COMP SCI 7015

Software Engineering & Project

Sprint Retrospective 2

Rail Break Prediction AI

a1882259

Nilangi Maheesha Sithumini Edirisinghe

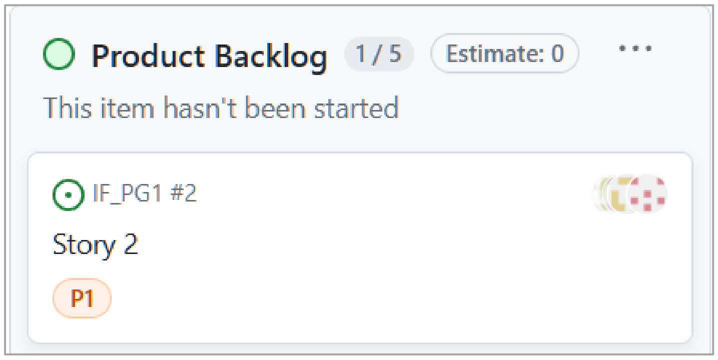
Group IF_PG1

List of Group Members:

a1873818 Zewei You, a1898921 Yong Kheng Beh, a1882259 Nilangi Edirisinghe, a1879038 Bhagya Indeewari Senanayake Senanayake Mudiyanselage, a1878048 Wenjing Shen, a1879321 Xinyue Ma, a1877644 Jiemin Zhanga, 1878387 Qiaoqiao Chen

1. Snapshots (Group)

1.1 Product Backlog and Task Board



Product Backlog 1 / 5 Estimate: 0

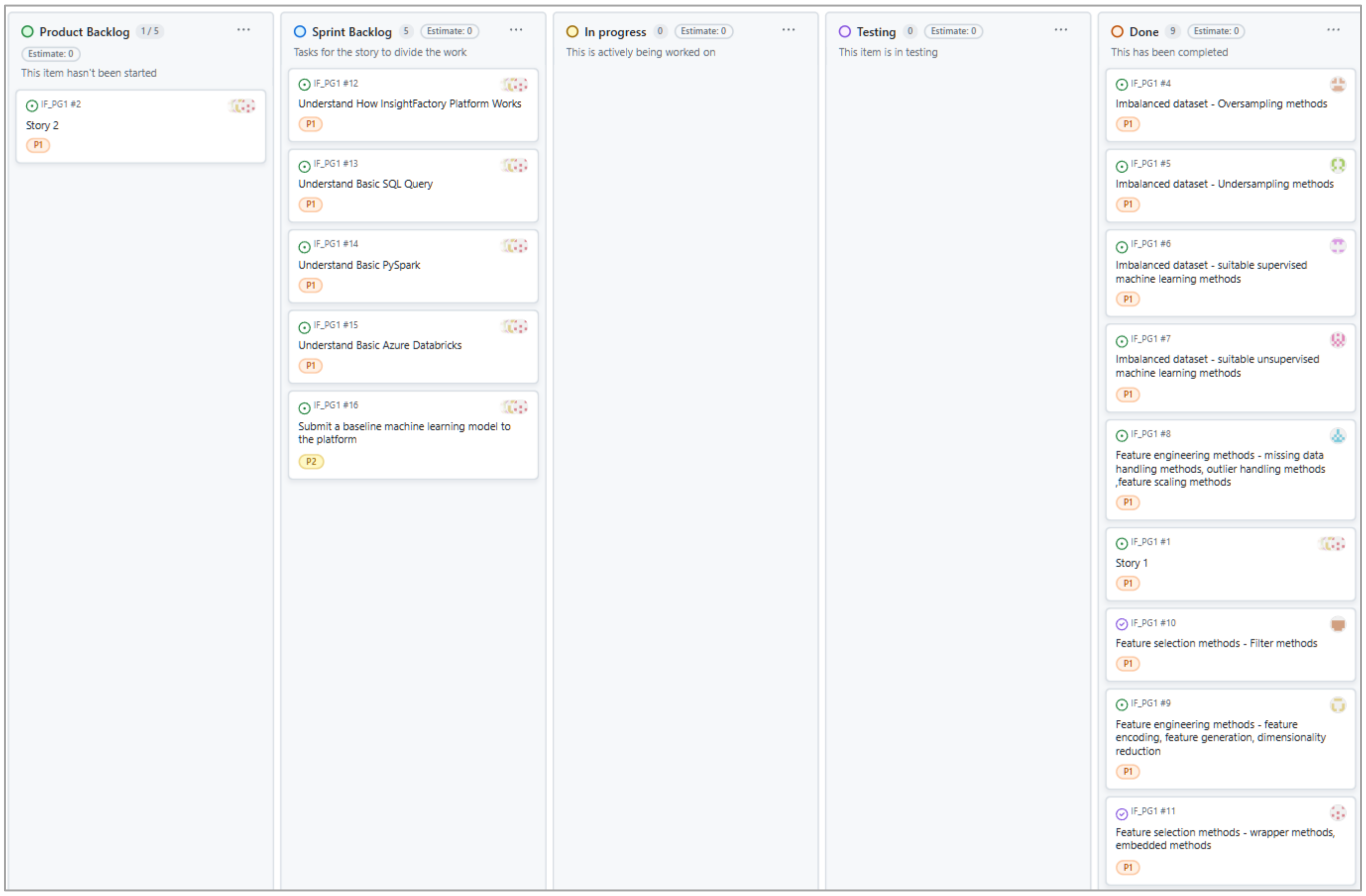
This item hasn't been started

IF_PG1 #2

Story 2

P1

Figure 1. The Screenshot of the Product Backlog



Product Backlog 1 / 5
Estimate: 0
This item hasn't been started

- IF_PG1 #2 — Story 2 — P1

Sprint Backlog 5 Estimate: 0
Tasks for the story to divide the work

- IF_PG1 #12 — Understand How InsightFactory Platform Works — P1
- IF_PG1 #13 — Understand Basic SQL Query — P1
- IF_PG1 #14 — Understand Basic PySpark — P1
- IF_PG1 #15 — Understand Basic Azure Databricks — P1
- IF_PG1 #16 — Submit a baseline machine learning model to the platform — P2

In progress 0 Estimate: 0
This is actively being worked on

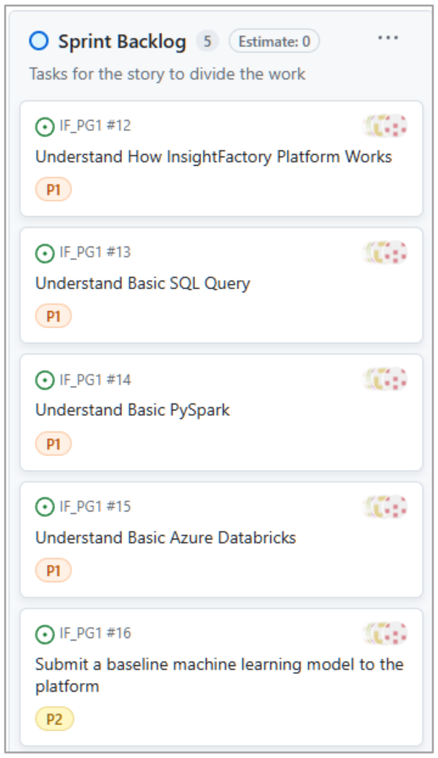
Testing 0 Estimate: 0
This item is in testing

Done 9 Estimate: 0
This has been completed

- IF_PG1 #4 — Imbalanced dataset - Oversampling methods — P1
- IF_PG1 #5 — Imbalanced dataset - Undersampling methods — P1
- IF_PG1 #6 — Imbalanced dataset - suitable supervised machine learning methods — P1
- IF_PG1 #7 — Imbalanced dataset - suitable unsupervised machine learning methods — P1
- IF_PG1 #8 — Feature engineering methods - missing data handling methods, outlier handling methods ,feature scaling methods — P1
- IF_PG1 #1 — Story 1 — P1
- IF_PG1 #10 — Feature selection methods - Filter methods — P1
- IF_PG1 #9 — Feature engineering methods - feature encoding, feature generation, dimensionality reduction — P1
- IF_PG1 #11 — Feature selection methods - wrapper methods, embedded methods — P1

Figure 2. The Screenshot of the Task Board

1.2 Sprint Backlog and User Stories



Sprint Backlog 5 Estimate: 0

Tasks for the story to divide the work

IF_PG1 #12
Understand How InsightFactory Platform Works
P1

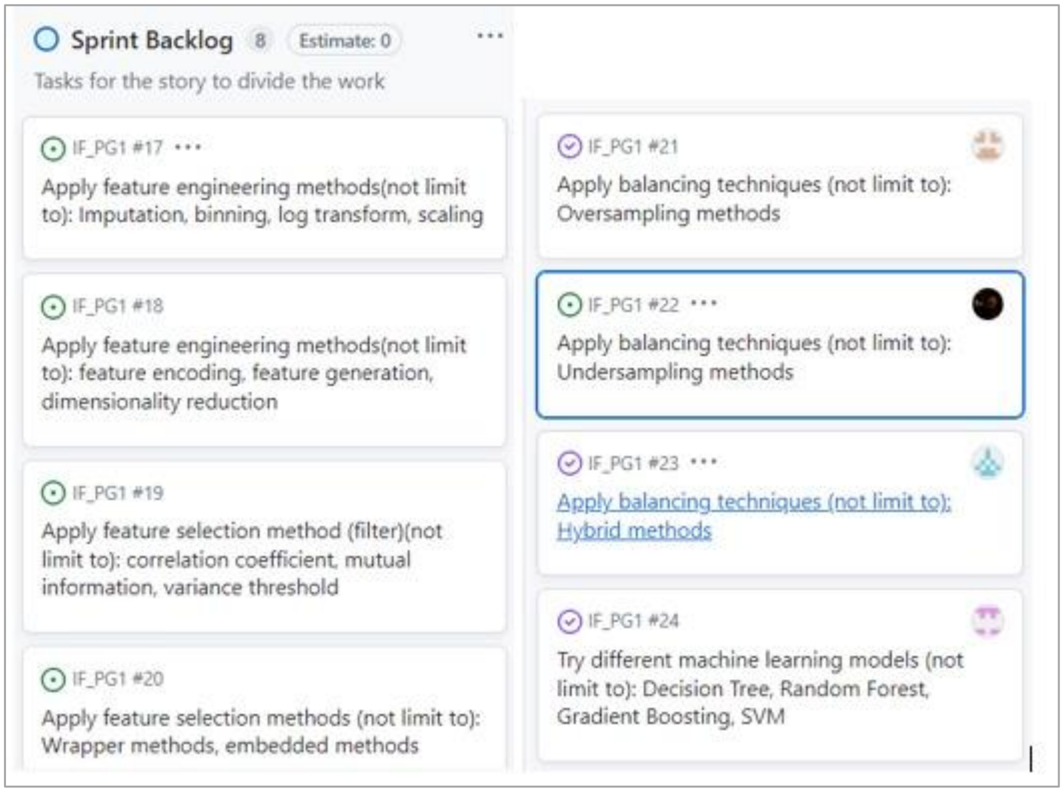
IF_PG1 #13
Understand Basic SQL Query
P1

IF_PG1 #14
Understand Basic PySpark
P1

IF_PG1 #15
Understand Basic Azure Databricks
P1

IF_PG1 #16
Submit a baseline machine learning model to the platform
P2

Figure 3. The Screenshot of the Sprint Backlog for First Week of Sprint 2



Sprint Backlog 8 Estimate: 0

Tasks for the story to divide the work

IF_PG1 #17
Apply feature engineering methods(not limit to): Imputation, binning, log transform, scaling

IF_PG1 #18
Apply feature engineering methods(not limit to): feature encoding, feature generation, dimensionality reduction

IF_PG1 #19
Apply feature selection method (filter)(not limit to): correlation coefficient, mutual information, variance threshold

IF_PG1 #20
Apply feature selection methods (not limit to): Wrapper methods, embedded methods

IF_PG1 #21
Apply balancing techniques (not limit to): Oversampling methods

IF_PG1 #22
Apply balancing techniques (not limit to): Undersampling methods

IF_PG1 #23
Apply balancing techniques (not limit to): Hybrid methods

IF_PG1 #24
Try different machine learning models (not limit to): Decision Tree, Random Forest, Gradient Boosting, SVM

Figure 4. The Screenshot of the Sprint Backlog for Second Week of Sprint 2

User Story 2

As a software engineer, I want to try out:

1) **Feature engineering methods**
2) **Feature selection methods and**
3) **Machine Learning (ML) techniques to approach a problem having an imbalanced dataset, so that I can produce my initial model to InsightFactory leaderboard**

Acceptance criteria :

1) **Implement and try out at least 3 techniques each for:**
    - **feature engineering methods**
    - **feature selection methods**
    - **ML techniques to approach a problem with an imbalanced dataset**
2) **Report what combination of techniques worked so far**
3) **Submit a model to InsightFactory leaderboard**

1. Definition of Done (DOD)

In the second sprint, as we gain access to the InsightFactory platform, the team will begin committing code while adhering to the following DoD to ensure that all defined standards are met and maintained. The DoD remains the same as in Sprint 1 but will be modified as needed when we begin actively committing code.

<u>Code Standards</u>

1) Readability and Maintainability:
    - Code must be written with clarity and follow established style guides (e.g., PEP 8 for Python).
    - All functions and classes should have clear, concise docstrings explaining their purpose, parameters, and return values.
2) Testing:
    - Unit tests must cover critical functions, especially for data preprocessing, feature engineering, and model evaluation.
    - Code should pass all tests before being merged into the main branch.

<u>Documentation</u>

1) Reproducibility:
    - All scripts and notebooks must be structured to allow easy reruns by other team members.
    - Include a README file that details how to set up the environment, run the analysis, and where to find each component of the project.
2) Data Pipeline Documentation:
    - The entire data pipeline from raw data ingestion to final model output should be documented.

- Include details on data cleaning steps, feature selection/engineering process, and the reasoning behind selecting specific machine learning models and hyperparameters.

3) Balanced Data Handling:
   - Clearly document the rebalancing techniques used (e.g., SMOTE, undersampling), along with the rationale for choosing them.
   - Include performance metrics before and after rebalancing to illustrate the impact of the techniques.

Machine Learning Model

1) Model Selection and Validation:
   - All selected models must be validated using appropriate techniques, including cross-validation and/or out-of-sample testing.
   - Performance metrics must be calculated for each model and compared against baseline models (e.g., logistic regression).
2) Hyperparameter Tuning:
   - Document the hyperparameter tuning process, including which parameters were tuned, the ranges tested, and the final selected values.
1) Model Interpretability:
   - If possible, include model interpretability steps (e.g., feature importance, SHAP values) to provide insights into how the model makes predictions.

Review and Sign-Off

1) Peer Review:
   - All code and documentation must undergo peer review before being marked as complete.
   - Reviewers must verify that all DoD criteria have been met.
2) Sign-Off:
   - The team must collectively sign off on each completed user story, ensuring all acceptance criteria are met and all necessary documentation is provided.

2. Summary of Changes:

During the first sprint, the team concentrated on researching and identifying techniques for feature engineering, feature selection, and machine learning (ML) to manage imbalanced datasets.

In the first week of Sprint 2, the team gained access to the InsightFactory (IF) platform, which is essential for model implementation. Each member is individually exploring the platform to gain hands-on experience in creating a production line, including connecting databases, ingesting data into a data lake, performing data preprocessing, and training a machine learning model using Databricks notebooks. The team will discuss their findings and address any questions at the beginning of the second week of Sprint 2 while submitting initial models to the InsightFactory leaderboard.

I attended the sprint review/planning meetings on Monday, the 26th of August with my group and Monday, the 2nd of September 2024 with the tutor, and presented my results.

2. What went well in the sprint?

In the second sprint, we were assigned to put our research into practice and experiment with the feature selection, feature engineering methods and machine learning techniques to approach a problem as well as submit an initial machine learning model to the leaderboard. As a group, we decided to dedicate the first week to learning the platform and running a basic pipeline as introduced to us by Insight Factory. Hence, by the end of the first week of the sprint, almost everyone in the group had successfully created their own production line on the platform and run a pipeline to submit a basic model, according to the tutorial given by InsightFactory.AI.

By the end of the first week, we had a meeting within the group members and discussed all the issues we had.

In the second week of the second sprint, the leader delegated the tasks further, such as experimenting on feature engineering methods, feature selection methods and various machine learning techniques. I was assigned to implement the feature selection methods. First, we all got together and created a common SQL query to extract the data from the database and used that for each delegated task. I implemented many feature selection methods including Correlation Coefficient, Mutual Information and Variance Threshold and presented my findings at the sprint meeting.

Overall, as a group, we were successful in implementing and trying out many techniques for feature engineering, feature selection, machine learning methods and our group was able to submit a model to Insight Factory leaderboard and achieve a high score, ranking in the second place.

3. What Could be Improved?

During the sprint meeting where we presented our progress to the product owner, we found that we ran out of time. We put in a lot of effort in creating a presentation and when half of the group members presented their work and asked questions, the other half did not have time to present their work to the product owner. Luckily, as I was the second in line to present, I was able to present all my findings and discuss with the product owner, however, as a group we were not very good at managing the limited amount of time given to us for the sprint meeting. Hence, next time, time management during the sprint meeting should be improved.

Additionally, there was a lack of collaborative troubleshooting when certain members encountered issues with running pipelines. Although knowledge sharing is crucial in group projects, the collaboration was somewhat siloed, with individuals focusing primarily on their tasks rather than helping others. In the next sprint, fostering a more open and supportive environment for discussion and mutual assistance should be prioritized to ensure the entire team can overcome obstacles efficiently.

4. What will the Group commit to Improve in the Next Sprint?

For the upcoming sprint, the group has committed to expanding the experimentation with different machine learning models and fine-tuning their hyperparameters to improve model accuracy. We will build on the findings of this sprint, particularly the feature selection insights, and apply them to a more robust machine learning model. One of the goals is to increase the F1 score of our model beyond 55%, which will require continued testing of feature engineering methods and parameter optimization. Additionally, the group will work towards improving time management in sprint meetings, ensuring that all members have equal opportunities to present their work. Collaboration will also be a focal point, with the aim of increasing transparency and mutual support among team members, especially when issues arise in technical implementations. This commitment will foster a more unified approach to problem-solving, ultimately leading to a more refined and successful final product.

5. Comment on My Progress This Sprint

In the last sprint, what I did was research about feature selection methods to approach our problem. In this sprint, I improved on it vastly by putting it into practice with the real world data and experimenting with different feature selection methods and deriving insights from them.

As the first step, I built a machine learning pipeline in the insightfactory.ai platform according to Toby's instructions and submitted that basic machine learning model to the leaderboard.

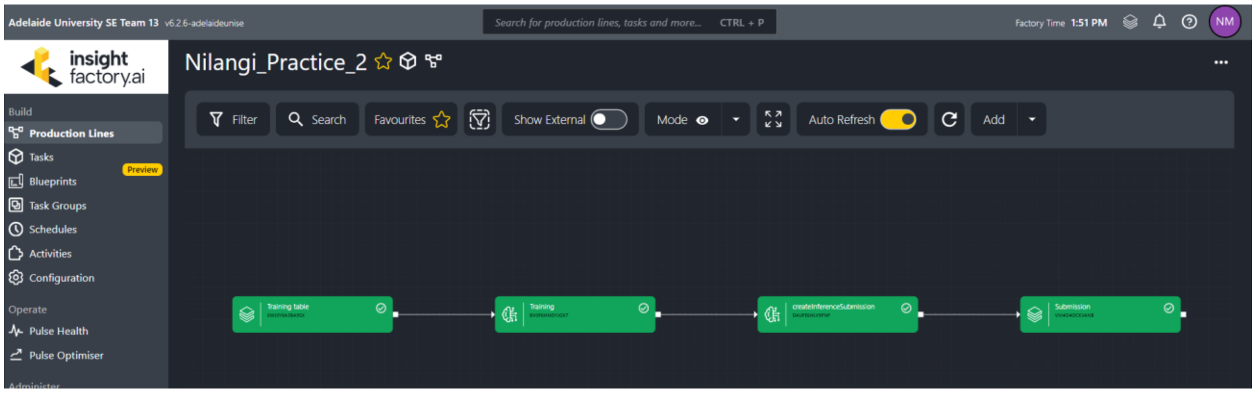
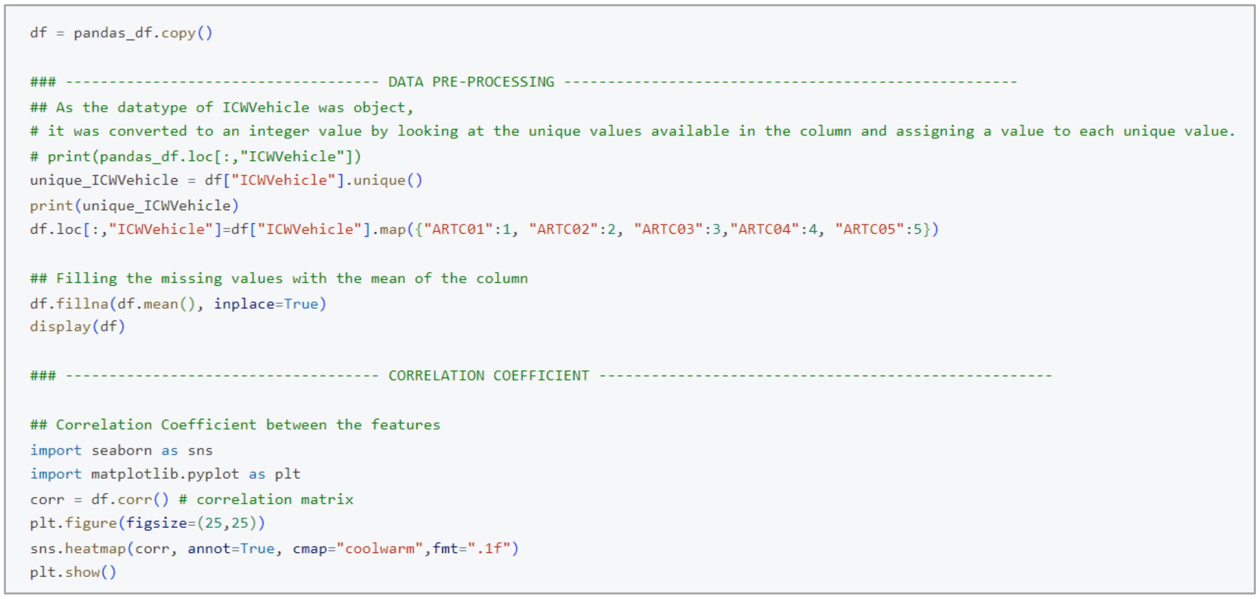


Figure 5. Screenshot of the Machine Learning Pipeline

Next, my contribution can be explained as follows:

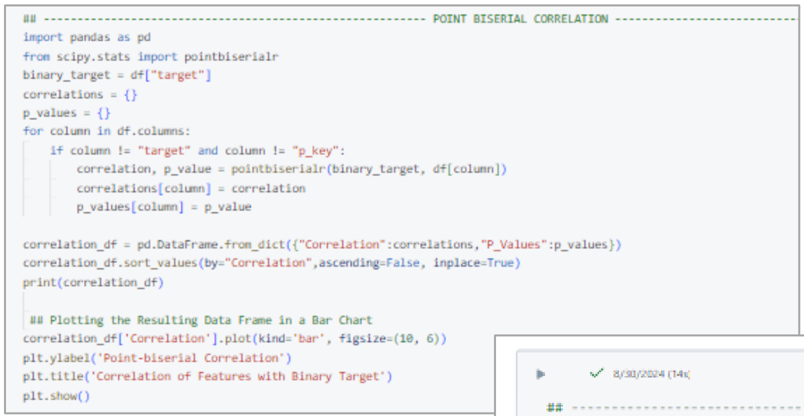
1. Firstly, I implemented the SQL query made by our group using the PySpark library where data from several tables were joined and selected and then converted to a Pandas Data Frame.
2. Using that data, I did data pre-processing as follows:
    a. Looked at the column data types of the data table made by the group and converted the "object" data type column "ICWVehicle" to "integer". Since I am focusing on feature selection, I did not focus on advanced encoding techniques. Essentially, I converted categorical variables into continuous data (later I learnt that the categorical variable "ICWVehicle" can be omitted and is not useful for the prediction.)
    b. Filled the missing values with the mean (average) of the column.
3. Next, I implemented the "Correlation Coefficient" method and generated a heatmap.



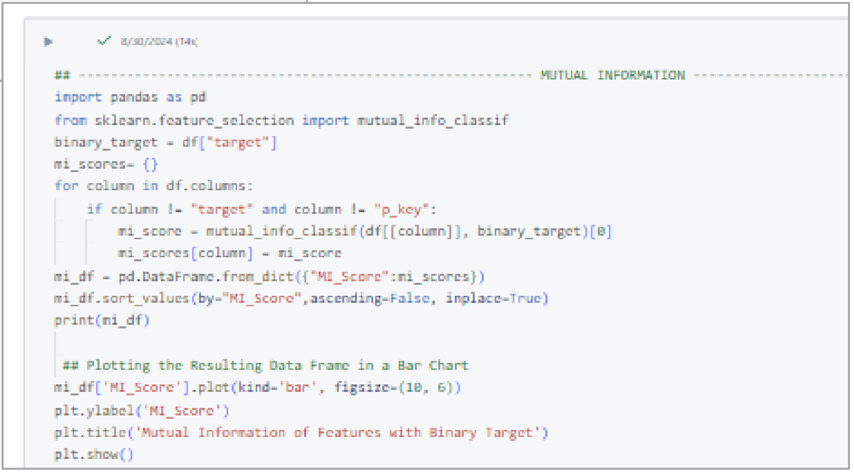
```
df = pandas_df.copy()

### ----------------------------------- DATA PRE-PROCESSING ---------------------------------------------------
## As the datatype of ICWVehicle was object,
# it was converted to an integer value by looking at the unique values available in the column and assigning a value to each unique value.
# print(pandas_df.loc[:,"ICWVehicle"])
unique_ICWVehicle = df["ICWVehicle"].unique()
print(unique_ICWVehicle)
df.loc[:,"ICWVehicle"]=df["ICWVehicle"].map({"ARTC01":1, "ARTC02":2, "ARTC03":3,"ARTC04":4, "ARTC05":5})

## Filling the missing values with the mean of the column
df.fillna(df.mean(), inplace=True)
display(df)

### ----------------------------------- CORRELATION COEFFICIENT ---------------------------------------------------

## Correlation Coefficient between the features
import seaborn as sns
import matplotlib.pyplot as plt
corr = df.corr() # correlation matrix
plt.figure(figsize=(25,25))
sns.heatmap(corr, annot=True, cmap="coolwarm",fmt=".1f")
plt.show()
```

Figure 6. Screenshot of Code for Data Preprocessing and Correlation Coefficient

4. It could be observed that the correlation coefficient of all the columns were 0 with the target, except for “tonnage” where the correlation coefficient was -0.1.
5. As it did not seem to provide a good indication of the correlations, next I implemented the “Point Biserial Correlation Method” which examines the relationship between a dichotomous variable and a metric variable. The implementation results of it are as follows:



```
## ----------------------------------------------------------- POINT BISERIAL CORRELATION -------------------------
import pandas as pd
from scipy.stats import pointbiserialr
binary_target = df["target"]
correlations = {}
p_values = {}
for column in df.columns:
    if column != "target" and column != "p_key":
        correlation, p_value = pointbiserialr(binary_target, df[column])
        correlations[column] = correlation
        p_values[column] = p_value

correlation_df = pd.DataFrame.from_dict({"Correlation":correlations,"P_Values":p_values})
correlation_df.sort_values(by="Correlation",ascending=False, inplace=True)
print(correlation_df)

 ## Plotting the Resulting Data Frame in a Bar Chart
correlation_df['Correlation'].plot(kind='bar', figsize=(10, 6))
plt.ylabel('Point-biserial Correlation')
plt.title('Correlation of Features with Binary Target')
plt.show()
```



```
## ----------------------------------------------------------- MUTUAL INFORMATION -------------------
import pandas as pd
from sklearn.feature_selection import mutual_info_classif
binary_target = df["target"]
mi_scores= {}
for column in df.columns:
    if column != "target" and column != "p_key":
        mi_score = mutual_info_classif(df[[column]], binary_target)[0]
        mi_scores[column] = mi_score
mi_df = pd.DataFrame.from_dict({"MI_Score":mi_scores})
mi_df.sort_values(by="MI_Score",ascending=False, inplace=True)
print(mi_df)

 ## Plotting the Resulting Data Frame in a Bar Chart
mi_df['MI_Score'].plot(kind='bar', figsize=(10, 6))
plt.ylabel('MI_Score')
plt.title('Mutual Information of Features with Binary Target')
plt.show()
```

Figure 7. Screenshots of Code for Point Biserial Correlation and Mutual Information

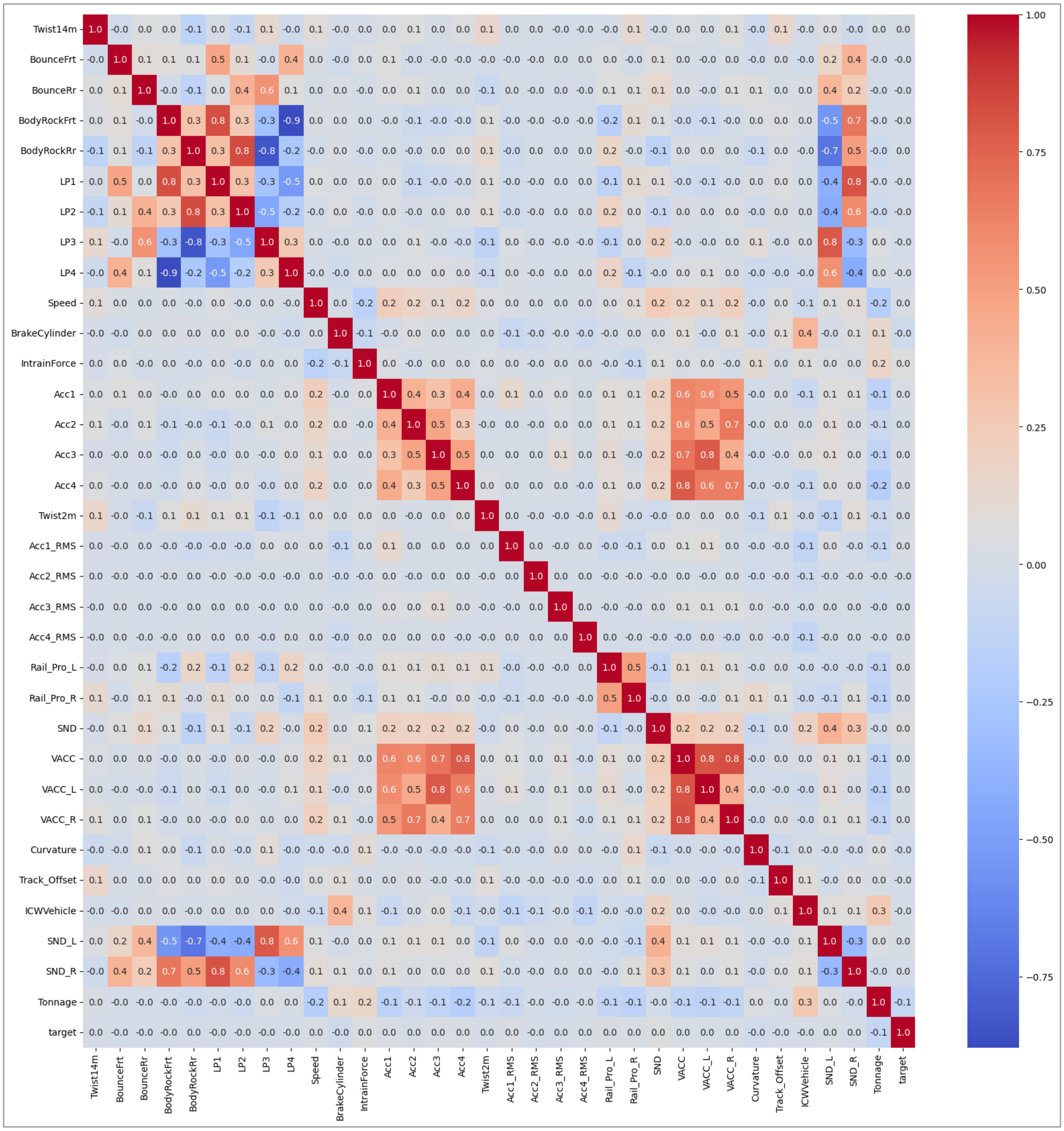


Figure 8. Heat Map that Displays the Correlation Coefficients

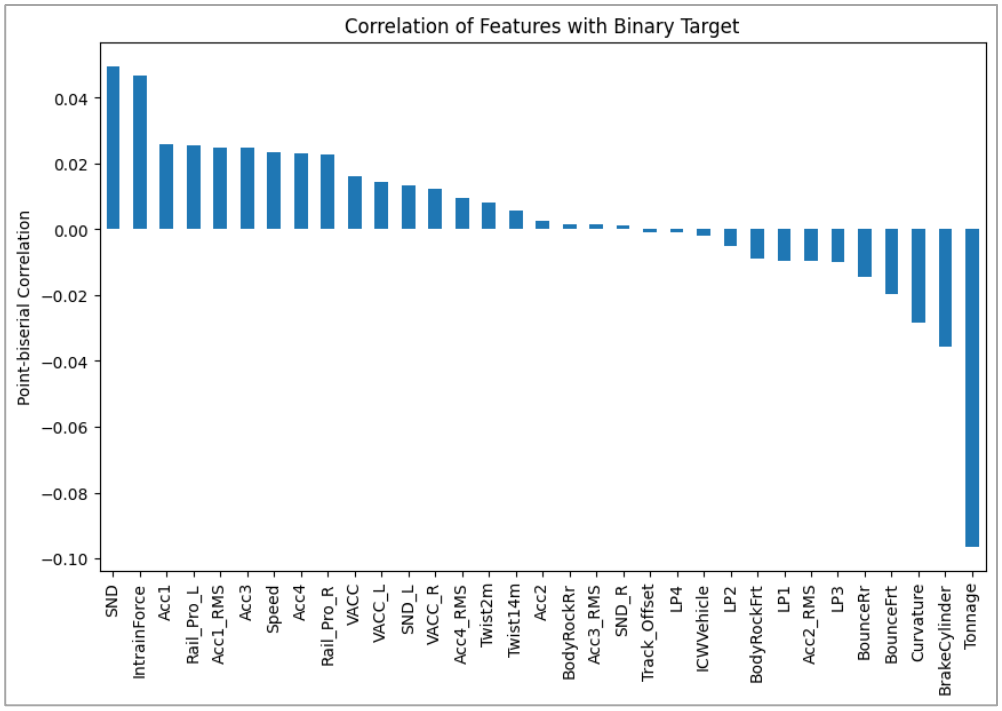


Figure 9. Results of Feature Selection – Correlation Coefficients

6. Accordingly, I implemented the Chi-Square test, which did not yield very good results as it is ideal for categorical variables and not very ideal for the given data set which is continuous data.

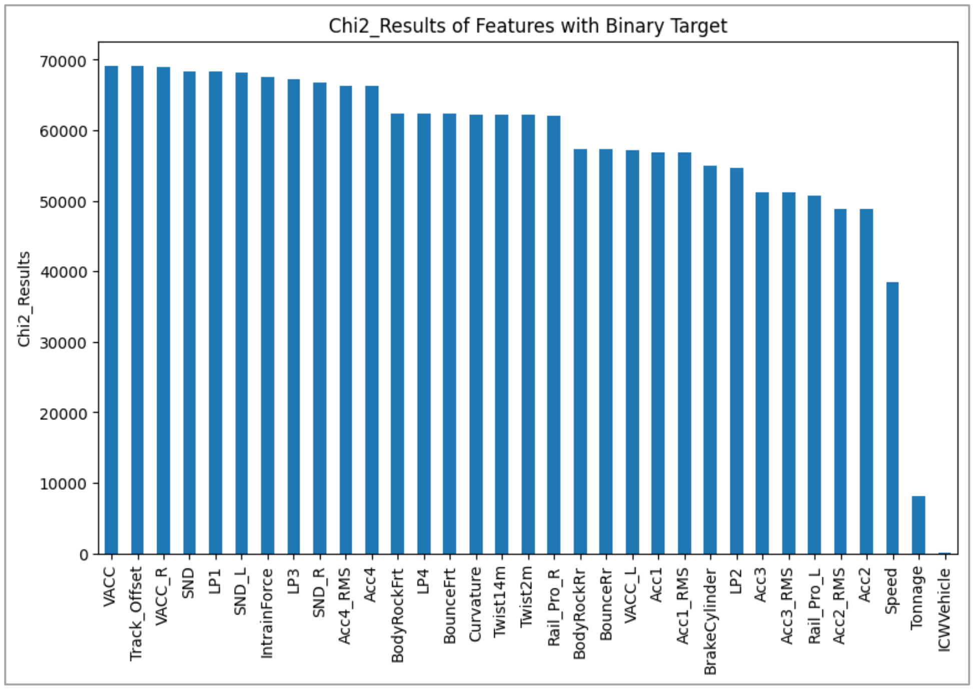


Figure 10. Results of Feature Selection – Chi Square Method

Next, I implemented the “Mutual Information” method and “Variance Threshold” method that yielded good results.

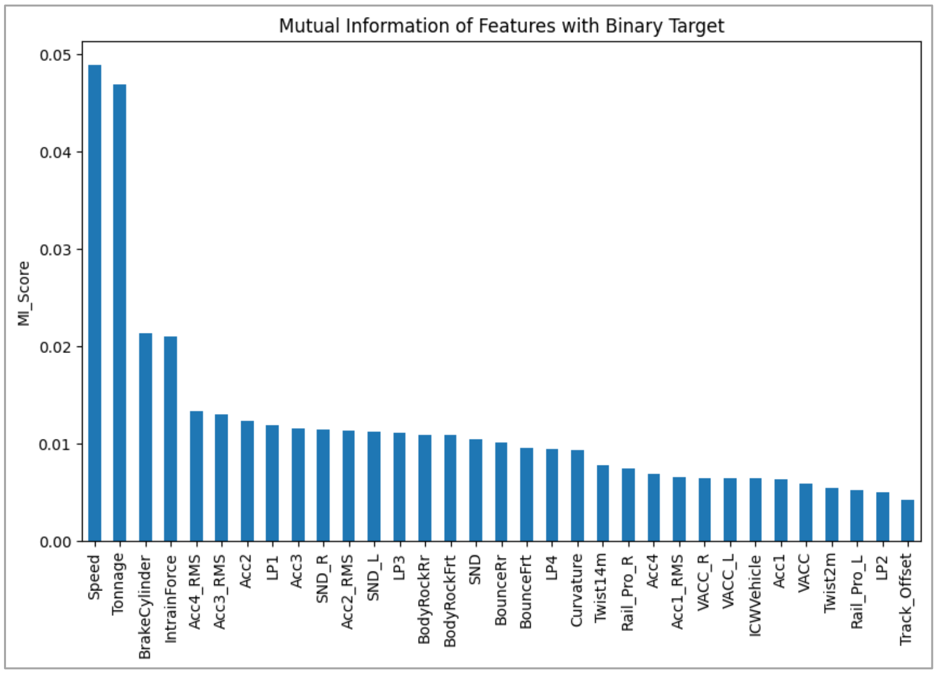


Figure 11. Results of Feature Selection – Mutual Information

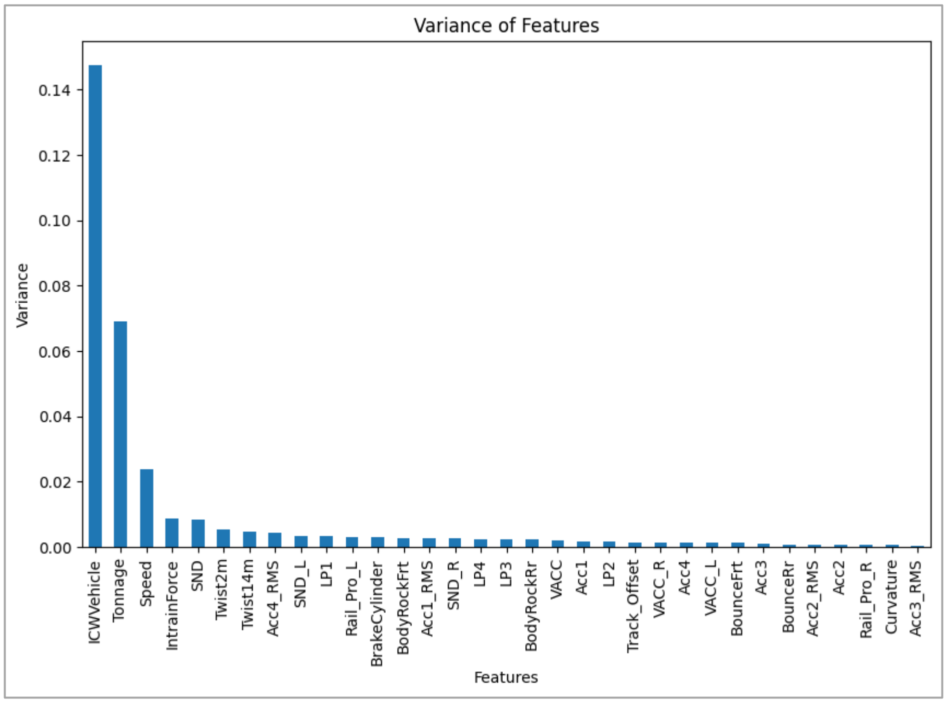
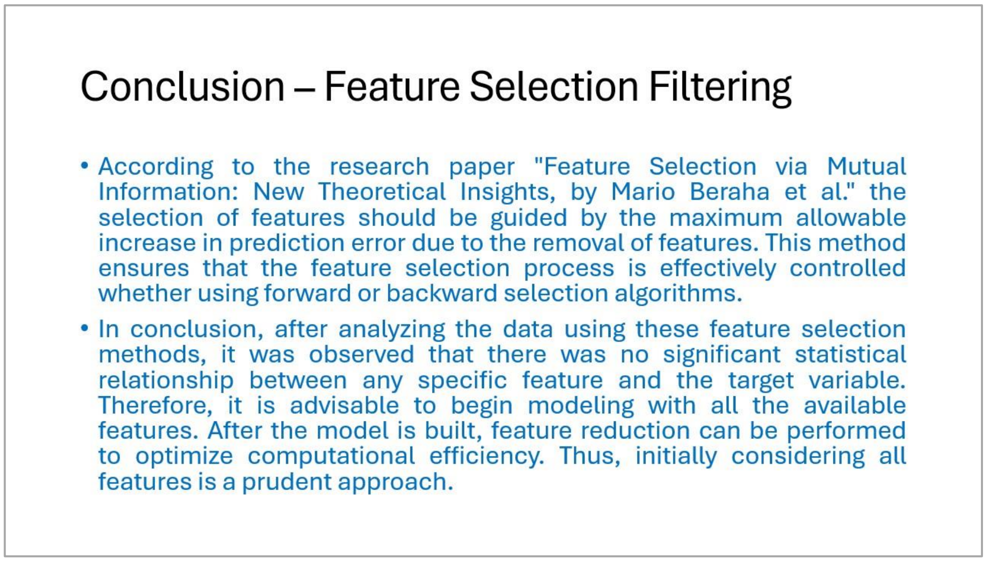


Figure 12. Results of Feature Selection – Variance Threshold

7. Finally, I presented all my findings at the sprint meeting as given in the and came up with the following conclusion.



Conclusion – Feature Selection Filtering

- According to the research paper "Feature Selection via Mutual Information: New Theoretical Insights, by Mario Beraha et al." the selection of features should be guided by the maximum allowable increase in prediction error due to the removal of features. This method ensures that the feature selection process is effectively controlled whether using forward or backward selection algorithms.
- In conclusion, after analyzing the data using these feature selection methods, it was observed that there was no significant statistical relationship between any specific feature and the target variable. Therefore, it is advisable to begin modeling with all the available features. After the model is built, feature reduction can be performed to optimize computational efficiency. Thus, initially considering all features is a prudent approach.

Figure 13. Conclusion I Derived from My Work of Feature Selection

6. Requirement Changes

During the second sprint, no major changes to the core project requirements were introduced. However, there were notable adjustments in the implementation process, particularly after initial experimentation with various feature selection methods. As we explored Correlation Coefficient, Point Biserial Correlation, Chi-Square Test, Mutual Information, and Variance Threshold, we concluded that the latter two methods provided the most promising results for feature selection. This shift in focus required re-evaluating our data preprocessing strategies and prioritizing techniques that yielded better results for the model's performance. Consequently, I had to adapt my project timeline by focusing more on refining these methods and applying them effectively to improve our model's ranking on the InsightFactory leaderboard.

Looking forward, potential requirement changes could stem from further findings in feature selection and model fine-tuning. As Mutual Information and Variance Threshold have shown to be beneficial, there may be a need to incorporate more advanced methods or additional feature engineering techniques to boost the F1 score beyond our current target of 55%. Additionally, feedback from future sprint meetings or leaderboard performance might prompt changes in our approach, such as using more sophisticated models or deeper analysis of the feature interactions, which could affect our current SQL data extraction methods or the machine learning pipeline structure.