OXFORD BROOKES UNIVERSITY

Faculty of Technology, Design and Environment

MASTER OF SCIENCE DISSERTATION

Dissertation Title:	Noise-Based Predictive Maintenance for Aircraft Models: A Machine Learning Perspective
Surname:	Pant
First Name:	Ashwini
Student Number:	19212510
Supervisor:	Dr Eleni Elia
Dissertation Module:	TECH7009 Dissertation in Data Analytics
Course Title:	MSc Data Analytics
Date Submitted:	29/09/2023

Statement of Originality

Except for those parts in which it is explicitly stated to the contrary, this project is my own work. It has not been submitted for any degree at this or any other academic or professional institution.

Signature of Author	Ashwini Pant	Date	29/09/2023

Regulations Governing the Deposit and Use of Master of Science Dissertations in the School of Engineering, Computing and Mathematics, Oxford Brookes University.

1. A copy of dissertation final reports submitted in fulfilment of Master of Science course requirements shall normally be kept by the School.

2. The author shall sign a declaration agreeing that, at the supervisor's discretion, the dissertation will be submitted in electronic form to any plagiarism checking service or tool.

3. The author shall sign a declaration agreeing that the dissertation be available for reading and copying in any form at the discretion of either the dissertation supervisor or in their absence the Programme Lead of Postgraduate Programmes, in accordance with 5 below.

4. The project supervisor shall safeguard the interests of the author by requiring persons who consult the dissertation to sign a declaration acknowledging the author's copyright.

5. Permission for anyone other than the author to reproduce in any form or photocopy any part of the dissertation must be obtained from the project supervisor, or in their absence the Programme Lead of Postgraduate Programmes, who will give his/her permission for such reproduction only to the extent to which he/she considers to be fair and reasonable.

I agree that this dissertation may be submitted in electronic form to any plagiarism checking service or tool at the discretion of my project supervisor in accordance with regulation 2 above.

I agree that this dissertation may be available for reading and photocopying at the discretion of my project supervisor or the Programme Lead of Postgraduate Programmes in accordance with regulation 5 above.

Signature of Author	Ashwini Pant	Date	29/09/2023

Abstract

The rise of the civil aviation industry has made noise pollution a big concern in recent years around the globe due to increase in take-offs, landings, flyovers, and construction of new airports. This issue has had a significant impact on residential areas and related industries on proximity to airport. Predicting noise level of the aircraft when specific aircraft needs to be maintained is crucial for efficient operation. The factors influencing the maintenance of aircraft models are quite complex and have non-linear relationships. Various factors, such as the specific conditions in different aircraft, as well as variations in aircraft noise level, all contribute to the complexity of predicting maintenance needs. Given the limitations of existing aircraft maintenance prediction models, this report introduces a new model that offers greater generality and employs a corresponding machine learning regression algorithm (i.e., Gradient Boosting). This model helps efficiently predict the aircraft maintenance to avoid sudden accidents, reduce environmental noise pollution and provide better experience for the customers. During the machine learning process, the model is trained using historical data and tested with the most recent data. The test results demonstrate that this model, combined with the machine learning algorithm, provides an efficient method for predicting aircraft maintenance needs.

Keywords: Noise level; aircraft model; maintenance prediction; machine learning; gradient boosting

Acknowledgment

I would like to express my deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude to my supervisor **Dr Eleni Elia**, whose contribution, guidance, constant supervision, stimulating suggestions and encouragement helped me to coordinate my project especially in development, writing and improving the content of this report.

Table of Contents

1 Introduction.. 1

1.1 Aims 2

1.2 Objectives.. 2

2 Background Research and Theory.. 3

2.1 Background 3

2.2 Literature Review.. 5

2.3 Knowledge of Technologies and Tools 10

2.3.1 Anaconda Navigator 10

2.3.2 Jupyter Notebook 10

2.3.3 Python 10

2.3.4 Libraries.. 10

2.4 Justification for Chosen Methodology 12

2.4.1 Gradient Boosting 12

2.4.2 Simple Decision Tree.. 13

2.4.3 Random Forest.. 13

2.4.4 XGBoost 14

2.5 Methodology.. 15

2.5.1 Use of Chosen Methodology.. 15

2.6 Use of Sources 17

3 Technical and Practical Problem Solving 18

3.1 Description of Development Process 18

3.1.1 Data Downloading 18

3.1.2 Data Exploration 19

3.1.3 Building Model 25

3.1.4 Model Evaluation 26

3.1.5 Prediction 27

3.2 Constraint and Limitations 30

3.3 Original Contribution 30

3.4 Understanding of Issues 31

3.5 Technical Achievement ... 31

3.6 Evaluation and Discussion ... 31

4 Critical Appraisal ... 32

4.1 Result and Achievement ... 32

4.1.1 Comparison of Performance of Different Algorithms with Gradient Boosting ... 32

4.1.2 Analysis of Noise Level of Aircraft Models ... 33

4.1.3 Predicting Maintenance of Aircraft Model Exceeding Threshold ... 35

4.2 Legal Issues ... 37

4.3 Social Issues ... 37

4.4 Ethical Issues ... 38

4.5 Wider Issues ... 38

4.6 Personal Reflection ... 39

5 Conclusion and Reflection ... 40

5.1 Future Work ... 40

5.2 Conclusion ... 40

6 Bibliography ... 41

7 Appendix ... 44

7.1 Appendix A: Timeline of the Project (Gantt Chart Representation) ... 44

7.2 Appendix B: Survey Results ... 44

7.2.1 Demographics ... 44

7.2.2 Aircraft Noise Pollution ... 45

7.2.3 Machine Learning in Aircraft Maintenance ... 47

7.3 Appendix C: Code ... 50

7.3.1 Appendix C1: Code Link ... 50

7.3.2 Appendix C2: Data Pre-Processing and Exploration ... 50

7.3.3 Appendix C3 : Data Visualisation without Using Algorithm ... 52

7.3.4 Appendix C4: Dividing Dataset into Training and Testing ... 54

7.3.5 Appendix C5: Model Evaluation ... 54

7.3.6 Appendix C6: Comparing the Actual Values with the Predicted Values ... 55

7.3.7 Appendix C7: Comparison of Different Algorithm with Gradient Boosting ... 56

7.3.8 Appendix C8: Analysis of Noise Level of Aircraft Models 58

7.3.9 Appendix C9: Predicting Maintenance of Aircraft Model Exceeding Threshold 60

7.4 Appendix D: Evidence of Weekly Meeting with Supervisor 63

Table of Figures

FIGURE 1: NOISE LEVEL PREDICTION USING HYENA AND DEBATS [11]........ 5
FIGURE 2: NOISE POLLUTION CYCLE BEFORE/AFTER COVID-19 [14] 6
FIGURE 3: MACHINE LEARNING APPLICATIONS IN AVIATION TECHNOLOGY [18] 7
FIGURE 4: CATEGORIZATION OF ML ALGORITHM [18] 8
FIGURE 5: TRADITIONAL ANOMALY DETECTION TECHNIQUES IN ML........ 9
FIGURE 6: WORKING MECHANISM OF GRADIENT BOOSTING ALGORITHM [35]........ 15
FIGURE 7: FLOWCHART OF DEVELOPMENT PROCESS 18
FIGURE 8: DATA GATHERING........ 18
FIGURE 9: DATA EXPLORATION 19
FIGURE 10: DATA EXPLORATION (CONTD.) 20
FIGURE 11: DATA EXPLORATION (CONTD.) 21
FIGURE 12: CHECKING DISTRIBUTION OF TARGET VARIABLE (MAX_SLOW)........ 22
FIGURE 13: SCATTER PLOT SHOWING NOISE LEVEL ACCORDING TO DATE 23
FIGURE 14: DENSITY PLOT SHOWING NOISE LEVEL........ 24
FIGURE 15:BOX PLOT SHOWING DIFFERENT AIRCRAFT MODEL NOISE LEVEL 24
FIGURE 16 : BOX PLOT SHOWING DIFFERENT AIRCRAFT TYPE NOISE LEVEL........ 25
FIGURE 17: DIVIDING TRAINING AND TESTING DATASET........ 25
FIGURE 18: MODEL EVALUATION DONE USING GRADIENT BOOSTING ALGORITHM 26
FIGURE 19: MODEL EVALUATION OF TRAINING AND TESTING DATASET........ 27
FIGURE 20: COMPARISON OF DIFFERENT ALGORITHM WITH GRADIENT BOOSTING ALGORITHM 32
FIGURE 21: PREDICTING AIRCRAFT NOISE LEVEL 33
FIGURE 22: PREDICTING MAINTENANCE OF THE AIRCRAFT MODEL EXCEEDING THE THRESHOLD 35
FIGURE 23: SHOWING AIRCRAFT MODEL 21 AND AIRCRAFT TYPE 21 ACCORDING TO PREDICTION 35
FIGURE 24: PREDICTING MAINTENANCE NOT REQUIRED BY THE AIRCRAFT MODEL 27 EXCEEDING THE THRESHOLD 36
FIGURE 25: SHOWING AIRCRAFT MODEL 27 AND AIRCRAFT TYPE 27........ 36
FIGURE 26: TIMELINE OF THE PROJECT 44
FIGURE 27: SURVEY DEMOGRAPHICS 45
FIGURE 28: SURVEY AIRCRAFT NOISE POLLUTION 46
FIGURE 29: SURVEY OF AIRCRAFT NOISE POLLUTION MITIGATING OPINION........ 47
FIGURE 30: SURVEY OF MACHINE LEARNING IN AIRCRAFT MAINTENANCE 1........ 47
FIGURE 31: SURVEY OF MACHINE LEARNING IN AIRCRAFT MAINTENANCE 2........ 48
FIGURE 32: SURVEY OF MACHINE LEARNING IN AIRCRAFT MAINTENANCE ADDITIONAL COMMENTS AND INSIGHTS........ 49
FIGURE 33: CODE OF DATA-PREPROCESSING AND DATA EXPLORATION........ 51
FIGURE 34: CODE OF DIFFERENT DATA VISUALISATION WITHOUT USING ALGORITHM 53
FIGURE 35: CODE OF DIVIDING DATASET INTO TRAINING AND TESTING DATASET........ 54
FIGURE 36: CODE OF MODEL EVALUATION USING GRADIENT BOOSTING MODEL 54
FIGURE 37: CODE OF MODEL EVALUATION OF TRAINING AND TESTING DATASET 54
FIGURE 38: COMPARING THE ACTUAL VALUE WITH THE PREDICTED VALUE........ 55
FIGURE 39: CODE OF CALCULATING SINGLE DECISION TREE MSE, RMSE AND R2 56
FIGURE 40: CODE OF CALCULATING RADOM FOREST MSE, RMSE AND R2 56
FIGURE 41: CODE OF CALCULATING GRADIENT BOOSTING MSE, RMSE AND R2........ 57
FIGURE 42: CODE OF CALCULATING XGBOOST MSE, RMSE AND R2 57

FIGURE 43: CODE OF COMPARING PERFORMANCE OF SINGLE DECISION TREE, RANDOM FOREST, XGBOOST WITH GRADIENT BOOSTING 58
FIGURE 44: CODE OF ANALYSIS OF NOISE LEVEL OF AIRCRAFT MODEL 59
FIGURE 45: CODE OF PREDICTION OF MAINTENANCE OF AIRCRAFT MODEL 27........ 60
FIGURE 46: CODE OF SHOWING MAINTENANCE FOR AIRCRAFT MODEL 27 AND AIRCRAFT TYPE 27.. 61
FIGURE 47: CODE OF PREDICTION OF MAINTENANCE OF AIRCRAFT MODEL 21........ 61
FIGURE 48: CODE OF SHOWING MAINTENANCE FOR AIRCRAFT MODEL 21 AND AIRCRAFT TYPE 21.. 62
FIGURE 49: EVIDENCE OF MEETING WITH SUPERVISOR FROM MAY TO SEPTEMBER 67

Table of Acronyms

WHO	World Health Organization
GB	Gradient Boosting Algorithm
XGBoost	Extreme Gradient Boosting
CIS	Commonwealth of Independent States
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
R^2	Coefficient of Determination R-squared
HYENA	Hypertension and Exposure to Noise near Airports
DEBATS	Discussion on the Health Effects of Aircraft Noise
EDA	Exploratory Data Analysis
RNN	Recurrent Neural Network
ML	Machine Learning
dB	Decibels
A*	A* Algorithm

Glossary

Mean Square Error: The Mean Squared Error (MSE) measures the proximity of a regression line to a set of points. It does this by computing the squared distances between the points and the regression line effectively eliminating negative values and emphasizing larger deviations. The lower MSE indicates the closer it is to the best-fit line.

Root Mean Square Error: The Root Mean Square Error (RMSE) is the square root of the mean square error. The root mean square error is used to measure how the magnitude of dispersion of residuals or prediction errors in a calculation. It denotes the difference between the predicted and observed results.

R-Squared (R^2): R-Squared (R^2 or the coefficient of determination) is a statistical measure in a regression model that determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit). R-squared can take any values between 0 to 1. Generally, a higher r-squared indicates more variability is explained by the model.

1 Introduction

The aviation industry has seen growth since the emergence of air travel with many people today considering it essential to journey across the globe. As a result, passengers play a role in generating profits for airlines. On the other hand, Airlines focus on ensuring that flying is both safe and enjoyable for passengers. According to [1] in 2018, airlines worldwide spent nearly $69 billion on fixing and maintaining their planes. This fact made up about 9% of all the money they used to run their operations. Between 2009 and 2019, the number of people flying on planes regularly increased by 183% all over the world. Looking ahead, from 2019 to 2039, experts predict that the number of planes in use worldwide will almost double. As older planes without as much sensor technology get retired and replaced with newer ones, the need for maintenance and the amount of data collected from planes will grow a lot during this time. This means more work will be needed to keep the planes safe and running well.

A significant concern that has emerged over time is the impact and noise pollution caused by aircraft. Residential area around or near airport have been a primary victim of global noise pollution. It not only contributes to noise pollution, but also air quality and climate change. According to [2], Noise primarily affects the mean expected damages for areas near airport boundaries. Average expected damages range from $100 to $400 per person annually for those living nearby. Smaller airports (i.e., <75,000 operations), show air quality damages below $20 per person per year, like climate impacts. However, at larger airports (i.e., >400,000 operations), environmental damage from noise and air quality reaches around $400 per person annually near the airport. Climate damages become more concerning beyond 2 to 6 kilometres from the airport boundary, depending on the airport's size. Therefore by utilizing machine learning techniques to predict aircraft noise levels, companies can achieve to maintain these standards and provide eco-friendly flying experience.

This report dives into using machine learning techniques to predict how noisy airplanes are going to be, change the way we take care of planes regularly and how we can use this technology to make flying safer and more enjoyable for passengers.

1.1 Aims

To analyse the noise levels of the aircraft models and to predict the maintenance if the aircraft model produces noise level higher than a threshold so that accident risks can be reduced using timeseries dataset.

1.2 Objectives

The following objectives will be carried out to achieve the aim.

- Execute the data gathering process.
- Implement the data cleaning process to attain the desired results.
- Perform Exploratory Data Analysis (EDA).
- Construct a model using the dataset and gradient boosting algorithm, with data division into training and test datasets.
- Evaluate the model's performance with various machine learning algorithm such as Simple Decision Tree, Random Forest, XGBoost with Gradient Boosting algorithms to prove Gradient Boosting provides the best accuracy.
- Utilizing the dataset for prediction using Gradient Boosting and draw conclusions accordingly.

In this report, I will be using data from existing data sets from Kaggle websites and apply machine learning techniques to analyse aircraft noise level and predict maintenance required that will allow agencies and aviation companies to develop their pattern for schedule maintenance for all aircraft models.

2 Background Research and Theory

2.1 Background

In recent times, the aviation industry has become a crucial part in our lives. It's not just about planes; it's about how it connects people, cultures, and businesses worldwide. This has made global trade and tourism much easier. Also, it's a big deal for the economy, providing lots of jobs and helping other industries like tourism and manufacturing. Imagine a world where time is super important. That's what aviation does; it helps people and leaders meet quickly and provide medical aid in emergencies to places that need it fast. According to a recent statistics [3], based on multiple aviation companies such as Airbus, Boeing etc., quarter of global air transport takes place in North America with 25.6% followed by Europe with 22.7%, Asia-Pacific Region accounting for 32.5% and the rest of the world along with Africa, Commonwealth of Independent States (CIS), Latin America and Middle East together account for 19.2%. Scientists also benefit from aviation. They use special planes and technology to study things like the weather and the environment. And did you know aviation has a hand in space exploration too? Recently, Chandrayaan-3 had a successful lunar landing on the south pole making it the first ever to do so attracting millions of attention, enabling new possibilities in the aviation market [4]. Aviation is a big part of our world today. It helps us in many ways, from business to science to culture. It's always changing and adapting to our needs in this global world we live in.

Precision has always been extremely important for keeping air travel safe and efficient. As aviation has progressed, so has the desire for more accuracy in every part of flying. Evidence can be found as far as the World War I were the airplanes were used as a navigation tool and to map enemy territory using high resolution cameras [5]. As sophisticated the aviation technology gets, it still can affect environmental factors by a greater margin. Noise pollution is considered such a factor in recent times. Since, the planes are assembled with more high-end features, more fuel and power consumption generate huge amounts of sounds to fly unpleasing to the human ear. Thus, people living closer to airport areas experience more stress related to noise pollution and affected on overall wellbeing. According to [6], repeated exposures to loud sounds can cause hearing loss over time. The sounds are measured in decibels (dB) and classified in 3 phases, whisper which is about 30 dB, normal about 60dB and loud sounds over 70dB. Anything above 70dB is considered noise and contributes to hearing loss. The

World Health Organization (WHO) recommends maintaining environmental noises below 70dB to prevent hearing loss.

Furthermore, consistent exposure to aviation noise has been associated with increased stress levels, sleep disruptions, and a higher risk of heart-related issues such as high blood pressure and heart disease. Sleep disturbances can result in fatigue, reduced cognitive function, and a greater likelihood of accidents. Moreover, the continuous aircraft noise can contribute to mental health problems like anxiety and depression, reducing the overall well-being of affected communities [7]. Consequently, taking steps to reduce aviation noise by using Machine Learning is crucial not only for comfort but also for protecting public health.

Machine learning has become a strong helper in achieving this greater precision. Before, aviation engineers had to do a lot of calculations and use basic tools to find their way and manage the aircraft. It was quite hard to keep the plane exactly on timely maintenance and make sure everyone was safe. But as aviation technology got better, things like autopilots, incident prediction, better navigation tools, and air traffic control systems helped make flying more precise. The concept of aviation 4.0 has relatively taken the world by storm with features such as Automatic/ Autonomous Flight Rules, Eco-Friendly Flight Rules, Flight Simulator programs for young developing pilots, Virtual Cockpits, Improved Search and Rescue Services and Realtime Human Performance Monitoring/Alerting [8]. It has become a super important tool in aviation. It makes everything more precise and helps passengers have better experiences when they fly. As technology gets even better, the aviation industry will use machine learning more to make flying even safer, eco-friendly, and more enjoyable for people all over the world.

2.2 Literature Review

Noise pollution has been a pressing issue all over the globe with rise of new aviation companies and airports for convenience. Despite of great business value for aviation giants after the tourism industry captivated the market since the 1960s and 1970s, the sudden rise of flyovers, landings and take-offs have taken toll among the residents living near the airports. The residential community has been on a constant battle with the aviation companies ever since the 1960s. According to a statistical impact report outlined by the European Commission in 2020 surveyed across 47 European airports, noise level above 45dB were experienced by 3.2 million people and about 1.2 million people experienced noise level above 40dB. Furthermore, estimates reported about 1 million residents per day were exposed to more than 50 aircraft noise above 70dB equivalent of a busy office [9]. Subsequently, Imperial College London implied a chance of stroke or heart attack rose by 10-20 percent for residents living in high noise zone [10].

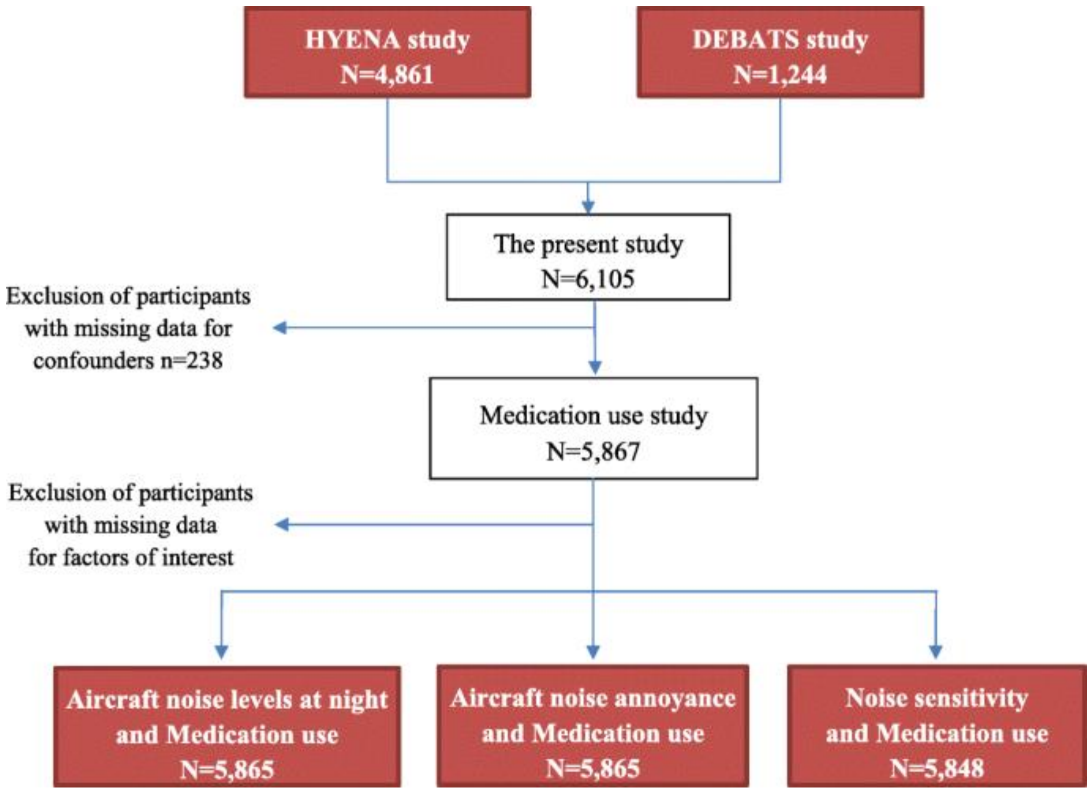
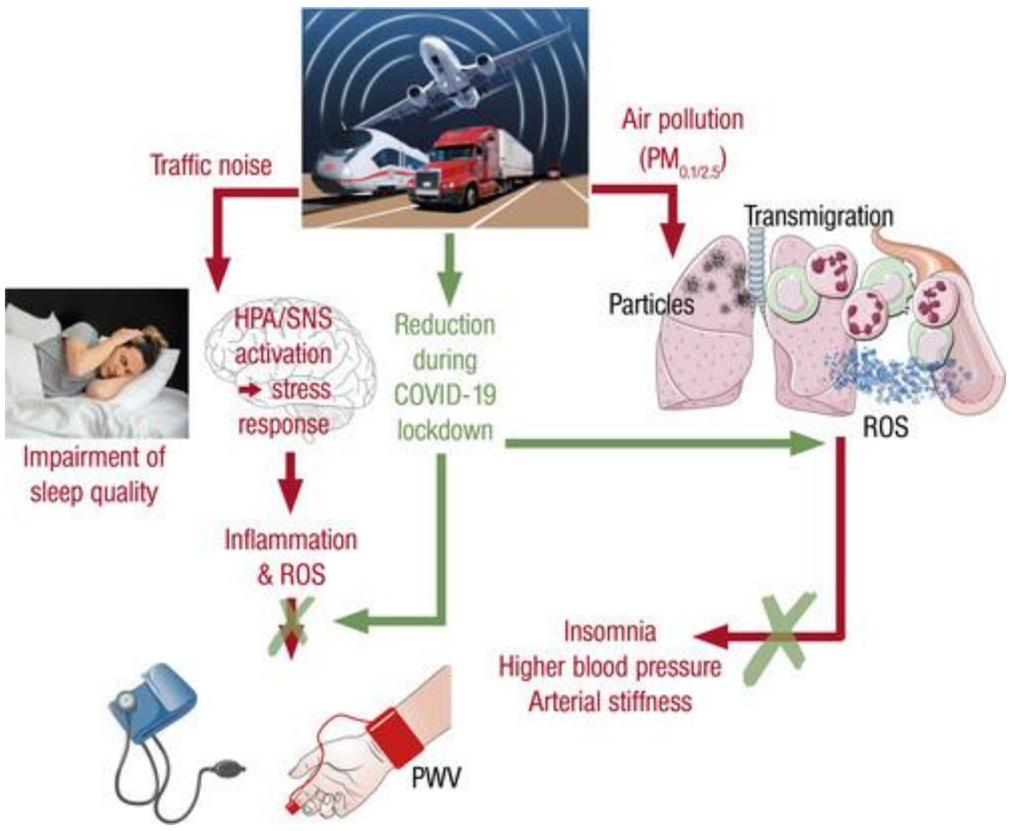

HYENA study
N=4,861
DEBATS study
N=1,244
The present study
N=6,105
Exclusion of participants with missing data for confounders n=238
Medication use study
N=5,867
Exclusion of participants with missing data for factors of interest
Aircraft noise levels at night and Medication use
N=5,865
Aircraft noise annoyance and Medication use
N=5,865
Noise sensitivity and Medication use
N=5,848

Figure 1: Noise Level Prediction Using HYENA and DEBATS [11]

Likewise, survey conducted by [12] among 5860 residents of ten European airports included in the studies HYENA (Hypertension and Exposure to Noise near Airports) and DEBATS (Discussion on the health effects of aircraft noise) concluded that about 25% participants used antihypertensive medications which varied among France to Germany from 16% to 33% respectively. Countries had varying medication usage, aircraft noise annoyance, and noise sensitivity rates among participants. For instance, in Greece, 43% found aircraft noise highly annoying, compared to just 10% in Sweden. Noise sensitivity was split with 35% having low, 32% medium, and 33% high sensitivity, with Swedes being less sensitive and Italians more so. Participants differed based on aircraft noise levels in relation to antacid use, age, education, physical activity, alcohol, and smoking. There were also distinctions between highly annoyed and less annoyed participants and among different noise sensitivity categories, impacting medication use, gender, age, education, BMI, physical activity, alcohol, and smoking habits [13].



Traffic noise
Air pollution ($PM_{0.1/2.5}$)
Transmigration
Particles
Impairment of sleep quality
HPA/SNS activation ➔ stress response
Reduction during COVID-19 lockdown
ROS
Inflammation & ROS
Insomnia
Higher blood pressure
Arterial stiffness
PWV

Figure 2: Noise Pollution Cycle Before/After COVID-19 [14]

During the pandemic, a major decline in aircraft noise pollution were observed with aviation market at risk. An article published by [15] concluded that short term reduction in aircraft noise exposure can reverse the effects of arterial stiffness, blood pressure, improve overall sleep cycle, better mood, decrease emotional and cognitive stress. The article also describes how the lockdown became a major factor in improving overall health and stability of the residents and how noise exposure affected residents at different time of the day [16].

Despite the commotion, Recent policies and framework have provided a path to reducing aircraft noise. Efforts have been taken by different airports which have had some major impact in improving lifestyles of the residents. Airports in Frankfurt, Vienna and UK have all been a major contributors actively participating in reducing noise level. Concepts such as active/passive model used by Frankfurt airport which consist of Noise Respite Model and Soundproofing along with implementing noise monitoring programs with stations and noise reports, offering to buy houses or provide compensations to properties where aircrafts need to be flown below 350m. Similarly, Vienna airport has developed virtual chatbots to engage with the residents and foster public participation for improving land use, noise mitigation and night-time restrictions. Likewise, UK has developed their own concept known as League Table were the airlines are ranked based on their environmental performances further encouraging customers and providing awareness to the community indirectly to contribute to sustainable environment [17].

The aviation industry has undergone significant technological advancements over the years, and one of the most transformative influences has been the integration of machine learning (ML) techniques. Machine learning is all about using smart computer programs that can analyse tons of data and figure out complex patterns. Machine learning's integration into the aviation industry can be traced back several decades. Early applications primarily focused on data analysis for improving flight safety and optimizing flight routes. As computing power advanced, ML algorithms became increasingly sophisticated, ushering in a new era of capabilities.

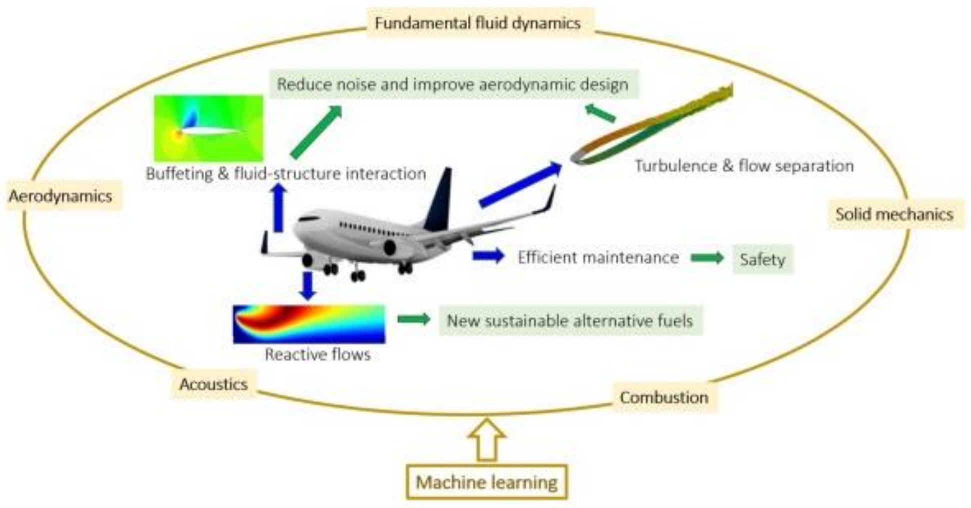

Fundamental fluid dynamics
Reduce noise and improve aerodynamic design
Buffeting & fluid-structure interaction
Turbulence & flow separation
Aerodynamics
Solid mechanics
Efficient maintenance
Safety
New sustainable alternative fuels
Reactive flows
Acoustics
Combustion
Machine learning

Figure 3: Machine Learning Applications in Aviation Technology [18]

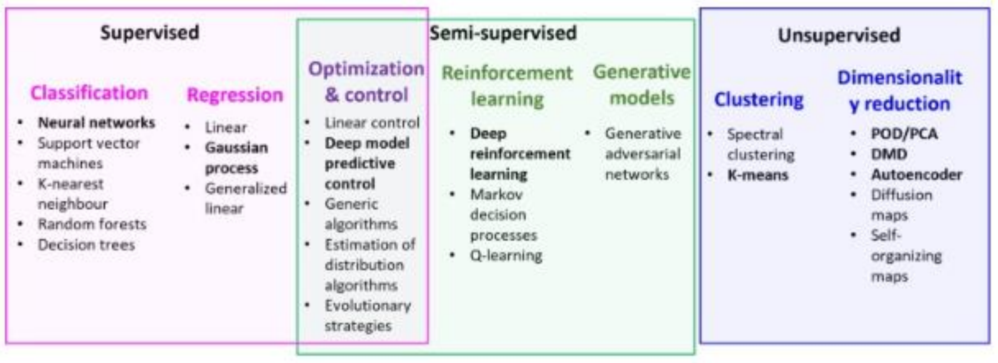


Figure 4: Categorization of ML Algorithm [18]

Predictive maintenance, a cornerstone of ML in aviation, allows for the prediction of equipment failures before they occur. [19] highlight how ML algorithms can be trained on historical maintenance data which has been classified into unsupervised, semi-supervised and supervised learning to anticipate potential issues, enhancing aircraft reliability, and minimizing downtime. The author also sheds light into how emerging technology such as ML can be categorized based on regression and classification for supervised learning accepting uncertainty errors on trained dataset to be deduced and analysed properly to represent the overall dataset. Different data collected from specific components can also be classified according to the figure above.

Flight planning has also benefited from ML techniques. [20] discuss how ML models such as A* can optimize flight paths, considering various factors such as weather conditions, air traffic, and fuel efficiency. By putting all this together, they can figure out the most efficient path for a flight. This not only means planes arrive on time but also use less fuel, which is good for the environment. These models enable airlines to achieve more efficient operations and punctual arrivals.

Anomaly detection is another crucial application for flight. The amount of false positive data generated is directly proportional to the amount of trained dataset provided to the algorithm. They have predefined sets and rules to perform operations that produces result which may or may not have higher error margin depending upon algorithm quality.

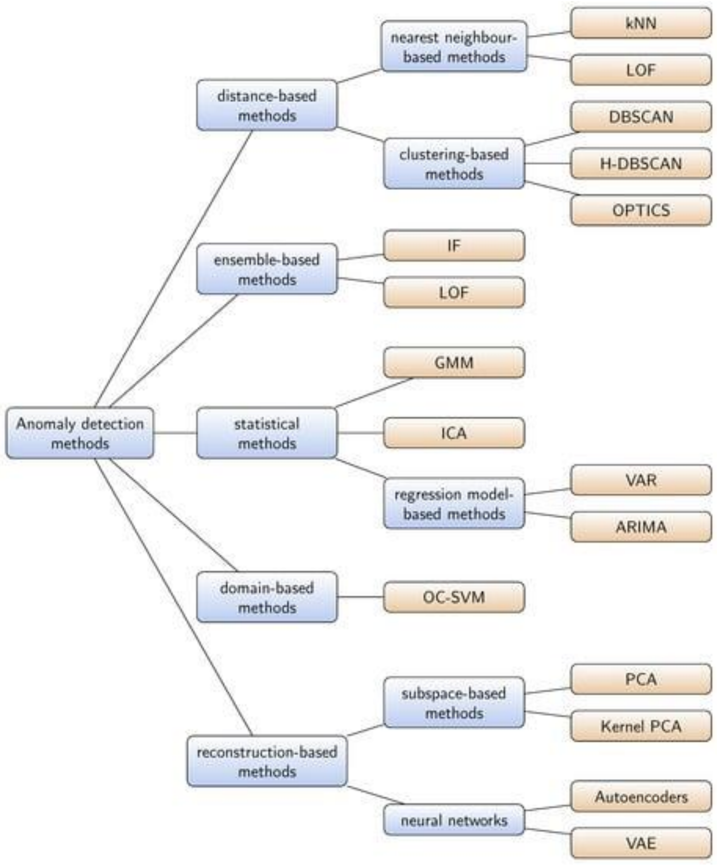


Figure 5: Traditional Anomaly Detection Techniques in ML

[21] emphasize how traditional methods such as the above figure were applied in the current aviation technology but still cannot perform accurately for large continuous datasets. The author demonstrates key methods outlines among several algorithms that are been used for different domains but represents Recurrent Neural Network (RNN) to be the most suitable anomaly detection mechanisms for a time series datasets. RNN can be used as a regression model which are more convenient to capture temporary and non-linear dependencies in multivariate time series when stacked in deep architectures. This method can surely replace many existing anomaly detection engines in the aircraft which work based upon incident rather than proactive approach with precision.

2.3 Knowledge of Technologies and Tools

2.3.1 Anaconda Navigator

Anaconda Navigator is a user-friendly tool for managing Python packages and environments in data science. It simplifies package installation, supports isolated environments, integrates with popular IDEs like Jupyter, and provides a graphical interface for easy navigation. It streamlines data science workflow [22].

2.3.2 Jupyter Notebook

Jupyter Notebook is an interactive tool for coding, text, and data visualization. It's great for creating documents that mix explanations, code, and graphs. We can share our work easily, and it supports multiple programming languages. It's widely used by data professionals and researchers for collaborative data analysis and problem-solving [23].

2.3.3 Python

Python is a versatile, user-friendly programming language known for its readability and broad application areas, including web development, data analysis, machine learning, and more. It features a large standard library, cross-platform compatibility, dynamic typing, and extensibility with other languages. Python is open-source and supports object-oriented programming, making it widely used in various domains by a large and active community of developers [24].

2.3.4 Libraries

2.3.4.1 Pandas

Pandas, a Python library, excels in handling structured data like tables. It provides DataFrames and Series for efficient data manipulation. With Pandas, you can import/export data, clean, transform, select, manage time series, and integrate with other data tools. It also integrates with data visualization and other data analysis libraries [25].

2.3.4.2 Numpy

NumPy, also known as "Numerical Python," is a crucial Python tool for numerical and scientific tasks. It's excellent at efficiently managing arrays and carrying out mathematical calculations. NumPy's speed and adaptability make it a must-have for activities such as analysing data, running simulations, and performing scientific calculations [26].

2.3.4.3 Matplotlib

Matplotlib is a popular Python library for creating high-quality data visualizations, including various types of charts and plots. It's highly customizable, supports interactive features in Jupyter notebooks, and integrates seamlessly with NumPy for data handling. Matplotlib is widely used in data analysis, research, and presentations due to its versatility and ability to produce publication-quality graphics [27].

2.3.4.4 Seaborn

Seaborn is a Python library for creating stylish statistical graphics, perfect for complex datasets. It offers various statistical plots, attractive themes, and seamless integration with data. You can visualize statistical relationships, handle categorical data, and explore correlations efficiently. It's a valuable tool to communicate data insights effectively [28].

2.3.4.5 Warnings

The "warnings" library in Python lets developers issue messages that aren't errors but serve to alert about potential issues or deprecated features in code. It supports different warning levels, allows custom warning messages, and provides options for filtering and handling warnings. This helps maintain code quality and communicate important information without causing program termination [29].

2.3.4.6 Sklearn

Scikit-Learn (sklearn) is a popular Python library for machine learning and data analysis. It provides a wide range of machine learning algorithms, tools for data preprocessing, model evaluation, and feature engineering. With an easy-to-use interface, it's suitable for both beginners and experts in the field. Scikit-Learn is a fundamental tool for developing machine learning models in Python [30].

2.4 Justification for Chosen Methodology

2.4.1 Gradient Boosting

The gradient boosting algorithm is an effective method for achieving high predictive accuracy on continuous (time series) datasets by building a strong predictive model by combining the outputs of multiple weak models and minimizing the loss function generated when making decisions, since the algorithm is designed to employ multiple decision trees sequentially to predict outcomes. This feature helps process complex data more efficiently and accurately compared to other algorithms that use only one decision tree at a time to predict outcomes. The following points explain why this algorithm works on the aircraft dataset [31].

- Aircraft noise data is intricate due to numerous factors.
- Gradient boosting excels in managing this complexity and uncovering hidden patterns. It also enhances decision-making and predictive accuracy through collaborative efforts.
- Applicable in various machine learning tasks such as regression, classification, and ranking problems and capable of handling both numerical and categorical features.
- Proficient in handling real-world data imperfections, including errors and outliers and effective at extracting valuable insights from noisy data.
- Identifies crucial factors for maintenance decision-making and focuses attention on the most significant variables for informed choices.
- It is adjustable to suit specific dataset characteristics and is tailored to optimize performance with our unique data.
- It is capable of efficiently processing extensive datasets, such as aircraft noise data and maintains accuracy and precision even under significant workloads and provides highly accurate results crucial for aircraft maintenance decisions.
- It ensures confidence in the predictions made and effective even when dealing with imbalanced datasets and utilizes regularization methods like shrinkage (learning rate) and tree pruning and prevents overfitting and ensures model generalization.

2.4.2 Simple Decision Tree

Simple Decision Tree is a visual representation of an algorithm designed to predict continuous numerical outcomes based on input features. It starts by considering the entire dataset as the root node of the tree. Initially, a feature from the dataset is chosen for splitting which are guided by conditions like minimum variance or mean squared errors within resulting subsets. A threshold for this chosen feature is set, creating two distinct subsets by dividing data points below and above the threshold that are usually assigned to the left and right child nodes. This process continues with treating each subset as a new dataset until a stopping condition is met such as reaching a maximum tree depth or a minimum number of data points in a node. Once finished, each leaf node receives a prediction value typically derived from the mean or median of the target variable for the data points it contains [32].

According to [32], the mathematical notation to calculate MSE for regression can be represented as follows:

$$\text{MSE} = \frac{1}{2}\sum_{i=1}^{N}(y_i - F(x_i))^2$$

2.4.3 Random Forest

Random Forest algorithm is a highly versatile ML technique. It leverages an ensemble of multiple decision trees to generate predictions or classifications. Through the combination of these tree outputs, the random forest algorithm provides a consolidated and accurate outcome. The algorithm has gained immense popularity due to its user-friendly interface and remarkable adaptability, rendering it effective for addressing both classification and regression challenges. What truly sets this algorithm apart is its exceptional capacity to handle intricate datasets and counteract overfitting that showcase it as an invaluable tool for a wide array of predictive tasks within the realm of machine learning. Random Forest Algorithm can process datasets featuring continuous variables and excels in delivering superior performance [33].

According to [33], the mathematical notation to calculate MSE for regression can be represented as follows:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f_i - yi)^2$$

2.4.4 XGBoost

XGBoost, or "Extreme Gradient Boosting is an algorithm known for its effectiveness in both regression and classification tasks. It has gained prominence in competitive arenas like Kaggle due to its powerful ensemble learning approach that are rooted in the gradient boosting framework. This method combines predictions from multiple weak models (i.e., decision trees) to create a strong ensemble model while preventing overfitting through L1 and L2 regularization. Furthermore, XGBoost excels in parallel and distributed computing making it suitable for large datasets which allows users to define custom loss functions for specific domains. It simplifies handling of missing data, supports cross-validation, and offers a wide range of loss functions for different problem types. XGBoost has found success in diverse domains but achieving optimal results requires careful hyperparameter tuning and data preparation [34].

According to [34], the mathematical notation to calculate MSE for regression can be represented as follows:

$$r_{im} = -\alpha \left[\frac{\partial\left(L\left(y_i\ ,\ F(x_i)\right)\right)}{\partial F\ (x_i)} \right]_{F(x)=F_{m-1}\ (x)}$$

2.5 Methodology

The theoretical background of the algorithms used to accomplish the objectives of the work are comprehensively presented in this section.

2.5.1 Use of Chosen Methodology

2.5.1.1 Gradient Boosting

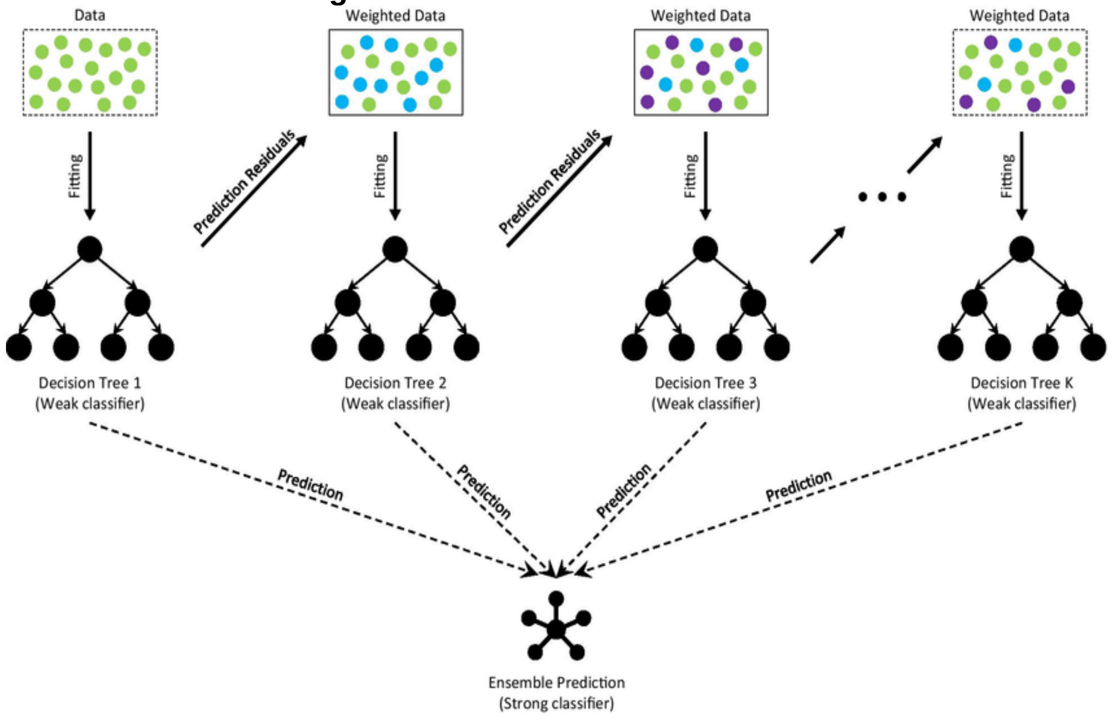


Figure 6: Working Mechanism of Gradient Boosting Algorithm [35]

Gradient Boosting is a powerful machine learning technique used for both regression and classification tasks. It builds an additive model in a forward stage-wise manner. In other words, it builds multiple weak learners (usually decision trees) sequentially and combines them to form a strong predictive model [36]. Additionally, to avoid making the model too complicated, hyperparameters are carefully tuned which allows gradient boosting to perform at its best for your specific data and task [37]. The mathematical notation for gradient boosting can be described as follows:

Let us consider, my training dataset as (X, y), where X represents the input features as aircraft model, type, datetime and y represents the target value as noise level produced by the aircraft model. The input features can be represented as:

$$X = \{(x_1, x_2, x_3)_1, (x_1, x_2, x_3)_2, .., (x_1, x_2, x_3)_N\} \qquad [38]$$

Where:

- x_1 represents the aircraft model
- x_2 represents the aircraft type
- x_3 represents the datetime features

The decision trees consist of weak learners which are represented as $h(x;\theta_i)$, where x is the input feature, θ_i are the parameters of the decision tree and $h(x;\theta_i)$ represents the prediction made by the decision tree.

For minimizing the loss function, let us consider the following notation.

$$L\big(y, F(x)\big) = \frac{1}{2}\sum_{i=1}^{N}(y_i - F(x_i))^2 \qquad [38]$$

Where N is the number of training samples, y_i is the noise level for the i-th sample, and $F(x_i)$ is the current ensemble's prediction for the i-th sample.

For calculating the gradient of the loss function with respect to current ensemble predictions, I considered the following notation.

$$r_i = -\frac{\partial L\big(y_i, F(x_i)\big)}{\partial F(x_i)} = 2(F(x_i) - y_i) \qquad [38]$$

The above function represents how much corrections are required for each training samples. Furthermore, new decision trees are fitted to the negative gradient r_i which aims to capture errors made by current ensemble represented as $h(x;\theta_{i+1})$ and update the current ensemble's prediction by adding new decision tree, weighted by a learning rate α:

$$F_{i+1}(x) = F_i(x) + \alpha \cdot h(x;\theta_{i+1}) \qquad [38]$$

The functions are then repeated for a fixed number of iterations until convergence to near accurate values represented as $F(x)$.

2.6 Use of Sources

The dataset used for this project is from Kaggle, an online platform and community that brings together data scientist, machine learning engineers and data enthusiast from around the globe. It is a popular platform for many professionals to build their data science skills and build portfolios [39]. It is important to keep in mind that the dataset used in this project might not have significant relevance or use-cases as it depends upon upvotes, notebooks shared, contributions and views. Despite of this, Kaggle sources are reliable for the following reasons [40] :

- Kaggle hosts data science competitions that tackle real-world problems, involving predictive modelling, machine learning, and data analysis. Participants stand to win cash prizes, recognition, and job prospects.
- Offers a free repository of publicly available datasets spanning various domains, serving as valuable resources for practice, research, and personal projects.
- It provides an IDE interface that empowers users to code, execute Python and R scripts, create data visualizations, and easily share their work with others. It fosters collaborative data analysis and insights sharing.
- It provides online courses and tutorials covering data science and machine learning topics, aiding users in honing their skills and advancing their knowledge in these fields.
- The Kaggle community boasts a robust and engaged user base of data scientists and machine learning practitioners. It facilitates collaborative efforts, with discussion forums and interactions serving as valuable resources for learning and networking.
- Kaggle permits organizations to host their own data science competitions and challenges on the platform, enabling companies to crowdsource solutions for specific data-related problems.

3 Technical and Practical Problem Solving

3.1 Description of Development Process

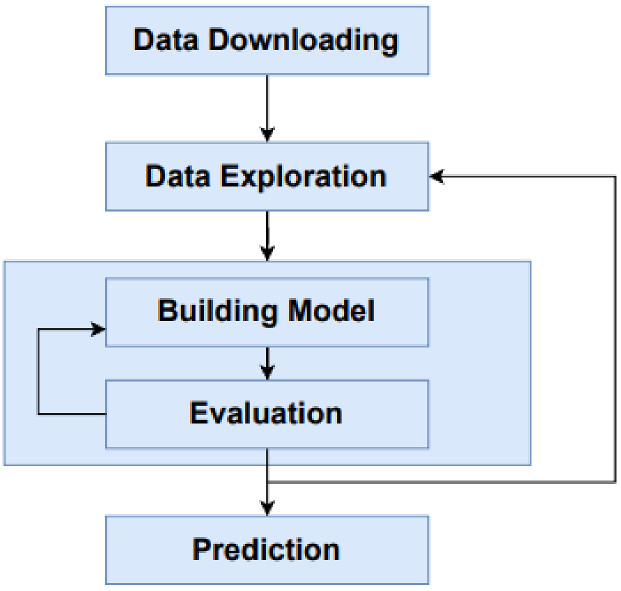
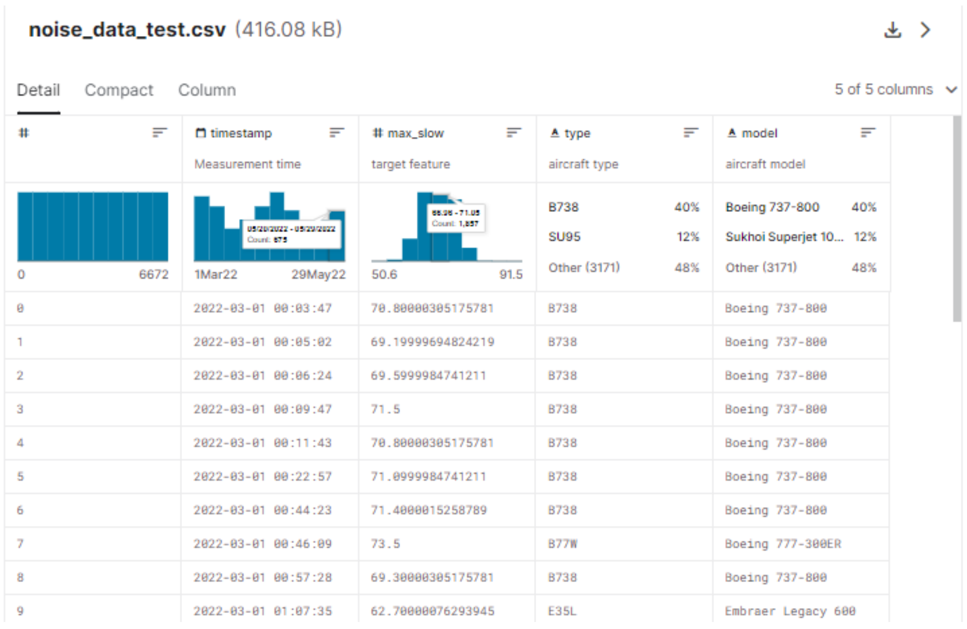


Figure 7: Flowchart of development process

3.1.1 Data Downloading


noise_data_test.csv (416.08 kB)

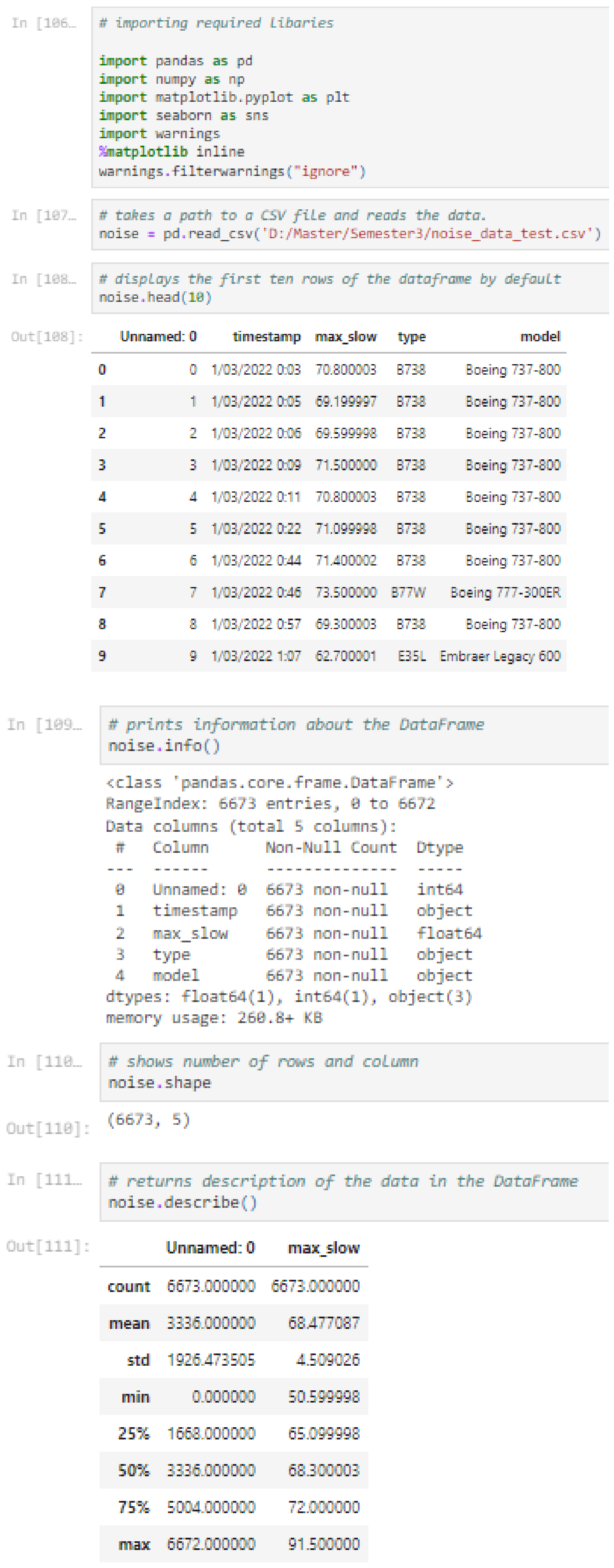
Detail Compact Column | 5 of 5 columns

#	timestamp	max_slow	type	model
	Measurement time	target feature	aircraft type	aircraft model
0 – 6672	1Mar22 – 29May22	50.6 – 91.5	B738 40% SU95 12% Other (3171) 48%	Boeing 737-800 40% Sukhoi Superjet 10... 12% Other (3171) 48%
0	2022-03-01 00:03:47	70.80000305175781	B738	Boeing 737-800
1	2022-03-01 00:05:02	69.19999694824219	B738	Boeing 737-800
2	2022-03-01 00:06:24	69.5999984741211	B738	Boeing 737-800
3	2022-03-01 00:09:47	71.5	B738	Boeing 737-800
4	2022-03-01 00:11:43	70.80000305175781	B738	Boeing 737-800
5	2022-03-01 00:22:57	71.0999984741211	B738	Boeing 737-800
6	2022-03-01 00:44:23	71.4000015258789	B738	Boeing 737-800
7	2022-03-01 00:46:09	73.5	B77W	Boeing 777-300ER
8	2022-03-01 00:57:28	69.30000305175781	B738	Boeing 737-800
9	2022-03-01 01:07:35	62.70000076293945	E35L	Embraer Legacy 600

Figure 8: Data Gathering

It is important to take reliable data so that machine learning model can find the correct patterns. The above timeseries (aircraft) data is downloaded from Kaggle [41] website where there are numerous data. The dataset contains 5 columns and 6674 rows of data, each representing aircraft. The variables in the dataset include sn, measurement of time, target feature, aircraft types, and aircraft model.

3.1.2 Data Exploration



```
In [106…  # importing required libaries

          import pandas as pd
          import numpy as np
          import matplotlib.pyplot as plt
          import seaborn as sns
          import warnings
          %matplotlib inline
          warnings.filterwarnings("ignore")
```

```
In [107…  # takes a path to a CSV file and reads the data.
          noise = pd.read_csv('D:/Master/Semester3/noise_data_test.csv')
```

```
In [108…  # displays the first ten rows of the dataframe by default
          noise.head(10)
```

Out[108]:

	Unnamed: 0	timestamp	max_slow	type	model
0	0	1/03/2022 0:03	70.800003	B738	Boeing 737-800
1	1	1/03/2022 0:05	69.199997	B738	Boeing 737-800
2	2	1/03/2022 0:06	69.599998	B738	Boeing 737-800
3	3	1/03/2022 0:09	71.500000	B738	Boeing 737-800
4	4	1/03/2022 0:11	70.800003	B738	Boeing 737-800
5	5	1/03/2022 0:22	71.099998	B738	Boeing 737-800
6	6	1/03/2022 0:44	71.400002	B738	Boeing 737-800
7	7	1/03/2022 0:46	73.500000	B77W	Boeing 777-300ER
8	8	1/03/2022 0:57	69.300003	B738	Boeing 737-800
9	9	1/03/2022 1:07	62.700001	E35L	Embraer Legacy 600

```
In [109…  # prints information about the DataFrame
          noise.info()

          <class 'pandas.core.frame.DataFrame'>
          RangeIndex: 6673 entries, 0 to 6672
          Data columns (total 5 columns):
           #   Column      Non-Null Count  Dtype
          ---  ------      --------------  -----
           0   Unnamed: 0  6673 non-null   int64
           1   timestamp   6673 non-null   object
           2   max_slow    6673 non-null   float64
           3   type        6673 non-null   object
           4   model       6673 non-null   object
          dtypes: float64(1), int64(1), object(3)
          memory usage: 260.8+ KB
```

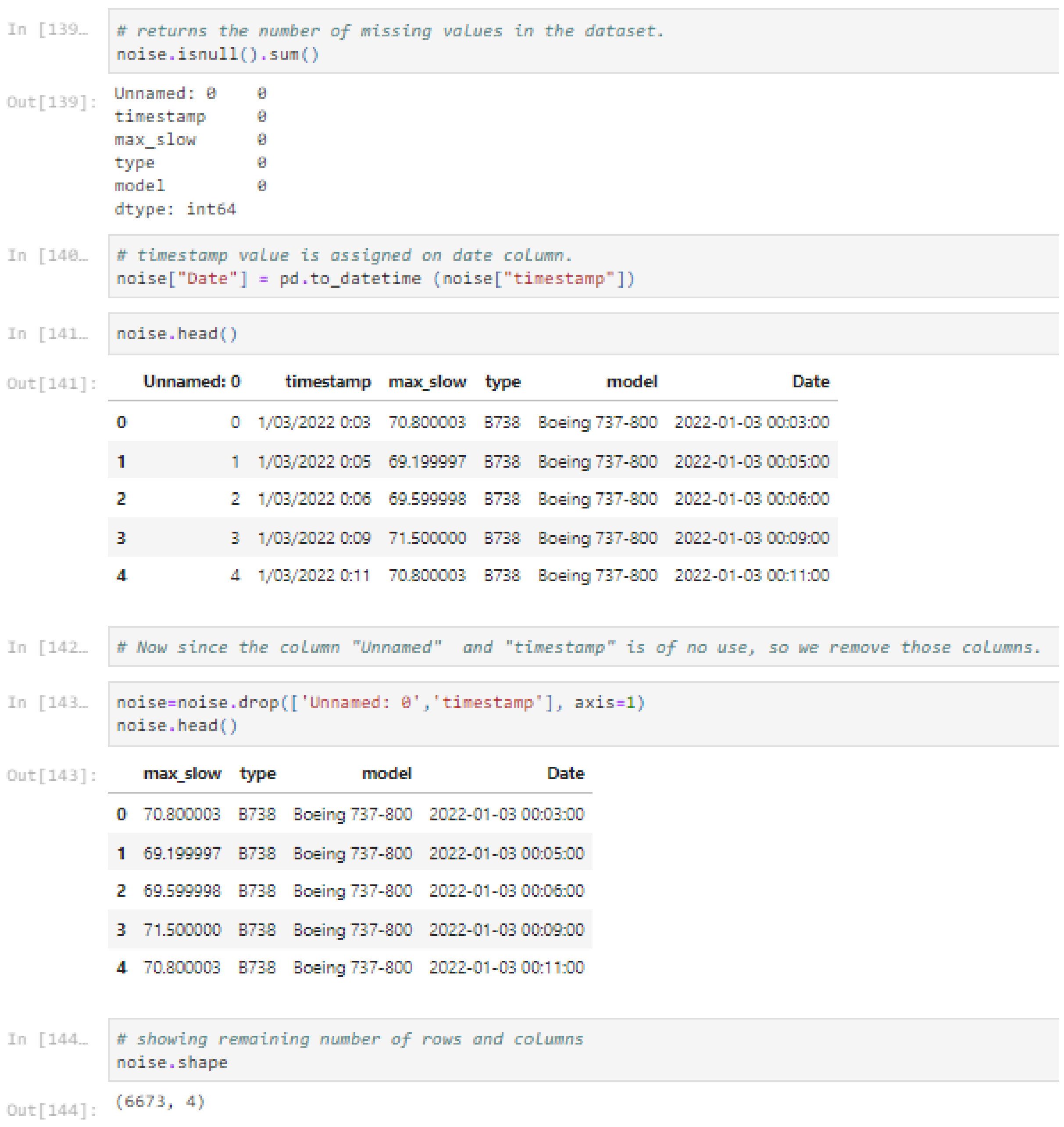
```
In [110…  # shows number of rows and column
          noise.shape

Out[110]: (6673, 5)
```

```
In [111…  # returns description of the data in the DataFrame
          noise.describe()
```

Out[111]:

	Unnamed: 0	max_slow
count	6673.000000	6673.000000
mean	3336.000000	68.477087
std	1926.473505	4.509026
min	0.000000	50.599998
25%	1668.000000	65.099998
50%	3336.000000	68.300003
75%	5004.000000	72.000000
max	6672.000000	91.500000

Figure 9: Data Exploration



```
In [139…  # returns the number of missing values in the dataset.
          noise.isnull().sum()

Out[139]: Unnamed: 0    0
          timestamp     0
          max_slow      0
          type          0
          model         0
          dtype: int64

In [140…  # timestamp value is assigned on date column.
          noise["Date"] = pd.to_datetime (noise["timestamp"])

In [141…  noise.head()
```

Out[141]:

	Unnamed: 0	timestamp	max_slow	type	model	Date
0	0	1/03/2022 0:03	70.800003	B738	Boeing 737-800	2022-01-03 00:03:00
1	1	1/03/2022 0:05	69.199997	B738	Boeing 737-800	2022-01-03 00:05:00
2	2	1/03/2022 0:06	69.599998	B738	Boeing 737-800	2022-01-03 00:06:00
3	3	1/03/2022 0:09	71.500000	B738	Boeing 737-800	2022-01-03 00:09:00
4	4	1/03/2022 0:11	70.800003	B738	Boeing 737-800	2022-01-03 00:11:00

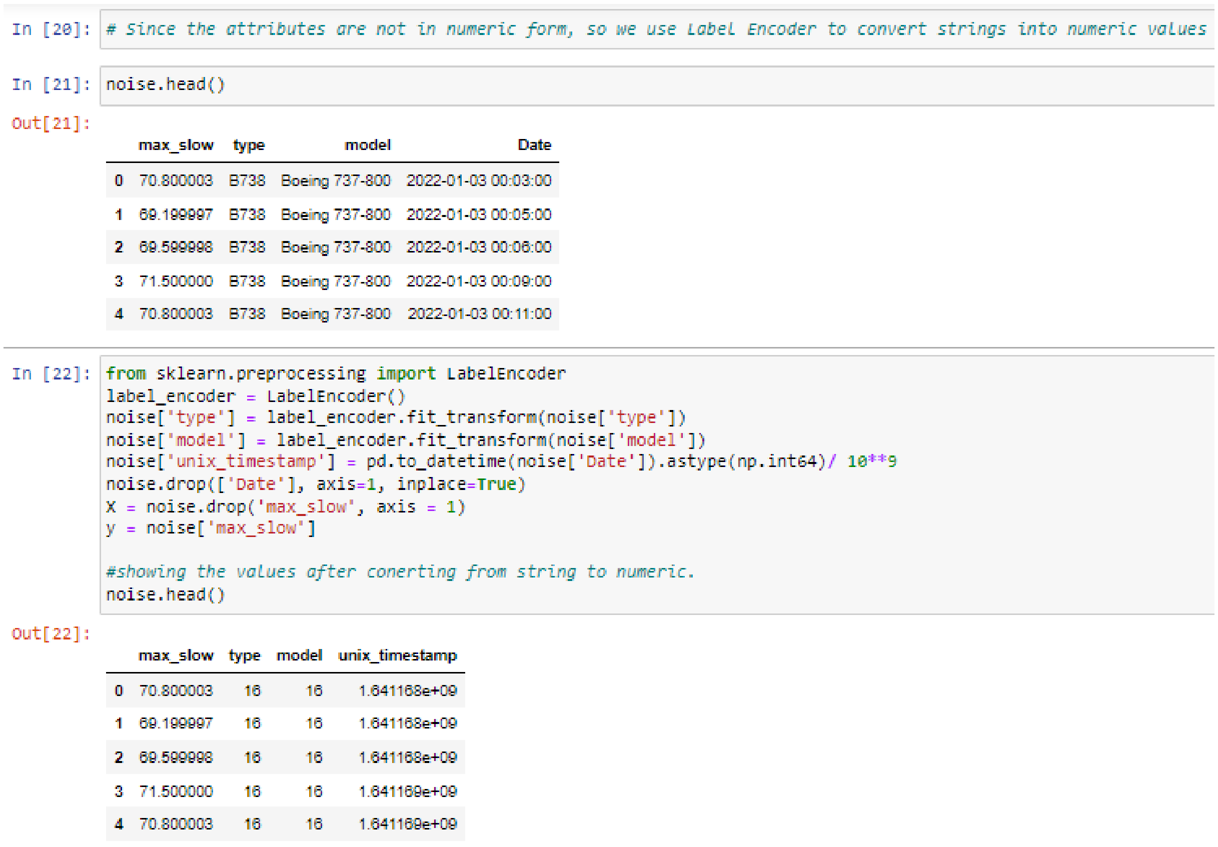
```
In [142…  # Now since the column "Unnamed"  and "timestamp" is of no use, so we remove those columns.

In [143…  noise=noise.drop(['Unnamed: 0','timestamp'], axis=1)
          noise.head()
```

Out[143]:

	max_slow	type	model	Date
0	70.800003	B738	Boeing 737-800	2022-01-03 00:03:00
1	69.199997	B738	Boeing 737-800	2022-01-03 00:05:00
2	69.599998	B738	Boeing 737-800	2022-01-03 00:06:00
3	71.500000	B738	Boeing 737-800	2022-01-03 00:09:00
4	70.800003	B738	Boeing 737-800	2022-01-03 00:11:00

```
In [144…  # showing remaining number of rows and columns
          noise.shape

Out[144]: (6673, 4)
```

Figure 10: Data Exploration (Contd.)



```
In [20]: # Since the attributes are not in numeric form, so we use Label Encoder to convert strings into numeric values
```

```
In [21]: noise.head()
```

Out[21]:

	max_slow	type	model	Date
0	70.800003	B738	Boeing 737-800	2022-01-03 00:03:00
1	69.199997	B738	Boeing 737-800	2022-01-03 00:05:00
2	69.599998	B738	Boeing 737-800	2022-01-03 00:06:00
3	71.500000	B738	Boeing 737-800	2022-01-03 00:09:00
4	70.800003	B738	Boeing 737-800	2022-01-03 00:11:00

```
In [22]: from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
noise['type'] = label_encoder.fit_transform(noise['type'])
noise['model'] = label_encoder.fit_transform(noise['model'])
noise['unix_timestamp'] = pd.to_datetime(noise['Date']).astype(np.int64)/ 10**9
noise.drop(['Date'], axis=1, inplace=True)
X = noise.drop('max_slow', axis = 1)
y = noise['max_slow']

#showing the values after conerting from string to numeric.
noise.head()
```

Out[22]:

	max_slow	type	model	unix_timestamp
0	70.800003	16	16	1.641168e+09
1	69.199997	16	16	1.641168e+09
2	69.599998	16	16	1.641168e+09
3	71.500000	16	16	1.641169e+09
4	70.800003	16	16	1.641169e+09

Figure 11: Data Exploration (Contd.)

The project starts with data pre-processing and exploration, where the raw dataset is meticulously prepared for analysis. This involves importing the libraries like **NumPy, pandas, matplotlib, seaborn** using **import** function, loading, and reading the dataset using **pd.read_csv** function, checking dataframe information using **noise.info()** function, checking number of columns and rows by using **noise.shape** function, checking summary statistics using **noise.describe()**, checking missing values using **noise.isnull()** function to ensure data integrity. There were zero missing value in the aircraft dataset. Little bit of restructure of dataset was done, timestamp value is assigned to date column by adding the date column and dropping the timestamp and unnamed column. Checking the target variable (**max_slow**) distribution by using **sns.displot(noise, x="max_slow").** Transformations like encoding categorical variables and normalising numerical features are carried out using **LableEncoder ()** function imported from sklearn library.

Data exploration is done to extract meaningful insights. Descriptive statistics provide a summary of key metrics, while data visualisation techniques like scatter plots, density plots, box plots and histograms uncover trends and relationships. Correlation analysis helps identify relevant variables, and exploratory data analysis reveals hidden patterns and anomalies that inform subsequent steps. Below is some visualisation without using gradient boosting algorithm.

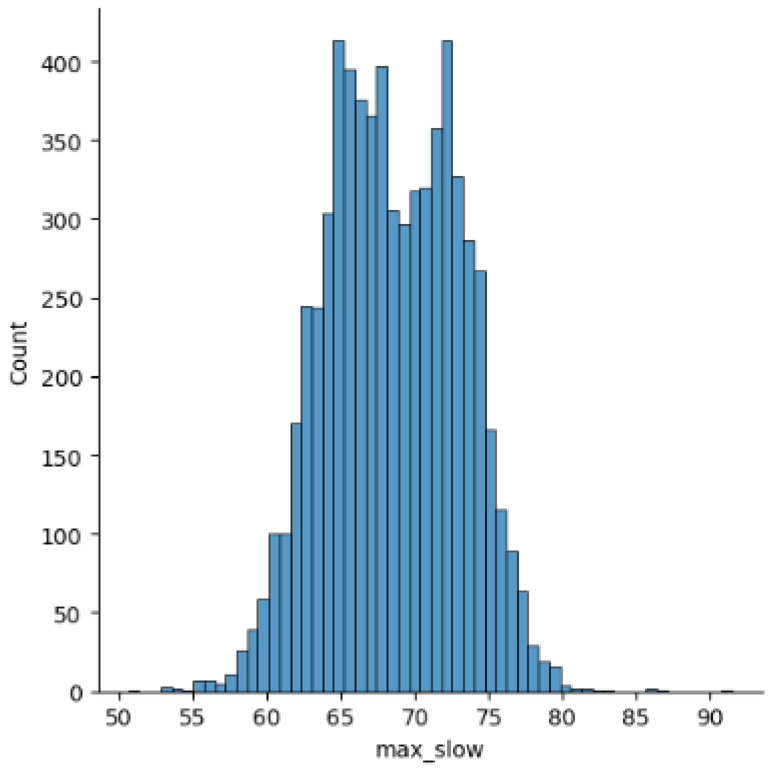


Figure 12: checking distribution of target variable (max_slow)

The above figure shows the distribution of max_slow column. In the figure, the distribution shows minimum value is at 50dB and maximux value is at 91dB.

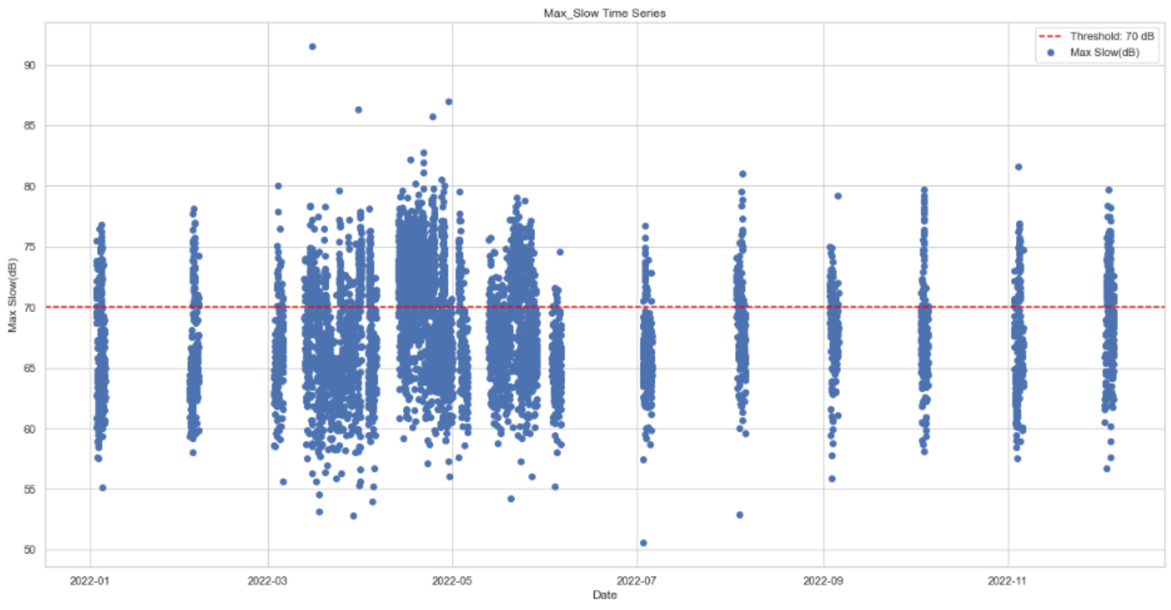
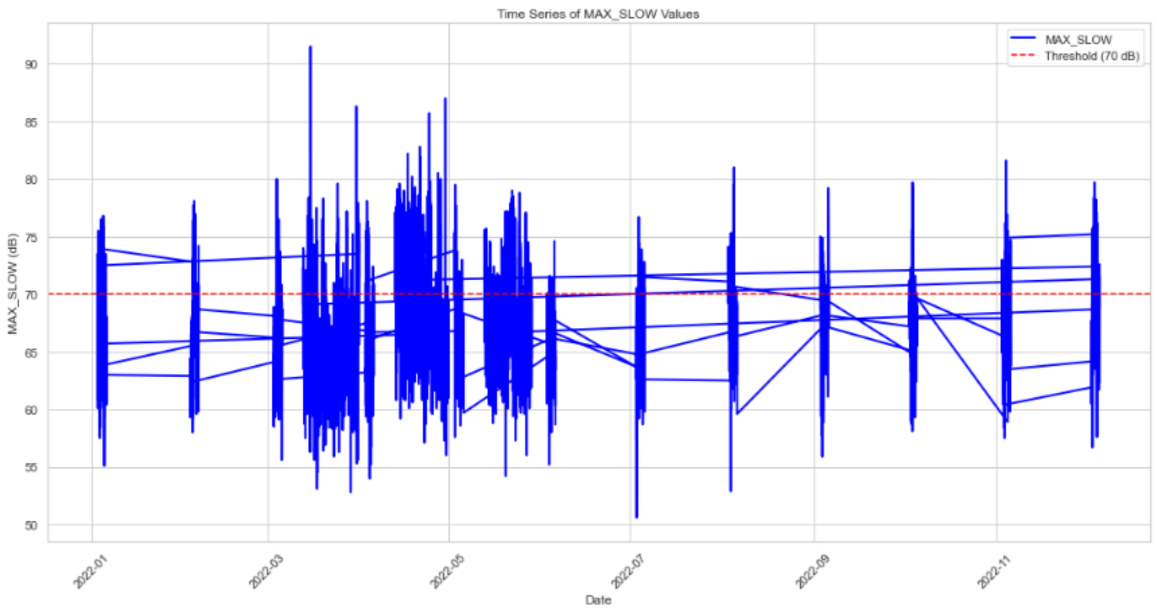


Figure 13: Scatter plot showing noise level according to date

The scatter plot show times series plotting the “max_show” noise value against dates where a red dashed threshold line at 70 decibels is added to visualise the limit.



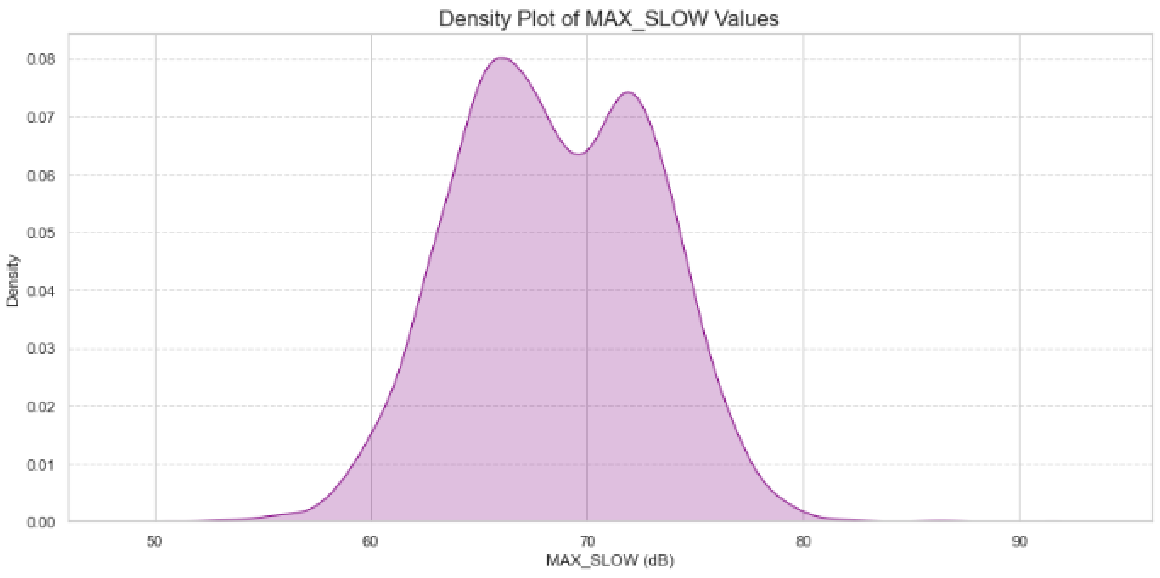


Figure 14: Density plot showing noise level

The density plot displays the distribution of noise levels, emphasizing the density of data along the decibel scale.

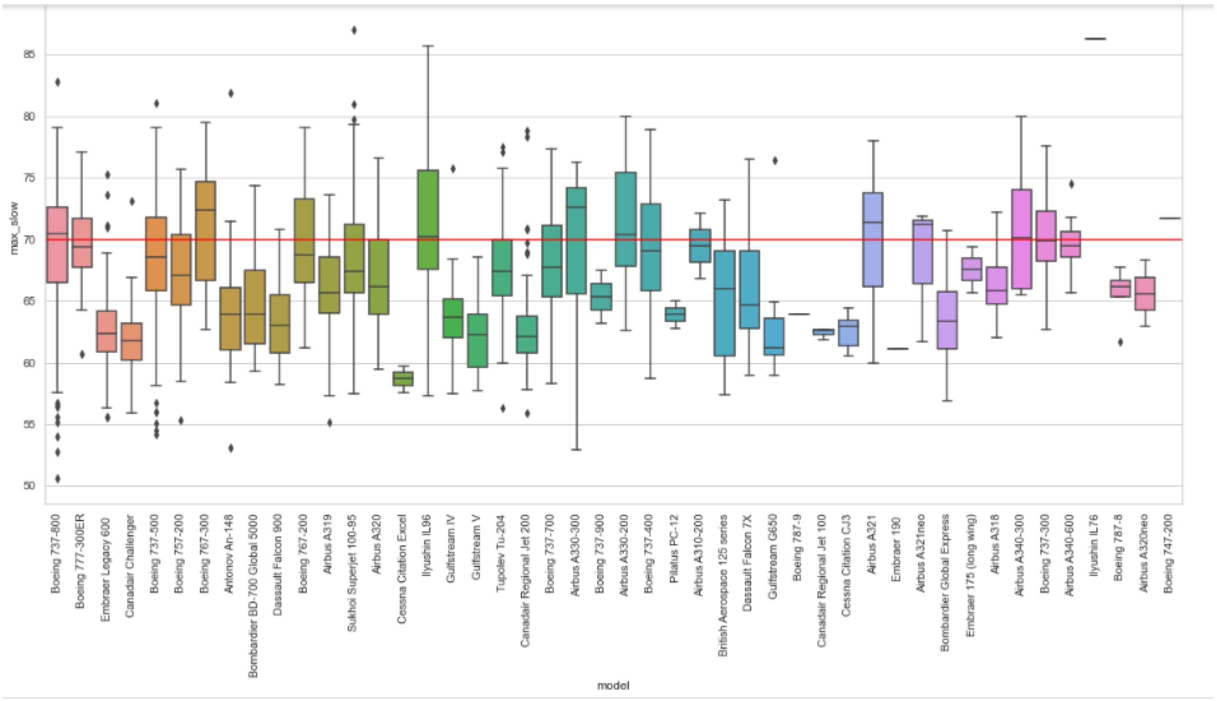


Figure 15:Box plot showing different aircraft model noise level

The box plot compared 'max_show' noise levels among different aircraft model categories.

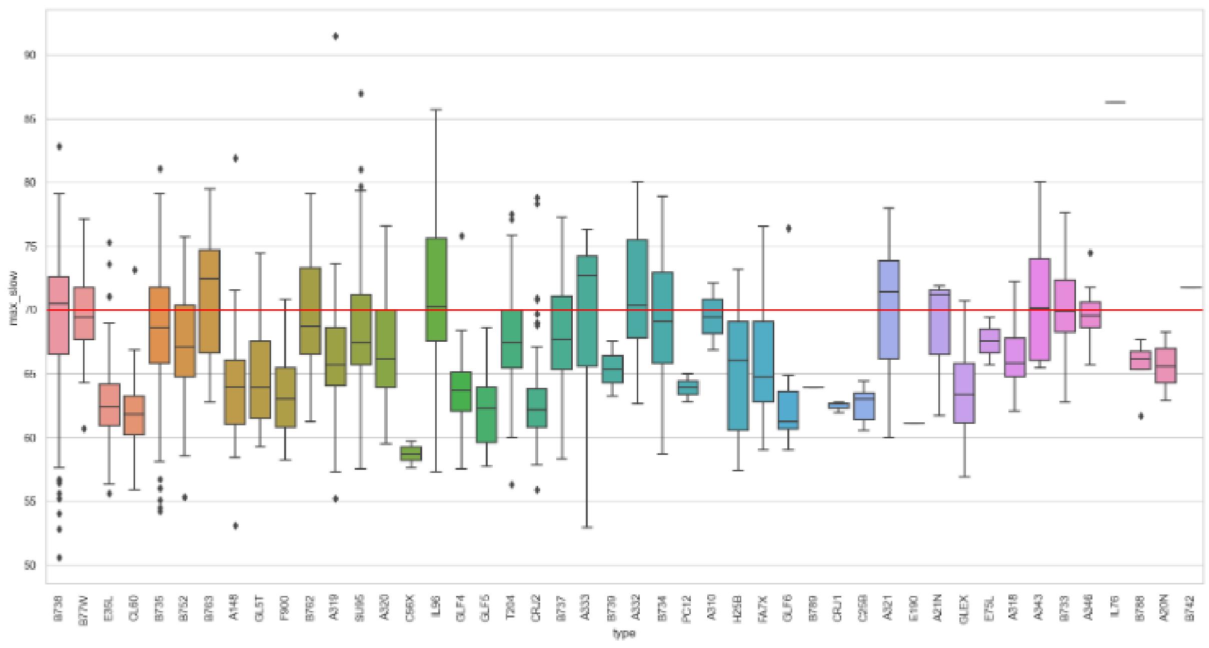
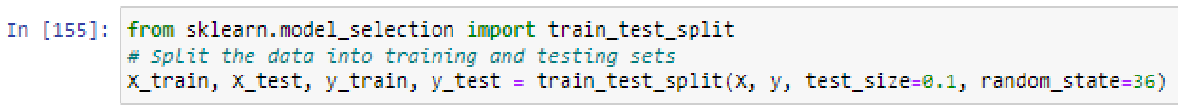


Figure 16 : Box plot showing different aircraft type noise level

The box plot compared ‘max_show’ noise levels among different aircraft type categories.

3.1.3 Building Model

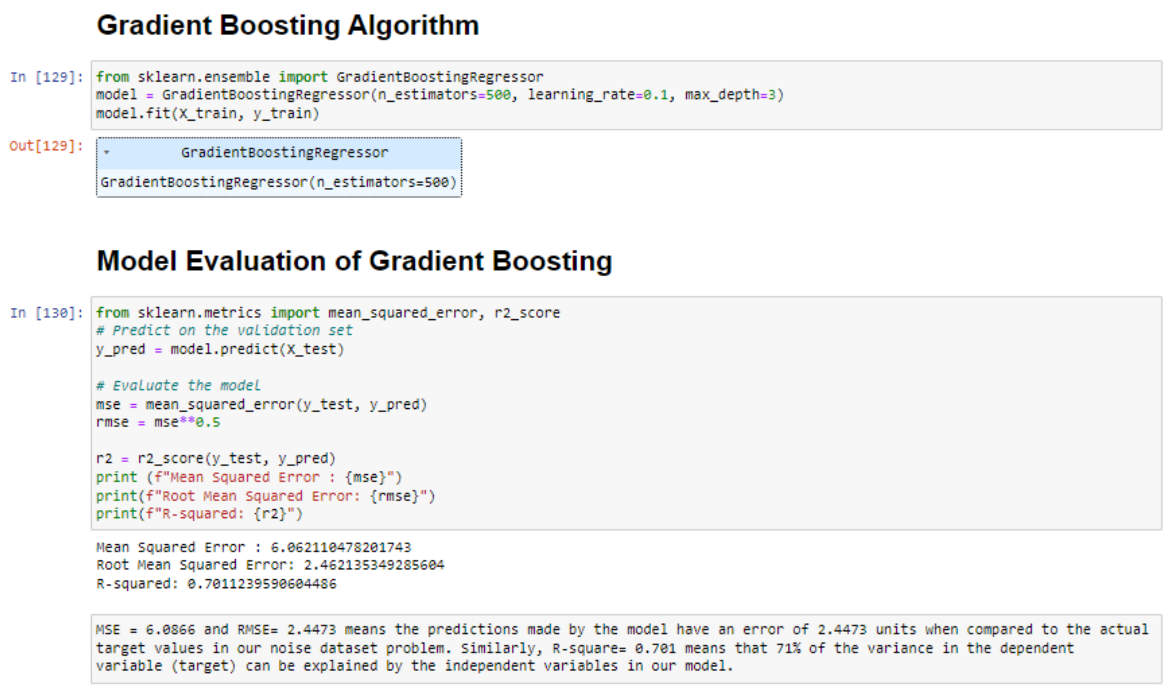


```
In [155]: from sklearn.model_selection import train_test_split
          # Split the data into training and testing sets
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=36)
```

Figure 17: Dividing training and testing dataset

After data exploration, the predictive models were developed. Features selected based on insights gained from data analysis are used to split the data into training and testing sets. Suitable machine learning algorithms like Gradient Boosting is chosen and trained on the training dataset, with hyperparameters optimised for performance. Here in the figure above, the aircraft dataset is divided into 90% training data and 10% testing data.

3.1.4 Model Evaluation



Gradient Boosting Algorithm

```
In [129]: from sklearn.ensemble import GradientBoostingRegressor
          model = GradientBoostingRegressor(n_estimators=500, learning_rate=0.1, max_depth=3)
          model.fit(X_train, y_train)

Out[129]:    GradientBoostingRegressor
          GradientBoostingRegressor(n_estimators=500)
```

Model Evaluation of Gradient Boosting

```
In [130]: from sklearn.metrics import mean_squared_error, r2_score
          # Predict on the validation set
          y_pred = model.predict(X_test)

          # Evaluate the model
          mse = mean_squared_error(y_test, y_pred)
          rmse = mse**0.5

          r2 = r2_score(y_test, y_pred)
          print (f"Mean Squared Error : {mse}")
          print(f"Root Mean Squared Error: {rmse}")
          print(f"R-squared: {r2}")

          Mean Squared Error : 6.062110478201743
          Root Mean Squared Error: 2.462135349285604
          R-squared: 0.7011239590604486
```

```
MSE = 6.0866 and RMSE= 2.4473 means the predictions made by the model have an error of 2.4473 units when compared to the actual
target values in our noise dataset problem. Similarly, R-square= 0.701 means that 71% of the variance in the dependent
variable (target) can be explained by the independent variables in our model.
```

Figure 18: Model Evaluation done using gradient boosting algorithm

Model evaluation on the testing dataset reveals its effectiveness in making predictions, guided by metrics such as accuracy and precision. Interpretation of the model's outcomes provides insights into the importance of various features. Depending on project goals, the option to deploy the model for real-time predictions is also explored. After training the model, I must check to see how it's performing. This is done by testing the performance of the model on previously unseen data. The unseen data used is the testing set that was split into my data earlier. If testing was done on the same data which is used for training, I will not get an accurate measure, as the model is already used to the data, and finds the same patterns in it, as it previously did. This will give me disproportionately high accuracy. When used on testing data, I got an accurate measure of how my model will perform and its speed. The model parameters are set as n_estimator(number of tress) is set 500, learning_rate 0.1 and max_depth 3. After changing the hyperparameter many time to find the best values for settings to optimize the model's performance. I have set the hyper tuning parameter as seen in the figure above. Model evaluation is done by using the testing dataset, which gave mean square error (MSE) is 6.06, root mean squared error (RMSE) is 2.462 means the predictions made by the model have an error of 2.462 units when compared to the actual target values in my noise dataset problem. Similarly, R-square (R^2) is 0.701 means that 71.12% of the variance in the dependent variable (target) can be explained by the independent variables in my model.

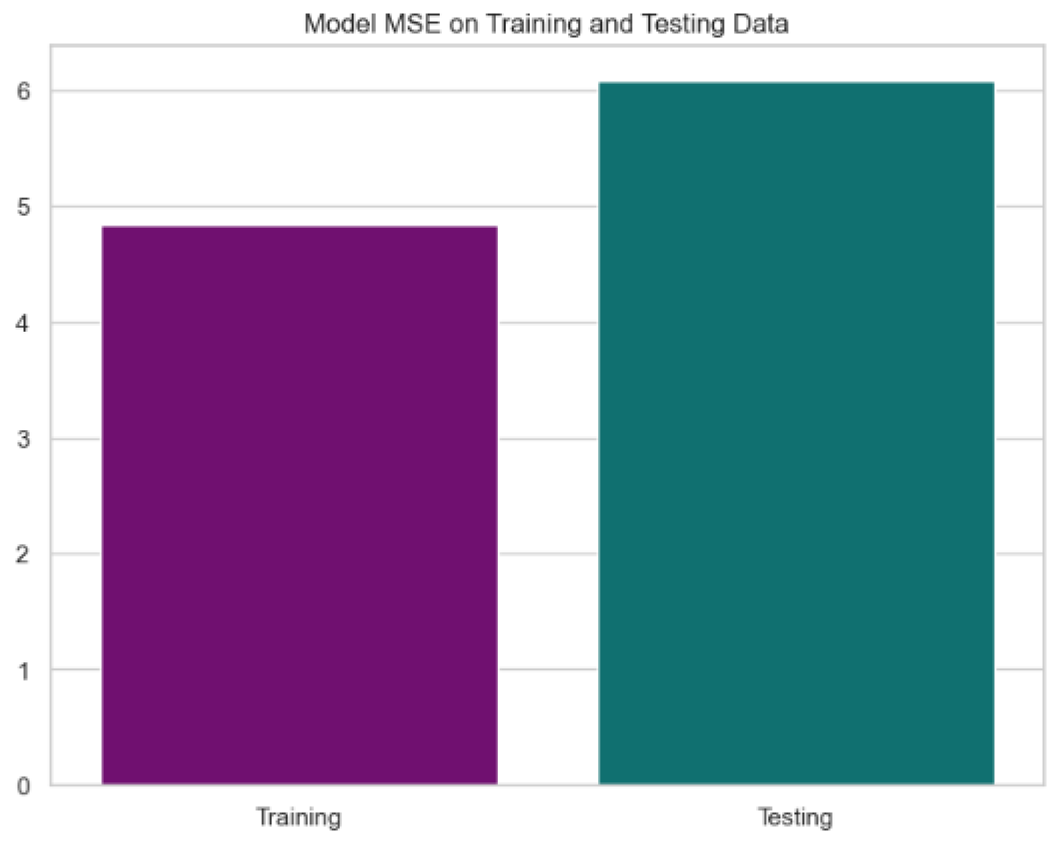
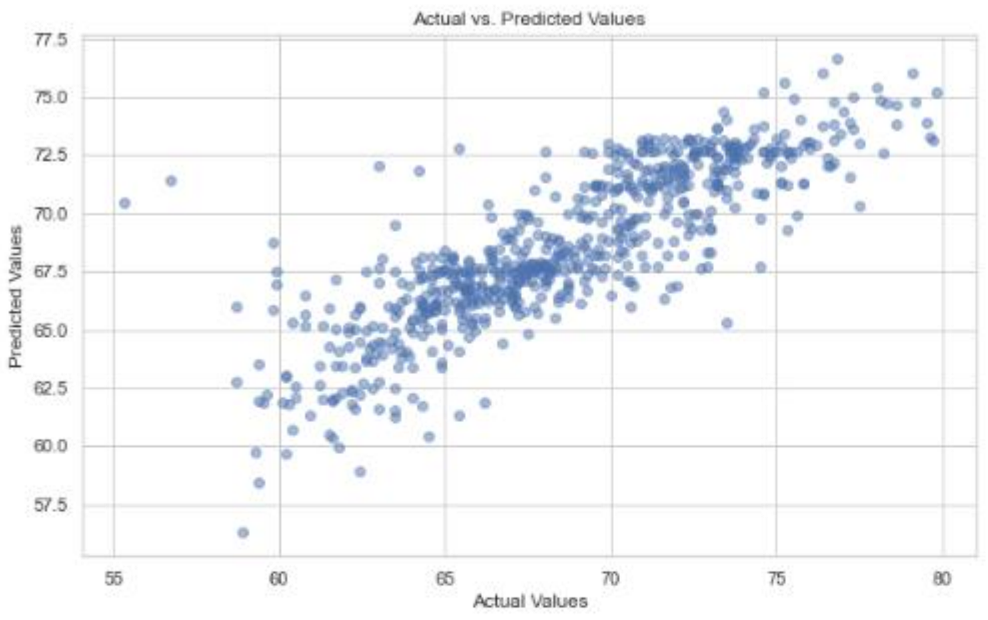


Figure 19: Model Evaluation of Training and Testing Dataset

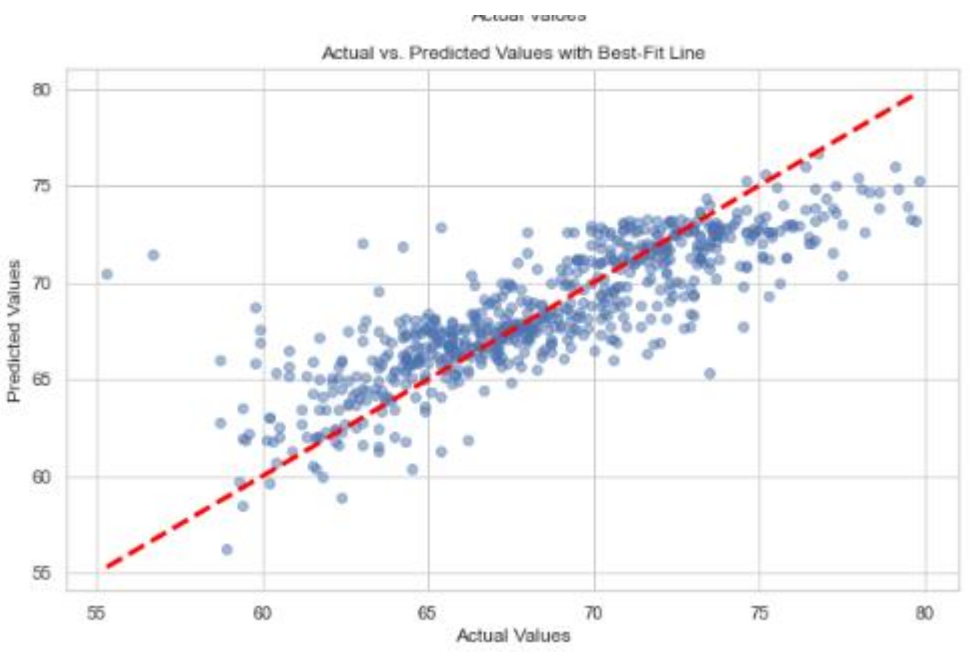
In my mode, I obtained the following MSE values of Training dataset as 4.63655823491841 and Testing MSE as 5.7913990496039505.The training MSE (4.63655823491841) is lower than the testing MSE (5.7913990496039505), which is generally expected. It suggests that the model can fit the training data quite well, and its predictions on the training set have lower error compared to predictions on unseen data (testing set).The difference between the training and testing MSE is not too large, which is a good sign. It indicates that the model is not significantly overfitting to the training data.

3.1.5 Prediction

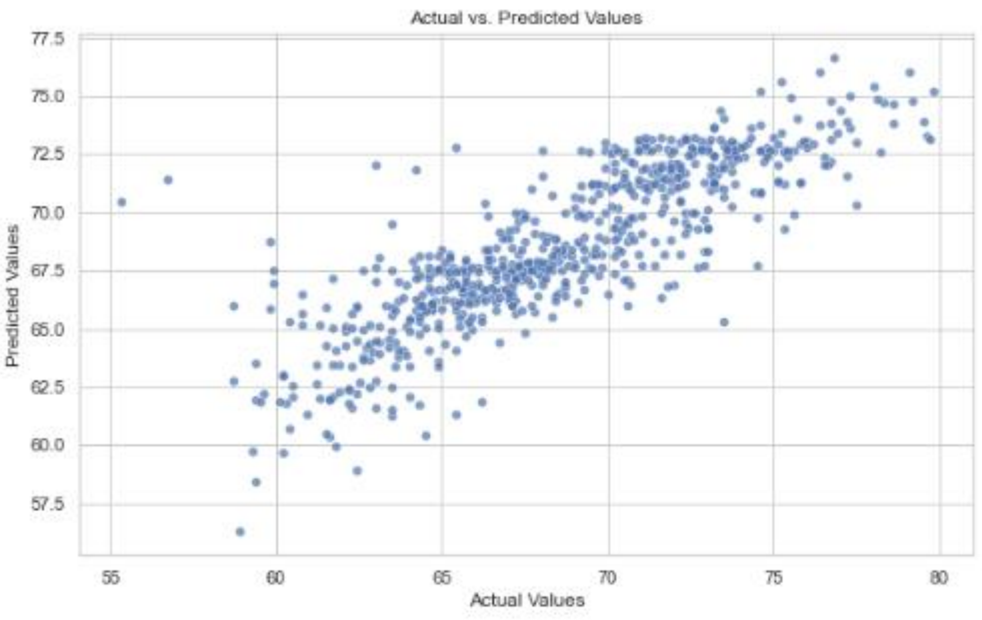
Once model have been created and evaluated, now its accuracy is checked to see whether it can be improved in any way. This is done by tuning the parameters present in the model. Parameters are the variables in the model that the programmer generally decides. At a particular value of my parameter, the accuracy will be the maximum. Parameter tuning refers to finding these values. In the end, I can use my model on unseen data to make predictions accurately. The matplotlib library is used to create visual representation comparing the actual values (y_test) with the predicted values (y_pred).



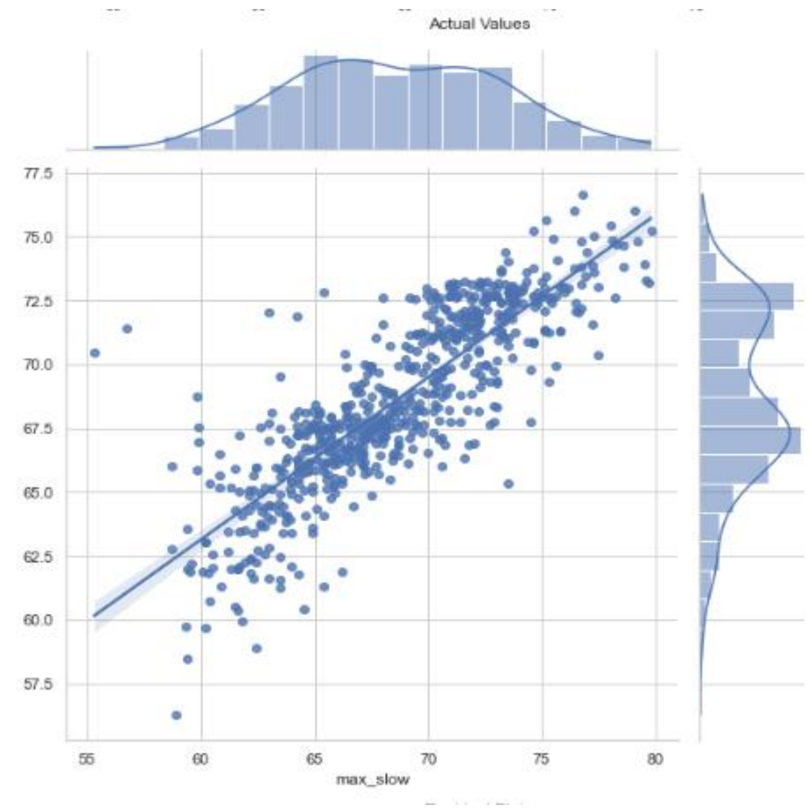
Each point on the plot represents a pair of actual and predicted values. The transparency of the point is set to 0.5 for better visualisation. This scatter plot indicates how closely the model's predictions align with the actual data.

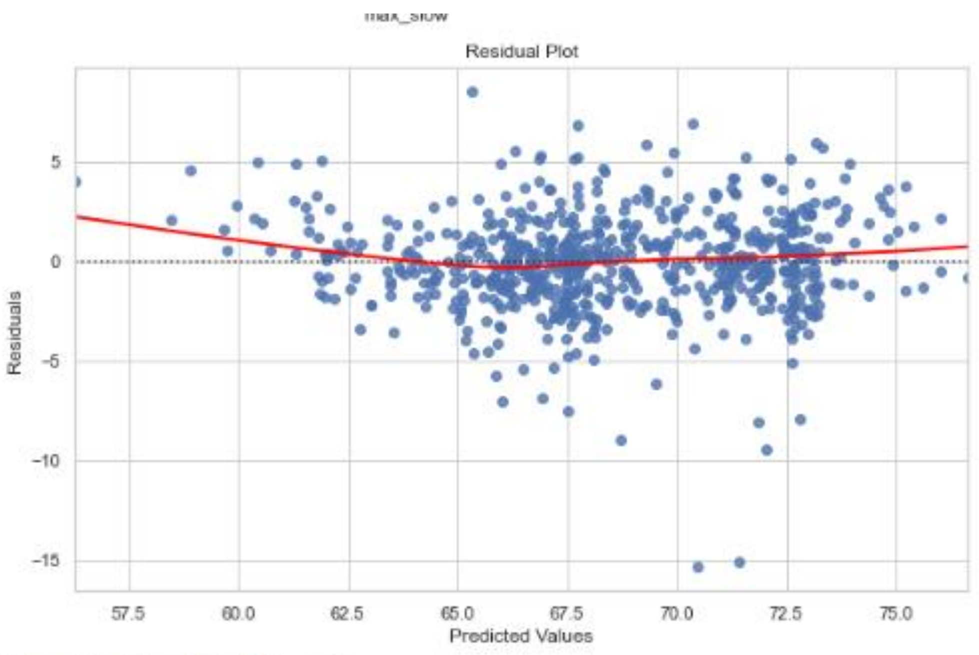


In this scatter plot the best-fit line is added. This best-fit line is presented as a red dashed line, running diagonally from the bottom-left corner to the top-right corner of the plot. It represents a precise line between actual and predicted values as a reference line. Data point variations from this line indicate whether the model tends to overpredict or underpredict.



The model predicts the target variable (y_pred) based on the test features (X_test). Then, key evaluation metrics are computed, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2).





```
Root Mean Squared Error: 2.46
R-squared: 0.70
```

The residual plot represents the differences between the actual and predicted values. This plot helps in identifying patterns or trends in the model's errors. The red line on the plot represents a smoothed curve, offering an indication of any underlying structure in the residuals.

Code used to produce results can be found in **Appendix C6: Comparing the Actual Values with the Predicted Values.**

Similarly, the aircraft noise level and aircraft maintenance prediction can be found in section **Result and Achievement.**

3.2 Constraint and Limitations

The constraint of the project could be reliance on the historical data. if the dataset is not up to date, it may not accurately reflect current trends or advancement in technical advancement in aircraft noise reduction. Likewise, the project’s findings may be specific to the dataset and may not generalise well to different regions or airports. Similarly, factors like weather conditions, proximity to urban areas which influence noise levels were not included in the dataset.

3.3 Original Contribution

The project comprehensively analysed different aircraft model noise pollution. Machine learning algorithms help to determine which aircraft models produce more noise pollution. It helps the aviation authorities to solve the issue at the beginning phase and send the aircraft for maintenance.

3.4 Understanding of Issues

The project focuses on gaining a deep understanding of issues related to aircraft noise pollution. It requires extensive research and analysis to identify key factors that influence pollution levels, including aircraft types, operational procedures, and aircraft models. It is important to understand these issues to make aircraft eco-friendly.

3.5 Technical Achievement

This project's technical achievement resides in its machine learning algorithms and data analysis techniques to resolve the complex problem of aircraft noise pollution. The utilisation of visualisation tools such as Matplotlib and Seaborn demonstrates a skilful use of data visualisation to improve interpretation and communication of findings. Likewise, the pre-processing stages, including label encoding and feature engineering, demonstrate an in-depth comprehension of data preparation for machine learning tasks.

3.6 Evaluation and Discussion

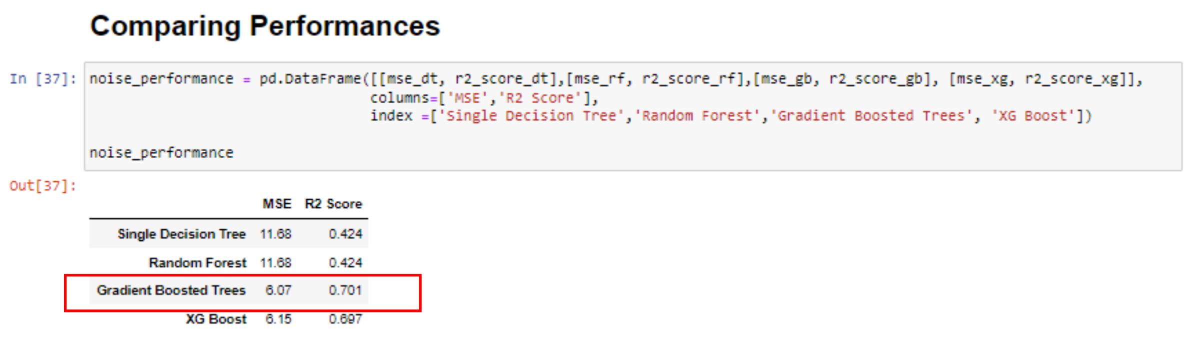
The evaluation and discussion of the project consist of several aspects. It effectively utilises regression models such as Gradient Boosting. Likewise, data processing, label encoding is implemented properly. The use of evaluation metrics MSE, RMSE and R^2 provide quantitative measures of the model performance. Furthermore, comparing Gradient Boosting with Simple Decision Tree, Random Forest and XGBoost reveals a minor advantage for Gradient Boosting, as indicated by a lower MSE and a higher R^2, indicating its potential as the preferred model for this task.

4 Critical Appraisal

4.1 Result and Achievement

The project utilizes machine learning regression models to accurately predict noise pollution levels produced by the aircraft. The comparison between Gradient Boosting, Decision Tree, Random Forest and XGBoost models has provided critical insights into algorithmic performance, recommending Gradient Boosting due to its slightly lower Mean Squared Error (MSE) and higher coefficient of determination (R^2).

4.1.1 Comparison of Performance of Different Algorithms with Gradient Boosting



Comparing Performances

```
In [37]: noise_performance = pd.DataFrame([[mse_dt, r2_score_dt],[mse_rf, r2_score_rf],[mse_gb, r2_score_gb], [mse_xg, r2_score_xg]],
                                   columns=['MSE','R2 Score'],
                                   index =['Single Decision Tree','Random Forest','Gradient Boosted Trees', 'XG Boost'])

         noise_performance
```

Out[37]:

	MSE	R2 Score
Single Decision Tree	11.68	0.424
Random Forest	11.68	0.424
Gradient Boosted Trees	6.07	0.701
XG Boost	6.15	0.697

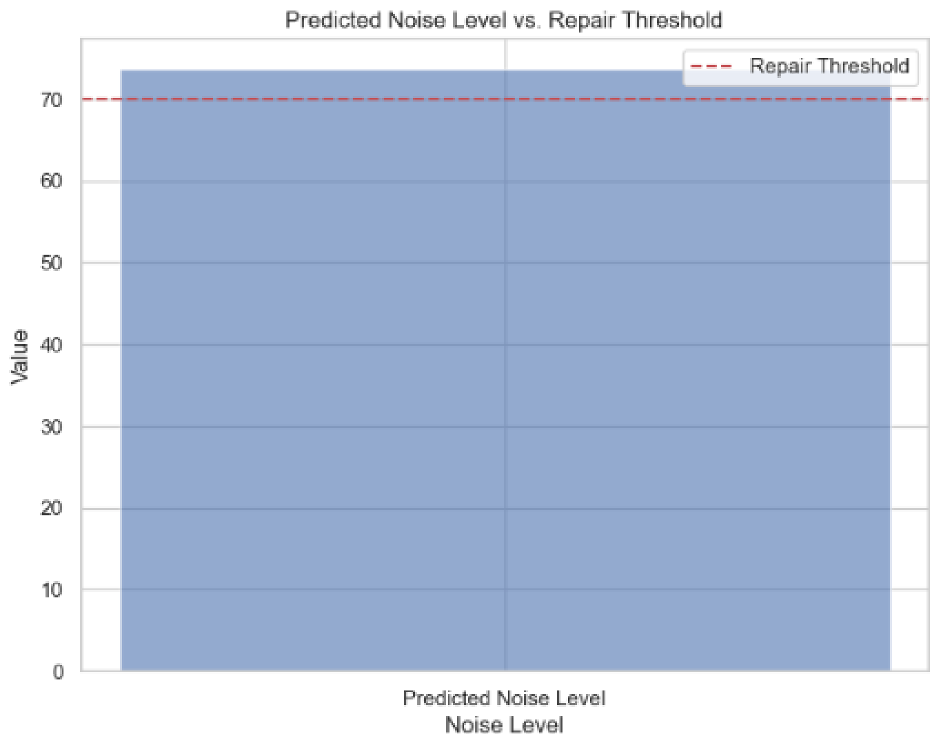
Figure 20: Comparison of different algorithm with gradient boosting algorithm

The comparison between the Single Decision Tree, Random Forest, Gradient Boosting (GB) and XGBoost algorithms in this project yielded valuable insights. Compared to Single Decision Tree, Random Forest and XGBoost, Gradient Boosting algorithm exhibited better training times, indicating comparable computational efficiency. However, in terms of predictive accuracy, gradient boosting demonstrated a slight advantage. It yielded a Mean Square Error (MSE) of 6.07 compared to Single Decision Tree MSE of 11.68, Random Forest MSE of 11.68 and XGBoost MSE of 6.15 indicating that Gradient Boosting provided more precise predictions. Additionally, Gradient Boosting outperformed in terms of coefficient of determination R-squared (R^2) value, with an R^2 of 0.701 compared to Single Decision Tree and Random Forest R^2 of 0.424, and XGBoost R^2 of 0.697. This suggests that gradient boost captured a higher proportion of variance in the noise pollution data, indicating its superior predictive power. These findings affirm the suitability of gradient boosting for this specific noise prediction task and underscore its potential for achieving more accurate and reliable results.

Code used to produce results can be found in **Appendix C7: Comparison of Different Algorithm with Gradient Boosting.**

4.1.2 Analysis of Noise Level of Aircraft Models

```
Input DataFrame:
   type  model  unix_timestamp
0    21     21    1.641182e+09
predicted noise level :  73.61920258789024
```



```
Input DataFrame:
   type  model  unix_timestamp
0    27     27    1.641173e+09
predicted noise level :  61.260728419245055
```

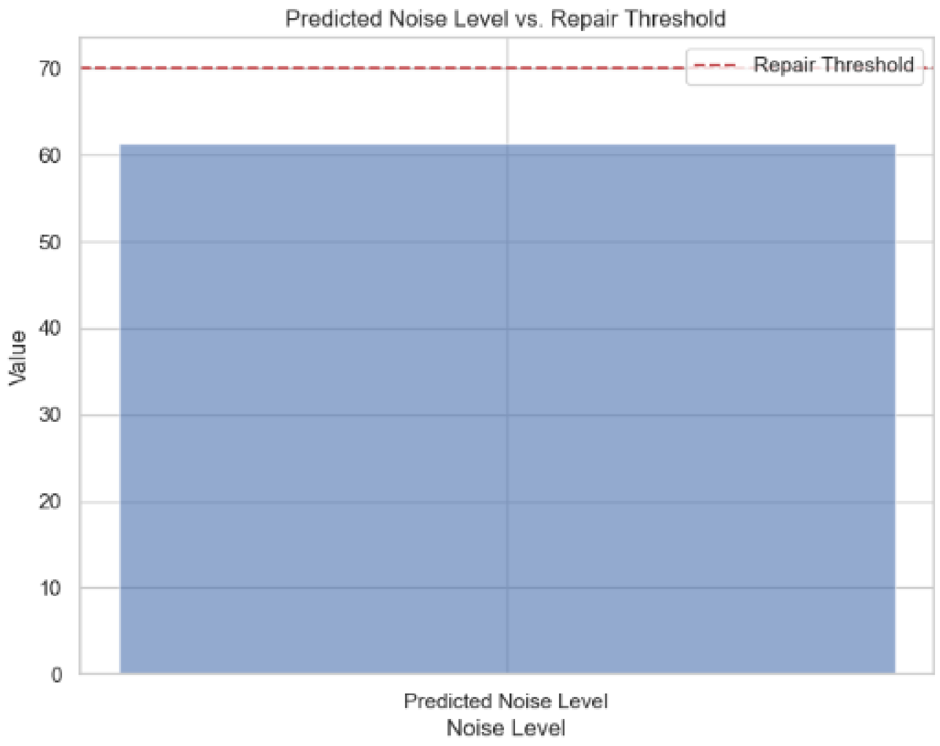
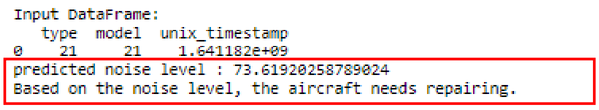


Figure 21: Predicting aircraft noise level

According to [6], The World Health Organization (WHO) recommends maintaining environmental noises below 70dB to prevent hearing loss. To predict the noise level produced by an aircraft model that exceeds threshold(70dB), machine learning model was trained on aircraft data. This model would consider various parameters such as aircraft type, aircraft model and date_time potential factors influencing noise levels. By inputting these variables into the trained model, it generates a prediction of the noise level. In the first figure, the noise level above threshold is predicted for aircraft model 21 and in second figure, the noise level less than threshold is predicted for aircraft model 27. Similarly in other models, noise level can also be predicted by inputting the required parameter. This prediction serves as a valuable tool for assessing whether the noise level of a particular aircraft model exceeds a predefined threshold.

Code used to produce results can be found in **Appendix C8: Analysis of Noise Level of Aircraft Models.**

4.1.3 Predicting Maintenance of Aircraft Model Exceeding Threshold



```
Input DataFrame:
   type  model  unix_timestamp
0    21     21    1.641182e+09
predicted noise level : 73.61920258789024
Based on the noise level, the aircraft needs repairing.
```

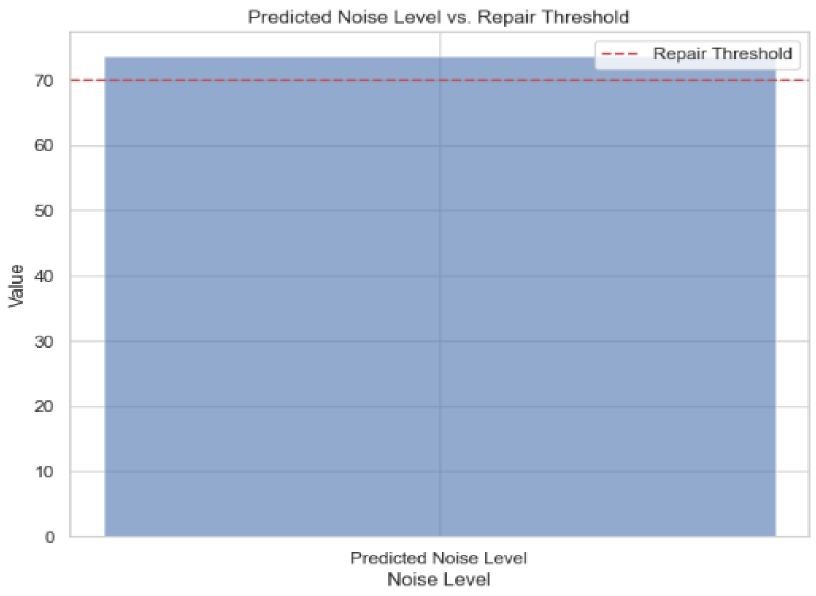


Figure 22: Predicting Maintenance of the aircraft model exceeding the threshold

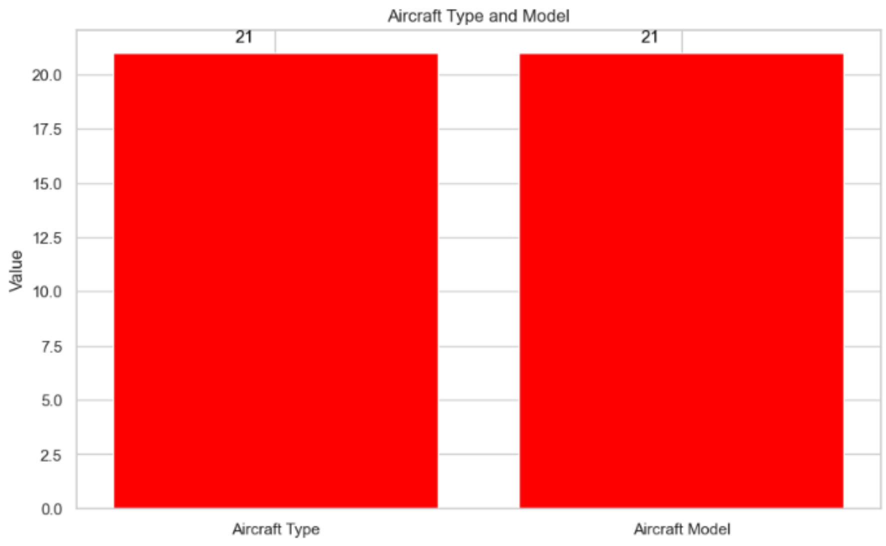
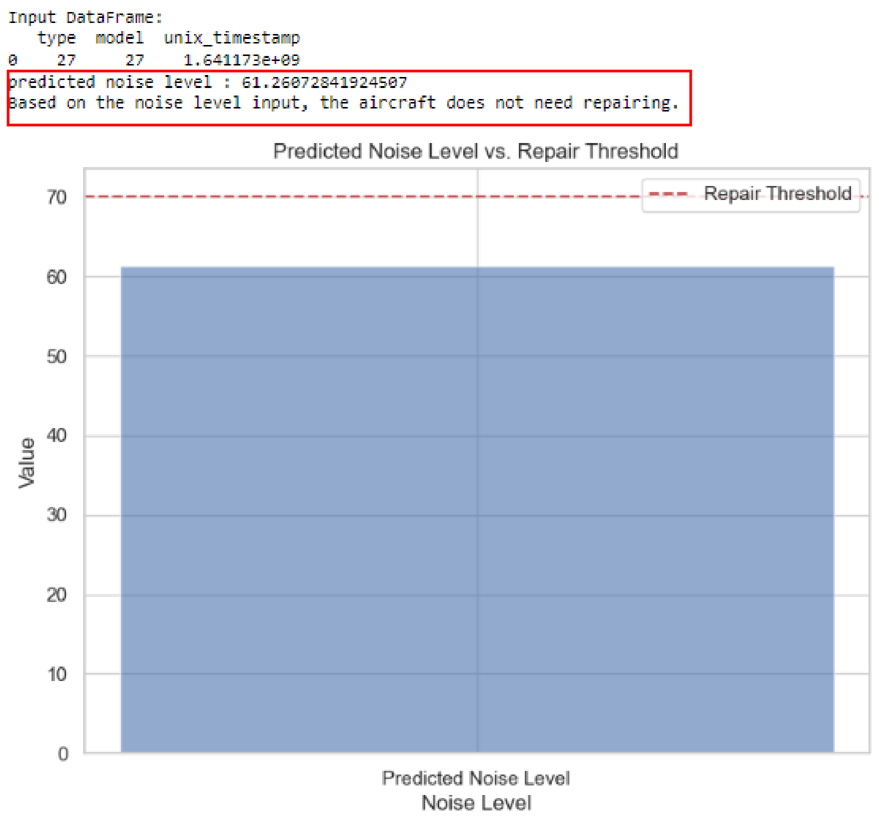
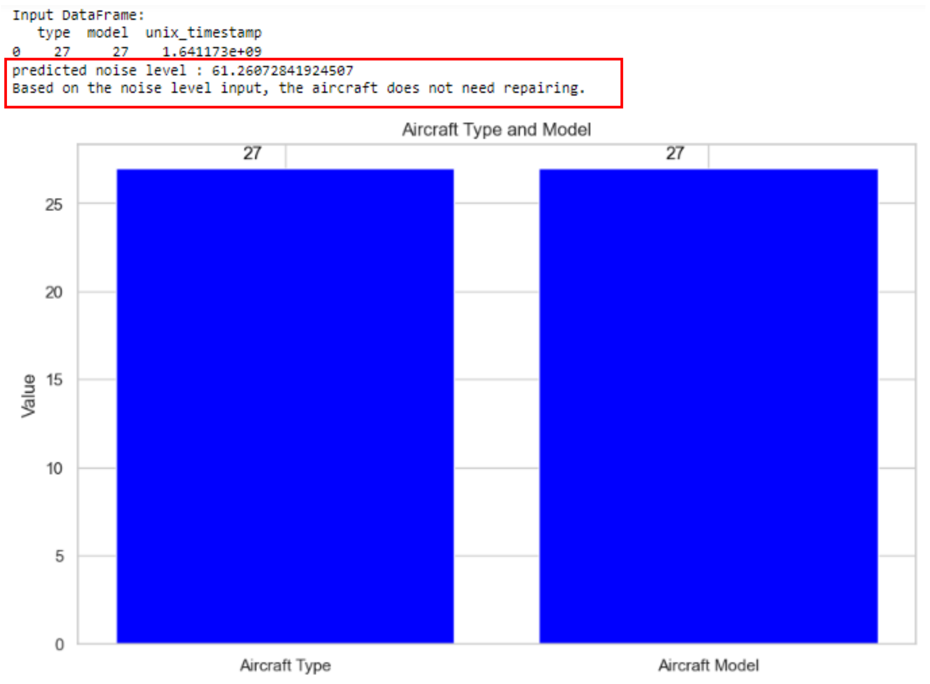


Figure 23: Showing Aircraft Model 21 and Aircraft Type 21 according to prediction



```
Input DataFrame:
   type  model  unix_timestamp
0    27     27    1.641173e+09
predicted noise level : 61.26072841924507
Based on the noise level input, the aircraft does not need repairing.
```

Figure 24: Predicting Maintenance not required by the aircraft model 27 exceeding the threshold



```
Input DataFrame:
   type  model  unix_timestamp
0    27     27    1.641173e+09
predicted noise level : 61.26072841924507
Based on the noise level input, the aircraft does not need repairing.
```

Figure 25: Showing Aircraft Model 27 and Aircraft Type 27

In the context of this project on aircraft noise pollution, predicting maintenance for aircraft models with noise levels exceeding threshold is an essential proactive measure for enhancing safety and reducing the probability of accidents. Using the machine learning models developed in this project, specifically the Gradient Boosting algorithms, noise levels for various aircraft models are predicted. These models are trained on aircraft historical data to reliably estimate noise emissions based on several variables, including aircraft type, aircraft model and date_time. Once the models have been developed, they predict aircraft noise levels. When the decibel level of an aircraft exceeds the predetermined threshold, it predicts that maintenance is required. In the above figure, aircraft model 21 noise level is above threshold so based on noise level produced by the aircraft model, the aircraft model needs maintenance. Likewise, aircraft model 27 noise level is below threshold so based on noise level produced by the aircraft model, the aircraft model does not need maintenance. Similarly, in same way other remaining aircraft model, maintenance can also be predicted. This proactive approach enables airlines and maintenance personnel to proactively address potential issues before they worsen, thereby reducing the likelihood of incidents caused by aircraft malfunctions or failures.

Code used to produce results can be found in **Appendix C9: Predicting Maintenance of Aircraft Model Exceeding Threshold.**

4.2 Legal Issues

Machine Learning in aviation comes with many benefits, but it also brings some legal challenges. These include concerns about keeping data safe and private, figuring out who's responsible if something goes wrong with the machine learning systems, and making sure these systems follow safety rules. There's also the issue of machine learning models unintentionally favouring certain groups, creating autonomy, self-aware, the need to protect new technology ideas, and making sure decisions made by machines are ethical [42].These systems might need special certifications, but different countries might have different rules. We also need to make sure we can understand and explain how these machines make decisions. Finally, people who work with these systems, like pilots and air traffic controllers, might need extra training and certifications. To tackle these challenges, it's important for aviation experts, technology companies, lawyers, and regulators to work together and create clear rules for using machine learning in aviation, so it can be safe and responsible.

4.3 Social Issues

Aviation technology holds the promise of transformative advancements in safety, efficiency, customer service, and maintenance but holds potential social risks. Safety and reliability concerns emerge as machine learning algorithms, while powerful, may not always make flawless decisions, executing rigorous testing, validation, and transparent deployment. Bias and fairness issues manifest as algorithms may inadvertently perpetuate biases, impacting passenger screening and organizational hiring practices. Aircraft noise may lead to irritation in the community, disturb people's sleep, have a negative impact on children's school performance, and potentially raise the chances of cardiovascular problems for those residing near airports [16]. The potential for job displacement looms as automation driven by machine learning threatens various aviation professionals, including pilots, air traffic controllers, and maintenance crews. The digital gap widens as the rapid adoption of technology in aviation may leave smaller companies and regions struggling to keep pace as ongoing training and education for aviation professionals becomes vital, addressing the need for necessary skillsets. Furthermore, while machine learning can optimize efficiency, the aviation industry's environmental impact remains a significant concern.

4.4 Ethical Issues

Privacy is a major concern when handling vast amounts of passenger and operational data, necessitating clear policies and safeguards. It becomes a focal point due to the substantial data required for effective training, raising issues surrounding passenger data protection and security profiling. Security risks, data ownership, and sharing agreements also require attention [43]. Additionally, ethical decision-making during emergencies and environmental considerations in optimization processes must be considered as regulatory bodies might face the complexity of establishing suitable regulations that strike the right balance between innovation and safety which ultimately affect the people in general.

4.5 Wider Issues

The project's wider consequences extend beyond the prediction of aircraft noise. It explores wider environmental, social, and technological issues. It emphasizes the importance of sustainable aviation practices and technological innovation for the development of quieter, more environmentally friendly aircraft. Additionally, this project highlights the need for robust regulatory frameworks to control noise pollution. It promotes societal equity in the distribution of pollution burdens and sparks conversations about urban planning and land use policies. In addition, it raises awareness about potential health effects, highlighting the interdependence of aviation, the environment, and society. In essence, this initiative functions as a catalyst for a more holistic approach to addressing the complex challenges posed by aircraft noise pollution.

4.6 Personal Reflection

The research on aircraft noise pollution has been an extremely informative academic project, providing invaluable insights into the complexities of environmental impact analysis in the aviation industry. Likewise, surveys were taken to validate my findings on the how the global population were affected by aircraft noise. As a solution, the use of machine learning models, such as Gradient Boosting provided a robust framework for predicting noise levels that highlighted the potential of data-driven approaches to address complex environmental issues. By handling the complexities of data pre-processing and feature engineering, I gained a greater understanding of the crucial role data quality plays in the efficiency of a model. In addition, the comparative analysis of algorithms revealed the subtle distinctions between predictive models thus highlighting the real-world implications of algorithm selection.

Surveys conducted can be found in the **Appendix B: Survey Results.**

5 Conclusion and Reflection

5.1 Future Work

In the future, integrating real-time data feeds and advanced sensor technologies could improve the accuracy and reliability of aircraft noise pollution predictions. Likewise, exploring various machine learning models and even delving into deep learning techniques may further improve the ability to predict. The project has lot of potential to explore on which are promising such as:

- Predict flight paths and patterns of take-off, landing to prevent major issues such as accidents, air traffic and provide cost effective decisions.
- Weather prediction and chatbots integration to assist passengers during flight can be provided for safe and efficient travel.
- Autonomous flight rules can be integrated based on sensory flight data and various other datasets from each component for full automation of aircraft to reduce human-errors by a great margin.

5.2 Conclusion

In short, Aircraft noise is a major global issue that needs to be addressed in recent times. It affects more in environmental health. However, steps have been taken by many aviation companies to reduce the aircraft noise pollution in accordance with government rules and regulations, but this has not been an efficient solution until now. Hence, by using machine learning algorithm we can minimize such problem and provide safe experience for passengers as well as improve overall environmental health. This project has made significant progress in comprehending and addressing aircraft noise pollution. Using Gradient Boosting, it accurately predicts noise levels. In this context, the comparative analysis with other algorithms demonstrated that Gradient Boosting is preferable with more precision and accuracy. Data pre-processing and visualisation techniques played a pivotal role in optimising model performance. Ethical considerations were diligently addressed, ensuring fairness and transparency in predictions. While working with the dataset, I came with the conclusion that noise of aircraft is directly proportional to aircraft quality.

6 Bibliography

[1] K. M. Izaak Stanton, Ahsan Ikram, Murad El-Bakry, "Predictive maintenance analytics and implementation for aircraft: Challenges and opportunities," vol. 26, no. 2, pp. 216-237, 2022.

[2] Philip J. Wolfe, Steve H.L. Yim, Gideon Lee, Akshay Ashok, Steven R.H. Barrett, and I. A. Waitz, "Near-airport distribution of the environmental costs of aviation," *34,* pp. 102-108, 2014, doi: https://doi.org/10.1016/j.tranpol.2014.02.023.

[3] Gössling Stefan and H. Andreas, "The global scale, distribution and growth of aviation: Implications for climate change," vol. 65, 2020.

[4] M. Jatan, "Chandrayaan-3 Makes Historic Touchdown on the Moon," *Scientific American,* 2023.

[5] R. Dave, "How Airplanes Were Used in World War I," 2022.

[6] N. C. f. E. Health. "What Noises Cause Hearing Loss?" https://www.cdc.gov/nceh/hearing_loss/what_noises_cause_hearing_loss.html (accessed 8 November.

[7] U. C. A. Authority. "Aviation noise and healthThe effects of aviation noise." (accessed.

[8] Valdés Rosa Arnaldo, Gómez Comendador Víctor Fernando, Sanz Alvaro Rodriguez, and C. J. Perez, "Aviation 4.0: More Safety through Automation and Digitization," 2018.

[9] T. Gallagher, "Noise pollution: How are airports and airlines addressing the issue?," 2021.

[10] M. Sayers and S. Wong, "Aircraft noise linked to higher rates of heart disease and stroke near Heathrow," no. 8 October, 2013, doi: 10.1136/bmj.f5432.

[11] C. Baudin, "The role of aircraft noise annoyance and noise sensitivity in the association between aircraft noise levels and medication use: results of a pooled-analysis from seven European countries," ed, 2021.

[12] M. L. Clémence Baudin, Wolfgang Babisch, Ennio Cadum, Patricia Champelovier, Konstantina Dimakopoulou, Danny Houthuijs, Jacques Lambert, Bernard Laumon, Göran Pershagen, Stephen Stansfeld, Venetia Velonaki, Anna L. Hansell & Anne-Sophie Evrard, "The role of aircraft noise annoyance and noise sensitivity in the association between aircraft noise levels and medication use: results of a pooled-analysis from seven European countries," vol. 21, 2021, doi: https://doi.org/10.1186/s12889-021-10280-3.

[13] C. A. Authority. "Noise - Overview." (accessed.

[14] O. Hahad, "Reduced Aircraft Noise Pollution During COVID-19 Lockdown Is Beneficial to Public Cardiovascular Health: a Perspective on the Reduction of Transportation-Associated Pollution," ed, 2021.

[15] Hahad Omar , Daiber Andreas , and M. Thomas, "Reduced Aircraft Noise Pollution During COVID-19 Lockdown Is Beneficial to Public Cardiovascular Health: a

Perspective on the Reduction of Transportation-Associated Pollution," vol. 79, no. 2, 2021, doi: https://doi.org/10.1161/HYPERTENSIONAHA.121.18607.

[16] C. C. Basner Mathias, Hansell Anna,Hileman James I., Janssen Sabine, Shepherd Kevin, Sparrow Victor, "Aviation Noise Impacts: State of the Science," pp. 41-50, 2017, doi: 10.4103/nah.NAH_104_16.

[17] Mr Ted ELLIFF, Mr Michele CREMASCHI, and Ms Violaine HUCK, *Impact of aircraft noise pollution on residents of large cities*. September 2020.

[18] S. L. Clainche, "Improving aircraft performance using machine learning: A review," ed, 2023.

[19] Soledad Le Clainche, Esteban Ferrer, Sam Gibson, Elisabeth Cross, Alessandro Parente, and R. Vinuesa, "Improving aircraft performance using machine learning: A review," vol. 138, 2023, doi: https://doi.org/10.1016/j.ast.2023.108354.

[20] Coline Ramee, Junghyun Kim, Junghyun KimMarie Deguignet, and D. N. Mavris, "Aircraft Flight Plan Optimization with Dynamic Weather and Airspace Constraints," 2020.

[21] e Silva Lucas Coelho and M. M. C. Rocha, "A data analytics framework for anomaly detection in flight operations," vol. 110, 2023, doi: https://doi.org/10.1016/j.jairtraman.2023.102409.

[22] I. Anaconda. "Anaconda Navigator." (accessed.

[23] J. Team. "Jupyter Notebook." (accessed.

[24] S. W3. "Python." (accessed.

[25] W. School. "Pandas Introduction." (accessed.

[26] W. School. "NumPy Introduction." (accessed.

[27] P. S. Contribution. "Matplotlib." (accessed 15 September.

[28] W. School. "Seaborn." (accessed.

[29] P. S. Foundation. "warnings — Warning control." (accessed 24 September.

[30] P. F. *et al.*, "scikit-learn," vol. 12, pp. 2825-2830, 2011.

[31] D. K. "Implementing Gradient Boosting in Python." https://blog.paperspace.com/implementing-gradient-boosting-regression-python/ (accessed.

[32] S. Ronaghan, "The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark," ed, 2018.

[33] M. Schott, "Random Forest Algorithm for Machine Learning," ed, 2019.

[34] guest_blog. "Introduction to XGBoost Algorithm in Machine Learning." https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/ (accessed.

[35] D. Haowen, "The architecture of Gradient Boosting Decision Tree," ed, 2021.

[36] N. Alexey. and K. Alois., "Gradient Boosting Machines, A Tutorial," 2013, doi: 10.3389/fnbot.2013.00021.

[37] M. B. Fraj. "In Depth: Parameter tuning for Gradient Boosting." https://medium.com/all-things-ai/in-depth-parameter-tuning-for-gradient-boosting-3363992e9bae (accessed.

[38] T. Parr. and J. Howard, "The MSE function gradient," 2018.

[39] Ç. Uslu, "What is Kaggle and what is it used for?," ed, 2022.

[40] R. Shaikh, "About Kaggle," ed, 2018.

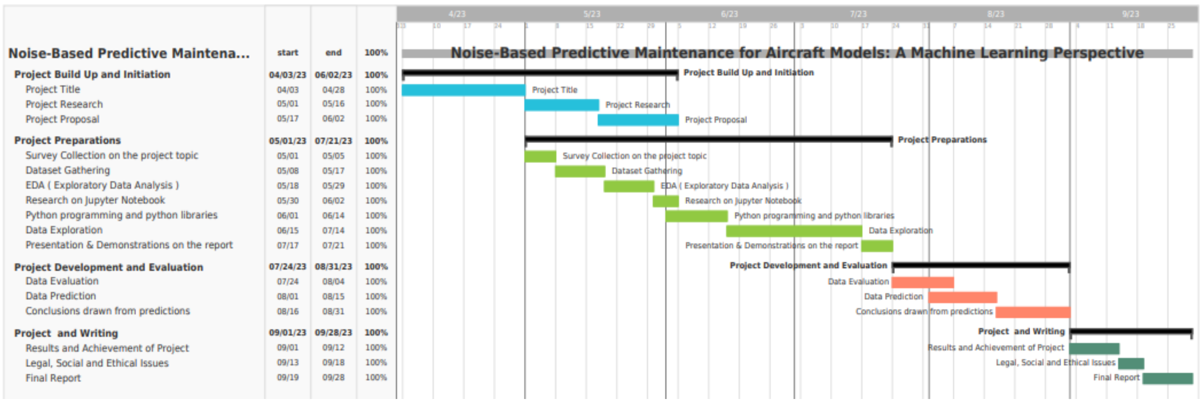
[41] T. Sasaki. *How noise pollution of aircraft has been improved?*,

[42] F. Alfie. "IS ARTIFICIAL INTELLIGENCE PRECEDING GOVERNANCE IN AVIATION?" (accessed.

[43] F. Marcin. "The Ethics of Artificial Intelligence in Autonomous Air Traffic Control." (accessed 14 May.

7 Appendix

7.1 Appendix A: Timeline of the Project (Gantt Chart Representation)



Noise-Based Predictive Maintena...	start	end	100%
Project Build Up and Initiation	**04/03/23**	**06/02/23**	**100%**
Project Title	04/03	04/28	100%
Project Research	05/01	05/16	100%
Project Proposal	05/17	06/02	100%
Project Preparations	**05/01/23**	**07/21/23**	**100%**
Survey Collection on the project topic	05/01	05/05	100%
Dataset Gathering	05/08	05/17	100%
EDA (Exploratory Data Analysis)	05/18	05/29	100%
Research on Jupyter Notebook	05/30	06/02	100%
Python programming and python libraries	06/01	06/14	100%
Data Exploration	06/15	07/14	100%
Presentation & Demonstrations on the report	07/17	07/21	100%
Project Development and Evaluation	**07/24/23**	**08/31/23**	**100%**
Data Evaluation	07/24	08/04	100%
Data Prediction	08/01	08/15	100%
Conclusions drawn from predictions	08/16	08/31	100%
Project and Writing	**09/01/23**	**09/28/23**	**100%**
Results and Achievement of Project	09/01	09/12	100%
Legal, Social and Ethical Issues	09/13	09/18	100%
Final Report	09/19	09/28	100%

Noise-Based Predictive Maintenance for Aircraft Models: A Machine Learning Perspective
4/23
5/23
6/23
7/23
8/23
9/23

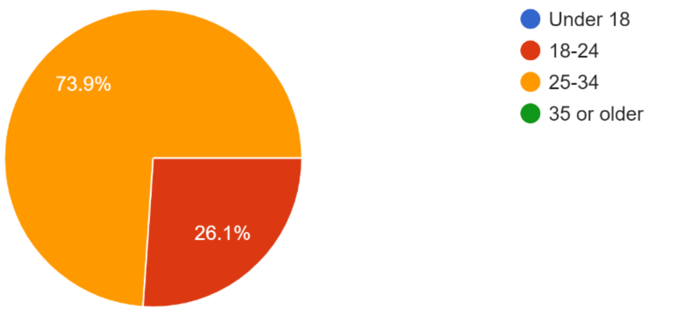
Figure 26: Timeline of the Project

7.2 Appendix B: Survey Results

7.2.1 Demographics

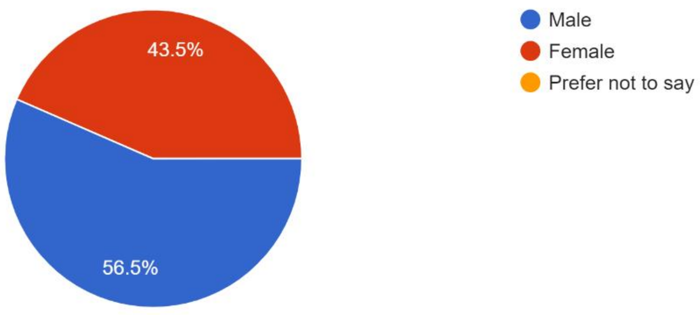
Age

23 responses



Gender

23 responses



Location (City/Country)

26 responses

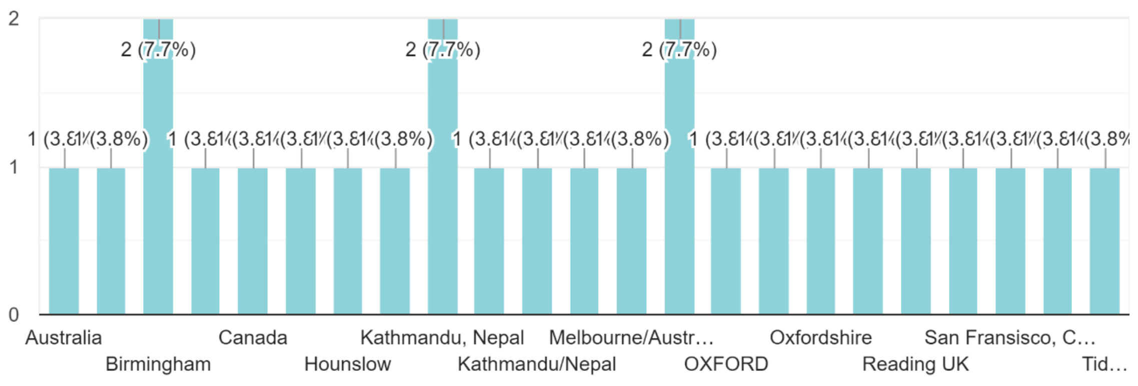
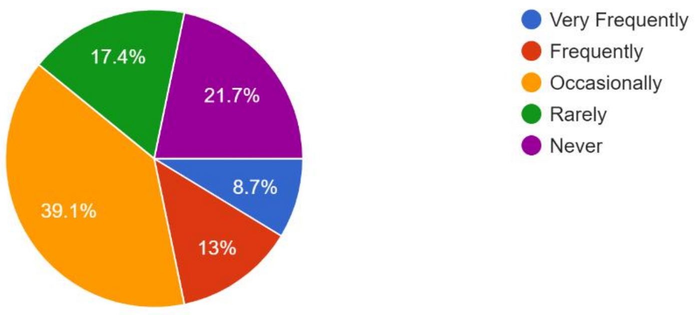


Figure 27: Survey Demographics

7.2.2 Aircraft Noise Pollution

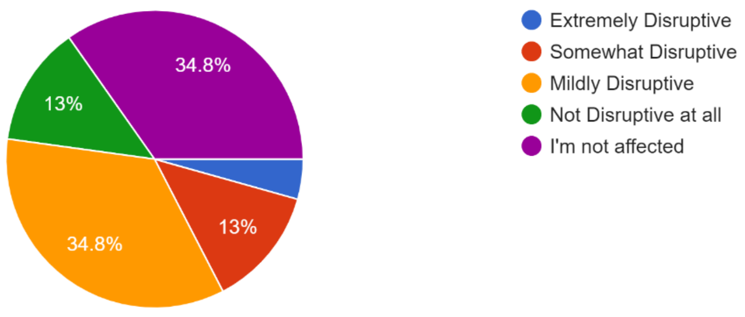
How often do you hear aircraft noise in your area?

23 responses



How does aircraft noise affect your daily life?

23 responses



Do you believe that aircraft noise pollution is a significant environmental concern?

23 responses

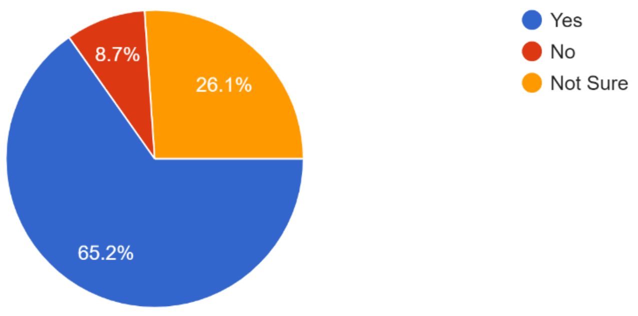
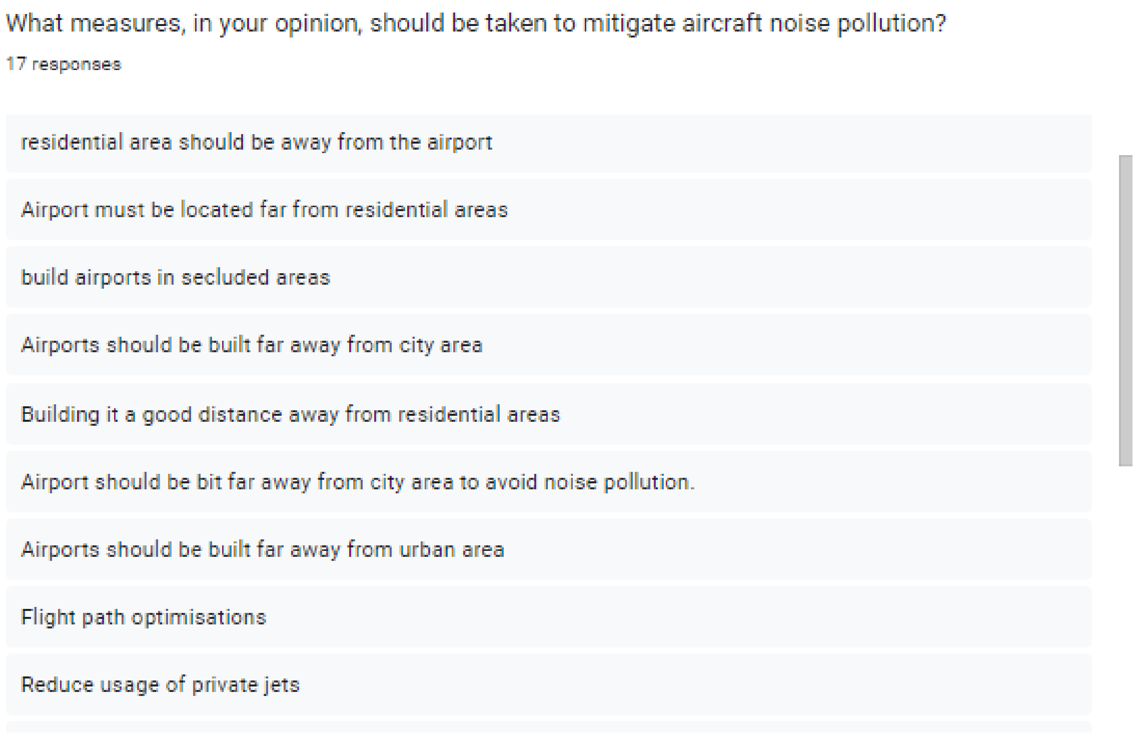


Figure 28: Survey aircraft noise pollution



What measures, in your opinion, should be taken to mitigate aircraft noise pollution?

17 responses

residential area should be away from the airport

Airport must be located far from residential areas

build airports in secluded areas

Airports should be built far away from city area

Building it a good distance away from residential areas

Airport should be bit far away from city area to avoid noise pollution.

Airports should be built far away from urban area

Flight path optimisations

Reduce usage of private jets

Figure 29: Survey of aircraft noise pollution mitigating opinion

7.2.3 Machine Learning in Aircraft Maintenance

Are you familiar with the use of machine learning algorithms in aircraft maintenance?

23 responses

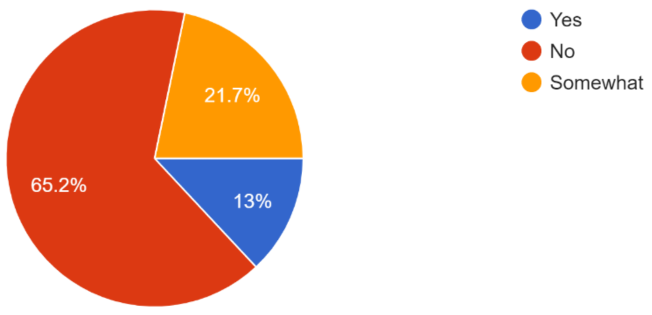
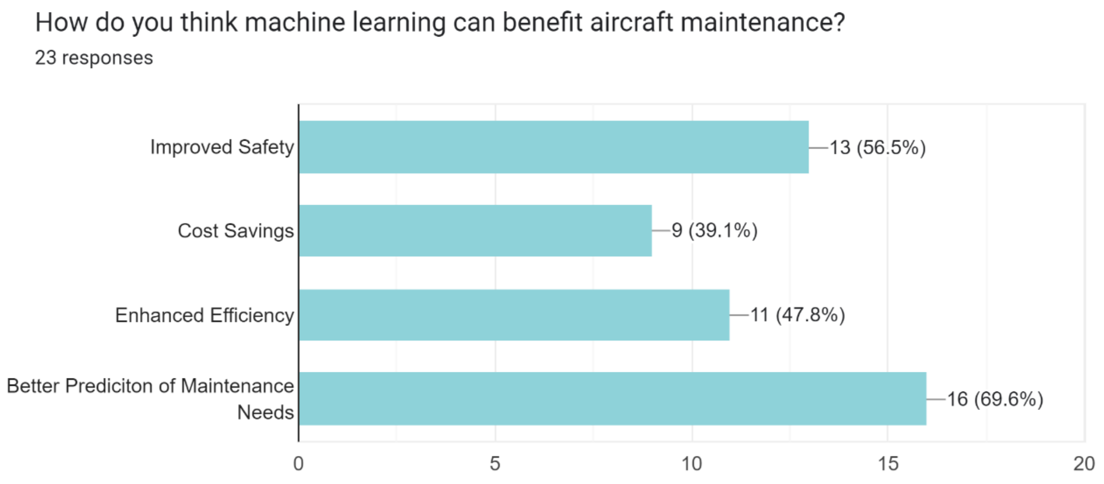
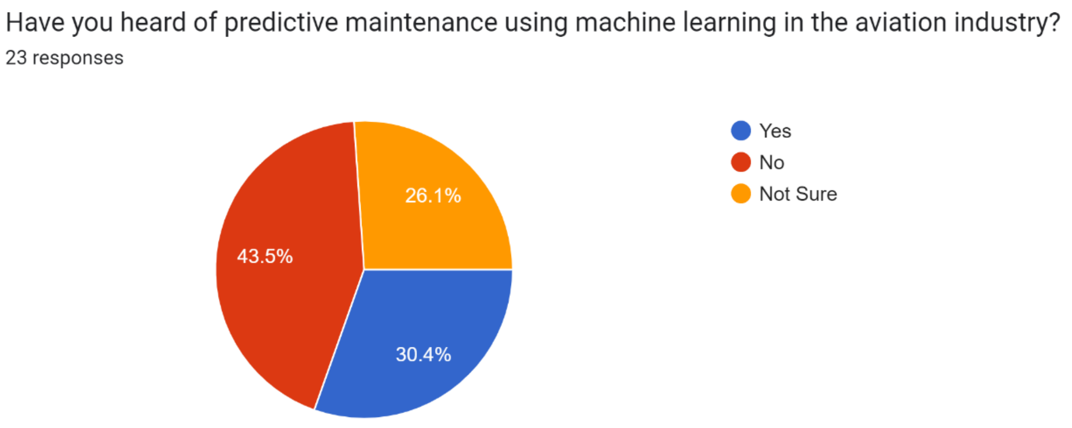


Figure 30: Survey of Machine Learning in Aircraft Maintenance 1





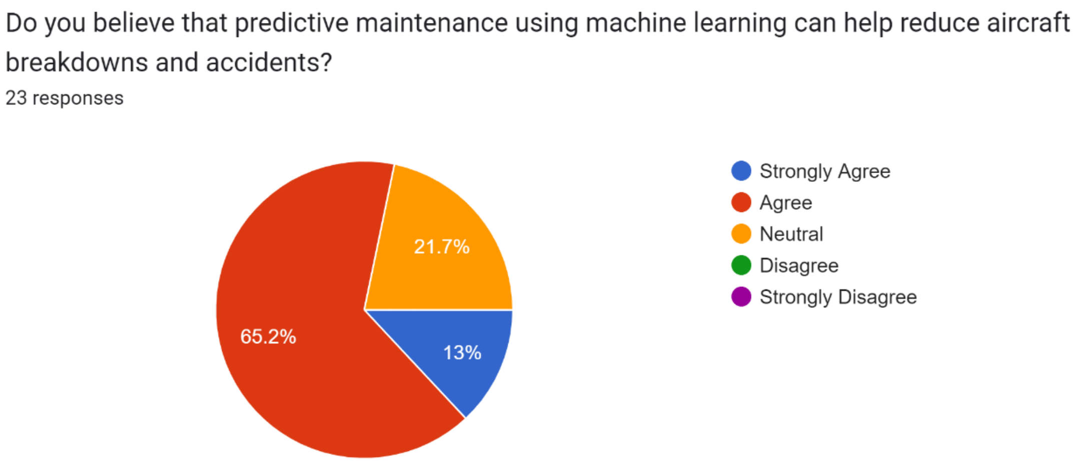
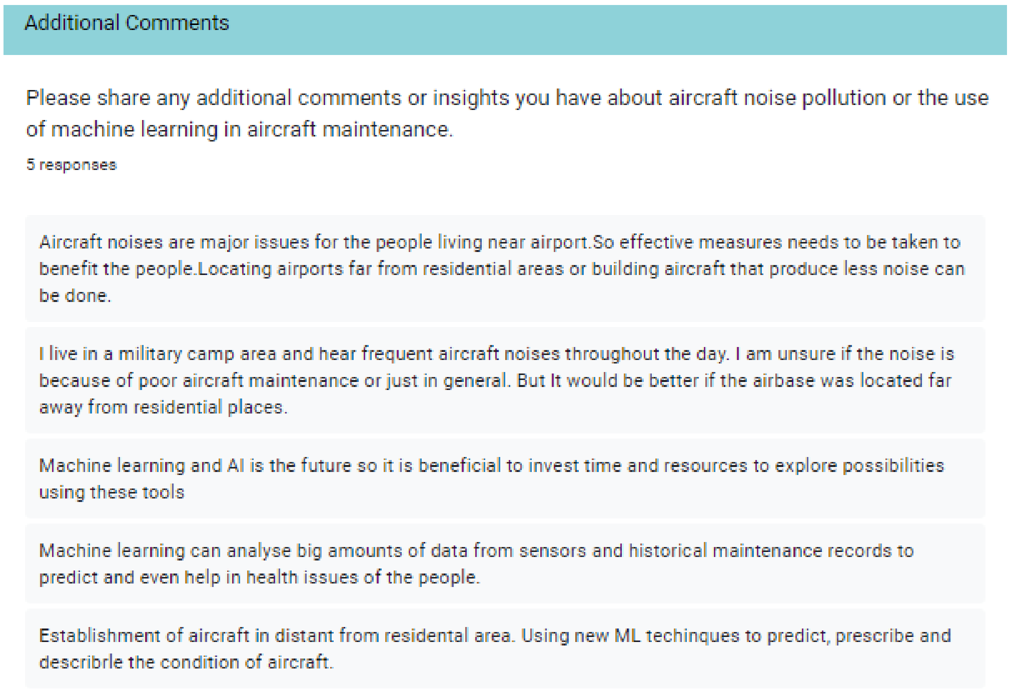


Figure 31: Survey of Machine Learning in Aircraft Maintenance 2


Additional Comments

Please share any additional comments or insights you have about aircraft noise pollution or the use of machine learning in aircraft maintenance.

5 responses

Aircraft noises are major issues for the people living near airport.So effective measures needs to be taken to benefit the people.Locating airports far from residential areas or building aircraft that produce less noise can be done.

I live in a military camp area and hear frequent aircraft noises throughout the day. I am unsure if the noise is because of poor aircraft maintenance or just in general. But It would be better if the airbase was located far away from residential places.

Machine learning and AI is the future so it is beneficial to invest time and resources to explore possibilities using these tools

Machine learning can analyse big amounts of data from sensors and historical maintenance records to predict and even help in health issues of the people.

Establishment of aircraft in distant from residental area. Using new ML techinques to predict, prescribe and describrle the condition of aircraft.

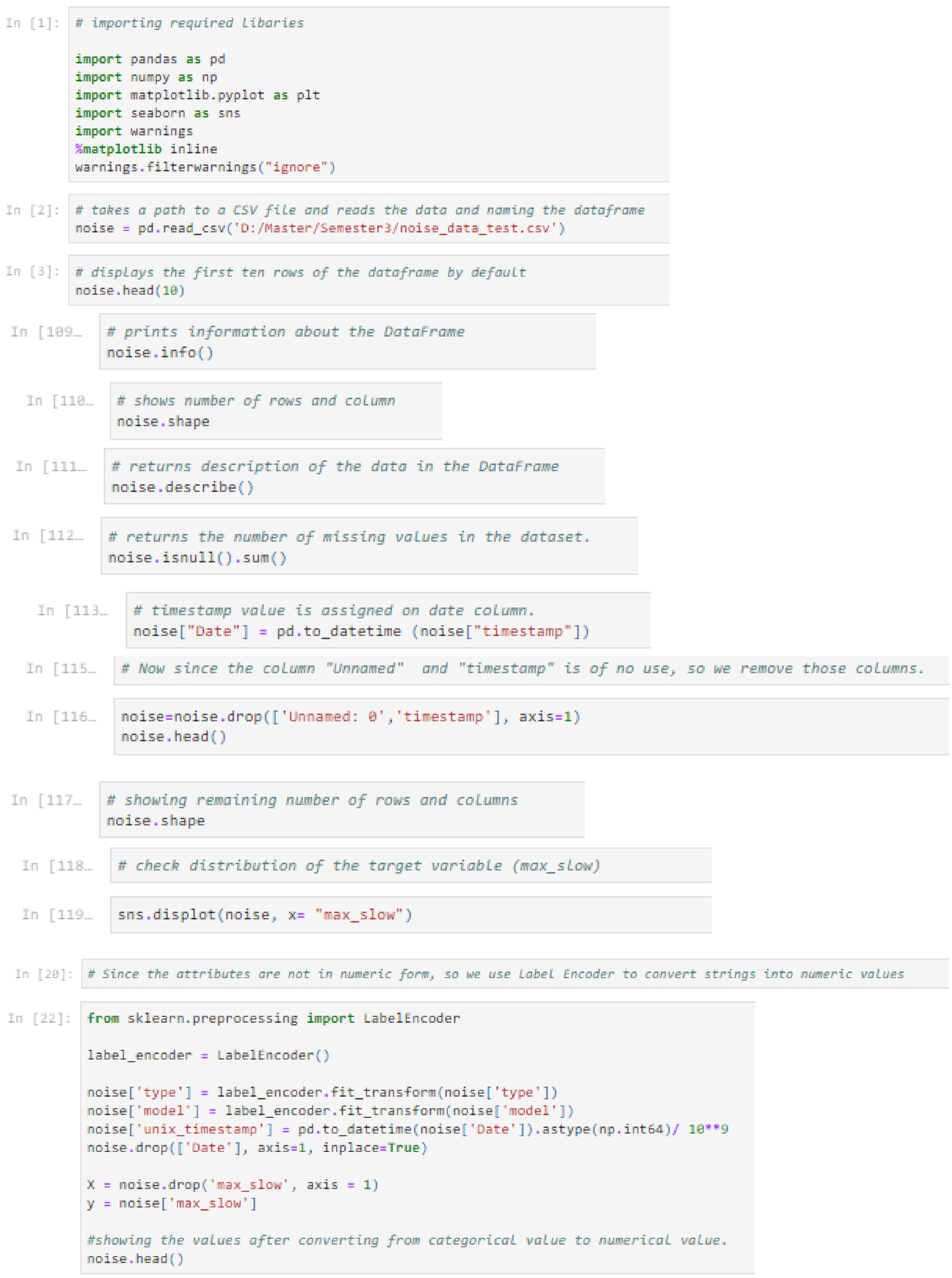
Figure 32: Survey of Machine Learning in Aircraft Maintenance additional comments and insights

7.3 Appendix C: Code

7.3.1 Appendix C1: Code Link

Google drive code link : https://drive.google.com/drive/folders/1GIxztneZFly-tRw2MaFvNxDW_NTcA3sA?usp=drive_link

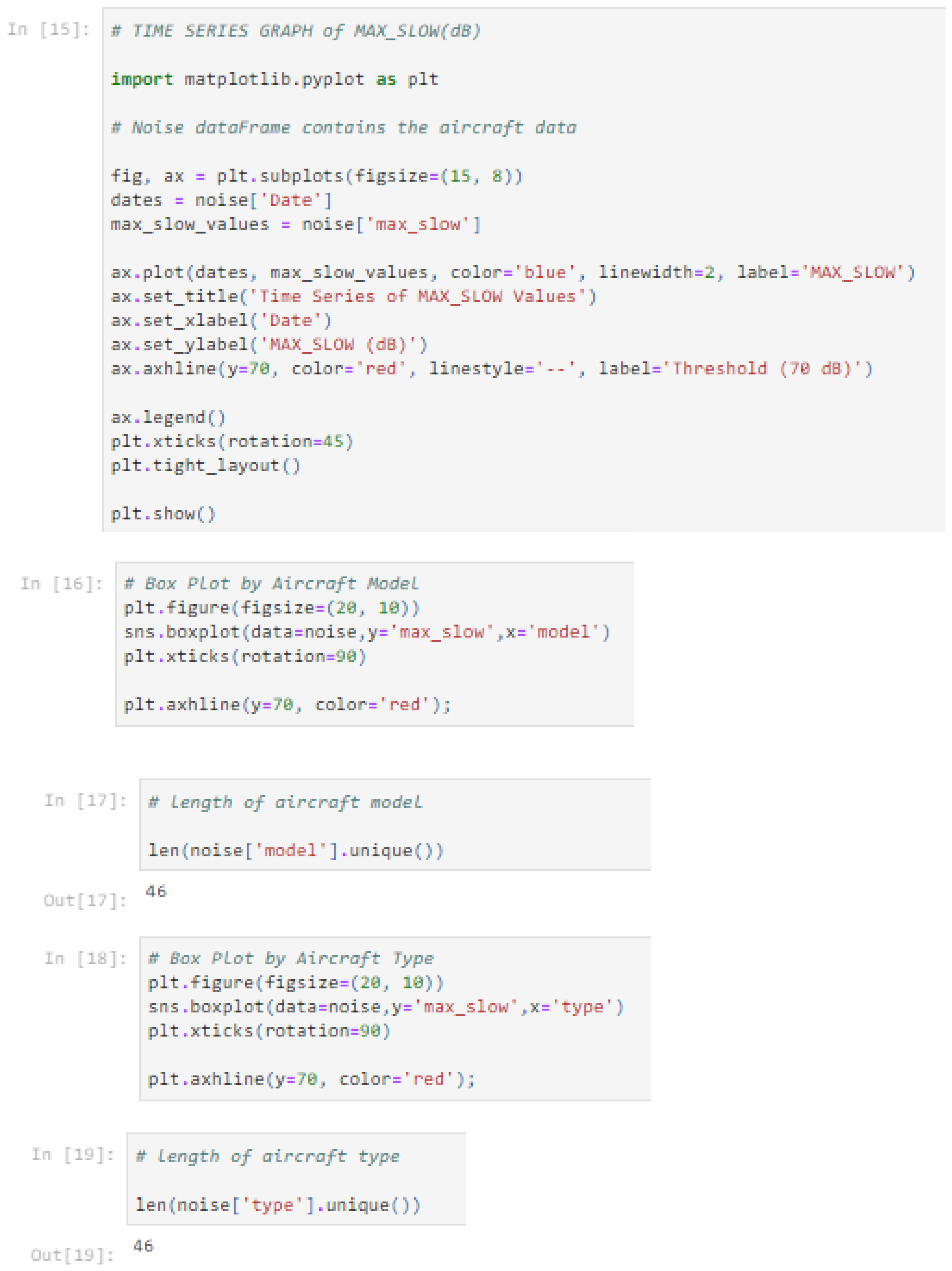
7.3.2 Appendix C2: Data Pre-Processing and Exploration



```
In [1]:  # importing required libaries

         import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
         import warnings
         %matplotlib inline
         warnings.filterwarnings("ignore")

In [2]:  # takes a path to a CSV file and reads the data and naming the dataframe
         noise = pd.read_csv('D:/Master/Semester3/noise_data_test.csv')

In [3]:  # displays the first ten rows of the dataframe by default
         noise.head(10)

In [109…  # prints information about the DataFrame
          noise.info()

In [110…  # shows number of rows and column
          noise.shape

In [111…  # returns description of the data in the DataFrame
          noise.describe()

In [112…  # returns the number of missing values in the dataset.
          noise.isnull().sum()

In [113…  # timestamp value is assigned on date column.
          noise["Date"] = pd.to_datetime (noise["timestamp"])

In [115…  # Now since the column "Unnamed"  and "timestamp" is of no use, so we remove those columns.

In [116…  noise=noise.drop(['Unnamed: 0','timestamp'], axis=1)
          noise.head()

In [117…  # showing remaining number of rows and columns
          noise.shape

In [118…  # check distribution of the target variable (max_slow)

In [119…  sns.displot(noise, x= "max_slow")

In [20]:  # Since the attributes are not in numeric form, so we use Label Encoder to convert strings into numeric values

In [22]:  from sklearn.preprocessing import LabelEncoder

          label_encoder = LabelEncoder()

          noise['type'] = label_encoder.fit_transform(noise['type'])
          noise['model'] = label_encoder.fit_transform(noise['model'])
          noise['unix_timestamp'] = pd.to_datetime(noise['Date']).astype(np.int64)/ 10**9
          noise.drop(['Date'], axis=1, inplace=True)

          X = noise.drop('max_slow', axis = 1)
          y = noise['max_slow']

          #showing the values after converting from categorical value to numerical value.
          noise.head()
```

Figure 33: Code of data-preprocessing and data exploration

7.3.3 Appendix C3 : Data Visualisation without Using Algorithm



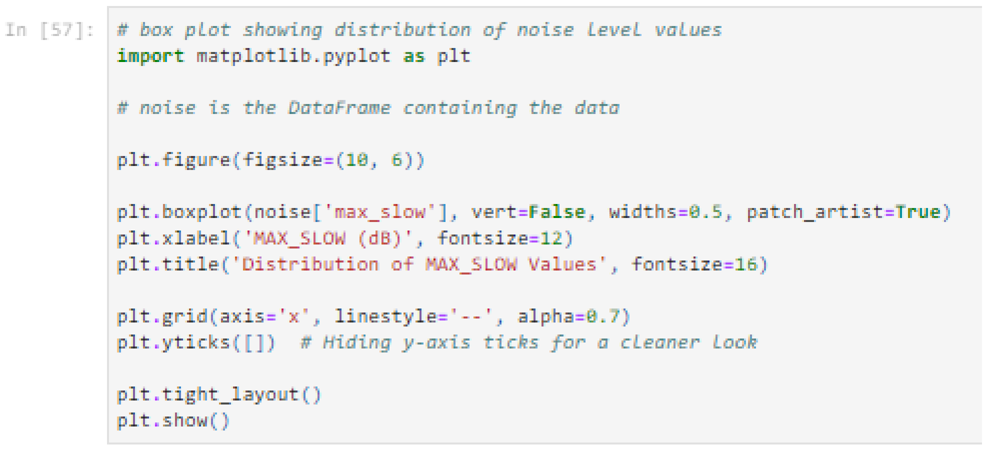
```
In [15]: # TIME SERIES GRAPH of MAX_SLOW(dB)

         import matplotlib.pyplot as plt

         # Noise dataFrame contains the aircraft data

         fig, ax = plt.subplots(figsize=(15, 8))
         dates = noise['Date']
         max_slow_values = noise['max_slow']

         ax.plot(dates, max_slow_values, color='blue', linewidth=2, label='MAX_SLOW')
         ax.set_title('Time Series of MAX_SLOW Values')
         ax.set_xlabel('Date')
         ax.set_ylabel('MAX_SLOW (dB)')
         ax.axhline(y=70, color='red', linestyle='--', label='Threshold (70 dB)')

         ax.legend()
         plt.xticks(rotation=45)
         plt.tight_layout()

         plt.show()
```

```
In [16]: # Box Plot by Aircraft Model
         plt.figure(figsize=(20, 10))
         sns.boxplot(data=noise,y='max_slow',x='model')
         plt.xticks(rotation=90)

         plt.axhline(y=70, color='red');
```

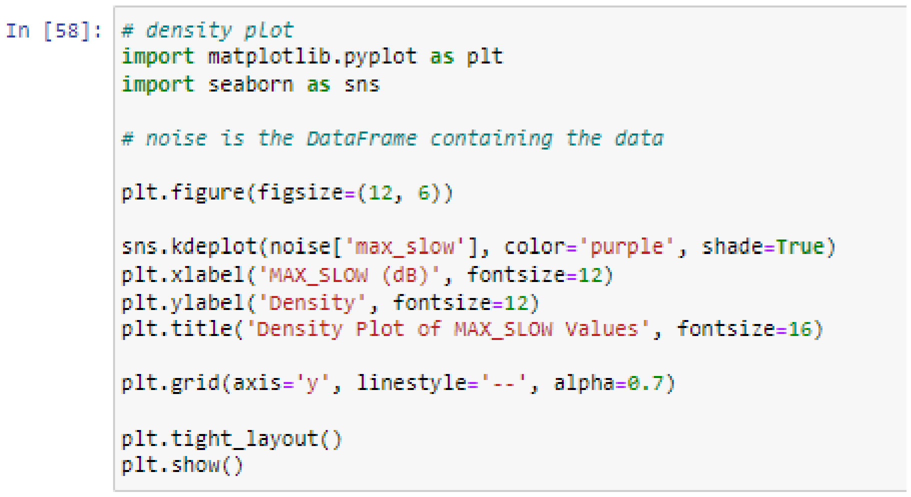
```
In [17]: # Length of aircraft model

         len(noise['model'].unique())

Out[17]: 46
```

```
In [18]: # Box Plot by Aircraft Type
         plt.figure(figsize=(20, 10))
         sns.boxplot(data=noise,y='max_slow',x='type')
         plt.xticks(rotation=90)

         plt.axhline(y=70, color='red');
```

```
In [19]: # Length of aircraft type

         len(noise['type'].unique())

Out[19]: 46
```



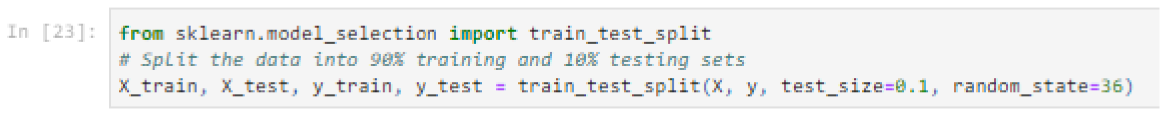
```
In [57]: # box plot showing distribution of noise level values
         import matplotlib.pyplot as plt

         # noise is the DataFrame containing the data

         plt.figure(figsize=(10, 6))

         plt.boxplot(noise['max_slow'], vert=False, widths=0.5, patch_artist=True)
         plt.xlabel('MAX_SLOW (dB)', fontsize=12)
         plt.title('Distribution of MAX_SLOW Values', fontsize=16)

         plt.grid(axis='x', linestyle='--', alpha=0.7)
         plt.yticks([])  # Hiding y-axis ticks for a cleaner look

         plt.tight_layout()
         plt.show()
```



```
In [58]: # density plot
         import matplotlib.pyplot as plt
         import seaborn as sns

         # noise is the DataFrame containing the data

         plt.figure(figsize=(12, 6))

         sns.kdeplot(noise['max_slow'], color='purple', shade=True)
         plt.xlabel('MAX_SLOW (dB)', fontsize=12)
         plt.ylabel('Density', fontsize=12)
         plt.title('Density Plot of MAX_SLOW Values', fontsize=16)

         plt.grid(axis='y', linestyle='--', alpha=0.7)

         plt.tight_layout()
         plt.show()
```

Figure 34: Code of different data visualisation without using algorithm

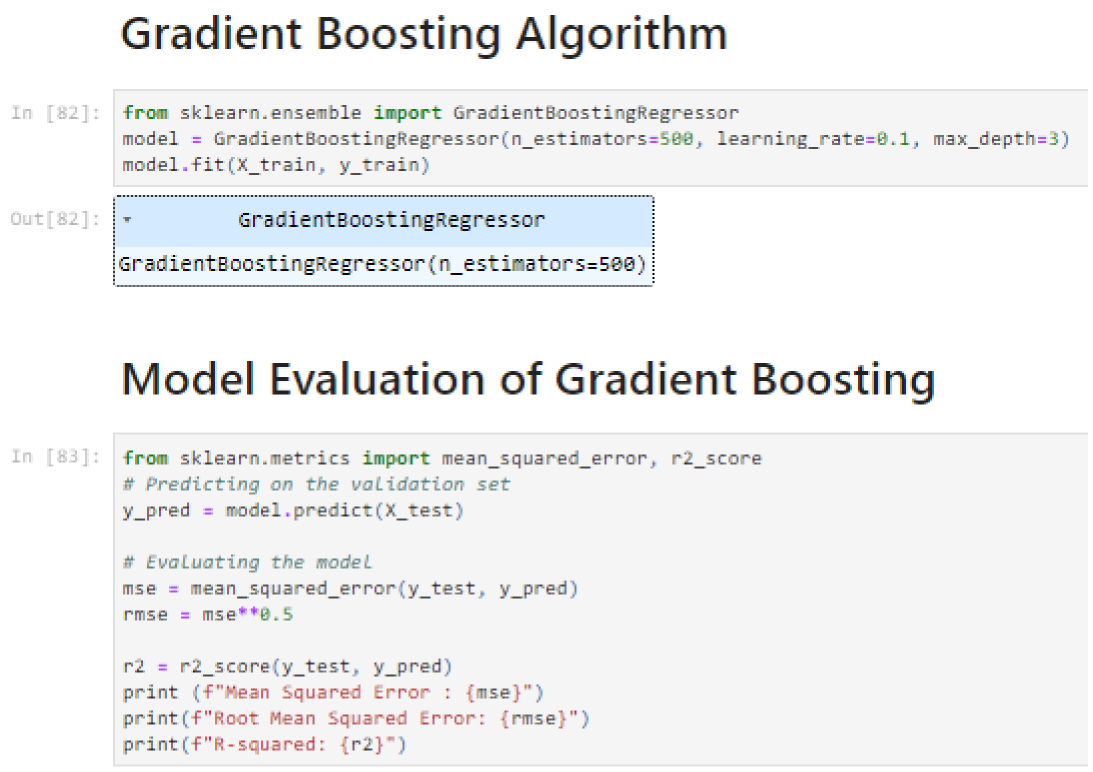
7.3.4 Appendix C4: Dividing Dataset into Training and Testing



```
In [23]: from sklearn.model_selection import train_test_split
         # Split the data into 90% training and 10% testing sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=36)
```

Figure 35: Code of dividing dataset into training and testing dataset

7.3.5 Appendix C5: Model Evaluation



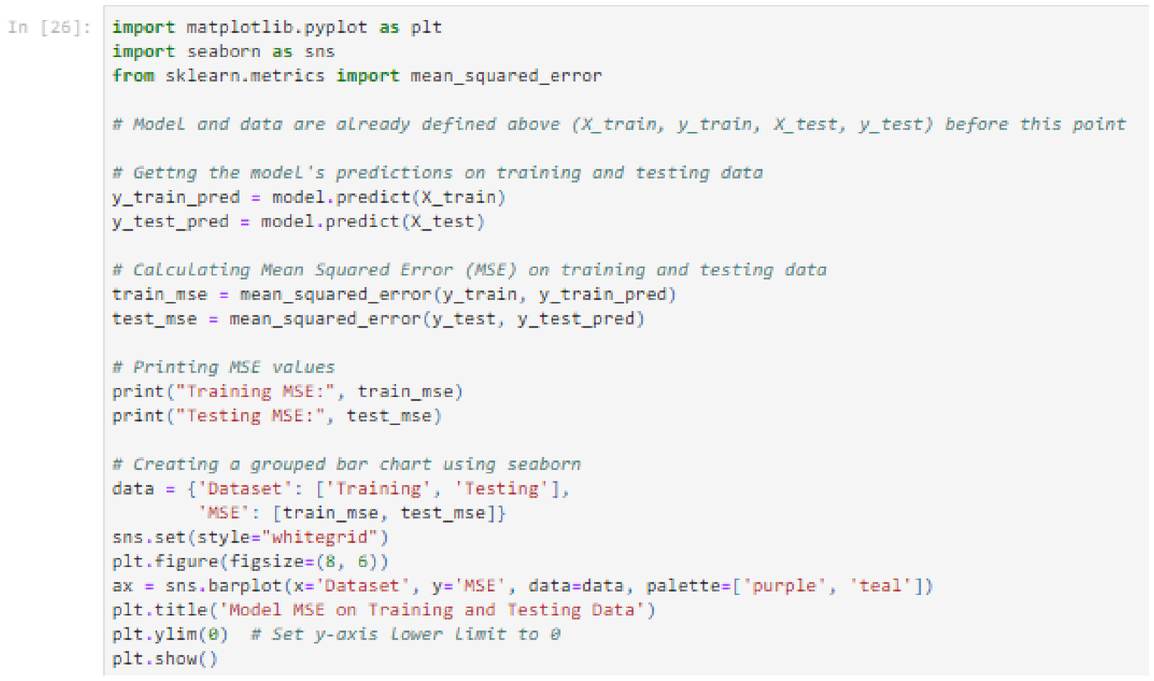
Gradient Boosting Algorithm

```
In [82]: from sklearn.ensemble import GradientBoostingRegressor
         model = GradientBoostingRegressor(n_estimators=500, learning_rate=0.1, max_depth=3)
         model.fit(X_train, y_train)

Out[82]:         GradientBoostingRegressor
         GradientBoostingRegressor(n_estimators=500)
```

Model Evaluation of Gradient Boosting

```
In [83]: from sklearn.metrics import mean_squared_error, r2_score
         # Predicting on the validation set
         y_pred = model.predict(X_test)

         # Evaluating the model
         mse = mean_squared_error(y_test, y_pred)
         rmse = mse**0.5

         r2 = r2_score(y_test, y_pred)
         print (f"Mean Squared Error : {mse}")
         print(f"Root Mean Squared Error: {rmse}")
         print(f"R-squared: {r2}")
```

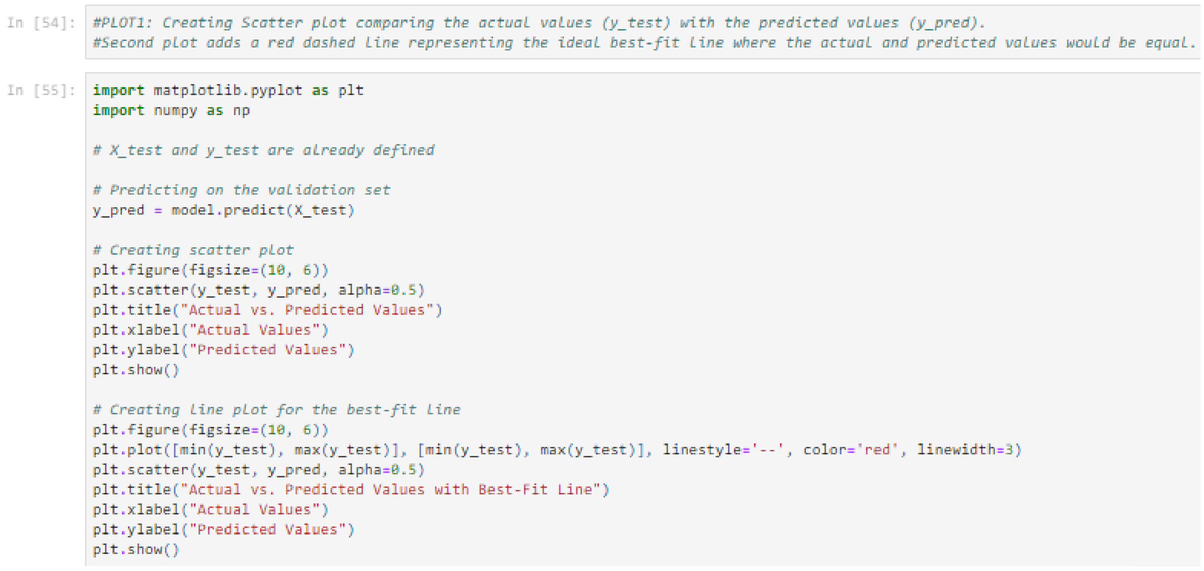
Figure 36: Code of model evaluation using gradient boosting model



```
In [26]: import matplotlib.pyplot as plt
         import seaborn as sns
         from sklearn.metrics import mean_squared_error

         # Model and data are already defined above (X_train, y_train, X_test, y_test) before this point

         # Gettng the model's predictions on training and testing data
         y_train_pred = model.predict(X_train)
         y_test_pred = model.predict(X_test)

         # Calculating Mean Squared Error (MSE) on training and testing data
         train_mse = mean_squared_error(y_train, y_train_pred)
         test_mse = mean_squared_error(y_test, y_test_pred)

         # Printing MSE values
         print("Training MSE:", train_mse)
         print("Testing MSE:", test_mse)

         # Creating a grouped bar chart using seaborn
         data = {'Dataset': ['Training', 'Testing'],
                 'MSE': [train_mse, test_mse]}
         sns.set(style="whitegrid")
         plt.figure(figsize=(8, 6))
         ax = sns.barplot(x='Dataset', y='MSE', data=data, palette=['purple', 'teal'])
         plt.title('Model MSE on Training and Testing Data')
         plt.ylim(0)  # Set y-axis lower limit to 0
         plt.show()
```

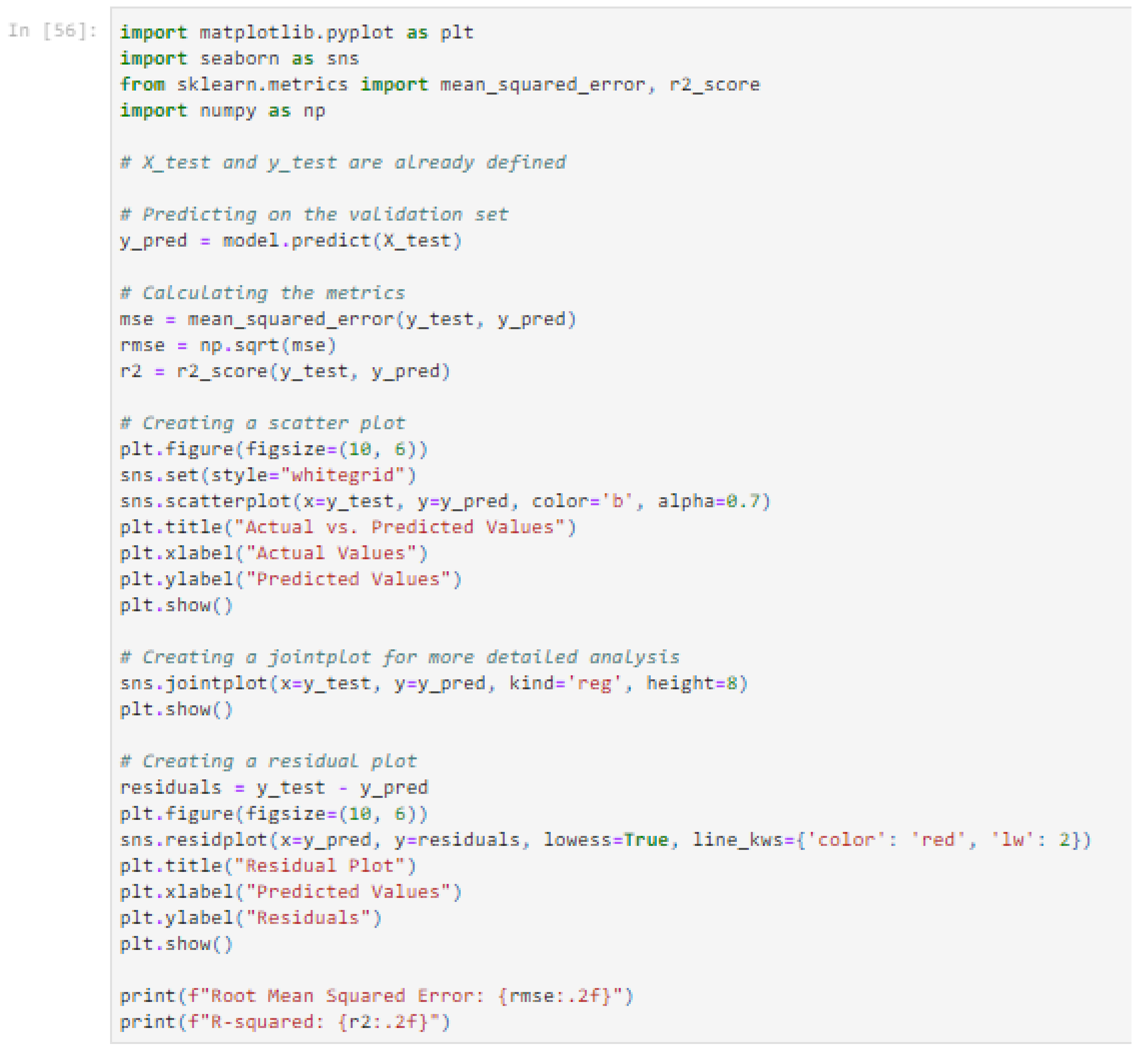
Figure 37: Code of model evaluation of training and testing dataset

7.3.6 Appendix C6: Comparing the Actual Values with the Predicted Values



```
In [54]: #PLOT1: Creating Scatter plot comparing the actual values (y_test) with the predicted values (y_pred).
         #Second plot adds a red dashed line representing the ideal best-fit line where the actual and predicted values would be equal.
```

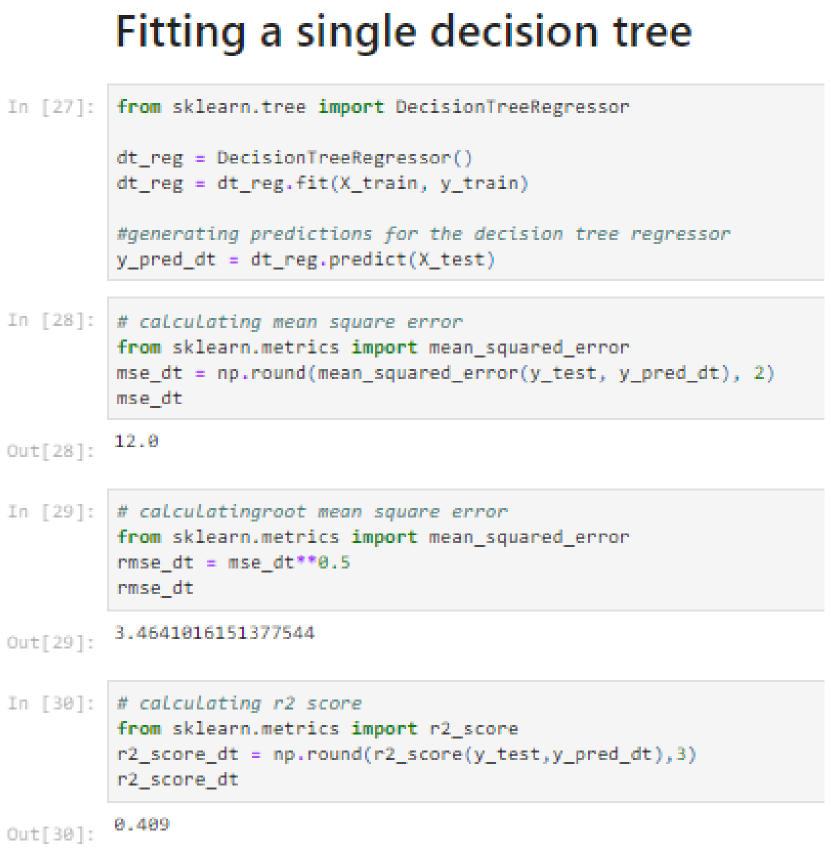
```
In [55]: import matplotlib.pyplot as plt
         import numpy as np

         # X_test and y_test are already defined

         # Predicting on the validation set
         y_pred = model.predict(X_test)

         # Creating scatter plot
         plt.figure(figsize=(10, 6))
         plt.scatter(y_test, y_pred, alpha=0.5)
         plt.title("Actual vs. Predicted Values")
         plt.xlabel("Actual Values")
         plt.ylabel("Predicted Values")
         plt.show()

         # Creating line plot for the best-fit line
         plt.figure(figsize=(10, 6))
         plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], linestyle='--', color='red', linewidth=3)
         plt.scatter(y_test, y_pred, alpha=0.5)
         plt.title("Actual vs. Predicted Values with Best-Fit Line")
         plt.xlabel("Actual Values")
         plt.ylabel("Predicted Values")
         plt.show()
```



```
In [56]: import matplotlib.pyplot as plt
         import seaborn as sns
         from sklearn.metrics import mean_squared_error, r2_score
         import numpy as np

         # X_test and y_test are already defined

         # Predicting on the validation set
         y_pred = model.predict(X_test)

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

         # Creating a scatter plot
         plt.figure(figsize=(10, 6))
         sns.set(style="whitegrid")
         sns.scatterplot(x=y_test, y=y_pred, color='b', alpha=0.7)
         plt.title("Actual vs. Predicted Values")
         plt.xlabel("Actual Values")
         plt.ylabel("Predicted Values")
         plt.show()

         # Creating a jointplot for more detailed analysis
         sns.jointplot(x=y_test, y=y_pred, kind='reg', height=8)
         plt.show()

         # Creating a residual plot
         residuals = y_test - y_pred
         plt.figure(figsize=(10, 6))
         sns.residplot(x=y_pred, y=residuals, lowess=True, line_kws={'color': 'red', 'lw': 2})
         plt.title("Residual Plot")
         plt.xlabel("Predicted Values")
         plt.ylabel("Residuals")
         plt.show()

         print(f"Root Mean Squared Error: {rmse:.2f}")
         print(f"R-squared: {r2:.2f}")
```

Figure 38: Comparing the actual value with the predicted value

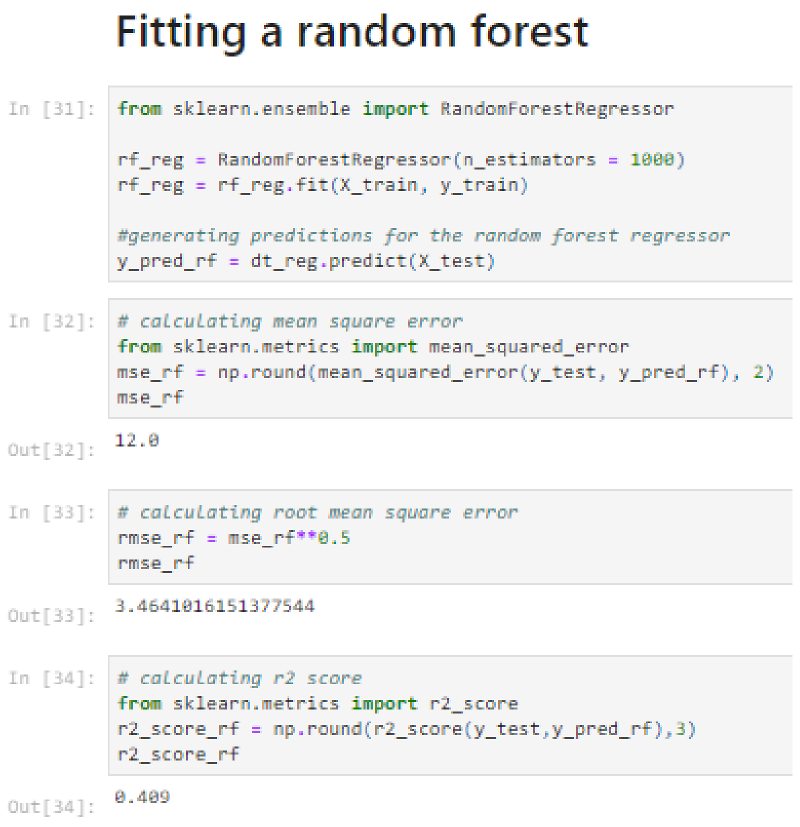
7.3.7 Appendix C7: Comparison of Different Algorithm with Gradient Boosting



Fitting a single decision tree

```
In [27]: from sklearn.tree import DecisionTreeRegressor

         dt_reg = DecisionTreeRegressor()
         dt_reg = dt_reg.fit(X_train, y_train)

         #generating predictions for the decision tree regressor
         y_pred_dt = dt_reg.predict(X_test)

In [28]: # calculating mean square error
         from sklearn.metrics import mean_squared_error
         mse_dt = np.round(mean_squared_error(y_test, y_pred_dt), 2)
         mse_dt

Out[28]: 12.0

In [29]: # calculatingroot mean square error
         from sklearn.metrics import mean_squared_error
         rmse_dt = mse_dt**0.5
         rmse_dt

Out[29]: 3.4641016151377544

In [30]: # calculating r2 score
         from sklearn.metrics import r2_score
         r2_score_dt = np.round(r2_score(y_test,y_pred_dt),3)
         r2_score_dt

Out[30]: 0.409
```

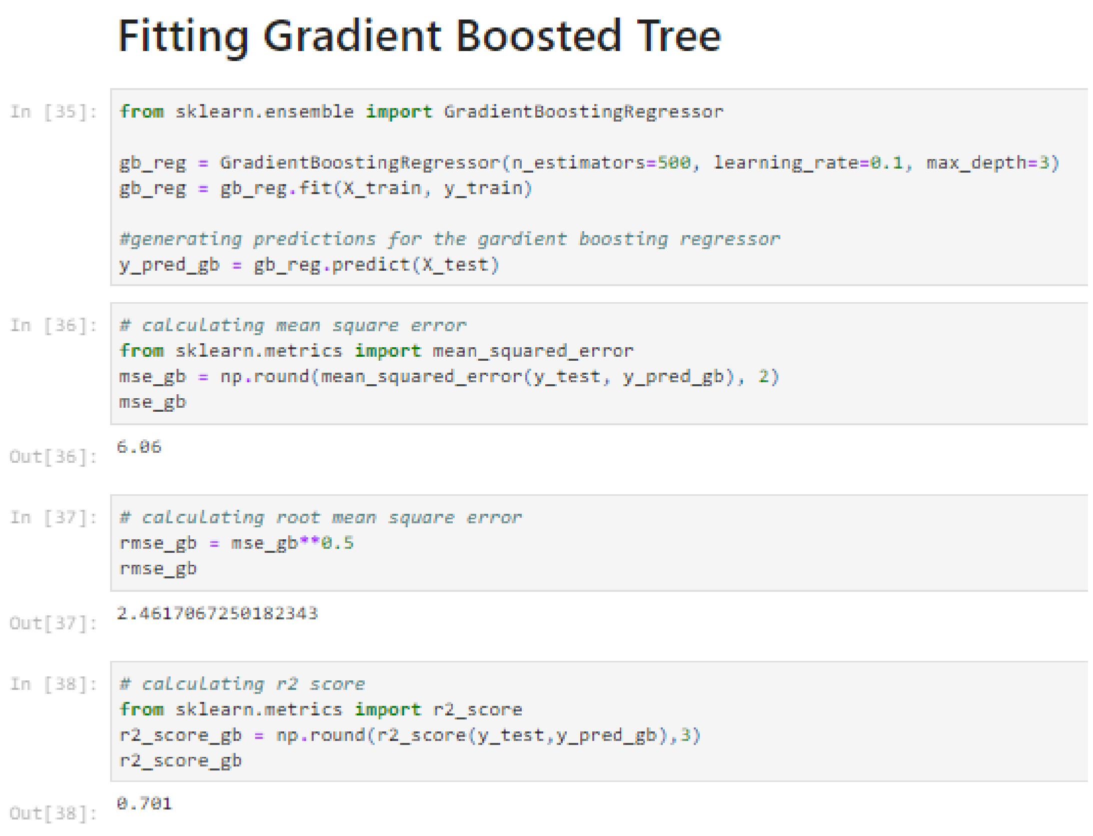
Figure 39: Code of calculating Single Decision Tree MSE, RMSE and R2



Fitting a random forest

```
In [31]: from sklearn.ensemble import RandomForestRegressor

         rf_reg = RandomForestRegressor(n_estimators = 1000)
         rf_reg = rf_reg.fit(X_train, y_train)

         #generating predictions for the random forest regressor
         y_pred_rf = dt_reg.predict(X_test)

In [32]: # calculating mean square error
         from sklearn.metrics import mean_squared_error
         mse_rf = np.round(mean_squared_error(y_test, y_pred_rf), 2)
         mse_rf

Out[32]: 12.0

In [33]: # calculating root mean square error
         rmse_rf = mse_rf**0.5
         rmse_rf

Out[33]: 3.4641016151377544

In [34]: # calculating r2 score
         from sklearn.metrics import r2_score
         r2_score_rf = np.round(r2_score(y_test,y_pred_rf),3)
         r2_score_rf

Out[34]: 0.409
```

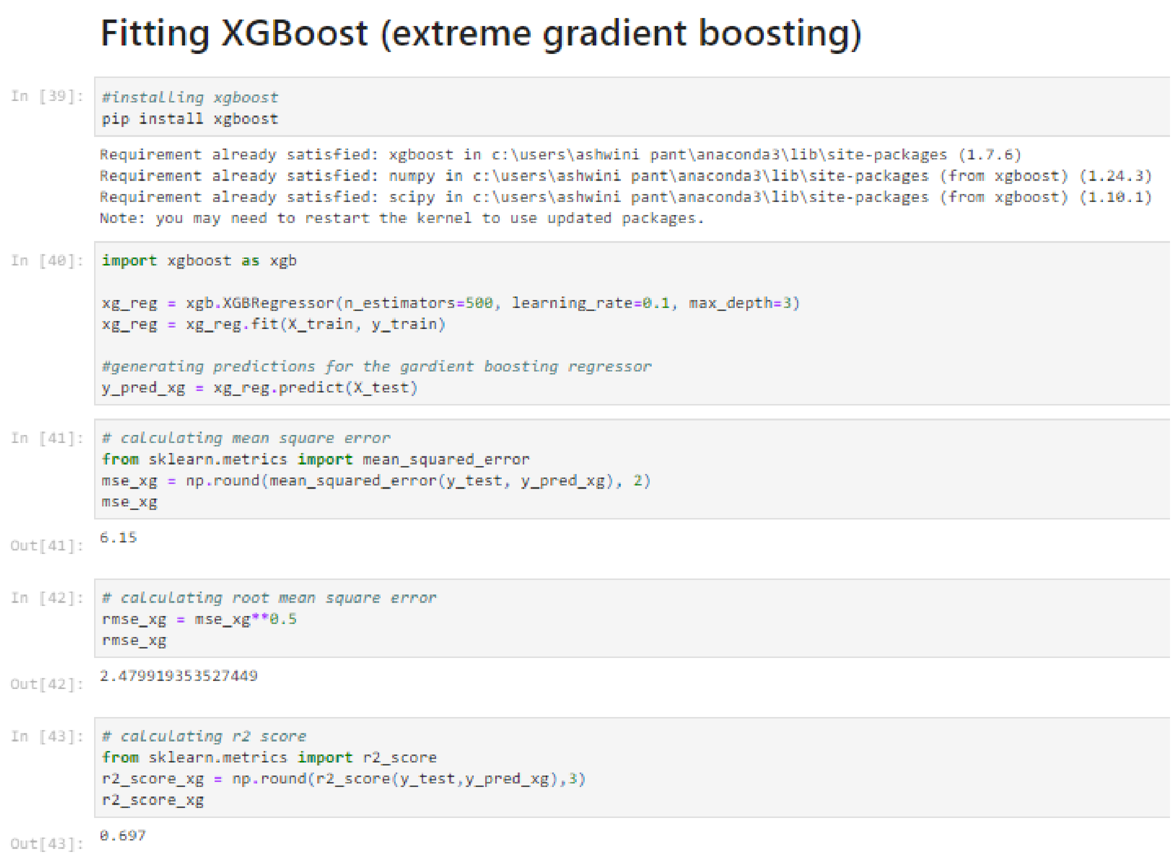
Figure 40: Code of calculating Radom Forest MSE, RMSE and R2



Fitting Gradient Boosted Tree

```
In [35]: from sklearn.ensemble import GradientBoostingRegressor

         gb_reg = GradientBoostingRegressor(n_estimators=500, learning_rate=0.1, max_depth=3)
         gb_reg = gb_reg.fit(X_train, y_train)

         #generating predictions for the gardient boosting regressor
         y_pred_gb = gb_reg.predict(X_test)

In [36]: # calculating mean square error
         from sklearn.metrics import mean_squared_error
         mse_gb = np.round(mean_squared_error(y_test, y_pred_gb), 2)
         mse_gb

Out[36]: 6.06

In [37]: # calculating root mean square error
         rmse_gb = mse_gb**0.5
         rmse_gb

Out[37]: 2.4617067250182343

In [38]: # calculating r2 score
         from sklearn.metrics import r2_score
         r2_score_gb = np.round(r2_score(y_test,y_pred_gb),3)
         r2_score_gb

Out[38]: 0.701
```

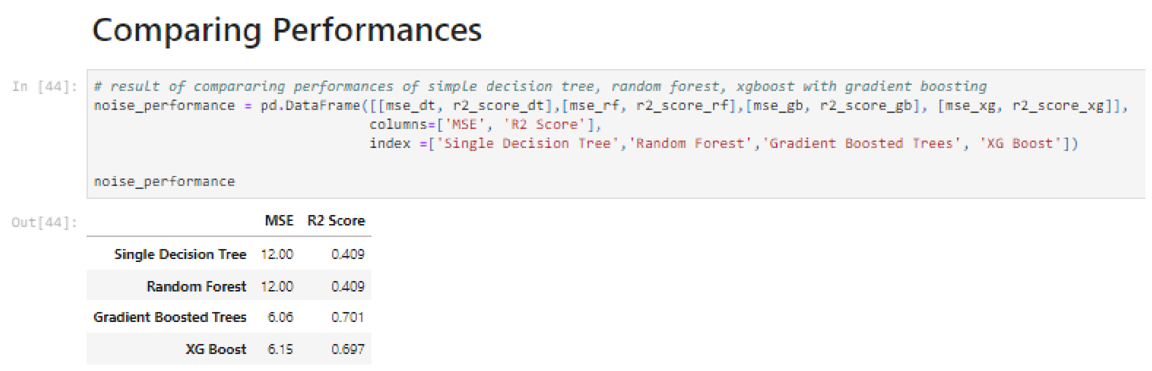
Figure 41: Code of calculating Gradient Boosting MSE, RMSE and R2



Fitting XGBoost (extreme gradient boosting)

```
In [39]: #installing xgboost
         pip install xgboost

         Requirement already satisfied: xgboost in c:\users\ashwini pant\anaconda3\lib\site-packages (1.7.6)
         Requirement already satisfied: numpy in c:\users\ashwini pant\anaconda3\lib\site-packages (from xgboost) (1.24.3)
         Requirement already satisfied: scipy in c:\users\ashwini pant\anaconda3\lib\site-packages (from xgboost) (1.10.1)
         Note: you may need to restart the kernel to use updated packages.

In [40]: import xgboost as xgb

         xg_reg = xgb.XGBRegressor(n_estimators=500, learning_rate=0.1, max_depth=3)
         xg_reg = xg_reg.fit(X_train, y_train)

         #generating predictions for the gardient boosting regressor
         y_pred_xg = xg_reg.predict(X_test)

In [41]: # calculating mean square error
         from sklearn.metrics import mean_squared_error
         mse_xg = np.round(mean_squared_error(y_test, y_pred_xg), 2)
         mse_xg

Out[41]: 6.15

In [42]: # calculating root mean square error
         rmse_xg = mse_xg**0.5
         rmse_xg

Out[42]: 2.479919353527449

In [43]: # calculating r2 score
         from sklearn.metrics import r2_score
         r2_score_xg = np.round(r2_score(y_test,y_pred_xg),3)
         r2_score_xg

Out[43]: 0.697
```

Figure 42: Code of calculating XGBoost MSE, RMSE and R2



Comparing Performances

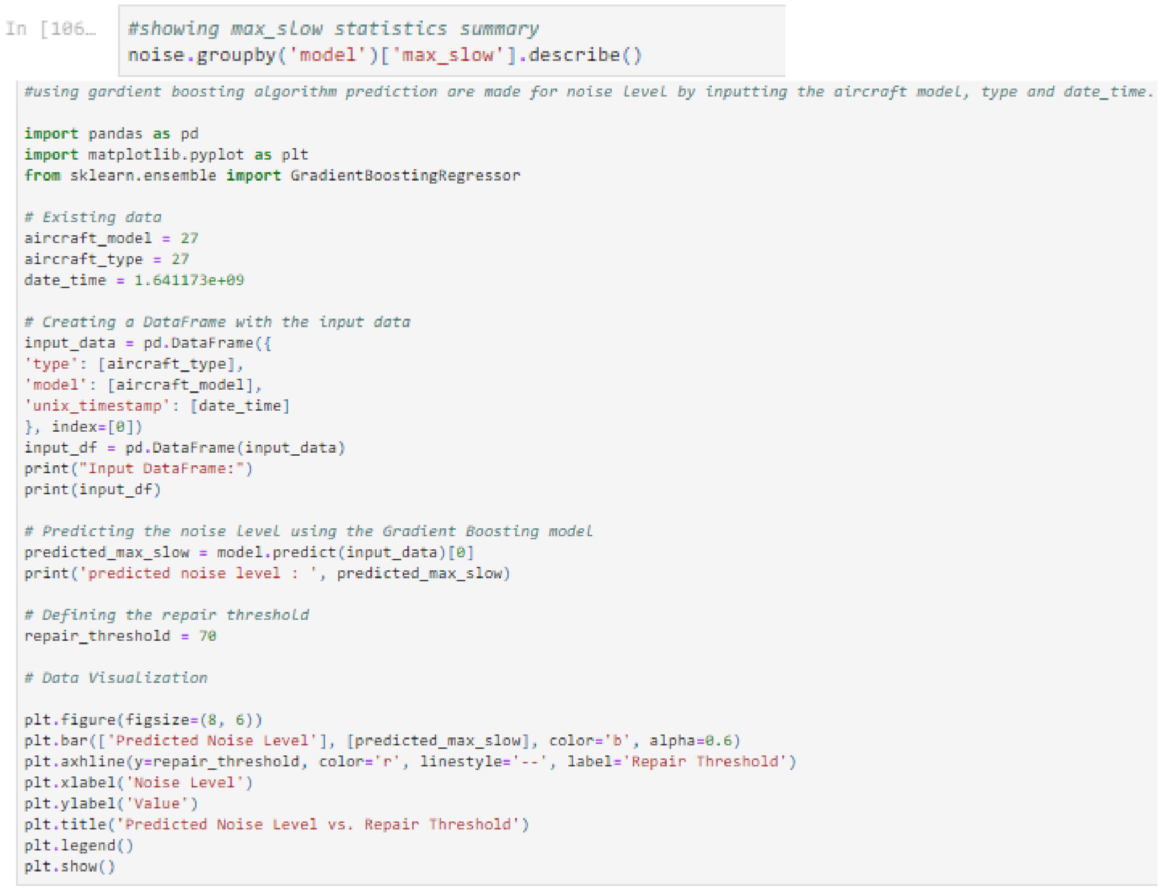
```
In [44]: # result of compararing performances of simple decision tree, random forest, xgboost with gradient boosting
         noise_performance = pd.DataFrame([[mse_dt, r2_score_dt],[mse_rf, r2_score_rf],[mse_gb, r2_score_gb], [mse_xg, r2_score_xg]],
                                          columns=['MSE', 'R2 Score'],
                                          index =['Single Decision Tree','Random Forest','Gradient Boosted Trees', 'XG Boost'])

         noise_performance
```

Out[44]:

	MSE	R2 Score
Single Decision Tree	12.00	0.409
Random Forest	12.00	0.409
Gradient Boosted Trees	6.06	0.701
XG Boost	6.15	0.697

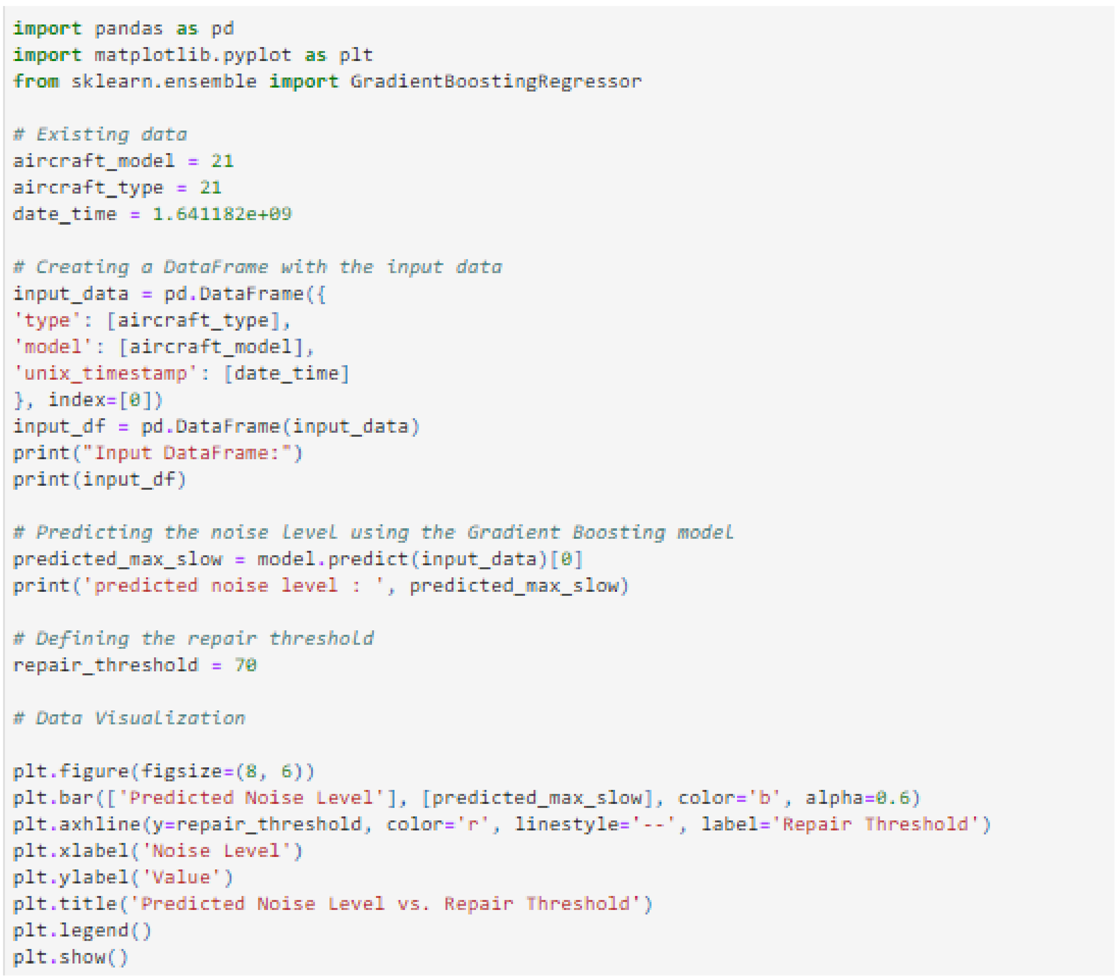
Figure 43: Code of comparing performance of Single Decision Tree, Random Forest, XGBoost with Gradient Boosting

7.3.8 Appendix C8: Analysis of Noise Level of Aircraft Models



```
In [106… #showing max_slow statistics summary
         noise.groupby('model')['max_slow'].describe()
```

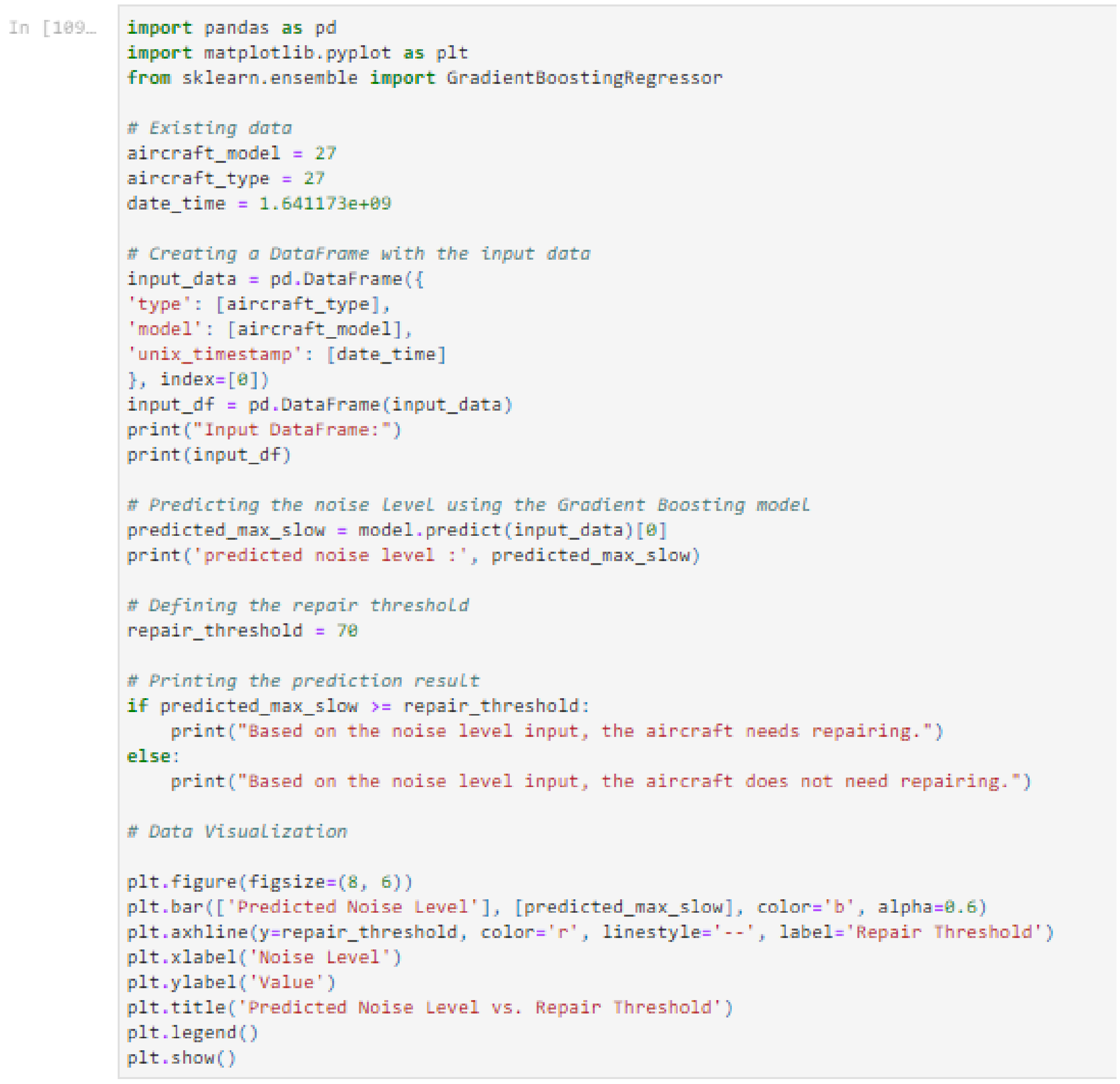
```
#using gardient boosting algorithm prediction are made for noise level by inputting the aircraft model, type and date_time.

import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import GradientBoostingRegressor

# Existing data
aircraft_model = 27
aircraft_type = 27
date_time = 1.641173e+09

# Creating a DataFrame with the input data
input_data = pd.DataFrame({
'type': [aircraft_type],
'model': [aircraft_model],
'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_data)[0]
print('predicted noise level : ', predicted_max_slow)

# Defining the repair threshold
repair_threshold = 70

# Data Visualization

plt.figure(figsize=(8, 6))
plt.bar(['Predicted Noise Level'], [predicted_max_slow], color='b', alpha=0.6)
plt.axhline(y=repair_threshold, color='r', linestyle='--', label='Repair Threshold')
plt.xlabel('Noise Level')
plt.ylabel('Value')
plt.title('Predicted Noise Level vs. Repair Threshold')
plt.legend()
plt.show()
```



```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import GradientBoostingRegressor

# Existing data
aircraft_model = 21
aircraft_type = 21
date_time = 1.641182e+09

# Creating a DataFrame with the input data
input_data = pd.DataFrame({
'type': [aircraft_type],
'model': [aircraft_model],
'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_data)[0]
print('predicted noise level : ', predicted_max_slow)

# Defining the repair threshold
repair_threshold = 70

# Data Visualization

plt.figure(figsize=(8, 6))
plt.bar(['Predicted Noise Level'], [predicted_max_slow], color='b', alpha=0.6)
plt.axhline(y=repair_threshold, color='r', linestyle='--', label='Repair Threshold')
plt.xlabel('Noise Level')
plt.ylabel('Value')
plt.title('Predicted Noise Level vs. Repair Threshold')
plt.legend()
plt.show()
```

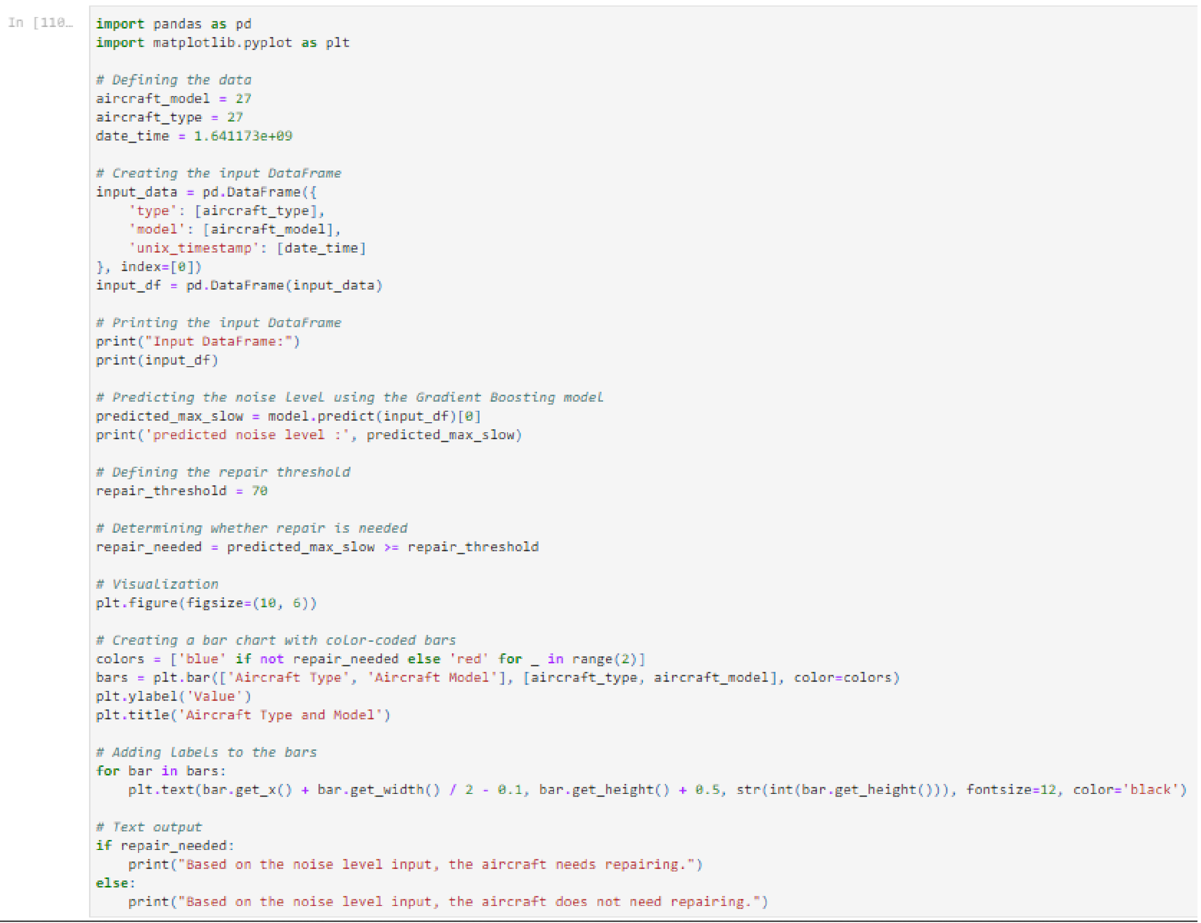
Figure 44: Code of analysis of noise level of Aircraft Model

7.3.9 Appendix C9: Predicting Maintenance of Aircraft Model Exceeding Threshold



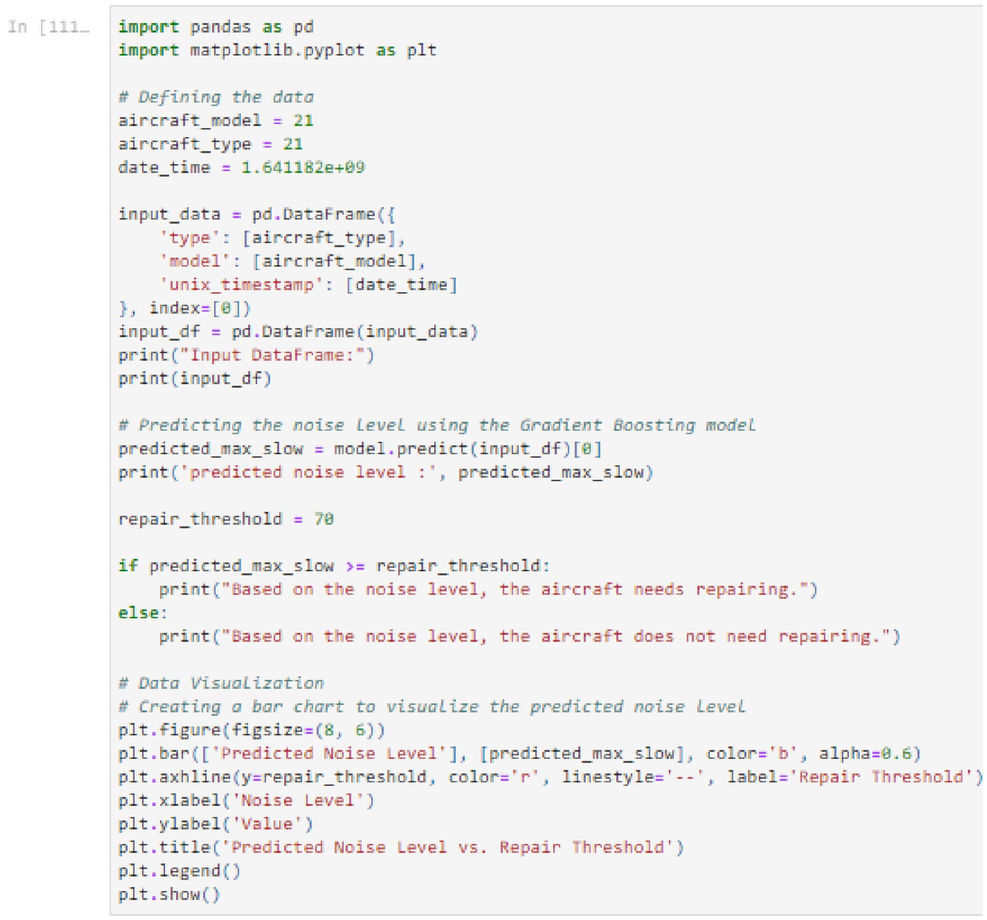
```
In [109…
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import GradientBoostingRegressor

# Existing data
aircraft_model = 27
aircraft_type = 27
date_time = 1.641173e+09

# Creating a DataFrame with the input data
input_data = pd.DataFrame({
'type': [aircraft_type],
'model': [aircraft_model],
'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_data)[0]
print('predicted noise level :', predicted_max_slow)

# Defining the repair threshold
repair_threshold = 70

# Printing the prediction result
if predicted_max_slow >= repair_threshold:
    print("Based on the noise level input, the aircraft needs repairing.")
else:
    print("Based on the noise level input, the aircraft does not need repairing.")

# Data Visualization

plt.figure(figsize=(8, 6))
plt.bar(['Predicted Noise Level'], [predicted_max_slow], color='b', alpha=0.6)
plt.axhline(y=repair_threshold, color='r', linestyle='--', label='Repair Threshold')
plt.xlabel('Noise Level')
plt.ylabel('Value')
plt.title('Predicted Noise Level vs. Repair Threshold')
plt.legend()
plt.show()
```

Figure 45: Code of prediction of maintenance of Aircraft Model 27



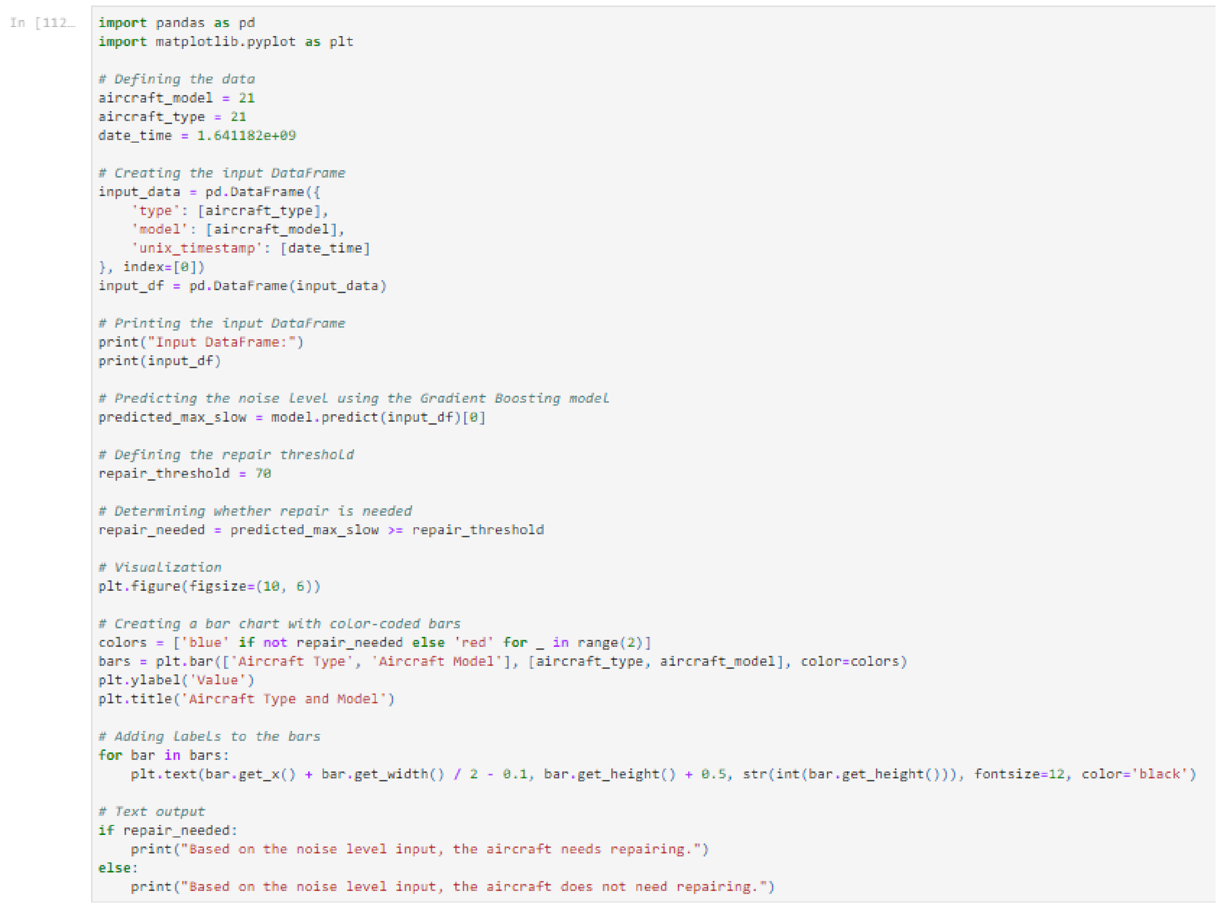
```
In [110…
import pandas as pd
import matplotlib.pyplot as plt

# Defining the data
aircraft_model = 27
aircraft_type = 27
date_time = 1.641173e+09

# Creating the input DataFrame
input_data = pd.DataFrame({
    'type': [aircraft_type],
    'model': [aircraft_model],
    'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)

# Printing the input DataFrame
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_df)[0]
print('predicted noise level :', predicted_max_slow)

# Defining the repair threshold
repair_threshold = 70

# Determining whether repair is needed
repair_needed = predicted_max_slow >= repair_threshold

# Visualization
plt.figure(figsize=(10, 6))

# Creating a bar chart with color-coded bars
colors = ['blue' if not repair_needed else 'red' for _ in range(2)]
bars = plt.bar(['Aircraft Type', 'Aircraft Model'], [aircraft_type, aircraft_model], color=colors)
plt.ylabel('Value')
plt.title('Aircraft Type and Model')

# Adding labels to the bars
for bar in bars:
    plt.text(bar.get_x() + bar.get_width() / 2 - 0.1, bar.get_height() + 0.5, str(int(bar.get_height())), fontsize=12, color='black')

# Text output
if repair_needed:
    print("Based on the noise level input, the aircraft needs repairing.")
else:
    print("Based on the noise level input, the aircraft does not need repairing.")
```

Figure 46: Code of showing maintenance for Aircraft Model 27 and Aircraft Type 27



```
In [111…
import pandas as pd
import matplotlib.pyplot as plt

# Defining the data
aircraft_model = 21
aircraft_type = 21
date_time = 1.641182e+09

input_data = pd.DataFrame({
    'type': [aircraft_type],
    'model': [aircraft_model],
    'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_df)[0]
print('predicted noise level :', predicted_max_slow)

repair_threshold = 70

if predicted_max_slow >= repair_threshold:
    print("Based on the noise level, the aircraft needs repairing.")
else:
    print("Based on the noise level, the aircraft does not need repairing.")

# Data Visualization
# Creating a bar chart to visualize the predicted noise level
plt.figure(figsize=(8, 6))
plt.bar(['Predicted Noise Level'], [predicted_max_slow], color='b', alpha=0.6)
plt.axhline(y=repair_threshold, color='r', linestyle='--', label='Repair Threshold')
plt.xlabel('Noise Level')
plt.ylabel('Value')
plt.title('Predicted Noise Level vs. Repair Threshold')
plt.legend()
plt.show()
```

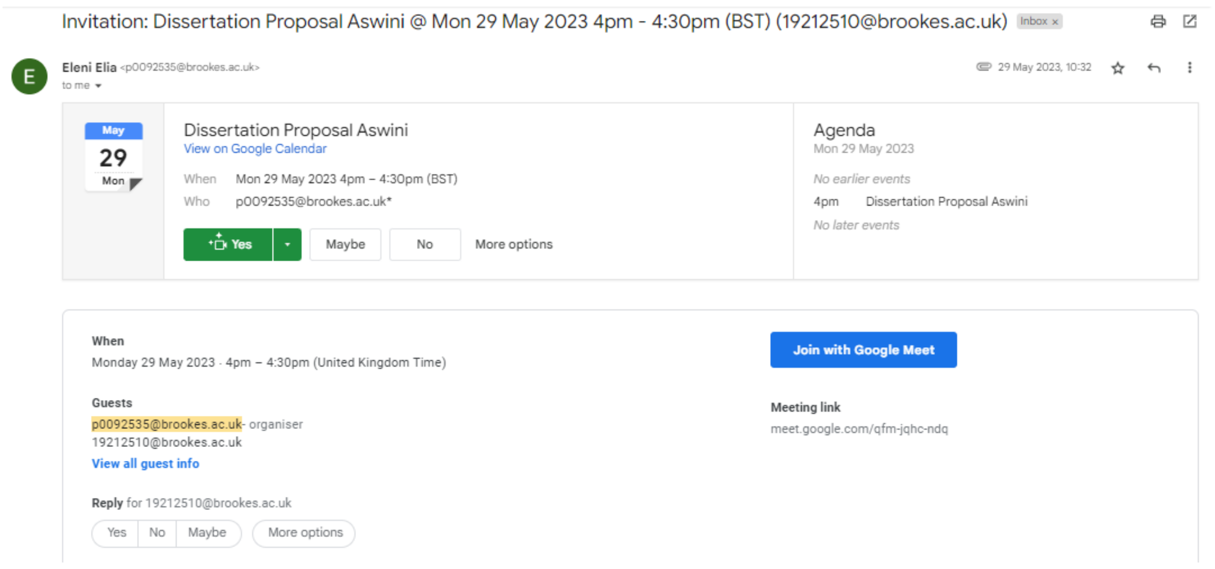
Figure 47: Code of prediction of maintenance of Aircraft Model 21



```
In [112…
import pandas as pd
import matplotlib.pyplot as plt

# Defining the data
aircraft_model = 21
aircraft_type = 21
date_time = 1.641182e+09

# Creating the input DataFrame
input_data = pd.DataFrame({
    'type': [aircraft_type],
    'model': [aircraft_model],
    'unix_timestamp': [date_time]
}, index=[0])
input_df = pd.DataFrame(input_data)

# Printing the input DataFrame
print("Input DataFrame:")
print(input_df)

# Predicting the noise level using the Gradient Boosting model
predicted_max_slow = model.predict(input_df)[0]

# Defining the repair threshold
repair_threshold = 70

# Determining whether repair is needed
repair_needed = predicted_max_slow >= repair_threshold

# Visualization
plt.figure(figsize=(10, 6))

# Creating a bar chart with color-coded bars
colors = ['blue' if not repair_needed else 'red' for _ in range(2)]
bars = plt.bar(['Aircraft Type', 'Aircraft Model'], [aircraft_type, aircraft_model], color=colors)
plt.ylabel('Value')
plt.title('Aircraft Type and Model')

# Adding labels to the bars
for bar in bars:
    plt.text(bar.get_x() + bar.get_width() / 2 - 0.1, bar.get_height() + 0.5, str(int(bar.get_height())), fontsize=12, color='black')

# Text output
if repair_needed:
    print("Based on the noise level input, the aircraft needs repairing.")
else:
    print("Based on the noise level input, the aircraft does not need repairing.")
```

Figure 48: Code of showing maintenance for Aircraft Model 21 and Aircraft Type 21

7.4 Appendix D: Evidence of Weekly Meeting with Supervisor



Invitation: Dissertation Proposal Aswini @ Mon 29 May 2023 4pm - 4:30pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

29 May 2023, 10:32

May 29 Mon

Dissertation Proposal Aswini
View on Google Calendar

When Mon 29 May 2023 4pm – 4:30pm (BST)
Who p0092535@brookes.ac.uk*

Yes | Maybe | No | More options

Agenda
Mon 29 May 2023
No earlier events
4pm Dissertation Proposal Aswini
No later events

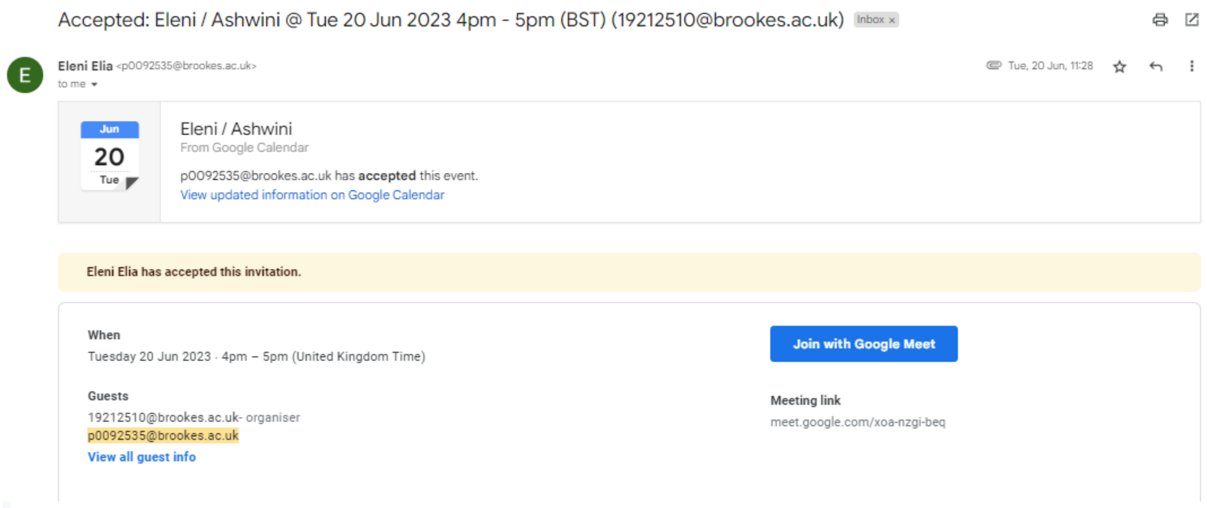
When
Monday 29 May 2023 · 4pm – 4:30pm (United Kingdom Time)

Guests
p0092535@brookes.ac.uk- organiser
19212510@brookes.ac.uk
View all guest info

Reply for 19212510@brookes.ac.uk
Yes | No | Maybe | More options

Join with Google Meet

Meeting link
meet.google.com/qfm-jqhc-ndq



Accepted: Eleni / Ashwini @ Tue 20 Jun 2023 4pm - 5pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Tue, 20 Jun, 11:28

Jun 20 Tue

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

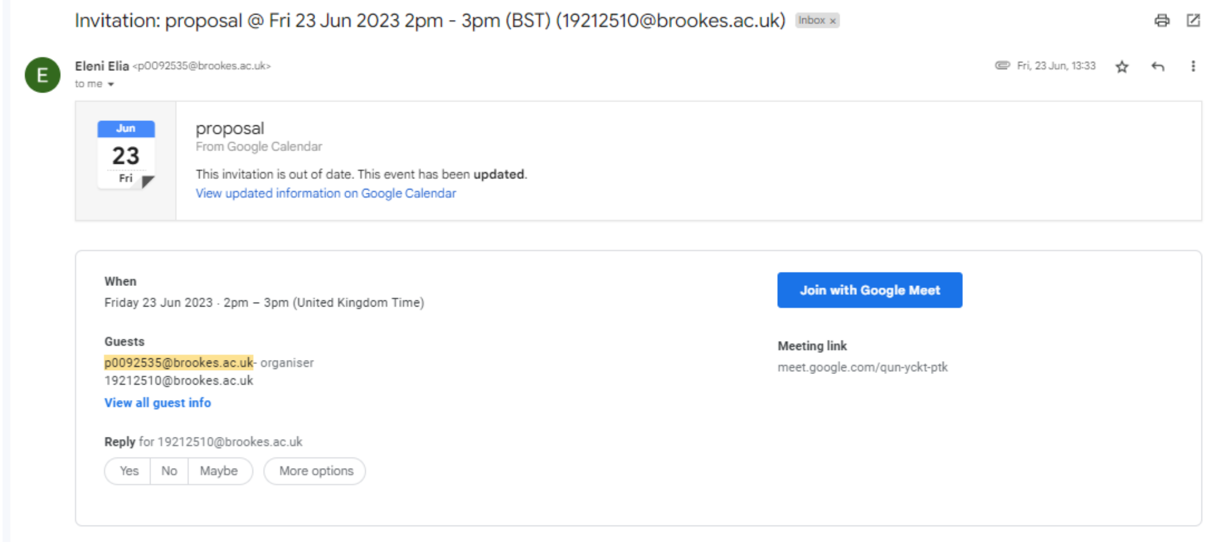
Eleni Elia has accepted this invitation.

When
Tuesday 20 Jun 2023 · 4pm – 5pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/xoa-nzgi-beq



Invitation: proposal @ Fri 23 Jun 2023 2pm - 3pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Fri, 23 Jun, 13:33

Jun 23 Fri

proposal
From Google Calendar
This invitation is out of date. This event has been **updated**.
View updated information on Google Calendar

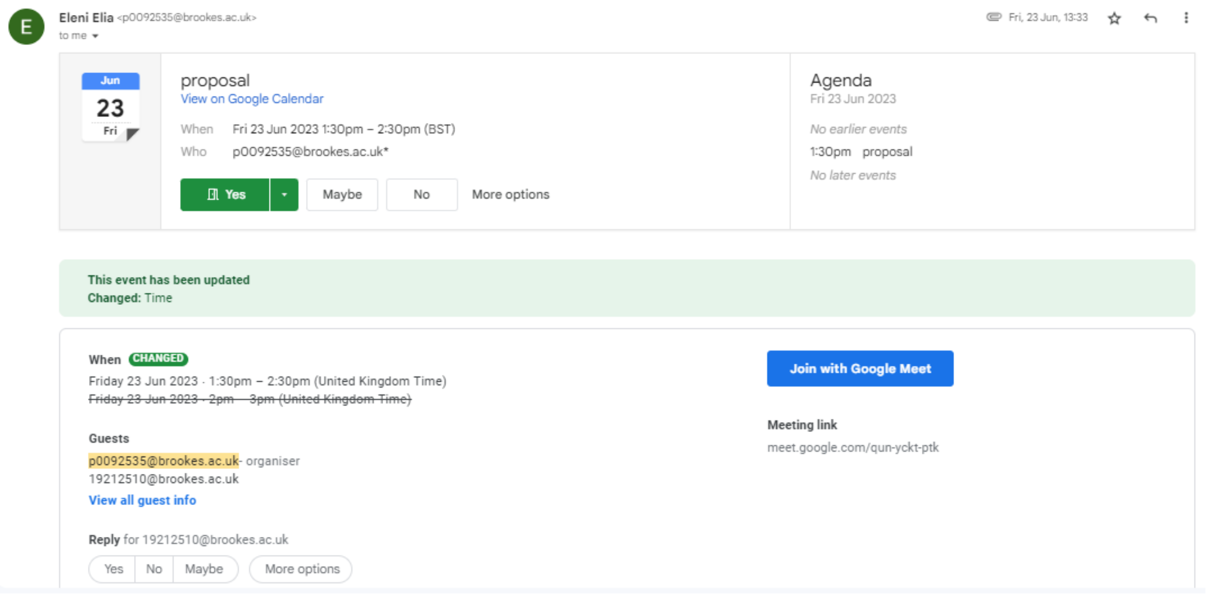
When
Friday 23 Jun 2023 · 2pm – 3pm (United Kingdom Time)

Guests
p0092535@brookes.ac.uk- organiser
19212510@brookes.ac.uk
View all guest info

Reply for 19212510@brookes.ac.uk
Yes | No | Maybe | More options

Join with Google Meet

Meeting link
meet.google.com/qun-yckt-ptk



Eleni Elia <p0092535@brookes.ac.uk>
to me

Fri, 23 Jun, 13:33

Jun 23 Fri

proposal
View on Google Calendar

When Fri 23 Jun 2023 1:30pm – 2:30pm (BST)
Who p0092535@brookes.ac.uk*

Yes | Maybe | No | More options

Agenda
Fri 23 Jun 2023

No earlier events
1:30pm proposal
No later events

This event has been updated
Changed: Time

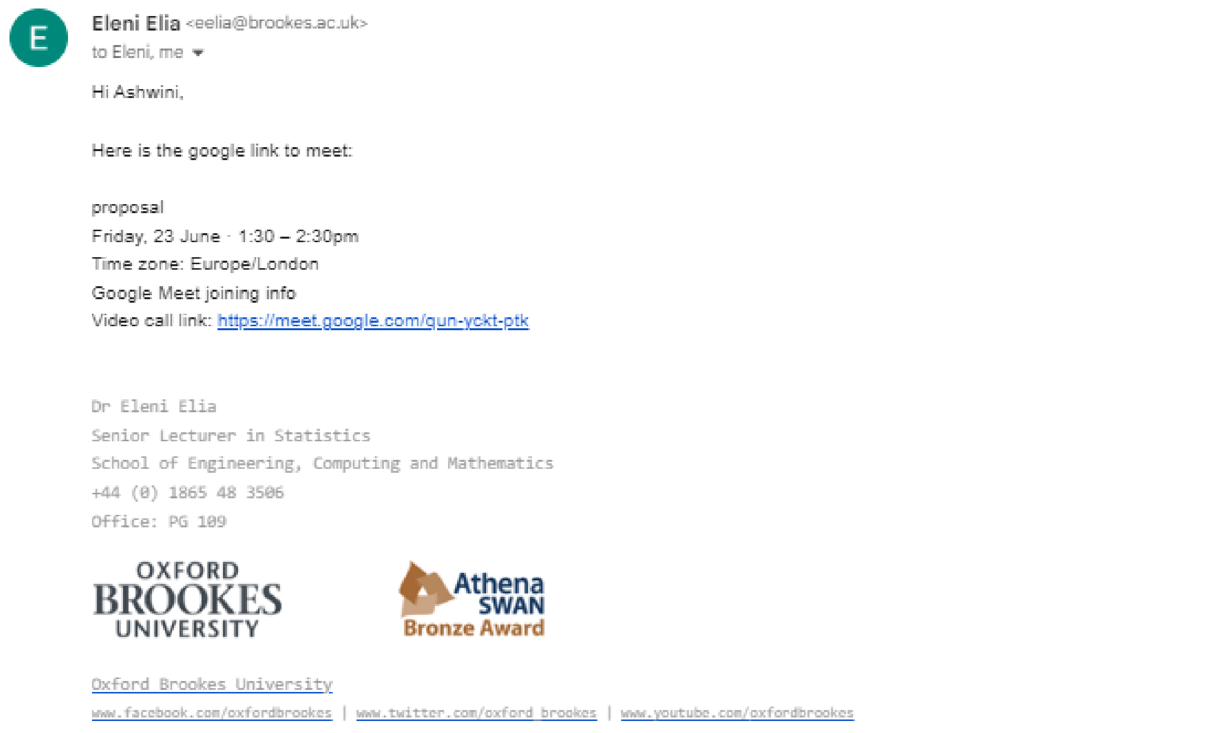
When CHANGED
Friday 23 Jun 2023 · 1:30pm – 2:30pm (United Kingdom Time)
~~Friday 23 Jun 2023 · 2pm – 3pm (United Kingdom Time)~~

Guests
p0092535@brookes.ac.uk- organiser
19212510@brookes.ac.uk
View all guest info

Reply for 19212510@brookes.ac.uk
Yes | No | Maybe | More options

Join with Google Meet

Meeting link
meet.google.com/qun-yckt-ptk



Eleni Elia <eelia@brookes.ac.uk>
to Eleni, me

Hi Ashwini,

Here is the google link to meet:

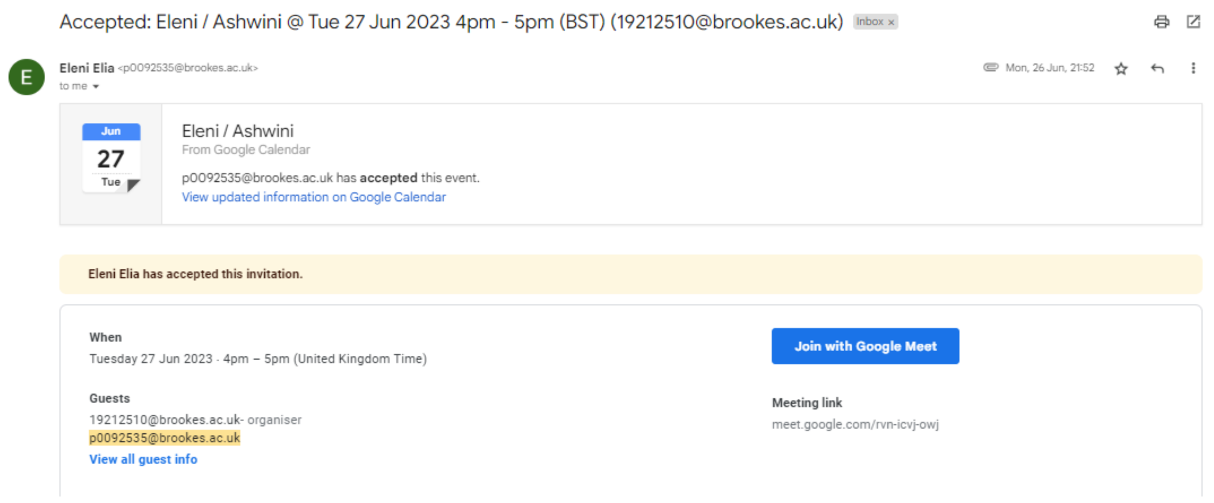
proposal
Friday, 23 June · 1:30 – 2:30pm
Time zone: Europe/London
Google Meet joining info
Video call link: https://meet.google.com/qun-yckt-ptk

Dr Eleni Elia
Senior Lecturer in Statistics
School of Engineering, Computing and Mathematics
+44 (0) 1865 48 3506
Office: PG 109

OXFORD BROOKES UNIVERSITY

Athena SWAN Bronze Award

Oxford Brookes University
www.facebook.com/oxfordbrookes | www.twitter.com/oxford_brookes | www.youtube.com/oxfordbrookes



Accepted: Eleni / Ashwini @ Tue 27 Jun 2023 4pm - 5pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Mon, 26 Jun, 21:52

Jun 27 Tue

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

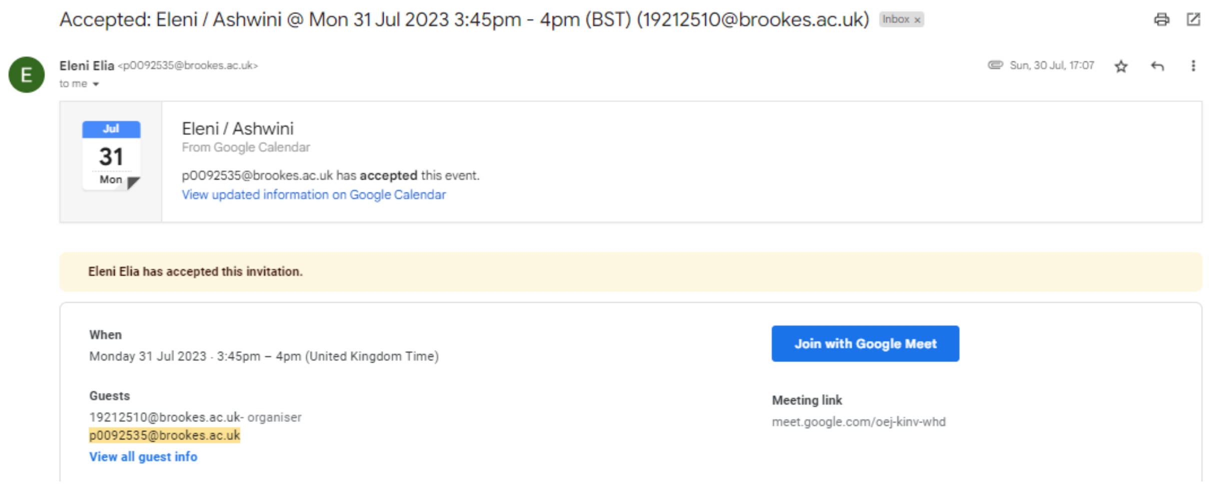
Eleni Elia has accepted this invitation.

When
Tuesday 27 Jun 2023 · 4pm – 5pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/rvn-icvj-owj



Accepted: Eleni / Ashwini @ Mon 31 Jul 2023 3:45pm - 4pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Sun, 30 Jul, 17:07

Jul
31
Mon

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

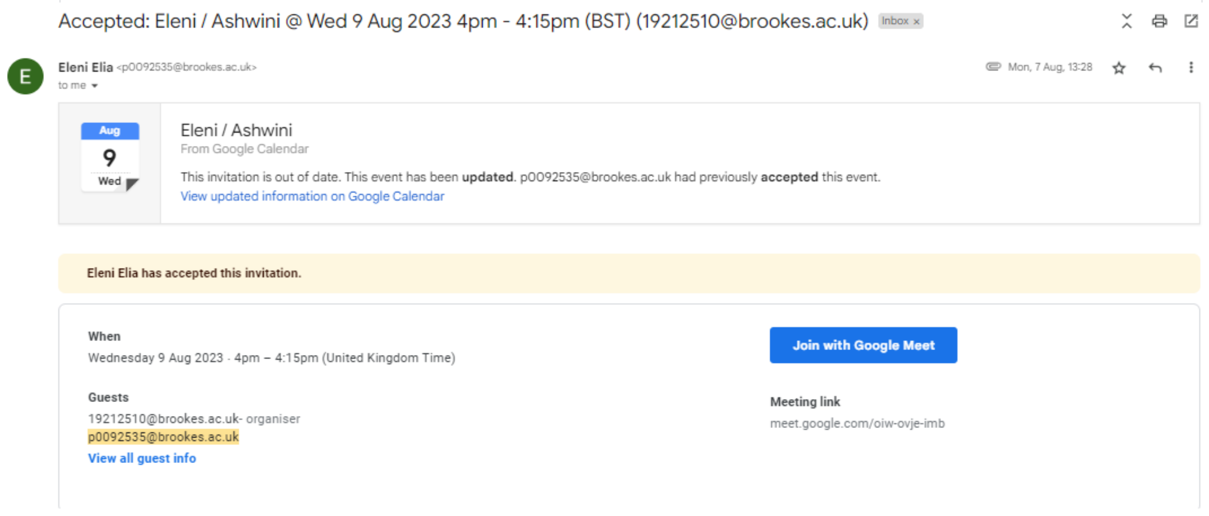
Eleni Elia has accepted this invitation.

When
Monday 31 Jul 2023 · 3:45pm – 4pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/oej-kinv-whd



Accepted: Eleni / Ashwini @ Wed 9 Aug 2023 4pm - 4:15pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Mon, 7 Aug, 13:28

Aug
9
Wed

Eleni / Ashwini
From Google Calendar
This invitation is out of date. This event has been **updated**. p0092535@brookes.ac.uk had previously **accepted** this event.
View updated information on Google Calendar

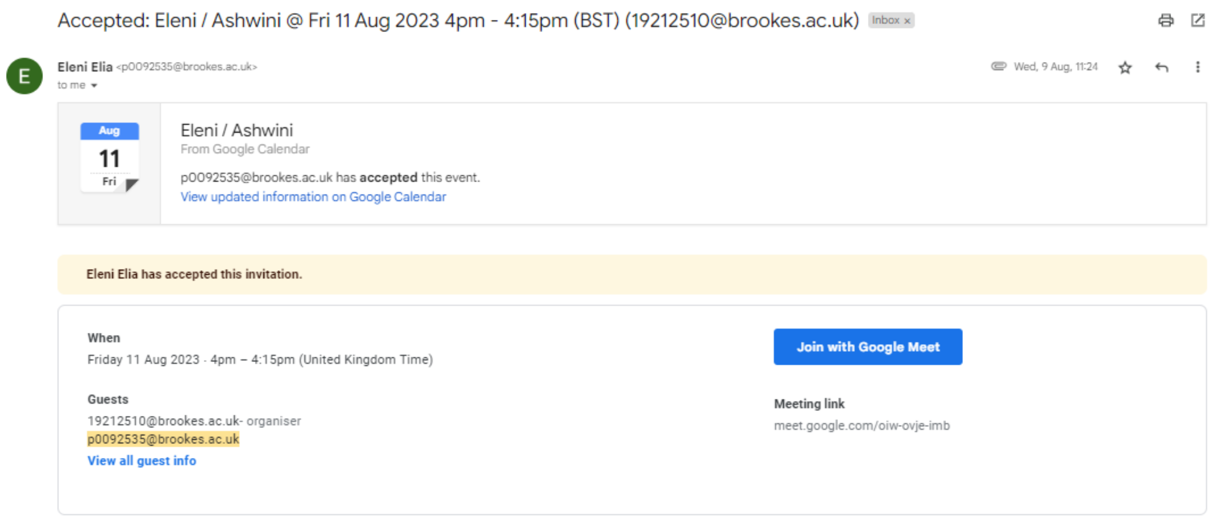
Eleni Elia has accepted this invitation.

When
Wednesday 9 Aug 2023 · 4pm – 4:15pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/oiw-ovje-imb



Accepted: Eleni / Ashwini @ Fri 11 Aug 2023 4pm - 4:15pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Wed, 9 Aug, 11:24

Aug
11
Fri

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

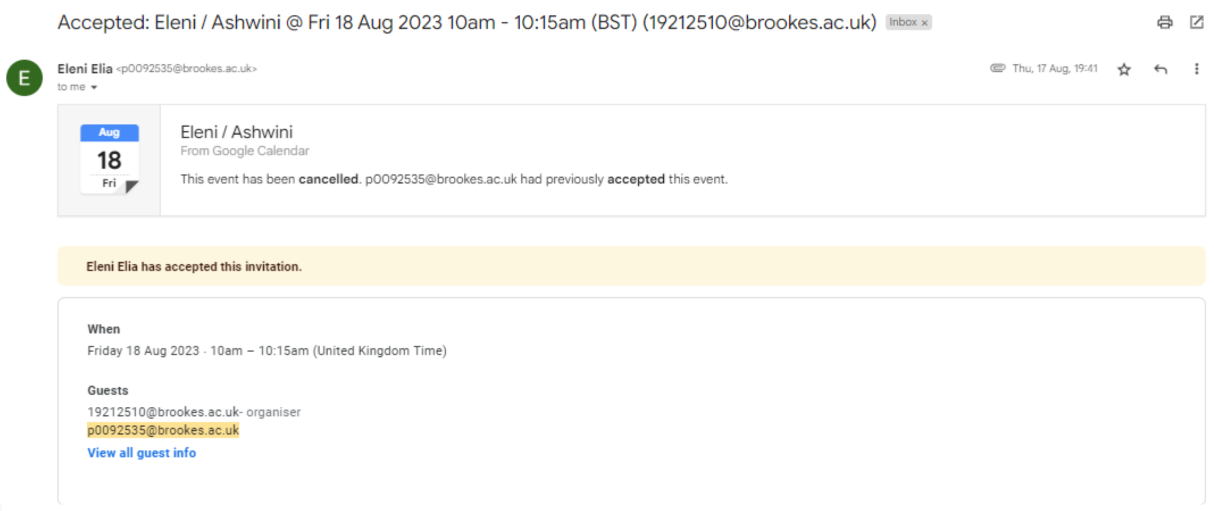
Eleni Elia has accepted this invitation.

When
Friday 11 Aug 2023 · 4pm – 4:15pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/oiw-ovje-imb



Accepted: Eleni / Ashwini @ Fri 18 Aug 2023 10am - 10:15am (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia <p0092535@brookes.ac.uk>
to me

Thu, 17 Aug, 19:41

Aug
18
Fri

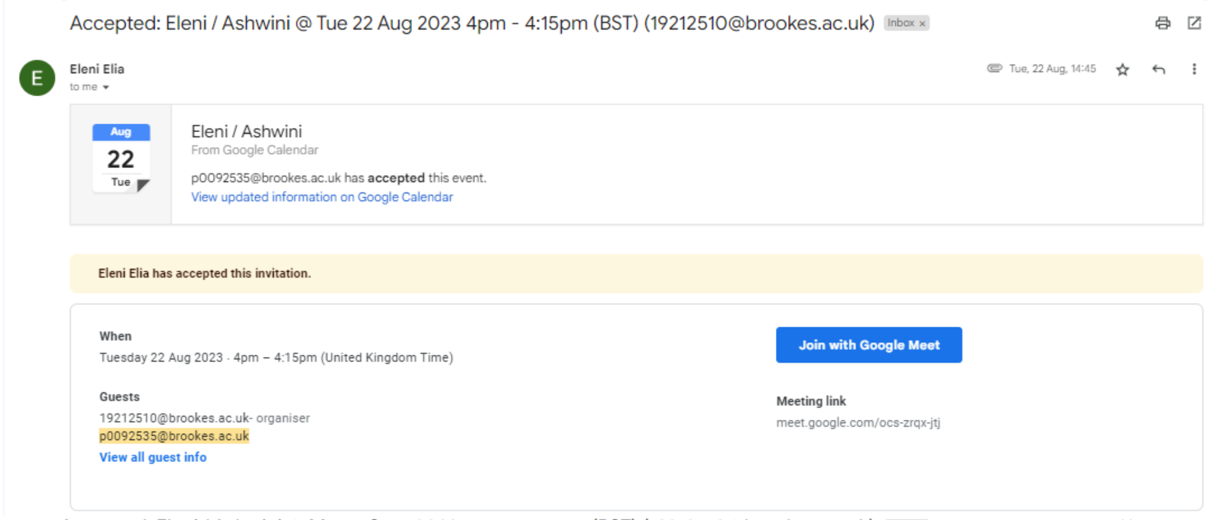
Eleni / Ashwini
From Google Calendar

This event has been **cancelled**. p0092535@brookes.ac.uk had previously **accepted** this event.

Eleni Elia has accepted this invitation.

When
Friday 18 Aug 2023 · 10am – 10:15am (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info



Accepted: Eleni / Ashwini @ Tue 22 Aug 2023 4pm - 4:15pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Tue, 22 Aug, 14:45

Aug
22
Tue

Eleni / Ashwini
From Google Calendar

p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

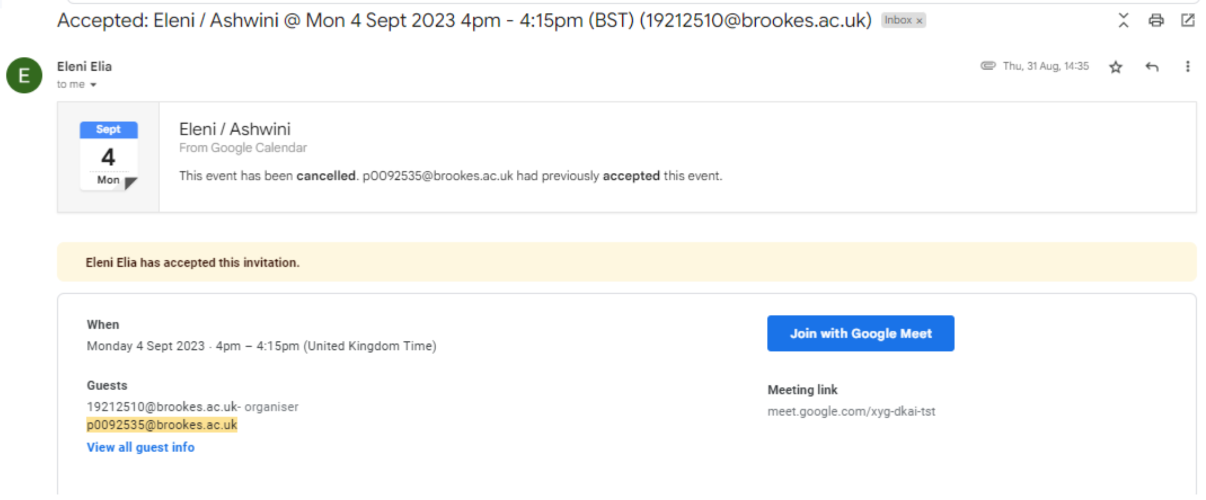
Eleni Elia has accepted this invitation.

When
Tuesday 22 Aug 2023 · 4pm – 4:15pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/ocs-zrqx-jtj



Accepted: Eleni / Ashwini @ Mon 4 Sept 2023 4pm - 4:15pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Thu, 31 Aug, 14:35

Sept
4
Mon

Eleni / Ashwini
From Google Calendar

This event has been **cancelled**. p0092535@brookes.ac.uk had previously **accepted** this event.

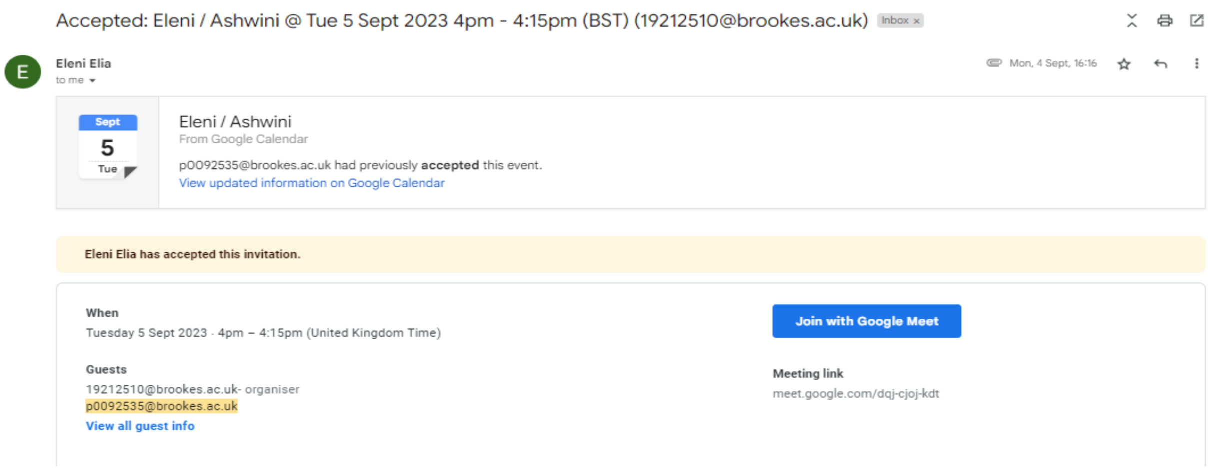
Eleni Elia has accepted this invitation.

When
Monday 4 Sept 2023 · 4pm – 4:15pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/xyg-dkai-tst


Accepted: Eleni / Ashwini @ Tue 5 Sept 2023 4pm - 4:15pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Mon, 4 Sept, 16:16

Sept 5 Tue

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk had previously **accepted** this event.
View updated information on Google Calendar

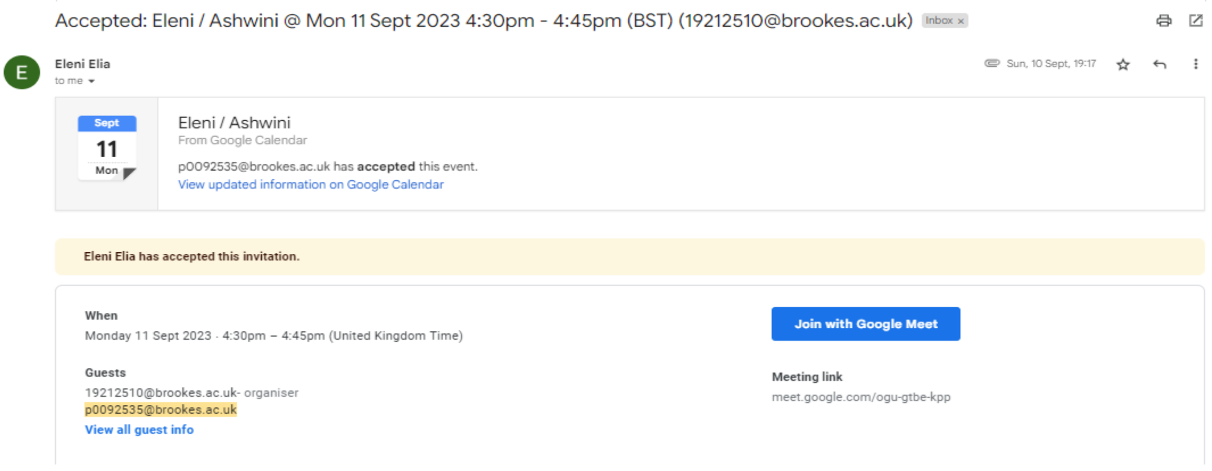
Eleni Elia has accepted this invitation.

When
Tuesday 5 Sept 2023 · 4pm – 4:15pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/dqj-cjoj-kdt


Accepted: Eleni / Ashwini @ Mon 11 Sept 2023 4:30pm - 4:45pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Sun, 10 Sept, 19:17

Sept 11 Mon

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

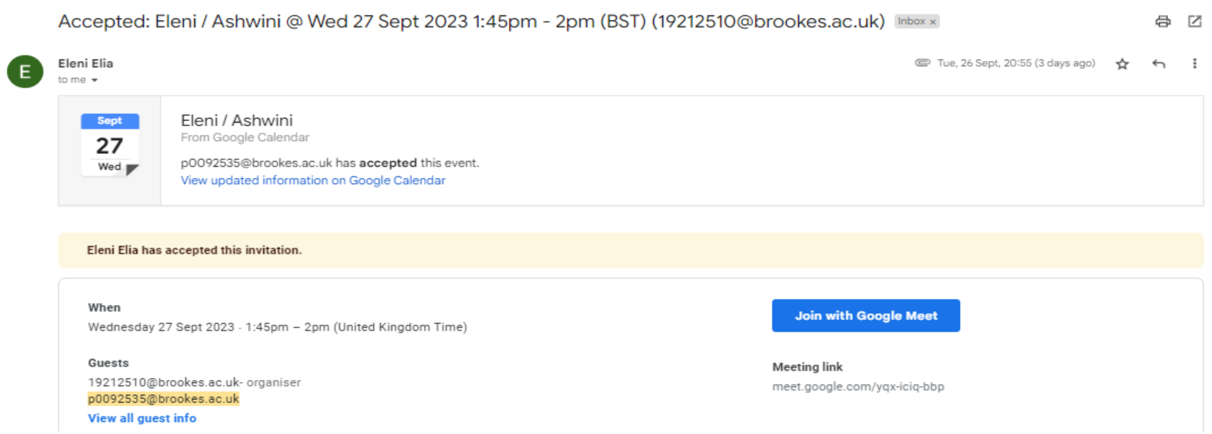
Eleni Elia has accepted this invitation.

When
Monday 11 Sept 2023 · 4:30pm – 4:45pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/ogu-gtbe-kpp


Accepted: Eleni / Ashwini @ Wed 27 Sept 2023 1:45pm - 2pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Tue, 26 Sept, 20:55 (3 days ago)

Sept 27 Wed

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

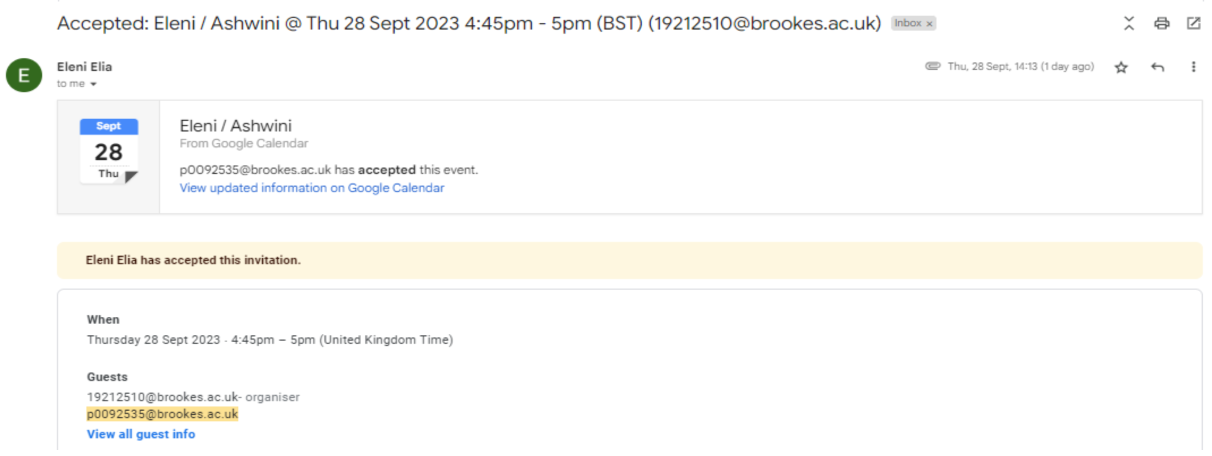
Eleni Elia has accepted this invitation.

When
Wednesday 27 Sept 2023 · 1:45pm – 2pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Join with Google Meet

Meeting link
meet.google.com/yqx-iciq-bbp


Accepted: Eleni / Ashwini @ Thu 28 Sept 2023 4:45pm - 5pm (BST) (19212510@brookes.ac.uk) Inbox ×

Eleni Elia
to me

Thu, 28 Sept, 14:13 (1 day ago)

Sept 28 Thu

Eleni / Ashwini
From Google Calendar
p0092535@brookes.ac.uk has **accepted** this event.
View updated information on Google Calendar

Eleni Elia has accepted this invitation.

When
Thursday 28 Sept 2023 · 4:45pm – 5pm (United Kingdom Time)

Guests
19212510@brookes.ac.uk- organiser
p0092535@brookes.ac.uk
View all guest info

Figure 49: Evidence of Meeting with Supervisor from May to September