Project #1

Step 2: Data Visualization

After performing statistical data analysis on the dataset, the visualized plot is shown in Figure 1.   
A line graph with different colored lines

Description automatically generated

Figure 1: X,Y,Z Values Plotted Against Step

The X value (blue line) starts steady at 9, then starts falling as drastically as Step increases in value. The Y value (orange line) starts steady at 3, wavers around 5 and 6 then returns back to 3. And finally, Z value (green line) has a stead rate of climb from 1 to 2 with some major wavering between steps 6-9 but evens out afterwards.

Step 3: Correlation Analysis

Figure 2 shows the correlation between each variable. The heatmap shows that X and Step have a high positive correlation (close to 1), suggesting that X is a good predictor for Step. The closer the value of correlation is to 1 the better the correlation.

For example, Y and Step, and Z and Step are not very good predictors for Step as their correlation is very low. Even worse would be the correlation between X, Y, and Z which are almost near zero, which makes sense as they are not key predictors for one another as they are independent variables.

A colorful squares with black letters

Description automatically generated with medium confidence

Figure 2: Correlation Matrix

Step 4: Classification Model Development

The classification models used in this project are:

1. **Logistic Regression**
2. **K-Neighbors Classifier**
3. **Random Forest Classifier**

The reason why Logistic Regression was chosen is because it is usually used for one or more independent variable’s which is applicable in this case. But since it is mainly used for linear relationships, it may underperform compared to other models.

KNeighbors would be important for non-linear data with its regression being determined by its neighboring points making it perform better than standard linear models.

Random Forest classifier is a supervised classification method that provides better accuracy and generalization and reduces overfitting.

The choice between these three classification models mainly comes from choosing an easier to more complex classifier to help better understand the main computations that occur to arrange the data. The results are shown in the table below.

|  |  |
| --- | --- |
| **Classifier** | **Accuracy Score** |
| Logistic Regression Accuracy | 0.95348 |
| Random Forest Accuracy | 0.97093 |
| KNeighbors Accuracy | 0.97674 |
| CV Accuracy | 0.96511 |

Table 1: Classifier and its Accuracy Score

The highest performing classifier in the list appeared to be the K-Neighbors followed by Random forrest, then CV Accuracy and finally Logistic Regression.

As mentioned earlier, the logistic regression is a linear model dealing with a nonlinear target variable, hence why its score appears to be the lowest. K-Neighbors makes decisions based on the similarity of nearby data points. Its very effective if the classes are well-separated in feature space, also its highly influence by the choice of distance metric, the more it aligns with the data structure the higher the accuracy.

Step 5: Model Performance Analysis

F1 Score is a parameter measured on precision and recall, any trade off between the two would lower the score. The outcome of the data is shown in the Table 2 below.

|  |  |
| --- | --- |
| **Classifier** | **F1 Score** |
| Logistic Regression Accuracy | 0.95351 |
| Random Forest Accuracy | 0.97048 |
| KNeighbors Accuracy | 0.97601 |
| CV Accuracy | 0.96434 |

Table 2: Classifier and its F1 Score

Since the data is not linear, once again the score is reflected with the Logistic regression classifier depicting the lowest score of the four. Random forest is great with handling both precision and recall and hence why its score is close to best. Since KNeighbors can make decisions based on local information it allows for higher recall and better precision and hence the best score. CV Accuracy has a good score but not the best, lower F1 score here could be due to variability across folds where some subsets of data are harder to classify.

The next step was to create a confusion matrix for each classifier to help understand the overall performance of a classification model by comparing the target values with predicted values. The following three figures show the confusion matrix for each classification model.

A graph of numbers and a graph

Description automatically generated

Figure 3: Logistic Regression Confusion Matrix

A graph of numbers and a graph

Description automatically generated

Figure 4: Random Forrest Confusion Matrix

A graph of numbers and a graph

Description automatically generated

Figure 5: KNeighbors Confusion Matrix

Comparing all three confusion matrices it is evident that some high and correct predictions were made for classes 7, 8, and 9. All models seem to perform quite similarly, with mostly correct classifications across classes. However, Random Forest seems to have fewer misclassifications compared to KNeighbors, especially in class 0.

Step 6: Stacked Model Performance Analysis

The stacking Classifier combined RandomForrest with KNeighbors classifier achieving an accuracy of 0.9709 suggesting that this approach is very effective at making predictions. The F1 score appears to be 0.9704 indicating that this model is performing very well in both precision and recall. This happens to model diversity allowing to combine global generalization of random forest with local specialization of KNeighbors which leads to better predictions.

A graph of a number and a number

Description automatically generated with medium confidence

Figure 6: Stacking Classifier Confusion Matrix

Step 7: Model Evaluation

Given the data set   
[9.375,3.0625,1.51], [6.995,5.125,0.3875], [0,3.0625,1.93], [9.4,3,1.8], [9.4,3,1.3]

The predicted steps given are  
Input: [9.375 3.0625 1.5], Predicted Maintenance Step: 5

Input: [6.995 5.125 0.3875], Predicted Maintenance Step: 8

Input: [0 3.0625 1.93], Predicted Maintenance Step: 13

Input: [9.4 3. 1.8], Predicted Maintenance Step: 6

Input: [9.4 3. 1.3], Predicted Maintenance Step: 4