Data analysis summary from analysis\_report.docx

1. The dataframe did not have any null values
2. 385 of the 498 feature columns were deemed non-normal by the shapiro test. Hence, IQR seemed to be a good choice to determine outliers instead of z-score.
3. Interestingly though, the top 10 correlated features with target were all normal. However, the correlation itself was quite less <= 0.1, implying that the features aren’t linearly related to the target variable.
4. There was high collinearity amongst the features. Hence PCA seemed a good option to keep only the top 20 percent features for model training.
5. All feature columns had less than 4% outliers
6. The target column had 16% outliers and a skewed tail. Hence, clipping its range using IQR seemed valid for model accuracy.
7. After treating the data with outliers, a new analysis report was generated and saved in transformed\_df\_analysis.csv

Training a simple baseline regression model

1. The data was split train.csv and test.csv, with a 80/20 split.
2. RegressionNN defined in src/model/model.py was trained on train.csv with 5 fold cross-validation. The parameters are defined in configs/conf.yaml
3. As the features did not show linear collinearity with the target feature, a three layered neural network with Relu activations was chosen to capture non-linear patterns.
4. Top 20% which is 89 features were used to train, rest were dropped.
5. Evaluation was run on the trained model and results are stored in baseline\_report.docx
6. All metrics suggested that the model was undertrained both on test and train set. This could likely be because of lack of training data, as the initial data only had 500 rows, of which 20 % was used for evaluation.
7. Since MSE was high and R square < 0.15 on both train and eval sets, the model was likely guessing around mean. Since, the model was fairly simple with
8. The MIR scores were also split quite evenly across the features, with the top feature having a score of ~0.12. This along with poor predictions also suggested a likely lack of predictive power amongst features.

Training a transformer

1. A transformer network was added in src/model/model.py and evaluation report is in transormer\_report.docx.
2. The transformer performed even poorly than the baseline model, likely because of its complexity combined with the lack of data/poor predictive power.

Improvements

1. Better data treatment. While neural networks can theoretically approximate any function, transforming features to a space using kernel-pca or other methods which explain more relation between them and the target might have helped the training process.
2. Training simpler tree based models which capture nonlinear relations might have produced better results.