Netflow Data

Thomas Ng

Department of Mathematics

Imperial College

MSc Statistics - CID: 00741795

Supervisor: Dr. Nicholas Heard

Declaration

The work contained in this thesis is my own work unless otherwise stated.

All the IP addresses are provided by the ICT department and all of them are anonymised.

Abstract

The focus of this project is to build statistical models to understand IP behaviour within college and primarily, to detect anomalies within college network traffic. We mainly focus on IP count data. The aim is to examine different college IPs and check for anomalous behaviour based on IP counts and the destination ports they use .

Chapter 3 is the crux of this project, as the Markov Chain model is built to monitor IP activity at different times within a day. The data is split into two sets: training and testing. And anomalous behaviour were discovered for several IPs over the testing period. However, we need to be cautious of false alarms.

Chapter 4 builds a bayesian model for port scanning. The aim is to detect whether an IP of interest has been using any destination ports that it does not normally use on a typical day in college. The idea of continuously updating the parameters is illustrated and Perl is heavily used to generate the p-values. No anomalies were found at the end.

We also tried to model IP count data using graphs in chapter 2 and agglomerative clustering technique in chapter 5. Since the IPs are anonymised and given limited amount of time, the results in these two chapters are not as profound as those found in chapter 3.

An introduction to the dataset is given in Chapter 1. The last two chapters are conclusion and appendix. The appendix simply includes some sample perl scripts and shell scripts used for this project.

Acknowledgements

I would like to express my deepest appreciation to my project supervisor, Dr Nicholas Heard, who guided me throughout the project. Without his guidance and persistent help this dissertation would not have been possible.

I would like to thank the Course Director of this master programme, Dr Axel Gandy, who created this interesting and challenging programme and provided me with this opportunity to learn statistics from top-notch experts at Imperial College.

In addition, I would like to thank my parents and friends who supported me financially and mentally in all these years. Lastly, I would like to express my deepest gratitude to an important friend and brother , Karlson Lee, who is a truly inspirational individual and his presence added so much colour to my life at Imperial.

Contents

List of Figures

LIST OF FIGURES

LIST OF FIGURES

List of Tables

LIST OF TABLES

1

Introduction

1.1 Pre-processing data with Perl and UNIX

This chapter describes and examines the structure of the netflow dataset, which requires data pre-processing using Perl and UNIX. By simply looking at the count data, we obtained different kind of plots, histograms as well as summary tables. The data are divided into two categories: College VS Non-College plus College, and one day VS one month.

The netflow dataset is provided by the ICT department at Imperial College. The data is known as Netflow version 9 format, orginally invented by Cisco. For security reasons, the data were anonymised by ICT. The data were then streamed onto a storage server in Maths and FIRA in turn accessed this data directly over the network.

Record of each network entering or leaving the device on any or all of its interfaces is captured by built-in netflow sensors in many network routers and various network devices such as switches, servers , hubs. A switch controls network traffic flow based on the address information within each packet. Its main function is to reduce the amount of unneccessary traffic. A hub is a network device that connects computers together and passes on all the information or data it receives, so that the data is shared among all devices that are connected to its ports. Put simply, hubs are used to extend the network. Not only does the built-in sensors keep a record of which interface or network the packets came in, but it also captures the network protocols used on any of the 7 OSI layers , packet's progress through the device and where it went to when it left the device, along with its starting point (source) and final destinations. All these

information are stored as a table in memory and can be downloaded to a computer later for analysis.

As with most of the statistics projects, the dataset look nasty and some data processing is required. The internet traffic data appears in the following format. See figure 1.1.

Date first seen	Duration	Proto	Src IP Addr:Port		Dst IP Addr:Port	Packets	Bytes	Flows
2013-03-04 03:59:43.428	0.000	UDP	100.253.15.130:17500	->	100.253.11.248:17500	5	950	1
2013-03-04 03:59:43.476	0.000	UDP	100.253.244.217:17500	->	100.253.244.58:17500	1	140	1
2013-03-04 03:59:43.500	0.000	UDP	100.253.12.245:631	->	100.253.12.62:631	3	600	1
2013-03-04 03:59:43.544	0.000	UDP	100.253.173.229:17500	->	100.253.171.194:17500	4	560	1
2013-03-04 03:59:43.548	0.000	UDP	100.253.174.48:17500	->	100.253.171.194:17500	4	556	1
2013-03-04 03:59:43.576	0.000	PIM	100.253.243.198:0	->	24.24.240.226:0	1	58	1
2013-03-04 03:59:43.580	0.000	PIM	100.253.167.193:0	->	24.24.240.226:0	1	58	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.247.160:1345	2	608	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.247.72:1345	2	608	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.247.66:1345	2	608	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.247.129:1345	2	608	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.247.189:1345	2	608	1
2013-03-04 03:59:43.624	0.004	UDP	100.253.4.249:1346	->	100.253.117.182:1345	2	608	1
2013-03-04 03:59:43.632	0.000	PIM	100.253.39.199:0	->	24.24.240.226:0	1	58	1
2013-03-04 03:59:43.708	0.000	UDP	100.253.79.240:17500	->	100.253.79.196:17500	1	162	1
2013-03-04 03:59:43.712	0.000	UDP	100.253.169.182:17500	->	100.253.171.194:17500	4	684	1
2013-03-04 03:59:43.724	0.000	UDP	100.253.35.233:64248	->	24.24.240.238:8612	1	44	1
2013-03-04 03:59:43.812	0.000	PIM	76.127.35.194:0	->	24.24.240.226:0	1	58	1
2013-03-04 03:59:43.832	0.000	UDP	100.253.174.80:17500	->	100.253.171.194:17500	4	724	1
2013-03-04 03:59:43.868	0.000	PIM	100.253.240.1:0	->	24.24.240.226:0	1	58	1
2013-03-04 03:59:43.888	0.000	UDP	100.253.119.214:17500	->	100.253.116.56:17500	1	140	1
2013-03-04 03:59:43.928	0.000	UDP	100.253.169.28:64018	->	100.253.171.194:1947	4	272	1
2013-03-04 03:59:43.980	0.000	UDP	100.253.52.42:57621	->	100.253.52.4:57621	19	1368	1
2013-03-04 03:59:34.912	9.000	TCP	100.253.68.131:49748	->	100.253.100.103:2800	3	152	1
2013-03-04 03:59:34.652	8.720	UDP	100.253.173.235:137	->	100.253.171.194:137	48	3744	1
2013-03-04 03:59:35.984	8.980	TCP	100.253.231.17:2172	->	100.253.75.15:9100	3	144	1

Figure 1.1: UNIX - Data Extraction using nfdump command

As you can see, the data is easy to read and understand. To interpret this, each line is an instance of data transfer across computers over a large network.

The first line can be read as "at 2013-03-04 03:59:43.428 (up to millisecond), under the protocol UDP, 5 packets, which carry a total of 950 bytes of data, travel from the Source with IP address 100.253.15.130 through the port 17500 to a destination with IP address 100.253.11.248 ". Notice that these IPs are displayed in dotted-decimal format.

When a file is transferred, it is broken up into smaller pieces called packets, and each packet is sent individually. An internet protocol is simply a set of rules for communcation between computers. There are many different protocols collectively carrying out the transfer (3, William & Don 2000).

To extract netflow data efficiently, we use the nfdump command to (13, Poulain and Vantorre 2006-2013) print flow record statistics in legacy machine readable format, with fields separated by "|". This is also known as the "pipe" format. See figure 1.2.

```
[tn12@fira netflow]$ /usr/local_machine/bin/nfdump -r /home/flowdata/ac-core/2013/03/04/04/nfcapd.201
303040400 -o pipe > output_files.txt
[tn12@fira netflow]$ head -n 10 output_files.txt
2|1362369583|428|1362369583|428|17|0|0|0|1694306178|17500|0|0|0|1694305272|17500|0|0|602|0|16|0|5|950
2|1362369583|476|1362369583|476|17|0|0|0|1694364889|17500|0|0|0|1694364730|17500|0|0|255|0|16|0|1|140
2|1362369583|500|1362369583|500|17|0|0|0|1694305525|631|0|0|0|1694305342|631|0|0|602|0|16|0|3|600
2|1362369583|544|1362369583|544|17|0|0|0|1694346725|17500|0|0|0|1694346178|17500|0|0|618|0|16|0|4|560
2|1362369583|548|1362369583|548|17|0|0|0|1694346800|17500|0|0|0|1694346178|17500|0|0|618|0|16|0|4|556
2|1362369583|576|1362369583|576|103|0|0|0|1694364614|0|0|0|0|404287714|0|0|0|237|0|16|192|1|58
2|1362369583|580|1362369583|580|103|0|0|0|1694345153|0|0|0|0|404287714|0|0|0|518|0|16|192|1|58
2|1362369583|624|1362369583|628|17|0|0|0|1694303481|1346|0|0|0|1694365600|1345|0|0|510|0|16|0|2|608
2|1362369583|624|1362369583|628|17|0|0|0|1694303481|1346|0|0|0|1694365512|1345|0|0|510|0|16|0|2|608
2|1362369583|624|1362369583|628|17|0|0|0|1694303481|1346|0|0|0|1694365506|1345|0|0|510|0|16|0|2|608
```

Figure 1.2: UNIX "pipe" format - Data extraction using nfdump -o pipe command

In pipe format, the fields are separated by "|" and the IPs are displayed in decimal format. (See *Appendix* for code to convert IP from decimal format to dotted-decimal format and vice versa)

- Column 2 and 4 corresponds to UNIX timestamp. Column 3 and 5 corresponds to milliseconds.
  - In fact, Column 2 and 3 is the starting time of data transfer from Source to Destination, Column 4 and 5 is the ending time. The time difference is the "duration" in figure 1.1.
- Column 6 corresponds to "Protocol" in figure 1.1.
- While Column 7 - 10 corresponds to Source IP address, Column 12 - 15 corresponds to Destination IP address.
- Column 11 and 16 corresponds to Ports.
- Column 23 corresponds to the number of Packets.

- Column 24 corresponds to "bytes", the total size of data carried by the packets.

Since the dataset provided by Imperial College is massive, it would be of our interest to extract the Source IPs and Destination IPs and obtain some count data from them so that we could obtain some basic plots and histograms. Data extraction of this sort is also known as *data processing* and the Perl programming language and UNIX are the most suitable tools for tasks like this.

This kind of data processing is carried out throughout the entire project. So, data of different time frame and various types of data of interest could be extracted and written to text files.

Note that the data incorporates both College & non-College IPs. The concept of College IPs is self-explanatory. It refers to any college computers connection or any college activities, i.e. checking college mailbox, using blackboard etc. Though the notion of non-College IPs may not be as obvious and this is best understood by examples. For instance, if one from outside college campus (in Hong Kong) initiates a connection to a college computer (FIRA), this represents a case of non-college IP connecting to a college IP. Using a college computer to connect to "google.com" is a case of college IP making a connection to non-college IPs.

1.2 Basic Plots and Histograms of College & non-College IPs - 4^{th} March

Using Perl, we first look at one day worth of traffic flow data (4th March), which involves college and non-college IPs. A list of unique source IPs and destination IPs and their corresponding occurrence are extracted, i.e. count data. Plots of count data and $\log_{10}(count)$ data are shown in Figure 1.3. See Figure 1.4 for the corresponding histograms. Given a massive amount of data within a day, there are vast amount of distinct IP addresses and counts. Thus, a figure like 1.3 is not helpful at all. In fact, a table like 1.1 seems to give a better representation of the data. As we can see there is a total number of 1358641 distinct IPs, this is why a plot like 1.3 is not informative at all.

Note that most of the histograms on the left hand side of the figures below are truncated at the 95^{th} or higher quantile of the distribution (removing anything above

1.2 Basic Plots and Histograms of College & non-College IPs - 4^{th} March

	All IPs Count	Source IPs Count	Dest. IPs Count
Maximum	21798834	19430449	21798821
95%	38	0	25
75%	3	0	3
Median	1	0	1
Mean	246.967	123.483	123.483
25 %	1	0	1
5%	1	0	1
Minimum	1	0	0
Total number of distinct IPs	1358641	1358641	1368641
Total number of counts	335539274	167769637	167769637

Table 1.1: Overview of IPs Count

95^{th} or higher quantile), as most of the counts are concentrated at low values with few extremely high values at the right tail. If we do not truncate the data at the 95^{th} or higher quantile of the distribution, we cannot see any distribution of the data (See third plot in figure 1.3). However, truncating the data at the 95^{th} or higher quantile can be dangerous, as this may neglect important IPs with very high counts, which are the hubs. Nonetheless, the histograms on the right hand side of the figure represent the *logarithmic transformation* (base 10) of the data without any truncation. For the sake of completeness, both types of histograms will be included so that readers can have a better understanding of the data. Readers can simply treat the histograms on the left hand side as the distribution of data excluding the hubs and the histograms on the right hand side as the distribution of log data including the hubs.
See table 1.1, figure 1.5, and figure 1.6 for tables and histograms of source IPs count data as well as destination IPs count data.

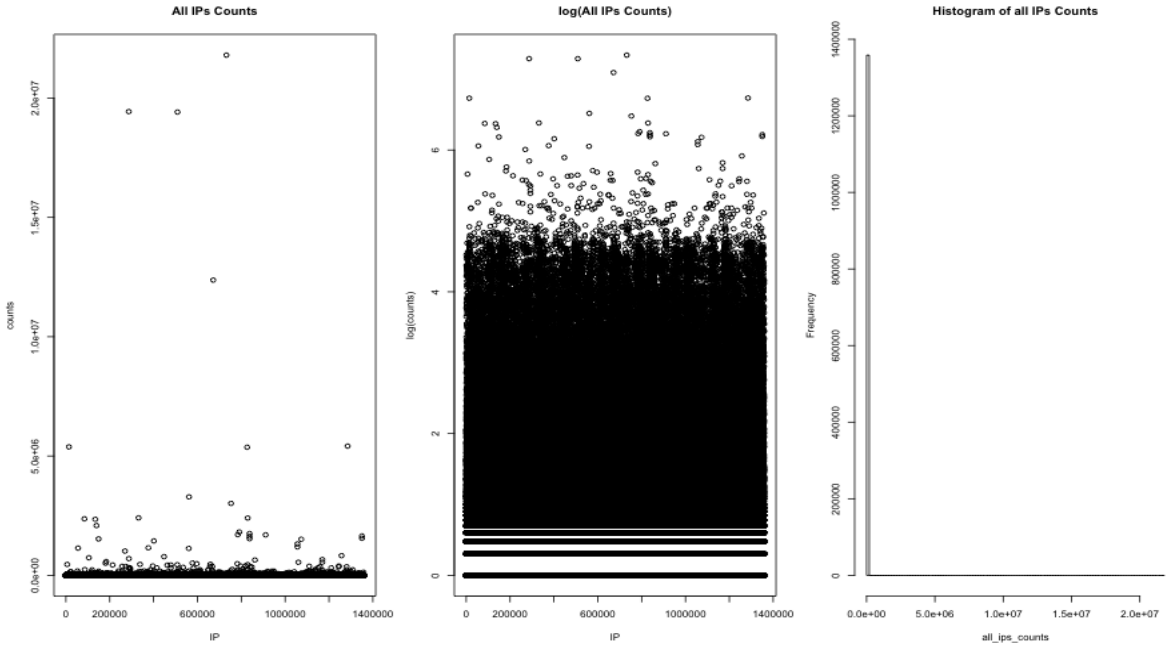


Figure 1.3: College + Non College IPs: All IPs Count

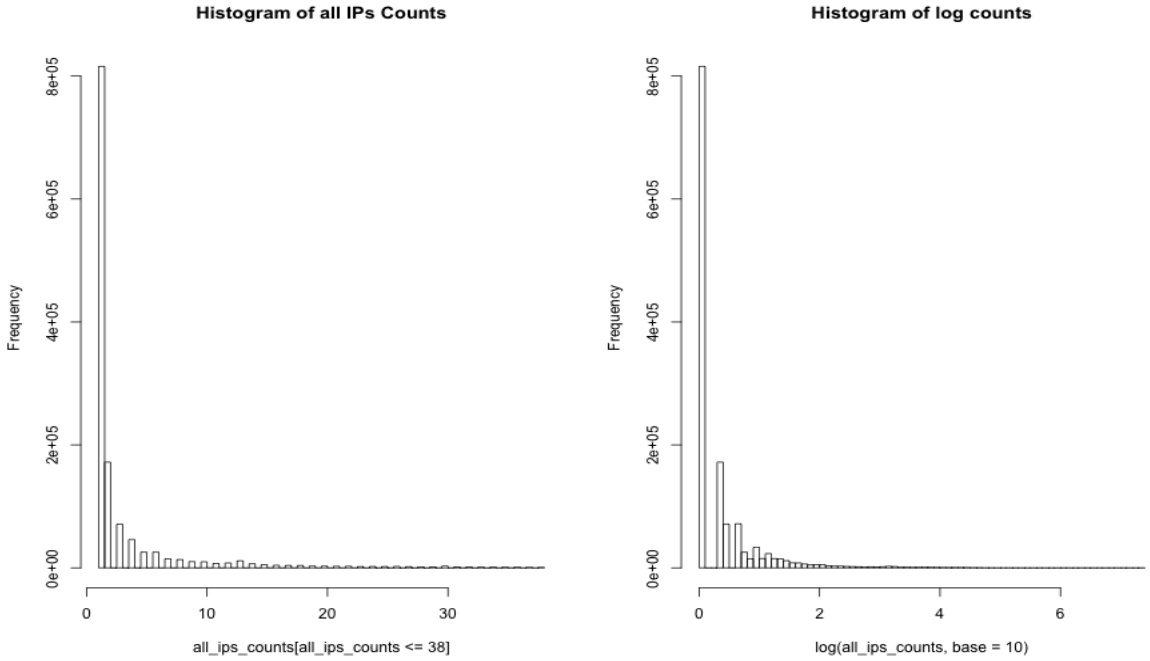


Figure 1.4: Histogram All IPs Counts - College + Non College IPs

1.2 Basic Plots and Histograms of College & non-College IPs - 4^{th} March

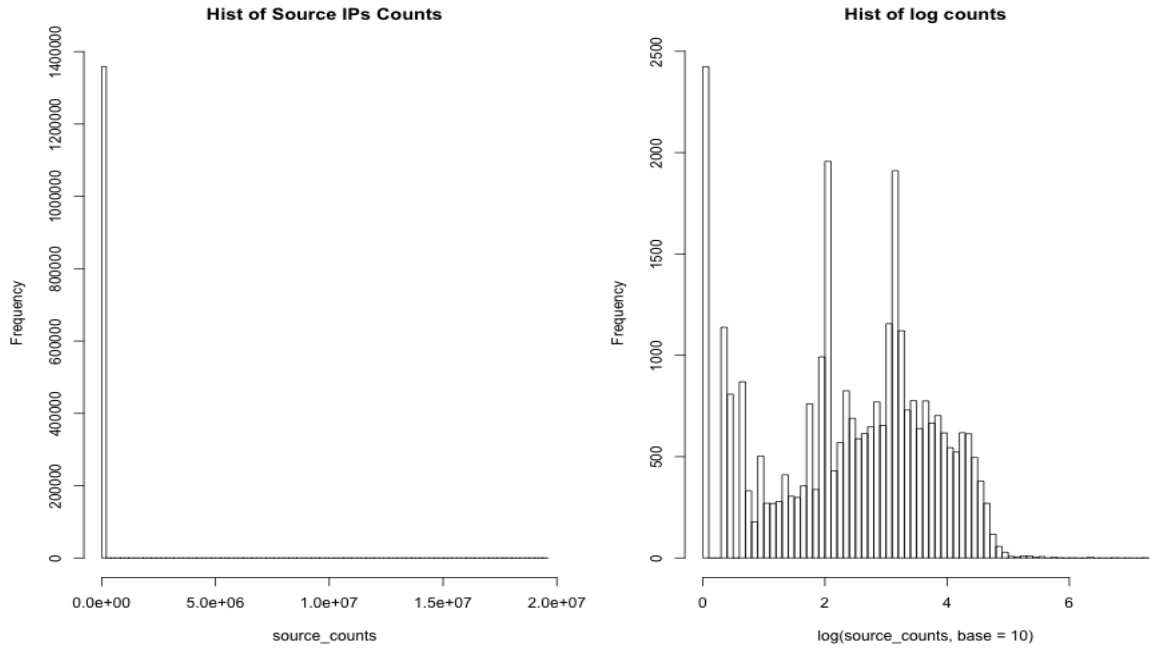


Figure 1.5: Histogram of Source IPs Counts - College + Non College IPs

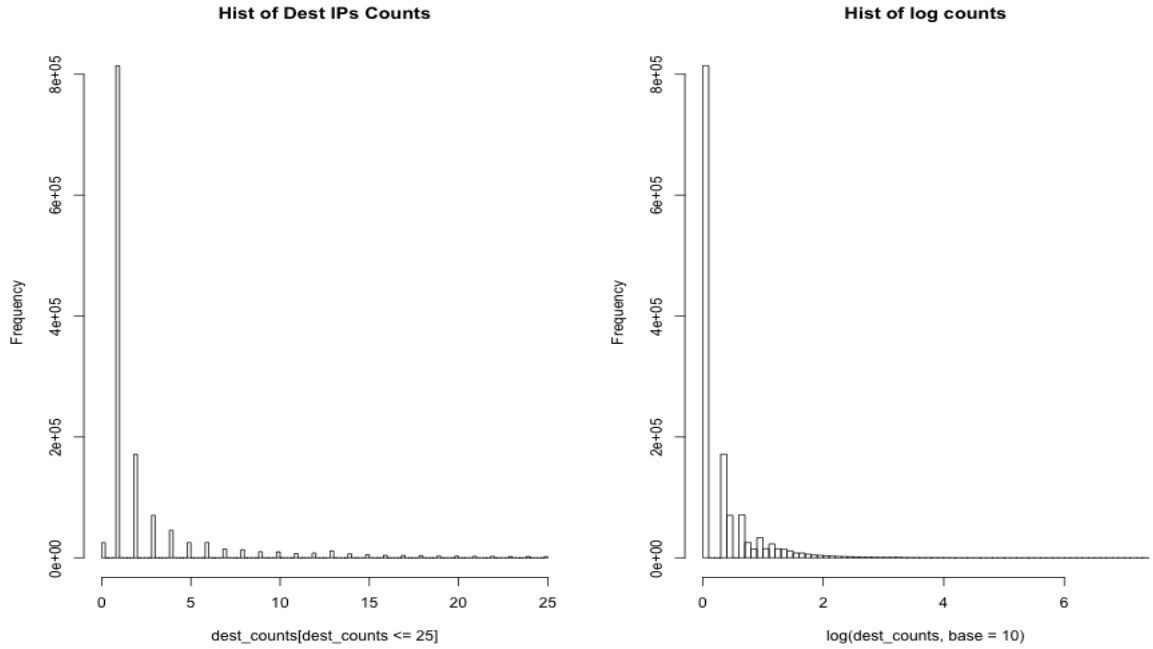


Figure 1.6: Histogram of Destination IPs Counts - College + Non College IPs

A list of unique edges is also extracted. In this context, an edge corresponds to a pair of source IP and destination IP. See table 1.2, figure 1.7 for tables and histograms.

From the list of unique edges, indegree and outdegree are easily obtained. The definition of indegree of a vertex v refers to the number of edges with v ever appear as terminal vertex. See table 1.2 , figure 1.8.

Similarly, the definition of outdegree of a vertex v corresponds to the number of edges with v ever appear as initial vertex. See table 1.2 , figure 1.9.

	Edge Count	Indegree	Outdegree
Maximum	19408655	7290	234740
95%	49	2	0
75%	7	1	0
Median	2	1	0
Mean	41.939	2.944	2.944
25 %	1	1	0
5%	1	1	0
Minimum	1	0	0
Total number of distinct IPs (Edges)	(4000282)	1358641	1358641
Total number of counts	167769637	4000282	400282

Table 1.2: Overview of Edge Count, Indegree and Outdegree

1.2 Basic Plots and Histograms of College & non-College IPs - 4^{th} March

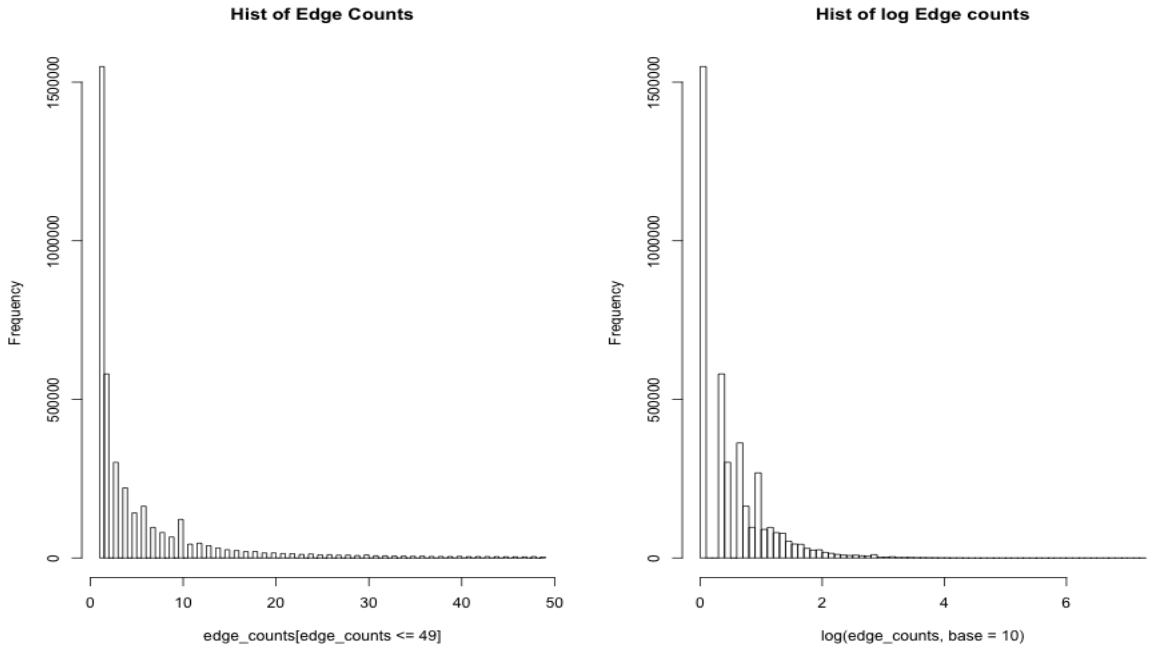


Figure 1.7: Histogram of Edge Counts - College + Non College IPs

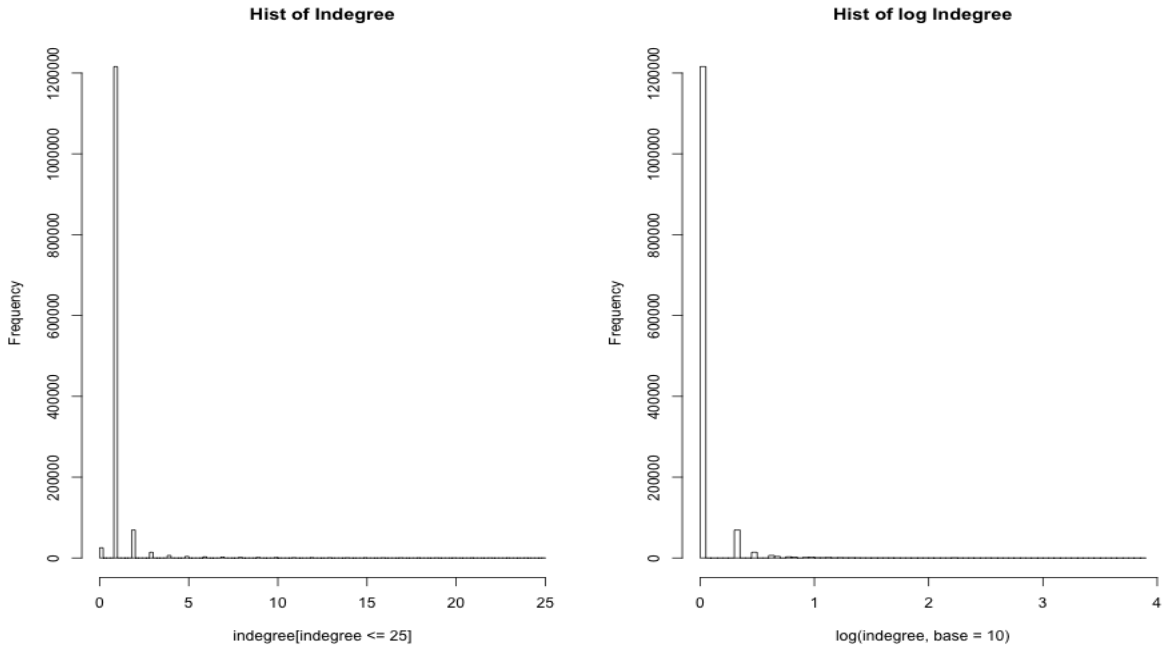


Figure 1.8: Histogram of Indegree - College + Non College IPs

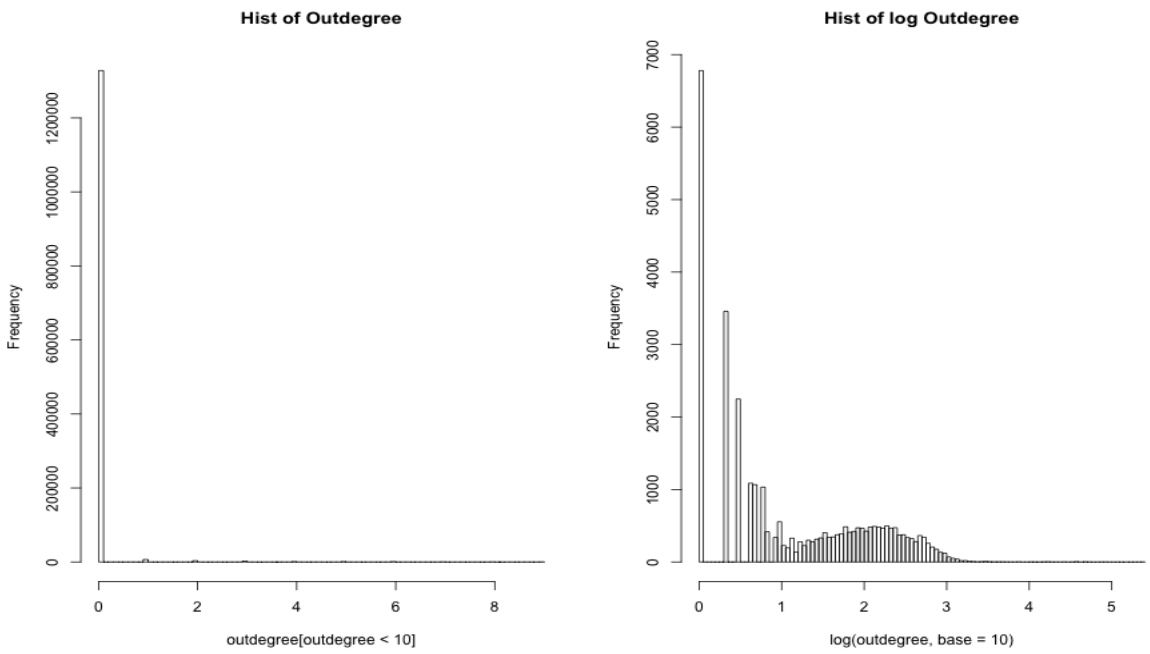


Figure 1.9: Histogram of Outdegree

1.3 College Network IPs - 4^{th} March

Since our aim is to model user behaviour within college network, IP addresses that satisfy any one the following 6 criteria below are considered as College IPs:

Decimal form	Dotted-decimal form
• 2130509824 $\leq$ IP $\leq$ 2130575359	• 126.253.0.0/16
• 1694302208 $\leq$ IP $\leq$ 1694367743	• 100.253.0.0/16
• 1823014912 $\leq$ IP $\leq$ 1823080447	• 108.169.0.0/16
• 1824260096 $\leq$ IP $\leq$ 1824325631	• 108.188.0.0/16
• 1045335552 $\leq$ IP $\leq$ 1045335807	• 62.78.142.0/24
• 1045216000 $\leq$ IP $\leq$ 1045216255	• 62.76.187.0/24

/16 and /24 mean that first 16 bit (first 6 digit) and 24 bit (first 9 digit) of the dotted-decimal IP are fixed, and the remaining digits are wildcards. Again, it would be

of our interest to look at histograms of the count data of one day worth of data within the college network.

See table 1.3 and figure 1.10 for table and histogram of all IPs Count data.
See table 1.3 and figure 1.11 for table and histogram of Source IPs Count data.
See table 1.3 and figure 1.12 for table and histogram of Destination IPs Count data.
See table 1.4 and figure 1.13 for table and histogram of Edge Counts.
See table 1.4 and figure 1.14 for table and histogram of Indegree. Again, we observe a high proportion of ones.
See table 1.4 and figure 1.15 for table and histogram of Outdegree.

	All IPs Count	Source IPs Count	Dest. IPs Count
Maximum	21796182	19430443	21796177
95%	30327.6	26763.6	2323.6
75%	6687	5692	48
Median	1033	435	2
Mean	14112.827	7056.413	7056.413
25 %	14	0	0
5%	1	0	0
Minimum	1	0	0
Total number of distinct IPs	15809	15809	15809
Total number of counts	223109680	111554840	111554840

Table 1.3: Overview of IPs Count

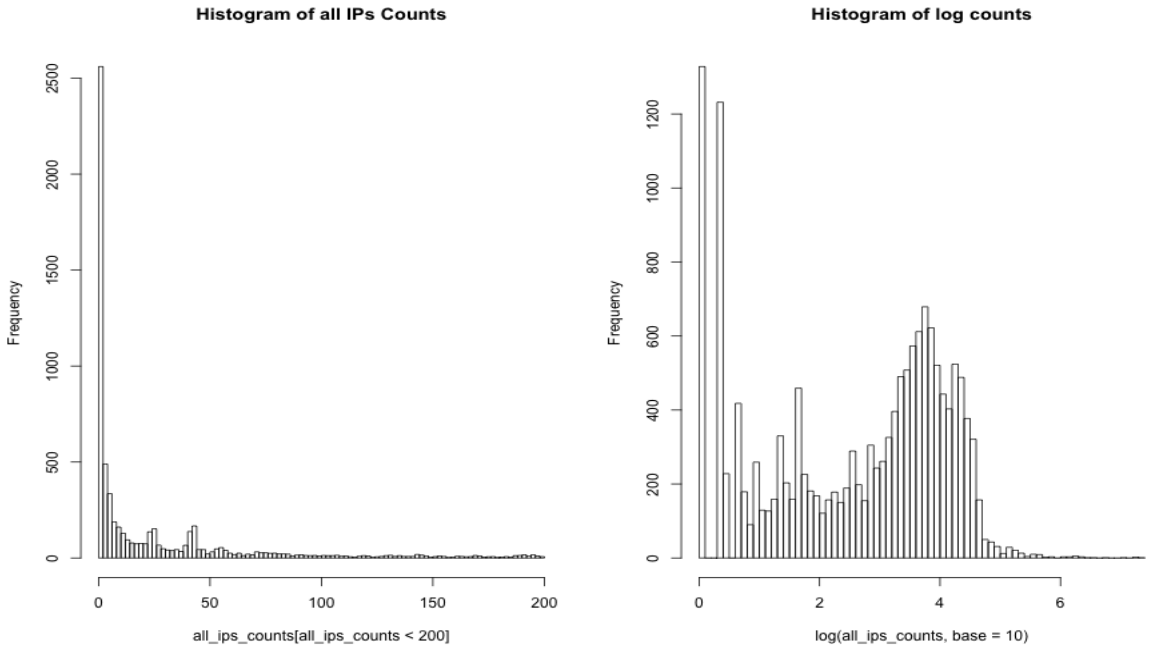


Figure 1.10: Histogram of All IPs Counts - 1 day & College IPs

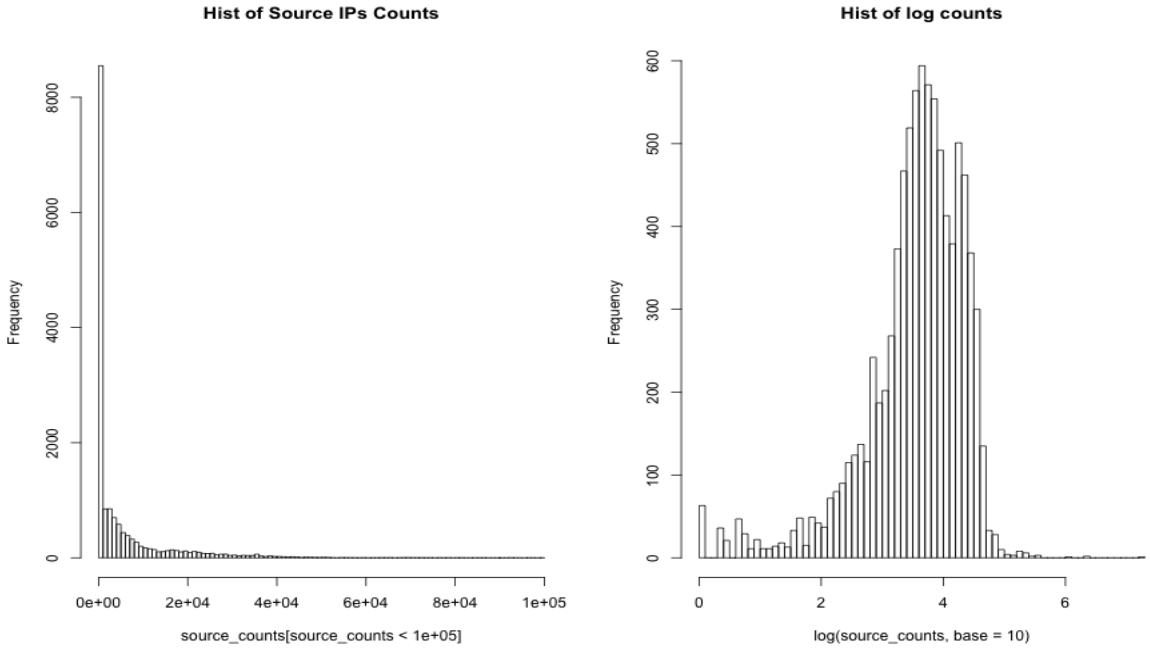


Figure 1.11: Histogram of Source IPs Counts - 1 day & College IPs

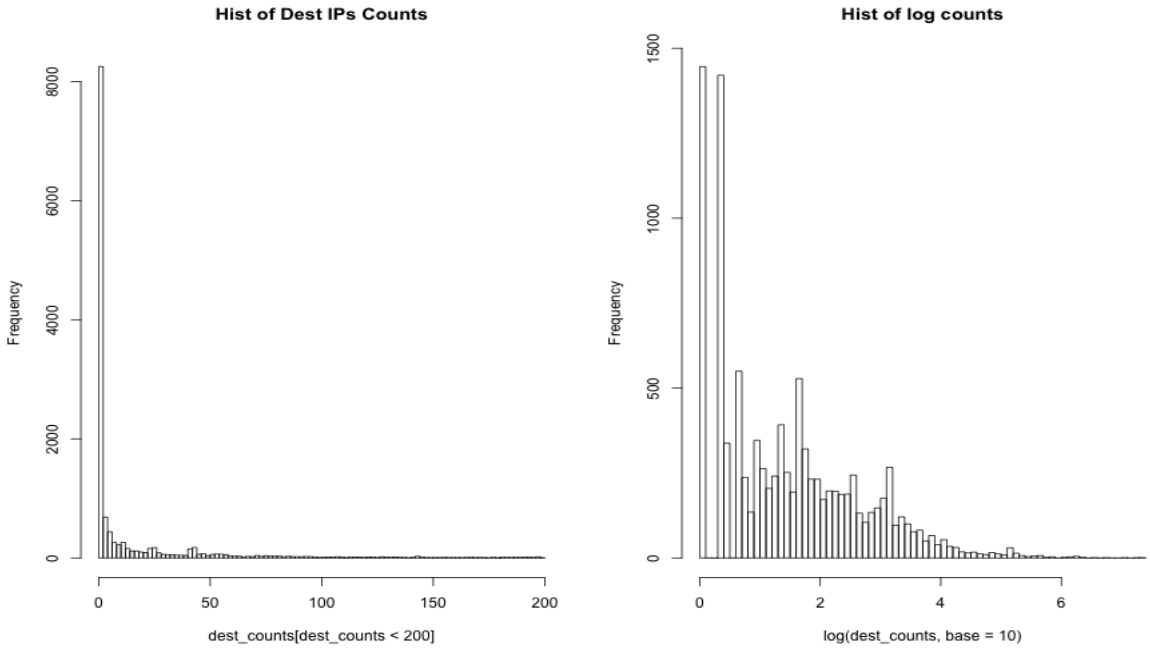


Figure 1.12: Histogram of Destination IPs Counts - 1 day & College IPs

	Edge Count	Indegree	Outdegree
Maximum	19408655	7245	2036
95%	1443	6	44
75%	136	2	24
Median	27	1	4
Mean	514.002	13.728	13.728
25 %	4	0	0
5%	1	0	0
Minimum	1	0	0
Total number of distinct IPs (Edges)	(217032)	15809	15809
Total number of counts	111554840	217032	217032

Table 1.4: Overview of Edge Count, Indegree and Outdegree

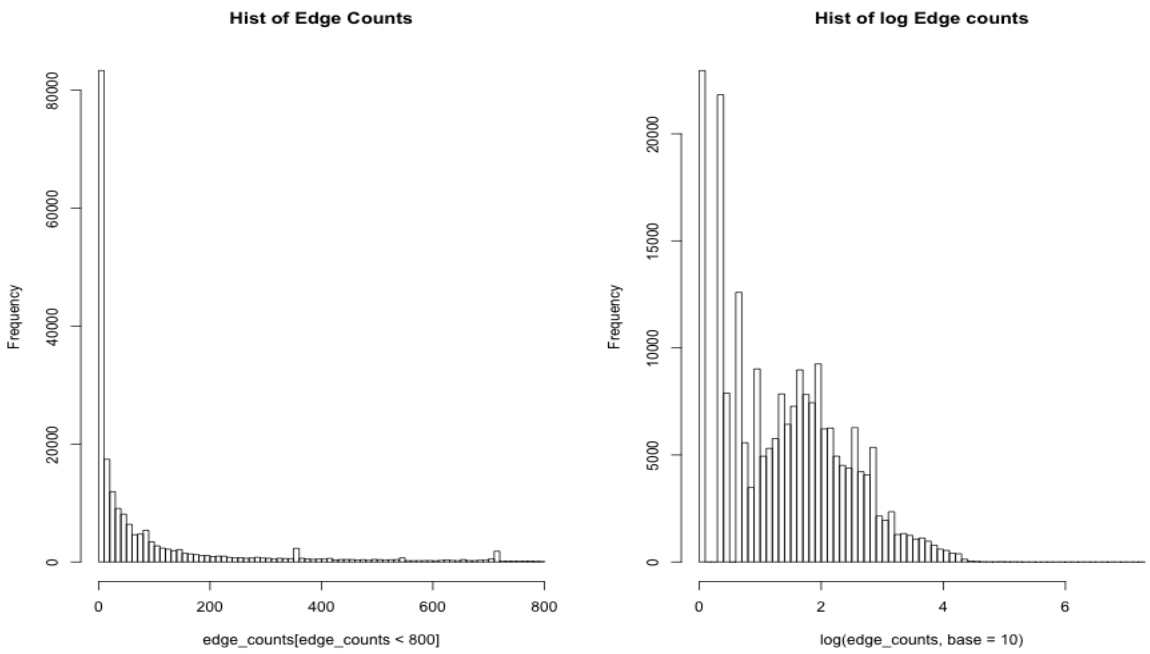


Figure 1.13: Histogram of Edge Counts - 1 day & College IPs

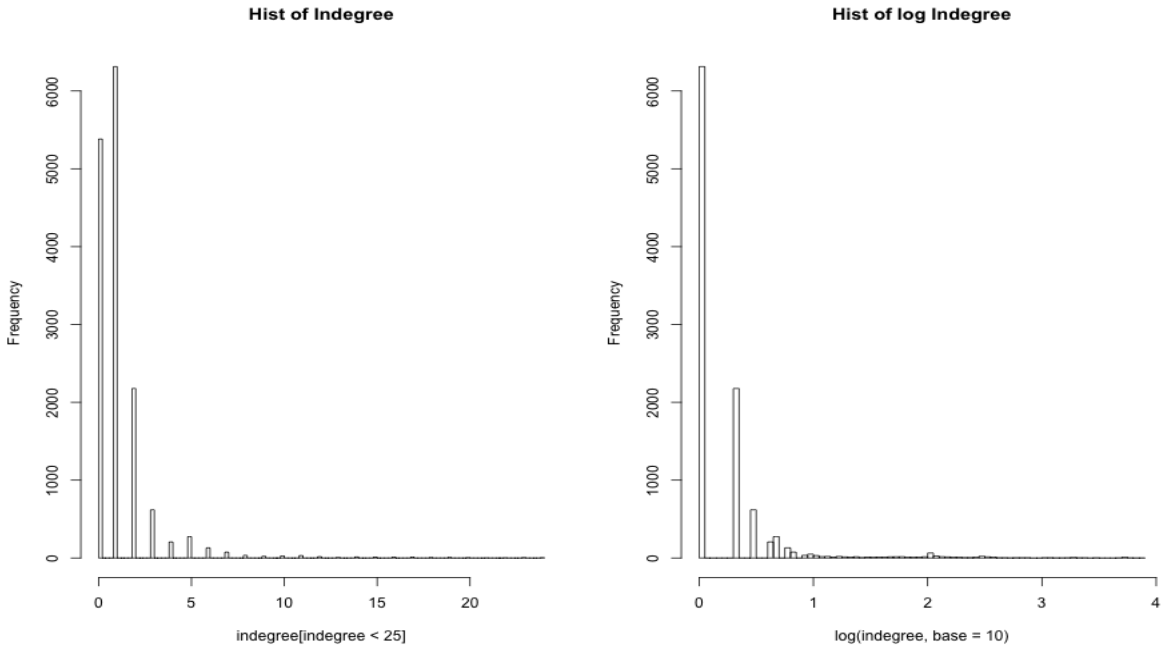


Figure 1.14: Histogram of Indegree- 1 day & College IPs

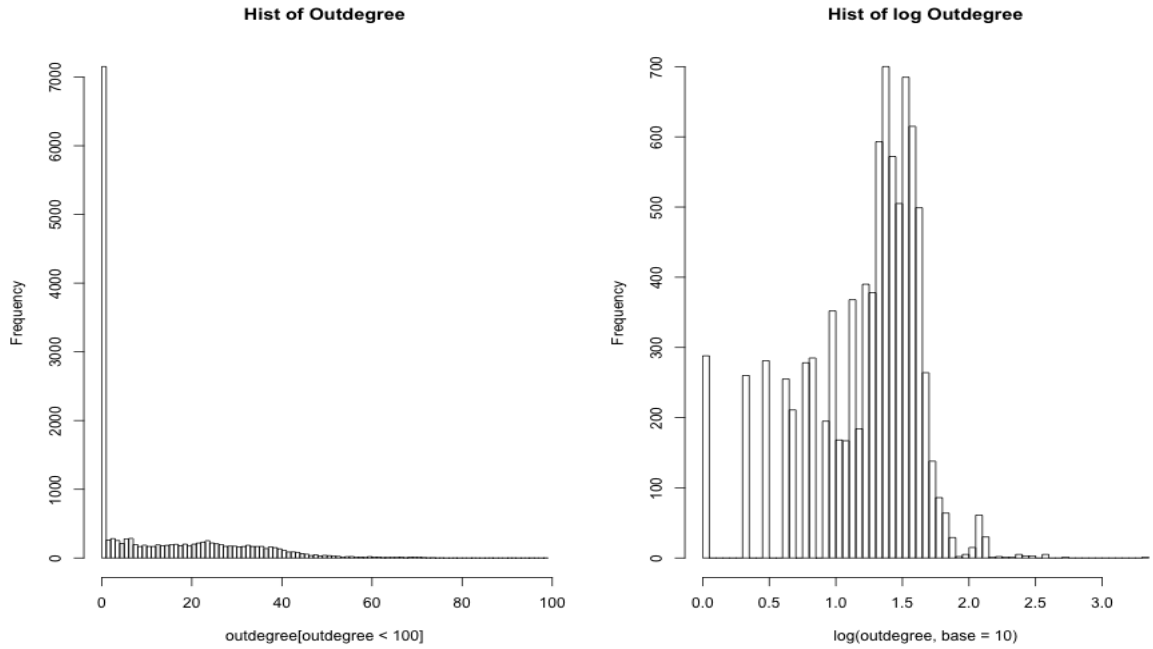


Figure 1.15: Histogram of Outdegree - 1 day & College IPs

1.4 College Network IPs - March

Finally, we look at the data at a longer time frame, i.e. one month, entire March.

See table 1.5 and figure 1.16 for table and histogram of all IPs Count data.

See table 1.5 and figure 1.17 for table and histogram of Source IPs Count data.

See table 1.5 and figure 1.18 for table and histogram of Destination IPs Count data.

See table 1.6 and figure 1.19 for table and histogram of Edge Counts.

See table 1.6 and figure 1.20 for table and histogram of Indegree.

See table 1.6 and figure 1.21 for table and histogram of Outdegree.

	All IPs Count	Source IPs Count	Dest. IPs Count
Maximum	522735282	438184282	522735112
95%	420843	350696.65	11317.95
75%	8880	1974.5	128
Median	55	0	14
Mean	147573.396	73786.698	73786.698
25 %	4	0	2
5%	1	0	0
Minimum	1	0	0
Total number of distinct IPs	41782	41782	41782
Total number of counts	6165911632	3082955816	3082955816

Table 1.5: Overview of All IPs Count

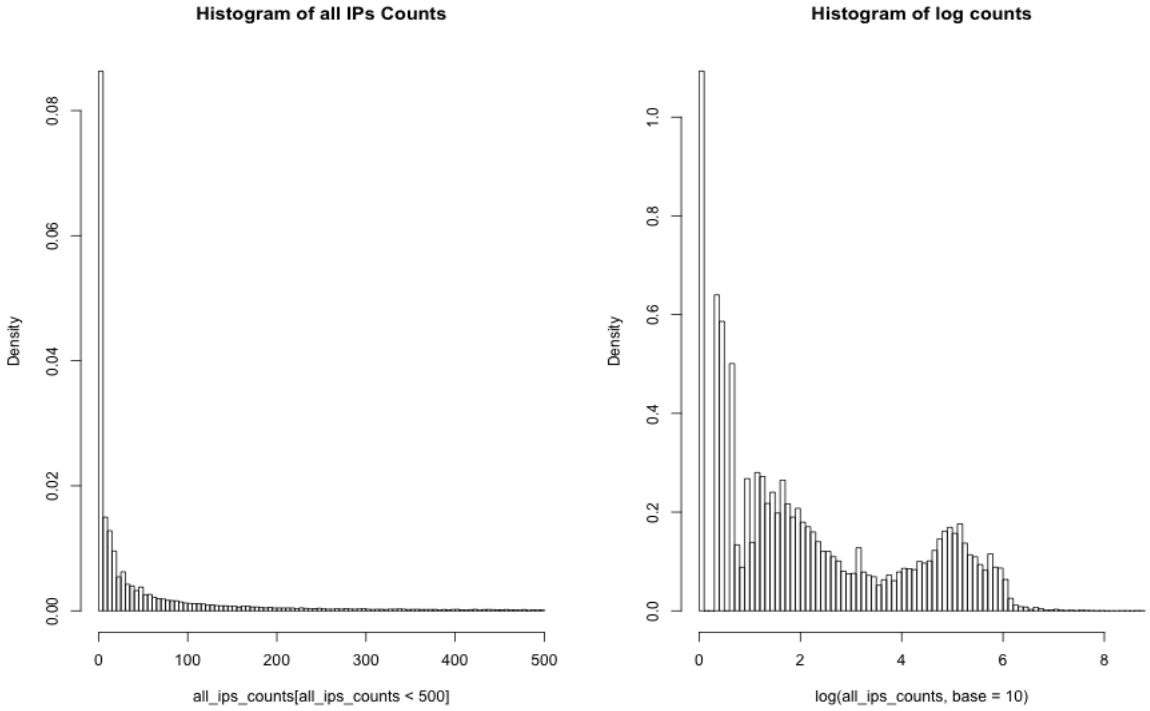


Figure 1.16: Histogram of All IPs Counts - 1 month & College IPs

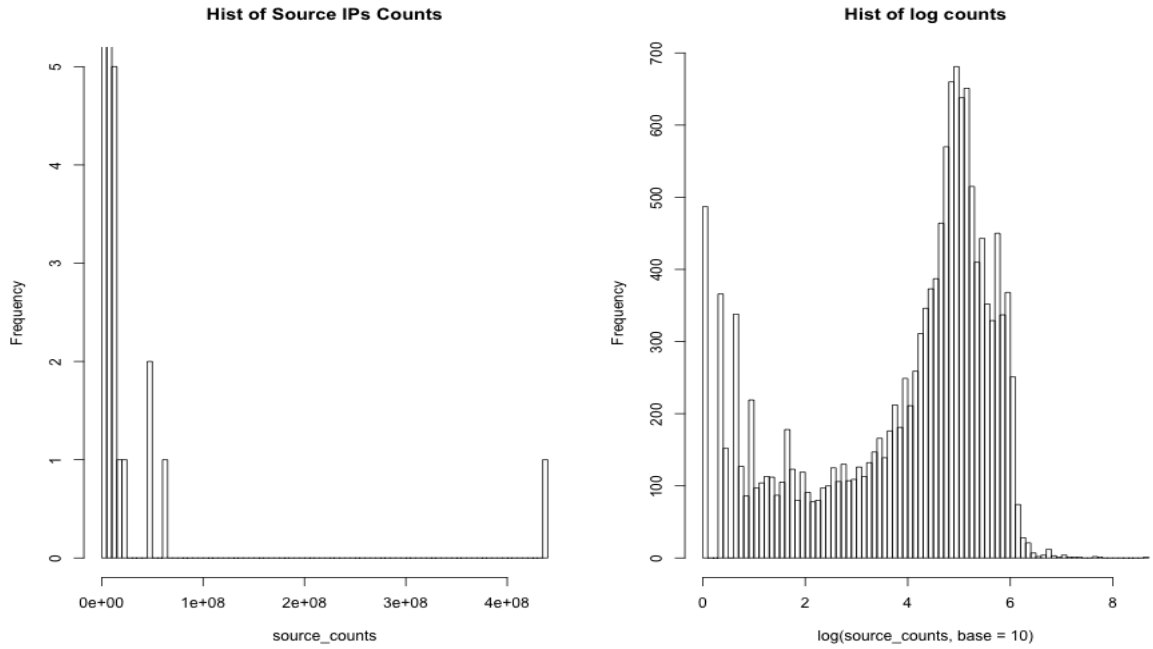


Figure 1.17: Histogram of Source IPs Counts - 1 month & College IPs

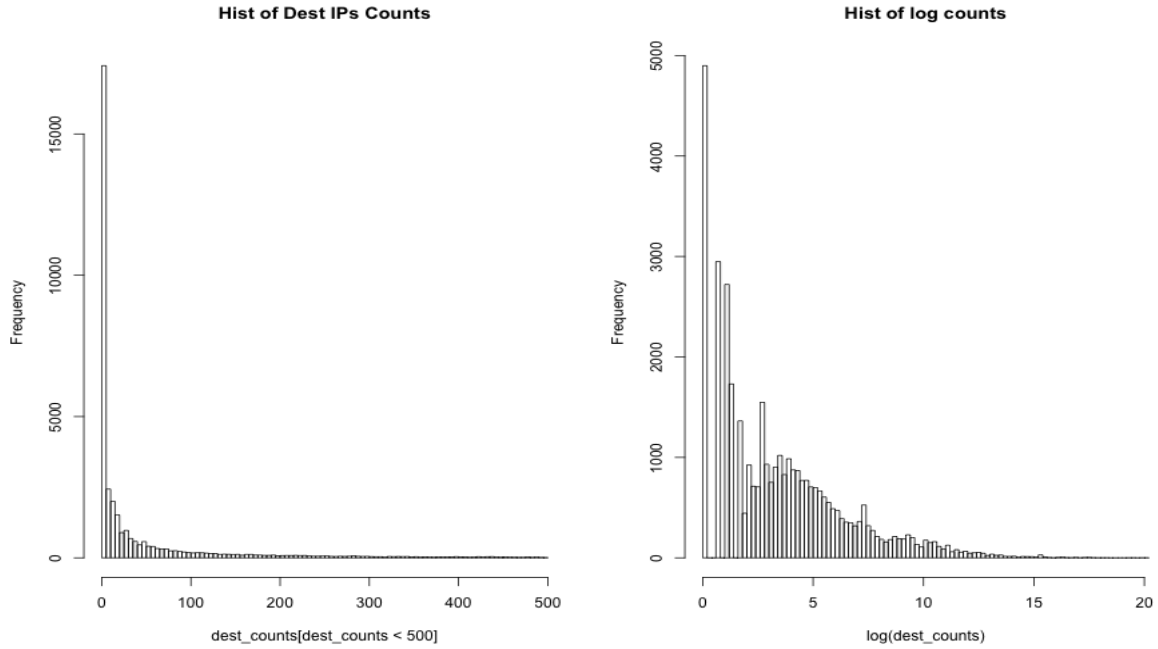


Figure 1.18: Histogram of Destination IPs Counts - 1 month & College IPs

	Edge Count	Indegree	Outdegree
Maximum	436987535	10450	6200
95%	12422.85	11	74
75%	547	4	9
Median	34	2	0
Mean	5153.859	14.317	14.317
25 %	6	1	0
5%	1	0	0
Minimum	1	0	0
Total number of distinct IPs (Edges)	(598184)	41782	41782
Total number of counts	3082955816	598184	598184

Table 1.6: Overview of Edge Count, Indegree and Outdegree

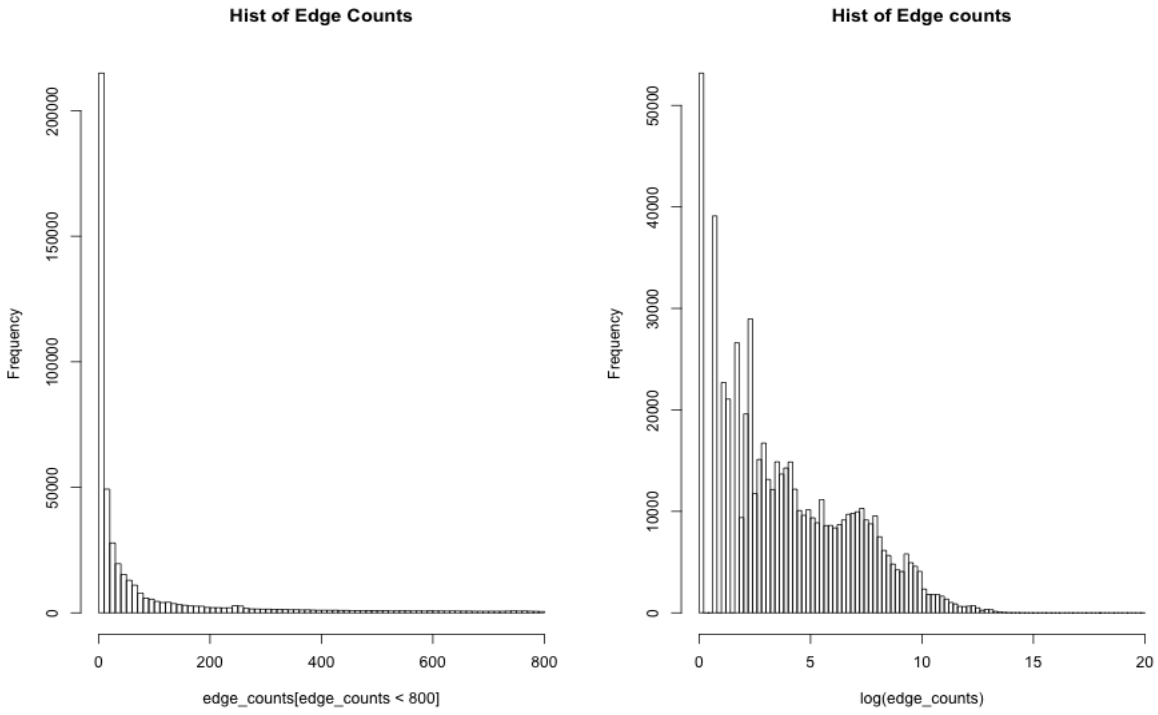


Figure 1.19: Histogram of Edge Counts - 1 month & College IPs

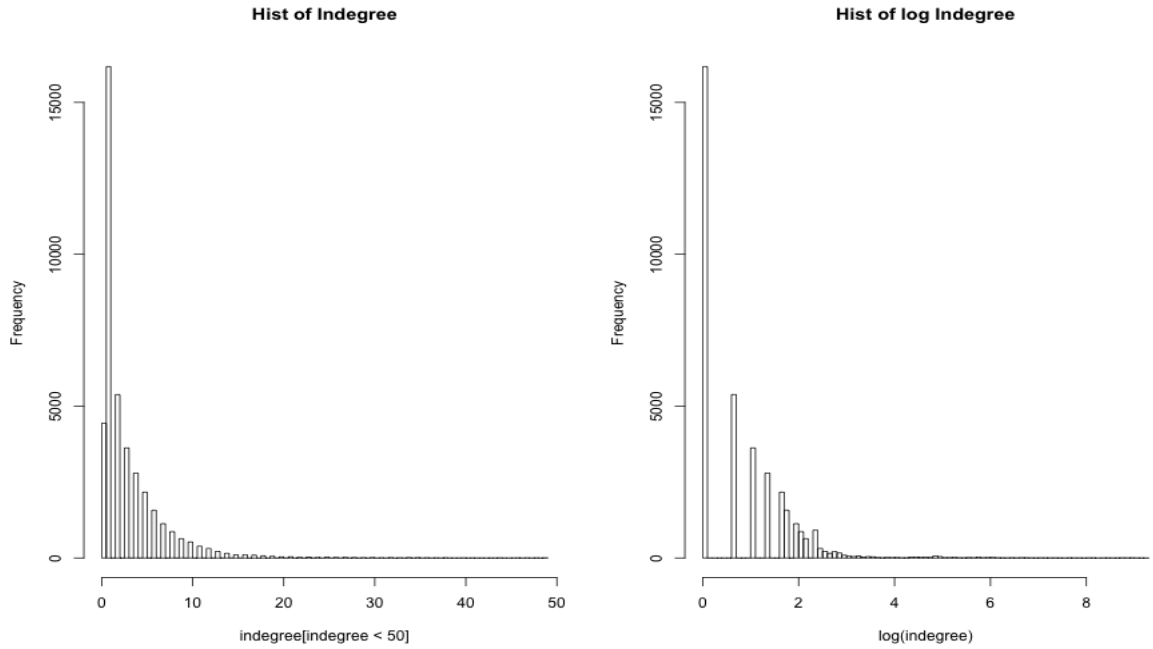


Figure 1.20: Histogram of Indegree - 1 month & College IPs

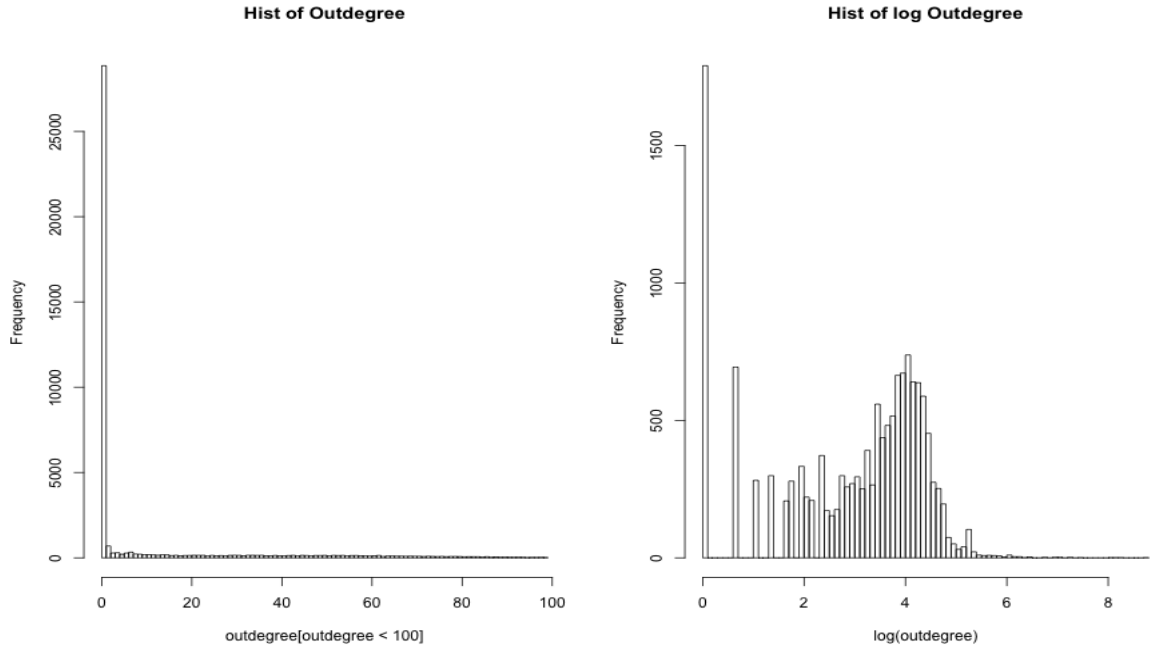


Figure 1.21: Histogram of Outdegree - 1 month & College IPs

Observations: It is evident that most of the distributions are positively skewed. From the 1 day college & non-college IP data, we observe that the source counts are mostly zeros(95%) and this is within our expectation, but the indegree have a high proportion of ones ($> 75\%$), which we should be suspicious about. This is a good opportunity to question the ICT department what are those distinct pairs of IPs with indegree one. High proportion ($> 50\%$) of ones in indegree also occur within the college network on the same day. Nonetheless, if we look at a month worth of data in March, we observe a much reduced proportion of ones in indegree. This is to some degree what we would expect from normally behaving IPs.

2

Modelling using graphs

2.1 Directed Network

In this chapter, we are interested in modelling the Edges: pairs of source IPs and destination IPs. Ultimately, we would like to construct a directed graph and decompose it into many smaller clusters. These smaller clusters may represent different groups of users within the network.

An edge is formed when there is a pair of source IP and destination IP. In this case, the edge is directed, since the source IP always point towards the destination IP.

To construct a directed graph, we used the "igraph" package, which incorporates many functions for plotting graphs and obtaining clusters.

We build a graph from all the college edges obtained in March.

```
IGRAPH DN-- 41782 598184 --
+ attr: name (v/c)
```

"D" indicates the kind of graph drawn, i.e. a Directed Network in this case. 41782 corresponds to the number of vertices in graph and 598184 corresponds to the number of edges.

We then compute the number ($no) of maximal weakly connected components of a graph and each component's size ($csize). See table 2.1. Basically, we build a bayesian network directly from the data and get a massive cluster with plenty of nodes in it and several clusters with only a few nodes in them. On the one hand, a cluster of size 41769 is massive and somewhat uninformative, because we cannot observe much from

Number of Cluster:	7						
Individual Cluster Size:	41769	2	3	2	2	2	2

Table 2.1: igraph output: 1 month, before filtering and thresholding, College IPs

a single graph with 41769 vertices. On the other hand, a cluster of size 2 or 3 is too small to explain the data. Therefore, some thresholding or filtering is required.

2.2 Filtering & Thresholding

The notion of filtering is to filter out a small part of the data without affecting too much the main structure of the data. It is believed that low values of edge count can be taken out, since they may represent occasional connections to random websites, and do not contribute much to help us identify different groups of users within the network. At the same time, nodes with high indegree or outdegree should be ignored, as we can safely assume that they are probably server IPs.

1. To achieve this, we first set a threshold for edge counts at the 5% quantile of the edge counts distribution and filter out edges with edge counts *lower* than 5% quantile of the distribution. .

2. Then, we set a threshold for indegree at the 95% quantile of the indegree distribution and filter out edges with indegree *greater* than the threshold. Repeat the same procedure on outdegree as well.

2.3 Filtering & Thresholding - March College IPs

In fact, the standard filtering at 5% quantile of edge counts distribution and 95% quantile of indegree and outdegree distribution does not work well here. It does not decompose the directed graph at all.

Nonetheless, given a month worth of data, we can be aggressive and filter out a little bit more data. So, we first set a threshold for edge counts at the 5% quantile (edges counts = 1). For indegree and outdegree, we set threshold at the 85% quantile of their distributions (indegree = 6, outdegree = 39). See histogram 2.3 2.4 2.5. Note that the red vertical line indicates the threshold level.

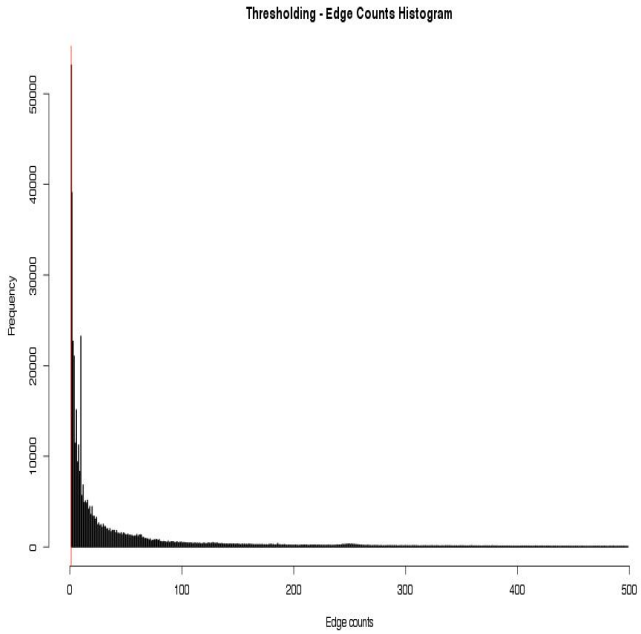


Figure 2.3: Filtering Edge Counts - 1 month College IPs

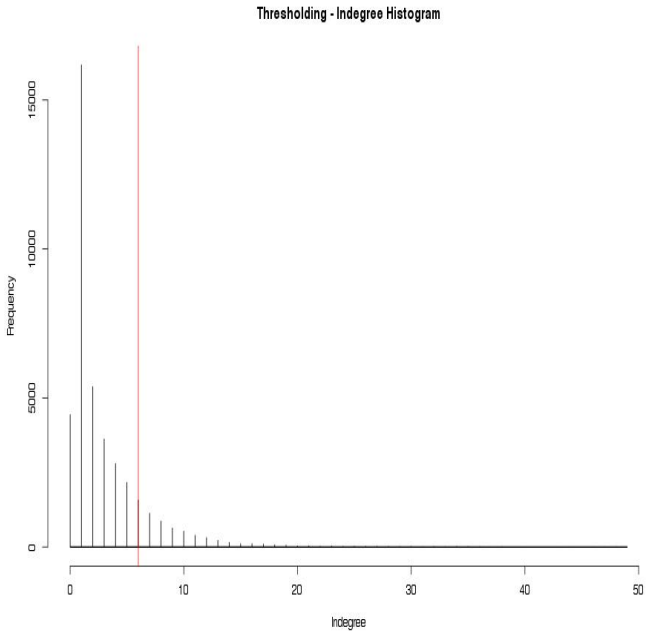


Figure 2.4: Filtering Indegree - 1 month College IPs

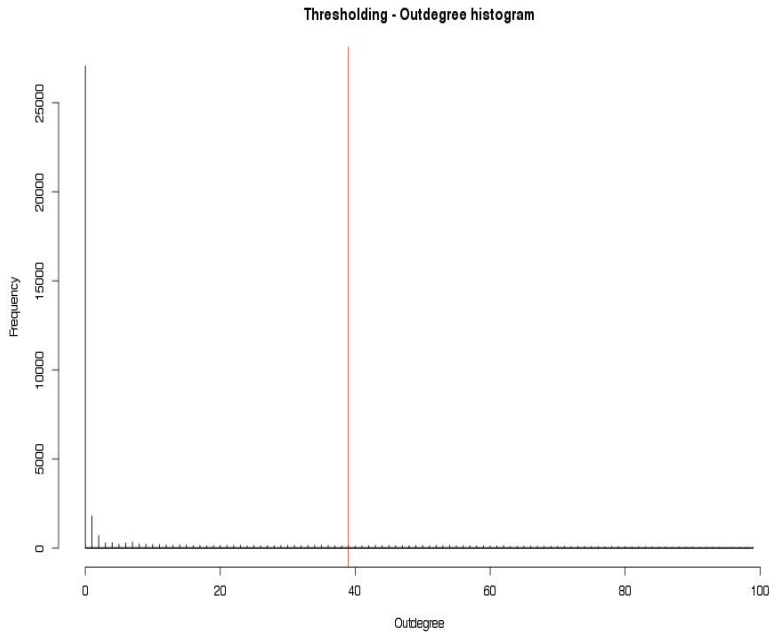


Figure 2.5: Filtering Outdegree - 1 month College IPs

After thresholding the edge counts, indegree and outdegree, a directed network is

Number of Cluster:	966						
Individual Cluster Size:	21	2	19	2	2	22	330
	12	3	11	5	35	67	56
	5	11	9	2	2	58	2
	23	13	17	18	58	40	7
	2	3	3	2	2	31	3.....

Table 2.2: igraph output: 1 month, after filtering and thresholding, College IPs

decomposed into many smaller components. See figure 2.6, 2.7 , 2.8. There are 966 clusters, the components are now nicer than those without thresholding or filtering. These components should be relatively easier to work with now than before. Part of the resulting clusters are in table 2.2:

Each component in 2.6, 2.7, 2.8 may correspond to different groups of users from different departments within College.

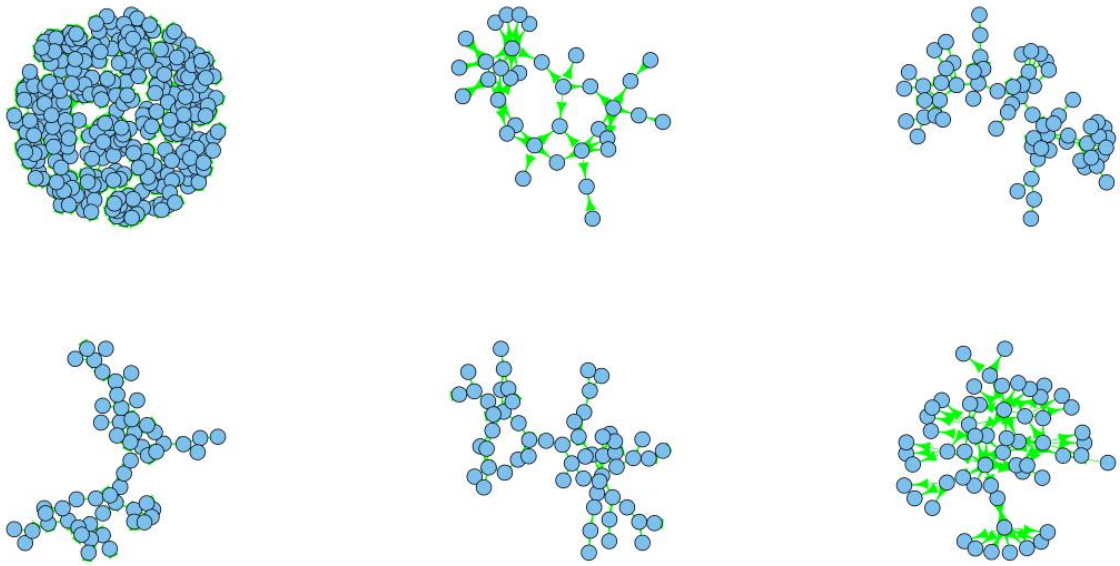


Figure 2.6: Directed Graph - 1 month College IPs

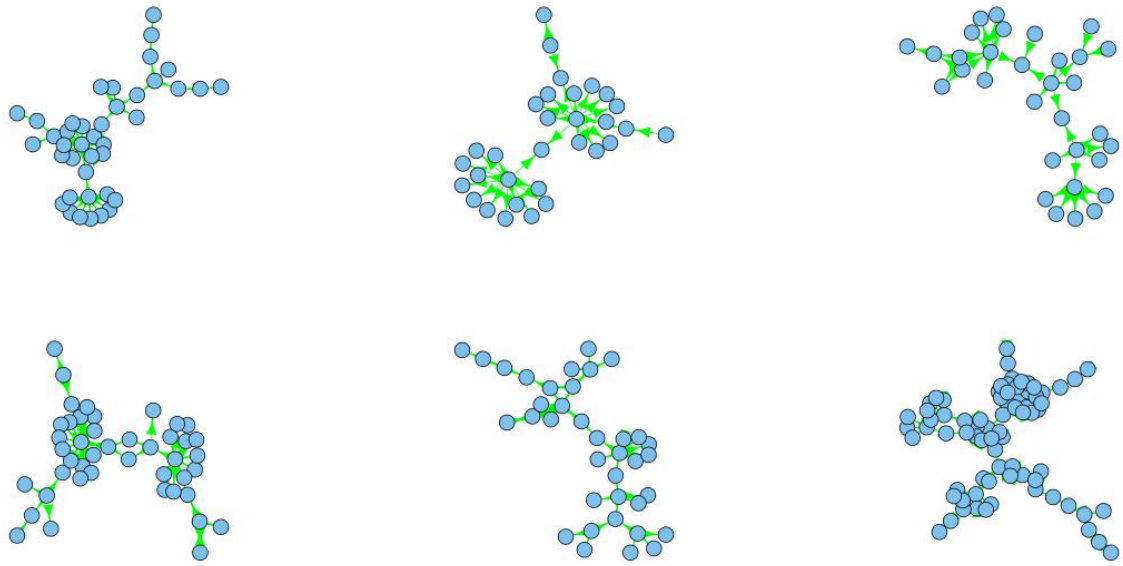


Figure 2.7: Directed Graph - 1 month College IPs

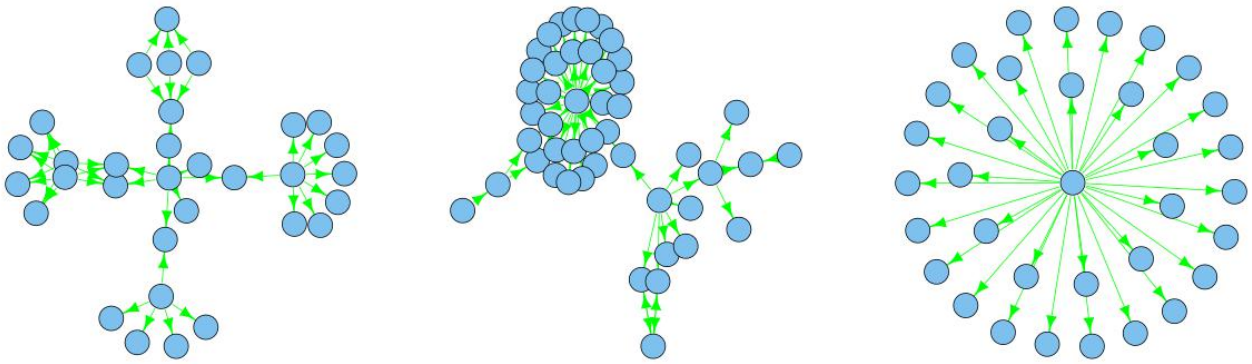


Figure 2.8: Directed Graph - 1 month College IPs

2.4 Filtering & Thresholding - 4^{th} March College IPs

Let's look at the college IPs only on the same day. Without any filtering at all, we have the output table 7.1, which is again uninformative. The filtering and thresholding process is shown in figure 7.1, 7.4, 7.5, in which the threshold for edge counts at 5% quantile is 1, the threhsold for indegree at 95% is 6 and threshold for outdegree at 95% is 44.

Results after filtering and thresholding are shown in table 7.2, figure 7.7, 7.8, 2.6

2.5 Filtering & Thresholding - 4^{th} March College & Non-CollegeIPs

We also build graphs from all the edges obtained on 4^{th} March, including all the college & non-college IPs. Again, without having done any filtering and thresholding, the clusters are difficult to model. See table 7.3 on appendix. See figure 7.11, 7.12, 7.13 for filtering and thresholding procedures, in which the threshold for edge counts at 5% quantile is 1, the threhsold for indegree at 95% is 1 and threshold for outdegree at 95% is 0. After filtering and thresholding, we have decent results. See table 7.4, and graph 7.14 and 7.15.

Remarks: It would be of great interest to carry on with some further analysis and classification at this point, to identify different group of users within the college network. However, due to privacy and security issues, the IPs are anonymised, which prohibits us from doing further analysis. Nonetheless, since working with college IPs plus non-college IPs involves massive amount of data, we would limit our interest in College IPs for the remaining chapters. This allows us to stick to the goal of anomaly detection within college network.

3

Discrete time modelling

3.1 Markov Chain Model

In this section, our goal is to perform anomaly detection in a discrete time monitoring framework. We first build a Markov Chain model to model minute counts of different IPs in March to gain an understanding of IP behaviour. We then try to examine the active counts by finding a distribution that fit and describe the data nicely. We illustrate the difficulty in fitting the data with simple *shifted* discrete distributions and mixture distributions. This leads us to use empirical distribution, which is the ideal model for active counts. The idea here is to train our model with active count data in the period 26^{th} Feburary -26^{th} March, then we set out to test our model upon active count data in the period 27^{th} March -2^{nd} April to see if the IPs' behaviour is significantly different from that in the training period. Now, it probably occurs to the reader that idea of *training and testing our model* is vague here. Please bear with us for now, as it will be discussed in depth later on. The issue of discrete p-values also arises in there and we explain how this can be tackled. Ultimately, we construct EWMA control chart to monitor IPs' behaviour in discrete time. At the end, we present the results of several different IPs.

We *assume* in here that the average weekday daily counts are very similar if not the same across time, which in turn justifies our focus on *minute counts*.

3.1.1 Sampling Data using Perl

The following briefly describes how time-series data are collected using Perl.

- Based on March's source IPs count data collected in chapter 1 and 2, a descending ordered list of IP address is created.

- Nodes are chosen selectively from the list to be analysed. Since source counts around the median levels are zero, we choose nodes with source counts at the 60^{th} quantile. The justification of why we choose not to work with IPs, with source counts at the median level, will be presented later. (Note that these source counts are based on "college $\rightarrow$ college" traffic.)

- We use perl to measure within each minute:

  - the number of outgoing connections a particular node makes with other nodes

  - the number of incoming connections other nodes make with the node of interest.

Note that the IPs of interest here are all college IPs and the data is collected from "college $\rightarrow$ non-college + college " traffic. Selecting nodes based on March source counts, which come from the "college $\rightarrow$ college" traffic, seems to create inconsistency. But this is the best we can do here because it takes tremendous amount of time (at least a month) to collect data of all IPs from "college $\rightarrow$ non-college + college " traffic. In addition, we will restrict our interest and concentrate in the outgoing connections of IPs, because we are more concerned about the client behaviour than server behaviour.

3.1.2 Markov Chain Model for Minute Counts

In this subsection, we study the activities of minute counts via a Markov Chain Model. As we mentioned at the end of Chapter 1, one of the main features of the minute count data is high proportion of zeroes. Therefore, it is natural to model the activity of a particular IP, say, i, by $Z_t^{(i)}$ as follows: Let $X_t^{(i)}$ be the number of outgoing connections of the IP i at the t^{th} minute. Then, at such time, i is in the active state $Z_t^{(i)} = 1$ if $X_t^{(i)} > 0$ and it is in the inactive state $Z_t^{(i)} = 0$ if $X_t^{(i)} = 0$. For the sake of presentation clarity, we also denote the active state by A and the inactive state by I.

We further assume $Z_t^{(i)}$ being a **homogeneous** two-state Markov Chain. That is,

1. $P\{Z_t^{(i)} = j_t | Z_{t-1}^{(i)} = j_{t-1}, Z_{t-2}^{(i)} = j_{t-2}, \cdots\} = P\{Z_t^{(i)} = j_t | Z_{t-1}^{(i)} = j_{t-1}\}$ where $j_s = 0$ or 1 for $s = t, t-1, \ldots$. [Given the full history of $Z_t^{(i)}$, it only depends on the most immediate history].

2. $P\{Z_t^{(i)} = j_t | Z_{t-1}^{(i)} = j_{t-1}\}$ is independent of t. [***time homogeneity***]

(4, Richard Durrett (1999), Essentials of Stochastic Processes , Springer text in Statistics).

Before dwelling any further into the Markov Chain model, we divide the time series data of each node into three separate time frames, such that we can monitor IP behaviour at discrete time:

- Night time $10:00pm - 8:00am$
- Weekday Daytime $08:00am - 10:00pm$
- Weekend Daytime $08:00am - 10:00pm$

These time segments are not arbitrariliy chosen. The cut between day time and night time is chosen as we observe there are relatively low or even zero activity after 10pm for most IPs, as shown in figure 3.1 3.2 7.16, which illustrates the normal behaviour we expect from the IPs.

The reason behind is that without classifying the discrete time series data into different time segments, the Markov Chains will be time-inhomogeneous and difficult to model. Therefore, by creating distinct time segments, we can simply assume the Markov Chains are *time-homogeneous* within those time periods. From now on, we can focus on homogeneous chains. Let $i, j \in S$. We can define transition probability from state i to state j as follow:

$$p(i,j) = P(Z_{t+1} = j | Z_t = i)$$

The transition matrix in Night time, Weekday and Weekend daytime are formulated as follow:

$$\begin{array}{c} \\ Inactive \\ Active \end{array} \begin{array}{c} \begin{array}{cc} Inactive & Active \end{array} \\ \begin{pmatrix} p(I,I) & p(I,A) \\ p(A,I) & p(A,A) \end{pmatrix} \end{array}$$

For instance, the transition probability of the node being active next minute given that it is inactive this minute is computed below in R.

$$p(I,A) = P(Z_{t+1} = A | Z_t = I) = \frac{P(Z_{t+1} = A, Z_t = I)}{P(Z_t = I)} = \frac{\#\ of\ \{Z_{t+1} = A \cap Z_t = I\}}{\#\ of \{Z_t = I\}}$$

Note that the rows of the transition matrix must sum up to 1. A normal IP i.e. 100.253.169.162 would have the transition matrices below and figure 3.1 basically shows the density of the IP's March minute count by days and by weeks, which help us to understand the IP behaviour. To interpret the "density plot of count by days", we read off from the x-axis, for instance, if the time is $8:17am$, then y-axis represents the group of all the counts from March, that occur at $8:17am$, in a density representation.

Night time

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 0.99 & 0.01 \\ 0.22 & 0.78 \end{pmatrix} \end{array}$$

Weekday Daytime

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 0.84 & 0.16 \\ 0.33 & 0.67 \end{pmatrix} \end{array}$$

Weekend Daytime

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 0.99 & 0.01 \\ 0.41 & 0.59 \end{pmatrix} \end{array}$$

, where I & A refer to the *inactive* and *active* state. We expect a normal IP should have fairly low or zero activity during night time and weekend compared to weekday daytime. So, we would expect the transition probability of an IP being active in the next minute given that it is inactive this minute ,i.e. $P(I, A)$, to be lower during night time and weekend daytime than during weekday daytime.

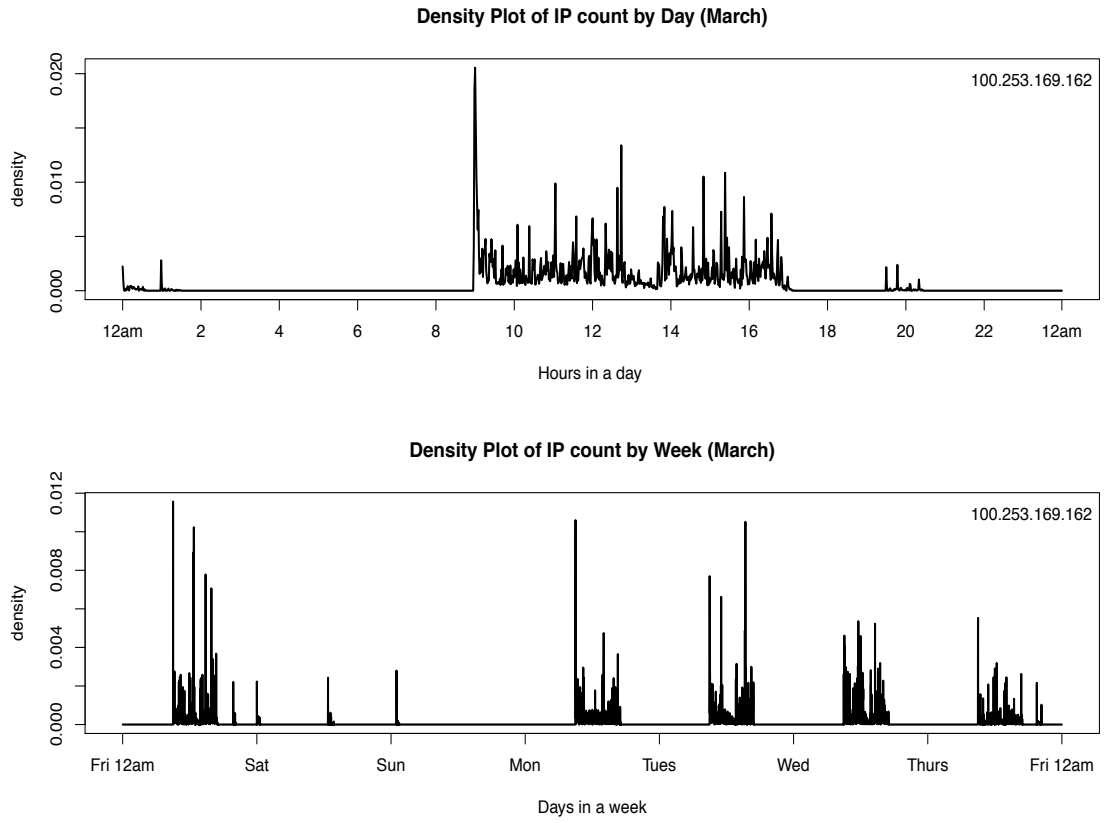


Figure 3.1: 100.253.169.162

Other example IPs, i.e. 100.253.170.35 and 126.253.151.115 with transition matrices are displayed below together with their density plots . The "−" simply means the particular IP is inactive in that period of time. In figure 3.2 , 7.16 (in Appendix), the density plots illustrate that there are some patterns within a day and between days. We tend to see, from normal-behaving IPs, high level of activities during daytime between $8am$ and $10pm$ on weekdays and low activity level during night time and weekends. This also justifies why and how we choose the time segments for our model.

	Night time		Weekday Daytime		Weekend Daytime	
	I	A	I	A	I	A
100.253.170.35	I 1.00	0.00	I 0.93	0.07	I 1.00	0.00
	A −	−	A 0.16	0.84	A −	−

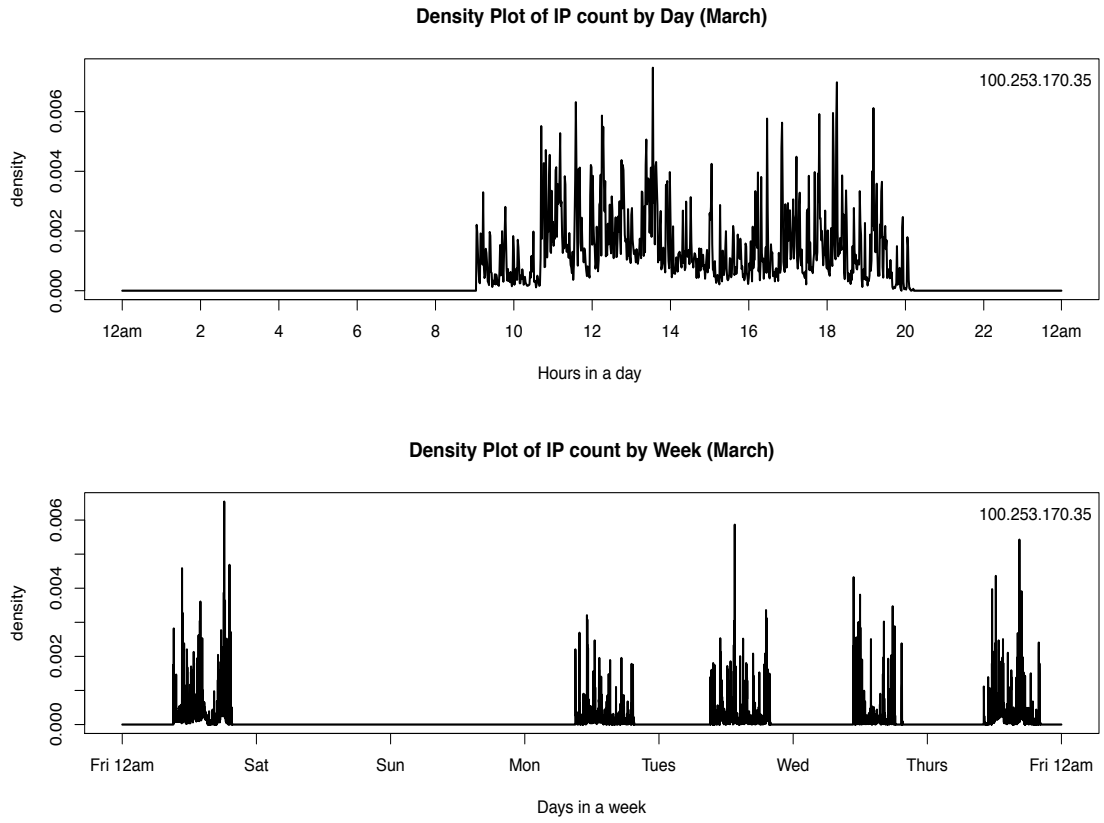


Figure 3.2: 100.253.170.35

Here we address why we choose not to look at IPs with median source counts. Although we think that IPs with median source counts should be regular college users, this turns out to be untrue. We collected data of outgoing connections of 100 IPs. We display one of these IPs' transition matrices below. As shown in figure 3.3, this IP shows very little and sometimes even zero activity on weekdays, weekends and night time. This justifies looking at IPs with source counts beyond the median level.

$$\begin{array}{cc} & \text{Night time} \\ & \begin{array}{cc} I & A \end{array} \\ \text{IP 1} & \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.99 & 0.01 \\ 0.06 & 0.94 \end{pmatrix} \end{array} \qquad \begin{array}{cc} & \text{Weekday Daytime} \\ & \begin{array}{cc} I & A \end{array} \\ & \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.99 & 0.01 \\ - & - \end{pmatrix} \end{array} \qquad \begin{array}{cc} & \text{Weekend Daytime} \\ & \begin{array}{cc} I & A \end{array} \\ & \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 1.00 & 0.00 \\ - & - \end{pmatrix} \end{array}$$

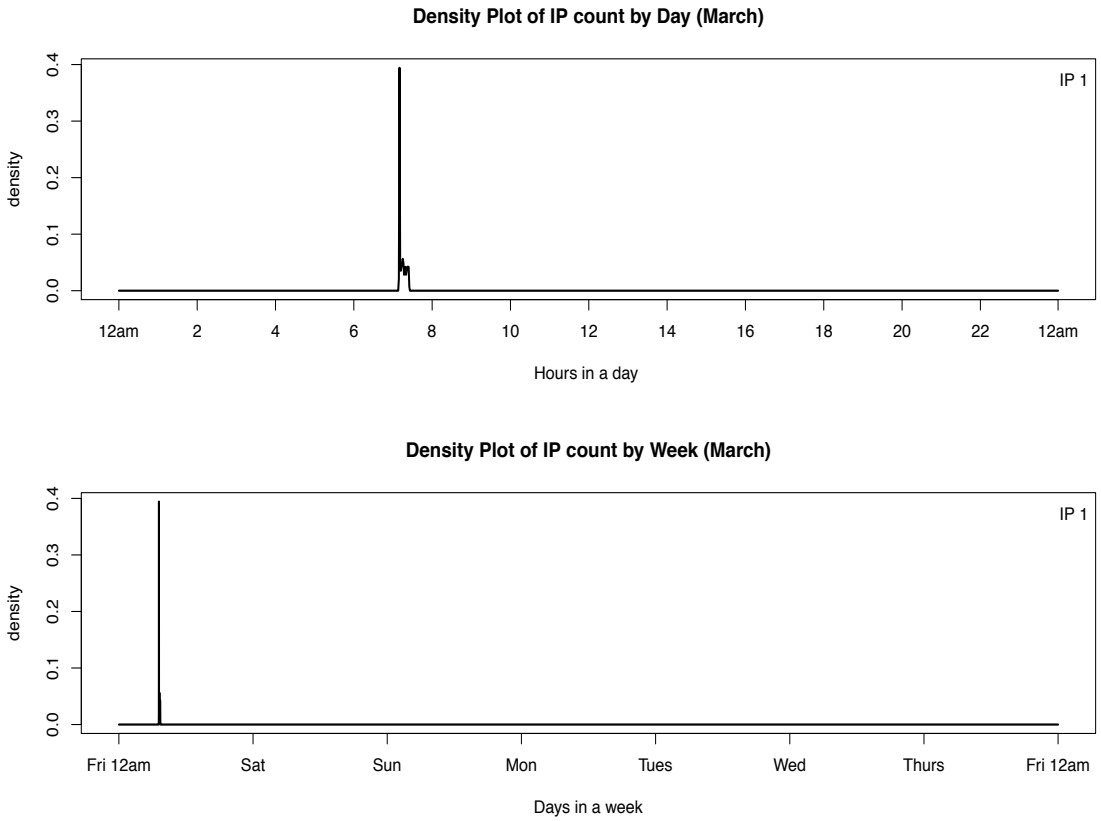


Figure 3.3: IP1

3.2 Minimum Chi-Square Estimation

After dividing up the time series data and computing the transition matrices for different time periods, we are now in a good position to model the *active* counts distribution and thereby, allowing us to understand the normal behaviour of each IP address at different time period. Due to limited data for weekend daytime active counts, we concentrate on modelling active counts at a) nightime and b) daytime during weekdays only.

The majority of the distribution of IPs' active counts are right-skewed. Our first attempt is to perform minimum chi-square estimation such that we find the parameter estimate of the chosen simple discrete distribution that will minimise the chi-squared statistic and fit the distribution to data. As we are looking at active counts only, the support is $\{1, ...\infty\}$ and thus, *shifted* discrete distributions are considered. The probability mass function of a shifted Poisson distribution with parameter λ is as follow:

$$P(X = x) = \begin{cases} \frac{\lambda^{x-1}e^{-\lambda}}{(x-1)!}, & \text{for } x = 1, 2, 3, ... \\ 0, & \text{otherwise} \end{cases}$$

In fact, shifted distributions can be easily dealth with by simply minusing 1 from all the active counts, which in turn bring us back to simple discrete distributions.

The methodology of minimum chi square estimation is outlined below:

Suppose we have the following frequency table of active counts (after combining classes)

Observed Frequency	O_0	O_1	$\cdots$	O_{j-1}	O_j
Expected Frequency	E_0	E_1	$\cdots$	E_{j-1}	E_j

Let $\sum_{i=0}^{j} O_i = N$. Our interest is to test if the distribution O_i are generated from $X \sim Poission(\lambda)$.

$$E_i = N \times \Pr\{X = i\} \text{ for } i = 0, \ldots, j-1$$

and class j is obtained by combining classes

$$E_j = N \times \Pr\{X \geq j\}$$

Note that the choice of j is crucial. The hypotheses of the Chi-Squared Goodness of fit test are:

H_0 : *Data* is from a Poisson Distribution VS H_1 : *Data* is not from a Poisson disrtibution.

The minimum chi-square estimation finds the parameter λ to minimise the Pearson chi-square test statistic:

$$\min_{\lambda} \sum_{k} \frac{(O_i - E_i)^2}{E_i} \sim \chi^2(k - p - 1)$$

,where k is the number of classes remaining after combining classes and p is the number of parameters estimated from the data (1, Mark L. Berenson (2005)). And $p = 1$ in this case.

Note that a common pitfall is to plug in the maximum likelihood estimate from original observation, in which case the test statistic does not have a limiting χ^2 distribution (2, Herman Chernoff , E.L. Lehmann, 1954):

$$\sum_{k} \frac{(O_i - \hat{E}_i)^2}{\hat{E}_i}$$

where $\hat{E}_i = N \times \widehat{\Pr}\{X = i\} = N \times e^{-\hat{\lambda}} \hat{\lambda}^i / i!$ for $i = 0, \ldots, j-1$ and $\hat{\lambda}$ is the MLE from the original observation. Also, $\hat{E}_j = N \times \Pr\{X \geq j\}$.

Figure 3.4 and 3.5 show the distribution of active counts during night time, weekday daytime and weekend daytime of various IPs respectively. Although it looks like some of discrete distributions fit the data quite well, all the chi-squared goodness-of-fit tests return p-values significantly less than 0.5. We reject the null hypothesis and hence, none of the simple shifted discrete distributions fit the data.

Looking closely at the topright corner in 3.5, it looks like that the active count data follow a shifted poisson distribution (red line), but the chi-squared goodness-of-fit test suggests otherwise. In fact, there is a second mode at count level 26.

Multiple modes motivate us to use discrete mixture distributions with parameters estimated by the *EM algorithm*.

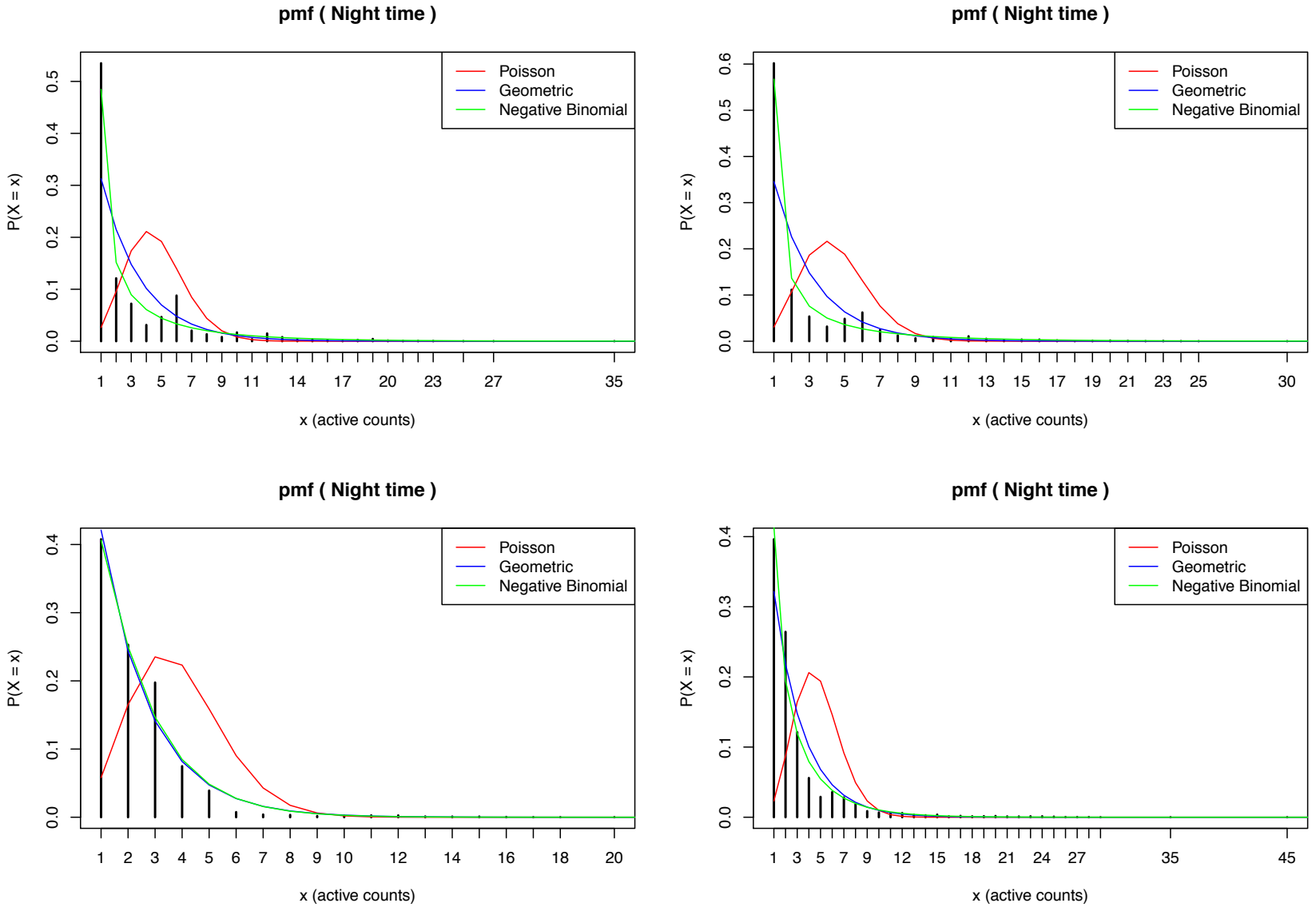


Figure 3.4: IPs Night time behaviour

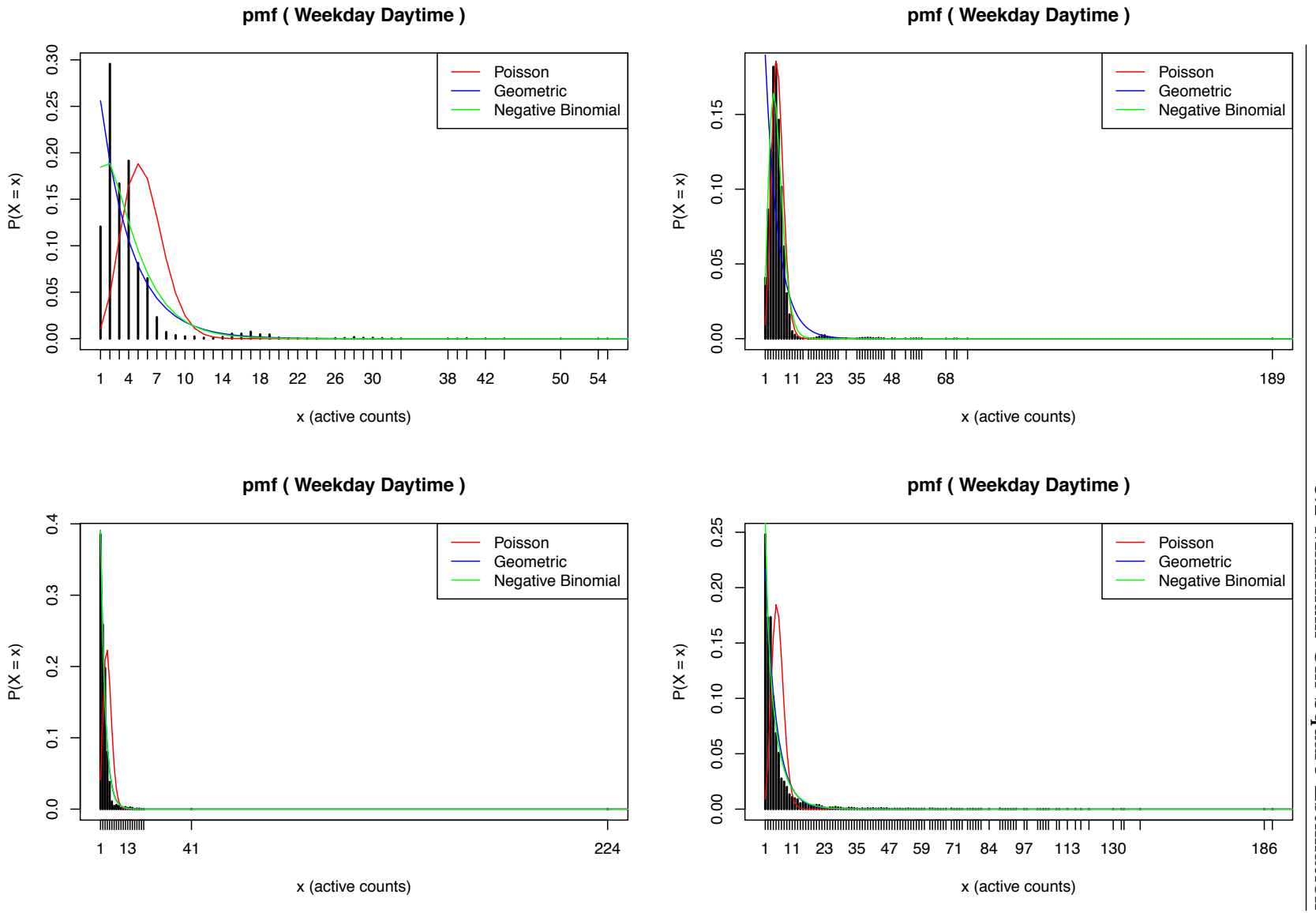


Figure 3.5: IPs Weekday Daytime behaviour

3.3 EM-algorithm

Here, we try to fit a poisson mixture distribution and a poisson-geometric mixture distribution.

3.3.1 Poisson Mixture Distribution

Given the active count data of a particular IP address $\{x_1, x_2, \cdots, x_m\}$, we define the latent variable $z_i \in \{1, \cdots, k\}$, for $i = 1, \ldots, m$, where k denotes the number of mixture components and m the size of the data. Here, $z_i \sim Multinomial(\boldsymbol{\phi})$, where $\phi_j \geq 0, \sum_{j=1}^{k} \phi_j = 1$

$$P(z_i|\boldsymbol{\phi}) = \prod_{j=1}^{k} \phi_j^{\mathbb{1}\{z_i=j\}}$$

And if z_i belongs to the j^{th} mixture component, $P(z_i = j) = \phi_j$. We are interested in:

$$x_i|z_i \sim Poi(\lambda_j)$$

Let $\boldsymbol{\theta} = (\boldsymbol{\lambda}, \boldsymbol{\phi})$ and $f(x_i; \lambda_j) = e^{-\lambda_j}\lambda_j^{x_i}/x_i!$ denotes the poisson probability mass function with parameter λ_j of x_i.

1. The **JOINT** log-likelihood:

$$l(\boldsymbol{\phi}, \boldsymbol{\lambda}|\boldsymbol{x}, \boldsymbol{z}) = \log \prod_{i=1}^{m}\prod_{j=1}^{k} P(x_i|z_i; \lambda_j)^{\mathbb{1}\{z_i=j\}} P(z_i = j)^{\mathbb{1}\{z_i=j\}} = \sum_{i=1}^{m}\sum_{j=1}^{k} \log\left[f(x_i; \lambda_j)\phi_j\right]^{\mathbb{1}\{z_i=j\}}$$

2. The E-step basically constructs lower bound for the log-likelihood function at the current value of $\boldsymbol{\theta}^{(t)}$.

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = E\left[l(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{z})|\boldsymbol{x}, \boldsymbol{\theta^{(t)}}\right] = \sum_{i=1}^{m}\sum_{j=1}^{k} E\left[\mathbb{1}\{z_i = j\}|\boldsymbol{x}\right] \log\left[\phi_j f(x_i; \lambda_j^{(t)})\right]$$

3. note that by Bayes' rule, $E\left[\mathbb{1}\{z_i = j\}|\boldsymbol{x}\right]$ as follow:

$$w_j^i = E\left[\mathbb{1}\{z_i = j\}|\boldsymbol{x}\right] = p(z_i = j|x) = \frac{P(x_i|z_i = j)P(z_i = j)}{\sum_{l=1}^{k} P(x_i|z_i = l)P(z_i = l)} = \frac{\phi_j f(x_i; \lambda_j)}{\sum_{l=1}^{k} \phi_l f(x_i; \lambda_j)}$$

which is our first updating equation.

4. The M-step maximises the lower bound $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$ with respect to λ_j and ϕ_j. Maximising it w.r.t. λ_j gives the second updating equation:

$$\lambda_j^{(t+1)} = \arg\max_{\lambda_j} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \Longrightarrow \lambda_j = \frac{\sum_{i=1}^{m} x_i w_j^i}{\sum_{i=1}^{m} w_j^i}$$

Note: there is an additional constraint $\sum_{l=1}^{k} \phi_l = 1$. To maximise $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$ with respect to ϕ_j, we need to construct a Lagrangian

$$\mathcal{L}(\phi) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_j^i \log \phi_j^i + \beta \left(\sum_{j=1}^{k} \phi_j - 1 \right)$$

,where β is the Lagrange multipler. And the third updating equation:

$$\phi_j^{(t+1)} = \arg\max_{\phi_j} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \Longrightarrow \phi_j = \frac{\sum_{i=1}^{m} w_j^i}{m}$$

Following similar procedures , we can also derive the updating equations of a Poisson-Geometric mixture distribution. See 7.1.3 in appendix.

Technically speaking, the mixture distributions mentioned above need to be shifted to the right by 1 unit, since we are focusing on active counts. However, we can simply minus one from the data vector and get back to simple framework.

We fit poisson mixture distributions and poisson geometric mixture distributions to the data in figure 3.6, 3.7. Note that the IPs in figure 3.6, 3.7 are the same IPs in figure 3.5, 3.4, in which we use simple discrete distributions.

Figures 3.6, 3.7 illustrate that the mixture distributions seem to perform better than the simple discrete distributions. However, multiple modes still remain as an issue, as mixture distributions fail to fit those modes. Unfortunately, we cannot do a chi-squared goodness of fit test, as we mentioned earlier that plugging the maximum likelihood estimates into the chi-squared test statistic is a common pitfall that a lot of people fall into. Nonetheless, we are going to illustrate in the next subsection that modelling active counts with mixture distributions will fail to train our model, as their tail probabilities (p-values) computed from the training data are not uniformly distributed. This motivates us to use empirical distribution to estimate the distribution function of the data.

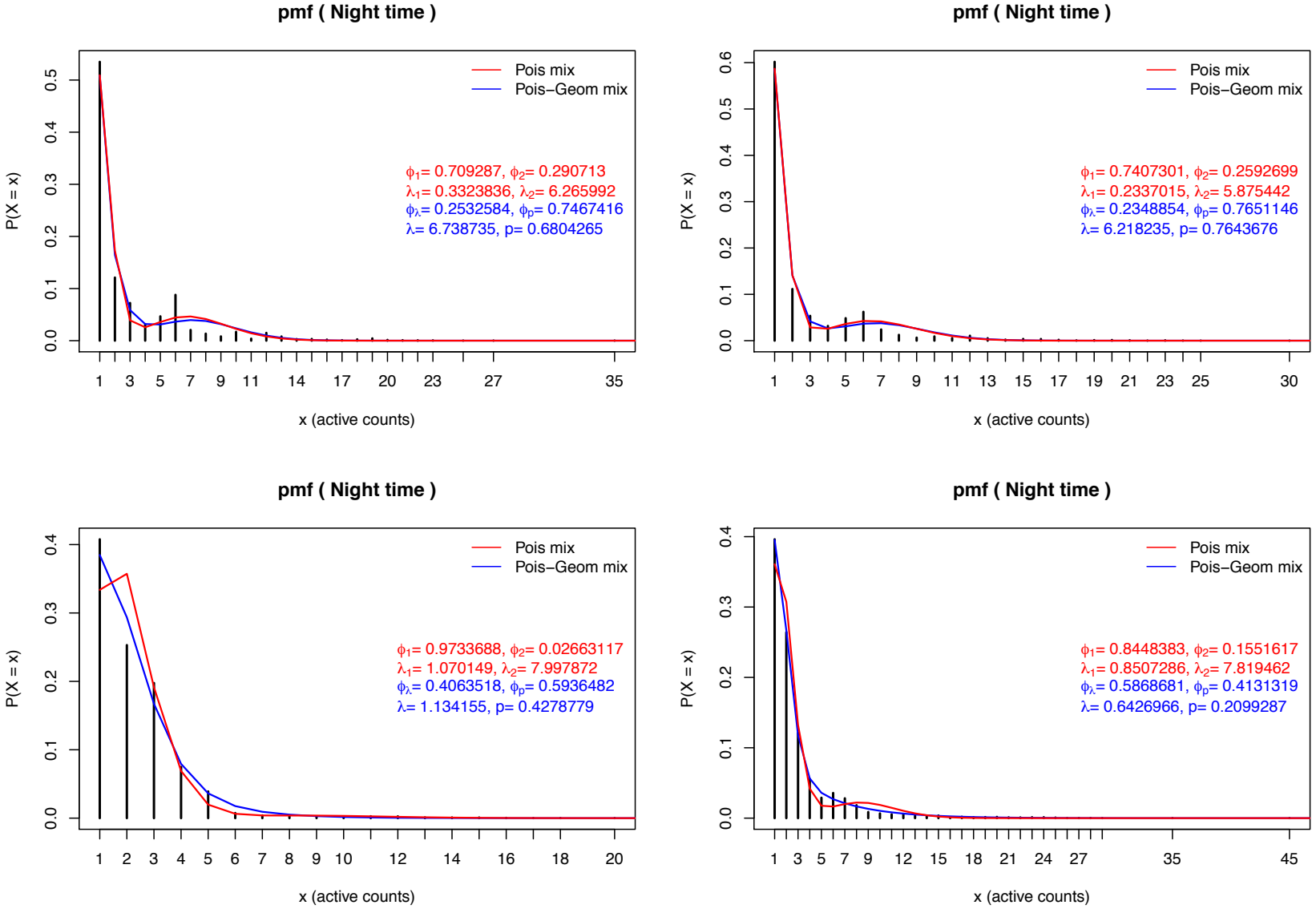


Figure 3.6: IPs Night time behaviour

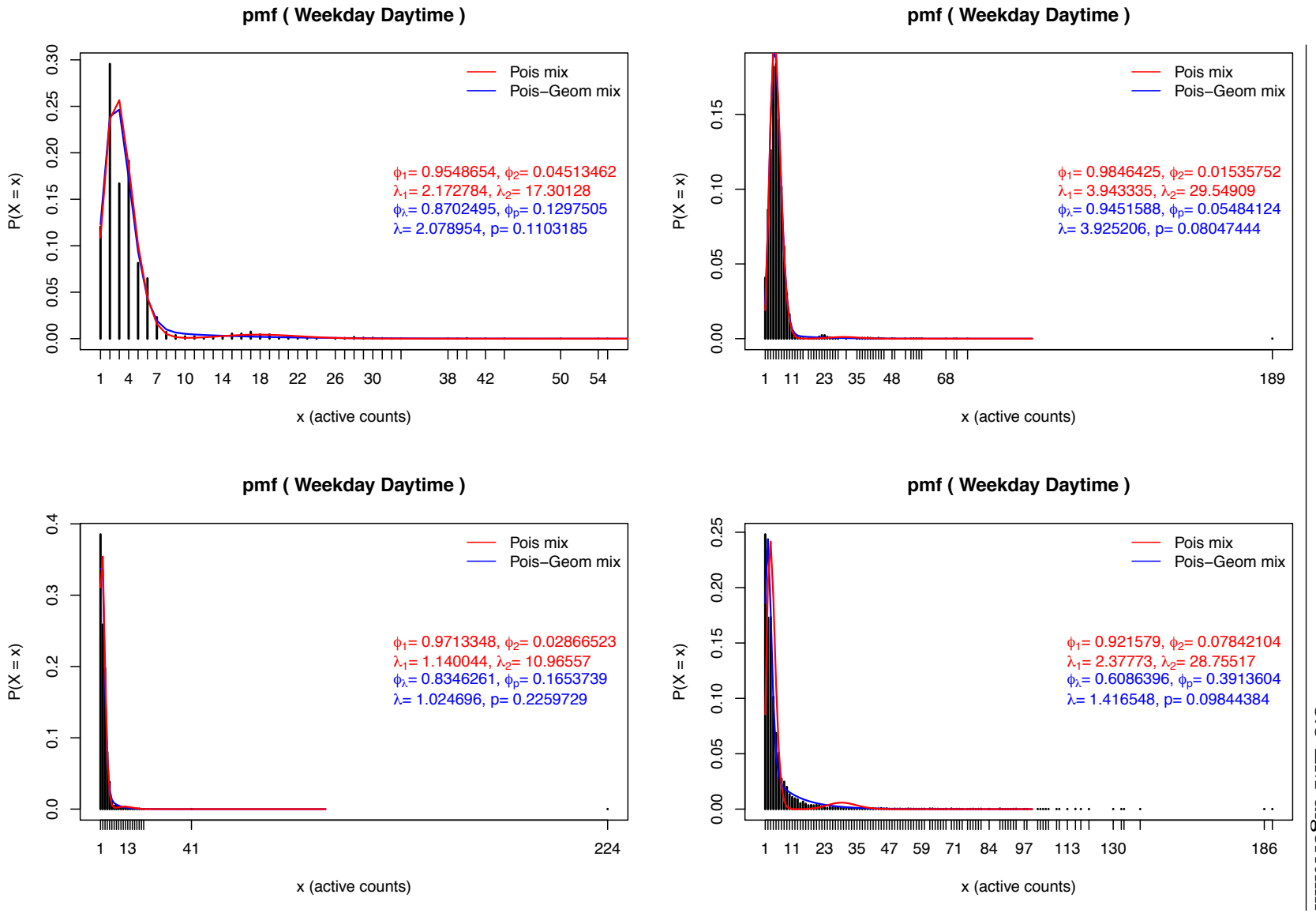


Figure 3.7: IPs Weekday Daytime behaviour

3.4 Training

So far we have touched upon several times the idea of *training the model*. And it was vaguely explained. Here, we are going to elaborate this notion. First, we need to understand what a p-value is. The standard definition of p-value is the probability, computed assuming that the **null hypothesis** is true, that the test statistic would take a value as extreme or more extreme than that actually observed. (12, Duncan J Murdoch 2008). The paper highlights that p-value is a random variable and it has a Uniform(0,1) distribution under the **simple** null hypothesis. We address what the **simple** null hypothesis is under a hypothesis testing framework in the appendix (this is meant for avid readers only).

It is worth noting here that an important assumption is made when training our model. We are assuming that there is no anomalous behaviour in the training period for the IP of interest.

To explain what training does, the following model is presented:
The model assumes: $X_1, \ldots, X_m$ are independent, identically distributed (i.i.d) from a distribution function $F_\theta(x)$, where X_i is the active count of the i^{th} minute from the training period and $F_\theta(x)$ is the proposed distribution function, where θ is the vector of parameters.

- That is, for Poisson distribution $\theta = \lambda$ and for Poisson mixture distribution $\theta = (\lambda_1, \ldots, \lambda_k, \phi_1, \ldots, \phi_{k-1})$

The hypothesis test is as follow:

$$H_0 : X_i \sim F_\theta \quad \text{vs} \quad H_1 : X_i \text{ is not distributed as } F_\theta$$

If the parameter θ is known, then we compute the following upper tail probabilities (or upper tail p-values):

$$U_{x_1} = \bar{F}_\theta(X_1), \cdots, U_{x_m} = \bar{F}_\theta(X_m)$$

,where $\bar{F}_\theta(X_1) = P_\theta(X_1 \geq x_1) = 1 - P_\theta(X_1 < x_1)$, and x 's are the observed active minute counts from the training period.(Note that a surge in count of a particular IP is considered as an anomaly or abnormal behaviour. This is why we are interested in monitoring upper tail probabilities or upper tail p-values. In addition, the p-values are **independent** in this context.)

If $\bar{F}_\theta$ is **continuous**, monotonic and strictly increasing, one only needs to verify whether $U_1, \cdots, U_m \sim U[0,1]$ to test our hypothesis. This is because for any $u \in [0,1]$,

$$\begin{aligned} P\{U_1 \leq u\} &= P\{\bar{F}_\theta(X_1) \leq u\} \\ &= P\{F_\theta(X_1) \geq 1-u\} \\ &= P\{X_1 \geq F_\theta^{(-1)}(1-u)\} \\ &= 1 - P\{X_1 \leq F_\theta^{(-1)}(1-u)\} \\ &= 1 - F_\theta\{F_\theta^{(-1)}(1-u)\} \quad (\text{under } H_0 : X_i \sim F_\theta) \\ &= u \end{aligned}$$

Therefore, we do not reject the $H_0 : X_i \sim F_\theta$ if $U_1, \cdots, U_m \sim U[0,1]$.
Instead of a proper test statistic, we can "think" that the p-value serves as a test statistic here. To test whether they are uniformly distributed, the one-sample Kolmogorov-Smirnov test is carried out to measure the distance between the empirical distribution function of the p-values and the uniform cumulative distribution function. To summarise, training essentially means finding a distribution that can explain and fit the active count data in March nicely such that we can obtain uniformally distributed p-values from it.

However, there are two issues here. First, the count levels are discrete and hence the upper tail p-values are discrete, i.e. $\bar{F}_\theta$ is not continuous. Thus, the p-values cannot be uniformly distributed. Second, θ is unknown in our case, which needs to be estimated by the maximum likelihood estimates $\hat{\theta}$.

3.4.1 Discrete p-values

The issue of **discrete** p-values arises here. See figure 3.8. Since we are working with discrete count levels, the p-values computed are also discrete and hence, not uniformally distributed.

By adopting N.A. Heard & M.O.M. Turcotte's approach, we can overcome the issue of discrete p-values. Let X be a discrete random variable on $\mathbb{N} = \{0, 1, 2, \ldots\}$. Denote the probability mass function , survivor function respectively as (7, N.A. Heard , M.O.M. Turcotte)

$$p_x = P(X = x) \quad \text{and} \quad s_x = U_x = P(X \geq x) = 1 - \textstyle\sum_{j=0}^{x-1} p_j$$

(Note that s_x is the same as U_x defined on page 42, they both refer to the discrete upper tail probabilities or p-values.) Then, one-sided upper tail p-values of draws from X are discrete random variables and therefore not $U[0, 1]$ random variables. Instead they are only approximately uniform when a latent variable U is drawn conditional on the realised value of X (7, N.A. Heard , M.O.M. Turcotte).

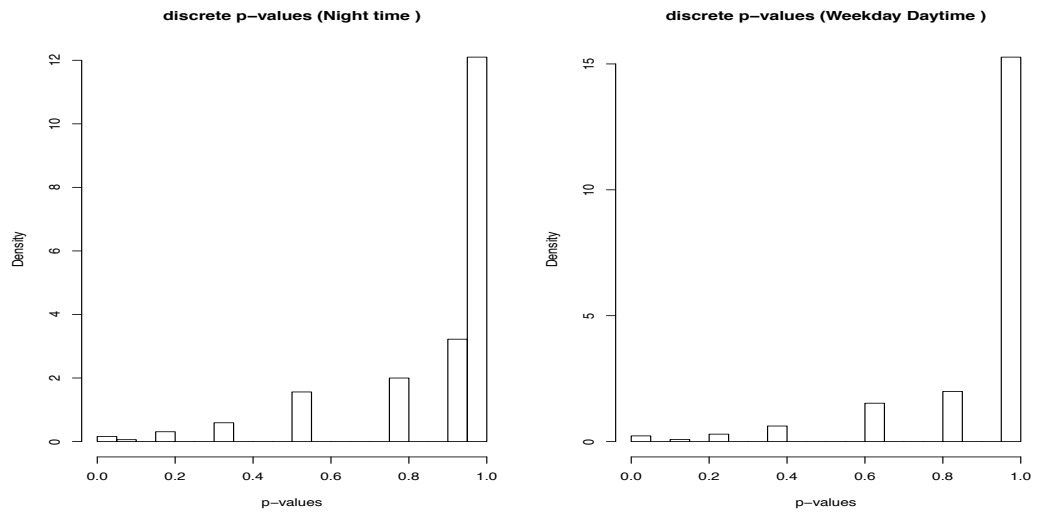


Figure 3.8: Discrete p-values

Theorem Suppose

$$X \sim p_x \quad \& \quad [U|X = x] \sim U(s_{x+1}, s_x]$$

Then $U \sim U[0,1]$. (7, N.A. Heard , M.O.M. Turcotte)

Proof: For $u \in [0,1]$, U has density function (using law of total probability)

$$f(u) = \sum_x p_x f(u|x) = \sum_x (s_x - s_{x+1}) \frac{\mathbb{1}_{(s_{x+1}, s_x]}(u)}{s_x - s_{x+1}} = \sum_x \mathbb{1}_{(s_{x+1}, s_x]}(u) = \mathbb{1}_{[0,1](u)} \quad \blacksquare$$

Therefore, the theorem above suggests that discrete p-values should be regarded as an interval censored observation of a truly uniform p-value. This means we interval censor a random variable X such that it is only known that X lies between two integers x and $x+1$, and then we sample X from a continuous distribution. (7, N.A. Heard , M.O.M. Turcotte) A discrete observation of $X = x$ then corresponds to the censored observation of a p-value

$$s \sim U(s_{x+1}, s_x]$$

(7, N.A. Heard , M.O.M. Turcotte)

3.4.2 Empirical Distribution

Having explained what we meant by training our model and how to deal with discrete p values. The second issue of unknown parameter θ is addressed here. We also illustrate here that although mixture distributions seemingly fit the data well, they fail in training. So, the hypotheses are as follow:

$$H_0 : \theta \sim F_\theta \quad \text{vs} \quad H_1 : \theta \text{ is not distributed as the proposed mixture distribution}$$

,where F_θ is the proposed mixture distribution. However, the parameter θ is unknown, so the computation is as follow:

$$U_1 = \bar{F}_{\hat{\theta}}(X_1), \cdots, U_m = \bar{F}_{\hat{\theta}}(X_m)$$

where θ is estimated by the maximum likelihood estimate $\hat{\theta}$, which is computed from the EM-algorithm. Again, we need to correct the discrete p-values, and then check if the corrected p-values ,$(s_1, \ldots, s_m)$, are distributed as uniform[0,1]. Figure 3.9 shows that the corrected p-values computed from poisson mixture and poisson-geometric mixture distributions are not uniformly distributed (supported by p-value significantly less than 0.5 from the K-S test)and hence, we reject the null hypothesis and conclude that the data do not follow our proposed mixture distributions.

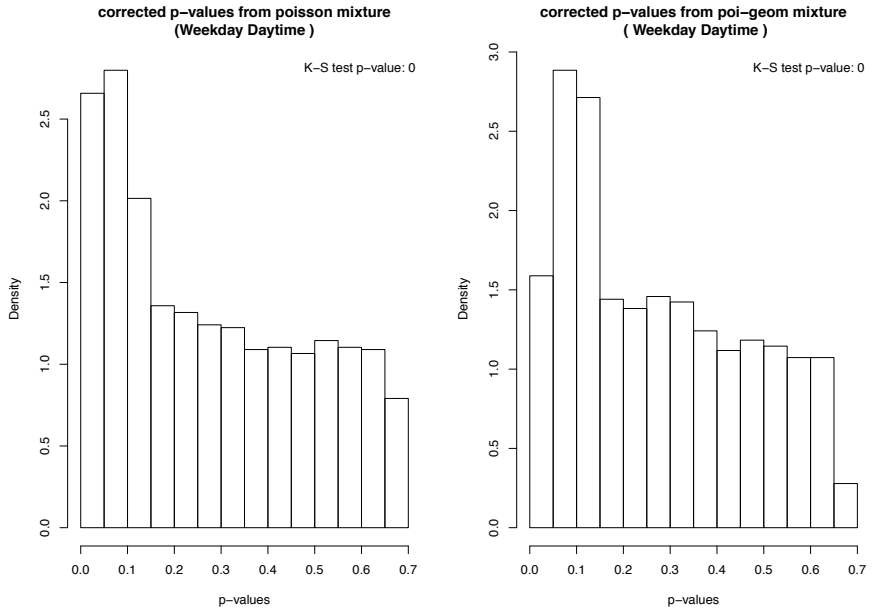


Figure 3.9: Corrected p-values from Mixture Distributions

This justifies the use of empirical distribution to estimate the distribution function of acive counts data. The **empirical distribution function** is defined as follow (14, van der Vaart, A.W. , 1998):

$$\hat{F}_n(t) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_i \leq t\}$$

Figure 3.10 shows the distribution of active counts on the left, the empirical cumulative distribution function (ecdf) in the middle and uniformly distributed corrected p-values computed from *ecdf* on the right. In fact, since empirical distribution fits the active count data so nicely, the corrected p-values computed from *ecdf* are always uniformly distributed, as shown in the figure. Put simply, using empirical distribution to train our model will always return uniformly distributed p-values for all IPs.

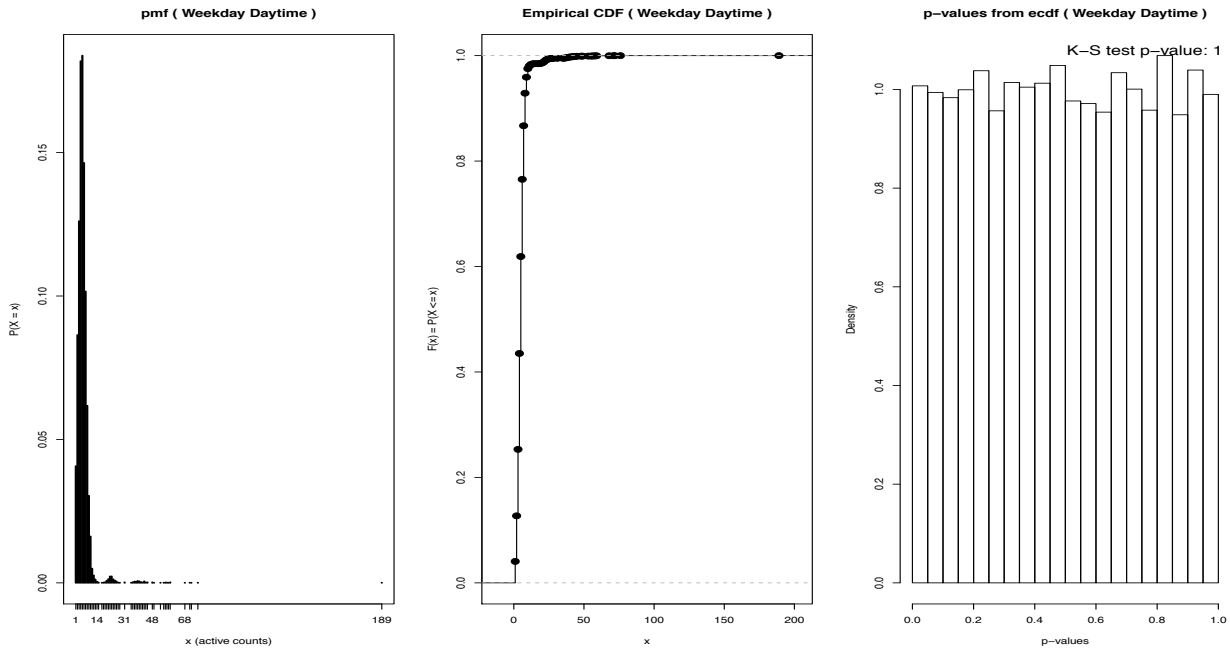


Figure 3.10: Training - Empirical Distribution

3.5 Testing

Once we train our model in the training period 26^{th} Feburary -26^{th} March, then we want to test our model in the testing period 27^{th} March -2^{nd} April. For testing, the following model is presented:

Assume $X_1, \cdots, X_m$ are i.i.d. from a distribution $F(x)$ where X_i is the count of the i^{th} minute from the training period.

First, with the training data, we use the empirical distribution function $\hat{F}^{(Training)}$ to estimate for the unknown distribution function $F^{(training)}(x)$. Similarly, one can get the estimate for the unknown distribution function of testing data, $F^{(Testing)}(x)$.

The hypothesis test is set up as follow:

$$H_0 : F^{(Training)} = F^{(Testing)} \text{ vs } H_1 : F^{(Training)} \neq F^{(Testing)}$$

And suppose we know $F^{(Training)}$ (not by estimation), then the upper tail probabilities are computed as follow:

$$U_1 = \bar{F}^{(Training)}(Y_1), \cdots, U_r = \bar{F}^{(Training)}(Y_r)$$

where $Y_1, \cdots, Y_r$ are the testing data and r is the size of testing data. Then, by checking the distribution of $U_1, \ldots, U_r$ and see if they are uniformly distributed $U[0,1]$ can help us test H_0 vs H_1 if $\bar{F}^{(Training)}$ is **continuous**. This is because for any $u \in [0,1]$

$$\begin{aligned}
P\{U_1 \leq u\} &= P\{\bar{F}^{(Training)}(Y_1) \leq u\} \\
&= P\{F^{(Training)}(Y_1) \geq 1-u\} \\
&= P\left\{Y_1 \geq \left[F^{(Training)}\right]^{(-1)}(1-u)\right\} \\
&= 1 - P\left\{Y_1 \leq \left[F^{(Training)}\right]^{(-1)}(1-u)\right\} \\
&= 1 - F^{(Testing)}\left\{\left[F^{(Training)}\right]^{(-1)}(1-u)\right\} \quad (\text{under } H_0 : F^{(Training)} = F^{(Testing)}) \\
&= u
\end{aligned}$$

But now (1)$F^{(Training)}$ is not continuous, as X_i's are discrete random variables and (2)$F^{(Training)}$ is unknown. Thus, we estimate the $F^{(Training)}$ by the empirical distribution function $\hat{F}^{(Training)}$:

$$\hat{F}^{(Training)}(x) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}\{X_i \leq x\}$$

Once again, if we compute the upper tail p-values , they will be discrete and need to be corrected by sampling from a uniform distribution as below:

$$s \sim U(s_{y+1}, s_y]$$

,where $s_y = P(X \geq y) = 1 - \sum_{j=0}^{y-1} p_j$. (Note that s_y is equivalent to the U_y above, as they both refer to discrete upper-tail p-values. And X is the discrete random variable of active minute count, which follows the empirical distribution function in the training period.) We do not reject the null hypothesis, if the corrected p-values, $(s_1, \ldots, s_r)$, are uniformly distributed.

3.6 Anomaly detection using EWMA Control Charts

Ultimately, if the corrected p-values returned from both the training period and the testing period are uniformly distributed, we can proceed further with the model to detect anomalies using EWMA control charts. Suppose t_n are the active minutes for a particular IP, and $n \in \{1, 2, \ldots\}$. Given that the time-indexed p-values $\{p_n\}$ are independent and uniformly distributed on $[0, 1]$ under the null hypothesis of normal behaviour, they can be transformed to real values $\{Z_n\}$ which independently follow a standard normal distribution under the same hypothesis, using Stouffer's Z-score method (7, N.A. Heard , M.O.M. Turcotte)

$$Z_n = \Phi^{-1}(1 - p_n)$$

Note that a surge in count at time t_n may be deemed as an anomalous event, which yields a very low p-value. This corresponds to a large value of Z_n (7, N.A. Heard , M.O.M. Turcotte).

To accumulate evidence of anomalous behaviour over time, we construct an exponentially weighted moving average chart $\{S_n\}$based on Z-scores $\{Z_n\}$,

$$S_n = (1 - w)S_{n-1} + wZ_n, \quad n \geq 1$$

with $S_0 = 0$ (7, N.A. Heard , M.O.M. Turcotte). w is weight. It is a parameter chosen by the user and it is set equal to 0.25 in our case (11, Montgomery, Douglas 2005). The control chart also has upper control limit (UCL) and lower control limit (LCL) under the null hypothesis (7, N.A. Heard , M.O.M. Turcotte) :

$$L\sqrt{\frac{w}{2 - w}[1 - (1 - w)^{2n}]}$$

L is another parameter and it is set to 3 in our case (11, Montgomery, Douglas (2005)).

It is worth noting that we are more concerned about going above the UCL than going below the LCL. This is because going below the LCL means the count level is very low at time t_n, which yields a very high p-value and hence a, very low Z-score. Low activity level is not an anomalous behaviour. However, high activity level tends to generate low p-value and thus, a very high Z-score that goes above the UCL. And this warrants a lot of attention.

3.7 Results

We will devote this section to analyse several IP addresses, namely 100.253.169.162, 100.253.181.118, 100.253.70.50 and 100.253.230.15. We compute their transition matrices to see how they behave at different times within a day . Then we train our model using empirical distribution to explain the active count data in training period, and then test the models upon data from testing period to see whether the p-values are uniformly distributed or not. Finally, we construct an EWMA control chart for anomaly detection. Note that the training period starts from $6pm$ on 26^{th} Feb to $23:59pm$ on 26^{th} March and testing period goes from $00:00am$ on 27^{th} March to $12:46pm$ on 2^{nd} April. In addition, we cannot not present the analysis of night time and weekend behaviour for most IPs, because there is very little activity during night time and weekend in the testing period.

3.7.1 Results of IP 100.253.169.162

The transition matrices during night time, weekday daytime and weekend nighttime in the training period are as follow:

Night time

	I	A
I	1.00	0.00
A	0.21	0.78

Weekday Daytime

	I	A
I	0.81	0.19
A	0.32	0.68

Weekend Daytime

	I	A
I	1.00	0.00
A	0.41	0.59

, where I and A refer to the *inactive* and *active* state .Combined with the density plots 3.11, we observe that this IP is busy at daytime during weekdays and fairly quiet during night time and weekends. Thus, we can infer this IP is a regular college user.

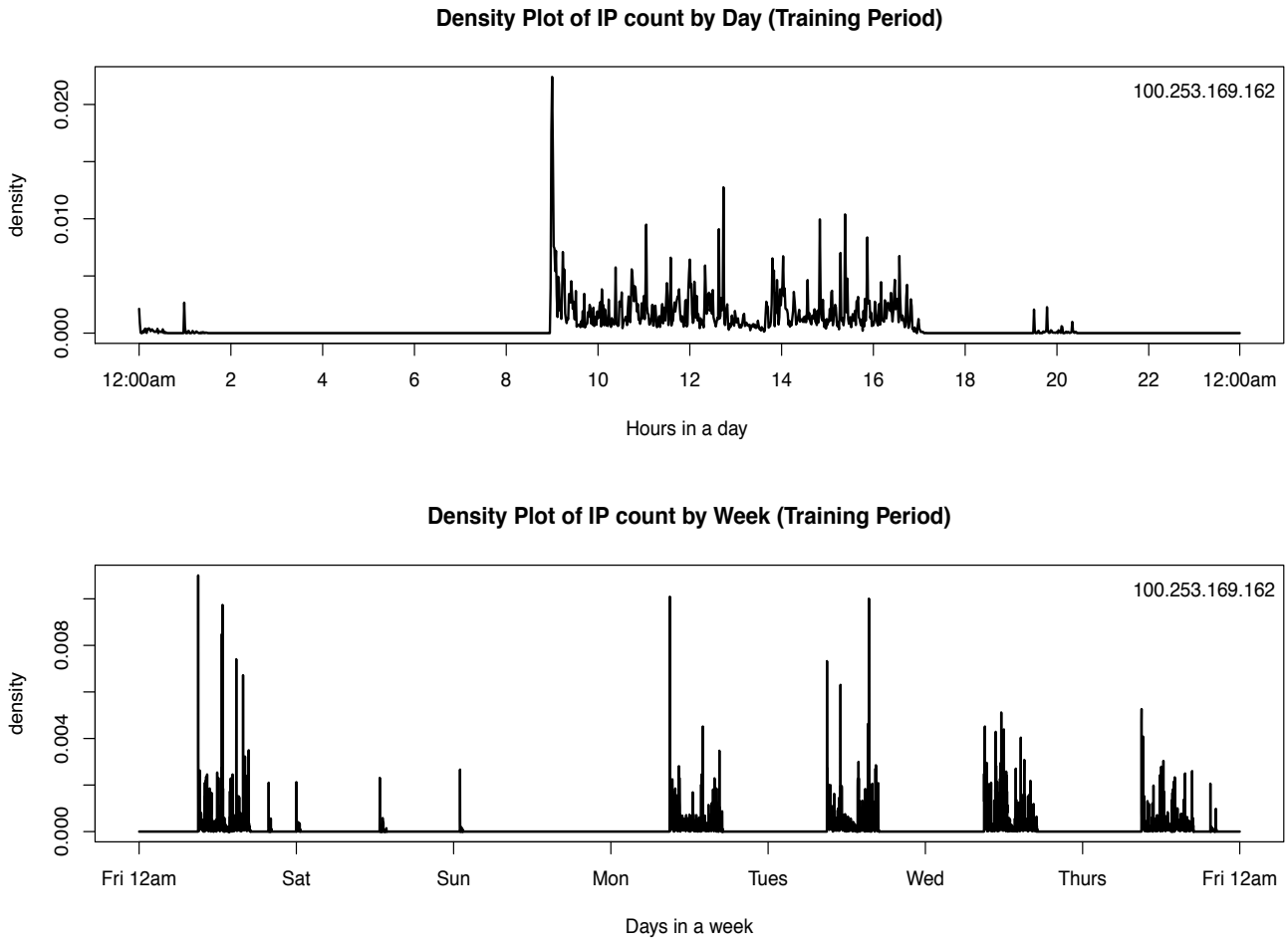


Figure 3.11: Density of counts by day and by week (Training period)

We train our model using empirical distribution in the training period. See the top row of figure 3.12. As expected, the corrected p-values are uniformly distributed in $[0, 1]$. We then test our model and the corrected p-values at night time and weekday daytime are both greater than 0.05. Hence, we do not reject the null hypothesis and conclude that active count data from the testing period follows the empirical distribution in the training period. See figure bottom row of figure 3.12. As a result, we construct an EWMA control chart for this IP's corrected p-values, as shown in figure 3.13. The blue circle marks the the highest exponentially-weighted average value (EWMA value), which is below the upper control limit (dotted red line). This suggests there is no anomalous behaviour at the testing period.

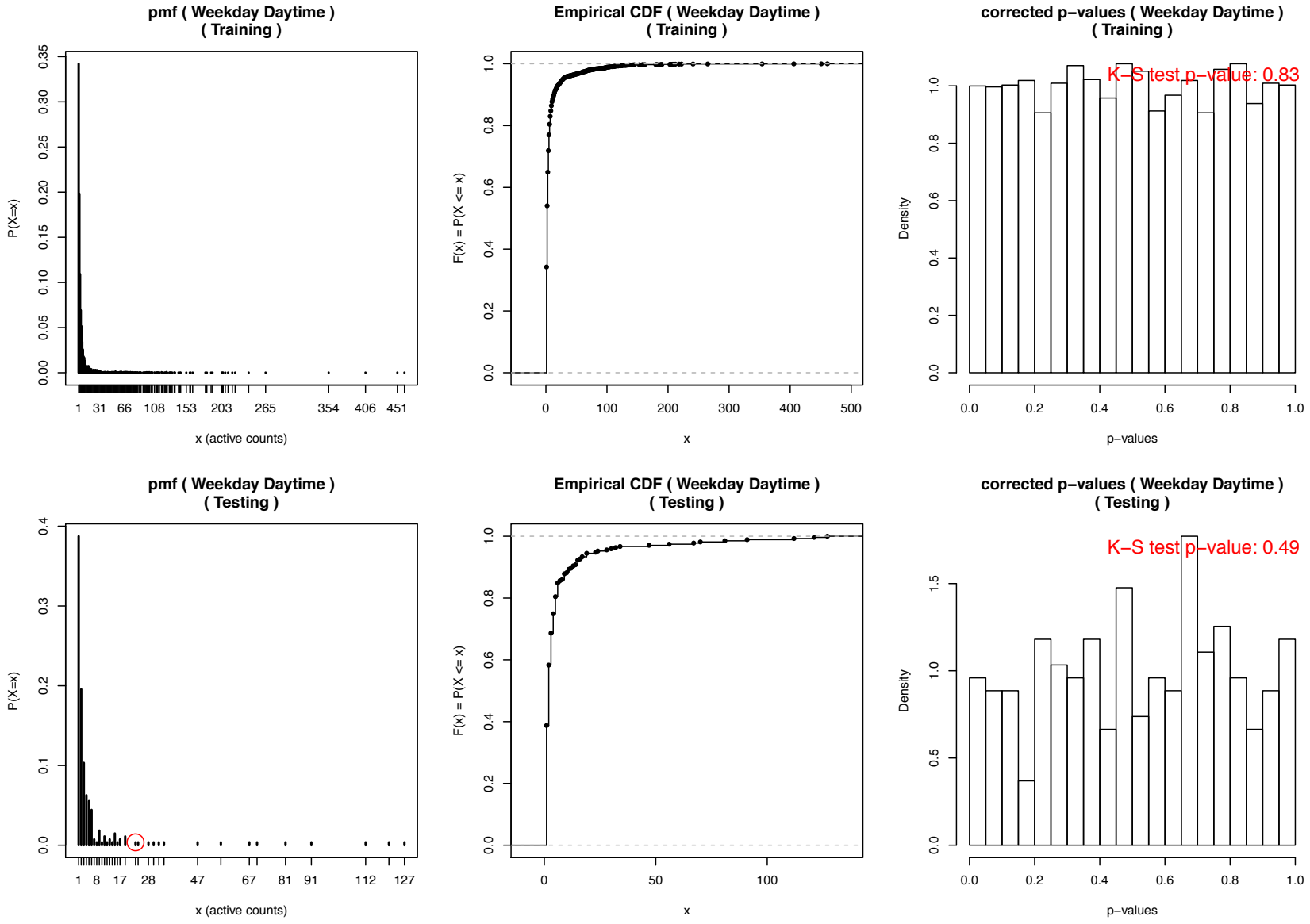


Figure 3.12: Training & Testing - 100.253.169.162 at Weekday Daytime

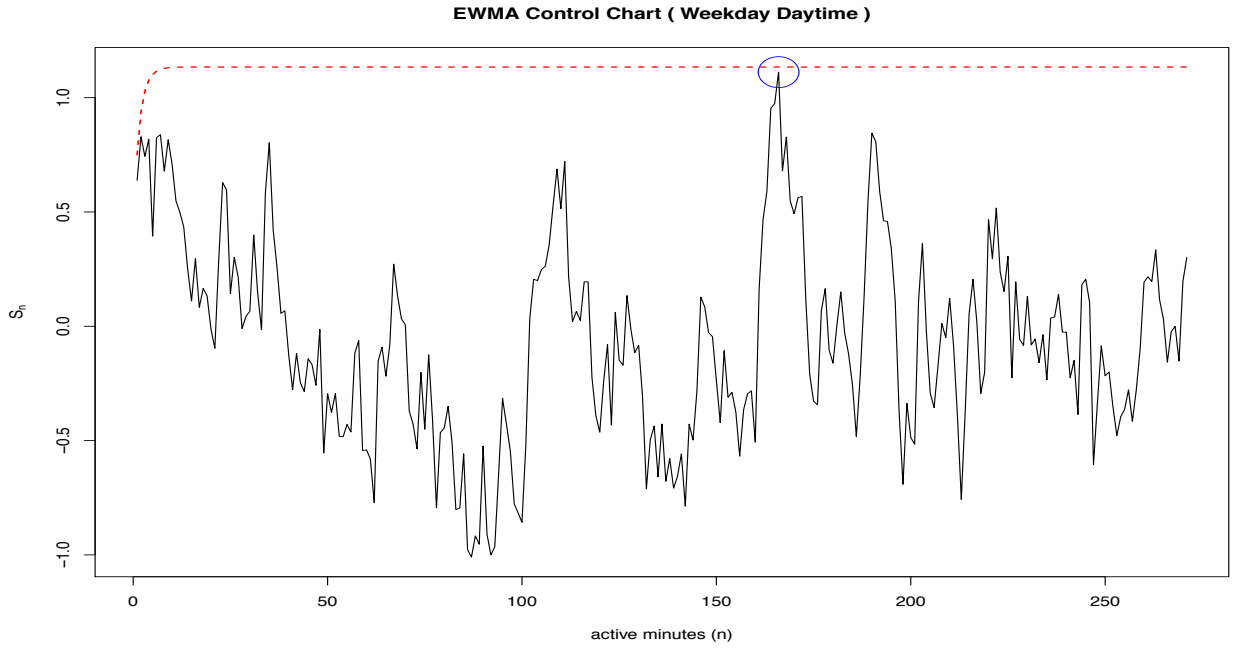


Figure 3.13: EWMA Control Chart - 100.253.169.162 at Weekday Daytime - No anomalies detected

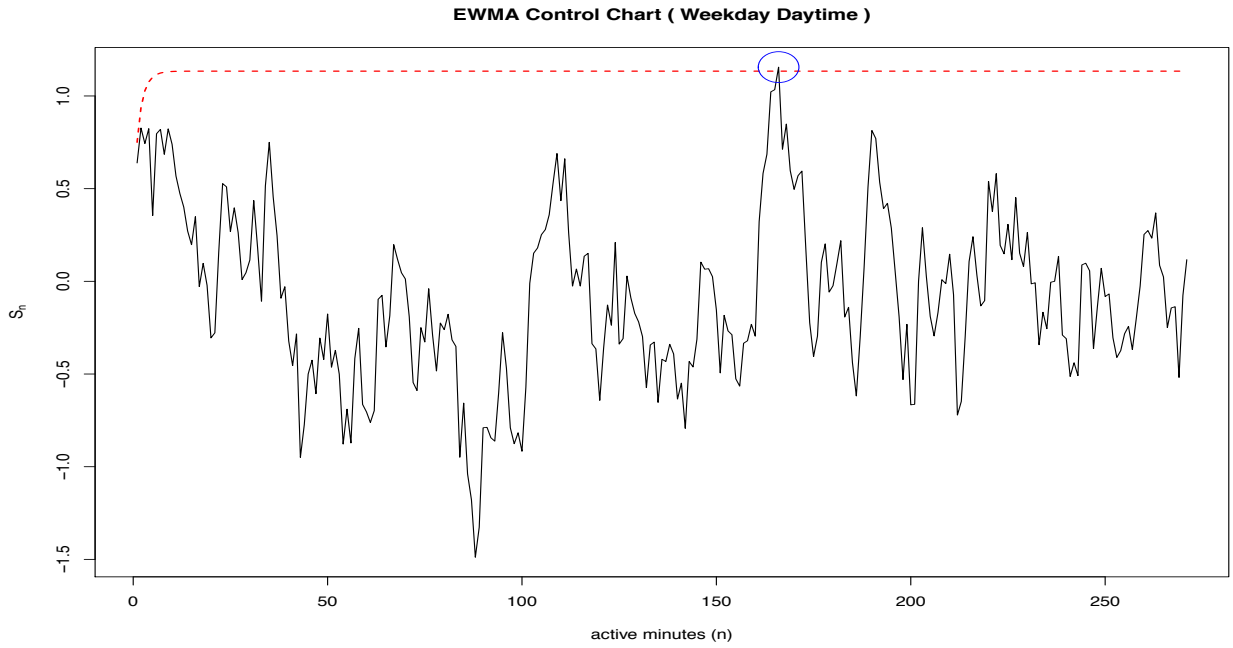
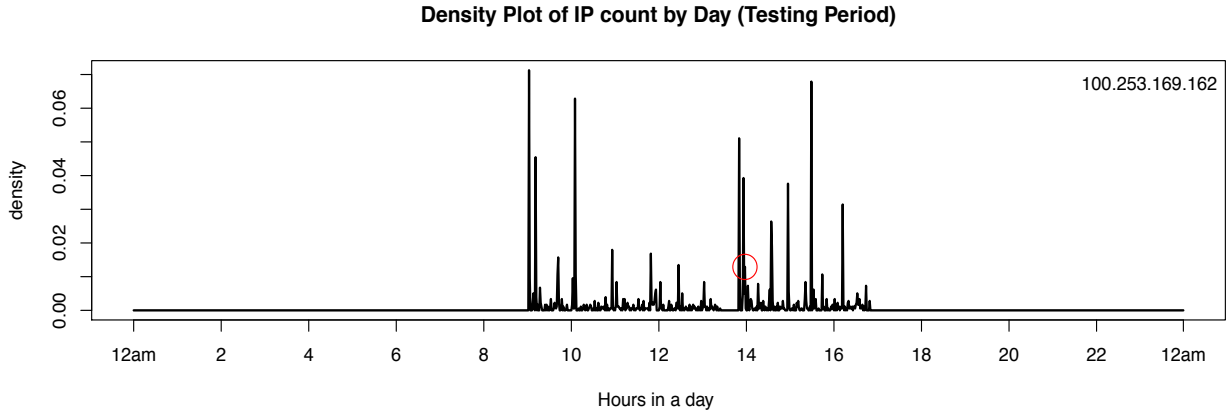


Figure 3.14: EWMA Control Chart - 100.253.169.162 at Weekday Daytime - Anomalies detected

However, if we interval censor the discrete p-values and sample again the corrected p-values from the uniform distribution between the intervals, we obtain an EWMA control chart that shows that there is anomalous behaviour, as shown in figure 3.14. This hints at the possibility of **false alarms**. We should **raise the value of L** in the control limit equation to avoid false alarms. Figure 3.15 is the density plot of this particular IP over the testing period, which also gives us a rough idea of this IP's activity level over time. The red circle indicates the time during the day and week in which the false alarm takes place. It can be seen the red circle occurs at around $14:00pm$ on Wednesday 27^{th} March and we observe from figure 3.15 the activity level is not high at all. In addition, the corresponding red circle in figure 3.12 is a count of 22, which is not high at all. This highlights an important feature of EWMA control chart, as it accumulates suprises over time. These evidence further reinforces the idea that this is really just a false alarm and the need to raise the parameter L.



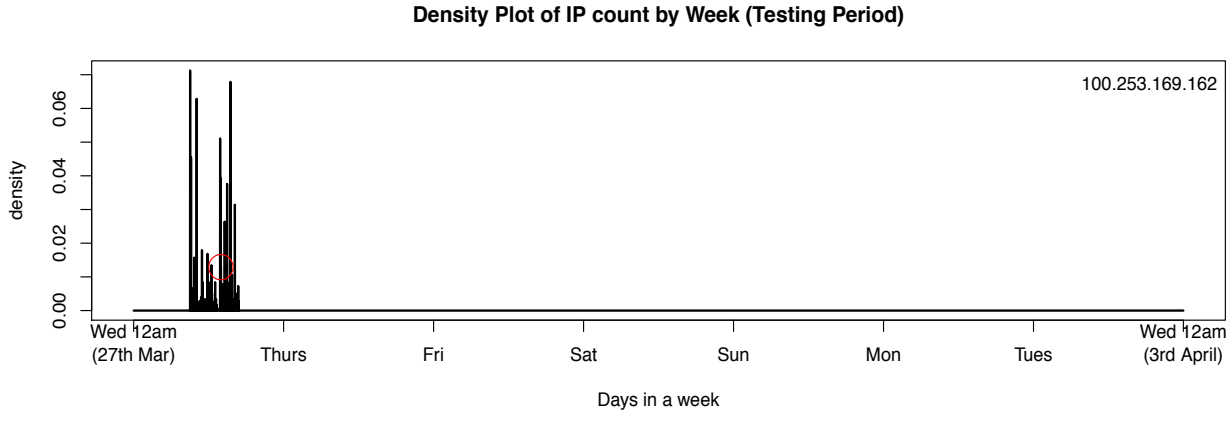


Figure 3.15: Density of counts by day and by week (Testing period)

3.7.2 Results of IP 100.253.181.118

Night time

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 1.00 & 0.00 \\ 0.08 & 0.92 \end{pmatrix} \end{array}$$

Weekday Daytime

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{pmatrix} \end{array}$$

Weekend Daytime

$$\begin{array}{c} \\ I \\ A \end{array} \begin{array}{c} \begin{array}{cc} I & A \end{array} \\ \begin{pmatrix} 1.00 & 0.00 \\ 0.03 & 0.97 \end{pmatrix} \end{array}$$

The transition matrices above and density plots 3.16 below suggests normal behaviour of a regular college user. Figure 3.17 shows how we train and test our model to obtain uniformly distributed corrected p-values. Hence we do not reject the null hypothesis. Based on the corrected p-values, we produce an EWMA control chart in figure 3.18.

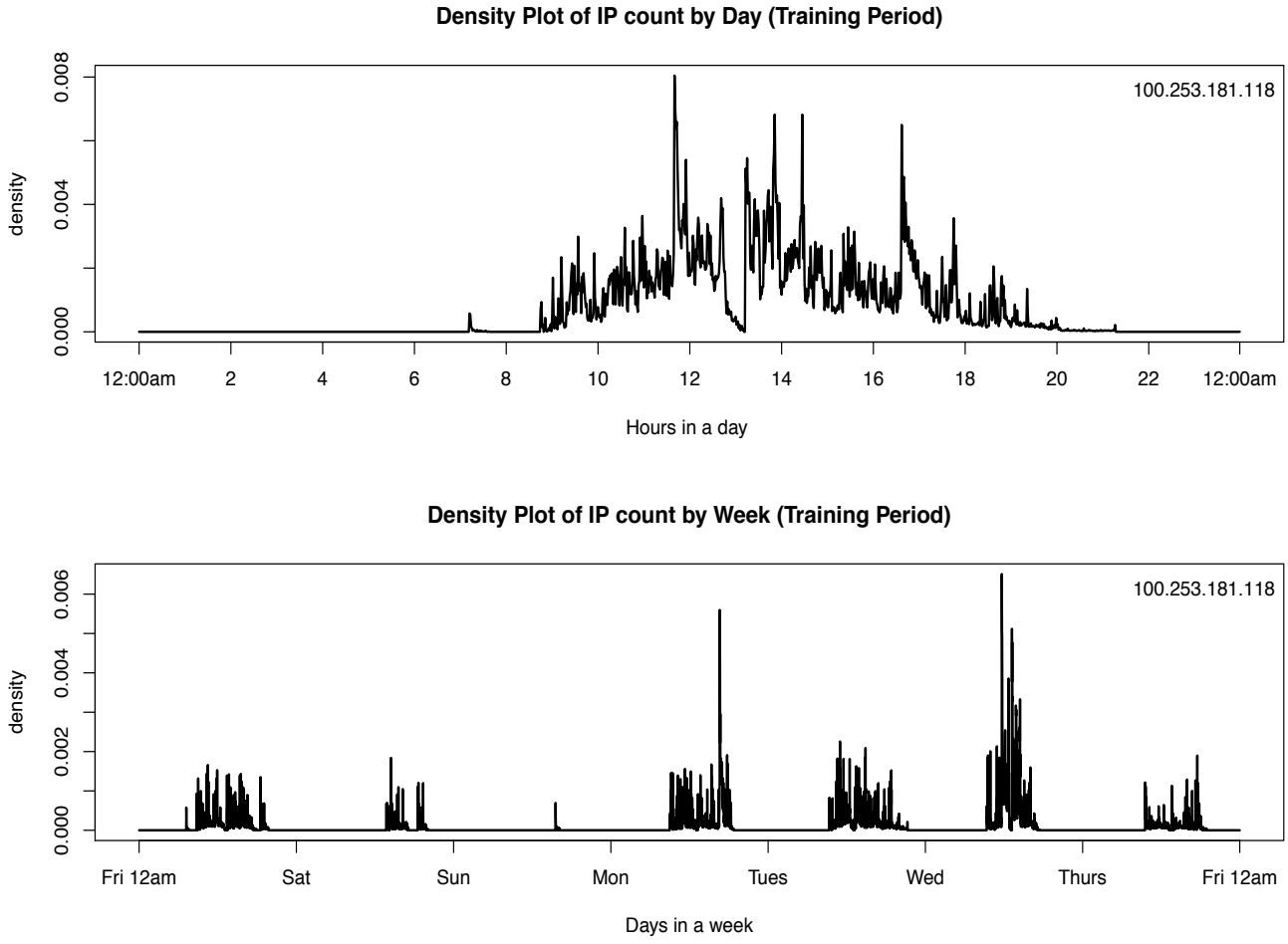


Figure 3.16: Density of counts by day and by week (Training period)

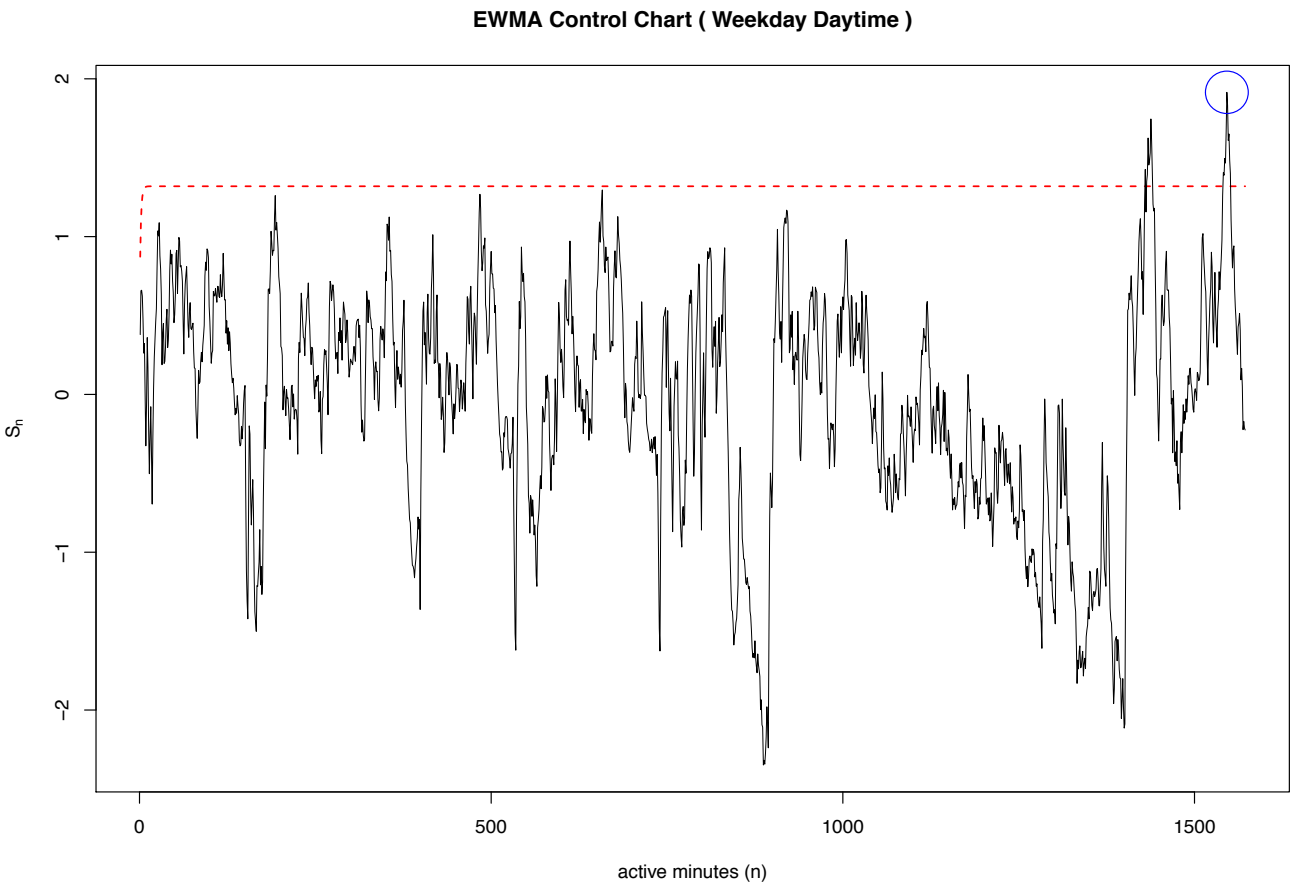


Figure 3.18: EWMA Control Chart - 100.253.181.118 at Weekday Daytime

Given high count levels in March as shown in figure 3.17, we raise the value of L from 3 to 3.49 and thus, increase upper control limit such that it prevents false alarms. Nonetheless, there are still anomalies. The blue circle captures the anomaly with highest EWMA value, which has a corresponding count level of 778, as shown by the red circle in figure 3.17. This occurs at midday on 2^{nd} April in figure 3.19.

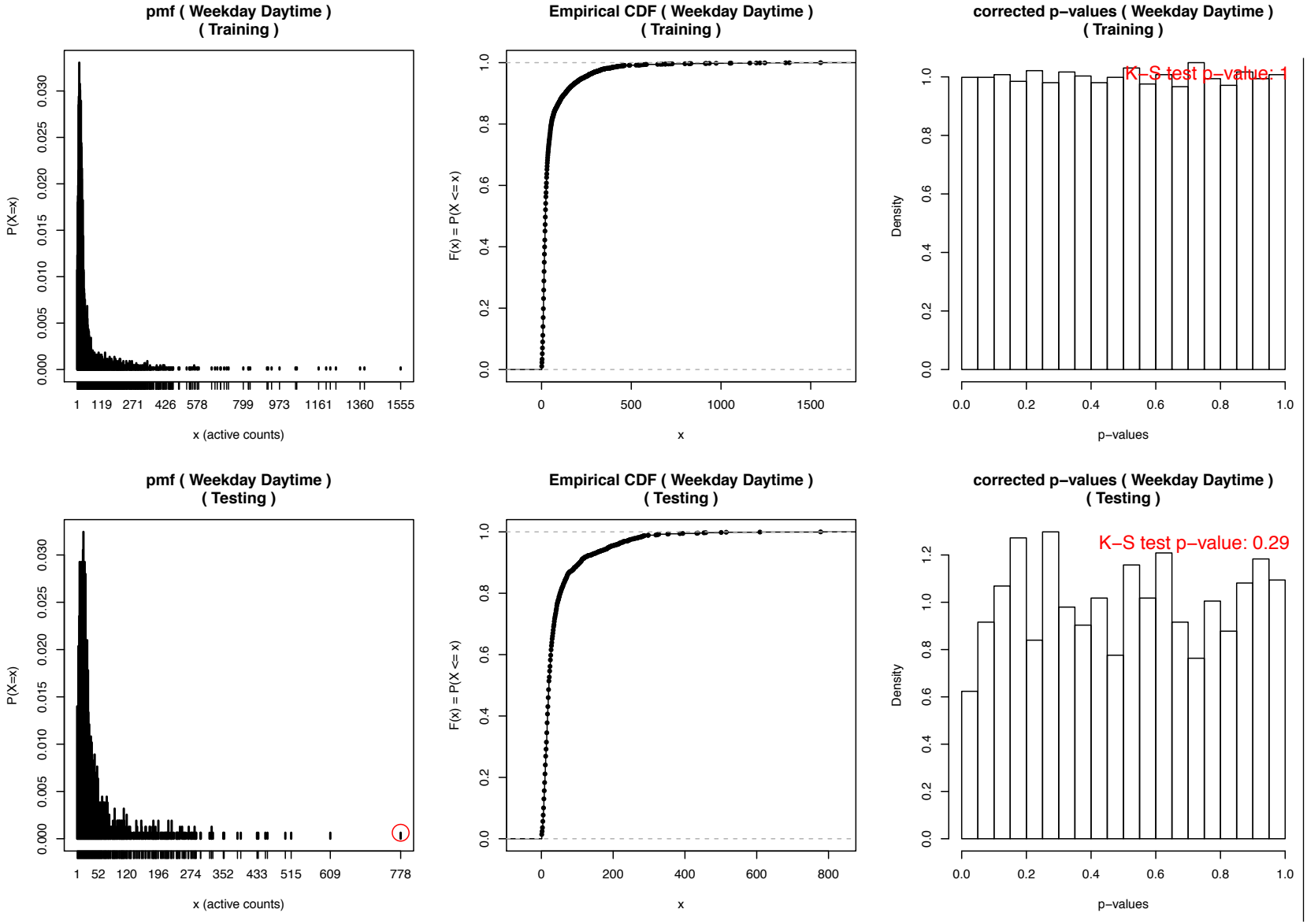


Figure 3.17: Training & Testing - 100.253.181.118 at Weekday Daytime

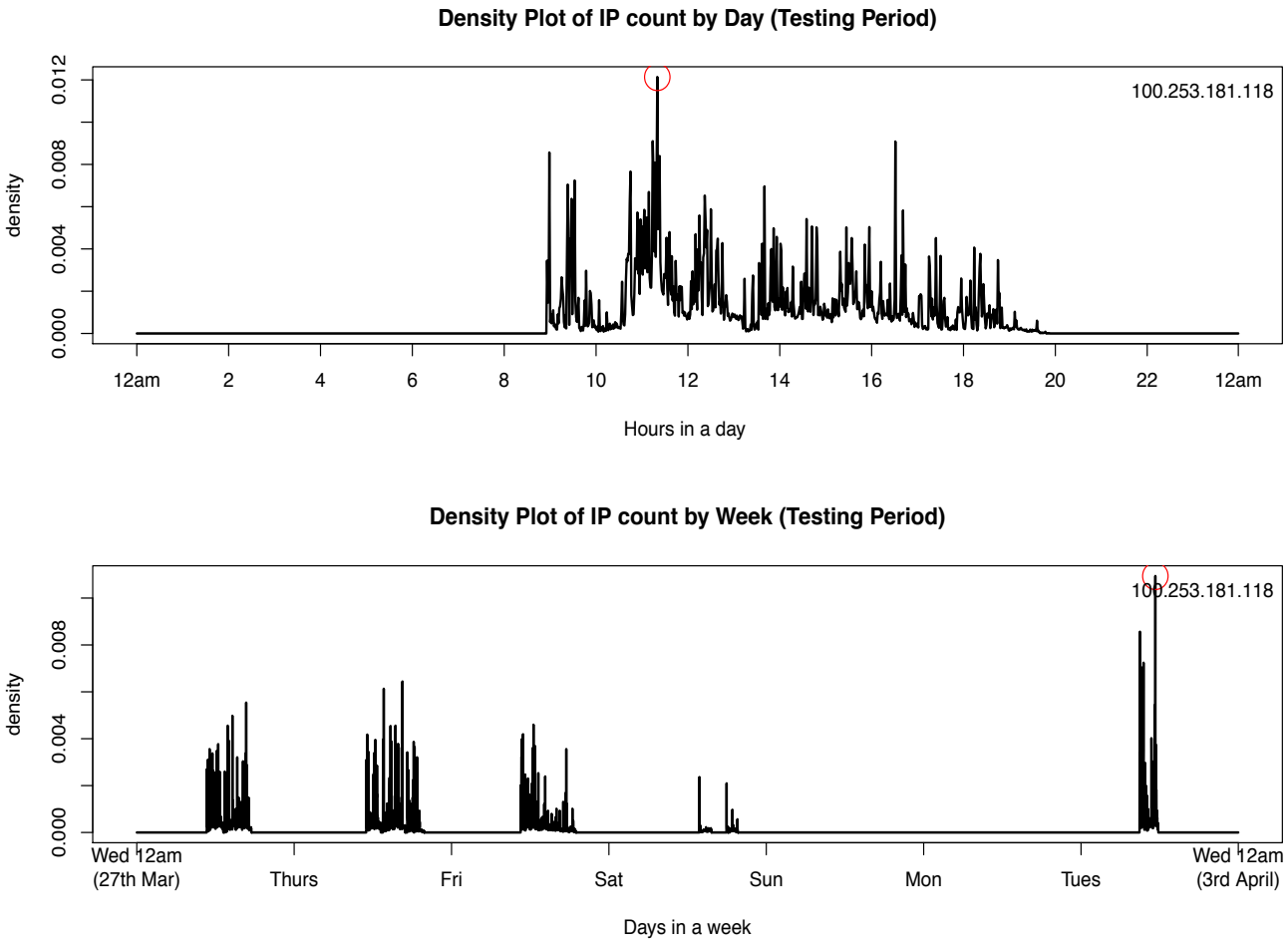


Figure 3.19: Density of counts by day and by week (Testing period)

3.7.3 Results of IP 100.253.70.50

Let's look at a very different IP with the transition matrices below and density plots of counts in figure 3.20. It can be seen that this IP behaves the same across time, as the transition matrices are almost identical. It is never quiet. This is not a normal college user, but it may be a machine that is constantly making connections to other machines. Although its behaviour is far from normal, it does illustrate a regular pattern over time, as it peaks every hour within a day.

Again, we train and test the model for night time and weekday daytime behaviour in figure 3.21, 3.22. The corrected p-values are significantly greater than 0.05 and hence

we do not reject the null hypothesis.

$$\begin{array}{c} \text{Night time} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.65 & 0.35 \\ 0.54 & 0.46 \end{pmatrix} \end{array} \quad \begin{array}{c} \text{Weekday Daytime} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.66 & 0.34 \\ 0.55 & 0.45 \end{pmatrix} \end{array} \quad \begin{array}{c} \text{Weekend Daytime} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.64 & 0.36 \\ 0.54 & 0.46 \end{pmatrix} \end{array}$$

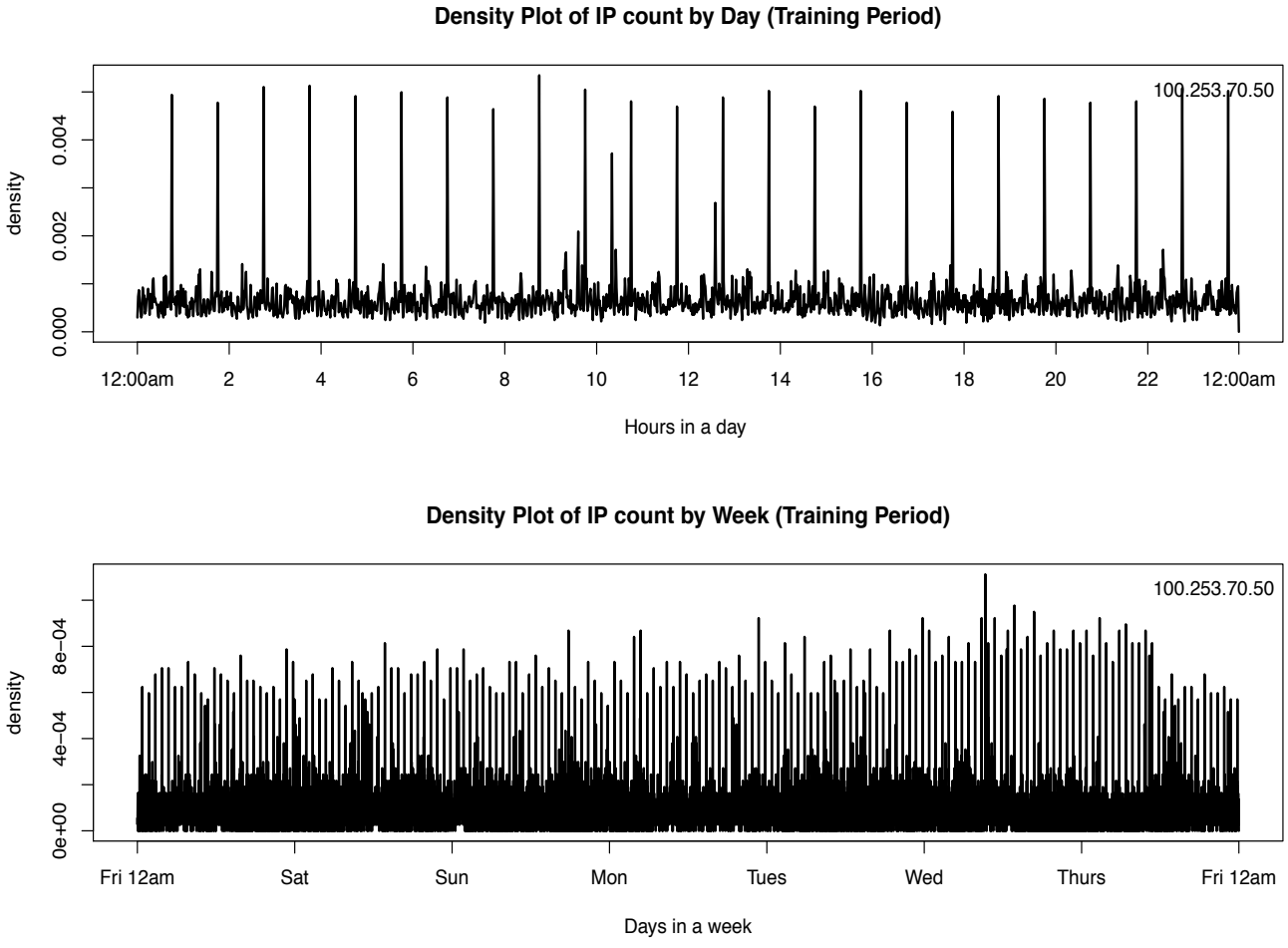


Figure 3.20: Density of counts by day and by week (Training period)

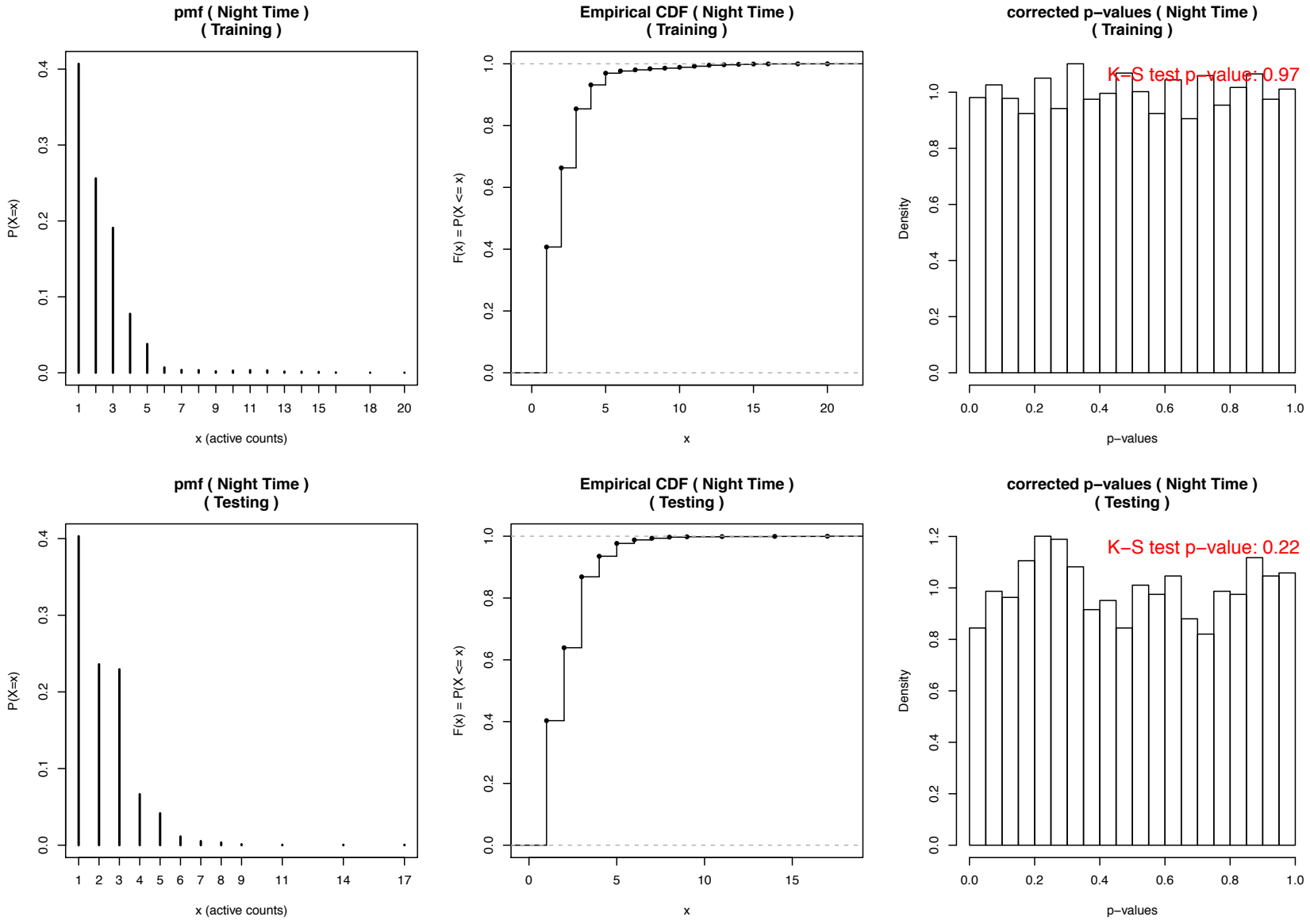


Figure 3.21: Training & Testing - 100.253.70.50 at Night Time

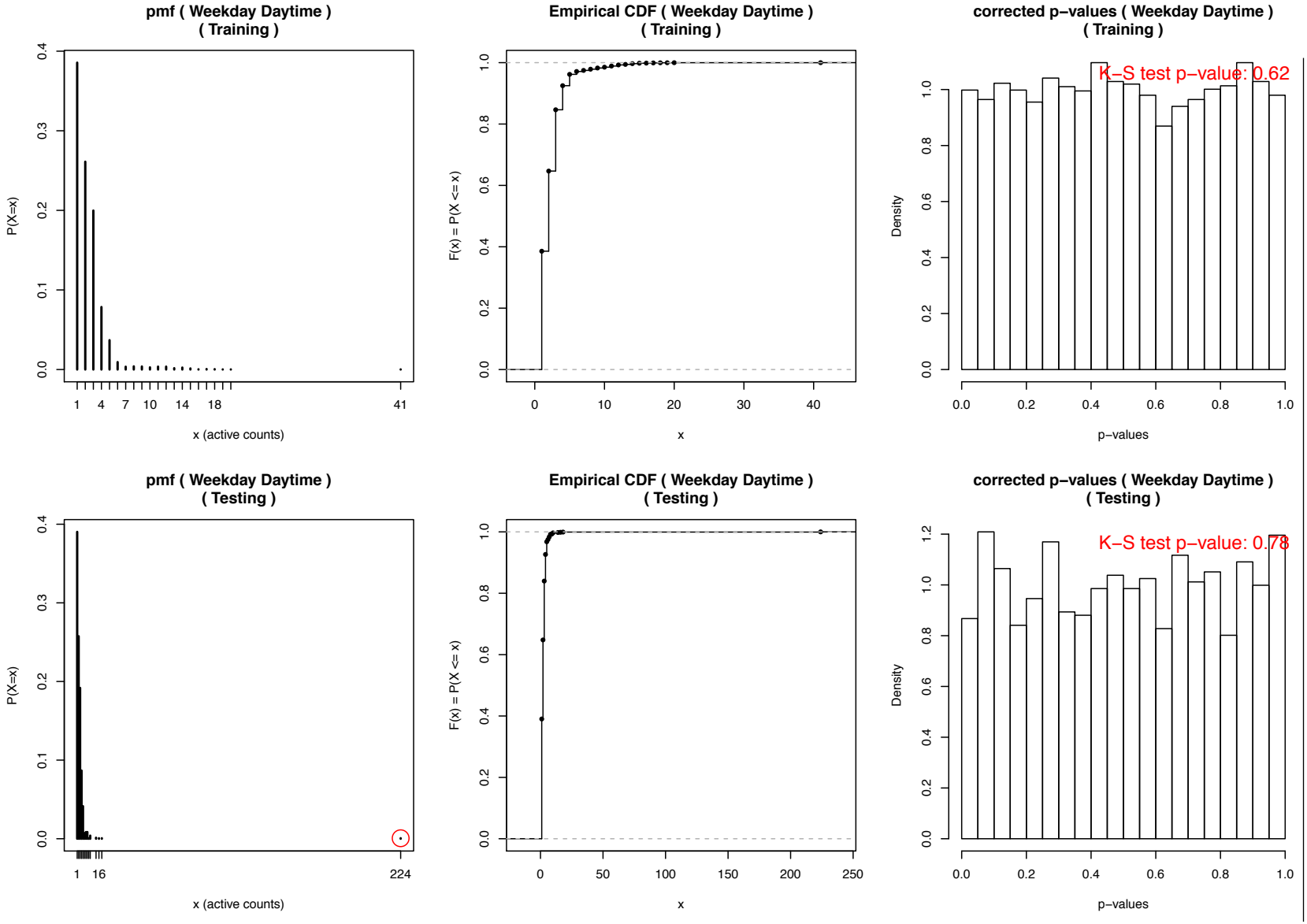


Figure 3.22: Training & Testing - 100.253.70.50 at Weekday Daytime

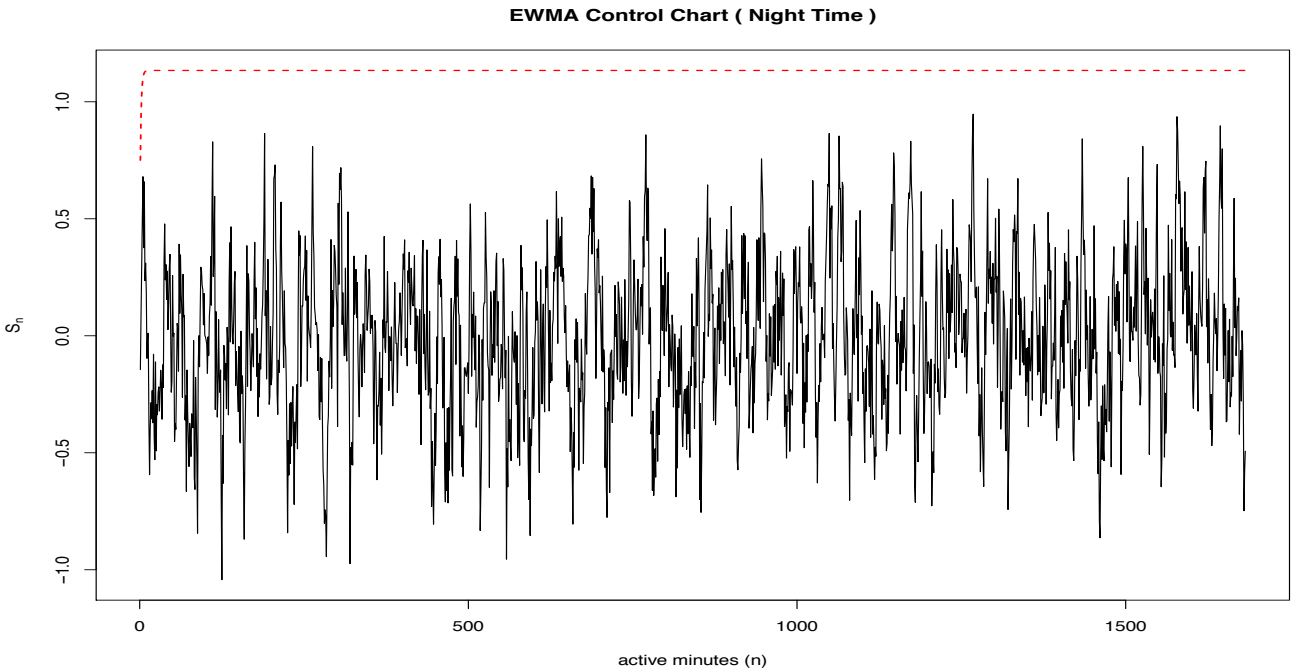


Figure 3.23: EWMA Control Chart - 100.253.70.50 at Night Time

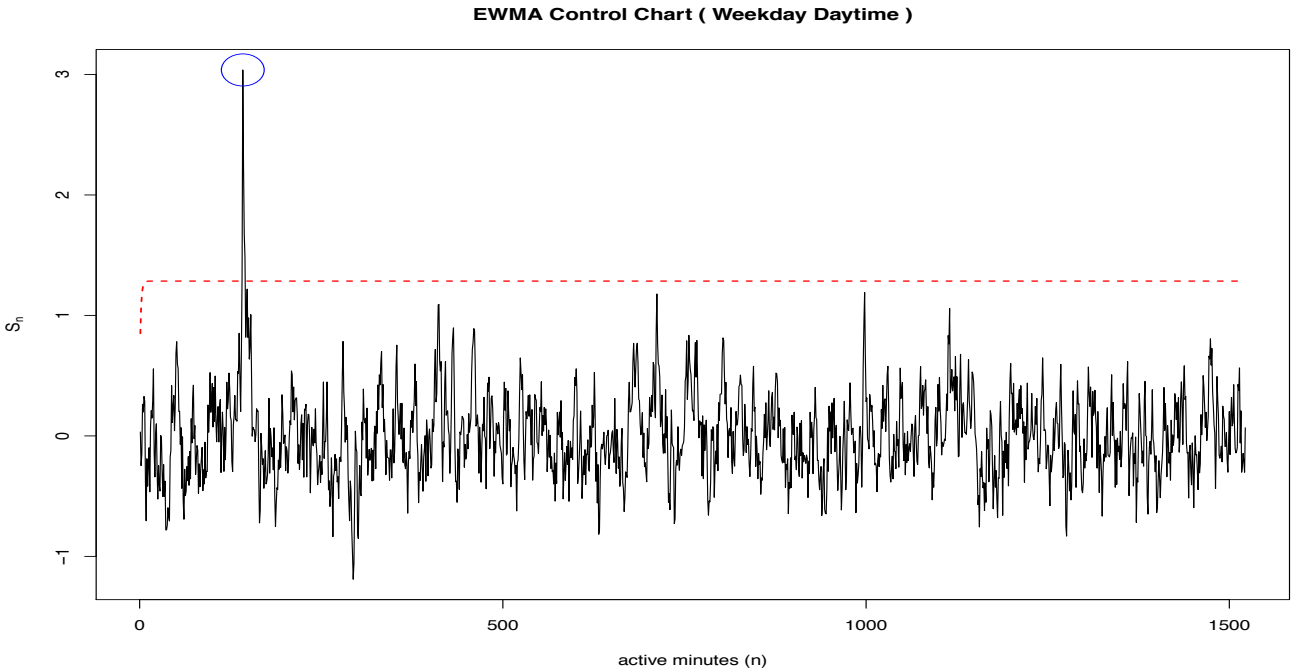


Figure 3.24: EWMA Control Chart - 100.253.70.50 at Weekday Daytime

The EWMA control charts for night time and weekday daytime are shown respectively in figure 3.23 , 3.24. Note that we have raised the parameter L of the control limit equation to 3.49 in order to prevent false alarms. Night time behaviour is normal, as there are no anomalies detected. However, we are certain that there was an anomaly at $1pm$ on 27^{th} March (figure 3.25) with count level 224, which is significantly higher than all other count levels occurred across the training period and testing period.

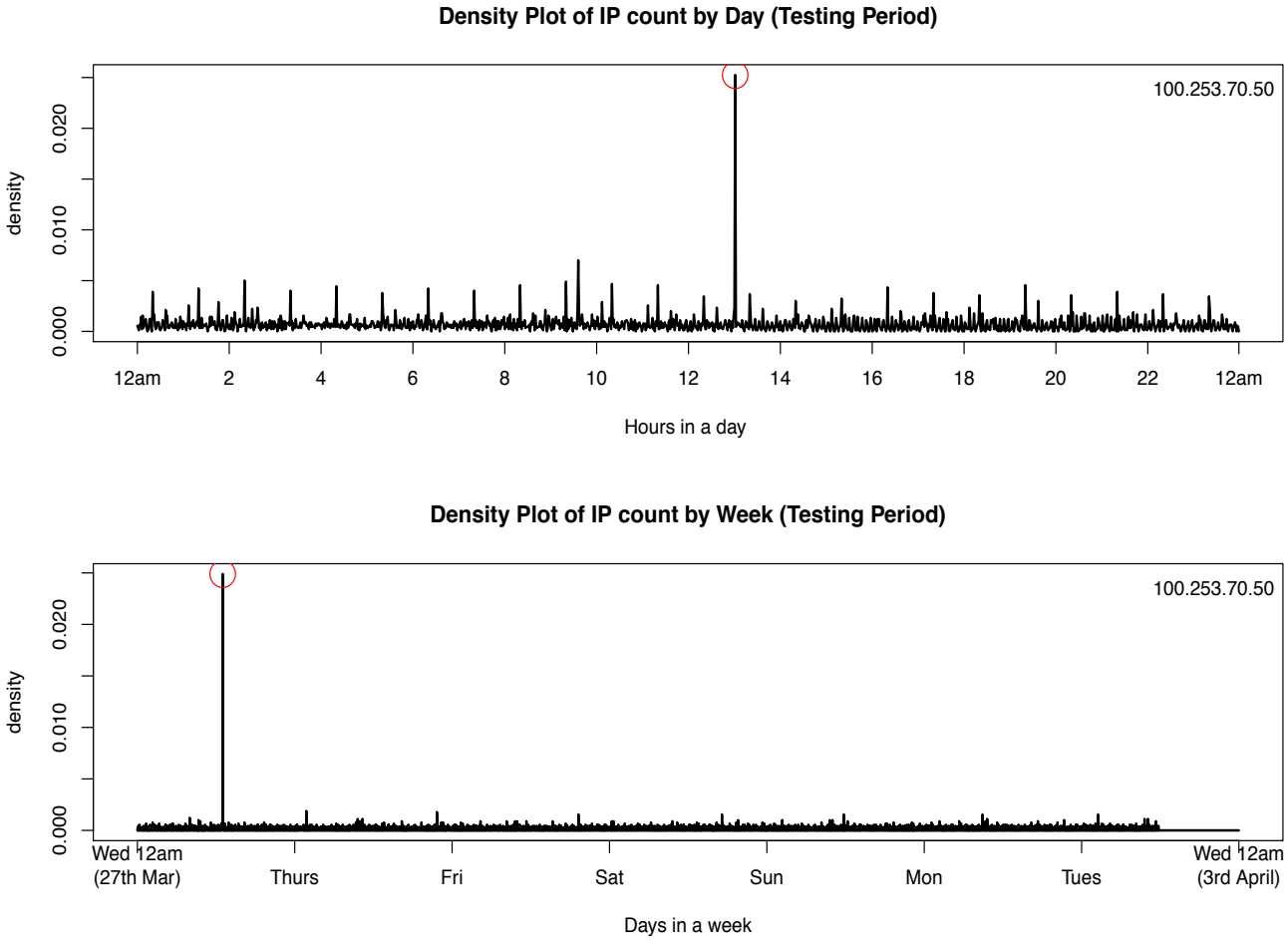


Figure 3.25: Density of counts by day and by week (Testing period)

3.7.4 Results of IP 100.253.230.15

Our last example is a normal IP with transition matrices below and density plot in figure 3.26. This IP is pretty quiet during night time and busy during daytime. It has

zero activity during weekends. Training and testing is displayed in figure 3.27. The EWMA control chart is shown in figure 3.28.

$$\begin{array}{c} \text{Night time} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 1.00 & 0.00 \\ 0.14 & 0.86 \end{pmatrix} \end{array} \qquad \begin{array}{c} \text{Weekday Daytime} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 0.91 & 0.10 \\ 0.17 & 0.83 \end{pmatrix} \end{array} \qquad \begin{array}{c} \text{Weekend Daytime} \\ \begin{array}{cc} I & A \end{array} \\ \begin{array}{c} I \\ A \end{array} \begin{pmatrix} 1.00 & 0.00 \\ - & - \end{pmatrix} \end{array}$$

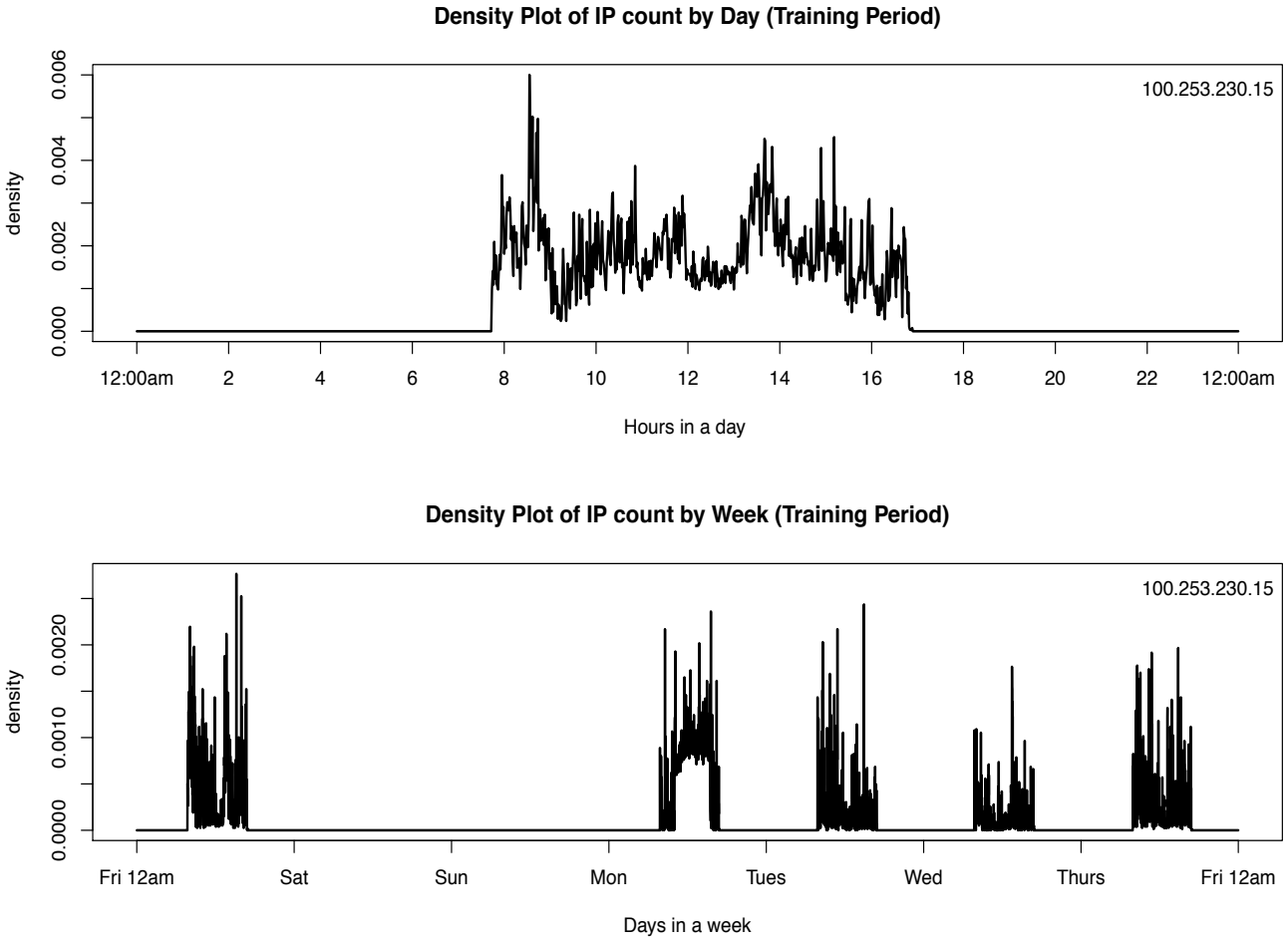


Figure 3.26: Density of counts by day and by week (Training period)

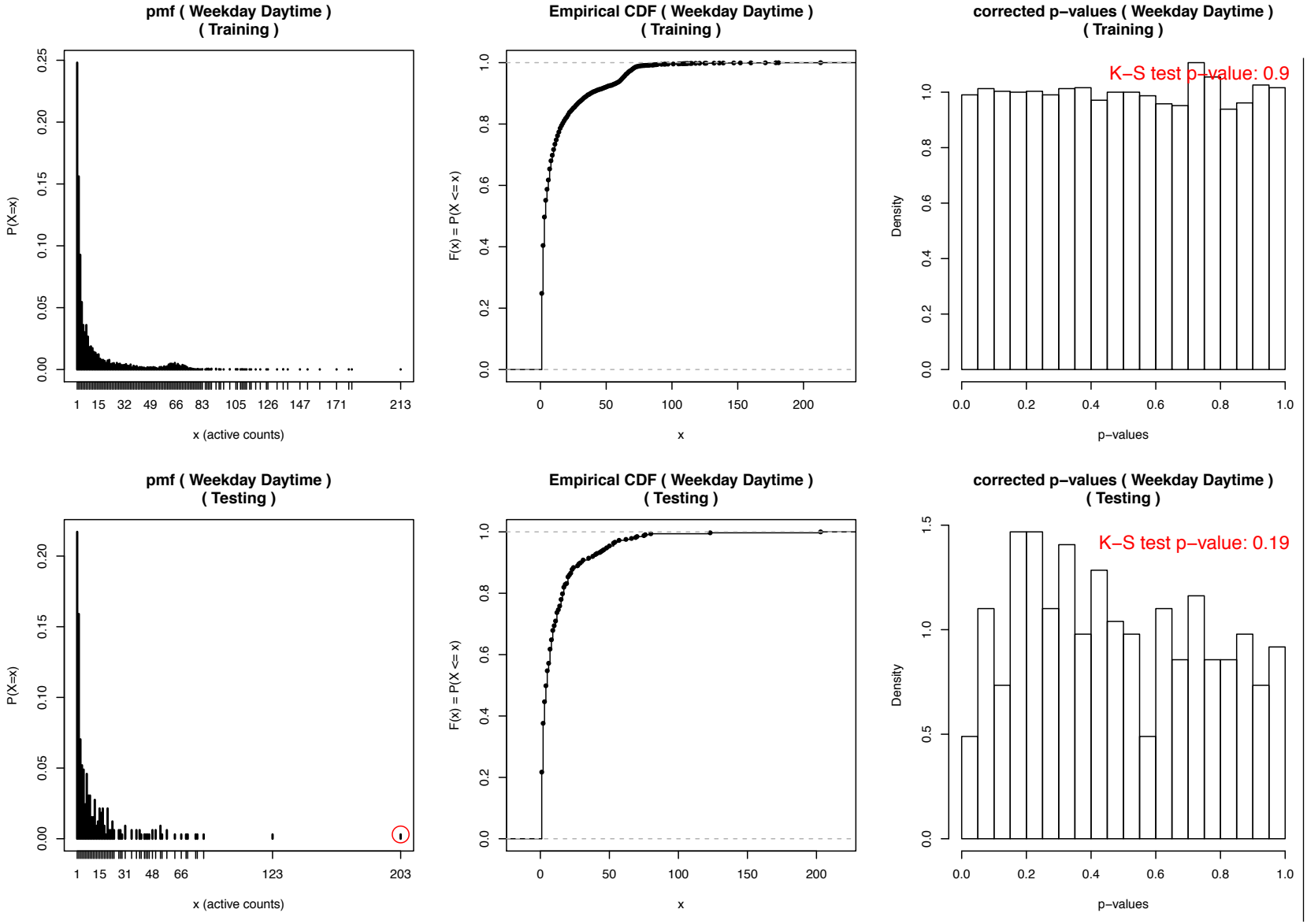


Figure 3.27: Training & Testing - 100.253.230.15 at Weekday Daytime

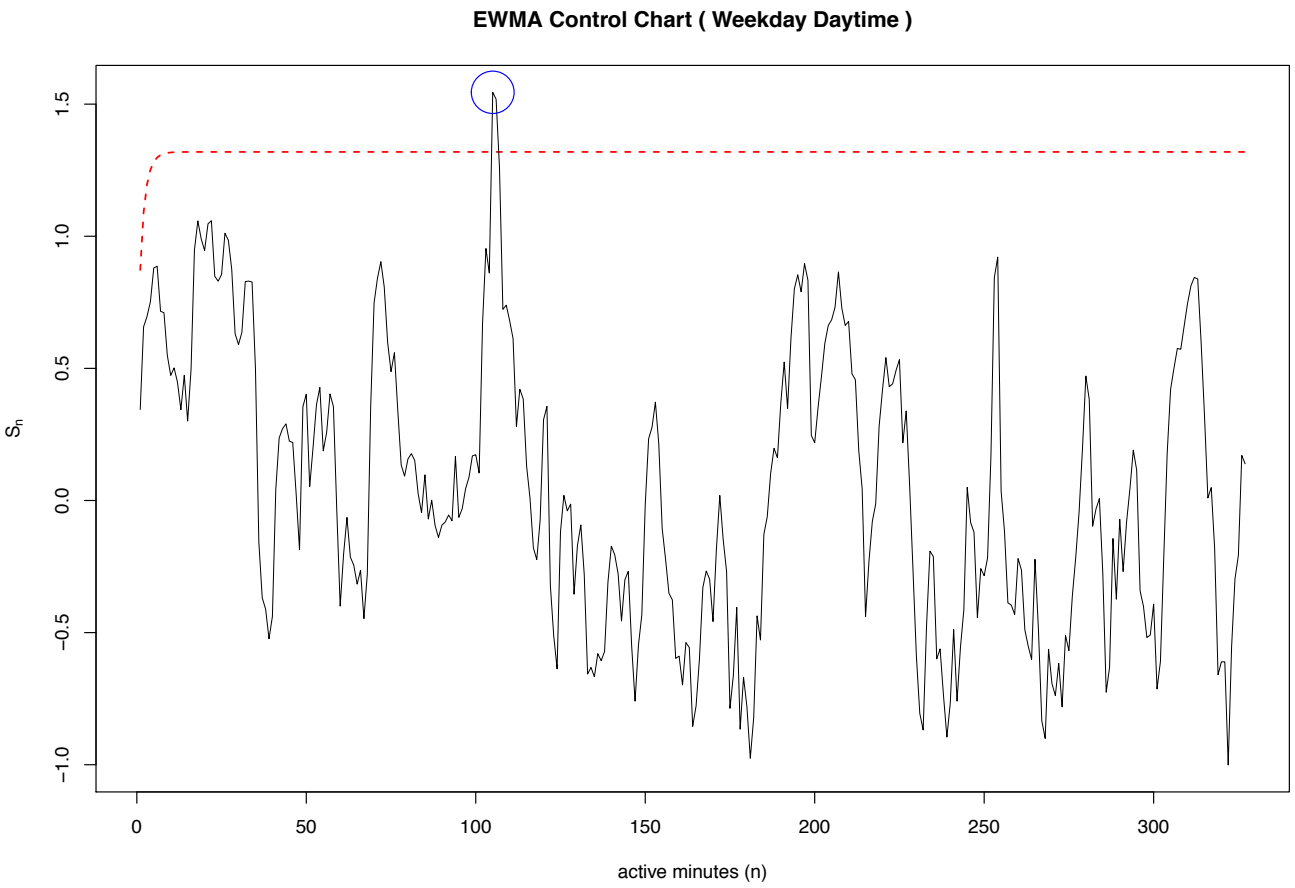


Figure 3.28: EWMA Control Chart - 100.253.230.15 at Weekday Daytime

We detect that an anomaly with count level 203 took place at around $10:00am$ on 27^{th} March, as shown in figure 3.28 and 3.29. We observe from figure 3.29 that this IP had zero activity level after 27^{th} March and this happened to some IPs. We believe this is because lent term ended at the end of March and hence, activity level in the college network plummeted.

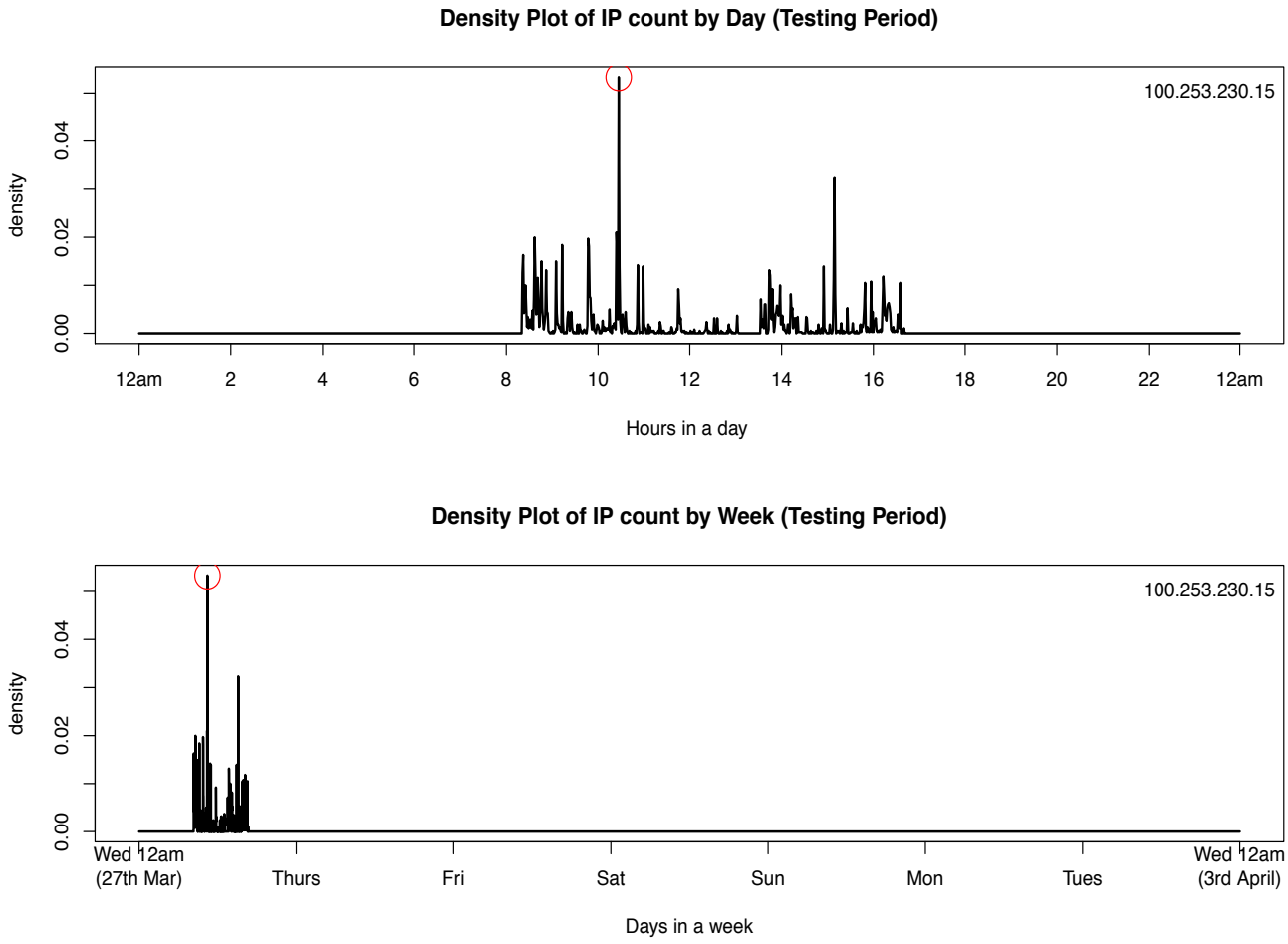


Figure 3.29: Density of counts by day and by week (Testing period)

4

Port Scanning

4.1 Ports observed on a typical day in College

The interest in this section is to build a bayesian model to see which destination ports an IP serves on so that we can detect anomalous behaviour. The focus here lies upon **destination ports**. The goal is to detect anomalous behaviour, when intruders or any IP address use ports that are not often seen being used on a typical day.

To understand what a Port is and what it does, it is essential to know what is an Internet Protocol. An internet protocol is simply a set of rules for communcation between computers. There are many different protocols collectively carrying out the transfer (3, William & Don 2000). To establish a communication between two hosts, there are protocols that use ports. They are the Transport Layer protocol, such as the Transmission Control Protocol (TCP)and User Datagram Protocol (UDP). Port numbers, a 16-bit number are needed to identify various applications or processes running on a single computer and thereby enable them to share the network resources simultaneously. So a port is associated with the host's IP address plus the type of protocol used to facilitate communication.

It is believed that normal IPs within the college tend to use several or more ports on a regular basis. To better understand this, we have written a perl script to look through all the ports used within a typical day in College. Table 4.1 was created based on their occurrence on 4^{th} March. This includes the top 5 ports and a few rarely used ports. It can be seen that on a typical day, Port 53, 80, 443, 1935, 445 are used very often. For instance, port 53 corresponds to name-domain server, i.e. connection to school

server. Port 445 corresponds to Microsoft-DS SMB file sharing. Port 1935 corresponds to Adobe Systems Macromedia Flash Real Time Messaging Protocol (RTMP) . A lot of the ports are rarely used. For instance, port 32967 corresponds to LogMeIn Hamachi (VPN tunnel software)used to connect to Mediation Server, which is a software that connects people together in a LAN-like network so that they can play computer games together. It is also worth noting that on a typical day in College, there are 65184 ports being used.

	Port Number	Frequency
Top 5	53	30720508
	80	21068818
	443	7816857
	1935	7791959
	445	5193460
Uncommon	6609	1
	5077	2
	17	3
	9065	4
	32976	58

Table 4.1: Ports used on a typical day in College - 4^{th} March

4.2 Bayesian Model

4.2.1 likelihood

Let x be a discrete random variable of Port Numbers and since port number is a 16 bit integer, we have $x \in \{0, 1, 2, ..., 2^{16} - 1\}$.

Probability distributions on x can be parameterised by a vector $\boldsymbol{\theta}$, where

$$p(x = k) = \theta_k \Leftrightarrow \;\; P(next\ destination\ port\ = k\ for\ a\ particular\ IP) = \theta_k$$

- *with 2 conditions:* $\sum_{k=0}^{2^{16}-1} \theta_k = 1$ & $\theta_k \in [0, 1]$

The joint probability of N iid samples $\boldsymbol{X} = \{x_1, ..., x_N\}$ follows a multinomial distribution :

$$P(\boldsymbol{X}|\boldsymbol{\theta}) = \prod_{k=1}^{2^{16}-1} \theta_k^{n_k} \quad , where\ \ n_k = \sum_i \mathbb{1}\{x_i = k\}$$

, where n_k counts the of times k^{th} outcome or port number k occurs at a certain point of time. It is important to note that **samples is not equivalent to counts**, because samples occur in a particular order (i.e. specific sequence of outcome). Thus, the normalising factor is one, since there is only 1 ordering. Note that if we are modelling counts, then the joint probabiltiy of set of counts follows a multinomial distribution, which must take into account of all possible permutations:

$$P(n_1, .., n_{2^{16}-1}|\boldsymbol{\theta}) = \begin{pmatrix} N \\ n_1, ...n_{2^{16}-1} \end{pmatrix} \prod_{k=0}^{2^{16}-1} \theta_k^{n_k}$$

4.2.2 Conjugate Prior

A conjugate prior for $\boldsymbol{\theta}$ would be a Dirichlet distribution, $\boldsymbol{\theta} \sim Dirichlet(\boldsymbol{\alpha})$, where $\boldsymbol{\alpha} = \{\alpha_0, ..., \alpha_{2^{16}-1}\}$ & each $\alpha_k > 0$. Laplace smoothing is applied to the hyperparameters α_k and α_k is defined as follow:

$$\alpha_k = \left\{ \frac{\#\ dest.\ port\ k\ observed\ in\ 1\ day + 1}{total\ \#\ dest\ ports\ observed\ in\ 1\ day + 2^{16}} \right\} \times \sum_j \alpha_j$$

The density is:

$$p(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1}$$

with mean:

$$E(\theta_k) = \frac{\alpha_k}{\sum_j \alpha_j}$$

Thus, the posterior distribution: $\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{\alpha} \sim Dirichlet(\boldsymbol{\alpha^*})$, where $\boldsymbol{\alpha^*} = \{\alpha_1 + n_1, ..., \alpha_{2^{16}-1} + n_{2^{16}-1}\}$

Furthermore, the posterior predictive distribution would be:

$$p(x = k|\boldsymbol{X}, \boldsymbol{\alpha^*}) = E[\theta_k|\boldsymbol{X}, \boldsymbol{\alpha}] = \frac{\alpha_k^*}{\sum_k \alpha_k^*} = \phi_k^*$$

The interest in this task is anomaly detection, i.e. to detect whether the IP of interest is connecting to ports that are rarely used.

In our context, an anomalous event is one that is less probable or as probable as ϕ_k^*. Thus, p-value is defined as:

$$p - value(x = k) = f_k = \sum_{j=0}^{2^{16}-1} \mathbb{1}(\phi_j^* \leq \phi_k^*)\phi_j^*$$

The main challenge here is the concept of **continuously updating** the vector $\boldsymbol{\phi_k^*}$ for every observed destination ports associated with the IP of our interest. And all of these are carried out in Perl.

Figure 4.1 depicts the distribution of p-values for several IP address and they clearly do not exhibit a continuous uniform distribution. The problem of discrete p-values resurface in this context again.

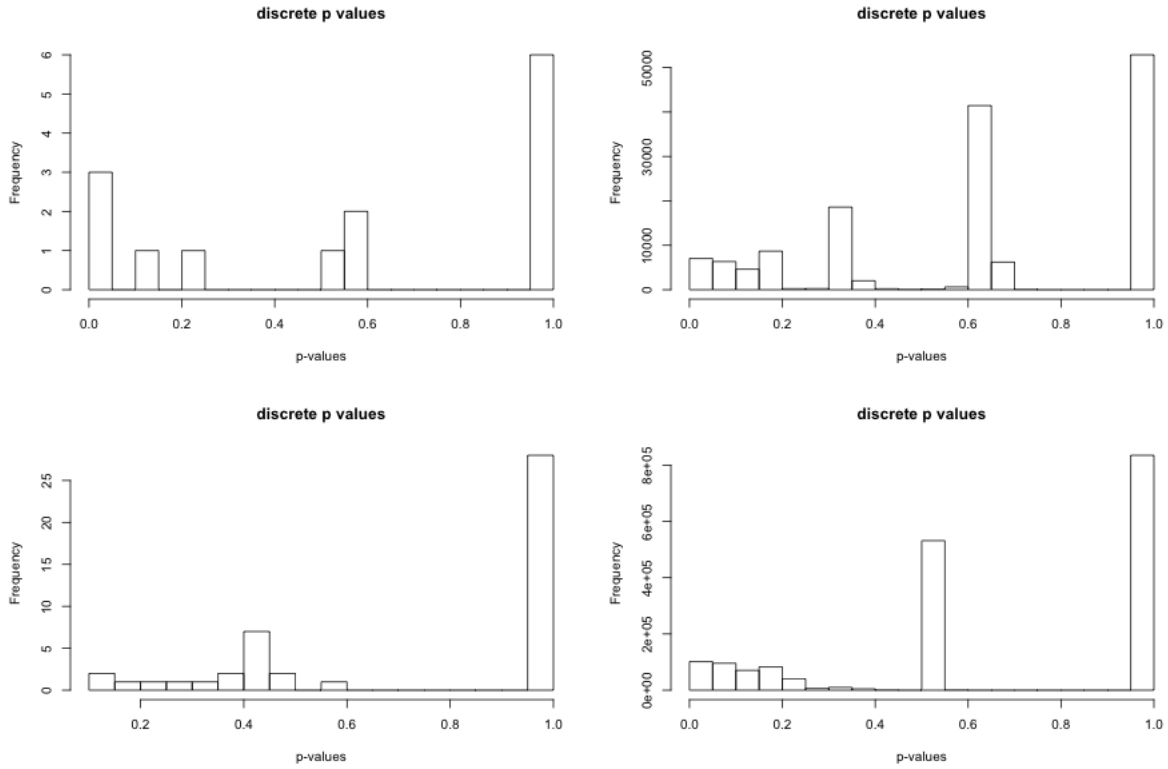


Figure 4.1: Discrete p-values

As mentioned in Chapter 3, discrete p-values can be overcome by interval censoring these discrete p-values and then draw uniform sample f between the interval.

$$f \sim U(f_{k-1}, f_k]$$

, where f_k is defined above, and f_{k-1} is just

$$f_{k-1} = \sum_{j=0}^{2^{16}-1} \mathbb{1}(\phi_j^* < \phi_k^*)\phi_j^*$$

Due to time constraints, we did not find any anomalous behaviour. Nonetheless, the corrected p-values from the IPs that we have examined all have uniform distribution (supported by KS-test p-value > 0.05), as show in the figure 4.2 below.

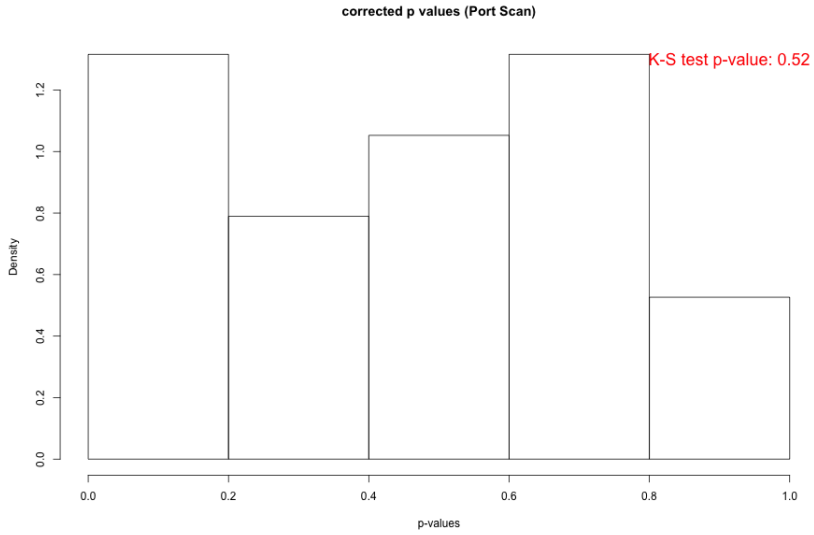


Figure 4.2: Corrected p-values from Port Scanning IP 126.253.232.37

5

Hierarchical Clustering

Since time series count data was collected for two hundred nodes (IP) in chapter 3, the interest in this section is to perform hierarchical clustering such that groups of *similar* objects can be identified and assigned into clusters. Hierarchical clustering techniques can be classified into 2 categories, *agglomerative* methods and divisive methods. Agglomerative methods are used here.

5.1 Agglomerative Methods

In agglomerative clustering, each node represents an individual cluster. Then the two most similar nodes are merged sequentially. The same procedure is carried out repeatedly until there is only one large cluster left, which includes all the nodes.

The symmetric distance matrix captures the similarity between each and every node. The measure of similarity would be based upon a symmetrised Kullback - Leibler divergence, namely the Jensen - Shannon divergence (10, Lin , 1991), which measures the simillarity between the probability distributions of two nodes. The Jensen-Shannon divergence is defined as follow :

$$JSD(P||K) = \frac{1}{2}\{D(P||M) + D(K||M)\}$$

, where $D(P||K)$ is the Kullback-Leibler divergence (8, S. Kullback, 1959). M is the average of the two distributions.

$$D(P||K) = \mathbb{E}[\log(P_i) - \log(K_i)] = \sum_i \log\left(\frac{P_i}{K_i}\right) P_i$$

$$M = \frac{1}{2}(P + K)$$

The square root of the Jensen - Shannon divergence will convert it to a metric (5, Endres, D. M.; J. E. Schindelin 2003) (15, sterreicher, F.; I. Vajda 2003). Note that the Kullback - Leibler divergence (9, S. Kullback and R. A. Leibler, 1951)is defined if

- P & K sum to 1
- $K_i = 0 \Rightarrow P_i = 0$

It can be easily verified that these two properties are also defined for the Jensen-Shannon divergence, since the data we are dealing with are discrete and P & K are probability mass functions. In addition base 2 logarithm (10, Lin , 1991) is used such that $0 \leq JSD(P||K) \leq 1$. A distance matrix of 5 nodes would look like the following:

$$\begin{bmatrix} 0.00 & 0.40 & 0.37 & 0.76 & 0.58 \\ 0.40 & 0.00 & 0.21 & 0.46 & 0.29 \\ 0.37 & 0.21 & 0.00 & 0.48 & 0.33 \\ 0.76 & 0.46 & 0.48 & 0.00 & 0.32 \\ 0.58 & 0.29 & 0.33 & 0.32 & 0.00 \end{bmatrix}$$

In our analysis, the agglomerative clustering procedure chosen is "single linkage", as it is good for outlier detections. The single linkage algorithm is based on minimum distances. The defining feature of this algorithm is that the distance between groups is defined as that of the closest pair of individuals, where only pairs consisting of one individual from each group are considered (6, Brian Everitt (2011), *Cluster Analysis*, Wiley series in probability and statistics)

The dendrogram below 5.2 illustrates not only the fusions made at each stage of the analysis, but also the *height* or distance level at which the nodes and different clusters merge together. Figure 5.1 is created by the *image.plot* function in R. The y-axis represents $y = \lfloor \log_2(x + 1) \rfloor$, where is x corresponds to counts. The x-axis presents a list of IPs in the order of which they merge with a cluster. The colours corresponds to probabilities $p(y)$. The image serves the purpose to justify the fact that agglomerative clustering has done something sensible here. And from these two figures, we can make a decision as to how many clusters we would like to set.

In fact, if we look at first 6 digits of the dotted IP address, we identify that all the IP address belong to either one of the 4 groups: 108.169(2%) , 108.188(3%) , 100.253(51%) and 126.253(44%).

Interestingly, if we set the the number of clusters to be four, we obtain three singleton clusters and a massive cluster of 246 nodes, as shown in figure 5.4. The three singleton clusters are 100.253.43.234 , 100.253.179.128 and 108.169.6.109. Figure 5.3 shows the probability mass function of these three IPs and one IP randomly chosen from the large cluster. We can see that these three IPs have much higher counts than the majority of IPs from the large cluster, which in turn separates them out to form singleton clusters.

We also observe that within the massive cluster, it contains 4 types of IPs 108.169(1.6%) , 108.188(3.25%) , 100.253(50.8%) and 126.253(44.3%). In fact, we observe that the more number of clusters we choose, the more singleton clusters there are. This means maybe the IPs' distributions are not as similar as we thought they might be. This essentially tells us that applying agglomerative clustering on the probability mass functions (*pmf*)does not give interesting results. Although we believe the pmf s are similar as most of their distributions are all right tailed and multi-modal, it may be the case that the counts are distributed at different bins for different IPs such that agglomerative clustering do not treat these IPs as similar objects. Nonetheless, if more time is given, we would try to run clustering on their cumulative distribution functions and see how they behave.

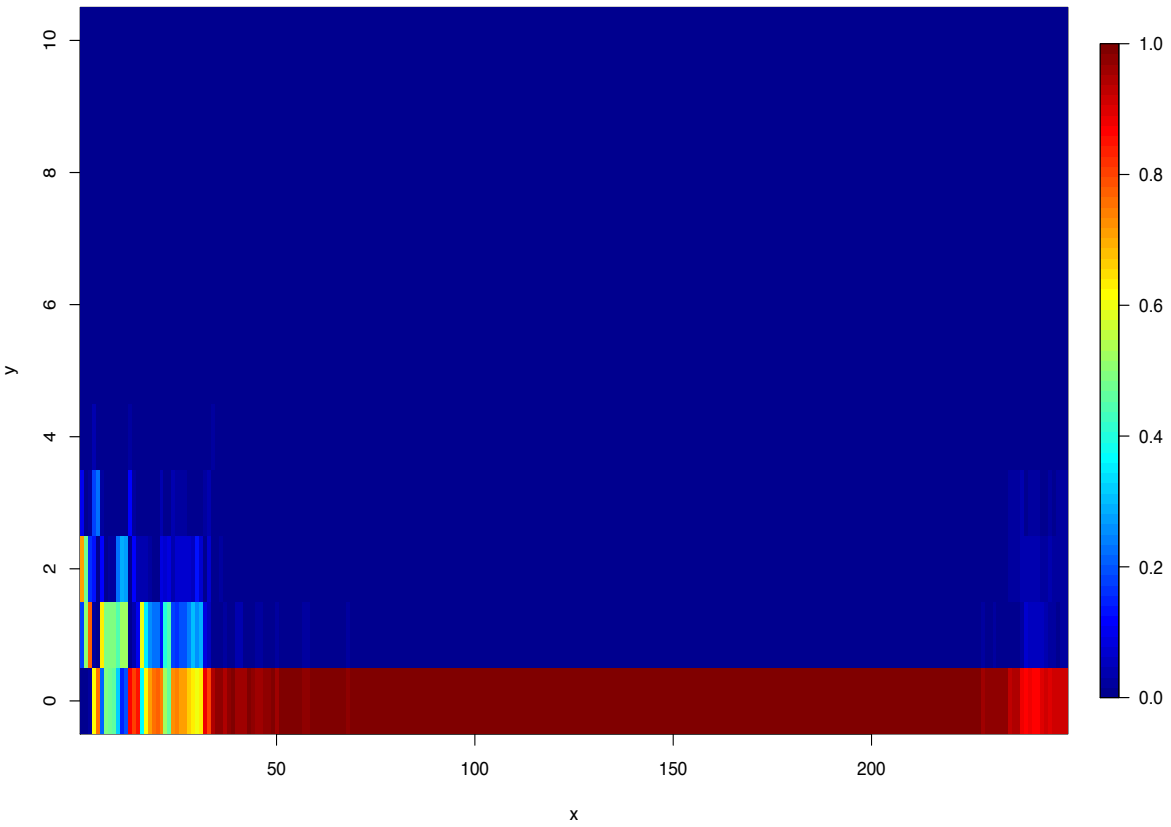


Figure 5.1: Image.plot

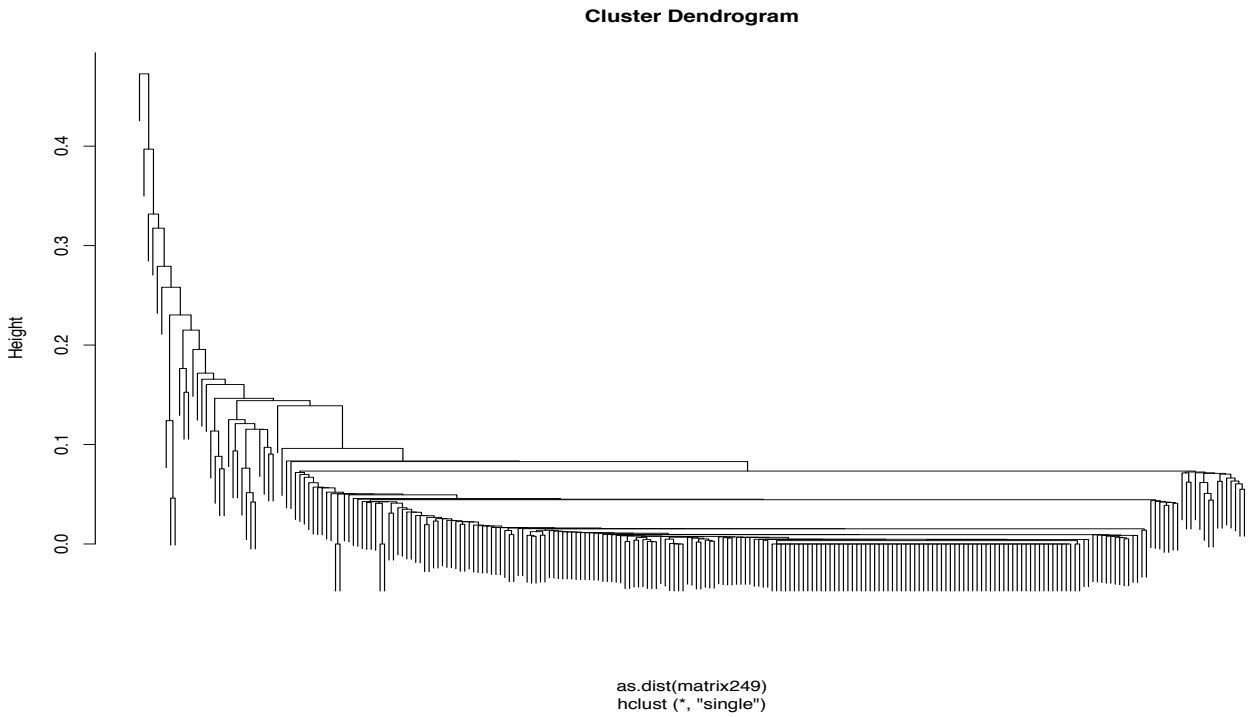


Figure 5.2: Dendrogram - 249 IPs

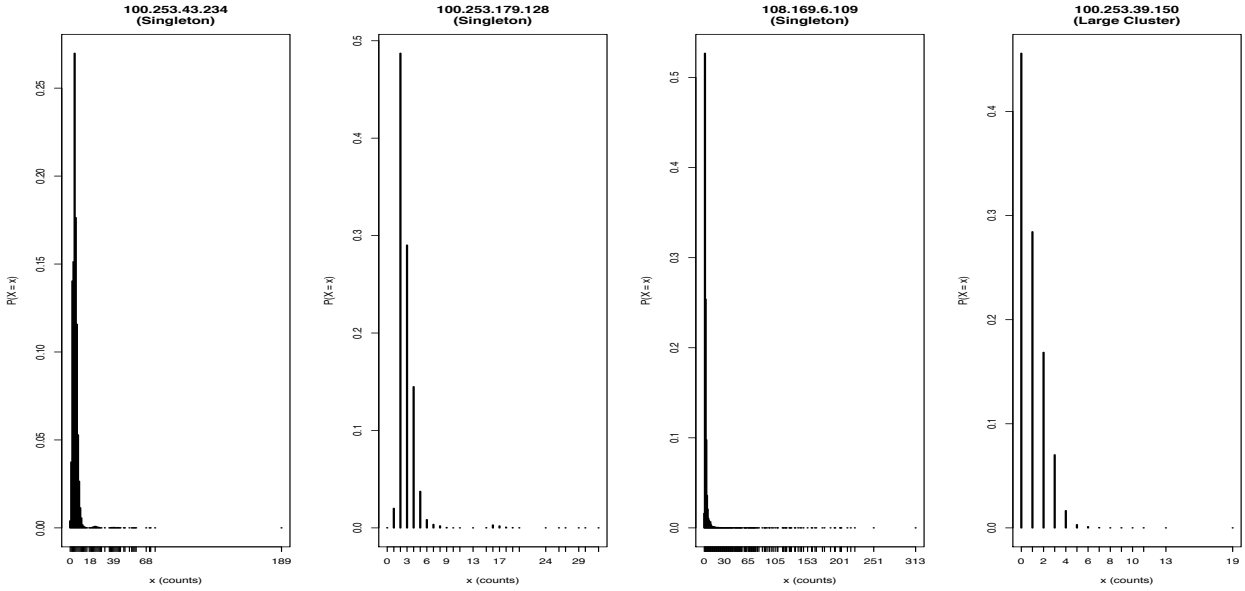


Figure 5.3: Difference in pmf

Figure 5.4: Dendrogram - 4 clusters

6

Conclusion

This project sets out to use statistics to model IP behaviour as well as to detect anomalous behaviour.

In Chapter 2, we successfully decomposed a massive cluster of nodes into smaller ones by means of filtering and thresholding. These smaller clusters may represent different groups of similar users in the college network. However, since the IPs are anonymised, we could not go any further to classify which clusters belong to which department at Imperial.

Chapter 3 developed a discrete time markov chain model, in which the transition matrices were created to monitor IP behaviour at different times in a day. We also conclude that we could not use simple discrete distributions or even mixture distributions to fit the active count data. Thus, empirical distributions were used to train and test the data. We overcame the problem of discrete p-values. We detected anomalies for several IPs. However, we need to be concerned about the problem of false alarms. Although we can always raise the value of parameter L in the control limit equation to prevent false alarms, by how much we should increase L is controversial and this requires a knowledgeable person who truly understands the college network. In addition, the corrected p-values computed from the testing period were not uniformly distributed for some IPs, which suggested the data from testing period follows a very different distribution from the empirical distribution in training period. We believe this was because lent term ended at the end of March 2013, which led to a drastic change in activity level in the testing period. A solution that we recommend would be to collect more data such that we can train our model in March 2011 and test it upon

data from March 2012 and March 2013. Furthermore, it is questionable whether there are no anomalies in the training period, which may invalidate our model assumption.

In Chapter 4, we built a bayesian model for port scanning. The major difficulty was to cotinuously update the p-values vector for every observation in the netflow data, which was all done in Perl. Nonethesless, given limited amount of time, we were only able to analyse a few IPs, and no anomalies were found.

Chapter 5 applied agglomerative clustering based on IPs' probability distribution of count. We chose four clusters and three of them were singleton clusters. Looking at the distribution of counts of these singletons, we observed they all had very high counts. Furthermore, the more number of clusters we chose, the more singleton clusters we got. Hence, we conclude that working with probability mass functions do not give interesting results, as their pmfs maybe very different from one another. If more time is given, we can perform clustering on the IP's cumulative distribution funcitons and see how they behave.

In conclusion, these are all interesting results. If time permits, I would love to continue with this project to obtain more results from the dataset to understand more about IP behaviour.

7

Appendix

7.1 Tables and Graph

7.1.1 Chapter 2 - Table& Graphs

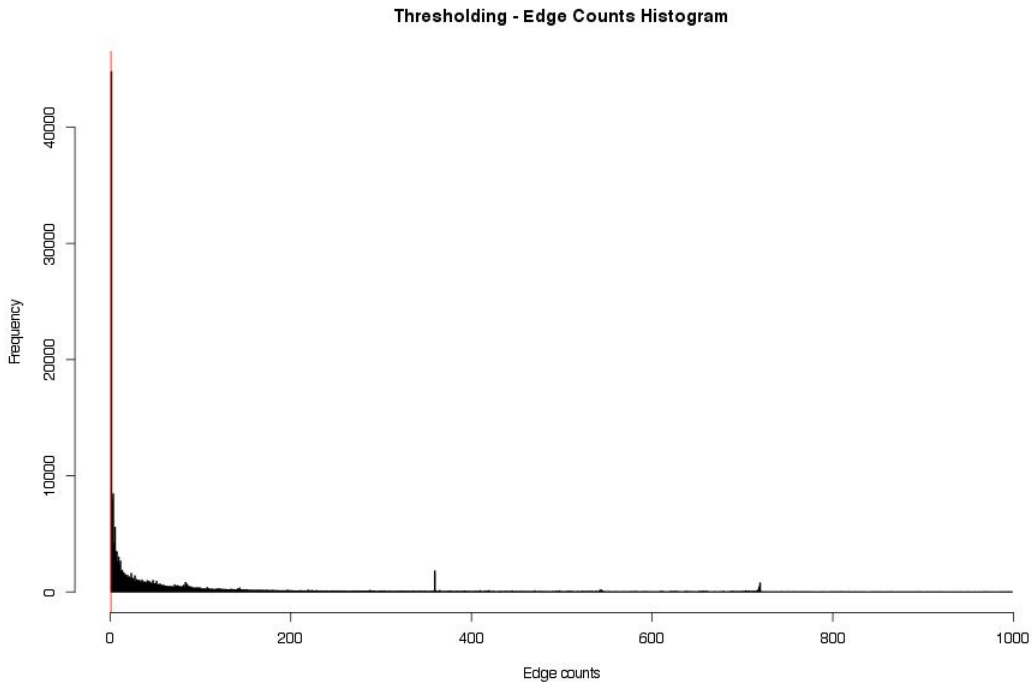


Figure 7.1: Filtering Edge Counts - 4th March College IPs

Number of Cluster:	18									
Individual Cluster size:	15769	3	3	2	3	2	4	2	2	2
	2		2	2	3	2	2	2	2	

Table 7.1: igraph output: 4th March College IPs; before filtering and thresholding

Number of Cluster:	683									
Individual Cluster Size:	1665	11	3	7	62	3	31	2	5	5
	10	76	5	45	2	3	2	2	8	3
	10	38	6	2	4	41	3	17	3	3
	3	10	5	14	2	6	3	15	6	28
	2	25	3	14	5	2	3	4	5	3...

Table 7.2: igraph output: 4th March College IPs; after filtering and thresholding

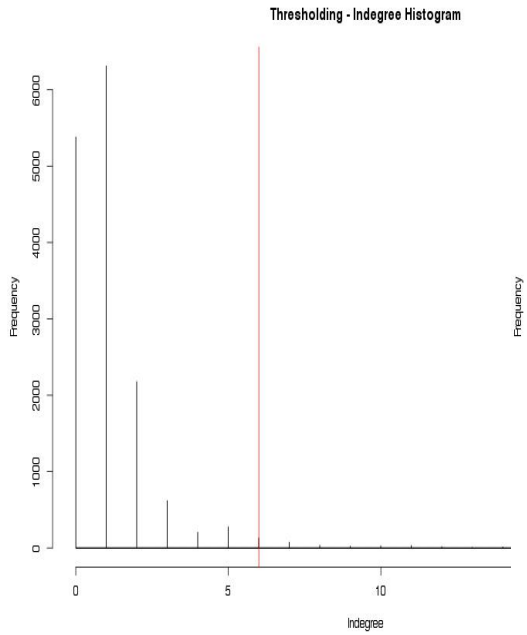


Figure 7.4: Filtering Edge Counts - 4th March College IPs

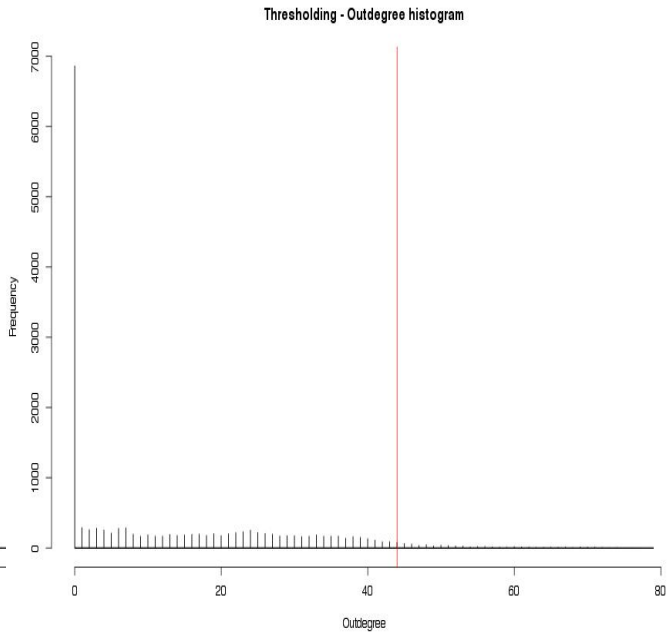


Figure 7.5: Filtering Edge Counts - 4th March College IPs

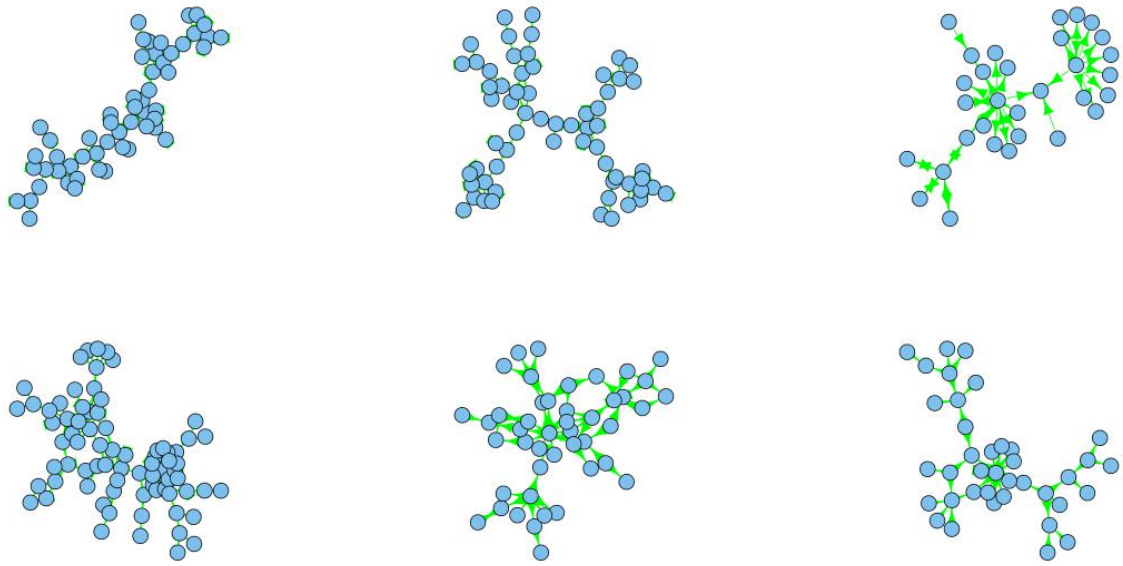


Figure 7.6: Directed Graph - 4th March College IPs

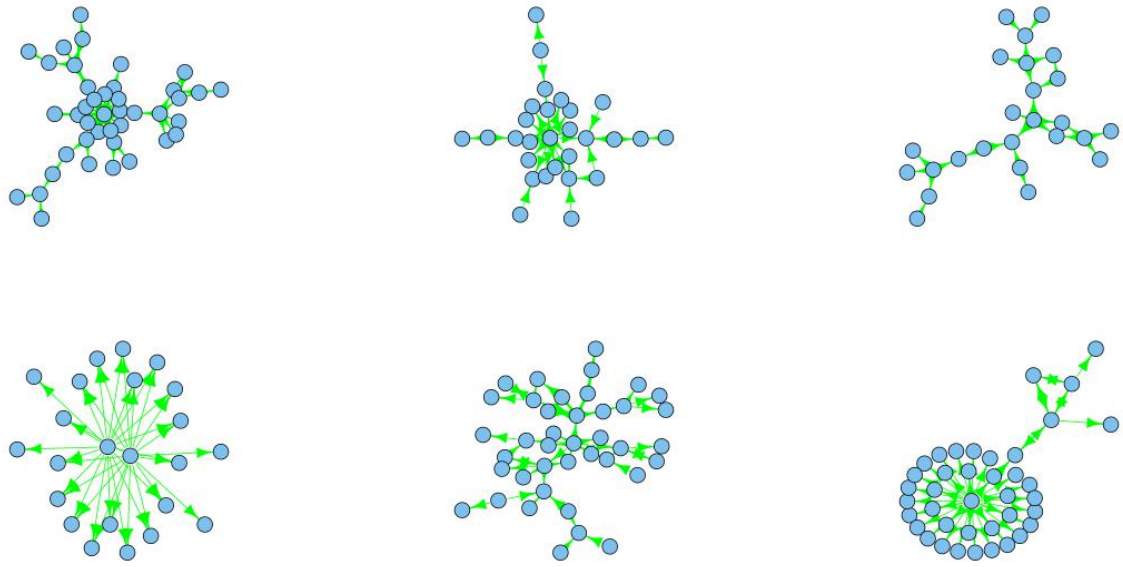


Figure 7.7: Directed Graph - 4th March College IPs

Number of Clusters:	40									
Individual Cluster size:	1334863	23617	11	2	2	9	4	23	2	9
	2	5	2	2	5	3	3	2	3	9
	2	6	2	5	2	2	2	3	2	2
	2	2	9	2	2	3	9	2	2	2

Table 7.3: igraph output: 4th March College & Non-College IPs; before filtering and thresholding

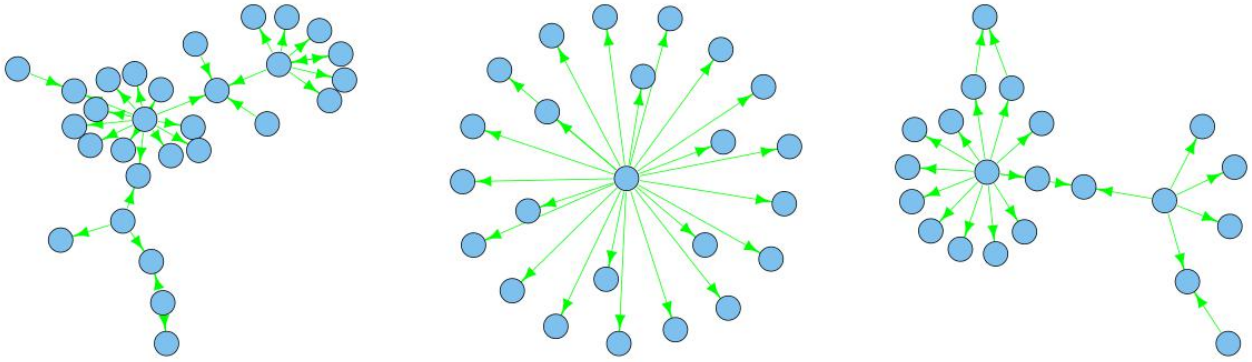


Figure 7.8: Directed Graph - 4th March College IPs

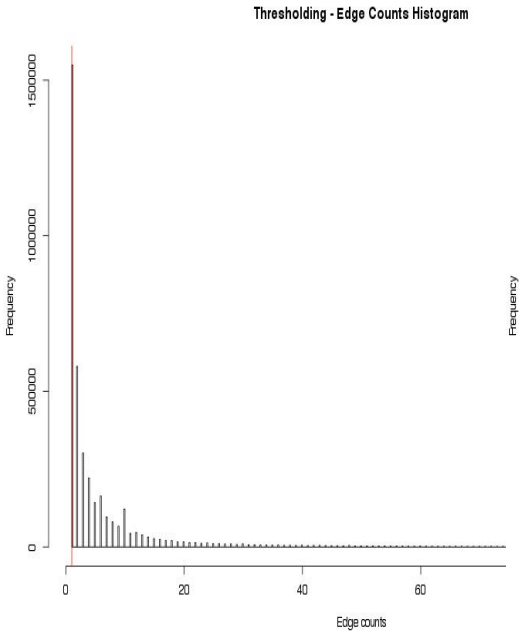


Figure 7.11: Filtering Edge Counts - 4th March College & Non-College IPs

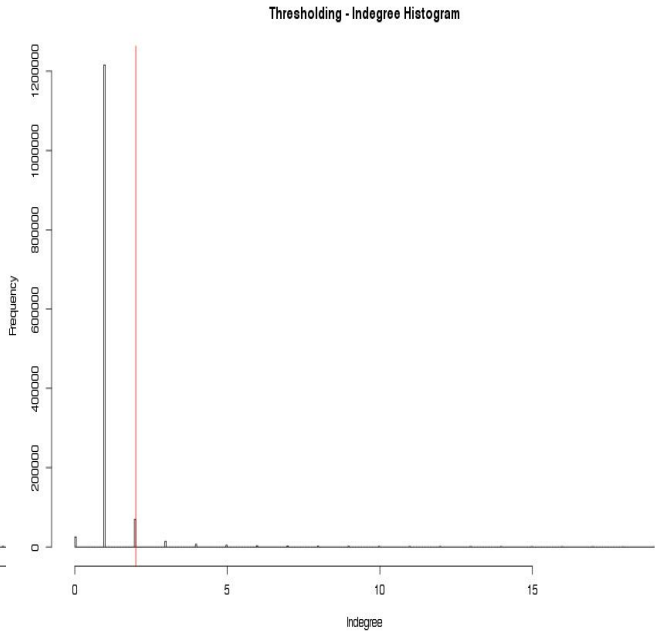


Figure 7.12: Filtering Edge Counts - 4th March College & Non-College IPs

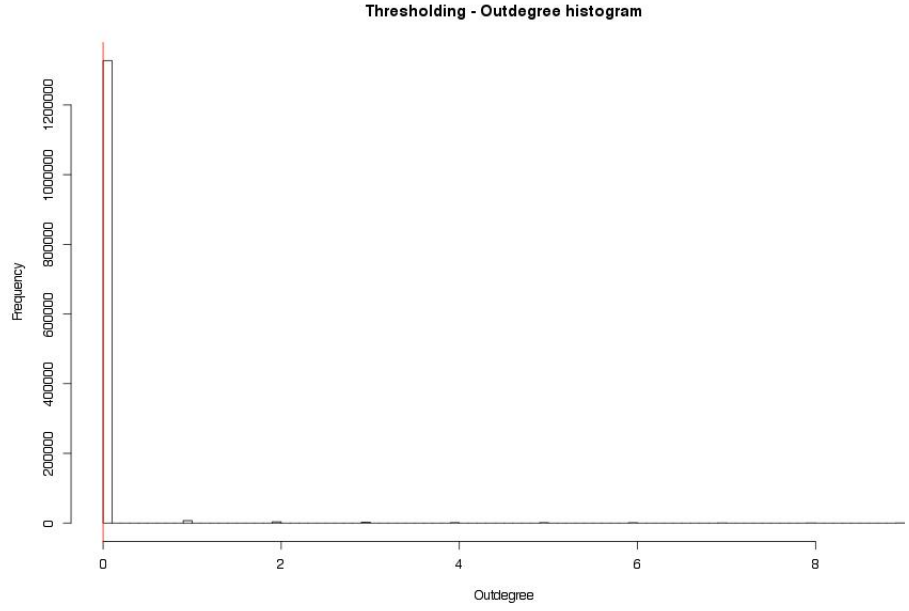


Figure 7.13: Filtering Edge Counts - 4th March College & Non-College IPs

Number of Cluster:	5186								
Individual Cluster size:	378	7762	72040	9793	31126	1374	14	34611	9
	18492	4	106	3744	1685	77	2295	25	540
	783	11967	2787	12	40	98	2	659	185
	2220	290	1723	11	42	10174	9069	449	1110
	8	413	605	15	218	7221	223	1668	8243 ...

Table 7.4: igraph output: 4th March College & Non-College IPs; after filtering and thresholding

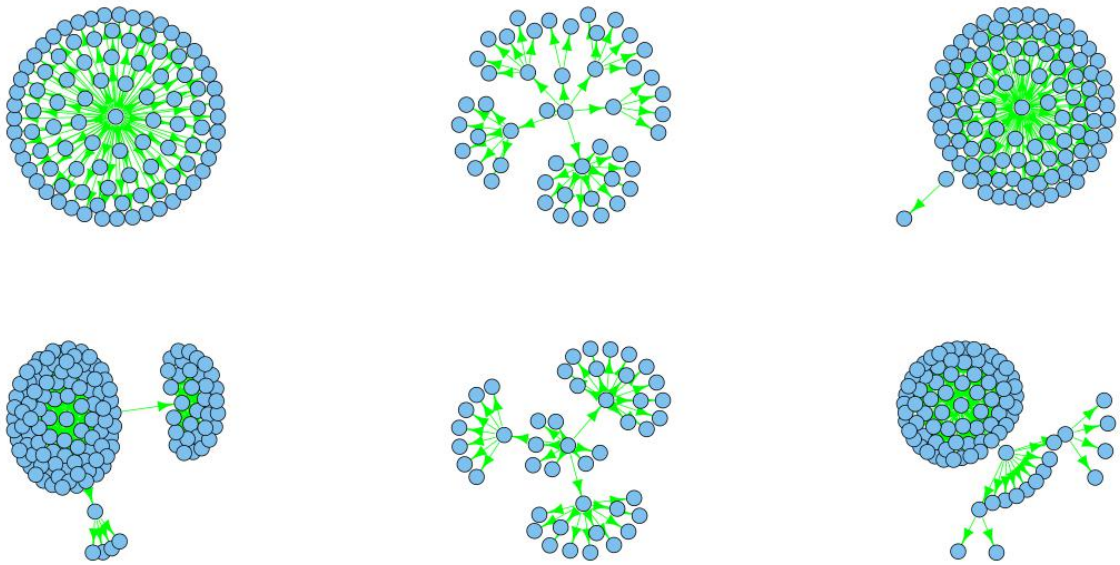


Figure 7.14: Directed Graph - 4th March College & Non-College IPs

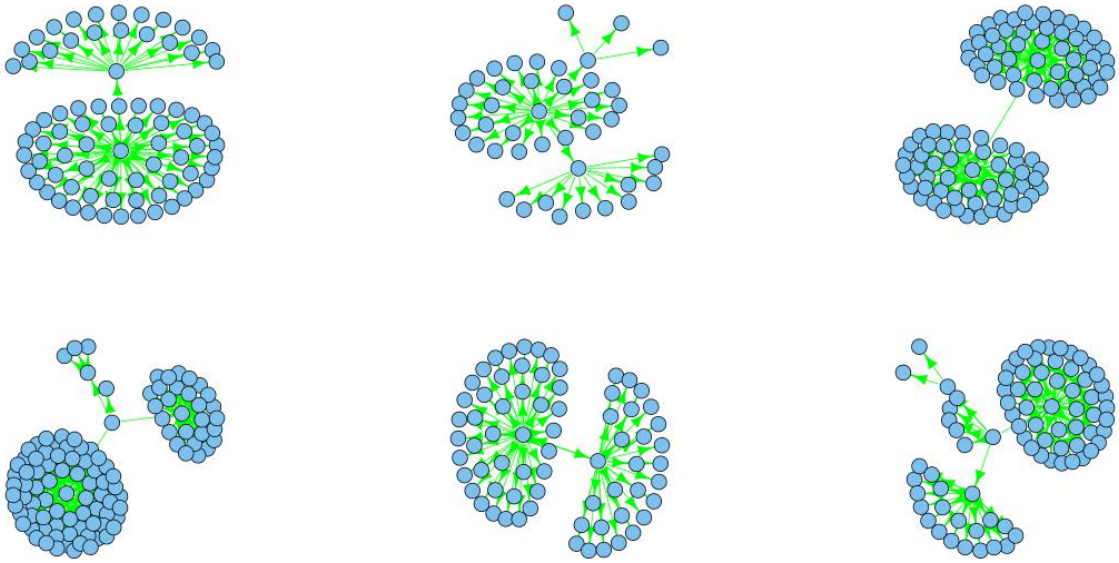


Figure 7.15: Directed Graph - 4th March College & Non-College IPs

7.1.2 Chapter 3 - IP with Normal behaviour

Transition matrices and density plot of counts of 0002130548595/126.253.151.115 are shown here.

	Night time	Weekday Daytime	Weekend Daytime
	I A	I A	I A
100.253.170.35	$I \begin{pmatrix} 0.99 & 0.01 \\ A & 0.44 & 0.56 \end{pmatrix}$	$I \begin{pmatrix} 0.99 & 0.01 \\ A & 0.01 & 0.99 \end{pmatrix}$	$I \begin{pmatrix} 0.99 & 0.01 \\ A & 0.04 & 0.96 \end{pmatrix}$

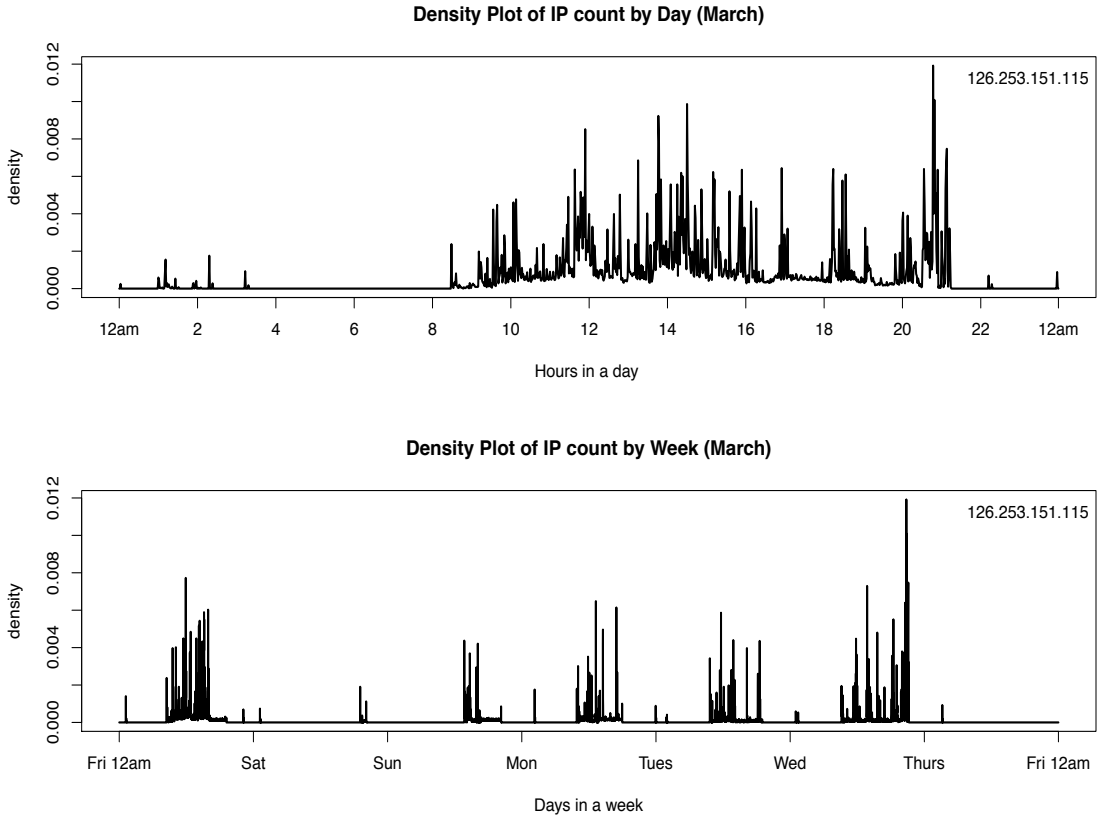


Figure 7.16: 126.253.151.115

7.1.3 Chapter 3 - EM algorithm for Poisson-Geometric mixture distribution

Let $z \in 1, 2$, and same as before $p(z_i = j) = \phi_j$.

$$x_i|z_i = 1 \sim Poisson(\lambda) \quad x_i|z_i = 2 \sim Geometric(p)$$

1. We obtain 2 equations from the E-step:

$$w_1^i = \frac{Poi_i(\lambda)\phi_1}{Poi_i(\lambda)\phi_1 + Geometric_i(p)\phi_2} \qquad w_2^i = \frac{Geometric_i(p)}{Poi_i(\lambda)\phi_1 + Geometric_i(p)\phi_2}$$

2. in the M-step, we have these 3 updating equations:

$$\lambda = \frac{\sum_{i=1}^m x_i w_1^i}{\sum_{i=1}^m w_1} \qquad p = \frac{\sum_{i=1}^m w_2^i}{\sum_{i=1}^m w_2^i x_i + \sum_{i=1}^m w_2^i} \qquad \phi_j = \frac{\sum_{i=1}^m w_j^i}{m} \text{ for } j = 1, 2$$

7.1.4 Chapter 3 - P-value is a random variable

Here we address what the **simple** null hypothesis is under a hypothesis testing framework and how uniformly distributed p-value under the hypothesis testing framework can be treated as a test statistic in our context to test whether the data truly follows a proposed distribution. Following the paper's approach, we will consider a one-sided t-test with $N(\mu, \sigma^2)$ data with

$$H_0 \ : \ \mu \leq 0 \qquad \text{VS} \qquad H_1 \ : \ \mu > 0$$

(12, Duncan J Murdoch 2008) And both μ and σ^2 are unknown. Since we are interested in μ, the test statistic is $T = \bar{X}/(s/\sqrt{n})$, where $\bar{X}$ and s ares the sample mean and sample standard deviation. Under the boundary case $\mu = 0$ in H_0, $T \sim t_{n-1}$. Suppose the *true* distribution of data follows $N(\mu, 1)$, with $\mu = -0.5, 0, 0.5,$, we then simulated 20000 observations from the normal distribution for each of the μ . (12, Duncan J Murdoch 2008) Here, it is worth noting that as the sample size n increases, the degree of freedom of the t-distribution increases, the t-distribution approaches the standard normal distribution. Consider the case where $\mu = 0$, i.e. under the *simple* null hypothesis, given that we are simulating 20000 observations, the t-distribution approaches the normal distribution and thus they overlap in the topleft of figure 7.17. The corresponding p-values plot are uniformly distributed on $[0, 1]$. Note that p-values are computed by $P(T \geq x)$, where x s are the simulated data.

If the data follows the true distribution $N(0.5, 1)$, the normal distribution is now shifted to the right (second row of figure 7.17). There are higher frequency of extreme observations compared to the previous case where $\mu = 0$ and hence, high proportion of low p-values.

If the true distribution is $N(-0.5, 1)$, it makes sense that there should be fewer extreme observations compared to the previous case where $\mu = 0$, and hence high proportion of high p-values.

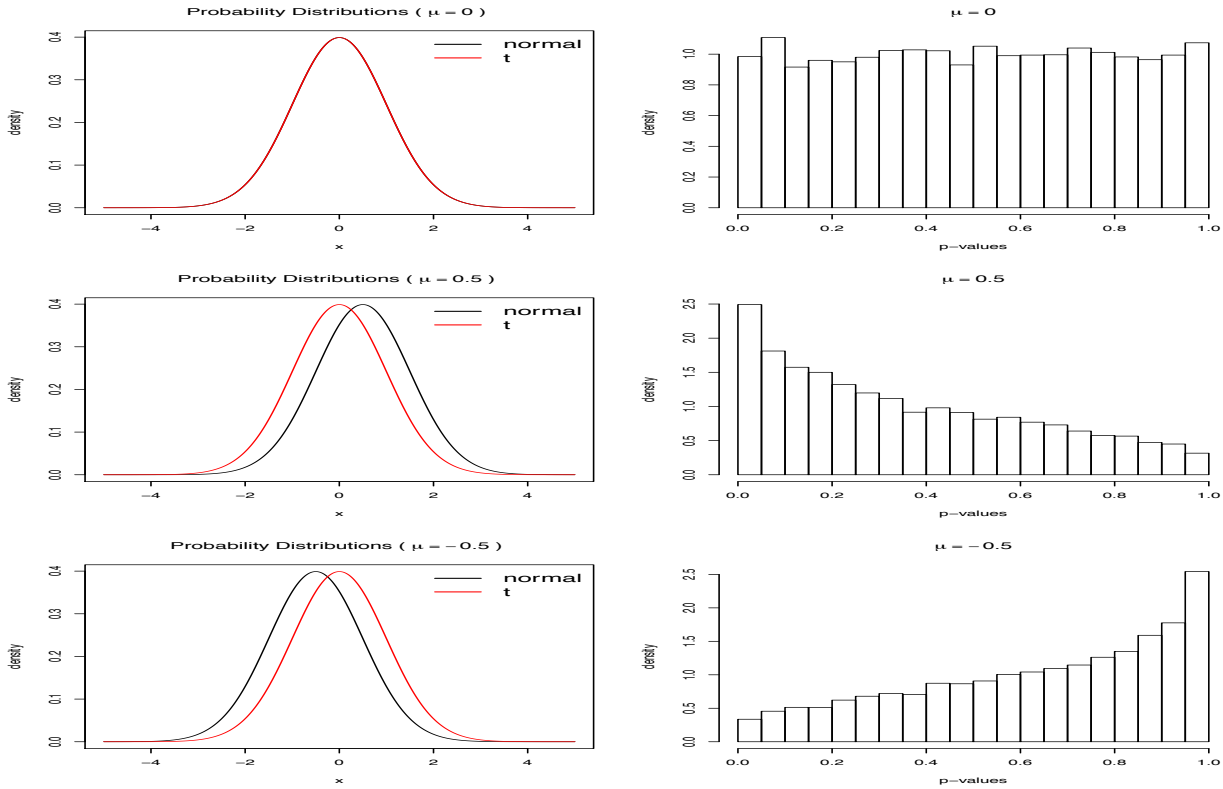


Figure 7.17: Simulation

7.2 Code

This section includes samples of perl scripts, shell scripts and R scripts used.

- Sample shell script for identifying college ips and collecting corresponding count data in April.

```
#!/bin/sh
time /usr/local_machine/bin/nfdump -R /home/flowdata/ac-core/2013/04
-o pipe | awk -F'|' 'function college(ip){ return
 (ip>=2130509824&&ip<=2130575359)|| (ip>=1694302208&&ip<=1694367743)||
  (ip>=1823014912&&ip<=1823080447)|| (ip>=1824260096&&ip<=1824325631)||
 (ip>=1045335552&&ip<=1045335807)||(ip>=1045216000&&ip<=1045216255)} {if($7==0
 && $12==0 && college($10) && college($15)){print $7$8$9$10,$12$13$14$15}}' |
 awk '{ec[$0]++; sc[$1]++; dc[$2]++; uc[$1]++; uc[$2]++}{ if(ec[$0]==1){sdc[$1]++
 ;ddc[$2]++ } }  END
  { for  (a in ec){ print a > "src_dst_ips.txt"; print ec[a] >
 "src_dst_ips_counts.txt" } } END { for (a in uc){ print a > "ips.txt";
 print uc[a] > "ips_counts.txt"; print sc[a]+0 >
 "src_ips_counts.txt"; print sdc[a]+0 > "src_ips_degrees.txt"; print
 dc[a]+0 > "dst_ips_counts.txt"; print ddc[a]+0 >"dst_ips_degrees.txt" } }'
```

- R code for converting IP in decimal format to dotted quad format

```
#108.169.14.35 -> 1823018531  #converting IPv4 address to a decimal
to.decimal <- function(string) {
string <- as.character(string); y <- unlist(strsplit(string, "[:.:]"))
return {
as.integer(y[1])* 16777216 + as.integer(y[2]) *
 65536 + as.integer(y[3]) * 256 + as.integer(y[4]) } }
# 1823018531 -> 108.169.14.35  #convertting decimal to dotted quad
to.dotted.quad <- function(v) {
int <- v; vec <- c()
#for(j in 1:length(v)) {
#int <- as.integer( unlist(strsplit(fnn[v[j]], "[:_.txt:]"))[1])
four <- int %% 256; three <- int %/% 256 %% 256;
two <- int %/% 256 %/% 256 %% 256; one <- int %/% 256 %/% 256 %/% 256 %% 256
```

```
vec <- append(vec, paste(one,".",two,".",three,".",four, sep = "" ));
cat(paste(one,".",two,".",three,".",four, sep = "" ), "\n")
#}
return
{
list(vec)
}
}
```

- Sample Perl script for sorting IPs address based on source counts

```
#!/usr/bin/perl
use strict;
use warnings;
use File::Find;
use Cwd;
my $dir = getcwd;
use File::Find;
use Cwd;
my %src = ();
my $count = 0;
#store IP + corresponding value into hash
open(IP , "ips.txt");
open(SRC, "src_ips_counts.txt");
while(defined($_ = <IP>)) {
my $ip = $_;
my $src_count = <SRC> ;
 $src{$ip} = $src_count ; }
close IP;
close SRC;
#filter out IPs that are non-College
open(I , ">filtered_ips.txt"); open(S , ">filtered_src_ips_counts.txt");
while(my ($key, $value) = each(%src) ) {
if ( ($key >= 2130509824 && $key <= 2130575359)||( $key >= 1694302208 && $key <=
 1694367743)||( $key >= 1823014912 && $key <= 1823080447)||($key >= 1824260096 &&
  $key <= 1824325631)||($key >= 1045335552 && $key <= 1045335807)||($key >=
```

```
   1045216000 && $key <= 1045216255))
{
chomp($key);
chomp($value);
$count += 1;
print "$key"."\t"."TRUE\t"."$value\n";
print "Line = $count\n";
} else {
print "FALSE";
delete $src{$key};
}
}
foreach my $key (sort { $src{$b} <=> $src{$a} } keys %src) {
print I "$key";
print S "$src{$key}";
}
```

- Sample Perl Script for Collecting Minute count data in March (for Section 3)

```
#!/usr/bin/perl
use strict;
use warnings;
use diagnostics;
use File::Find;
use Cwd;
#global var
my $flow_data_path= "/home/flowdata/ac-core/2013/03/";
my $nfdump_command="/usr/local_machine/bin/nfdump -o pipe -r";
my $dir = getcwd;
sub process_node {
    my $node = $_[0];
    my $outgoing_file= "$dir/$node"."_.txt";
    my $incoming_file= "$dir/_$node".".txt";
    $node += 0;
    unlink($outgoing_file, $incoming_file);
    #pass by reference
    my $wanted = sub{&eachFile(\$node, \$outgoing_file , \$incoming_file)};
```

```
    find ({wanted => $wanted, preprocess =>
     \&whichFiles}, $flow_data_path);
}
sub whichFiles { return sort @_; }
sub eachFile {
    my ($ref_node, $ref_outgoing_file , $ref_incoming_file) = @_;
    if(/^nfcapd\./){
        #awk -F field separator defined by |
    system("$nfdump_command $File::Find::name
     | grep $$ref_node | awk -F'|' '{print
      \$7\$8\$9\$10,\$12\$13\$14\$15}' |
      awk 'BEGIN {src_count=0; dst_count=0}
       {if(\$1==$$ref_node) {src_count++}}
        {if(\$2==$$ref_node) {dst_count++}} END
         {print src_count >> \"$$ref_outgoing_file\"; print dst_count >>
          \"$$ref_incoming_file\"}' ");
    }
}
my $IP_list_file="2323.txt";
open(IPS,$IP_list_file);
while(defined($_ = <IPS>)){
    chomp;
    &process_node($_);
}
```

Bibliography

Mark L. Berenson, David M. Levine, and Timothy C. Krehbiel. Basic business statistics: Concepts and applications, 2005. `http://courses.wcupa.edu/rbove/Berenson/10th%20ed%20CD-ROM%20topics/section12_5.pdf`. 35

Herman Chernoff and E.L. Lehmann. The use of maximum likelihood estimates in χ^2 tests for goodness of fit. *Ann.Math.Statist*, 25:579–586, 1954. 35

William S. Cleveland and Don X.Sun. Internet traffic data. *Journal of the American Statistical Assoication*, 95:451:979–985, 2000. 2, 69

Richard Durrett. *Essentials of Stochastic Processes*. Springer, 1999. 29

D. M. Endres and J. E. Schindelin. A new metric for probability distributions. *IEEE Trans. Inf. Theory*, 49 (7):1858–1860, 2003. 76

Brian S. Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. *Cluster Analysis 5th ed.* John Wiley & Sons, Ltd, 2011. 76

N.A. Heard and M.O.M. Turcotte. *Data Analysis for Network Cyber Security*, chapter Monitoring a device in a communication network. Imperial College Press. 44, 45, 49

S. Kullback. Information theory and statistics. *Wiley, New York*, 1959. 75

S. Kullback and R. A. Leibler. On information and suffciency. *Ann. Math. Stat.*, 22:79–86, 1951. 76

J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Inform. Theory*, 37(1):145–151, 1991. 75, 76

BIBLIOGRAPHY

Douglas C. Montgomery. *Introduction to Statistical Quality Control*. Hoboken, New Jersey: John Wiley & Sons, 2005. 49

Duncan J. Murdoch, Yu-Ling Tsai, and Jams Adcock. P-values are random variables. *The American Statistician*, pages 242–245, 2008. 42, 91

Benjamin Poulain and Maxime Vantorre. nfdump - manual, 2006. `http://www.linuxcertif.com/man/1/nfdump/`. 3

A.W. van der Vaart. Asymptotic statistics. *Cambridge University Press*, 1998. 46

F. sterreicher and I. Vajda. A new class of metric divergences on probability spaces and its statistical applications. *Ann. Inst. Statist. Math.*, 55 (3):639–653, 2003. 76