Group 6 BAE Systems

Mars Express Challenge

Ishmeet Arora 11353387

Dylan Besson 10724837

Soe Lin Thant (Roger) 11473492

Danni Hu 10701242

Langyu Gu 10778076

1. Introduction

1.1. Background

The Mars Express Orbiter (MEX), a spacecraft launched by the European Space Agency (ESA) in 2003, has been orbiting Mars and providing valuable scientific data for over two decades (*Kelvins - Mars Express Power Challenge*, n.d.). Since beginning operations in 2004, MEX has made significant discoveries, including:

- Providing stunning three-dimensional images of Mars
- Mapping the chemical composition of the atmosphere
- Providing detailed information about Phobos, Mars's innermost moon
- Tracing the history of water on Mars, providing evidence that the red planet was once home to environmental conditions that may have been suitable for life (ESA, n.d.).

The spacecraft's operation is managed by a dedicated mission control team stationed at the ESA Operations Centre (ESOC) in Darmstadt, Germany. The team is responsible for preparing the mission plans and sending telecommands to the spacecraft, such as:

- Orbit and altitude control
- Science payload ON/OFF commands
- Radio reception and transmission plans (*Kelvins - Mars Express Power Challenge - Home*, n.d.).

MEX uses electric power generated by its solar arrays, or batteries during eclipses, to supply power to the platform units and the autonomous thermal subsystem, which regulates the temperature of various components within the spacecraft system within predefined operational ranges (*Kelvins - Mars Express Power Challenge*, n.d.). The remaining available power can be used to conduct science operations, as per the equation:

Science Power = Produced Power - Platform Power - Thermal Power

The thermal subsystem contains multiple nodes, including heaters and coolers, which activate as necessary and consume electrical power. Predicting the behaviour of this subsystem by modelling the power consumption of each of these nodes is of utmost importance to mission operators as it enables them to optimise science operations and ensure the spacecraft's safety (*Kelvins - Mars Express Power Challenge*, n.d.).

The MEX Flight Control Team from ESA collected three Martian years' (2061 Earth days, from 22/08/2008 to 14/04/2014) worth of MEX's telemetry data, comprising context data (explanatory variables) and measurements of the electric current (target variables) in each thermal subsystem node (Lucas & Boumghar, 2017). The data was then released as part of a machine learning challenge in 2016, encouraging participants to develop predictive models that could predict the average electric current of 33 power lines per hour for the fourth Martian year (687 Earth days, from 14/04/2014 to 01/03/2016). The response to the challenge was incredible; participants from all over the world built accurate machine learning models, some of which improved the sensitivity and accuracy of power consumption predictions compared to the existing empirical models (Lucas & Boumghar, 2017).

Mr. Gordon from BAE Systems presented us with the same challenge from the 2016 competition. This report documents our approach to developing a machine learning model for the MEX's power consumption, with the intention of presenting our findings to Mr. Gordon and the mentor team at the University of Manchester.

1.2. Project Objectives

The primary objective of this project is to develop a machine learning model that accurately predicts the average hourly electric current for each of the 33 power lines in MEX, using the provided context data. The model's performance is evaluated using root mean squared error (RMSE), which was the chosen metric in the original competition. Accurate predictions of power consumption are crucial for optimising space operations and ensuring the longevity and safety of the spacecraft.

1.3. Technical Terms

In order to understand the overall project, we must define some scientific terms that will come up repeatedly throughout the report.

Table 1: Description of technical terms used in the report.

Term	**Definition**
Unix time	A system for describing a point in time, defined in this report as the number of milliseconds that have elapsed since the Unix epoch, which is 00:00:00 UTC on 01/01/1970.
Martian year	The time it takes for Mars to complete one orbit around the sun, which is approximately 687 Earth days.
Occultation	When one celestial body moves in front of another, temporarily blocking the orbiter's view of the background object.
Eclipse	Refers to when an object obscures light from another object. For MEX, an eclipse occurs when Mars or one of its two moons, Deimos or Phobos, casts its shadow on the spacecraft, preventing sunlight from reaching the solar arrays.
Umbra	The darkest part of a shadow, where the light source is completely blocked by an opaque object. MEX experiences umbra during a total eclipse when Mars or one of its two moons entirely obscures the sun.
Penumbra	The outer part of a shadow, where the light source is only partially blocked by an opaque object. MEX experiences penumbra during partial eclipses when Mars or one of its two moons partially blocks sunlight from reaching the spacecraft.

Pericentre	The point in an orbit where a celestial body is closest to the body it orbits. For MEX, the pericentre is the point in its orbit where it is closest to Mars.
Solar constant	The amount of radiant energy received from the Sun per unit area per unit time at a distance of one astronomical unit (AU) from the Sun. In the context of MEX, the solar constant at Mars is related to the amount of solar energy available to the spacecraft's solar arrays.
Solar aspect angle	The angle between the Sun vector (the direction of the Sun from MEX) and MEX's reference axis. Solar aspect angle is important as it affects the amount of sunlight that falls on the spacecraft's solar arrays and the overall thermal state of the spacecraft.

1.4. About the Data

The Mars Express Power Challenge dataset consists of context data and observation data collected over four Martian years, each spanning approximately 687 Earth days:

- Year 1: 22/08/2008 to 10/07/2010
- Year 2: 10/07/2010 to 27/05/2012
- Year 3: 27/05/2012 to 14/04/2014
- Year 4: 14/04/2014 to 01/03/2016

The context data includes explanatory variables which describe the spacecraft's environment and operations, available for all four years. The observation data, available for only the first three years, contains the target variables representing the electric current measurements of 33 power lines.

In the original challenge, the dataset was divided into a training set (first three Martian years) and a test set (fourth Martian year). The training set included both context and observation data, while the test set only provided context data. The goal of the challenge was to predict the average electric

current per hour for the 33 power lines during the fourth Martian year. However, since the real values for the target variable for the fourth year were never released and the official challenge has ended, we could not evaluate the performance using Year 4 data. Therefore, for this project, we treated 20% of the data from Years 1-3 as test data.

The context data is organised into five different file types, each containing specific information for each Martian year:

1. SAAF (Solar Aspect Angles) files, which provide information about the spacecraft's orientation with respect to the Sun:
    - ut_ms: Unix timestamp in milliseconds.
    - sa: Solar aspect angle (angle between the spacecraft's solar panels and the Sun-Mars line).
    - sx, sy, sz: Solar angles of the X, Y, and Z axes of the satellite, respectively.
2. DMOP (Detailed Mission Operations Plan) files, which list the commands sent to various subsystems of the spacecraft:
    - ut_ms: Unix timestamp in milliseconds.
    - subsystem: Name of the operated subsystem command.
3. FTL (Flight Dynamics Timeline) files, which contain spacecraft pointing events:
    - utb_ms: Unix timestamp in milliseconds of the beginning of the pointing period.
    - ute_ms: Unix timestamp in milliseconds of the end of the pointing period.
    - type: Type of pointing or action.
    - flagcomms: Boolean indicating if any communication device was used.
4. EVTF (Event) files, which contain descriptions of other events such as eclipses or changes in altitude:
    - ut_ms: Unix timestamp in milliseconds.
    - description: Short description of the event .
5. LTDATA (Long-Term Data) files, which provide daily measurements of various parameters related to the spacecraft's environment:
    - ut_ms: Unix timestamp in milliseconds (one sample per day).
    - sunmars_km: Distance between the Sun and Mars in kilometres.
    - earthmars_km: Distance between Earth and Mars in kilometres.

- sunmarsearthangle_deg: Sun-Mars-Earth angle in degrees.
- solarconstantmars: Solar constant at Mars in W/m².
- eclipseduration_min: Total duration of all eclipses in the day, in minutes.
- occultationduration_min: Total duration of all occultations in the day, in minutes.

For each file type, there are separate files for each Martian year. For example, there are four SAAF files, one for each year, and the same applies to the other file types.

The observation data files contain the electric current measurements for the 33 power lines, each named using the convention "NPWDxxxx", where "xxxx" is a unique four-digit identifier for each power line, such as NPWD2372, NPWD2401, NPWD2402, and so on. In addition, these files contain timestamps in 'ut_ms' just like the other files.

While observations are typically made every 30 or 60 seconds in the observation data, anomalies exist. For example, sometimes the observations would be off by a few milliseconds or even seconds. To simplify the analysis and align the observation data with the goal of predicting hourly average electric currents, we resampled the data to a regular hourly interval during the preprocessing stage, which is discussed in more detail in section 3.

2. Project Timeline and Group Meetings

This section provides an overview of the key meetings and milestones of the project, from the initial kick-off meeting with Mr. Gordon from BAE Systems, up to the final stages of the project.

We held regular meetings to discuss progress, address issues, and ensure collaboration. The following table (Table 2) summarises the meetings we had and their outcomes.

Table 2: Dates of group meetings, along with any key discussions and findings.

Date	Meeting Summary
02/02 (Fri)	● Held initial kick-off meeting with Mr. Gordon from BAE Systems. ● Received task explanation and initial advice from Mr. Gordon. ● Gained resources explaining the background of the challenge and Mr. Gordon's expectations.
04/02 (Sun)	● Conducted initial group meeting to discuss first steps and project approach. ● Acknowledged lack of domain knowledge in astronomy. ● Agreed on the importance of background research and understanding technical terms. ● Decided to dedicate the first few weeks to research and literature review.
10/02 (Sat)	● Discussed findings from initial independent research. ● Examined the provided dataset. ● Assigned each group member to a specific context file for in-depth analysis: ○ Ishmeet - SAAF ○ Dylan - DMOP ○ Roger (Soe Lin Thant) - EVTF ○ Langyu - FTL ○ Danni - LTDATA ● Tasked members with understanding assigned files, explaining variables, and considering feature engineering techniques.

19/02 (Mon)	● Presented individual findings and insights about assigned context files to the group. ● Ensured thorough understanding of all variables among team members. ● Proceeded with data cleaning, exploratory data analysis (EDA), and feature engineering.
03/03 (Sun)	● Identified certain context files that required more work than others. ● Recognised DMOP and EVTF as the most challenging: ○ DMOP: Struggled to understand the different subsystems and their functions due to lack of documentation. ○ EVTF: Could not understand some of the events, which were not described clearly. ● Discussed issues and challenges, brainstorming solutions.
09/03 (Sat)	● Completed feature creation for all context files except DMOP. ● Discussed approach for merging context files into a single dataset. ● Decided to resample data to hourly granularity and established start and end datetimes. ● Assigned each member the task of implementing interpolation for integer features and forward filling for other features.
15/03 (Fri)	● Held progress update meeting between Mr. Gordon and Dylan. ● Discussed ongoing challenges with the DMOP file. ● Received valuable insights and guidance from Mr. Gordon. ● Obtained permission to use existing solution code from original challenge submissions to overcome DMOP issues.

17/03 (Sun)	• Resolved final resampling and data alignment issues across all context files. • Combined context files into a single consistent dataset before the Easter break.
10/04 (Wed)	• Discussed the best machine learning approach. • Assigned Isha and Danni to handle the machine learning process. • Assigned the project report planning and writing tasks to Roger, Dylan and Langyu.
Subsequent meetings	• Held meetings as needed to discuss progress and address any issues collaboratively.

A detailed, visual project timeline can be found in the Gantt chart below (Figure 1).

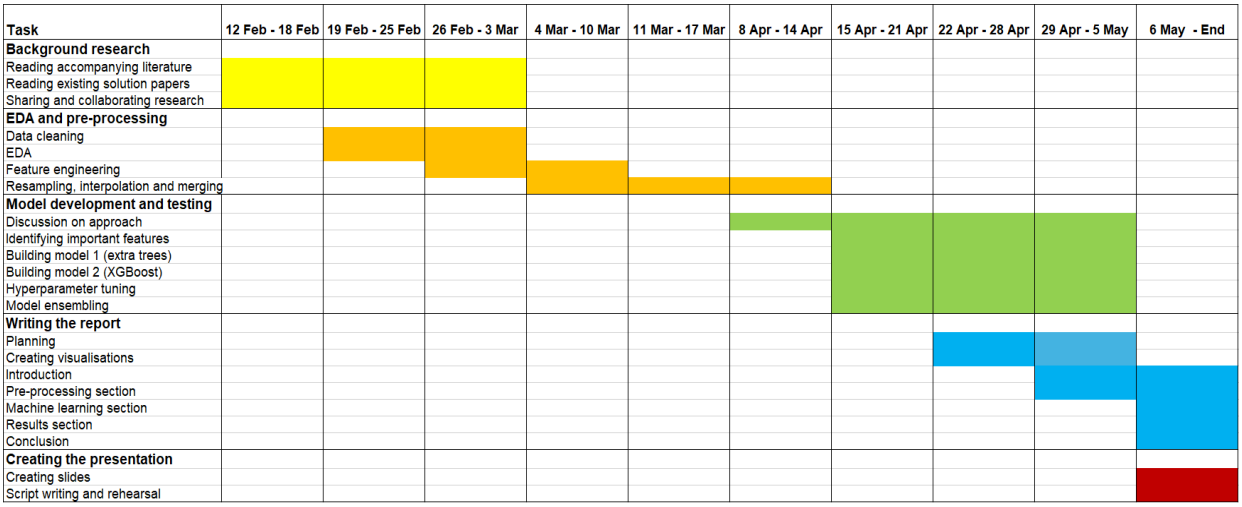

Figure 1: Gantt chart showing the progress of the project.

3. Data Pre-Processing

In this section, we will go through each of the five context files, highlighting any key points found during exploratory data analysis (EDA), and the features we created for each. The context files had no issues, such as missing or erroneous values, since The Advanced Mission Concepts' Data Analytics Team from ESOC had already prepared the data (*Kelvins - Mars Express Power Challenge - Home*, n.d.). Please refer back to Section 1.4 for descriptions of the original features.

3.1. SAAF

The angles given in the SAAF file are important because it details which side of the spacecraft would be heating up in the Sun. They also affect the amount of solar energy the solar panels on the spacecraft receive.

Apart from this observation, there was little else that we could observe or infer about the SAAF file, so we decided to keep all of the features.

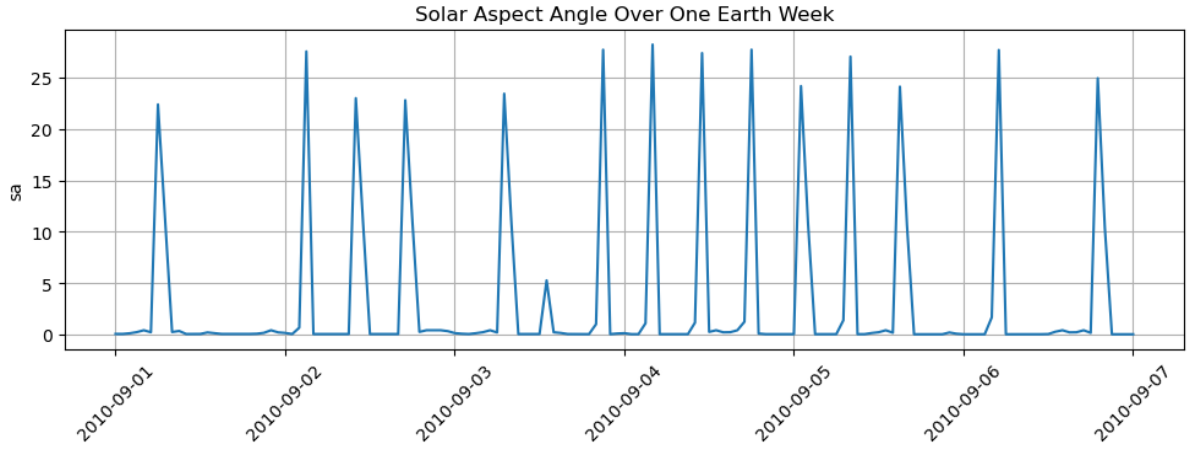

Figure 2: The solar aspect angle of the spacecraft over one Earth week, from 09/01/2010 to 09/07/2010.

The above graph (Figure 2) shows how the angles may be somewhat correlated, given that they have similar spiking patterns. However, nothing conclusive could be said.

3.2. DMOP

The DMOP files give raw strings under the ‘subsystem’ feature, which refer to subsystem commands being executed (e.g. “ASXX383C” or “TRIGGER”), or flight dynamics events which occur (e.g. “MAPO.000005”). There is no accompanying documentation on what these commands and events refer to in the physical sense. Furthermore, the string names are code names which are not interpretable using common sense. With over 29,000 unique strings, deciding which would be of importance to the target variable was a complex task.

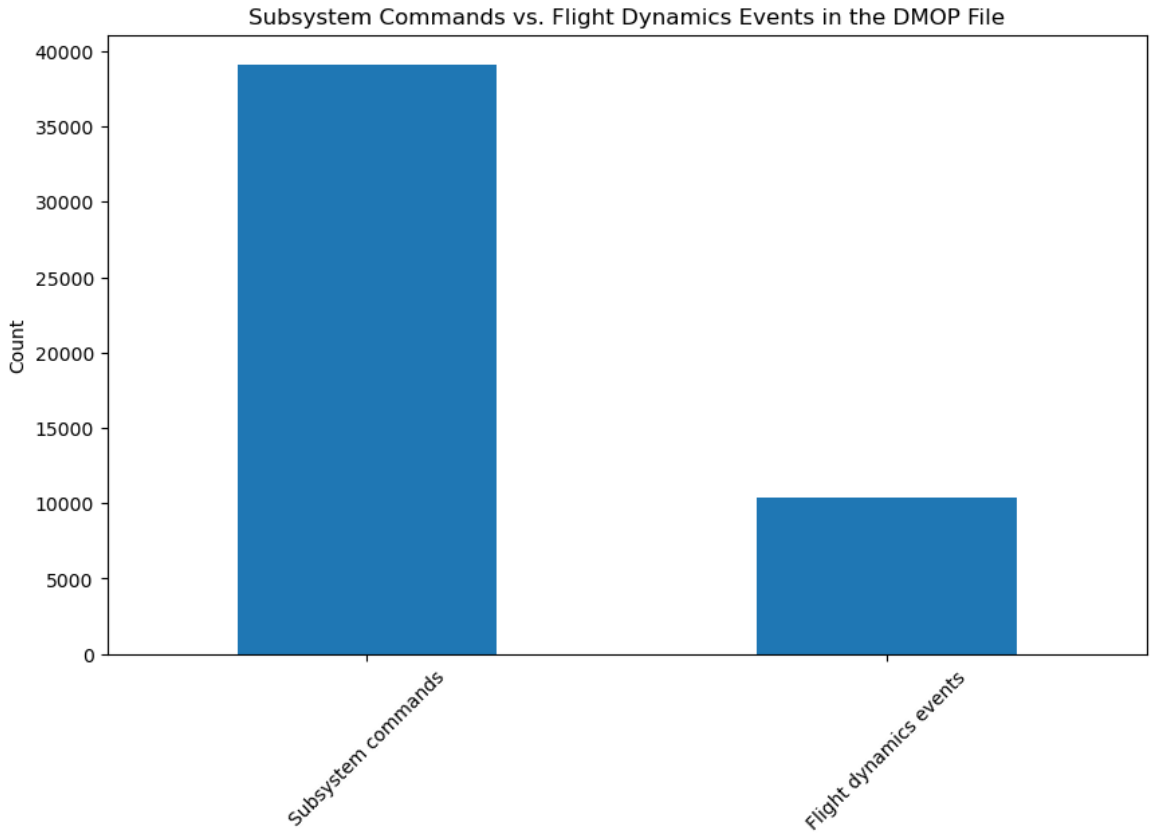

Figure 3: A bar chart comparing the number of subsystem commands and flight dynamics events in the DMOP file.

One attempt at identifying important strings was to group the strings based on the first four letters of the string, which was suspected to have some meaning. “ASXX383C” would be classified in the group “ASXX”, for instance. A simple XGBoost model was built, testing on a few of the 33

power lines and giving the most important four letter groups. Unfortunately, each powerline would give different groups different importances, so there was not enough reliable evidence to draw any conclusions here.

Some existing solutions for the project also described the importance of some of these commands as ON/OFF switches for different functions on the spacecraft. The strings suspected to be ON/OFF switches were very similar, generally only differing by one or two characters. Various attempts were made to identify such ON/OFF switches, such as using string comparison, or building simple models to find correlations between pairs of strings. Again, however, without any explicit description of these commands in any available literature, it was somewhat impossible to know which commands were ON/OFF switches for sure.

Although every effort was made to study and manipulate the DMOP file by ourselves, the difficulty of this task was extremely high, and it was delaying the overall progress of the project. Given these time constraints, we decided to use the set of ON/OFF switches that Github user stephanos-stephani (the second place user in the official competition) designed in order to proceed (stephanos-stephani, n.d.). This was a set of 18 binary-valued switches, called ‘pair0’, ‘pair1’, and so on.

A binary column, called ‘command’, was also added to differentiate between subsystem commands and flight dynamics events.

3.3. FTL

The FTL files originally had columns ‘utb_ms’ and ‘ute_ms’, giving the start and end times of any pointing actions. We believed that the duration of these pointing commands would be of little importance, as long as we captured where the spacecraft was pointing at regular intervals. Thus, we dropped the ‘ute_ms’ feature, and renamed ‘utb_ms’ to ‘ut_ms’ to make it consistent with the other context files.

In our EDA, we found there to be 18 different types of pointing action, shown in the graph below (Figure 4).

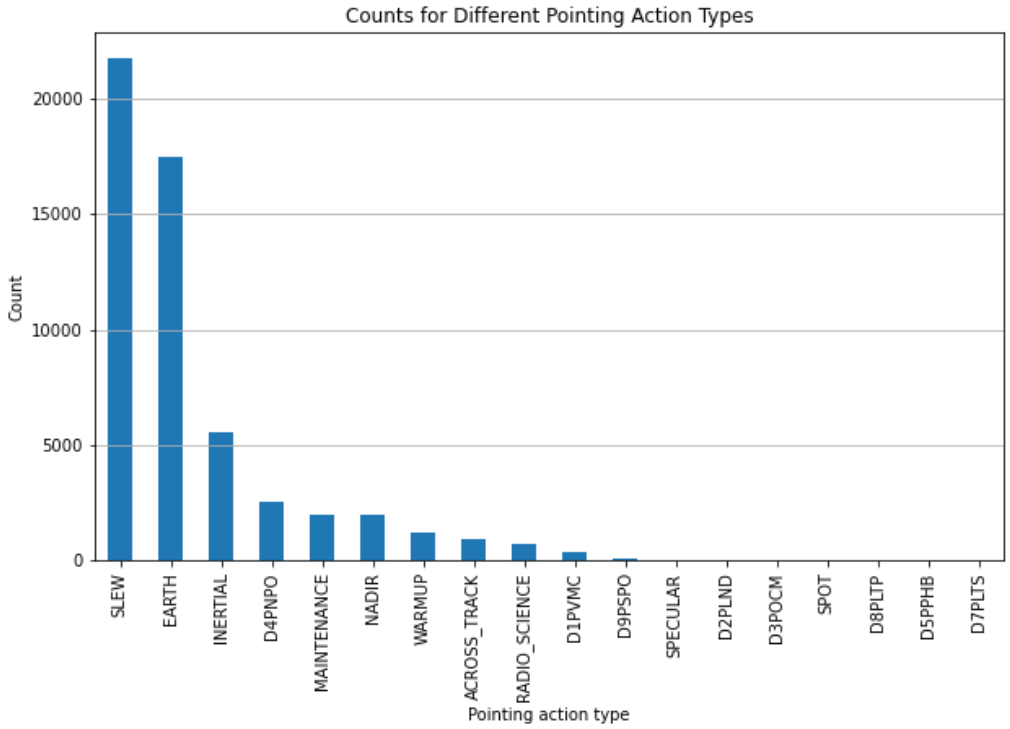

Figure 4: Bar chart showing the counts for different types of pointing actions.

Some pointing actions, like ‘EARTH’, obviously refer to the spacecraft facing towards Earth. Given that one of the main purposes of the MEX is to gather data about Mars, we suspected that the pointing actions starting with the letter ‘D’, such as ‘D4PNPO’, could refer to the pointing of the spacecraft to different areas of Mars. The other pointing actions are much less clear. Again, without further documentation, these are only speculations based on our intuition. Since there were only 18 different types, we decided to create dummy variables for each.

The binary column denoting whether a communications device was used, ‘flagcomms’, was also kept.

3.4. EVTF

The EVTF files contain over 89355 unique strings in the ‘description’ column. While these strings have clearer physical meanings, such as “MAR_UMBRA_START”, there were still too many different events, some of which required extensive research to understand. We also had to ensure

that our designed features captured as much information as possible, even when we would later downsample to an hourly time granularity.

After more thorough research, we decided to drop certain data entries. For example, there were strings such as "MLG_LOS_05_/_RTLT_02373", referring to a 'loss of signal', which we dropped. We believed a loss of signal would not have a great enough impact on the target variable to be worth keeping. This is consistent with the approach that some of the best submissions to the official challenge adopted (stephanos-stephani, n.d.; shelfwise, n.d.; alex-bauer, n.d.).

In contrast, other events that we deemed important, such as "MAR_PENUMBRA_START", were manipulated and transformed into binary variables. For example, a '1' for the 'penumbra' variable indicates that the spacecraft is currently in a penumbra. Figure 5 below shows what is meant by a penumbra, demonstrated on Earth instead of Mars.

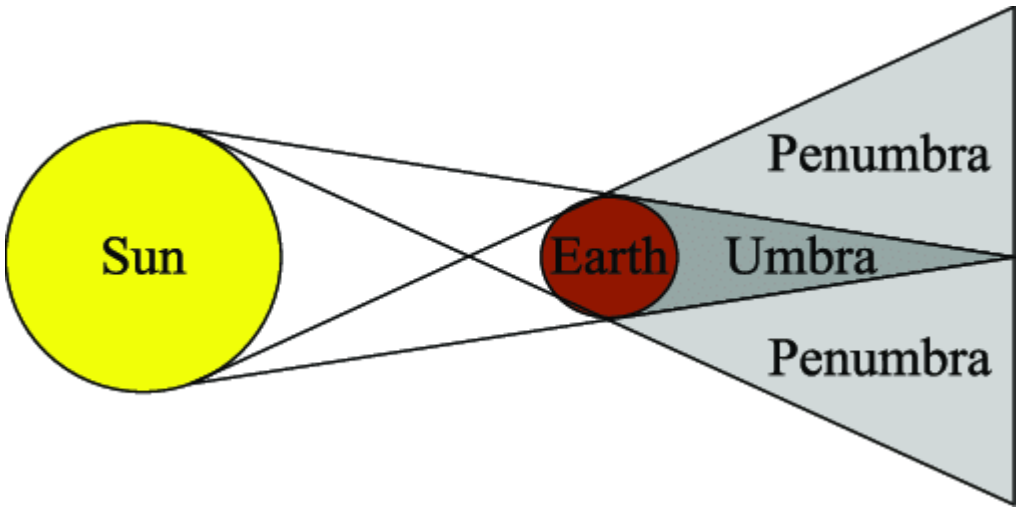

Figure 5: Diagram showing an umbra and penumbra on Earth (Jia et al., 2017).

We engineered the following features:

- 'altitude' (the current altitude of the spacecraft)
- 'direction' (whether the spacecraft is ascending or descending)
- 'umbra' (whether the spacecraft is currently in an umbra)
- 'penumbra' (whether the spacecraft is currently in a penumbra)

- ‘time_since_last_pericentre_ms’ (the time in milliseconds since the spacecraft’s last pericentre).

3.5. LTDATA

The LTDATA files did not require much manipulation. The features given are easy to interpret in the physical sense, and there were no useful new features to engineer.

The ‘sunmars_km’ feature was dropped, however, due to its very strong correlation with ‘solarconstantmars’. This is because the solar constant for Mars is related to the Sun-Mars distance by the inverse square law (Campbell, 2017).

Figure 6 below shows how the spacecraft spends varying amounts of time in occultation throughout its days.

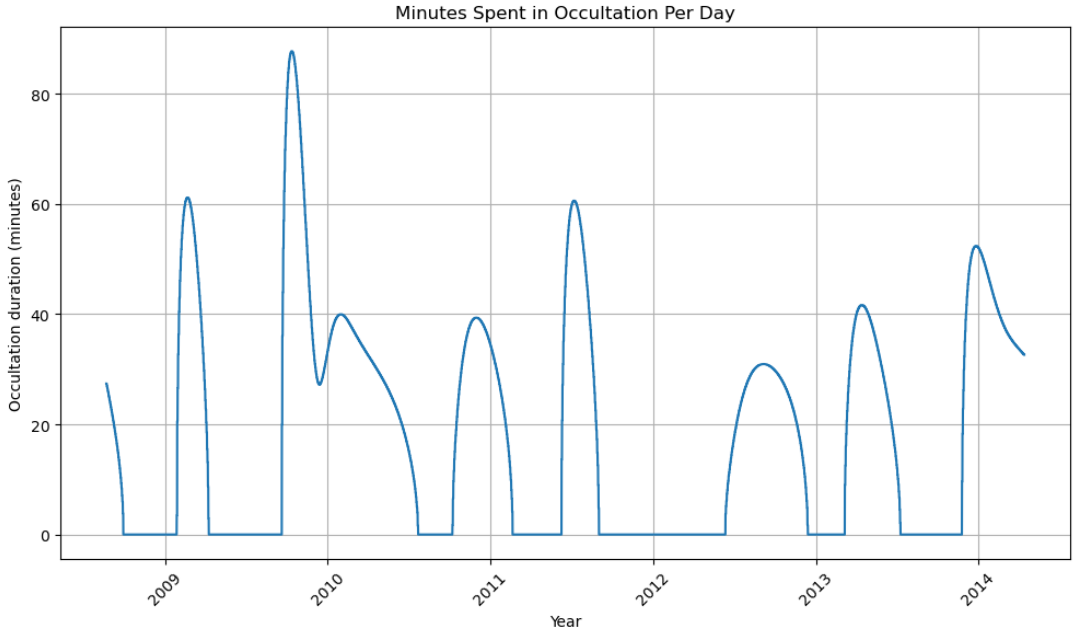

Figure 6: Line graph showing the minutes spent in occultation per day.

3.6. Resampling and Merging

Once the features were engineered for each context file, we looked to merge them as one before moving onto the machine learning stage. Given that each of the context files had different time granularities and start and end datetimes, we had to resample them all.

Within one of our meetings, we discussed what the best time granularity and start and end datetimes would be. Initially, we chose a time granularity of 30 seconds. We believed that the smaller the granularity, the more information we would retain and the more accurate our final predictions would be. Although this is true, considering we have three Martian years (2061 Earth days) and over 50 features, we realised model training would be extremely slow.

Therefore, we opted for an hourly time granularity. Many of our features do not fluctuate very much within each hour, so the loss of information here was not too great. In addition, this approach may have even avoided the capturing of unwanted noise. The start and end datetimes were then chosen to line up with the power datasets.

When resampling, we used a combination of linear interpolation and mean imputation for the numerical features. For the binary features, we selected the last entry for each hour to resample, then used forward or backward filling for the missing entries based on the scenario. Once resampled in a consistent manner, the five context files were merged into one seamlessly.

One more feature was also created at this stage, called ‘no_of_subsystem_commands’, which is an integer showing the number of commands in the DMOP file that appeared within the resampled hours.

3.7. Completed Feature Set

After the feature engineering phase, we ended up with over 50 features. The table below (Table 3) outlines them all.

Table 3: Descriptions of the final features, the context file they came from, and their data type.

Feature	Context File	Description	Data Type
sa	SAAF	Solar aspect angle.	Float
sx	SAAF	Solar angle of the x-axis of the satellite.	Float
sy	SAAF	Solar angle of the y-axis of the satellite.	Float
sz	SAAF	Solar angle of the z-axis of the satellite.	Float
command	DMOP	Determining between flight dynamics events (0) and subsystem commands (1).	Binary
pair0	DMOP	ON/OFF switch determined by strings 'AAAAF40B0' and 'AAAAF40C0'.	Binary
pair1	DMOP	ON/OFF switch determined by strings 'AAAAF40E0' and 'AAAAF40F0'.	Binary
…	…	…	…

pair17	DMOP	ON/OFF switch determined by strings containing 'UPBS’ and 'UPBE'.	Binary
no_of_subsystem_commands	DMOP	The number of commands within the resampled hour.	Integer
flagcomms	FTL	Whether a communications device was used in the pointing command.	Binary
type_ACROSS_TRACK	FTL	Dummy variable for ACROSS_TRACK pointing command types.	Binary
…	…	…	…
type_WARMUP	FTL	Dummy variable for WARMUP pointing command types.	Binary
altitude	EVTF	The current altitude of the spacecraft.	Integer
direction	EVTF	Whether the spacecraft is currently ascending or descending.	Binary
umbra	EVTF	Whether the spacecraft is currently in an umbra.	Binary

penumbra	EVTF	Whether the spacecraft is currently in a penumbra.	Binary
time_since_last_pericentre_ms	EVTF	Time since the spacecraft's last pericentre, in milliseconds.	Integer
earthmars_km	LTDATA	The average distance between the Earth and Mars per 24 hour day, in kilometres.	Float
sunmarsearthangle_deg	LTDATA	The average angle between the Sun, Mars and Earth per 24 hour day, in degrees.	Float
solarconstantmars	LTDATA	The solar constant for Mars per 24 hour day.	Float
eclipseduration_min	LTDATA	The total number of minutes spent in eclipse per 24 hour day.	Float
occultationduration_min	LTDATA	The total number of minutes spent in occultation per 24 hour day.	Float

4. Machine Learning

4.1. Approach

We recognised that each feature we had would have varying effects on the 33 different power lines, as the power lines were distinct from each other. Therefore, we decided to build 33 separate machine learning models, each with its own set of features. We came up with the following approach:

1. Reserve 20% of the data from Years 1 to 3, treating it as a test set.
2. Build 33 simple Extra Trees models on the training set without any hyperparameter tuning for each power line, with the sole purpose of extracting feature importances.
3. Extract the top 20 most important features for each power line.
4. Rebuild the 33 Extra Trees models with the features from step 2. Use hyperparameter tuning and 5-fold cross-validation to optimise performance on the validation set, measuring performance using the root mean squared error (RMSE) metric. Retrain the final model on both the training and validation set using the optimal set of hyperparameters.
5. Develop 33 XGBoost models, also using hyperparameter tuning and 5-fold cross-validation. Retrain the final model on both the training and validation set using the optimal set of hyperparameters.
6. For each power line, ensemble the results of its accompanying Extra Trees and XGBoost model using a simple mean to create a third, combined model.
7. Compare the performance of all three model types (Extra Trees, XGBoost, and ensembled) on the reserved test set, keeping the best performing model.

We chose Extra Trees over other similar methods because:

1. It is well-suited to deal with large datasets with high dimensionality, which is the case in our project with over 50 features.

2. It can quickly identify feature importances, allowing us to focus on the most relevant features and reduce computational complexity.
3. The random thresholds for splitting in trees helps to counter overfitting issues.

As for XGBoost, we chose it because:

1. It is also well-suited to deal with large datasets.
2. It has high predictive performance compared to other methods, especially when combined with hyperparameter tuning.
3. It is able to handle complex interactions and relationships among variables.

We decided to ensemble the Extra Trees and XGBoost models in order to combine the strengths of the two models. This approach allowed us to increase the robustness of the final predictions, as the ensemble model is less sensitive to the weaknesses of the individual models. Later in the report (section 5), we will see that this model performed slightly better than the individual models.

4.2. Train and Test Sets

As mentioned earlier, the original challenge required users to predict the power values for a fourth Martian year. Without the true values for this year being available, we had to improvise an unseen test set to evaluate our model on.

We decided that of the three Martian years we had available, we would reserve 20% as the unseen data set. We then used cross validation methods on the remaining 80% in order to train the model. We opted for a simple k-fold cross-validation approach assuming independent and identically distributed (i.i.d.) data points, rather than using time series cross-validation. This decision was based on the following factors:

1) Lack of strong temporal features: The dataset did not contain any direct temporal features. The absence of these features suggested that the i.i.d assumption was reasonable.
2) Computational efficiency: Time series cross validation can be very computationally intensive, especially when dealing with large datasets. Adopting a simple k-fold cross-validation approach allowed us to train our models much more efficiently.

3) Robustness of the models: The two models we used in this project, Extra Trees and XGBoost, can capture complex relationships in the data, even in the absence of temporal features.
4) Simplicity and interpretability: Using a simple k-fold cross-validation method allowed us to focus on the core aspects of the project, such as feature engineering, model training, and evaluation, without introducing additional complexity. This simple approach also enabled us to produce more interpretable results.

4.3. Feature Importances

Feature importance measures indicate how much a feature contributes to the model's prediction accuracy. We employed an Extra Trees regressor to calculate the feature importances for each power line, then selected the top 20 features. The Extra Trees regressor enhances predictive accuracy and mitigates overfitting by averaging the outputs of numerous randomised decision trees (i.e. extra trees).

When training the Extra Trees model, we used a consistent random state to ensure reproducibility of the results. We did not use hyperparameter tuning for this step, as it was solely for getting feature importances. Instead, we set 'n_estimators' parameter, which refers to the number of decision trees used for each model, to 100 for balanced accuracy and computational efficiency. Each tree in the model is constructed using a random subset of the available features.

During the construction of each tree, the Extra Trees algorithm differentiates itself from similar models by the way it chooses the splits. Instead of calculating the optimal thresholds for all features, Extra Trees randomly selects the split points for each feature, and then chooses the best one among these random splits to partition the data. This approach increases the model's randomness and helps to prevent overfitting.

The algorithm assesses each feature's split quality based on how much it decreases the node impurity (a measure of the homogeneity of the labels at the node). It uses mean squared error (MSE) to do so. A larger decrease in impurity indicates a higher feature importance. Our model averaged impurity reductions across all trees to derive feature importance scores for each feature

in each power line. These feature importance scores were then captured in a dataframe, and the 20 most significant features for each power line were identified.

Subsequent models were then built using only these features for each of the power lines, reducing model complexity and improving overall performance.

In the following graph (Figure 7), we can see the features which appeared in the top 10 most frequently.

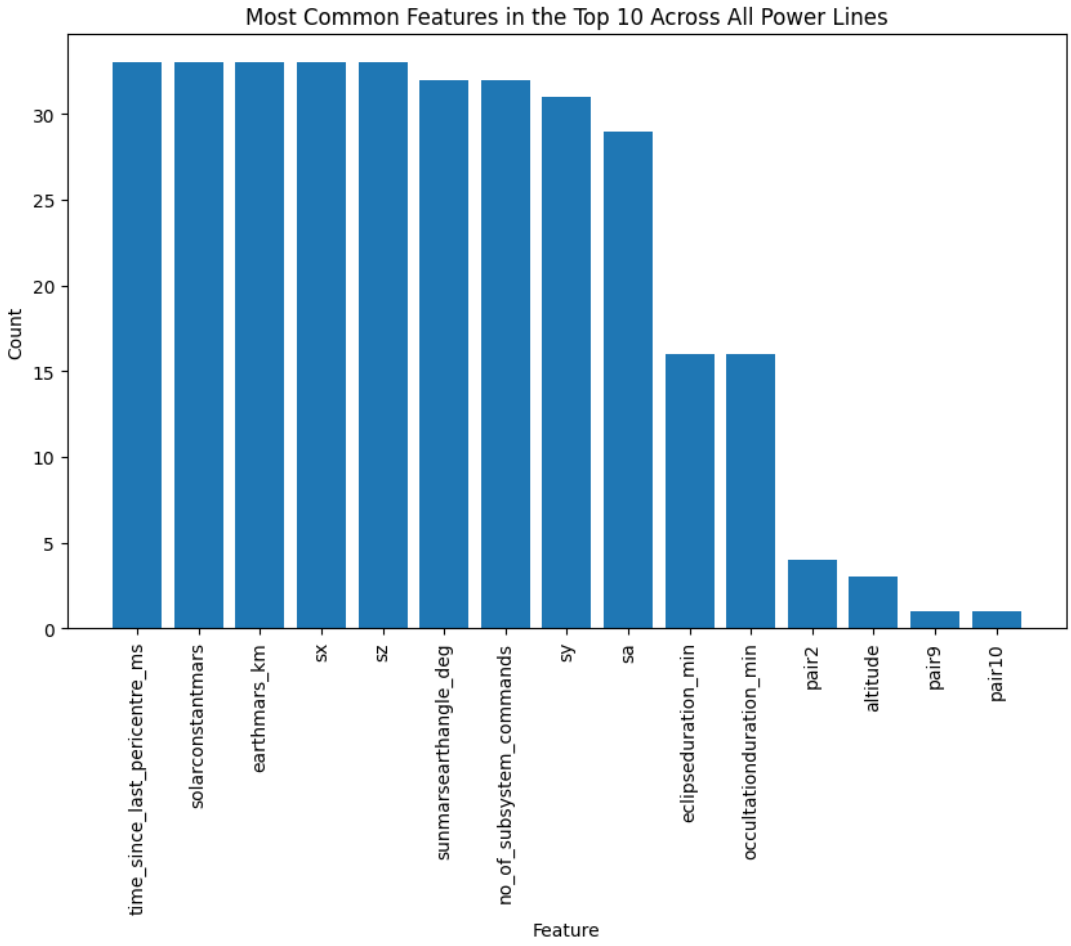

Figure 7: Bar chart showing the features that ranked top 10 at least once across all power lines.

We can see that 15 features appear on the top 10 most important features, with ‘time_since_last_pericentre’, ‘solarconstantmars’, ‘earthmars_km’, ‘sx’, and ‘sz’ present in every

power line. Overall, features related to solar angles, eclipses, or distances between objects seem to be the most important, which makes sense as MEX uses solar energy.

4.4. Extra Trees

After getting the top 20 most important features, we trained another set of Extra Trees models, this time using hyperparameter tuning. We set up a grid for: 'n_estimators', which refers to the number of trees in the forest; 'max_depth', which refers to the maximum depth of each tree; 'min_samples_split', which is the minimum number of samples required to split an internal node; 'min_samples_leaf', which is the minimum number of samples required to be at a leaf node, and 'bootstrap', which indicates whether bootstrap samples are used when building trees. Due to the high number of hyperparameter combinations, we used 'RandomSearchCV', using 50 iterations to perform a 5-fold cross-validation. This approach, as opposed to using 'GridSearchCV', allowed us to efficiently explore a wide range of hyperparameters without going through every combination. As for the scoring metric, we used negative mean squared error (neg_mean_squared_error), which effectively minimises the mean squared error (MSE). We then converted this metric into root mean squared error (RMSE), the metric used in the official challenge.

The complete grid for the candidate hyperparameters is shown below:

- 'n_estimators': [50, 100, 150]
- 'max_depth': [6, 8, 10, 12]
- 'min_samples_split': [2, 5, 10]
- 'min_samples_leaf': [1, 2, 4]
- 'bootstrap': [True, False]

4.5. XGBoost

In addition to Extra Trees, we also used XGBoost, an optimised gradient boosting algorithm. The algorithm works by employing a series of trees, with each new tree gradually improving the model's performance by correcting mistakes made by earlier trees in the sequence. In addition,

XGBoost introduces more complex regularisation techniques (L1 and L2) that help control overfitting (Chen & Guestrin, 2016).

The algorithm selects the most significant split points based on the gradient of losses, resulting in faster convergence and improved performance compared to traditional gradient boosting methods. Furthermore, XGBoost operates 10 times faster than typical implementations on a single processor and can handle billions of examples in environments with distributed resources or limited memory (Akande et al., 2022). In our trials, using comparable setups, XGBoost significantly outpaces other methods in processing speed, confirming its computational advantage which is often highlighted in studies.

For XGBoost models of each power line, we used a grid containing the following parameters: 'n_estimators', which determines the number of boosting rounds; 'max_depth', which refers to the maximum depth of each tree; 'learning_rate', which controls the step size at each boosting iteration; 'subsample', which is the fraction of samples to be used for each tree, and 'colsample_bytree', which represents the fraction of features to be used for each tree. The complete grid is shown below:

- 'n_estimators': [50, 100, 150, 200]
- 'max_depth': [6, 8, 10, 12]
- 'learning_rate': uniform distribution from 0.01 to 0.16
- 'subsample': uniform distribution from 0.6 to 1
- 'colsample_bytree': uniform distribution from 0.5 to 1

Key to XGBoost's configuration is its 'learning_rate'. The range of this hyperparameter we used is crucial for preventing the model from reacting too strongly to individual samples, thus avoiding overfitting. The goal is to find a learning rate that makes gradual updates and generalises well to unseen data.

We set the booster's 'objective' to negative mean squared error (neg_mean_squared_error) just like we did for Extra Trees, and converted it into RMSE.

4.6. Ensembling

In our predictive modelling, we implemented an ensemble strategy to harness the strengths of both the Extra Trees and XGBoost models. After finding the best Extra Trees and XGBoost models using hyperparameter tuning, we used a simple average of the predictions of the two models to derive the predictions of a new, third model. Like the individual models, the ensemble method's effectiveness was quantified by calculating the root mean squared error (RMSE), which provided a measure of the ensemble's performance against actual observed outcomes. In the next section, we will discuss the performance of all three models.

5. Results

For the Extra Trees models, while different models chose different hyperparameters, the most frequently chosen hyperparameters were:

- 100 for ‘n_estimators’
- 10 for ‘min_samples_split’
- 4 for ‘min_samples_leaf’
- 6 for ‘max_depth’
- ‘False’ for ‘bootstrap’

A complete table showing the optimal hyperparameters for each Extra Trees model (for each power line) is shown in Appendix 1A.

As for XGBoost, the most frequently chosen hyperparameters were:

- 100 for ‘n_estimators’
- 0.544246251 for ‘colsample_bytree’
- 0.039397429 for ‘learning_rate’
- 6 for ‘max_depth’
- 0.730132132 for ‘subsample’

A complete table showing the optimal hyperparameters for each XGBoost model is shown in Appendix 1B.

In summary, ‘n_estimators’ of 100 and a ‘max_depth’ of 6 were the most common for both types of models.

As for performance, the ensemble model did the best, with a mean RMSE of 0.0976 across all power lines. In comparison, Extra Trees and XGBoost had a mean RMSE of 0.0981 and 0.0978 respectively. It was interesting to see that XGBoost performed marginally better than Extra Trees despite taking significantly less time to train. While the performances of all three models were similar, the results demonstrate the power of ensembling, which produced the best performing model by simply combining the predictions of the two base models without the need for extra training. Therefore, if we had to choose one model for this challenge, it would be this ensemble model.

The accuracy (RMSE) of the three models’ predictions of each power line's power consumption can be seen in Appendix 2.

6. Discussion

In the following section, we will discuss parts of the project that went well, and parts of the project that we found challenging or areas where improvement could be made for next time.

6.1. Successes

We felt that changing from a one hour to a 30 second time granularity was a good decision. Given how long the machine learning models took to run at times, increasing the time granularity would not have been possible. The loss of information was greatly minimised by the use of interpolation and forward filling or back filling, and many of our features were designed to still be able to capture much of the same information.

Another success was in the use of ensembling to improve the machine learning model. We likely would have had to design multiple different types of machine learning model (Extra Trees,

XGBoost, Random Forest, etc.) to decide which would be best for our data anyway. Using ensembling on two of these lowered our final RMSE, with virtually no extra training time. This is because the calculation for the ensembling process was very simple.

6.2. Challenges and Improvements

The greatest challenge of the project was the lack of official documentation on the dataset. Many of the features given in the original data were impossible to interpret fully in the physical sense without any further context. As discussed earlier, this issue mainly arose when attempting to interpret the strings in the DMOP, FTL and EVTF files. This meant that we had to make our best assumptions at certain points, or it would prevent us from being able to design features that we otherwise may have been able to create.

Given the lack of official documentation, we often referred to the work of other users on Github, as well as other written reports available online. However, many of these past projects came with little to no explanation on their methodology and approach, or used knowledge much outside the scope of this course (expert knowledge in astronomy, for example). Therefore, we could not use these resources for much more than as guiding ideas. The only time we directly used another person's work was in the aforementioned creation of the ON/OFF switches for the DMOP file.

An improvement we could have made in the feature engineering process was the inclusion of delay effects in some of the features. Let us use the example of a subsystem command from the DMOP file in order to illustrate this point: if such a command moves a certain part of the spacecraft into sunlight at 2009-09-22 19:12:00, the temperature effect of this would not be felt at this time exactly. It may take an hour for the spacecraft to adjust, then heat up in its new position.

Unfortunately, again, without accompanying literature on how long such a command would take to come into effect, we felt it would be unwise to implement delay effects based on guesswork. In the end, we came to the conclusion that hourly resampling would indirectly implement a delay effect. When we resampled the files to an hourly time granularity, a command at timestamp 2009-09-22 19:12:00 would be captured at timestamp 2009-09-22 20:00:00, effectively creating a delay

of 48 minutes. If we were to do this project again, we may have tried adding different delay effects and retraining the model to see whether improvements were made.

Machine learning was another challenging aspect. Although Extra Trees and XGBoost models are relatively easy to understand, implementing them in our project proved to be a challenge due to the sheer size of the dataset. Hyperparameter tuning was the most time consuming in particular, even with the use of a random search as opposed to a grid search. In addition, due to time constraints, we were not able to examine the differences in the models' predictive accuracy across different power lines.

Additionally, it is important to acknowledge that the simple mean approach we used for ensembling may not always yield the best results. Further experimentation with advanced techniques such as stacking or weighted averaging could potentially lead to better performance, given more time and resources. That being said, using a simple mean provides a good balance between performance and complexity, which is suitable for our project.

7. Conclusion

In this project, our goal was to develop a machine learning model that accurately predicts the average hourly electrical current for each of the 33 power lines in the Mars Express Orbiter (MEX), using the telemetry data provided by the European Space Agency (ESA). In order to achieve this goal, we performed data-preprocessing, feature engineering, data merging, model development, ensembling, and model selection.

We began by preprocessing the five context files, which included data on solar aspect angles, mission operations, flight dynamics, events, and long-term data. We then performed feature engineering, extracting new, relevant features, before resampling the data to an hourly granularity and merging it into a single dataset.

As for the machine learning stage, we used Extra Trees and XGBoost models. We first identified the top 20 most important features for each power line using a simple Extra Trees model, reducing complexity. Next, we trained 33 Extra Trees models as well as 33 XGBoost models, one for each power line, using hyperparameter tuning to optimise performance. To enhance the

predictions, we used a simple averaging technique to ensemble the two models.. The ensembled model achieved the best performance, demonstrating the effectiveness of ensembling.

We also encountered challenges, the main one being the lack of documentation on the context files. This made it difficult to understand the context files and limited our ability to create new features. Moreover, time constraints and the large size of the dataset also posed challenges in the machine learning phase.

Despite the challenges, we successfully created a model that accurately predicts the average hourly electric current for the MEX's power lines, with a low mean RMSE 0.0976. This achievement can serve as a foundation for similar projects in the future.

In conclusion, this project showcases the power of applying machine learning to space operations. By utilising advanced machine learning techniques, we can develop models that can help spacecrafts optimise their scientific operations and ensure their safety and longevity, providing valuable insights to the aerospace industry.

References

Akande, Y. F., Idowu, J., Misra, A., Misra, S., Akande, O. N., & Ahuja, R. (2022). Application of XGBoost Algorithm for Sales Forecasting Using Walmart Dataset. *Lecture Notes in Electrical Engineering*, 147–159. https://doi.org/10.1007/978-981-19-1111-8_13

alex-bauer. (n.d.). *alex-bauer/kelvin-power-challenge*. GitHub. Retrieved May 7, 2024, from https://github.com/alex-bauer/kelvin-power-challenge

Campbell, A. (2017). *Inverse square law - Energy Education*. Energyeducation.ca. https://energyeducation.ca/encyclopedia/Inverse_square_law

Chen, T., & Guestrin, C. (2016). XGBoost: a Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, 785–794. https://doi.org/10.1145/2939672.2939785

ESA. (n.d.). *Mars Express*. Www.esa.int. https://www.esa.int/Science_Exploration/Space_Science/Mars_Express

Jia, X., Xu, M., Pan, X., & Mao, X. (2017). Eclipse Prediction Algorithms for Low-Earth-Orbiting Satellites. *IEEE Transactions on Aerospace and Electronic Systems*, *53*(6), 2963–2975. https://doi.org/10.1109/taes.2017.2722518

Kelvins - Mars Express Power Challenge. (n.d.). Kelvins.esa.int. Retrieved May 7, 2024, from https://kelvins.esa.int/mars-express-power-challenge/home/

Lucas, L., & Boumghar, R. (2017). *Machine Learning for Spacecraft Operations Support - the Mars Express Power Challenge*. https://doi.org/10.1109/smc-it.2017.21

shelfwise. (n.d.). *shelfwise/Mars-Express-Challenge: 3rd place solution to the Mars Express Power Challenge hosted by the European Space Agency*. GitHub. Retrieved May 7, 2024, from https://github.com/shelfwise/Mars-Express-Challenge

stephanos-stephani. (n.d.). *stephanos-stephani/MarsExpressChallenge*. GitHub. Retrieved May 7, 2024, from https://github.com/stephanos-stephani/MarsExpressChallenge

Appendix 1A: Best Hyperparameters for Extra Trees Models (for all 33 Power Lines).

	n_estimators	min_samples_split	min_samples_leaf	max_depth	bootstrap
NPWD2372	100	10	4	10	TRUE
NPWD2401	150	5	4	12	TRUE
NPWD2402	50	2	2	6	TRUE
NPWD2451	100	5	1	6	FALSE
NPWD2471	50	2	1	6	FALSE
NPWD2472	150	10	4	8	FALSE
NPWD2481	100	5	2	10	TRUE
NPWD2482	50	10	4	10	FALSE
NPWD2491	100	10	4	10	TRUE
NPWD2501	50	2	4	6	TRUE
NPWD2531	100	2	2	12	FALSE
NPWD2532	100	10	2	12	FALSE
NPWD2551	150	5	4	12	TRUE
NPWD2552	100	10	4	6	TRUE
NPWD2561	150	2	4	10	FALSE
NPWD2562	50	5	2	12	FALSE
NPWD2691	50	10	2	6	TRUE
NPWD2692	100	10	4	6	TRUE
NPWD2721	100	10	2	12	FALSE
NPWD2722	100	10	2	12	FALSE
NPWD2742	150	2	4	10	FALSE
NPWD2771	50	10	2	10	FALSE
NPWD2791	150	5	4	12	TRUE
NPWD2792	50	2	4	8	TRUE
NPWD2801	100	5	1	6	FALSE
NPWD2802	100	10	2	12	FALSE
NPWD2821	50	2	2	6	TRUE
NPWD2851	150	5	4	12	TRUE
NPWD2852	50	2	2	6	TRUE
NPWD2871	100	5	1	6	FALSE
NPWD2872	50	5	1	6	FALSE
NPWD2881	100	2	1	12	FALSE
NPWD2882	100	5	1	6	FALSE

Appendix 1B: Best Hyperparameters for XGBoost Models (for all 33 Power Lines).

	colsample_bytree	learning_rate	max_depth	n_estimators	subsample
NPWD2372	0.544246251	0.03939743	6	100	0.73013213
NPWD2401	0.544246251	0.03939743	6	100	0.73013213
NPWD2402	0.544246251	0.03939743	6	100	0.73013213
NPWD2451	0.544246251	0.03939743	6	100	0.73013213
NPWD2471	0.728034992	0.12777639	10	200	0.80569378
NPWD2472	0.640467255	0.09140441	6	50	0.92087879
NPWD2481	0.544246251	0.03939743	6	100	0.73013213
NPWD2482	0.544246251	0.03939743	6	100	0.73013213
NPWD2491	0.640467255	0.09140441	6	50	0.92087879
NPWD2501	0.885635173	0.0211067	10	100	0.64634762
NPWD2531	0.77335514	0.03772817	10	100	0.91005313
NPWD2532	0.901836038	0.03798551	8	100	0.8157369
NPWD2551	0.77335514	0.03772817	10	100	0.91005313
NPWD2552	0.652121121	0.08871346	12	50	0.71649166
NPWD2561	0.614399083	0.02154699	10	150	0.66448851
NPWD2562	0.614399083	0.02154699	10	150	0.66448851
NPWD2691	0.931551713	0.10349472	8	50	0.62542334
NPWD2692	0.761366415	0.07413115	8	200	0.64315657
NPWD2721	0.77335514	0.03772817	10	100	0.91005313
NPWD2722	0.640467255	0.09140441	6	50	0.92087879
NPWD2742	0.953783237	0.04739383	6	150	0.90222046
NPWD2771	0.901836038	0.03798551	8	100	0.8157369
NPWD2791	0.57800932	0.03339918	10	150	0.94647046
NPWD2792	0.544246251	0.03939743	6	100	0.73013213
NPWD2801	0.687270059	0.15260715	10	200	0.83946339
NPWD2802	0.77335514	0.03772817	10	100	0.91005313
NPWD2821	0.544246251	0.03939743	6	100	0.73013213
NPWD2851	0.953783237	0.04739383	6	150	0.90222046
NPWD2852	0.537275322	0.15803304	12	50	0.67948627
NPWD2871	0.728034992	0.12777639	10	200	0.80569378
NPWD2872	0.903720078	0.14441369	10	100	0.64402077
NPWD2881	0.805926447	0.03092408	12	200	0.74654474
NPWD2882	0.728034992	0.12777639	10	200	0.80569378

Appendix 2: The accuracy (RMSE) of the three models' (Extra Trees, XGBoost, and ensemble).

Power Line	XGBoost	ExtraTrees	Ensembled
NPWD2372	0.172283732	0.172470277	0.172244628
NPWD2401	0.008497694	0.008214061	0.008314558
NPWD2402	0.002605643	0.002601956	0.002600569
NPWD2451	0.846815897	0.846772208	0.846153745
NPWD2471	6.74E-05	6.74E-05	6.73E-05
NPWD2472	0.001347602	0.001343576	0.001344166
NPWD2481	0.024491871	0.024393309	0.024315925
NPWD2482	0.008235698	0.007897437	0.008002202
NPWD2491	0.206600939	0.20675045	0.206464057
NPWD2501	0.002161888	0.001291974	0.001493405
NPWD2531	0.116125174	0.116893884	0.115412053
NPWD2532	0.217897891	0.218421517	0.217489241
NPWD2551	0.511905096	0.513833462	0.510592508
NPWD2552	0.000355442	0.000355502	0.000355443
NPWD2561	0.360405847	0.361883782	0.360658542
NPWD2562	0.011609651	0.01197975	0.011660304
NPWD2691	1.95E-07	1.95E-07	1.95E-07
NPWD2692	4.86E-05	4.86E-05	4.86E-05
NPWD2721	0.099365801	0.099217629	0.09898885
NPWD2722	0.006906986	0.006725759	0.006684328
NPWD2742	0.035820551	0.035654786	0.035620161
NPWD2771	0.017690452	0.017331451	0.017445912
NPWD2791	0.076013638	0.077028202	0.076210782
NPWD2792	0.000968832	0.000969361	0.000968138
NPWD2801	0.000132738	0.000132595	0.000132627
NPWD2802	0.093159829	0.093458895	0.093017724
NPWD2821	0.000503032	0.000503375	0.000503103
NPWD2851	0.286320026	0.286514708	0.285621515
NPWD2852	4.64E-05	4.63E-05	4.63E-05
NPWD2871	4.53E-05	4.52E-05	4.52E-05
NPWD2872	6.40E-05	6.39E-05	6.40E-05
NPWD2881	0.118127875	0.12336866	0.119165371
NPWD2882	0.000294271	0.000294063	0.000294142

Appendix 3: Code repository link.

https://github.com/Isha-2001/MarsExpressChallenge