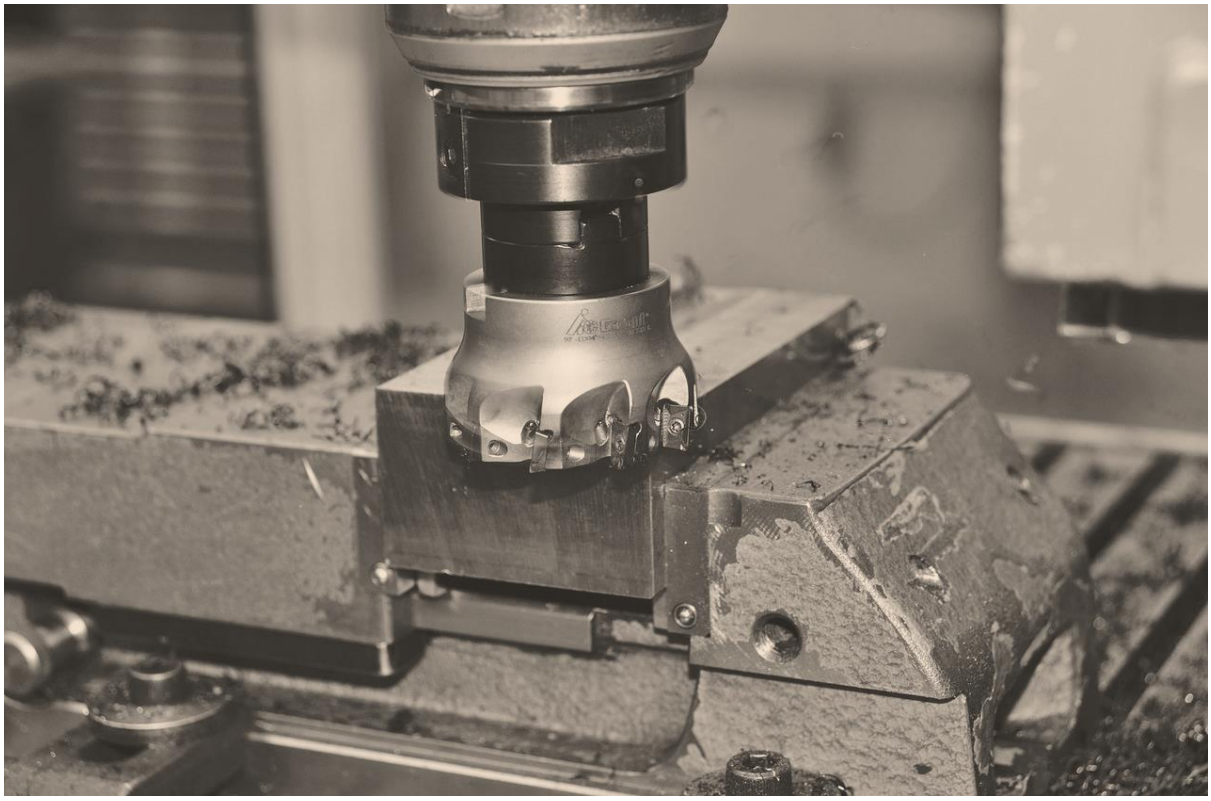

Milling Equipment:

Predictive Maintenance & Asset Life Optimisation

Author: Dr Kieran Jervis

Portfolio: www.chemdigital.co.uk

Email: kieran@chemdigital.co.uk

Table of Figures

Table of Tables

Contents

1. The Problem Statement

A Fabrication company operates a diamond-tipped milling machine, where failures in the milling process can occur via five primary mechanisms. When a failure happens, it can cause damage to the current workpiece within the machine, resulting in both material waste and additional loss due to downtime to reset and replace damaged materials and tooling.

The goal of this study is to predict the need for maintenance ahead of operation, specifically in the form of tool tip replacements, to avoid these losses. Currently, tool tips are replaced at random intervals between 200-240 minutes of accumulated usage, which may not align with actual wear and tear or failure risk. Instead, we propose the use of a supervised offline classification model to predict the onset of failure. Using historical environmental and operational data from previous run instances, along with data on prior failures if occurred, the aim of the model will be to predict whether the tool tip will fail under upcoming operational conditions.

Because the material loss and downtime from an unexpected tool tip failure are more costly than the tool tip, our primary success metric will be to focus on maximising the detection of likely failures, with scheduled tool tip replacement we see **3.49%** of projects experiencing an interruption due to unexpected tool tip failure or scheduled replacement. A secondary goal is to minimise false failure predictions to reduce unnecessary replacements, as we have no measure of false failure predictions from this dataset to improve upon we will instead aim to train a model to make false failure predictions at or less than the tool tip replacement rate, determined from the raw dataset to be **0.4%** of instances.

2. Data Description

This report in-part presents a detailed analysis of the dataset structured for predictive maintenance applications, aiming to explore machine learning models' capabilities in predicting failures across various process and quality parameters. Key attributes and descriptions of the dataset:

- **(1) Unique Identifier (UDI)**: Sequential ID for each entry, ranging from 1 to 10,000.
- **(2) Product ID and (3) Type**: Each product is labelled as low (L), medium (M), or high (H) quality. Low quality (50%) is most frequent, followed by medium (30%) and high (20%), each accompanied by a variant-specific serial number.
- **Environmental and Process Conditions**:
    - **(4) Air Temperature**
    - **(5) Process Temperature**
    - **(6) Rotational Speed and (7) Torque**
- **(8) Tool Wear and Durability**: Based on quality type (H, M, L), tool wear time adds 5, 3, or 2 minutes respectively, representing the tool's effective usage time in the production process.
- **(9) Machine Failure Label**: This binary indicator is triggered by any of five failure modes: tool wear failure, heat dissipation failure, power failure, overstrain, or random failure. Each failure mode has specific parameters:

- **(10) Tool Wear Failure (TWF)**: Randomly replaced tooltip between 200-240 minutes of tool use, with 69 instances of replacement.
- **(11) Heat Dissipation Failure (HDF)**: Triggered if the difference between air and process temperature is below 8.6 K and rotational speed is under 1380 rpm, with 115 recorded instances.
- **(12) Power Failure (PWF)**: Defined by torque and rotational speed producing power outside the range 3500-9000 W, observed 95 times.
- **(13) Overstrain Failure (OSF)**: Caused if the product of tool wear and torque exceeds limits set by product type, totalling 98 instances.
- **(14) Random Failures (RNF)**: A 0.1% chance of failure per data point, with 5 cases recorded.

This dataset is derived from the following publication: Matzka, S. (2020). "Explainable Artificial Intelligence for Predictive Maintenance Applications," *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*, pp. 69-74.

2.1. Exclusion of Unpredictable Outcomes

Random failures (RNF) and scheduled tool changes (TWF) are considered inherently unpredictable and thus are excluded from the predictive modelling goals. These events do not correlate with observable process conditions and therefore cannot contribute meaningfully to model training or predictive accuracy in maintenance applications but rather have the potential to hinder training.

2.2. Assumptions

Each row of data represents a single milling project state space and failure outcome, with no temporal sequencing between instances.

3. Data-set Pre-Processing

The dataset is modified to consolidate individual failure columns under a unified "Machine Failure" column, with specific integer values assigned to each type of failure – Label Encoded with the respective column names.

As indicated in the prior section, the TWF and RNF positive rows are removed from the data set reducing the dataset column length from 14 to 12. Additionally, rows indicating multiple failure modes are removed due to the intractability of assigning a single root cause of failure.

Furthermore, eliminating rows associated with random or scheduled failure modes (RNF & TWF respectively) results in an alteration in the quantitative business metrics as the total number of instances are reduced from 10,000 to 9,916. Our business objectives are altered due to this. We will maintain our goal to improve and meet the false failure rate to **<=0.4%.** The primary goal to *reduce occurrence of failures by prediction driven proactive maintenance*; with the elimination of TWF and RNF failure modes eliminated is recalculated at **<2.70%** of instances.

4. Train & Test sets

Before conducting exploratory data analysis (EDA), the dataset is split into a target variable stratified 80:20 ratio. The larger portion (80%) is used for EDA, model training, and validation, while the remaining 20% is reserved as an isolated test set for the final model. This isolated data will represent unseen real-world data.

The training set is formed of 7,932 rows. The train and test sets are stratified across class outcomes. This is reproducible and deterministic, as we used a fixed random_state value of 38513497 (ChemDig) in the train_test_split function, allowing for reruns of the code without concern for data leakage between the test set and training set.

This methodology ensures that the evaluation of the final tuned model on the test set is unbiased, as neither the machine learning models, nor the analyst has prior exposure to this data subset. This provides a realistic assessment of model performance on unseen data.

5. Exploratory Data Analysis

5.1. Assessment of Raw Data

Following the consolidation of failure columns to the "Machine failure" column, the failure-mode specific Machine failure label columns are temporarily (for EDA) dropped from the database. Further to this the "UDI" column is removed as it serves no contribution to the data patterns, it is an indexing column. Reducing the column count from 12 to 11.

The data set is complete, with no missing values or duplicate rows in any of the 11 columns, Figure 1 shows the dataset prior the machine failure column addition and UDI removal.

```
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   UDI                     7932 non-null   int64
 1   Product ID              7932 non-null   object
 2   Type                    7932 non-null   object
 3   Air temperature [K]     7932 non-null   float64
 4   Process temperature [K] 7932 non-null   float64
 5   Rotational speed [rpm]  7932 non-null   int64
 6   Torque [Nm]             7932 non-null   float64
 7   Tool wear [min]         7932 non-null   int64
 8   HDF                     7932 non-null   int64
 9   PWF                     7932 non-null   int64
 10  OSF                     7932 non-null   int64
```

Figure 1 - Dataset information pre- label encoding of target variables – No missing data

Before conducting data analysis, it would be beneficial to transform the object data type columns into numeric format.

As the 'Type' column contains the strings 'L', 'M', and 'H', which refer to a hierarchical system, we can confidently map these values to [0, 1, 2] respectively.

The 'Product ID' column uses a regex pattern where the first letter represents the quality of the tool, which is synonymous with the 'Type' column data. This is followed by a serial number for the product. We will drop the letter, as it is already captured in the 'Type' column, and retain the serial number, as it may prove useful for prediction.

The data set target columns are unbalanced with only **2.7% of the data reflecting failures**, this is as expected in industrial data, we will not typically observe a successful company with substantial failure data.

Table 1 - Unbalanced dataset, failure outcome targets variable distribution.

Machine failure	**Count**	**Contribution%**
No Failure	7721	97.34
HDF	85	1.07
PWF	64	0.81
OSF	62	0.78

5.2. Outlier handling

Failure data is scarce; to retain as much of the failure data as possible, we will opt for a less severe outlier test—Z-Score analysis. Although IQR was performed, it highlighted many more points related to failure instances at the boundaries of the data distributions.

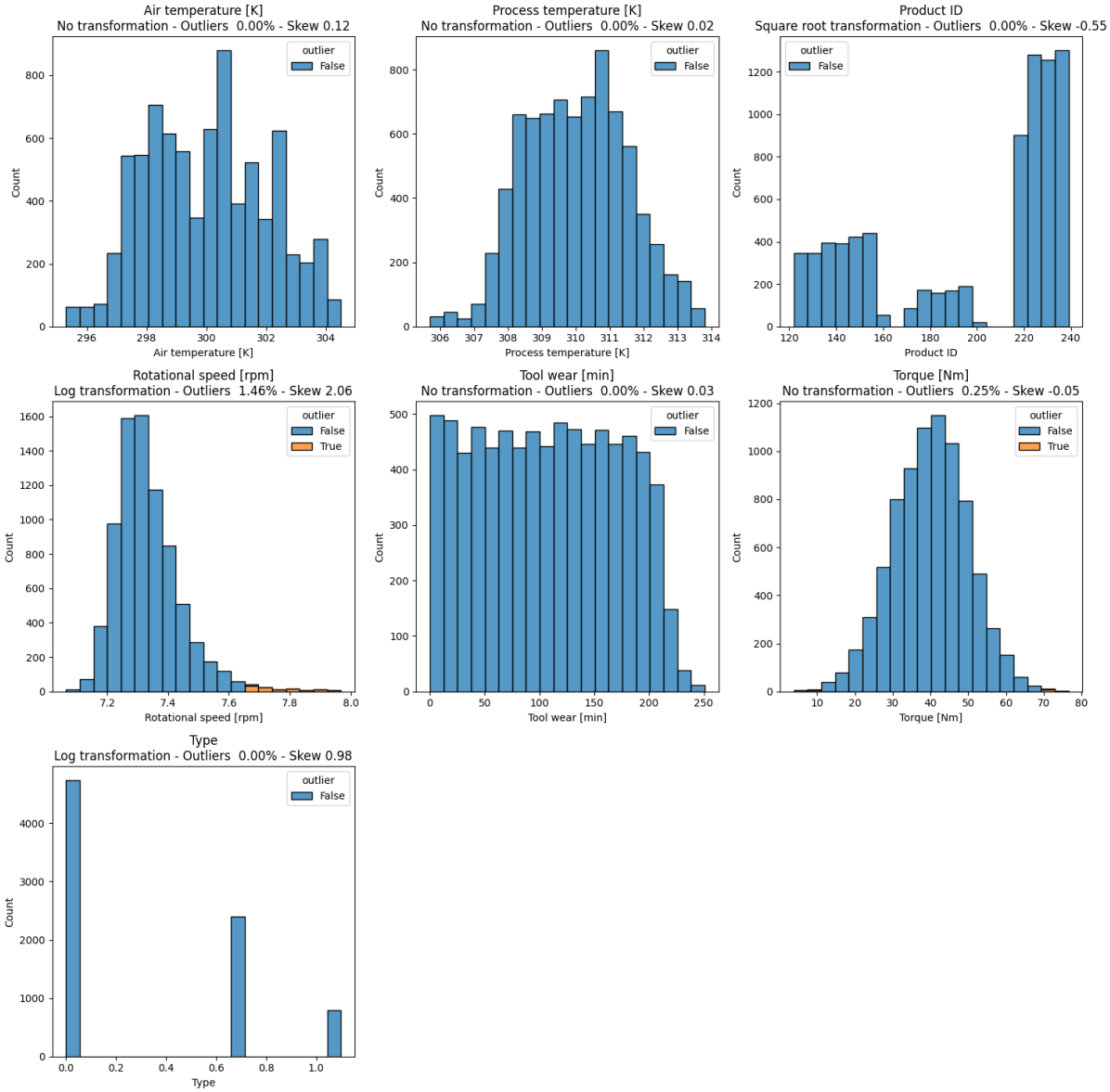

Figure 2 - Outlier detection, Z-Score analysis, with titled transformations applied. Outliers in both Rotational speed & Torque distributions indicated by orange colouring.

The plot above shows the outliers in the feature space data, where the threshold for outlier classification is a modulus Z-score value of 3 or greater. Of these, 89 are non-failure, and 37 are machine failure PWF; 55% of PWF failures are identified as outlier data. The outliers for these rows are found under the Rotational Speed and Torque columns. To preserve these limited instances, the outlier data will be clipped to the column's minimum, mean, or maximum value, depending on which value the outlier is closest to.

The plot titles include the transformations applied to each distribution to reduce the skew and normalise the data for accurate Z-score analysis. These transformations are retained in the dataset, as the reduction of skew normalises the distributions, which is required for many of the algorithms applied going forward.

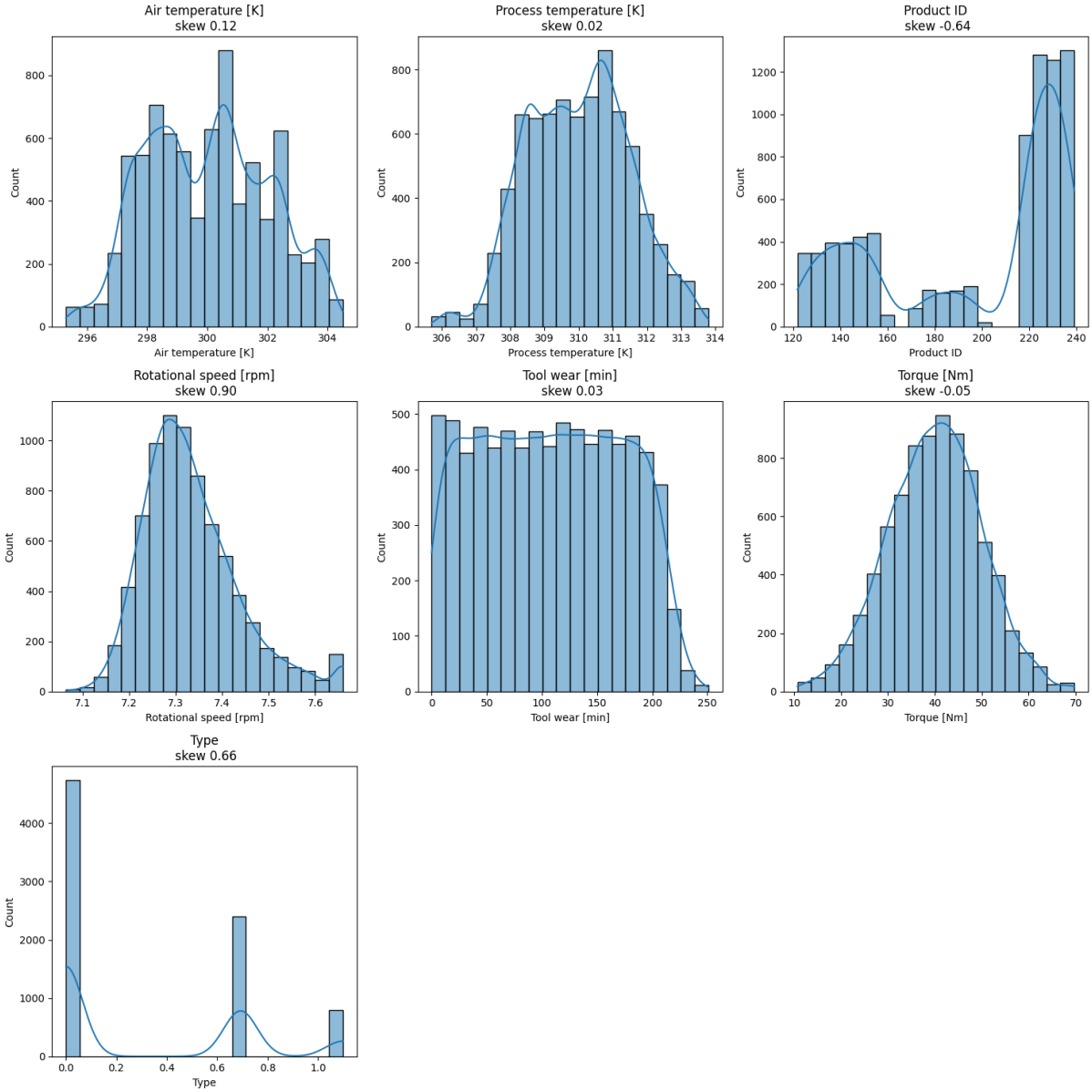

Figure 3 - Outliers clipped to nearest min, max, mean value of column. And skew reduced to acceptable levels.

The application of transformations has improved the skew of the plots, most dramatically with the rotational speed. This is a combination of the transformation and the clipping of outliers—the latter can be seen in the far-right bin of the rotational speed plot, which now contains many of the original outliers that were previously in the training tail. Similarly, the torque outliers have been clipped to their nearest min/max/mean, flattening the histogram. The following description of the data provides the distributions of the data post outlier handling and transformations; all features fall within the expected ranges.

	Product ID	Type	Air temperature [K]	Process temperature [K]	\
5645	217.892175	0.0	298.0	308.5	
7926	221.972971	0.0	299.4	309.2	
4635	230.449561	0.0	300.2	310.1	
5547	238.849325	0.0	298.9	309.9	
7572	227.953943	0.0	303.7	312.5	

	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	\
5645	7.446585	25.3	124.0	
7926	7.275865	48.2	24.0	
4635	7.408531	30.5	43.0	
5547	7.372118	31.3	98.0	
7572	7.403670	36.0	125.0	

Figure 4 - Sample of feature space data post-outlier removal.

5.3. Feature relationships

With the data processed for missing values, outliers and skew we will now explore the multivariate relationships. The following pair plot present the features with failure mode colouring (see key), non-failure instances are removed as the number of datapoints obfuscate the minority failures underlying pattern.

Notable patterns identified:

- **Clustering of failure modes:** There is clear clustering with well-defined boundaries between failure modes throughout the feature space, which is a positive sign for successful classification. This clustering is particularly noticeable in the Tool Wear and Torque rows of the pair plot.
- **Correlation between temperatures:** Process temperature and air temperature are correlated, which could introduce multicollinearity and negatively affect the accuracy or performance of algorithms. It may be advisable to remove one of the two correlated features or combine them into a new feature. When plotted against each other, the two features clearly separate HDF failures from the other two failure modes.
- **Product ID and failure mode correlation:** There appears to be a correlation between higher Product ID values and an increase in OSF failures, which also correlates with products of type L (converted to value 0 in the earlier data manipulation).

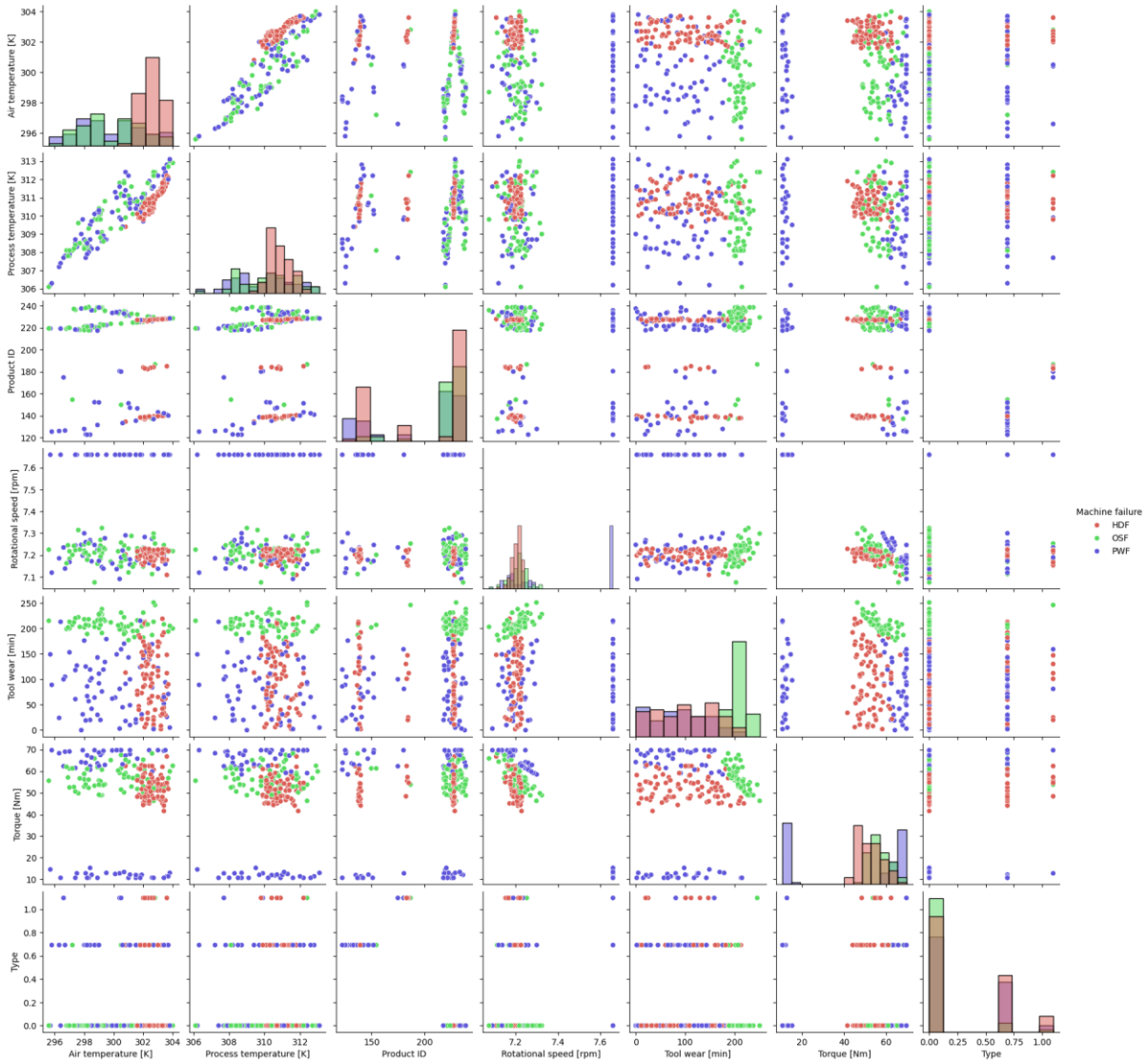

Figure 5 - Pair plot of feature space interaction, demonstrating clustering nature of future space with target failure highlighting.

The heatmap below shows the bivariate Pearson correlation of the features, confirming patterns observed in the pair plot.

- **Type and Product ID correlation:** Unsurprisingly, Type and Product ID are highly correlated, a score of 0.81. Product ID is a numeric serial number that is variant (Type)-specific. The key difference is that Type is categorical, while Product ID is continuous data with a greater range, potentially capturing more nuanced relationships between the exact product and the failure modes.
- **Process and air temperature correlation:** Process and air temperature are highly correlated with a score of 0.88.
- **Torque and rotational speed correlation:** Torque and rotational speed are also highly correlated, with a score of 0.91.
- **Non-correlation of remaining features:** The remaining features are effectively non-correlated, with the maximum correlation score being that of 0.062 between Product ID and Process Temperature. The low feature-feature correlations are ideal data for proceeding with classification model training.

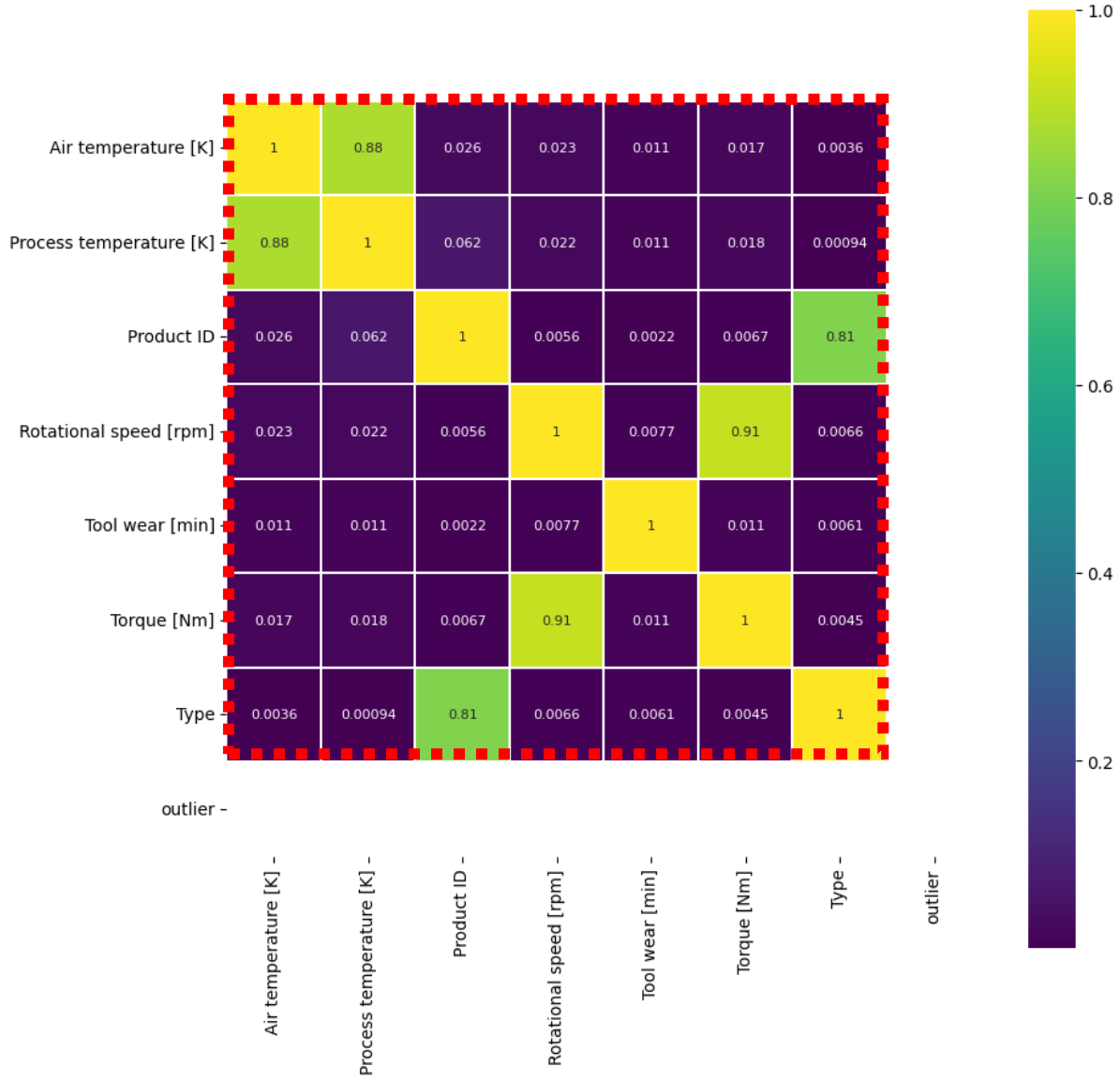

Figure 6 - Feature space correlation matrix – multicollinearity indicated by values closer to 1.

5.4. Feature Engineering

There is opportunity to combine the highly correlated components identified in the heat map via domain knowledge feature engineering.

Physical relationships:

$$Specific\ Torque = \frac{Torque(Nm)}{Rotational\ Speed\ (rpm)}$$

$$Power = Torque(Nm) * Rotational\ Speed\ (rpm)$$

Thermal features

$$\Delta T = Process\ Temp.\ (K) - Air\ Temp.\ (K)$$

$$Normalised\ Process\ Temp = \frac{Process\ Temp.(K)}{Air\ Temp.(K)}$$

The heatmap below shows the Pearson correlations, including the domain-engineered features. As expected, there is a correlation between the foundational features and those derived. The red dotted lined box indicates the boundary between original features and new feature correlations.

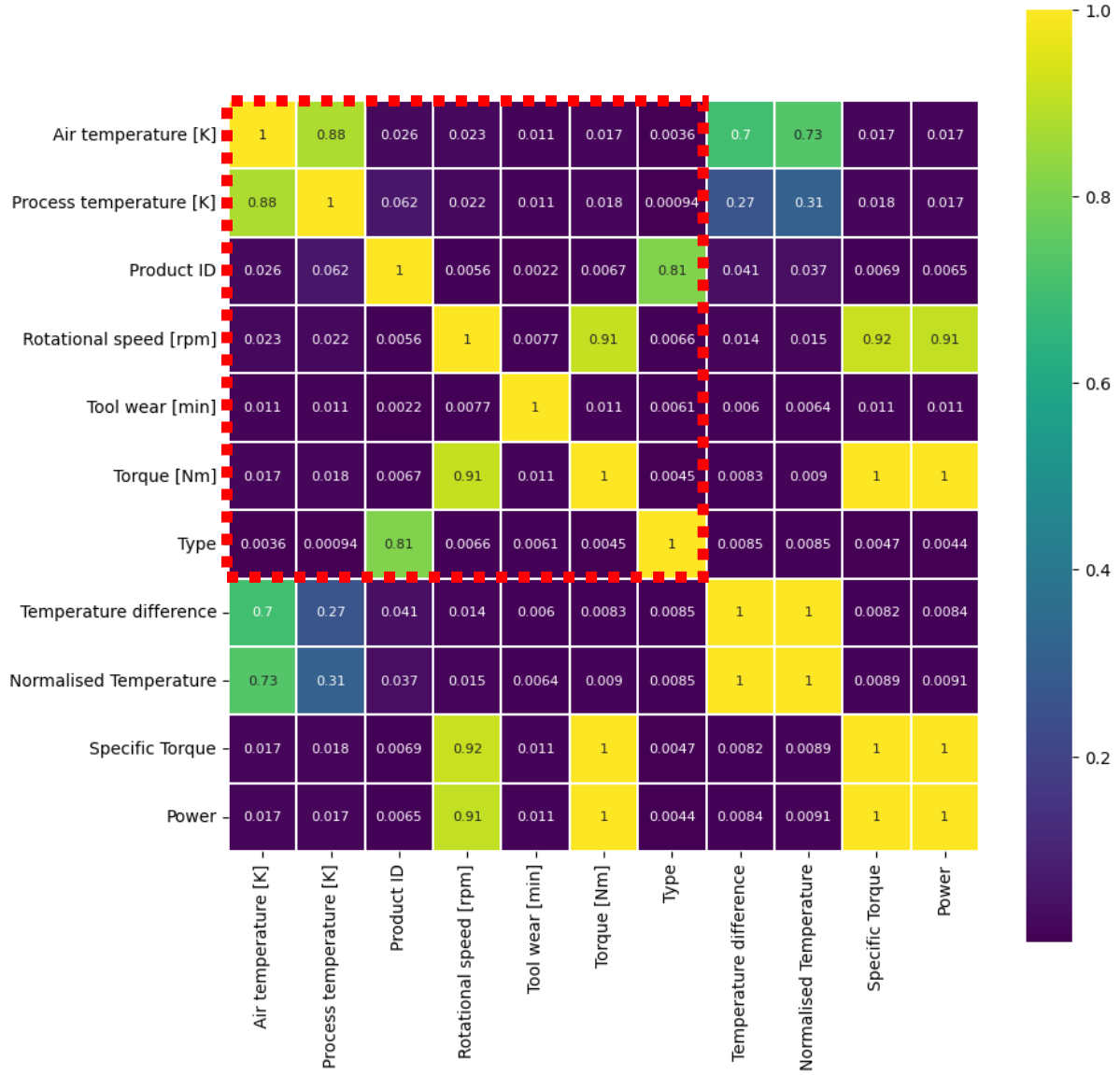

Figure 7- Domain expansion feature space correlation matrix– multicollinearity indicated by values closer to 1.

Mutual information— a feature importance test—treats outcomes as nominal categories. Here, we employ this test to evaluate which features provide the most information gain in predicting the outcome. Due to the unbalanced data, with the majority being non-failure, non-failure instances have been removed from evaluation, like with the pair plots above to provide a clear indication of failure modes in relation to the feature space.

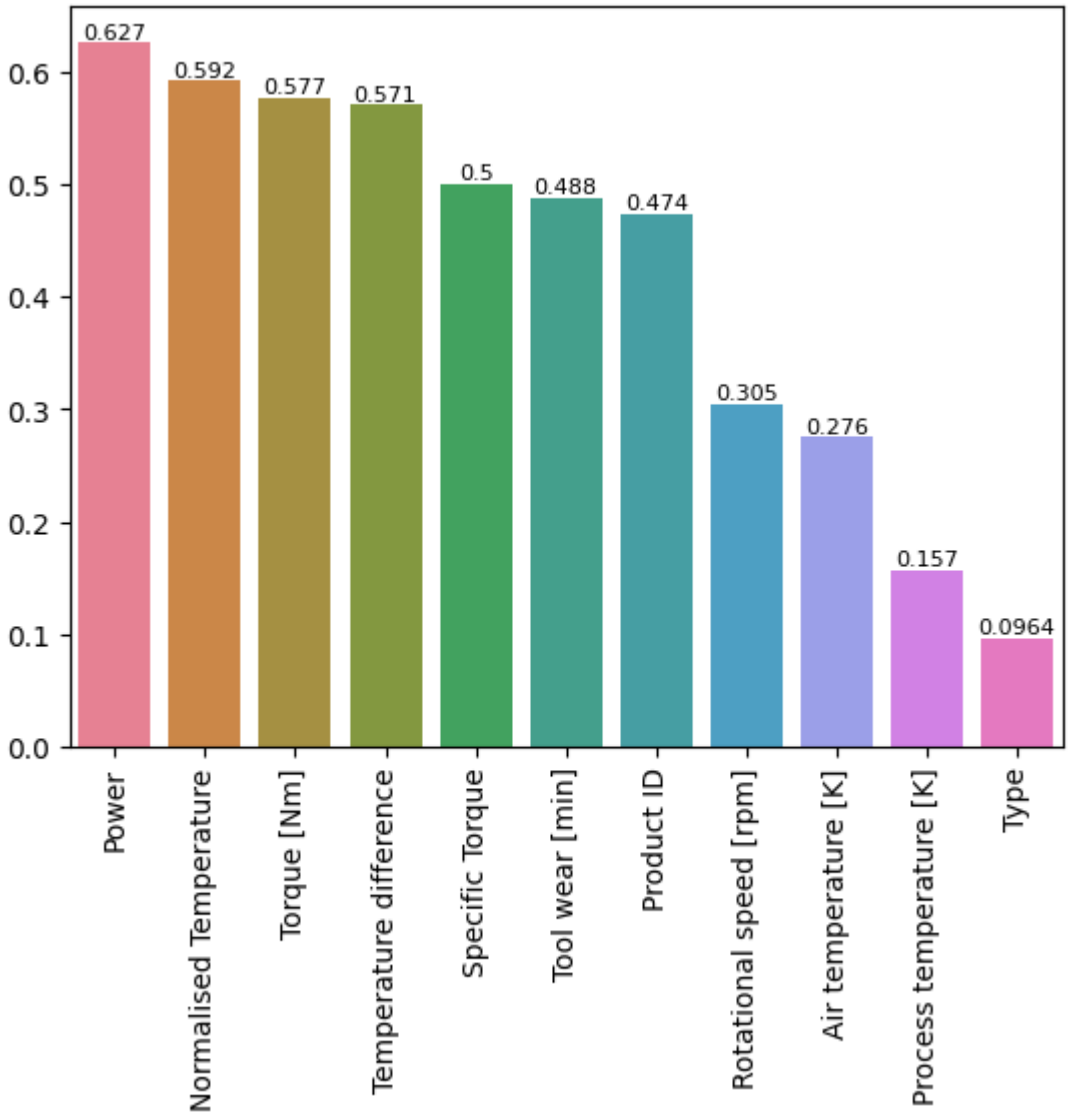

Figure 8 - Mutual Information ranked feature space; higher value indicates greater feature contribution to target prediction.

From the figure we can see the descending rank of feature importance revealed via mutual information assessment. Although this graphic indicates poor feature contribution from Rotational speed [rpm], Air temperature [K], Process temperature [K] and Type – our feature space is small, having only 11 components. It would be prudent to retain as much information as possible at this stage and instead handle feature reduction as a '*hyperparameter'* of model training.

Although not employed in this report, there is opportunity to expand the feature space with methods such as polynomial features and radial basis function and subsequently reduce the dimension of the data feature space via methods such as PCA, UMAP or t-SNE to capture patterns and eliminate multicollinearity. As well, feature selection could be applied when dealing with larger sets of features to reduce the number of features to those most influential upon outcome, such as ranked methods as that above (Mutual information) or exploitation of feature importance parameters gained from coarse trained decision trees. Although not employed here, an example of a plausible automated workflow for feature selection combining Pearsons and mutual information is given below; where persons correlation assesses whether

collinearity exists between features (*Value>0.9*) and Mutual Information determines the features importance in gaining information towards a target failure mode value.

1. *Evaluate features via Pearsons's correlation (PC) assessment, setting diagonal to 0.*
2. *Evaluate mutual information (MI) of each feature.*
3. *For each feature in order of least to most important, evaluated from MI.*
    a. *If the feature's PC column or row has a correlation > 0.9 – remove feature*
        i. *Restart from step 1*
    b. *Else proceed to next least important feature, until end of list.*

The final step of the EDA is the scaling of the data, here we apply a Min-Max Scaler, the complete data processing pipeline is as follows, note many of the pipeline transforms are custom functions – the transformer naming aligns with the operations performed above to assist in clarity of their role in the data manipulation.

```
df_target_preprocessing = Pipeline([('Intractable_Machine_Failure_Highlight',
HighlightingIntractableMachineFailures()),
                        ('Drop_Columns', ColumnDropper(['TWF','RNF'])),
                        ('Drop_intractable_failure_rows', NanRowDropper()),
                        ('Drop_Machine_Failure_column',
ColumnDropper(['Machine failure'])),
                                    ])
preprocessing = Pipeline([
    ('Type_Mapping', MappingTypeTransformer()),
    ('ProductID_Numeric_Extraction', NumericExtractionProductIDTransformer()),
    ('Domain_Feature_Engineering', DomainFeatureAdditionTransformer()),
    ('Drop_Columns', ColumnDropper(['UDI'])),
    ('Imputer', PandasSimpleImputer()),
    ('Outlier_Detecton_and_tranformation', OutlierTransformer(method='zscore',
threshold=3)),
    ('Skew_correction', SkewnessTransformer()),
    ('Scaler', MinMaxScaler())])
```

Within this pipeline we generate the above new domain features and then apply outlier detection wide – It would be prudent to investigate the newly generated domain feature distributions as we did with the original feature space, but here we assumed investigation of the original feature space was sufficient as the domain features are parent to their creation.

The dataset, information shown below, is now cleaned, transformed and prepared via the above pipeline for modelling. The label encoded 'Machine failure' column is not present as the format is string which is not a data type captured in the data frame quantitative description.

	Product ID	Type	Air temperature [K]	Process temperature [K]	\
count	7932.000000	7932.000000	7932.000000	7932.000000	
mean	0.646600	0.288203	0.511855	0.532025	
std	0.343779	0.367539	0.216912	0.182635	
min	0.000000	0.000000	0.000000	0.000000	
25%	0.259336	0.000000	0.336957	0.382716	
50%	0.845637	0.000000	0.521739	0.543210	
75%	0.924108	0.630930	0.673913	0.666667	
max	1.000000	1.000000	1.000000	1.000000	

	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	\
count	7932.000000	7932.000000	7932.000000	
mean	0.464963	0.496660	0.428330	
std	0.175929	0.168448	0.251662	
min	0.000000	0.000000	0.000000	
25%	0.342233	0.381601	0.211155	
50%	0.437780	0.499148	0.430279	
75%	0.558547	0.611584	0.645418	
max	1.000000	1.000000	1.000000	

	Temperature difference	Power	Normalised Temperature	\
count	7932.000000	7932.000000	7932.000000	
mean	0.533409	0.493792	0.529809	
std	0.222713	0.168464	0.224888	
min	0.000000	0.000000	0.000000	
25%	0.377778	0.380166	0.357437	
50%	0.488889	0.492786	0.497793	
75%	0.755556	0.608789	0.748518	
max	1.000000	1.000000	1.000000	

	Specific Torque
count	7932.000000
mean	0.491005
std	0.169292
min	0.000000
25%	0.371329
50%	0.488684
75%	0.604551
max	1.000000

Figure 9 - Description of features post EDA-database.

6. Preliminary Model Evaluation

In this section of the report the cleaned data prepared in the earlier chapter will be used to train three classification models, Logistic Regression, SVM and Random Forest (An ensemble of decision trees) with default hyperparameter settings. The algorithm training metrics are as follows, as aligned with the business objectives detailed in the problem statement.

1. **Primary Objective:** Prevent failures, achieved by Maximising Recall: True Positives / (True Positives + False Negatives) – Dataset comparative business statistic; **2.7%** of instances resulted in failure.
2. **Secondary Objective:** Reduce unnecessary maintenance/tool change, achieved by Maximising Precision: True Positives / (True Positives + False Positives). – Dataset comparative business statistic; **0.4%** of tooltips replaced.

Where a *True Positive* is a correctly predicted failure; a *False Negative* is an incorrectly predicted non-failure; and a *False Positive* is an incorrectly predicated failure.

A 5-fold stratified split of the data is performed, and the cross-validation results of the model are assessed against each of the respective folds validation set. An average is taken across the folds to obtain the precision, recall and f1-score for each failure mode. A confusion matrix is presented to showcase the combined spectrum of prediction and actual values across the validation of the fitted models on the 5-folds.

Accuracy is a measure of correct values, which is a poor metric to observe for progression of our models as the minority failures do not equally influence this value, instead the majority non-failure outcome mostly determines this figure. The *macro avg* is the average of the column metrics weighting the contributions from target variable outcomes equally, whereas the *weighted avg* is the average with weighting relative to support of the outcome class.

6.1. Logistic Regression

Table 2- Logistic Regression 5-fold validation metrics

	precision	**recall**	**f1-score**	**support**
No Failure	0.98	1.00	0.99	7721
HDF	0.59	0.12	0.20	85
PWF	0.00	0.00	0.00	64
OSF	0.78	0.11	0.20	62
accuracy			0.98	7932
macro avg	0.59	0.31	0.35	7932
weighted avg	0.96	0.98	0.97	7932

1. The recall is near-perfect for capturing non-failure's however ranges from 0-0.12 for failure mode instances – a poor performance overall.
2. Precision is improved in comparison to recall with no-failure, HDF & OSF exhibiting reasonable values of 0.98, 0.59 & 0.78 respectively. However, the precision value of 0.00 for PWF is poor.

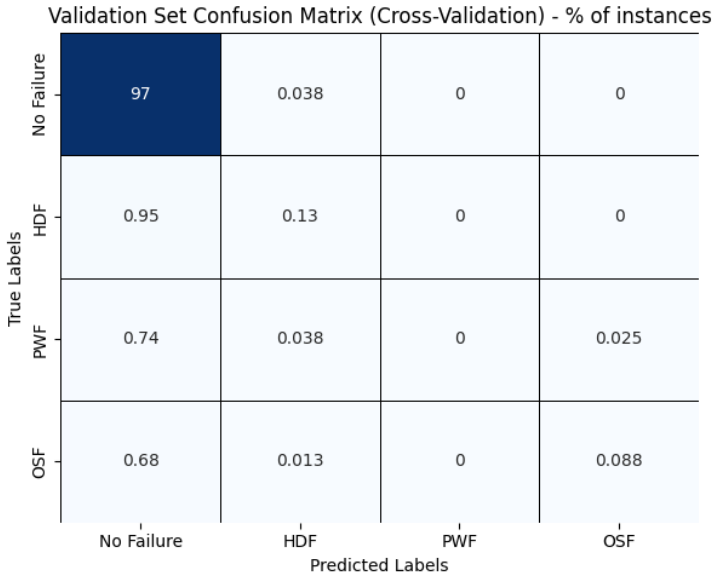

Figure 10 - Confusion matrix LR

From a review of the confusion matrix, the model seems to default predicting non-failure. This is likely due to the model being unable to adjust to the imbalance of the target data leaning heavily towards non-failure outcome, with default hyperparameters.

6.2. SVC

Table 3 - SVC 5-fold validation metrics

	precision	**recall**	**f1-score**	**support**
No Failure	0.98	1.00	0.99	7721
HDF	0.67	0.05	0.09	85
PWF	0.00	0.00	0.00	64
OSF	1.00	0.29	0.45	62
accuracy			0.98	7932
macro avg	0.66	0.33	0.38	7932
weighted avg	0.97	0.98	0.97	7932

1. Performance is poor on recall, however, shows an improvement in OSF recall compared to Logistic Regression.
2. SVC improved precision when compared to Logistic Regression across the classes; however, remaining ineffective for PWF prediction.

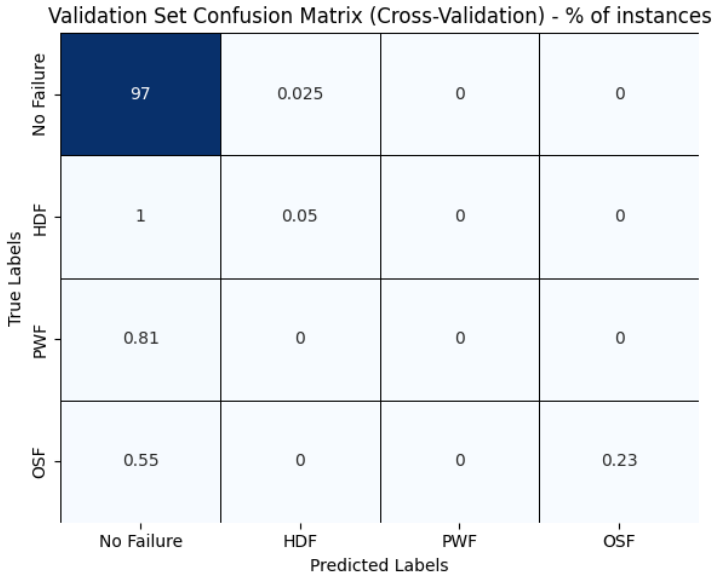

Figure 11 - Confusion matrix SVC

SVC is better at predicting OSF but does not consider HDF or PWF for any predictions across the validation set bar one instance of class HDF. Instead opting to predict the majority class (no failure). This, like Logistic Regression could be due to the imbalance in the data set heavily weighting the model to place importance on predicting the majority no failure class. The comparative improved capability to predict OSF does result in a model which has higher accuracy, but this is marginal.

6.3. Random Forest

Table 4 - Random Forest classifier 5-fold validation metrics

	precision	**recall**	**f1-score**	**support**
No Failure	0.99	1.00	1.00	7721
HDF	0.96	0.95	0.96	85
PWF	0.97	0.97	0.97	64
OSF	1.00	0.47	0.64	62
accuracy			0.99	7932
macro avg	0.98	0.85	0.88	7932
weighted avg	0.99	0.99	0.99	7932

1. A wide range of recall values, scores from 0.47 to 1.00. Prediction of OSF is weakest; all models seem to conflate OSF with no failure to some degree.
2. Precision is near-perfect; less than 0.1% of instances of incorrectly predicting a failure from the 7935 instances in the data set.

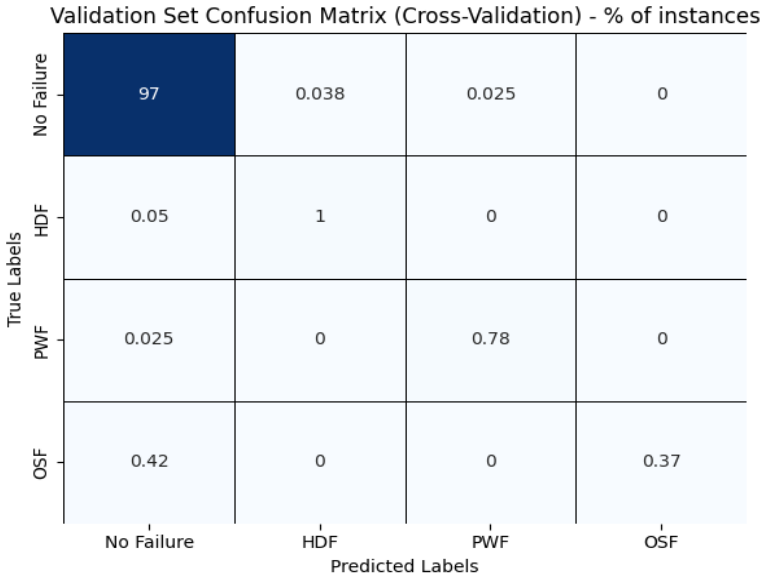

Figure 12 - Confusion matrix RF

The diagonal of the Random Forest Classifier confusion matrix is dense, the predictive capability of this model is significant – this is to be expected as an ensemble method composed of 100 decision Trees its relative performance will be greater than the LR and SVC models above. The percentage of predicted failures in the validation dataset is 2.2%, 0.06% of the instances are false positives for failure prediction, the precision is great at a value of 0.97.

The tool-changes due to incorrect failure predictions are as low as 0.06% of instances, which with this preliminary model is already ***over a 6-fold improvement*** *upon the business metric of 0.4%, the reduction of unnecessary tool changes.*

The number of failures missed by this predictive model equate to 39 instances, with 171 failures correctly predicted – the recall is 0.85. 19% of failures are not predicted, the majority are failure mode OSF.

Only ***0.5% of failures are not predicted,*** *this predictive model could serve over* ***a 5-fold reduction in failures*** *compared to the 2.7% business metric achieved from the scheduled approach.*

6.4. Review of preliminary models

The three models evaluated without hyperparameter tuning were Logistic Regression (LR), SVC and Random Forest (RF). Although benefiting from being highly interpretable, the LR and SVC proved incapable of distinguishing failures from non-failures, this is likely due to both (i) the imbalance of the data set, and (ii) the linear nature of these approaches – whereas RF is inherently capable of linear and non-linear trend identification and capture. Whilst weighting of the dataset and/or SMOTE or other sampling technique could address issue (i);

and further, employment of the kernel trick, either RBF or Polynomial could address the latter issue (ii); the random forest classifier solve time was instantaneous and without modification managed to produce results of significant improvement in comparison – capturing majority of the failures, and reducing failures by prediction by 77% compared to the scheduled tool-tip replacement technique employed prior.

Hence the model selected for fine tuning, at the expense of model interpretability, in the next section is Random Forest as it inherently handles the non-linearity of feature to target relationships present in this dataset. In the following section we will focus on regularisation of the model – overfitting could be a reason to the model's weak performance when predicting >50% of OSF incorrectly – the focus of the tuning is to improve the OSF predictions.

7. Fine-Tuning

Although the preliminary random forest model is sufficient in improving upon the scheduled approach to tool tip replacement. There is concern for the model's ability to correctly predict OSF failures which is only correct in <47% of the validation instances. Hence, we will be performing fine tuning of the model in aim of refining the tools capability regarding all failure modes but primarily driven by this poor performance in OSF prediction represented prior.

7.1. Overfitting assessment

We will evaluate the preliminary random forest model's performance on the training set data and compare it to the validation data results to determine whether the model is overfitting. The confusion matrix results are presented below as percentages of the total instances.

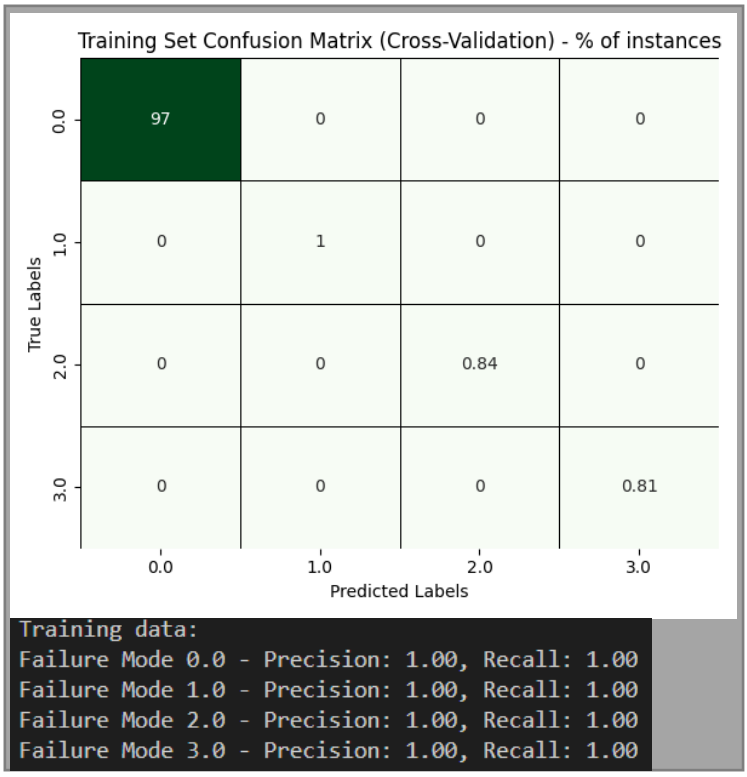

Figure 13 - Percentile confusion matrix - training data.

The trained classifier performs perfectly on the training data, a score of 1.0 for precision and 1.0 for recall – indicating overfitting as performance on unseen validation data was significantly poorer for OSF in comparison.

Fortunately, the precision of this initial classifier is significantly high, ranging from 96-100% per failure outcome. There is potential for improvement of recall and in affect improving OSF predictions, through regularisation, at the cost of some precision. Knowledge of the model at present being overfit will help guide parameter scope exploration in the next section.

The overfit random forest model is indicative of a model requiring regularisation, this will be achieved by addressing the overfitting via hyperparameter tuning.

7.2. Hyperparameter Tuning Pipeline

Fine-tuning of the model hyperparameters is particularly important for attaining a more accurate prediction of failure mode OSF, with the present model 53% of OSF failures are not predicted.

Reducing the complexity of the model and thus reduction of the bias towards the training set – will result in an increase in the variance, this will improve the model's aptitude to correctly predict values in the non-trained data spaces – the models generalisation.

Below are the hyperparameter's of the random forest model and ranges we will explore via RandomisedSearchCV – this search algorithm is chosen for its efficiency of exploration through large hyperparameter space.

Table 5 - RandomisedSearchCV distribution ranges.

Parameter	**Exploration Space**
n_estimators	$50 \rightarrow 400$
max_depth	$1 \rightarrow 100$
max_features	$\mathrm{sqrt}, \mathrm{log2}, \mathrm{None}$
max_samples	$0.1 \rightarrow 0.9$
class_weight	$None, balanced$
min_samples_split	$2 \rightarrow 30$
min_samples_leaf	$1 \rightarrow 30$

To rank the models, the RandomsedGridSearchCV requires a score to measure and compare estimators produced. The score selected here is the F-beta score, an adaptation of the f-score with a beta value used to prioritise precision ($\beta < 1$) or recall ($\beta > 1$). We will use the same hyperparameter exploration pipeline with RandomisedGridSearchCV and an external loop to explore a set of beta values of $\beta = 1,\ 2, 3, 4, 5, 6$ & 7; an increased beta value above the value of 1 favours improved recall compared to standard f-score, and the expected effect is to see improved OSF predictions.

Within this scorer, the multiclass outcomes are averaged to discount the influence of the imbalanced set, to overcome avoid the majority non-failure class overcontributing to the score in this unbalanced dataset.

The dataset is split by the RandomisedSearchCV via a 3-fold stratification. See the Randomised search code and associated pipeline below, where the preprocessing variable is the pipeline introduced at the end of the data processing chapter.

```
full_pipeline = Pipeline([('Feature_Processing', preprocessing),
                          ('Classification_Model',
RandomForestClassifier(random_state = 38513497, oob_score = True))])


search = RandomizedSearchCV(full_pipeline, param_distributions=distrubutions,
                                n_iter=100, cv=StratifiedKFold(n_splits=3,
shuffle=True, random_state=38513497), scoring=scorer, random_state=38513497,
verbose = 1, n_jobs=-1)
```

For each of the f-score values the search is run for 100 iterations. The recall and precision values are derived from the out-of-bag validation, along with the business metrics for percentage of tool changes and percentage of missed failures. The results of the beta value runs are given below in the plot. The x axis is the Beta value, and the two-y axis represent the business goals of this study.

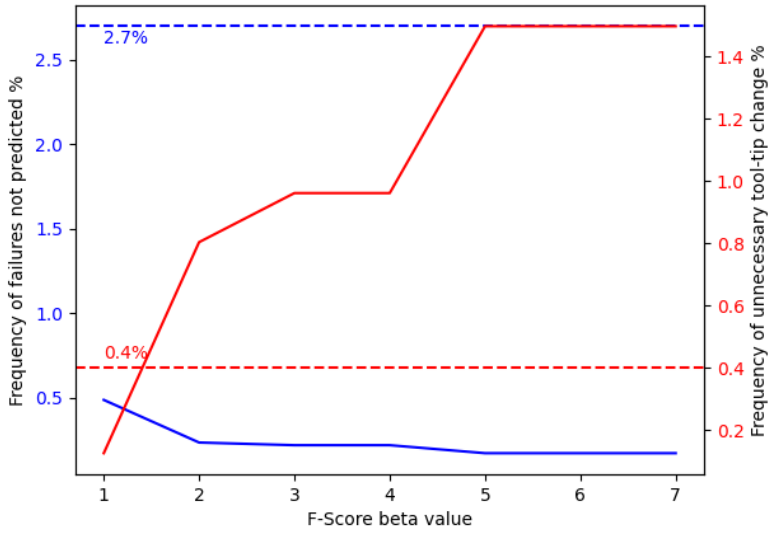

Figure 14 - Business metrics vs F-score Beta value.

More detailed results including the recall and precision are given in the table below. Demonstrating the positive trend between increasing recall and decreasing the frequency of non-predicted failures, and positive trend between the precision and decreasing frequency of unnecessary tool tip changes.

Table 6 - Beta value affect upon business metrics & classification metrics (recall & precision)

F-Score beta value	*Frequency of unnecessary tool-tip change %*	*Frequency of failures not predicted %*	*Precision*	*Recall*
1	0.13	0.49	0.96	0.85
2	0.80	0.24	0.81	0.92
3	0.96	0.22	0.79	0.92
4	0.96	0.22	0.79	0.92
5	1.50	0.17	0.72	0.93
6	1.50	0.17	0.72	0.93
7	1.50	0.17	0.72	0.93

We can see from the figure above, the tool tip (red line) metric has a positive trend, whereas the failures not predicted metric has a negative trend with increasing beta value. The business metrics are opposing, in that an increase in one inevitably results in a decrease of the other, this is a disguised precision/recall trade off.

As indicated in our problem statement our priority goal is the minimisation of failures below the baseline of 2.7%, and secondary to this the minimisation of unnecessary tool tip changes to at or below 0.4%. Thus, from this plot we see the beta value of 5 gives us the minimum frequency of failures not predicted equal to 0.17%, this comes with the greatest frequency of tool tip changes of 1.49%. Any further increases in beta result in the same business metric outcomes – as per out of bag metric evaluation.

*The tool-changes due to incorrect failure predictions are as frequent as **1.49%** of instances, which with this hyperparameter tuned model is **a 4-fold increase** upon the business metric of **0.4%,** unnecessary tool changes.*

*Only **0.17% of failures are not predicted,** this predictive model could serve over **a 15-fold reduction in failures** compared to the **2.7%** achieved via the scheduling approach.*

The validation set classification matrix for this model is given below, based on out of bag (OOB) data, reflecting the above statements regarding business metrics.

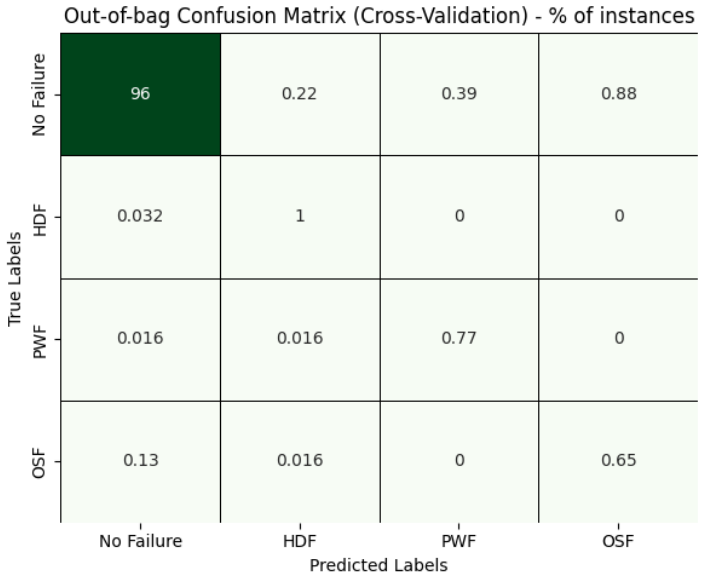

Figure 15- Tuned model OOB classification matrix

Compared to prior classification matrices we see a much-improved model, with the recent tuning we now only miss 17% of the OSF failures as opposed to the prior 53%. The sparsity observed for non-diagonal and non-edge cells of the classification matrix in the prior model is now absent, there are mis predicted failures. For instance, 0.032% of failure predictions are for HDF, but in fact are failures a result of OSF & PWF. We expect the model to be imperfect as we have regularised the model – reducing the overfitting nature of its prior default parameter set. The fortunate perspective is that a failure will still be predicted for these 0.032% of instances, although not the correct failure, this will still reduce failures at production which is our business objective.

The hyperparameters for this optimal run are as follows.

Table 7 - Hyperparameter settings derived for optimal estimator towards minimisation of failures not predicted.

Parameter	**Exploration Space**	**Optimal Value resultant from search**
n_estimators	$50 \rightarrow 400$	374
max_depth	$1 \rightarrow 100$	58
max_features	$\mathrm{sqrt}, \log 2, \mathrm{None}$	$sqrt$
max_samples	$0.1 \rightarrow 0.9$	0.973814
class_weight	$None, balanced$	$balanced$
min_samples_split	$2 \rightarrow 30$	2
min_samples_leaf	$1 \rightarrow 30$	15

This model is the final product of this study and will be used in the following section for full dataset training and assessment against our isolated test set.

8. Final Model Assessment

In this section we will first train the hyperparameter tuned model of the full validation set, then evaluate the model on the isolated test set from earlier in this report. The classification of this run is given below.

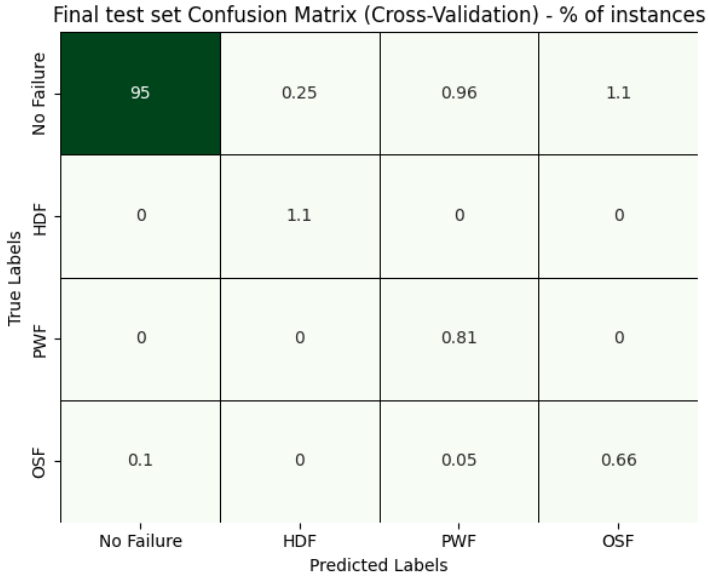

Figure 16 - Final test set evaluation of tuned model.

*The tool-changes due to incorrect failure predictions are as frequent as **1.97%** of instances, which with this hyperparameter tuned model is **a 5-fold increase** upon the business metric of **0.4%**.*

*Only **0.11% of failures are not predicted,** this predictive model could serve over **a 25-fold reduction in failures** compared to the **2.7%** figure reported above.*

With this final model, based on the trained and test set evaluated model; for every 1000 production runs of the milling equipment, with this predictive model guiding tool-tip change, 20 of 56 tool-tip changes would be unnecessary, however 26 of the total 56 would avoid failures – to the extent that only 1 failure per 1000 runs would not be predicted. Although greater in number than the scheduled approach, the sacrifice of 56 tool tip changes results in operation whereby failure only occurs 1/1000 production runs of the milling tool. A vast improvement upon the scheduled approach which 27/1000 failures.

9. Conclusion

9.1. Outcome

Qualitatively, we met our target. We improved the prediction of failures by 27-fold, although our secondary objective of minimising unnecessary tool-tips changes was not successful the overbearing financial importance of predicting failures over changing tool tips defines this model as a significant successor over the prior scheduled approach.

This predictive model can be extended to a secondary application, to help identify safe operating regions for tool tips based on their wear state, contributing to asset life optimisation.

9.2. Alternative/Future Exploration

While further refinement of the current model may require significant effort, there are several avenues for potential improvement. Specifically, addressing errors in the current approach through boosting methods or alternative strategies could enhance performance. These approaches may be iterated and tested throughout the workflow for better outcomes.

9.2.1. Deployment and Monitoring:

The next stage for this project would be the development of a basic flowchart for deployment, including steps for updates and monitoring model drift. Additionally, estimating potential cost savings based on the tool's tip costs (represented by diamonds) would provide useful insights.

Additionally, given the trends in tooltip and project costs over time, it would be beneficial to track their changing relative costs. A transition point may be reached where minimising tool-tip changes becomes more critical due to rising tool-tip costs and decreasing project material expenses.

9.2.2. Model Refinements and Techniques:

Several advanced techniques could be applied to further improve predictive capabilities:

- **Recursive Feature Elimination (RFE)**: This could streamline the model by selecting the most relevant features.
- **Dimensionality Reduction**: Techniques like PCA or UMAP could be employed to simplify the feature space and capture only the most important patterns, removing redundancies – although this would result in an uninterruptible model.
- **Hyperparameter Tuning**: A more refined search, such as Bayesian Optimisation (BOHO) or Hyperband, could replace the current random search for more efficient hyperparameter optimisation.
- **Feature Engineering**: Additional domain-specific features could be generated to further enhance model performance.

9.2.3. Model Selection and Ensemble Methods:

- **Outcome Representation**: Instead of using the current hierarchical approach, it may be beneficial to treat the outcome column as binary (e.g., 0/1) for simplicity.
- **Stacked Estimators**: Using multiple trained models as stacked estimators could potentially improve overall performance.
- **Boosting Methods**: Applying boosting techniques could help address the underperformance of certain models, though caution should be taken to avoid overfitting and lack of generalisability.

9.2.4. Alternative Approaches:

- **Anomaly Detection**: Rather than identifying failures explicitly, treating the problem as anomaly detection could provide a more effective way to spot failures or other issues.
- **Threshold Tuning (PR-AUC)**: Further tuning of the threshold using metrics like Precision-Recall Area Under the Curve (PR-AUC) could enhance model evaluation and performance.
- **Alternative Approach (Binary + Multiclass Hybrid):**
  Use a two-stage model:

  - **Stage 1:** Binary classifier to predict failure (`0` vs. `1, 2, 3`).
  - **Stage 2:** Multiclass classifier (trained only on failure instances) to predict specific failure modes (`1` vs. `2` vs. `3`).

  The potential advantages of this approach are two-fold, (i**)** Simplifies the primary classification task (failure detection) and, (ii) allows more focused optimisation of specific failure mode predictions in Stage 2.

Further exploration of these techniques, along with a more extensive preliminary study of models and grid search, could lead to improvements in predictive capability and overall model effectiveness.