Credit Card Fraud Detection

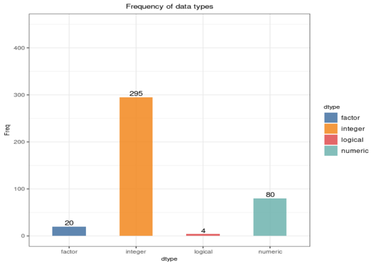
Nirbhaya Shaji, Lúcia Moreira

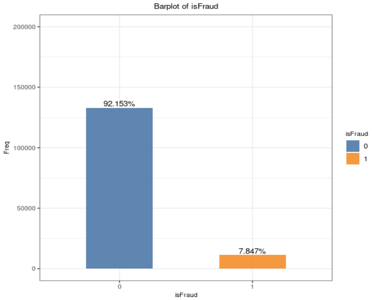
Introduction to Data Science Project, FCUP, University of Porto

December 17, 2019

Kaggle: IEEE-CIS Fraud Detection

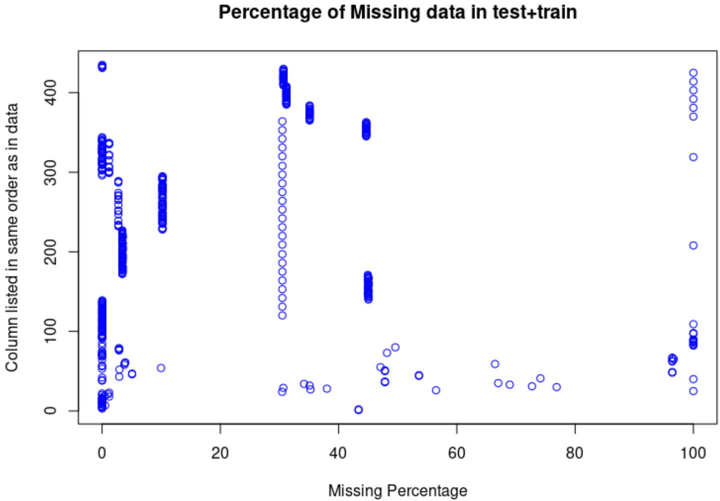
- Predicting the probability that an online transaction is fraudulent
- Vesta Corporation - e-commerce payment solutions
- Data is broken into two files: identity and transaction.
- Categorical and Numerical features
- train_transaction, identity.csv - the training set
- test_transaction, identity.csv - the test set





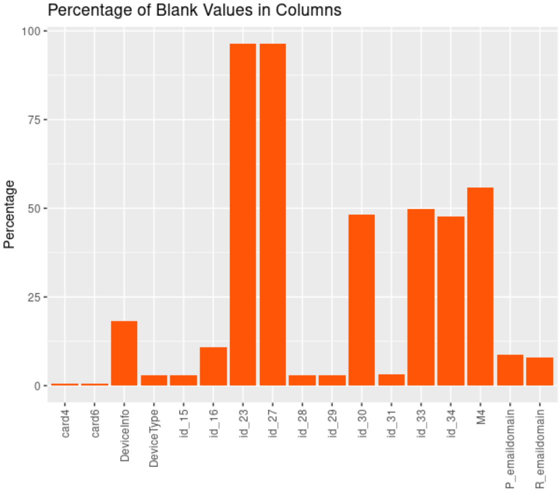
Missing Data

- Transaction and Identity merged over TransactionID
- Test and Train Data Combined

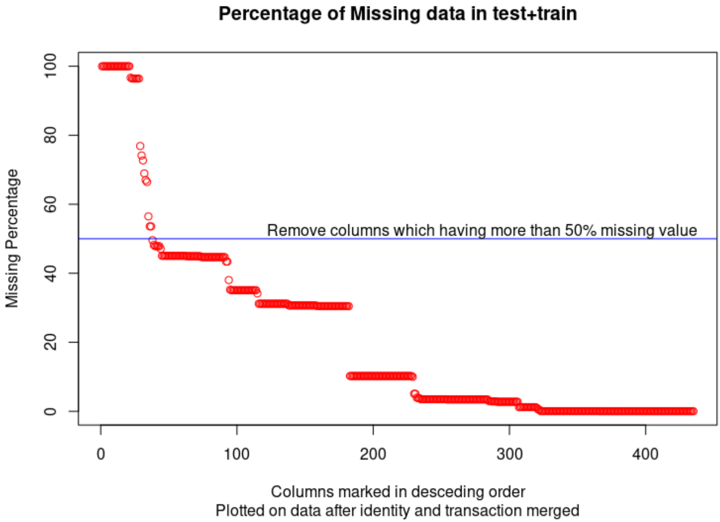


Sparsity

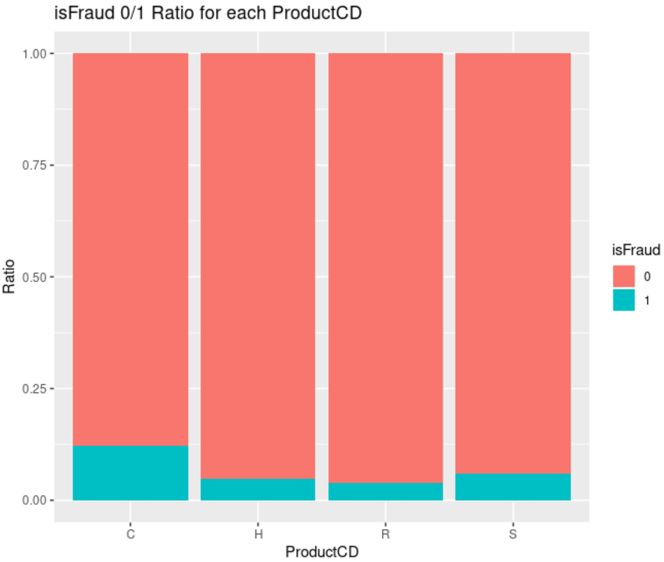
- Random Sparsity
- 15 percent of the data has NA values (17514759/114169860)
- Columns with blank values present



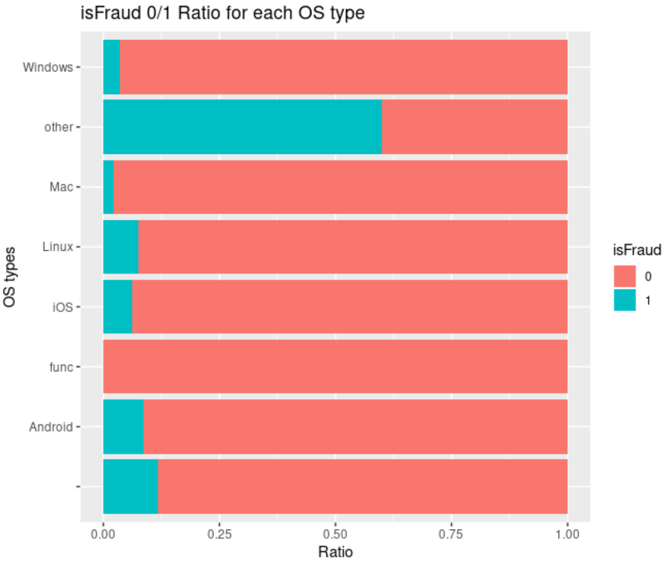
- Columns with more than 50 percentage NA removed for further EDA



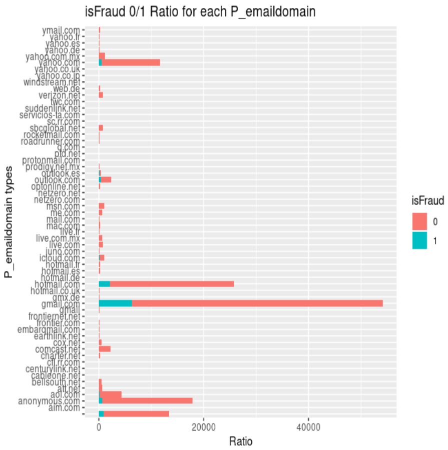
ProductCD

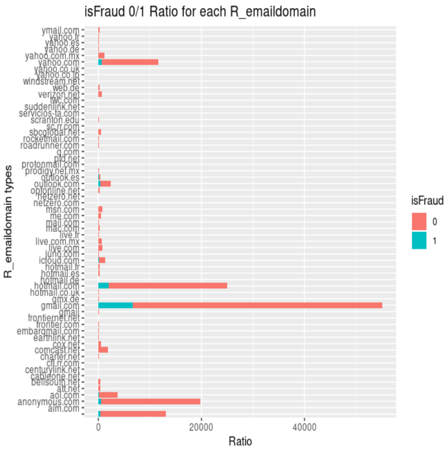

isFraud 0/1 Ratio for each ProductCD
1.00
0.75
0.50
0.25
0.00
Ratio
C
H
R
S
ProductCD
isFraud
0
1

OS type


isFraud 0/1 Ratio for each OS type
Windows
other
Mac
Linux
iOS
func
Android
OS types
0.00
0.25
0.50
0.75
1.00
Ratio
isFraud
0
1

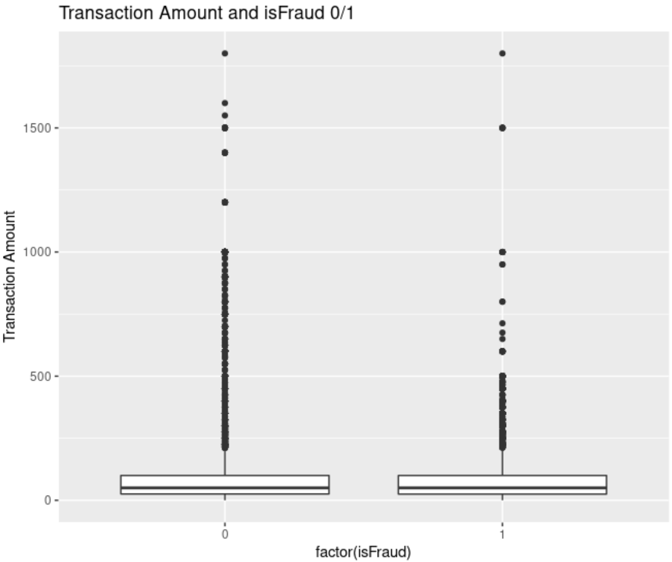
Email Domain



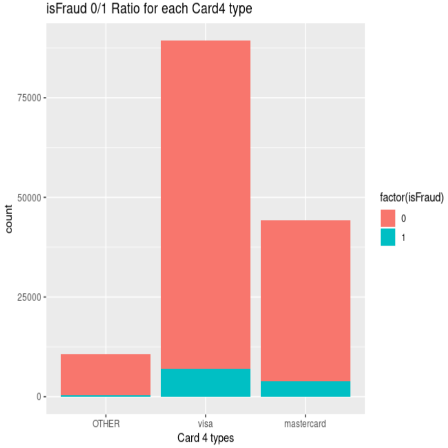


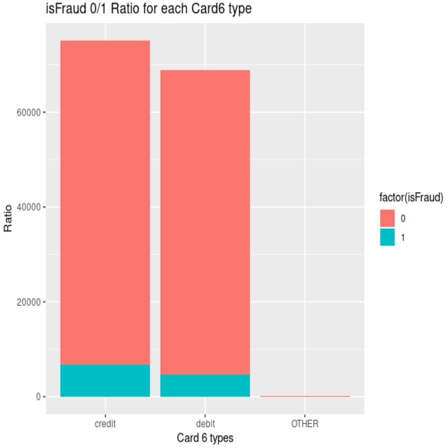
- Pretty much same as P-emaildomain with an exception of scranton.edu

Transaction Amount Box-plot

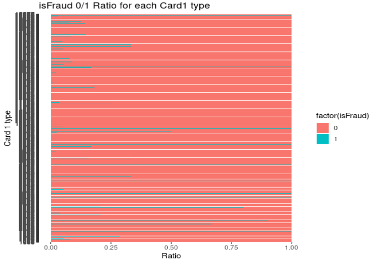


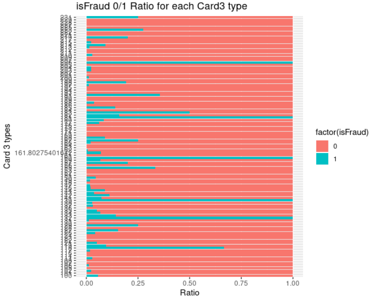
CardType 4: Visa/Master/Other and CardType 6: Credit/Debit/Other

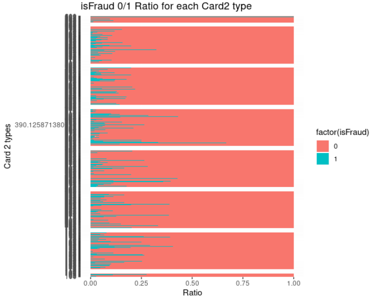


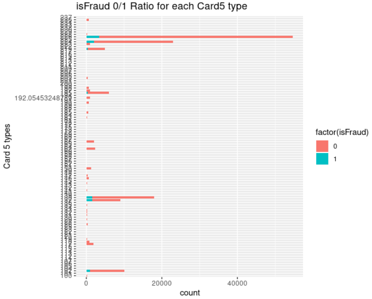


EDA - Rest of the Card types









Strategy I – decrease dataset size

- Very high sparse initial dataset files:
  - Transactions (590540, 394): **41 %** (670 MB)
  - Identity (144233, 41): **36 %** (26 MB)

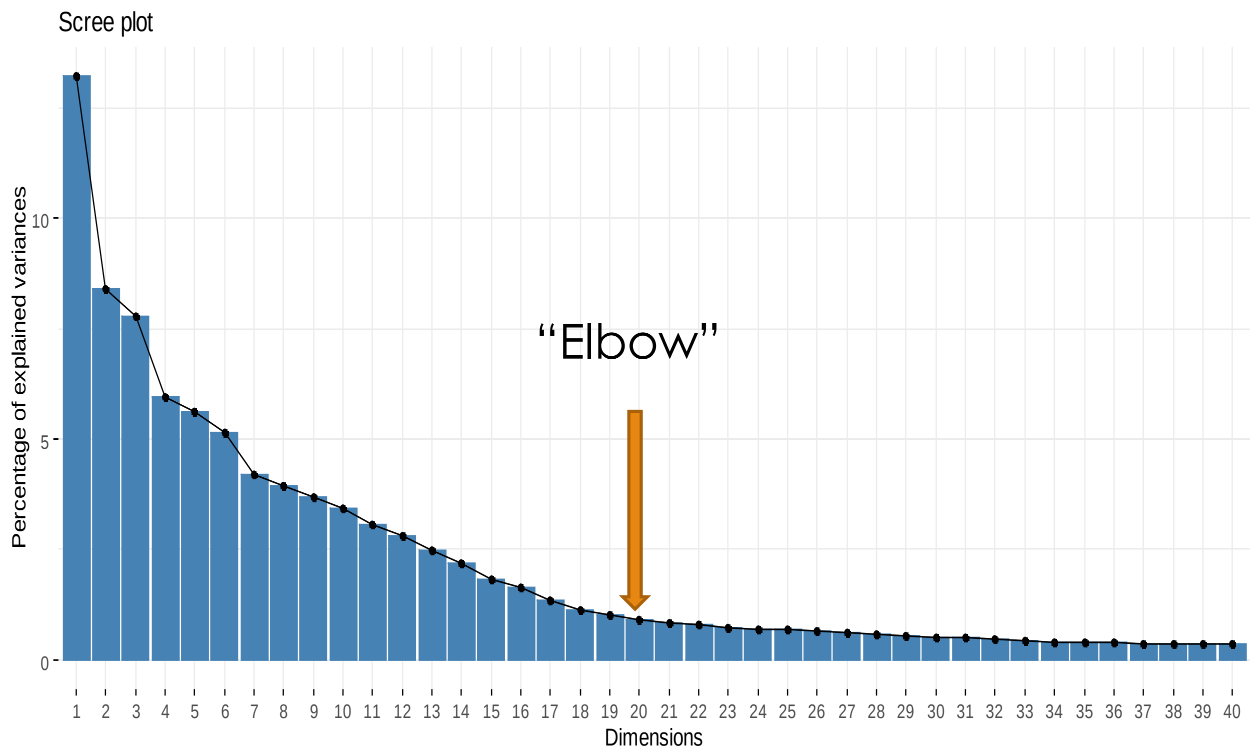
- Merge by identification ID (144233, 434): **27 %**
  - Combine information relevant in both files
  - Also allows decreasing dataset size

- Drop columns with more than 50% NaNs (144233, 303): **11%**
  - Drop 131 columns – allowed further decrease in dataset size (180 MB yet)

- Umbalanced feature was kept:
  - 8 % fraud
  - 92% not fraud

Strategy II – Decrease number of numerical variables

- We still have **270 numerical variables** in the new dataset
- **PCA analysis**:
  - Mean was input in the remaining NaNs



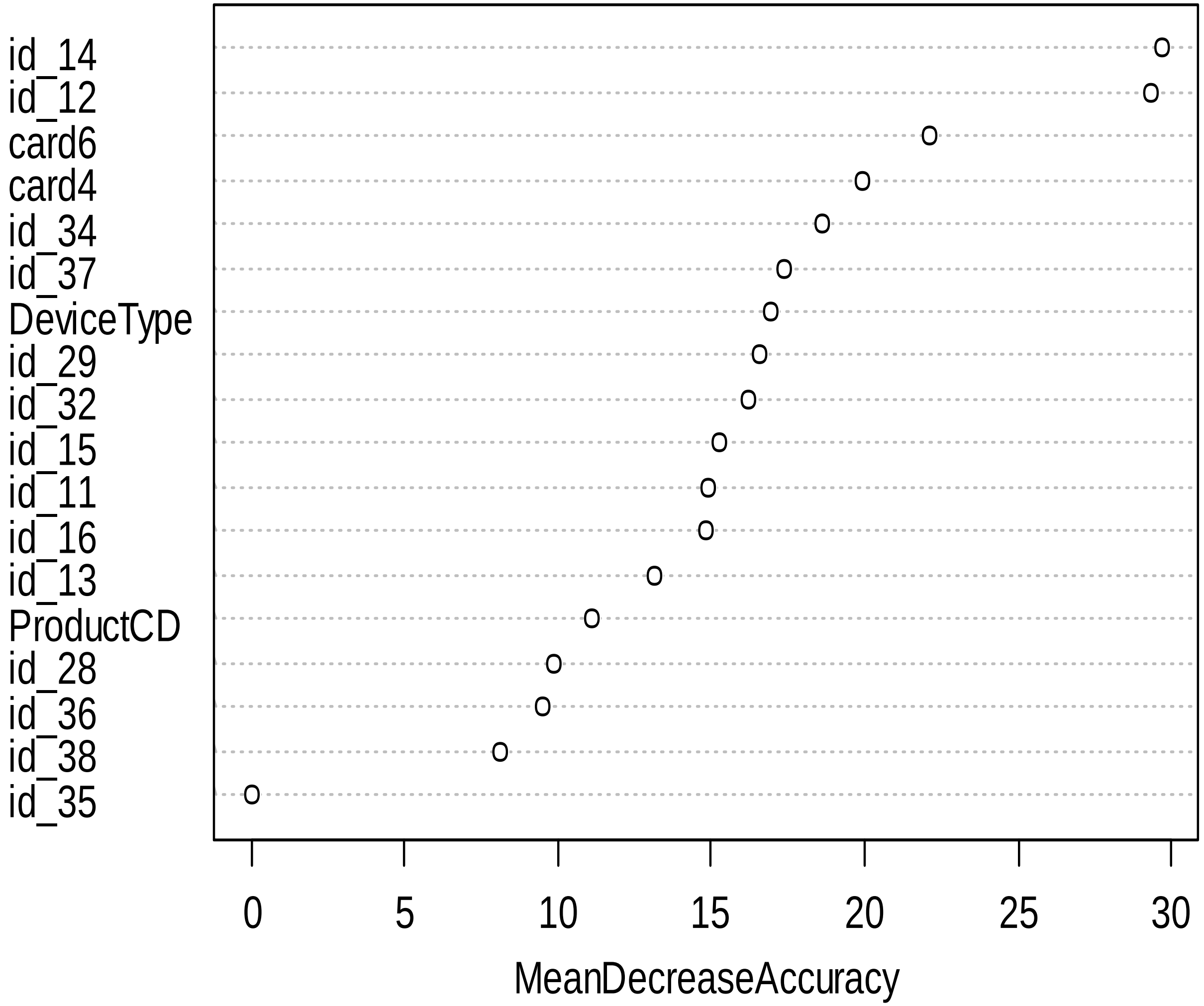
Comply with the 3 common criteria

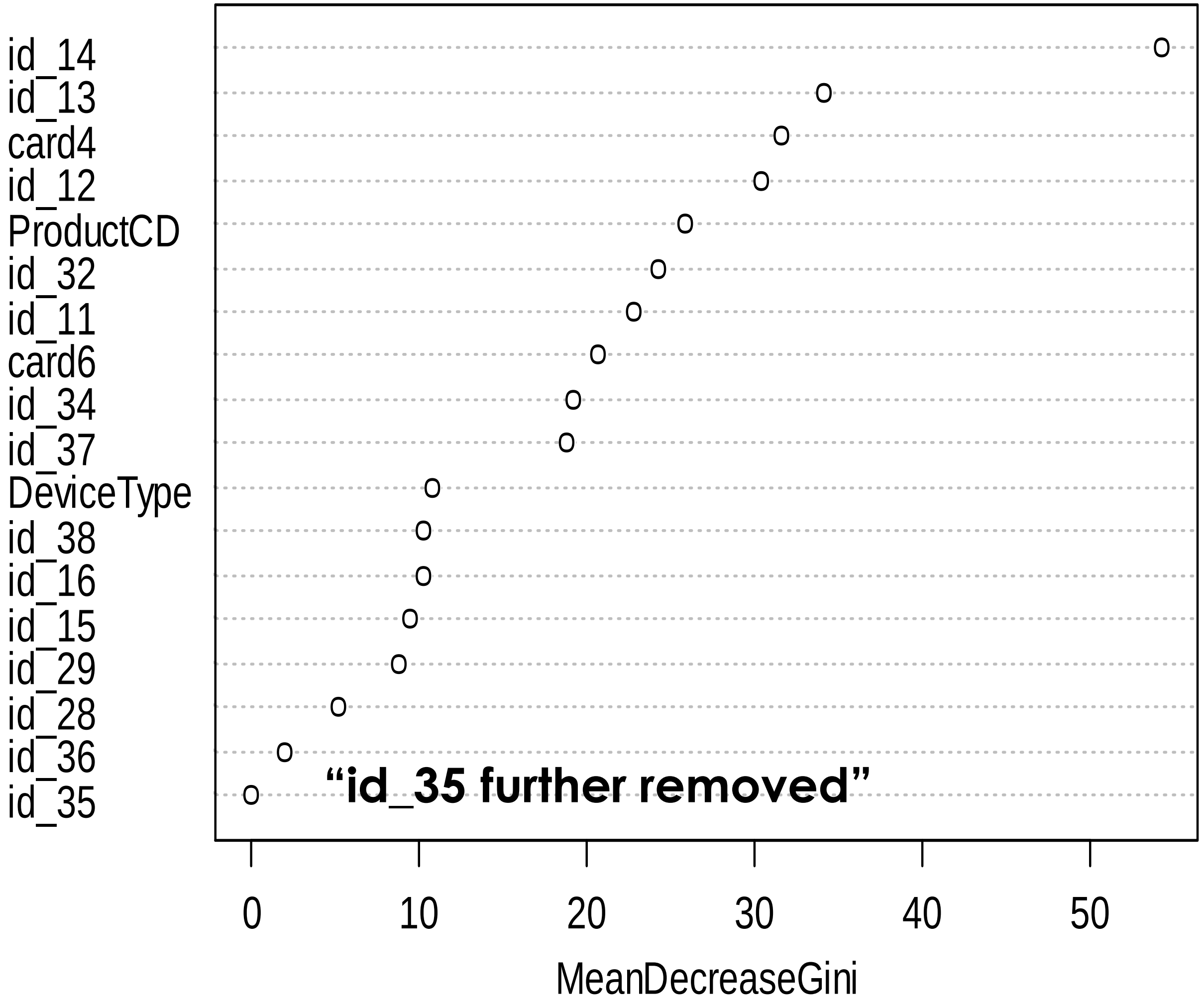
Component	eigenvalue	% of variance	cumulative % of variance
1	34,9	13,2	13,2
2	22,2	8,4	21,6
3	20,5	7,8	29,4
4	15,7	6,0	35,4
5	14,9	5,6	41,0
6	13,6	5,1	46,1
7	11,1	4,2	50,3
8	10,4	4,0	54,3
9	9,7	3,7	58,0
10	9,0	3,4	61,4
11	8,1	3,1	64,4
12	7,4	2,8	67,2
13	6,6	2,5	69,7
14	5,8	2,2	71,9
15	4,8	1,8	73,7
16	4,3	1,6	75,4
17	3,6	1,4	76,7
18	2,9	1,1	77,8
19	2,7	1,0	78,9
20	2,4	0,9	79,8
21	2,2	0,8	80,6

Strategy III – Identify relevance of the categorical variables

- Data set has 33 categorical variables
- Random Forest – handles NaNs
- Ensemble with 500 trees
- CPU Memory intensive
- Categorical variables with more than 53 factos: removed!
  - 15 variables removed

Strategy III – Random Forest





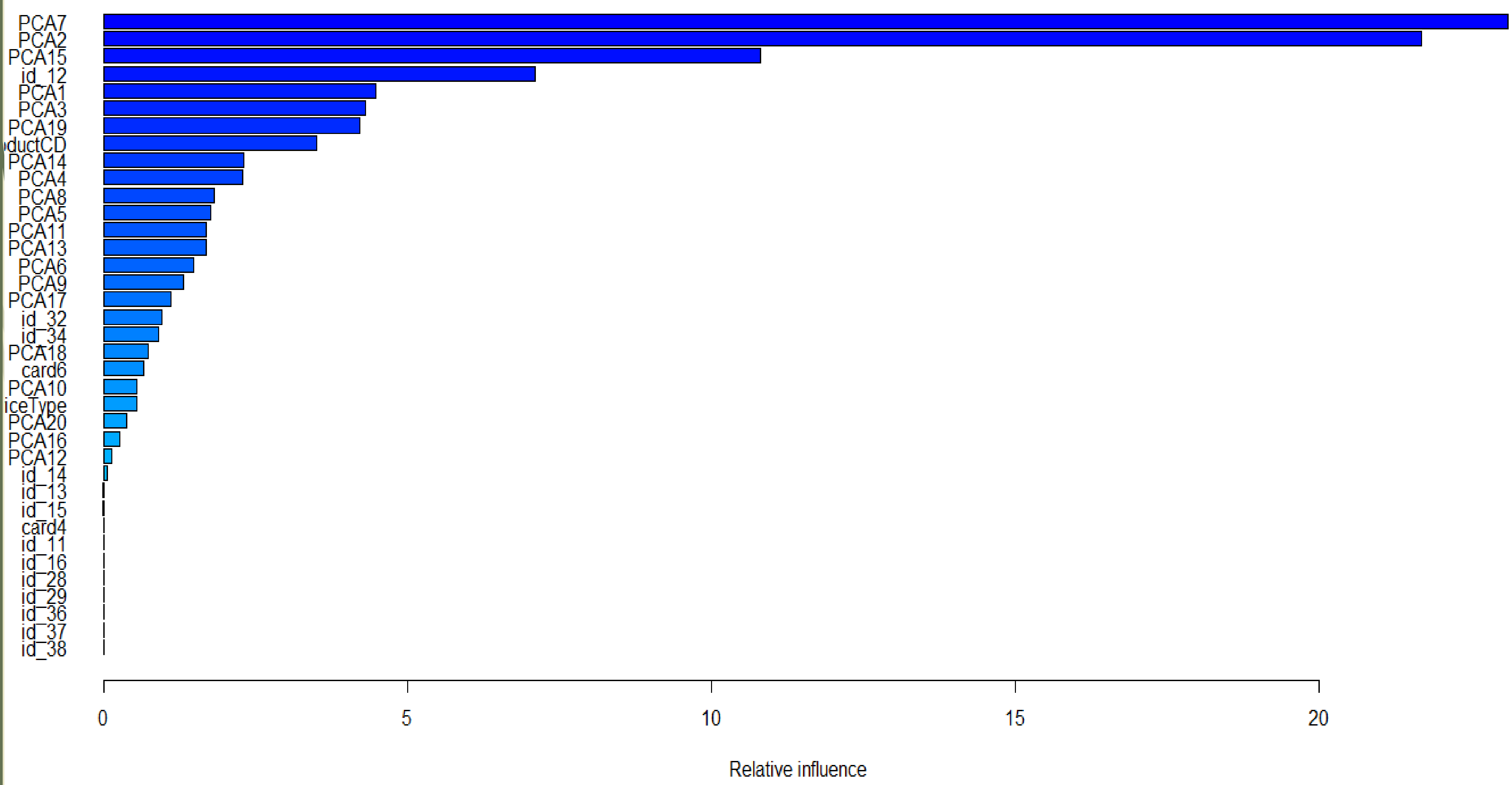
Model – Gradient Boosting Machine

- **Relevant algorithm for umbalanced data sets**
- **20 numerical dimensions from PCA**
- **18 categorical significant variables with up to 53 factors**

- Time domain consideration:
  - 50% past for training
  - 50% future for testing
- Hyperparameters:
  - Holdout method with 30% future for validation based on accuracy:
    - Shrinkage (regularization),
    - nr. trees
    - depth

- Random splitting
  - 65 % for training
  - 35 % for testing
- Hyperparameters:
  - 10-fold CV based on accuracy:
    - shrinkage (regularization)
    - nr. trees
    - depth

Model – Gradient Boosting Machine



Model – Gradient Boosting Machine

- Time domain consideration
  - Logistic Regression (Bernoulli distribution):
    - Accuracy train: **96.2 %**

prediction	0	1
0	68588	2526
1	253	1133

    - Accuracy test: **89.8 %**

prediction	0	1
0	63696	6945
1	377	714

  - AdaBoost (exponential loss for 0-1 outcomes):
    - Accuracy train: **96. 0 %**

prediction	0	1
0	68595	2626
1	246	1033

    - Accuracy test: **89.7 %**

prediction	0	1
0	63903	7250
1	170	409

Model – Gradient Boosting Machine

- Random splitting
  - Logistic Regression:
    - Accuracy train: **93.9 %**

prediction	0	1
0	85771	5072
1	643	2265

- Accuracy test: **93.4%**

prediction	0	1
0	46142	2969
1	359	1012

- Hyperparameters:
  - 10-fold Cross-validation
- nr trees: 800
- depth= 2
- shrinkage =0.01

In a binary setting:

		Predicted	
		P	N
Condition	P	TP	FN
	N	FP	TN

- Recall=TP/TP+FN=46142/46142+359= 46142/46501= **99.2 %**
- Precision=TP/TP+FP=46142/46142+2969=46142/49111= **93.9** %
- True negative rate: TN/TN+FP = 1012/1012+2969 = 1012/3981= **25.4** %
- Balanced accuracy= Recall + TNR/2 = **62 %**
- **Still better than random guess!**

Conclusions

- We had a quite high sparse data set
- Data reduction analysis:
  - PCA analysis
    - 270 numerical values allowed reduction to 20 variables that explain 80 % of the variance,
    - Average was input in the empty instances
  - Random Forests
    - Allowed discriminate the most relevant categorical variables for this classification problem
    - Variables with more than 53 factors were removed
    - Most of the remaining categorical variables were relevant except one
- Gradient Boosting Machine with the selected 38 variables:
  - Apparently not a time series problem ("reported chargeback on the card as fraud transaction ")
  - 93.4 % accuracy
  - **In future use the balanced accuracy for optimizing the hyperparameters**