How Severe is an Insurance Claim?

Machine Learning Kaggle Team Project
SVC Team

Allstate®
You're in good hands.

Why Do We Care & What Do We Want to Find Out

Allstate, a personal insurer in the United States, is continually seeking fresh ideas to improve their claims service for the over 16 million households they protect.

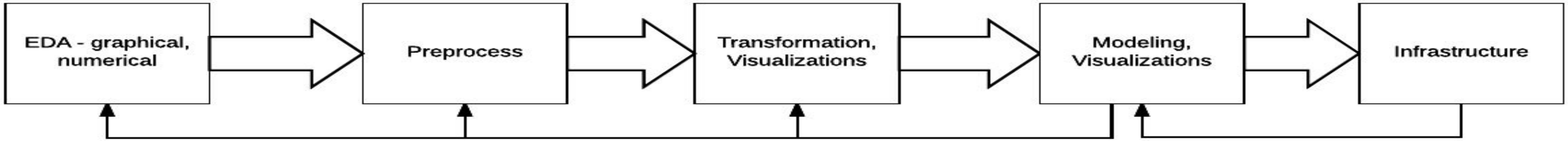
Allstate is currently developing automated methods of predicting the cost, and hence severity, of claims. In this recruitment challenge, Kagglers are invited to show off their creativity and flex their technical chops by **creating an algorithm which accurately predicts claims severity.** Aspiring competitors will demonstrate insight into better ways to predict claims severity for the chance to be part of Allstate's efforts to ensure a worry-free customer experience.

Background knowledge of the dataset

Each row in this dataset represents an insurance claim. You must predict the value for the 'loss' column. Variables prefaced with 'cat' are categorical, while those prefaced with 'cont' are continuous.

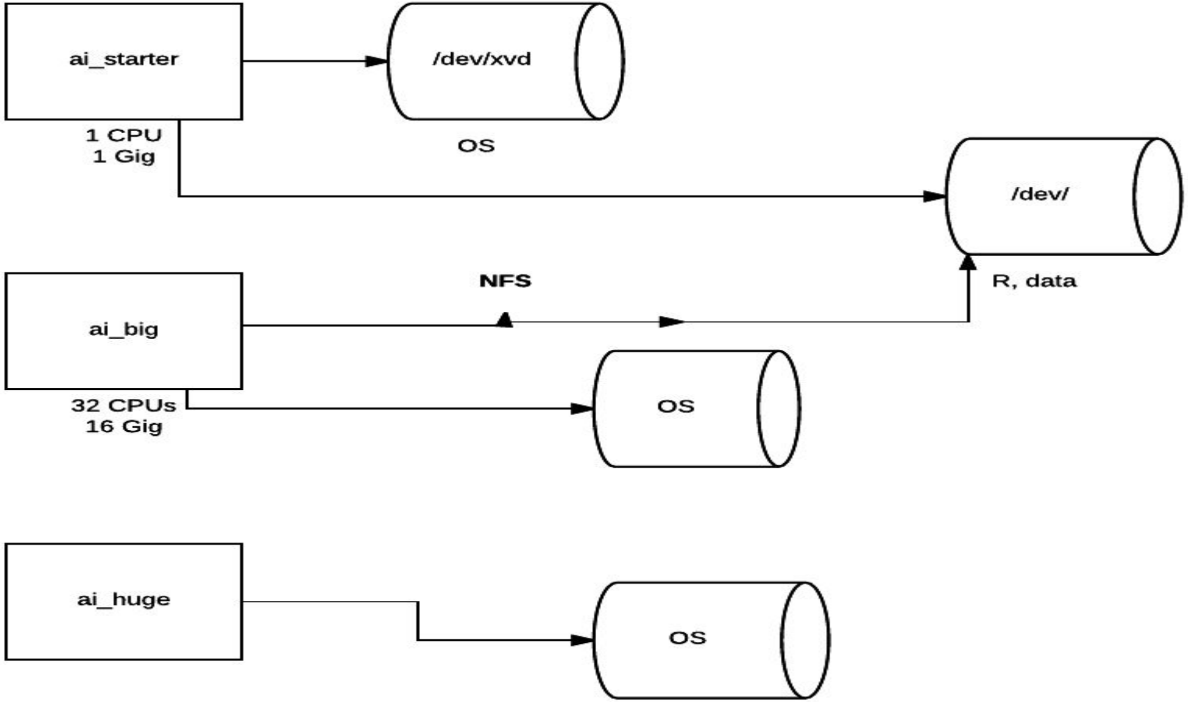
File descriptions

- **train.csv** - the training set
- **test.csv** - the test set. You must predict the loss value for the ids in this file.
- **sample_submission.csv** - a sample submission file in the correct format



SVC Team: Sid, Valerie, Connie

Infrastructure - AWS



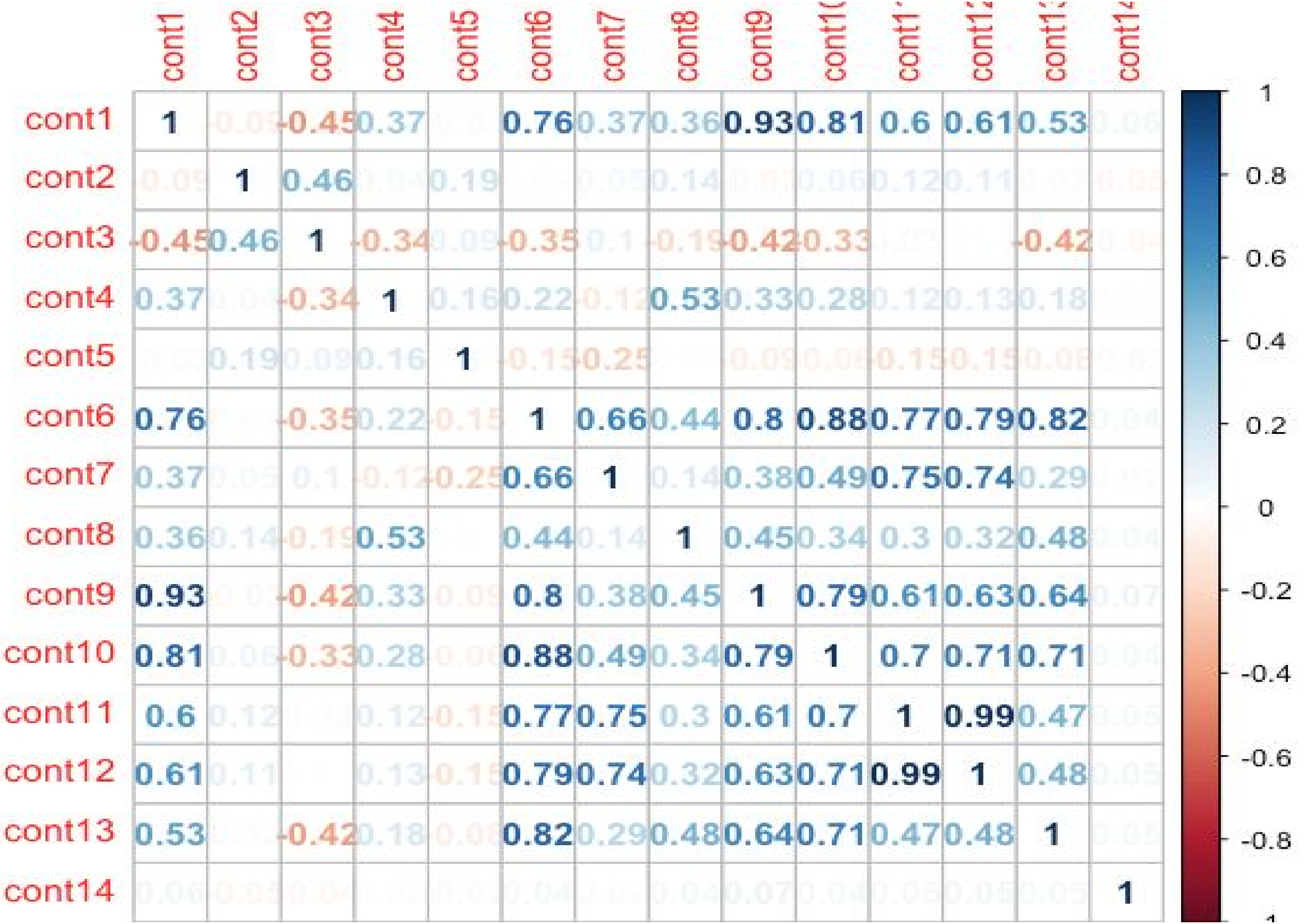
The Data

Training Dataset

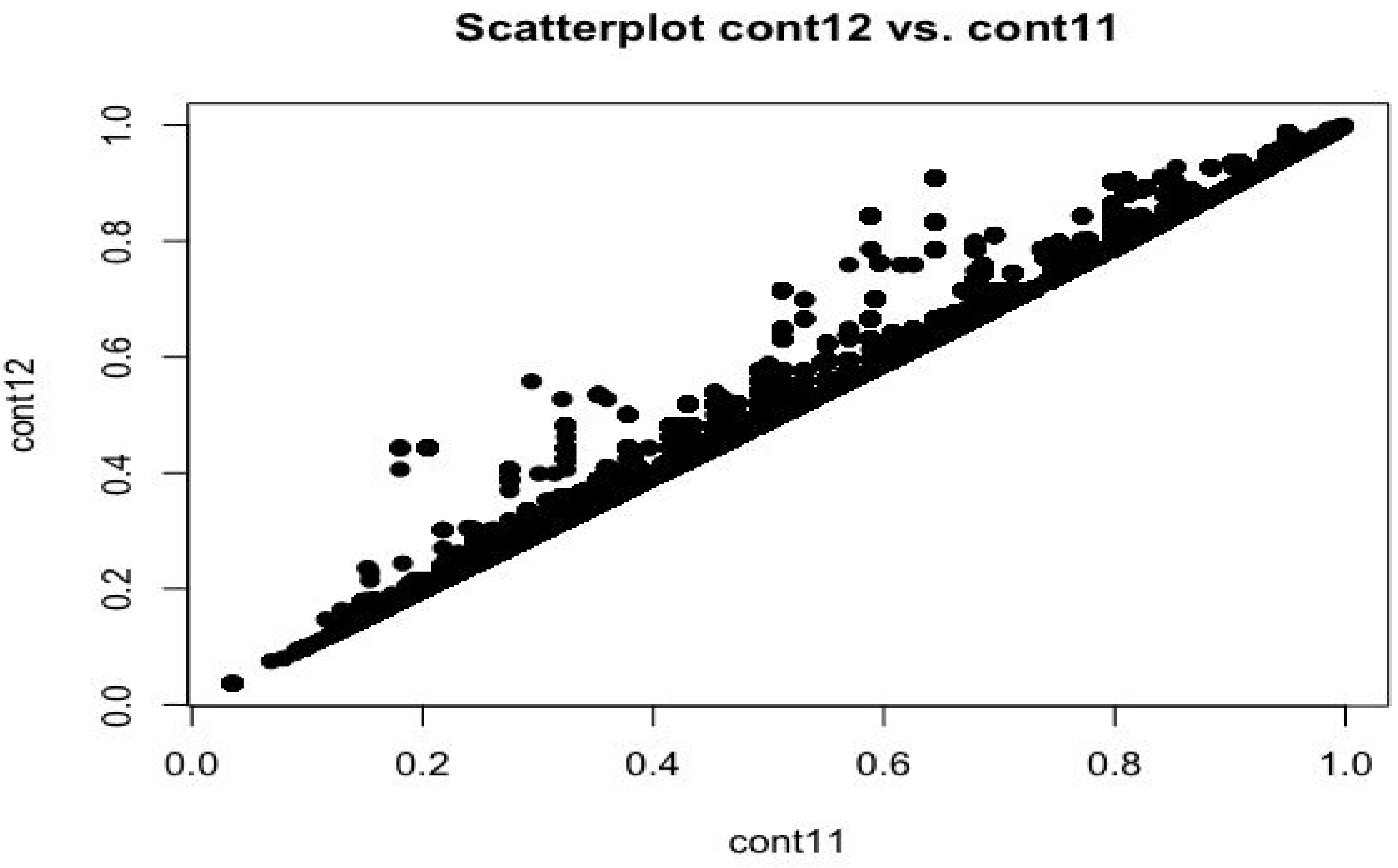
Observations: 188318

Categorical Variables: 116

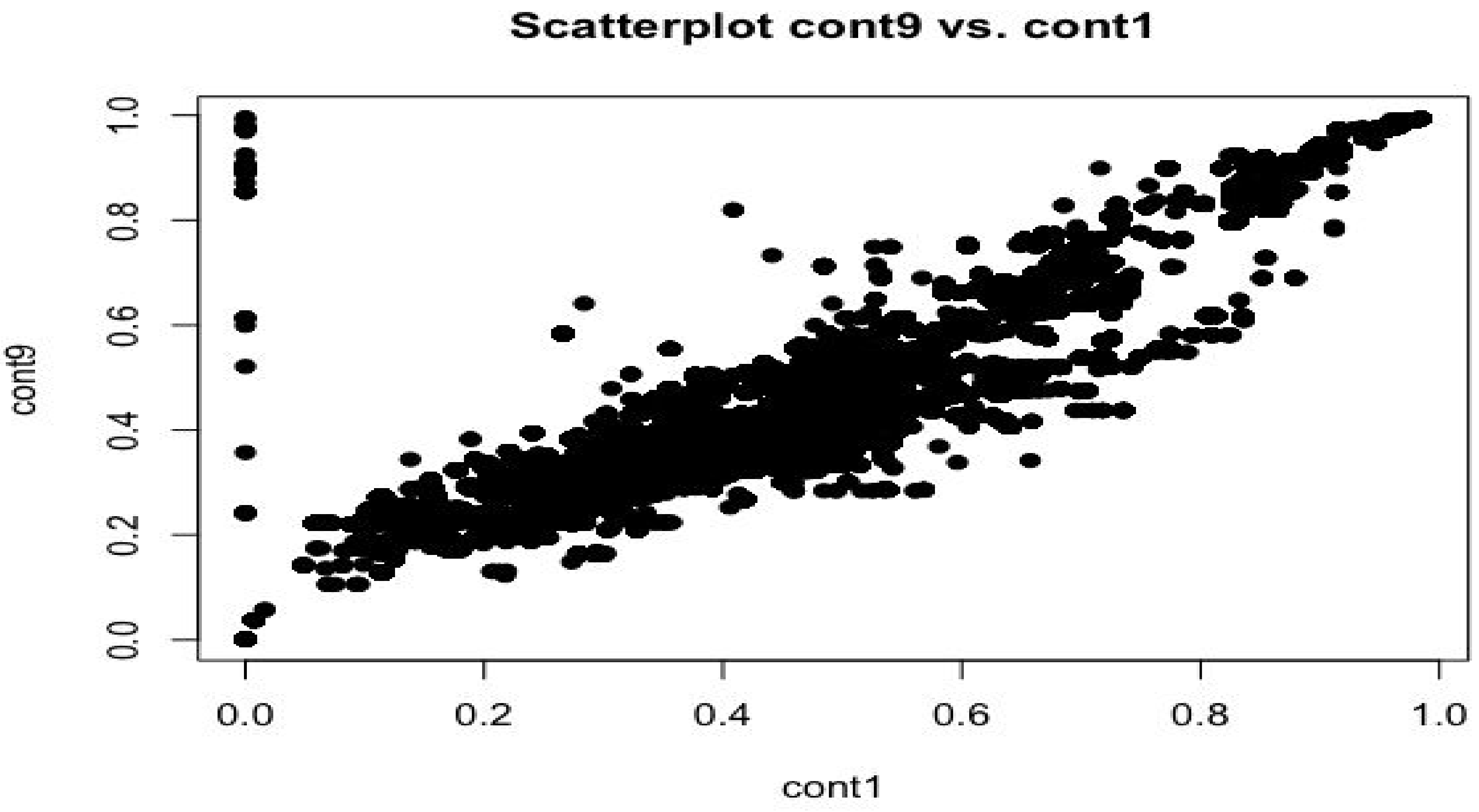
Continuous Variables: 14

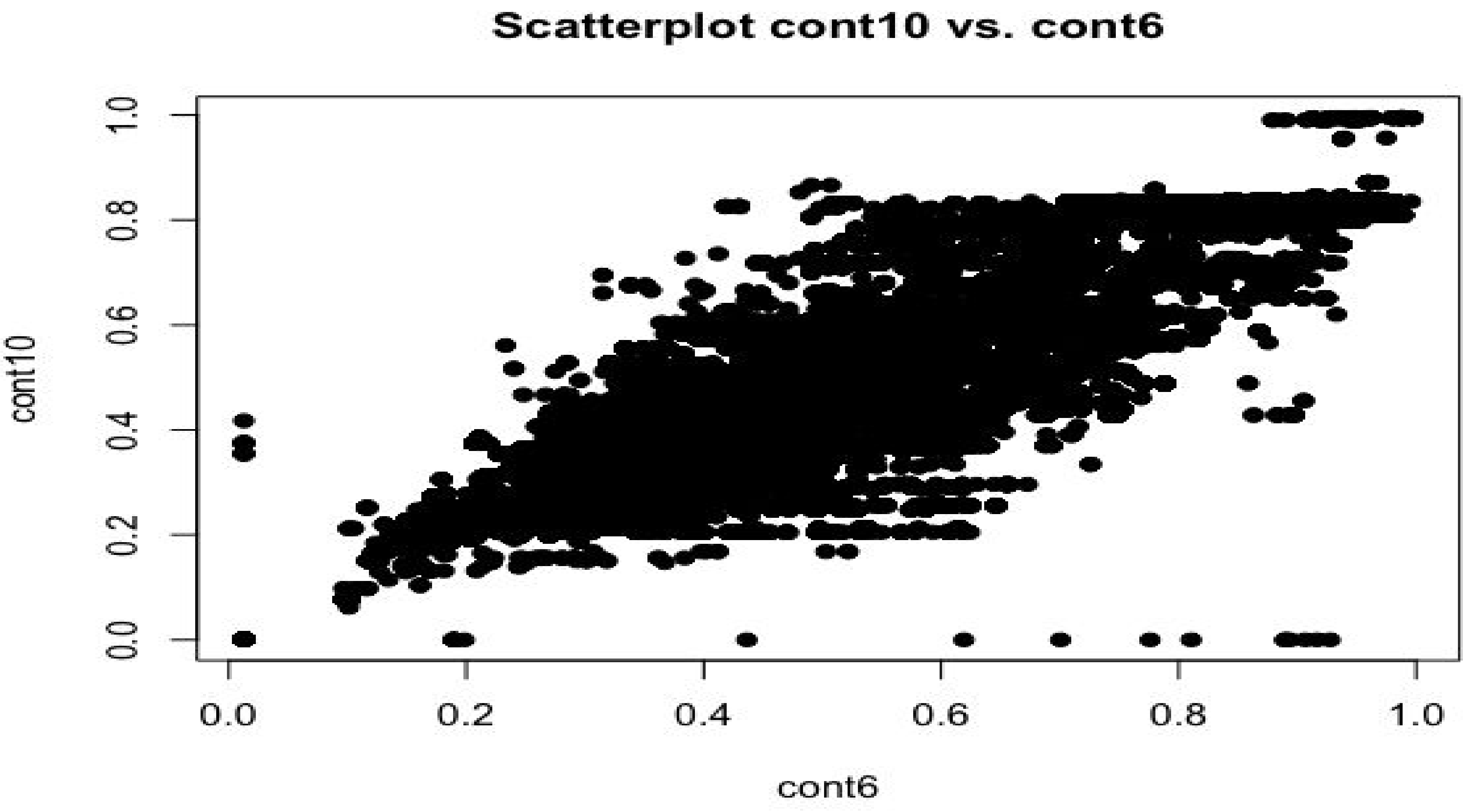


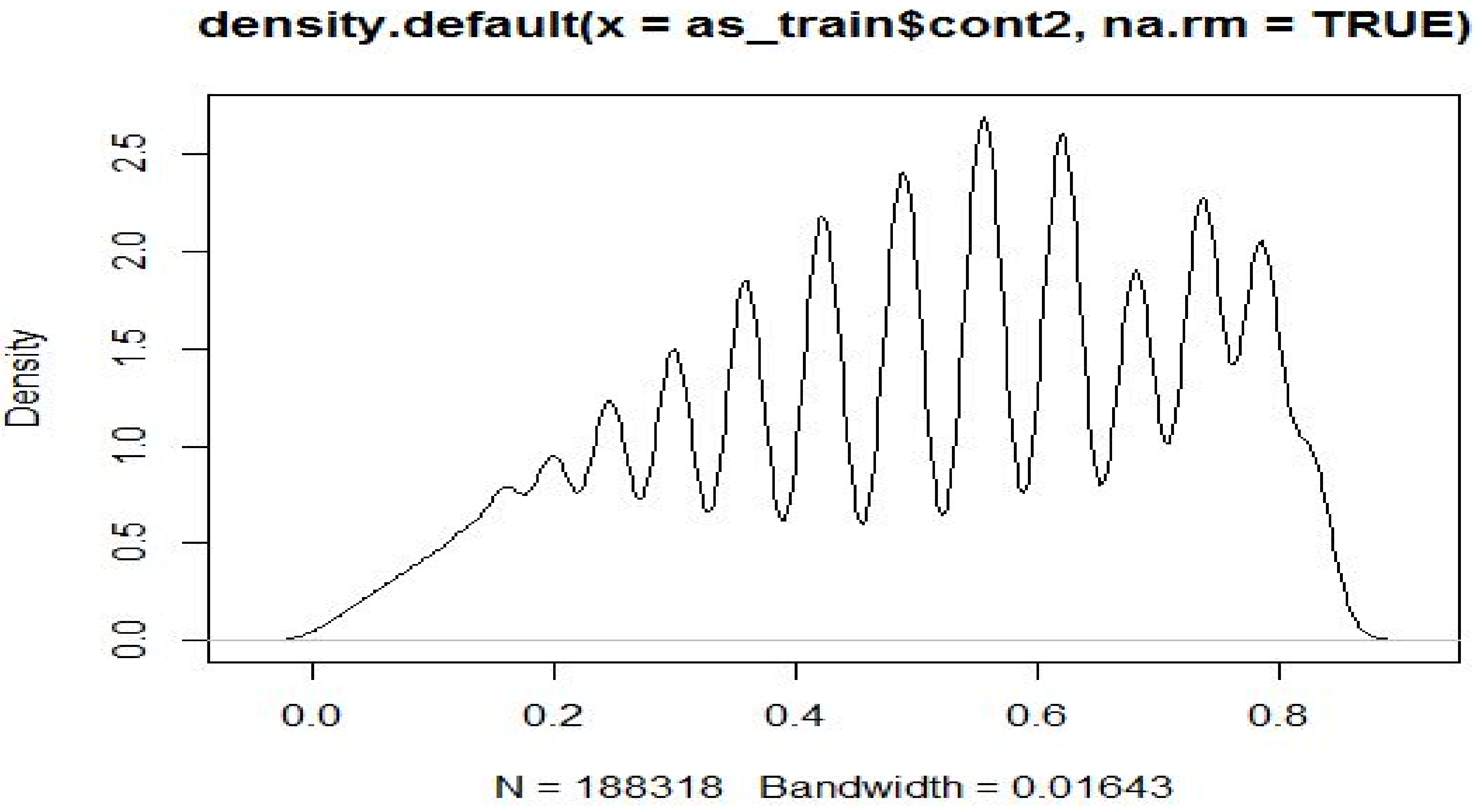
cont1
cont2
cont3
cont4
cont5
cont6
cont7
cont8
cont9
cont10
cont11
cont12
cont13
cont14
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8

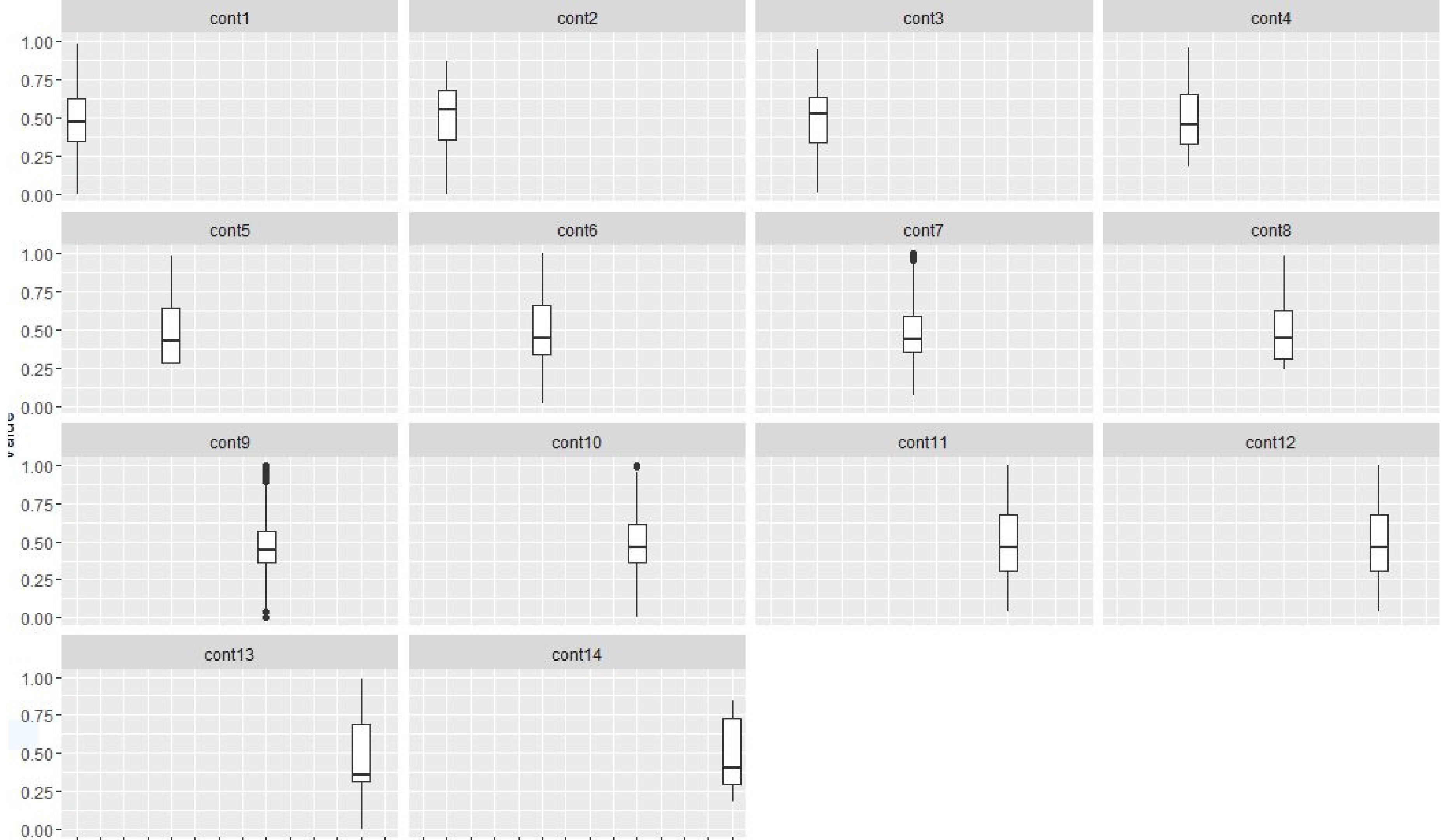


Scatterplot cont12 vs. cont11
cont12
cont11
0.0
0.2
0.4
0.6
0.8
1.0

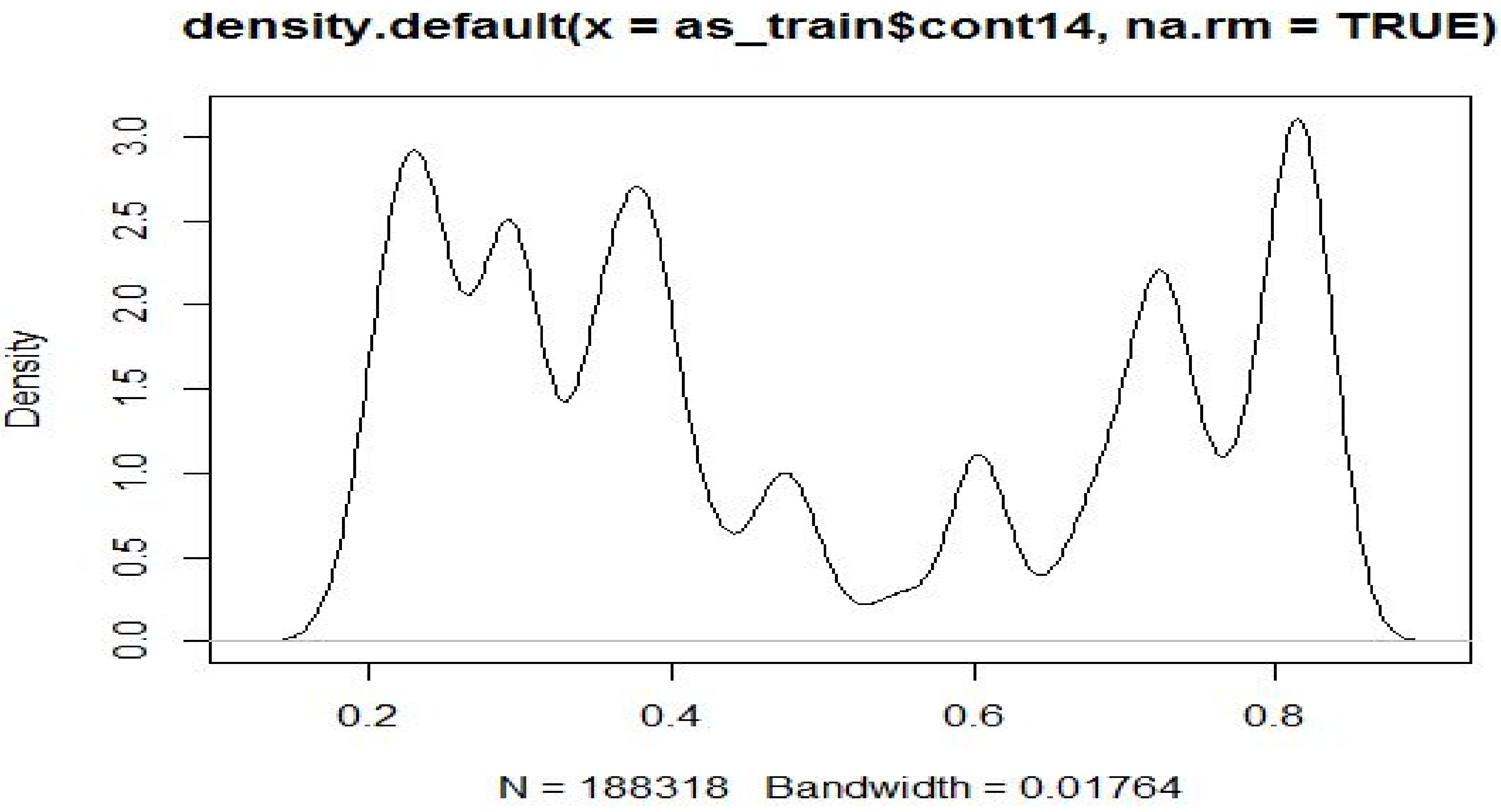

Scatterplot cont9 vs. cont1
cont9
cont1
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0

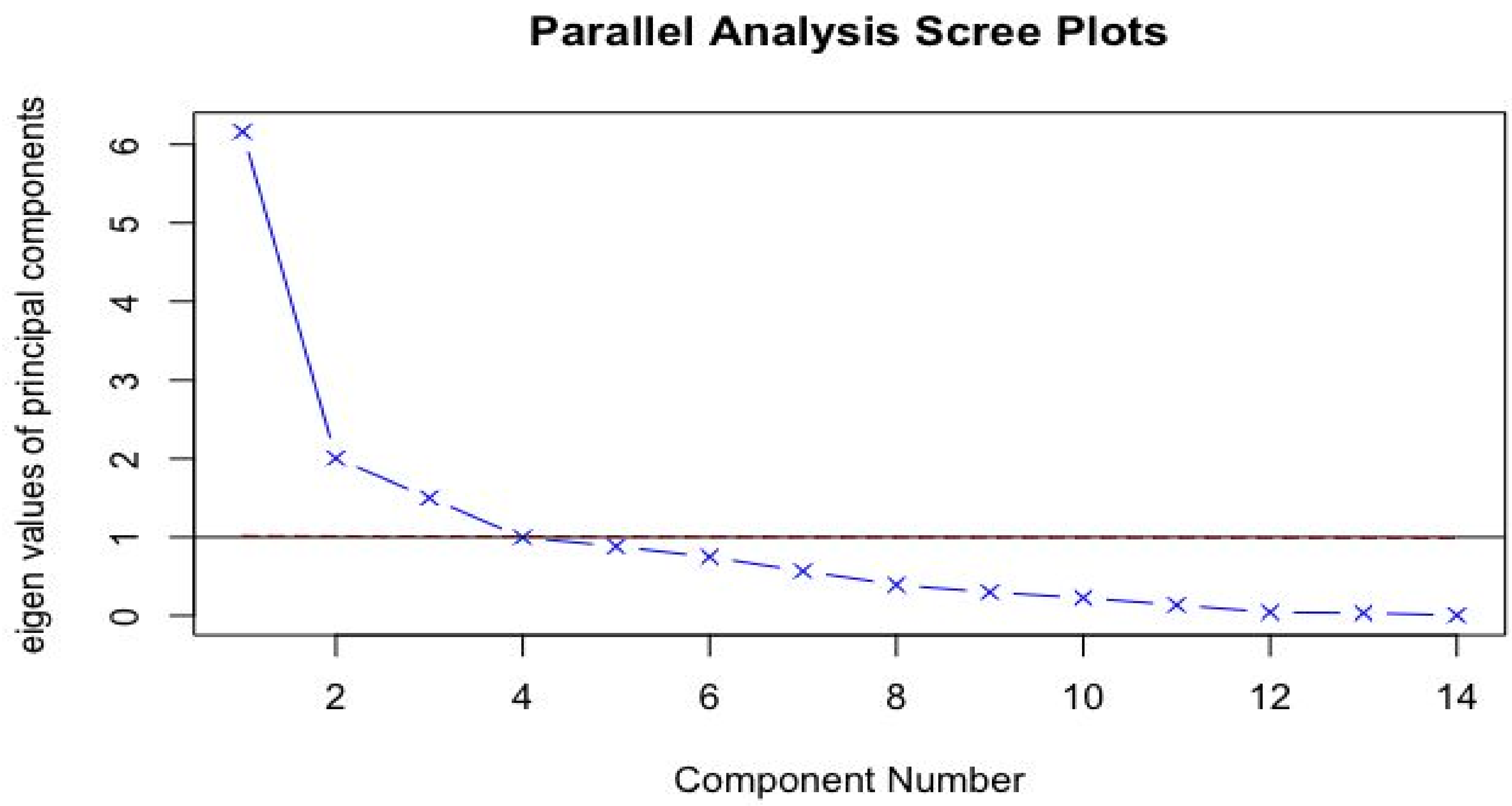

Scatterplot cont10 vs. cont6
cont10
0.0
0.2
0.4
0.6
0.8
1.0
cont6
0.0
0.2
0.4
0.6
0.8
1.0


density.default(x = as_train$cont2, na.rm = TRUE)
Density
0.0
0.5
1.0
1.5
2.0
2.5
0.0
0.2
0.4
0.6
0.8
N = 188318   Bandwidth = 0.01643



cont1
cont2
cont3
cont4
cont5
cont6
cont7
cont8
cont9
cont10
cont11
cont12
cont13
cont14
1.00
0.75
0.50
0.25
0.00


density.default(x = as_train$cont14, na.rm = TRUE)
Density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.2
0.4
0.6
0.8
N = 188318 Bandwidth = 0.01764


Parallel Analysis Scree Plots
eigen values of principal components
0
1
2
3
4
5
6
2
4
6
8
10
12
14
Component Number

Principal Components Analysis
Call: principal(r = st[c(129:142)], nfactors = 3, residuals = TRUE,
rotate = "none")

Standardized loadings (pattern matrix) based upon correlation matrix

	PC1	**PC2**	**PC3**
cont1	0.86	-0.23	-0.04
cont2	0.02	0.38	0.74
cont3	-0.35	0.74	0.38
cont4	0.33	-0.57	0.45
cont5	-0.15	- 0.19	0.60
cont6	0.96	0.04	-0.06
cont7	0.63	0.59	-0.17
cont8	0.51	-0.30	0.44
cont9	0.88	-0.21	-0.02
cont10	0.91	-0.03	0.01
cont11	0.84	0.44	0.03
cont12	0.85	0.42	0.02
cont13	0.76	-0.20	0.00
cont14	0.07	-0.05	-0.08

	PC1	**PC2**	**PC3**	
SS loadings	6.16	2.00	1.50	eigen values of components (magnitude and direction)
Proportion Var	0.44	0.14	0.11	PC1 explains 44% of variability
Cumulative Var	0.44	0.58	0.69	PC2 explains 14%
Proportion Explained	0.64	0.21	0.16	PC3 explains 11%
Cumulative Proportion	0.64	0.84	1.00	All 3 explain 69%

Mean item complexity = 1.6
Test of the hypothesis that 3 components are sufficient.

The root mean square of the residuals (RMSR) is 0.07

Fit based upon off diagonal values = 0.97

Should 4th component be extracted?
[1] 6.158504761 2.004966733 1.499272561 **0.995995109** 0.882372169
[6] 0.750377613 0.566765246 0.394354906 0.300828147 0.228558295
[11] 0.136052628 0.043266719 0.033515221 0.005169893

Linear Regression

1. Removed the columns of continuous variables with correlation >0.75

   So I removed: 4 cont columns

2. Removed the columns of binary levels which one of the level contains < 1% records: 29 category columns removed

   Ex: cat22: A:188275 B: 43

3. Removed the columns of category variables which have highly dependent to other category variables (can be explained >75%)

   24 columns of category columns removed

The dependency was determined by:

GKtau(as_train_f09$cat3,as_train_f09$cat90)

xName	yName	Nx	Ny	tauxy	tauyx
as_train_f09$cat3	as_train_f09$cat90	2	7	0.923	1

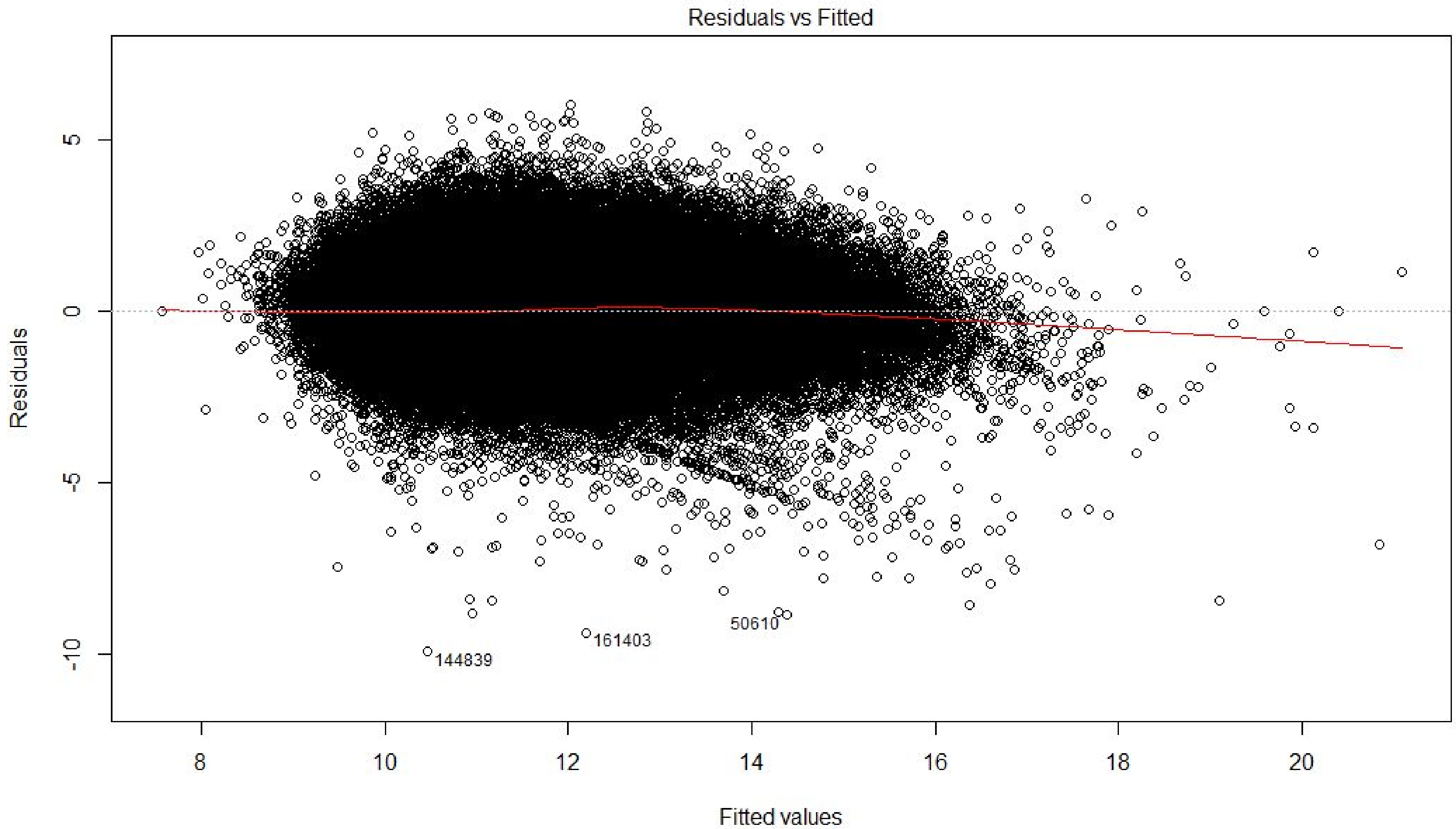
And the columns were removed by X-square test on these two columns

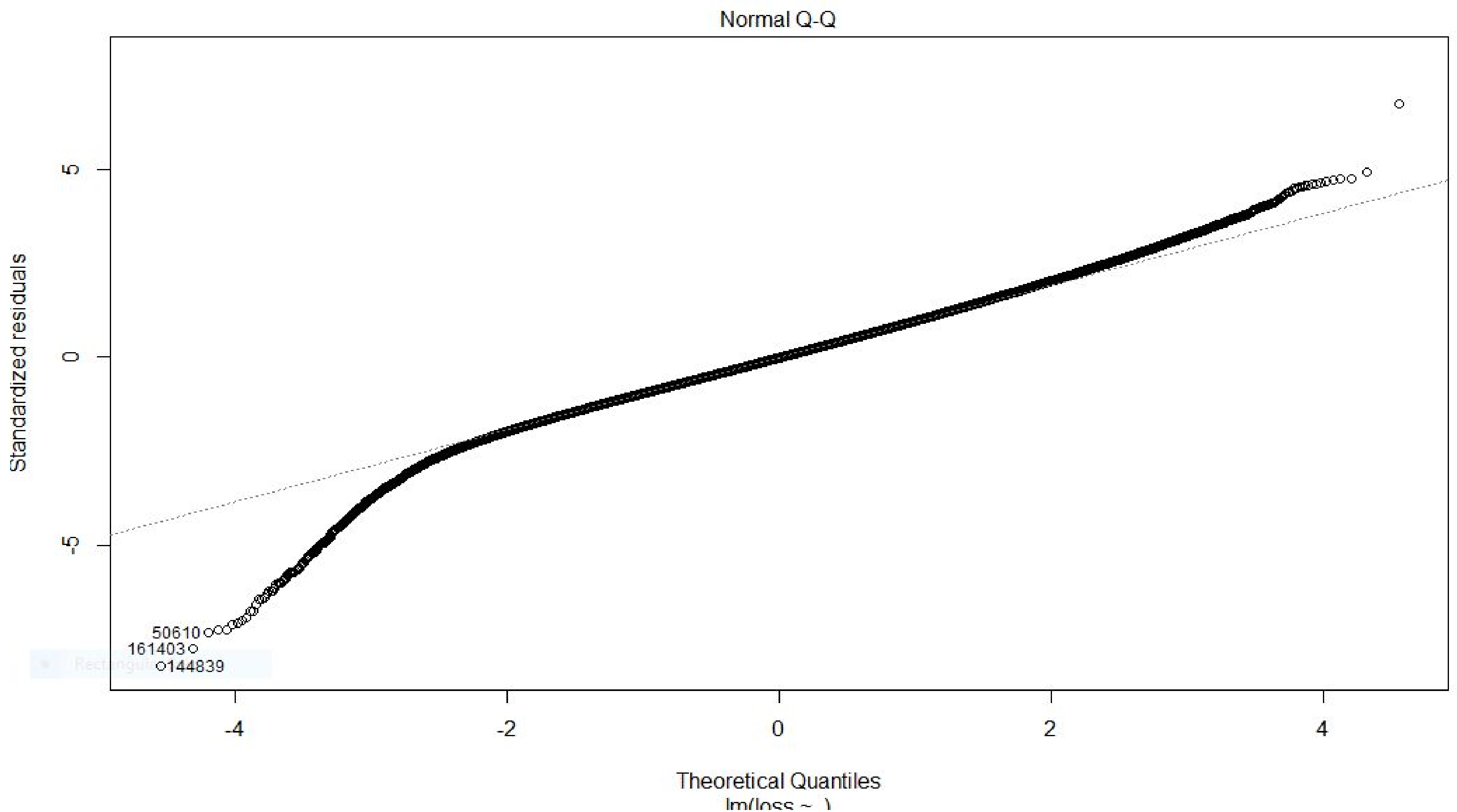
4. BoxCox transformed the loss variable.

Lambda = 0.1

5. Preprocess to cut columns which has <5% rows to avoid the singularity on linear regression process.

6. Adjusted R-squared: 0.50


Residuals vs Fitted
Residuals
Fitted values
5
0
-5
-10
8
10
12
14
16
18
20
144839
161403
50610


Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(loss ~ .)
5
0
-5
-4
-2
0
2
4
50610
161403
144839

The plots show:

Variables are still not independent

Residual is not normally distributed

The simple run of the lm does not give much predict power

Power transformation and feature combination have been tried to reduce non-normal distribution and did not show much improvement so far.

Machine Learning

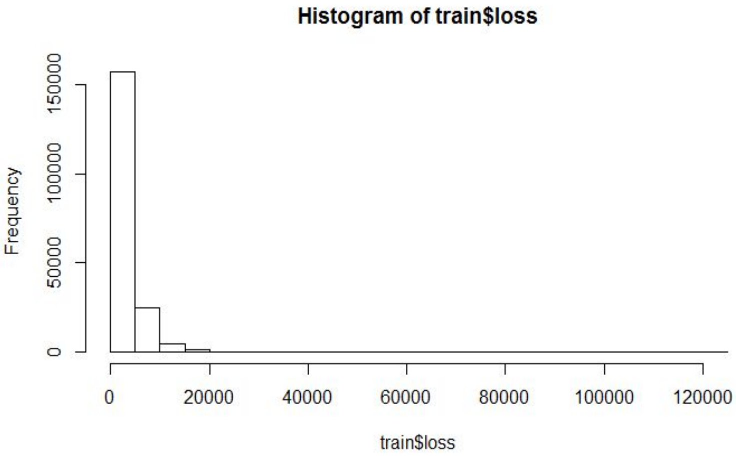
- Linear Regression (lm)
- Lasso (glmnet)
- Stochastic Gradient Boosting (gbm)
- eXtreme Gradient Boosting (xgboost)

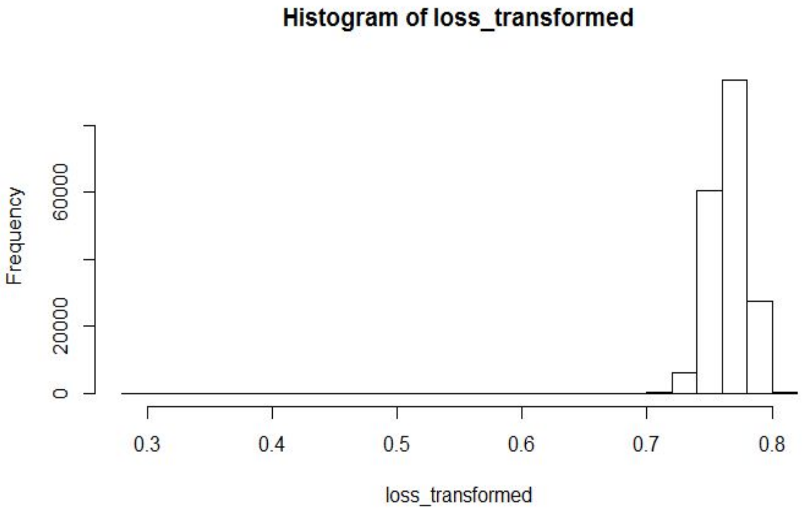
Work Flow

- Pre-processing
- Data Splitting
- Model Tuning Using Resampling
- Fitting Models
- Predict

Pre-processing

- Data Transformation: log(loss)
- Creating Dummy Variables
- Excluding near zero-variance predictors





Data Splitting

- Creating Partition of 80 percent for training data from train dataset.
- 20 percent for testing.

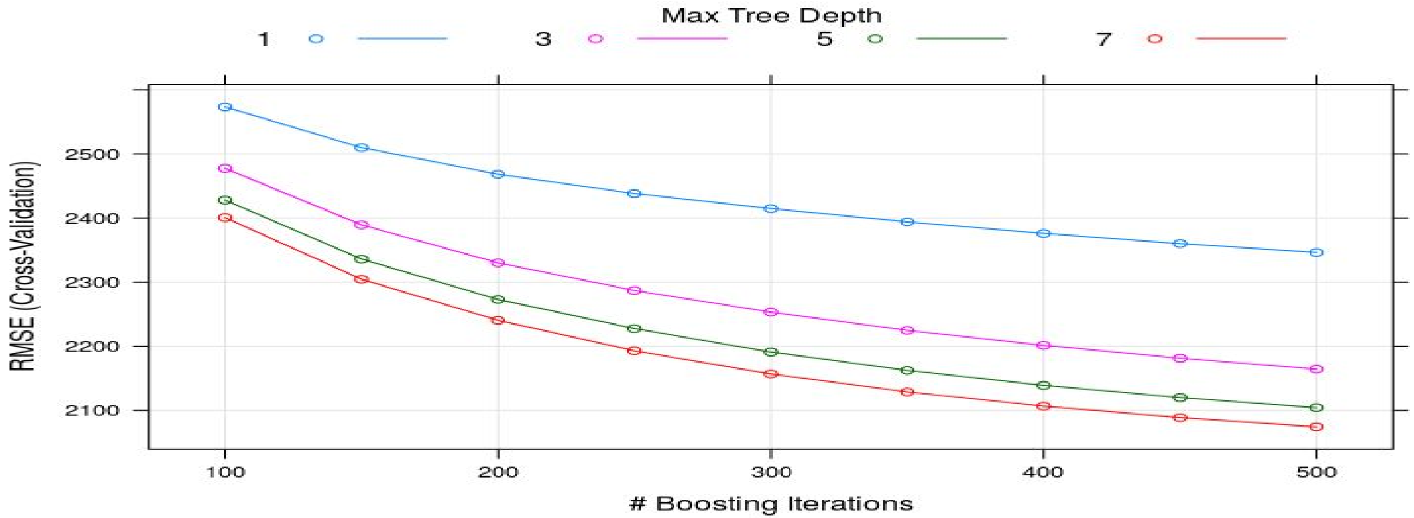
Resampling Method and Tuning

- Repeated Cross-Validation
  - 10-fold cv
  - Repeats 3 times.
- gbm
  - n.trees = 100, 150, 200, 250,...500, interaction.depth = 1,3,5,7, shrinkage = 0.01, n.minobsinnode = 20
- glmnet (lasso)
  - lambda range from 0.01 to 100,000 with 100 equal space
- xgboost
  - nrounds = 1000, max_depth = 4,6,8,10,12,14,16, eta = 0.01, gamma = 1, colsample_bytree = 0.5, min_child_weight = 80,100,120, subsample = 0.7

Metric: RMSE (Root mean square error)

Gradient Boosting

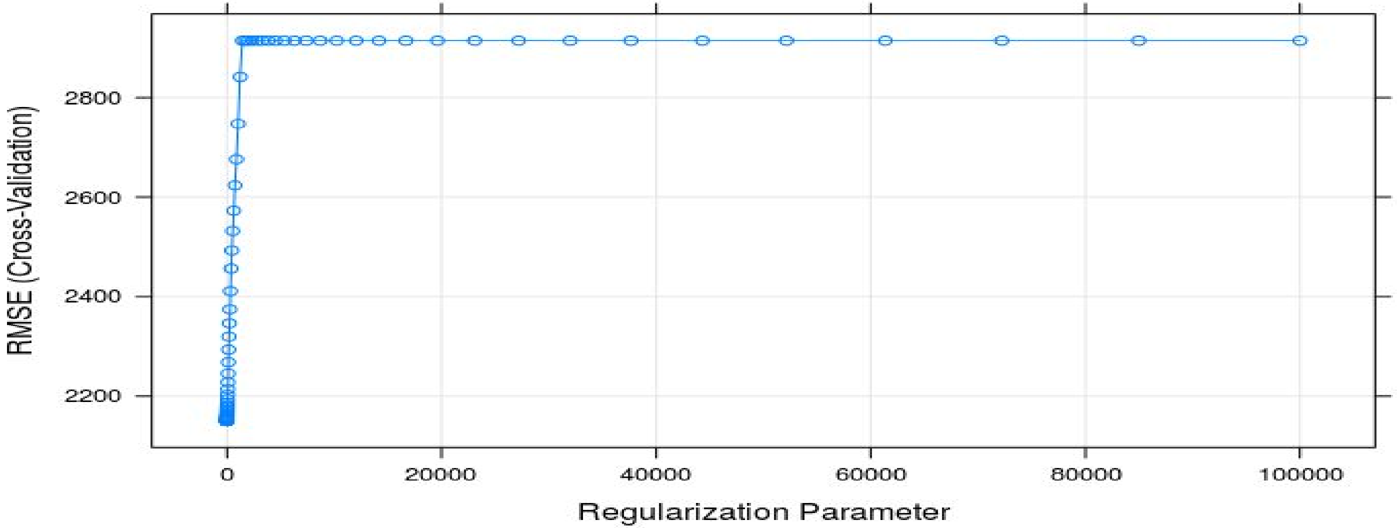
- Fitting n.trees = 500, interaction.depth = 7, shrinkage = 0.01, n.minobsinnode = 20
- RMSE = 2074.632



Lasso

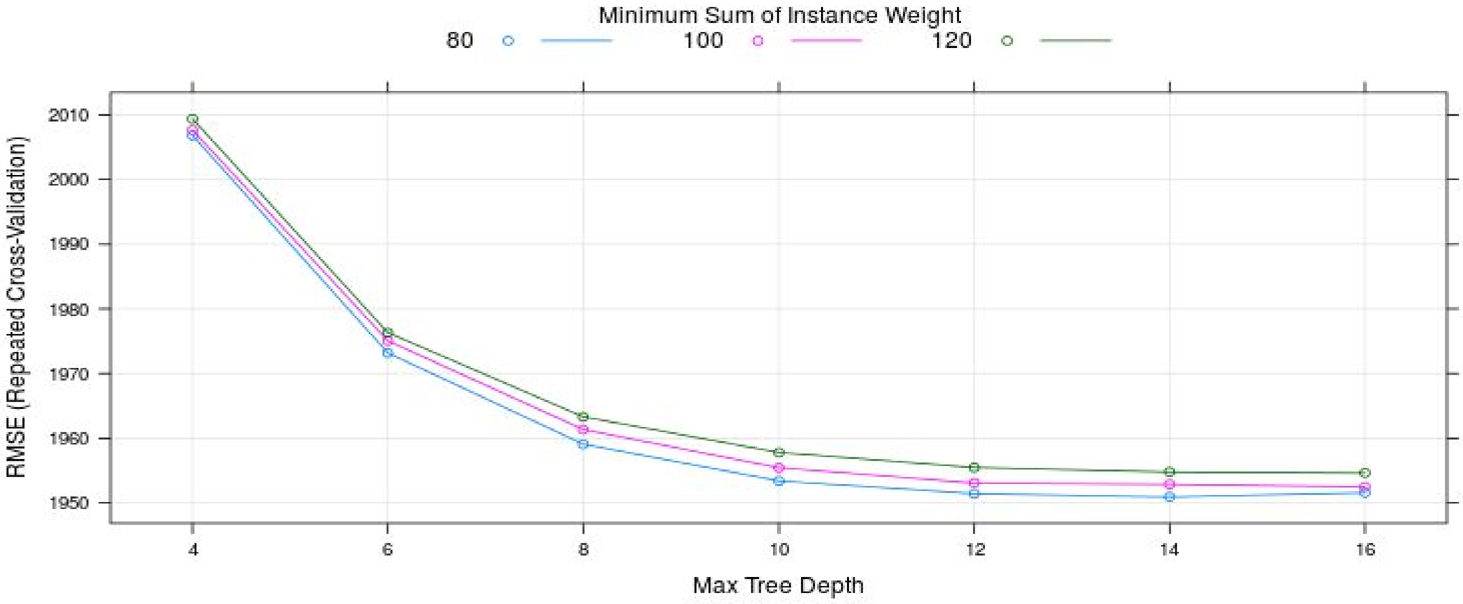
- alpha = 1, lambda = 0.221

RMSE = 2085.1974905 Rsquared = 0.4679428 MAE = 1351.189



eXtreme Gradient Boosting

- nrounds = 1000, max_depth = 14, eta = 0.01, gamma = 1, colsample_bytree = 0.5, min_child_weight = 80, subsample = 0.7



eXtreme Gradient Boosting

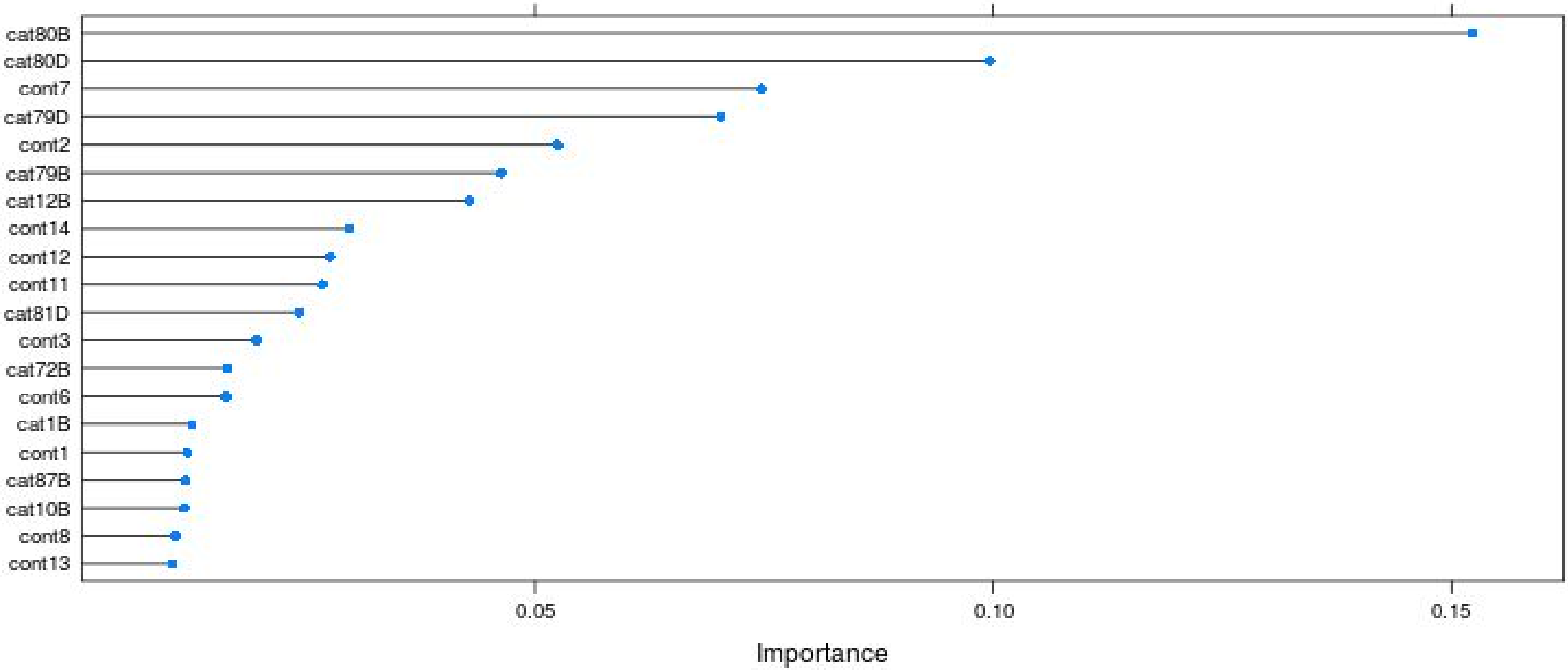
max_depth	min_child_weight	RMSE	Rsquared
14	80	1950.950	0.5523391

Mean Square Error

```
> sum(abs(predicted - lossTest)) / length(lossTest)

[1] 1199.845
```

Variance Importance



Overfitting

- Xgboost Mean square error on 20% test data is 1199.845, submission MAE is 1343.23
- Severely overfitted.
- Reason:
  - Only one data splitting. More.
  - Test dataset may have additional level in categorical variable.
    - example : cat92 -- 7 (train) vs. 8 (test), cat103 -- 13 (train) vs. 14 (test)
- Ensemble

Conclusion

- Best Model: xgboost
- Variable importance to determine which feature impact loss more