Statistical Analysis of Permian Basin Equipment Maintenance at Chevron

Intelligent Systems Concentration Final Project

Einar López Altamirano
A01656259

Mentor
Dr. Raul Ramirez Velarde

February 2023 - June 2023

Index

Problem Definition and Objectives

One of Chevron's significant areas of opportunity is streamlining the maintenance activities for their upstream, midstream, and downstream equipment. The proper and optimal performance of their instruments is essential for all other processes within the company, and any failure or disruption in their operations results in a loss of company resources. Given the nature of the processes for which the equipment is used, constant preventive maintenance operations are expected and scheduled in advance. However, there is also a need for corrective maintenance operations when equipment fails, stops, or underperforms.

Instrument failures are often unexpected and sudden, making them challenging to predict, even for experienced workers or teams. Additionally, the corrective maintenance operations required to address and resolve these issues tend to be time-consuming and result in significant resource losses. In the past, before the growth and expansion of information technologies, companies had limited opportunities to proactively prepare for such challenges. However, with the advent of modern methodologies such as statistical analysis and prediction models, there are now numerous possibilities to forecast equipment failures.

In this report, we will delve into the methodology and results derived from the development of a statistical analysis on equipment maintenance activities. The primary purpose of this analysis is to present a proof of concept for streamlining corrective maintenance operations using machine learning techniques. The main objective of the statistical analysis is to accurately forecast the frequency, duration, and monetary impact of maintenance activities related to Permian Basin equipment.

Methodology

The successful completion of this project was made possible thanks to a dataset provided by Chevron, comprising 1.5 million entries. This dataset encompasses maintenance activities conducted on Permian Basin equipment from 2006 to 2018. By utilizing this dataset, we were able to train machine learning models to recognize patterns from past operations and make predictions about future failures. As a result, we can now more effectively schedule preventive maintenance operations to mitigate equipment disruptions caused by malfunctions or breakdowns.

The methodology employed to achieve this objective was as follows:

1. **Data understanding:** Extensive research was conducted to gain a comprehensive understanding of the dataset, including its columns, classes, magnitudes, and categories.
2. **Data clean-up:** Non-relevant columns were dropped, redundant data was removed or grouped, and efforts were made to reduce the dimensions of the dataset.

3. **Data splitting:** A sub-dataset was created specifically for corrective maintenance operations, excluding all entries related to preventive maintenance activities.
4. **Descriptive statistics:** A Pareto Frontier analysis was performed to focus solely on the equipment that accounted for 80% of the maintenance activities.
5. **Data recoding and class creation:** Categorical columns were transformed into numeric equivalents, and thresholds based on quantiles were established to group numeric values into classes.
6. **Predictive model creation, testing, and evaluation**: Various machine learning models were applied for both classification and regression analysis to identify the best performers. Rigorous testing and evaluation were carried out to assess their effectiveness in predicting future outcomes.
7. **Hyperparameter tuning:** An extensive search for optimal parameter configurations was conducted to fine-tune the selected models and achieve the highest performance.

Data Description

The dataset mentioned earlier consists of 55 variables and 1,429,919 samples. Each entry represents a maintenance operation performed on a specific equipment, identified by a unique code. The dataset provides information about instrument malfunctions, their impact on production, and offers insights into the tasks involved in the maintenance operations. This includes details about dates, roles, and teams associated with the maintenance activities. This comprehensive dataset serves as a valuable resource for analyzing and optimizing the maintenance processes to enhance efficiency.

The dataset provided did not include any accompanying documentation, manuals, or references. This created challenges and uncertainties throughout the project as there were no clear explanations for certain variables, their magnitudes, or their classes.

That's why the research process was necessary before anything else. Finding information about the dataset was an enormous struggle. There was minimal information available online, and even finding other databases similar to Chevron's was unsuccessful. After these findings, an examination was done to find any official standards or protocols that the maintenance process data recording could have implemented. The hope was that these documents would describe some of the database variables, their meanings, possible values, and magnitudes.

Based on the available information, the ISO 14224:2016 standard titled "Petroleum, petrochemical and natural gas industries — Collection and exchange of reliability and maintenance data for equipment" appeared to be a suitable reference as it aligned with the industry, processes, and activities. However, it turned out to be less useful than anticipated. While some variables in Chevron's dataset were described in the standard, there was insufficient information to enhance our understanding of the data.

The research was temporarily halted to initiate a more in-depth analysis of the provided data. One significant issue with the database was the abundance of NaN values. Out of the 55 columns, three columns (DaysLeftForSchCompliance, DaysLeftForPM, SystemStatus) contained entirely NaN values, while 12 other columns (PMOverdue, PMOriginalDueDate, EquipmentDown, Cause, FailureReason, Requirement, CompletedWeek, ScheduleWindowStart, TargetDate, ScheduleWindowEnd, ScheduleCompliant, and ActualDuration) had at least 50% NaN values.
If all rows containing at least one NaN value were eliminated, the database would be reduced from 1.5 million entries to approximately 400 thousand entries.

Other columns served no purpose, such as No Column Name which contained a written description of equipment failure that was not standardized. Similarly, EquipmentDescription faced the same issue. Additionally, StatusShortDescription provided a more descriptive version of StatusDescription rendering it unhelpful. Furthermore, certain columns lacked sufficient descriptive information. For example, Model contained two frequently occurring values, "Conventional" and "Conv," while "Failure Reason" included entries like "No code available, See comments" and "No code needed, See comments." Such issues were pervasive throughout the database, necessitating a new approach to minimize the deletion of entries whenever possible.

Avoiding the removal of values was necessary due to the nature of the problem. Since the goal was to predict the periodicity of corrective maintenance activities, any modifications to the data frequency would impact the results and introduce bias. To illustrate, suppose there were 12 recorded operations for a specific equipment within a one-year timeframe, indicating a failure rate of approximately once a month. If 6 of those operations were dropped due to unusable data in certain columns, the same equipment would appear to fail once every two months. This issue persisted throughout the six years of data recording, underscoring the importance of data integrity for the effectiveness of the statistical analysis.

There were 14 columns that contained no NaN values, prompting the decision to proceed with the project's development while focusing solely on these variables. In addition to the aforementioned reasons, it is reasonable to assume that these variables hold significant relevance, as they cannot be left empty when recording maintenance activity logs. The selected columns are as follows: WorkOrder, SupervisorRole, TradeGroup, CreatedDate, IsAffectingProduction, AffectedProduction, GrossProductionLoss, Duration, Safety, WOType, Reopened, StatusDescription, EquipmentCode, StatusCode, and EquipmentType.

Data Transformation

As mentioned previously, it is necessary to divide the dataset into two categories: corrective maintenance operations and preventive maintenance operations. This division was determined based on the values found in the WOType column, which include "PPM", "JOB", "XA", "AA", "STAT", "IN", "MRC", "XL", and "PL". With reference to the information obtained from ISO 14224:2016, it can be confirmed that PPM refers to Planned Preventive

Maintenance, while the remaining values were classified as corrective activities. After the splitting process, the number of entries in the dataset was reduced to 778,291.

With the new dataset, a further analysis was conducted, leading to an interesting observation. It was noticed that many equipment had logs for the same day, particularly on the date 27/05/2006, where at least 30 different equipment had reported between 80 to 362 instances. It is plausible that activities were registered with this date because it was the earliest one in the dataset, and previously unrecorded entries were labeled with the earliest possible date. This occurrence was observed multiple times throughout the data. To mitigate potential bias caused by unusually high numbers of operations on the same day, all activities for each unique equipment that occurred on the same day were grouped together, retaining only the operation with the longest duration. This resulted in a significant reduction for many samples. For instance, the three most frequently observed equipment, "C35", "V60", and "MWTP", went from 5,465, 2,943, and 2,286 entries to 1,152, 1,160, and 984 entries, respectively. Additionally, a new column called FailureCount was added, representing the number of failures for a specific instrument throughout its history. These transformations resulted in a reduction of the dataset to 604,397 entries.

The subsequent approach to further reducing the number of samples involved defining a Pareto Frontier to identify the most significant equipment. The objective was to retain only the instruments that accounted for 80% of the maintenance operations. By implementing this strategy, the dataset could be narrowed down to 73,274 distinct equipment, a considerable reduction compared to the previous count of 177,000. As a result, the database was streamlined to 483,517 samples. This marked the final reduction of the dataset, and preparations for the prediction models commenced thereafter.

Data Preparation

In order to utilize machine learning models, data preparation is essential, and one of the initial steps involved recoding categorical variables into numerical representations. This recoding process was applied to several columns, including SupervisorRole, TradeGroup, IsAffectingProduction, Safety, WOType, Reopened, StatusCode, and EquipmentType. The methodology employed for recoding entailed replacing each possible value with a number ranging from 0 to the total number of possible values, starting with the value that had the highest frequency of occurrences. For instance, in the SupervisorRole column, the possible values of "Surface Maintenance Supervisor", "Production Supervisor", "IE Supervisor", and "Surface Maintenance Advisor" were replaced with 0, 1, 2, and 3, respectively.

Some columns required type casting to facilitate the generation of new insights. For instance, the column CreatedDate, which indicates the date when the report for the corrective maintenance operation was filed, was cast to a DateTime object. This casting was necessary to enable sorting the dataset by EquipmentCode and CreatedDate, which in turn allowed the calculation of the column DaysFromLastFailure. DaysFromLastFailure represents the number of days that have passed since the last filing for a maintenance operation of a specific equipment. Initially, the first instance of each equipment had a NaN value since there was no previous sample to compare it with. To address this, the NaN values were replaced with the mean value of DaysFromLastFailure for each equipment.

Another important transformation involved translating the dates in the column CreatedDate to represent only the day of the year when each sample was created. This transformation aimed to capture the seasonality in the data and enhance the performance of the models.

One crucial methodology in data analysis involved exploring the presence of outliers and examining the quantiles of the numerical variables. Only four columns met the criteria for this analysis: Duration, AffectedProduction, GrossProductionLoss and DaysFromLastFailure.
An interesting insight emerged from this examination, revealing that three out of the four variables had highly imbalanced distributions. In the variable AffectedProduction, which represents the impact of failure on production, the range of values varied significantly, from 0 to 318,455,302. However, 99.9% of the data fell below 940, indicating that only a very small portion of the values exceeded this threshold. This significant difference greatly affected the predictions, as reflected by the mean value of 671, despite the fact that 99.9% of the data was smaller than this mean.

Similar issues were observed in the other variables, although to a lesser extent. To address these imbalances, the Winsorize technique was employed. This technique involves assigning the value at a certain quantile, such as 99%, to all values beyond that threshold. By applying this technique, the distributions became more balanced, leading to improved performance of the machine learning algorithms. For instance, after applying winsorization, the mean of the AffectedProduction variable decreased to 4.4843.

Predictions

Both regression and classification models were employed for the statistical analysis, aiming to provide a flexible and diverse solution. The methodology for both techniques was consistent. First, a subset of ten thousand random entries was created from the database. Then, four distinct data frames were generated, each corresponding to a specific variable. The data was subsequently normalized and scaled. The data frames were further split into "X_train", "X_test", "y_train", and "y_test" sets. Multiple models were trained for each variable, and corresponding metrics were obtained for evaluation. The best-performing model for each variable was selected. Hyperparameter tuning was conducted to optimize the models, and the best configuration, along with the respective hyperparameters, was chosen as the optimal model for each variable.

Regression

Four different regression models were trained: Logistic Regression, Decision Tree Regressor, Random Forest Regressor, and K-Nearest Neighbors Regressor. All models were implemented with their default configurations, except for the Logistic Regression model, which required a max_iter of 10000 to prevent errors.
The following metrics were calculated for each model and its respective variable: mean square error, mean absolute error, and r2 Score. Here are the results:

Duration			
Model	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
Logistic Regression	196.970	2.737	-0.039
Decision Tree Regressor	135.892	2.962	0.282
Random Forest Regressor	106.059	2.844	0.440
Support Vector Regressor	195.732	2.770	-0.033
K-Nearest Neighbors Regressor	164.178	3.245	0.133

Affected Production			
Model	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
Logistic Regression	1692.467	4.286	0.017
Decision Tree Regressor	3066.124	6.633	-0.779

Random Forest Regressor	1677.099	5.089	0.026
Support Vector Regressor	1667.301	4.103	0.032
K-Nearest Neighbors Regressor	1775.713	5.290	-0.030

Gross Production Loss			
Model	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
Logistic Regression	501.704	2.717	-0.014
Decision Tree Regressor	979.359	5.276	-0.981
Random Forest Regressor	444.151	4.871	0.101
Support Vector Regressor	501.168	2.806	-0.013
K-Nearest Neighbors Regressor	559.98324	4.783	-0.132

Days From Last Failure			
Model	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
Logistic Regression	562640.764	381.006	-0.346
Decision Tree Regressor	623198.243	424.807	-0.491

Random Forest Regressor	290954.435	324.231	0.303
Support Vector Regressor	472865.573	338.951	-0.131
K-Nearest Neighbors Regressor	379977.897	385.053	0.090

The highlighted rows indicate the best performing models for which hyperparameter tuning will be conducted. The following hyperparameter grids were defined:

Random Forest Regressor	
Parameters	**Possible Values**
n_estimators	5, 10, 50, 100, 250
max_features	1.0 , sqrt, log2
max_depth	None, 10, 50, 100
min_samples_split	2, 5, 10
min_samples_leaf	0.01, 0.1, 1,3,5
bootstrap	True, False

Support Vector Regressor	
Parameters	**Possible Values**
kernel	linear, rbf, poly, sigmoid
degree	3,5,7,10
C	0.001, 0.01, 0.1, 1, 10
epsilon	0.1, 0.01, 0.001
gamma	scale, auto

And the following configurations produced the best results.

Duration			
Random Forest			
Hyperparameters	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
n_estimators: 250, min_samples_split: 10, min_samples_leaf: 1, max_features: log2, max_depth: 50, bootstrap: False	111.717	2.921	0.410

Affected Production			
Support Vector Regressor			
Hyperparameters	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
kernel: poly, degree: 5, C: 0.1, epsilon: 0.001, gamma: scale,	1636.52	4.008	0.050

Gross Production Loss			
Random Forest			
Hyperparameters	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
n_estimators: 50, min_samples_split: 5, min_samples_leaf: 5, max_features: 1.0, max_depth: None, bootstrap: True	416.372	4.673	0.157

Days From Last Failure			
Random Forest			
Hyperparameters	**Mean Square Error**	**Mean Absolute Error**	**R2 Score**
n_estimators: 250, min_samples_split: 10, min_samples_leaf: 1, max_features: log2, max_depth: 100, bootstrap: True	270971.196	318.420	0.351

Classification

For the implementation of the classification models, some adjustments to the dataset were required. Firstly, unlike the regression models, the outliers were not dropped. Since the creation of classes was needed, all values could be accommodated within the defined class boundaries, making winsorization unnecessary.

The second adjustment involved defining the classes based on the quantiles of the predicted variables. The quantiles used were as follows:

Variable / Quantile	0.25	0.5	0.9	0.99	0.999	0.9999	Min	Max	Mean
Affected Production	0	0	4	75	940.96	7000	0	318455302	671.43
Gross Production Loss	0	0	0	100	594062.48	595199.29	0	595336	1436.39
Duration	1	1	7	93	259.48	621.64	1	687389	9.07
Days From Last Failure	24	127	1171	3565	4216	4328	1	5081	410.14

And the created classes were assigned as shown below:

Days From Last Failure										
Limits	0	1	24	36	127	597	1171	1936	3916	5081
Class	0	1	2	3	4	5	6	7	8	9

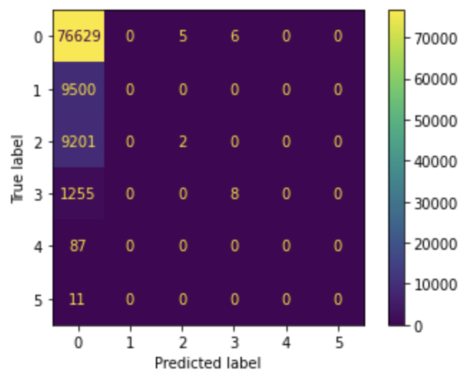
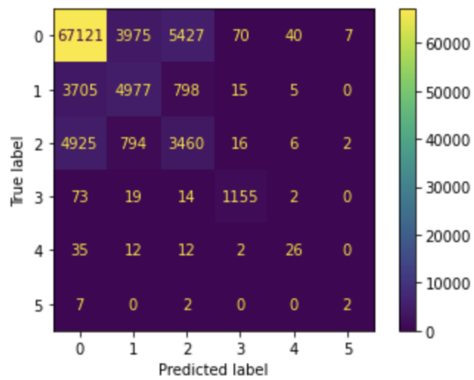
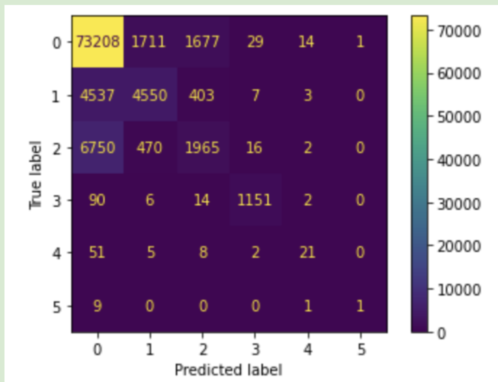
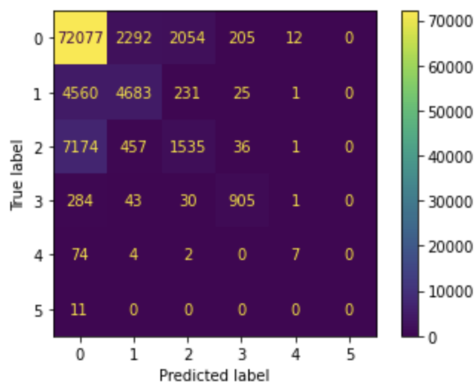
Affected Production								
Limits	0	4	14	75	150	940	7000	318455302
Class	0	1	2	3	4	5	6	7

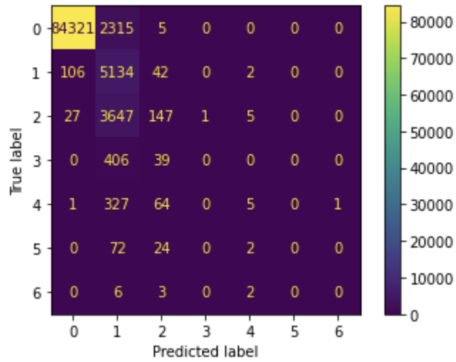
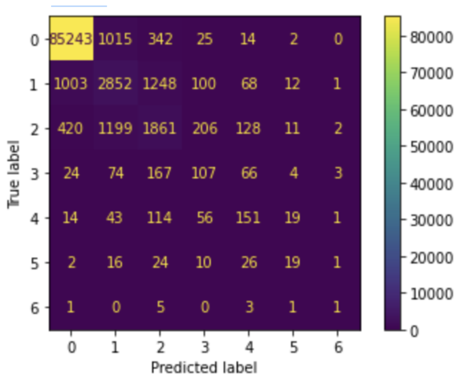
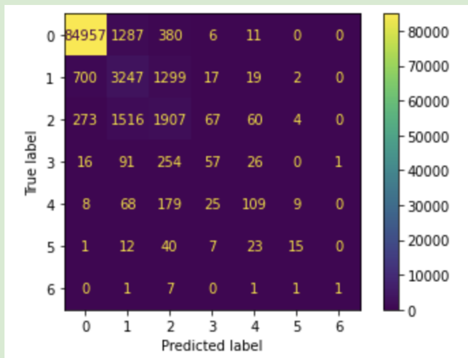
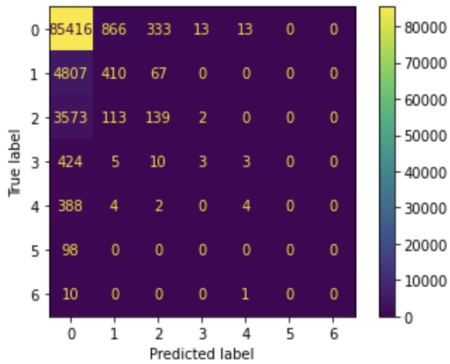
Duration							
Limits	1	2	7	93	259	621	687389
Class	0	1	2	3	4	5	6

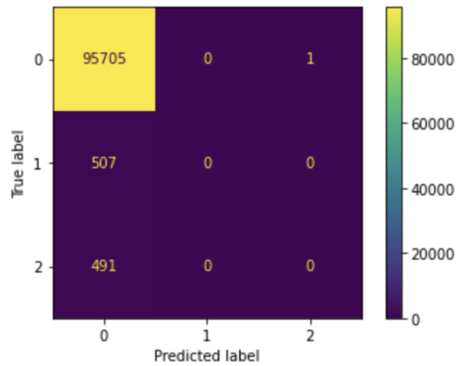
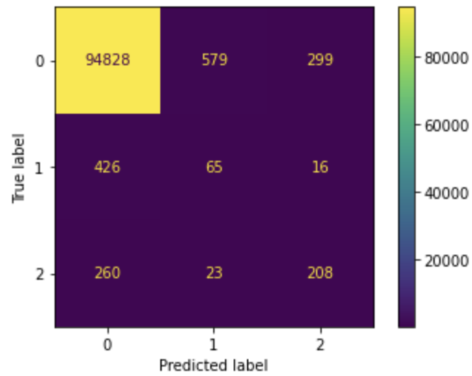
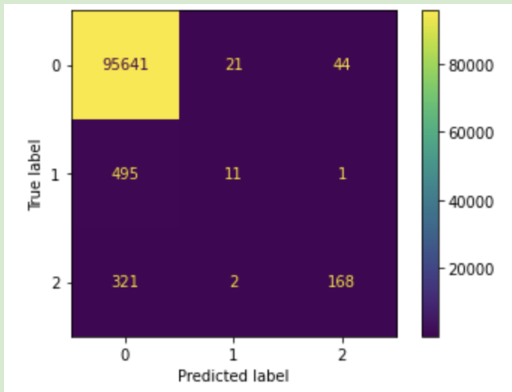
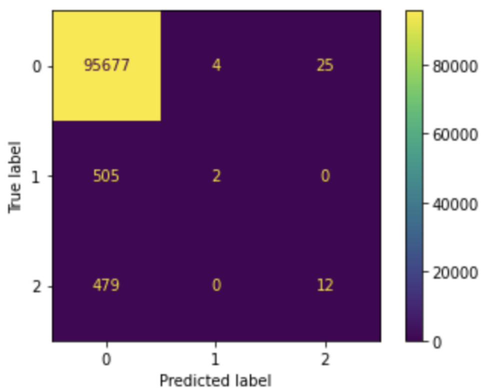
Gross Production Loss				
Limits	0	100	300	595336
Class	0	1	2	3

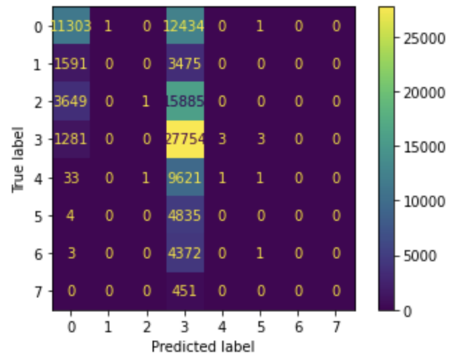
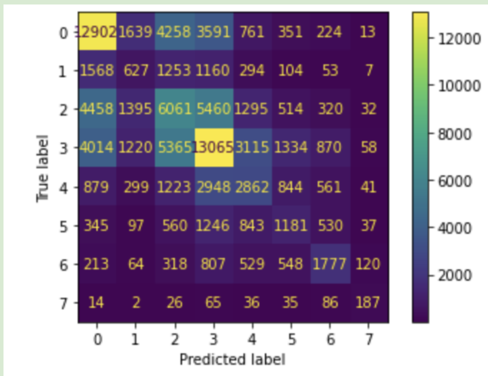
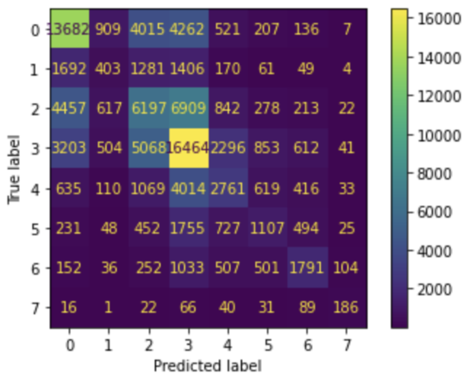
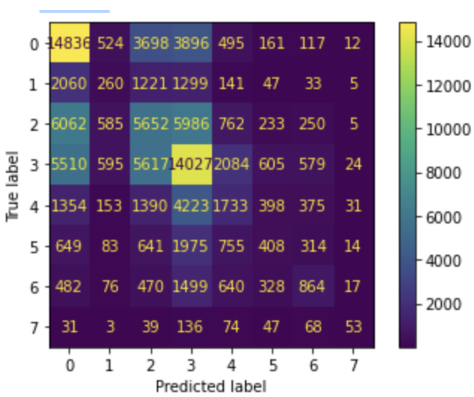
Similarly to the regression methodology, five different classification models were implemented: Multi-Layer Perceptron, Decision Tree Classifier, Random Forest Classifier, and K-Nearest Neighbors Classifier. All of the models were implemented with their default configurations.

The metrics of accuracy, precision, and recall were calculated for each model and their respective variables. Here are the results:

Duration					
Model	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
Multi Layer Perceptron	0.792	0.167	0.274	0.149	
Decision Tree Regressor	0.793	0.528	0.529	0.528	
Random Forest Regressor	0.836	0.481	0.661	0.528	
K-Nearest Neighbors Regressor	0.819	0.399	0.495	0.425	

<table>
<tr><th colspan="6">Affected Production</th></tr>
<tr><th>Model</th><th>Accuracy</th><th>Recall</th><th>Precision</th><th>F1 Score</th><th>Confusion Matrix</th></tr>
<tr><td>Multi Layer Perceptron</td><td>0.926</td><td>0.285</td><td>0.313</td><td>0.239</td><td>
</td></tr>
<tr><td>Decision Tree Regressor</td><td>0.933</td><td>0.416</td><td>0.422</td><td>0.418</td><td>
</td></tr>
<tr><td>Random Forest Regressor</td><td>0.933</td><td>0.391</td><td>0.531</td><td>0.419</td><td>
</td></tr>
<tr><td>K-Nearest Neighbors Regressor</td><td>0.889</td><td>0.159</td><td>0.257</td><td>0.165</td><td>
</td></tr>
</table>

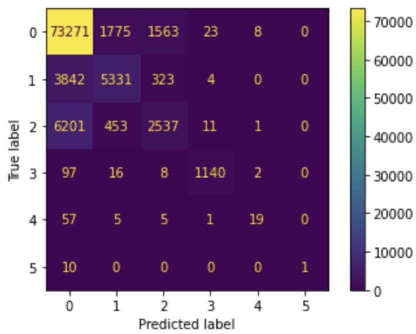
<table>
<tr><th colspan="6">Gross Production Loss</th></tr>
<tr><th>Model</th><th>Accuracy</th><th>Recall</th><th>Precision</th><th>F1 Score</th><th>Confusion Matrix</th></tr>
<tr><td>Multi Layer Perceptron</td><td>0.989</td><td>0.333</td><td>0.329</td><td>0.331</td><td>
</td></tr>
<tr><td>Decision Tree Regressor</td><td>0.983</td><td>0.514</td><td>0.495</td><td>0.504</td><td>
</td></tr>
<tr><td>Random Forest Regressor</td><td>0.990</td><td>0.454</td><td>0.701</td><td>0.504</td><td>
</td></tr>
<tr><td>K-Nearest Neighbors Regressor</td><td>0.989</td><td>0.342</td><td>0.549</td><td>0.349</td><td>
</td></tr>
</table>

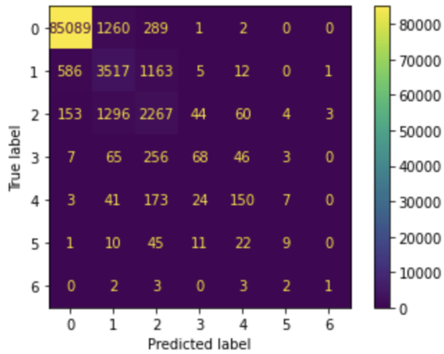
Days From Last Failure					
Model	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
Multi Layer Perceptron	0.403	0.178	0.216	0.132	
Decision Tree Regressor	0.399	0.348	0.342	0.345	
Random Forest Regressor	0.440	0.359	0.385	0.368	
K-Nearest Neighbors Regressor	0.391	0.253	0.302	0.263	

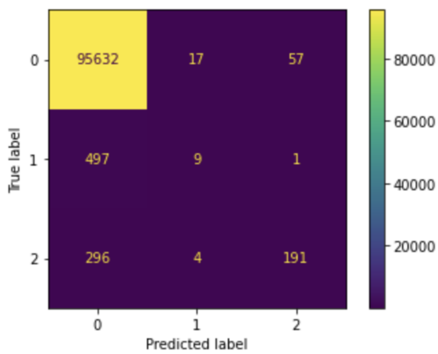
The highlighted rows indicate the best performing models for which hyperparameter tuning will be conducted. The following hyperparameter grids were defined:

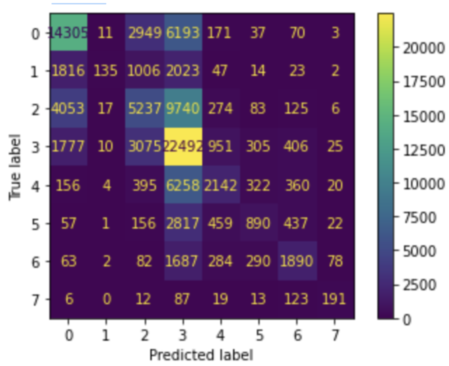
Random Forest Regressor	
Parameters	**Possible Values**
n_estimators	5, 10, 50, 100, 250
max_features	1.0 , sqrt, log2
max_depth	None, 10, 50, 100
min_samples_split	2, 5, 10
min_samples_leaf	0.01, 0.1, 1,3,5
bootstrap	True, False

And the following configurations produced the best results.

Duration					
Random Forest					
Hyperparameters	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
n_estimators: 50, min_samples_split:5, min_samples_leaf: 3, max_features: 1.0, max_depth: None, bootstrap: True	0.851	0.500	0.792	0.556	

Affected Production					
Random Forest					
Hyperparameters	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
n_estimators: 50, min_samples_split:5, min_samples_leaf: 3, max_features: 1.0, max_depth: None, bootstrap: True	0.942	0.421	0.516	0.442	

Gross Production Loss					
Random Forest					
Hyperparameters	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
n_estimators: 50, min_samples_split:5, min_samples_leaf: 3, max_features: 1.0, max_depth: None, bootstrap: True	0.990	0.468	0.686	0.515	

Days From Last Failure					
Random Forest					
Hyperparameters	**Accuracy**	**Recall**	**Precision**	**F1 Score**	**Confusion Matrix**
n_estimators: 250, min_samples_split:2, min_samples_leaf: 3, max_features: log2, max_depth: 100, bootstrap: True	0.488	0.366	0.535	0.385	

Conclusions

The established methodology yielded promising results, with both regression and classification models delivering relevant outcomes. While the regression models initially had slightly lower performance, this was expected given the wider margin of error associated with regression analysis.

Among the implemented models, Random Forest consistently provided the most impressive results. This could be attributed to the model's ability to capture a wide range of possibilities and discover intricate patterns within the data.

The success of the hyperparameter tuning process was evident, as most models exhibited improved performance after fine-tuning. In some cases, there were significant jumps in prediction accuracy before and after tuning. For instance, the Random Forest classification model for "Days From Last Failure" saw a remarkable increase in accuracy, soaring from 0.399 to 0.488.

However, it's worth noting that the Random Forest regression model for "Duration" experienced a decline in R2 score after hyperparameter tuning, dropping from 0.44 to 0.41. Nonetheless, for all other implementations and variables, there was an increase in performance ranging from 0.02 to 0.3 points, solidifying the success of the planned methodology.