Milestone Report

## The general problem description (based upon early submission)

**What is the problem you want to solve?**

The general problem is that software development companies might have a clear idea on the significance of different types of software architectural styles. Hence, the specific problem that companies need to understand the popularity of these software architectural styles.

**Who is your client and why do they care about this problem? In other words, what will your client DO or DECIDE based on your analysis that they wouldn’t have otherwise?**

The potential client for this project could be software development companies, or research agencies. Based upon the data analysis, organiziations could gain a clear understanding on the usage of the different architechtural styles for software development, among various level of software developers.

-**What data are you going to use for this? How will you acquire this data?**

The data are available via Kaggle <http://www.kaggle.com> in a single dataset with various updated versions. These datasets are publicly available for downloading.

**In brief, outline your approach to solving this problem (knowing that this might change later).**

The variables of the project are software architechtural styles, software developers’ job experiences, and their education levels. The data analysis will include linear regression and ranking, as well as relevant graph for data visualization.

**What are your deliverables? Typically, this would include code, along with a paper and/or a slide deck.**

The deliverables of the project include R code, a report (possibly be written by R markdown).

## Data wrangling process

The following texts summarizes the important steps for cleaning up the data of software architechtural styles.

**Renamed variables names to keep simple and meaningful**

Some of the column names in the original dataset were relatively long. Short and simple names were given to represent the actual meaning of the columns. (See comments in R codes)

**Separated variable “Industry”**

The variable “Industry” contained various format of values taking from the original survey. Separating the responses to tidy the data.

**Shortened variable “Education”**

The variable “Education” contained redundant information. Removing the additional information to tidy the data.

**Checked the uniqueness of observations**

Some of the observations might be duplicated entries that have happened due to various reasons. Removing the duplicates when found to tidy the data.

**Checked missing values**

Some of the observations might be missing values that have happened due to various reasons. Omitting the observations that contain missing values when found to tidy the data (given the large amount of samples).

*After the data wrangling step, a cleaned copy of dataset is ready for statistical analysis.*

## Preliminary findings

The following information shows the preliminary analysis to the software architechtural styles survey data.

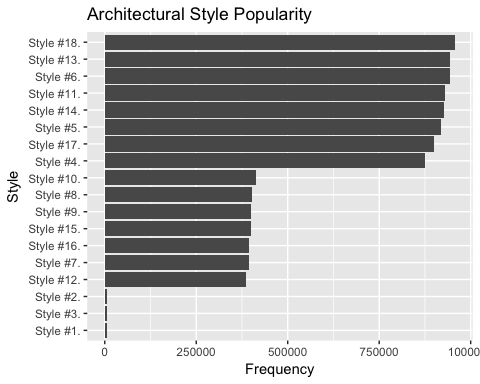
**First, read the cleaned data as a R object into the workspace and explore the structure of the data frame.**

## 'data.frame': 1121 obs. of 23 variables:  
## $ Timestamp : Factor w/ 999 levels "2016/12/11 1:41:33 PM GMT+5",..: 47 49 50 53 54 60 61 61 62 2 ...  
## $ Name : Factor w/ 669 levels " Inayat m","aalia",..: 356 390 171 660 290 26 300 300 153 621 ...  
## $ Organization.: Factor w/ 253 levels " KUST","Abbsen uni",..: 116 180 65 37 9 157 182 182 30 128 ...  
## $ Education : chr "BSC" "MS" "BSC" "BSC" ...  
## $ Industry : chr "Other" "Education" "Other" "Other" ...  
## $ Style #1. : num 3 5 1 2 2 2 1 1 2 2 ...  
## $ Style #2. : int 4 3 0 3 1 3 10 10 3 2 ...  
## $ Style #3. : int 2 2 0 2 1 1 1 1 0 0 ...  
## $ Style #4. : int 1455 50 0 1000 2 0 400 400 3 1000 ...  
## $ Style #5. : int 2342 20 0 1500 1 6 15 15 100 50 ...  
## $ Style #6. : num 1000 20 0 1700 0 18 10 10 100 30 ...  
## $ Style #7. : int 859 10 0 500 3 6 5 5 100 20 ...  
## $ Style #8. : num 456 20 0 600 0 7 10 10 15 0 ...  
## $ Style #9. : num 232 5 0 600 1 2 5 5 0 1 ...  
## $ Style #10. : num 965 10 0 500 4 1 0 0 2 1 ...  
## $ Style #11. : num 2045 50 0 1500 3 ...  
## $ Style #12. : num 543 10 0 400 1 1 50 50 2 0 ...  
## $ Style #13. : int 2955 10 0 1600 1 6 300 300 5 0 ...  
## $ Style #14. : int 1004 0 0 1400 1 13 1 1 500 2 ...  
## $ Style #15. : num 356 0 0 300 0 0 0 0 100 0 ...  
## $ Style #16. : int 289 10 0 500 0 0 10 10 3 0 ...  
## $ Style #17. : num 2006 10 1 600 2 ...  
## $ Style #18. : num 1594 20 0 1450 1 ...

A table to map styles

|  |  |
| --- | --- |
| Style #1. | Repository |
| Style #2. | Client Server |
| Style #3. | Abstract Machine |
| Style #4. | Object Oriented |
| Style #5. | Function Oriented |
| Style #6. | Event Driven |
| Style #7. | Layered |
| Style #8. | Pipes Filters |
| Style #9. | Data centeric |
| Style #10. | Blackboard |
| Style #11. | Rule Based |
| Style #12. | Publish Subscribe |
| Style #13. | Asynchronous Messaging |
| Style #14. | Plug ins |
| Style #15. | Micro kernel |
| Style #16. | Peer to Peer |
| Style #17. | Domain Driven |
| Style #18. | Shared Nothing |

**Then, using a barplot to show the popularity of the different software architechtural styles.**



**These styles have actually been divided into three groups that are ranked by the popularity.**

The most poupular styles are:

## [1] "Style #4." "Style #5." "Style #6." "Style #11." "Style #13."  
## [6] "Style #14." "Style #17." "Style #18."

Less poupular styles are:

## [1] "Style #7." "Style #8." "Style #9." "Style #10." "Style #12."  
## [6] "Style #15." "Style #16."

The least poupular styles are:

## [1] "Style #1." "Style #2." "Style #3."

## Limitations

The dataset that has been utilized in the project might only contain the limited information about the software architechtual styles being used in the software development process. However, the dataset may demonstrate the popularity of the styles.

## Updated approach

The next step going forward will be using the current data to build a logistic regression model (with 70% of the samples) as training data. Then, using the rest of 30% data to test the model, regarding the prediction on a particular software architectural style might be selected.