REPORT ON THE OUTCOMES OF THE PROPOSED MODEL SOLUTION IN THE FAULT IMPACT ANALYSIS COMPETITION

Report Date: August 21, 2023

Prepared by: Julius Mwangi

INTRODUCTION

The following report presents a comprehensive analysis of the results obtained from the "Fault Impact Analysis: Towards Service-Oriented Network Operation & Maintenance" competition, conducted by the International Telecommunication Union (ITU). The report outlines the proposed model solution, the methodology employed, and emphasizes the outcomes of the model's performance, particularly focusing on the F1 score as the primary evaluation metric.

COMPETITION OVERVIEW

The "Fault Impact Analysis" competition organized by the ITU aimed to address the challenges in network operation and maintenance by leveraging advanced data analysis techniques. Contestants were required to develop models capable of predicting the impact of network faults on services, contributing to more effective and efficient maintenance strategies.

PROPOSED MODEL SOLUTION

The proposed model solution for the competition was a novel ensemble approach that combined two traditional machine learning algorithms. The ensemble comprised of catboost and lightgbm models. The ensemble approach was chosen to harness the strengths of various algorithms and provide a more robust and accurate prediction.

METHODOLOGY

Data Collection and Preprocessing A diverse dataset containing historical network records on KPIs and corresponding data rate was provided. This dataset was subjected to extensive preprocessing, including data cleaning, normalization, and feature engineering to enhance the model's ability to capture relevant patterns. For data cleaning.

Data Reading

- Prepared data by organizing provided training and validation datasets into corresponding "train" and "validation_clean" folders.
- Generated "train.csv" by reading and processing data from the "train" folder and "test.csv" from the "validation_clean" folder. This preprocessing was performed using the "1. DATA READING AND VISUALIZATIONS.ipynb" notebook.
- In a separate notebook named "2. MODEL SOLUTIONS.ipynb," I read the processed "train.csv" and "test.csv" files. The data was then combined, grouped by "NE ID" and "endTime," and sorted in ascending order. This set the stage for subsequent model operations.

Data Cleaning

- Converted the "endTime" column to a datetime object.
- Shifted previous hour KPIs to predict the current hour's values.
- The shifting caused the first rows of each ID to contain NAN values after shifting the previous day. These rows were dropped.

	NE ID	endTime	fault_duration	relation	predict_rows	data_roc	data_rate_t+1_trend	access_success_rate	resource_utilition_rate	TA	
37561	B0017-13_8	2023-02-17 05:00:00	0.0	0.000000	0	NaN	1	NaN	NaN	NaN	
37562	B0017-13_8	2023-02-17 06:00:00	0.0	0.000000	0	-0.460840	1	100.000000	1.054	3.035211	1.88
37563	B0017-13_8	2023-02-17 07:00:00	0.0	0.000000	0	-0.825678	1	100.000000	1.702	2.638554	5.71
37564	B0017-13_8	2023-02-17 08:00:00	46.0	0.654162	0	-0.657071	1	99.747899	14.521	2.963437	11.29
37565	B0017-13_8	2023-02-17 09:00:00	57.0	0.654162	0	-0.172830	1	99.829787	36.649	2.542375	12.06
37566	B0017-13_8	2023-02-17 10:00:00	2938.0	0.654162	0	0.528440	0	99.860807	58.506	2.412703	12.03
37567	B0017-13_8	2023-02-17 11:00:00	1089.0	0.654162	0	0.570455	0	99.844781	50.280	2.428769	12.14
37568	B0017-13_8	2023-02-17 12:00:00	25.0	0.654162	0	0.085498	0	99.942639	46.411	2.510660	12.26

```python
# Print the length of the combined DataFrame before the filtering process
before_filtering_length = len(combined_df)
print("Before:", before_filtering_length)

# Create a mask to identify rows where "NE ID" changes compared to the previous row
mask = combined_df['NE ID'] != combined_df['NE ID'].shift()

# Apply the mask to filter out rows where "NE ID" changes
combined_df = combined_df[~mask]

# Print the length of the combined DataFrame after the filtering process
after_filtering_length = len(combined_df)
print("After:", after_filtering_length)
```

```
Before: 957480
After: 948292
```

- Eliminated duplicated instances of fault duration after the initial occurrence

B0017-13_8											
NE ID	endTime	access_su	resource_	TA	bler	cqi	mcs	data_rate	fault_dura	relation	
B0017-13	2/17/2023 5:00	100	1.054	3.035211	1.884315	10.79236	2.912705	89.90918	0	0	
B0017-13	2/17/2023 6:00	100	1.702	2.638554	5.715271	11.27395	9.487271	48.47541	0	0	
B0017-13	2/17/2023 7:00	99.7479	14.521	2.963437	11.29741	8.910144	9.713915	8.45035	0	0	
B0017-13	2/17/2023 8:00	99.82979	36.649	2.542375	12.06214	7.125136	7.46731	2.897868	46	0.654162	
B0017-13	2/17/2023 9:00	99.86081	58.506	2.412703	12.03958	6.916188	7.704837	2.397028	57	0.654162	
B0017-13	2/17/2023 10:00	99.84478	50.28	2.428769	12.14568	7.321369	8.263317	3.663715	2938	0.654162	dropped
B0017-13	2/17/2023 11:00	99.94264	46.411	2.51066	12.26415	7.599301	8.270461	5.7537	1089	0.654162	
B0017-13	2/17/2023 12:00	90.73839	60.956	2.461951	12.09317	7.61272	9.045568	6.24563	25	0.654162	

```
combined_df[combined_df["NE ID"] == "B0017-13_8"]
```

```
Before: 1084072
After: 995544
```

	NE ID	endTime	fault_duration	relation	predict_rows	data_roc	data_rate_t+1_trend	access_success_rate	resource_utilition_rate	TA	
35624	B0017-13_8	2023-02-17 05:00:00	0.0	0.000000	0	NaN	1	NaN	NaN	NaN	
35625	B0017-13_8	2023-02-17 06:00:00	0.0	0.000000	0	-0.460840	1	100.000000	1.054	3.035211	1.8
35626	B0017-13_8	2023-02-17 07:00:00	0.0	0.000000	0	-0.825678	1	100.000000	1.702	2.638554	5.7
35627	B0017-13_8	2023-02-17 08:00:00	46.0	0.654162	0	-0.657071	1	99.747899	14.521	2.963437	11.2

- Filtered out rows with both 'data_rate' and 'fault_duration' equal to 0, addressing discrepancies where data rate is normally 0 only when a fault is present.

DROP ALL ROWS WITH 0 IN THE 'DATA_RATE' COLUMN AND 0 IN THE 'FAULT_DURATION' COLUMN ¶

```
print("Before:", len(combined_df))
combined_df = combined_df.loc[(combined_df['data_rate'] != 0) | (combined_df['fault_duration'] != 0)]
print("Before:", len(combined_df))
combined_df.head()
```

```
Before: 995544
Before: 957480
```

NE ID	endTime	access_success_rate	resource_utilition_rate	TA	bler	cqi	mcs	data_rate	fault_duration	relation
B0015-13	2/18/2023 1:00	100	16.554	2.258698092	9.48853348	8.465944098	9.783498834	26.92568568	0	0
B0015-13	2/18/2023 2:00	100	1.668	2.017857143	7.148969273	10.38826668	8.240137945	39.74733299	0	0
B0015-13	2/18/2023 3:00	100	1.648	1.965023847	6.082627119	11.0563597	6.477267842	42.55616385	0	0
B0015-13	2/18/2023 4:00	0	0.794	0	0	0	0	0	0	0
B0015-13	2/18/2023 5:00	0	0.792	0	0	0	0	0	0	0
B0015-13	2/18/2023 6:00	0	0.797	0	0	0	0	0	0	0
B0015-13	2/18/2023 7:00	100	3.039	2.044897959	6.307375345	10.63304194	5.333392633	31.96576879	0	0
B0015-13	2/18/2023 8:00	100	7.403	2.057271557	11.79037878	10.25236586	7.847369686	12.02940289	0	0
B0015-13	2/18/2023 9:00	100	7.121	2.031876138	7.448801932	9.846690935	7.65737449	34.62266106	0	0
B0015-13	2/18/2023 10:00	100	15.626	2.110306791	11.09223686	9.61310949	8.932995694	32.55145375	0	0
B0015-13	2/18/2023 11:00	100	16.432	2.213155445	8.799953935	10.00690104	10.68746538	70.06881045	0	0

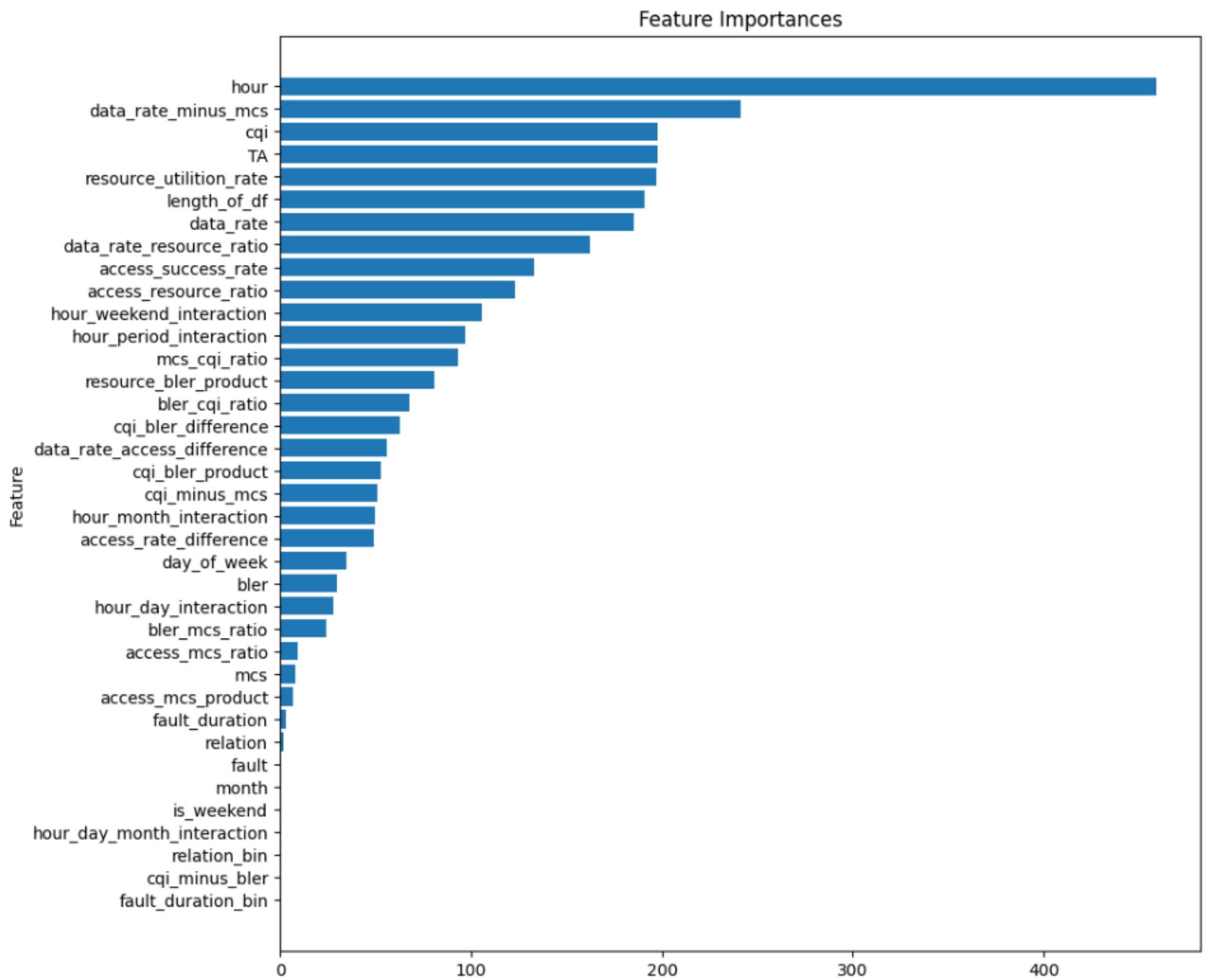
Feature Engineering

Apart from the KPIS additional features were engineered including the target variable.

- *Target Variable*: Was created based on this information by the competition host:

  "If data rate of previous non fault row is greater than data rate of the fault row , then the label is 1 else 0"

- *Time features such as hour and month were created. Actually fature hour proved to be very important as per the feature importance reports.*



- *Some interaction features created also proved very important as evidenced in the chart above*
- *Binned features relation and fault_duration.*

<u>*Model Training*</u>

- Dropped columns "predict_rows", 'data_roc', 'period_of_day','data_roc_bin' before training. Also dropped duplicates in the train before proceeding to training the models

- Modelling Entailed training individual constituent models, namely CatBoost and LightGBM on the dataset. The individual models were basic with no parameters but default.
- I also trained on all train data without splitting to X_train and X_test.
- I did not predict the binary classes but the probabilities instead. The probabilities were then rounded off to integers.
- To ensure accurate prediction of the 'ones', a confidence level of around 0.485 was incorporated. This decision was rooted in maintaining alignment with the distribution observed in the training dataset, where approximately 52% of the data were 'ones' and the remaining 48% were 'zeros'.

```
combined_df[((combined_df["fault"]==1)&(combined_df["predict_rows"]==0))]["data_rate_t+1_trend"].value_counts()
```

```
1    3789
0    3435
Name: data_rate_t+1_trend, dtype: int64
```

Ensemble Integration

- The approach used for creating the ensemble predictions involved combining the outputs of individual models. This process, aimed to leverage the diverse strengths of these models and arrive at a more robust and accurate final prediction.

RESULTS AND DISCUSSION

The proposed ensemble model demonstrated exceptional performance in the competition, leading to the eventual victory at position 2. F1 Score The harmonic mean of precision and recall, known as the F1 score, stood at around 0.748. This balanced metric provides a comprehensive assessment of the model's overall performance, considering both false positives and false negatives.

FUTURE WORK AND INTERNSHIP GOALS:

The successful outcome of the proposed model solution in the ITU's Fault Impact Analysis competition serves as a foundation for future endeavors in advancing the field of service-oriented network operation and maintenance. I look forward for the internship. As an intern, there are several exciting avenues for further research and development that can build upon the accomplishments of the current model:

1. Hyperparameter tuning

Perform hyperparameter tuning using grid search and cross-validation techniques to optimize each model's performance. This was never done for this competition submissions due to time factor. I resorted to using the default parameters

2. Enhanced Feature Engineering:

During the internship, a deeper exploration of feature engineering could be undertaken. This involves identifying and incorporating additional relevant features from the provided dataset to potentially enhance the model's predictive capabilities. The integration of historical patterns, contextual data, or network topology information might provide valuable insights for improved fault impact analysis.

3. Multi-Modal Data Fusion:

Considering the complexity of network fault analysis, integrating multi-modal data sources could prove beneficial. Combining RAN KPI data with other types of network-related data, such as maintenance logs, weather conditions, or geographical information, could offer a more holistic view of network behavior and contribute to more accurate fault impact predictions.

3. Transfer Learning and Generalization:

To broaden the applicability of the model, exploring transfer learning techniques might be beneficial. Pre-training the model on a larger, more diverse dataset e.g. those based on time series problem could enhance its ability to generalize across different network scenarios and fault types, ensuring robust performance even in previously unseen conditions.

4. Real-time and Adaptive Analysis:

Adapting the model for real-time analysis could lead to proactive fault management and quicker response times. Investigating methods to incorporate streaming data and designing algorithms that dynamically adjust to changing network conditions could be a promising area of research during the internship.

CONCLUSION

In conclusion, the proposed ensemble model solution for the "Fault Impact Analysis" competition proved to be highly effective in predicting service impacts resulting from network faults. The exceptional F1 score achieved by the model reflects its ability to strike a balance between precision and recall