

# Data Science Final Report

第七組

107703001 資科三 鄭家宇

107306046 資管三 王品蓁

107306060 資管三 陳庭萱

107306064 資管三 陳建佑

107306079 資管三 邱聖雅

# **Content**

• **EDA**

• **Dealing With  
Missing Value**

• **Model  
Evaluation**

# **INTRODUCTION**

# D Data Description

- Data 來源： IEEE-CIS Fraud Detection Goal: we want to detect fraud from customer transactions, and prevent them.

## Categorical Features

ProductCD  
card1 - card6  
addr1, addr2  
P\_emaildomain  
R\_emaildomain  
M1 - M9  
DeviceType  
DeviceInfo  
id\_12 - id\_38

## Model Tried

Null model  
LightGBM Regressor  
KNN  
SVM

# **EXPLORATORY DATA ANALYSIS**

# D Data Description

## Transaction Table

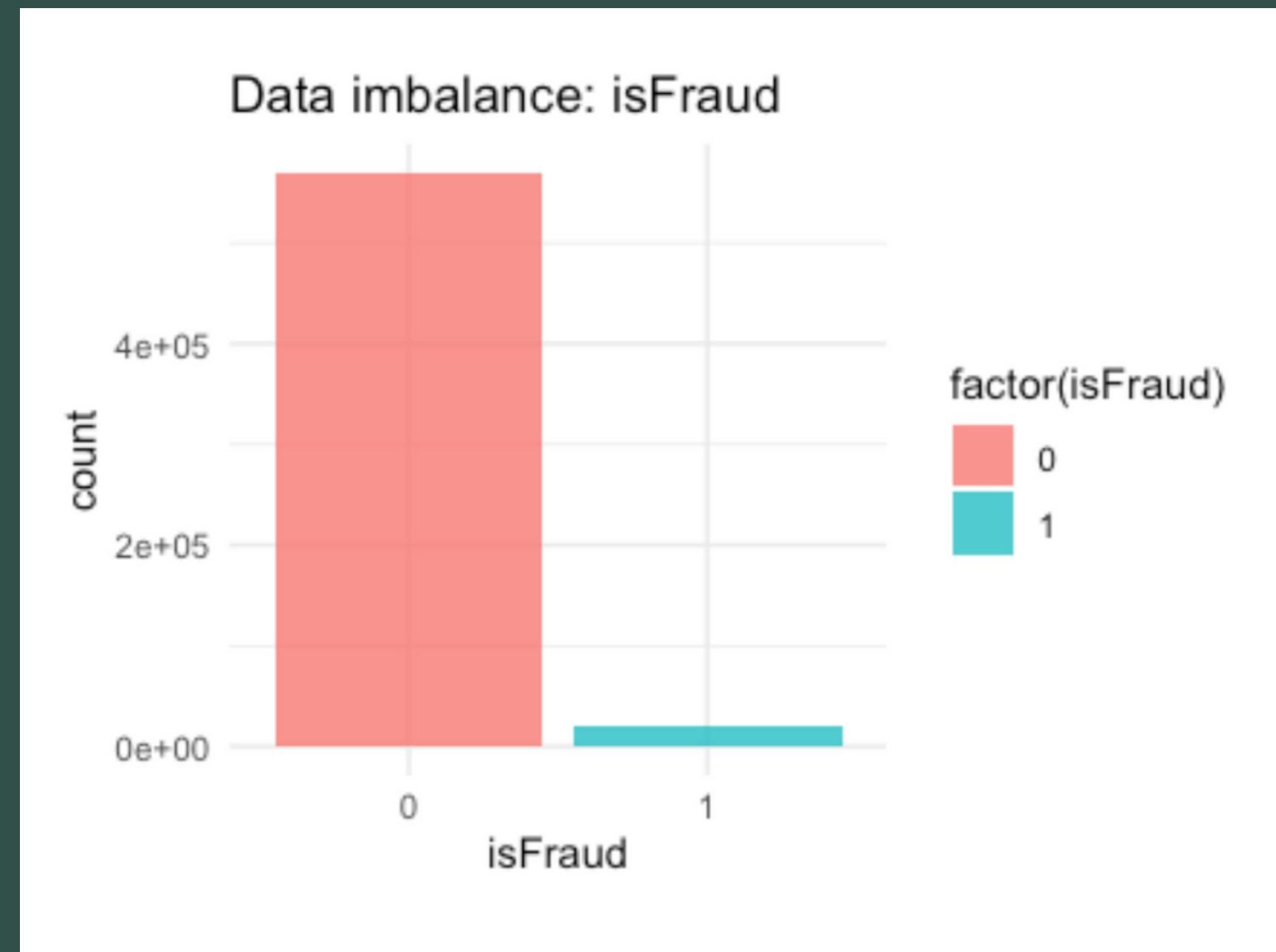
- **TransactionDT**: timedelta from a given reference datetime (not an actual timestamp)
- **TransactionAMT**: transaction payment amount in USD
- **ProductCD**: product code, the product for each transaction
- **card1 - card6**: payment card information, such as card type, card category, issue bank, country, etc.
- **addr**: addressdist: distanceP\_ and (R\_\_)
- **emaildomain**: purchaser and recipient email domain
- **C1-C14**: counting, such as how many addresses are found to be associated with the payment card, etc. The actual meaning is masked.
- **D1-D15**: timedelta, such as days between previous transaction, etc.
- **M1-M9**: match, such as names on card and address, etc.
- **Vxxx**: Vesta engineered rich features, including ranking, counting, and other entity relations."Our team finished projects one week earlier on average last year thanks to TaskWhiz."

# D Data Description

*Variables in this table are identity information – network connection information (IP, ISP, Proxy, etc) and digital signature (UA/browser/os/version, etc) associated with transactions.*

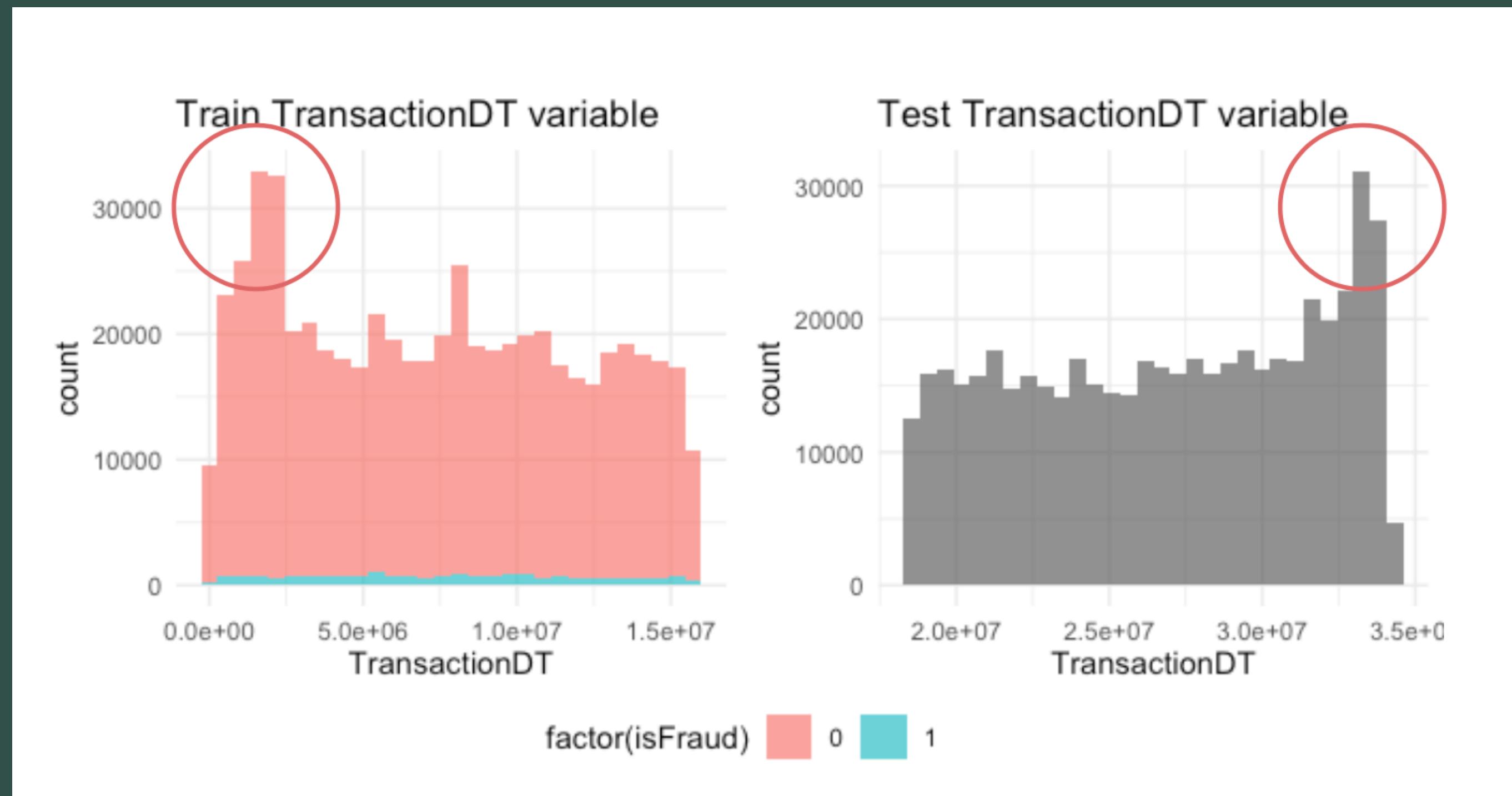
# ▷ Target Variable

The dataset is *imbalanced*, most transactions are non-fraud: might “overfit”

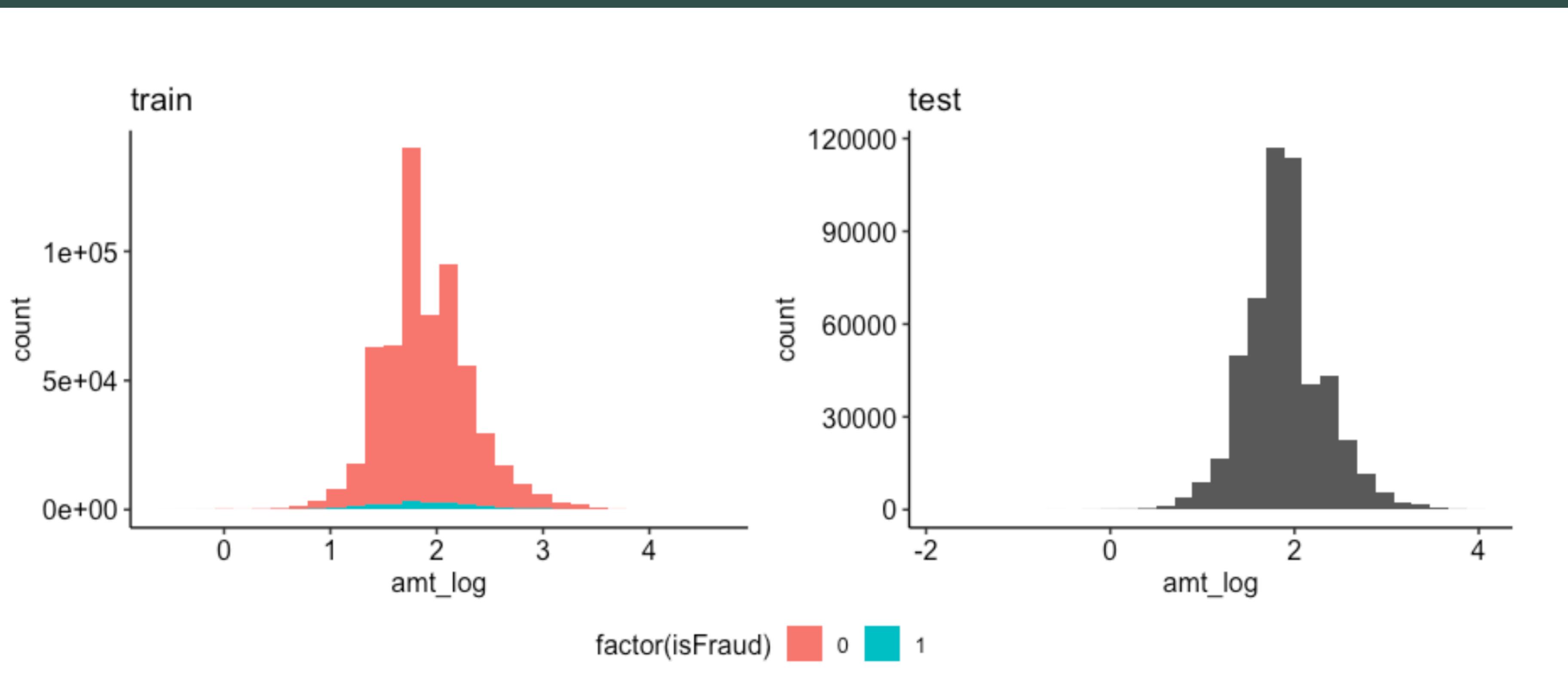


# ▷ TransactionDT

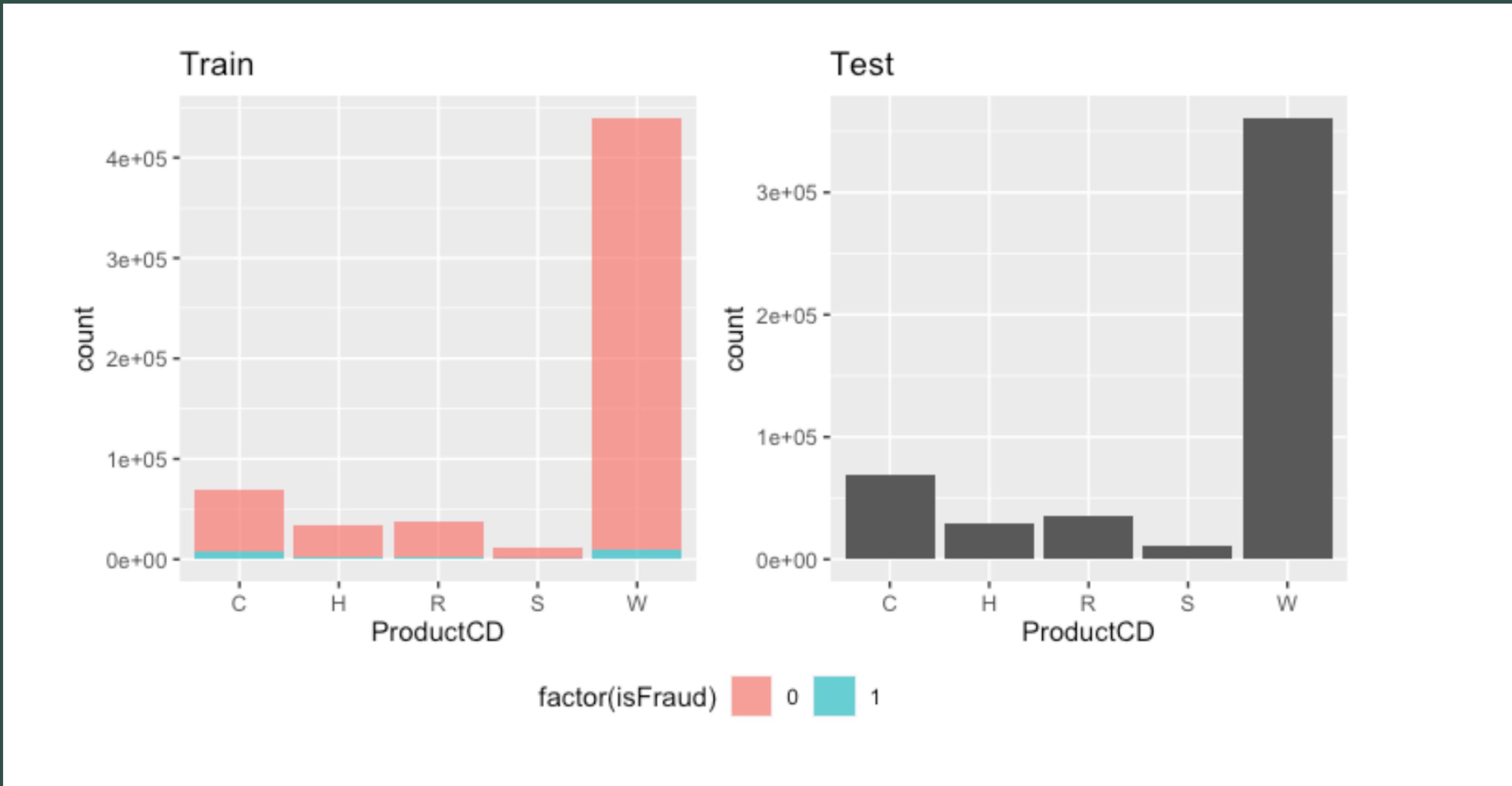
minimum and maximum values



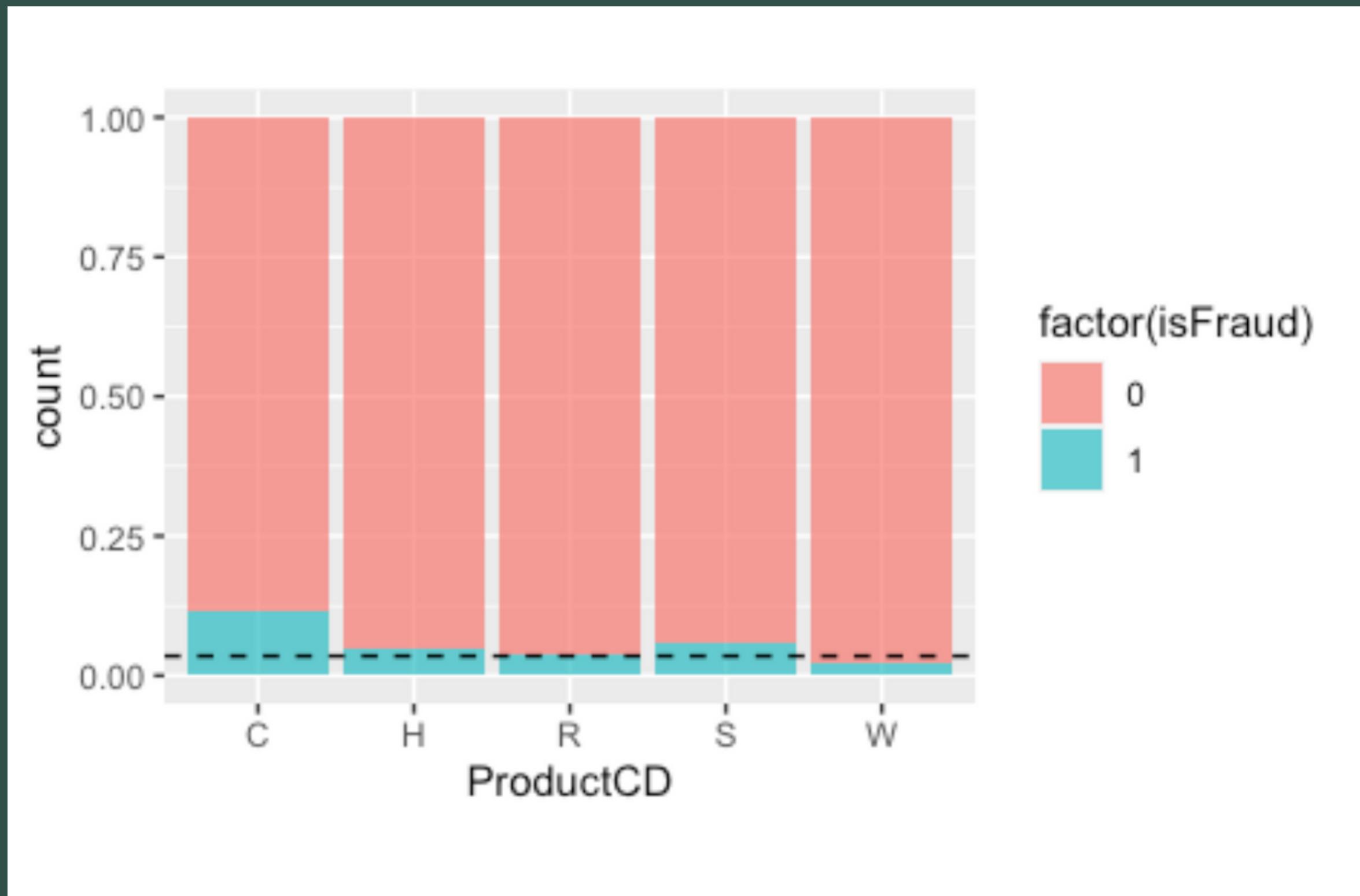
# ▷ TransactionAMT



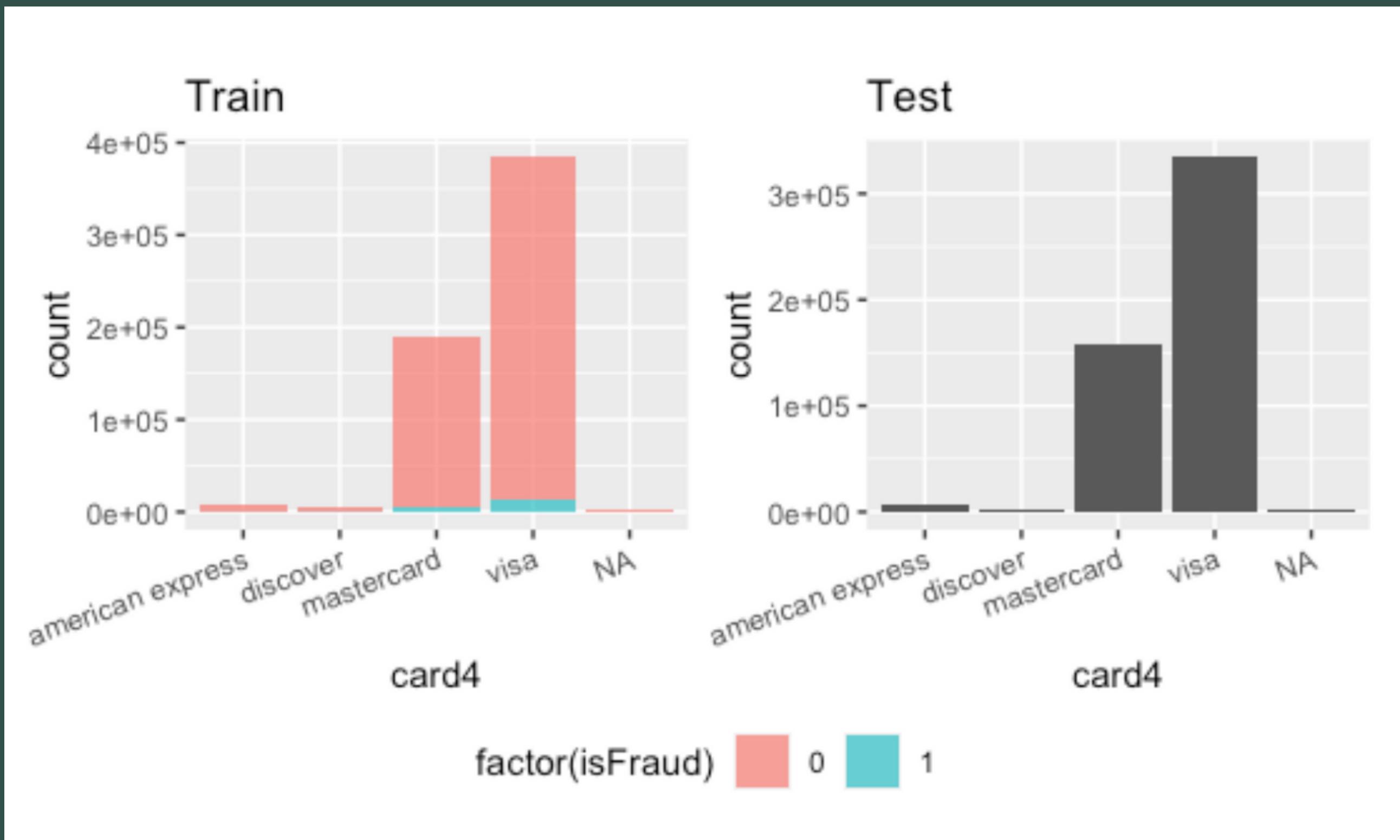
# D ProductCD



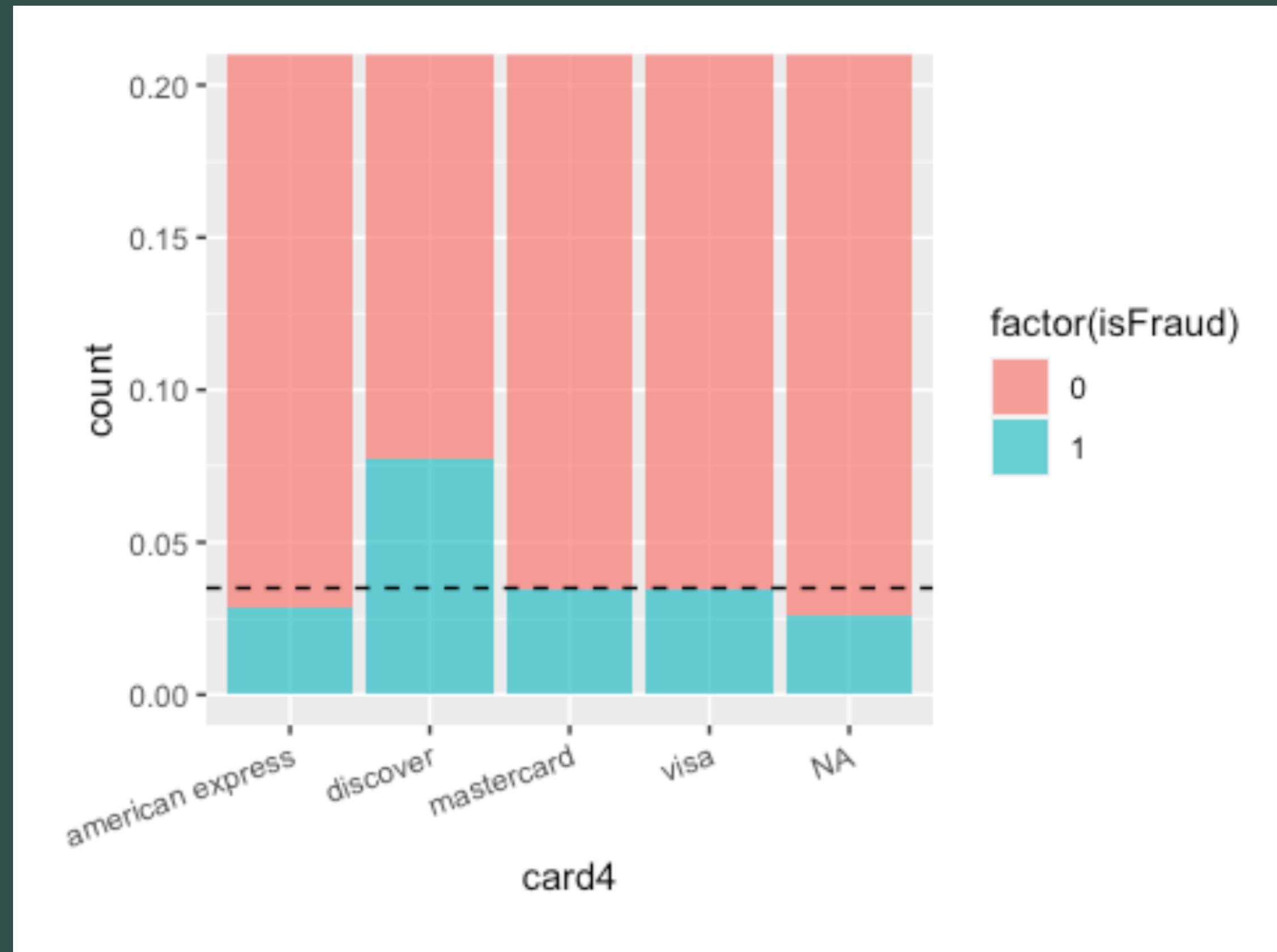
# D ProductCD



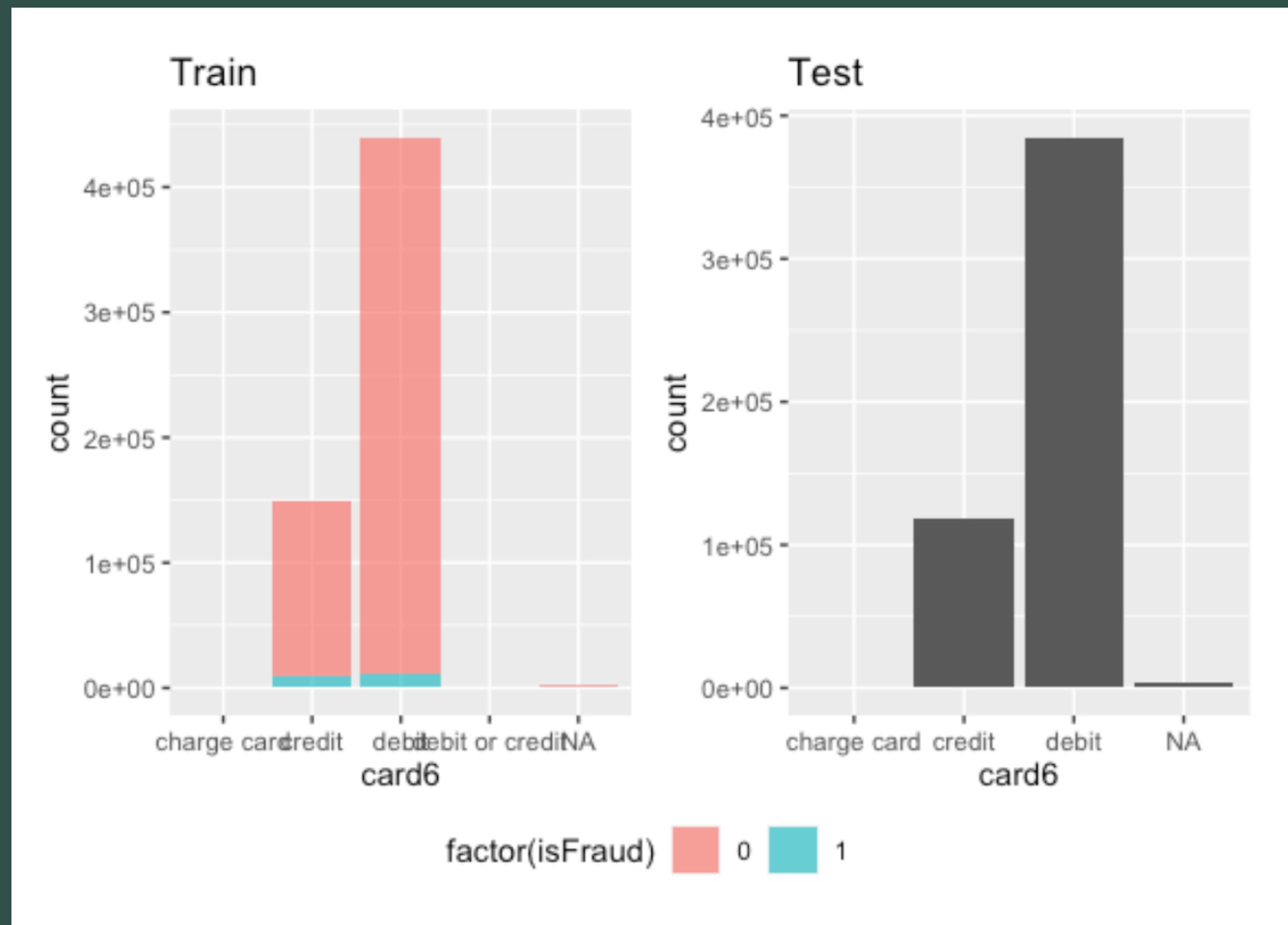
# D Card 4



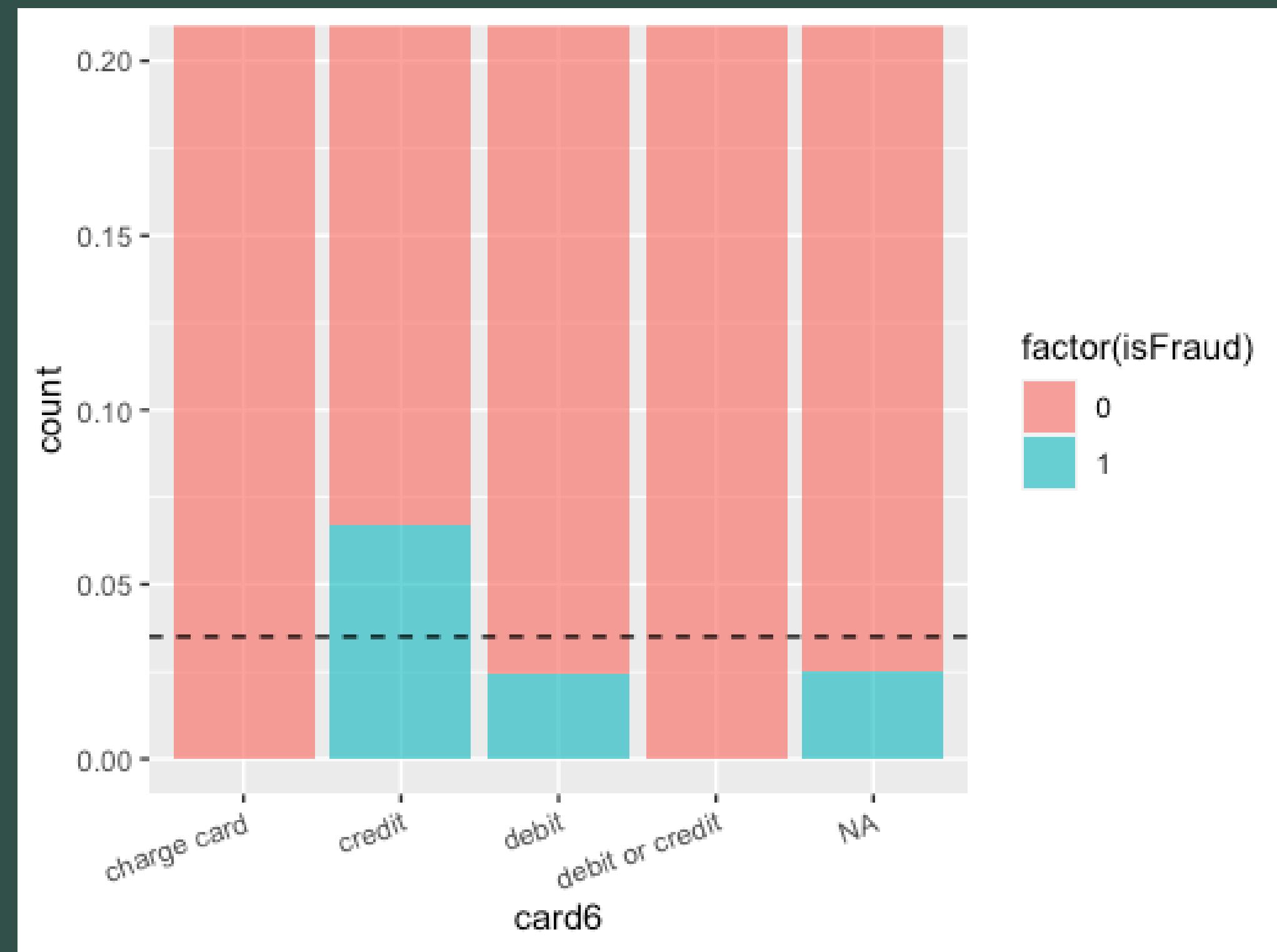
# D Card 4



# Card 6



# Card 6



# **DEALING WITH MISSING VALUE**

# Missing Rate

- missing values in the most of the columns.
- data full of missing values = low complete\_rate

skim_variable	n_missing	complete_rate
V138	<u>508595</u>	0.139
V139	<u>508595</u>	0.139
V140	<u>508595</u>	0.139
V141	<u>508595</u>	0.139
V142	<u>508595</u>	0.139
V143	<u>508589</u>	0.139
V144	<u>508589</u>	0.139
V145	<u>508589</u>	0.139
V146		
V147		
V148		

Variable	MissPercentage	
	<fct>	<dbl>
id_24	id_24	99.19616
id_25	id_25	99.13096
id_07	id_07	99.12707
id_08	id_08	99.12707
id_21	id_21	99.12639
id_26	id_26	99.12572
id_22	id_22	99.12470
id_23	id_23	99.12470
id_27	id_27	99.12470

# Different type of the columns

Two types of column in the data:

- Category columns
- Numeric columns

```
# Get the categorical and numeric columns
category_columns = [
    'ProductCD', 'card1', 'card2', 'card3', 'card4', 'card5', 'card6',
    'addr1', 'addr2', 'P_emaildomain', 'R_emaildomain', 'DeviceType', 'DeviceInfo',
    'M1', 'M2', 'M3', 'M4', 'M5', 'M6', 'M7', 'M8', 'M9', "id_12",
    "id_13", "id_14", "id_15", "id_16", "id_17", "id_19", "id_20",
    "id_28", "id_29", "id_30", "id_31", "id_32", "id_33",
    "id_34", "id_35", "id_36", "id_37", "id_38"]
```

# ► Fill NA in Numeric columns

NA imputation:

- KNN imputation
  - Error in knnlmpuation in r: Not sufficient complete cases for computing neighbors [closed]
- MICE imputation
- give mean, median and mode to fill NA

	V219	V206	V50	V292	V170	V101	V191	V120	V258	V84	...	V192	V151	V254	V30	V155	V283	D1	V338	C11	V328
0	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	1.0	14.0	0.0	2	0.0
1	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	1	0.0
2	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	1	0.0
3	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	0.0	112.0	0.0	1	0.0
4	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	49.0	1.0	0.0	0.0	1.0	0.0	0.0	1	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
590535	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	1.0	1.0	1.0	29.0	0.0	1	0.0
590536	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	1	0.0
590537	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1	0.0
590538	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	0.0	1.0	7.0	22.0	0.0	1	0.0
590539	0.0	0.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	...	1.0	1.0	1.0	1.0	1.0	2.0	0.0	0.0	1	0.0

590540 rows × 384 columns

# Memory Reduction

- Some float64, int64 column take too much memory
- but some columns can be int8 or int16.
- Do downcasting!

```
train.memory_usage().sum() → train.memory_usage().sum()  
2223973768 → 1071239688
```

TransactionID	float64
isFraud	int64
TransactionDT	float64
TransactionAmt	float64
ProductCD	object
...	...
V2_isna	bool
V15_isna	bool
V94_isna	bool
V147_isna	bool
V312_isna	bool
Length: 791, dtype: object	

# D Fill NA in Category columns

- Encoding: convert string to numeric
  - Label encoding
  - one hot encoding
- we have columns with very high cardinality, so we choose **LABEL ENCODING!**
- Two thing that we need to deal with:
  - Missing value
  - Unknown value

	ProductCD	card1	card2	card3	card4	card5	card6	addr1	addr2	P_emaildomain	...	id_29	id_30	id_31	id_32	id_33	id_34	id_35	id_36	id_37	id_38
0	4	10095	500	42	1	38	1	166	62		31	...	2	74	96	4	260	4	2	2	2
1	4	1372	303	42	2	2	1	173	62		16	...	2	74	96	4	260	4	2	2	2
2	4	2833	389	42	4	58	2	178	62		36	...	2	74	96	4	260	4	2	2	2
3	4	13341	466	42	2	14	2	282	62		54	...	2	74	96	4	260	4	2	2	2
4	1	2712	413	42	2	2	1	241	62		16	...	1	7	124	3	164	3	1	0	1
...	...	...	...	...	...	...	...	...	...		...	...	...	...	...	...	...	...	...	...	
590535	4	4305	500	42	4	108	2	132	62		31	...	2	74	96	4	260	4	2	2	2
590536	4	7354	124	42	2	106	2	78	62		16	...	2	74	96	4	260	4	2	2	2
590537	4	8621	494	42	2	106	2	98	62		16	...	2	74	96	4	260	4	2	2	2
590538	4	5297	380	42	2	106	2	219	62		2	...	2	74	96	4	260	4	2	2	2
590539	4	10950	69	42	2	2	1	151	62		16	...	2	74	96	4	260	4	2	2	2

590540 rows × 41 columns

# **MODEL EVALUATION**

# D Model

- Naive Bayes
- KNN
- LGB
- null model: give random prob.
  - for comparison
  - negative : positive = 0.97 : 0.03

# D Evaluation

- without cross validation

	knn	naive bayes	null model	lgb
accuracy	0.961	0.744	0.923	<b>0.982</b>
precision	0	0.982	0.032	0.772
<b>sensitivity</b>	0	<b>0.78</b>	0.234	0.033
specificity	0.962	0.615	<b>0.998</b>	0.969
f1	0	0.869	0.36	0.033

# D Performance



model	AUC score from kaggle
LGB	0.892
Naive Bayes	0.746
null model	0.5
KNN	0.5

# SHINY

**<https://sourlab.shinyapps.io/datascience/>**

# What's the problem?

## An error has occurred

The application failed to start: exited normally with code 137, signal 9 (SIGKILL)

### LOGS

```
2021-01-11T15:55:30.454277+00:00 shinyapps[system]: Out of memory!
2021-01-11T15:55:37.385770+00:00 shinyapps[system]: Out of memory!
2021-01-11T15:57:13.862446+00:00 shinyapps[system]: Out of memory!
2021-01-11T15:58:22.374907+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:02:11.703267+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:03:08.180247+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:08:55.741235+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:11:10.707325+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:15:17.140324+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:15:39.592174+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:17:06.037234+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:17:22.004832+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:29:40.242106+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:29:52.377532+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:34:50.398618+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:40:58.575547+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:41:11.185870+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:41:17.893742+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:42:05.304995+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:42:41.824297+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:43:39.551810+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:49:39.991687+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:55:14.405872+00:00 shinyapps[system]: Out of memory!
2021-01-11T16:58:49.929991+00:00 shinyapps[system]: Out of memory!
2021-01-11T17:00:04.099779+00:00 shinyapps[system]: Out of memory!
2021-01-11T19:05:11.397143+00:00 shinyapps[system]: Out of memory!
```

## D What we have tried?

- reduce dataset: from 3millions to 500
- reduce the feature image
- turn png to jpeg



**However,  
shiny still be out of memory :(**

<https://reurl.cc/Ag1QrK>

# IEEE CIS fraud detection

[homepage](#)[transaction](#)[identity](#)[transaction structure](#)[identity structure](#)[isFraud](#)[model result plot](#)[model result table](#)

## IEEE CIS fraud detection

IEEE-CIS works across a variety of AI and machine learning areas, including deep neural networks, fuzzy systems, evolutionary computation, and swarm intelligence. Today they're partnering with the world's leading payment service company, Vesta Corporation, seeking the best solutions for fraud prevention industry, and now you are invited to join the challenge.

This competition is a binary classification problem - i.e. our target variable is a binary attribute (Is the user making the click fraudulent or not?) and our goal is to classify users into "fraudulent" or "not fraudulent" as well as possible.

### Data

In this competition you are predicting the probability that an online transaction is fraudulent, as denoted by the binary target `isFraud`.

The data is broken into two files **identity** and **transaction**, which are joined by `TransactionID`.

Note: Not all transactions have corresponding identity information.

### Categorical Features - Transaction

- `ProductCD`
- `card1 - card6`
- `addr1, addr2`
- `P_emaildomain`
- `R_emaildomain`
- `M1 - M9`

### Categorical Features - Identity

- `DeviceType`
- `DeviceInfo`
- `id_12 - id_38`

### Files

- `train_{transaction, identity}.csv` - the training set
- `test_{transaction, identity}.csv` - the test set (you must predict the `isFraud` value for these observations)
- `sample_submission.csv` - a sample submission file in the correct format

### Model Tried

# Data Introduction

# Data Preview

http://127.0.0.1:4352 | [Open in Browser](#) |

## IEEE CIS fraud detection

homepage transaction identity transaction structure identity structure isFraud model result plot

model result table

Show 5 entries Search:

X.1	X	TransactionID	isFraud	TransactionDT	TransactionAmt	ProductCD	card1	card2	card3	card4
191	191	191	2987190	0	89565	32.325	0	3454	165	73
201	201	201	2987200	0	89666	34	4	9044	167	42
310	310	310	2987309	0	91339	25.95	4	771	10	42
104	104	104	2987103	0	88184	47.95	4	13533	446	42
200	200	200	2987199	0	89665	19.455	0	11593	444	73

Showing 1 to 5 of 50 entries Previous [1](#) [2](#) [3](#) [4](#) [5](#) ... [10](#) Next

# Data Preview

http://127.0.0.1:4352 | [Open in Browser](#) |

## IEEE CIS fraud detection

homepage transction identity transaction structure identity structure isFraud model result plot

model result table

Show 5 entries Search:

X	TransactionID	id_01	id_02	id_03	id_04	id_05	id_06	id_07	id_08	id_09	id_10	id_
133	133	2987504	-5	78385		0	0					
316	316	2988204	-5	127122		0	0					
496	496	2989451	-10	19389	0	0	0	0		0	0	
202	202	2987683	-5	30602		0	-12			0	0	95.0800018310
182	182	2987620	0	9297	0	0	0	-9		0	0	

Showing 1 to 5 of 50 entries Previous 1 2 3 4 5 ... 10 Next



# Data Structure

<http://127.0.0.1:4352> |  [Open in Browser](#) | 

IEEE CIS fraud detection

homepage transaction identity transaction structure identity structure isFraud model result plot

## model result table

```
'data.frame': 500 obs. of 436 variables:  
 $ X.1          : int 1 2 3 4 5 6 7 8 9 10 ...  
 $ X            : int 1 2 3 4 5 6 7 8 9 10 ...  
 $ TransactionID : int 2987000 2987001 2987002 2987003 2987004 2987005 2987006 2987007 2987008 2987009 ...  
 $ isFraud      : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ TransactionDT : int 86400 86401 86469 86499 86506 86510 86522 86529 86535 86536 ...  
 $ TransactionAmt: num 68.5 29 59 50 50 ...  
 $ ProductCD    : int 4 4 4 4 1 4 4 4 1 4 ...  
 $ card1         : int 10095 1372 2833 13341 2712 3816 8827 9143 1412 12782 ...  
 $ card2         : int 500 303 389 466 413 454 259 389 0 10 ...  
 $ card3         : int 42 42 42 42 42 42 42 42 42 42 ...  
 $ card4         : int 1 2 4 2 2 4 4 4 4 2 ...  
 $ card5         : int 38 2 58 14 2 108 58 108 108 106 ...  
 $ card6         : int 1 1 2 2 1 2 2 2 2 2 ...  
 $ addr1         : int 166 173 178 282 241 132 17 173 183 78 ...  
 $ addr2         : int 62 62 62 62 62 62 62 62 62 62 ...  
 $ dist1         : int 19 8 287 8 8 36 0 8 8 19 ...  
 $ dist2         : int 37 37 37 37 37 37 37 37 37 37 ...  
 $ P_emaildomain : int 31 16 36 54 16 16 54 29 1 54 ...  
 $ R_emaildomain : int 31 31 31 31 31 31 31 31 31 31 ...  
 $ C1            : int 1 1 1 2 1 1 1 1 1 2 ...  
 $ C2            : int 1 1 1 5 1 1 1 1 1 2 ...  
 $ C3            : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ C4            : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ C5            : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ C6            : int 1 1 1 4 1 1 1 1 1 3 ...  
 $ C7            : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ C8            : int 0 0 0 0 1 0 0 0 1 0 ...  
 $ C9            : int 1 0 1 1 0 1 1 0 0 3 ...  
 $ C10           : int 0 0 0 0 1 0 0 0 1 0 ...  
 $ C11           : int 2 1 1 1 1 1 1 1 1 1 ...  
 $ C12           : int 0 0 0 0 0 0 0 0 0 0 ...  
 $ C13           : int 1 1 1 25 1 1 1 1 1 12 ...  
 $ C14           : int 1 1 1 1 1 1 1 1 1 2 ...  
 $ D1            : int 14 0 0 112 0 0 0 0 0 61 ...  
 $ D2            : int 97 97 97 112 97 97 97 97 97 61 ...  
 $ D3            : int 13 8 8 0 8 8 8 8 8 30 ...
```

# Data Structure

IEEE CIS fraud detection

homepage transaction identity transaction structure identity structure isFraud model result plot

## model result table

```
'data.frame': 500 obs. of 42 variables:  
$ X : int 1 2 3 4 5 6 7 8 9 10 ...  
$ TransactionID: int 2987004 2987008 2987010 2987011 2987016 2987017 2987022 2987038 2987040 2987048 ...  
$ id_01 : int 0 -5 -5 -5 0 -5 -15 0 -10 -5 ...  
$ id_02 : int 70787 98945 191631 221832 7460 61141 NA 31964 116098 257037 ...  
$ id_03 : int NA NA 0 NA 0 3 NA 0 0 NA ...  
$ id_04 : int NA NA 0 NA 0 0 NA 0 0 NA ...  
$ id_05 : int NA 0 0 0 1 3 NA 0 0 0 ...  
$ id_06 : int NA -5 0 -6 0 0 NA -10 0 0 ...  
$ id_07 : int NA NA NA NA NA NA NA NA NA ...  
$ id_08 : int NA NA NA NA NA NA NA NA NA ...  
$ id_09 : int NA NA 0 NA 0 3 NA 0 0 NA ...  
$ id_10 : int NA NA 0 NA 0 0 NA 0 0 NA ...  
$ id_11 : num 100 100 100 100 100 100 NA 100 100 100 ...  
$ id_12 : chr "NotFound" "NotFound" "NotFound" "NotFound" ...  
$ id_13 : int NA 49 52 52 NA 52 14 NA 52 52 ...  
$ id_14 : int -480 -300 NA NA -300 -300 NA -300 NA NA ...  
$ id_15 : chr "New" "New" "Found" "New" ...  
$ id_16 : chr "NotFound" "NotFound" "Found" "NotFound" ...  
$ id_17 : int 166 166 121 225 166 166 NA 166 121 225 ...  
$ id_18 : int NA NA NA NA 15 18 NA 15 NA NA ...  
$ id_19 : int 542 621 410 176 529 529 NA 352 410 484 ...  
$ id_20 : int 144 500 142 507 575 600 NA 533 142 507 ...  
$ id_21 : int NA NA NA NA NA NA NA NA NA ...  
$ id_22 : int NA NA NA NA NA NA NA NA NA ...  
$ id_23 : chr "" "" "" "" ...  
$ id_24 : int NA NA NA NA NA NA NA NA NA ...  
$ id_25 : int NA NA NA NA NA NA NA NA NA ...  
$ id_26 : int NA NA NA NA NA NA NA NA NA ...  
$ id_27 : chr "" "" "" "" ...  
$ id_28 : chr "New" "New" "Found" "New" ...  
$ id_29 : chr "NotFound" "NotFound" "Found" "NotFound" ...  
$ id_30 : chr "Android 7.0" "iOS 11.1.2" "" "" ...  
$ id_31 : chr "samsung browser 6.2" "mobile safari 11.0" "chrome 62.0" "chrome 62.0" ...  
$ id_32 : int 32 32 NA NA 24 24 NA 32 NA NA ...  
$ id_33 : chr "2220x1080" "1334x750" "" "" ...  
$ id_34 : chr "match_status:2" "match_status:1" "" "" ...
```

# Model Result Plot

## IEEE CIS fraud detection

homepage

transction

identity

transaction structure

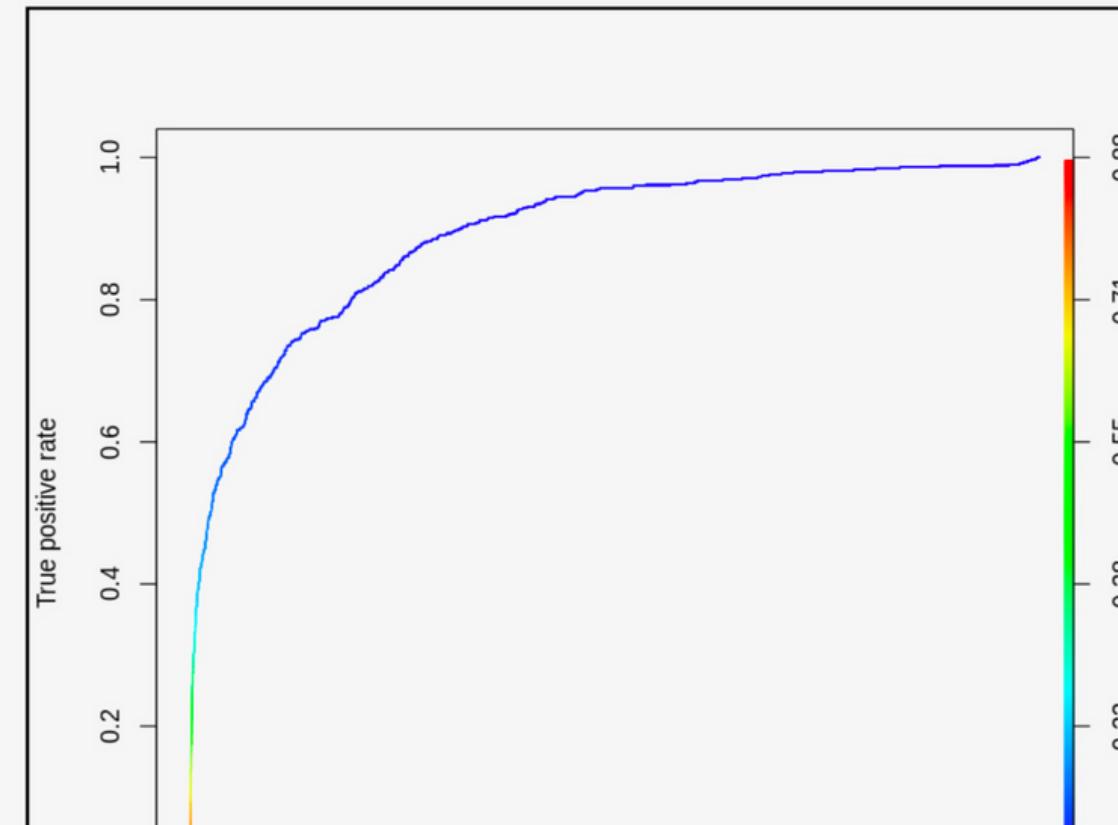
identity structure

isFraud

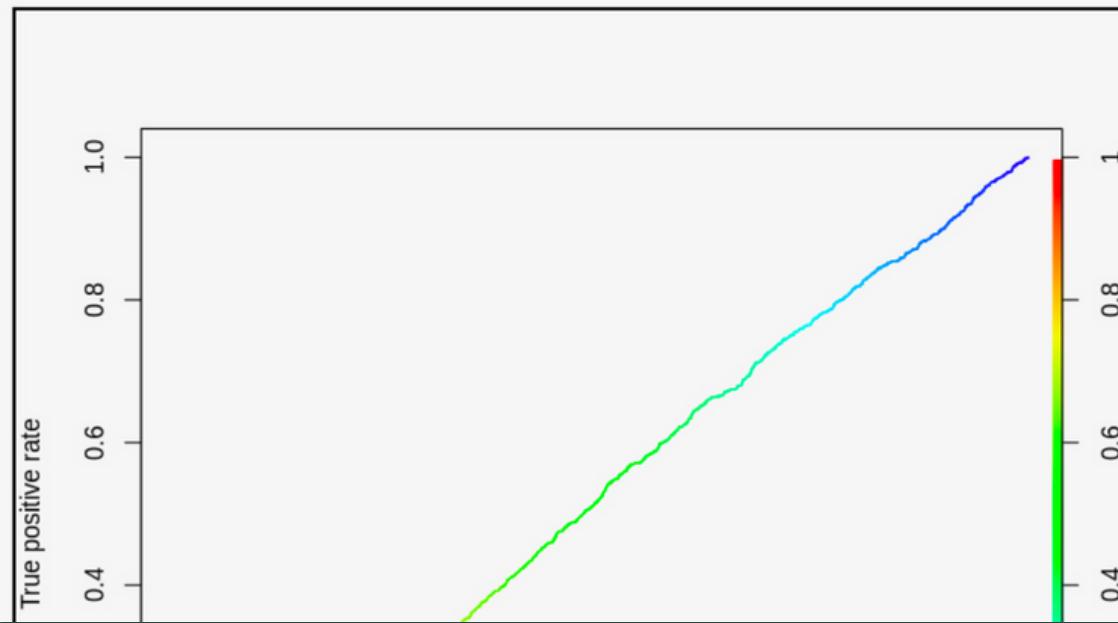
model result plot

model result table

**LGB**  
(1)AUC



**Null**  
(1)AUC



# Model Result Table

## IEEE CIS fraud detection

homepage    transction    identity    transaction structure    identity structure    isFraud    model result plot

model result table

Show 10 entries    Search:

	model	accuracy	precision	sensitivity	specificity	recall	F1	kappa
1	KNN	0.9613	0	0	0.9615	0	0	-0.00059
2	Naviebayes	0.774	0.982	0.78	0.615	0.78	0.869	0.105
3	Null	0.9425	0.0314	0.234	0.998	0.235	0.36	0.3536
4	LGB	0.982	0.7716	0.033	0.969	0.0337	0.0325	0.0029

Showing 1 to 4 of 4 entries    Previous 1 Next

# D Challenge

- huge datasets
- AUC score improved from 0.5 to 0.89

# D Reference

- <https://www.kaggle.com/yoongkang/beginner-s-random-forest-example>
- <https://www.kaggle.com/yoongkang/beginner-memory-reduction-techniques>
- <https://www.kaggle.com/sunghun/ieee-fraud-detection-eda-step1#3.-transactionAmt>
- <https://www.kaggle.com/c/ieee-fraud-detection/discussion/101203>