2024-Spring Qualification Project

Security Risk Evaluation via Log Analysis

Chan Gyu Lee (2023-29914)

ViBA
Visualization and Business Analytics

Problem Definition

01 Identify Patterns in Log Data

02 Classify Security Risk

03 Prevent Potential Cyber Threats

Exploratory Data Analysis

y	X
Level	**full_log**
level 0	log 0
level 0	log 1
level 0	log 2
level 0	log 3
level 0	log 4
. . .	. . .
level 3	log 472940
level 3	log 472941
level 3	log 472942

01 Characters

Examples:
{“Type”: [“error”], "tags":["warning","stats-collection"], "message":"No Living connections"}

02 Numbers

Examples:
{“msg=audit”: 16118892144.855:247124, ”exit": 3, ”sshd":6677, “rhost” = 44.222.184.35}

03 Special Characters

Examples:
{\(\)./, \<\> , :=\(\)./ ,\<\>, ?}

EDA

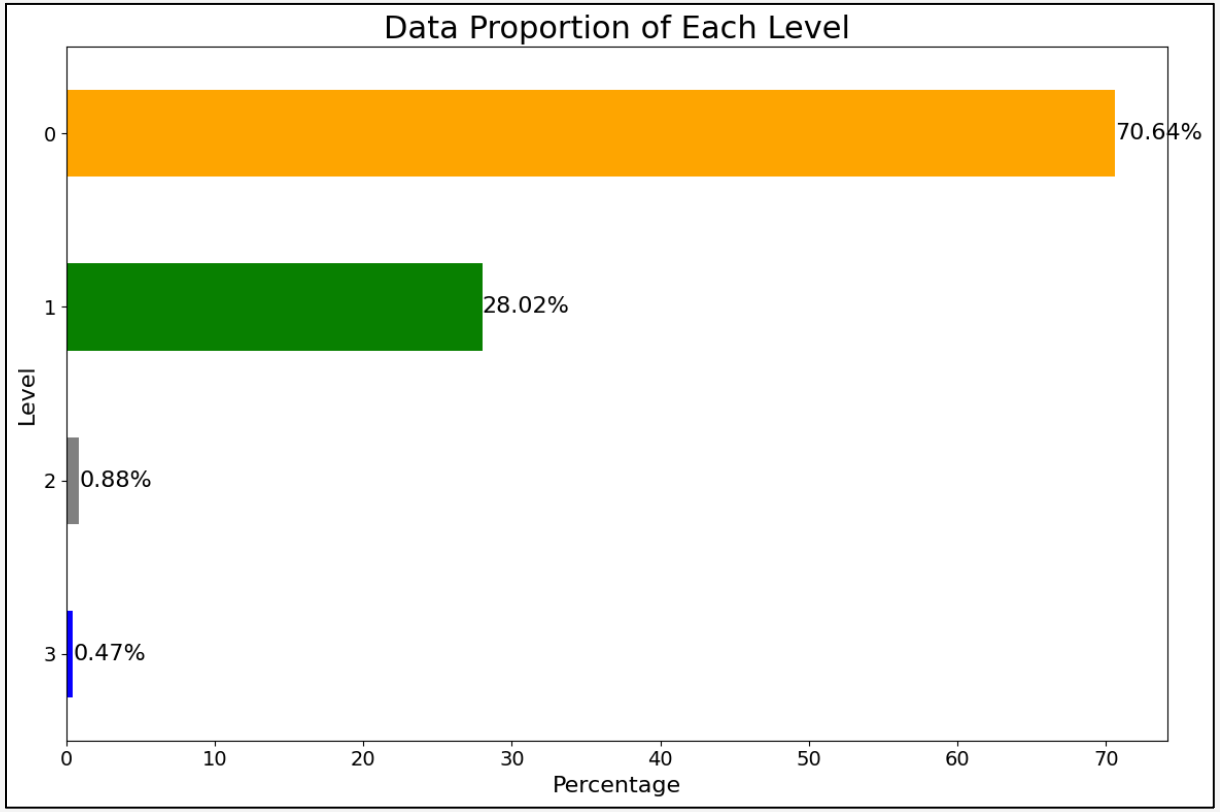
Imbalance of Level (y)

KEY Finding 01 – Level Imbalance

Explanation: The levels are not distributed equally. Most observations fall into the levels of 0 and 1.

- Data Count by Level

Level	Count
Level = 0	334065
Level = 1	132517
Level = 2	4141
Level = 3	2219



EDA

Top 10 most common words

Level 0:

Word	Frequency
error	741440
elasticsearch	449353
localhost	412213
kibana	406857
no	405525
connections	390450
living	376626
at	354785
js	354705
message	274852

Level 1:

Word	Frequency
audit	502668
type	501812
msg	501796
syscall	233676
proctitle	232372
cwd	232188
s	214346
bin	199805
systemu	185124

Level 2:

Word	Frequency
localhost	3490
jan	3417
nist	3246
failed	2927
the	2603
http	2463
service	2225
systemd	2210
unit	2210
entered	2210

Level 3:

Word	Frequency
tcp	35239
listen	12156
udp	12028
established	2820
java	2759
nist	1962
the	1907
dnsmasq	1863
sshd	1732
is	1585

EDA

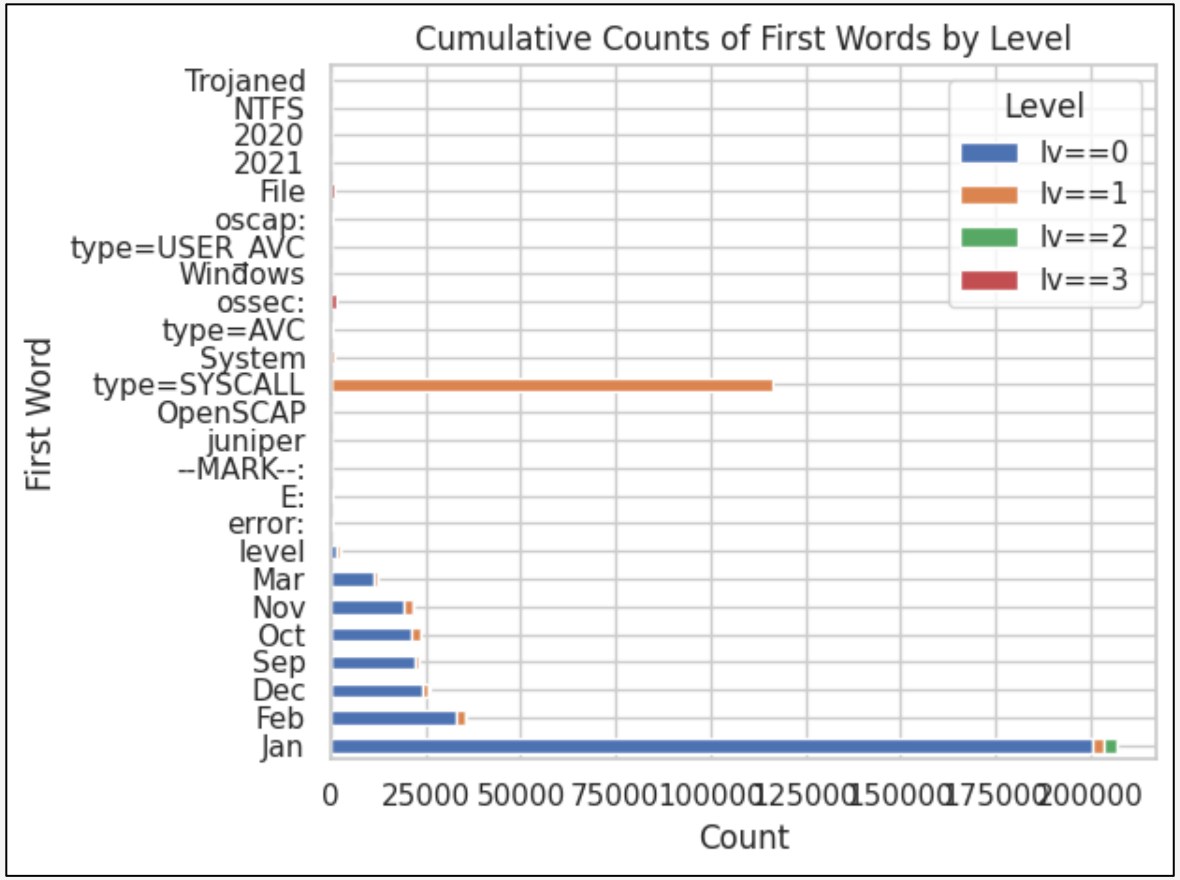
Cumulative Counts of Words

KEY Finding 02 - Common Words

1. Commonly used words are different in each level.
2. Words (texts) in log data may possibly explain the levels.
- ex) months related first words imply that the data comes from the level 0.

Level 0 Example)

1. Sep 24 10:02:22 localhost kibana: {…}
2. Feb 8 16:21:00 localhost logstash : {…}
3. Jan 13 01:50:40 localhost kibana: {…}
4. Jan 4 10:18:31 localhost kibana: {…}



EDA

Possible Outliers

KEY Finding 03 – Label Confusion

1. Exists identical logs that have different levels.
2. Without the existence of timeseries feature, hard to interpret.
- ex) months related first words imply that the data comes from the level 0.

Examples of Such Cases)

level	full_log
Level = 0	level : 5, log : No mode specified for interfa...
Level = 1	level : 5, log : No mode specified for interfa...
Level = 1	level : 5, log : No mode specified for interfa...

Model Development

이곳에 내용을 입력하세요

- Data Preprocessing
- Learning Method Selection
- Evaluation Metrics
- Experiment
- Result Analysis

Data Preprocessing

Consideration

01

Consideration 01

Data Imbalance

Need to control data imbalance issue as the original dataset is vulnerable to biases in its setting.

02

Consideration 02

Frequency of Words

The format of data and frequencies of words may capture the level feature.

Possible Approaches

Sampling Methods

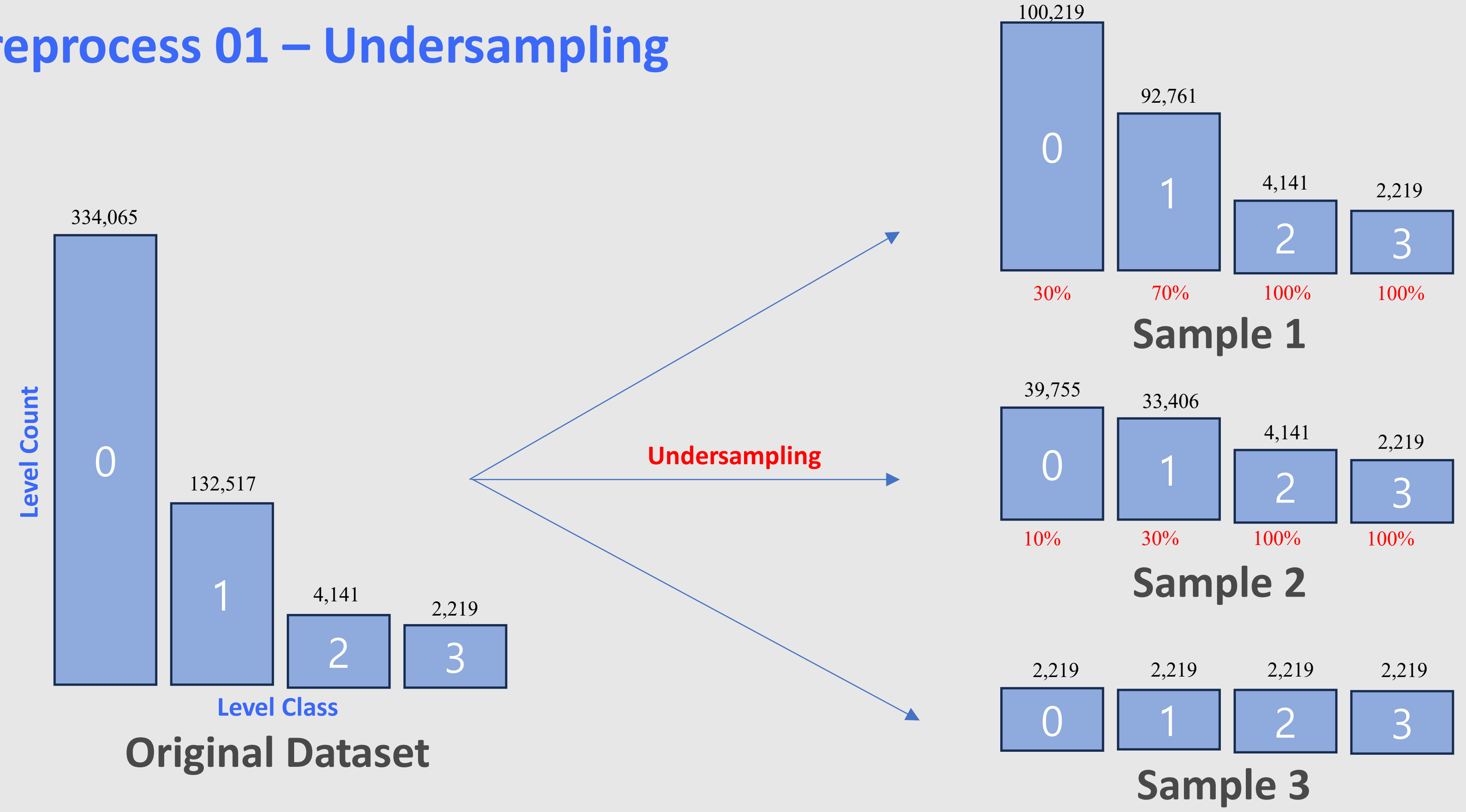
Oversampling Vs. Undersampling

Feature Extraction using Text Vectorizing Method

Capture the frequencies of tokens.

Data Preprocessing

Preprocess 01 – Undersampling



Level	Count
Level = 0	100,219
Level = 1	92,761
Level = 2	4.141
Level = 3	2.219

Level	Count
Level = 0	100,219
Level = 1	92,761
Level = 2	4.141
Level = 3	2.219

Level	Count
Level = 0	100,219
Level = 1	92,761
Level = 2	4.141
Level = 3	2.219

Data Preprocessing

Preprocess 02 – Mask using Tokens

1) Convert numerical data into <num> token.
2) Remove special characters.

Preprocess 03 - Vectorize

Preprocessed log texts -> count frequencies -> vectorize

Raw log format

Localhost logstash: 18862 Java::ComMysqlJdbcExceptionsJdbc4::Communications

Masking

Localhost logstash: <NUM> Java ComMysqlJdbcExceptionsJdbc<NUM> Communications

Vectorize

Learning Method Selection

Logistic Regression

Classification Task

Tree-Based Methods

Approach	Non-Tree	Tree-Based	
Method	**Logistic Regression**	**Random Forest**	**XGBoost**
Distinction	- Simplicity and Interpretability - Not suitable for multi-classes - Not preferable when classes are well-separated.	Create multiple decision trees and aggregate their output to enhance robustness of the model. - Decorrelation - Increase accuracy	Boosting ensemble technique that sequentially combine multiple small trees. - Parallel Learning - Large computation amount

Evaluation Metrics

Task: Cyber Security

Metric 01: Macro-Recall

Average recall scores across all classes.
- For this particular task to enhance cyber security, identifying hazardous threat is the most critical goal.

Formula:

$$Macro\_Recall = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i}$$

- C is the number of classes

Metric 02: Macro-F1 Score

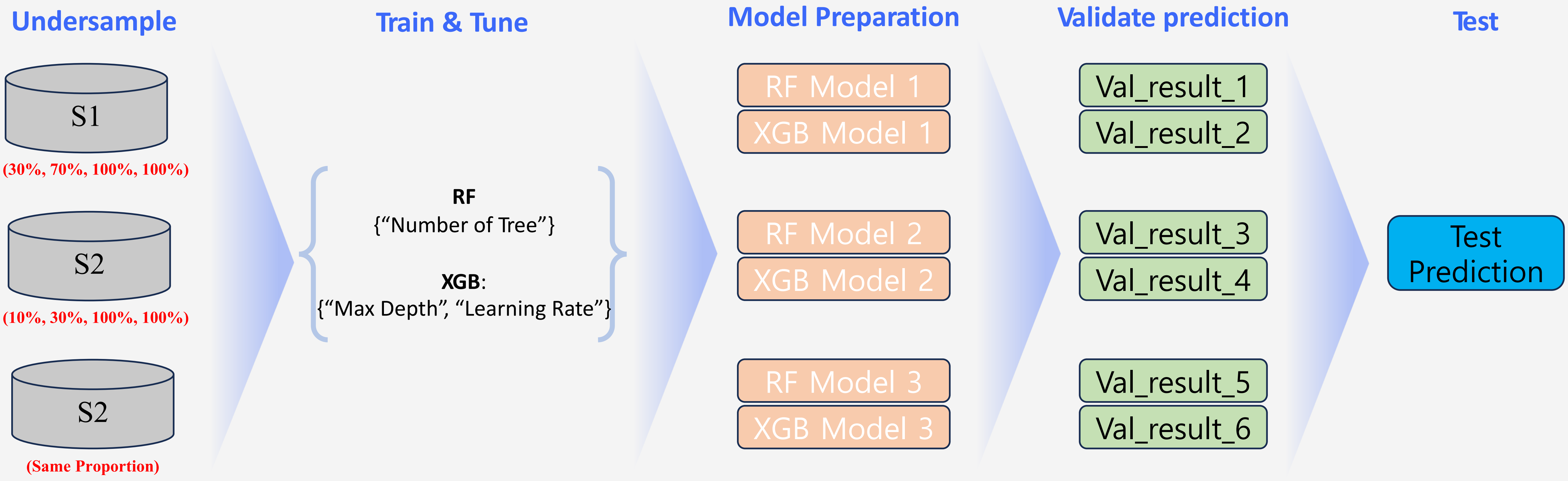
Considers the balance between precision and recall.
- As the levels of given data are highly imbalanced, macro-F1 score is computed as well.

Formula:

$$Macro\_F1\ Score = \frac{1}{C}\sum_{i=1}^{C} F1_i$$

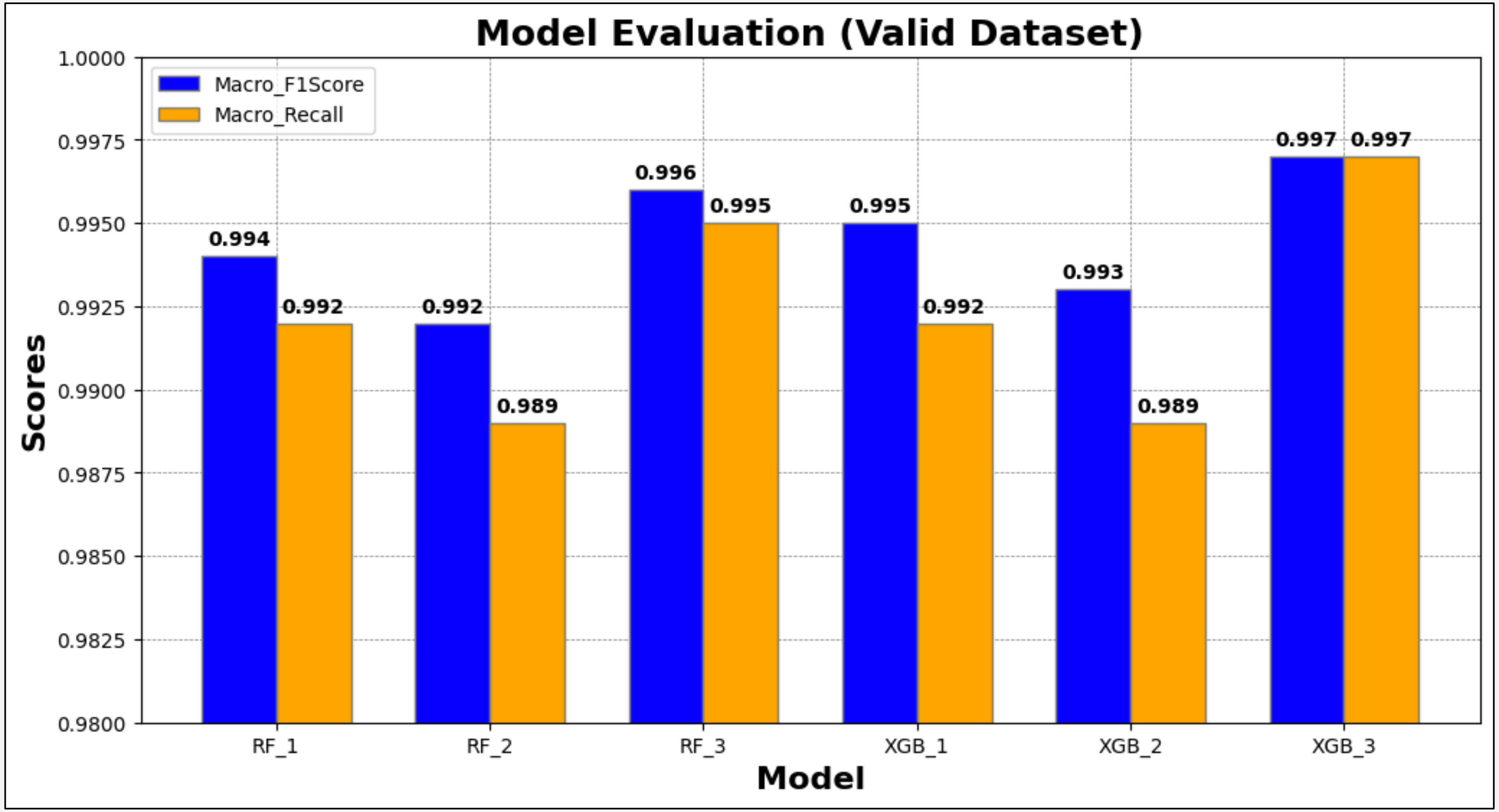
- $F1 = 2\times\frac{Precision \times Recall}{Precision + Recall}$

Overview of Experiment



Result Error Analysis

Model Performance



Result Error Analysis

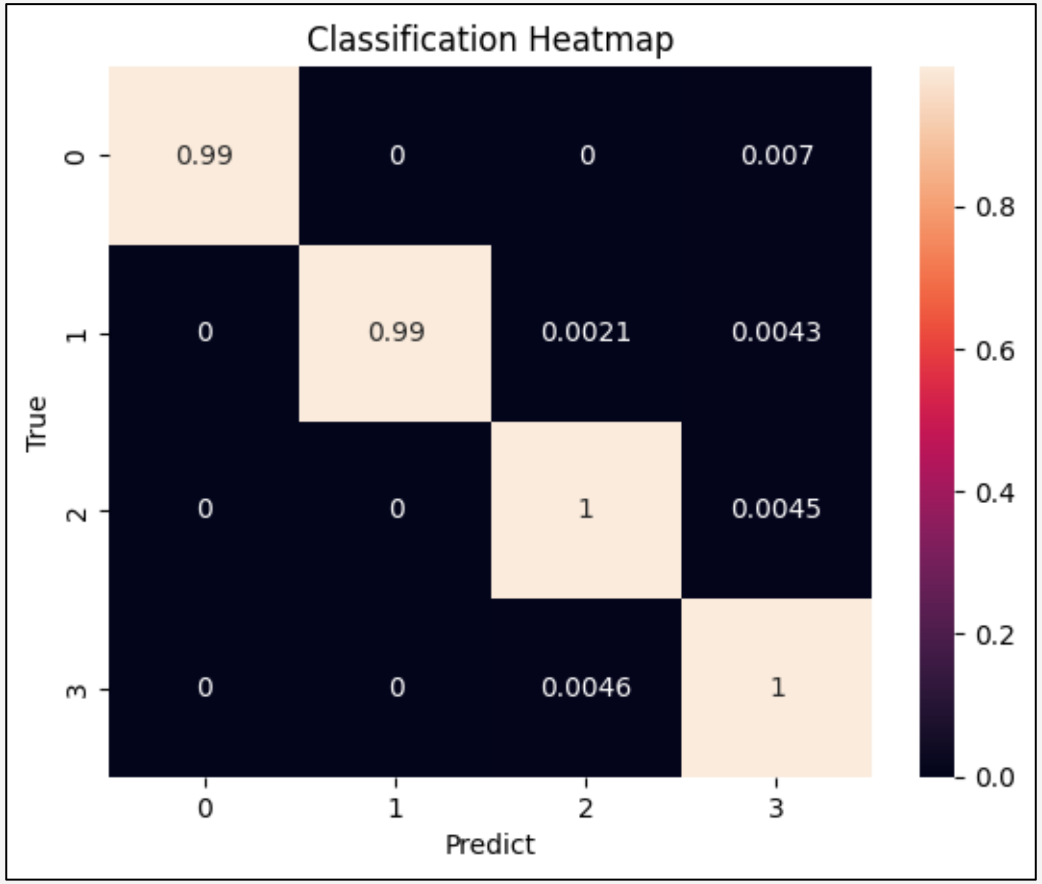
Model Performance

Comparisons to non-tree methods

The suggested tree model surpasses the overall performance.

Result Table

Method	Macro_F1	Macro_Recall
Logistic Reg.	0.9937	0.9938
Naïve Bayes	0.9770	0.9774
XGB_3	**0.9943**	**0.9943**



Discussion

Limitation and Future Work

Limitation	Future Work
1. Handling outliers (identical log, different labels)	1. Add inference time for model selection criteria
2. Absence of time series feature	2. Acquire more features for robustness
3. Lack of prior knowledge	3. Consider online learning algorithms

Q&A

Appendix

Model Optimization: Grid Search Result

Model/ Sampling	Parameter Selection	Macro_Recall
RF_1	{'n_estimators': 1}	0.982883
	{'n_estimators': 50}	0.988259
	{'n_estimators': 100}	0.988366
RF_2	{'n_estimators': 1}	0.984327
	{'n_estimators': 50}	0.990118
	{'n_estimators': 100}	0.990307
RF_3	{'n_estimators': 1}	0.984998
	{'n_estimators': 50}	0.995864
	{'n_estimators': 100}	**0.996061**

Model / Sampling	Parameter Selection	Macro_Recall
XGB_1	{'learning_rate': 0.01, 'max_depth': 1}	0.768638
	{'learning_rate': 0.01, 'max_depth': 2}	0.926331
	{'learning_rate': 0.01, 'max_depth': 3}	0.975613
	{'learning_rate': 0.1, 'max_depth': 1}	0.955064
	{'learning_rate': 0.1, 'max_depth': 2}	0.984936
	{'learning_rate': 0.1, 'max_depth': 3}	0.987962
	{'learning_rate': 0.2, 'max_depth': 1}	0.983606
	{'learning_rate': 0.2, 'max_depth': 2}	0.988041
	{'learning_rate': 0.2, 'max_depth': 3}	0.988384
XGB_2	{'learning_rate': 0.01, 'max_depth': 1}	0.780537
	{'learning_rate': 0.01, 'max_depth': 2}	0.928141
	{'learning_rate': 0.01, 'max_depth': 3}	0.983537
	{'learning_rate': 0.1, 'max_depth': 1}	0.974304
	{'learning_rate': 0.1, 'max_depth': 2}	0.988017
	{'learning_rate': 0.1, 'max_depth': 3}	0.990149
	{'learning_rate': 0.2, 'max_depth': 1}	0.985603
	{'learning_rate': 0.2, 'max_depth': 2}	0.990387
	{'learning_rate': 0.2, 'max_depth': 3}	0.990181
XGB_3	{'learning_rate': 0.01, 'max_depth': 1}	0.881486
	{'learning_rate': 0.01, 'max_depth': 2}	0.963035
	{'learning_rate': 0.01, 'max_depth': 3}	0.980178
	{'learning_rate': 0.1, 'max_depth': 1}	0.984041
	{'learning_rate': 0.1, 'max_depth': 2}	0.994168
	{'learning_rate': 0.1, 'max_depth': 3}	0.994164
	{'learning_rate': 0.2, 'max_depth': 1}	0.993609
	{'learning_rate': 0.2, 'max_depth': 2}	**0.994356**
	{'learning_rate': 0.2, 'max_depth': 3}	0.994354