Fault Impact Analysis

Towards Service-Oriented Network Operation & Maintenance by ITU

Predict an NE's average data rate change when a fault occurs

Krishna Priya
krishnapriyakejriwal@gmail.com

Abstract - The efficient management of faults in the telecom Radio Access Network (RAN) is crucial for maintaining stable and reliable network services. Traditional fault management methods lack the ability to assess the impact of faults on key performance indicators (KPIs), hindering the optimal allocation of Operation & Maintenance (O&M) resources. In response, this competition aimed to leverage machine learning to predict the impact of faults on RAN KPIs, thereby enabling service-centric fault management. This report presents my winning solution, which is based on feature engineering and an ensemble of gradient boosting models. My approach addresses the challenges of uncertainty and complexity in RAN fault analysis and demonstrates the potential for ML-driven advancements in fault management.

1. Introduction

In the realm of telecom Operation & Maintenance (O&M), the primary objective is to ensure the stability and reliability of network services, with a significant focus on fault management within the Radio Access Network (RAN). Fault management encompasses the monitoring, analysis, diagnosis, and repair of network faults, with fault analysis being a pivotal component. Conventionally, network experts have relied on rule-based methods, which often prioritize faults based on predefined criteria such as fault duration or fault categories.

However, this conventional approach falls short of explicitly assessing the impact of faults on critical network KPIs, including coverage and data rate. Consequently, the reliability of network services remains unquantified, and O&M resources cannot be optimally scheduled. The challenge is exacerbated by the complexity of modern RANs, where diverse network configurations, multiple layers of coverage (e.g., 4G and 5G coexistence), and various fault types create a dynamic landscape.

To address these limitations and empower the transformation of O&M from equipment-centric to service-centric, this competition tasks participants with developing machine learning-based models capable of predicting how faults impact the average data rate of Network Elements (NEs). This prediction is grounded in network topology and historical data, bridging the gap between fault occurrence and network KPIs.

In this report, I present my solution to this problem, highlighting the challenges posed by uncertainty, non-stationarity and complex networking mechanisms. Our solution demonstrates the potential of machine learning to revolutionize fault management and ultimately contribute to the autonomy of next-generation communication systems.

2. Dataset

The dataset provided for this competition is a comprehensive collection of RAN Key Performance Indicator (KPI) data, drawn from over 100 4G and 5G Network Elements (NEs). It encompasses hourly records of six RAN KPIs related to data rate, including Access Success Rate, Resource Utilization Rate, TA (Time Advanced), BLER (Block Error Rate), CQI (Channel Quality Indicator), and MCS (Modulation and Coding Scheme).

The provided data had 7256 train NEs and 1932 test NEs. Each NE contains essential information, including:

- NE ID: A unique identifier for each NE.
- Hourly timestamp: Precise timestamps corresponding to hourly observations.
- RAN KPIs: Six distinct KPIs providing insights into the NE's performance.
- Data rate: Data rate measurement for each hour.
- Fault duration: Duration of faults in seconds occurring within the hour.
- "Distance" to the fault: An indicator of fault proximity, ranging from 0 (no fault) to 1 (fault occurred at the NE) and values in between to denote adjacency between NEs.

The task involves predicting the status of data rate in the first hour following the occurrence of a fault, where the fault duration is greater than 0. The status is binary, with '1' indicating a decrease in data rate compared to the hour preceding the fault and '0' indicating no decrease.

In the subsequent sections of this report, we detail our approach, feature engineering techniques, model selection, and performance evaluation, highlighting the steps taken to achieve the best possible F1 score in predicting fault impacts.

2.1 Data Pre-Processing

Train and test data in different folders with csv chunks were concatenated to create train.csv and test.csv

2.2 EDA and Data Processing

Upon filtering the data for a single NE ID, I observed that it was broken into multiple IDs where based on ‘n’ past data where no fault, once we observed fault, we had to predict data rate change. After the particular fault hour, again for some hours there was no fault and again ‘n’ faults. This made me believe that past ‘n’ data of an NE ID is not only present in that ID but we should get all data for an NE ID. Thus I sorted the whole data by NE ID and endtime in ascending order to utilize the complete data and from now on, ID column would be just used for submission and no analysis.

Second thing I observed was that it was not necessary that we have datapoint for just the hour before the fault. We also had to predict for fault time when the last data point was more than 9 hours ago. This gave me an intuition that along with last values, how far in time was the last value was also important.

Third thing that I observed was that we did not have the 6 KPIs for data points where fault occurred in the test data, where as we had these values in the training data so I deleted these data from training as well to avoid overfitting.

2.3 Feature Engineering

Feature Set 1:

- Lagged value of 1,2,3 for all KPIs+data_rate
- Difference of current sample’s endtime to endtime of lagged value
- Lagged value of 24,48,72 to get an idea of that particular hour’s values of KPI+data_rate.

- Extracting month,day of month, hour, day of week from current samples endtime.
- Sin and Cos transformation of hour column to give model an idea of hour 0 and hour 23 are the closest one and not the farthest one. This would help the model split the leaf/node on this feature easier and get better gini impurity.

Feature Set 2:

- Group by 'NE ID' and 'hour' to calculate aggregate statistics of 6 RAN KPIs and data rate. Statistics included:
    - Mean
    - Std
    - Min
    - Max
    - Skew

Feature Set 3:

- Group by 'NE ID' and 'hour-1' to calculate aggregate statistics of 6 RAN KPIs and data rate.

2.4 Feature Selection

Removed all columns with <=1 unique values in column.

Any column with null percent more than 75% were set to be removed although there were no such columns

3. Modelling

3.1 Model Selection

Since we had lag features, there were many null values in the feature set so I wanted to try models which handle null values internally by splitting them in a different leaf/node and calculating their impurity.
These models are: lightgbm, catboost, xgboost, histgradientboosting

3.2 Models

Model set 1:

- Lightgbm: Trained on Feature Set 1
- catboost: Trained on Feature Set 1
- histgradientboost: Trained on Feature Set 1
- xgboost: Trained on Feature Set 1

Ensemble1: harmonic mean of 4 predictions of above 4 models

Model set 2:
Introduced month, day, hour and weekday number as categorical features in all the 4 models.

- Lightgbm: Trained on Feature Set 1 + categorical features
- catboost: Trained on Feature Set 1 + categorical features
- histgradientboost: Trained on Feature Set 1 + categorical features
- xgboost: Trained on Feature Set 1 + categorical features

Ensemble2: harmonic mean of 4 predictions of above 4 models

Model set 3:

- Lightgbm: Trained on Feature Set 1+2
- catboost: Trained on Feature Set 1+2
- histgradientboost: Trained on Feature Set 1+2
- xgboost: Trained on Feature Set 1+2

Ensemble3: harmonic mean of 4 predictions of above 4 models

Model set 4:

- Lightgbm: Trained on Feature Set 1+2+3
- catboost: Trained on Feature Set 1+2+3
- histgradientboost: Trained on Feature Set 1+2+3
- xgboost: Trained on Feature Set 1+2+3

Ensemble4: harmonic mean of 4 predictions of above 4 models

3.2 Model Setup and hyperparameters:

- Cross Validation: 10 fold StratifiedKfold for all 16 models above.
- Early stopping for overfitting detection was employed on all models
- Feature fraction was tuned for best CV score as I didnot want the models to overfit on any feature.
- Introducing a small L1 regularization term to tackle overfitting again
- The most important parameter to tune was **scale_pos_weight** for the best CV score. This parameter is important for imbalanced problems or problems where F1 score is the metric. Basically it adds more weight to the minority class in model's binary logistic function.
- Evaluation metric for each of the model was set to be f1 score:
    - For lightgbm and xgboost since f1 score is not available as evaluation metric by default, a custom evaluation metric was coded and passed as callback
- No manual thresholding was applied to convert probilities to binary values as this generally overfits the leaderboard and it should be taken care by scale_pos_weight in training itself.

3.2 Final Submission

The final submission for the competition was a harmonic mean of 4 ensembles. However some of the individual models have scored much better than the ensemble model on both local CV and private leaderboard and could be selected for future work. The individual model benchmarks is published in the benchmark section.

4. Conclusion

In this competition, we developed a robust solution for predicting the impact of faults on Network Elements' average data rates in the Radio Access Network. By meticulously preprocessing data, engineering relevant features, and employing a combination of gradient boosting models, we successfully addressed the inherent uncertainties and complexities in fault management. Our approach, which included ensemble techniques, yielded competitive results and demonstrated the potential of machine learning in transforming network operation and maintenance towards a service-centric paradigm. This competition underscores the broader implications of AI and ML in tackling intricate real-world challenges, with the potential to revolutionize fault management in telecommunications and beyond.

4. Future work and Analysis

If we can introduce latitude/longitude of NE IDs we could better predict the behavior of an NE ID using it's neighbours past trends. I tried finding the neighbours but did not get clean neighbours just from the past data.

5. Model Benchmarks

Model	Local CV (f1 score)	Private LB (f1 score)
Lightgbm (1)	0.7293	0.7544
Catboost (1)	0.7228	0.7375
Histgradientboost (1)	0.7184	0.7444
Xgboost (1)	0.7282	0.7462
Lightgbm (2)	0.7300	0.7472
Catboost (2)	0.7208	0.7384
Histgradientboost (2)	0.7180	0.7528
Xgboost (2)	0.7290	0.7379
Lightgbm (3)	0.7379	0.7598
Catboost (3)	0.7314	0.7566
Histgradientboost (3)	0.7248	0.7554
Xgboost (3)	0.7344	0.7587
Lightgbm (4)	0.7354	0.7661
Catboost (4)	0.7286	0.7591
Histgradientboost (4)	0.7259	0.7596
Xgboost (4)	0.7323	0.7639
Hmean (ens1,ens2,ens3,ens4)		0.7545

We see that single lightgbm model with feature set 1+2+3 performed 2nd best on local cross validation and best on private leaderboard. Although I didnot choose this for the final submission and went ahead with an ensemble to be safe for the sake of the competition, we should choose this single model for any analysis in the future. Even if you remove some features, this is still the highest quality model.

Feature importance for lightgbm(4) single model:

feature	importance
data_rate_T-1	441.3
data_rate_mean	194.5
data_rate_min	96.8
data_rate_T-2	61.6
data_rate_T-3	59.7
data_rate_T-72	56.7
data_rate_skew	56.3
fault_duration	55.1
hour	49.5
data_rate_T-24	44.4
mcs_T-1	44.0
TA_skew_T-1	43.8
data_rate_max	42.8
bler_T-1	41.3
mcs_T-48	40.9
bler_T-3	40.5
data_rate_skew_T-1	37.0
resource_utilition_rate_skew	36.9
cqi_skew	36.5
bler_T-2	34.9
resource_utilition_rate_T-2	33.7
resource_utilition_rate_T-3	33.2
data_rate_std_T-1	30.4

6. References

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., … Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30, 3146–3154.

Prokhorenkova, L. O., Gusev, G., Vorobev, A., Dorogush, A. V. & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features.. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi & R. Garnett (eds.), *NeurIPS* (p./pp. 6639-6649), .

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189–1232.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.2939785