FRAUD DETECTION

Fast Campus Data Science School 14th
Jaewook Kim
Sangwoo Im
Jinseo Lee

PROBLEM

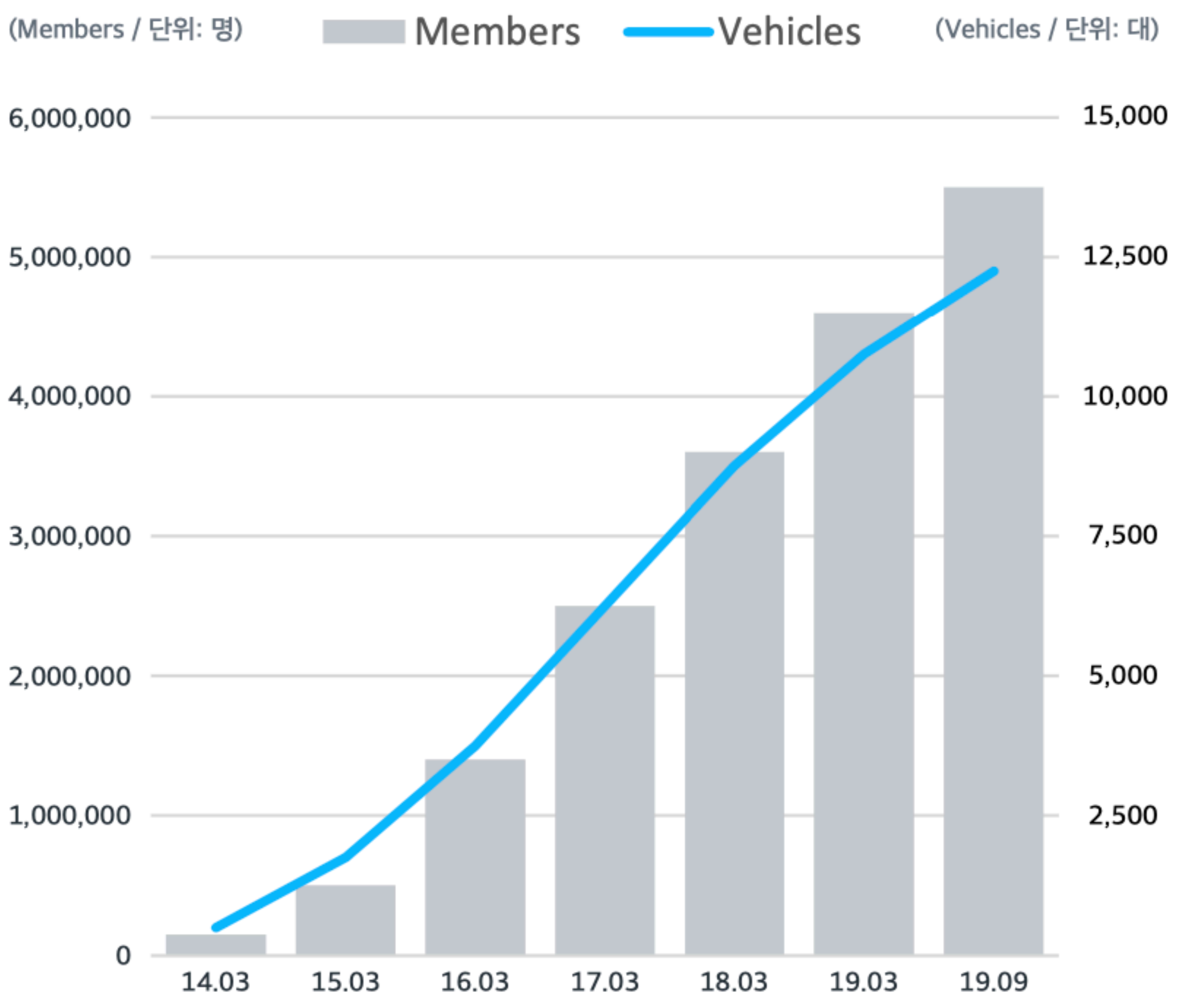
ACCELERATED GROWTH

APP DOWNLOAD
6.5M

ACCUMULATED USERS
5.8M

MONTHLY USERS
200K

- 12,000 Vehicles Ready to Go
- 5.8M Accumulated Users
- Upward Trend of Demand



Source : SOCAR 회사 소개서

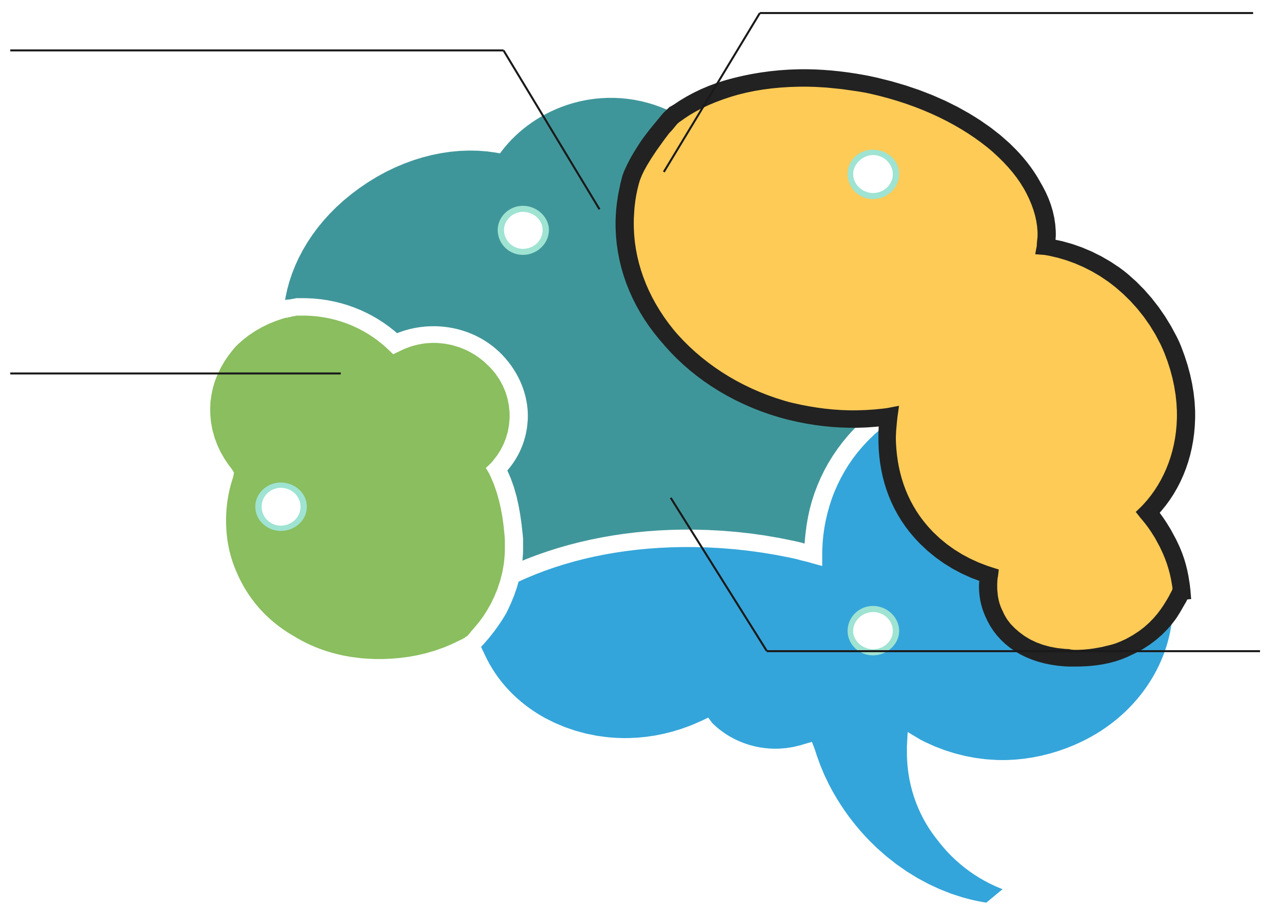
PROBLEM

"CRASH FOR CASH?"

NEWS :https://www.chosun.com/economy/2020/10/16/RS2SPARHKBAQVKMQ2AGI62XWNU/

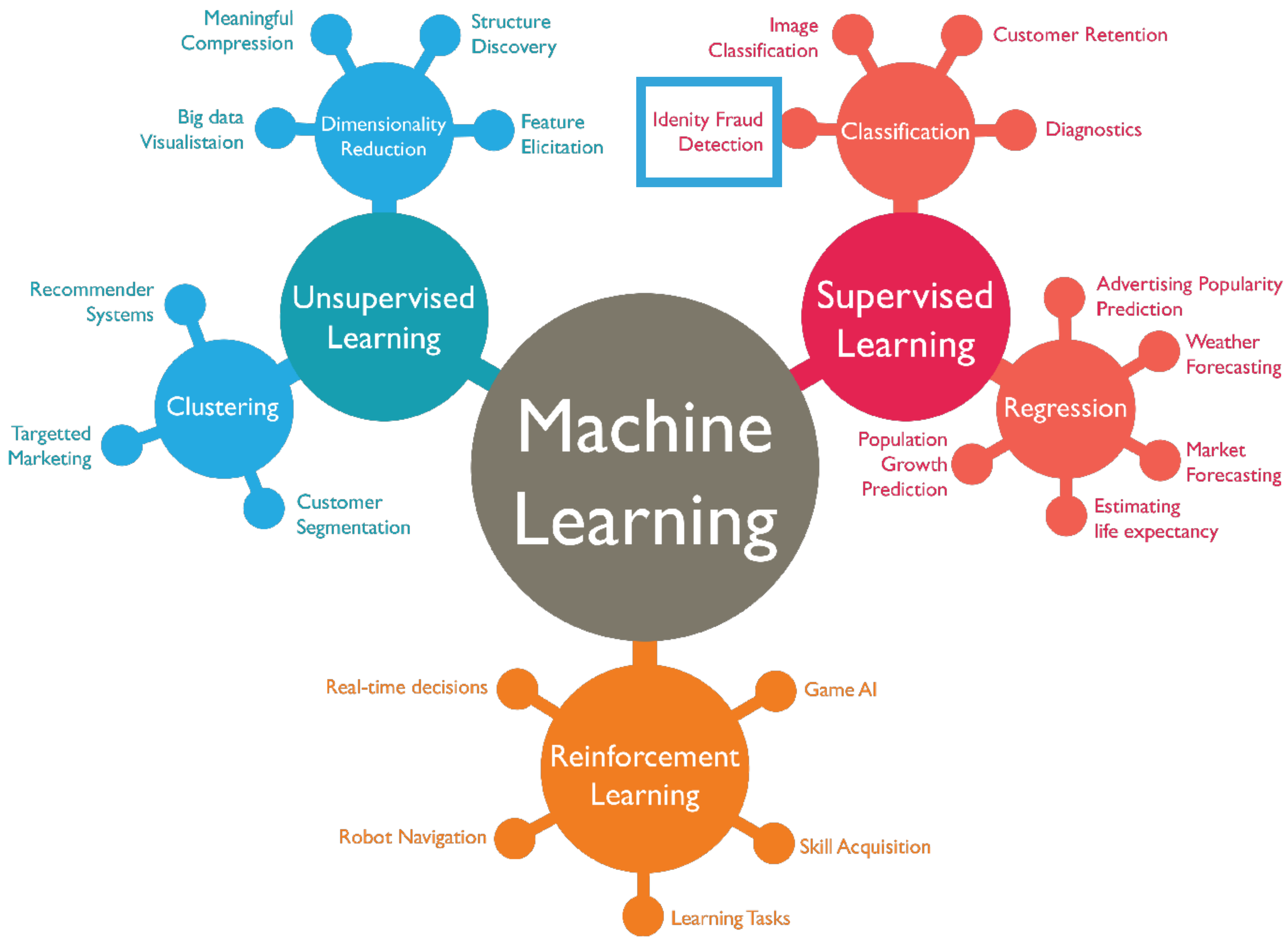
CAN WE DETECT SCAMS?

LET'S SOLVE WITH MACHINE LEARNING



Machine Learning

HOW IS MACHINE LEARNING APPLIED IN FRAUD DETECTION?



KEY POINT IN FRAUD DETECTION

	Predicted N	Predicted P
Actual N	TN	FP
Actual P	FN	TP

Recall

Precision

- **Recall ↑ = TP / (FN + TP)**
  - Detection Rate ↑
  - Loss ↓
- **Precision ↑ = TP / (FP + TP)**
  - Misidentification Rate ↓
  - Customer Complaint ↓

GOAL

LET’S FIND THE OPTIMAL MODEL!

DATASET

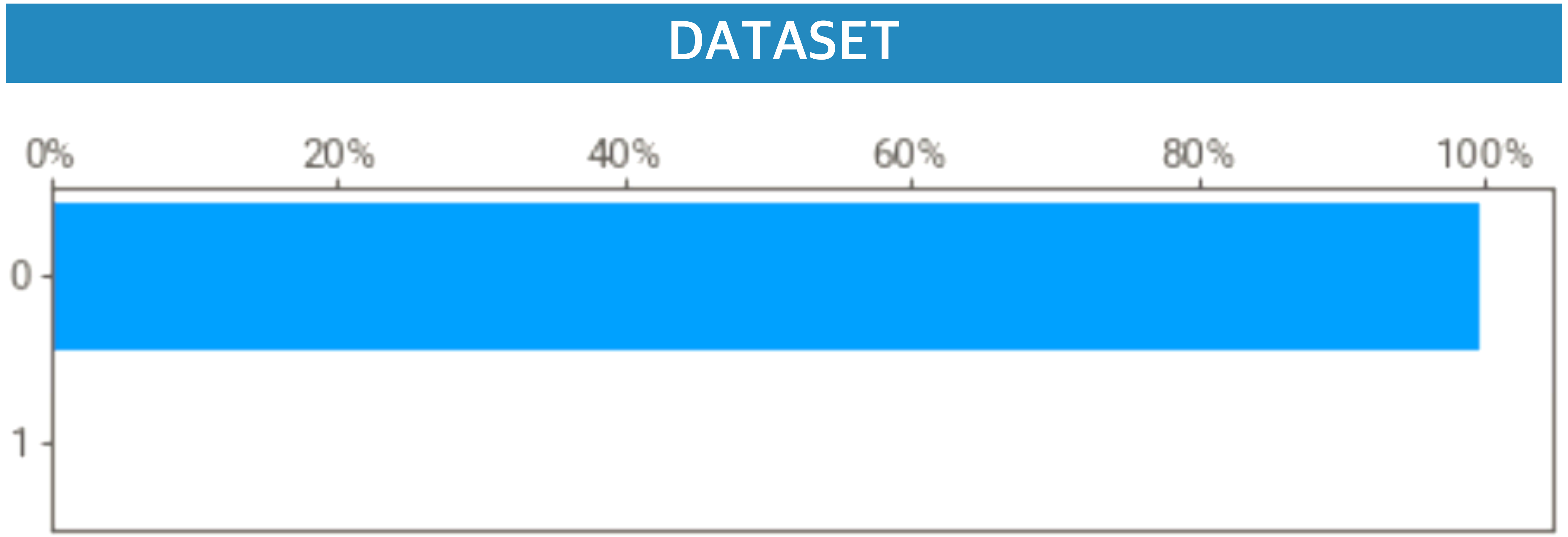
RAW DATA : 16000 X 25

- **Data Unrevealed upon the Request of Data Provider**

EDA

1. FRAUD_YN : IMBALANCED DATA

- Of 16,000 accident samples
  - Case 0: 99.74%
  - Case 1: 0.26%
- Definition of Fraud
  - Staged Crash



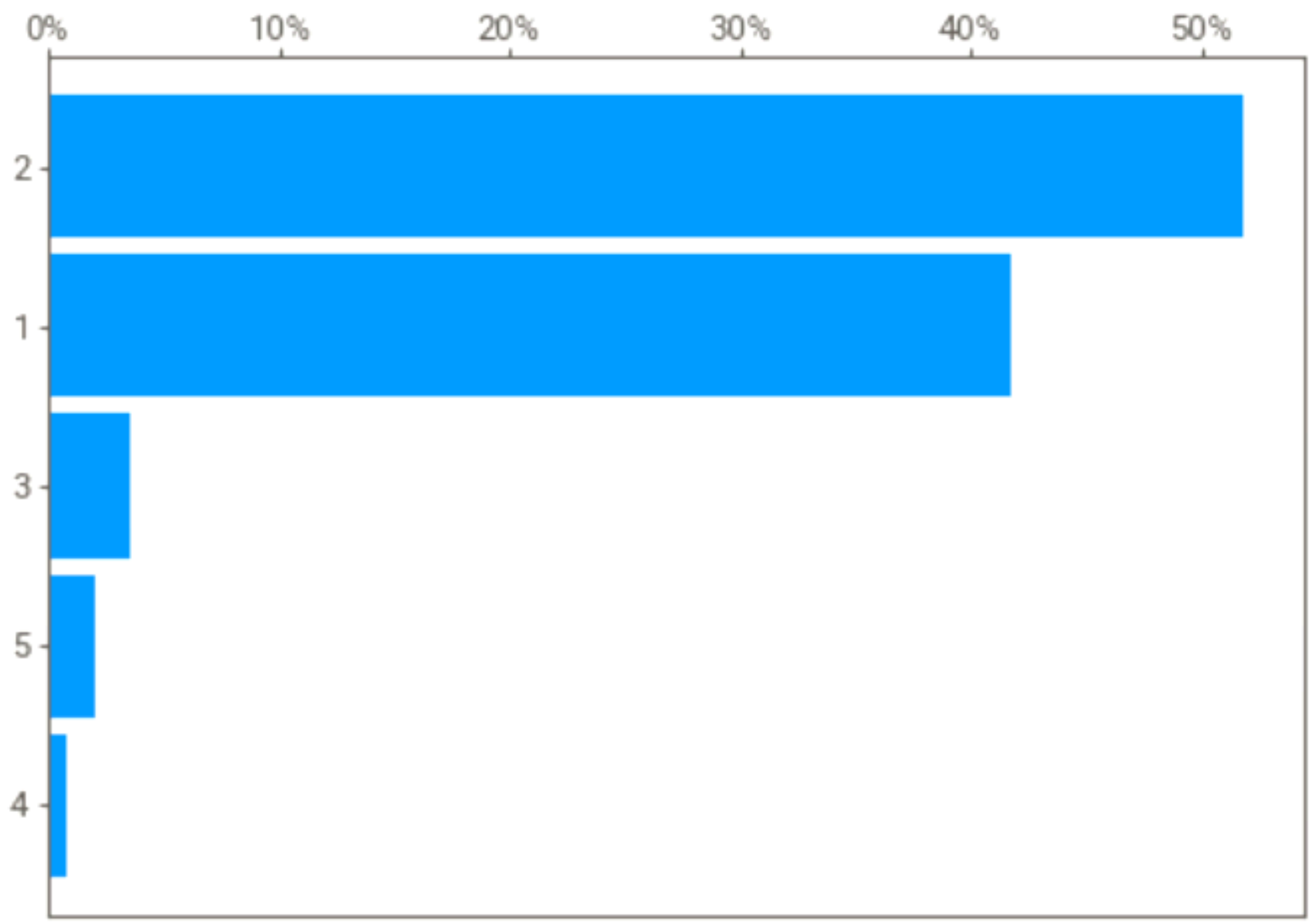
TOP CATEGORIES

0	15,959	99%
1	41	0%
ALL	16,000	100%

2. C1 -> ONE_HOT_ENCODING

- 1
- 2
- 3
- 4
- 5
- Dataset & Fraud
  - Case 1,2 : High frequency
- Different ratio

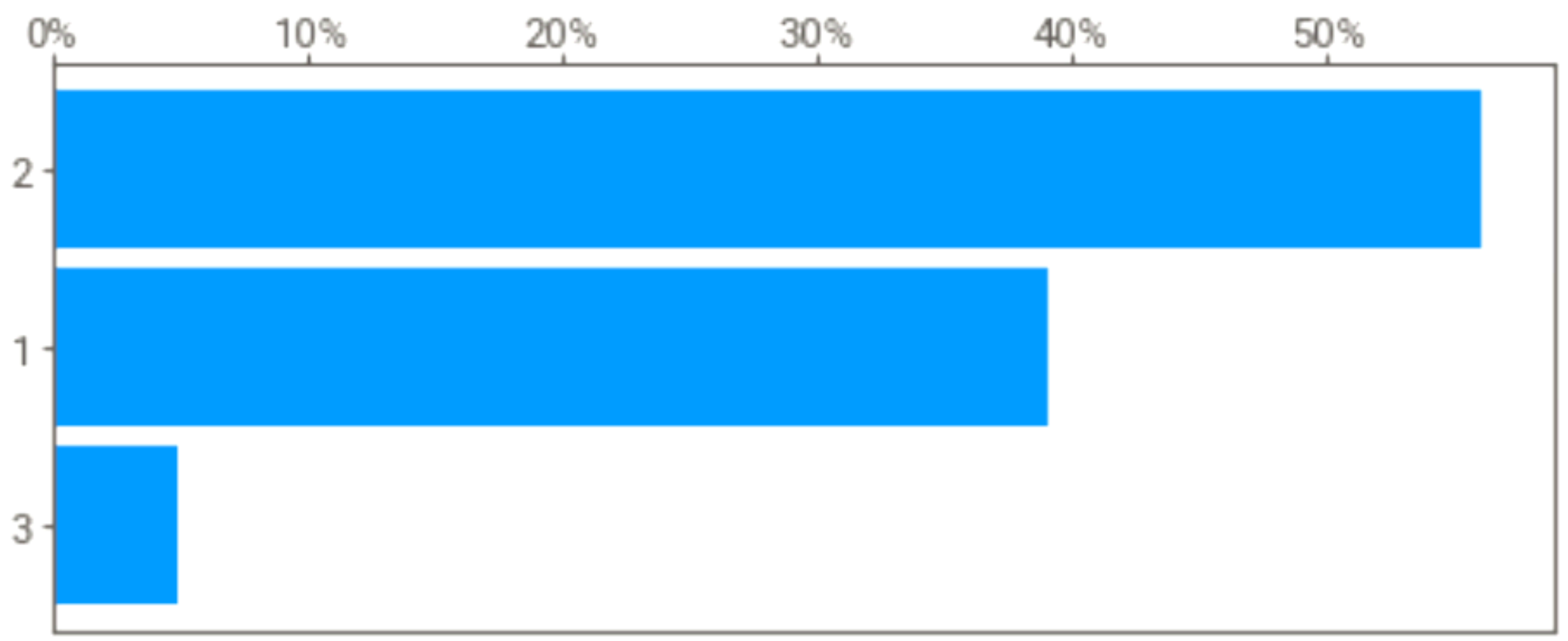
DATASET



TOP CATEGORIES

2	8,296	52%
1	6,685	42%
3	563	4%
5	332	2%
4	124	1%
ALL	16,000	100%

FRAUD



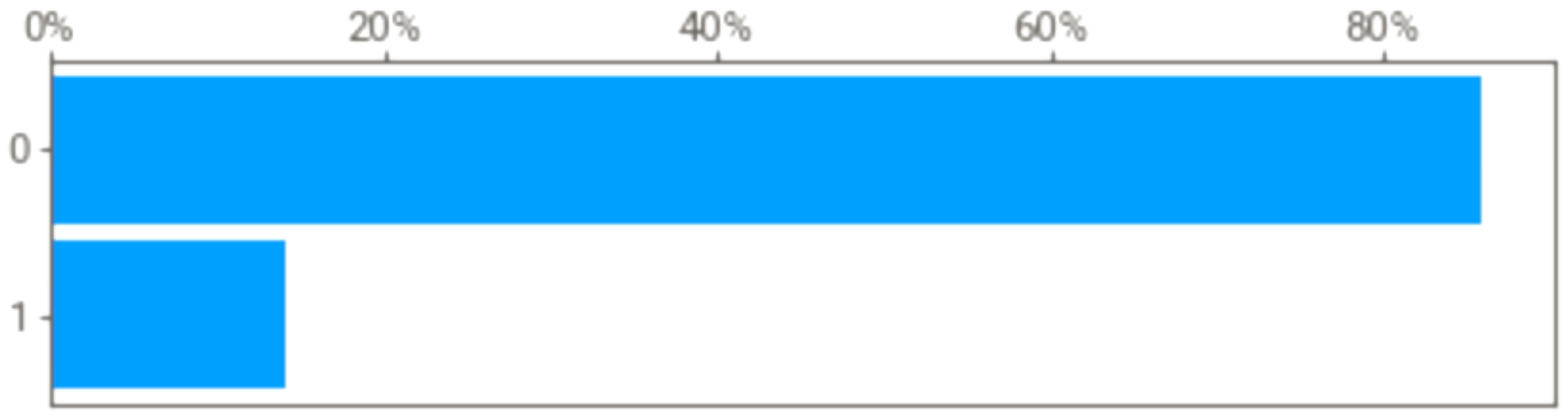
TOP CATEGORIES

2	23	56%
1	16	39%
3	2	5%
ALL	41	100%

3. C2

- Similar Ratio

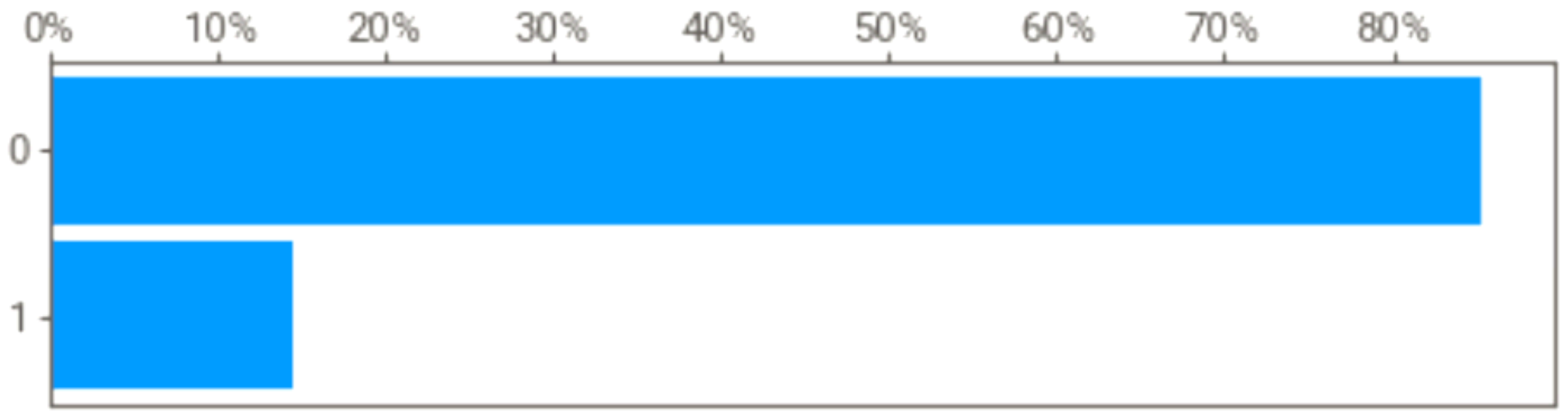
DATASET



TOP CATEGORIES

0	13,751	86%
1	2,249	14%
ALL	16,000	100%

FRAUD



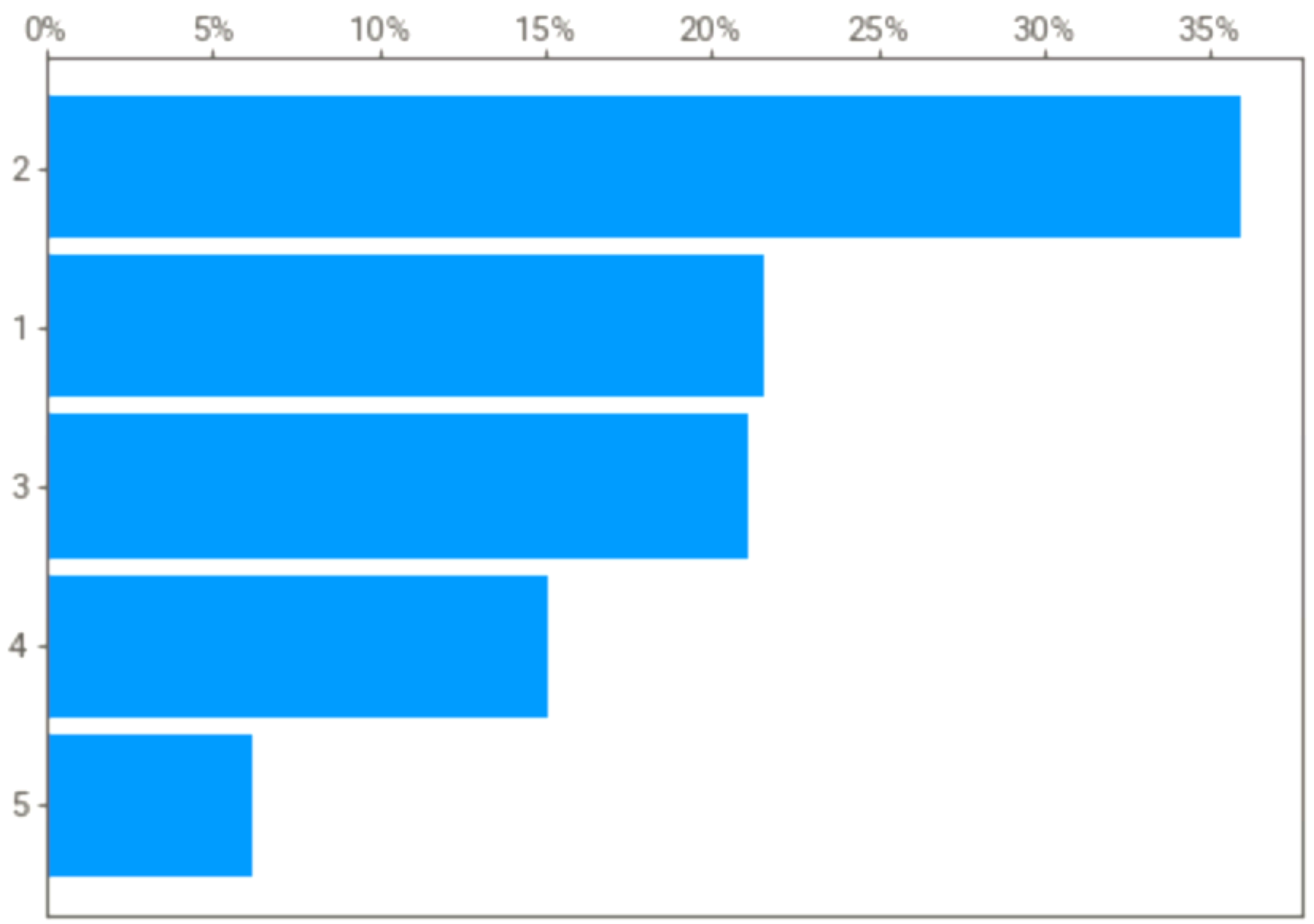
TOP CATEGORIES

0	35	85%
1	6	15%
ALL	41	100%

4. C3 -> ONE_HOT_ENCODING

- 1
- 2
- 3
- 4
- 5
- High frequency of Case 1-3 in Dataset and Fraud
- Different Ratio
-

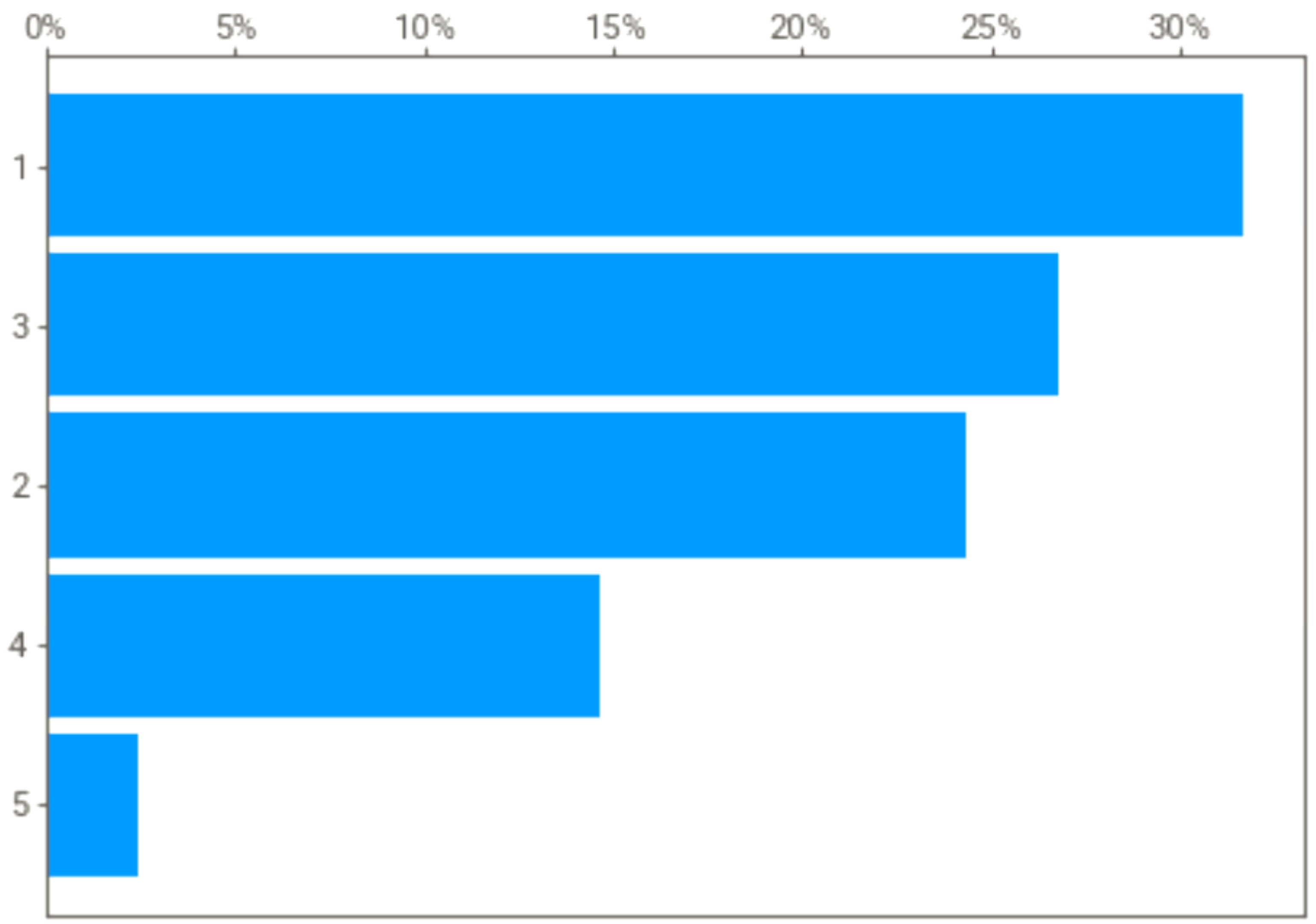
DATASET



TOP CATEGORIES

2	5,753	36%
1	3,454	22%
3	3,376	21%
4	2,419	15%
5	998	6%
ALL	16,000	100%

FRAUD



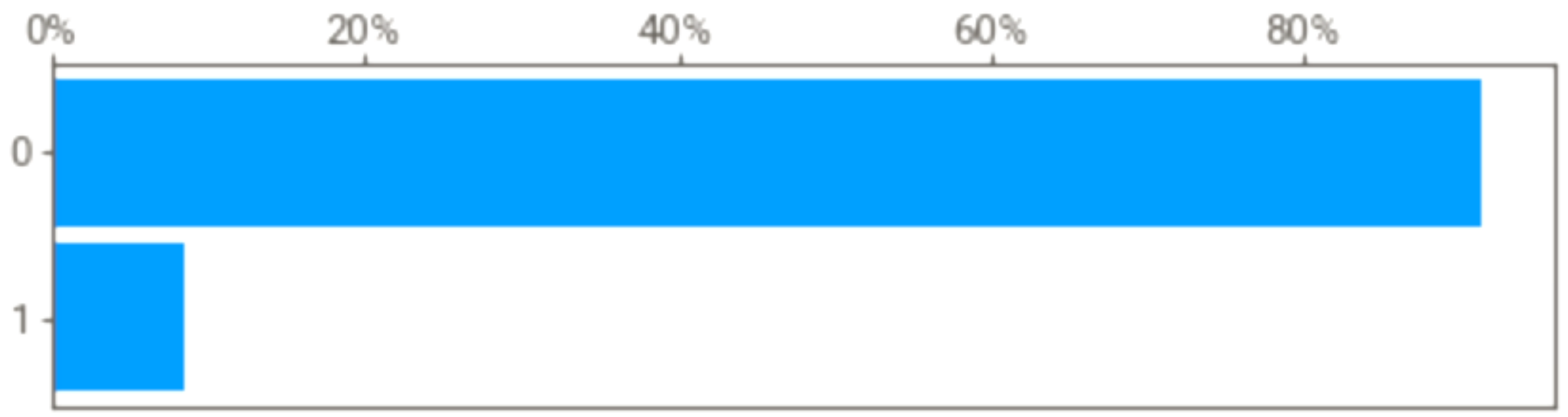
TOP CATEGORIES

1	13	32%
3	11	27%
2	10	24%
4	6	15%
5	1	2%
ALL	41	100%

5. C4

- Similar Ratio

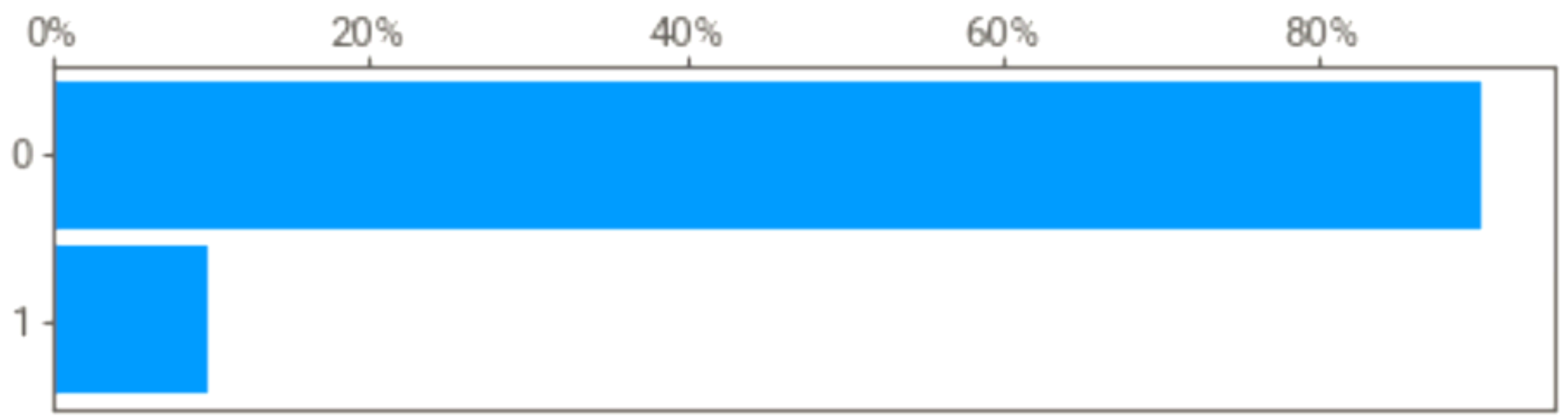
DATASET



TOP CATEGORIES

0	14,635	91%
1	1,365	9%
ALL	16,000	100%

FRAUD



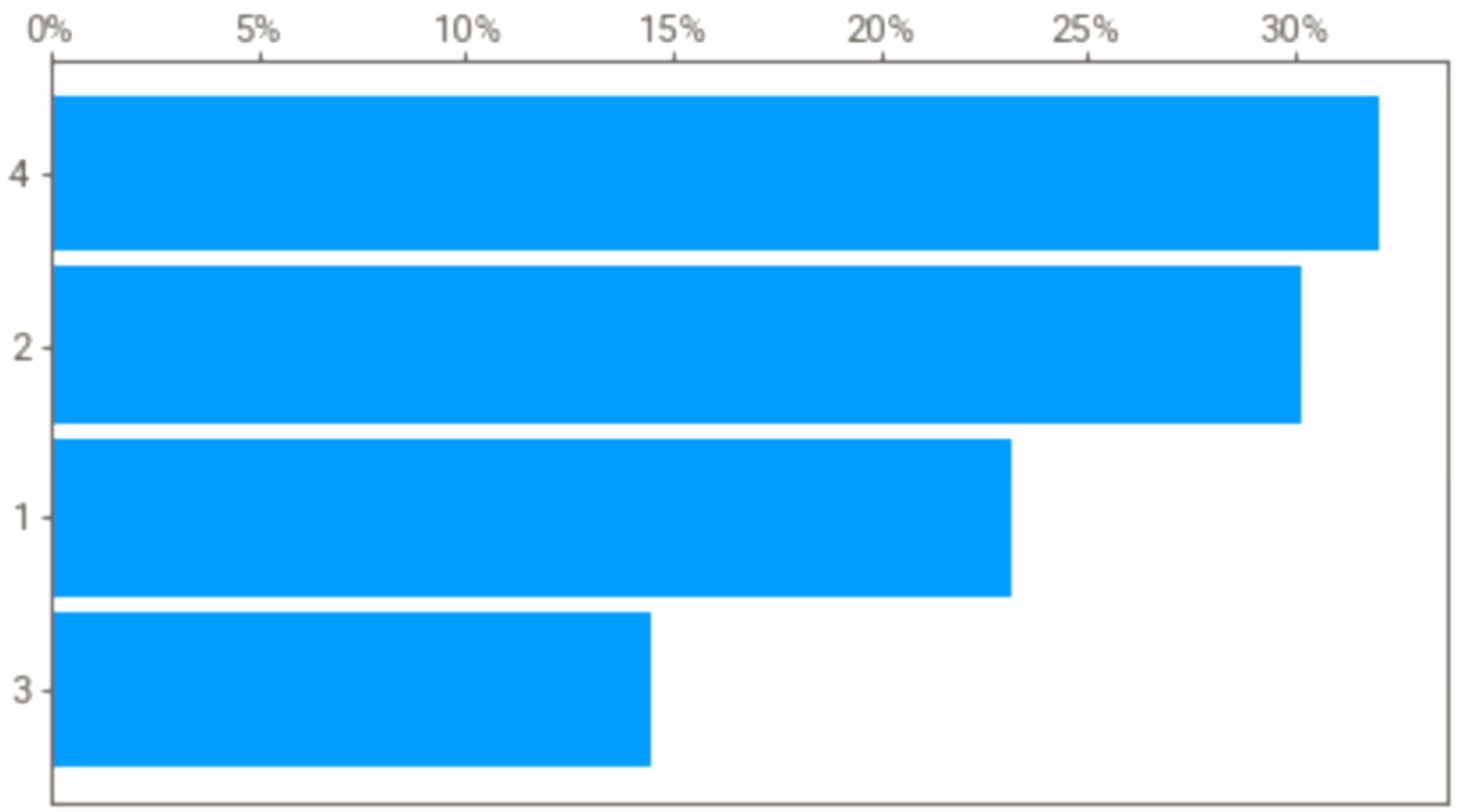
TOP CATEGORIES

0	37	90%
1	4	10%
ALL	41	100%

6. C5

- 1
- 2
- 3
- 4
- High frequency of Case 4 in Dataset and Fraud
- Different Ratio

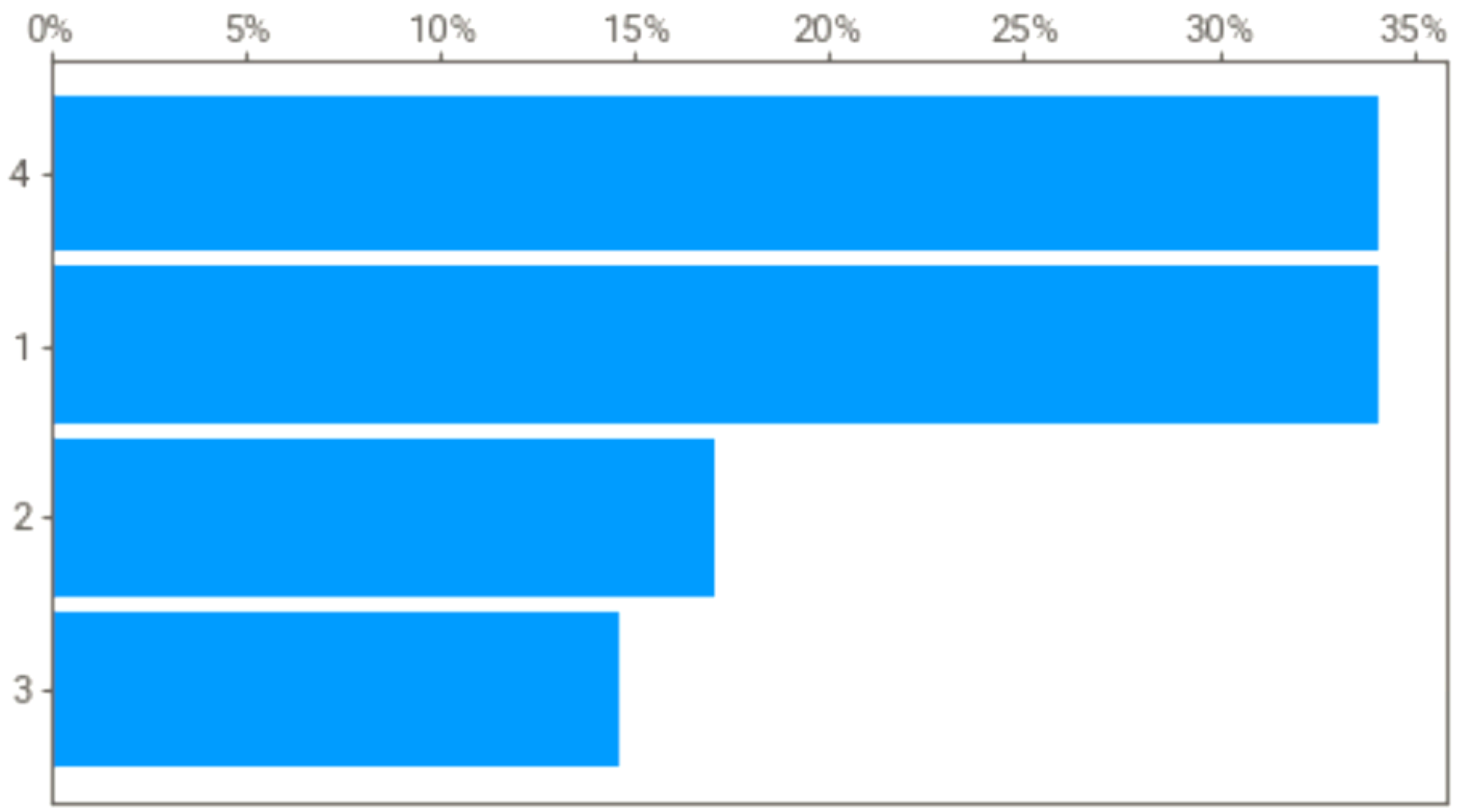
DATASET



TOP CATEGORIES

4	5,132	32%
2	4,836	30%
1	3,709	23%
3	2,323	15%
ALL	16,000	100%

FRAUD



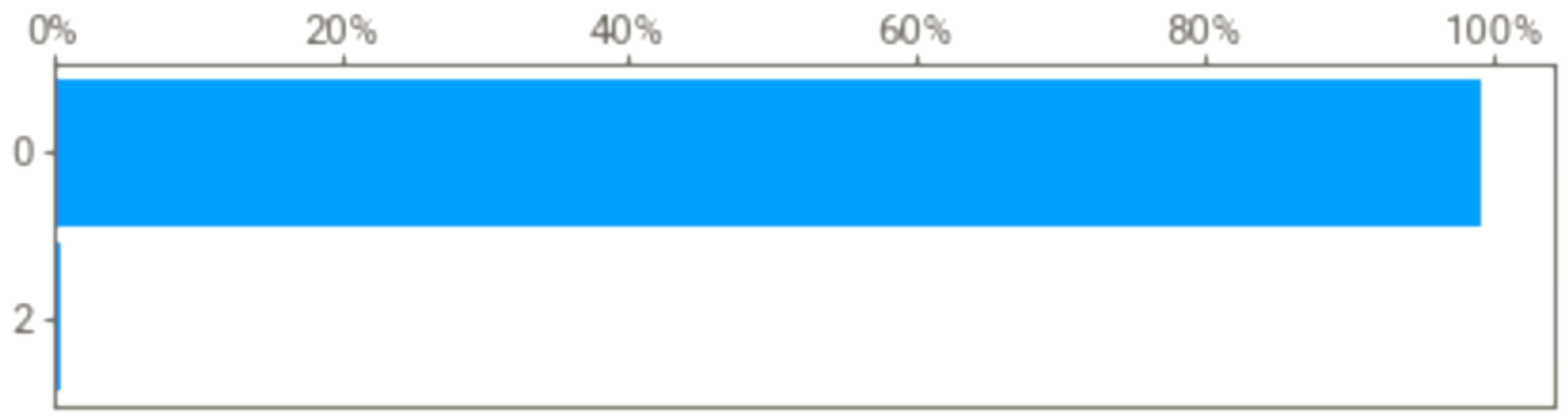
TOP CATEGORIES

4	14	34%
1	14	34%
2	7	17%
3	6	15%
ALL	41	100%

7. C6

- 0
- 2
- Dataset : 0 (99%)
- Fraud : 0 (100%)
- Similar Ratio

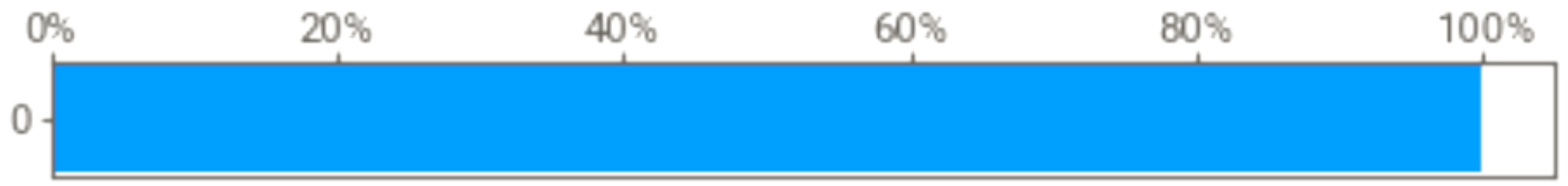
DATASET



TOP CATEGORIES

0	15,895	99%
2	105	1%
ALL	16,000	100%

FRAUD



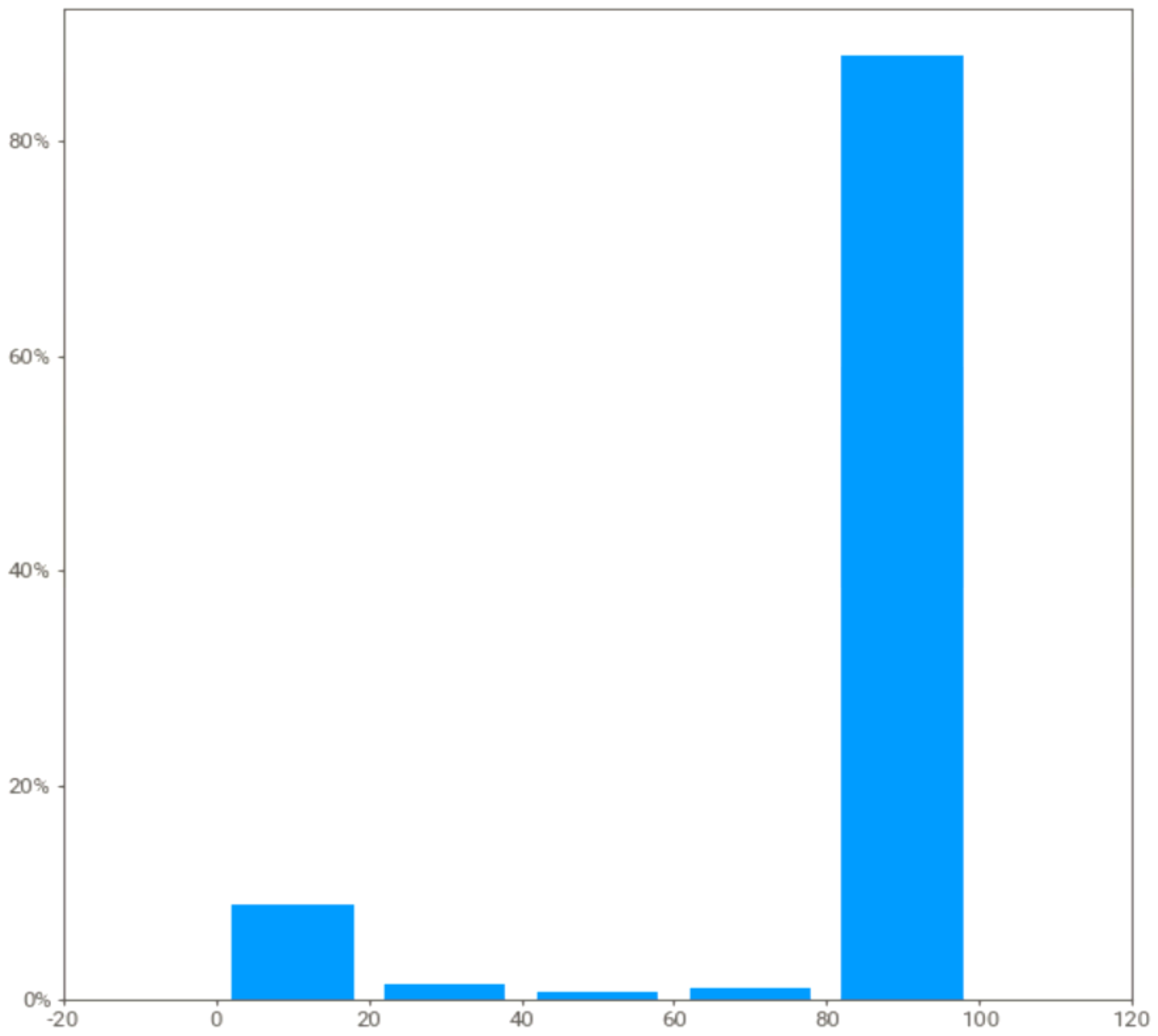
TOP CATEGORIES

0	41	100%
ALL	41	100%

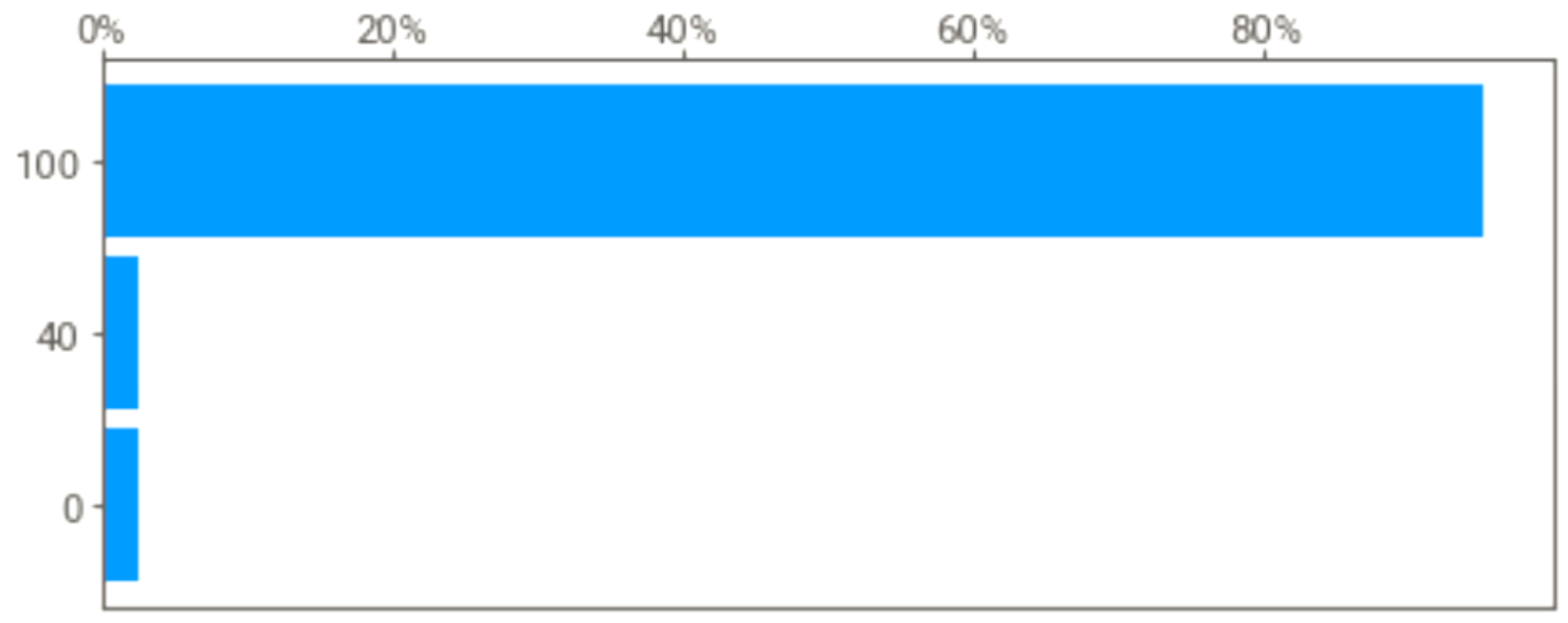
8. C7

- Dataset
  - 100% : 13781/86%
  - 0% : 1364/9%
  - Others : 855/5%
- Fraud
  - 100% : 39/95%
- Different Ratio

DATASET



FRAUD



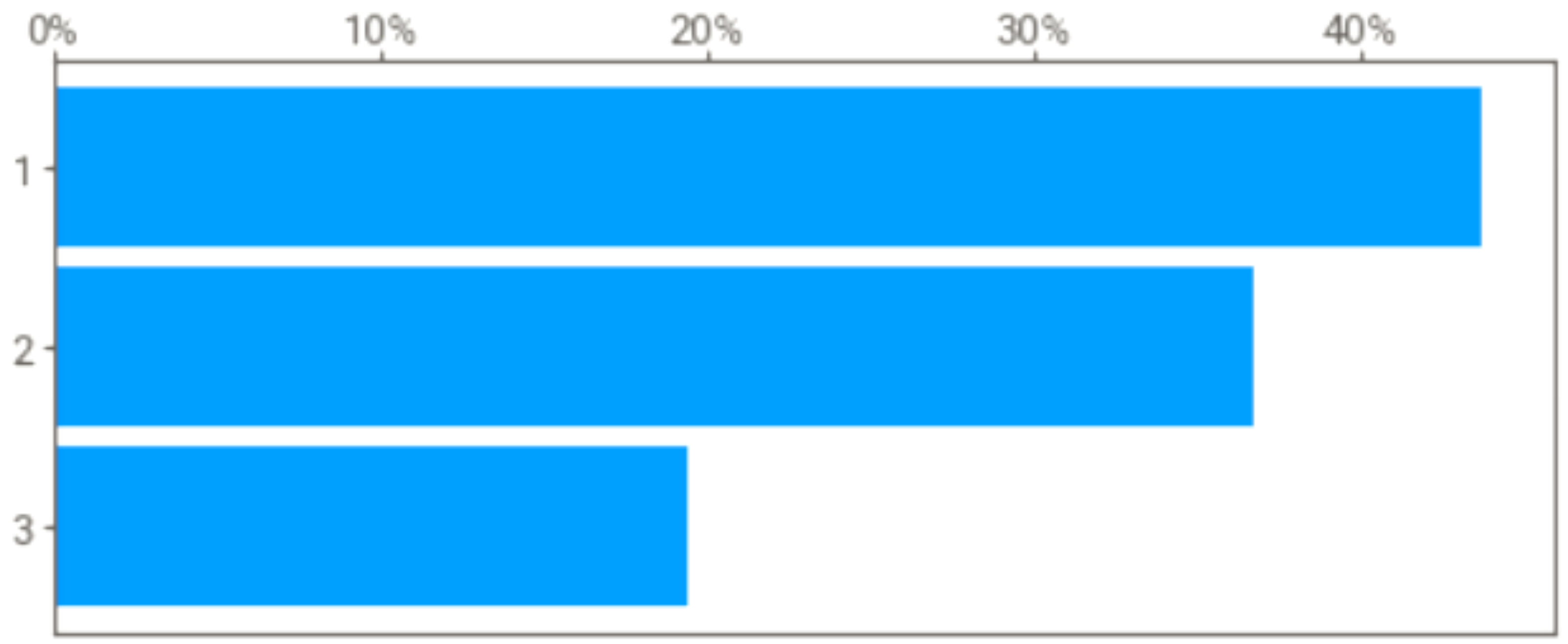
TOP CATEGORIES

100	39	95%
40	1	2%
0	1	2%
ALL	41	100%

9. C8

- 1 :
- 2 :
- 3 :
- Case1
  - default option in SOCAR APP
  - High frequency in Dataset and Fraud
- Different Ratio
-

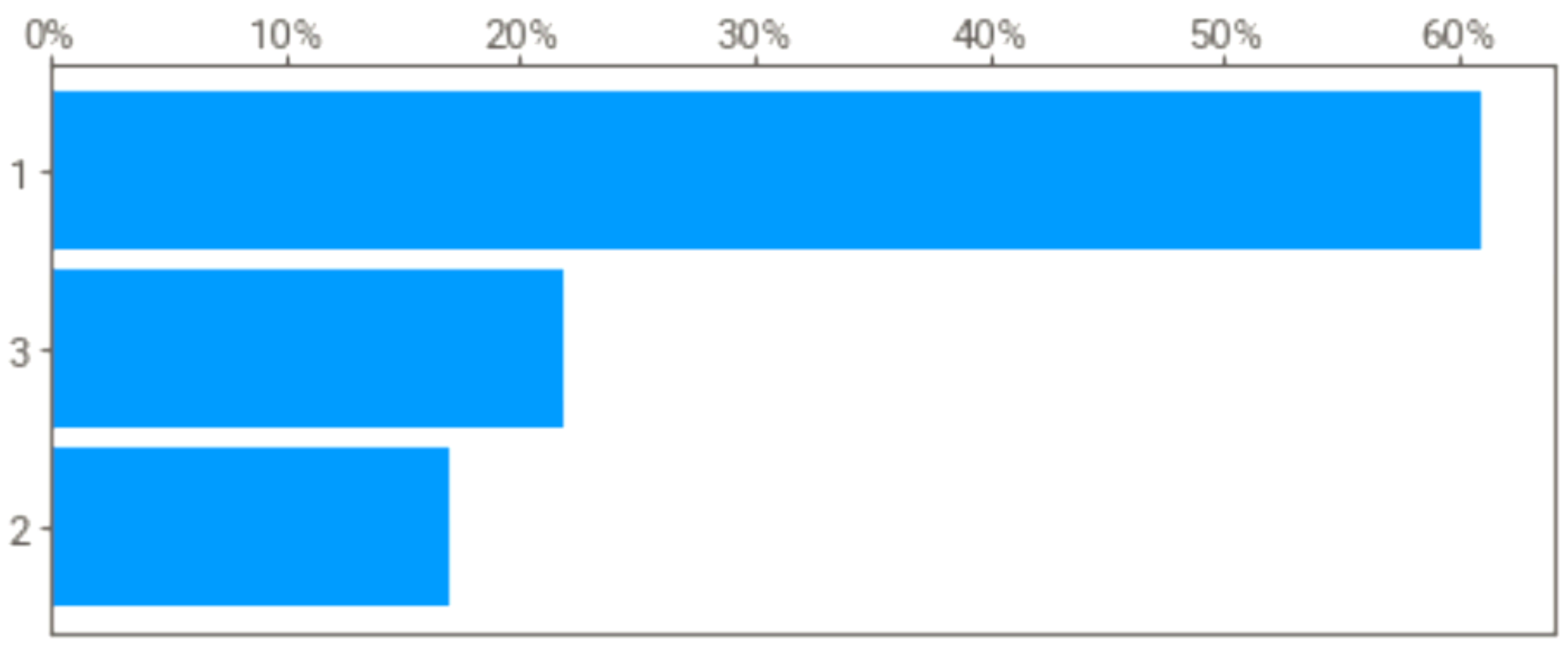
DATASET



TOP CATEGORIES

1	7,002	44%
2	5,882	37%
3	3,116	19%
ALL	16,000	100%

FRAUD



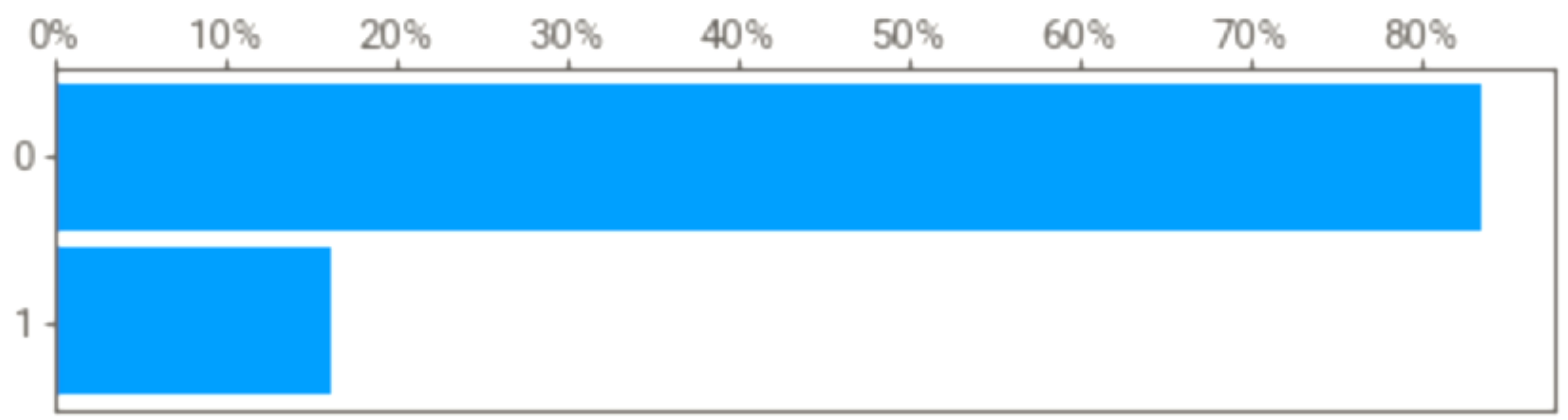
TOP CATEGORIES

1	25	61%
3	9	22%
2	7	17%
ALL	41	100%

10. C9

- Similar Ratio

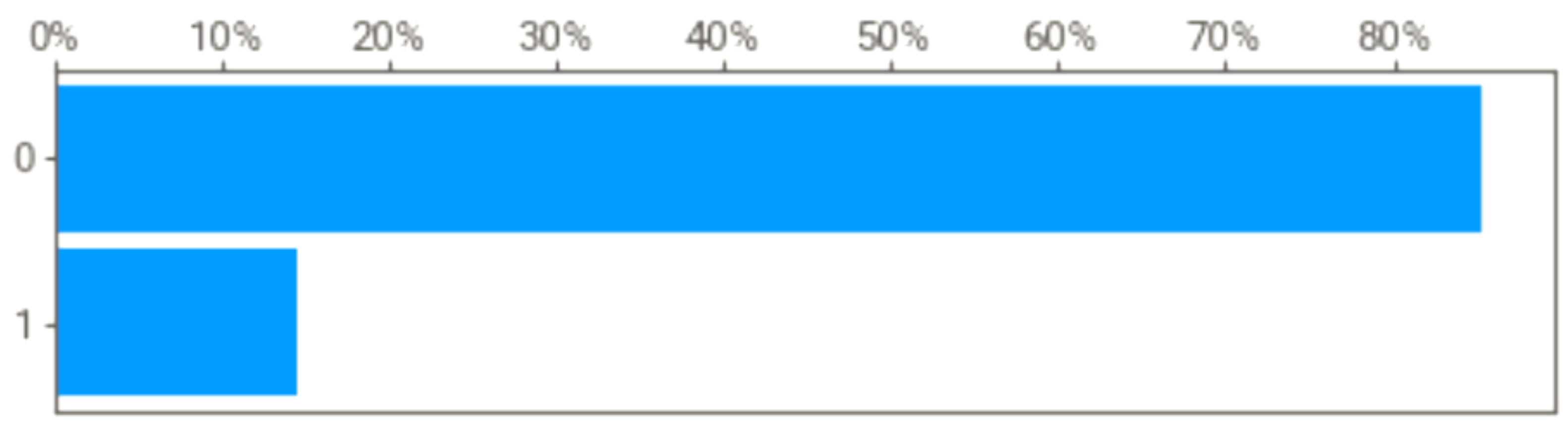
DATASET



TOP CATEGORIES

0	13,394	84%
1	2,606	16%
ALL	16,000	100%

FRAUD



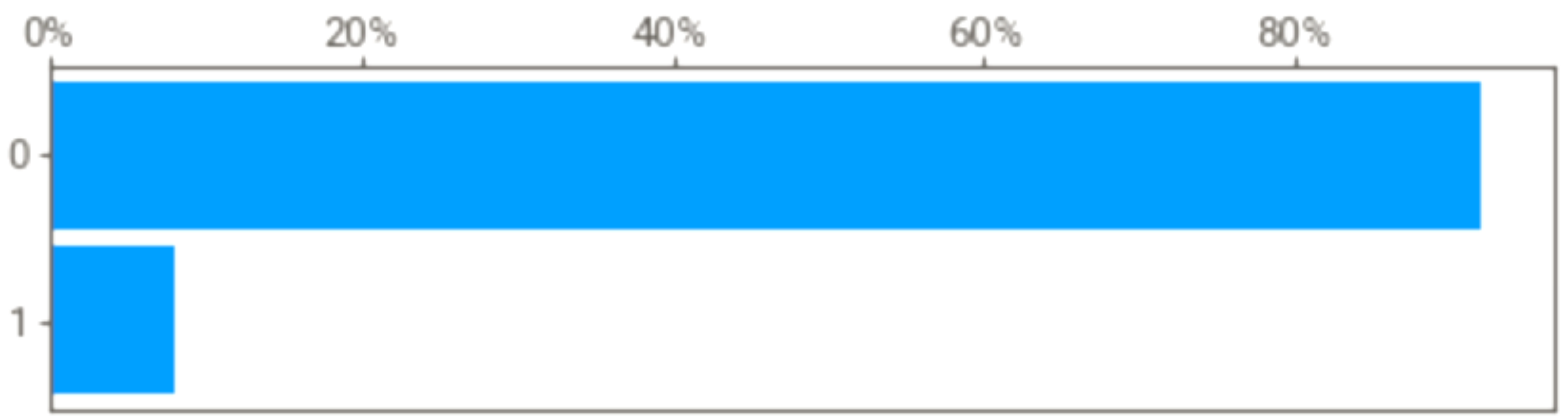
TOP CATEGORIES

0	35	85%
1	6	15%
ALL	41	100%

11. C10

- Ratio of Case 0
  - Dataset 92%
  - Fraud 85%

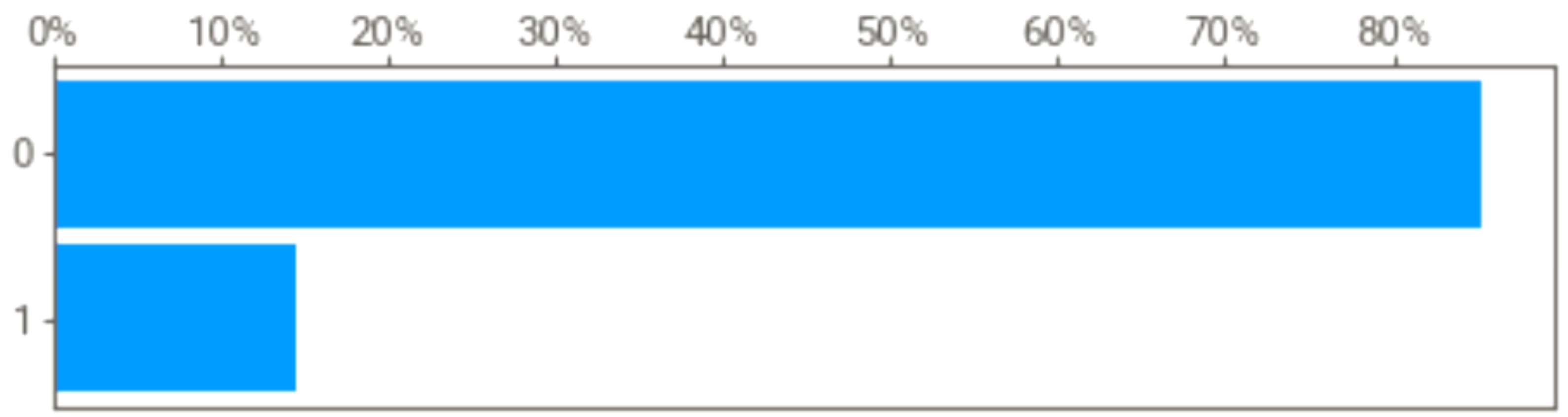
DATASET



TOP CATEGORIES

0	14,719	92%
1	1,281	8%
ALL	16,000	100%

FRAUD



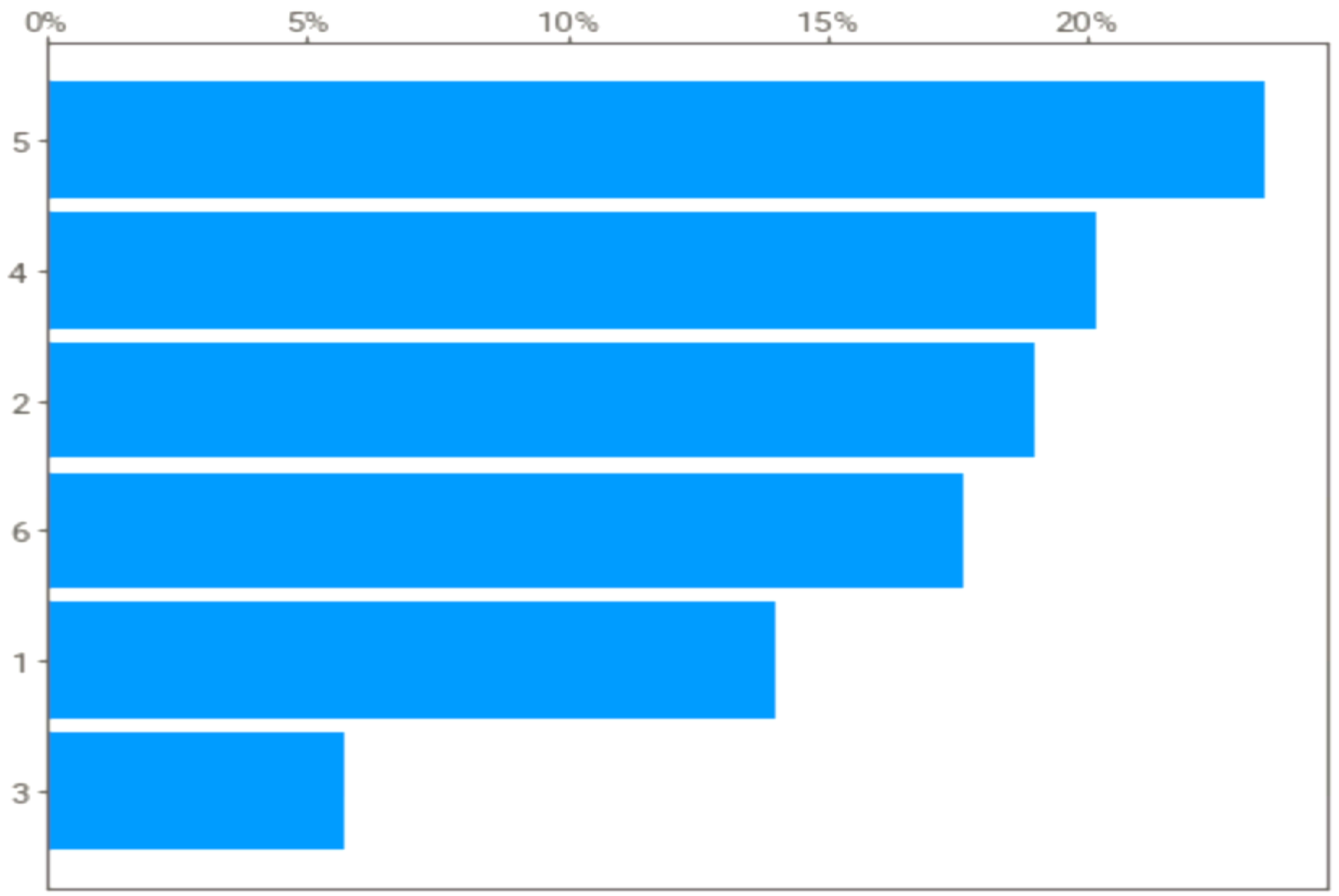
TOP CATEGORIES

0	35	85%
1	6	15%
ALL	41	100%

12. C11-> ONE_HOT_ENCODING

- 1
- 2
- 3
- 4
- 5
- 6
- Case 4 & 5 : High frequency in Dataset and Fraud

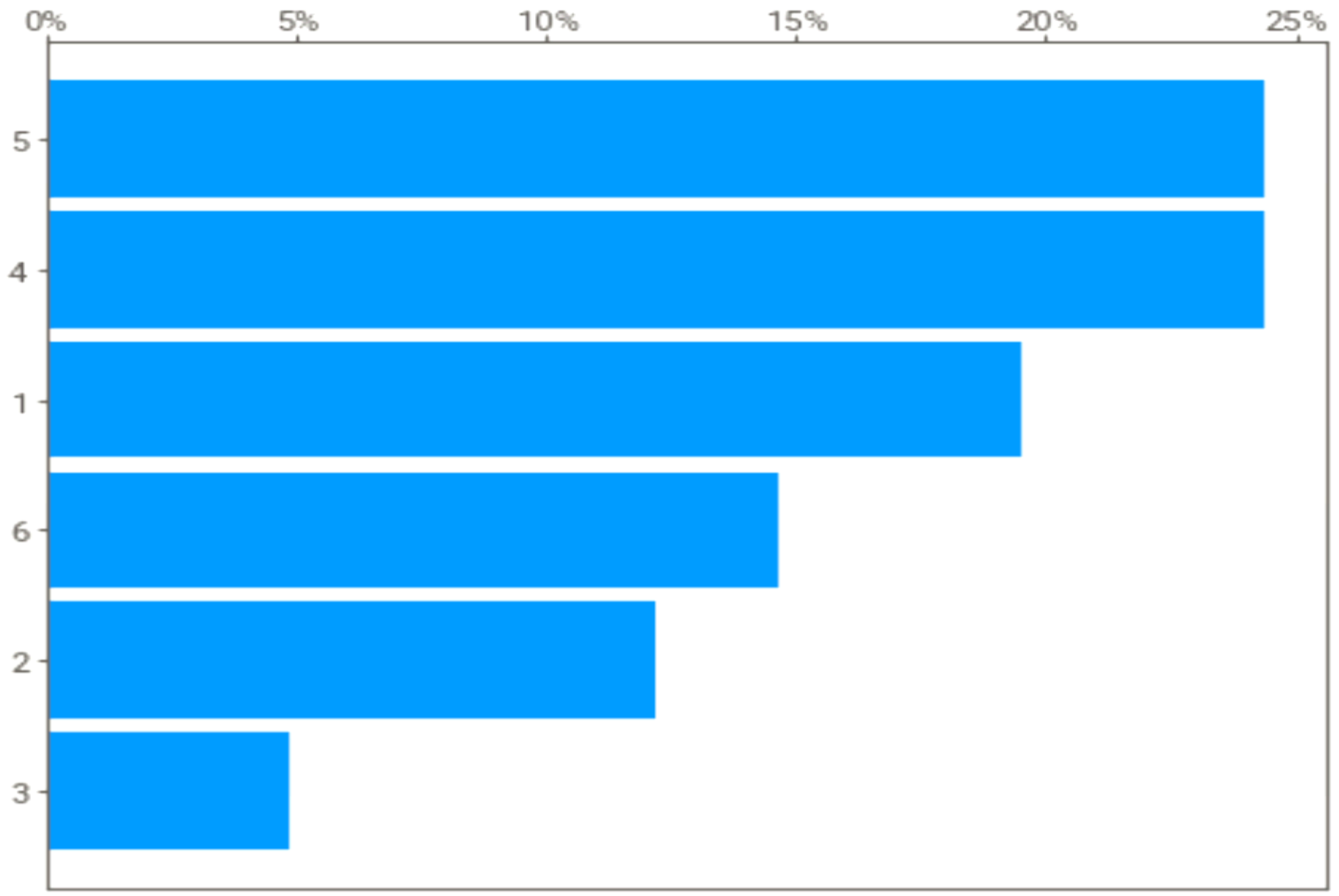
DATASET



TOP CATEGORIES

5	3,750	23%
4	3,228	20%
2	3,040	19%
6	2,823	18%
1	2,242	14%
3	917	6%
ALL	16,000	100%

FRAUD



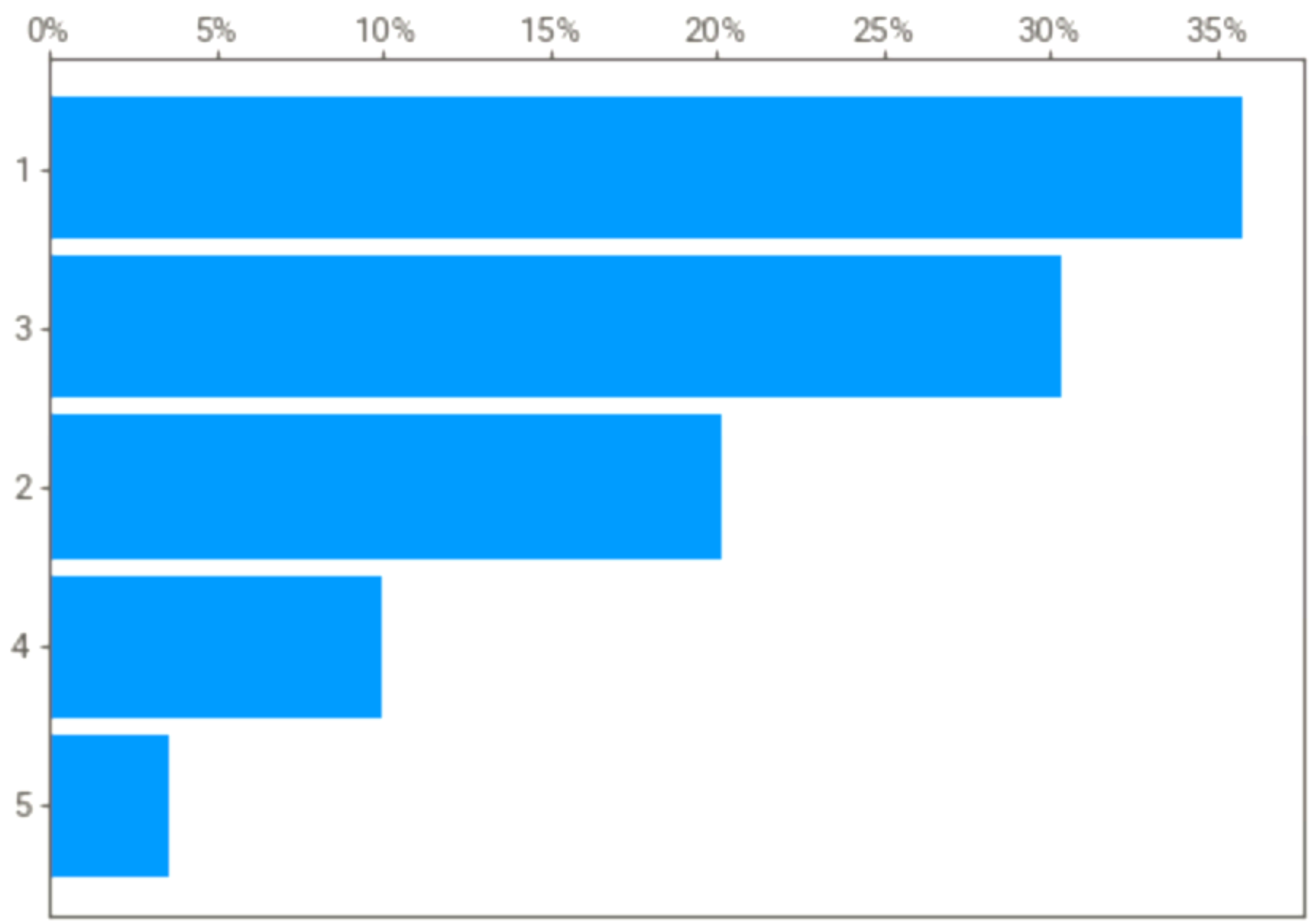
TOP CATEGORIES

5	10	24%
4	10	24%
1	8	20%
6	6	15%
2	5	12%
3	2	5%
ALL	41	100%

13. C12 -> ONE_HOT_ENCODING

- 1
- 2
- 3
- 4
- 5
- Different Ratio

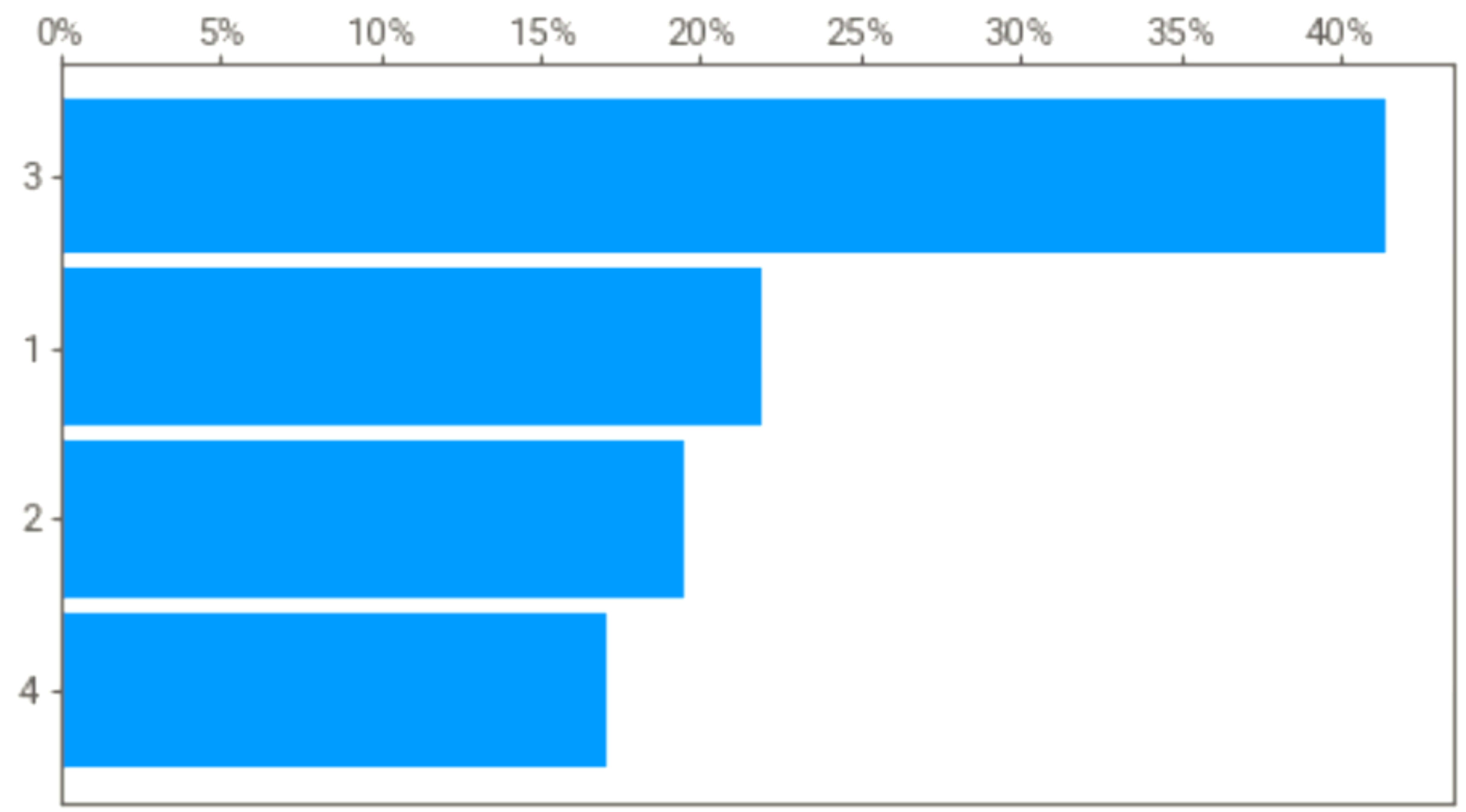
DATASET



TOP CATEGORIES

1	5,727	36%
3	4,858	30%
2	3,234	20%
4	1,605	10%
5	576	4%
ALL	16,000	100%

FRAUD



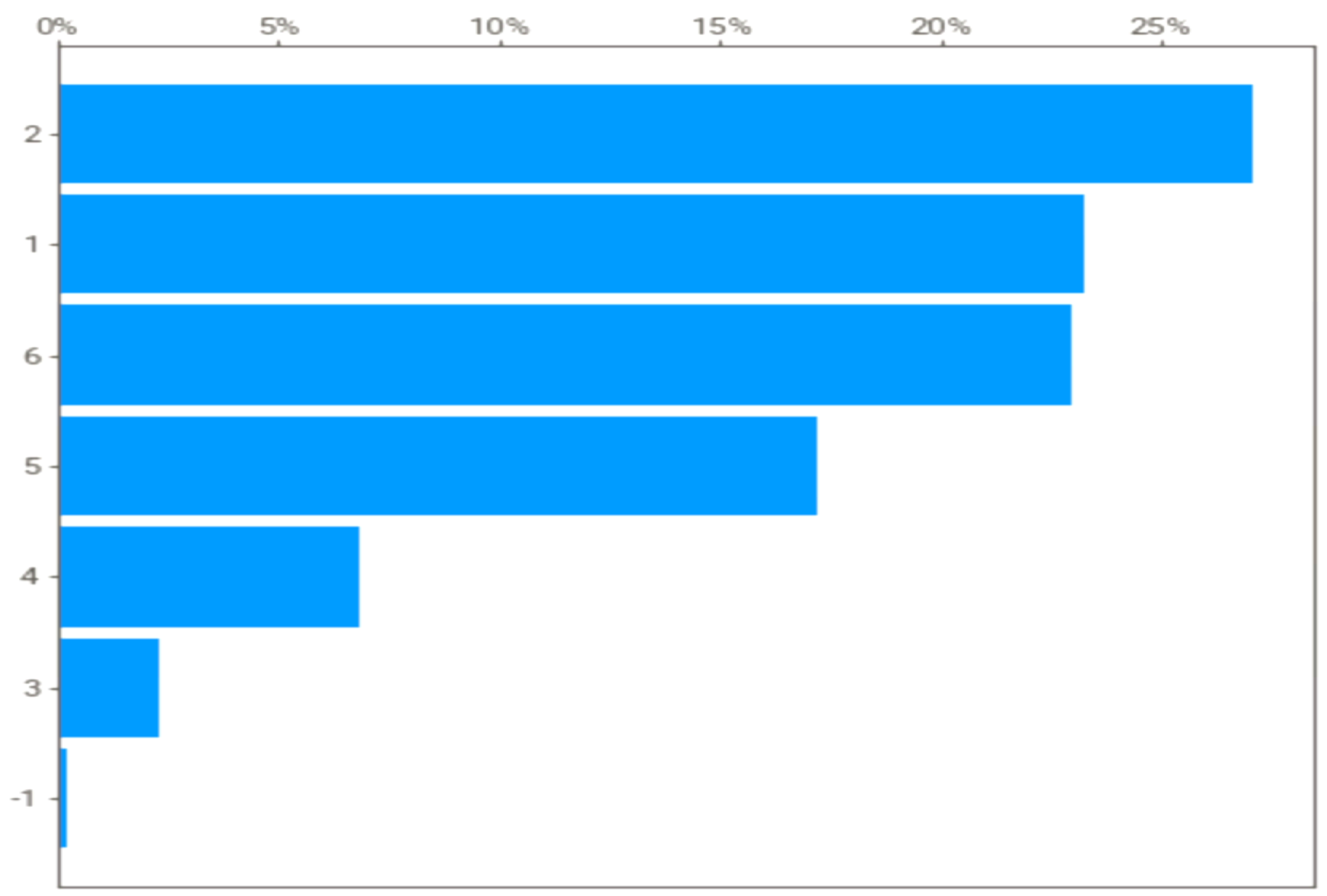
TOP CATEGORIES

3	17	41%
1	9	22%
2	8	20%
4	7	17%
ALL	41	100%

14. C13 -> ONE_HOT_ENCODING

- 1
- 2
- 3
- 4
- 5
- 6
- -1
- High frequency of Case 1 in Fraud

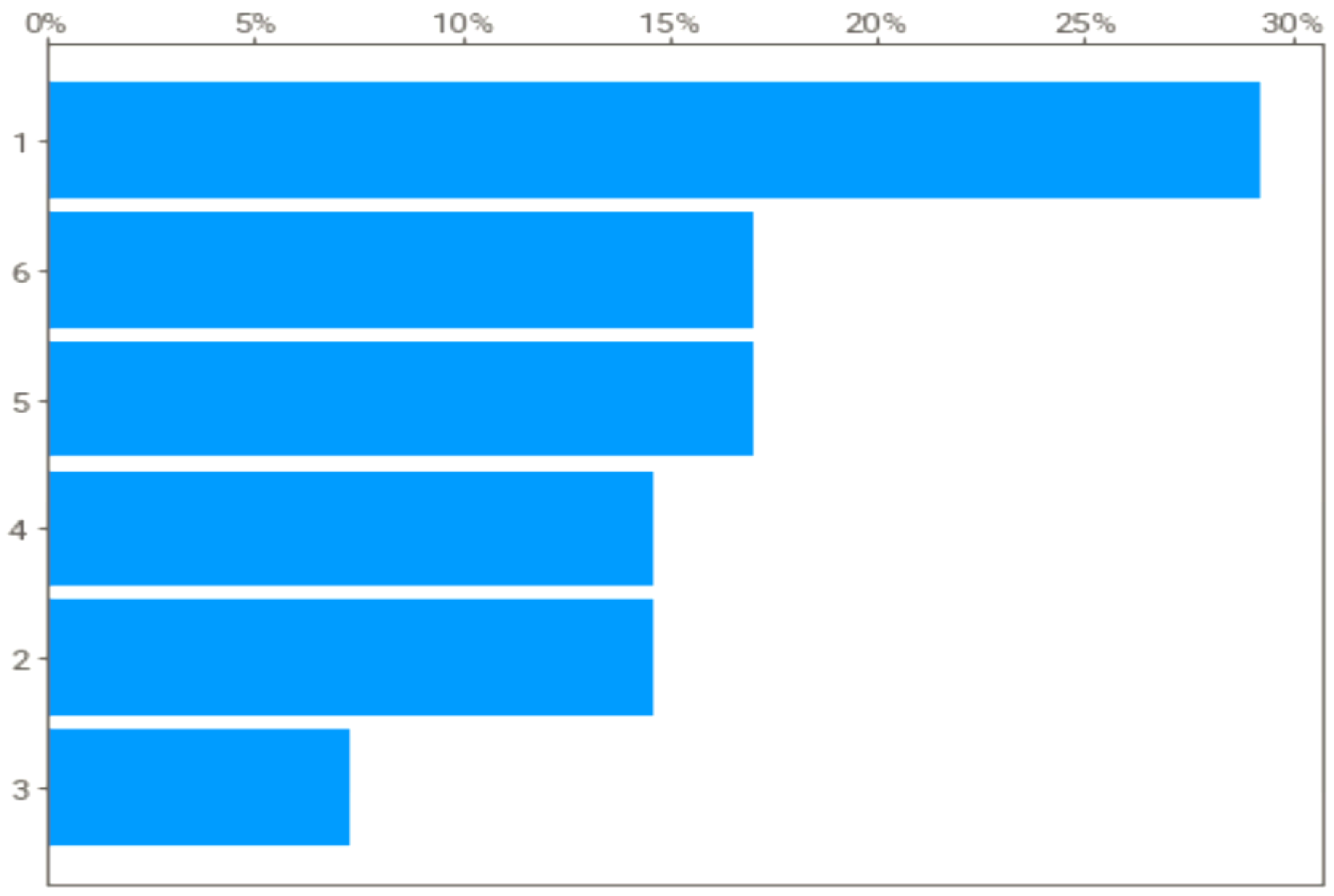
DATASET



TOP CATEGORIES

2	4,337	27%
1	3,727	23%
6	3,680	23%
5	2,752	17%
4	1,094	7%
3	374	2%
-1	36	0%
ALL	16,000	100%

FRAUD



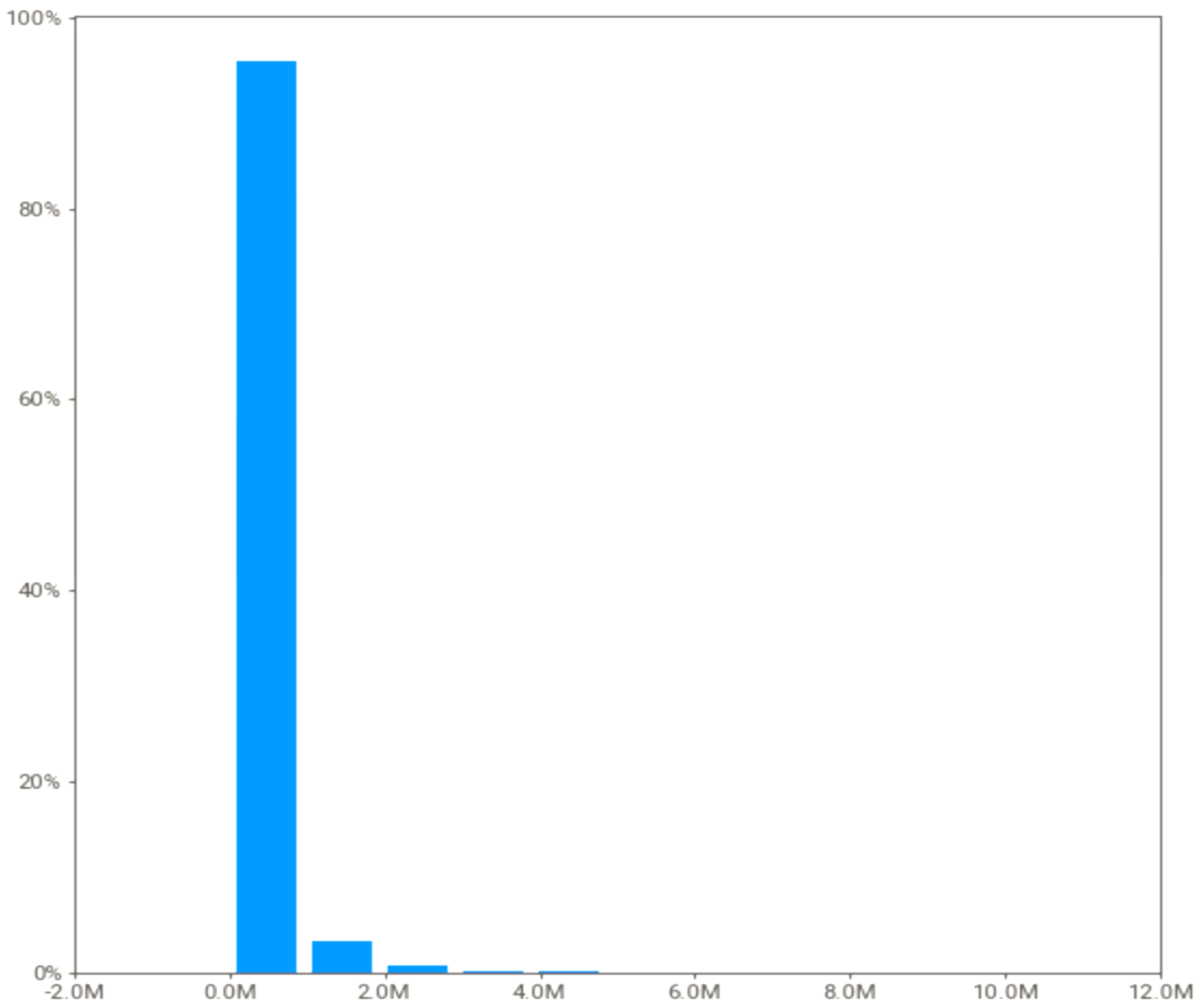
TOP CATEGORIES

1	12	29%
6	7	17%
5	7	17%
4	6	15%
2	6	15%
3	3	7%
ALL	41	100%

15. C14

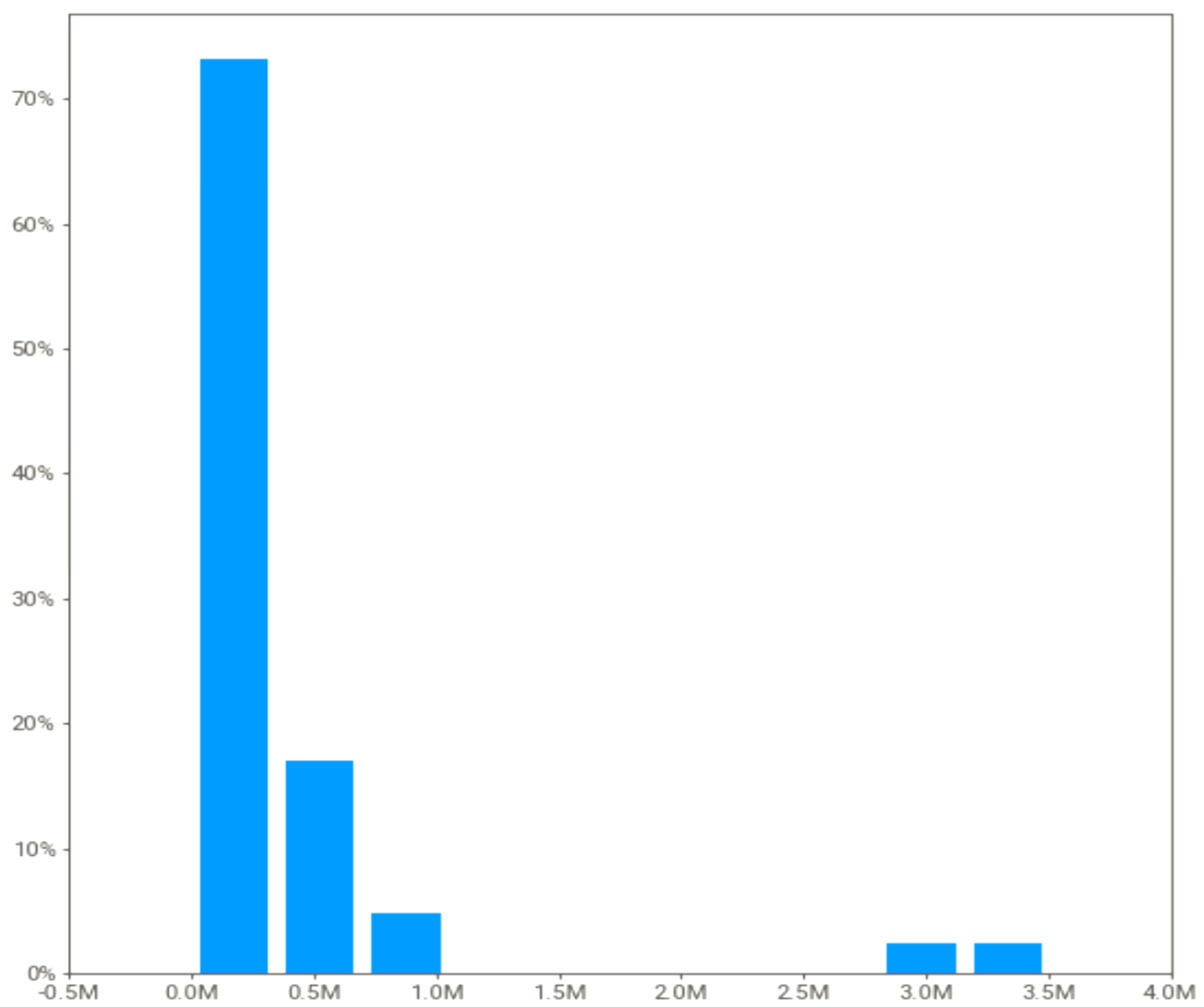
- Case 0
  - Dataset : 6,006 / 38%
  - Fraud : 23 / 56%
- Q. Outlier Handling?
- C14 == 0 & C15==0
  - Dataset : 4,073 / 25%
  - Fraud : 12 / 29%

DATASET



	count	mean	std	min	50%	max

FRAUD

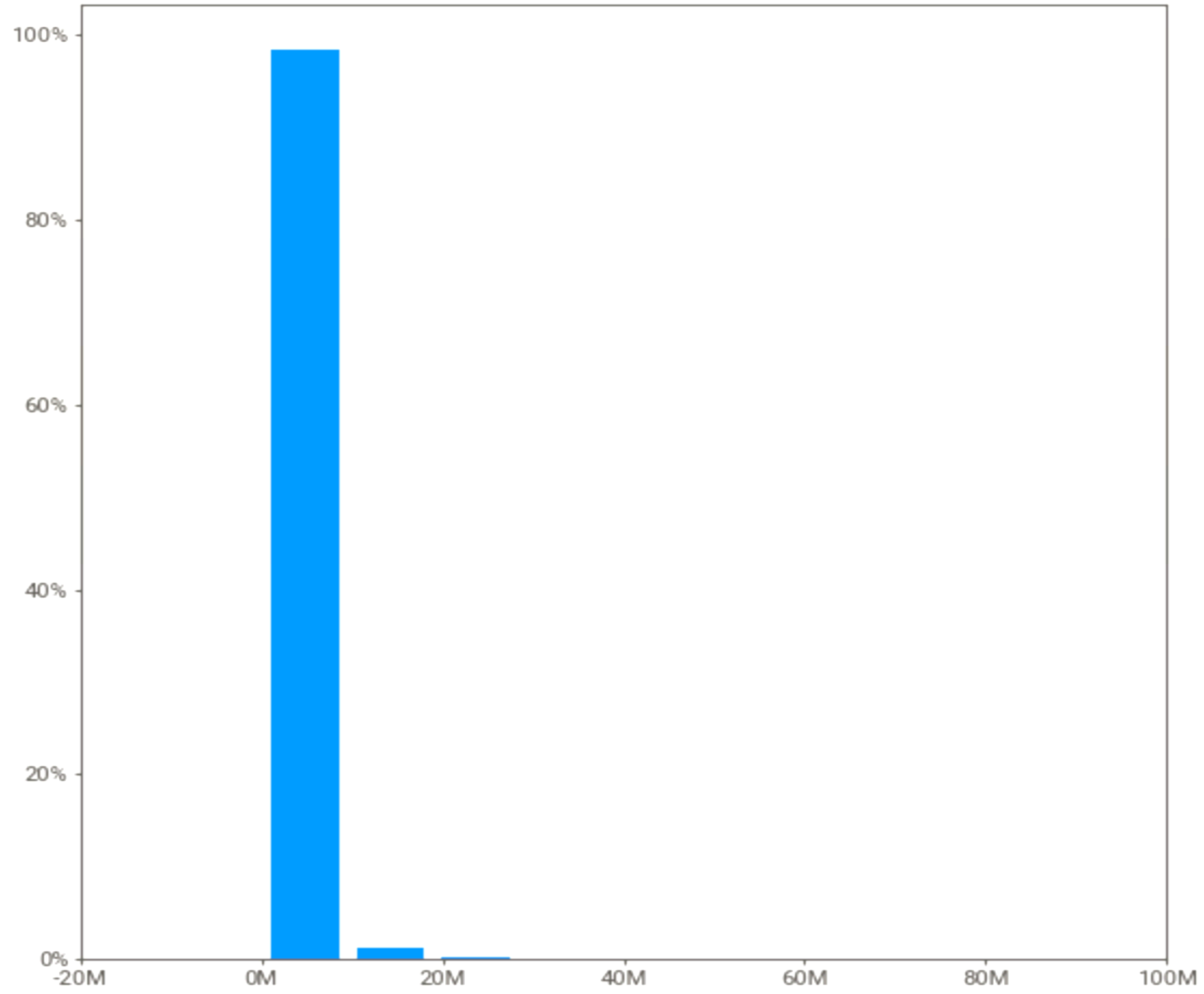


	count	mean	std	min	50%	max

16. C15

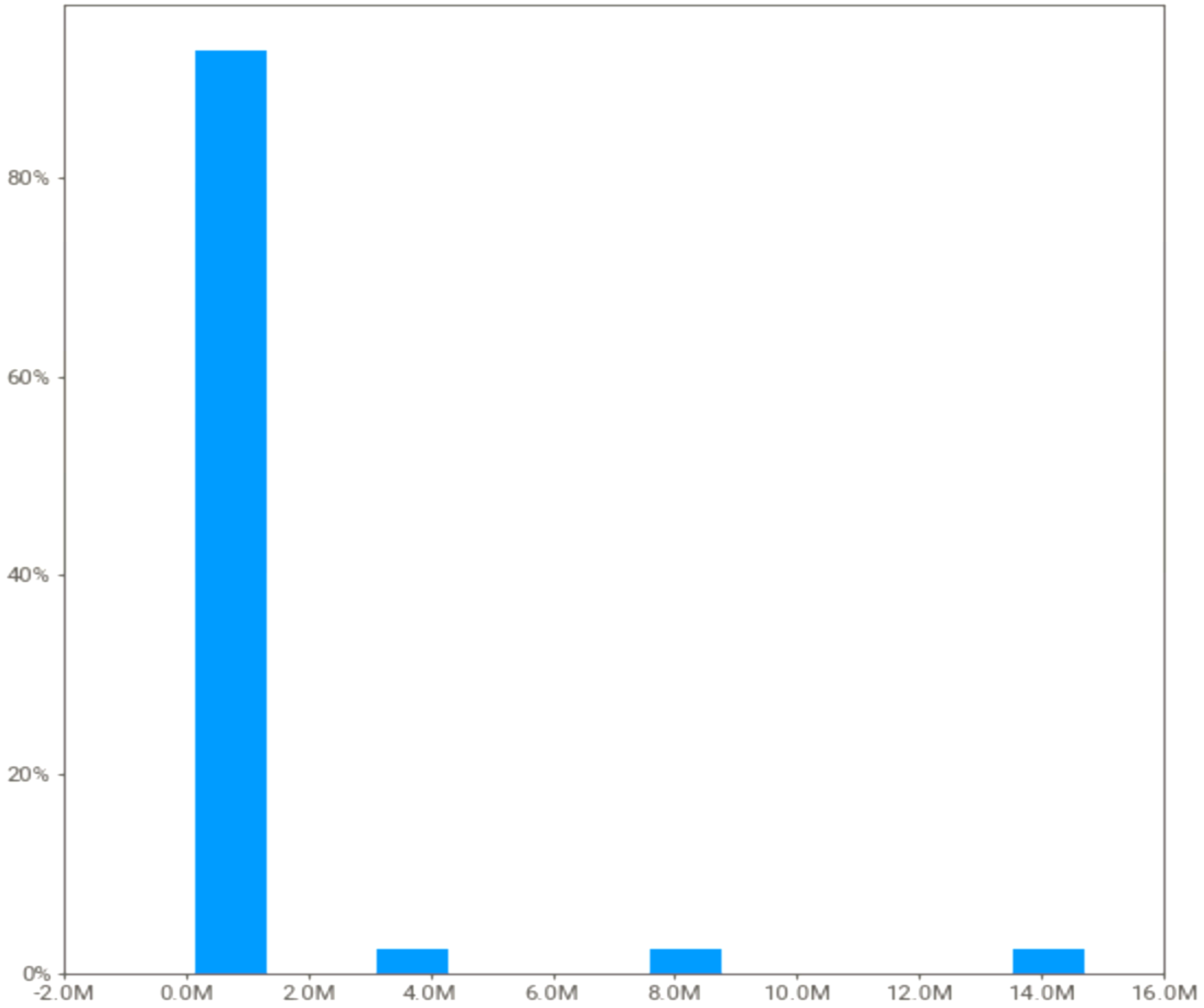
- Case 0
  - Dataset : 10,424 / 65%
  - Fraud : 21 / 51%
- Q. Outlier Handling?
-

DATASET



count	mean	std	min	50%	max

FRAUD

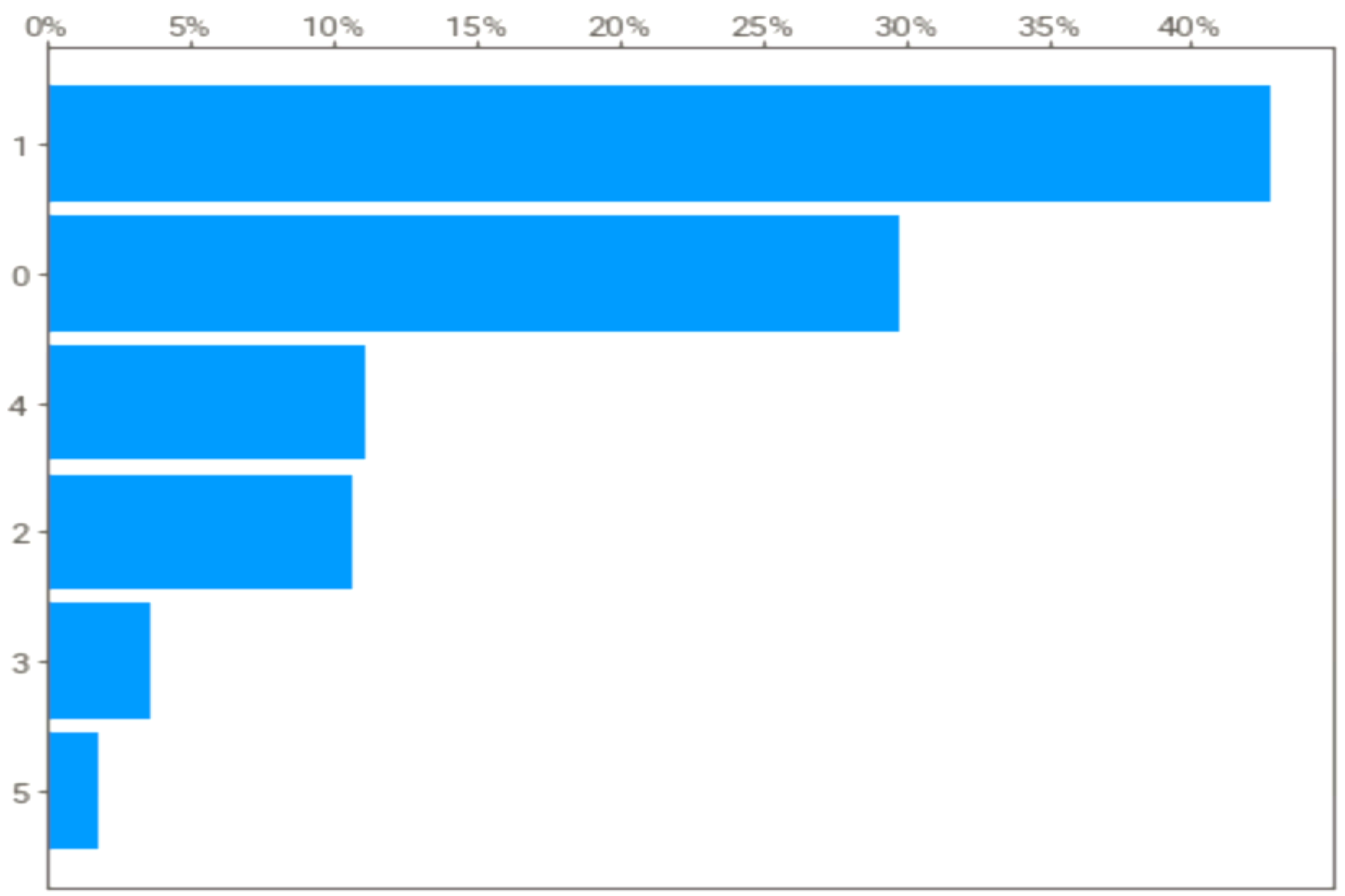


count	mean	std	min	50%	max

17. C16 -> ONE_HOT_ENCODING

- 0
- 1
- 2
- 3
- 4
- 5
- Different Ratio

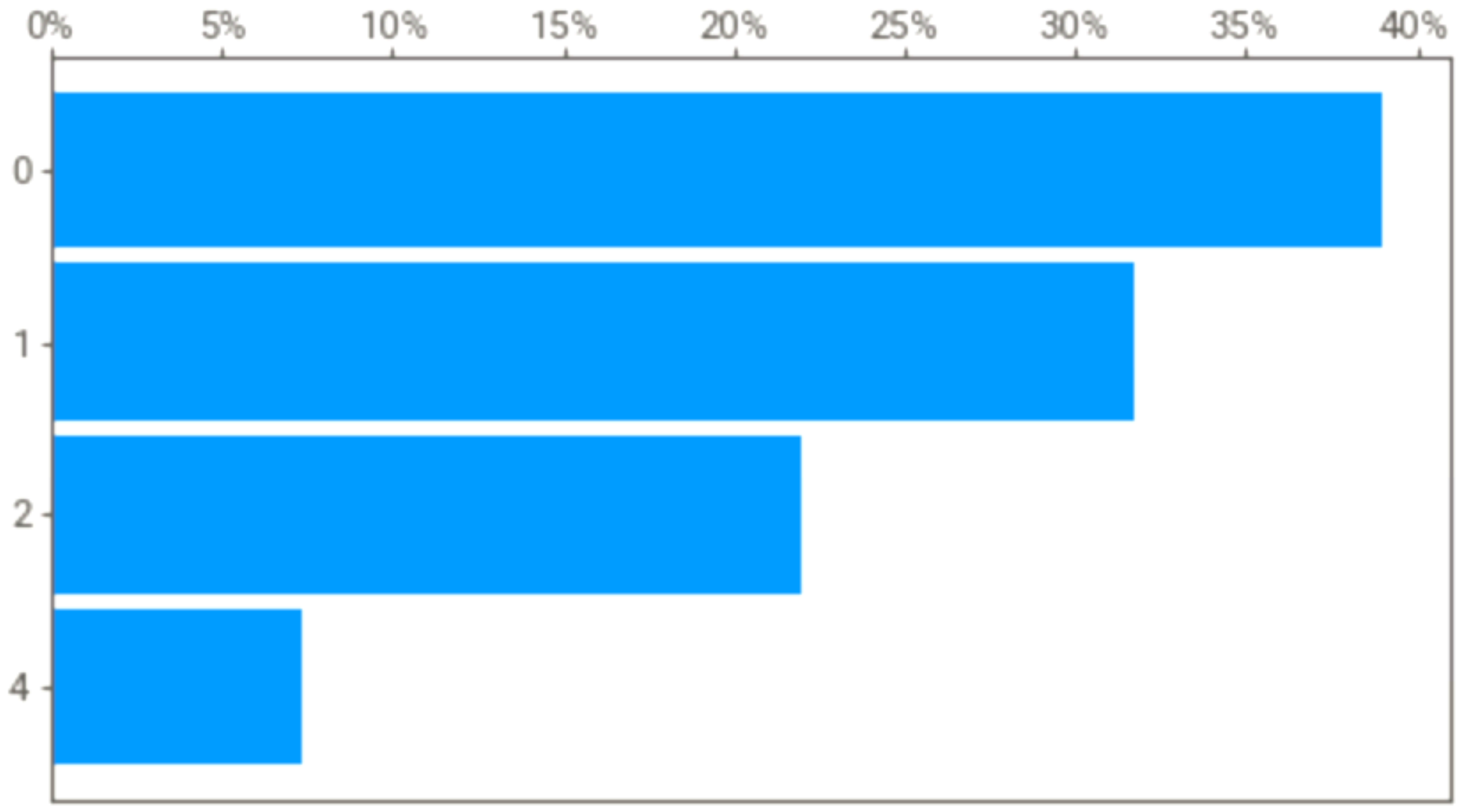
DATASET



TOP CATEGORIES

1	6,850	43%
0	4,778	30%
4	1,786	11%
2	1,710	11%
3	588	4%
5	288	2%
ALL	16,000	100%

FRAUD



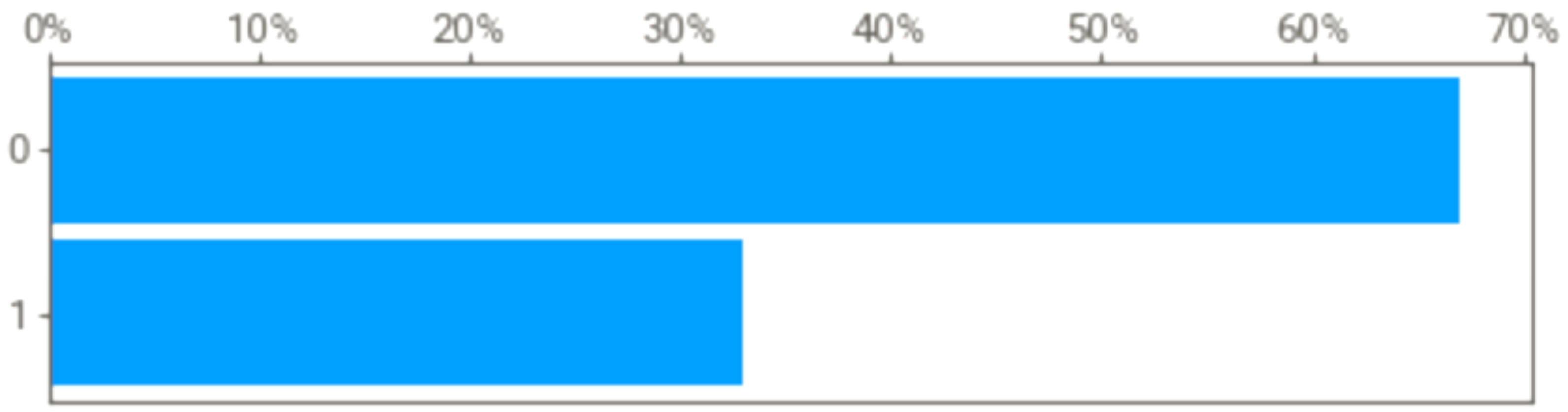
TOP CATEGORIES

0	16	39%
1	13	32%
2	9	22%
4	3	7%
ALL	41	100%

18. C17

- Different Ratio
- Ratio of Case 0
  - Dataset 67%
  - Fraud 56%

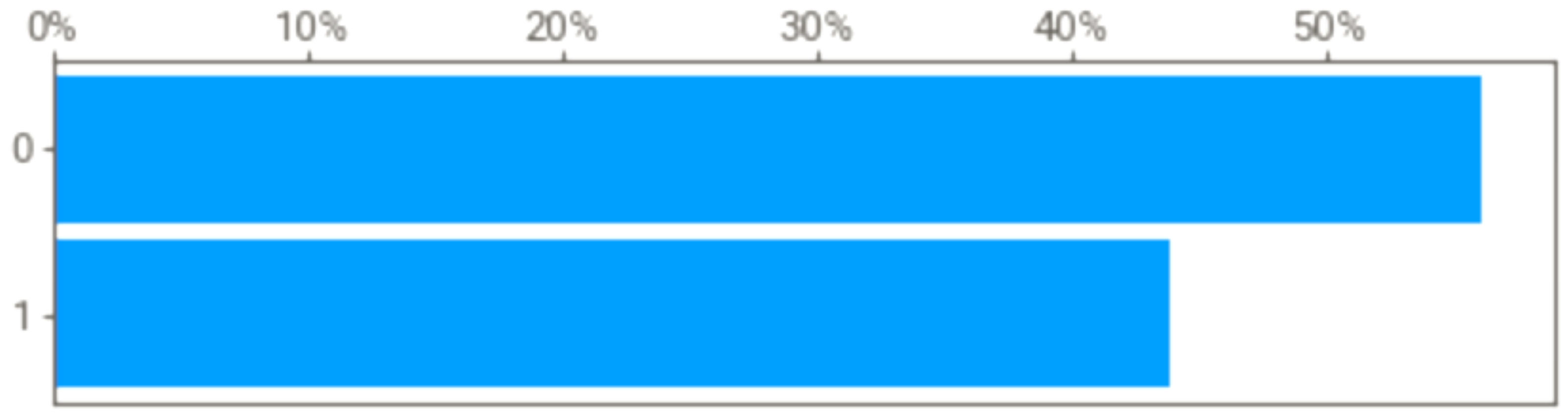
DATASET



TOP CATEGORIES

0	10,731	67%
1	5,269	33%
ALL	16,000	100%

FRAUD



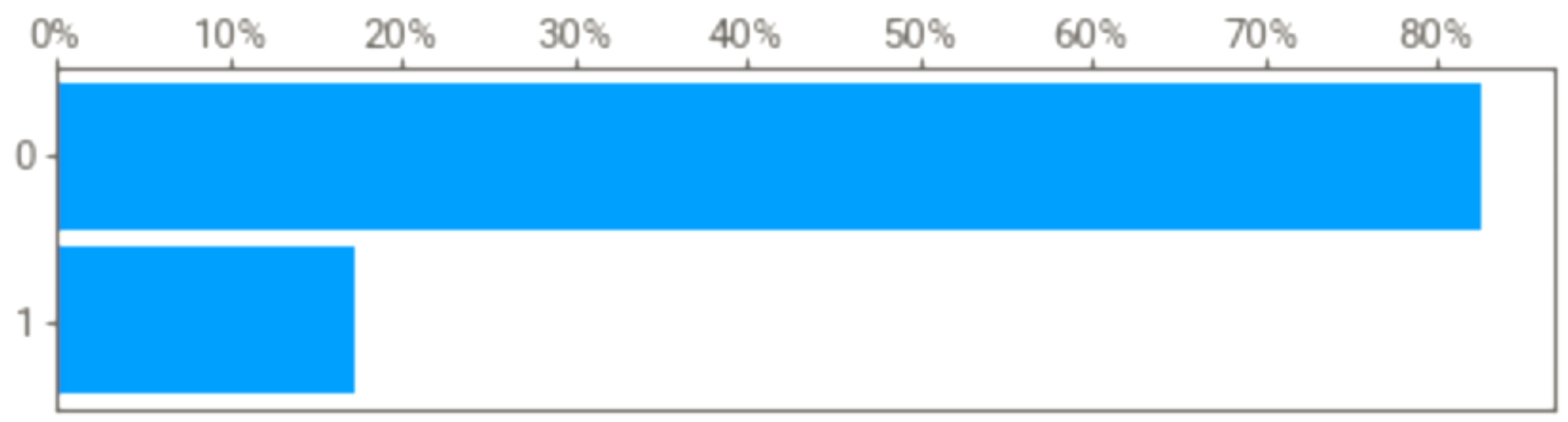
TOP CATEGORIES

0	23	56%
1	18	44%
ALL	41	100%

19. C18

- Similar Ratio

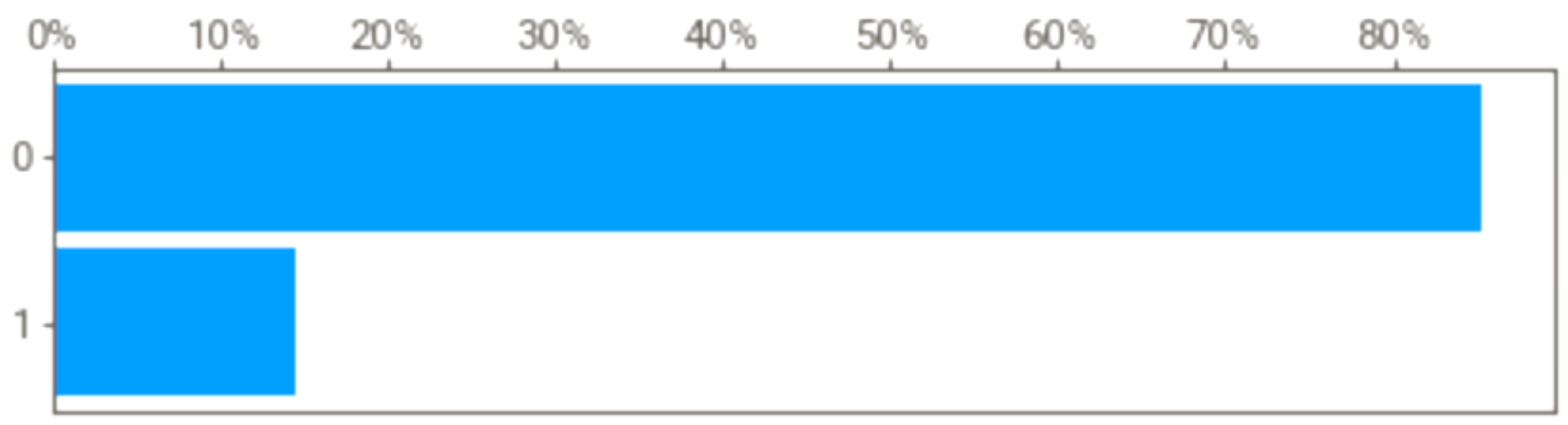
DATASET



TOP CATEGORIES

0	13,228	83%
1	2,772	17%
ALL	16,000	100%

FRAUD



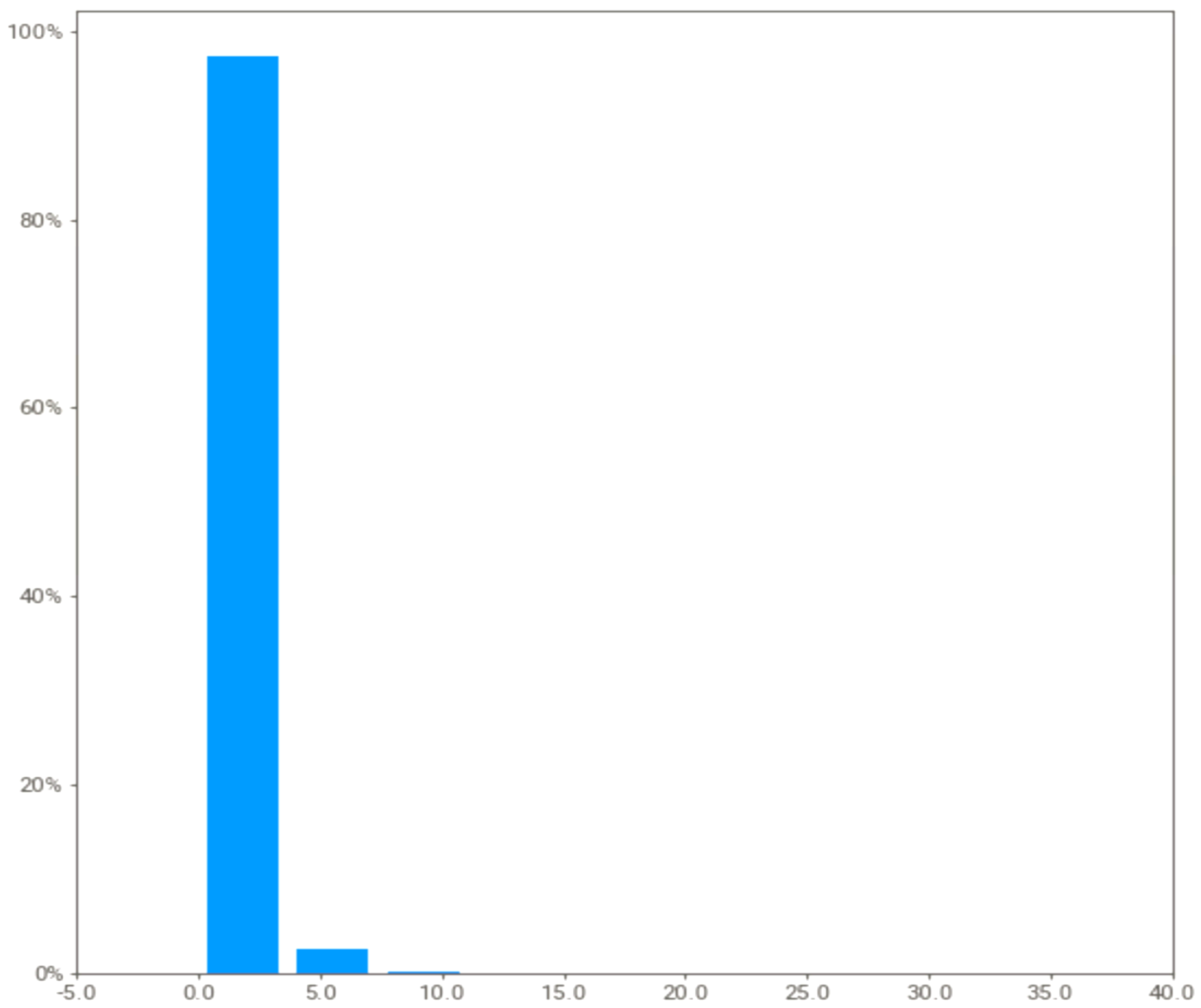
TOP CATEGORIES

0	35	85%
1	6	15%
ALL	41	100%

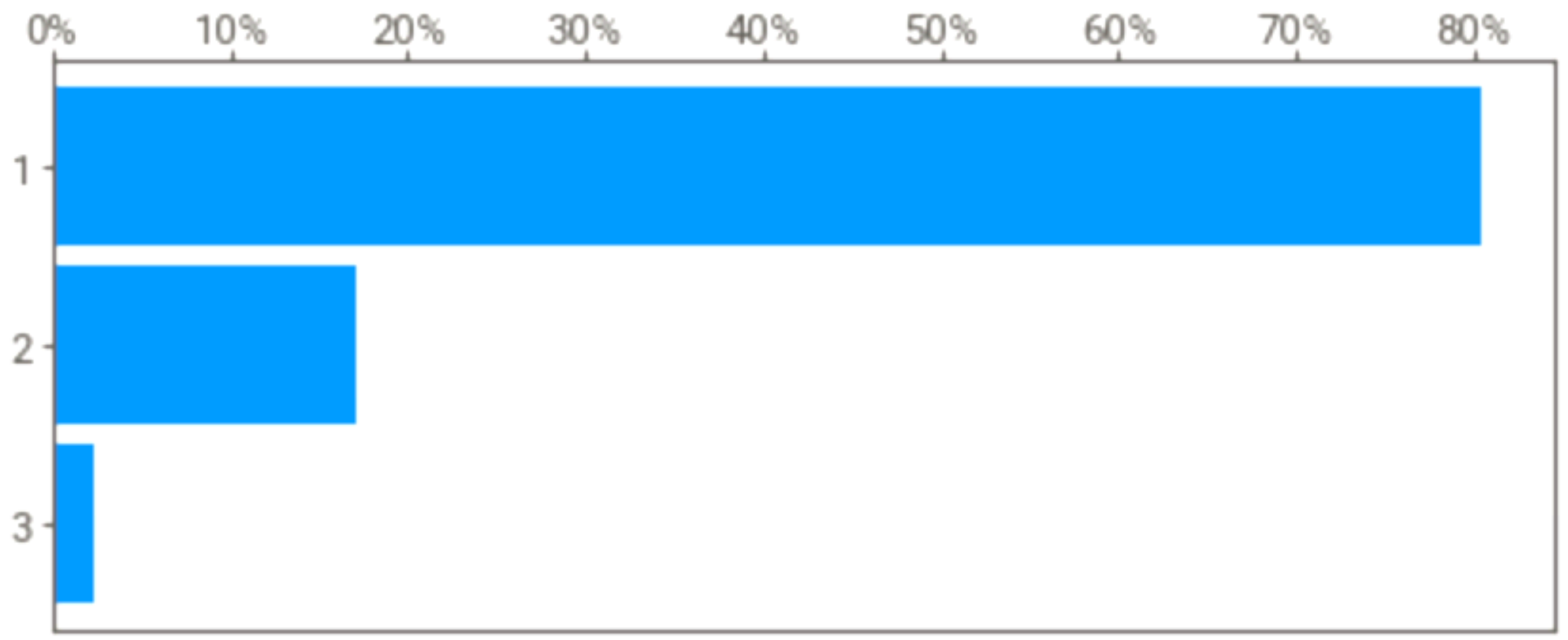
20. C19

- Dataset: Case 1,2,3 = 95%
- Fraud: Case 1,2,3 = 100%
- Q. Outlier Handling?

DATASET



FRAUD



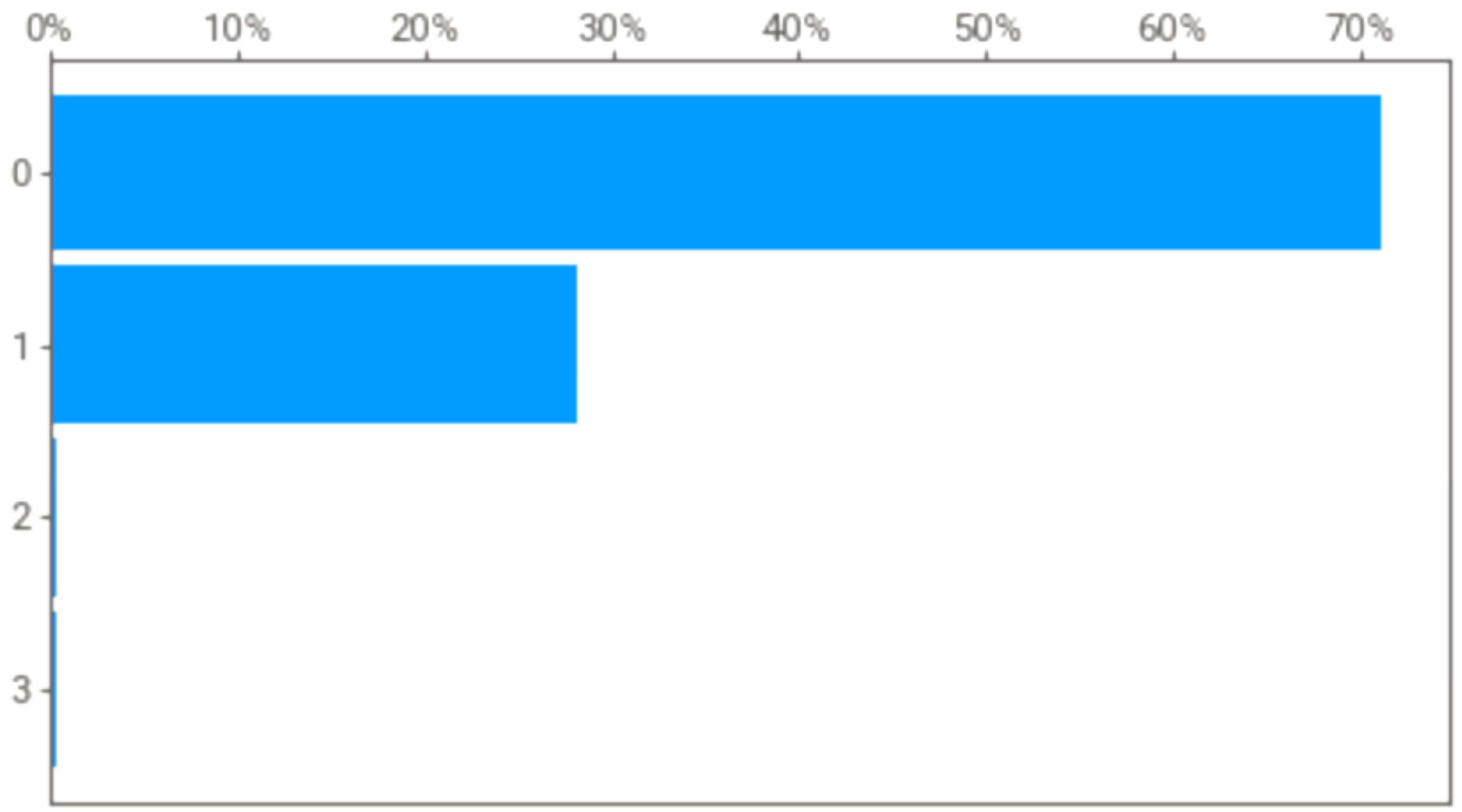
TOP CATEGORIES

1	33	80%
2	7	17%
3	1	2%
ALL	41	100%

21. C20 -> ONE_HOT_ENCODING

- 0
- 1
- 2
- 3
- Fraud
  - 0 & 1 : 95%
  - 2 & 3 : one case each

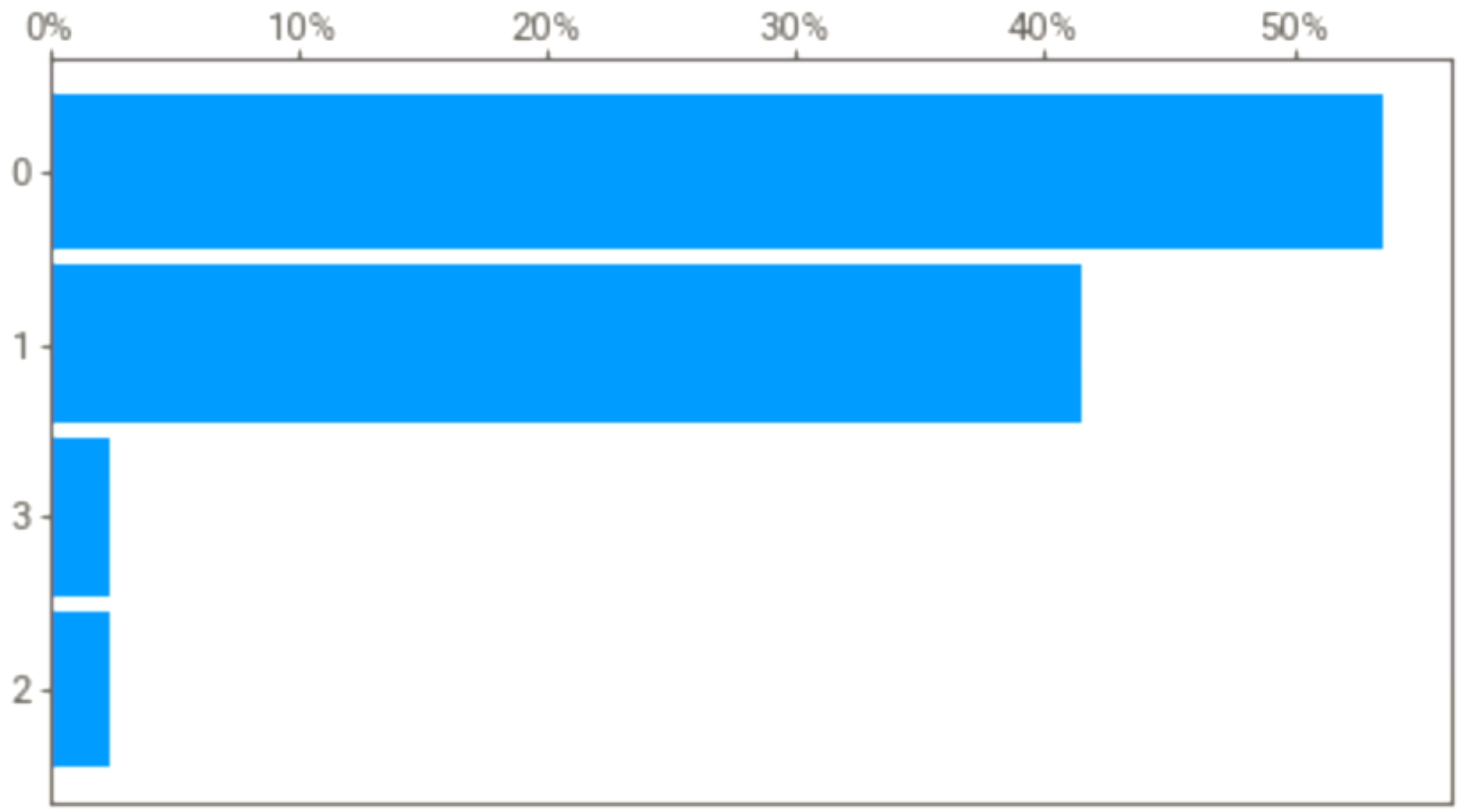
DATASET



TOP CATEGORIES

0	11,384	71%
1	4,496	28%
2	67	0%
3	53	0%
ALL	16,000	100%

FRAUD



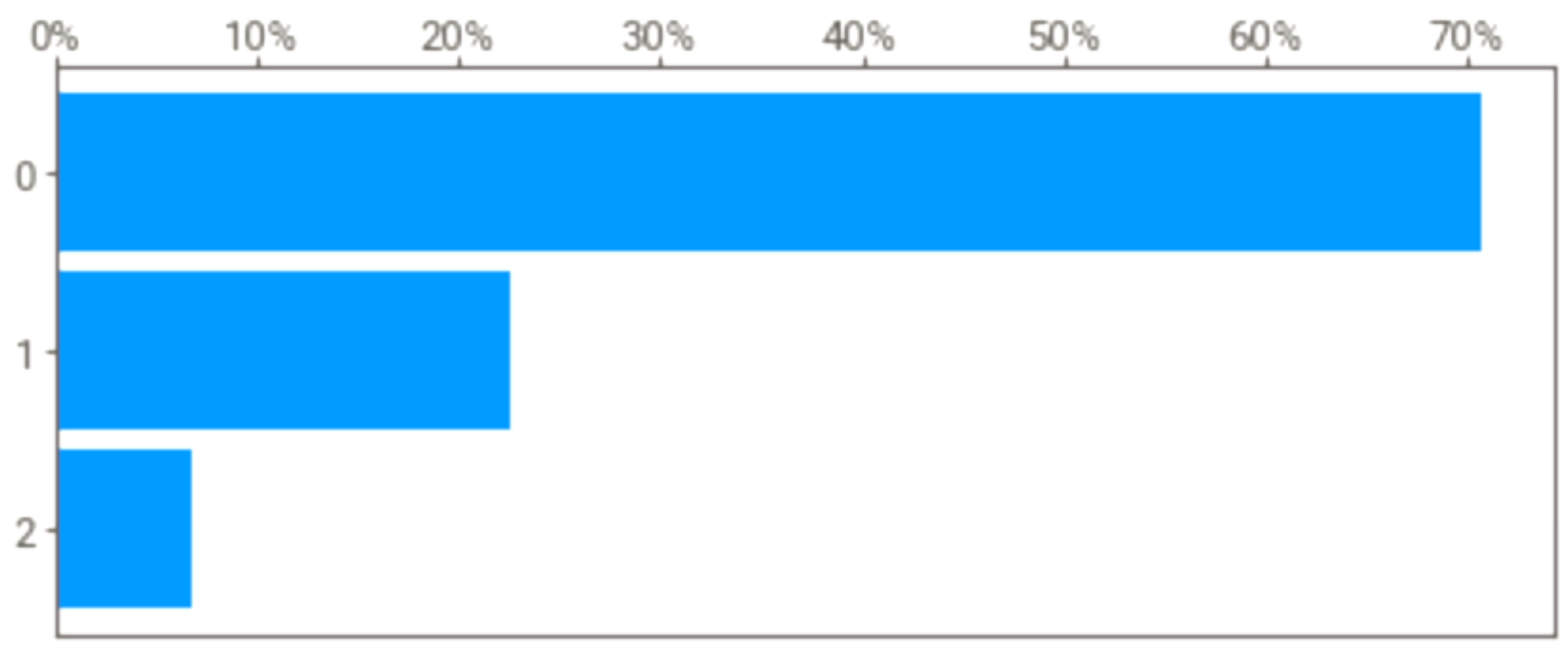
TOP CATEGORIES

0	22	54%
1	17	41%
3	1	2%
2	1	2%
ALL	41	100%

22. C21 -> ONE_HOT_ENCODING

- 0
- 1
- 2

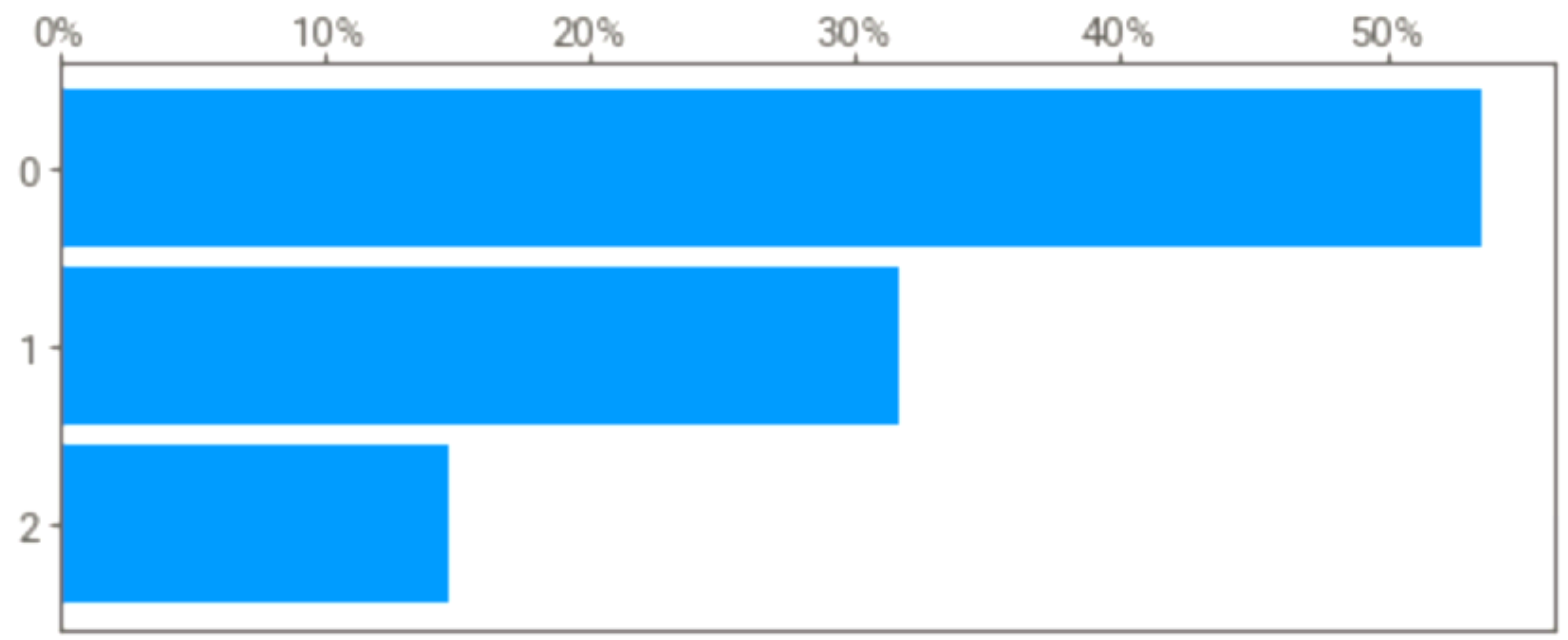
DATASET



TOP CATEGORIES

0	11,325	71%
1	3,598	22%
2	1,077	7%
ALL	16,000	100%

FRAUD



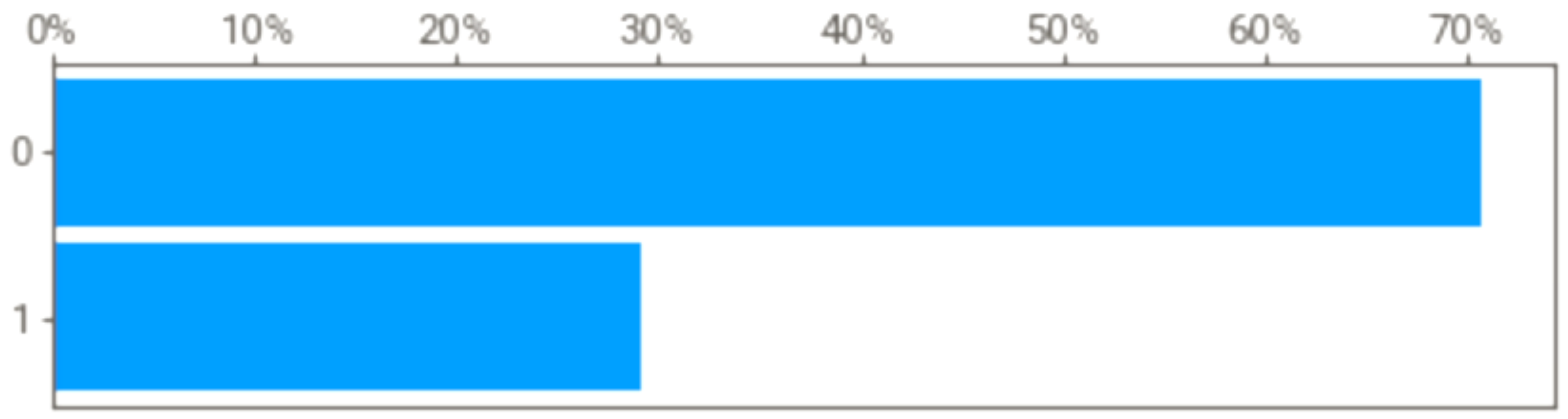
TOP CATEGORIES

0	22	54%
1	13	32%
2	6	15%
ALL	41	100%

23. C22

- 0
- 1
- 2

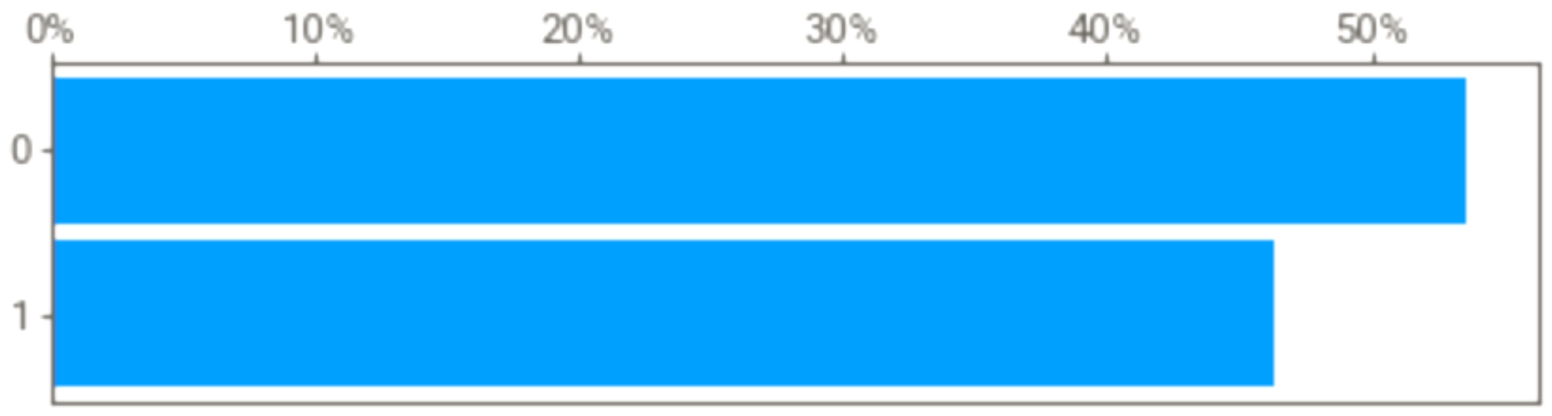
DATASET



TOP CATEGORIES

0	11,328	71%
1	4,672	29%
ALL	16,000	100%

FRAUD



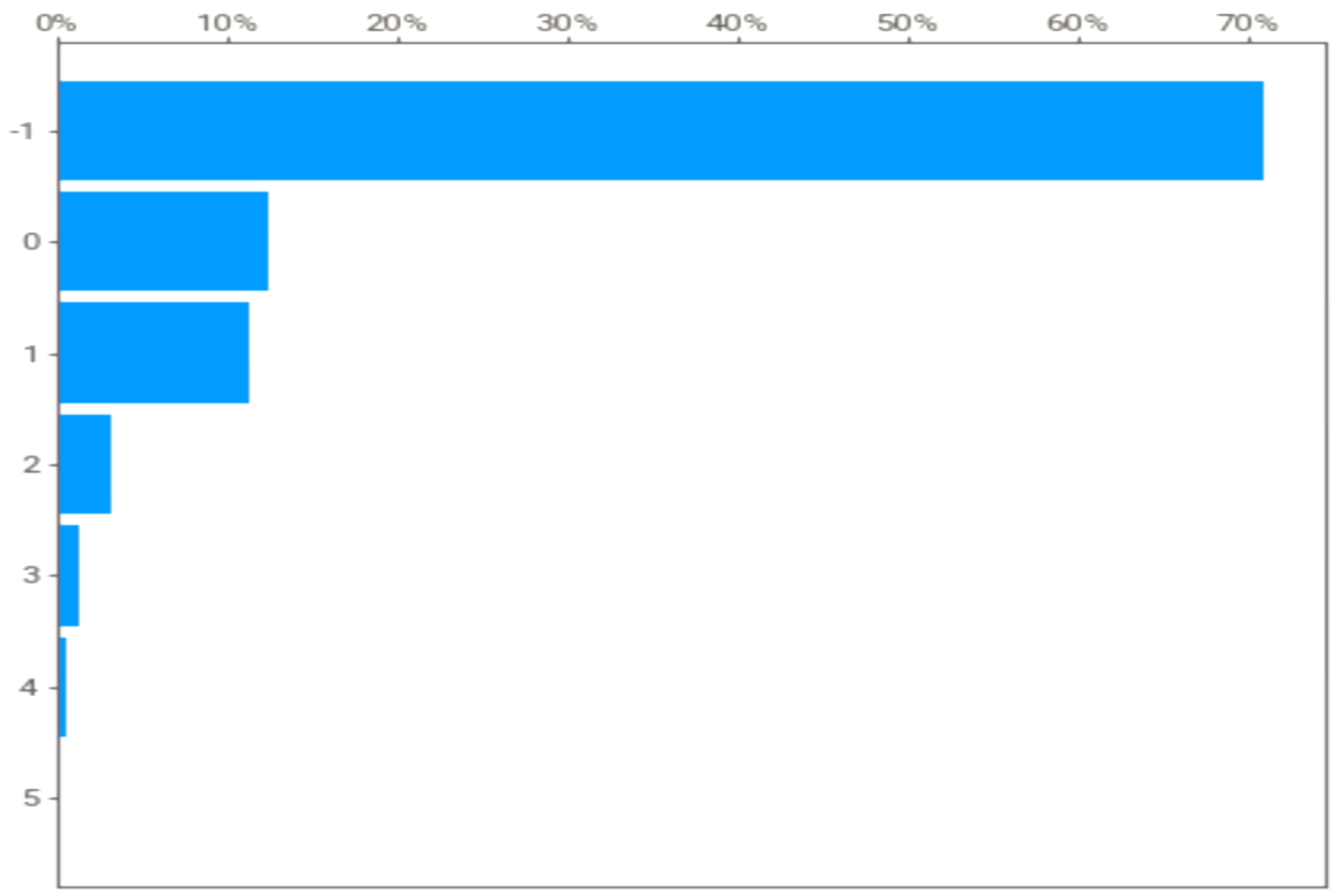
TOP CATEGORIES

0	22	54%
1	19	46%
ALL	41	100%

24. C23

- -1
- 5
- Fraud : -1 or 0 = 66%
-

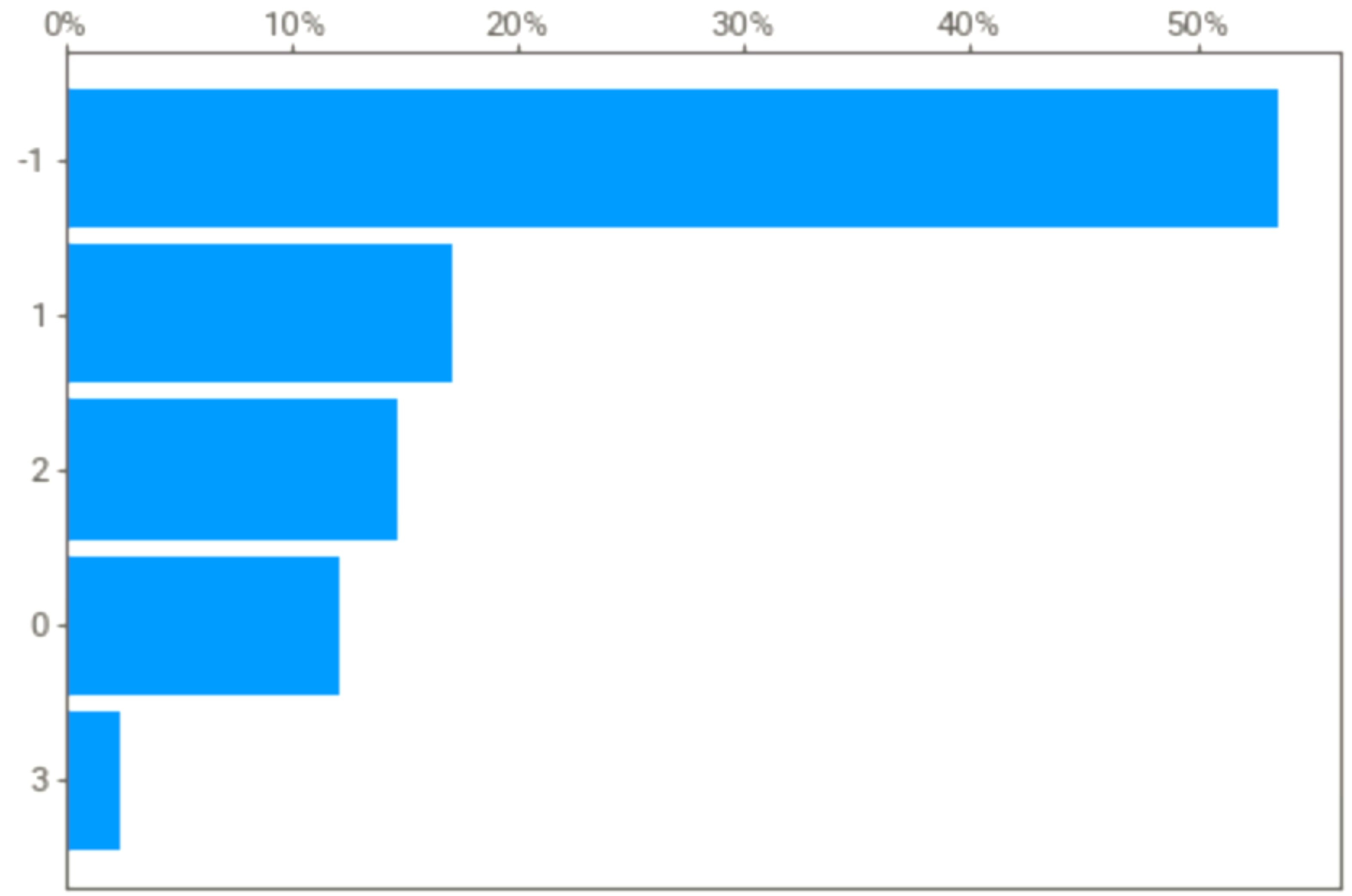
DATASET



TOP CATEGORIES

-1	11,358	71%
0	1,997	12%
1	1,804	11%
2	502	3%
3	222	1%
4	91	1%
5	26	0%
ALL	16,000	100%

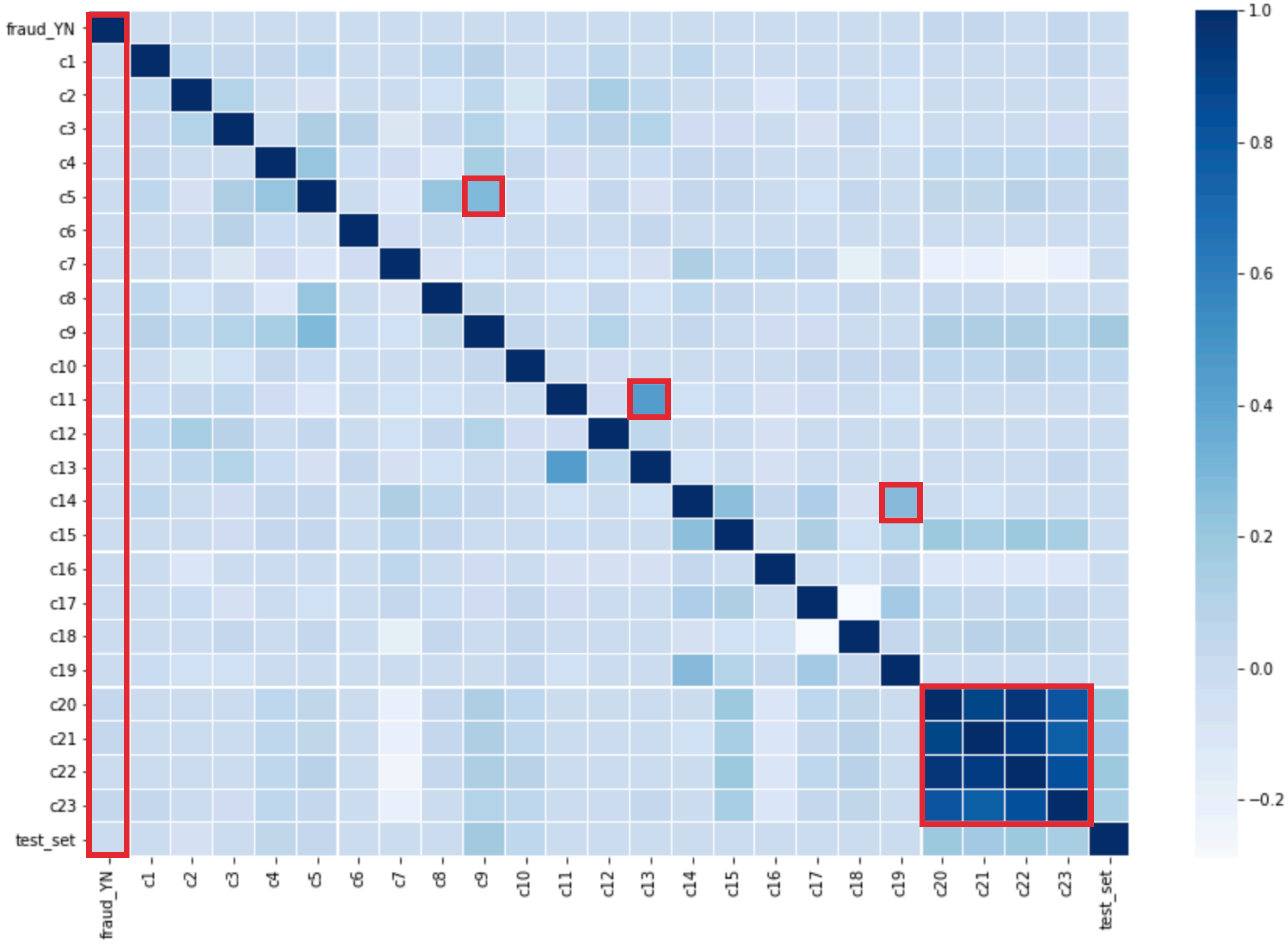
FRAUD



TOP CATEGORIES

-1	22	54%
1	7	17%
2	6	15%
0	5	12%
3	1	2%
ALL	41	100%

- Fraud_YN
- No significant correlations (~0.02)



- C11 & C13 (0.43)
- C5 & C9 (0.29)
- C14 & C19 (0.27)
- C20, C21, C22, C23 (high correlations)

PRE-PROCESSING 1

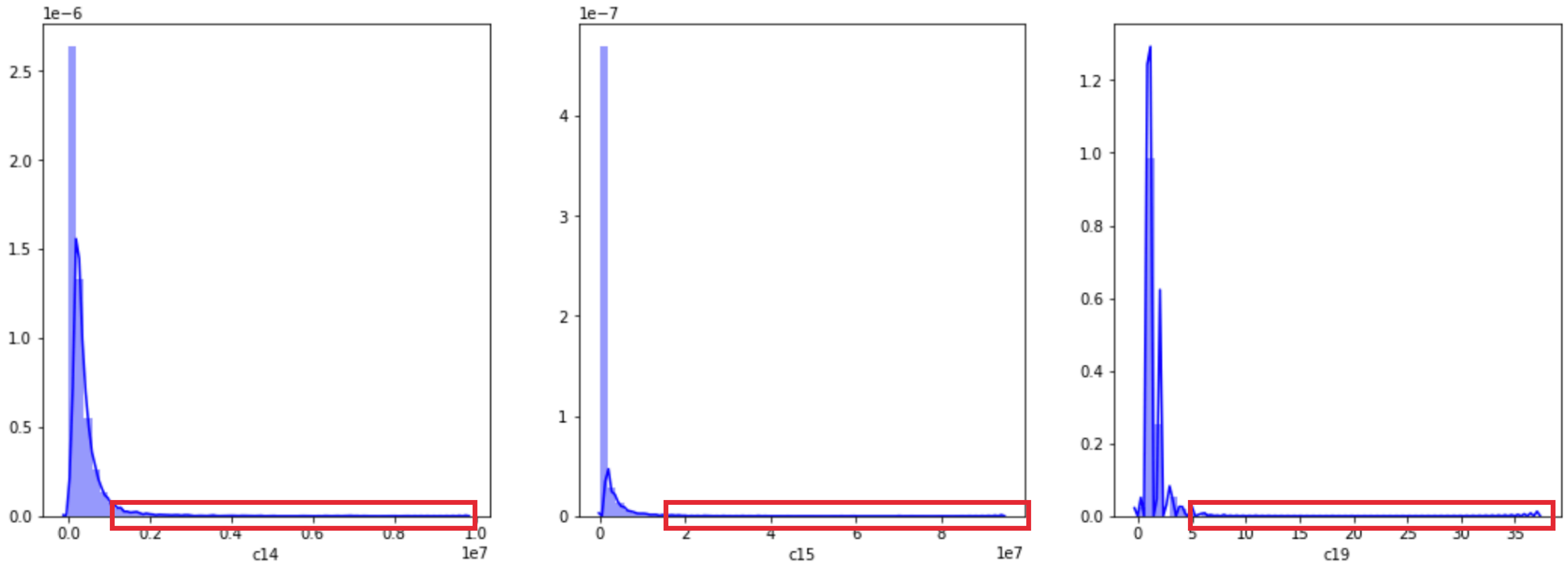
ONE_HOT_ENCODING, OUTLIER, RESAMPLING

CATEGORICAL DATA : APPLY ONE-HOT-ENCODING

- C1
- C3
- C11
- C12
- C13
- C16
- C20
- C21

c16_1	c16_2	c16_3	c16_4	c16_5	c20_0	c20_1	c20_2	c20_3	c21_0	c21_1	c21_2
1	0	0	0	0	1	0	0	0	1	0	0
1	0	0	0	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	0	0	0	1	0
0	0	1	0	0	1	0	0	0	1	0	0
1	0	0	0	0	1	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...
1	0	0	0	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	1	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	0	0	1	0	0

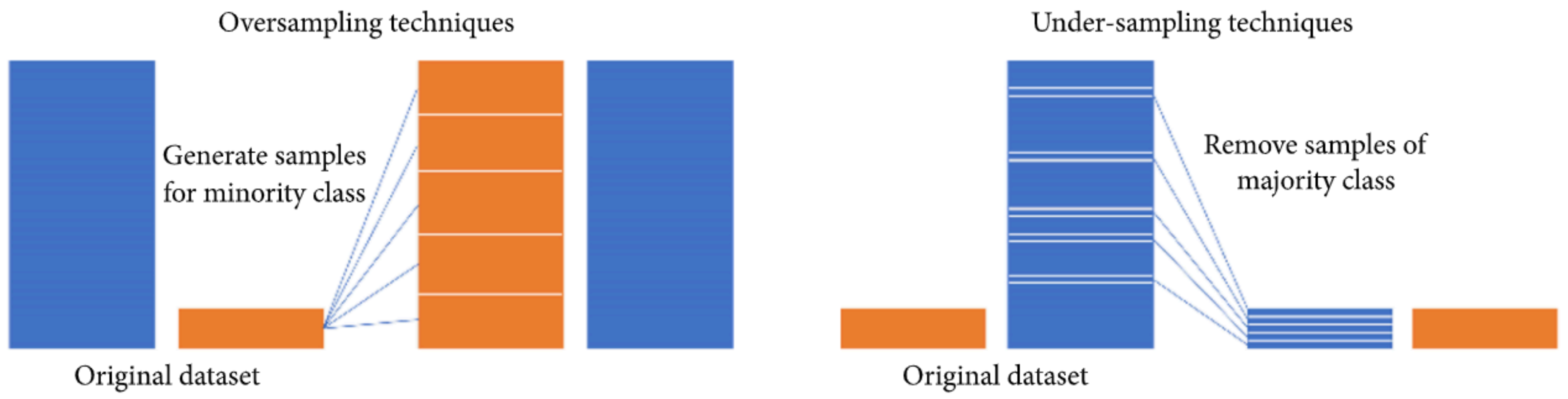
OUTLIERS IN C14, C15, C19



PRE-PROCESSING 1 : OUTLIERS CHECK (SCALER)

- Standard Scaler (X)
  - May not be optimal if feature is not normally distributed
  - Cannot guarantee balanced feature scales in the presence of outliers
- Min Max Scaler (X)
  - Effective if the distribution is not normal or standard deviation is small
  - Sensitive to the presence of outliers
- **Robust Scaler (O)**
  - Centering and scaling statistics of this scaler based on percentiles
  - Not influenced by a few number of very large marginal outliers

- Re-sampling techniques used in our dataset due to strong between-class imbalance



Source : https://www.hindawi.com/journals/complexity/2019/8460934/fig1/

MODELING 1

USING ROBUSTSCALER

BASELINE

- Logistic Regression and SVM were not able to detect any fraud cases in both train and test datasets whereas RandomForest seemed overfitted.

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.99736	0.0	0.0	0.997757	0.0	0.0
1	SupportVectorMachine	0.99736	0.0	0.0	0.997757	0.0	0.0
2	RandomForest	1.00000	1.0	1.0	0.997757	0.0	0.0

GRID SEARCH (SCORING = “RECALL”)

- Although we expected better results from GridSearch, the performance of Random Forest in train set depreciated and the performances of Logistic Regression and SVM remained the same

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.99736	0.0	0.0	0.997757	0.0	0.0
1	SupportVectorMachine	0.99736	0.0	0.0	0.997757	0.0	0.0
2	RandomForest	0.99736	0.0	0.0	0.997757	0.0	0.0

OVER-SAMPLING (SMOTE)

```
X_train.shape, y_train.shape
```

```
((12879, 53), (12879,))
```

```
X_train_over.shape, y_train_over.shape
```

```
((25690, 53), (25690,))
```

- Although data was successfully over-sampled, over-sampling led to overfitting.
- Potential Solution:
  - Try another over-sampling method!

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.998677	0.999844	0.997509	0.997116	0.0	0.0
1	SupportVectorMachine	0.998443	0.999376	0.997509	0.997116	0.0	0.0
2	RandomForest	0.996497	0.994648	0.998365	0.992951	0.0	0.0

OVER-SAMPLING (RANDOM OVERSAMPLING)

- Random_state = 3

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.827092	0.793176	0.884936	0.685037	0.004065	0.571429
1	SupportVectorMachine	0.839665	0.780904	0.944258	0.660045	0.003766	0.571429
2	RandomForest	0.990308	0.980984	1.000000	0.970522	0.000000	0.000000

- Random_state = 11

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.827287	0.794522	0.882912	0.690804	0.004141	0.571429
1	SupportVectorMachine	0.831102	0.758039	0.972674	0.618392	0.003356	0.571429
2	RandomForest	0.996730	0.993503	1.000000	0.987824	0.000000	0.000000

- Random_state = 22

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.828883	0.795647	0.885091	0.690804	0.004141	0.571429
1	SupportVectorMachine	0.844492	0.788086	0.942390	0.679590	0.003003	0.428571
2	RandomForest	0.989763	0.979936	1.000000	0.967959	0.000000	0.000000

- Random Oversampling improved performances of Logistic Regression and SVM.

- However, Random Forest did not perform well in both over-sampling methods.

UNDER-SAMPLING

- Random Undersampling

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.867647	0.878788	0.852941	0.522269	0.002681	0.571429
1	SupportVectorMachine	0.955882	0.969697	0.941176	0.567767	0.002963	0.571429
2	RandomForest	0.852941	0.852941	0.852941	0.492470	0.002524	0.571429

- NearMiss (Version 3)

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	1.000000	1.0	1.000000	0.201538	0.001604	0.571429
1	SupportVectorMachine	0.941176	1.0	0.882353	0.192887	0.001587	0.571429
2	RandomForest	0.926471	1.0	0.852941	0.238706	0.001683	0.571429

- Condensed NN

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.859551	0.680000	0.500000	0.829221	0.005639	0.428571
1	SupportVectorMachine	0.792135	0.434783	0.294118	0.790131	0.004587	0.428571
2	RandomForest	0.876404	0.700000	0.617647	0.810958	0.005093	0.428571

- We applied three different under-sampling methods.
- However, under-sampling did not improve performances of our models.

DILEMMA & SOLUTION

- Yet outputs of over-sampling and under-sampling may be reasonable, all of results either show low performances on test recall or have considerable difference between train and test precision which may imply overfitting.
- Therefore, we decided to apply ensemble models specialized in handling imbalanced data.

BALANCED RANDOM FOREST CLASSIFIER

- A balanced random forest randomly under-samples each boostrap sample to balance it

EASY ENSEMBLE CLASSIFIER

- An ensemble of AdaBoost learners trained on different balanced boostrap samples
- Balancing achieved by random under-sampling

RUSBOOST CLASSIFIER

- Random under-sampling integrated in the learning of AdaBoost
- During learning, the problem of class balancing is alleviated by random under-sampling the sample at each iteration of the boosting algorithm.

ENSEMBLE WITH RESAMPLING

- Since the difference between recalls of RUSBoost in train/test is significant, we decided to exclude RUSBoost in our modeling.
- Compared to Balanced Random Forest, Easy Ensemble performed better in every score.
- Hence, we chose EasyEnsemble as our main ensemble model.

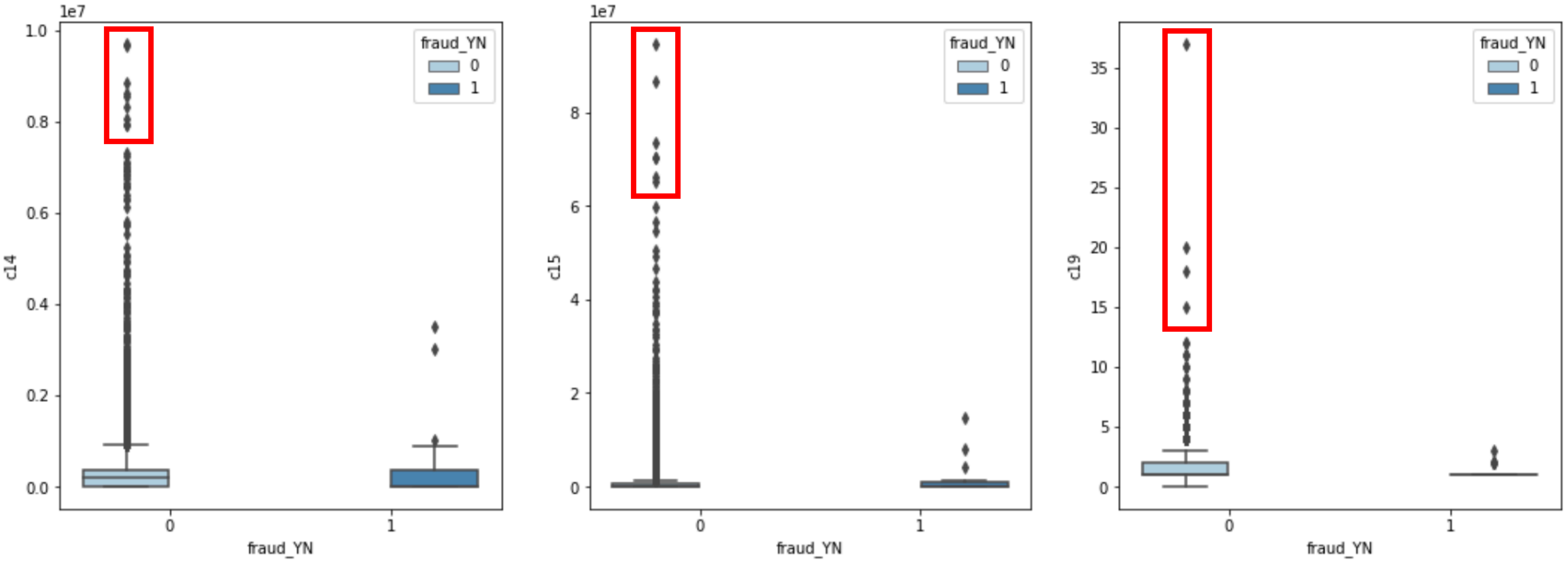
	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	BalancedRandomForest	0.576054	0.005829	0.941176	0.470682	0.001817	0.428571
1	EasyEnsemble	0.677848	0.008128	1.000000	0.521948	0.002679	0.571429
2	RUSBoost	0.765976	0.009223	0.823529	0.696892	0.001063	0.142857

PRE-PROCESSING 2

USING MIN-MAX SCALER

AFTER OUTLIER REMOVAL

OUTLIERS IN C14, C15, C19 : REMOVED



MODELING 2

USING MIN-MAX SCALER
AFTER OUTLIER REMOVAL

BASELINE

- With hope that outlier removal may increase our models' performances, we proceeded outlier removal. However, there was no significant improvement in the performance.

Before Outlier Removal

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.99736	0.0	0.0	0.997757	0.0	0.0
1	SupportVectorMachine	0.99736	0.0	0.0	0.997757	0.0	0.0
2	RandomForest	1.00000	1.0	1.0	0.997757	0.0	0.0

After Outlier Removal

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.997357	0.0	0.0	0.997755	0.0	0.0
1	SupportVectorMachine	0.996424	0.0	0.0	0.995189	0.0	0.0
2	RandomForest	1.000000	1.0	1.0	0.997755	0.0	0.0

GRID SEARCH (SCORING = “RECALL”)

- Similar to the comparison of baselines, after outlier removal, there was no improvement at all on the performance

Before Outlier Removal

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.99736	0.0	0.0	0.997757	0.0	0.0
1	SupportVectorMachine	0.99736	0.0	0.0	0.997757	0.0	0.0
2	RandomForest	0.99736	0.0	0.0	0.997757	0.0	0.0

After Outlier Removal

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	LogisticRegression	0.997357	0.0	0.0	0.997755	0.0	0.0
1	SupportVectorMachine	0.997357	0.0	0.0	0.997755	0.0	0.0
2	RandomForest	0.997357	0.0	0.0	0.997755	0.0	0.0

COMPARISON BETWEEN MODELING 1 AND 2

- Modeling 1 : Using Robust Scaler

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	BalancedRandomForest	0.576054	0.005829	0.941176	0.470682	0.001817	0.428571
1	EasyEnsemble	0.677848	0.008128	1.000000	0.521948	0.002679	0.571429
2	RUSBoost	0.765976	0.009223	0.823529	0.696892	0.001063	0.142857

- Modeling 2 : Using MinMaxScaler After Outlier Removal

	model name	train accuracy	train precision	train recall	test accuracy	test precision	test recall
0	BalancedRandomForest	0.597170	0.006140	0.941176	0.441309	0.002295	0.571429
1	EasyEnsemble	0.576881	0.006208	1.000000	0.440026	0.002290	0.571429
2	RUSBoost	0.809468	0.006130	0.441176	0.654586	0.002788	0.428571

- Although Balanced Random Forest and Easy Ensemble using MinMaxScaler showed decent performances on recall, their accuracies and precisions were lower than those of EasyEnsemble.
- Hence, we chose **EasyEnsemble** using RobustScaler as our main model.

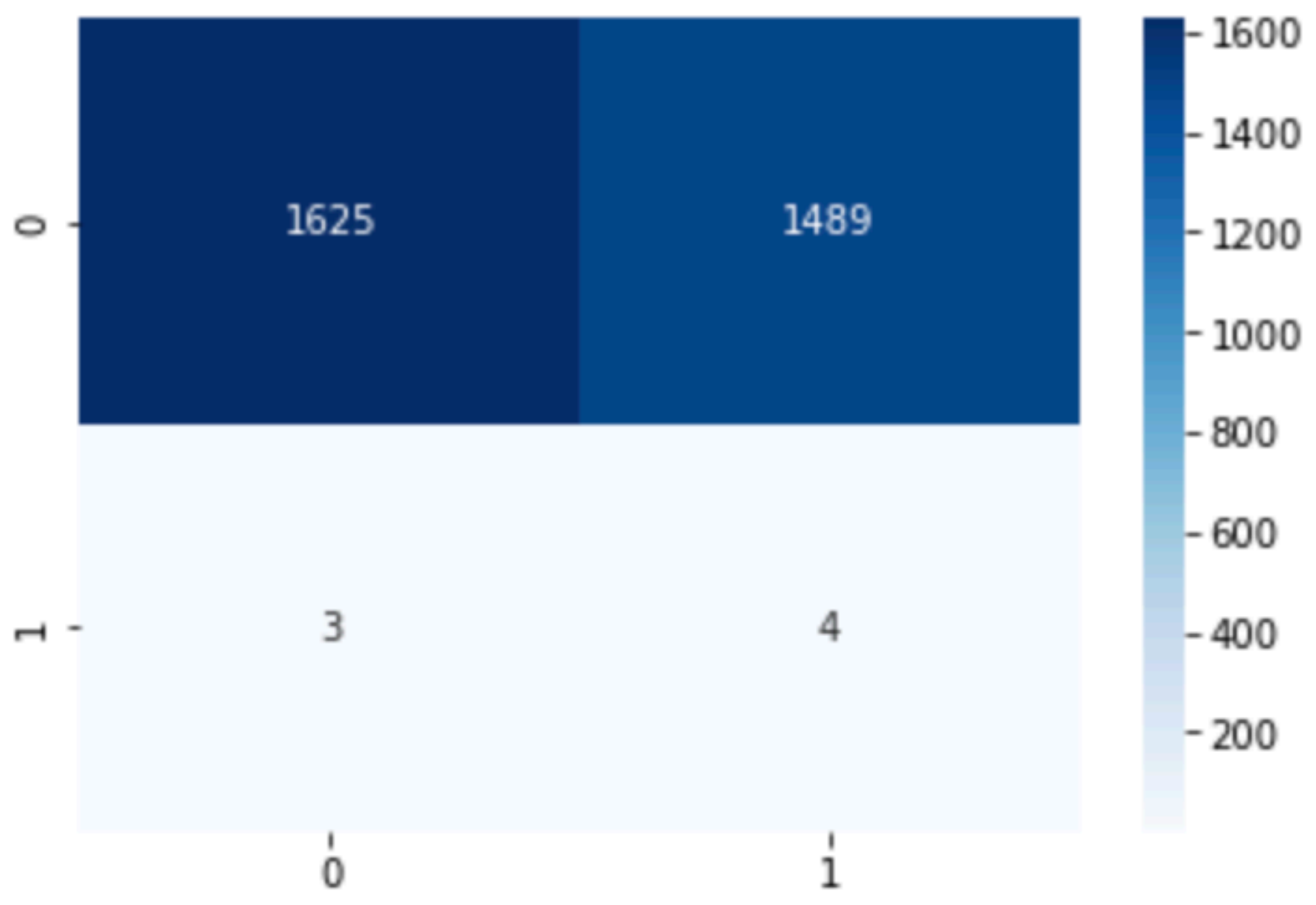
VALIDATION

EASY ENSEMBLE CLASSIFIER

- 3-Fold Cross Validation

```
each accuracy : [0.5658048  0.57861635 0.53226182]
each precision : [0.00428036 0.00441014 0.00496032]
each recall : [0.72727273 0.72727273 0.83333333]
average accuracy : 0.5588943240934855
average precision : 0.004550274873633767
average recall : 0.7626262626262627
```

- Test
  - Accuracy : 0.5219
  - Precision : 0.0027
  - Recall : 0.5714



CONCLUSION

LIMITATION

- Extremely imbalanced dataset
  - Avoid overfitting when dealing imbalanced data
- Time constraint
  - Insufficient time to study the newly found models in depth
  - Not able to confirm if our model is optimal
- Lack of understanding of specific columns
  - No further explanation from the company due to security issues

LESSON

- GOOGLE it FIRST!
  - What we want to implement is already implemented by smart ones!
- Let's approach the problem in various ways!
  - You will never know before you try.
- The difference between training data and field data is large.
- How can the insights from EDA be better applied in modeling?
- Are the best parameters really the best ones?

THANK YOU!