Jasmine Cooper

D214

Task 2: Data Analytics Report and Executive Summary

July 14, 2024

Western Governor's University

Research Question

Research Question
To what extent does demand and parts-per-tool usage affect tooling (or equipment) failure?

Justification for the Research Question
The research question addresses a critical aspect of operational efficiency in test work centers. Equipment failures can lead to significant downtime, increased costs, and suboptimal resource allocation. By investigating the relationship between demand, parts-per-tool usage, and tooling failures, the study aims to develop a predictive maintenance and scheduling model. This model can provide actionable insights to minimize downtime, costs, and optimize resource allocation, thereby enhancing overall operational efficiency.

Context
In test work centers, efficient predictive tool usage, optimal reorder point management, and effective capacity planning are vital for minimizing downtime and costs. Traditionally, these tasks are managed through manual approaches and rely on tribal knowledge, leading to unexpected equipment failures and inefficient inventory management. The introduction of a data-driven approach through advanced analytics and visualization tools like Tableau offers a solution to these challenges.

By leveraging historical data and machine learning algorithms, the proposed model aims to forecast tooling failures before they occur. This proactive approach is expected to reduce unplanned downtime and extend the lifespan of test tools through timely maintenance interventions. In addition to enhancing reliability, this strategy can reduce repair costs and improve operational continuity.

Furthermore, optimizing reorder points for inventory management through data analysis can help determine the optimal inventory levels needed to meet demand, thereby minimizing the risk of stockouts or overstocking. This precision in inventory management can lead to reduced holding costs, improved cash flow utilization, and a more streamlined procurement process.

Visualizing demand and capacity constraints using interactive tools like Tableau empowers planners to make informed decisions, ensuring optimal resource utilization and operational efficiency. An interactive Tableau dashboard can centralize the visualization of key metrics, explore data trends, and simulate different scenarios, facilitating collaborative decision-making across teams.

Hypothesis
Demand and parts-per-tool usage statistically significantly affect tooling (or equipment) failure.

Discussion of Hypothesis
The hypothesis suggests a significant statistical relationship between demand, parts-per-tool usage, and equipment failure. This hypothesis is based on the premise that higher demand and

increased usage of parts per tool are likely to lead to greater wear and tear on equipment, resulting in more frequent failures. By validating this hypothesis, the study aims to provide a foundation for developing predictive maintenance and scheduling models that can preventively address equipment failures, leading to improved operational efficiency and reduced costs.

Data Collection

Methodology

For the data collection phase of this project, I utilized the AI4I 2020 Predictive Maintenance Dataset from the UCI Machine Learning Repository and created a supplemental dataset to fill in the gaps specific to my project's requirements. Additional data points were created in Excel, particularly for demand, parts per tool usage, lead times, and inventory levels. Below is the detailed process, advantages, and disadvantages of my data-gathering methodology, and how I overcame challenges during this process.

Data Collected

1. Historical Data on Tool Usage: Pulled from the UCI Machine Learning Repository, this dataset includes records of how often each tool has been used over a specific period of time. It provides a reflection of real predictive maintenance based on industry domain knowledge.
2. Demand Per Tool Usage: Randomized in Excel, representing the frequency and volume of requests for each tool. Demand ranges from 50 to 50,000 units per week to simulate real-world demand based on domain knowledge regarding prototype and production planning.
3. Parts Per Tool Usage: Created based on tool wear in minutes provided in the AI4I 2020 Predictive Maintenance Dataset with methodology described below in the Data Preparation and Extraction portion of this report.
4. Lead Times for Reordering Tools: Randomized in Excel, ranging from 7 to 90 days, based on domain knowledge.
5. Current Inventory Levels: Created using randomization logic based on product quality variants, discussed in detail in the Data Preparation and Extraction portion of this report.
6. Reorder Point Calculation: Created using industry standard reorder point calculation, which is the average daily demand multiplied by lead time.

Advantages and Disadvantages

One advantage of this data-gathering process was using a publicly available dataset from the UCI Machine Learning Repository, which ensures that the data is credible, well-documented, and relevant to predictive maintenance. Additionally, this dataset did not have any missing values, meaning data cleaning was not needed to properly use this dataset for analysis. Although, this dataset had many advantages, it lacked certain specific details for my project, necessitating the creation of supplemental data, which introduced additional steps and some complexity.

Challenges and Solutions

One challenge was ensuring the supplemental data accurately reflected realistic scenarios. To address this, I used a combination of domain knowledge and data from the AI4I 2020 Predictive Maintenance dataset to randomize values within realistic ranges to simulate real-world scenarios.

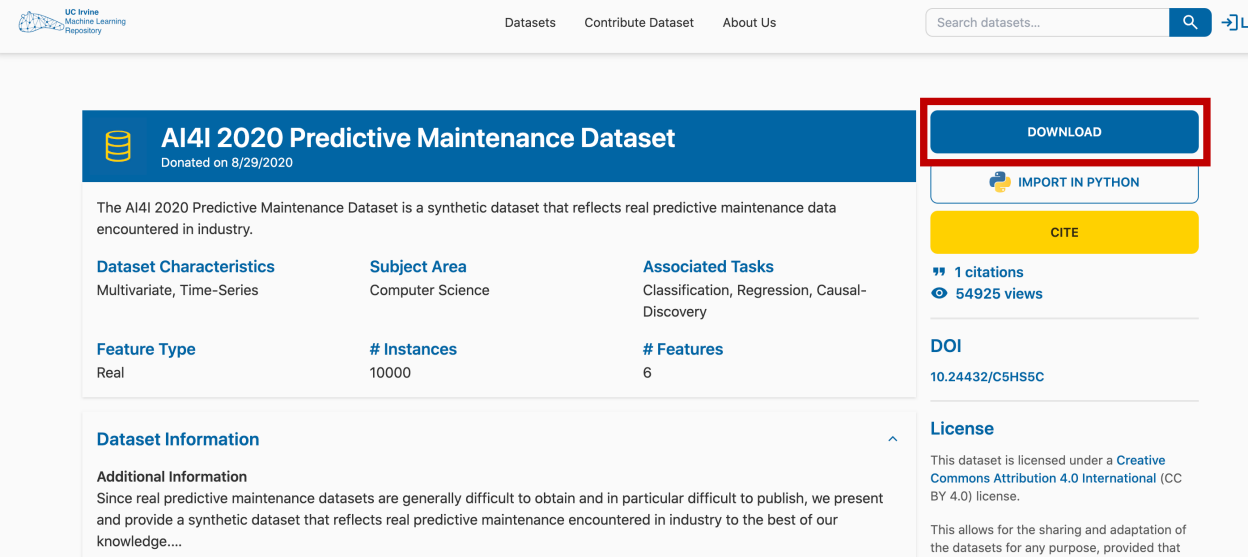
Data Extraction and Preparation

Overview

The data extraction and preparation phase involved several steps to ensure the data was ready for analysis and visualization. I used Excel and Python for data extraction and preparation, detailing each step below.

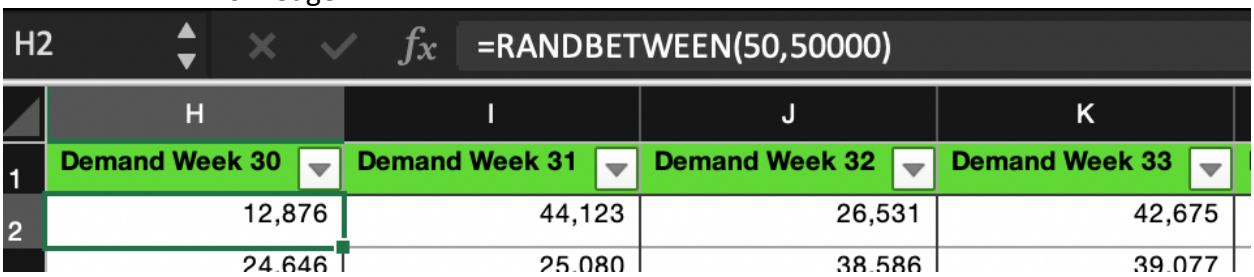
Data Extraction

- Pulling Data from UCI Repository: Downloaded the AI4I 2020 Predictive Maintenance Dataset.

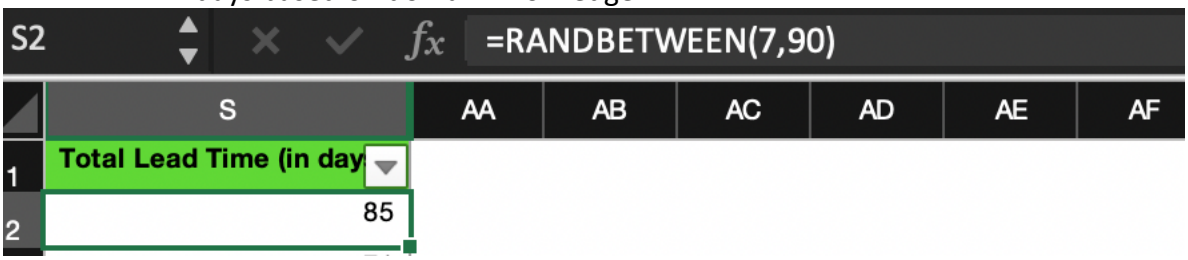


Data Preparation

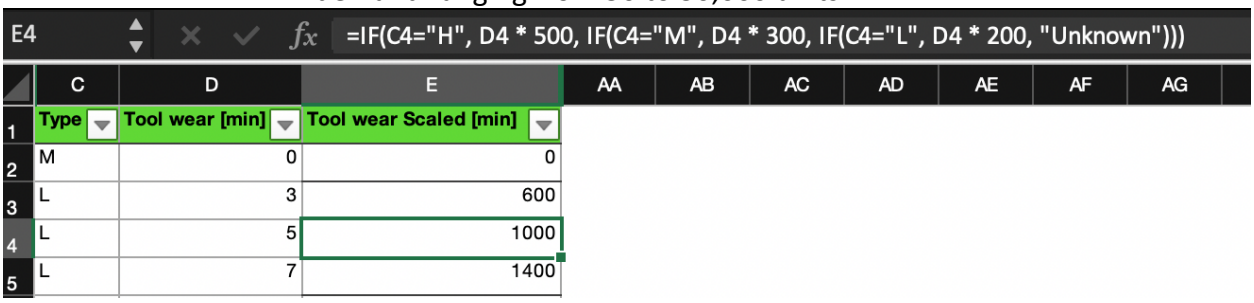
- Creating Supplemental Data in Excel:
  - Demand for Tool Usage: Randomized values between 50 to 50,000 to simulate real-world demand for prototype and production planning based on domain knowledge.



- Lead Times for Reordering Tools: Generated random values between 7 to 90 days based on domain knowledge.



- Parts Per Tool Usage Calculation: Based on tool wear and product quality variants.
    1. Tool wear is scaled by 100 relative to tool type to accommodate demand ranging from 50 to 50,000 units.

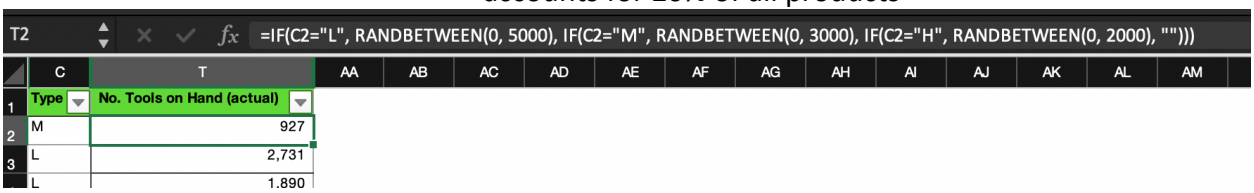


    2. The relationship between tool type, tool wear, and parts usage is defined in the dataset description:
        - High Quality (H): Adds 5 minutes of wear per part
        - Medium Quality (M): Adds 3 minutes of wear per part
        - Low Quality (L): Adds 2 minutes of wear per part
    3. Parts per tool were calculated using the formula: =IF(C2="H",IF(ROUNDDOWN(E2/5,0)=0,50,ROUNDDOWN(E2/5,0)),IF(C2="M",IF(ROUNDDOWN(E2/3,0)=0,30,ROUNDDOWN(E2/3,0)),IF(C2="L",IF(ROUNDDOWN(E2/2,0)=0,20,ROUNDDOWN(E2/2,0)))))
        - This formula ensures parts per tool values are practical by rounding down and setting minimums. For instance, if the parts per tool calculation yields 5800.6667 then the parts per tool calculation will output 5800. If the parts per tool calculation yields 0.6667, then the calculation will round down to 0. If the calculation rounds to 0, then a minimum quantity is set based on the tool type (e.g. H=50, M=30, L=20).

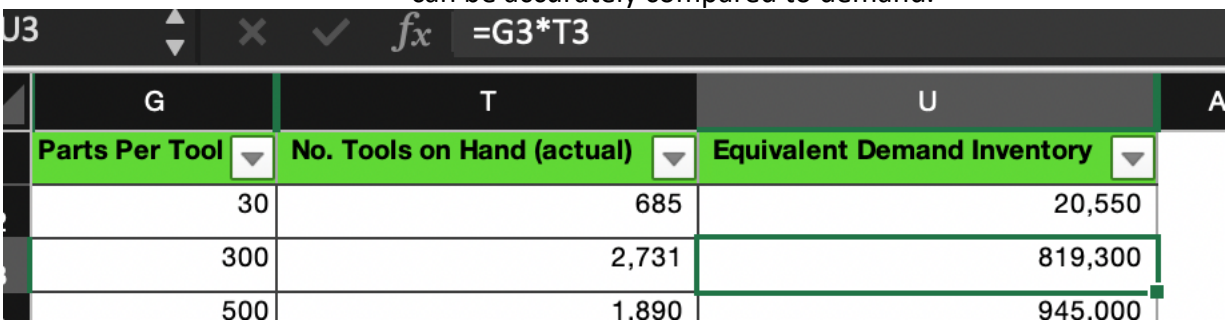
- Current Inventory Levels: Created using randomization logic based on product quality variants. The tool types help to identify the product quality, and in the dataset description it provides descriptions for the following tool types: Low (L) accounts for 50% of all products, Medium (M) accounts for 30% of all products,

and High(H) accounts for 20% of all products. There are 10,000 unique products in the dataset. Using a combination of the product quality variants and the number of products in the dataset, inventory levels were randomized.

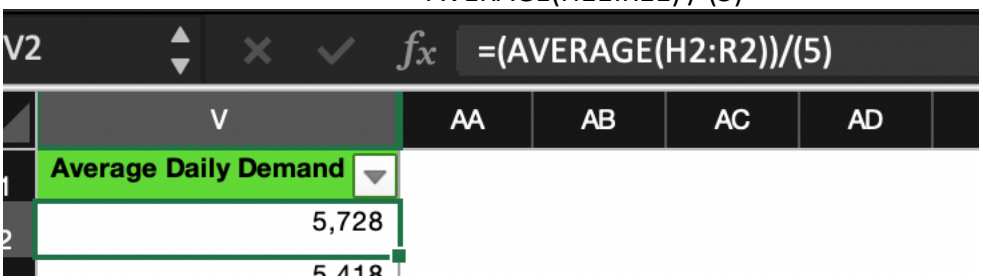
- Randomized Number of Tools on Hand:
  a. Low tool types inventory levels are between 0 to 5000 parts using =RANDBETWEEN(0,5000) since it accounts for 50% of all products
  b. Medium tool types inventory levels are between 0 to 3000 parts using =RANDBETWEEN(0,3000) since it accounts for 30% of all products
  c. High tool types inventory levels are between 0 to 2000 parts using the formula =RANDBETWEEN(0,2000) since it accounts for 20% of all products



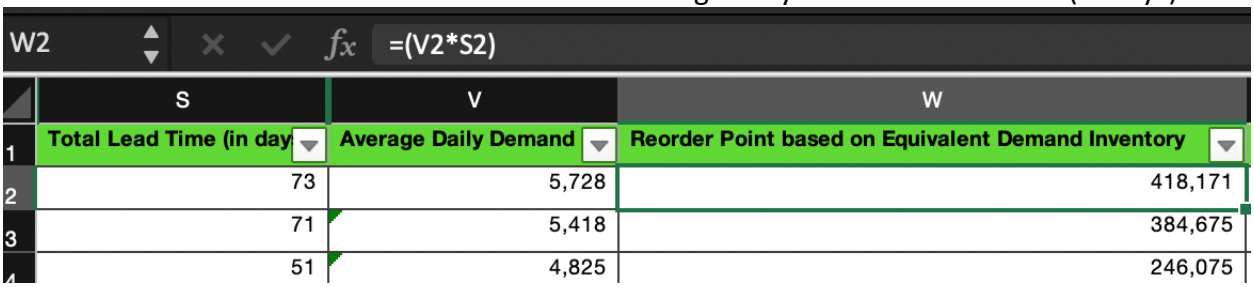
- Equivalent Demand Inventory: This indicates that inventory levels are converted to a comparable metric that aligns with demand. Multiplying the parts per tool usage by the number of tools on hand provides a realistic view of inventory levels that can be accurately compared to demand.



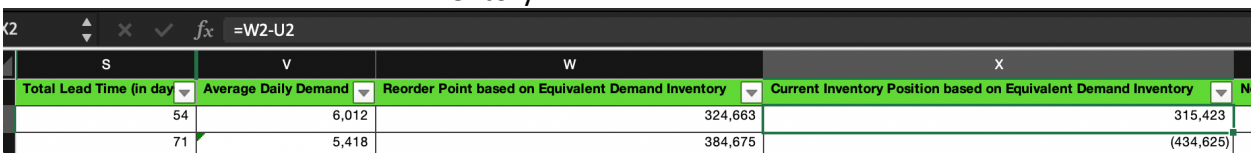
- Reorder Point Calculation
  - Calculate Average Daily Demand: Take the average weekly demand over a 10-week period and then divide by 5 to represent a 5-day work week. This was calculated using =AVERAGE(H11:R11) / (5)



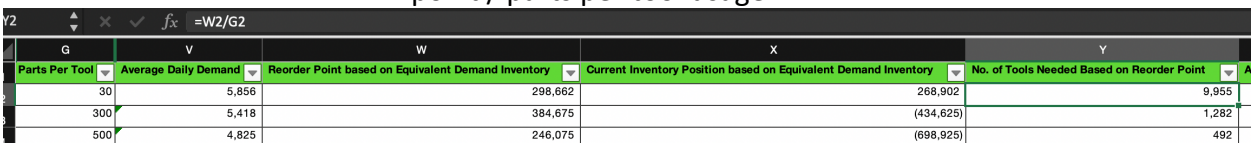
- Reorder Point = average daily demand * lead time (in days)



- Current Inventory Position = reorder point – equivalent demand inventory

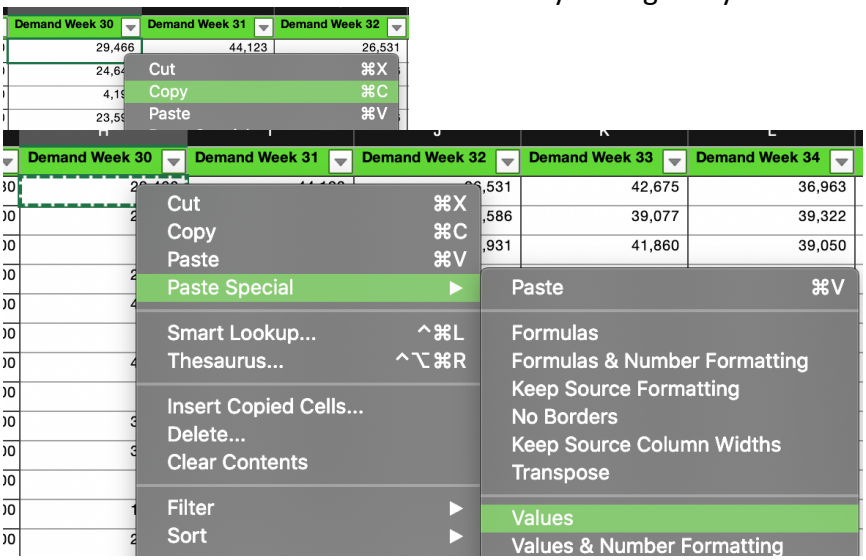


- Number of tools needed based on reorder points = reorder point / parts per tool usage



- Adjusted reorder point: Number of Tools Needed Based on Reorder Point – Number of tools on hand

- Data Consistency: After randomization was completed for Total lead time, Demand data, and Number of tools on hand, outputs were copied and pasted as values to ensure consistency during analysis and visualization.



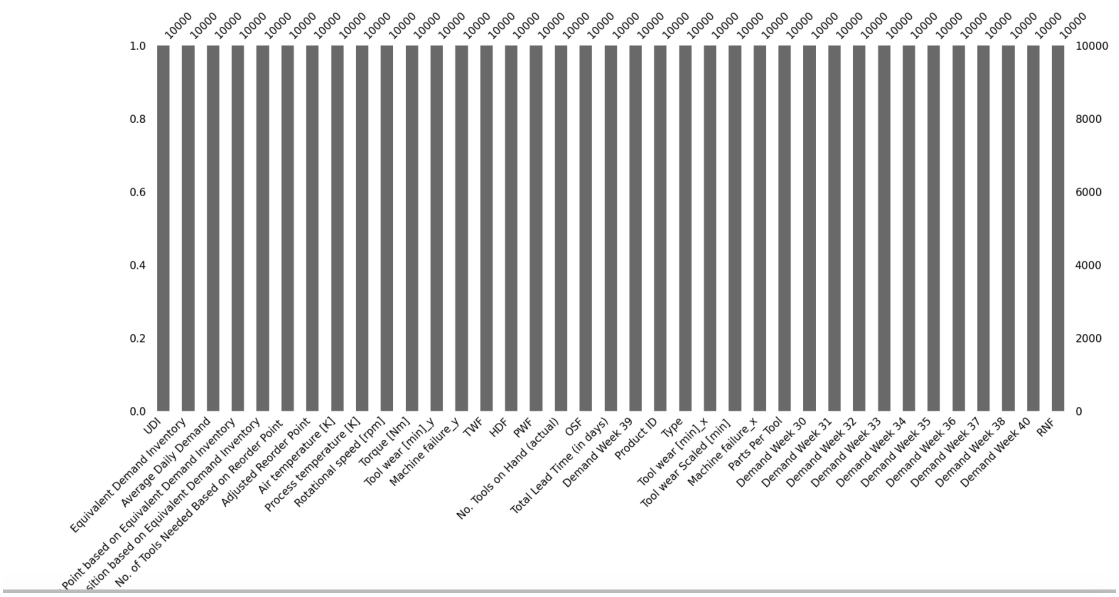
- Data Integration: Combined the primary and supplemental datasets, ensuring consistency and accuracy in the merged data.

```
#Import and merge datasets

import pandas as pd
import numpy as np

# Load the Excel and CSV files
supplemental_data_path = '/Users/jasminemoniquecooper/Downloads/Supplemental Predictive Maintenance Dataset.xlsx'
ai4i2020_data_path = '/Users/jasminemoniquecooper/Downloads/ai4i2020_pmd.csv'

# Load the data
supplemental_data = pd.read_excel(supplemental_data_path, sheet_name='Supplemental Data')
maint_data = pd.read_csv(ai4i2020_data_path)

#View all columns
pd.set_option('display.max_columns', None)

# Merge the two datasets on common columns
merged_data = pd.merge(
    supplemental_data,
    maint_data,
    on=['UDI', 'Product ID', 'Type']
)

# Display the first few rows of the merged dataset
print(merged_data.head())
```

```
   UDI Product ID Type  Tool wear [min]_x  Tool wear Scaled [min]   \
0    1     M14860    M                  0                      0
1    2     L47181    L                  3                    600
2    3     L47182    L                  5                   1000
3    4     L47183    L                  7                   1400
4    5     L47184    L                  9                   1800
```

- Handling Missing or Inconsistent Data: There were no missing or inconsistency data in the core dataset and the supplemental data was developed to ensure completeness and consistency, maintaining data integrity.

```
#ensure I pulled correct data and no missingness
import missingno as msno
import matplotlib.pyplot as plt
column_order = merged_data.isnull().sum().sort_values().index
msno.bar(merged_data[column_order])
plt.show()
```



Justification of Tools and Techniques

Excel was chosen for its accessibility and ease of use for initial data manipulation tasks. It is widely available and familiar, making it an ideal choice for quick data randomization and preliminary calculations. Excel's user-friendly interface allows for rapid data entry, formula application, and simple data manipulations, which is beneficial for creating and randomizing supplemental data efficiently. While Excel is excellent for small to medium-sized datasets, it struggles with more cumbersome tasks like merging data, conducting complex data analysis, and building predictive models. These tasks are often less efficient in Excel compared to more robust data manipulation tools.

Python was selected for its powerful data manipulation capabilities and extensive libraries such as pandas and NumPy, which are essential for importing, handling, cleaning, and merging large datasets. Its versatility and efficiency make it suitable for more complex data preparation tasks. It also supports automation of repetitive tasks which increases efficiency and consistency. The main drawback of Python is creating, and randomizing data can be more complex and time-consuming in Python compared to Excel, where such tasks are more straightforward due to built-in functions.

Using Excel and Python together leverages the strengths of both tools. Excel was ideal for the initial stages of data preparation, particularly for creating and randomizing the supplemental dataset quickly. This allowed for easy visualization and manipulation of data in a familiar environment. Once the preliminary data was prepared, Python was employed to handle larger-scale data integration and complex calculations. This combination ensured that the data preparation process was both efficient and effective, allowing for seamless transition from initial data creation to sophisticated data analysis and modeling.

Overall Summary

This structured approach to data collection, extraction, and preparation ensures that the dataset is comprehensive, realistic, and ready for predictive modeling and Tableau visualization. The combination of publicly available data and domain-specific supplemental data provides a solid foundation for the analysis.

In text citation:

(AI4I 2020 Predictive Maintenance Dataset, 2020)
("Merging datasets", n.d.)
("Importing flat files using pandas", n.d.)
("Analyze the amount of missingness", n.d.)

Analysis

Feature Selection

To start the data-analysis process, I used the Recursive Feature Elimination (RFE) method for feature selection. This method helps to identify the most significant features that contribute to

the predictive model. The dataset was split into training and testing sets to ensure the model's performance could be evaluated on unseen data. The selected features for the analysis were:

- Parts Per Tool
- Demand Week 30 to Demand Week 40
- No. Tools on Hand (actual)
- Average Daily Demand
- No. of Tools Needed Based on Reorder Point
- Adjusted Reorder Point

```
#Feature selection

from sklearn.ensemble import RandomForestRegressor
from sklearn.feature_selection import RFE
from sklearn.model_selection import train_test_split

selected_features = ['Machine failure_x', 'Parts Per Tool',
       'Demand Week 30', 'Demand Week 31', 'Demand Week 32', 'Demand Week 33',
       'Demand Week 34', 'Demand Week 35', 'Demand Week 36', 'Demand Week 37',
       'Demand Week 38', 'Demand Week 39', 'Demand Week 40',
       'Total Lead Time (in days)', 'No. Tools on Hand (actual)',
       'Equivalent Demand Inventory', 'Average Daily Demand',
       'Reorder Point based on Equivalent Demand Inventory',
       'Current Inventory Position based on Equivalent Demand Inventory',
       'No. of Tools Needed Based on Reorder Point ', 'Adjusted Reorder Point',
       'Air temperature [K]', 'Process temperature [K]',
       'Rotational speed [rpm]', 'Torque [Nm]',
       'Machine failure_y', 'TWF', 'HDF', 'PWF', 'OSF', 'RNF']

target = 'Tool wear Scaled [min] '

X = merged_data[selected_features]
y = merged_data[target]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize a random forest regressor
estimator = RandomForestRegressor(random_state=42)

# Initialize RFE with the estimator, and let it choose the number of features automatically
rfe = RFE(estimator)

# Fit RFE on the training data
rfe.fit(X_train, y_train)

# Print the selected features
print("Selected Features:")
for i, feature in enumerate(selected_features):
    if rfe.support_[i]:
        print("-", feature)

# Transform the data to include only the selected features
X_train_rfe = rfe.transform(X_train)
X_test_rfe = rfe.transform(X_test)

# Print the shape of the resulting datasets
print("\nX_train_rfe shape:", X_train_rfe.shape)
print("X_test_rfe shape:", X_test_rfe.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
Selected Features:
- Parts Per Tool
- Demand Week 30
- Demand Week 31
- Demand Week 32
- Demand Week 33
- Demand Week 34
- Demand Week 35
- Demand Week 36
- Demand Week 37
- Demand Week 39
- Demand Week 40
- No. Tools on Hand (actual)
- Average Daily Demand
- No. of Tools Needed Based on Reorder Point
- Adjusted Reorder Point

X_train_rfe shape: (8000, 15)
X_test_rfe shape: (2000, 15)
y_train shape: (8000,)
y_test shape: (2000,)
```

Analysis Technique: Linear Regression

I conducted a linear regression analysis to model the relationship between the selected features and tool wear. The outputs from this analysis were:

- Mean Squared Error (MSE): 122418245.718
- Root Mean Squared Error (RMSE): 11064.27
- R-squared: 0.706

These metrics indicate how well the linear regression model fits the data, with the R-squared value showing that approximately 70.6% of the variability in the tool wear is explained by the selected features in the model.

```python
#Linear Regression Analysis

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Initialize the linear regression model
model = LinearRegression()

# Fit the model on the training data
model.fit(X_train_rfe, y_train)

# Make predictions using the test data
y_pred = model.predict(X_test_rfe)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)
print("R-squared:", r2)
```

```
Mean Squared Error: 122418245.71876074
Root Mean Squared Error: 11064.277912216447
R-squared: 0.7063801803263088
```

Cross Validation

To ensure the robustness of the model, I performed cross-validation to compare RMSE scores. The cross-validation RMSE scores of 5 folds were: 11062.01, 10239.66, 10687.84, 9774.83, and

10979.93. The mean RMSE of the validation set was 10548.85. This validation step confirms the model's stability across difference subsets of the data. Since the RMSE of the validation set and the RMSE of the linear regression model have a small delta of 515.42, it provides confidence that the linear regression model is reliable.

```
#Cross Validation

from sklearn.model_selection import cross_val_score
from sklearn.metrics import make_scorer

# Define a function to calculate RMSE (since cross_val_score by default uses negative MSE)
def rmse(y_true, y_pred):
    return np.sqrt(mean_squared_error(y_true, y_pred))

# Perform cross-validation with RMSE as the scoring metric
scores = cross_val_score(model, X, y, cv=5, scoring=make_scorer(rmse))

# Print RMSE scores for each fold and the mean RMSE across all folds
print("Cross-Validation RMSE Scores:", scores)
print("Mean RMSE:", scores.mean())
```

```
Cross-Validation RMSE Scores: [11062.01849498 10239.66111718 10687.84807057  9774.8383211
 10979.93022778]
Mean RMSE: 10548.859246323424
```

Ordinary Least Squares (OLS) Regression

To further understand the significant of each feature, I performed an OLS regression analysis. P-values were extracted from this analysis to address the hypothesis. P-values less than 0.05 indicate statistically significant features. From this analysis, significant features include Parts Per Tool, Demand Week 35, Demand Week 40, No. Tools on Hand (actual), No. of Tools Needed Based on Reorder Point, and Adjusted Reorder Point. These features indicate they have a meaningful relationship with tooling failure. The hypothesis states that "Demand and parts-per-tool usage statistically affect tooling (or equipment) failure". The significant features identified support this hypothesis, indicating that demand (specifically weeks 30 to 40) and parts-per-tool usage indeed play a role in tooling failure.

```
#Use statsmodels to perform detailed analysis including p-values
#Use to answer research question / discuss hypothesis

import statsmodels.api as sm

#Selected features
selected_feature_names = [
    'Parts Per Tool', 'Demand Week 30', 'Demand Week 31', 'Demand Week 32', 'Demand Week 33', 'Demand Week 34',
    'Demand Week 35', 'Demand Week 36', 'Demand Week 37', 'Demand Week 39', 'Demand Week 40',
    'No. Tools on Hand (actual)', 'Average Daily Demand', 'No. of Tools Needed Based on Reorder Point',
    'Adjusted Reorder Point'
]

# Add a constant to the independent variables matrix for intercept
X_train_const = sm.add_constant(X_train_rfe)

# Fit OLS (Ordinary Least Squares) model
model_sm = sm.OLS(y_train, X_train_const)
results = model_sm.fit()

# Print summary with p-values
print(results.summary())

# Get coefficients and their associated feature names
coefficients = results.params[1:]  # Exclude the intercept
pvalues = results.pvalues[1:]      # Exclude the intercept's p-value

print("\nCoefficients:")
for i, (coef, pval) in enumerate(zip(coefficients, pvalues)):
    if i < len(selected_feature_names):
        feature_name = selected_feature_names[i]
        print(f"{feature_name}: {coef:.4f} (p-value: {pval:.4f})")
    else:
        print(f"Index {i} exceeds the length of selected_feature_names.")

# Interpretation
print("\nSignificance level (α) = 0.05")
print("P-values less than 0.05 indicate statistically significant results.")
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     Tool wear Scaled [min]    R-squared:                       0.711
Model:                                OLS   Adj. R-squared:                  0.710
Method:                     Least Squares   F-statistic:                     1402.
Date:                    Sun, 07 Jul 2024   Prob (F-statistic):               0.00
Time:                            12:38:51   Log-Likelihood:                -85584.
No. Observations:                    8000   AIC:                         1.712e+05
Df Residuals:                        7985   BIC:                         1.713e+05
Df Model:                              14
Covariance Type:                nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       6658.6621    753.698      8.835      0.000    5181.218    8136.106
x1             2.5883      0.019    133.693      0.000       2.550       2.626
x2             0.0042      0.012      0.361      0.718      -0.019       0.027
x3            -0.0203      0.012     -1.721      0.085      -0.043       0.003
x4             0.0040      0.012      0.341      0.733      -0.019       0.027
x5            -0.0093      0.012     -0.797      0.426      -0.032       0.014
x6            -0.0153      0.012     -1.304      0.192      -0.038       0.008
x7            -0.0194      0.012     -1.664      0.096      -0.042       0.003
x8            -0.0293      0.012     -2.468      0.014      -0.053      -0.006
x9            -0.0157      0.012     -1.335      0.182      -0.039       0.007
x10           -0.0173      0.012     -1.476      0.140      -0.040       0.006
x11           -0.0246      0.012     -2.098      0.036      -0.048      -0.002
x12           -1.9134      0.067    -28.646      0.000      -2.044      -1.782
x13            0.5868      0.457      1.285      0.199      -0.308       1.482
x14           -0.9738      0.067    -14.482      0.000      -1.106      -0.842
x15            0.9396      0.042     22.278      0.000       0.857       1.022
==============================================================================
Omnibus:                     3210.527   Durbin-Watson:                   2.030
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14868.640
Skew:                           1.928   Prob(JB):                         0.00
Kurtosis:                       8.453   Cond. No.                     8.86e+16
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 6.73e-21. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

```
Coefficients:
Parts Per Tool: 2.5883 (p-value: 0.0000)
Demand Week 30: 0.0042 (p-value: 0.7182)
Demand Week 31: -0.0203 (p-value: 0.0853)
Demand Week 32: 0.0040 (p-value: 0.7334)
Demand Week 33: -0.0093 (p-value: 0.4255)
Demand Week 34: -0.0153 (p-value: 0.1923)
Demand Week 35: -0.0194 (p-value: 0.0962)
Demand Week 36: -0.0293 (p-value: 0.0136)
Demand Week 37: -0.0157 (p-value: 0.1819)
Demand Week 39: -0.0173 (p-value: 0.1399)
Demand Week 40: -0.0246 (p-value: 0.0360)
No. Tools on Hand (actual): -1.9134 (p-value: 0.0000)
Average Daily Demand: 0.5868 (p-value: 0.1988)
No. of Tools Needed Based on Reorder Point: -0.9738 (p-value: 0.0000)
Adjusted Reorder Point: 0.9396 (p-value: 0.0000)

Significance level (α) = 0.05
P-values less than 0.05 indicate statistically significant results.
```

Linear Regression Analysis on the Entire Dataset & Handling Negative Predictions

After validating the robustness of the model, I re-conducted linear regression analysis on the entire dataset. The outputs from this analysis were:

- Mean Squared Error: 116232582.26
- Root Mean Squared Error: 10781.12

- R-squared: 0.710

The model evaluation metrics are like those of the original linear regression analysis with the train and test split and the cross-validation set, indicating that the model on the entire dataset is robust.

While analyzing the predictions, I noticed that some predicted tool wear values were negative, which is not practical. To address this, I calculated the average predicted tool wear based on tool type and replaced negative values with these averages, ensuring no negative values were present in the final dataset.

```
from sklearn.linear_model import LinearRegression
import pandas as pd

# Entire dataset is merged_data with all features and target
X = merged_data[['Parts Per Tool', 'Demand Week 30', 'Demand Week 31', 'Demand Week 32', 'Demand Week 33',
                 'Demand Week 34',
    'Demand Week 35', 'Demand Week 36', 'Demand Week 37', 'Demand Week 39', 'Demand Week 40',
    'No. Tools on Hand (actual)', 'Average Daily Demand', 'No. of Tools Needed Based on Reorder Point ',
    'Adjusted Reorder Point']]
y = merged_data['Tool wear Scaled [min] ']

# Initialize the linear regression model
model = LinearRegression()

# Fit the model on the entire dataset
model.fit(X, y)

# Make predictions using the entire dataset
predictions_full = model.predict(X)

# Create a DataFrame for predictions
predictions_df = pd.DataFrame({
    'Product ID': merged_data['Product ID'],  # Replace with actual identifier column
    'Predicted_Tool_wear_Scaled': predictions_full
})

# Optionally set to view all rows
pd.set_option('display.max_rows', None)

# Append Tool_Type column from merged_data to predictions_df
predictions_df['Type'] = merged_data['Type']

# Print predictions_df
print(predictions_df)
```

	Product ID	Predicted_Tool_wear_Scaled	Type
0	M14860	538.658809	M
1	L47181	-962.800533	L
2	L47182	2254.241922	L
3	L47183	4318.212668	L
4	L47184	6733.095070	L
5	M14865	4686.224561	M
6	L47186	-3046.074386	L
7	L47187	4170.836875	L
8	M14868	7406.787775	M
9	M14869	5413.443149	M
10	H29424	9965.754911	H
11	H29425	11648.333231	H
12	M14872	8224.417303	M
13	M14873	13610.823800	M
14	L47194	7127.071143	L
15	L47195	6800.267855	L
16	M14876	14407.929188	M
17	M14877	12784.342254	M
18	H29432	17011.864961	H
19	M14879	16613.130355	M
20	H29434	16440.509044	H
21	L47201	21853.052959	L
22	M14882	17781.952272	M
23	L47203	15977.302392	L

Since there are 10,000 tools in the dataset, this is a partial output of the predictions on the entire dataset. All outputs will be included in the submission of the Python notebook.

```
#model evaluation metrics for full predictions on full dataset

from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score

# Calculate Mean Squared Error (MSE)
mse = mean_squared_error(y, predictions_full)

# Calculate Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)

print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)

# Calculate R-squared
r2 = r2_score(y, predictions_full)
print("R-squared:", r2)
```

```
Mean Squared Error: 116232582.2669682
Root Mean Squared Error: 10781.1215681379
R-squared: 0.7100346333787111
```

```python
#Determine number of rows with negative tool wear in min

import numpy as np

# Assuming total dataset size is 10000 records
total_dataset_size = 10000

# Find indices of rows where predictions are negative
negative_indices = np.where(predictions_full < 0)[0]

# Calculate count of negative predictions
negative_count = len(negative_indices)

# Calculate percentage of negative predictions relative to total dataset
negative_percentage = (negative_count / total_dataset_size) * 100

# Display results
print(f"Number of rows with negative predictions: {negative_count} ({negative_percentage:.2f}% of total)")
```

```
Number of rows with negative predictions: 315 (3.15% of total)
```

```python
#determine the average predicted tool wear based on tool type

# Group by Type and calculate average predicted tool wear
average_predicted_wear = predictions_df.groupby('Type')['Predicted_Tool_wear_Scaled'].mean()

# Convert series to DataFrame
average_predicted_wear_df =
average_predicted_wear.reset_index().rename(columns={'Predicted_Tool_wear_Scaled': 'Average_Predicted_Tool_Wear'})

# Print or display the average_predicted_wear_df
print("Average Predicted Tool Wear by Tool Type:")
print(average_predicted_wear_df)
```

```
Average Predicted Tool Wear by Tool Type:
  Type  Average_Predicted_Tool_Wear
0    H                 30958.794098
1    L                 26867.104467
2    M                 29402.670242
```

```python
#Replace negative predicted tool wear with the avg predicted tool wear based on tool type

import numpy as np
import pandas as pd

# Ensure 'Type' is the index in average_predicted_wear_df for easy lookup
average_predicted_wear_df.set_index('Type', inplace=True)

# Create a copy of predictions_df to make modifications
predictions_updated_df = predictions_df.copy()

# Function to apply the average based on 'Type' when the value is negative
def replace_with_average(row):
    if row['Predicted_Tool_wear_Scaled'] < 0:
        return average_predicted_wear_df.loc[row['Type'], 'Average_Predicted_Tool_Wear']
    else:
        return row['Predicted_Tool_wear_Scaled']

# Apply the function to replace negative values
predictions_updated_df['Predicted_Tool_wear_Scaled'] = predictions_updated_df.apply(replace_with_average, axis=1)

# Print the modified predictions_updated_df
print(predictions_updated_df)
```

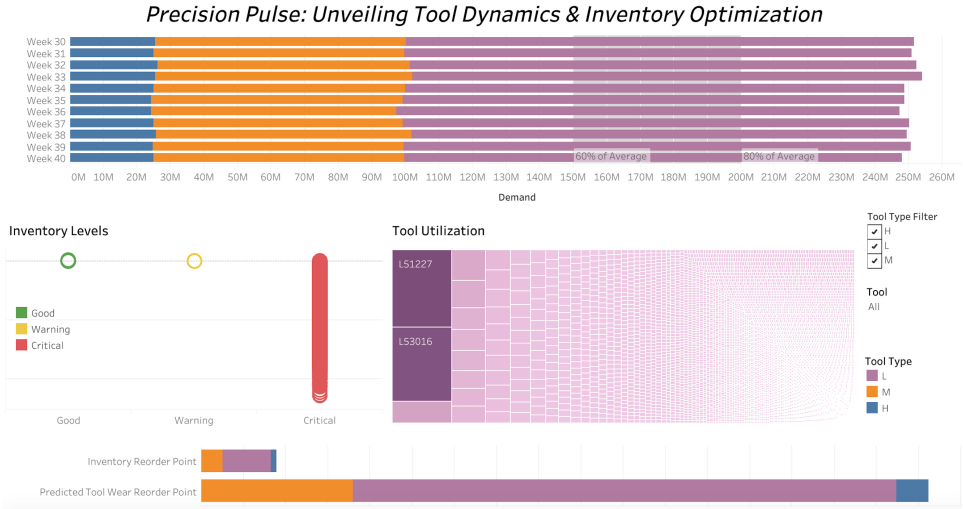
	Product ID	Predicted_Tool_wear_Scaled	Type
0	M14860	538.658809	M
1	L47181	26867.104467	L
2	L47182	2254.241922	L
3	L47183	4318.212668	L
4	L47184	6733.095070	L
5	M14865	4686.224561	M
6	L47186	26867.104467	L
7	L47187	4170.836875	L
8	M14868	7406.787775	M
9	M14869	5413.443149	M
10	H29424	9965.754911	H
11	H29425	11648.333231	H
12	M14872	8224.417303	M
13	M14873	13610.823800	M
14	L47194	7127.071143	L
15	L47195	6800.267855	L
16	M14876	14407.929188	M
17	M14877	12784.342254	M
18	H29432	17011.864961	H
19	M14879	16613.130355	M
20	H29434	16440.509044	H
21	L47201	21853.052959	L
22	M14882	17781.952272	M
23	L47203	15977.302392	L
24	M14884	23432.507603	M

Since there are 10,000 tools in the dataset, this is a partial output of the predictions with updated average predictions to replace negative predictions. All outputs will be included in the submission of the Python notebook.

Tableau Dashboard Creation

In addition to the statistical analysis conducted, a Tableau dashboard was created to visualize key metrics and trends identified during the analysis. This dashboard provides an interactive interface where stakeholders can view demand fluctuations, inventory levels, tool capacity, predicted tool wear, and inventory reorder points in real-time. By integrating these visual insights, decision makers can better understand the implications, ensuring that inventory management strategies are responsive to actual demand patterns.

The dashboard highlights critical findings, such as variability in demand illustrates by the distribution bands, which can range from 150M to 200M for the 60% average, and 200M to 230M for the 80% average. This visualization reinforces the need for a dynamic inventory management system that adjusts reorder points based on real-time data.



Justification of Analysis Techniques

Recursive Feature Elimination systematically selects the most relevant features, improving model performance and interpretability. By focusing on the most significant features, RFE helps reduce overfitting and enhances the model's predictive power. However, it can be computationally intensive, especially with large datasets and complex models, which may limit its practicality in certain situations.

Linear regression provides a simple and interpretable model, making it easy to understand the relationships between variables. This transparency is beneficial for stakeholders who need to grasp the impact of different features quickly. However, a drawback of this analysis is that it assumes a linear relationship between variables, which may not always hold true in real-words data. This assumption can lead to biased or inaccurate predictions if the underlying relationships are non-linear.

Cross-validation offers a more robust evaluation of the model's performance by averaging results across multiple data splits. This technique helps ensure that the model generalizes well to unseen data and reduced the likelihood of overfitting. However, cross-validation can be time-consuming and computationally expensive, particularly with large datasets, which may delay the analysis process and require additional computational resources.

Tableau offers an intuitive interface and powerful capabilities for creating interactive and visually appealing dashboards. Tableau allows for real-time data exploration, making it easy to identify trends, patterns, and outliers, which is essential for effective decision-making. The ability to quickly visualize complex data sets enhances interpretability and facilitates communication of insights to stakeholders. However, one disadvantage of Tableau is that it may require supplementary analysis in more advanced statistical context, which could lead to potential gaps in analysis if not used alongside other analytical methods.

The analysis process involved careful feature selection, regression modeling, and validation to ensure the accuracy and reliability of the predictions. By addressing issues like negative predictions and using multiple evaluation techniques, the final model provides valuable insights into tool wear and helps optimize maintenance schedules and inventory management.

In text citations:

(“Importing flat files using pandas”, n.d.)
(“Analyze the amount of missingness”, n.d.)
(“Mean, median, & mode imputations”, n.d.)
(“More than explanatory variables”, n.d.)
(“Selecting features for model performance”, n.d.)
(“Working with model objects”, n.d.)
(“Making predictions”, n.d.)
(Quantifying model fit”, n.d.)
(“Tableau: adding lines and distribution bands”, n.d.)
(“Creating dashboards”, n.d.)

("Adding elements to the dashboard", n.d.)
("Dashboard objects and actions", n.d.)
("Dashboard interactivity", n.d.)
("Sharing data insights", n.d.)
("Advanced manipulations with Story Points", n.d.)
("Calculated Fields to extend data", n.d.)
("Visualizations for exploratory analysis of trends", n.d.)
("Slicing and dicing", n.d.)
("Make your data visually appealing", n.d.)
("Dashboard and stories", n.d.)

Data Summary and Implications

The research question focused on understanding the extent to which demand and parts-per-tool usage affect tooling (or equipment) failure. To answer this question, we conducted a linear regression analysis, evaluated the model's performance using various metrics, and developed a Tableau dashboard for visualization.

Selected Features and Model Performance

The selected features as a result of the analysis were: Parts Per Tool, Demand Week 30 to Week 40, No. Tools on Hand (actual), Average Daily Demand, No. of Tools Needed Based on Reorder Point, and Adjusted Reorder Point. These features represent the most significant features that contributed to the model.

The model evaluation metrics, Mean Squared Error (MSE): 122418245.718, Root Mean Squared Error (RMSE): 11064.27, and R-squared: 0.706 were calculated to determine the model performance. The R-squared value of 0.706 indicates that our model explains approximately 70.6% of the variance in tool wear, suggesting a good fit. However, the large value RMSE of 11,064.28 indicates a significant prediction error, which might require further analysis.

The cross-validation RMSE scores varied across different folds but maintain a mean RMSE of 10,548.86. Although the RMSE score suggests a potential prediction error, the consistency between the cross-validation and single train-test split RMSE suggests that the model's performance is stable across different data subsets, indicating good prediction accuracy.

Significance of Selected Features

Using the OLS regression results, the impact of each selected feature on tool wear was assessed. Selected features with p-values less than 0.05 were deemed statistically significant. These features include Parts Per Tool, No. Tools on Hand (actual), Adjusted Reorder Point, Average Daily Demand, Demand Week 36, and Demand Week 40. This suggests that while overall demand impacts tool wear, specific weeks may have a strong individual effect. Practically, peaks in demand could lead to increased tool wear failure, which should be considered in inventory management strategies when anticipating forecasted demand spikes.

Answering the Research Question & Addressing the Hypothesis

The research question aimed to determine the extent to which demand and parts-per-tool usage affect tooling (or equipment) failure. The analysis provided compelling evidence that both factors play a significant role in predicting tool wear and subsequent failures.

Results indicate that the hypothesis that demand and parts-per-tool usage significantly affect tooling failure is supported by the analysis. The statistically significant predictors of tooling Parts Per Tool, and Demand Week 30 to Demand Week 40 confirm their strong impact on tool wear. This suggests that higher parts-per-tool usage correlates with increased wear and higher likelihood of failure. Additionally, the demand spikes observed during this period further wear on tools, indicating a relationship between demand intensity and tooling performance.

In conclusion, the analysis robustly confirms that demand and parts-per-tool usage significantly affect tooling failure. This insight underscores the critical need for organizations to adopt dynamic inventory management systems that account for demand variability and tool usage patterns, ultimately enhancing operational efficiency and reducing downtime.

Key Results of Tableau Dashboard Analysis

Based on the creation of the Tableau dashboard some key results were observed:

1. Predicted tool wear reorder points are significantly larger than Inventory reorder points. This discrepancy suggests high variability in inventory management and poses a challenge to maintaining critical inventory levels effectively.
2. There is high variability in the demand, which likely drives the large variability in reorder points, and the need for higher inventory on hand. Distribution bands were applied to the demand to observe variability. The 60% average band ranges from 150M to 200M, while the 80% average band extends from approximately 200M - 230M.
3. Most demand is concentrated on tool type L, which also shows the most variability between inventory reorder points and predicted tool wear reorder points. Additionally, tool type L is the most utilized among all tools.
4. Maintaining critical inventory levels remains essential despite predictions based on tool wear, emphasizing the importance of robust inventory management strategies.

Recommendations

Based on the linear regression and Tableau dashboard analysis, here are the recommended courses of action:

1. Implement a Dynamic Inventory Management System: Adjust reorder points in real-time based on demand, parts-per-tool usage, and predictive tool wear forecasts.
2. Develop Dynamic Inventory Policies: Create policies that adjust reorder points based on demand variability bands, ensuring preparedness for fluctuating demand and reducing risks of overstocking or stockouts.
3. Prioritize Tool Type L: Focus on monitoring and replenishing inventory for tool type L, which shows high demand, variability, and utilization. Implement predictive

maintenance schedules specifically for this tool type to ensure optimal availability and performance.

Limitation

One key limitation of the analysis is the reliance on artificially created historical data, which may not accurately reflect real-world demand patterns, lead times or tool usage. This could impact predictions and the robustness of the findings.

Future Directions

1. Alternative Modeling Techniques: Explore hybrid models that integrate linear regression with machine learning techniques like random forests or gradient boosting. This could help capture both linear relationships and complex interactions. These models often provide better predictive accuracy and robustness, especially in larger and more complex datasets, which can be ideal for real-world scenarios. Additionally, by changing randomized supplemental data to include dates, time series forecasting techniques can be used to predict tool wear and demand based on historical patterns. This method could also provide more accurate predictions over time.
2. Collaboration and Integration: Integrate the predictive model with Enterprise Resource Planning (ERP) systems to streamline operations and provide real-time insights for decision-makers is valuable. After integration, foster collaboration between departments such as production, maintenance, and supply chain to provide feedback and share insights and data, leading to more comprehensive and holistic predictive maintenance strategies.

By leveraging organizational synergies and alternative modeling techniques, future studies can build on our findings to develop more accurate and reliable predictive models, leading to improved operational efficiency.

Conclusion

In conclusion, the finding from the analysis highlight the statistically significant relationship between demand, parts-per-tool, and tooling failure. Implementing the recommended strategies will enhance inventory management and predictive maintenance efforts. Future research directions aim to refine modeling techniques and promote interdepartmental collaboration, ultimately contributing to a more resilient operational framework.

In text citations:

(Quantifying model fit", n.d.)
("Selecting features for model performance", n.d.)
("Working with model objects", n.d.)
("Making predictions", n.d.)
("Tableau: adding lines and distribution bands", n.d.)
("Sharing data insights", n.d.)
("Visualizations for exploratory analysis of trends", n.d.)
("Slicing and dicing", n.d.)

(“Make your data visually appealing”, n.d.)
(“Dashboard and stories”, n.d.)

Citations for code:

DataCamp. (n.d.). Importing flat files using pandas [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-importing-data-in-python/introduction-and- flat-files-1?ex=15

DataCamp. (n.d.). Analyze the amount of missingness [Video file]. Retrieved from https://campus.datacamp.com/courses/dealing-with-missing-data-in-python/the-problem-with-missing- data?ex=9

DataCamp. (n.d.). More than explanatory variables [Video file]. Retrieved from https://campus.datacamp.com/courses/intermediate-regression-with-statsmodels-in-python/multiple-linear-regression-3?ex=7

DataCamp. (n.d.). Mean, median, & mode imputations [Video file]. Retrieved from https://campus.datacamp.com/courses/dealing-with-missing-data-in-python/imputation-techniques?ex=1

DataCamp. (n.d.). Making predictions [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/predictions-and-model-objects-2?ex=1

DataCamp. (n.d.). Working with model objects [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/predictions-and-model-objects-2?ex=5

DataCamp. (n.d.). Quantifying model fit [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/assessing-model-fit-e78fd9fe-6303-4048-8748-33b19c4222fe?ex=1

DataCamp. (n.d.). Selecting features for model performance [Video file]. Retrieved from https://campus.datacamp.com/courses/dimensionality-reduction-in-python/feature-selection-ii- selecting-for-model-accuracy?ex=1

DataCamp. (n.d.). Creating dashboards [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/getting-started-with-dashboards?ex=1

DataCamp. (n.d.). Adding elements to the dashboard [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/getting-started-with-dashboards?ex=7

DataCamp. (n.d.). Dashboard objects and actions [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/getting-started-with-dashboards?ex=10

DataCamp. (n.d.). Dashboard interactivity [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/getting-started-with-dashboards?ex=15

DataCamp. (n.d.). Sharing data insights [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/sharing-data-insights?ex=1

DataCamp. (n.d.). Advanced manipulations with Story Points [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/sharing-data-insights?ex=6

DataCamp. (n.d.). Calculated Fields to extend data [Video file]. Retrieved from https://campus.datacamp.com/courses/analyzing-data-in-tableau/preparing-for-analysis?ex=6

DataCamp. (n.d.). Visualizations for exploratory analysis of trends [Video file]. Retrieved from https://campus.datacamp.com/courses/analyzing-data-in-tableau/preparing-for-analysis?ex=10

DataCamp. (n.d.). Slicing and dicing [Video file]. Retrieved from https://campus.datacamp.com/courses/analyzing-data-in-tableau/preparing-for-analysis?ex=15

DataCamp. (n.d.). Make your data visually appealing [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-tableau/presenting-your-data?ex=1

DataCamp. (n.d.). Dashboards and stories [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-tableau/presenting-your-data?ex=7

DataCamp. (n.d.). Merging datasets [Video file]. Retrieved from h https://campus.datacamp.com/courses/analyzing-police-activity-with-pandas/analyzing-the-effect-of-weather-on-policing?ex=8

AI4I 2020 Predictive Maintenance Dataset. (2020). *UCI Machine Learning Repository.* https://archive.ics.uci.edu/dataset/601/ai4i+2020+predictive+maintenance+dataset

Citations for content:

DataCamp. (n.d.). Making predictions [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/predictions-and-model-objects-2?ex=1

DataCamp. (n.d.). Working with model objects [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/predictions-and-model-objects-2?ex=5

DataCamp. (n.d.). Quantifying model fit [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/assessing-model-fit-e78fd9fe-6303-4048-8748-33b19c4222fe?ex=1

DataCamp. (n.d.). Selecting features for model performance [Video file]. Retrieved from https://campus.datacamp.com/courses/dimensionality-reduction-in-python/feature-selection-ii- selecting-for-model-accuracy?ex=1

DataCamp. (n.d.). Sharing data insights [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/sharing-data-insights?ex=1

DataCamp. (n.d.). Visualizations for exploratory analysis of trends [Video file]. Retrieved from https://campus.datacamp.com/courses/analyzing-data-in-tableau/preparing-for-analysis?ex=10

DataCamp. (n.d.). Make your data visually appealing [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-tableau/presenting-your-data?ex=1

DataCamp. (n.d.). Creating dashboards [Video file]. Retrieved from https://campus.datacamp.com/courses/creating-dashboards-in-tableau/getting-started-with-dashboards?ex=1

DataCamp. (n.d.). Tableau: adding lines and distribution bands [Video file]. Retrieved from https://campus.datacamp.com/courses/statistical-techniques-in-tableau/measures-of-spread-and-confidence-intervals?ex=10

AI4I 2020 Predictive Maintenance Dataset. (2020). *UCI Machine Learning Repository.* https://archive.ics.uci.edu/dataset/601/ai4i+2020+predictive+maintenance+dataset