# Business Problem

Many industries complete maintenance on their products. There are also business devoted to just this task. If a company can predict when and what unexpected maintenance will occur, it can improve all facets of the pipeline. Without this knowledge, teams may be blindsided with large maintenance tasks that require parts, time, and money to resolve. Predicting failures ahead of time allows a team to solve the issue before it happens, or before it gets worse. This saves time, effort, and money. Additionally, if customers are waiting for their product, this can help speed up the process so they have a shorter wait. All of these elements improve the company workflow and the customer relationship. See Appendix A for more information on predictive maintenance.

# Background

Maintenance failures are a problem many companies face. This dataset was generated synthetically in order to provide a way to get a general understanding of how to predict maintenance failures. The same principals can be applied to any type of product within many industries, including, but not limited to, aerospace, manufacturing, energy, and automotive.

# Data Explanation

The predictive maintenance dataset contains 10 columns and 10,000 observations. There is a variation of numerical and categorical variables. See Appendix B for a data dictionary with more details on each column.

## Data Preparation

### Target

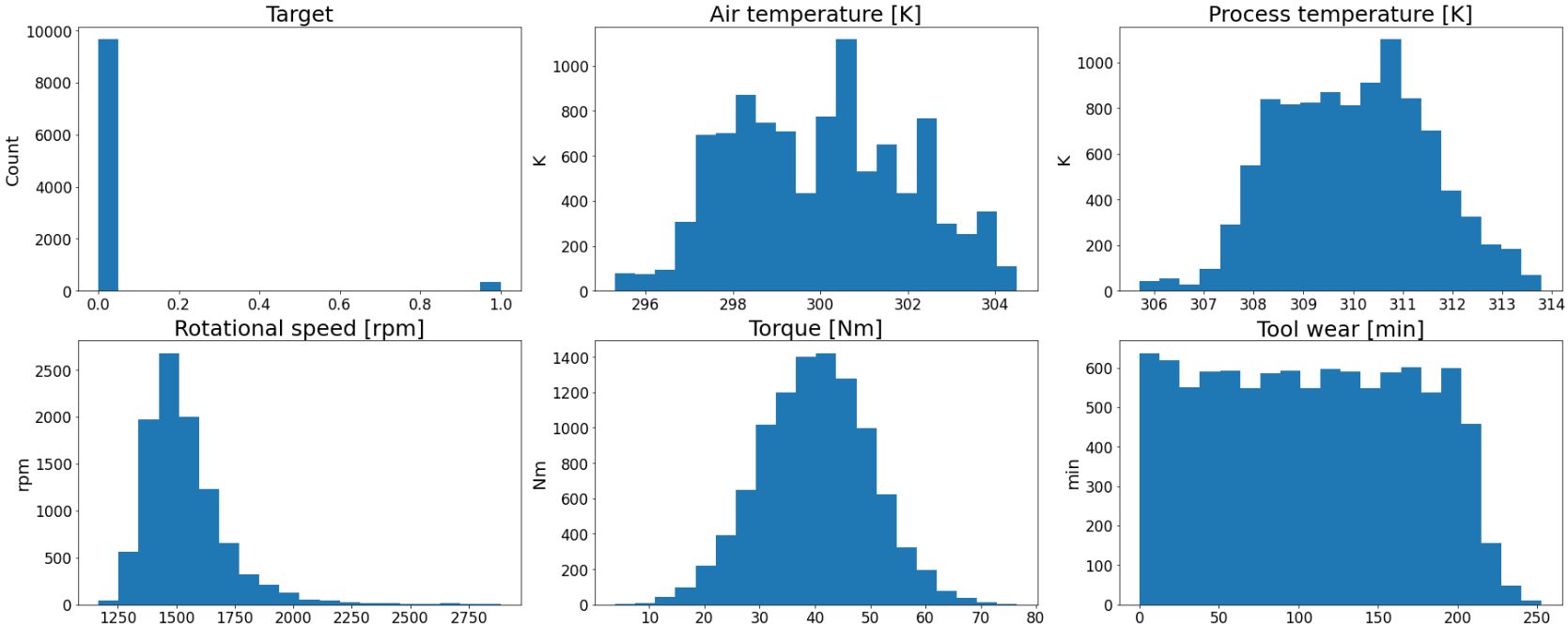
This dataset contains two target features, a binary indication of a failure or non-failure and the type of failure if one occurred. The data is heavily imbalanced as 3.39% of the records resulted in a failure. After further exploration, I found that there were 9 observations where a product was labeled as a failure where the failure type said non-failure. There were also 18 observations that were labeled as a non-failure but had a failure type of random. These two instances created observations that were unclear on whether or not a failure actually occurred therefore I opted to remove these from the dataset.

### Features

After removing the records with discrepancies around the target I was left with 9,973 observations. There are no null values within any columns. Majority of the columns are numeric values and each of them fall within a reasonable range of values, indicating there are no outliers. As shown in Figure 1, air temperature, process temperature, and torque all have a very normal distributions. Rotational speed has a slight positive skew, while tool wear contains minimal variation.

**Figure 1**

Numerical Features and Target Histograms



*Note. The distributions of the target and each numerical value in the dataset.*

After cleaning the data, I one-hot encoded the (product) type variable because it was the only categorical variable that would be trained in the model. I removed the UID, product ID, and failure type from the data in preparation for modeling. The ID columns are unique to each row which provides no value towards a prediction, and the failure type would introduce target leakage into training. Lastly, due to the imbalanced target class I chose to oversample the training dataset. This creates more records of maintenance failures based on the records that were already there. Since the data is already synthetic, I saw no issue in adding more synthetic data for oversampling.

# Methods

I am focusing on supervised binary classification machine learning algorithms within this project. These models fit the data well because it is already labeled, and the target I want to predict is whether or not a product is going to experience a maintenance failure or not so there are only two possible outcomes. To get an idea for what model will fit best with the data, I tested how six different algorithms perform without any hyperparameter tuning. These models included a logistic regression, decision tree, random forest, support vector machine, k-nearest neighbors, and a Gaussian Naïve Bayes. The simple logistic regression is a basic model that will provide a baseline to use for comparing how the other models perform.

# Analysis

After running the oversampled training data through the six models I chose, I saw decent results among each one. As displayed in Table 1, each model had an accuracy score within 0.82 – 0.89. The k-nearest neighbors classifier came out on top, it had the best accuracy and F1 score. The logistic regression also performed well landing itself in the middle.

**Table 1**

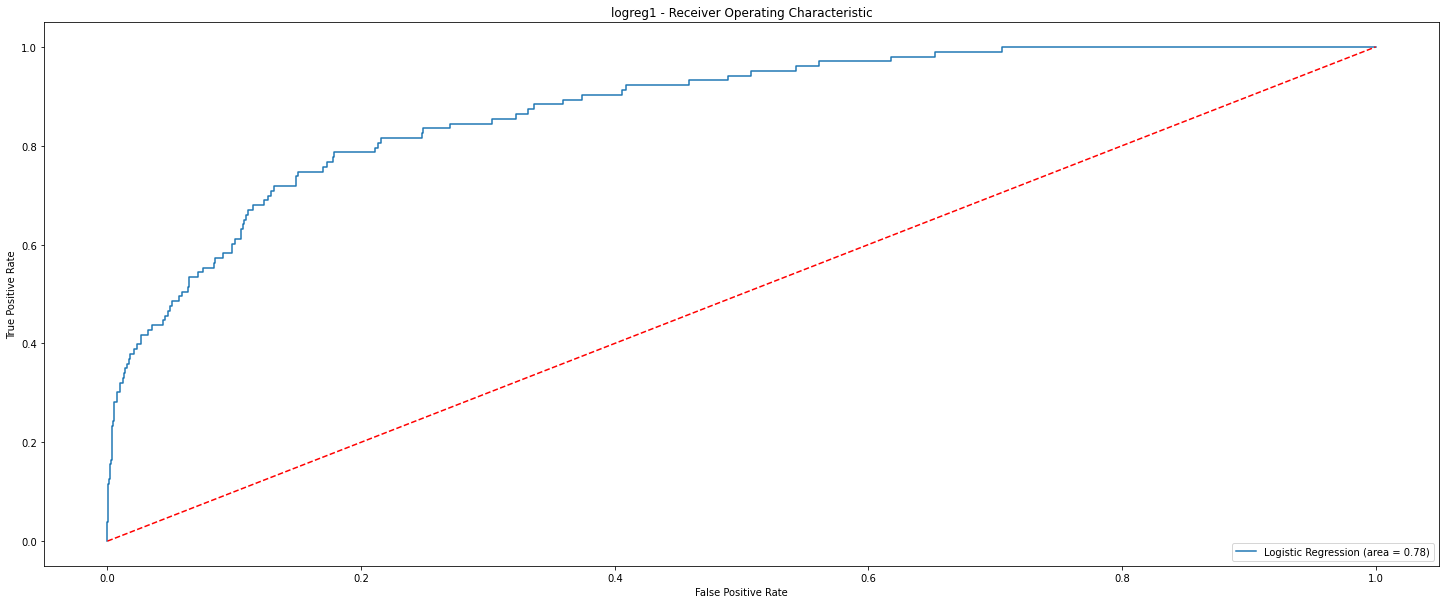
Initial Model Performance Metrics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **F1 Score** | **Recall** | **Precision** |
| K-Neighbors Classifier | 0.895963 | 0.894830 | 0.895980 | 0.902637 |
| Decision Tree Classifier | 0.887741 | 0.886408 | 0.887762 | 0.895027 |
| Logistic Regression | 0.886438 | 0.886431 | 0.886502 | 0.886640 |
| Random Forest Classifier | 0.878936 | 0.876856 | 0.878961 | 0.889899 |
| SVC | 0.868430 | 0.866764 | 0.868450 | 0.877532 |
| Gaussian Naïve Bayes | 0.829471 | 0.826353 | 0.829506 | 0.845825 |

Since the logistic regression is intended to be the baseline, that’s the model I started with to evaluate further. After evaluating this model on the test dataset, the accuracy dropped from 0.90 to 0.87, indicating the model generalized well to the unseen data.

**Figure 2**

Receiver Operating Characteristic for the Logistic Regression Model



*Note. The ROC Curve for the logistic regression model is not bad, but could be better.*

As shown in Figure 2, the ROC curve with an AUC of 0.78 is not a terrible metric, but it shows there is room for improvement. Although the accuracy and ROC curve metrics are decent, it is not the best metric for assessing model performance on imbalanced data. Oversampling the training data helped that issue some, but using the F1 score as the main evaluation metric when working with imbalanced classes is better. Unfortunately for this model, the F1 score for the main class (1 – failures) is 0.27. Recall for the main class is 0.68 and precision is 0.17 which shows that precision is what is brining down the F1 score. This means that the model is falsely predicting failures. In an effort to improve this, I started hyperparameter tuning for the logistic regression, and a KNN model. The final KNN model proved to do better than the final logistic regression model in most cases. Both models were better than the initial logistic regression. For the KNN, I was able to increase the main class metrics for the F1 score from 0.30 to 0.34. I was also able to increase the precision score from 0.19 to 0.25. That increase came at a cost though, as the recall metric dropped from 0.66 to 0.51. Overall, this was the best performing model out of each one I tested, yet there is always room for improvement. See Table 2 for the full evaluation metrics of the final KNN model.

**Table 2**

Evaluation metrics for the hyperparameter tuned KNN Model

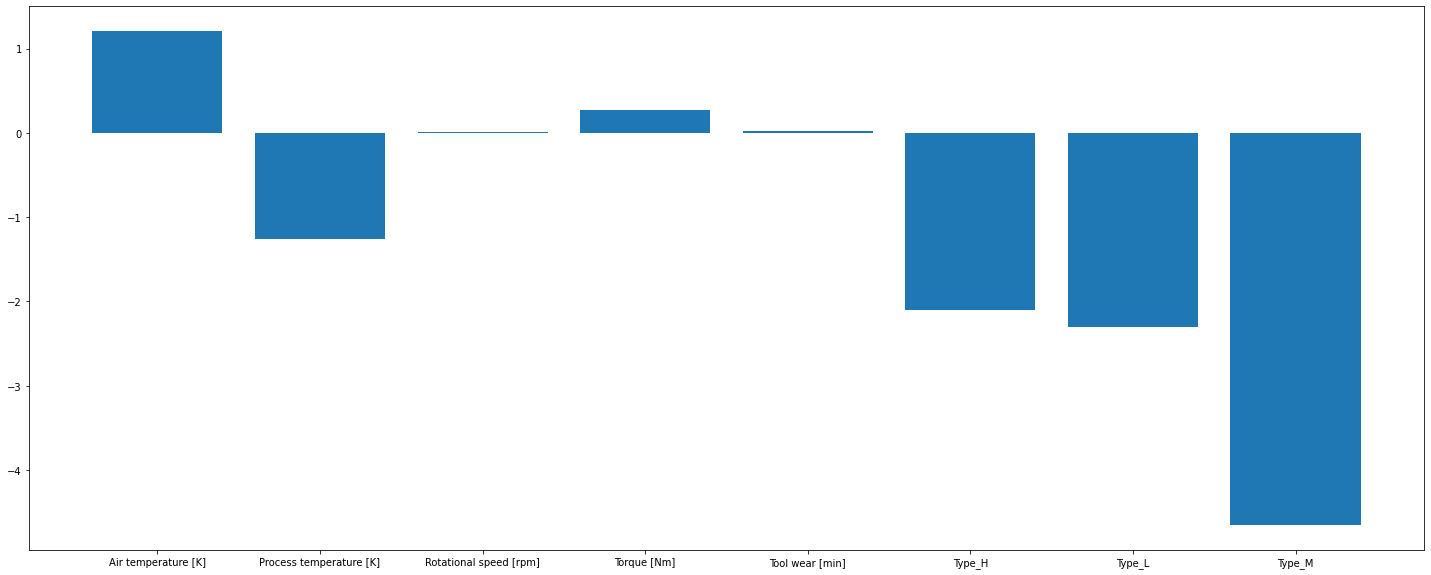
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Precision** | **Recall** | **F1-Score** | **Support** |
| 0 | 0.98 | 0.95 | 0.96 | 2889 |
| 1 | 0.25 | 0.51 | 0.34 | 103 |
| accuracy |  |  | 0.93 | 2992 |

# Conclusion

The initial exploration of the logistic regression model showed that there is room for improvement. In the first test of six models, the k-nearest neighbors performed better than the logistic regression so I explored that model further. An improvement was made through using KNN and hyperparameter tuning the leaf size, number of neighbors, and Minkowski formula. Although the evaluation metrics were not ideal for each model, additional information was recovered from the logistic regression model. As shown in Figure 3, the logistic regression suggests that air temperature and torque are the only features that play a strong role in predicting maintenance failures. Rotation speed and tool wear each had no impact on the model. The other features all had a negative impact, suggesting they better predict non-failures.

**Figure 3**

Logistic Regression Feature Importance



*Note. The positive and negative impact of each feature on predicting the target.*

# Assumptions

Due to the synthetic creation of this data, I am assuming it is a good example of a real maintenance dataset. This also leads to the assumption that it was created in an effort to have minimal data quality issues. I only discovered one problem with data quality and changed nothing else. I assume all feature distributions are in an ideal range for replicating a real life product.

# Limitations

The main limitation with this dataset is the number of features. I started with 10, but two of those were unique identifiers. The initial list also included two different target variables which left me with six features for training a model. It would be idea to have more features to choose from so I could test different combinations to see what creates a better model.

# Challenges

Initial challenges that I ran into during this project are around the data. The data is largely imbalanced as I have mentioned. This makes it difficult to predict a failure since there are so few examples of it. My first run at models had a hard time accurately predicting a failure. I used hyperparameter turning to solve this issue. I was able to see improved models, but there is still more room for improvement.

# Future Uses

The results of this model can be utilized in many ways. The first intended use case I see is to help those running maintenance be more informed on possible future failures. Seeing this information in advance and real time can help them better prepare. Additionally, I think this model could be used to test how new products might perform, especially if they are a variation of an existing product.

# Recommendations

To improve this project I would recommend getting more data. Additional information can better support and potentially improve the predictions. The current dataset does not include a time element, which could be helpful depending on the type of product. There also might be external factors that could play a role in predicting maintenance events that would be nice to include in the analysis.

# Implementation Plan

The initial implementation of this project would likely be in the form of a report. This creates an easy way to share out the information with suggested actions to be had from the predictions. The model will also need to be deployed in an MLOps pipeline so that predictions can be quickly provided and the model can be easier to maintain.

# Ethical Assessment

Predictive maintenance models can largely benefit a company. With that in mind, there are still ethical concerns to be aware of. The data and model used needs to be transparent, and also robust. If the data does not contain information on certain maintenance failures then the model could end up being biased and can miss major, rare, failures. Privacy is another concern. The data used in these models is tracked from products owned by customers. It is important to keep the customer anonymous to prevent introducing bias, and to make sure any sensitive data is stored safely.

# Questions

Below are ten questions an audience might ask over this project.

1. Why do we need to invest in this project?
2. How will stakeholders utilize this information?
3. Why can we trust these predictions?
4. How can we be sure this analysis doesn’t infringe on customer privacy?
5. How maintainable is this?
6. What happens if action is taken on an incorrect prediction?
7. How can this be used for upcoming products?
8. Why is this better than preventative maintenance?
9. How does this benefit our customers?
10. What resources are required for this project?

# References

Bansal, S. (2021, November 6). *Machine Predictive Maintenance Classification*. Kaggle. Retrieved September 18, 2022, from <https://www.kaggle.com/datasets/shivamb/machine-predictive-maintenance-classification>

*What Industries Use Predictive Maintenance Analytics?* Insights Success. (2021, October 4). Retrieved September 18, 2022, from <https://www.insightssuccess.com/what-industries-use-predictive-maintenance-analytics/>

*What Is Predictive Maintenance? Definition and FAQs*. HEAVY.AI. (n.d.). Retrieved September 18, 2022, from <https://www.heavy.ai/technical-glossary/predictive-maintenance>

**Appendix A**

**Predictive Maintenance**

Predictive maintenance is a data science process used to predict when a product or piece of equipment will need maintenance. There are many levels of what kind of maintenance that might be required, these can vary in severity. Predictive maintenance is often compared to preventative maintenance. Predictive is determining what will fail before it occurs using current and real time data. Preventative is focused on the historical data and focused on understanding different metrics like life expectancy of a product.

**Appendix B**

**Data Dictionary**

|  |  |
| --- | --- |
| **Feature** | **Description** |
| UID | Unique identifier for each instance |
| Product ID | Unique identifier for each product |
| Type | A letter L, M, or H to distinguish product quality. L is low (50% of all products), M is medium (30%), and H is high (20%) |
| Air Temperature (K) | Temperature surrounding the product |
| Process Temperature (K) | Temperature of the product during the process |
| Rotational Speed (rpm) | The speed of an object rotating around an axis |
| Torque (Nm) | The measure of force causing the product to rotate around an axis |
| Tool Wear (min) | For the quality variants H/M/L, 5/3/2 minutes are added for tool wear to the product used in the process |
| Failure Type | Description of the failure if one occurred. Possible values are heat dissipation, overstrain, power, tool wear, random, and no failure |
| Target | Identifier of a failure or non-failure |