

**IME672A**

**Data Mining and Knowledge Discovery**

Instructor: *Prof. Faiz Hamid*

Project report on

**Impurity Prediction in Mined Ore**

Submitted by

**Group 2**

Prashant K Sinha 18807542

Shyam Bahmani 180756

Harsh Raj 18807280

Prashant 180538

Desarda Rushabh 18807230

**Introduction**

A flotation plant is one of the various processes in ore mining. It is a very cumbersome process and with a slight variation in different input/variables, the quality of ore can be affected vastly. And fine tuning between different inputs is required in order to get the desired level of purity or % iron concentration.

Using the data available to us, we have developed a model which can predict the impurity level given the input variables, without using the % Iron Concentrate attribute.

**About the data**

We have a total of 24 attributes (dependent and independent). Time is a discrete value numeric attribute. The first column shows the date and time of recording the data. 2nd and 3rd columns show how much iron and silica are we feeding into the chamber respectively. Column 4 to column 8 have the important variables which impact the quality of the final iron ore. From column 9 to 22, the air flow and levels in the flotation columns are recorded. The last two columns are the output, iron concentration and impurity silica concentration.

% Silica concentrate is the dependent variable which we have to predict.

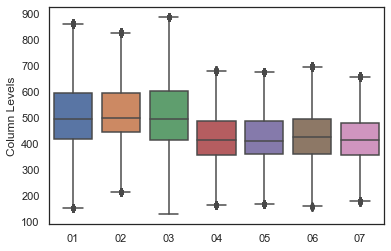
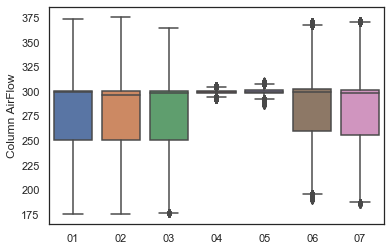
**Our approach in brief**

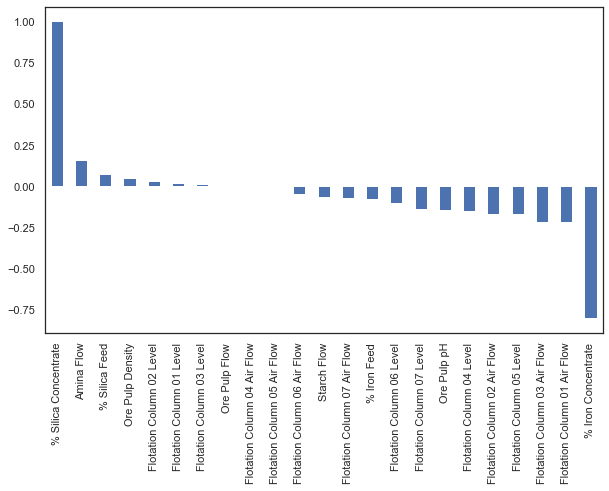
First, we will be analysing the data through visualization techniques and simultaneously checking for any discrepancies in the data and eliminating those anomalies. After doing pre-processing like principal component analysis, we have modelled using different ML algorithms to improve the accuracy to just under 90%. We found out that random forest regression gives the best result with the highest accuracy.

**Data visualization and cleaning**

Import the data, replacing comma(,) with decimal(.), parsing the first column as ‘date’. There are no null values, but 1171 tuples are duplicates to be dropped before further analysis. All the attributes are continuous numeric with data type as float64 except ‘date’ which is of datetime64[ns] type.

The features air flow column and level column have some outliers which are concentrated at the two extremes only so we decided to keep them as it is, as shown in the figure below.





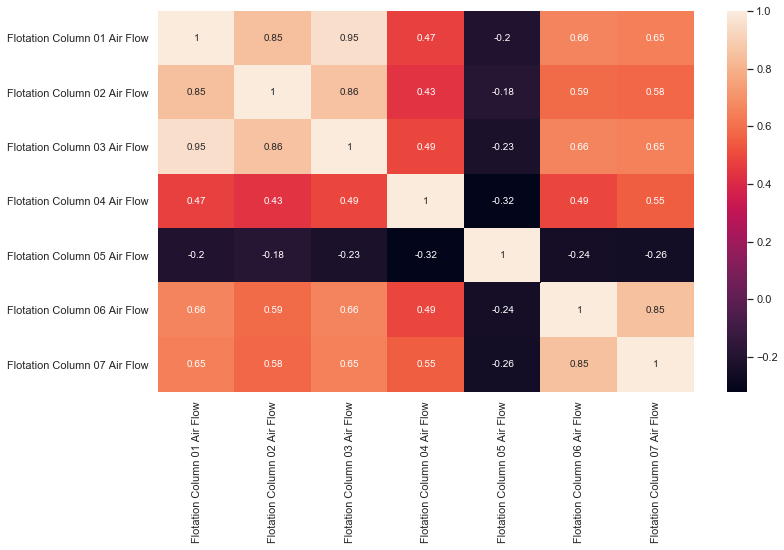
From the above correlation plot of % Silica Conc against other features, a high correlation is observed with % Iron Conc which basically represents the purity of the ore which is expected and will be dropped eventually before training the models. Others do not significant correlation but the important features are Alumina Flow, Level-5, and Air Flow-1-3.

Apart from intercorrelation between different features, not much can be inferred from the heatmap and the dependent variable does not depend on any single variable but all the independents combine to predict the % silica concentrate.

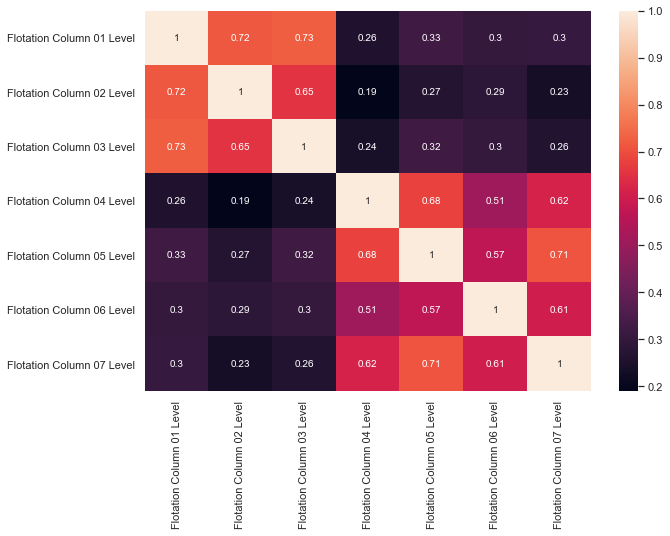
**Data pre-processing**

In this, we have reduced the dimension from 24 to 16 using principal component analysis (PCA).

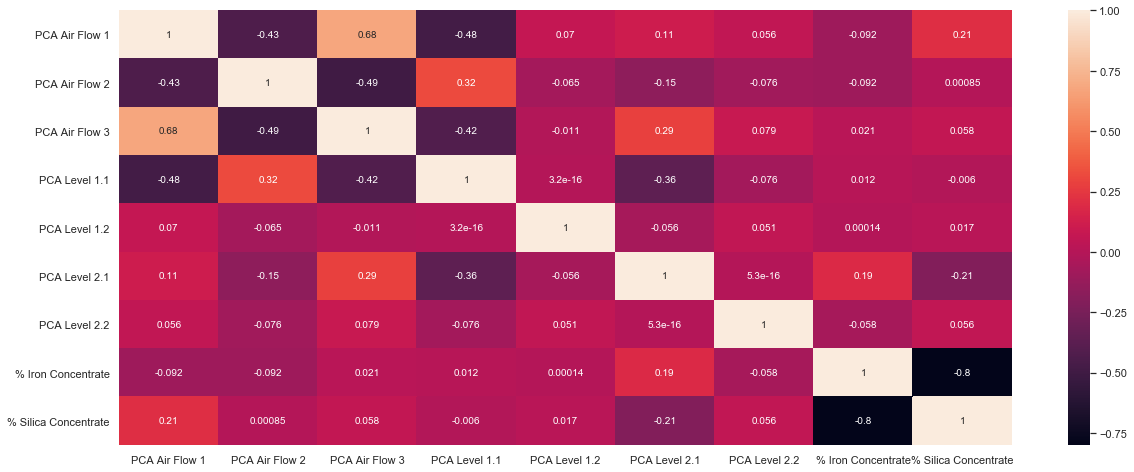
From the heatmap of ‘Air Flow’ intercorrelation between Flotation Column (FC) (1, 2 & 3), and (6 & 7) can be observed. According to correlations, we have clubbed attributes with similar patterns, column 01-02-03, 04-05, and 06-07. The data is first normalized and then used PCA library to fit the data. When n\_components is taken as 1, a variance of 92.7% is observed which is acceptable.



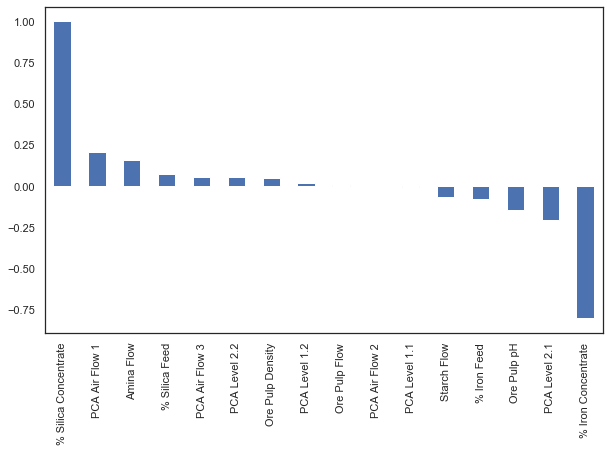
From the heatmap of ‘Level’ attributes FC (01-02-03) and (04-05-06-07) can be clubbed respectively. If we take n\_components as 1, the variance is 71.3% which is further improved by considering n\_components = 2.



The below heatmap shows correlations between different variables after PCA analysis.



Before training the data let's check if we can drop any attribute which is not significant. The correlation of ‘% Silica Concentrate’ is plotted against different variables below. Its correlation with ‘Ore Pulp Flow’, PCA Air Flow 2’ and PCA Level 1.1’ is almost zero so we dropped these features. Also, as mentioned before ‘% Iron Concentrate’ is an output of the experiment so we do not know its content in the ore beforehand, so we dropped this feature.



**Model training and evaluation**

Now the data is ready for training and building a model. After processing the data, we have 11 independent variables and 1 dependent variable (% Silica Concentrate) which is to be predicted.

For all the models we have stratified sampled the data in **7:3**, 70% being the training set percentage.

**Note:** From here onward accuracy, score and R2 represent the same thing and is the measure of the model accuracy.

**Linear regression**

The model is inefficient as the score is very bad on both training and test sets. The model is *underfitting.*

* Accuracy on training set = 15.10%
* Accuracy on test set = 15.01%

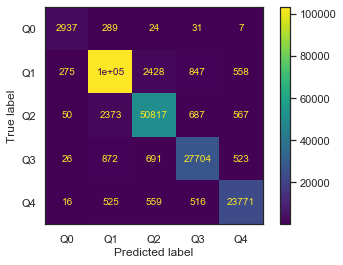
**Polynomial regression**

Regression with degree=2 does not indicate much improvement. Still the model is *underfitting*.

* Accuracy on training set = 21.27%
* Accuracy on test set = 21.17%

**Decision Tree Classifier**

Here we took a different approach where instead of predicting the impurity we have classified them into the range in which they are most likely to belong.

We categorize the ‘% Silica Concentrate’ values into 5 classes (0, 1, 2, 3, 4), 0 being the best quality ore while 4 is the worst quality with high impurity. 

The DecisionTreeClassifier model is able to predict the class with high accuracy, as shown in the confusion matrix.

* Accuracy on training set = 99.98%
* Accuracy on test set = 94.62%

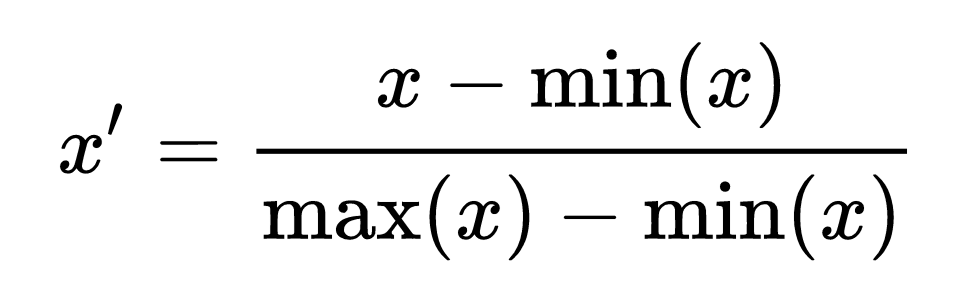
**Artificial Neural Networks**

The Artificial Neural Network is one of the models which we have built that has been statistically and practically accurate (with an R2 score of 0.857).

As seen before in the correlation table, the Silica Concentrate Percentage is not significantly correlated with any attribute in the data, thus it is possible and probable that the dataset is quite non-linear.

Therefore for our Artificial Neural Network, we kept all attributes for building and training the model so that we don’t leave out on assimilating any useful non-linear patterns in the data.

To make the data processing more efficient for the ANN, we pre-process the data by normalizing it with min-max normalization by using the following formula for each data object with respect to its attribute column.



After which we do random sampling of the dataset such that the sample contains 70% of the original dataset.

Then we split the data into training and test set with a fraction of 0.3 for test set. Thus the test set contains (70% \* 30% = ) 21% of the original dataset, and we train the model using (70% \* 70% =) 49% of the original dataset.

After which, we used the MLPRegressor model from sklearn package to build the neural network with the following parameters. We decided on the no. of hidden layers and neurons by trying out various combinations ( (64,64), (128,12,128,128), (10,10,10,10,10) etc.) and chose the most optimal and accurate one.

Hidden Layers = 4

Neurons in hidden layers = (256, 256, 256, 256)

Activation Function for each layer = ReLU

Learning Rate = ‘adaptive’

(keeps the learning rate constant to ‘learning\_rate\_init’ as long as training loss keeps decreasing. Each time two consecutive epochs fail to decrease training loss by at least tol, or fail to increase validation score by at least tol if ‘early\_stopping’ is on, the current learning rate is divided by 5.)

Batch size = min (200, no\_of\_samples)

No. of epochs = 35

Backpropagation/Weight Optimization algorithm used = ‘adam’

( ‘adam’ refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba)

We successfully built the model. The following are the model performance results on the testset:

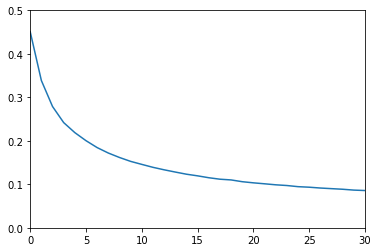
Mean Squared Error: 0.1812

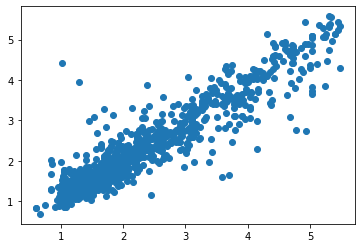
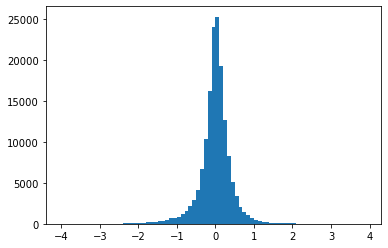
R2 Score: 0.8571

Mean Absolute Error: 0.2767

Mean Squared Error: 0.1811

Root Mean Squared Error: 0.4256

The following is a graph plotting loss per iteration on the y-axis and number of iterations on the x-axis

From scatter plots of Actual Test Values vs Predicted Values. You can tell how well the model is performing. For an ideal model, the points should be closer to a diagonal line. Here we plot the first 1000 points from test and predicted values. And residual histogram (predicted value - test value) to check for any bias towards positive or negative skew, which we don’t have.

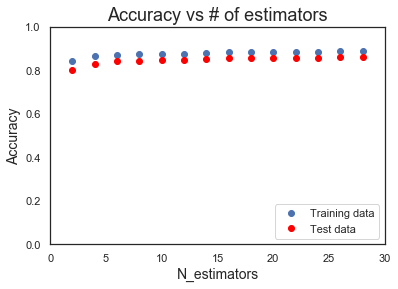
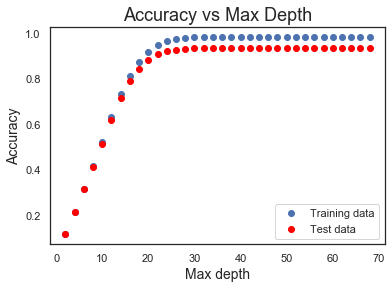
**Random Forest**

As we have seen even the ANN model has low accuracy. So to improve the accuracy, we have applied the ensemble method ‘Random Forest’.

With (n\_estimators=10, max\_depth=none), the model predict the test set with 93.98% accuracy while training set with 98.76% accuracy. This seems an overfit as the accuracy gap is high.

After fixing n\_estimators=7, maximum accuracy of 93.53% is obtained with depth=29. However, there is inconsistency in the accuracy of train and test data. So the optimal model at the accuracy of 84% where the model is able to predict train and test set with similar accuracy or the gap is low.

If we fix max\_depth=18, a model with n\_estimators=17 has high consistency with the train and test set and predicts train and test with similar accuracy.



**Results**

* Regression with degree 1 and 2 does not perform well on the test set and are under-fitted.
* The decision tree classifies the quality of the ore into 5 classes with an accuracy of 94%.
* ANN with ReLU function and 4 hidden layers of 256 neurons each gives an Mean Squared Error of 0.1812 and an R2 Score of 0.8571.
* The random forest model gives an accuracy of 85% with both training and test set, which can be considered a very good model as it is consistent with all seen and unseen data.
* Future Directions : Using GPU acceleration and tensorflow for faster, more efficient data processing and modeling.