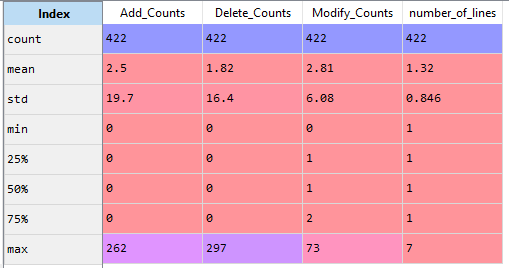
**CA 4 – Perfrom Analysis on a 5000 Line Dataset**

The task of this exercise is to extract, clean and format a large data source that is in an unstructured format and then to analyse statistical pieces of information. Unfortunately the data itself is not very ‘interesting’, the data is a log file from an SVN repository. However, what we can define as interesting is what information we can extract and the type of analysis we can perform on a unstructured dataset.

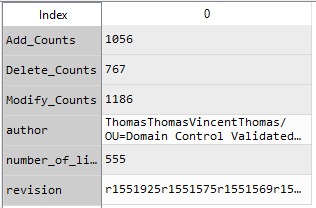
***Descriptive Statistics***

Using functions in the python programme “ConnorDunne\_10361551\_CA4.py” we have cleaned the data, formatted it and stored it into a Dataframe for analysis. The first bit of statistical analysis we can perform on this data set is some descriptive statistics. We can perform summary statistics on our data frame “commits” to give us an idea of the spread, max, min etc. of the data.

*commits.describe()*



This gives us the summary statistics on our integer columns in the data frame. We can see that the average number of lines committed to the SVN is 1. We can also gather information on the sum of the values in the data frame.



However as you can see above our string values return the sequence of strings instead of the sum of unique values. We can find out the sum of unique values in our strings using a mixture of the set() and len() function in python.

*unique\_auth = set(commits["author"])*

*unique\_rev = set(commits["revision"])*

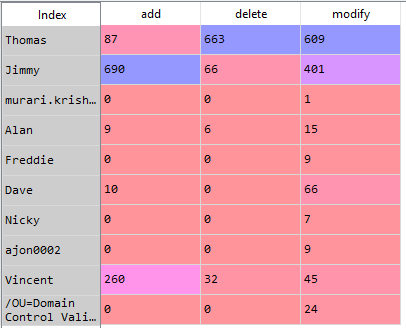
*len(unique\_auth)*

*len(unique\_rev)*

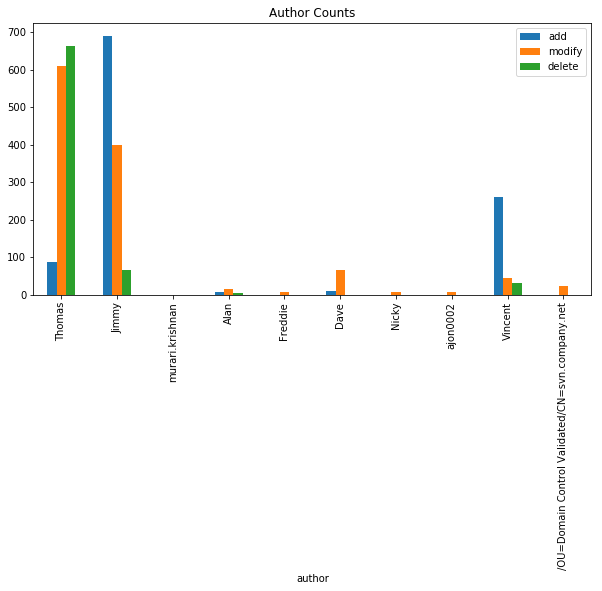
What’s interesting to note is that there are 422 unique revision ID’s, this tells us that each SVN revision is given a unique ID as there are 422 objects in our data.

***Actions per User***

We can go into some more analysis with our data by extracting all the unique author names and summing the total of all their actions (add, modify, delete). Again we store this in a data frame using each author name as the index. Looking at the output from this data frame we can very quickly identify who is performing the most number of actions. Using the table below we can see that the authors “Thomas” and “Jimmy” are performing the most number of actions.



We can also produce bar charts to display this data and again we can identify below that Thomas and Jimmy are executing the greatest number of actions. However, the plot below also reveals that “Vincent” is adding a lot to the repository. This may not have been as clearly evident by studying the table below.

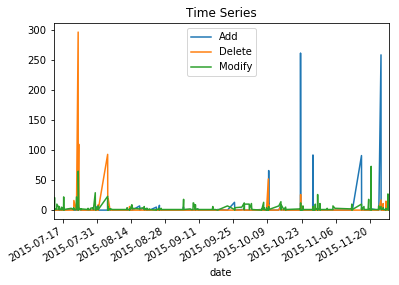


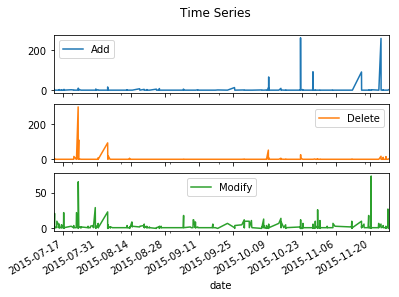
This is not ground breaking analysis and could be identified very quickly using a tool like Excel; however, what is interesting is that we were able to extract this level of information from an unstructured large data set.

***Time Series Analysis***

Looking at the data it was identified that each repository entry is linked with a date and time. This information can be used to perform a time series on the data and identify the changes over time. To accomplish this the first task is to create a series array of the count of each action entry by date and then resample the array by week.

We can see the results of our time series plotted on a line graph. We can compare each action on one chart or as three separate sub plots.





What is interesting from the time series graphs is that we can identify at different points where each action is occurring most frequently. Looking above Modify appears fairly constant however, there are some peaks at the same points the Add and Delete actions peak. Looking more closely it is also evident that Modify and Delete patterns are similar between ‘2015-07-31’ and ‘2015-08-14’. This could suggest some correlation between Modify and Delete actions, which might suggest these actions occur within a project. Correlation analysis tells us that there is not a lot of correlation between Modify and Delete, however there is more correlation between Modify and .Delete than there is correlation between Add and Delete or Add and Modify.

*action\_corr = action\_times.corr()*

*action\_corr*

*sns.heatmap(action\_corr,cmap=sns.diverging\_palette(220, 10,as\_cmap=True),square=True)*

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Add** | **Delete** | **Modify** |
| **Add** | 1.000000 | 0.023762 | 0.020378 |
| **Delete** | 0.023762 | 1.000000 | 0.051103 |
| **Modify** | 0.020378 | 0.051103 | 1.000000 |

