

Advanced Process Mining

Assignment Part 2 - Report RWTH Aachen – Summer Semester 2020

Tobias Brockhoff
Lisa Mannel MSc
Sebastiaan J. van Zelst MSc PhD





Submitted By:
Praveen Yadav (391288)
Malek Alhelwany (391558)

Report - Assignment Part 2


This report is about a *fine management process* which was followed by a local police department. Different areas of the process were analyzed, in the starting to get the idea of the process preliminary analysis was done. Then in later parts, we explored a bit deeper to find the bottlenecks in the process and the cases which were not compliant with the recommended process model. And lastly some advices were given to improve the process, so that the police department can handle future cases efficiently.

Q1. Preliminary Analysis

In this section some rudimentary tasks were performed to get a picture of the process.

- a)  From the information given by the police department, it was clear that the whole log should be divided into two sublogs, one with containing events related to Appeal other without those. For this we used Disco Tool. Disco is a lightweight process mining tool, which is also efficient.
- To create separate logs, we used Disco's Attribute Filter and specified activities ["Insert Date Appeal to Prefecture", "Send Appeal to Prefecture", "Receive Result Appeal from Prefecture", "Notify Result Appeal to Offender", "Appeal to Judge"]. In the case *with appeal* we got the cases where at least one of the above activities happened and in the other *without appeal* we remove all the cases where one of these activities happened.
- Both sublogs can be found in the same zip folder where this report was placed. From now on we refer With_Appeal_Log as the logs containing the events with appeal related events and the other sublog as Without_Appeal_Log which does not contain any events related to appeal.
- b)  Below table contains some basic statistics about the three logs, Original_Log (original log file which was given), With_Appeal_Log, Without_Appeal_Log. This information was also collected using Disco tool except for the last row. For last row python code was used along with PM4PY library. Code is available in the same zip folder as the report.

Information	Original_Log	With_Appeal_Log	Without_Appeal_Log
Number of traces and trace variants	150370, 231	4567 and 189	145803 and 42
Number of events and average trace length	561470, 48.8 weeks	30119, 15.5 months	531351 and 48.2 weeks
The covered timespan of the log	01.01.2000 00:00:00 to 18.06.2013 00:00:00	03.01.2000 00:00:00 to 14.06.2013 00:00:00	01.01.2000 00:00:00 to 18.06.2013 00:00:00
Number of activities	11	11	6
The fraction of traces containing an activity multiple time	0.0501	0.0188	0.0511


- c)  Below table shows statistics data about the activities involved in all the three logs. This information was also collected using the Disco Tool, the frequency of activities is available under Statistics tab, if you choose Activity in left side.

Activity	Original_Log	With_Appeal_Log	Without_Appeal_Log
Create Fine	150370	4567	145803
Send Fine	103987	4567	99420
Insert Fine Notification	79860	4411	75449
Add penalty	79860	4411	75449
Payment	77601	969	76632
Send for Credit Collection	59013	415	58598
Insert Date Appeal to Prefecture	4188	4188	NA
Send Appeal to Prefecture	4141	4141	NA
Receive Result Appeal from Prefecture	999	999	NA
Notify Result Appeal from Prefecture	896	896	NA
Appeal to Judge	555	555	NA

Below table contains the fraction of traces of each log where an activity was the first/last activity. To calculate the below information, Endpoint Filter was used in the Disco Tool.

Activity	Original_Log	With_Appeal_Log	Without_Appeal_Log
Create Fine	1.0	1.0	1.0
Send Fine	0.13	~0.01	0.14
Insert Fine Notification	0	0	0
Add penalty	0	0	0
Payment	0.44	0.16	0.45
Send for Credit Collection	0.39	0.08	0.40
Insert Date Appeal to Prefecture	0	0	NA
Send Appeal to Prefecture	0.02	0.68	NA

Receive Result Appeal from Prefecture	~0.01	0.01	NA
Notify Result Appeal from Prefecture	~0.01	0.01	NA
Appeal to Judge	~0.01	0.02	NA

- d)  Dotted chart was created using the Prom Tool, it is a basic functionality to visualize any log. So below are two dotted charts one with the activities over time other with activities over time duration.

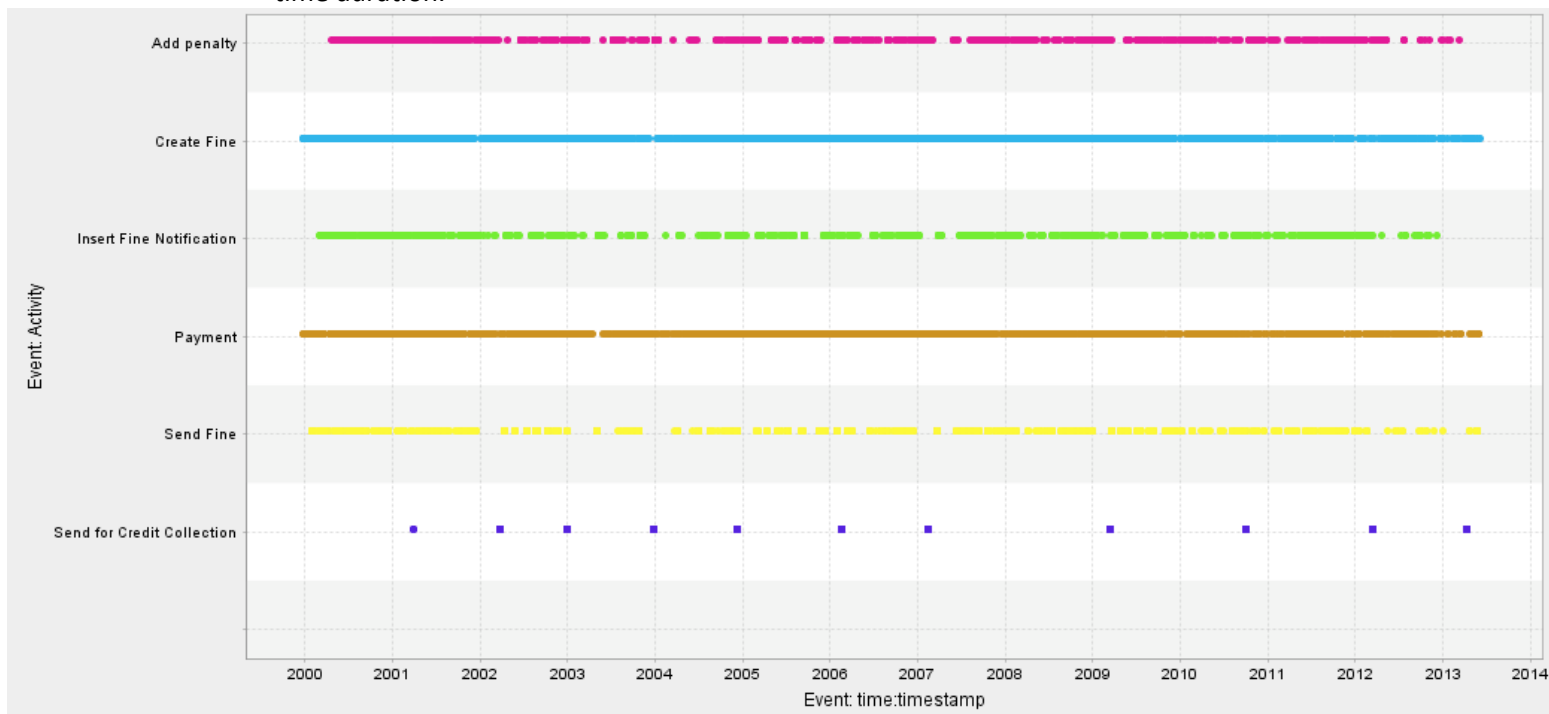


Figure 1 Shows Activity over time.

Figure 1 is showing a dotted chart of activities over time. We observed that while all the activities happened regularly throughout the year, “Send for Credit Collection” happened once in the whole year. This suggest that the police department has dedicate certain period of the year to do this event. And we noticed in one more dotted chart that our assumption was true, if you see the dotted chart of the Traces over time below (Figure 2), you will find vertical line patterns which were also highlighted with a rounded rectangle. That means multiple cases were handled in batches during this period which were not finished until up to that time. There was “Payment” activity highlighted using vertical ellipse also happening in specific time of the year suggesting people are paying money in the during a specific time of the year either in start or end of the year.

One more interesting finding which we got from Figure 1 was that there was no “Send for Credit Collection” Activity in the year 2008. That year a big financial crisis happened we think that was the primary reason the police department decided not to conduct this activity as people had no money.

Figure 3 a) and b) shows a dotted chart of the activities over time duration. There are many wave-like patterns in the diagram. On zooming on the waves, we found that in most of the

cases had “Send for Credit Collection” as the last activity as figure 3 b) shown by the horizontal arrows. Similarly, we found that cases involving “Payment” activity as the last activity also tend to take more time than usual.

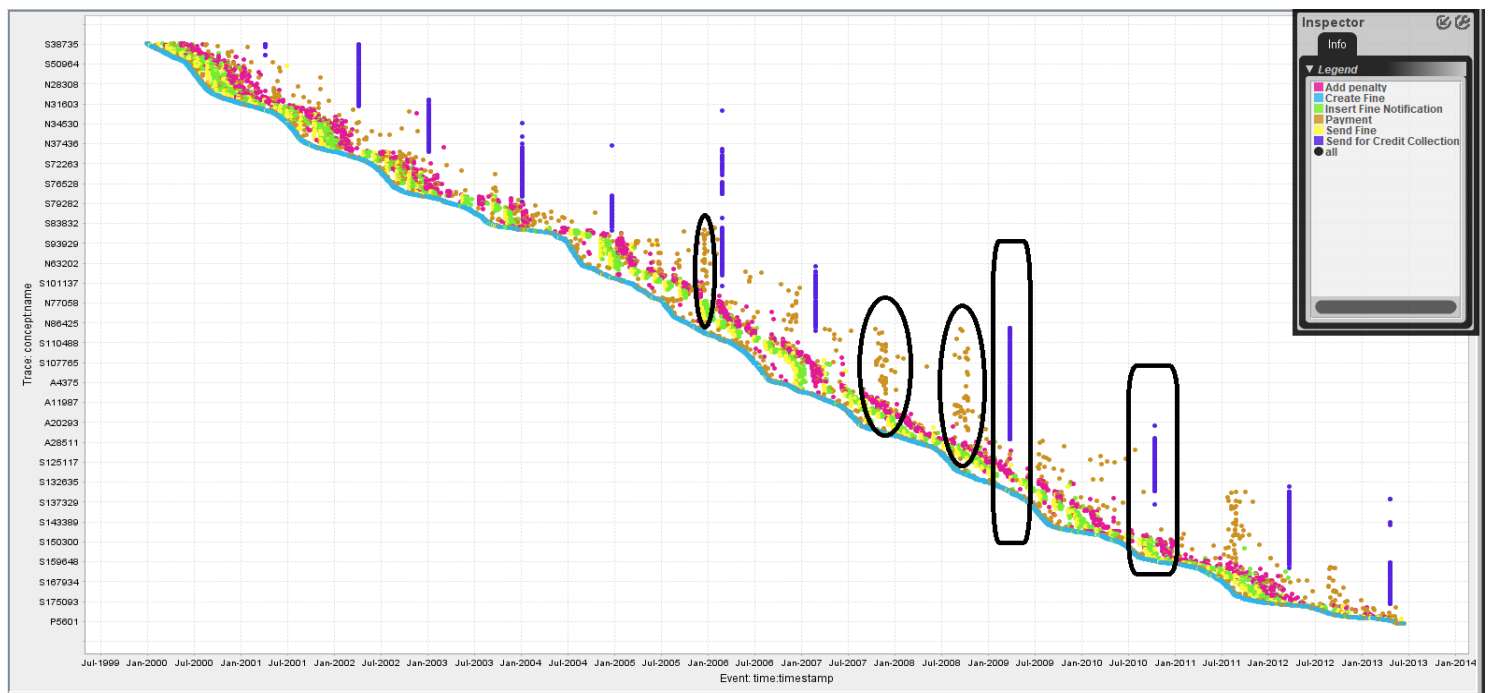


Figure 2 Activities over Time stamp

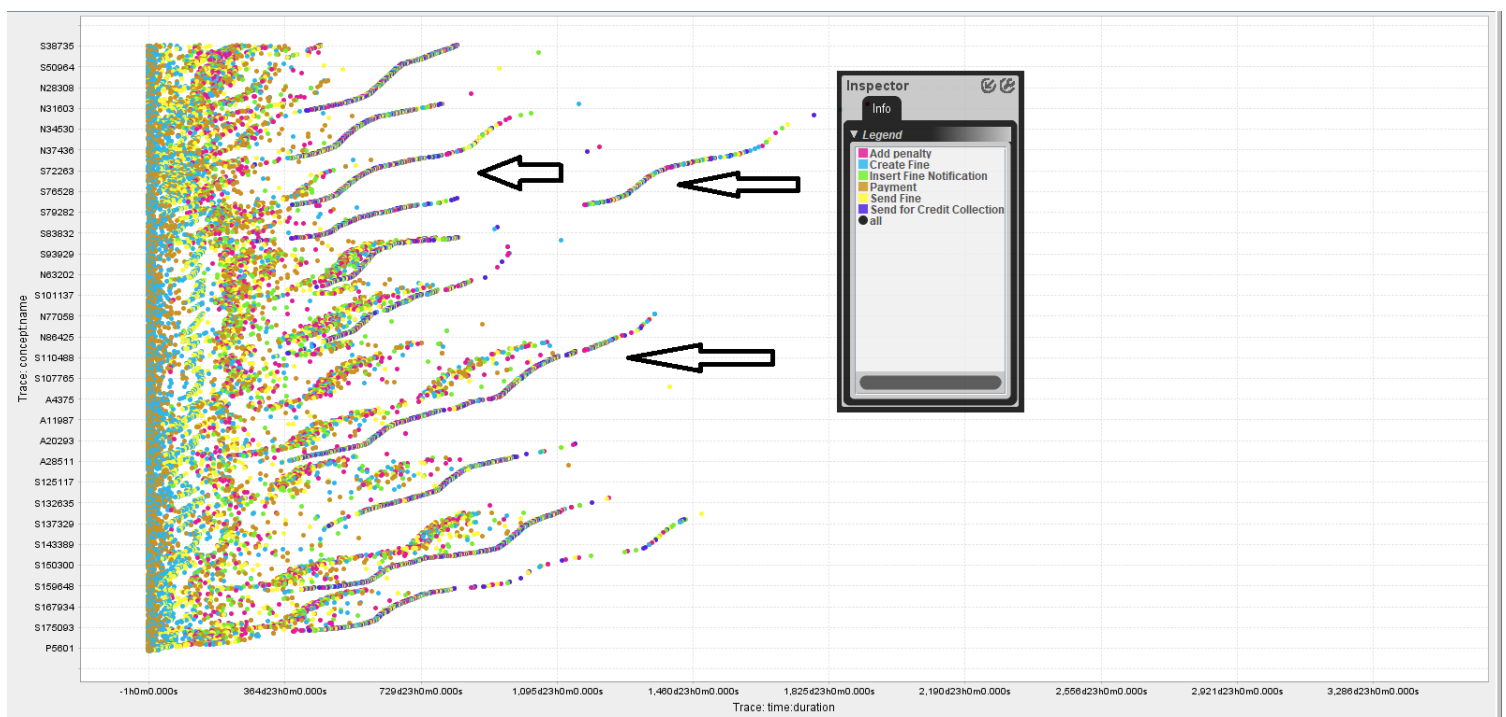


Figure 3 a) Activities over case duration

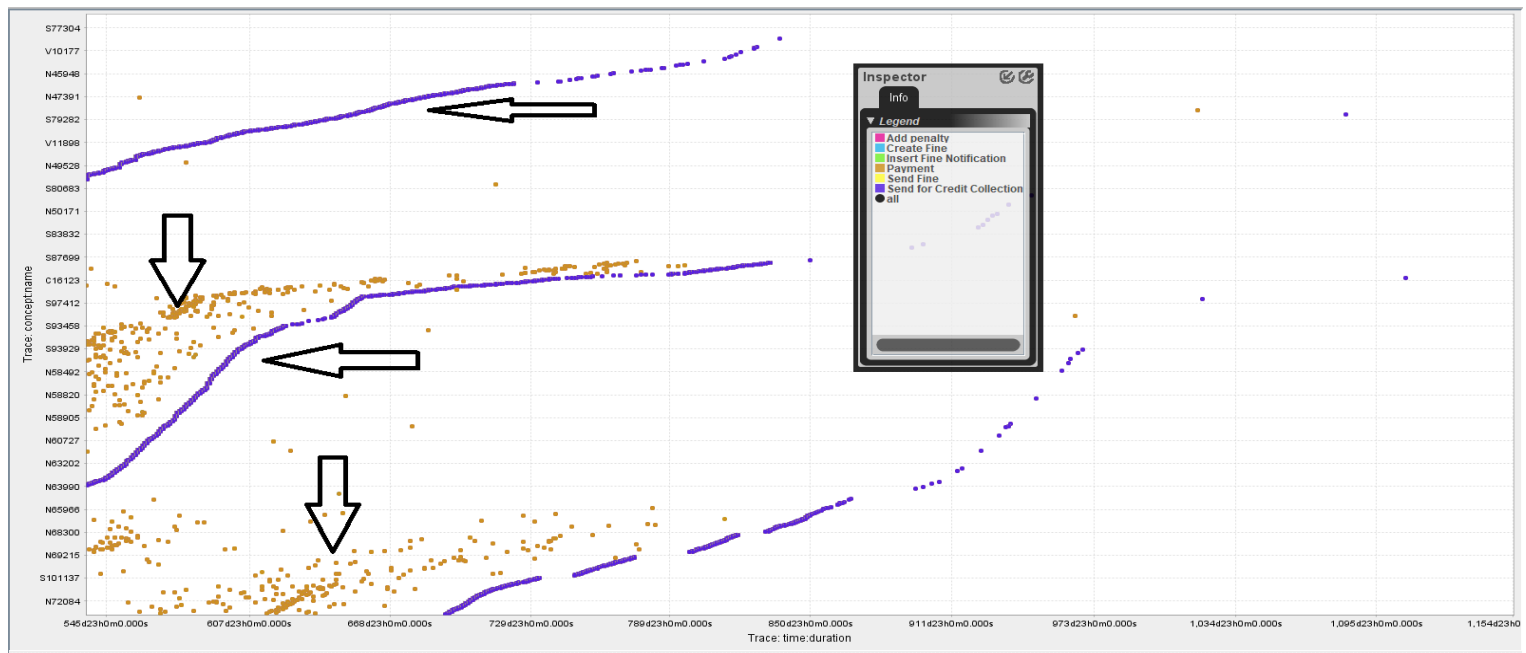


Figure 3 b) Activities over case duration

We can see the duration required for first “Send for Credit Collection” event is around 1 year, which also firms our claim that we made for the first dotted charts pattern (Figure 1), that the police department conducts this activity once a year.



e/ After considering the statistics in part b and c we got below insights in the process:

- Number of traces comparison suggest that 97% of the traces falls under Without_Appeal process.
- There are very small number of variants in the Without_Appeal_Log in comparison to the With_Appeal_Log, which suggest that the process having appeal related events is much more convoluted. Specially when With_Appeal_Log has only 3% of the total cases.
- The average trace length in case of the Without_Appeal_Log is also less than With_Appeal_Log.
- Time span suggest that both type of processes is frequent.
- With_Appeal_Log involves more activity than the Without_Appeal_Log, which is obvious, as we made them separate based on the activity names.

Part c, data suggest below highlights for the processes:

- “Create Fine” is the most frequent and first activity in all three logs.
- Second most frequent activity is “Send Fine” in all three logs.
- “Send for Credit Collection” is the least frequent in With_Appeal and Without_Appeal process but if we consider the Original_Log “Appeal to Judge” is least frequent.
- In case of With_Appeal_Log “Send Fine” activity always happened like “Create Fine”.
- In Original_Log 44% of the time “Payment” was the last activity, which is followed by “Send for Credit Collection” with 39%. This means in more than 80% of the cases the process ended with offender either paying the money or police department’s credit collection activity.

- In With_Appeal_Log 68% of the time “Send Appeal to Prefecture” was the last, followed by “Payment” at 16%.
- Without_Appeal_Log’s ending activities are like the Original_Log as the With_Appeal_Log only contains 3% of the cases. In Without_Appeal_Log 45% of the time “Payment” was the last followed by “Send for Credit Collection” with 40% to be exact.

Desirable Results: Since most of the cases (85%) out of all the cases ends with the two activities “Payment” and “Send for Credit Collection” and certainly if the process ends with the payment, that is desirable, if people pay on time and the police doesn’t had to the credit collection, then it saves huge amount of money and time.

Potential Problems:

- Why was the process involving appeals takes more time than without appeals and why does this process contain so many variants?
- Why only 1% of the cases results were received from the Prefecture while they sent appeal in 68% cases in the With_Appeal_Log.
- In the Original_Log it seems that almost in half of the cases Police had to perform “Send for Credit Collection” activity, means offenders were not paying their fine on their own, why was that the case.
- In Without_Appeal_Log process 14% of the time “Send Fine” was the last activity means in those cases offender didn’t paid and also the Police didn’t collected the money, why was that the case, was it because of less fine amount or it depended upon some other factors.

Q2. Discovery and Conformance

In this part we discovered model from the logs and checked the conformance of the model i.e. how much logs are matching with the discovered model.

a)  Figure 4 is showing the model generated from the Without_Appeal_Log.

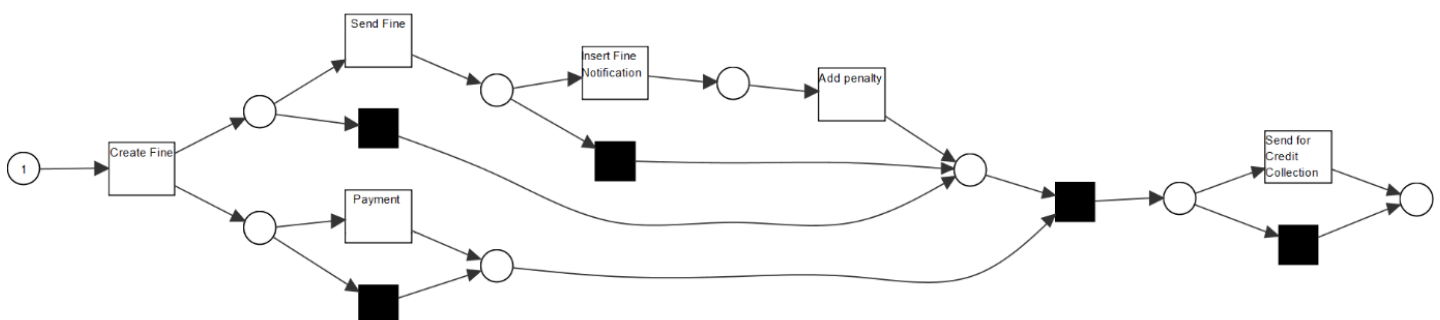


Figure 4 Model generated from the Without_Appeal_Log

Figure 5. (a) and 5. (b) is showing the model generated from the With_Appeal_Log.

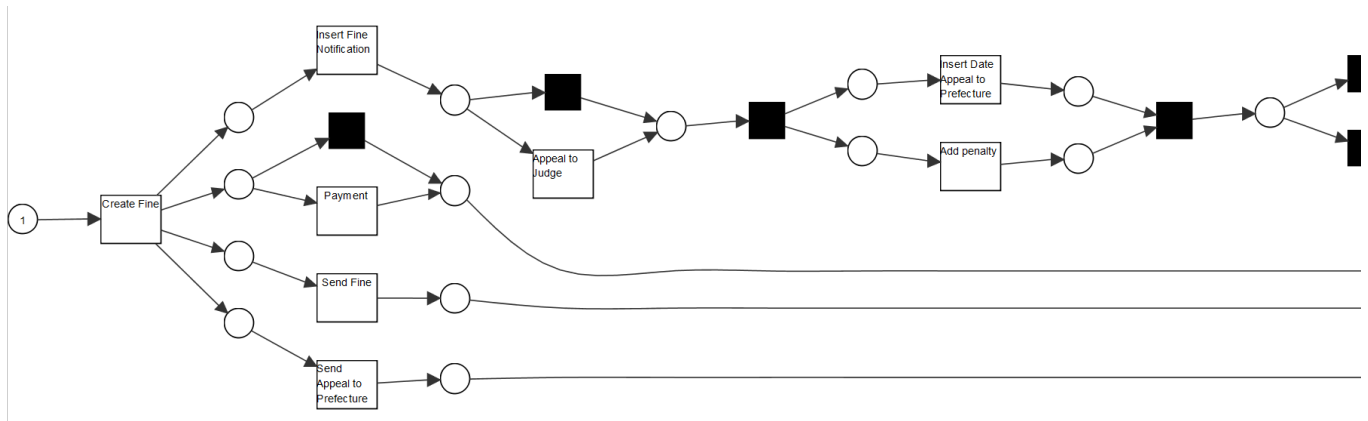


Figure 5. (a) Left part of the model generated from With_Appeal_Log

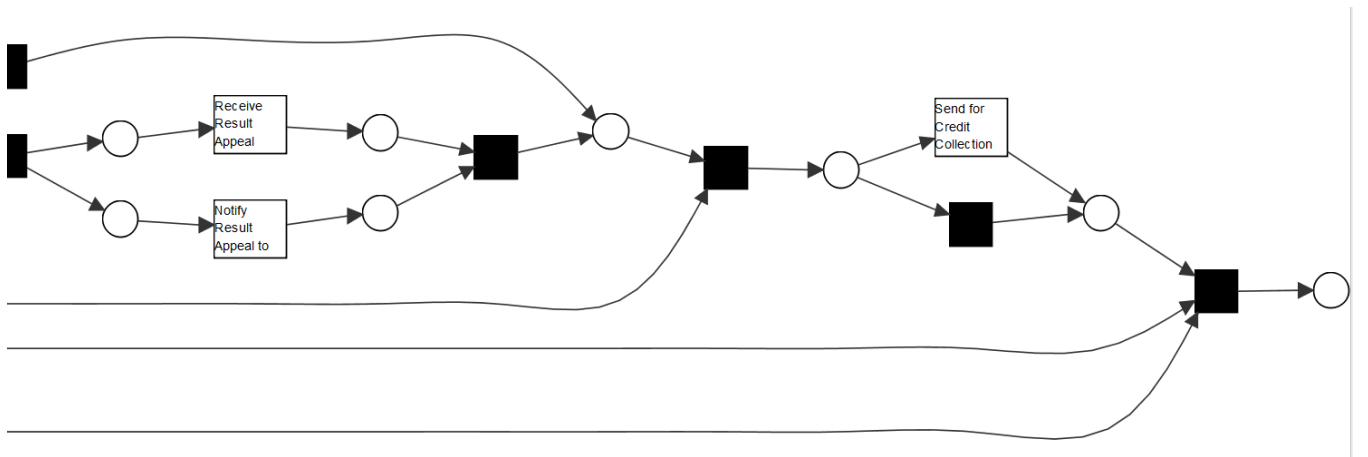



Figure 5. (b) Right part of the model generated from With_Appeal_Log


To generate the models, we used Inductive Miner. Both the models were generated after considering 20 % as the noise threshold, which was found to result in better model while keeping most of the behaviour. We used Inductive Miner as it gives a sound workflow net while maintaining the fitness and it can detect hidden tasks.

Model generated from Without_Appeal_Log is very simple, for generalization we can say it handles generalization also very gracefully as almost all the activity have a silent transition along with them, suggesting that this model can generate much more behaviour which was not present in the logs.

Model generated from With_Appeal_Log is also simple because we removed the noise, the model generated with including all the logs was very complex. But this model does not seem to generalize too well as there are certain activities in the start they have to be there in the logs. However, in the later parts, there is flexibility in the behaviour.

b)  Quality Scores for precision and fitness:

- **Model Without Appeals:** Precision = 0.72742, Fitness = 0.9925
- **Model With Appeals:** Precision = 0.68692, Fitness = 0.9672

c) Behaviour of both the models: 

- I. **Model Without Appeals:** Start Activity as we see in this model is “Create Fine”, after that there are two paths one goes to “Send Fine” activity and other go to “Payment” activity. So, if we see the path following “Send Fine” there are two more activities “Insert Fine Notification” and “Add Penalty”, both activities have silent transitions alongside them, suggesting that both can be skipped. Then there is a silent transition which waits for the “Payment” activity to be finished before going forward. Now the “Payment” activity path which we started after “Create Fine” has also a skip transition, means “Payment” activity can also be skipped. Once either the “Payment” activity is done or skipped both the paths are joined to finish the process except one activity “Send for Credit Collection”. The model suggest that this activity is last activity in some of the cases not all as it also have a skip transition alongside it.
- II. **Model With Appeals:** Start Activity in this model is also “Create Fine”, after this activity there are 4 paths, let see from bottom to top as the two bottom paths has one activity each “Send Fine” and “Send Appeal to Prefecture” after these activities there is only one silent transition before the process gets completed. However, this silent transition waits for other paths to be finished. Now about the top two paths, one of them have a “Payment” activity with silent transition in parallel suggesting that this activity is skippable. And with a series of this skippable transition the process could also be finished. The most complicated path which this model has is the top one which involves “Insert Fine Notification” activity. After this activity there is an option to perform “Appeal to Judge” activity. Then there are two activities which should happen concurrently to progress in this process which are “Insert Date Appeal to Prefecture” and “Add Penalty”. After these two activities there is a choice to either perform two more activities “Receive Result Appeal from Prefecture”, “Notify Result Appeal to Offender” concurrently or skip both. Now after this there is last choice to do “Send for Credit Collection” or skip it. Once this is finished the process ends as soon as the silent transition fires.



u) Deviations of the logs from the process, for this we used Inductive Visual Miner, check the corresponding figures:

- I. **Model Without Appeals (Figure 6):** In this model we found there were no such cases where there some activity is skipped in the model, however there were in total 7696 times “Payment” activity happened in various stages of the process as you can see in the Figure 6 but it was not supported by the model, these were log moves only.

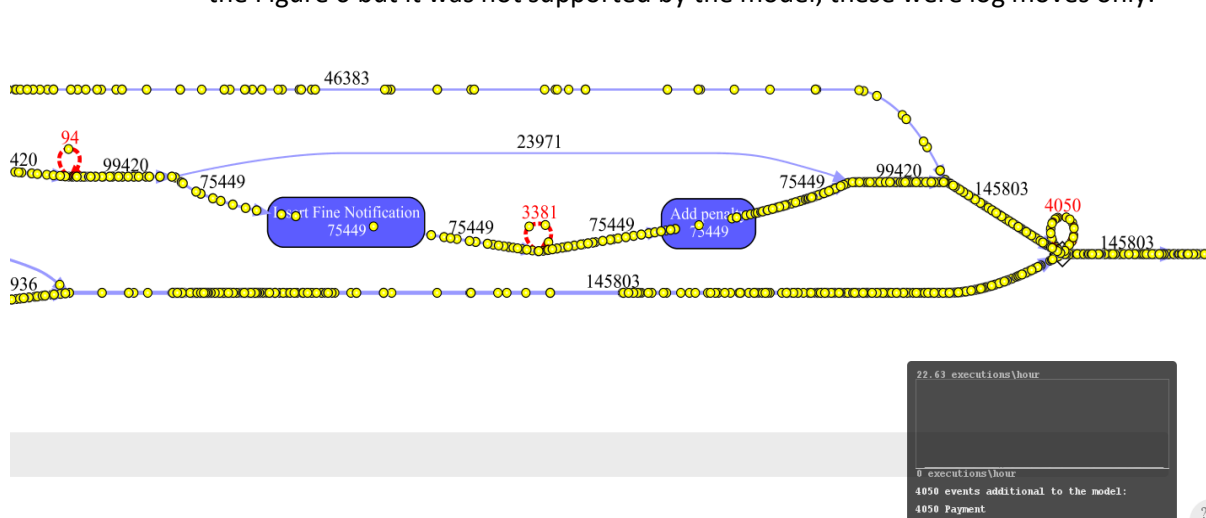


Figure 6 Deviations in model Without_Appeal

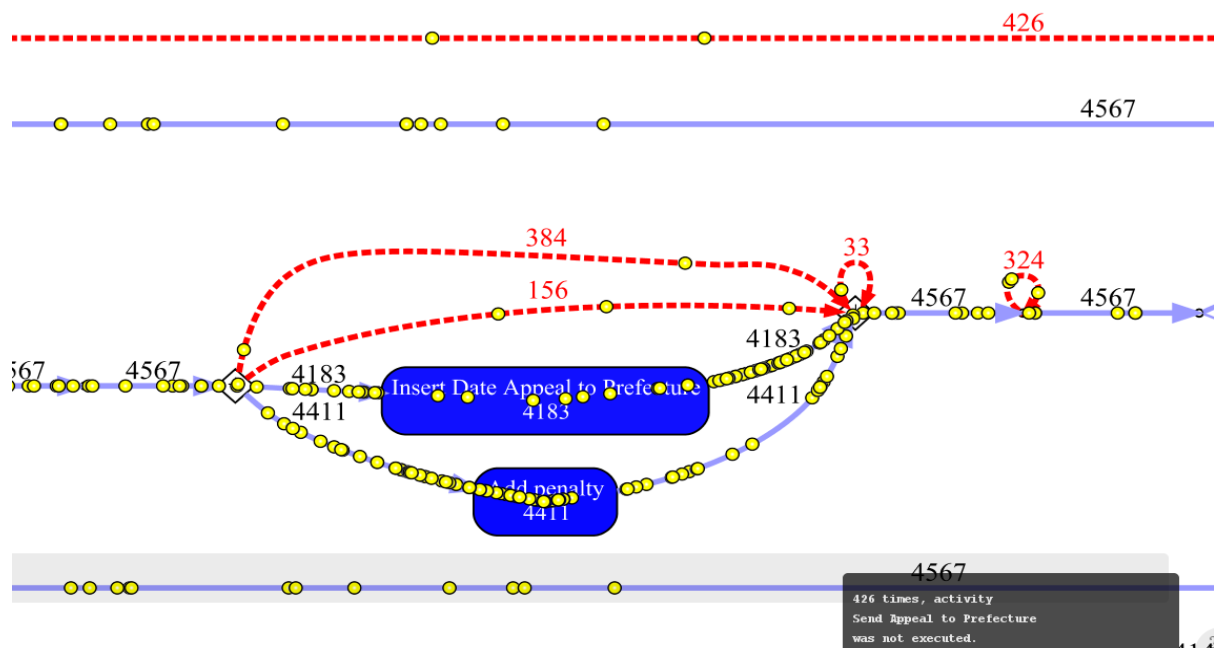



Figure 7 Deviations in the model With_Appeal

II. Model With Appeals (Figure 7):

- “Insert fine notification” was not executed 193 times,
- “send appeal to prefecture” was not executed 426 times,
- “add penalty” was not executed 156 times model move,
- “Insert Date Appeal to Prefecture” was not executed 384 times,
- And there were many times activities only executed in the log but was not supported by the model, the two major blocks of such activities were 324, 180, which involved these activities respectively [129 - Payment, 93 - Appeal to Judge, 52 - Notify Result Appeal to Offender, 50 - Receive Result Appeal from Prefecture], [136 - Appeal to Judge, 44 - Payment]. These were only log moves not on the model.

e)  The two model discovered tell two different stories, the model without the appeals is simple, yet involves multiple irregularities regarding “Payment” activity, as we saw there were multiple cases where the “Payment” activity is happening after activities where it shouldn’t happen according to the model. As we can see in Figure 6 most of these payment deviations were after “Insert Fine Notification” and “Add Penalty” activity, suggesting that there is some behaviour which was not completely captured by the model generated. And it makes some sense also because people may try to pay when the police inserts fine and adds penalty. They fear that more penalty can be given if they do not pay.


The model with appeals is little bit complicated, filtering the noise made it understandable but with that there are lots of deviation were introduced. The highest violation was the “Send Appeal to Prefecture” activity, it was skipped 426 times. There was one thing we thought was strange that why “Insert Date Appeal to Prefecture” and “Add Penalty” are happening concurrently. But when we were examining the deviation we found that in 384 cases “Insert Date Appeal to Prefecture” was not executed and in 156 cases “Add Penalty” was not executed, suggesting us that the concurrency in the model is not based on any mechanism, it’s a mere coincidence. Like the first model here also “Payment” activity happened in various

stages of the model which was not supported by the model, these were basically moves on the log.

Our model's behaviour which was described earlier pretty much matches up with the actual real process, as we saw that there were not many deviations compared to the number of traces in each model. And we also calculated the fitness of both models and it was high, however the precision is less and that is to be expected as we described the models would hold up to the generalization.


Q3. Attribute Analysis and Compliance

In this section we inspected the Process generated from Without_Appeal_Log as 97% of the traces follow this path.

- a)  Below table sums up the information which we collected about the payment status of each case using PM4PY library on the Without_Appeal_Log. As the Police department told us if the traces contained "Send for Credit Collection" activity means that the offender paid in full. The logic to calculate all this data was simple, we checked all the cases, we summed up all the expenses with events like "Create Fine", "Send Fine" and "Add Penalty" stored in one variable and in one variable we stored the amount paid by the offender, whenever we encountered the "Payment" event. Now if we do not saw the "Send for Credit Collection" activity in the whole traces we used the expense amount and paid amount to decide the Payment Status. The code for this can be found in a separate file in the same zip folder.

Sr	Payment Status	Fraction of Traces	Number of Traces
i)	Proper Payment	0.735	107169
ii)	Too Much Payment	0.006	882
iii)	Less Payment	0.119	17366
iv)	No Payment at all	0.139	20386

v) Due to this people not paying any payment or less payment the police department lost \$2210639.53. This is simple subtraction between the expense amount and amount paid in the cases where there was no payment or less payment. Code for this is also in the same file as the payment status calculation.

- b)  To discover how the amount paid by offender increases over time, we looked first on the data where amount increment happens. We found that there were two activities where the amount was increased "Send Fine" and "Add Penalty" after once the fine was created. And in the earlier analysis of the logs we found that the only "Payment" activity was repeated multiple time not these. So, we extracted first some data about these activities like *Expense Amount, Penalty Amount, Time to Send Fine (in days), Time Create Fine to Penalty (days) etc.* Below table gives a sneak peek in the data.

	Fine Amount	Expense Amount	Penalty Amount	Send Fine Time	Time to Send Fine	Penalty Time	Time Send Fine to Penalty	Time Create Fine to Penalty	Send Fine	Add Penalty
0	35.0	11.0	0.0	2006-12-05 00:00:00+01:00	134.0	None	0.0	0.0	True	False
1	35.0	11.0	71.5	2006-12-12 00:00:00+01:00	132.0	2007-03-16 00:00:00+01:00	94.0	226.0	True	True
2	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	129.0	2007-10-01 00:00:00+02:00	76.0	205.0	True	True
3	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	118.0	2007-09-22 00:00:00+02:00	67.0	185.0	True	True
4	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	118.0	2007-10-01 00:00:00+02:00	76.0	194.0	True	True
5	22.0	13.0	44.0	2007-07-17 00:00:00+02:00	118.0	2007-09-21 00:00:00+02:00	66.0	184.0	True	True
6	21.0	11.0	42.5	2006-12-12 00:00:00+01:00	132.0	2007-03-01 00:00:00+01:00	79.0	211.0	True	True
7	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	118.0	2007-09-21 00:00:00+02:00	66.0	184.0	True	True
8	36.0	18.6	74.0	2007-09-29 00:00:00+02:00	191.0	2007-12-11 00:00:00+01:00	73.0	265.0	True	True

Figure 8 Case data with extracted information.

On first look of the data we find out that *Penalty Amount* was always around the double of the *Fine Amount* and *Time Create Fine to Penalty* was around 200 days. But before drawing conclusions we looked on the descriptive statistics of these traces. So, in the below figure 9 we found that the average *Time Create Fine to Penalty* was now around 4 months. The initial hunch for the *Fine Amount* and *Penalty Amount* was also not adhering. And we also saw that the minimum amount of *Fine Amount*, *Expense Amount*, *Penalty Amount* were zero. So, we investigated further and thought that we should look on the cases where there is only “Send Fine” happened, and “Add Penalty” happened and in how many cases the amounts were zero.

	Fine Amount	Expense Amount	Penalty Amount	Time to Send Fine	Time Send Fine to Penalty	Time Create Fine to Penalty
count	99420.000000	99420.000000	99420.000000	99420.000000	99420.000000	99420.000000
mean	46.977901	11.860449	73.587968	87.244468	58.813237	123.842808
std	55.418873	3.889061	114.481481	42.296502	35.672647	79.457043
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	32.000000	10.000000	37.560000	53.000000	60.000000	60.000000
50%	35.000000	11.880000	68.770000	87.000000	71.000000	142.000000
75%	38.000000	13.500000	74.000000	121.000000	80.000000	186.000000
max	4351.000000	76.000000	8000.000000	732.000000	492.000000	792.000000

Figure 9 Descriptive values of the traces

Upon checking the data, we found that “Send Fine” always happened in these cases. The descriptive statistics generated from that was same as the Figure 9. However, when we checked for the data where “Send Fine” happened and the *Expense Amount* was zero. We found that the data was following our rule of *Penalty Amount* and *Fine amount* as shown in Figure 10, *Penalty Amount* was around double of the *Fine Amount* for these cases. Surprisingly for these cases the *Time to Send Fine* was less than a day which was very less in comparison to the mean of all traces which was around 3 months. Similarly, the *Time Create Fine to Penalty* was also around 2 months, almost half of all trace means.

	Fine Amount	Expense Amount	Penalty Amount	Time to Send Fine	Time Send Fine to Penalty	Time Create Fine to Penalty
count	2636.000000	2636.0	2636.000000	2636.000000	2636.000000	2636.000000
mean	140.671324	0.0	287.228323	0.821320	60.335736	61.157056
std	216.034688	0.0	467.425367	15.349797	6.795742	16.788776
min	0.000000	0.0	0.000000	0.000000	59.000000	59.000000
25%	32.800000	0.0	65.600000	0.000000	60.000000	60.000000
50%	68.000000	0.0	137.500000	0.000000	60.000000	60.000000
75%	143.000000	0.0	286.500000	0.000000	60.000000	60.000000
max	4351.000000	0.0	8000.000000	732.000000	226.000000	792.000000

Figure 10 Descriptive Statistics for the Cases where Send Activity happened but the expense amount was zero.

	Fine Amount	Expense Amount	Penalty Amount	Time to Send Fine	Time Send Fine to Penalty	Time Create Fine to Penalty
count	75449.000000	75449.000000	75449.000000	75449.000000	75449.000000	75449.000000
mean	47.919967	11.793445	96.967697	85.653567	77.498867	163.189068
std	58.588739	3.912294	122.486259	42.023576	15.123107	43.569819
min	0.000000	0.000000	0.000000	0.000000	59.000000	59.000000
25%	32.000000	10.000000	65.500000	52.000000	69.000000	132.000000
50%	35.000000	13.000000	71.500000	85.000000	75.000000	164.000000
75%	38.000000	13.500000	77.500000	120.000000	82.000000	196.000000
max	4351.000000	60.000000	8000.000000	732.000000	492.000000	792.000000

Figure 11 Descriptive Statistics for the Cases where Add Penalty Activity happened

There were also cases where the *Fine Amount* was zero, 22 to be exact. Now if looked on the traces where “Add Penalty” activity happened, which was the major cause of increase in expenses for the offender we found that out of 99420 traces, 75449 traces had this activity, almost $\frac{3}{4}$ of the total. Figure 11 shows a table about these trace’s data. So, in this table we can clearly see that our rule for doubling the amount is still valid. To establish our rule, we draw a scatter plot between the *Fine Amount* and the *Penalty Amount* shown in Figure 12. The plot also confirmed our rule that the *Penalty Amount* was always double to the *Fine Amount*.

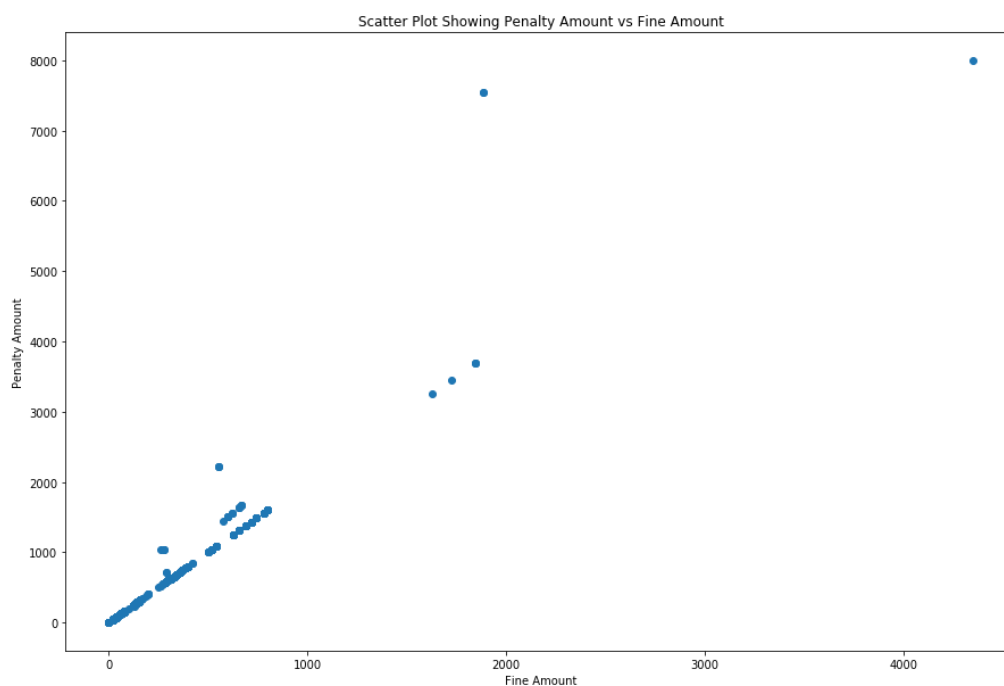


Figure 12 Penalty Amount vs Fine Amount

The other thing to notice here is that the *Time Create Fine to Penalty* is around 5.5 months, however the standard deviation is also too high. To get the idea about how the durations (*Time Create Fine to Penalty*, *Time to Send Fine*) were fluctuating over the time we draw some distribution chart as shown in Figure 13. We found some patterns in the below graph one was that there were two horizontal lines background corresponding to the cases where *Expense Amount* was zero. As we discussed earlier in those cases the Send Fine duration was around 1 day and Add Penalty two months.

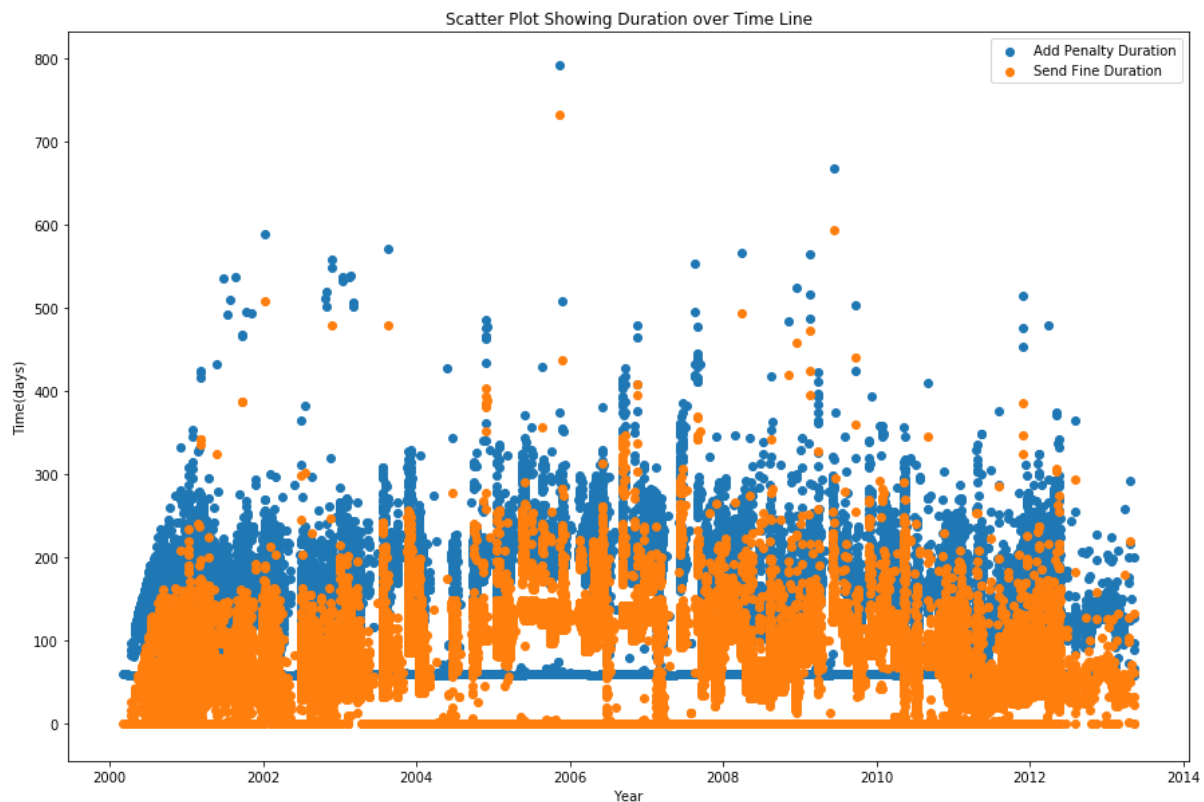


Figure 13 Send Fine and Add Penalty durations over timeline.

The other pattern was the vertical lines, which suggested when the *Time to Send Fine* increases the *Time Create Fine to Penalty* also increases. As we saw that the Send Fine duration and Add Penalty duration were more centered around 100 and 200 days, we thought of making frequency distribution of the durations. Figure 14 and 15 show that distribution and confirms that as the *Time to Send Fine* increased the *Time Create Fine to Penalty* also increased. And both the diagrams had similar pattern.

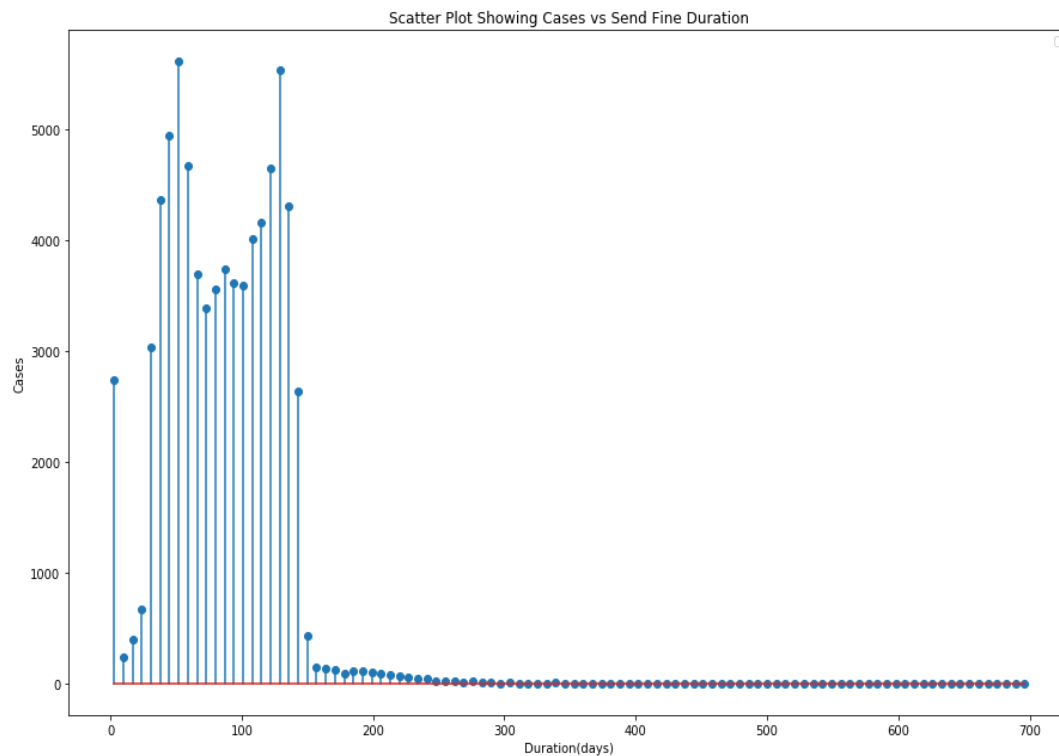


Figure 14 Cases vs Send Fine Duration

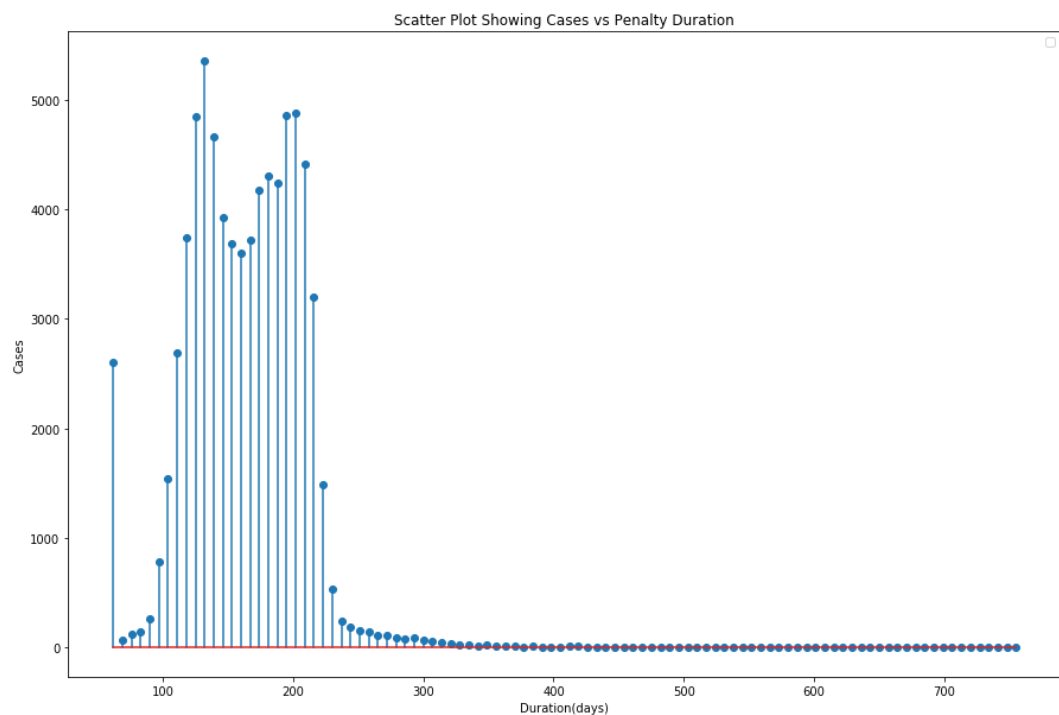



Figure 15 Cases vs Penalty Duration

- c)  we saw in part a) of this section there were some irregularities in Payment, sometime people were not paying at all, or they were paying less, or paying more. In some cases, police had to perform "Send for Credit Collection" activity to get the money. To know what is happening we mined a decision tree to understand hidden factors in these cases. We determined the *Payment Status*[No Payment, Less Payment, Full Payment]{A: Paid by the

offender by their own, B: Police had to do the credit collection activity}, More Payment] for each cases and used it as the target variable, while *Fine Amount*, *Penalty Amount*, *Time to Send Fine*, *Time Send Fine to Penalty*, *Time Create Fine to Penalty*, *Send Fine (Happened {True, False})*, *Add Penalty (Happened {True, False})* were features which we considered to base our decision Tree on. We discovered the below tree, we used *min_impurity_decrease* as 0.1.

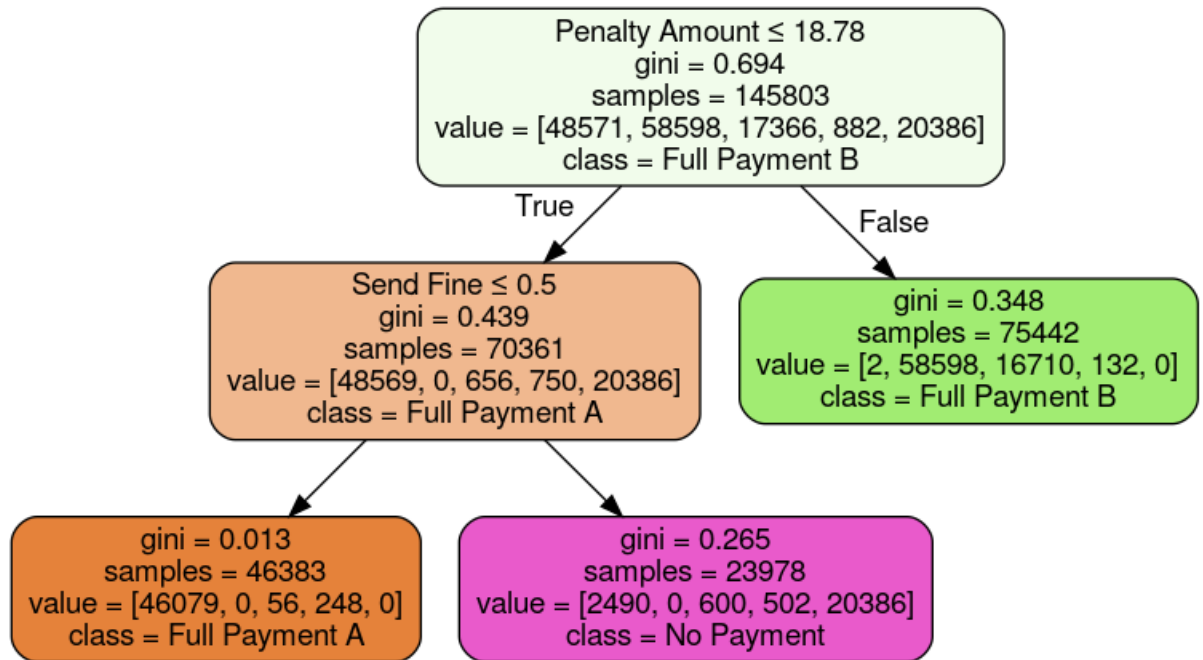


Figure 16 Decision Tree generated for all the traces where expense was increased.

From the tree we can see those traces fell under three major classes *Full Payment A* 46079 cases where people paid full payment on their own, *No Payment* 20386 cases where people didn't paid at all and *Full Payment B* 75442 cases where Police has to do the credit collection. So, the first factor on which the tree was split is *Penalty Amount* if it is higher than 18.78 then Police must do credit collection. In case the amount is less than 18.78 but the *Send Fine* activity happens means the people were not paying that's why Police must *Send Fine*. While *Send Fine* did not happen in case people paid on their own (Full Payment A). So, this decision tree gives pretty much good idea to predict the payment status of the cases.

	Fine Amount	Expense Amount	Penalty Amount	Send Fine Time	Time to Send Fine	Penalty Time	Time Send Fine to Penalty	Time Create Fine to Penalty	Send Fine	Add Penalty	Payment Status	Payment Balance
0	22.0	13.0	44.0	2007-07-17 00:00:00+02:00	118.0	2007-09-21 00:00:00+02:00	66.0	184.0	True	True	Less Payment	-22.0
1	36.0	13.0	74.0	2007-08-02 00:00:00+02:00	130.0	2007-10-08 00:00:00+02:00	67.0	197.0	True	True	Less Payment	-36.0
2	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	130.0	2007-09-21 00:00:00+02:00	66.0	196.0	True	True	Less Payment	-36.0
3	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	129.0	2007-10-01 00:00:00+02:00	76.0	205.0	True	True	Less Payment	-36.0
4	36.0	13.0	74.0	2007-06-27 00:00:00+02:00	113.0	2007-09-04 00:00:00+02:00	69.0	182.0	True	True	Less Payment	-36.0
5	22.0	13.0	44.0	2007-08-02 00:00:00+02:00	95.0	2007-10-08 00:00:00+02:00	67.0	162.0	True	True	Less Payment	-22.0
6	22.0	13.0	44.0	2007-09-29 00:00:00+02:00	90.0	2007-12-11 00:00:00+01:00	73.0	163.0	True	True	Less Payment	-22.0
7	21.0	11.0	42.5	2007-04-05 00:00:00+02:00	113.0	2007-06-12 00:00:00+02:00	68.0	181.0	True	True	Less Payment	-21.0
8	36.0	13.0	74.0	2007-07-17 00:00:00+02:00	120.0	2007-10-04 00:00:00+02:00	79.0	199.0	True	True	Less Payment	-36.0
9	22.0	13.0	44.0	2007-07-17 00:00:00+02:00	97.0	2007-09-22 00:00:00+02:00	67.0	164.0	True	True	Less Payment	-22.0

Figure 17 Showing Data of traces where Payment activity happened multiple times.

However, there was one more irregularity regarding the payment of fine which was with the “Payment” activity itself, there were some traces where this activity happened multiple times. To get a better idea we extracted all the traces where the payment activity happened more than once. The data looked like the below table shown in the Figure 17. On the first glance it looked like all those traces where “Payment” activity happened multiple times resulted in *Less Payment* Status. To explore this, we plotted bar chart showing the count of the cases vs *Payment Status* shown in Figure 18. So, we found that out of 7456 cases, 7198 resulted in *Less Payment* status.

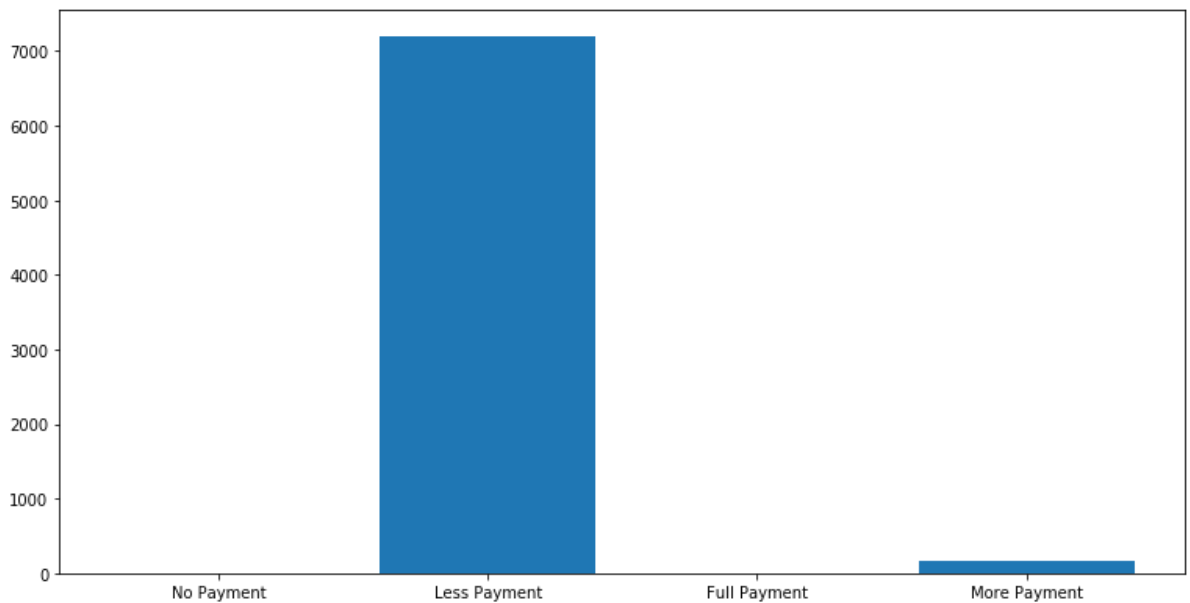


Figure 18 Number of Cases vs Payment Status

There was one more surprising finding we got about the *Payment Balance* (the amount, which is due on the offender, - means he must pay, + means he paid extra). This amount was same as the *Fine Amount*, suggesting us that the people were just paying the additional expenses which were incurred due to the “Send Fine” and “Add Penalty” activities. To investigate further we draw one graph between *Payment Balance* and *Fine Amount*, which is shown in the figure 19. This plot shows that in when the *Fine Amount* was less the people only paid the *Penalty Amount* and the *Expense Amount*, but as the amount of fine started to rise it is showing that *Payment Balance* was also double of that suggesting that people sopped paying even the other expenses and penalty amount.

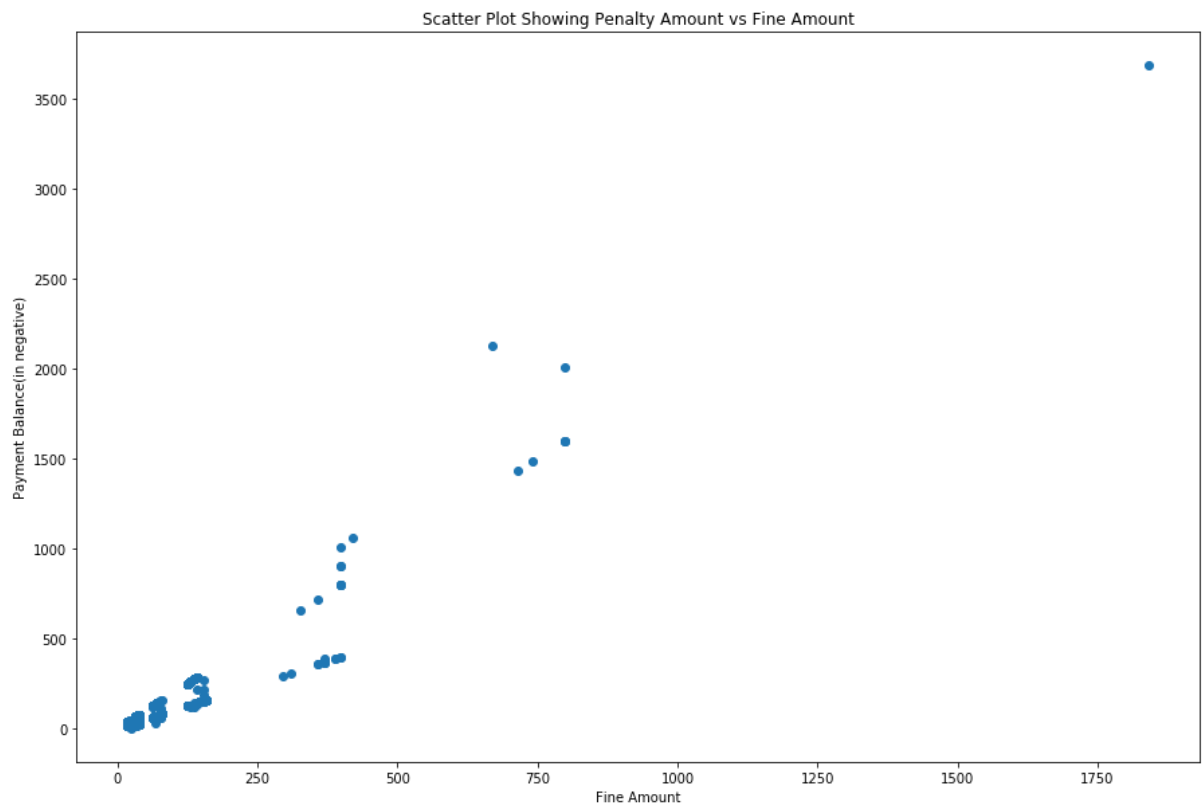


Figure 19 Payment Balance vs Fine Amount

There are certain deviations to our rule which we discussed in the part b) as we saw that there were many noises in the graph shown in the Figure 12, for example the *Fine Amount* was 2000 but the *Penalty Amount* was around 7500 which is clear violation of our doubling rule. We examined the data from such traces where the *Penalty Amount* was more than 100(threshold) of the doubled amounts of the *Fine Amount*. We found in those cases the *Penalty Amount* was 3-4 times the *Fine Amount* even though the *Time Create Fine to Penalty* was not much. Same was shown in the figure 20 showing descriptive data about such cases. The possible explanation for this behaviour seems to be the *Fine Amount*, check the mean value in the table it was more than 600 which is way more than the normal traces which was around 48.

	Fine Amount	Expense Amount	Penalty Amount	Time to Send Fine	Time Send Fine to Penalty	Time Create Fine to Penalty
count	24.000000	24.000000	24.000000	24.000000	24.000000	24.000000
mean	651.577083	5.758333	2013.930833	17.041667	67.208333	84.250000
std	406.039433	7.095064	1748.786692	43.036364	8.968443	45.949169
min	259.000000	0.000000	720.830000	0.000000	59.000000	59.000000
25%	555.000000	0.000000	1486.670000	0.000000	60.000000	60.000000
50%	600.000000	0.000000	1555.500000	0.500000	62.000000	62.000000
75%	653.000000	13.500000	1672.500000	8.000000	72.500000	85.750000
max	1886.000000	16.600000	7546.000000	180.000000	91.000000	249.000000


Figure 20 Showing Outlier trace data

After examining all this data, the advice which the police department can use is that they should not put high amount of penalty as we saw earlier that if the *Penalty Amount* is higher than 18.78 people tend not to pay themselves and Police has to send someone for credit collection which involves in more resource wastage. Instead they could use the same money to reduce the penalty amount of the offenders. However, we would tell the police to stick to fix duration of the *Time Create Fine to Penalty*, in some cases we found this value is way higher

than the mean duration. And, why Police department is not sending the fine immediately in all the cases. We would suggest them to send the fine immediately and send some reminders in between before adding a penalty as people sometimes cannot afford to pay the big amount in one payment. This was the case we saw where there were multiple payments were done by the offender but still the cases ended with less payment in total. It would be better if Police first send them a reminder about the Penalty, so that people can avoid such scenarios and try to pay on time.

Q4. Performance

In this section we looked on the performance aspects of the processes.

- a)  Time performance model of the whole process:
We found top 3 activities in this model which were taking too much time were “Send for Credit Collection”, “Payment” and “Send Appeal to Prefecture”, as shown in Figure 21 a) and b). These activities had average waiting time of 15.16, 12.35 and 3.46, 2.99 months respectively.

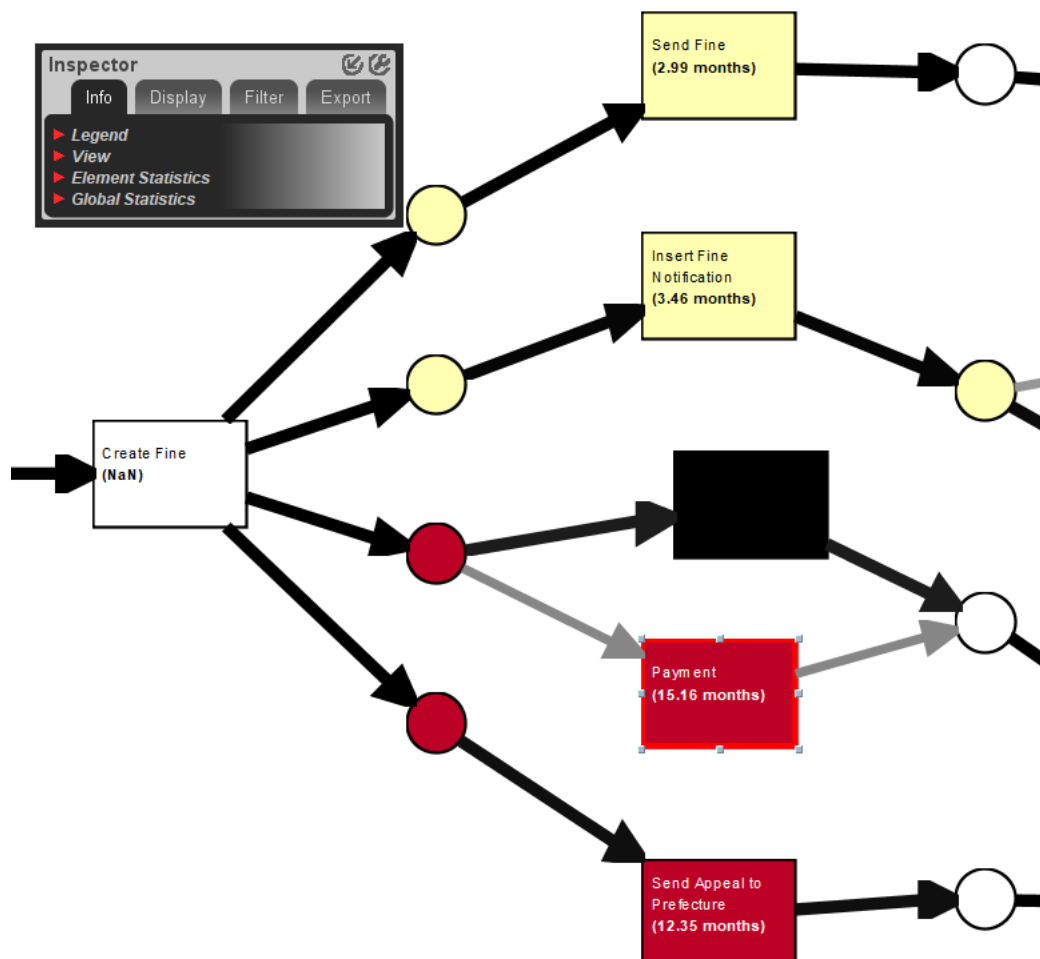


Figure 21 a) Showing bottleneck in process with appeals

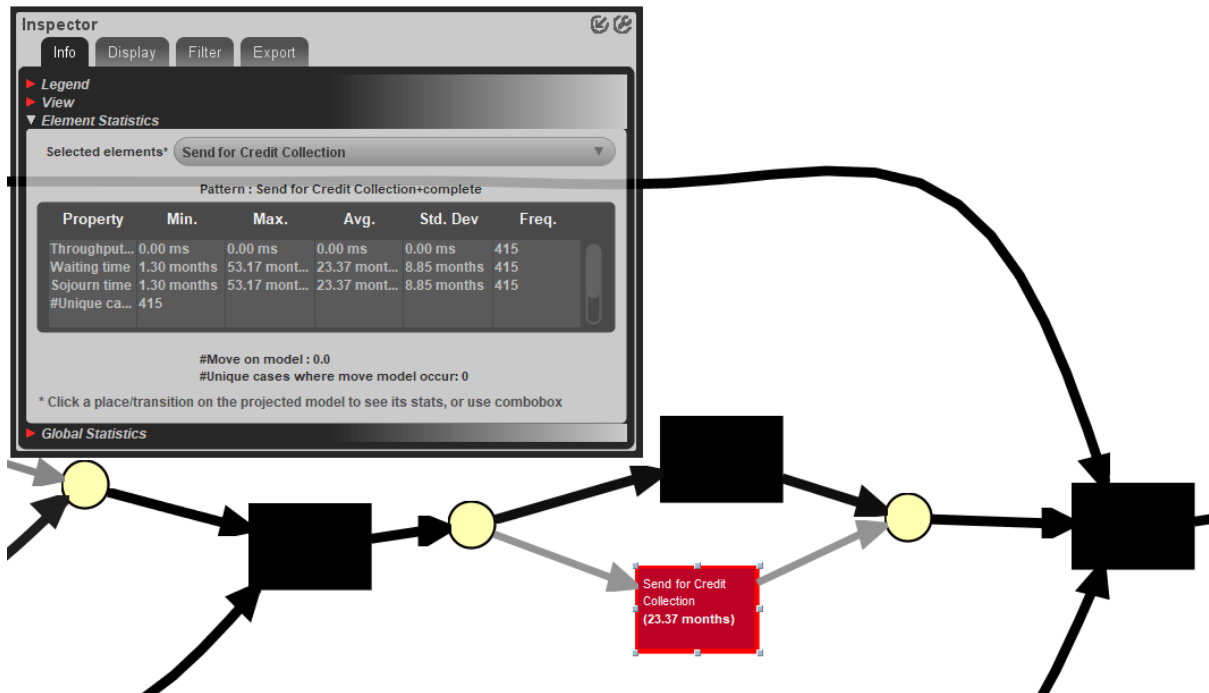


Figure 21 b) Showing bottleneck in process with appeals

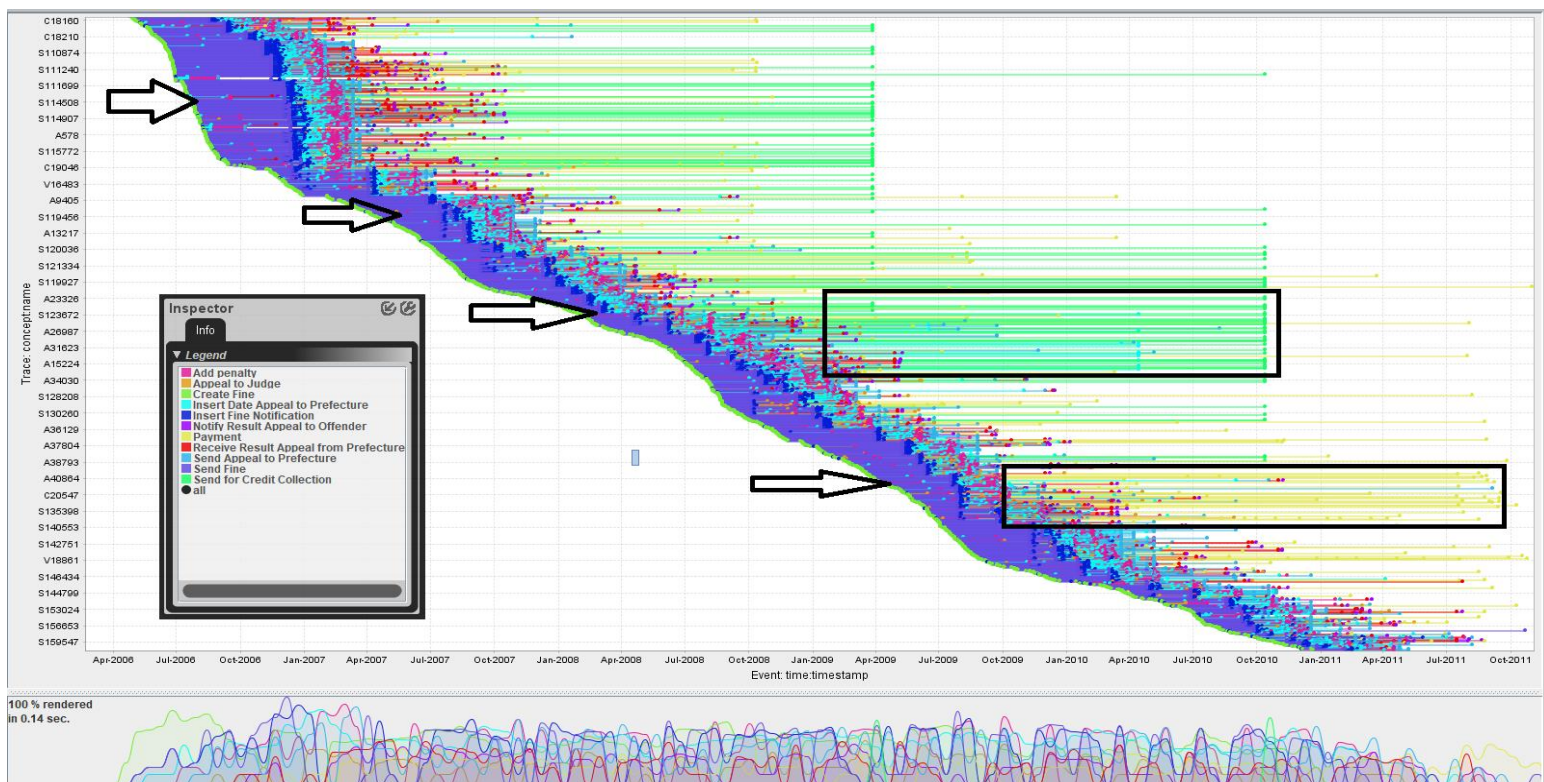


Figure 22 a) Showing Dotted chart of the With Appeal Process Traces over time.

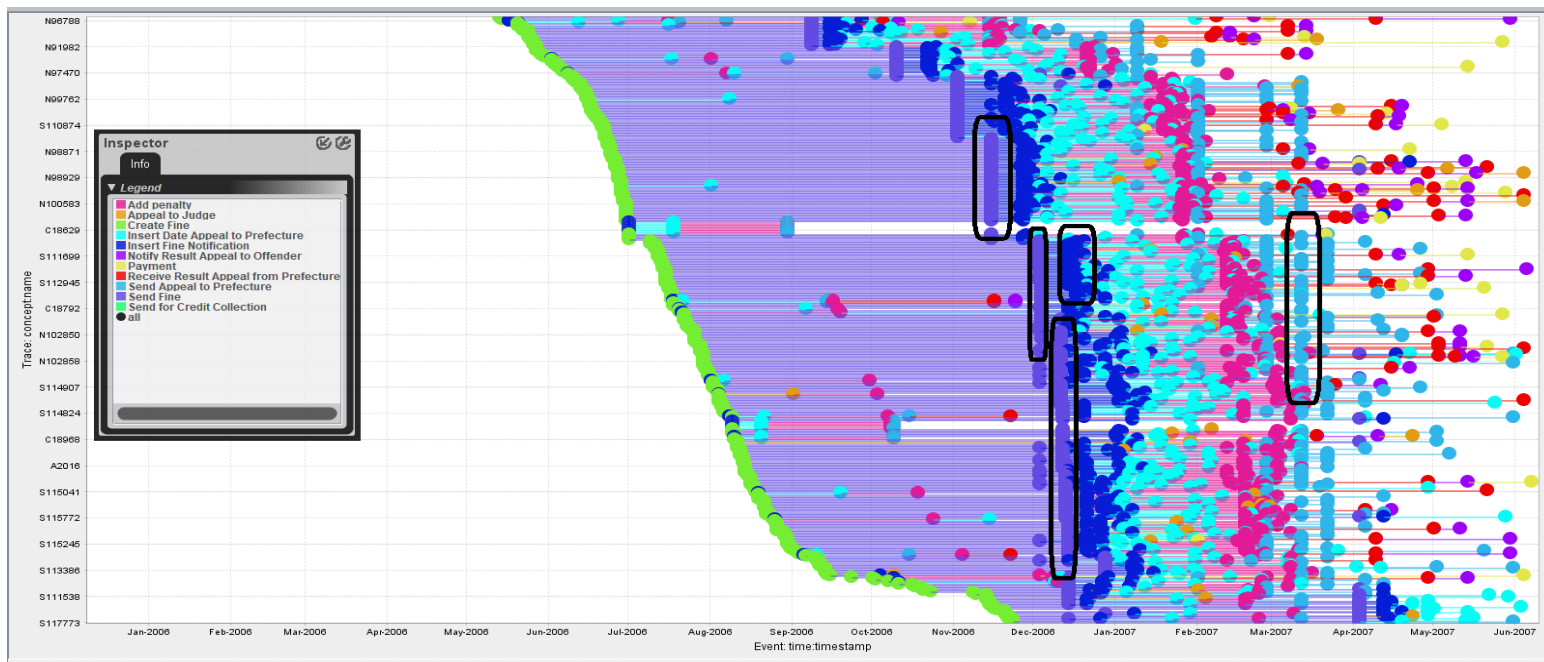


Figure 22 b) Zoomed in Dotted chart of the With Appeal Process Traces over time.

Dotted chart of the traces over time of the With_Appeal_Log are shown in the Figure 22 a) and b). As we can see there are many patterns in the first chart Figure 22 a), the two horizontal rectangle are for “Payment” (Yello lines) and “Send for Credit Collection” (Green lines) takes too much time which is coherent with the performance data of the model generated with the With_Appeal_Log. There was one more regular pattern which was indication with arrows in Figure 22 a) denotes that after “Create Fine” activity the police waits for some months before they send the fine. And in the Figure 22 b) you’ll see the vertical patterns which suggest the that Police is doing multiple activities in batches also the “Send Fine”, “Insert Fine Notification”, “Send Appeal to Prefecture” etc.

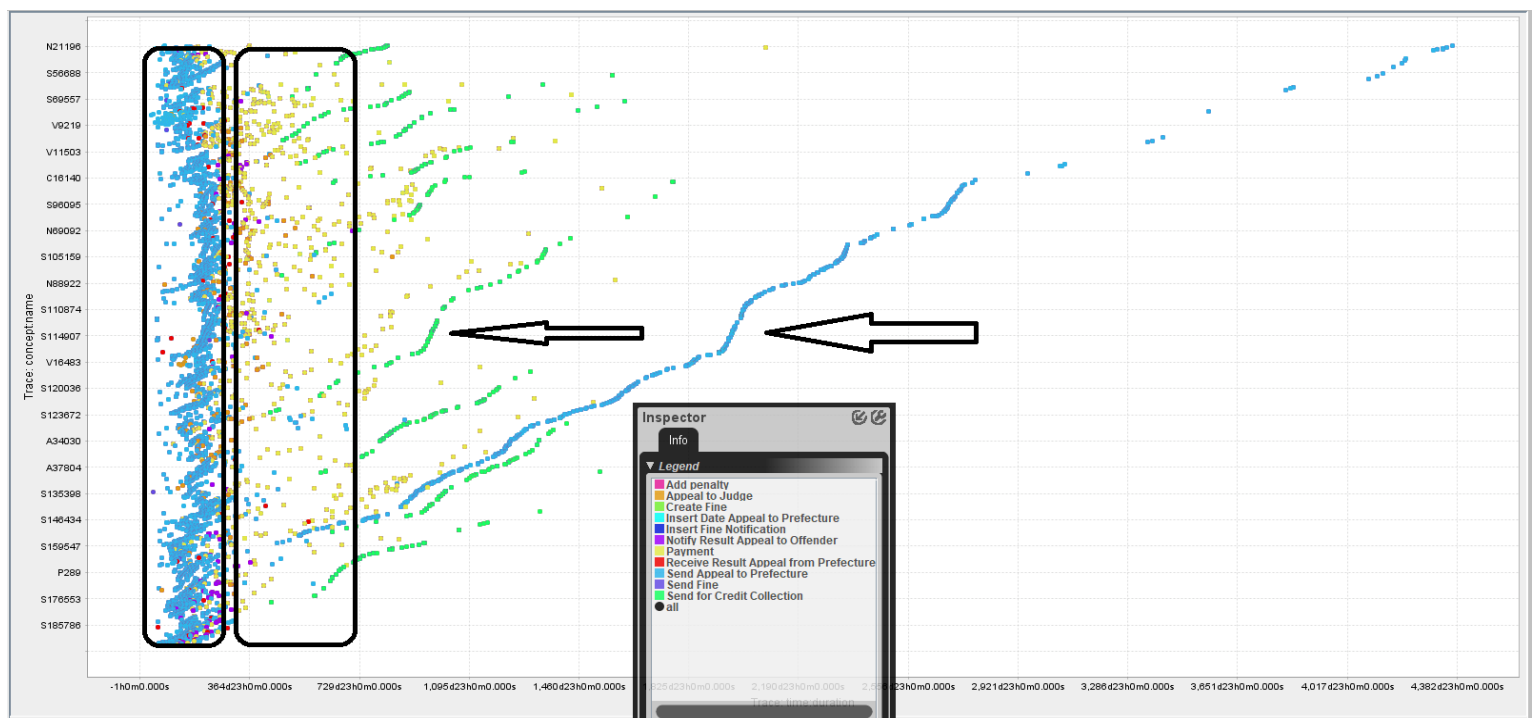


Figure 23 Showing Dotted chart of the With Appeal Process Traces over time duration of the cases.

If we see the dotted chart shown in the figure 23, it shows traces over case duration, so it also shows some patterns for example the green wave and blue waves denoted by the arrows tend to suggest that cases involving “Send for Credit Collect” (Green) and “Send Appeal to Prefecture” (Blue) as their last activity tend to take more time than others. Similarly the rectangle which contains yellow dots suggests that the cases involving “Payment” activity also have larger duration than the normal cases. The first rectangle which contains the blue dots suggest that the “Send Appeal to Prefecture” is very common activity to end these cases, which we also saw in question 1, of the report, this activity was the most frequent, 68% of the all cases ended with this activity.

b) Time performance of both the models, for time performance for model with appeal we will refer to the Figure 21 a) and b). Figure 24 a) and b) shows the Time performance for model without appeals.

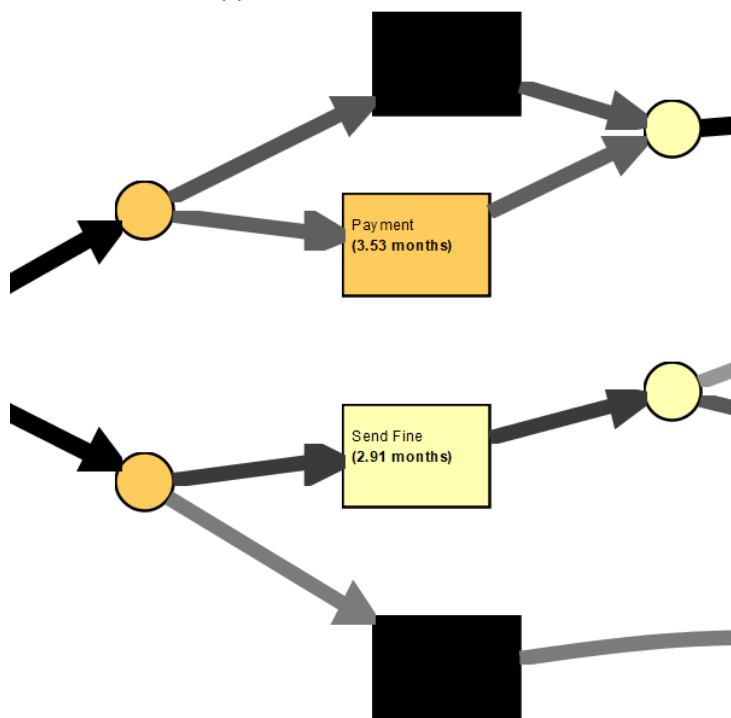


Figure 24 a) Time Performance of the Without Appeal Process

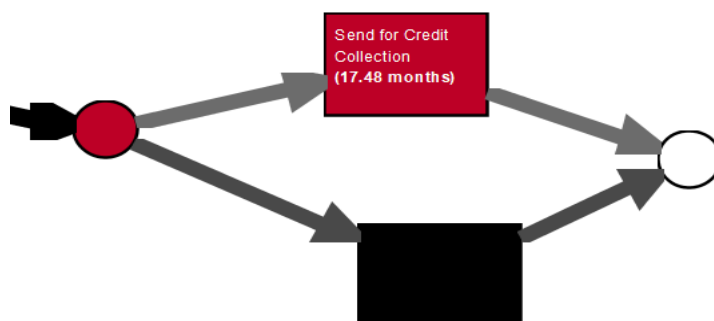



Figure 24 b) Time Performance of the Without Appeal Process

Model Without Appeal		Model With Appeal	
Activity	Average Waiting Time (months)	Activity	Average Waiting Time (months)
Send for Credit Collection	17.48	Send for Credit Collection	23.37
Payment	3.53	Payment	15.16
Send Fine	2.91	Send Appeal to Prefecture	12.35
Add Penalty	2	Notify Result Appeal to Offender	3.48
Insert Fine Notification	17.60 days	Insert Fine Notification	3.46
Create Fine	0	Send Fine	2.99
		Receive Result appeal from Prefecture	2.48
		Add Penalty	1.94
		Insert Date Appeal to Prefecture	1.54
		Appeal to Judge	1.12
		Create Fine	0

The above table shows activities in both the models with their corresponding average waiting times. “Send for Credit Collection” is the activity with the highest average waiting time followed by “Payment” in both the models. However the difference between the average waiting times of “Payment” activity is huge and that was to be expected. Because in the cases where there was appeal, people tend to wait for the results from the prefecture before they consider paying the fine and even then they could appeal to a judge all this extra steps increases the “Payment” activities average waiting time to more than a year. The other thing to notice is that “Send Fine” and “Add Penalty” has almost the same average waiting time in both the models. In Model with appeals “Appeal to Judge” activity has the least waiting time except the starting activity “Create Fine” while in case of Model without appeal “Insert Fine Notification” has the least.

c  oints to Improve the time performance:

- As “Send for Credit Collection” was the highest average waiting time activity in both the models we would suggest them to employ more resources in this activity and also increase the frequency of this activity from once a year to quarterly for better improvements.
- Next is the “Payment” activity in both the model, which has more waiting time than others, the model with appeal is complicated and there is less chance of improvement for this activity. But in case without appeal, this could be improved either by sending a reminder to the offender to pay the fine.
- In both type of models the “Send Fine” activity has average waiting time around 3 months, and we also saw this pattern in the dotted chart of both of these logs, Police department tend to wait for some time before sending the fine. Which delays the whole process. They should send the fine as soon as the fine is created.

- “Send Appeal to Prefecture” was also have higher average waiting time. Police should also assign more resources for performing this activity, so that the later parts of the process doesn’t get delayed because of late appeal sent.

Q5. Summary and Conclusion

In this report, the Police department’s log of the Fine Management System was analysed. We discovered that there were two type of processes which were happening in the System one involving appeal related activities and the other was without those activities. However, we found only 3% out of all the cases fall under With_Appeal, so making Process without appeals more important.

Then we looked on the different activities involved in both type of process and we found Process with appeal involved in total 11 activities while process without appeal involved only 6. “Create Fine” was the first activity in all the cases. In process without appeals there were two activities “Send for Credit Collection” and “Payment” which were the last activity in more than 80% of the cases. In process with appeals “Send Appeal to Prefecture” ended 68% of the cases. We found that “Payment” was the only activity which was happening multiple times.

Then we plotted the dotted chart of the process without appeals and we found some patterns that the Police department was performing certain activity of many cases in batches. Also, we saw the Activity “Send for Credit Collection” was performed once a time in each year, except 2008. Then we found that if the cases end with the “Payment” activity and police do not have to do credit collection that was desirable. And we find some potential problems for which we investigated further in the report.

In question 2 we discovered two models with considering noise threshold at 0.2 from both set of logs, calculated their fitness and precision values. We found that both models had high fitness values (>0.95) and but low precision (<0.75) as both the models were allowing for much more behaviour that was present in the logs. We looked on the deviations in the model, in model without appeals only “Payment” activity happened at arbitrary places in the model, while in model with appeals there were many such blocks where there were only log moves.

Then in question 3 we dig deeper to find out what was happening with the payment, we found out that in 73.5% of the cases the complete payment happened but it was the result of two type of payments either people paid by their own or due to Police’s credit collection activity. Around 13% of the cases people paid zero amount while in 12% of the cases they paid less than the total fine and expense amount resulting in a loss of \$2210639.53. In less than 1 % of the cases people paid extra than what they had to pay.

We looked at the rules which were governing the increment in the total expense amount as the process moves forward. Found that the *Penalty Amount* was always around the double of the *Fine Amount* except for some cases, where it was 3-4 times. We searched for the factors which decide *Payment Status* of a case whether the case is going to end in *No Payment*, *Less Payment*, *Full Payment*{A: Paid by the offender by their own, B: Police had to do the credit collection activity}, *More Payment*. We found that whenever the Penalty Amount is higher than the 18.78 Police must do the credit collection, people did not pay on their own. We also analysed the traces in which payment activity was happening multiple times, we found that the status of such traces ends up in the *Less Payment* resulting a loss to the department. We gave some advice to handle such cases, like sending a notification to the offender before doing the “Add Penalty” activity as offenders were not be able to pay if the amount gets cumulated too much.

In the last section we looked on the performance aspect of both type of processes, we found that “Send for Credit Collection” was the bottleneck in both the processes. There were some solutions were mentioned in the last question like assigning more resource to handle this activity and performing it more frequently as this is the last activity in 40% of the cases i.e. means the by doing

this activity the police department finishes the process for many offenders, but still this was being done once in a year.

Similarly, the “Send Fine” activity in both the logs have waiting time around 3 months, this should also be improved if people will receive the fine early, they will also pay early. We suggest that the department should send the fine either electronically or by other means, but it should be sent as soon as the fine was created.

For process with appeals, since all the activities were taking longer than usual and the number of cases which follow this process was also low, so we suggest making a separate small department to handle such cases. Because the number of activities in this process is almost double to the original one. Due to requirement of different type of activities everything was getting delayed. Or if you cannot make a separate department at least have a status check activity which keep things in control.

For future we would give the following advices to the process owner:

- Do the “Send for Credit Collection” quarterly as opposed to yearly, whether it improves the process or not.
- Look for recurrent “Payment” activities, ask the offenders what the real cause of not paying the full amount. As we saw in some cases people only paid the extra expenses but not the original *Fine Amount*. *Were they facing difficulty with the system or it was purely financial?* If it is because of money, police department can offer them instalment option, so that they can keep paying smaller amount of money.
- “Send Appeal to Prefecture” takes long time, the better idea would be to establish an office in the police department for the prefecture so appeal process can be handled in police department.
- Police department should invest more in Process Mining so that they could predict the Cases in real time and prevent from cases to result in *No Payment or Less Payment*.

Appendix

- At certain places we used some python code which is available in the same zip folder where the report was.
- Also, all the figures are available in respective question folders.