# Mining Process Quality Prediction

## Advanced-Data Analysis and Machine Learning

**BM29A6100**

Jhuma Mim

## Table of Content

## Week 2: Data Summary & Visualization

The current dataset is taken from a mining manufacturing process, from the froth flotation phase. We have collected this dataset from Keggal. The dataset contains design characteristics of iron ore froth flotation processing plants which were put together within three (3) months.

The aim of the process is to have as little silica content in the outlet as possible. There are 24 columns in data, as seen in the table below. The aim is to predict the impurity percentage in the ore, which in this case is represented by the 24th variable: silica ore. Therefore our target column is 24 which % Silica Concentrate. There are a total of 737453 rows × 24 columns.

```
]: df=pd.read_csv( mining_dataset.csv )
   df.head(5)
```

]:

| % ca ed | Starch Flow | Amina Flow | Ore Pulp Flow | Ore Pulp pH | Ore Pulp Density | Flotation Column 01 Air Flow | Flotation Column 02 Air Flow | ... | Flotation Column 07 Air Flow | Flotation Column 01 Level | Flotation Column 02 Level | Flotation Column 03 Level | Flotation Column 04 Level | Flotation Column 05 Level | Flotation Column 06 Level | Flotation Column 07 Level | % Concen |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 98 | 3019,53 | 557,434 | 395,713 | 10,0664 | 1,74 | 249,214 | 253,235 | ... | 250,884 | 457,396 | 432,962 | 424,954 | 443,558 | 502,255 | 446,37 | 523,344 | 6 |
| 98 | 3024,41 | 563,965 | 397,383 | 10,0672 | 1,74 | 249,719 | 250,532 | ... | 248,994 | 451,891 | 429,56 | 432,939 | 448,086 | 496,363 | 445,922 | 498,075 | 6 |
| 98 | 3043,46 | 568,054 | 399,668 | 10,068 | 1,74 | 249,741 | 247,874 | ... | 248,071 | 451,24 | 468,927 | 434,61 | 449,688 | 484,411 | 447,826 | 458,567 | 6 |
| 98 | 3047,36 | 568,665 | 397,939 | 10,0689 | 1,74 | 249,917 | 254,487 | ... | 251,147 | 452,441 | 458,165 | 442,865 | 446,21 | 471,411 | 437,69 | 427,669 | 6 |
| 98 | 3033,69 | 558,167 | 400,254 | 10,0697 | 1,74 | 250,203 | 252,136 | ... | 248,928 | 452,441 | 452,9 | 450,523 | 453,67 | 462,598 | 443,682 | 425,679 | 6 |

Fig 1: Sample snippet of dataset showing first 5 rows

### Modeling goals

For this project, we aim to evaluate the feasibility of using machine learning algorithms to predict the percentage of silica concentrate in the froth flotation processing plant. Before model development, we will determine which variable associated with silica concentrate ore extraction is statistically significant. To Investigate the most important variables in prediction and reduce the inlet variables from the dataset.If we are successful in model development we will try to

determine the minimum sampling frequency for quality prediction, in other words, what is the maximum interval between measurements so that the prediction quality is maintained.

## Data Summary

Initially, we thought in our dataset ‘date’ column might not be useful for analysis or model development. Therefore, we will not use this column in our analysis.

| | | |
|---|---|---|
| Silica Feed mean: 14.65171555339798<br>Silica Feed max: 33.4<br>Silica Feed min: 1.31<br><br>Starch Flow mean: 2869.1405694542955<br>Starch Flow max: 6300.23<br>Starch Flow min: 0.00202596<br><br>Amina Flow mean: 488.14469722409433<br>Amina Flow max: 739.538<br>Amina Flow min: 241.669<br><br>Ore Pulp Flow mean: 397.57837171046845<br>Ore Pulp Flow max: 418.641<br>Ore Pulp Flow min: 376.249<br><br>Ore Pulp pH mean: 9.767638747852406<br>Ore Pulp pH max: 10.8081<br>Ore Pulp pH min: 8.75334<br><br>Ore Pulp Density mean: 1.680379685600307<br>Ore Pulp Density max: 1.85325<br>Ore Pulp Density min: 1.51982<br><br>Flotation Column 01 Air Flow mean: 280.151856174563<br>Flotation Column 01 Air Flow max: 373.871<br>Flotation Column 01 Air Flow min: 175.51<br>Flotation Column 03 Level mean: 531.3526623255991<br>Flotation Column 03 Level max: 886.822<br>Flotation Column 03 Level min: 126.255<br>Flotation Column 04 Level mean: 420.3209730796404<br>Flotation Column 04 Level max: 680.359<br>Flotation Column 04 Level min: 162.201 | Flotation Column 02 Air Flow mean: 277.15996522286844<br>Flotation Column 02 Air Flow max: 375.992<br>Flotation Column 02 Air Flow min: 175.156<br><br>Flotation Column 03 Air Flow mean: 281.0823972775214<br>Flotation Column 03 Air Flow max: 364.346<br>Flotation Column 03 Air Flow min: 176.469<br><br>Flotation Column 04 Air Flow mean: 299.44779355836937<br>Flotation Column 04 Air Flow max: 305.871<br>Flotation Column 04 Air Flow min: 292.195<br><br>Flotation Column 05 Air Flow mean: 299.91781411764566<br>Flotation Column 05 Air Flow max: 310.27<br><br>Flotation Column 05 Air Flow min: 286.295<br>Flotation Column 06 Air Flow mean: 292.07148503904654<br>Flotation Column 06 Air Flow max: 370.91<br>Flotation Column 06 Air Flow min: 189.928<br><br>Flotation Column 07 Air Flow mean: 290.75485625863615<br>Flotation Column 07 Air Flow max: 371.593<br>Flotation Column 07 Air Flow min: 185.962<br>Flotation Column 01 Level mean: 520.244822748026<br>Flotation Column 01 Level max: 862.274<br>Flotation Column 01 Level min: 149.218<br>Flotation Column 02 Level mean: 522.6495545316109<br>Flotation Column 02 Level max: 828.919<br>Flotation Column 02 Level min: 210.752 | Flotation Column 05 Level mean: 425.2517058999014<br>Flotation Column 05 Level max: 675.644<br>Flotation Column 05 Level min: 166.991<br><br>Flotation Column 06 Level mean: 429.9410177177393<br>Flotation Column 06 Level max: 698.861<br>Flotation Column 06 Level min: 155.841<br><br>Flotation Column 07 Level mean: 421.021230553676<br>Flotation Column 07 Level max: 659.902<br>Flotation Column 07 Level min: 175.349<br><br>Iron Concentrate mean: 65.05006799077366<br>Iron Concentrate max: 68.01<br>Iron Concentrate min: 62.05<br><br>Silica Concentrate mean: 2.3267632513529675<br>Silica Concentrate max: 5.53<br>Silica Concentrate min: 0.6 |

Table 1: Each column mean, maximum and minimum range

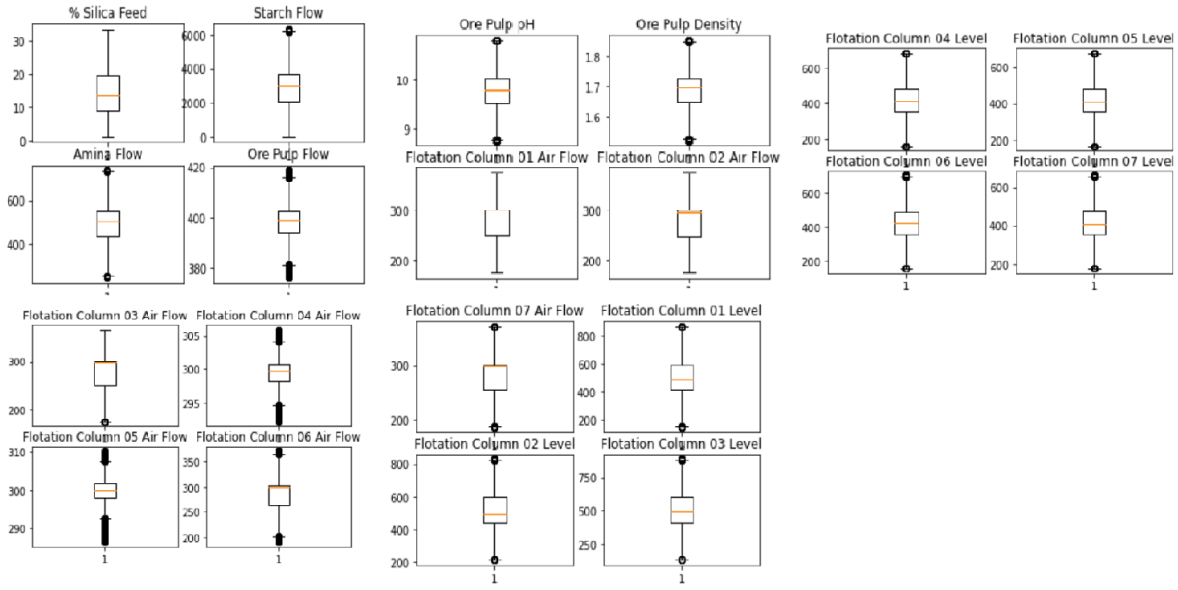



Fig 2: Outlier boxplot of different variables.

For outliers analysis, we have plotted each variable boxplot and we found some columns have a significant number of outliers. For example, ‘Ore Pump Flow’ and ‘Floating column 4 and 5’ have a high number of outliers. In contrast, some columns have no outliers at all (ex. Silica feed, Flotation 01 and 02 airflow column).

The scatter plotting can be shown as follows:

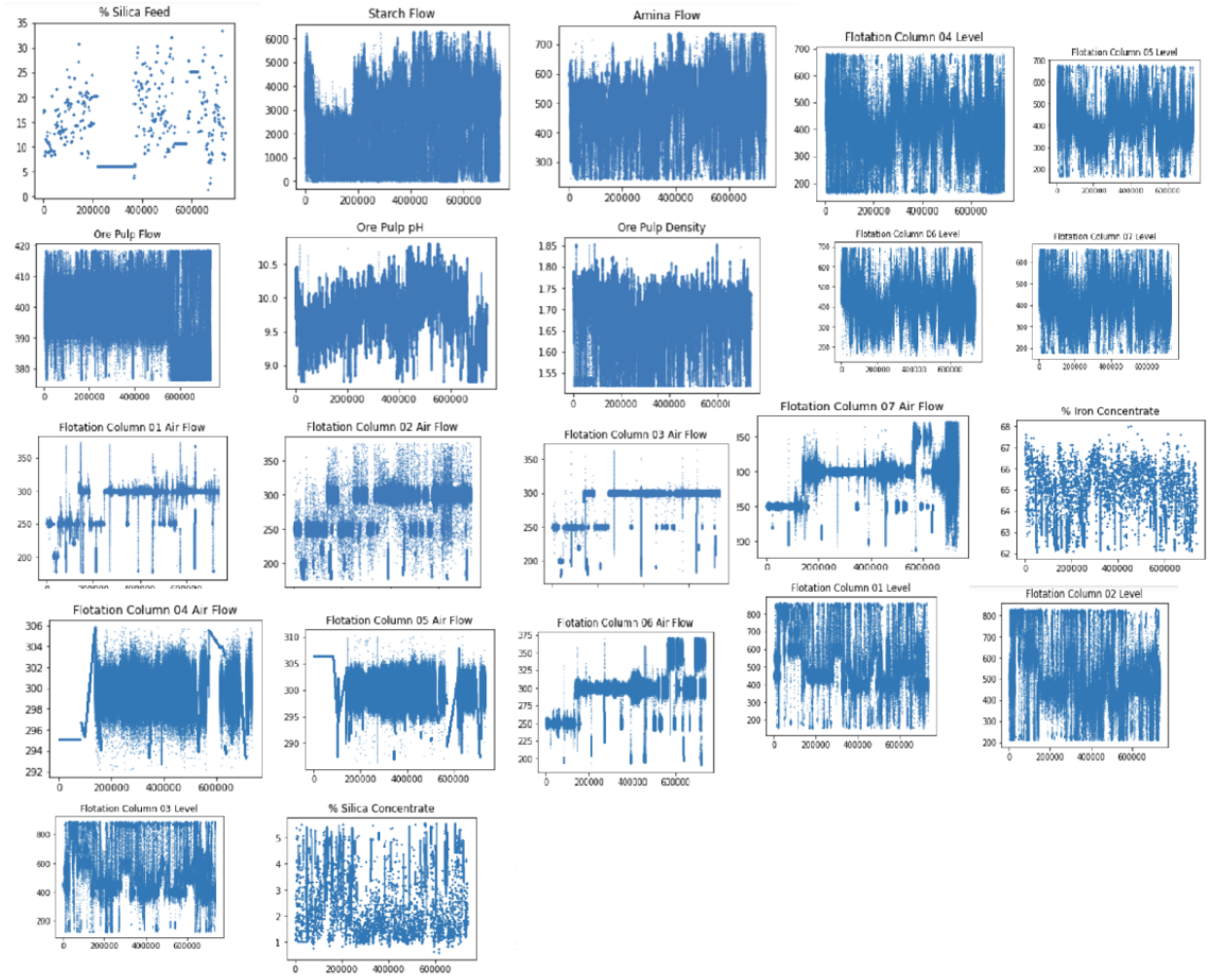



Fig 3: Plot of scattering value of dataset

From the scatter value plot we see that all variables have a good representation of values. We observe few data have outliers which verify our outliers plot.

The variable distribution can be shown as follows:

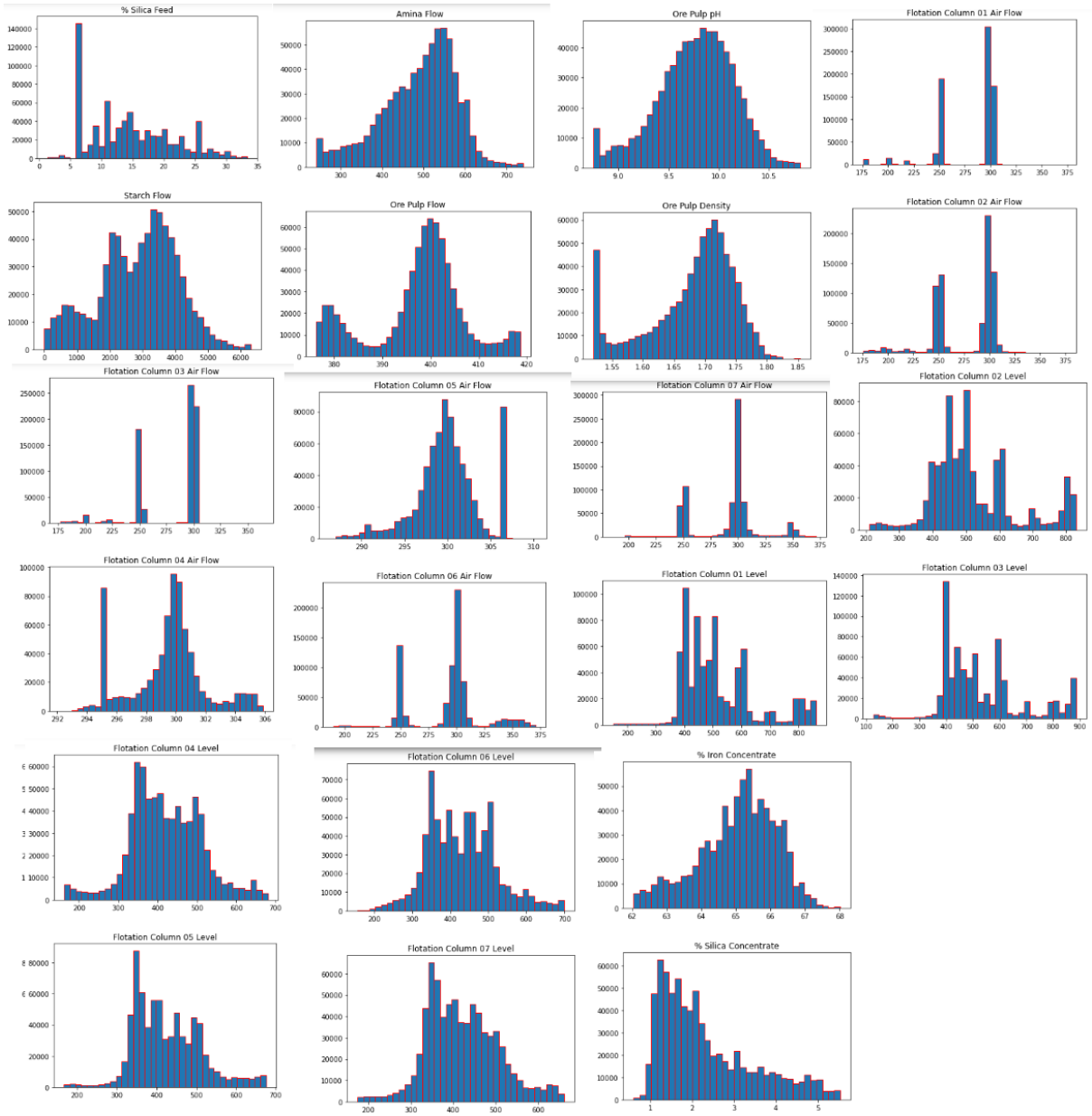



Fig 4: Distribution plot of different variables.

From the variable distribution plot, we can observe that almost 14 variables show some sort of normal distribution. Yes, it does not perfectly follow the normal distribution pattern for all cases but it's good enough to consider a normal distribution.

# Week 3: Data Pre-treatment

## Down-Sampling

Initially we thought there were no missing values, however after deep inspection, we noticed missing values in our data that have been replaced with a constant which usually appears in the same Hours time frame consecutively. To deal with this situation we're-sample the dataset so that the predicted variable has the same frequency (1h) as the rest of the measurements, by hourly averaging each column.

| | date | % Iron Feed | % Silica Feed | Starch Flow | Amina Flow | Ore Pulp Flow | Ore Pulp pH | Ore Pulp Density | Flotation Column 01 Air Flow | Flotation Column 02 Air Flow | ... | Flotation Column 07 Air Flow | Flotation Column 01 Level | Flotation Column 02 Level | Flotation Column 03 Level | Flotation Column 04 Level | Flotation Column 05 Level |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3019,53 | 557,434 | 395,713 | 10,0664 | 1,74 | 249,214 | 253,235 | ... | 250,884 | 457,396 | 432,962 | 424,954 | 443,558 | 502,255 |
| 1 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3024,41 | 563,965 | 397,383 | 10,0672 | 1,74 | 249,719 | 250,532 | ... | 248,994 | 451,891 | 429,56 | 432,939 | 448,086 | 496,363 |
| 2 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3043,46 | 568,054 | 399,668 | 10,068 | 1,74 | 249,741 | 247,874 | ... | 248,071 | 451,24 | 468,927 | 434,61 | 449,688 | 484,411 |
| 3 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3047,36 | 568,665 | 397,939 | 10,0689 | 1,74 | 249,917 | 254,487 | ... | 251,147 | 452,441 | 458,165 | 442,865 | 446,21 | 471,411 |
| 4 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3033,69 | 558,167 | 400,254 | 10,0697 | 1,74 | 250,203 | 252,136 | ... | 248,928 | 452,441 | 452,9 | 450,523 | 453,67 | 462,598 |
| 5 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3079,1 | 564,697 | 396,533 | 10,0705 | 1,74 | 250,73 | 248,906 | ... | 251,873 | 444,384 | 443,269 | 460,449 | 439,92 | 451,588 |
| 6 | 2017-03-10 01:00:00 | 55,2 | 16,98 | 3127,79 | 566,467 | 392,9 | 10,0713 | 1,74 | 250,313 | 252,202 | ... | 253,477 | 446,185 | 444,571 | 452,306 | 431,328 | 443,548 |
| 7 | 2017- | | | | | | | | | | | | | | | | |

Fig 5: Repeated same constant value in same hour marked as red

After sampling, the number of rows reduces to 4415 from 737453 rows which is a 99.4% reduction from the original dataset.

## Scaling and Centering Data:

After sampling, the total number of variables is 24 and there are 4415 rows. We have not deal with any categorical variables since all the variables in the dataset are numerical. The dataset has not many observations in terms of categories.

```
print(df.dtypes)
```

```
date                           datetime64[ns]
% Iron Feed                           float64
% Silica Feed                         float64
Starch Flow                           float64
Amina Flow                            float64
Ore Pulp Flow                         float64
Ore Pulp pH                           float64
Ore Pulp Density                      float64
Flotation Column 01 Air Flow          float64
Flotation Column 02 Air Flow          float64
Flotation Column 03 Air Flow          float64
Flotation Column 04 Air Flow          float64
Flotation Column 05 Air Flow          float64
Flotation Column 06 Air Flow          float64
Flotation Column 07 Air Flow          float64
Flotation Column 01 Level             float64
Flotation Column 02 Level             float64
Flotation Column 03 Level             float64
Flotation Column 04 Level             float64
Flotation Column 05 Level             float64
Flotation Column 06 Level             float64
Flotation Column 07 Level             float64
% Iron Concentrate                    float64
% Silica Concentrate                  float64
dtype: object
```

Fig 6: each observation data type,

However, from fig-7 we can observe that variables have different measurement range and different scale, therefore, we needed to scale each observation in the same range.

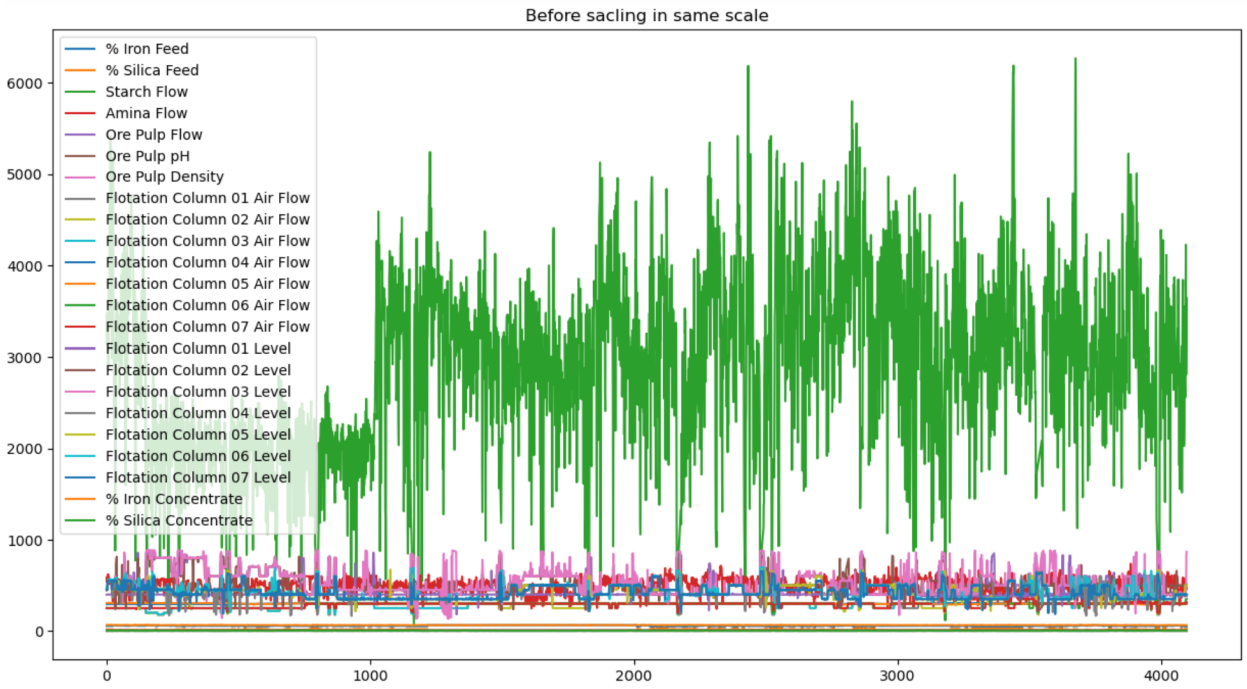



Fig 7:Before scaling the each column

```
from scipy.stats import zscore
f.apply(zscore)
```

| | % Iron Feed | % Silica Feed | Starch Flow | Amina Flow | Ore Pulp Flow | Ore Pulp pH | Ore Pulp Density | Flotation Column 01 Air Flow | Flotation Column 02 Air Flow | Flotation Column 03 Air Flow | ... | Flotation Column 07 Air Flow | Flotation Column 01 Level | Flotation Column 02 Level | Flotation Column 03 Level |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -0.212251 | 0.342020 | 0.308811 | 1.083194 | 0.140386 | 0.914982 | 0.771216 | -0.985665 | -0.915495 | -1.089311 | ... | -1.489760 | -0.571874 | -0.652849 | -0.583654 |
| **1** | -0.212251 | 0.342020 | 0.277909 | 0.586454 | 0.274018 | 0.957985 | -0.197532 | -1.029399 | -0.915904 | -1.094421 | ... | -1.501172 | -0.580146 | -0.623916 | -0.586492 |
| **2** | -0.212251 | 0.342020 | 0.642218 | 1.239983 | 0.141633 | 0.742794 | 0.820657 | -1.019853 | -0.919639 | -1.093962 | ... | -1.479619 | -0.575238 | -0.618578 | -0.580570 |
| **3** | -0.212251 | 0.342020 | 0.377639 | 1.255080 | 0.273440 | 0.399418 | 0.794703 | -1.018239 | -0.916220 | -1.091336 | ... | -1.491618 | -0.264436 | -0.268758 | -0.317273 |
| **4** | -0.212251 | 0.342020 | 0.482067 | 1.572251 | 0.243344 | -0.057179 | 1.340809 | -1.028134 | -0.917724 | -1.095120 | ... | -1.494520 | 0.235650 | 0.235556 | 0.130673 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **4092** | -1.268920 | 1.255731 | 0.481821 | 0.108340 | -1.999028 | -1.564830 | -0.305942 | 0.728207 | 0.782108 | 0.663327 | ... | 0.293283 | -1.002473 | -0.205925 | -0.920055 |
| **4093** | -1.268920 | 1.255731 | 1.427514 | 0.248788 | -1.967283 | -1.005608 | 0.182163 | 0.676674 | 0.775266 | 0.663532 | ... | 0.471946 | -0.984898 | -0.198207 | -0.953615 |
| **4094** | -1.268920 | 1.255731 | -0.064109 | 0.353772 | -1.973075 | 0.009618 | 0.866698 | 0.668653 | 0.769614 | 0.665880 | ... | 0.702546 | -0.984629 | -0.191003 | -0.429560 |
| **4095** | -1.268920 | 1.255731 | 0.339192 | 0.052174 | -2.047082 | 0.038307 | 0.568701 | 0.679143 | 0.765464 | 0.664254 | ... | 0.532765 | -0.987576 | -0.207121 | 2.200494 |
| **4096** | -1.268920 | 1.255731 | 0.809586 | -0.327145 | -2.001901 | -0.301179 | -0.196885 | 0.667449 | 0.779602 | 0.661216 | ... | 0.900561 | -0.984098 | -0.197579 | 2.433386 |

Fig 8: Scaling the dataset using z-score function

After down-sampling, we use the z-score function  for scaling each variable in the same scale and center the data.  A z-score is a numerical measurement that describes a value's relationship to the mean of a group of values.

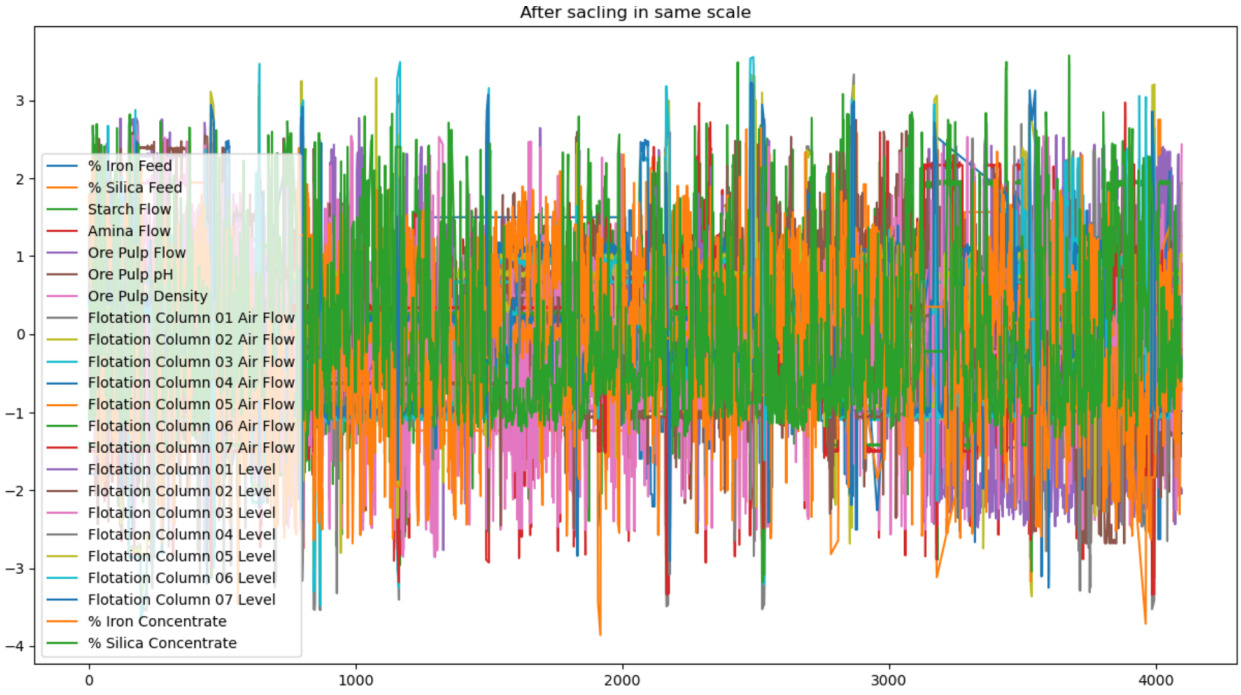



Fig 9: After scaling the each column z-score function

After the data has been standardized, we can see that every variable has the same ranges of measurement and scale. Moreover, we notice that the variables are all centered at zero.

## Outliers Handling:

From box plot observation (example fig 10), we initially thought there were some outliers and planned to replace them with a clipped lower and upper bound. However, after observation, those outliers are possible due to their distance from the mean  (fig:11), which seems not impossible, and ranges are quite small. Therefore, we have not yet to deal with outliers since we don't have much domain knowledge about mining and this range of outliers looks pretty possible.

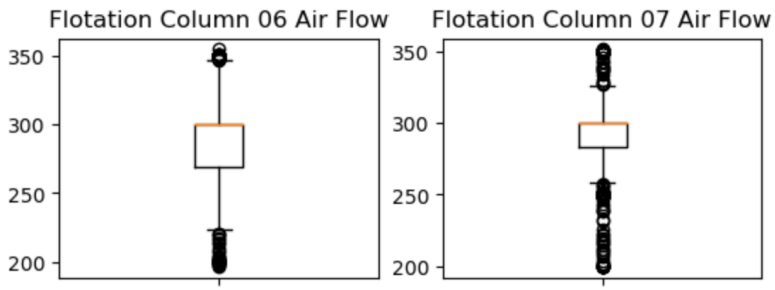



Fig 10: Example of box plot figure that shows outliers.

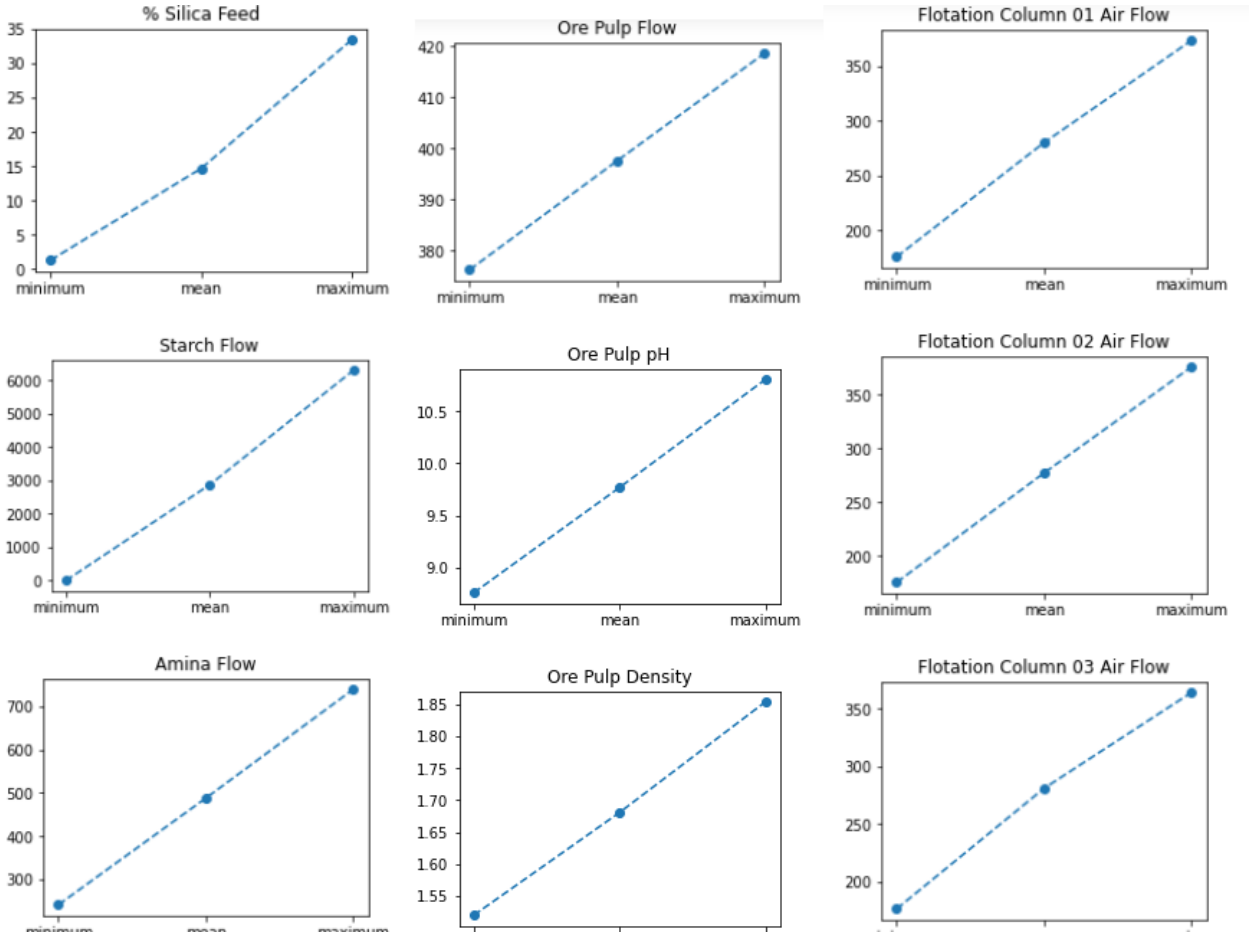



Fig 11: Plot of the  maximum, minimum and mean value of each column.

## Week 4 : Make a modeling plan

This week, the Mining process dataset contains different measurements of Mining 24 columns.These columns in data as seen in the table-2 below:

| Var. No. | Parameter |
|---|---|
| 1 | Date |
| 2 | Iron Conc. in Feed [%] |
| 3 | Silica Conc. In Feed [%] |
| 4 | Starch Flow |
| 5 | Amina Flow |
| 6 | Ore Pulp Flow |
| 7 | Ore Pulp pH |
| 8 | Ore Pulp Density |
| 9 | Flotation Column 01 Air Flow |
| 10 | Flotation Column 02 Air Flow |
| 11 | Flotation Column 03 Air Flow |
| 12 | Flotation Column 04 Air Flow |
| 13 | Flotation Column 05 Air Flow |
| 14 | Flotation Column 06 Air Flow |
| 15 | Flotation Column 07 Air Flow |
| 16 | Flotation Column 01 Level |
| 17 | Flotation Column 02 Level |
| 18 | Flotation Column 03 Level |
| 19 | Flotation Column 04 Level |
| 20 | Flotation Column 05 Level |
| 21 | Flotation Column 06 Level |
| 22 | Flotation Column 07 Level |
| 23 | Outlet Iron Conc. [%] |
| 24 | Outlet Silica Conc. [%] |

Table 2: Columns of mining dataset

The Silica concentrate % are important since silica ore is directly connected to the mining process. Here, we aim to predict silica ore based on other data. In short,

**Goals 1:** Predict the outlet silica content in the ore. So we would investigate the most important variables in prediction and reduce from the dataset the inlet variables as much as while maintaining closely the prediction accuracy.

**Goal 2:** Determine the minimum sampling frequency for quality prediction that is finding maximum interval between measurements so that the prediction quality is maintained. And

finally we will Lag the variables and try to forecast the quality of ore. How much into the future can you predict?

Our first goal is a prediction dataset using the Principal Component Regression (PCR) Model. It is a regression technique that serves the same goal as standard linear regression. It is also based on principal component analysis (PCA).— model the relationship between a target variable and the predictor variables.

$$y = mX + C$$

Where PCA performs based on X. Compute the linear regression score vectors and response variable y, and C is regression coefficients.

Another Model is Partial Least Squares Regression (PLS). It is related to (Ordinary Least Squares) the standard mathematical approach for fitting a Linear Regression.

The difference between both of them is PCR is focused on variance , on the other hand PLS is considered on covariance while reducing the dimensions.

During the model planning process we will follow these steps showing it by flow chart. A short description of modeling process are given below:

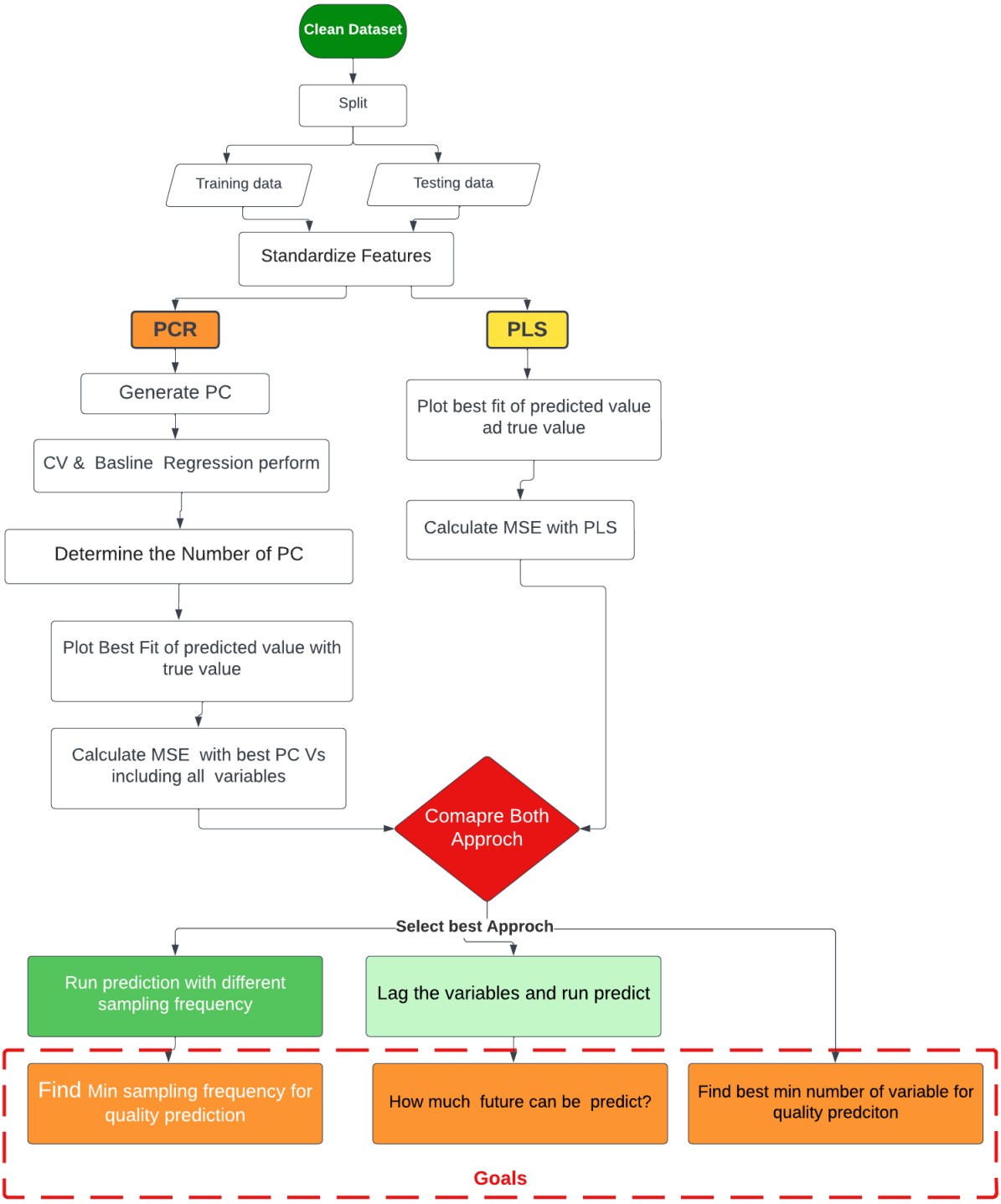



Fig 12: Flow chart of modeling process work

From the clean dataset, we will divide the dataset into training and test samples. We will select 20% random data for the test and the rest of 80% for training. It is important to standardize each original feature to be on the same scale and centering before generating the principal components. Because if we don't use it the features with larger ranges (and thus higher variance) will dominate and play an unfairly huge role in the principal components generated. To evaluate the performance of the PCR model, we will need to have benchmarks to compare with. So we will apply linear regression and cross validation (CV).After that, We will generate principal components (PC).Now from PC we will Keep the principal components that explain most of the variance (where N < p), where N is determined by cross-validation and visual analysis**.** We will iterate over an increasing number of principal components to include in regression modeling and assess the resulting Means square error (MSE) scores**.**

$$MSE = \frac{\sum(y_i - \hat{y}_i)^2}{n}$$

Where:

- yi is the ith observed value.
- ŷi is the corresponding predicted value.
- n = the number of observations.

Next, we will inspect the training set cross-validation MSE with the number of principal components needed to find the best Prediction result. After determining the best number of principal components, we proceed to run PCR on our test dataset. We will predict the MSE with the best PC by comparing the original date level with prediction data Level. The lower score of MSE will represent better performance of the model

We would compare both PCR and PLS approaches and will use the best one for answering our goals.

## Week 6- Result of Model Planning

According to our modeling plan, We divided the dataset for test 20% random data and 80% for training. We standardize each original feature to be on the same scale and centering before applying the PCR Model.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 12 | 13 | 14 | 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.047254 | -0.041941 | -0.457574 | 0.468376 | -0.137587 | 0.142140 | -0.026310 | -0.084942 | -0.100856 | -0.059005 | ... | -0.010258 | -0.000189 | 0.008202 | -0.003501 | -0.0 |
| 1 | -0.058343 | 0.042330 | 0.459303 | -0.455392 | 0.145255 | -0.152848 | 0.048017 | 0.096605 | 0.109352 | 0.067104 | ... | 0.013760 | 0.016497 | 0.023437 | 0.002805 | 0.0 |
| 2 | -0.167520 | 0.038420 | -0.314363 | -0.232610 | 0.000338 | 0.197711 | 0.143296 | -0.227652 | 0.528074 | 0.172668 | ... | 0.079877 | 0.050372 | -0.006632 | -0.035362 | 0.0 |
| 3 | -0.128897 | -0.171869 | -0.211914 | -0.386131 | -0.375406 | 0.143575 | -0.120690 | 0.175265 | -0.207287 | 0.097037 | ... | 0.084511 | 0.580206 | -0.139402 | 0.259406 | -0.2 |
| 4 | 0.046144 | -0.122049 | -0.342875 | -0.070434 | 0.062916 | -0.255600 | 0.064574 | 0.767141 | -0.078732 | 0.261357 | ... | -0.020156 | -0.233386 | 0.116476 | -0.072429 | 0.1 |

Fig 13: after scaling and centering the dataset.

## PCR Model

To perform the PCR Model, firstly we generate cross validation (CV) and Linear Regression. After that, we applied the principal components (PC) to generate the number of PCs in regression. We used a benchmark to compare it.

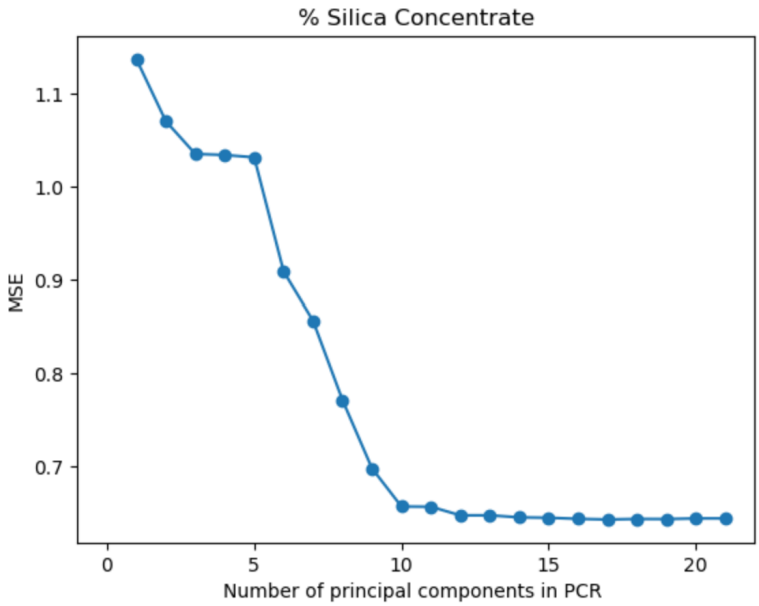



Fig 14: Number of PC in regression for PCR

We see that the training set performance of PCR improves (i.e., MSE decreases) with more principal components, in line with what we expect. The line is the MSE benchmark from the baseline standard linear regression model using all original features. The plot shows that the lowest cross-validation MSE (minimum point in the plot) occurs when there are 21 principal components. However, we can see if we use 12 PCs it also produces similar performance (0.41) as the rest of the other PCs are almost similar.

```
mean_squared_error with PC number 1 : 1.2941344159576758
mean_squared_error with PC number 2 : 1.1466921744665834
mean_squared_error with PC number 3 : 1.0726066345903815
mean_squared_error with PC number 4 : 1.0702670120200368
mean_squared_error with PC number 5 : 1.0650422012730407
mean_squared_error with PC number 6 : 0.8264788449078837
mean_squared_error with PC number 7 : 0.7336137521098713
mean_squared_error with PC number 8 : 0.5946614008024075
mean_squared_error with PC number 9 : 0.4869165731362718
mean_squared_error with PC number 10 : 0.43129873868902163
mean_squared_error with PC number 11 : 0.4309568470270762
mean_squared_error with PC number 12 : 0.4189091010415173
mean_squared_error with PC number 13 : 0.4189282529499689
mean_squared_error with PC number 14 : 0.41607722268044817
mean_squared_error with PC number 15 : 0.4155184767374524
mean_squared_error with PC number 16 : 0.4140282297030072
mean_squared_error with PC number 17 : 0.413140470303283
mean_squared_error with PC number 18 : 0.41387731661219074
mean_squared_error with PC number 19 : 0.41375476610040607
mean_squared_error with PC number 20 : 0.4145938564553757
mean_squared_error with PC number 21 : 0.4146014685728496
```

Fig 15:MSE Scores of PCR Model

**We notice the most important variables in our analysis. 'Iron Feed'** is the essential feature, and the least important part is 'Iron concentration' shown in fig-16. From the previous mean square error result, we have seen after using PC 12 components mean square error remains almost the same. Therefore, we will keep the first 12 significant features as most important variables in our dataset for the PCR calculation.

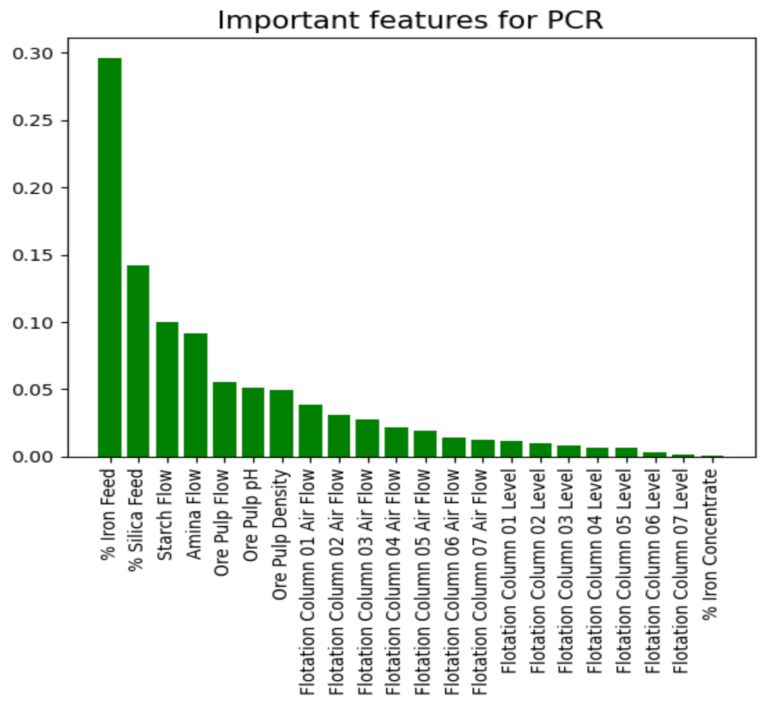



Fig 16: Important features of PCR Model

We have run the model and tested it with test data. We calculate the error from the original level and predicted level.

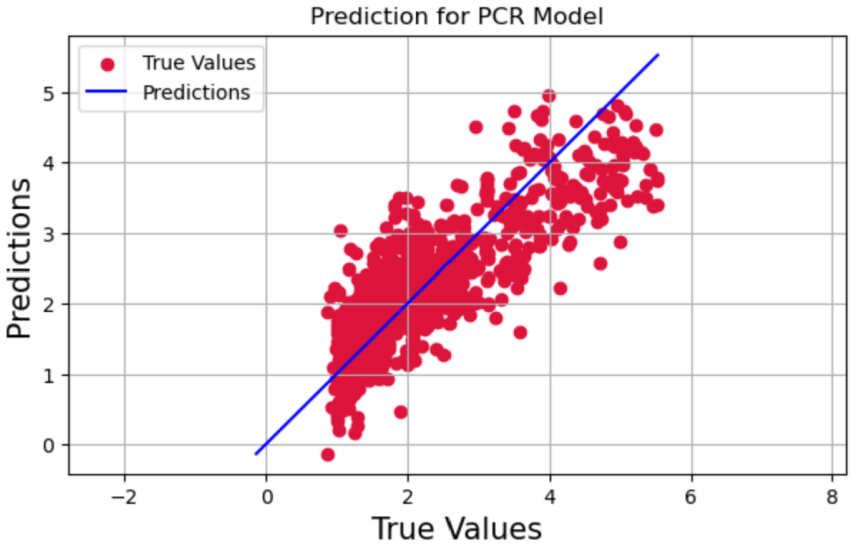



Fig 17: Plotting of PCR Model

## PLS Model

We used another model which was a PLS model. Again, we divided the dataset into two parts - training and testing. We run the model using five components and we got a MSE score of 0.41

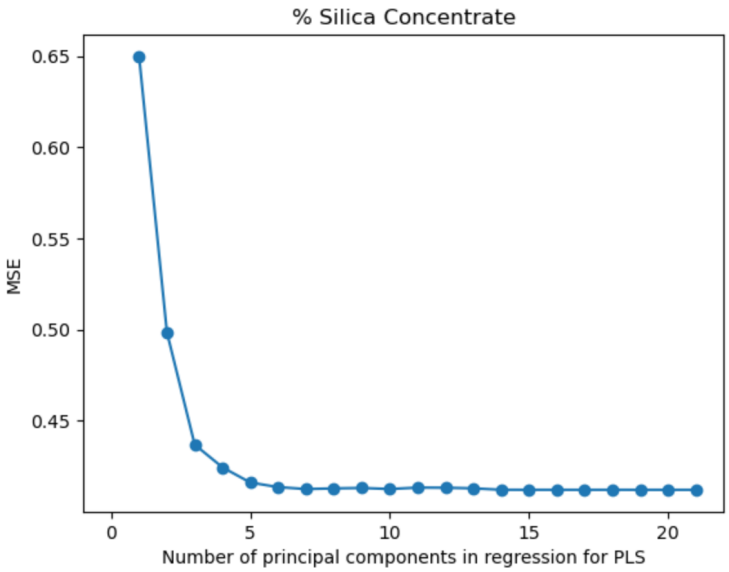



Fig 18: Number of PC in regression for PLS

```
PLSR = PLSRegression(n_components=22).fit(X_train_scaled , Y)

y_pred = PLSR.predict(X_test_scaled)

PLS_score_test = mean_squared_error(y_test, y_pred)

print('mean_squared_error','with PLS Methods',':', PLS_score_test)
```

```
mean_squared_error with PLS Methods : 0.4119085615098681
```

Fig 19: MSE Score of PLS Model

**For PLS, we can see from the importance bar-chart, 'Iron concentrate' is the most important** and for better model calibration. We will keep the only first five variables as the most important variables in our dataset for the PLS evaluation. These features are - % Iron Concentrate, % Silica Feed, Flotation Column 01 Air, % Iron Feed, Amina Flow.

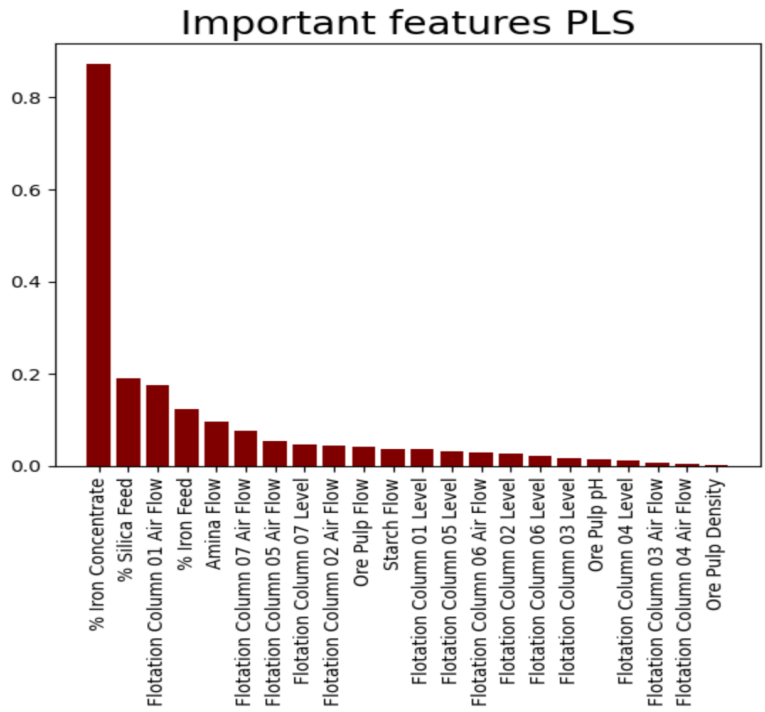



Fig 20: Important variables of PLS Model.

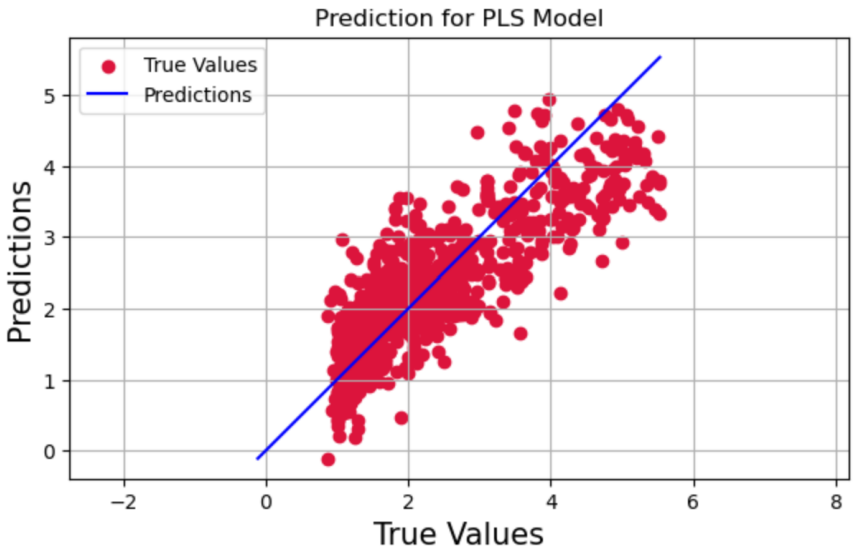



Fig 20: Plotting of PLS Model

Initially, we ran both PCR and PLS Model. From this observation, we can see that the PCR and PLS Model both have the same accuracy. However, both models use a different number of components. We can see the best result from PCR is with 12 components and for PLS with 5 components:

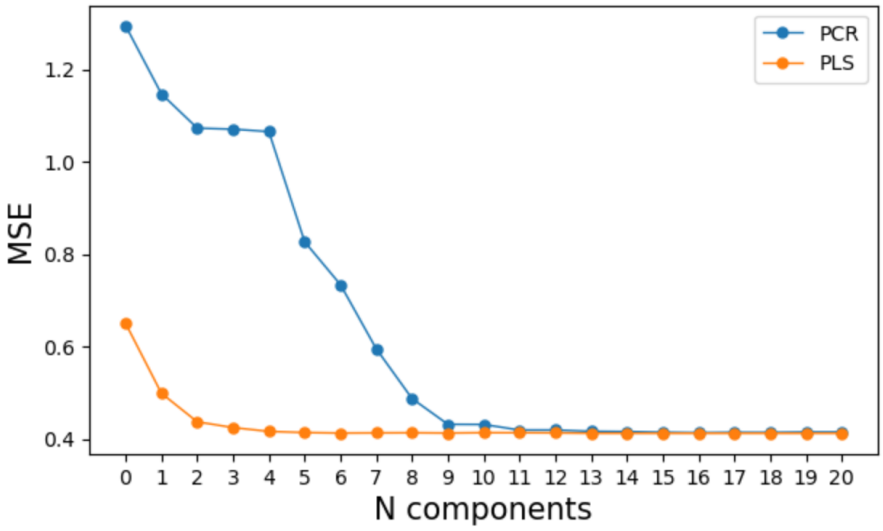



Fig 21: N- components of both model

## Variable Reduction

First, we will reduce 10 less important variables. We have already got into PCR, and then we will apply a linear regression model with this reduced from the initial dataset. Similarly, for the PLS model, we will only keep five important variables. We will remove 17 variables from the dataset, this number, and the variable name we have attained from the previous experiment.

The MSE value for PCR and PLS model can be shown as follows.

| **Model** | **MSE** |
|---|---|
| PCR | 1.14 |
| PLS | 0.45 |

After analyzing PCR and PLS, we find PLS works better even with only five components, whereas in a similar prediction, we get PCR with 12 components. Therefore we used the PLS model for minimum sampling frequency analysis and future prediction analysis.

| | Original label | predicted label by PCR | predicted label by PLS |
|---|---|---|---|
| **1299** | 2.09 | 2.288331 | 2.864514 |
| **1338** | 2.32 | 2.773141 | 1.914765 |
| **3875** | 2.14 | 2.752939 | 2.974309 |
| **2083** | 1.18 | 2.418110 | 2.221342 |
| **1498** | 1.90 | 1.645040 | 1.711748 |
| **...** | ... | ... | ... |
| **1767** | 1.29 | 1.319764 | 0.261347 |
| **3812** | 4.06 | 2.799984 | 3.977108 |
| **2629** | 1.41 | 2.030542 | 1.746782 |
| **1427** | 1.63 | 1.825720 | 1.919045 |
| **67** | 1.90 | 0.913847 | 0.580877 |

Fig 22: original vs predic label for both model

## Minimum Sampling Frequency

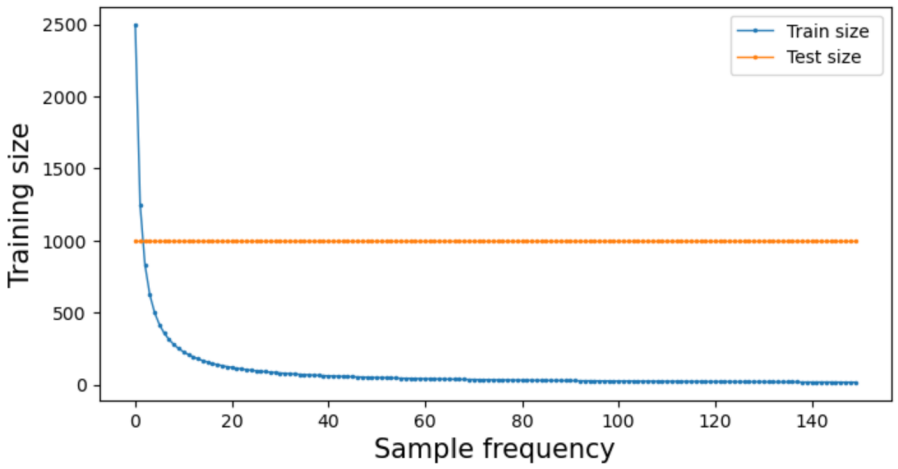



Fig 23: Train & Test datas size at different sample frequency for PLS Model

For minimum sampling frequency analysis, we collected a test sample size of 1000 from the end of the dataset; we kept this sample test data the same for all sample frequency tests so that we could compare our results. It is essential since if test data changes, there might be no ground truth for comparison. Now we sample train data at different frequencies (0 to 150). We start prediction with at 1-hour interval of time data. The graph shows that the model can predict at 25 sampling frequencies where MSE is still under 0.5. Therefore we can assume a sampling frequency of 25, that means we take a sample every 25 hours intervals, and the model actually does not affect much. Another observation is that we can see some good predictions at 140 samples as well, even in that case, the training sample is less than 500. This result is hard to describe since the model can predict even after 140 hours interval sample if there are enough train data.

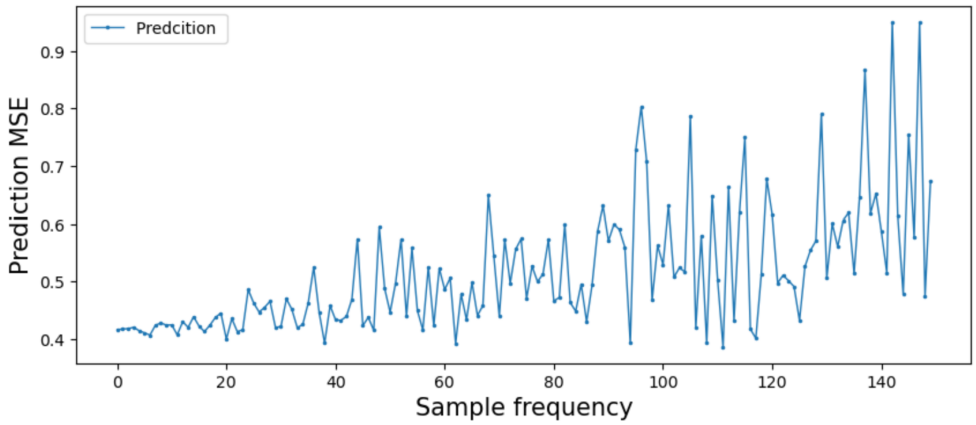



Fig 24:Minimum sample frequency of Prediction MSE

In that case, we could not determine any sharp cut-off point from the graph that we can say our model minimum sampling frequency must be this value; rather, for the current data, we can say **minimum sampling frequency approximate 25** would be a safe sampling value which will ensure MSE error below 0.5

## Lagged variables in Future prediction:

For time-lagged prediction, we tried two different approaches. One method is that, We split the dataset into small chunks for example- First, 100 data for train and second 100 data for test. In this way we fixed 2k test data same for all lag test experiments. Rest of the data collected for the test and we made 150 different lagged sample test sets, which start 1 hours to 150 hours. This first methods ensure we have taken the same proportion of train and test data from throughout the dataset.

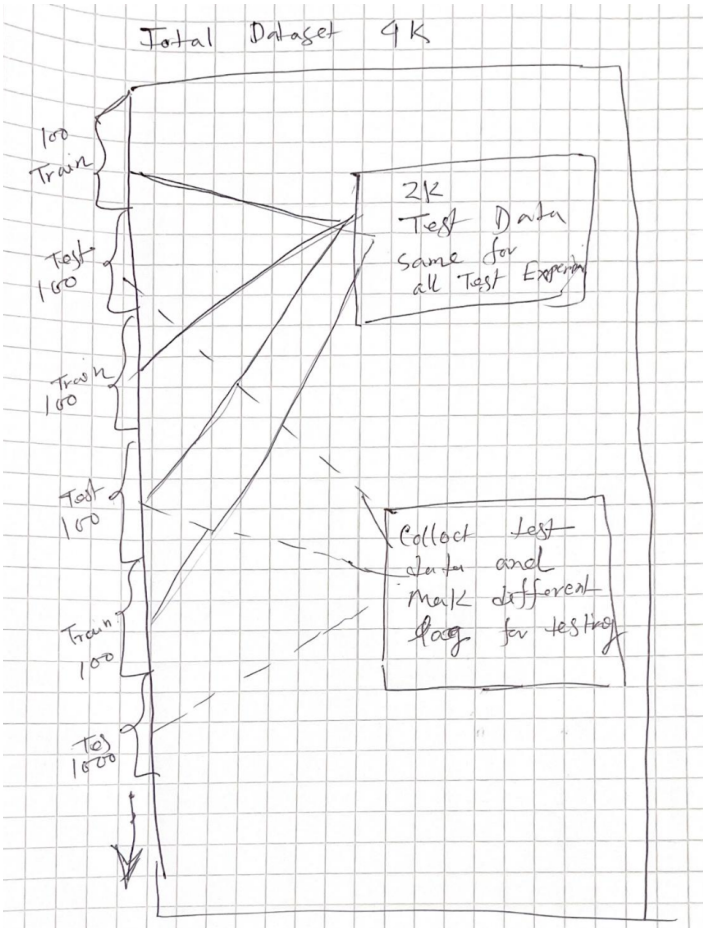



Figure 25: train and test data collection process of methods 1,

We start prediction with zero frequency which is a 1-hour interval of time data. We predicted test data at different frequencies (0 to 150).

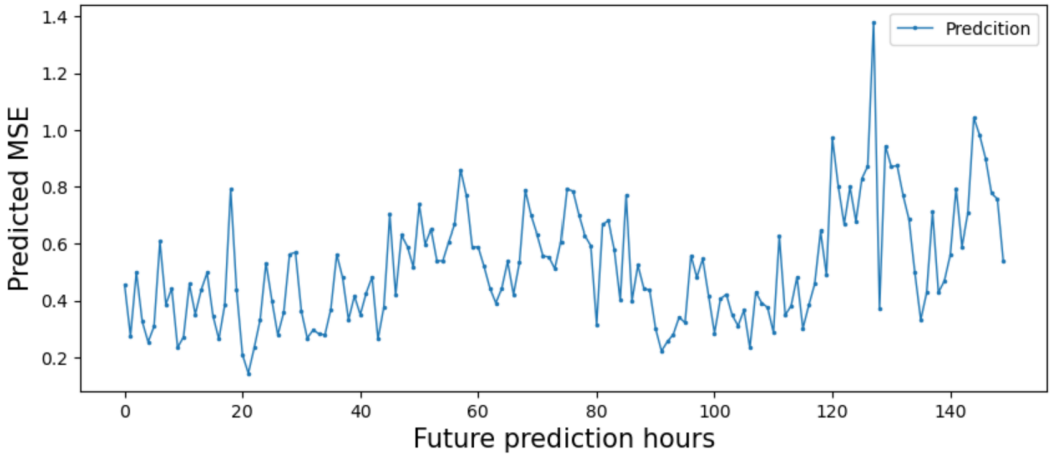



Figure 25: Result of methods 1, prediction of 140 hours future prediction

From methods 1, we can see that, majority of test data has MSE error is under 0.5 which suggests that our model prediction is not depending on time, in other words we might be able to predict reliably from any future test data. To justify our hypothesis, we did second approach,

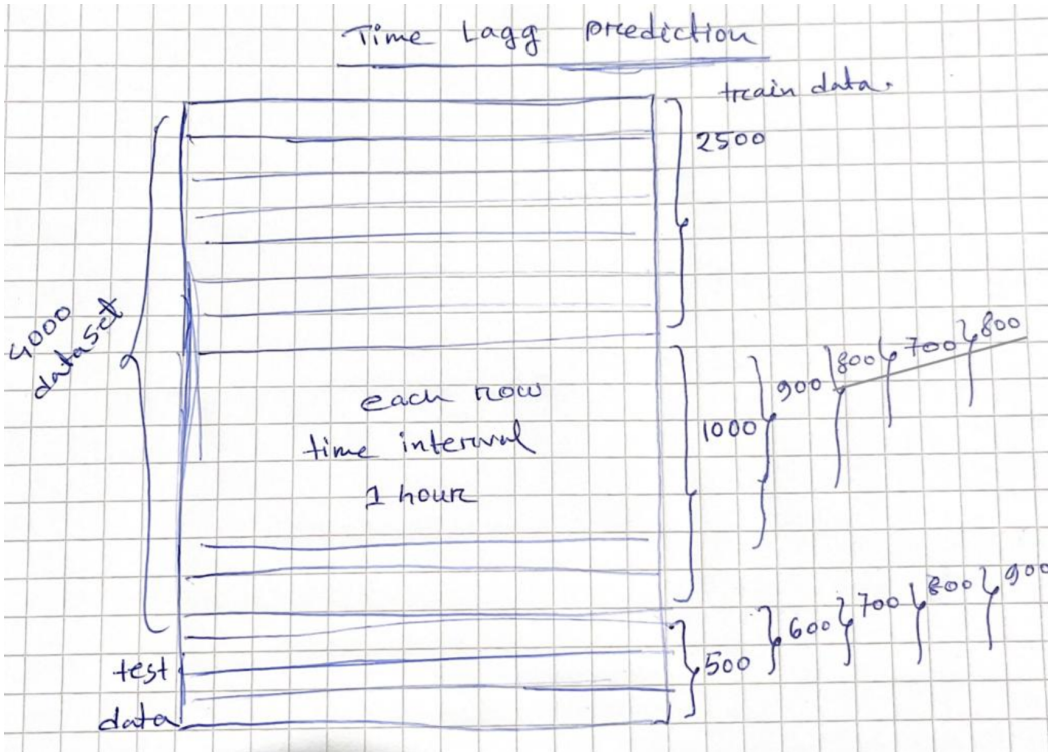
**Second approach,** We have approximately 4000 datasets. We have taken the first 2500 data as training data and made different test samples from the end of the dataset for testing.

Each row has (1) an hour time interval. We have made a total of 1000 test data that size varies from 500-1500 test samples. When the test sample is the last 500, and the training sample is the first 2500, the time difference is 1000 samples, which means we will predict 1000 hours ahead. When the training sample is first 2500, and the test sample is last 1500, then there would be zero time difference.

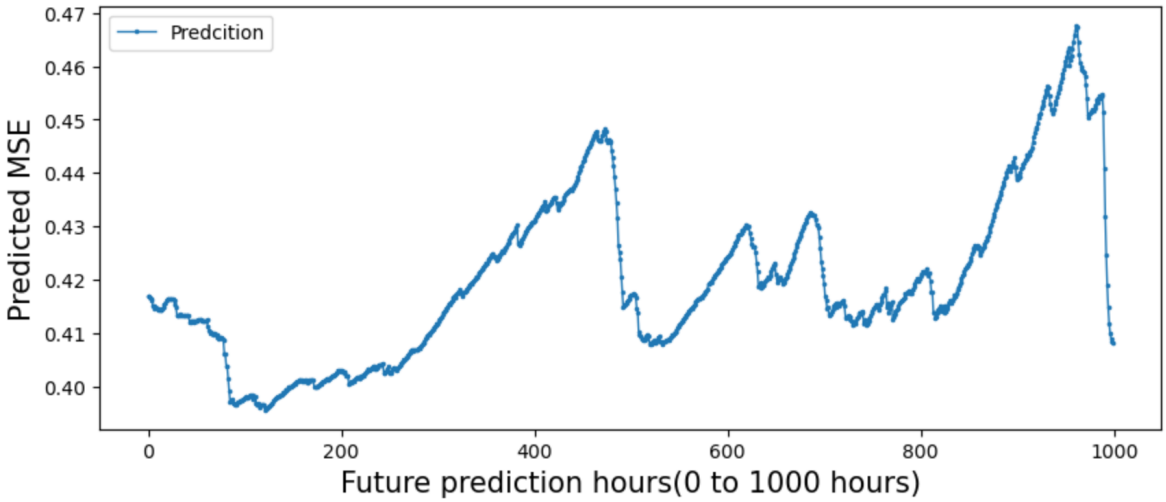
Train sample + Time difference + Test sample = 4000

| Train Sample<br>From first | Time difference or<br>Future prediction time | Test data size<br>From end | Total |
|---|---|---|---|
| 2500 | 1000 hours | 500 | 4000 |
| 2500 | 900 hours | 600 | 4000 |
| 2500 | 100 hours | 1400 | 4000 |
| 2500 | 1 hours | 1499 | 4000 |

Table: Lag time creation example for different size of train and test sample.




We start prediction with zero frequency which is a 1-hour interval of time data. We predicted test data at different frequencies upto 1000 hours.




From the result, we can see that our initial hypothesis has been proven, which is this dataset-trained model can be used for predicting a long time of future value. Due to dataset time

limitation, we were able to run up to 1000 hours of future sample prediction successfully without increasing MSE error. The model can confidently predict successfully from the results of 1000 hours of future prediction with approximately 0.43 MSE error, which is the result we received initially.

## Conclusion:

In this experiment, we predict **% Silica Concentrate** with mining datasets. We started with data preprocessing (missing value, outlier handling, sampling, scaling). After that, we chose two kinds of models: PCR and PLS. In both methods, we have determined the most significant variable that is required for the model. Therefore we have removed unnecessary variables from the dataset and applied them to PCR and PLS models. We compared and found PLS more suitable compared to PCR, which has a 0.40 MSE error compared to a 1.2 MSE error. After using the PLS model, we tried to determine the minimum sampling frequency, which is approximately 25. Finally, we experimented to determine the future prediction time, and in our case, we were able to successfully predict 1000 hours( 41 days) ahead of time.

**References**:

[1] – Dataset: Quality Prediction in a Mining Process [accessed online at: https://www.kaggle.com/datasets/edumagalhaes/quality-prediction-in-a-mining-process