Toronto Metropolitan University

Department of Aerospace Engineering
Faculty of Engineering & Architectural Science

Semester (Term, Year)	Fall, 2023
Course Code	AER850
Course Section	1
Course Title	Intro to Machine Learning
Course Instructor	Dr. Reza Faieghi
Submission	Project
Submission No.	1
Submission Due Date	October 15, 2023
Title	Machine Learning Project 1
Submission Date	October 15, 2023

Submission by (Name):	**Student ID (XXXX1234)**	**Signature**
Nima Dorali-Beni	xxxx63711	N.D.

By signing the above you attest that you have contributed to this submission and confirm that all work you contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a ”0” on the work, and “F” in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Academic Integrity Policy 60, which can be found at www.torontomu.ca/senate/policies/

AER850: Intro to Machine Learning

Project 1 Report

Nima Dorali-Beni

AER850

Oct 15, 2023

Dr. Reza Faieghi

GitHub Link:
nd-AER850-P1
https://github.com/nimadb/nd-AER850-P1.git

1.0 Background & Introduction

The application of Augmented Reality (AR) in the aerospace industry, particularly in the domains of manufacturing and maintenance, is an ongoing and rapidly developing field of study. This project is dedicated to enhancing AR-based instruction modules with predictive machine learning (ML) algorithms to improve efficiency and accuracy. The purpose is for the student to understand the process of an ML project and to develop their skills in python.
The primary focus of this project is to create a micro-level coordinate system-based ML model that can optimize AR-based animation techniques. These models can empower technicians in aerospace, allowing them to gain better insights while performing tasks such as aircraft maintenance, computer-aided design and modeling, part assembly, and routine maintenance for space-related applications.

2.0 Code Outline

This code developed is divided into several stages, each contributing to the development and evaluation of ML models for predicting maintenance steps based on given coordinates. A full code-walkthrough is not provided; however, a general walkthrough of each section of the code is provided to align the reader between the project description and the developed code.

2.1 Step 0: Configuration

In this step, necessary libraries are imported and configurations are set.

2.2 Step 1: Data Import and Splitting

Data is loaded from a CSV file, organized, and initial data checks are performed. We ensure that there are no null values in the dataset. The features (X, Y, Z) and the target variable (maintenance step) are separated. The data is then split into training and testing sets.

2.3 Step 2: Visualization and Data Analysis

In this step, the dataset is analyzed visually and statistically. Initial insights are obtained through pair plots and 3D visualization. Data is also grouped by maintenance steps, and statistical analysis and visualization are performed for each group.

2.4 Step 3: Correlation Analysis

This step explores the correlation between the features (X, Y, Z) and the target variable (maintenance step). The analysis shows that there is no significant correlation in the grouped data.

2.5 Step 4/5: Classification Model Development/Engineering

Three classification models are developed: Logistic Regression, Decision Tree, and Random Forest. Each model is fine-tuned using hyperparameter tuning via grid search. Model training, prediction, and evaluation are carried out for each model. The best-performing models are evaluated based on accuracy, classification reports, and confusion matrices. The Logistic Regression model achieved an accuracy of 0.99, the Decision Tree model achieved 0.988, and the Random Forest model achieved 0.97 on the test data.

2.6 Step 6: Model Evaluation

The best-performing models are saved in a joblib format and can be used to predict maintenance steps based on provided coordinates. For a sample set of coordinates, the models predict the maintenance steps.

3.0 Observations

This section highlights the main findings from each step of the project.
After being imported as a data frame, the data is printed; the print does not reveal much except for the columns and number of rows of data. This is further described using the info function. The info function does specify 860 "non-null" rows. A simple isna…any…sum function confirms this; there are no empty values in the entire data frame. The info function also revealed that the 'X', 'Y', and 'Z' columns are floating point values while the step column is integers. This will be important as it specifies that the desired prediction variable, the 'Step' variable, is an integer. This means that a model that will generate decimal predictions will not be suitable. Then, using a value_counts function, it is shown that there are more data points for steps 7,8, and 9. Steps 1-6, and 10-13 each have 24 data points while step 7 has 148, step 8 has 221, and step 9 has 251 data points.

Next, the data was split into training and test sets before visualization. This was done to prevent snooping on the testing data set. The next set of observations were made in the visualization step. To start, a seaborn pairplot (same as a scatter matrix) was made. This plot looks is shown in the figure below. The diagonals (the histograms) show that the X and Y values are skewed to either sides, the Z values are spread more evenly, and the Step values are roughly even except for steps 7, 8, and 9. The other plots show that the relationships between the X, Y, and Z data points are somewhat linear. Although they're linear, they are not continuous, they more resemble a piece-wise function. This signifies that the data needs to be segmented. The descriptive statistics were thus not valuable.

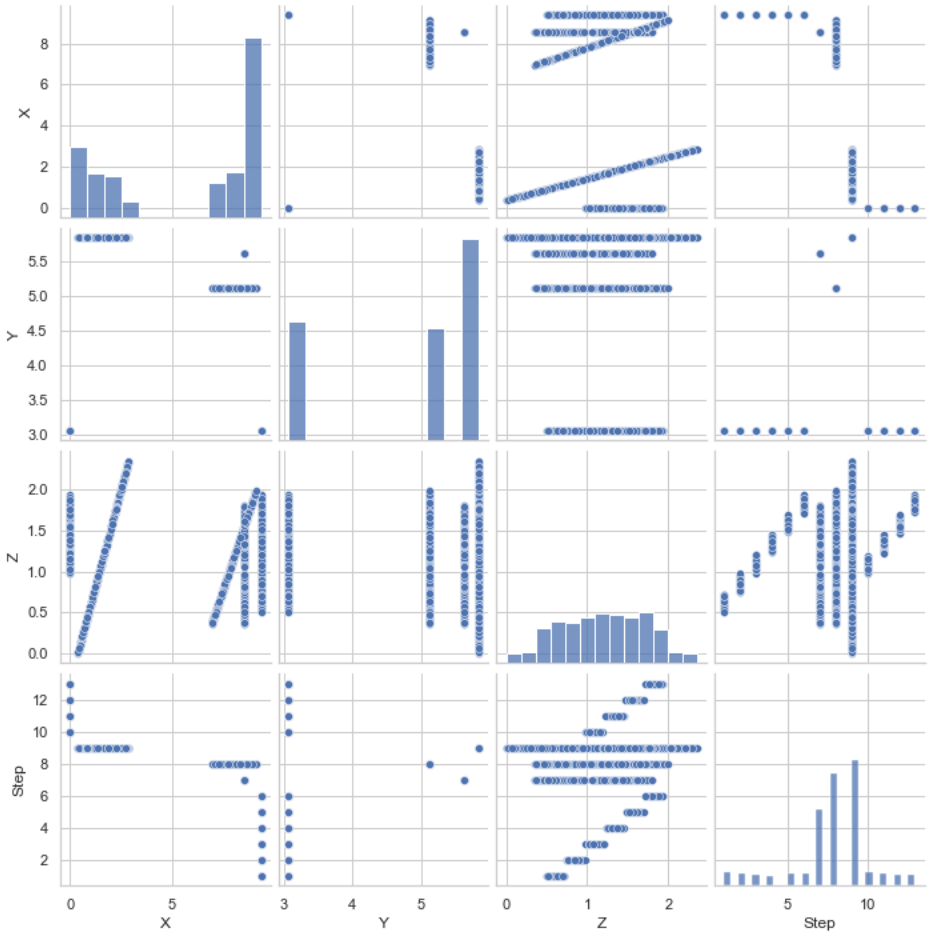

Figure 1: Pairplot of the training data set.

To get another point-of-view, the data was also 3D plotted and colour coded based on the step value. This is shown below in Figure 2. This 3D plot does a much better job in visualizing the data. It is immediately clear that finding correlations within the entire dataset is pointless. Separating the datasets 1-6, 7, 8, 9, and 10-13 will show much better statistics and correlations. This is exactly what was done next. Individual plots for each step were then generated. Three arbitrary examples are shown in Figure 3. It can be seen that within a step, the data is very linear, this is further proven with the pairplots shown in Figure 4 for the corresponding 3D plots in Figure 3. The overall correlation values between the entire data set is shown in the correlation matrix in Figure 5. This shows nothing of significance, again, since the data is structured and not continuous. The individual step correlation matrix between each of the 3 ‘X’, ‘Y’, ‘Z’ and the Step variable were also not there. The correlations were all ‘nan’; this makes sense since they were separated by the step. There were some correlations identified within the grouped data, however. The points in steps 1-6 showed a 0.99 correlation factor between the ‘Z’ and ‘Step’ values; and the points in steps 10-13 showed a 0.97 correlation factor between the ‘Z’ and ‘Step’ values. The steps 7, 8, and 9 do not have the correlations since they only consist of one step value. Also, there are no ‘X’ or ‘Y’ correlations to ‘Step’ for the two groups with ‘Z’ and ‘Step’ correlations since the data in those groups has identical ‘X’ and ‘Y’ values (only change in ‘Z’).

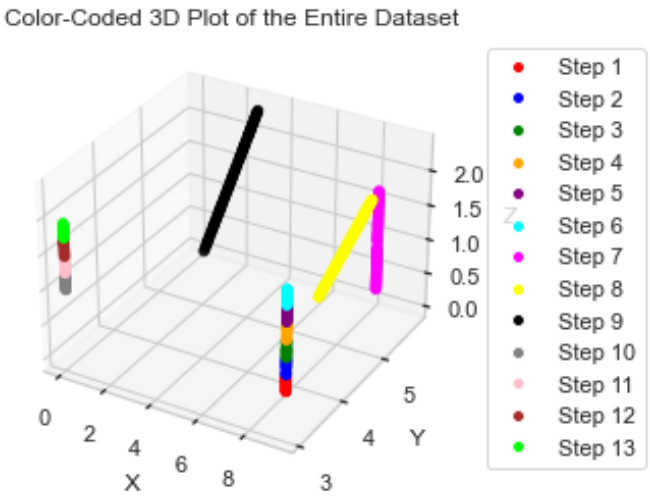

Figure 2: Colour-Coded 3D plot for the entire dataset.

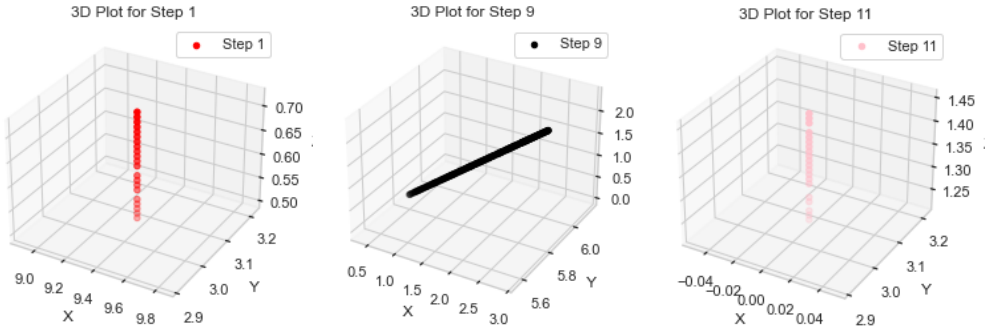

Figure 3: Examples of individual 3D plots for steps 1, 9, and 11.

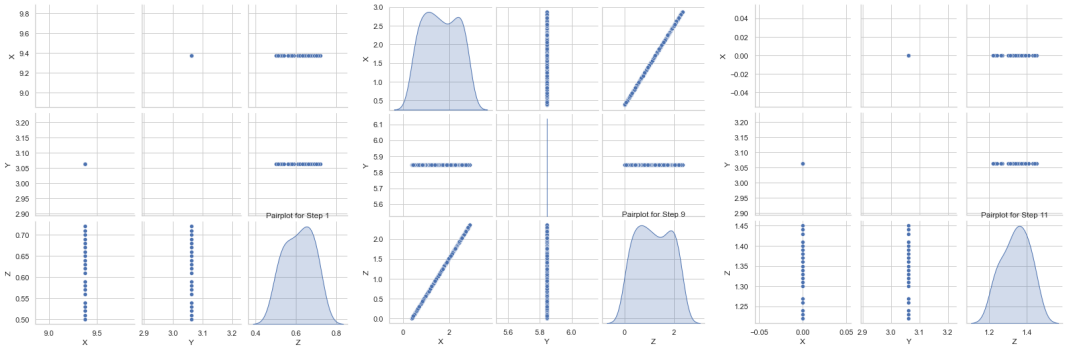

Figure 4: Examples of individual pairplots for steps 1, 9, and 11.

	X	Y	Z	Step
X	1.000000	-0.140759	0.013870	-0.749911
Y	-0.140759	1.000000	-0.133676	0.292810
Z	0.013870	-0.133676	1.000000	0.199531
Step	-0.749911	0.292810	0.199531	1.000000

Figure 5: Overall correlation matrix using the pearson correlation value.

After observation of the data, it was also noted that scaling was not necessarily required for this problem. This is due to the fact that the floating-point values of the data points were relatively close to the actual integer step values.

Then, the following 3 machine learning models were decided on: Logistic Regression, Decision Trees, and Random Forest. These 3 models were chosen since the predictor was integer and not a continuous value. Each model was developed with a grid search for a parameter grid in order to perform hyperparameter optimization. All 3 models were optimized using accuracy as the scoring value. Accuracy was chosen over f1 since the classes were more or less well distributed and the true positive and negative rates are important.
These are the results of the hyperparameter optimization:

Logistic Regression Results

Best Hyperparameters for Logistic Regression: {'C': 100, 'penalty': 'l2', 'solver': 'newton-cg'}
lr_Accuracy: 0.99
Confusion Matrix:

```
[[ 3  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  5  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  6  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  1  7  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  5  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  4  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 26  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 43  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0 52  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  3  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  4  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  6  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  1  6  0]]
```

2 single errors

Decicion Tree Results

Best Hyperparameters for Decision Tree: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2}
dt_Accuracy: 0.9883720930232558
Confusion Matrix:
[[3 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 5 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 6 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 8 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 5 0 0 0 0 0 0 0 0 0]
[0 0 0 0 1 3 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 26 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 43 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 52 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 3 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 4 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 6 0 0]
[0 0 0 0 0 0 0 0 0 0 0 1 6 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 6]]
2 single errors

Random Forest Results

Best Hyperparameters for Random Forest: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200}
rf_Accuracy: 0.9709302325581395
Confusion Matrix:
[[3 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 5 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 6 0 0 0 0 0 0 0 0 0 0 0]
[0 0 1 7 0 0 0 0 0 0 0 0 0 0]
[0 0 0 1 4 0 0 0 0 0 0 0 0 0]
[0 0 0 0 1 3 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 26 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 43 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 52 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 2 1 0 0 0]
[0 0 0 0 0 0 0 0 0 0 4 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 6 0 0]
[0 0 0 0 0 0 0 0 0 0 0 1 6 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 6]]
5 single errors

The results show very similar performance between the optimized logistic regression and decision tree models. Interesting enough, all 3 models fail to identify 1 step 12 value as a step 12, and falsely identified it as a step 13 value. Based on the confusion matrices, it was also shown that none of the models predicted a step value outside of the step groups that were identified previously (1-6, 7, 8, 9, and 10-13). This is valuable since a false positive there would mean an enormous error.

Overall, all models are performing quite well and the best-suited one is either the logistic regression model or the decision tree. The statistical results of the models do not provide any information that would determine whether one is actually better besides the slightly higher accuracy of the logistic regression model (by not very much).

In the last step, the given extra data points were predicted using all 3 models and they all produced the same results:
Logistic Regression Model Step Prediction: [5 8 13 6 4]
Decision Tree Model Step Prediction: [5 8 13 6 4]
Random Forest Model Step Prediction: [5 8 13 6 4]

4.0 Conclusion

In In conclusion, this project has focused on the integration of Augmented Reality (AR) with predictive machine learning (ML) algorithms in the aerospace industry. The primary real-life objective was to enhance AR-based instruction modules to improve efficiency and accuracy in tasks such as aircraft maintenance, computer-aided design, part assembly, and space-related applications. By developing micro-level coordinate-based ML models, technicians in the aerospace industry can gain valuable insights and guidance. The learning outcome was for the student to develop their understanding of machine learning projects and their application.

The project's code outline consists of several stages, including configuration, data import and splitting, visualization, data analysis, correlation analysis, and the development of classification models. Three ML models, namely Logistic Regression, Decision Tree, and Random Forest, were created and optimized using grid search for hyperparameter tuning. The evaluation results showed high accuracy, with Logistic Regression achieving 99%, Decision Tree achieving 98.8%, and Random Forest achieving 97%. The models were well-suited for predicting maintenance steps based on provided coordinates.

Throughout the project, observations were made regarding the dataset's characteristics, distributions, and the necessity of data segmentation. Data visualization and correlation analysis provided valuable insights, showing the importance of considering step-based groupings. The findings informed the selection of the appropriate ML models for integer prediction.