## Case Study – Airlines

Below data is used for the case study – Airline data.

<http://stat-computing.org/dataexpo/2009/the-data.html>

1. Begin Hpcc:

First execute the command hpcc.begin() to start hpcc on R. It is the first and the foremost thing command to be executed before writing any code.

hpcc.begin()

1. Read Data:

First read the whole data from the logical file into variable DS.

DS <- hpcc.dataset('hthor::test::hpcc::airlines\_\_p670296112','thor')

1. Filter :

Let’s take all the data for the year 2008 to carry our analysis. For that we need to filter the data with year = 2008 from the dataset DS.

FilterOut <- hpcc.filter(data=DS,condition='year=2008')

1. Sort Data :

To visualize how the flights are getting delayed. We will sort the data and then visualize how the delays are happening.

SortOut <- hpcc.sort(dataframe=FilterOut,fields='depdelay')

hpcc.output(SortOut,20)

Sample Output:

year month dayofmonth dayofweek deptime crsdeptime arrtime crsarrtime uniquecarrier flightnum tailnum

1 2008 9 29 1 136 1030 456 1335 HA 7 N587HA

2 2008 1 25 5 743 915 1051 1105 YV 7065 N77331

3 2008 2 9 6 1645 1804 2005 2058 AS 64 N794AS

1. Univariate Statistics

We can find the univariate statistics of the delay occurring.

Mean : mean <- hpcc.mean(dataframe=SortOut,fields='depdelay')

hpcc.output(mean)

Output:

Mean

field mean

deptime 1307.904787889059

Mode : mode <- hpcc.mode(dataframe=SortOut,fields='depdelay')

Hpcc.output(mode)

Output:

Mode

field Mode

deptime 0

Median :

median <- hpcc.median(dataframe=SortOut,fields='depdelay')

hpcc.output(median)

Output:

Median

field Median

deptime 1316

1. Percentile : To find out the percentile breakups of a variable from the dataset

percentile <- hpcc.percentile(dataframe=SortOut,'depdelay',55)

hpcc.output(percentile)

|  |  |  |
| --- | --- | --- |
| **Percentiles** | **value** | **pos** |
| 0 | -534 | 1 |
| 1 | -12 | 70097 |
| 5 | -9 | 350486 |
| 10 | -7 | 700972 |
| 25 | -4 | 1752432 |
| 50 | -1 | 3504864 |
| 55 | 0 | 3855350 |
| 75 | 8 | 5257296 |
| 90 | 37 | 6308756 |
| 95 | 69 | 6659242 |
| 99 | 162 | 6939631 |
| 100 | 2467 | 7009728 |

1. Count:

Count Elements of the Dataset: Number of flights which are delayed.

DelayedDS <- hpcc.filter(data=DS,condition=’depdelay>0’)

Count <- hpcc.count(dataset=DelayedDS)

Count

2700974

1. Minimum & Maximum Value function:

To find out the min delay that has occurred in the year 2008.

minDelay <- hpcc.min(dataframe=DS,fields='depdelay')

Similarly to find the maximum delay

maxDelay <- hpcc.max(dataframe=DS,fields='depdelay')

1. Frequency :

Finding out the top most source stations from which the flights are getting delayed

flightFreq <- hpcc.freq(dataframe=DelayedDS, fields=‘origin’)

Output:

|  |  |
| --- | --- |
| ATL | 175017 |
| DFW | 127749 |
| DEN | 104414 |
| EWR | 69612 |
| DTW | 59837 |
| CLT | 48936 |
| BWI | 46748 |
| BOS | 37570 |
| DAL | 31205 |
|  |  |
|  |  |

1. Correlation :

We can find the correlation between the arrival delay and the departure delay. This tells how the arrival delay affects departure delay.

For Spearman Correlation (method =’S’)

correlation <-hpcc.corr(dataframe=DelayedDS,fields='depdepay,arrdelay',method='S')

Output:

Correlation

Arrdelay DepDelay .82359

1. Execute Code:

The output which has been mentioned for every module above comes only we execute the code through the command. After we make sure all the program is done, we can execute the command hpcc.execute(). This command executes the ecl code in Hpcc and brings back the results.

Usage :

hpcc.execute()

1. Downloading Files & Plotting Graphs

After executing the code data will be available, and we have an option to download the entire data apart from viewing it on the console. For that we need to specify to download in the hpcc.output() command. Once we execute our code, we will be driven through that process.

Example(Look into the console part)

Code:

DS <- hpcc.dataset('hthor::test::hpcc::airlines\_\_p670296112','thor')

DelayedDS <- hpcc.filter(data=DS,condition='depdelay>400')

hpcc.output(DelayedDS,30,TRUE)

hpcc.execute()

Console:

|  |
| --- |
| [1] "Below are the files available for download"  [1] "1 -- DelayedDS"  Type the number of the file To Download or 0 to exit : 1  [1] <http://192.168.192.129:8010/FileSpray/DownloadFile?Name=DelayedDS.csv&NetAddress=192.168.192.129&Path=/var/lib/HPCCSystems/mydropzone/&OS=1> #(Download URL)  [1] "D:\\Users\\703134307\\Documents\\RWrkngDir" #(Download Location)  trying URL 'http://192.168.192.129:8010/FileSpray/DownloadFile?Name=DelayedDS.csv&NetAddress=192.168.192.129&Path=/var/lib/HPCCSystems/mydropzone/&OS=1'  Content type 'application/octet-stream' length 325745 bytes (318 Kb)  opened URL  downloaded 318 Kb  [1] "File Downloaded at D:\\Users\\703134307\\Documents\\RWrkngDir\\rhpcc\\downloads  \\DelayedDS.csv"  Type the number of the file To Download or 0 to exit : 0 (Type zero to exit from download process)  Do you want to plot any graph (Y/N): Y  [1] "1 Result 1"  [1] "2 DelayedDS"  Choose the dataset to plot : 2  [1] "1. Histogram"  [1] "2. Scatter Plot"  Choose the graph to plot : 2  [1] "1 year"  [1] "2 month"  [1] "3 dayofmonth"  [1] "4 dayofweek"  [1] "5 deptime"  [1] "6 crsdeptime"  [1] "7 arrtime"  [1] "8 crsarrtime"  [1] "9 uniquecarrier"  [1] "10 flightnum"  [1] "11 tailnum"  [1] "12 actualelapsedtime"  [1] "13 crselapsedtime"  [1] "14 airtime"  [1] "15 arrdelay"  [1] "16 depdelay"  [1] "17 origin"  [1] "18 dest"  [1] "19 distance"  [1] "20 taxiin"  [1] "21 taxiout"  [1] "22 cancelled"  [1] "23 cancellationcode"  [1] "24 diverted"  [1] "25 carrierdelay"  [1] "26 weatherdelay"  [1] "27 nasdelay"  [1] "28 securitydelay"  [1] "29 lateaircraftdelay"  Input the variable to plot as X-axis :15  Input the variable to plot as Y-axis :16 |
|  |
| |  | | --- | |  | |

Graph:

