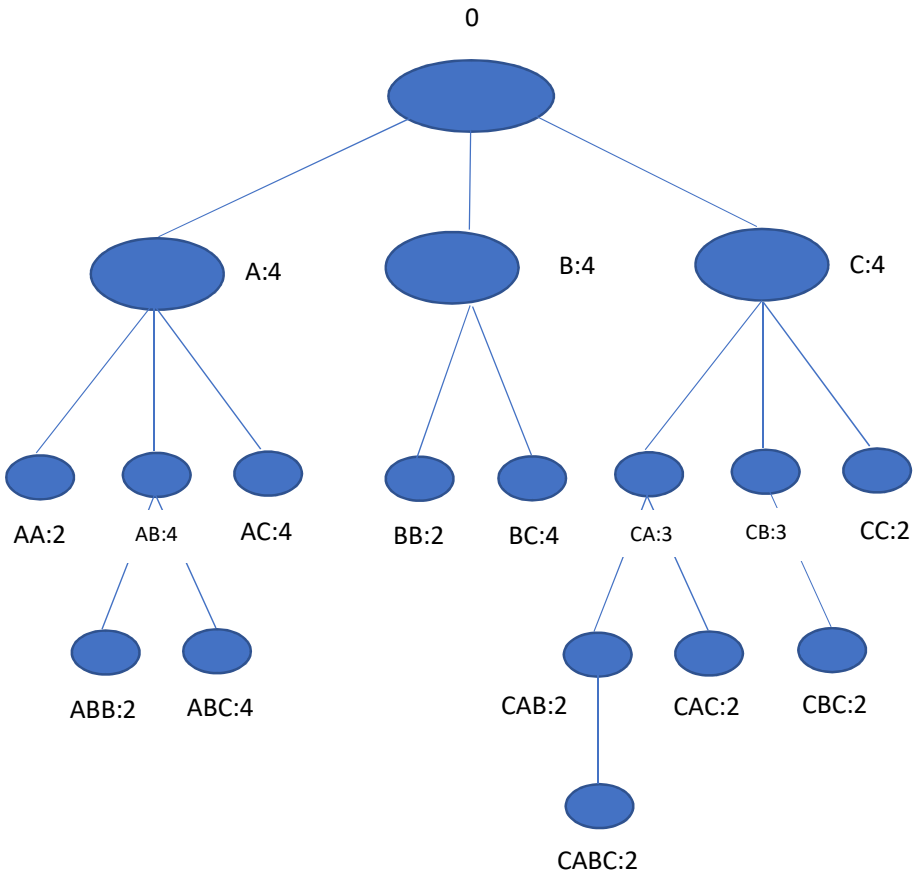


Figure 5.1: Sequence tree from example data

5.3.1 IMPLEMENTATION AND RESULTS

The interface we used to mine sequential patterns for each group (obtained after clustering) was of the form:

Output = gapBide(input_seq, support, min_gap, max_gap)

where min_gap and max_gap minimum and maximum gap constraints and support specifies the minimum of input sequences that should contain the sequential pattern for it to be a part of the output.

We obtained sequential patterns for all the clustered groups keeping support = 2 and min_gap and max_gap values to 0. Using a support value of 2 allowed us to obtain all possible sequential patterns of interest for each group. Keeping max_gap and min_gap values to 0 kept the result set to an interpretable size and gave us all the continuous sequential patterns in each group.

5.3.2 DRAWBACKS OF gap-BIDE FOR OUR CASE

The gap-Bide algorithm works well to mine sequential patterns for all the groups for our framework. However, it suffers from issues that hampers its performance for our work.

- Our dataset consists of a sequence of activities for a particular patient. These activities are stored as a set of strings, with each activity name consisting of a group of characters. Gap-BIDE algorithm however, requires the activities to be represented as single character. This required us to decode each activity name as a unique character before input and encode it back. This added task of decoding and encoding is not only tedious but also deteriorated the performance of an otherwise efficient algorithm.
- Although, the algorithm provides us a way to obtain sequential patterns that have occurred ***n*** number of times in the input sequence by using the parameter 'support', it provides us no way to obtain a sequential pattern of desired length without increasing the size of output set to undesirable limits. By keeping the value of support to 1, we can obtain all possible sequential patterns for the group. However, this also introduces redundancies along with the large size of output sequences.

These drawbacks motivated us to shift our focus to a new paradigm of sequential pattern mining called ***n-grams*** model. We have discussed about this approach in detail in the following section.

5.4 N-GRAMS ANALYSIS OF ACTIVITY SEQUENCE

Statistical analysis of sequence of activities of patients is one of the commonly performed tasks in the field of bioinformatics. However, in many tasks, the requirement for the features to be as symbols is restrictive and requires the need of a model that can handle the data in its raw form. It is for this reason we used n-grams feature in our work. The idea of *n*-grams has been borrowed from language processing technologies where n-grams of words form the basic unit in statistical language processing model.

A *n*-gram [18] is a continuous sequence of *n* items and a *n*-gram model is a type of probabilistic language model for predicting the next item in a sequence. It is implemented in the form of a *(n - 1)* Markov model. *N*-gram models provide benefits in the form of simplicity of implementation and scalability of the model. With a large value of n, the model can store more context, with an understood space-time tradeoff allowing small applications to scale up when needed.

5.4.1 IMPLEMENTATION AND RESULTS

The implementation of gap-BIDE algorithm, although not best suited for our case gave us an idea about the nature of sequential patterns that we may get as output. This helped us better analyze and validate the results we obtained after the implementation of *n*-grams model.

Clustering of patients into groups helped us obtain their features and the IDs from the attribute dataset. This helped us separate the patients into the designated groups from our trace dataset as well. N-grams model when applied to the IDs of patients in our trace dataset helped us obtain continuous sequential patterns along with the frequency of their appearance in the group. Unlike, the gap-BIDE algorithm where we input a parameter called support, the *n*-grams model helped us obtain the frequency of a specified length of sequential pattern in the group without using any such extra parameter. Table 5.2 shows the number of sequential patterns of specific length obtained for each group.

	Group 0	**Group 1**	**Group 2**	**Group 3**
Sequence Length = 1	44	44	44	44
Sequence Length = 2	69	74	64	67
Sequence Length = 3	48	56	35	48
Sequence Length = 4	24	17	9	15
Sequence Length = 5	6	2	0	0
Sequence Length = 6	2	0	0	0
Total Number of Sequences for each group	193	193	152	174

Table 5.2 Number of Sequences for each group obtained by n-grams

The results are in contrast to ones obtained with the help of gap-BIDE algorithm shown in table 5.3 below:

	Group 0	**Group 1**	**Group 2**	**Group 3**
Total Number of Sequences for each group	1300	528	706	1149

Table 5.3 Number of Sequences for each group obtained by gap-BIDE

The results shown above just show the number of sequences obtained as output for each group. However, there may be some sequences that are present in multiple groups or may be present in all the groups as a part of a common procedure. It is important to find the group for which the aforesaid sequence in question is of significance. This calls for a need to conduct significance test of the sequences obtained so that a proper relevance of the sequential pattern for a group can be obtained and the results can be standardized for the group. We discuss this in the next chapter and a novel 'one vs rest' technique to mine significant sequential patterns for each group.

CHAPTER 6

STATISTICAL ANALYSIS OF CLUSTERS AND SEQUENTIAL PATTERNS

In statistical analysis, there can be scenarios wherein the association between two variables is obtained by chance. Such associations pollute the result and hence they need to be dealt with through a proper mechanism. In our case, there can be a possibility that, not all the sequential patterns obtained for a group form a representative set of patterns. Such patterns need to filtered out for the output to be as accurate as possible. This calls for the need for a test of significance to find out the sequential patterns that are of significance to a particular group. We will discuss our procedure of a significance test in the sections that follow. We also discuss the results obtained after examining the probability value (p-value) for the significance test conducted.

6.1 CALCULATING Z-Score FOR TWO PROPORTIONS

A Z-score is the number of standard deviations from the mean a data point is. It is also known as 'standard score' and can be placed on a normal distribution curve. Z-scores range from -3 standard deviations (far left) upto +3 standard deviations (far right). Z-scores are a way to compare results from a test to a normal population.

In it's simplest form, the Z-score of a sample is given by the equation:

$$z = \frac{x - \mu}{\sigma}$$

Where x is the value of the sample

μ is the mean of the population

σ is the standard deviation

However, the mathematics of the Z-score formula [19] change when we are interested in checking whether one population p_1 equals another population p_2.

Applying this concept to our case, we can state that if p_1 is the population of patients that has observed a sequential pattern P and p_2 is another population of patients that has also observed the same sequential pattern P , then we are interested in testing the null hypothesis:

$$H_0: p_1 = p_2$$

against the hypothesis:

$$H_A: p_1 \neq p_2$$

The test statistic for testing the difference in two population proportions, that is, for testing the null hypothesis is:

$$Z = \sqrt{\frac{(\widehat{p_1} - \widehat{p_2})}{\hat{p}(1 - \hat{p})(\frac{1}{n_1} + \frac{1}{n_2})}}$$

where $\hat{p}$ is the proportion of successes in the two populations combined and is given by the equation:

$$\hat{p} = \frac{Y_1 + Y_2}{n_1 + n_2}$$

After calculating the Z-score, the next step in the process is the calculation of p-value from the Z-scores we obtained after implementing the above formulae. In the next section we discuss the calculation of p-value from Z-score and its implications.

6.1.1 CALCULATING p-value FROM Z-Scores

A p value is used in hypothesis testing to help you support or reject the null hypothesis. The p value is the evidence against a null hypothesis. The smaller the p-value, the strong the evidence that you should reject the null hypothesis. The p-value lies between 0 and 1 and is interpreted as follows:

- A small value of p (typically <= 0.05) indicates a strong evidence against the null hypothesis, hence the null hypothesis can be rejected.
- A large value of p (typically > 0.05) indicates a weak evidence against the null hypothesis, hence the null hypothesis cannot be rejected.

We calculated the p-value from Z-score using the *stats* module in python's *scipy* library.

Applying these concepts to an example scenario from our output sequential patterns we can easily understand the importance of these statistical analysis elements in our research work.

Let us consider a group (let's say group-0) of patients for which a particular sequential procedure has been observed n1 number of times. Let us also assume that the sequential procedure has been observed n2 number of times in rest of the groups combined. For the procedure in question to be of significance for group-0, the p-value for sequential procedure should be less than 0.05 otherwise it is discarded for the group and considered for significance with other groups. Table 6.1 shows a scenario from the result set for Group-0 wherein the sequential pattern obtained is of significance.

Sequential Pattern	**Support for Group-0**	**Support for rest of the groups combined**	**Z-score**	**p-Value**
pt_arrival → pre_cnmc_tasks →bair_hugger_ec	4	1	1.9926	0.0462

Table 6.1: p-value and Z-score for a significant sequential pattern for Group-0

In a similar manner, we mined sequential patterns that are a true significant representative for a particular group and presented the results.

Each activity in our dataset has a timestamp in the form of its starting and finishing time. The idea that duration and order of occurrence of a particular activity in a sequence can give us useful insight into the association of an activity with a group along with the sequential procedure motivated us to explore into the temporal aspects of the patterns. A brief discussion of the same has been presented in the section below.

6.2 EXPLORING THE DURATION AND FREQUENCY OF OCCURRENCE OF ACTIVITIES FOR EACH GROUP

Including information about the duration of occurrence of activities starting from their time of commencement, has the benefit that one can get to know about the order of occurrence of each activity separately within each group and hence gain insight about the behavioral aspects of the group. This information is different from the previously mined sequential patterns. While the sequential patterns give information about the procedural steps, the knowledge obtained from the duration analysis gives information about an individual activity. In a similar manner information about the frequency of occurrence of an activity within each group gives insight into the details of how a particular activity dominates the group depending on the type of treatment recommended for the group. We have shown visualizations [20] for these concepts in Chapter-7 and analyzed the results for a better understanding of the topic.

CHAPTER 7

A CASE STUDY WITH TRAUMA RESUSCITATION PROCESS

In this chapter, we present and discuss our final results obtained from a real-time data collected by our partner team. This data was collected in collaboration with Children's National Medical Center, a Level-1 trauma center in Washington D.C. The dataset consists of information about 122 patients and is divided into two parts. One part consists of information about the activities performed on each patient along with the starting and finishing time of each activity. The other part consists of information about the features of each patient, the type of injury, period of admittance etc. There was a total of 180 classes of activities performed. Each activity is stored in the data in a sorted manner based on its starting time giving us a logical order of occurrence.

We analyzed the data collected and ran our algorithm on it to obtain the final results including the visualizations. As discussed in Chapter-4, we applied various clustering algorithms on the data sets to find out groups of relevance. We found that Affinity Propagation algorithm is best suited for our data and that the appropriate number of clusters is 4. Figure 7.1 shows the groups in their raw form after clustering.

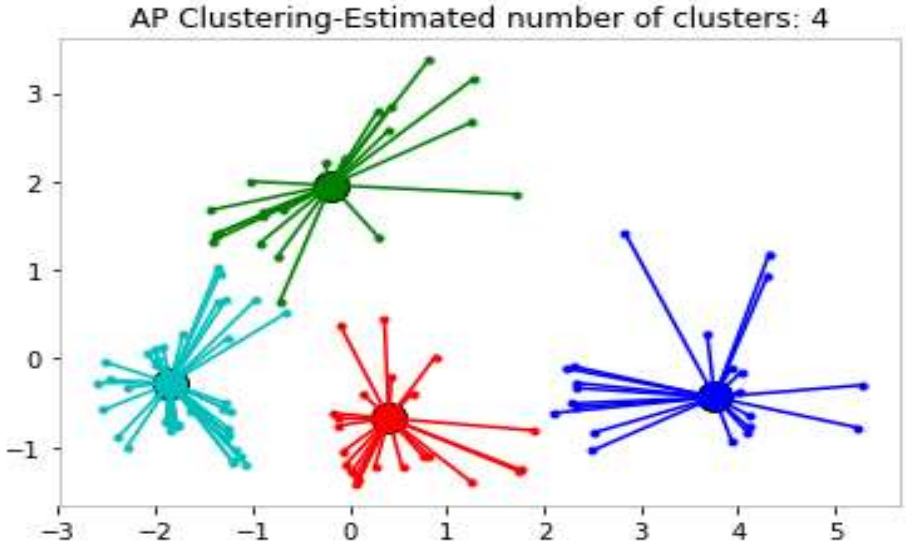
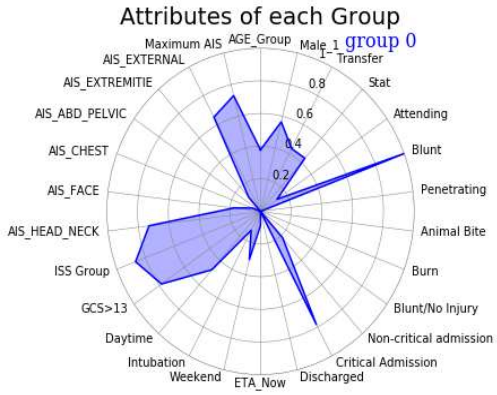


Figure 7.1: Affinity Propagation clustering on the data

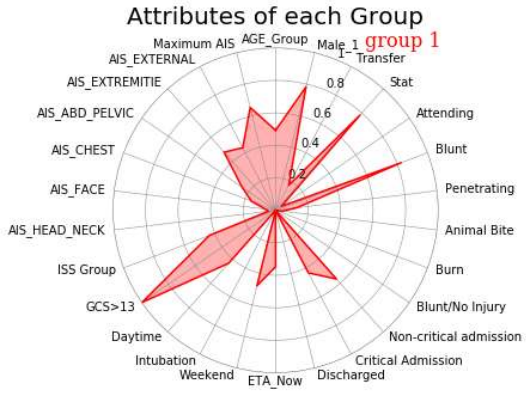
In chapter-5 and chapter-6 we discussed about mining the prototype sequential patterns for each and finding out the statistical significance of each sequential pattern to the groups. The results obtained after the analysis have been shown in Figures 7.2-7.5. In these figures, we have shown how the attributes of a group effect the type of procedure/treatment they have obtained and hence recommend the same for new incoming patients that belong to a particular group.



Recommended Steps
Total Verbalized-GCS
Visual inspection-G
L Visual inspection-EAR
Visual inspection-LUE → Palpation-LUE
Palpation-RLE → Palpation-LLE
Palpation-A → Visual inspection-A

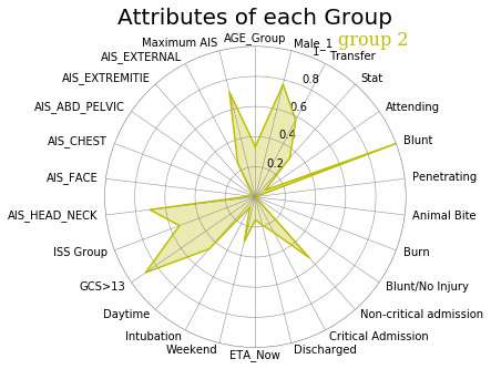
Figure 7.2: Recommended steps for Group-0

`



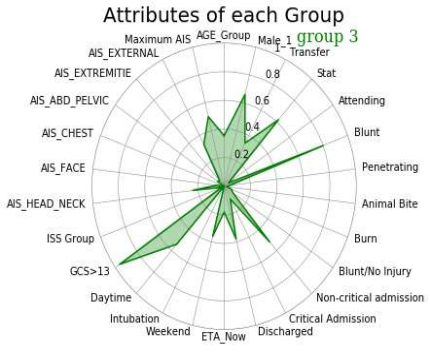
Recommended Steps
Visual inspection-N
Palpation-LUE→Visual inspection-LUE
Right pupil-PU → Left pupil-PU

Figure 7.3: Recommended steps for Group-1



Recommended Steps
Visual assessment-AA
Stability-PE
Verbal assessment-AA
R DP/PT-PC →L DP/PT-PC
Palpation-F → Visual inspection-F
Palpation-C →Visual inspection-C
Visual inspection-NE → Palpation-NE
Visual inspection-A → Palpation-A
Visual inspection-LUE → Palpation-LUE
Visual inspection-BK → T-Spine-BK
Chest Auscultation-BA →R DP/PT-PC → L DP/PT-PC
Visual inspection-BK → T-Spine-BK → L-Spine-BK

Figure 7.4: Recommended steps for Group-2



Recommended Steps
Rectal-BK
Palpation-NE → Visual inspection-NE
Palpation-C → Visual inspection-C
Palpation-RUE → Visual inspection-RUE
L DP/PT-PC → R DP/PT-PC

Figure 7.5: Recommended steps for Group-3

From our trained model, we can see that several context attributes such as, injury type, place of injury, severity of the injury etc., turn out to be statistically

significant for the trace clusters. Using the radar charts obtained from Affinity Propagation Clustering (APC) we can see that the attributes like AIS Extremital and ISS Group are heavily correlated within the clusters.

To dig deeper into the correlations, we plotted the average duration of an activity from an average starting time within a group. The visualization shown in Figure 7.6 throws light on the order of occurrence of each activity between the groups. Some activities occur earlier in a group and some occur later than the others.

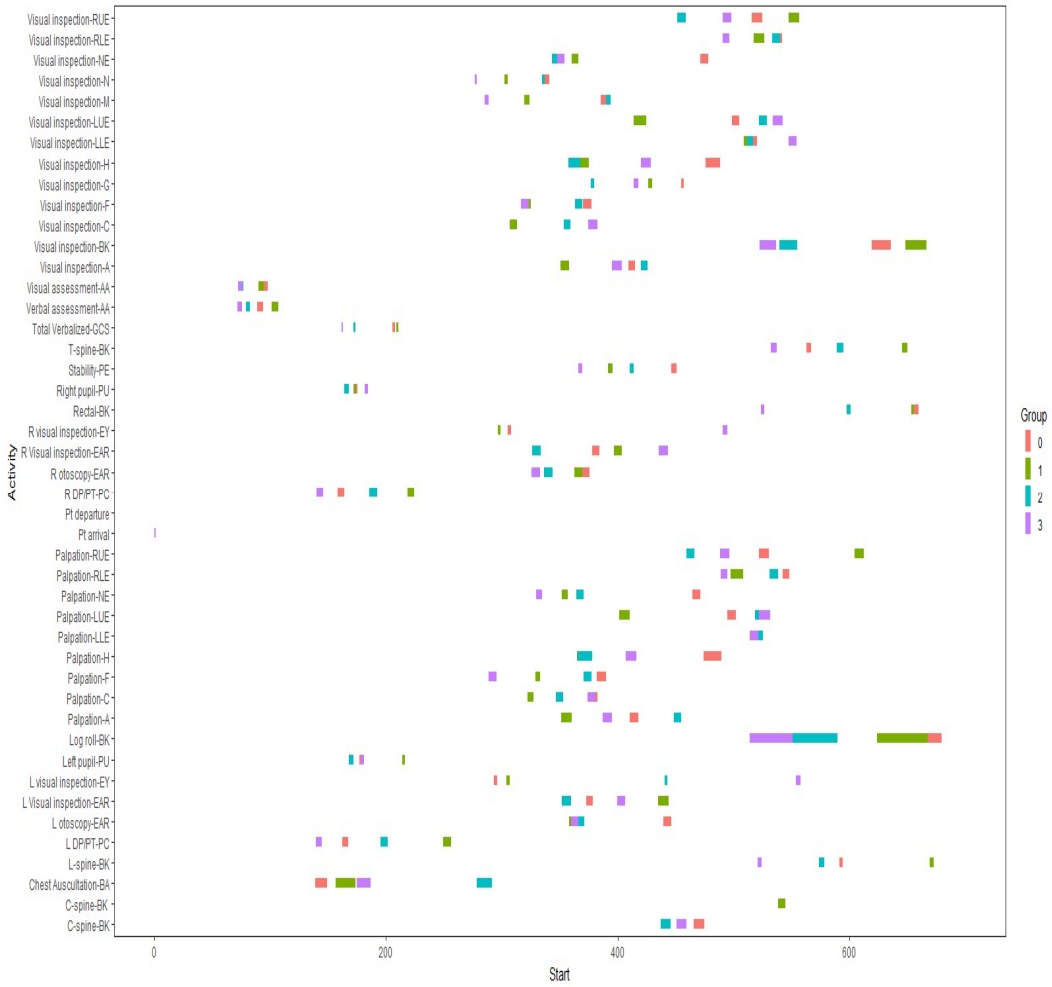


Figure 7.6: Average duration of an activity from average starting time for each group

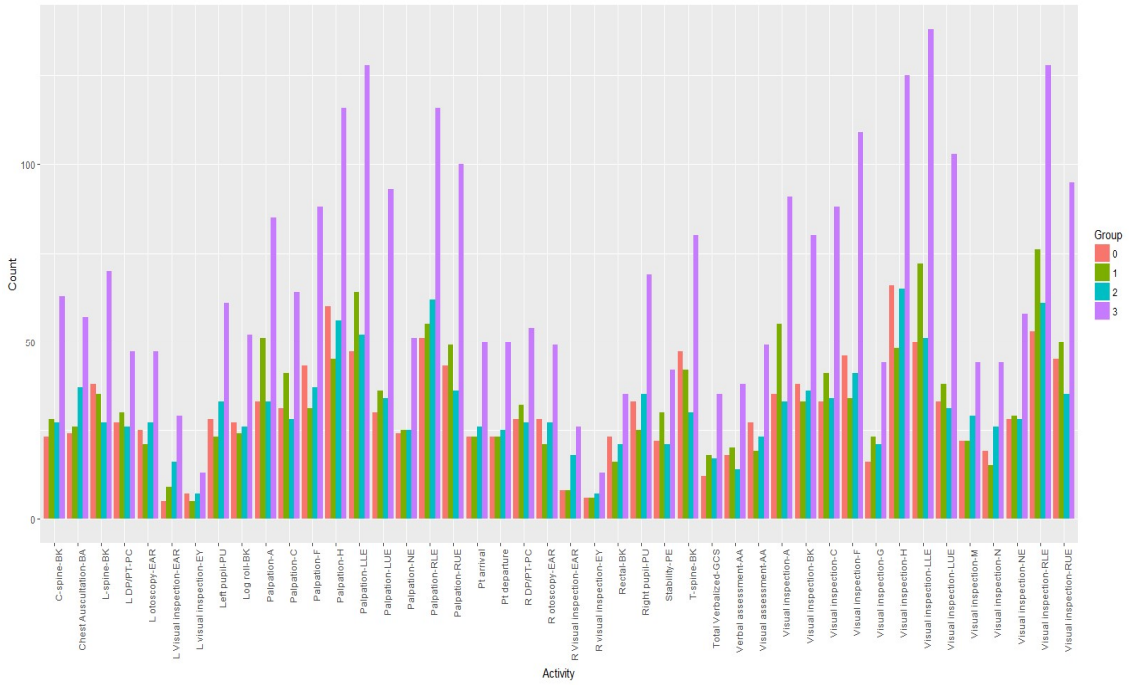


Figure 7.7: Frequency of occurrence of an Activity each group

Figure 7.7 shows the relative frequency of each significant activity in every group. It is very evident from the figure that group-3 (marked in violet) is the group where the activities have occurred more frequently than other groups. Group-1(marked in green) is the one to follow in most cases. This can well be attributed to some of the commonalities between the features of group-1 and group-3(Figures 7.3 & 7.4). Features like 'GCS>13','Non-Critical Admission', 'Stat' show similar normalized behavior.

CHAPTER 8

CONCLUSION

We have presented a process analysis and data-driven recommendation framework in this research work. Our framework clusters the process traces of patients into relevant groups and extracts some sequential recommendations for each one of them. Tests of significance are then conducted to mine the significant prototype recommendations. We introduced novel approaches to mine the prototype sequential patterns. Although, we have currently tested our framework only with the data from the field of medicine, it can also be extended to other real-world processes.

CHAPTER 9

REFERENCES

[1] S. Yang and I. Marsic, "A Data-driven Process Recommender Framework," in *KDD*, 2016.

[2] J. Clark and et al., "Computer-generated trauma management plans: comparison with actual care.," in *World Journal of Surgery*, 2002.

[3] M. Fitzgerald, "Trauma resuscitation errors and computer-assisted decision support," in *Archives of Surgery*, 2011.

[4] C. Li and J. Wang, "Efficiently Mining Closed Subsequences with Gap Constraints," in *Proceedings of the 2008 SIAM Internation Conference on Data Mining*, 2008.

[5] H. Mannila, H Toivonen and A.I. Verkamo, "Discovery of Frequent Episodes in Event Sequences," in *Data Mining and Knowledge Discovery,vol1,pp. 259-289*, 1997.

[6] X.Yan, R.Afshar and J.Han, "CloSpan: Mining Closed Sequential Patterns in Large Database," in *SDM*, 2003.

[7] J. Pei, J. Han and et al., "Mining Sequential Patterns by Pattern-Growth:The Prefix-Span Approach," in *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 2004.

[8] A. Floratos and I. Rigoutsos, "Combinatorial pattern discovery in biological sequences: the teiresias algorithm.," in *Bioinformatics, 14(1)*, 1998.

[9] M.Zhang, B.Kao and K.Yip, "Mining Frequent periodic patterns with gap requirement from sequences.," in *SIGMOD*, 2005.

[10] E. Forgey, "Cluster Analysis of Multivariate Data: Efficiency vs Interpretability of Classification," in *Biometrics, vol. 21, p. 768*, 1965.

[11] Murtagh, Fionn and et al., "Ward's hierarchical clustering method: Clustering criterion and agglomerative algorithm.," in *xrXiv*, 2011.

[12] D. Dueck and B. J.Frey, "Clustering by passing messages between data points," in *Science. 315*, 2007.

[13] Fournier-Viger, P., et al., "Fast Vertical Mining of Sequential Patterns Using Co-occurrence Information.," in *Proc. 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2014.

[14] R. Agrawal and R. Srikant, "Mining Sequential Patterns," in *ICDE*, 1995.

[15] B. Valaramathi and M. Saravanan, "Generalization of web log datas using WUM technique.," in *Proceedings of the 12th International Conference on Networking, VLSI and signal processing (ICNVS '10), pp. 157–165*, 2010.

[16] N. R. Mabroukeh and C. I. Ezeife, "A taxonomy of sequential pattern mining algorithms.," in *ACM Computing Surveys, vol. 43, no. 1, article 3*, 2010.

[17] J. Wang, C. Li and J. Han, "Frequent closed sequence mining without candidate maintenance.," in *IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 8, pp. 1042–1056*, 2007.

[18] "https://lagunita.stanford.edu/c4x/Engineering/CS-224N/asset/slp4.pdf".

[19] "https://onlinecourses.science.psu.edu/stat414/node/268".

[20] S. Yang, I. Marsic and et al., "VIT-PLA: Visual Interactive Tool for Process Log Analysis.," in *KDD Workshop on Interactive Data Exploration and Analytics*, 2016.