APAN5420 — HW 7, Credit Card Transactions

Megan Wilder

7/10/18

1 MeanShift Modeling Technique

1.1 MeanShift Method

The mean shift algorithm is a non parametric clustering technique. It can handle clusters of various shapes and sizes and prior knowledge of the number of clusters is not required (source: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/TUZEL1/MeanShift.pdf (http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/TUZEL1/MeanShift.pdf)). For each data point, mean shift defines a window around it and computes the mean of the data point. It then shifts the center of the window towards the mean and repeats the algorithm until it converges (source: https://saravananthirumuruganathan.wordpress.com/2010/04/01/introduction-to-mean-shift-algorithm/ (https://saravananthirumuruganathan.wordpress.com/2010/04/01/introduction-to-mean-shift-algorithm/)). For fraud analysis, data points far from the centroids can be ignored by the clustering process and flagged as anomalies. These anomaly points could be possible fraudulent transactions.

1.2 MeanShift Model

Hyperparameter tuned include: h: a positive bandwidth parameter. Larger values of h produce few and large clusters. Smaller values of h produce many small clusters. (source: https://www.rdocumentation.org/packages/MeanShift/versions/1.1-1/topics/msClustering (https://www.rdocumentation.org/packages/MeanShift/versions/1.1-1/topics/msClustering)) I analyzed h values of: 10, 20, 30, 50 and 75.

```
#load library
library(MeanShift)
library(ggfortify)

#view dimension of model_df_scale
dim(model_df_scale)
```

```
## [1] 5471   48
```

```
#transpose data
model_df_trans <- t(model_df_scale)

#view dimension of model_df_trans
dim(model_df_trans)
```

```
## [1]   48 5471
```

```
#view head of model_df_trans
kable(head(model_df_trans))
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	
Max_201307	0.0680335	0.0877499	0.1829367	-0.0479014	0.0364684	-0.0147671	0.0664152	-0.0511395	0.0594114	-0.0452959	-0.0511395	-0.0511395	0.0287402	-0.0
Max_201308	-0.0248850	0.0126459	-0.0125401	0.0074599	-0.0269110	-0.0269110	-0.0269110	-0.0269110	-0.0269110	0.0256481	0.0033035	-0.0269110	-0.0269110	-0.0
Max_201309	-0.3637536	-0.3637536	-0.1284096	-0.3637536	-0.3637536	1.1309317	-0.3637536	0.6013264	-0.3637536	3.5187725	-0.3637536	0.3670778	-0.3637536	0.9
Max_201310	-0.2787139	-0.2787139	2.2689856	-0.2787139	0.3158198	1.5839023	-0.2787139	-1.0509373	-0.2787139	-0.2787139	0.5490824	0.3286893	-0.1110626	0.9
Max_201311	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0135197	0.0
Max_201312	-0.3160999	0.3290033	0.0492077	0.7641744	-0.3160999	0.1503852	-0.3160999	0.4494435	0.4152202	0.2430983	-0.3160999	-0.3160999	1.9457632	-0.3

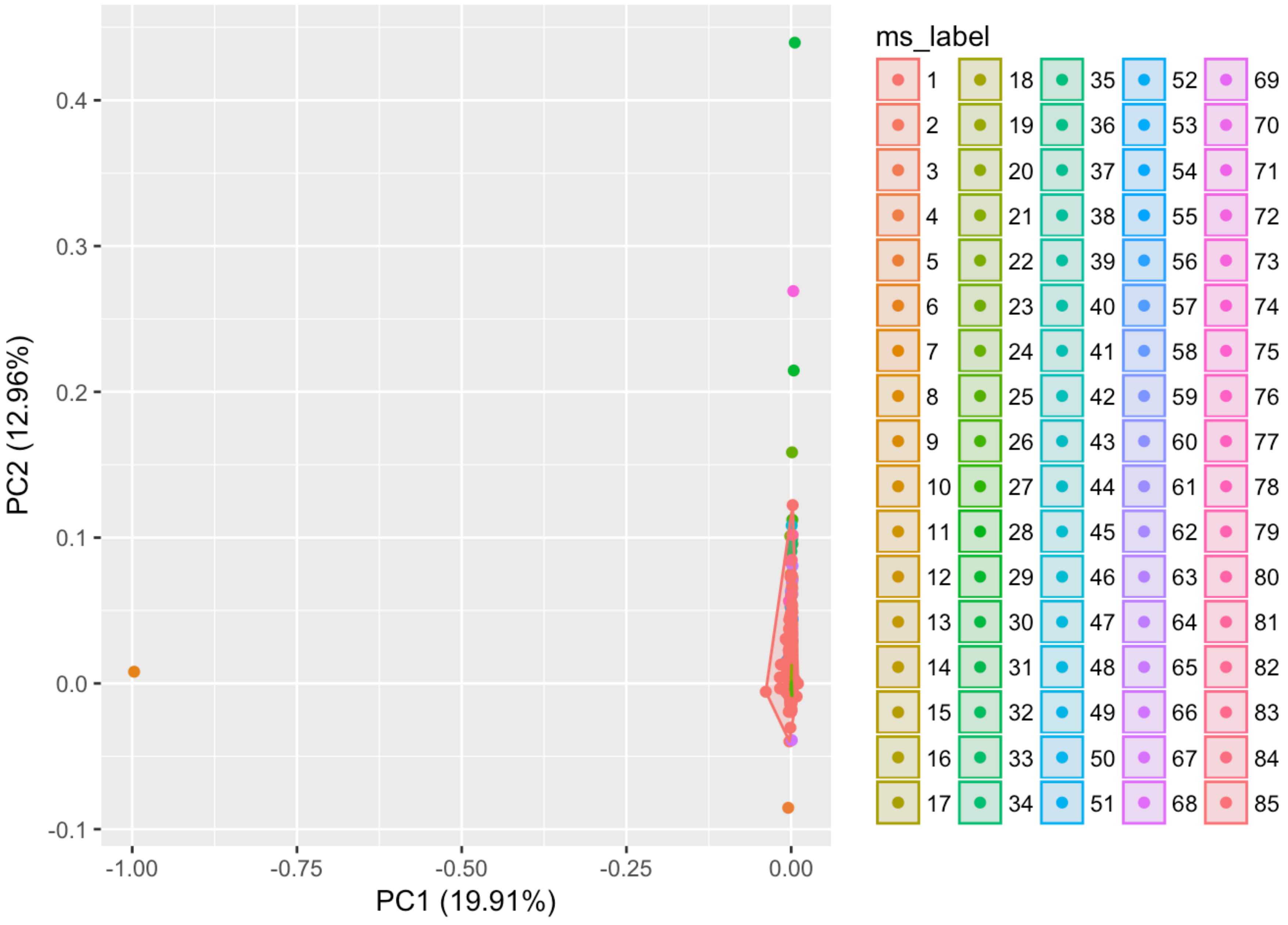
```
#H = 10.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 10.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'ms_label'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'ms_label'] <- as.factor(model_ms[, 'ms_label'])

#view factor levels of ms_label column
levels(model_ms$ms_label)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "23" "24" "25" "26" "27" "28"
## [29] "29" "30" "31" "32" "33" "34" "35" "36" "37" "38" "39" "40" "41" "42"
## [43] "43" "44" "45" "46" "47" "48" "49" "50" "51" "52" "53" "54" "55" "56"
## [57] "57" "58" "59" "60" "61" "62" "63" "64" "65" "66" "67" "68" "69" "70"
## [71] "71" "72" "73" "74" "75" "76" "77" "78" "79" "80" "81" "82" "83" "84"
## [85] "85"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(ms_label) %>%
summarise(no_rows = length(ms_label)) #cluster 1 has 5377 data points, cluster 2 has 3 data points,cluster 3 has
2 data points and clusters 4-85 have one point each.
```

```
## # A tibble: 85 x 2
##    ms_label no_rows
##      <fctr>   <int>
##  1        1    5377
##  2        2       3
##  3        3       2
##  4        4       1
##  5        5       1
##  6        6       1
##  7        7       1
##  8        8       1
##  9        9       1
## 10       10       1
## # ... with 75 more rows
```

```
#plot using principal component anlaysis to reduce high-dimensional data to two dimensions
autoplot(
prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE),
data = model_ms,
colour = 'ms_label',
frame = TRUE
)
```



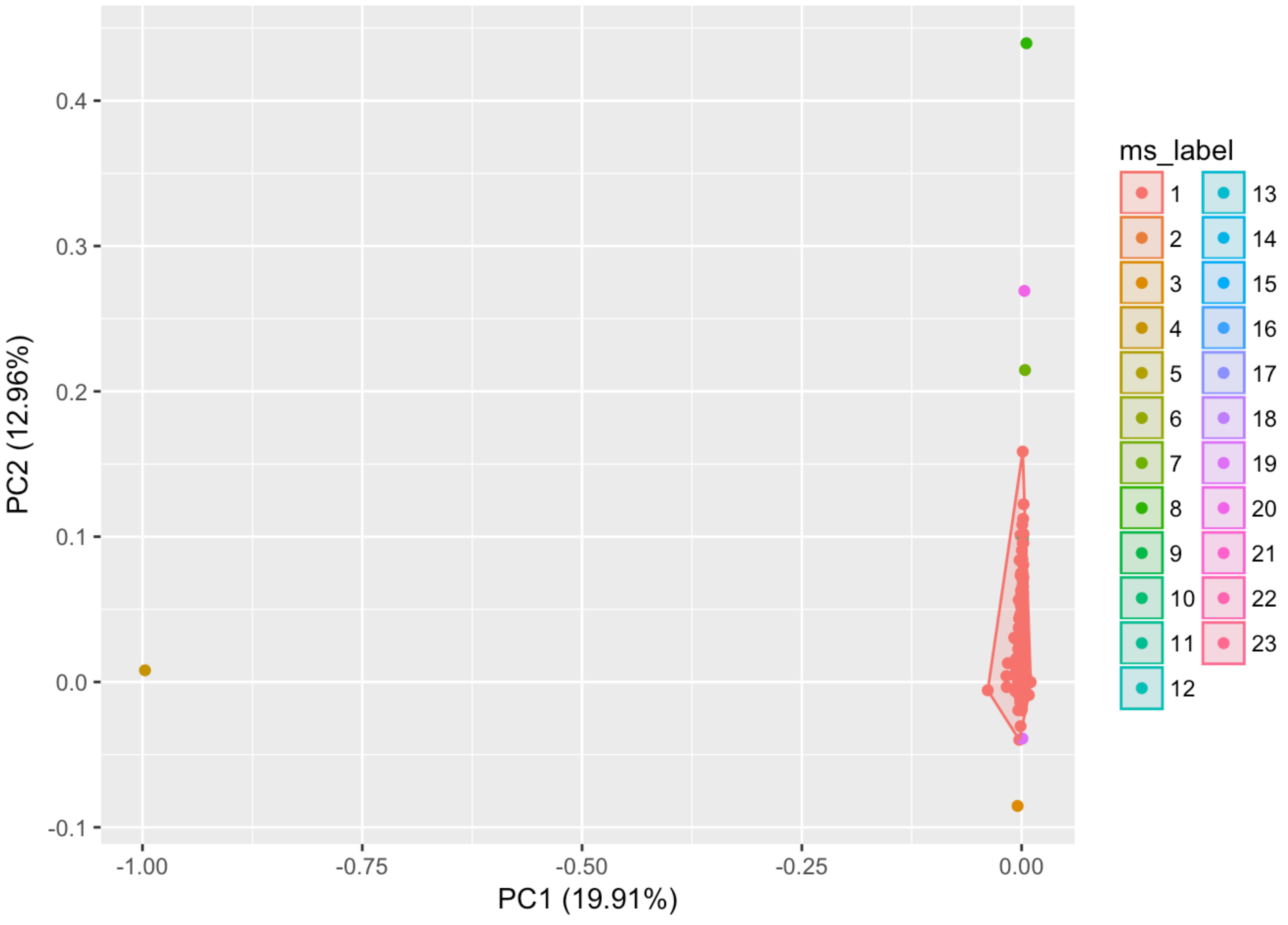
```
#H = 20.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 20.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'ms_label'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'ms_label'] <- as.factor(model_ms[, 'ms_label'])

#view factor levels of ms_label column
levels(model_ms$ms_label)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "23"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(ms_label) %>%
summarise(no_rows = length(ms_label)) #cluster 1 has 5449 data points, clusters 2-23 have one point each.
```

```
## # A tibble: 23 x 2
##    ms_label no_rows
##      <fctr>   <int>
##  1        1    5449
##  2        2       1
##  3        3       1
##  4        4       1
##  5        5       1
##  6        6       1
##  7        7       1
##  8        8       1
##  9        9       1
## 10       10       1
## # ... with 13 more rows
```

```
#plot using principal component anlaysis to reduce high-dimensional data to two dimensions
autoplot(
prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE),
data = model_ms,
colour = 'ms_label',
frame = TRUE
)
```



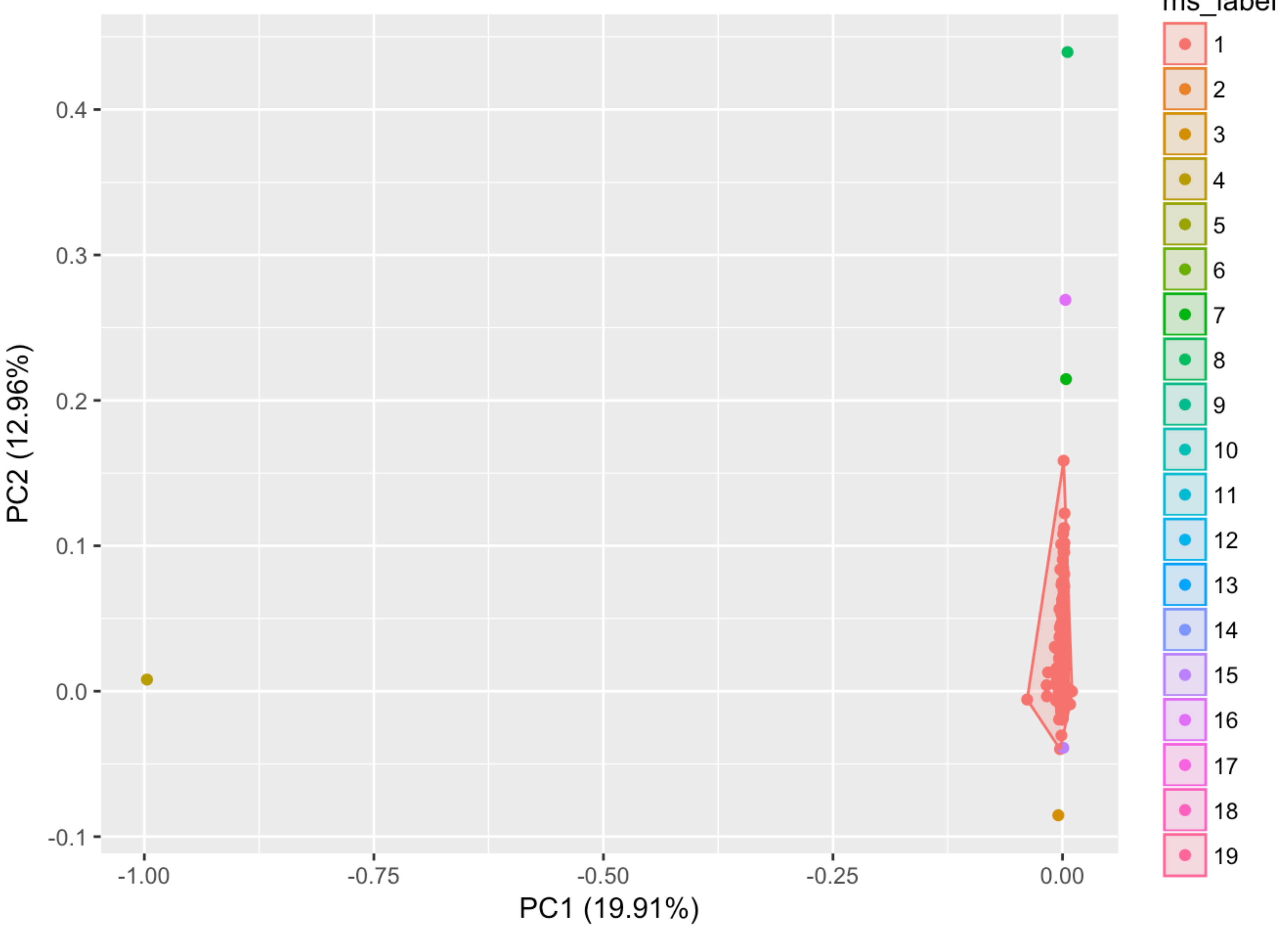
```
#H = 30.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 30.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'ms_label'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'ms_label'] <- as.factor(model_ms[, 'ms_label'])

#view factor levels of ms_label column
levels(model_ms$ms_label)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(ms_label) %>%
summarise(no_rows = length(ms_label)) #cluster 1 has 5453 data points, 2-19 have one point each.
```

```
## # A tibble: 19 x 2
##    ms_label no_rows
##      <fctr>   <int>
##  1        1    5453
##  2        2       1
##  3        3       1
##  4        4       1
##  5        5       1
##  6        6       1
##  7        7       1
##  8        8       1
##  9        9       1
## 10       10       1
## 11       11       1
## 12       12       1
## 13       13       1
## 14       14       1
## 15       15       1
## 16       16       1
## 17       17       1
## 18       18       1
## 19       19       1
```

```
#plot using principal component anlaysis to reduce high-dimensional data to two dimensions
autoplot(
prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE),
data = model_ms,
colour = 'ms_label',
frame = TRUE
)
```



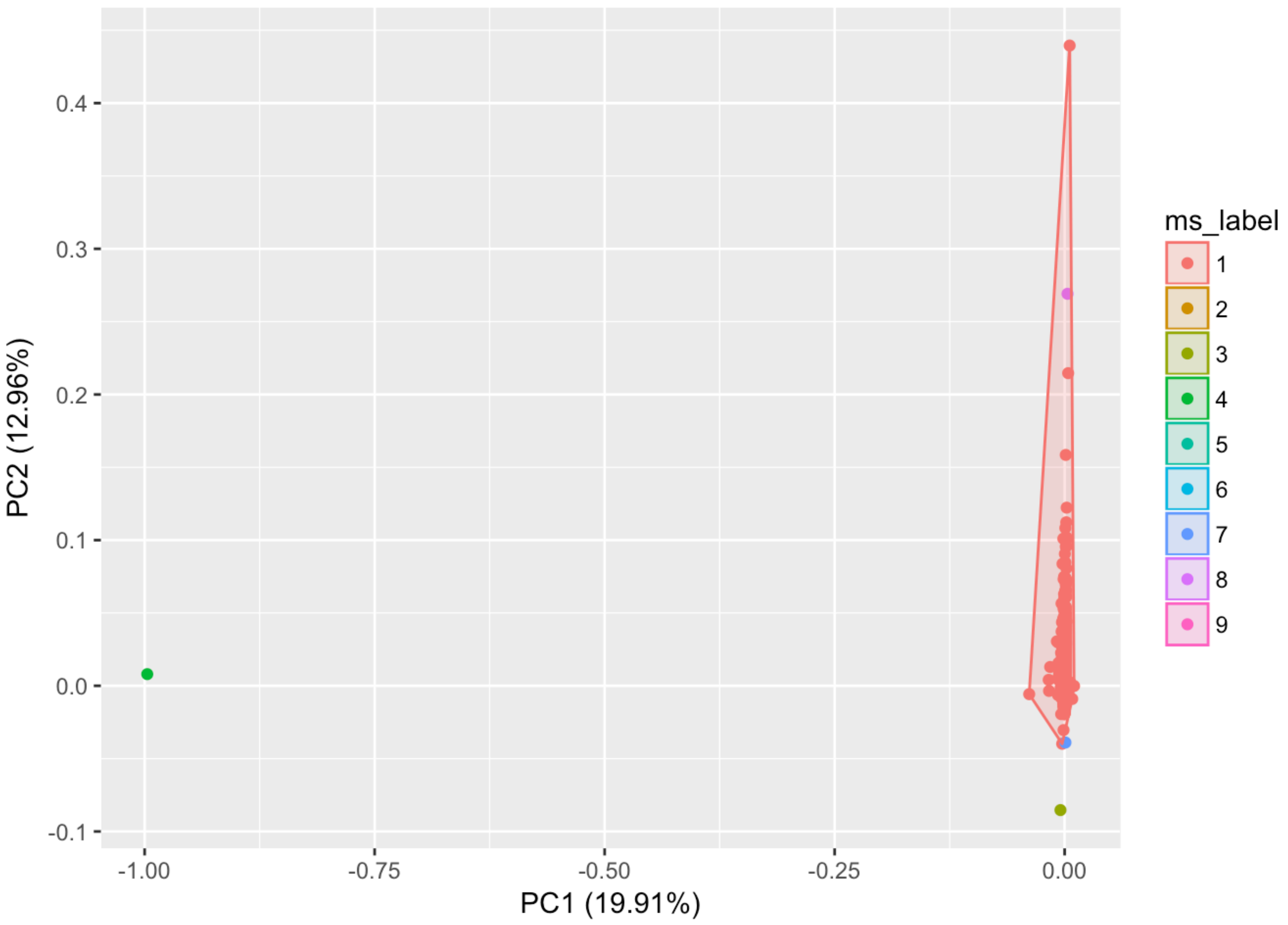
```
#H = 50.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 50.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'ms_label'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'ms_label'] <- as.factor(model_ms[, 'ms_label'])

#view factor levels of ms_label column
levels(model_ms$ms_label)
```

```
## [1] "1" "2" "3" "4" "5" "6" "7" "8" "9"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(ms_label) %>%
summarise(no_rows = length(ms_label)) #cluster 1 has 5463 data points, 2-9 have one point each.
```

```
## # A tibble: 9 x 2
##   ms_label no_rows
##     <fctr>   <int>
## 1        1    5463
## 2        2       1
## 3        3       1
## 4        4       1
## 5        5       1
## 6        6       1
## 7        7       1
## 8        8       1
## 9        9       1
```

```
#plot using principal component anlaysis to reduce high-dimensional data to two dimensions
autoplot(
prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE),
data = model_ms,
colour = 'ms_label',
frame = TRUE
)
```



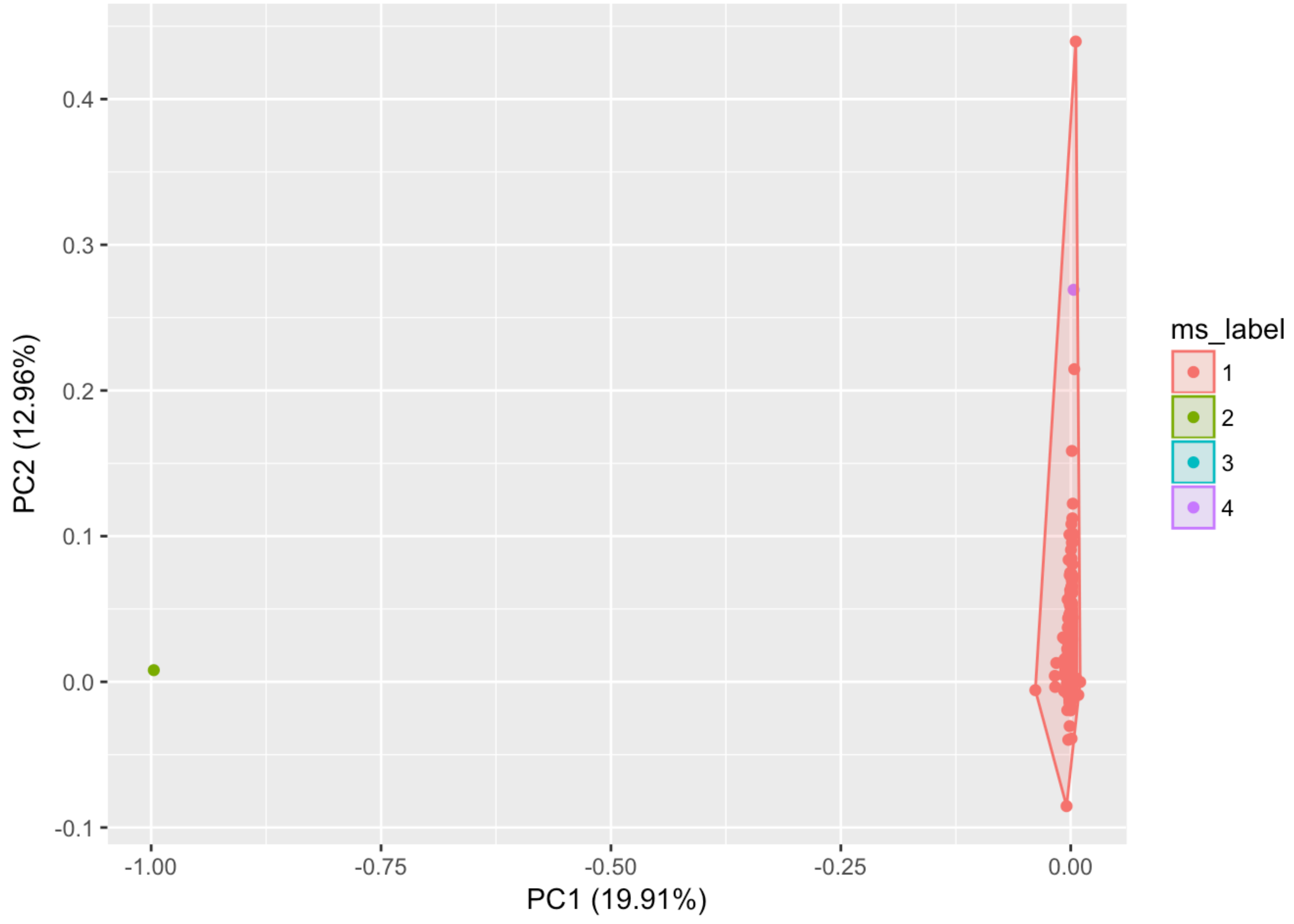
```
#H = 75.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 75.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'ms_label'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'ms_label'] <- as.factor(model_ms[, 'ms_label'])

#view factor levels of ms_label column
levels(model_ms$ms_label)
```

```
## [1] "1" "2" "3" "4"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(ms_label) %>%
summarise(no_rows = length(ms_label)) #cluster 1 has 5468 data points, 2-4 have one point each.
```

```
## # A tibble: 4 x 2
##   ms_label no_rows
##     <fctr>   <int>
## 1        1    5468
## 2        2       1
## 3        3       1
## 4        4       1
```

```
#plot using principal component anlaysis to reduce high-dimensional data to two dimensions
autoplot(
prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE),
data = model_ms,
colour = 'ms_label',
frame = TRUE
)
```



1.3 Best Model with hyperparameter of h=20

It appears that using h of 20 resulted in a reasonable model. It clustered the data points into 1 primary cluster with 22 anomalies.

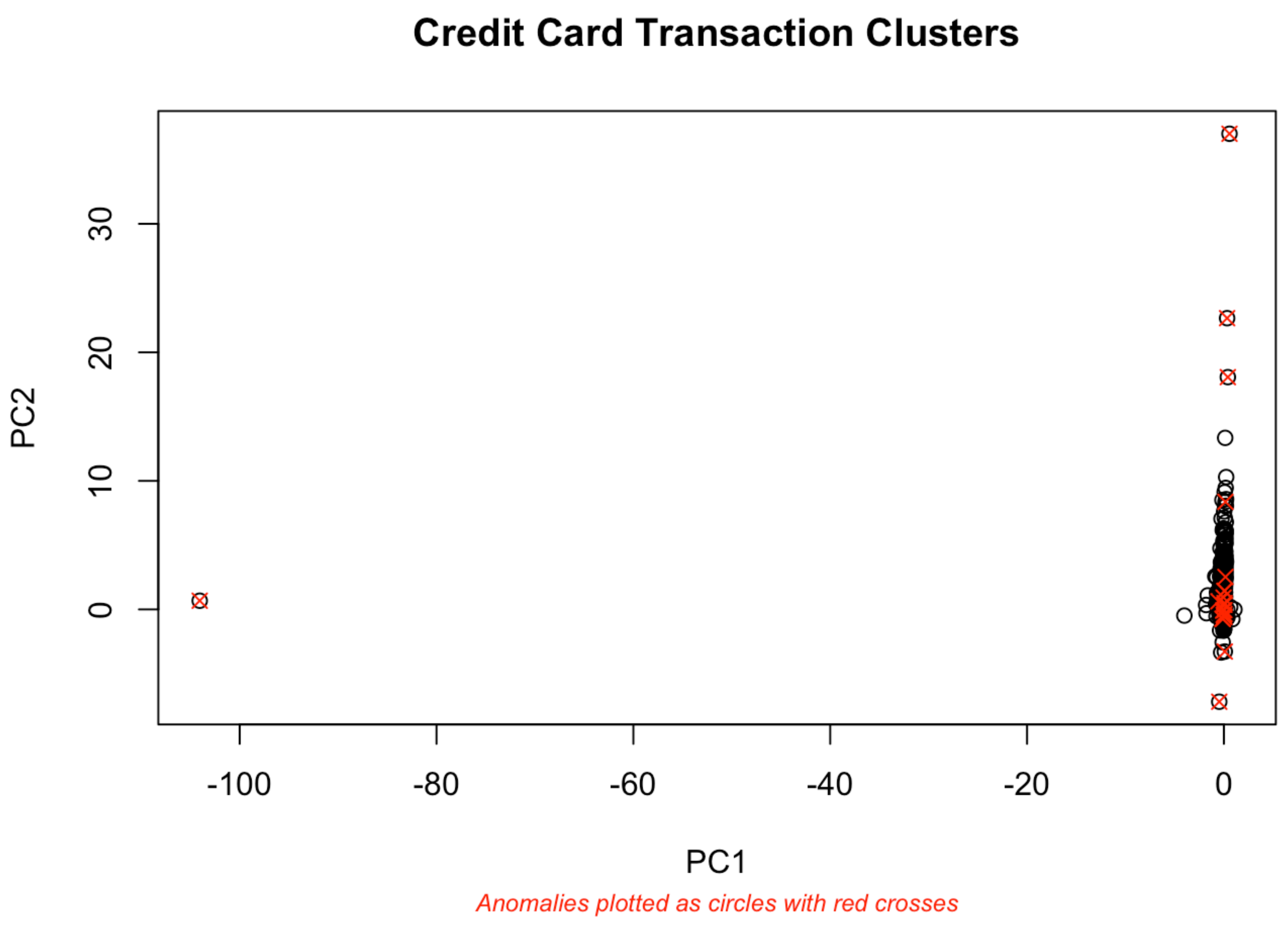
```
#H = 20.0
#MeanShift clustering using multi-core processing
options(mc.cores = 2)
ms_clustering <-
msClustering(model_df_trans, h = 20.0, multi.core = TRUE)

#attach clustering labels to model df
model_ms <- model_df
model_ms[, 'Cluster'] <- ms_clustering[2]

#change labels to factors
model_ms[, 'Cluster'] <- as.factor(model_ms[, 'Cluster'])

#view factor levels of ms_label column
levels(model_ms$Cluster)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14"
## [15] "15" "16" "17" "18" "19" "20" "21" "22" "23"
```

```
#view number of occurances per factor level
model_ms %>%
group_by(Cluster) %>%
summarise(no_rows = length(Cluster)) #cluster 1 has 5449 data points, clusters 2-23 have one point each.
```

```
## # A tibble: 23 x 2
##    Cluster no_rows
##     <fctr>   <int>
##  1       1    5449
##  2       2       1
##  3       3       1
##  4       4       1
##  5       5       1
##  6       6       1
##  7       7       1
##  8       8       1
##  9       9       1
## 10      10       1
## # ... with 13 more rows
```

```
#principal component anlaysis to reduce high-dimensional data to two dimensions
fraud_PCA <- prcomp(model_ms[, 3:12], center = TRUE, scale. = TRUE)
fraud_PCA2 <- fraud_PCA$x[, 1:2]

#plot cluster 1 add anomalies as crosses.
plot(
fraud_PCA2,
main = "Credit Card Transaction Clusters",
sub = "Anomalies plotted as circles with red crosses",
cex.sub = 0.75,
font.sub = 3,
col.sub = "red"
)
points(fraud_PCA2[model_ms$Cluster != 1, ], pch = 4, col = "red")
```



1.4 Anomalies

```
#create DF that include possible fraud transactions
fraud <- model_df[model_ms$Cluster != 1,]
fraud <- fraud[, 1:2]

#view fraud
kable(fraud) %>% kable_styling(latex_options = "scale_down")
```

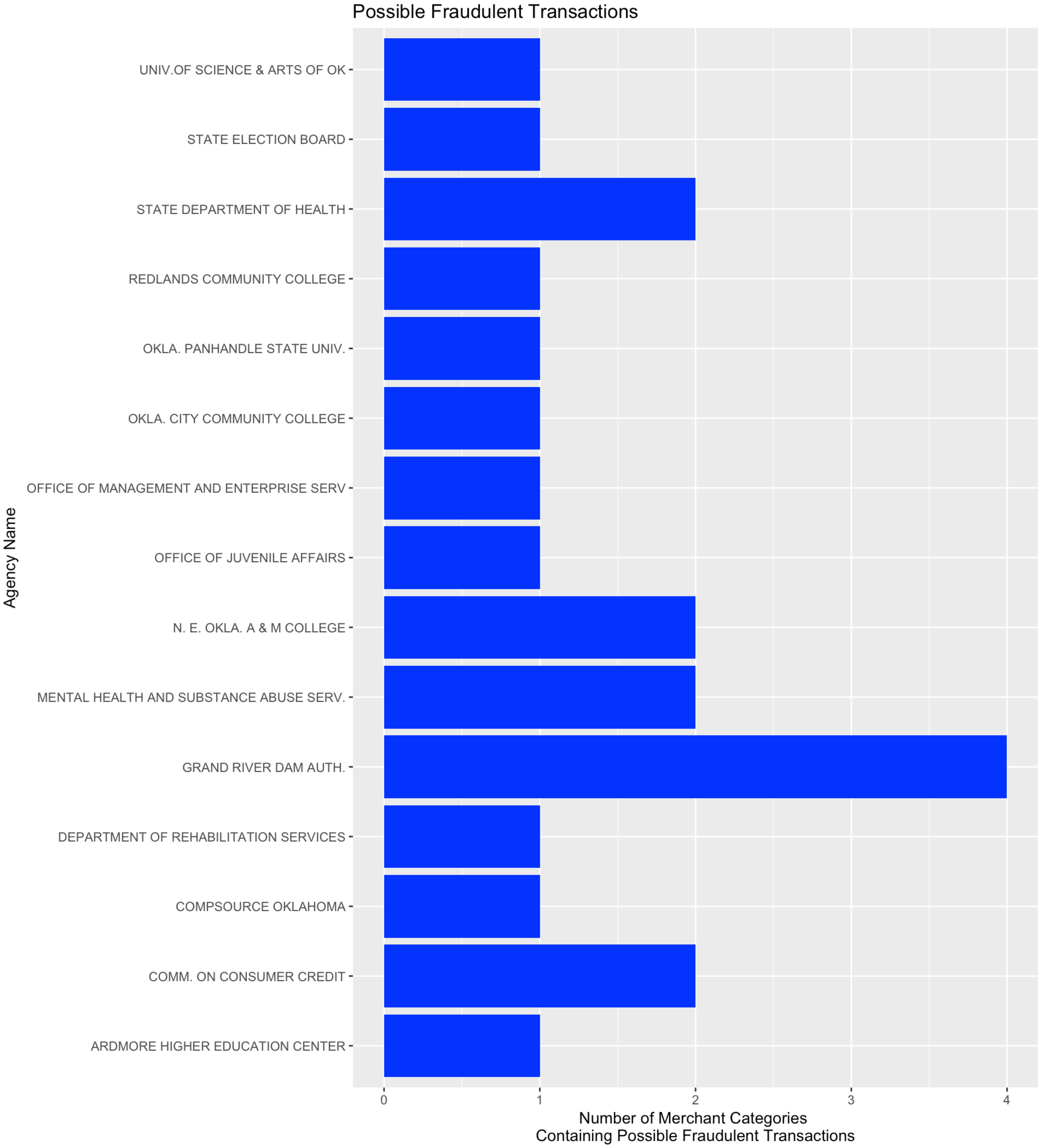
	Agency_Name	Merchant_Category
35	ARDMORE HIGHER EDUCATION CENTER	HOUSEHOLD APPLIANCE STORES
279	COMM. ON CONSUMER CREDIT	GOVERNMENT SERVICES–NOT ELSEWHERE CLASSIFIED
293	COMM. ON CONSUMER CREDIT	SHERATON
327	COMPSOURCE OKLAHOMA	DELTA
1328	DEPARTMENT OF REHABILITATION SERVICES	MISCELLANEOUS FOOD STORES-CONV STRS AND SPECIALTY MKTS.
2159	GRAND RIVER DAM AUTH.	COMMERCIAL EQUIPMENT, NOT ELSEWHERE CLASSIFIED
2165	GRAND RIVER DAM AUTH.	COMPUTERS, COMPUTER PERIPHERAL EQUIPMENT, SOFTWARE
2174	GRAND RIVER DAM AUTH.	DETECTIVE AGENCIES,PROTECTIVE AGENCIES,AND SECURITY SERVICES
2194	GRAND RIVER DAM AUTH.	GOVERNMENT SERVICES–NOT ELSEWHERE CLASSIFIED
2614	MENTAL HEALTH AND SUBSTANCE ABUSE SERV.	DEPARTMENT STORES
2687	MENTAL HEALTH AND SUBSTANCE ABUSE SERV.	RENAISSANCE HOTELS
2713	N. E. OKLA. A & M COLLEGE	AUTOMATED FUEL DISPENSER

2752	N. E. OKLA. A & M COLLEGE	RECORD STORES
2782	OFFICE OF JUVENILE AFFAIRS	COMFORT HOTEL INTERNATIONAL
2975	OFFICE OF MANAGEMENT AND ENTERPRISE SERV	TOLLS AND BRIDGE FEES
3330	OKLA. CITY COMMUNITY COLLEGE	GROCERY STORES,AND SUPERMARKETS
3485	OKLA. PANHANDLE STATE UNIV.	RECORD STORES
4165	REDLANDS COMMUNITY COLLEGE	LUXOR HOTEL AND CASINO
4713	STATE DEPARTMENT OF HEALTH	MISCELLANEOUS GENERAL MERCHANDISE
4717	STATE DEPARTMENT OF HEALTH	NON-DURABLE GOODS NOT ELSEWHERE CLASSIFIED
4752	STATE ELECTION BOARD	STATIONERY,OFFICE AND SCHOOL SUPPLY STORES
5156	UNIV.OF SCIENCE & ARTS OF OK	COMFORT HOTEL INTERNATIONAL

1.5 Business Insight

Agency transactions that occurred within the merchant category listed in the fraud data frame could possibly be fraud based on my MeanShift analysis. Transactions that occurred within these merchant categories at these agencies require further analysis to determine if fraud actually occurred. Grand River Dam Auth had the highest number of merchant categories, 4, that might contain fraudulent transactions.

```
#Plot number of merchant categories that contain
#possible fraudulent transaactions by agency
ggplot(data = fraud, mapping = aes(Agency_Name)) + geom_bar(fill = 'blue', stat = "count") + xlab("Agency Name")
+ ylab("Number of Merchant Categories \nContaining Possible Fraudulent Transactions") + ggtitle("Possible Fraudul
ent Transactions") + coord_flip()
```



2 Autoencoder Modeling Technique

2.1 Autoencoder Method

Autoencoder is an unsupervised, neural network algorithm that uses backpropagation, setting the target values equal to the inputs. Autoencoders compress the input into a latent-space representation, and then reconstructs the output from this representation. It is typically used for dimension reduction. (sources: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/ (http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/) https://en.wikipedia.org/wiki/Autoencoder (https://en.wikipedia.org/wiki/Autoencoder))

2.2 Autoencoder Model

Hyperparameters tuned: Hidden layers: 5, 2, 5 and 10, 2, 10 Epochs: 50, 100, 200

```
#source: https://shiring.github.io/machine_learning/2017/05/01/fraud
#load libraries
library(h2o)
library(dplyr)
library(ggplot2)
h2o.init()
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         1 days 21 hours
##     H2O cluster timezone:       America/New_York
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.20.0.2
##     H2O cluster version age:    27 days
##     H2O cluster name:           H2O_started_from_R_meganroesch_evw568
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   3.38 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  4
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         XGBoost, Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.4.1 (2017-06-30)
```

```
# convert data to H2OFrame
model_h <- as.h2o(model_df_scale)
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

```
#view head of h20 model
kable(head(model_h))
```

Max_201307	Max_201308	Max_201309	Max_201310	Max_201311	Max_201312	Max_201401	Max_201402	Max_201403	Max_201404	Max_201405	Max_201406	Med_20
0.0680335	-0.0248850	-0.3637536	-0.2787139	0.0135197	-0.3160999	0.7760440	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	0.076
0.0877499	0.0126459	-0.3637536	-0.2787139	0.0135197	0.3290033	0.2536490	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	0.111
0.1829367	-0.0125401	-0.1284096	2.2689856	0.0135197	0.0492077	-0.1967417	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	0.042
-0.0479014	0.0074599	-0.3637536	-0.2787139	0.0135197	0.7641744	0.6836222	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	-0.133
0.0364684	-0.0269110	-0.3637536	0.3158198	0.0135197	-0.3160999	-0.1967417	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	0.095
-0.0147671	-0.0269110	1.1309317	1.5839023	0.0135197	0.1503852	-0.1570854	-0.382932	-0.282552	-0.2563837	-0.3200084	-0.3912933	-0.032

```
#splitting the dataset into training and test sets
model_train_test <- h2o.splitFrame(model_h,
ratios = c(0.4, 0.4), # must add up to less than 1.0
seed = 42)

train  <- model_train_test[[1]]
test1  <- model_train_test[[2]]
test2  <- model_train_test[[3]]
dim(train)
```

```
## [1] 2184   48
```

```
dim(test1)
```

```
## [1] 2206   48
```

```
dim(test2)
```

```
## [1] 1081   48
```

```
#set features
x_col <- colnames(model_df_scale)

#Hyperparameters: hidden= 5,2,5 epochs = 100
#unsupervised neural network model using deep learning autoencoders (autoencoder = TRUE)
#“bottleneck” training, hidden layer in the middle is very small. Model will reduce the dimensionality of the inp
ut data (in this case, down to 2 nodes/dimensions).
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(5, 2, 5),
epochs = 100,
activation = "Tanh"
)
```

```
##
  |
  |                                                                 |   0%
  |
  |======                                                           |  10%
  |
  |=======================================                          |  60%
  |
  |=================================================================| 100%
```

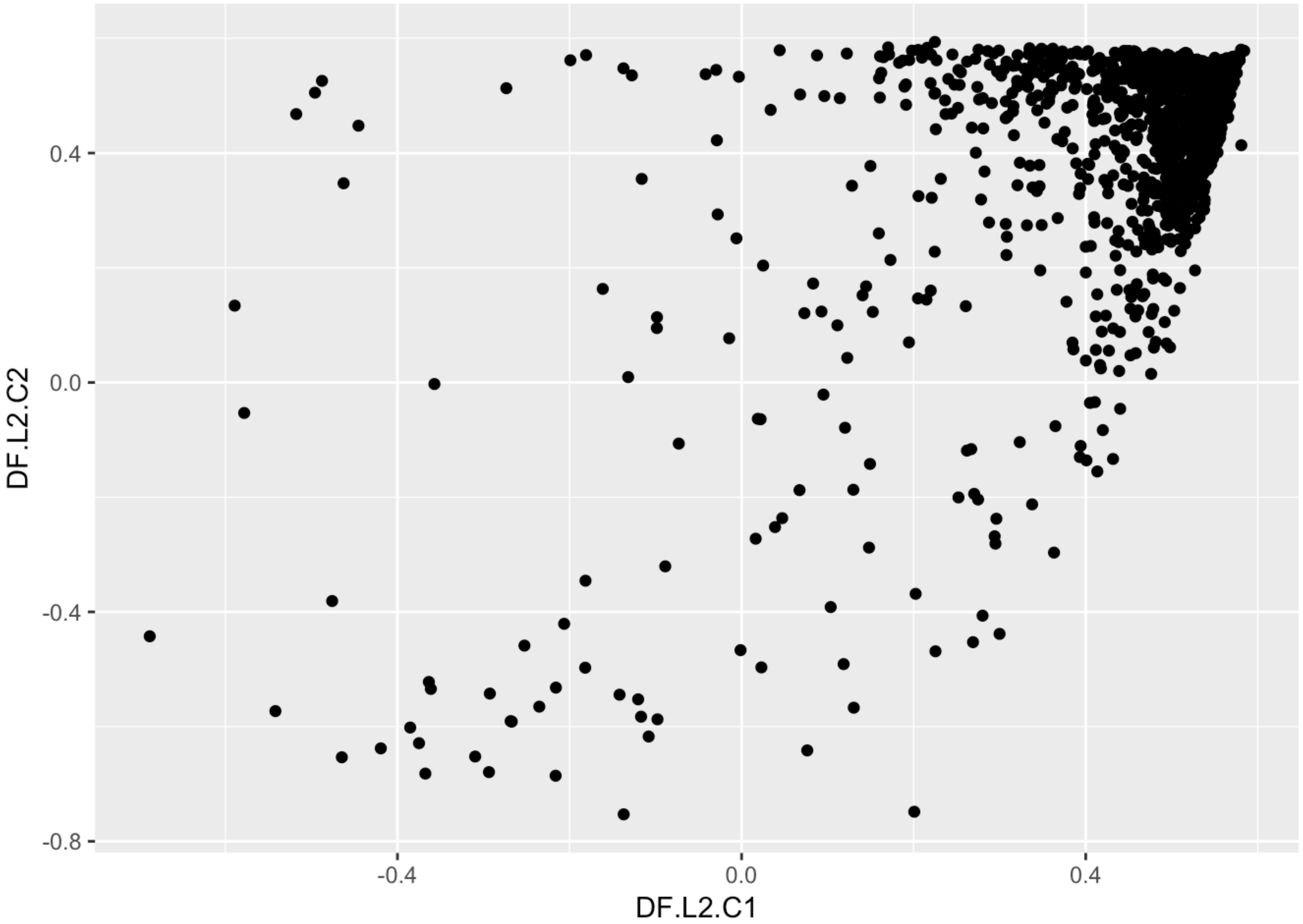
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 560 weights/biases, 16.5 KB, 218
,400 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2     5  Tanh  0.00 % 0.000000 0.000000  0.479026 0.337976 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.045984 0.025170 0.000000
## 4     4     5  Tanh  0.00 % 0.000000 0.000000  0.016172 0.001813 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.026387 0.063845 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.034244   0.195337  0.372873 0.481217
## 3    0.187328   0.411369 -0.076665 0.030236
## 4    0.273079   0.684253 -0.461672 0.760387
## 5   -0.023042   0.130015 -0.069759 0.093643
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002705178
## RMSE: (Extract with `h2o.rmse`) 0.05201132
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

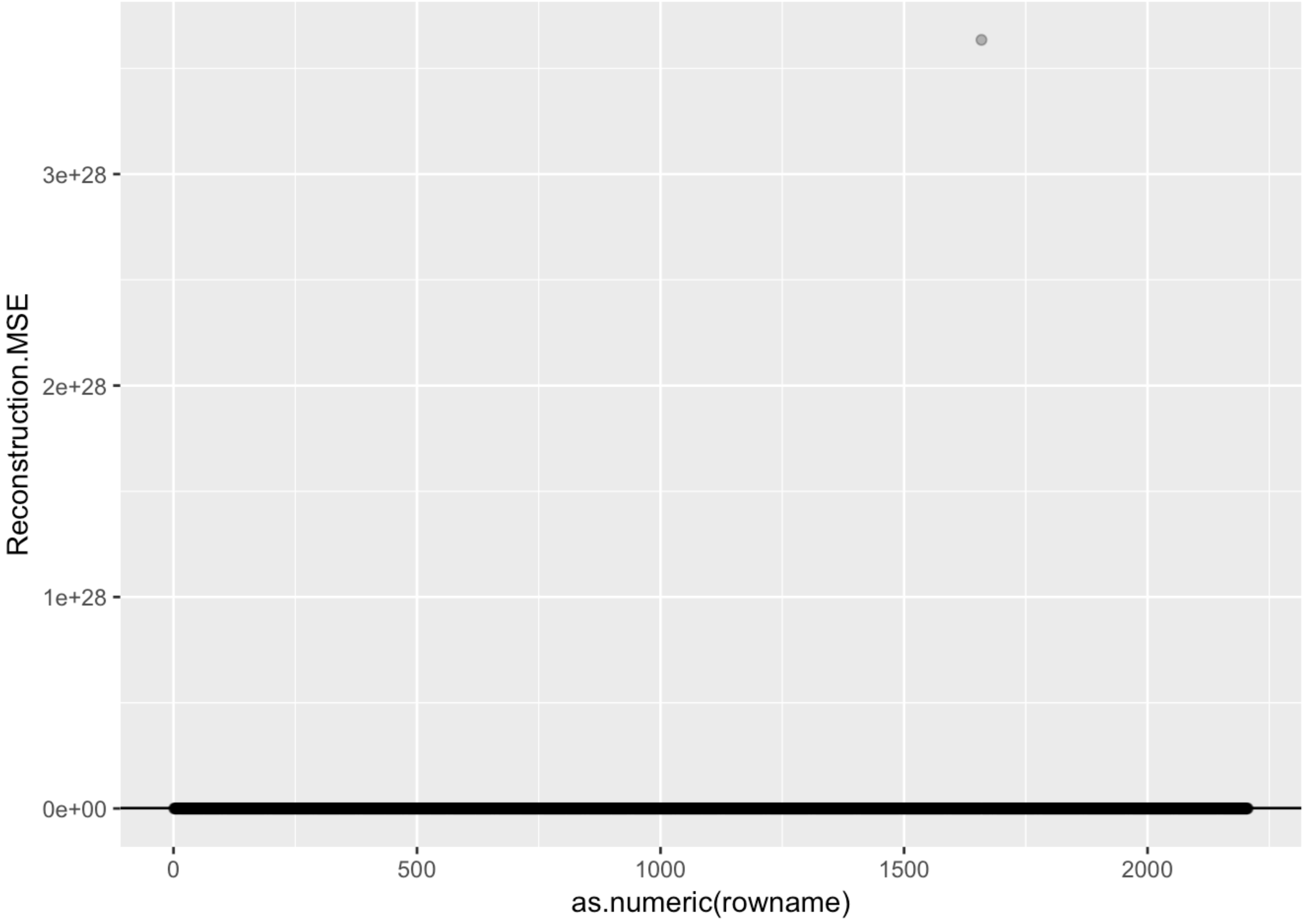
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers based on the graph
```



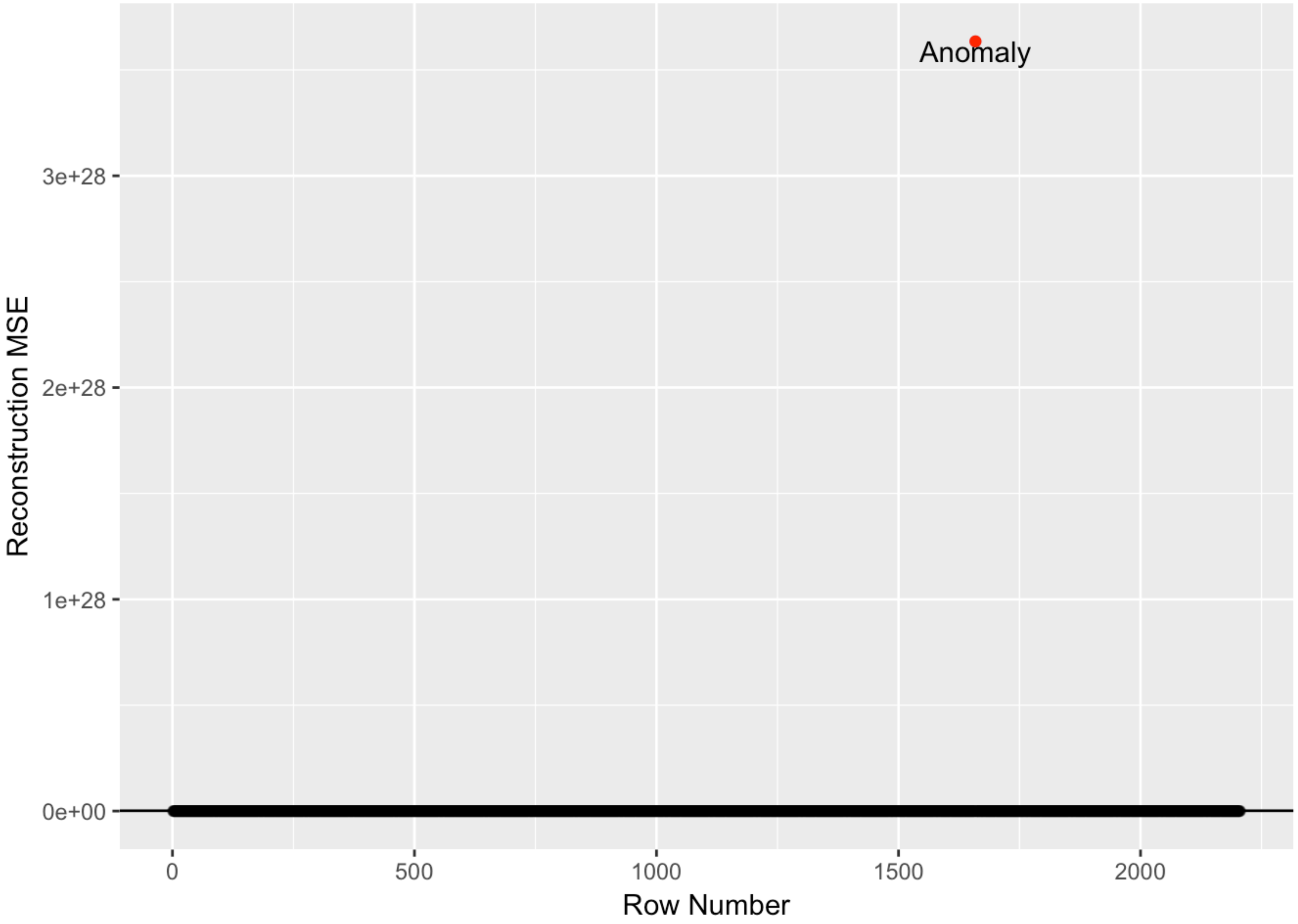
```
#use anomaly detection function

#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse
```

```
##           mean
## 1 1.647652e+25
```

```
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



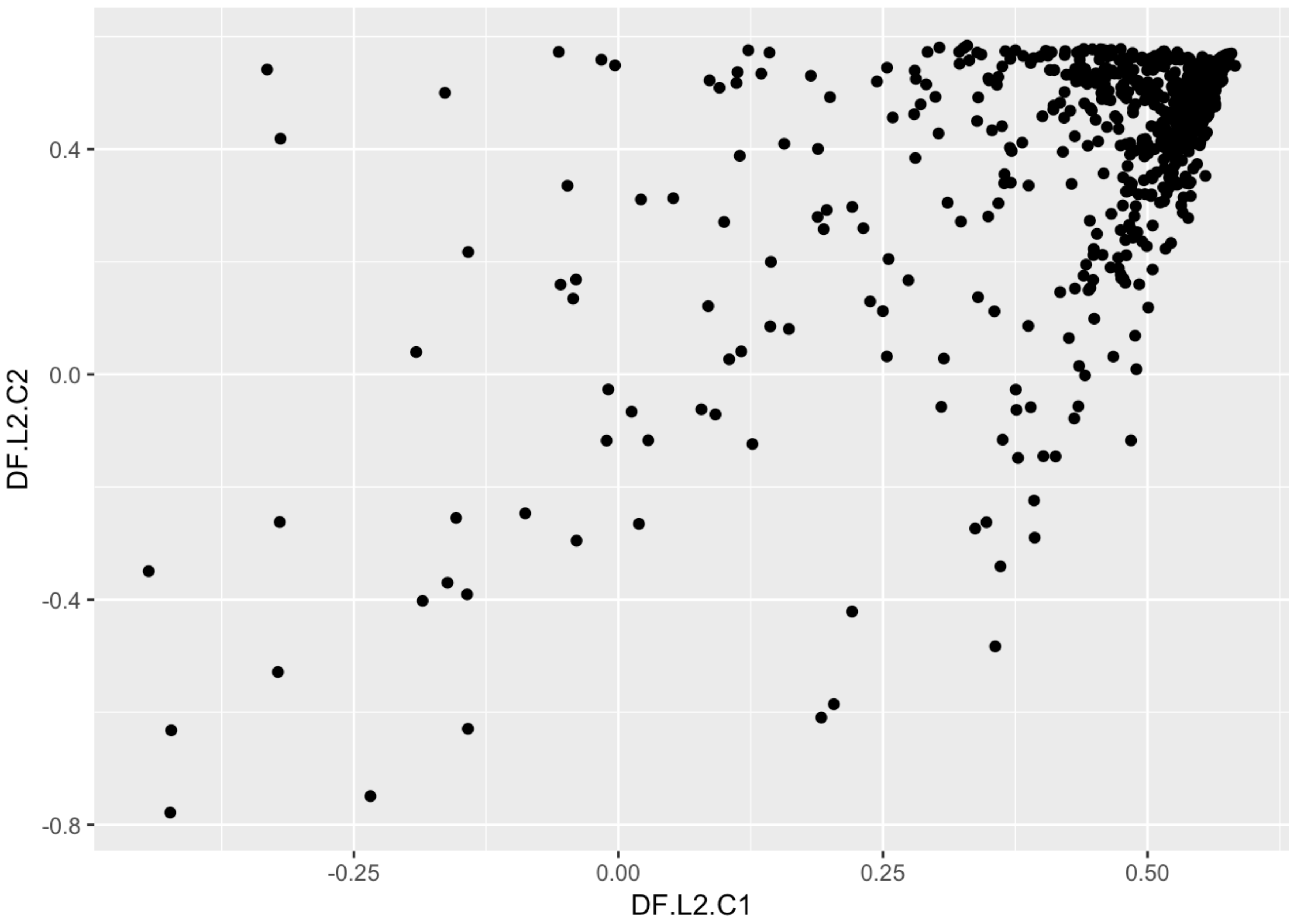
```
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

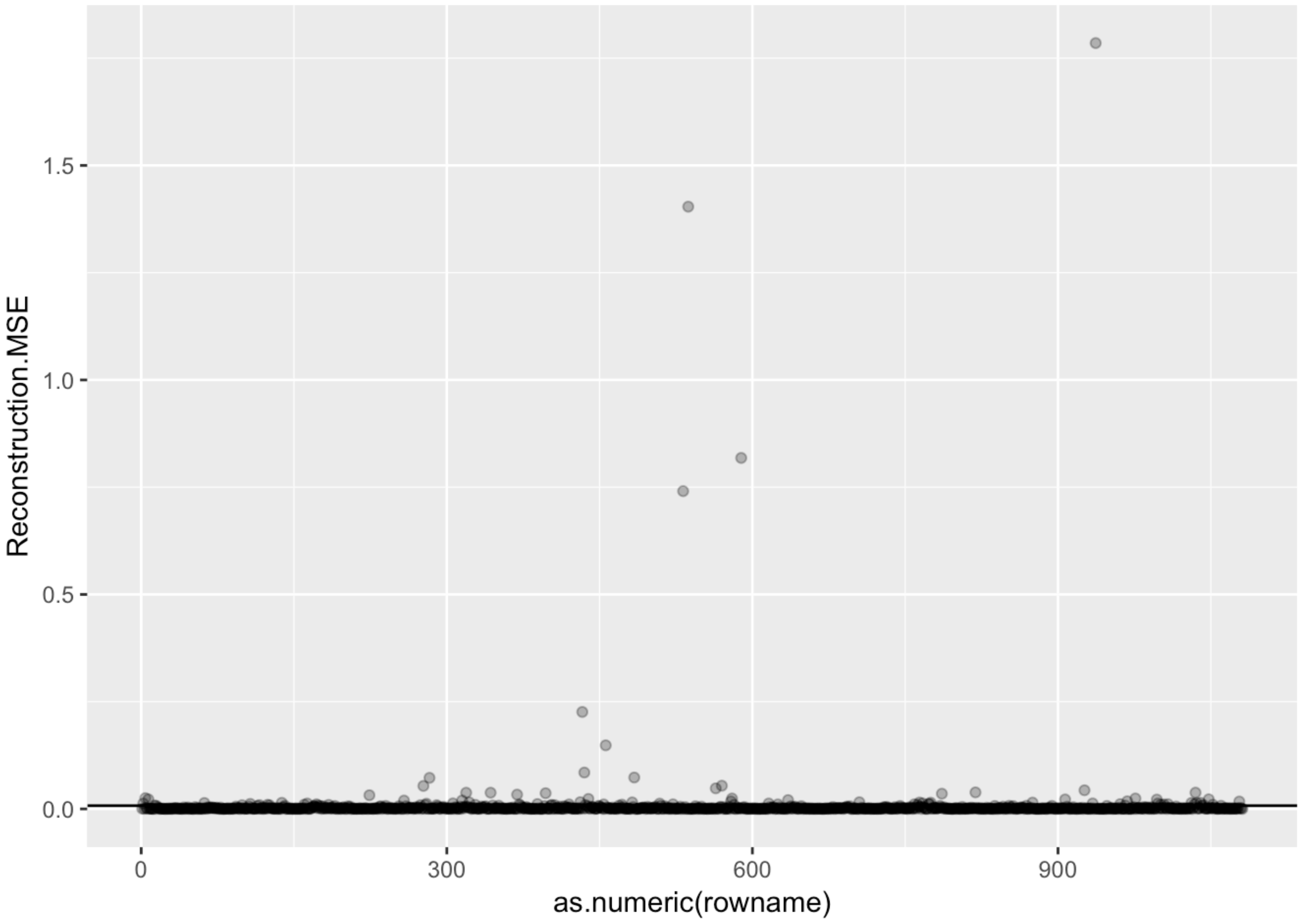
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point()
```



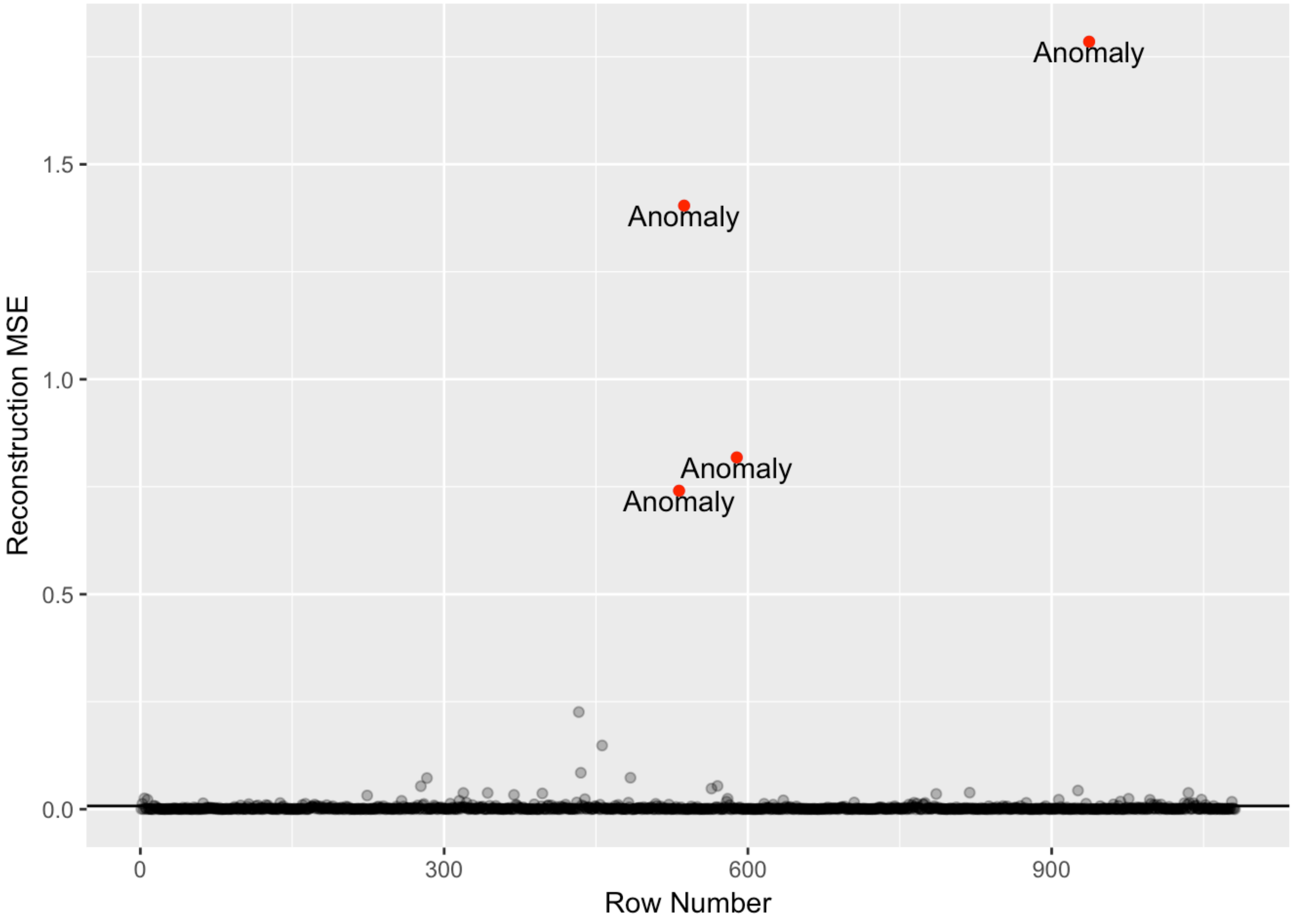
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007633034
```

```
##          mean
## 1 0.007630806
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto2 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto <- rbind(fraud_auto, fraud_auto2)

#Hyperparameters: hidden= 10,2,10 epochs = 100
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(10, 2, 10),
epochs = 100,
activation = "Tanh"
)
```

```
## 
  |
  |                                                                 |   0%
  |
  |======                                                           |  10%
  |
  |================================                                 |  50%
  |
  |====================================================             |  80%
  |
  |=================================================================| 100%
```

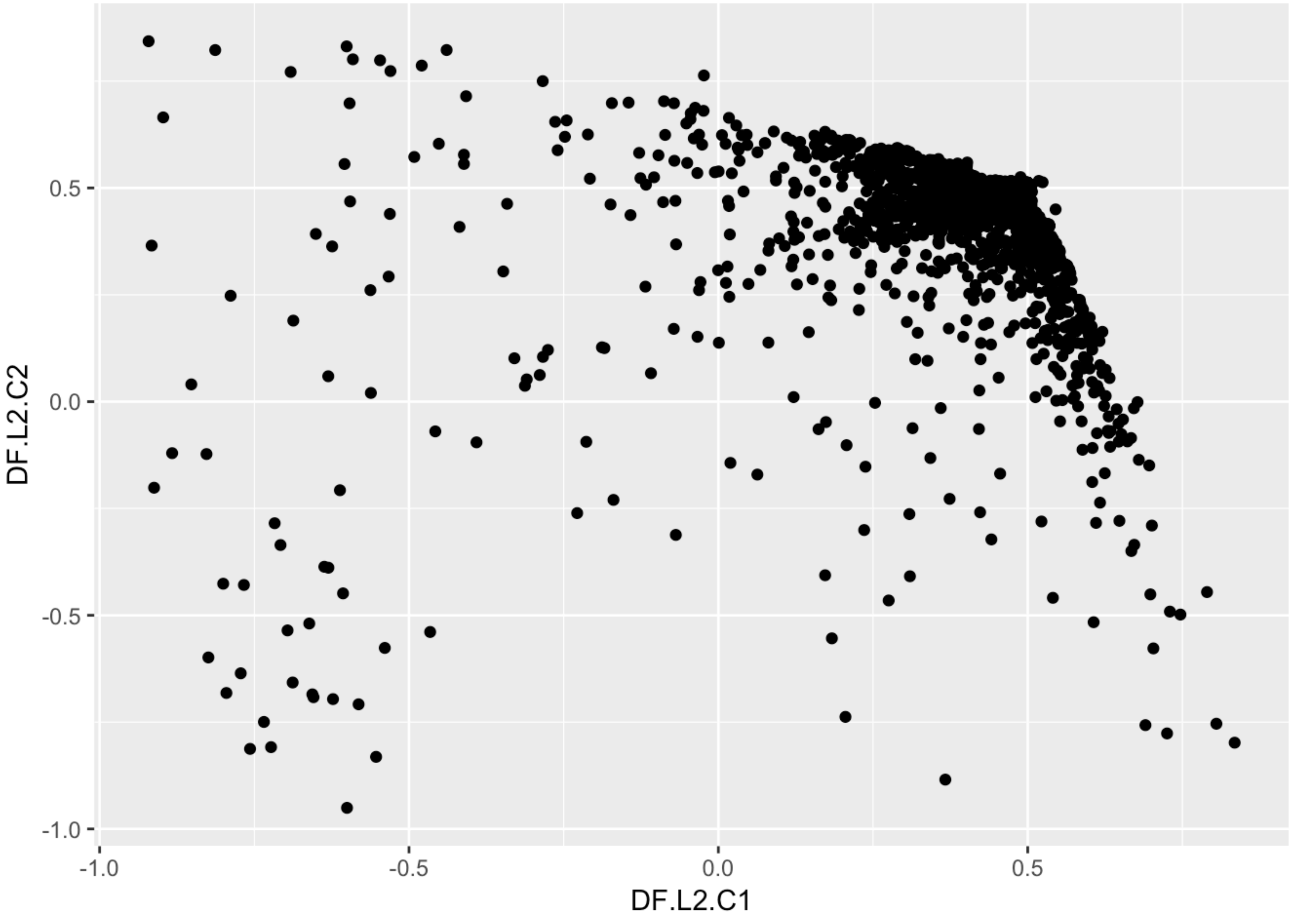
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 1,070 weights/biases, 22.6 KB, 2
18,400 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2    10  Tanh  0.00 % 0.000000 0.000000  0.542340 0.316112 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.038262 0.011778 0.000000
## 4     4    10  Tanh  0.00 % 0.000000 0.000000  0.014941 0.003893 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.017366 0.026094 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.058417   0.209880  0.320923 0.829139
## 3    0.202537   0.370560 -0.376802 0.041043
## 4   -0.028103   0.534001  0.077167 0.718914
## 5   -0.000205   0.133304 -0.066080 0.103163
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002686602
## RMSE: (Extract with `h2o.rmse`) 0.05183244
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |================================                                 |  50%
  |
  |=================================================================| 100%
```

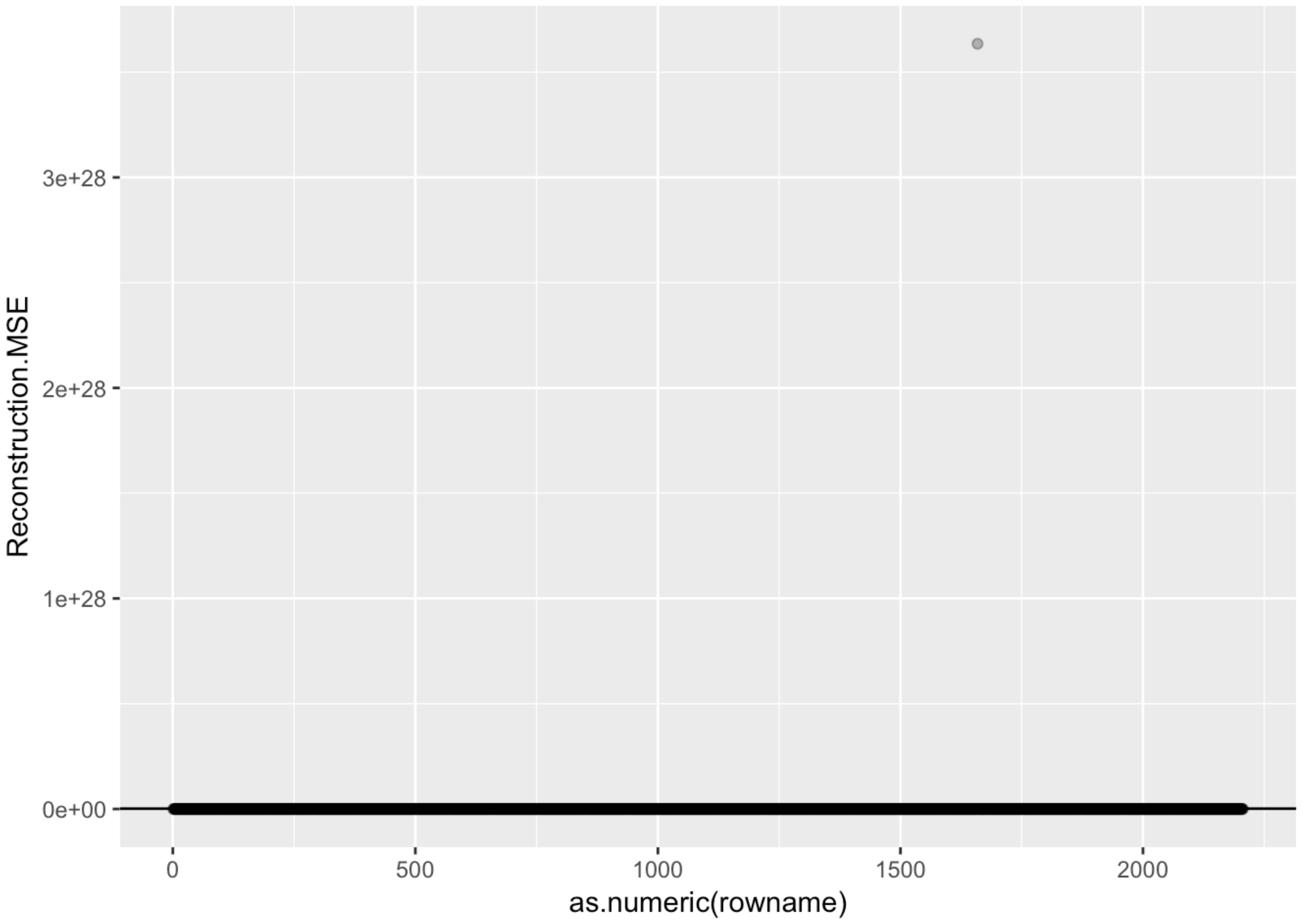
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see which points are outliers, use anomaly detection
```



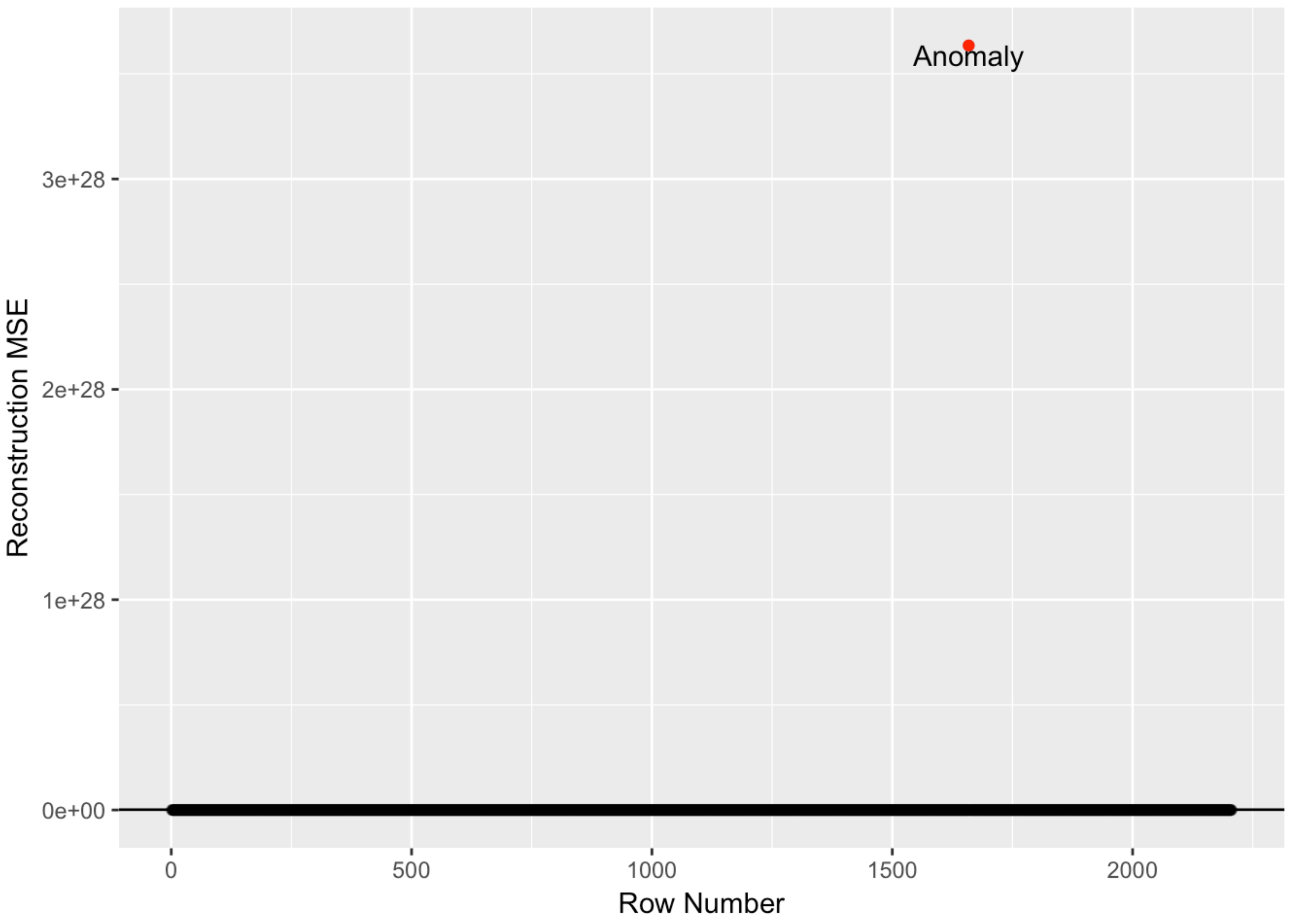
```
#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #1.647652e+25
```

```
##           mean
## 1 1.647652e+25
```

```
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



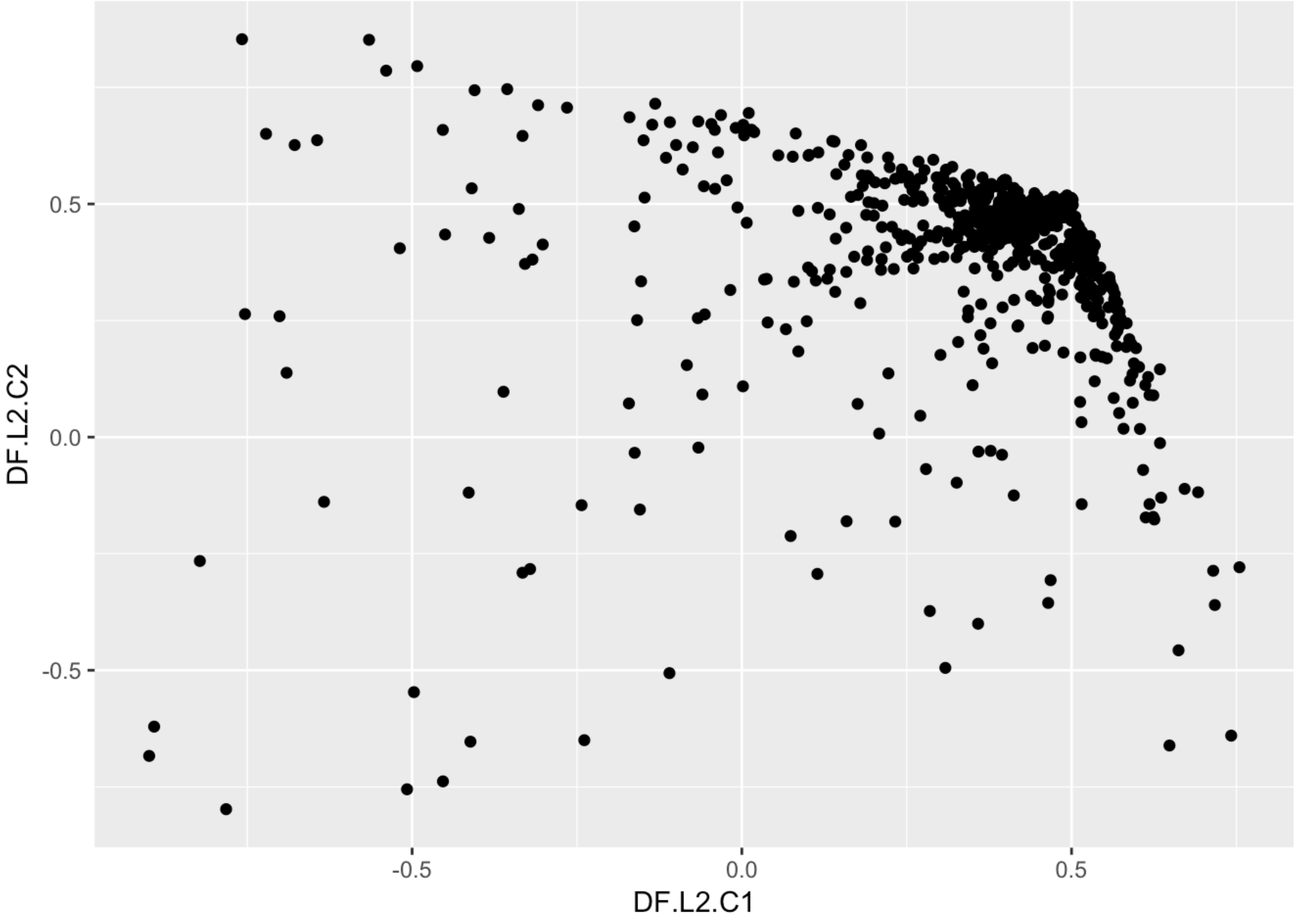
```
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #again row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto3 <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

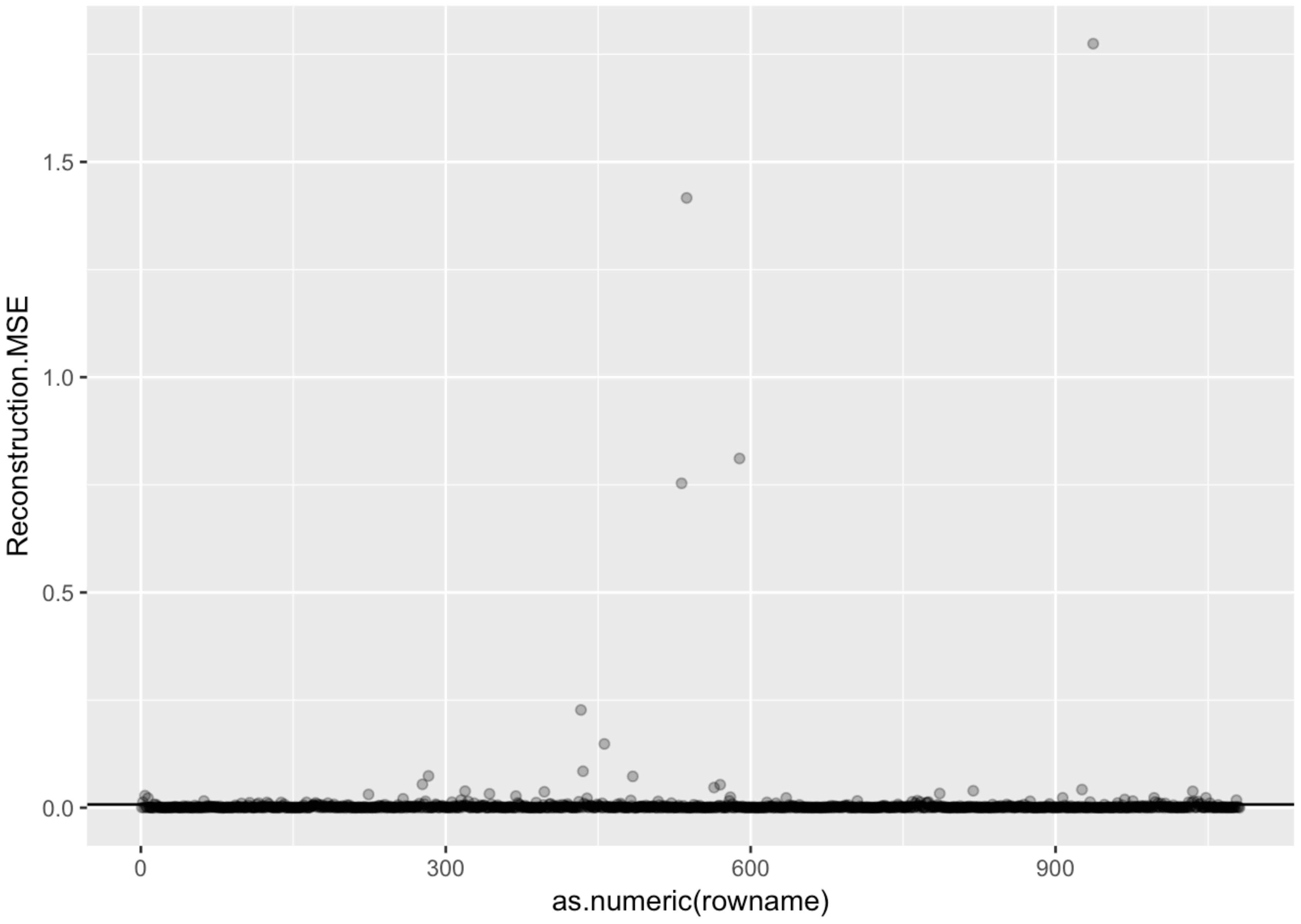
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point()  #difficult to see outliers, use anomaly detection function
```



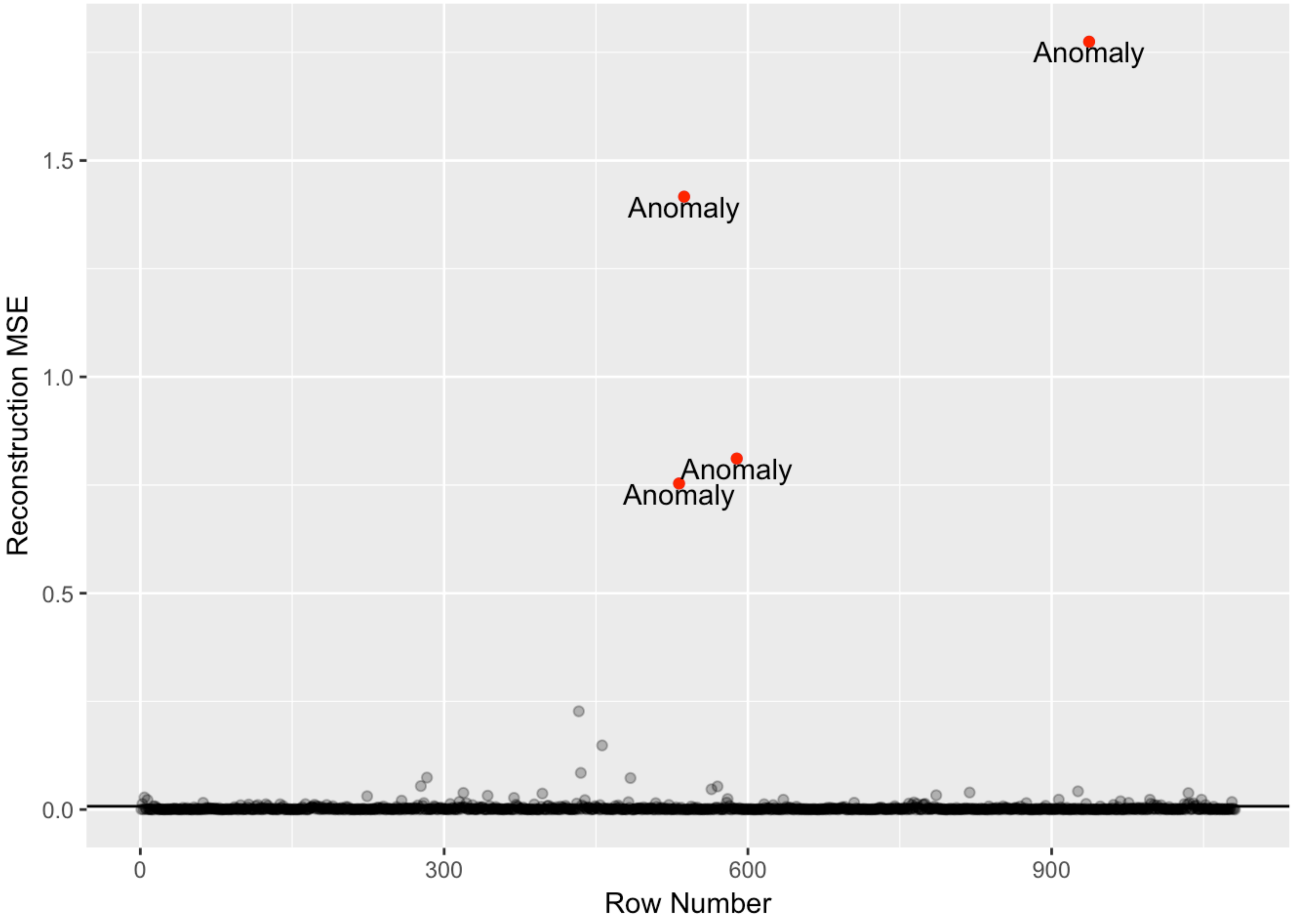
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007471474
```

```
##          mean
## 1 0.007605382
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #again, 4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto4 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto3 <- rbind(fraud_auto3, fraud_auto4)

#Hyperparameters: hidden= 5,2,5 epochs = 50
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(5, 2, 5),
epochs = 50,
activation = "Tanh"
)
```

```
##
  |
  |                                                                 |   0%
  |
  |======                                                           |  10%
  |
  |=================================================================| 100%
```

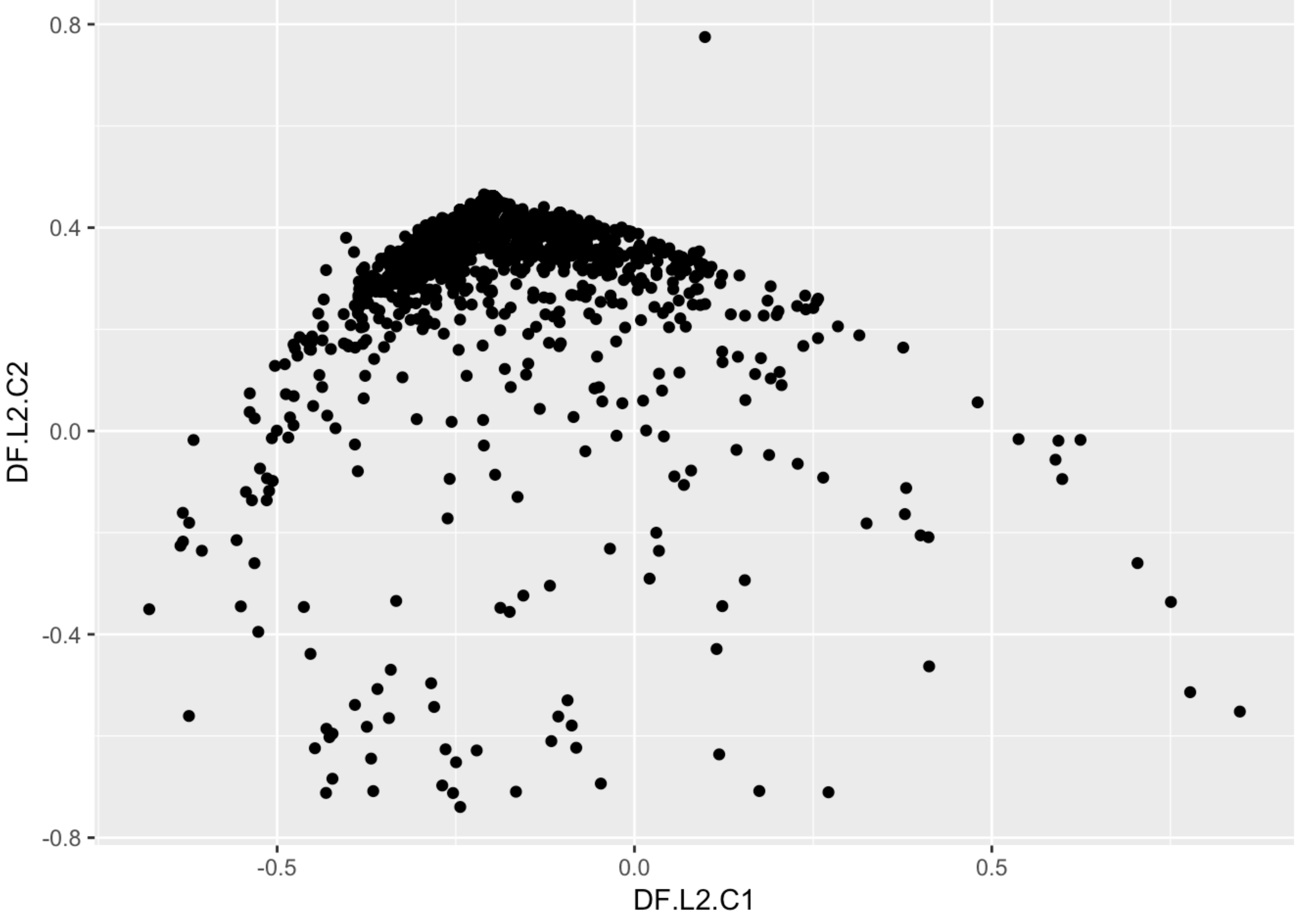
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 560 weights/biases, 16.5 KB, 109
,200 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2     5  Tanh  0.00 % 0.000000 0.000000  0.373738 0.298850 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.030857 0.011263 0.000000
## 4     4     5  Tanh  0.00 % 0.000000 0.000000  0.017275 0.005266 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.029557 0.048476 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.032488   0.186752  0.148808 0.452786
## 3    0.070456   0.443334 -0.291856 0.001799
## 4   -0.153768   0.698241 -0.002421 0.415632
## 5    0.007005   0.140633 -0.024118 0.061744
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002690766
## RMSE: (Extract with `h2o.rmse`) 0.05187259
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

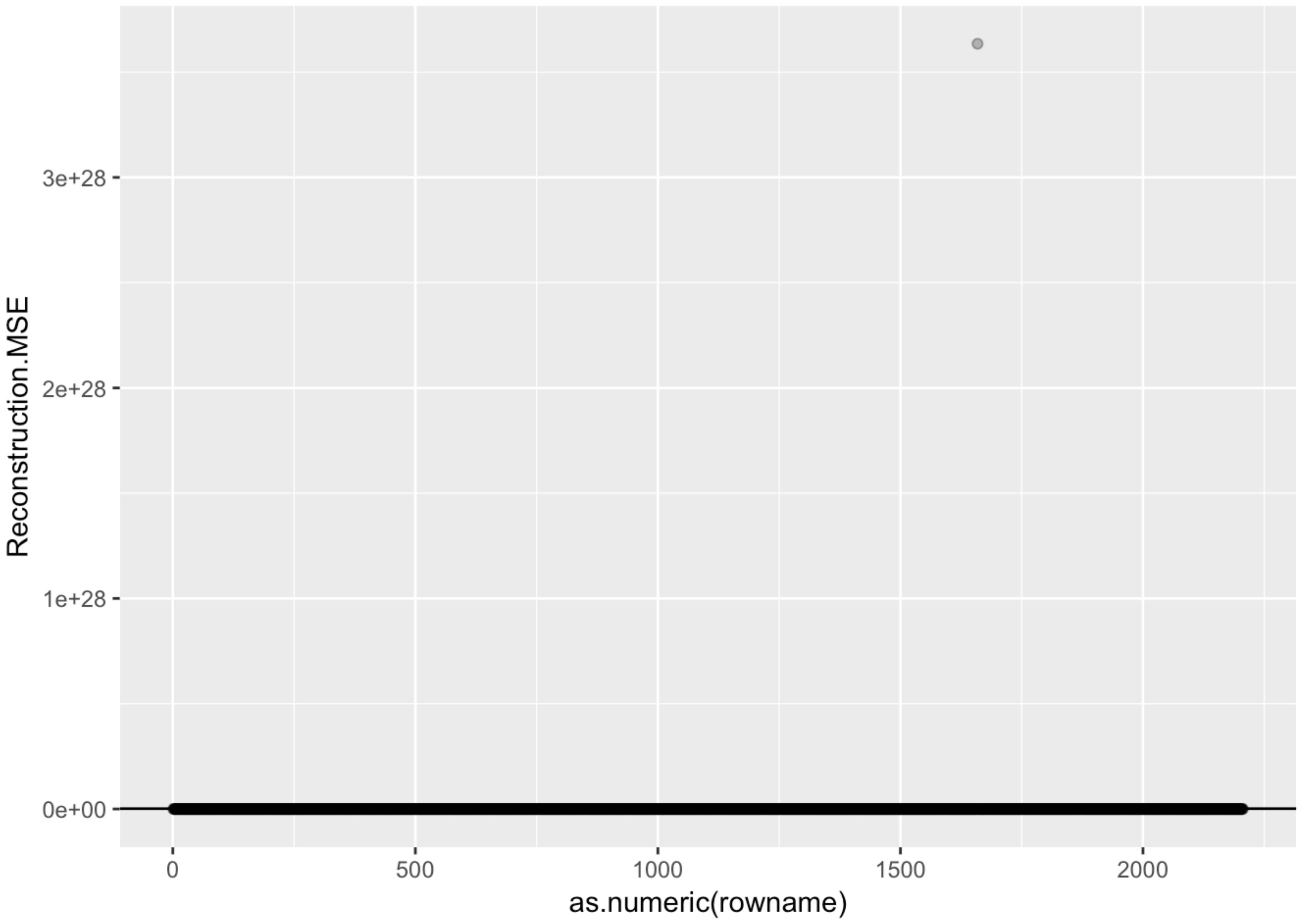
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly detection function
```



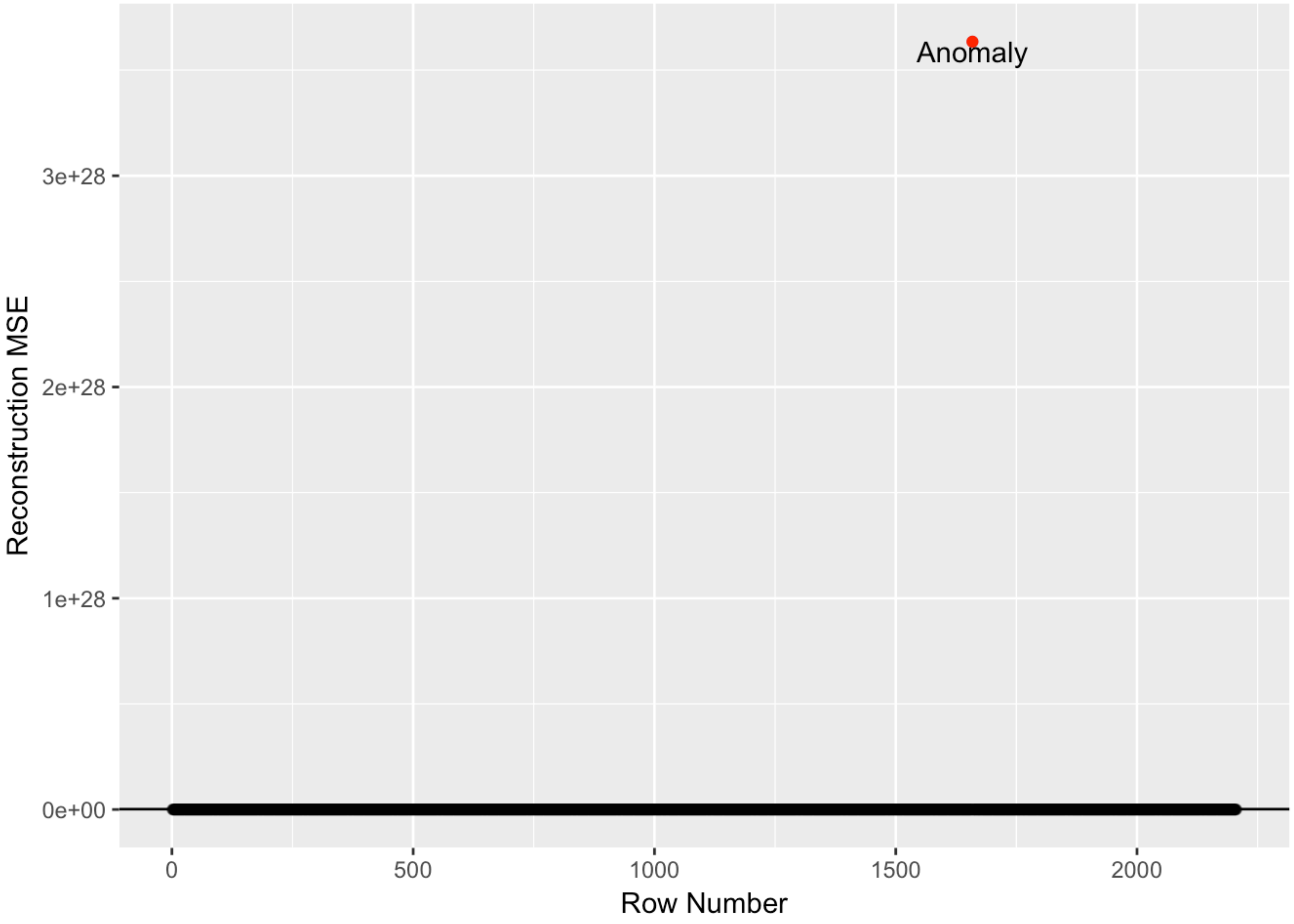
```r
#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #1.647652e+25
```

```
##           mean
## 1 1.647652e+25
```

```r
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



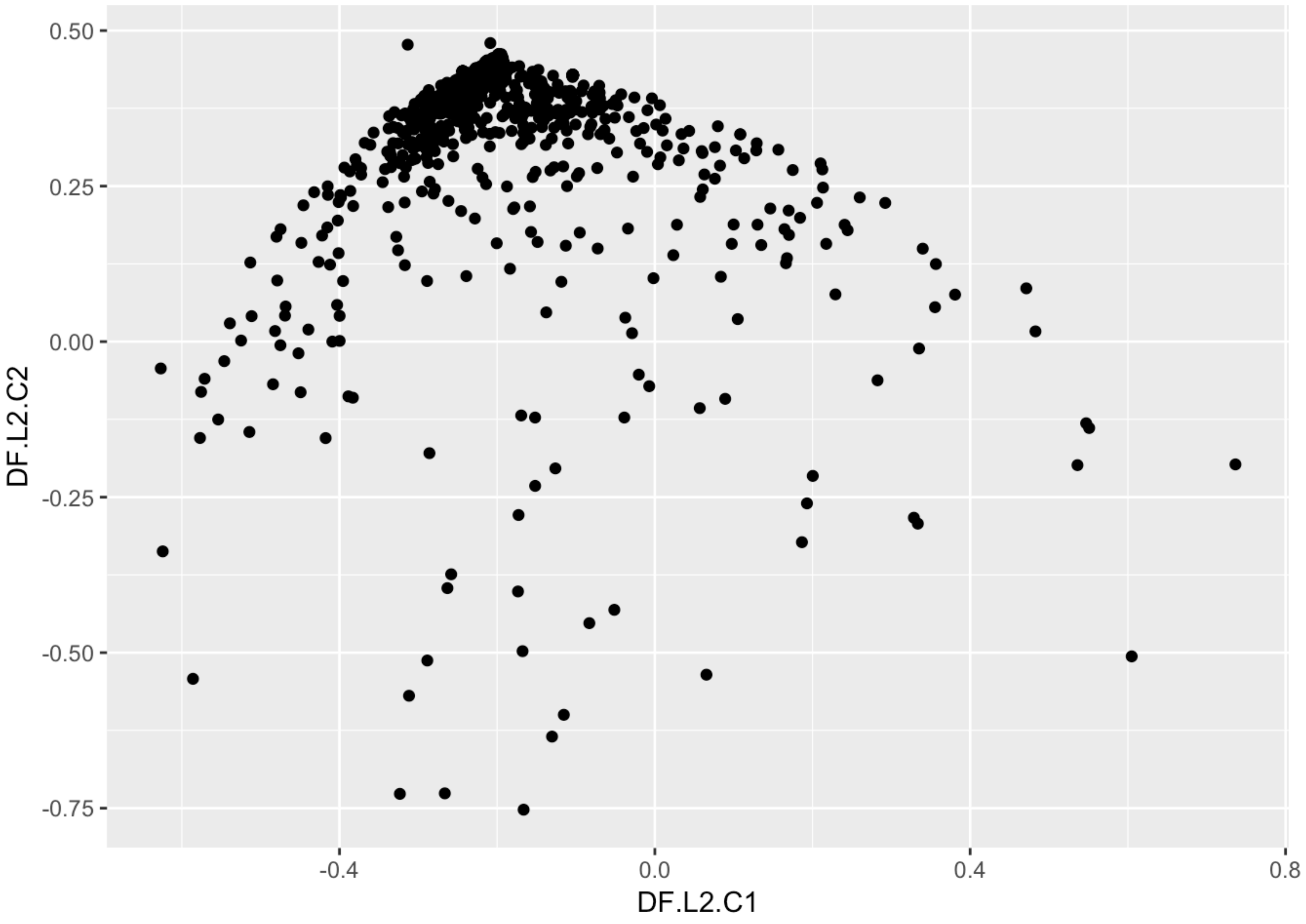
```r
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto5 <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

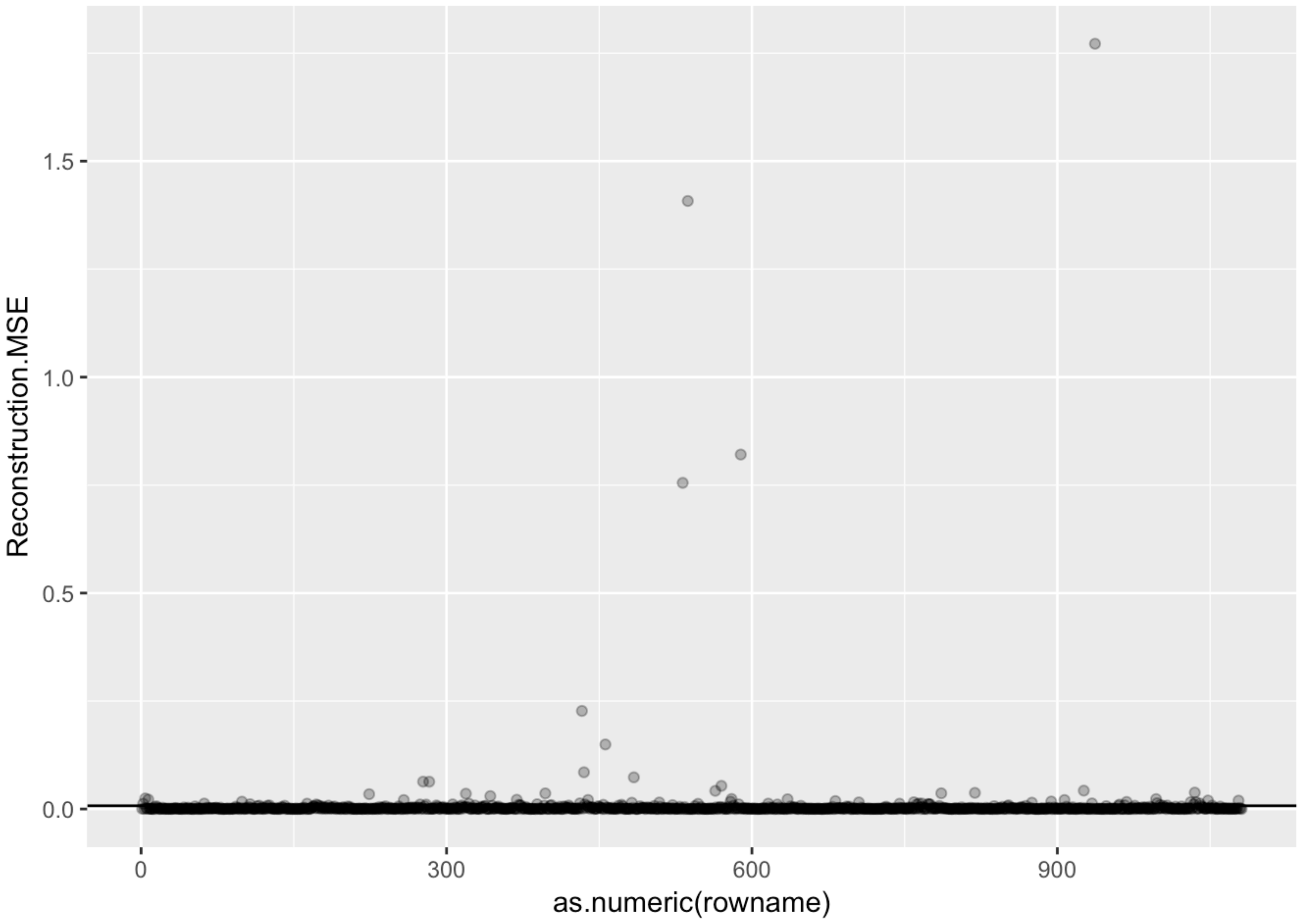
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly function
```



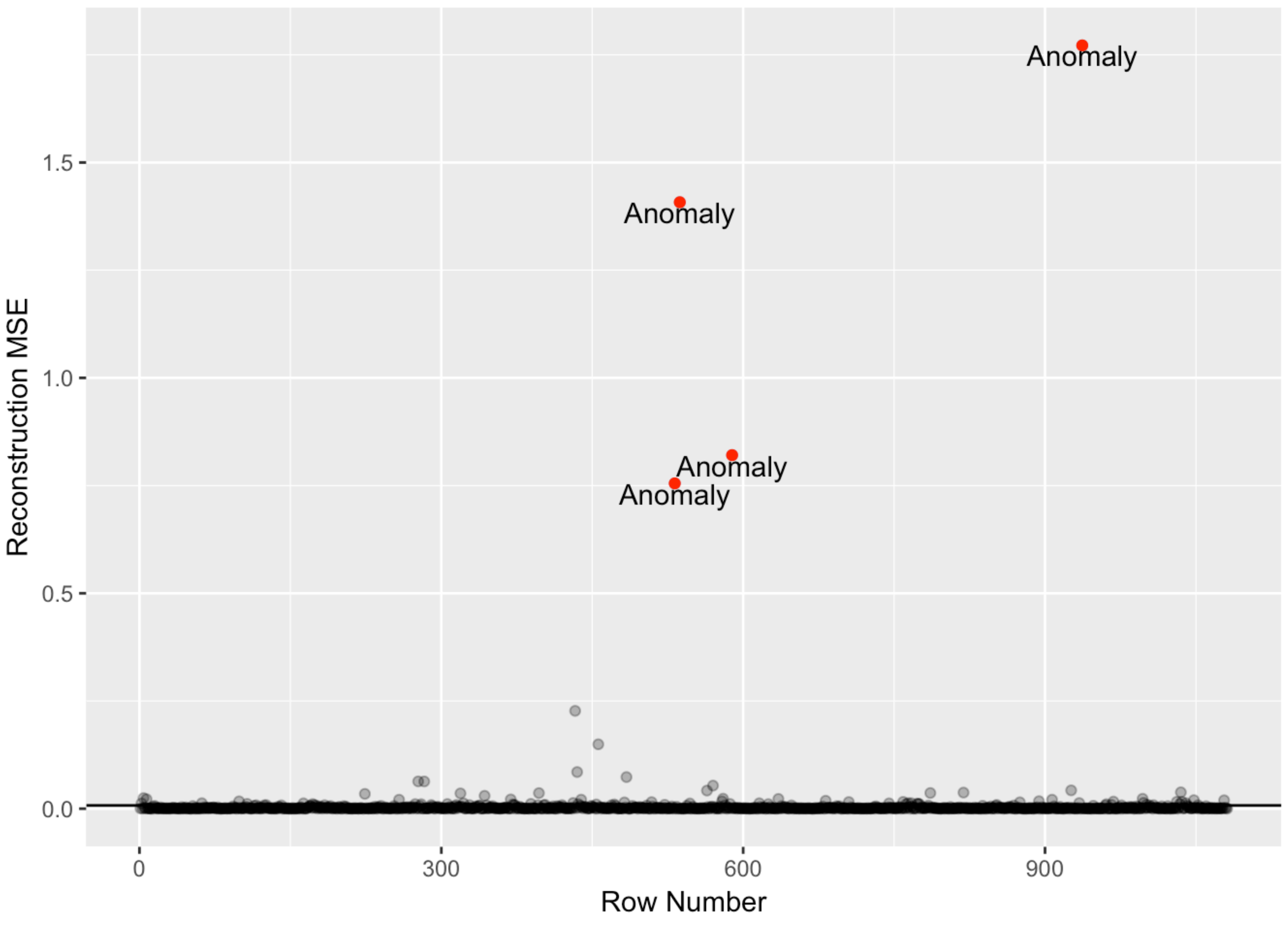
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007654928
```

```
##          mean
## 1 0.007590092
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto6 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto5 <- rbind(fraud_auto5, fraud_auto5)

#Hyperparameters: hidden= 10,2,10 epochs = 50
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(10, 2, 10),
epochs = 50,
activation = "Tanh"
)
```

```
##
  |
  |                                                                 |   0%
  |
  |======                                                           |  10%
  |
  |====================================================             |  80%
  |
  |=================================================================| 100%
```

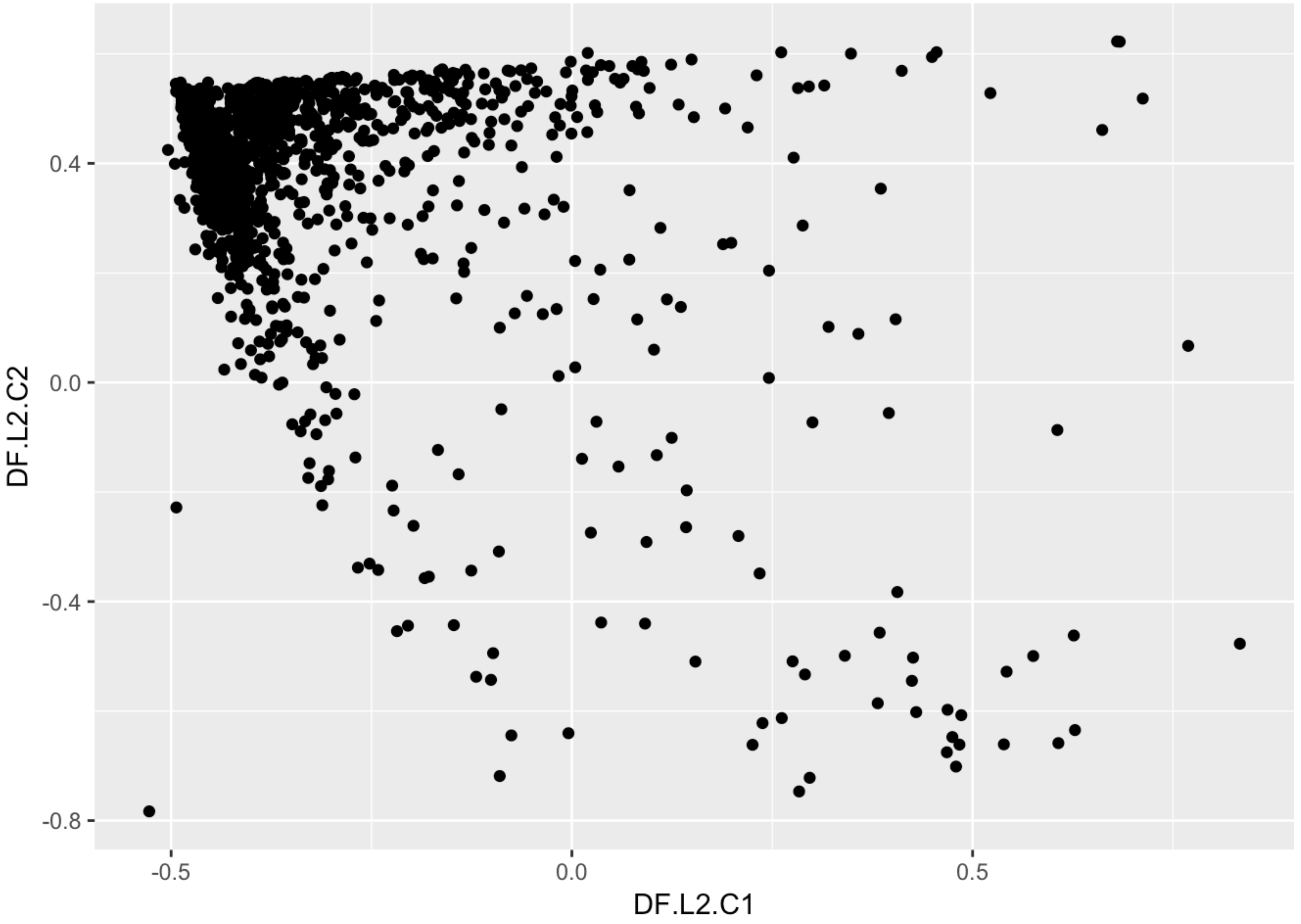
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 1,070 weights/biases, 22.6 KB, 1
09,200 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2    10  Tanh  0.00 % 0.000000 0.000000  0.568292 0.287090 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.079092 0.026032 0.000000
## 4     4    10  Tanh  0.00 % 0.000000 0.000000  0.012482 0.002350 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.048798 0.075271 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.004023   0.196240 -0.040712 0.363420
## 3   -0.008344   0.335481 -0.022985 0.032709
## 4   -0.003274   0.492689  0.094036 0.378860
## 5    0.002066   0.141298 -0.023892 0.055148
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002703982
## RMSE: (Extract with `h2o.rmse`) 0.05199982
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                      |   0%
  |
  |===================================                                   |  50%
  |
  |======================================================================| 100%
```

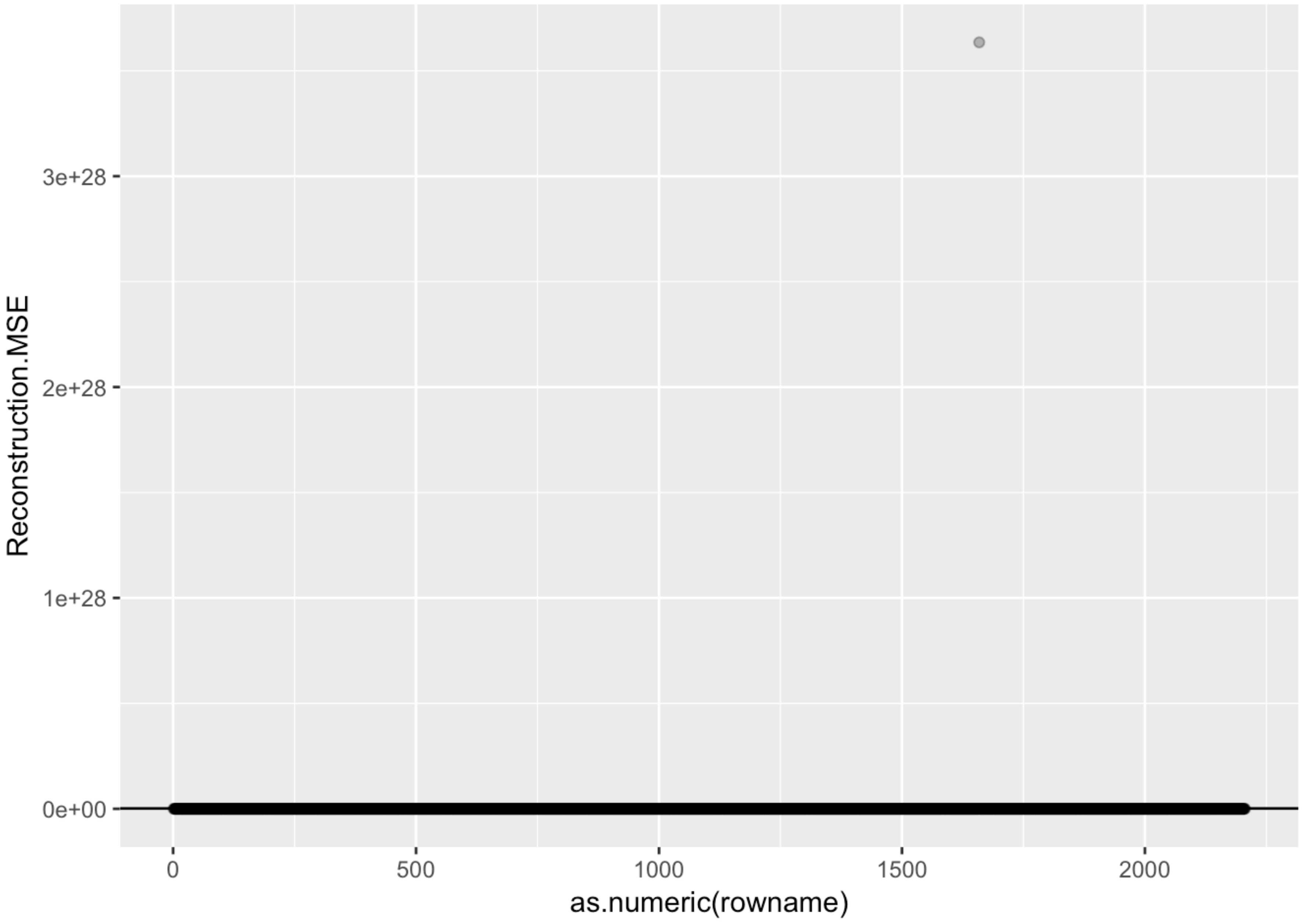
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly function
```



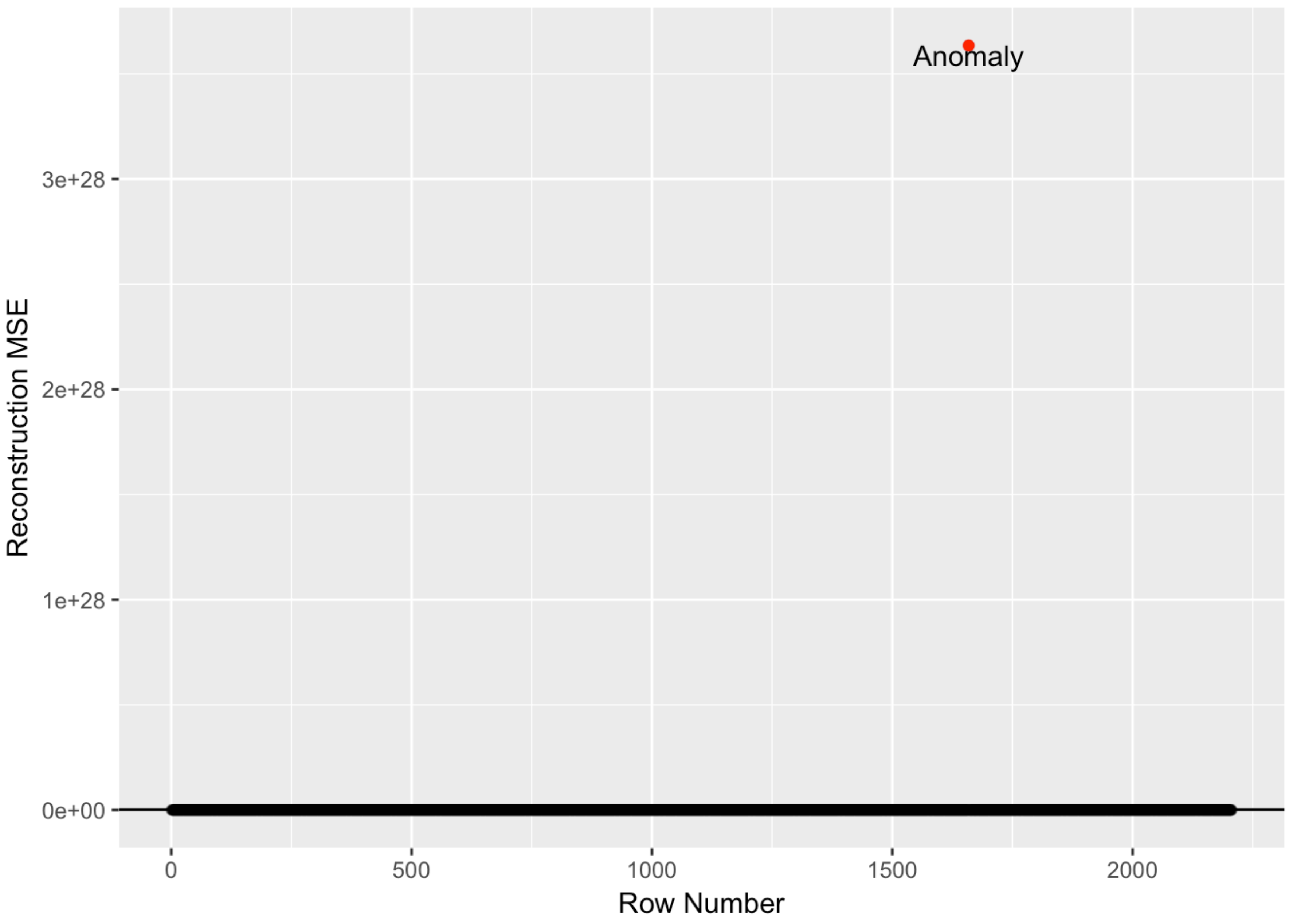
```
#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #1.647652e+25
```

```
##           mean
## 1 1.647652e+25
```

```
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



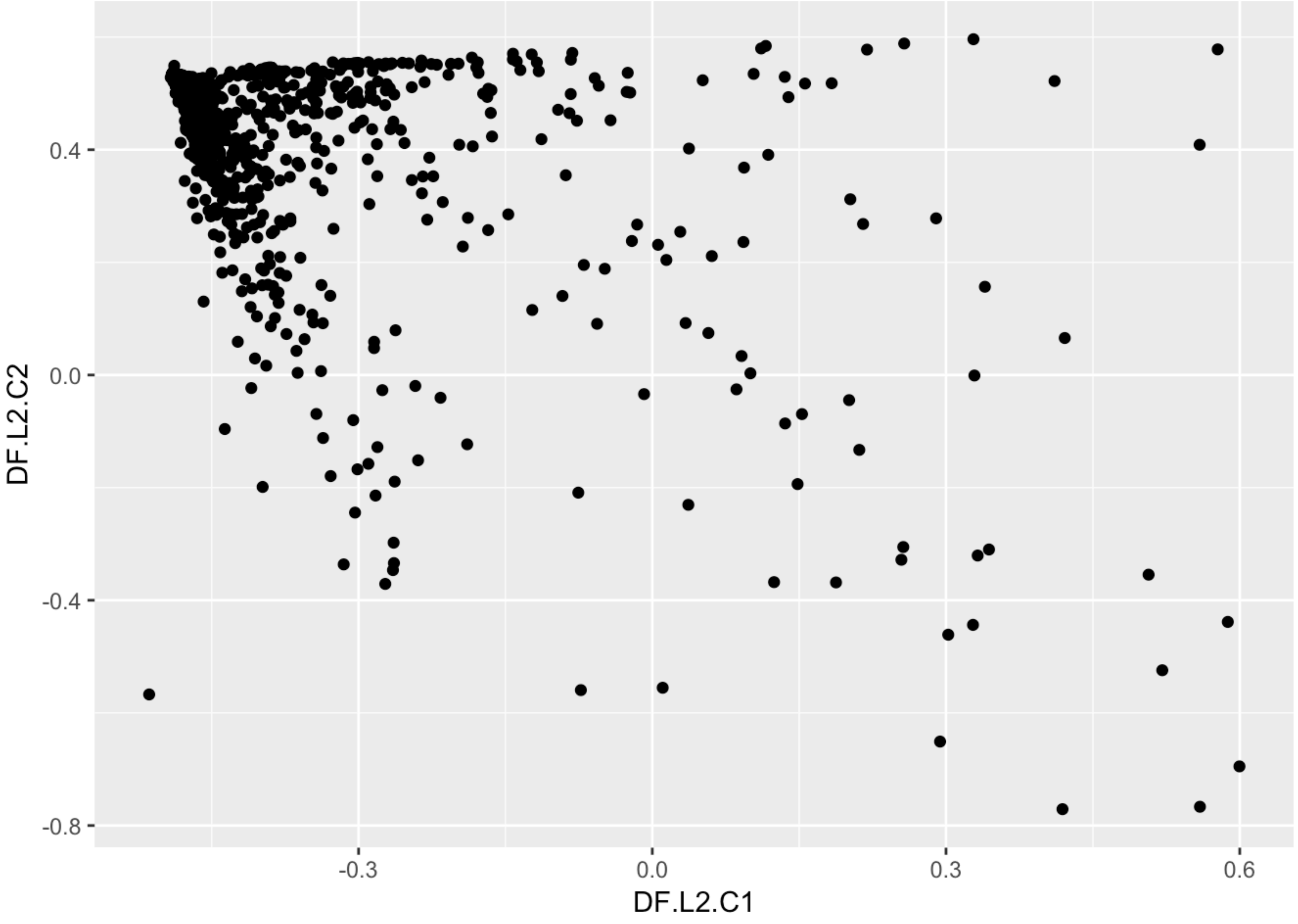
```
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto7 <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

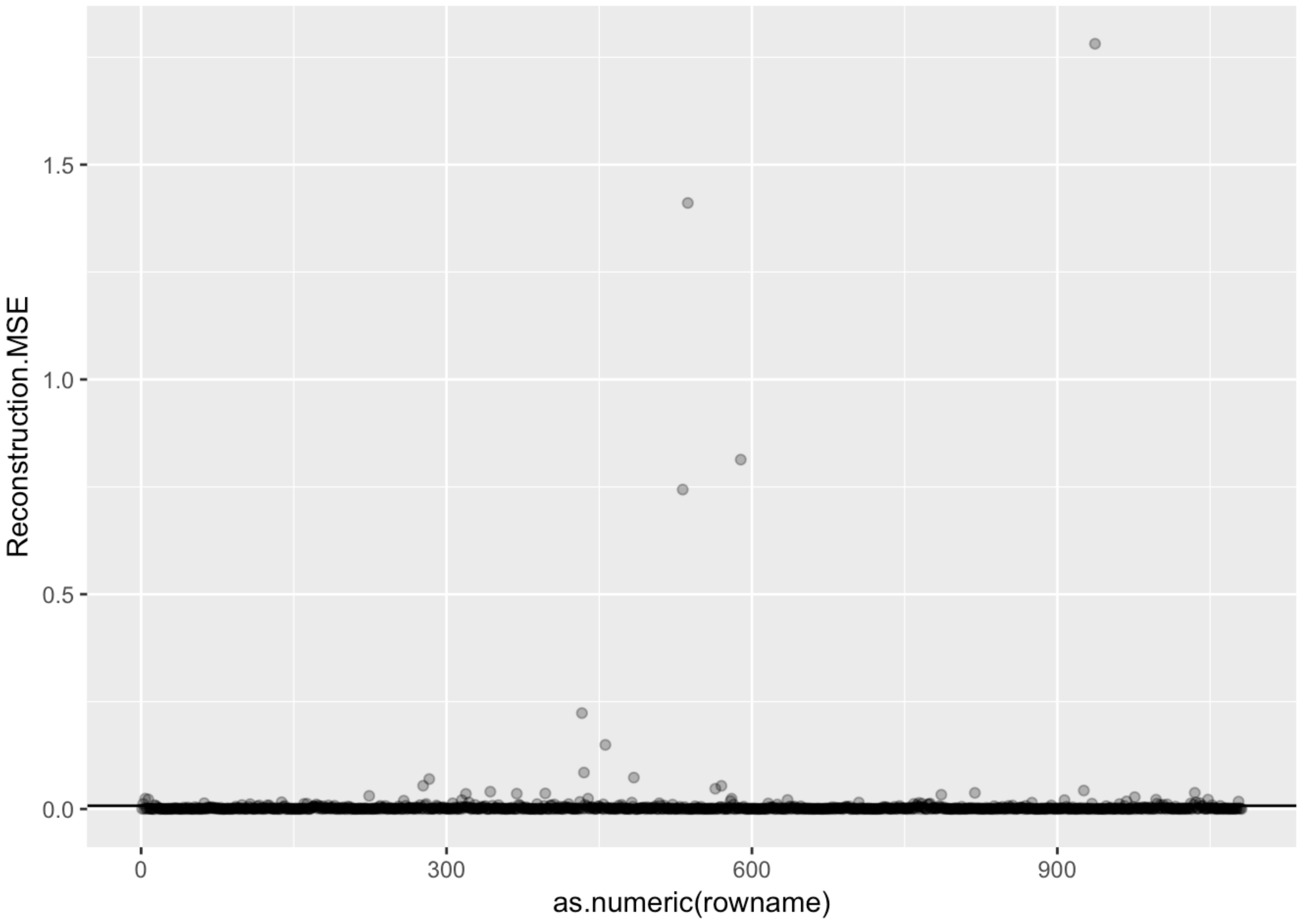
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly function
```



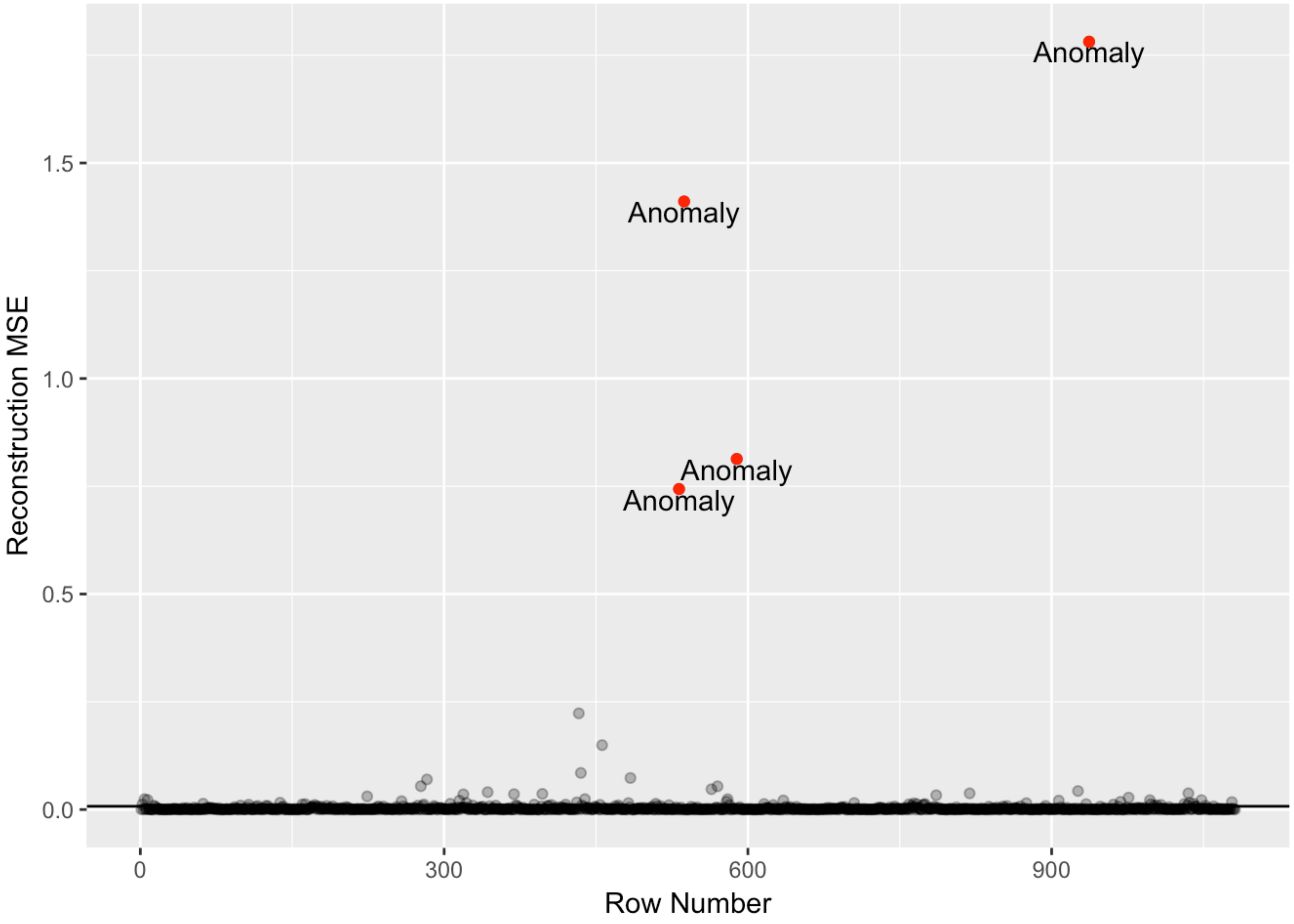
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007580477
```

```
##          mean
## 1 0.007647592
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto8 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto7 <- rbind(fraud_auto7, fraud_auto8)

#Hyperparameters: hidden= 5,2,5 epochs = 200
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(5, 2, 5),
epochs = 200,
activation = "Tanh"
)
```

```
##
  |
  |                                                                 |   0%
  |
  |===                                                              |   5%
  |
  |=======================                                          |  35%
  |
  |==========================================                       |  65%
  |
  |==============================================================   |  95%
  |
  |=================================================================| 100%
```

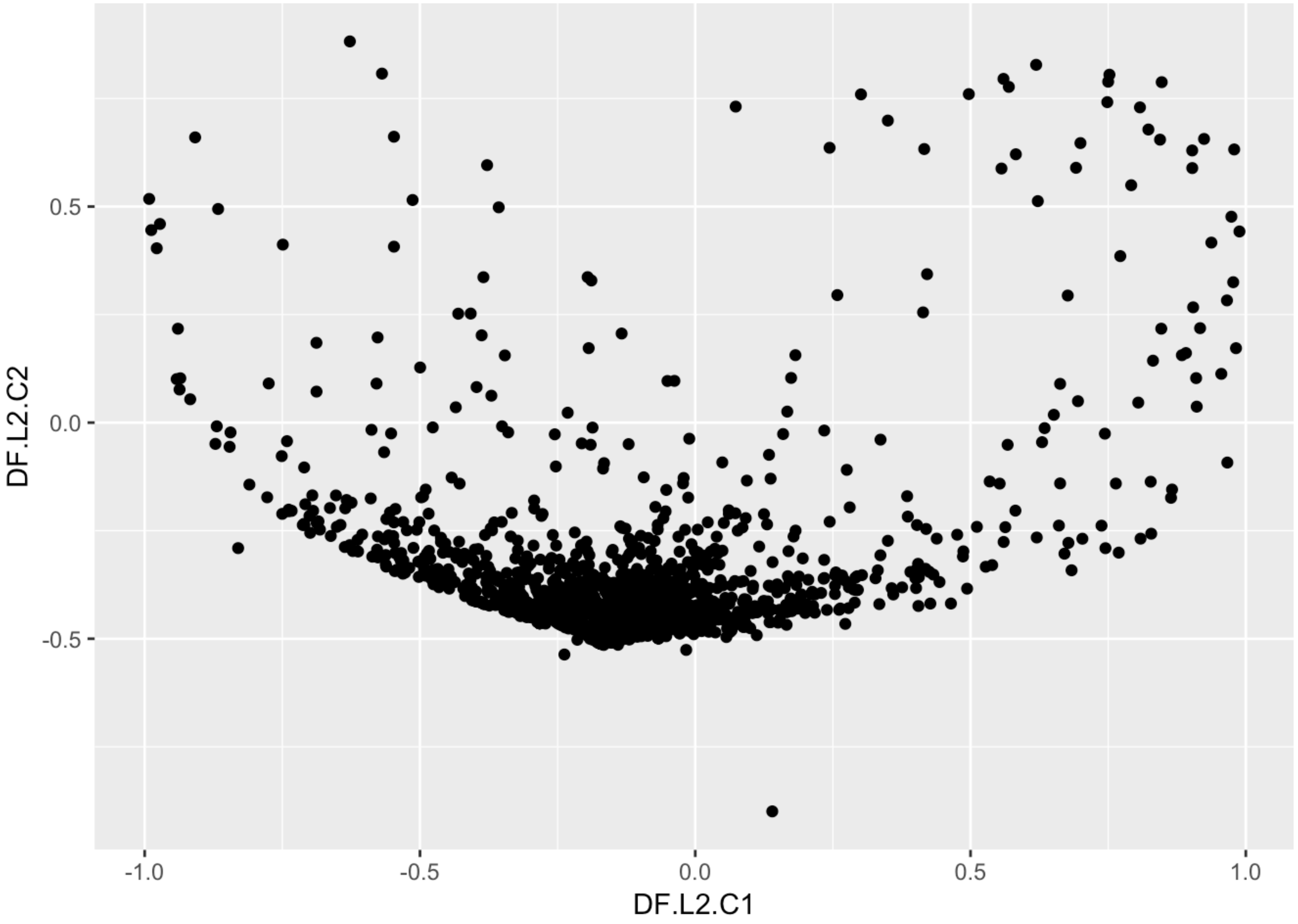
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 560 weights/biases, 16.5 KB, 436
,800 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2     5  Tanh  0.00 % 0.000000 0.000000  0.371088 0.323116 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.013339 0.005476 0.000000
## 4     4     5  Tanh  0.00 % 0.000000 0.000000  0.017925 0.005319 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.010522 0.019056 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.036832   0.273483  0.607661 0.860750
## 3   -0.278161   0.690642  0.064342 0.279239
## 4   -0.321768   0.880430 -0.403682 0.855633
## 5   -0.003448   0.134351 -0.070183 0.118387
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002615746
## RMSE: (Extract with `h2o.rmse`) 0.05114437
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |================================                                 |  50%
  |
  |=================================================================| 100%
```

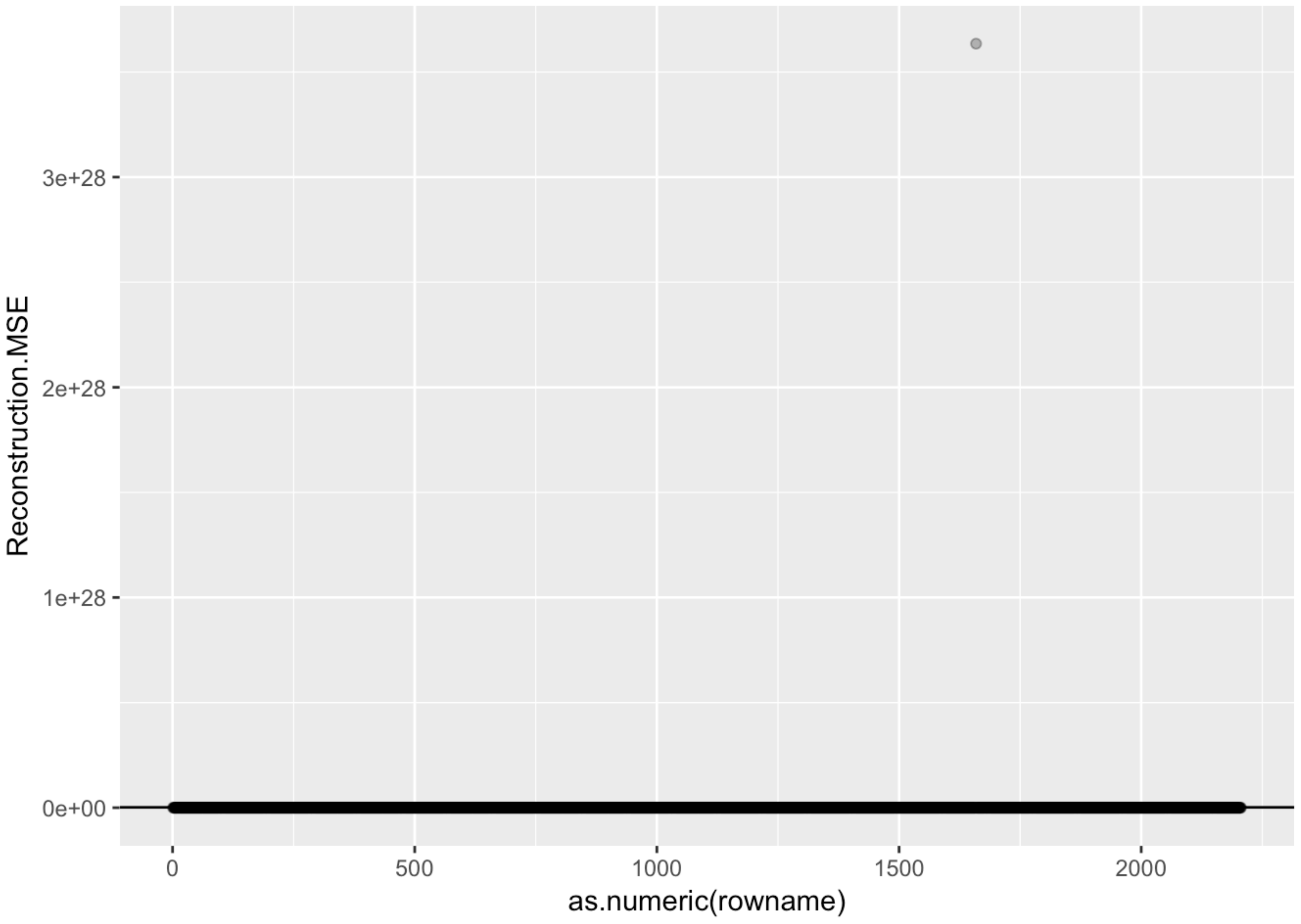
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #hard to see outliers, use anomaly detection
```



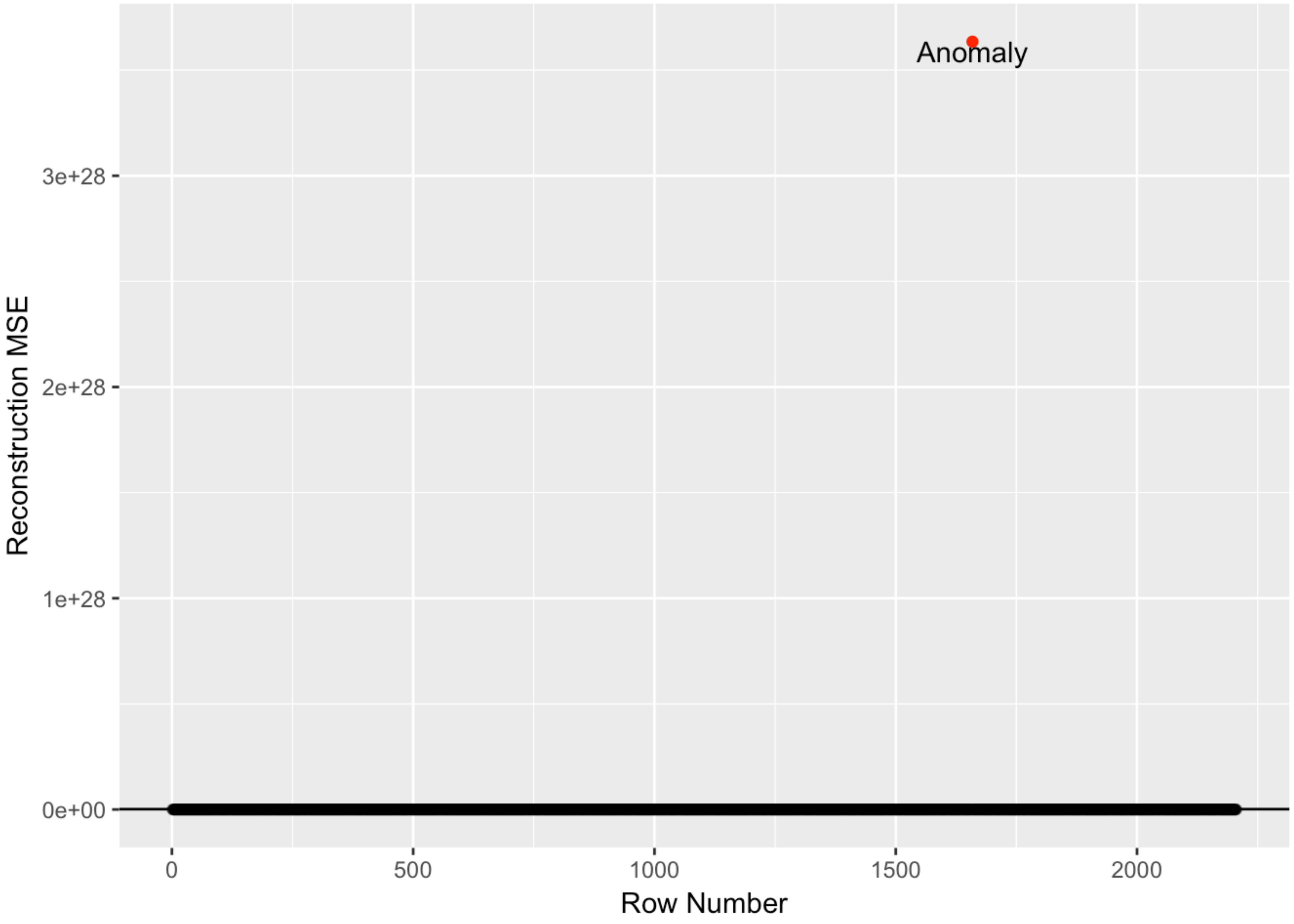
```
#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #1.647652e+25
```

```
##           mean
## 1 1.647652e+25
```

```
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



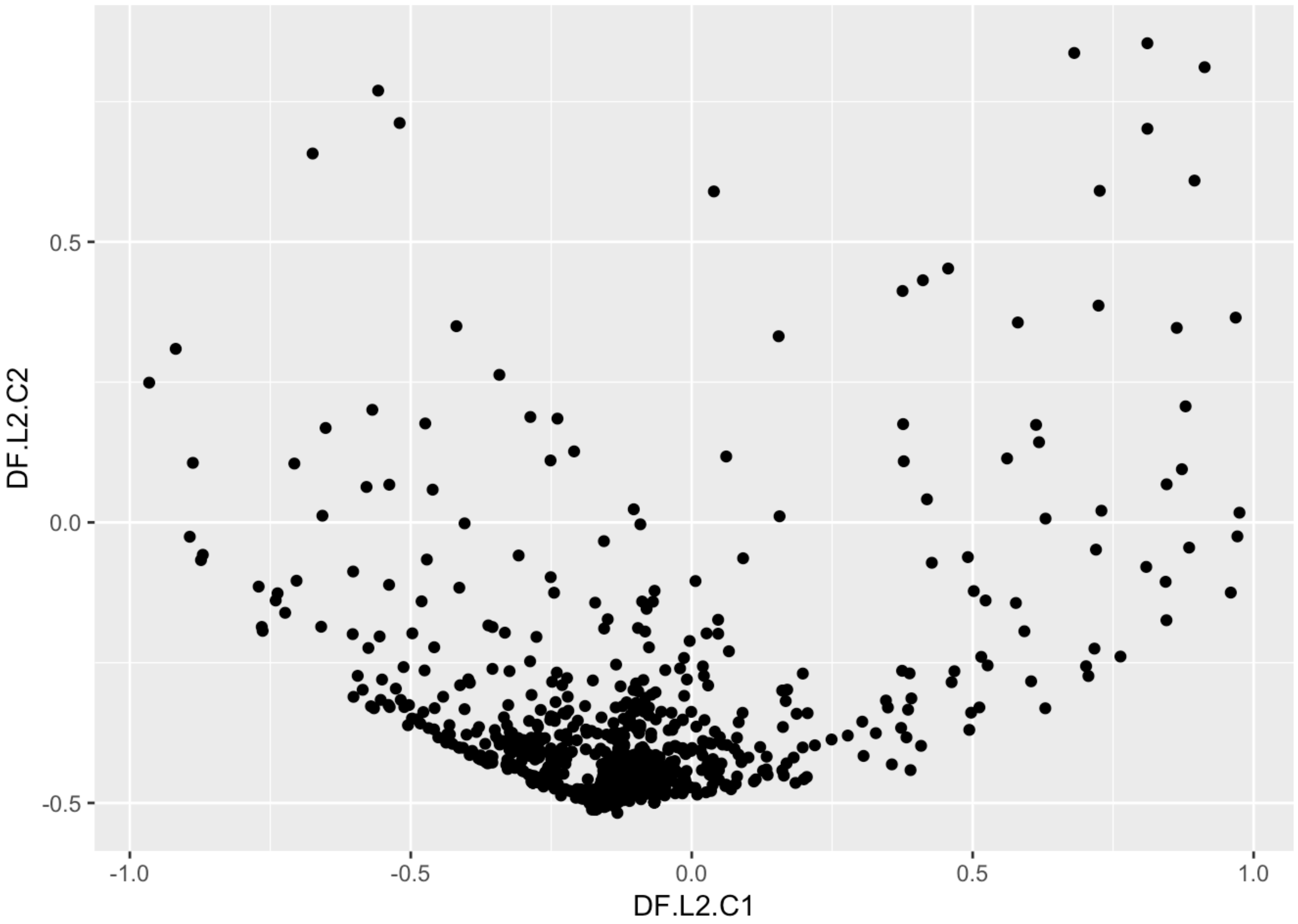
```
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto9 <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

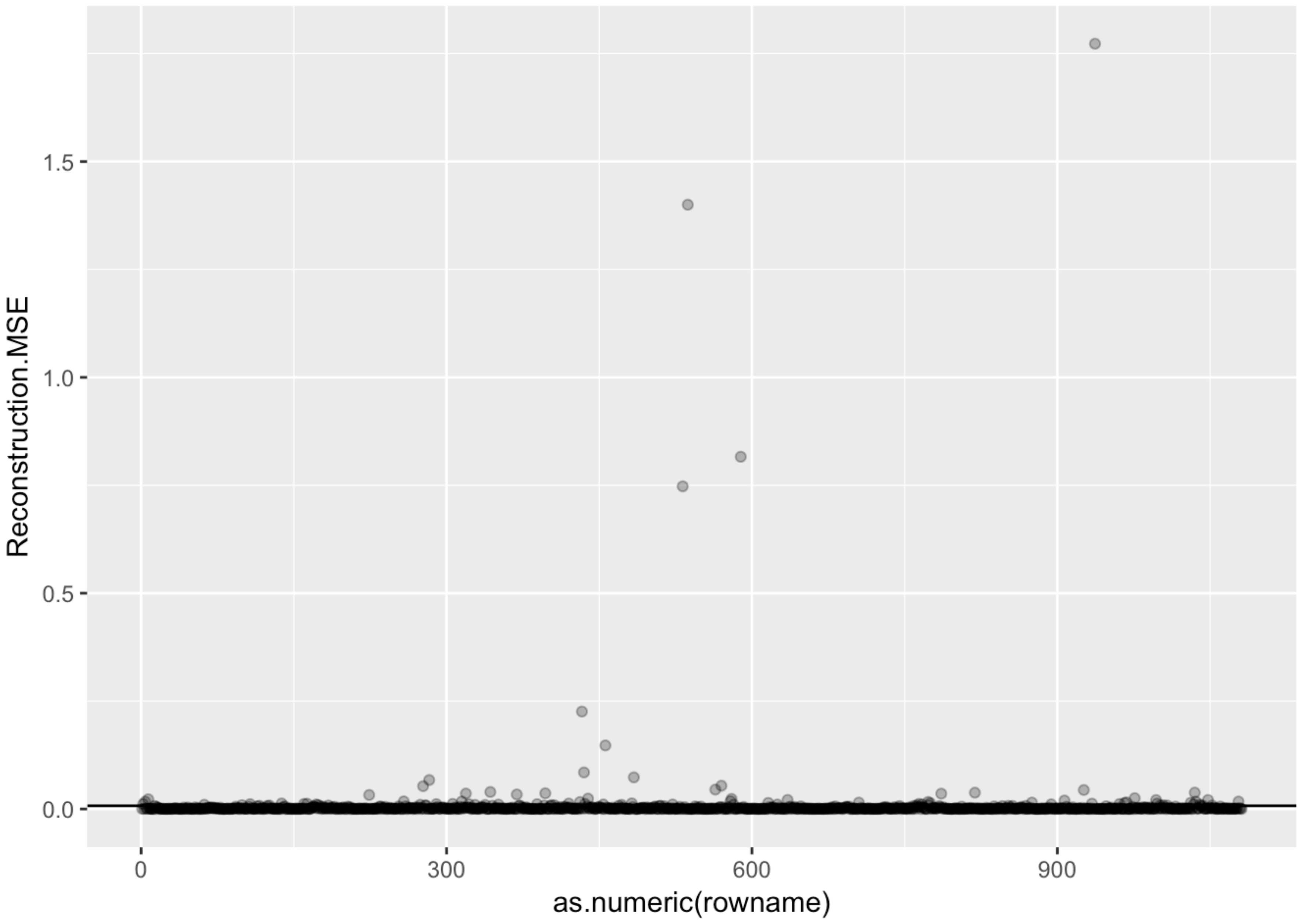
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly detection function
```



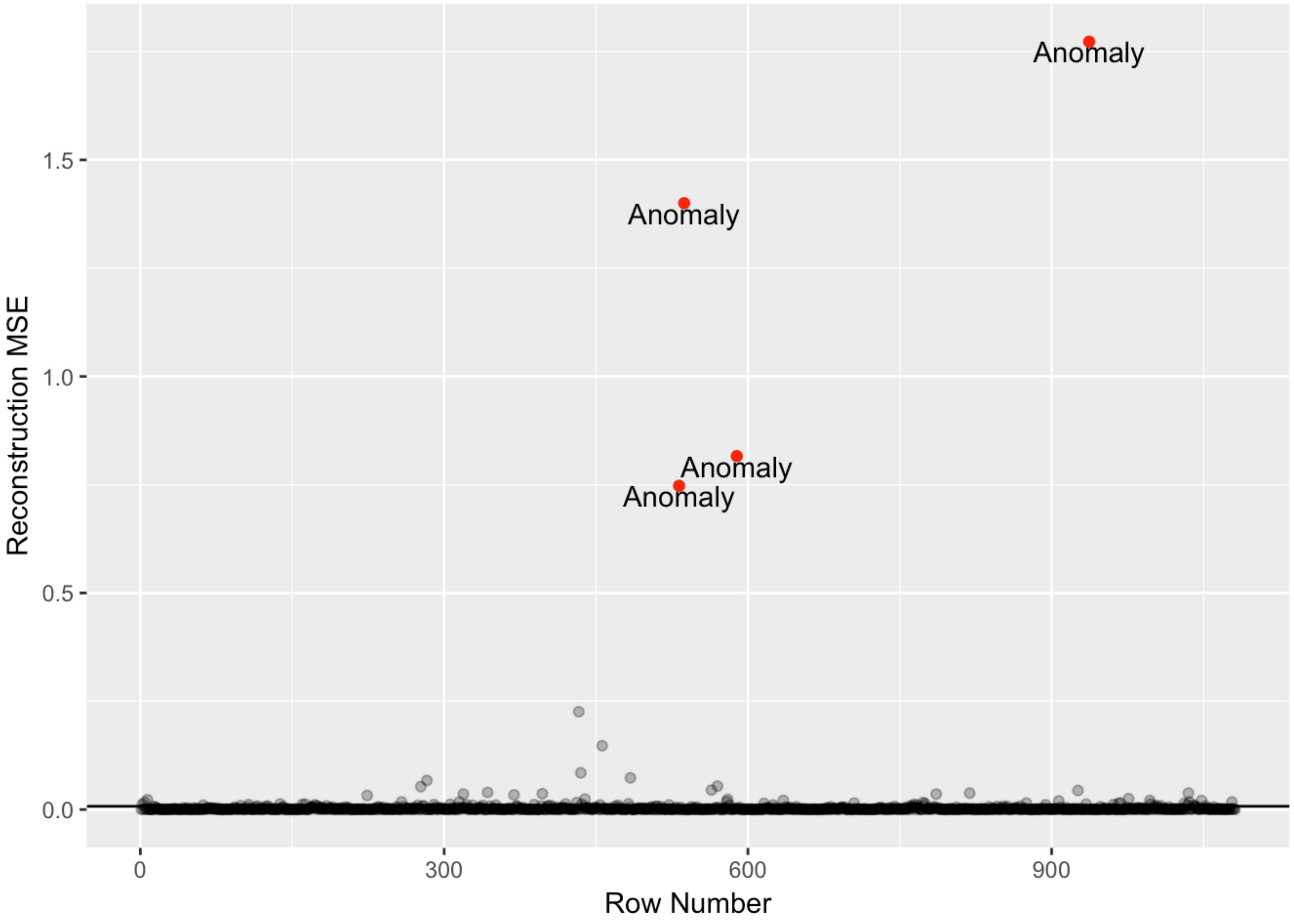
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007565647
```

```
##          mean
## 1 0.007541658
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto10 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto9 <- rbind(fraud_auto9, fraud_auto10)

#Hyperparameters: hidden= 10,2,10 epochs = 200
#unsupervised neural network model using deep learning autoencoders (autoencoder = TRUE)
#“bottleneck” training, hidden layer in the middle is very small. Model will reduce the dimensionality of the inp
ut data (in this case, down to 2 nodes/dimensions).
model <- h2o.deeplearning(
x = x_col,
training_frame = train,
model_id = "model",
autoencoder = TRUE,
hidden = c(10, 2, 10),
epochs = 200,
activation = "Tanh"
)
```

```
##
  |
  |                                                                 |   0%
  |
  |===                                                              |   5%
  |
  |================                                                 |  25%
  |
  |==========================                                       |  40%
  |
  |=======================================                          |  60%
  |
  |====================================================             |  80%
  |
  |=================================================================| 100%
```

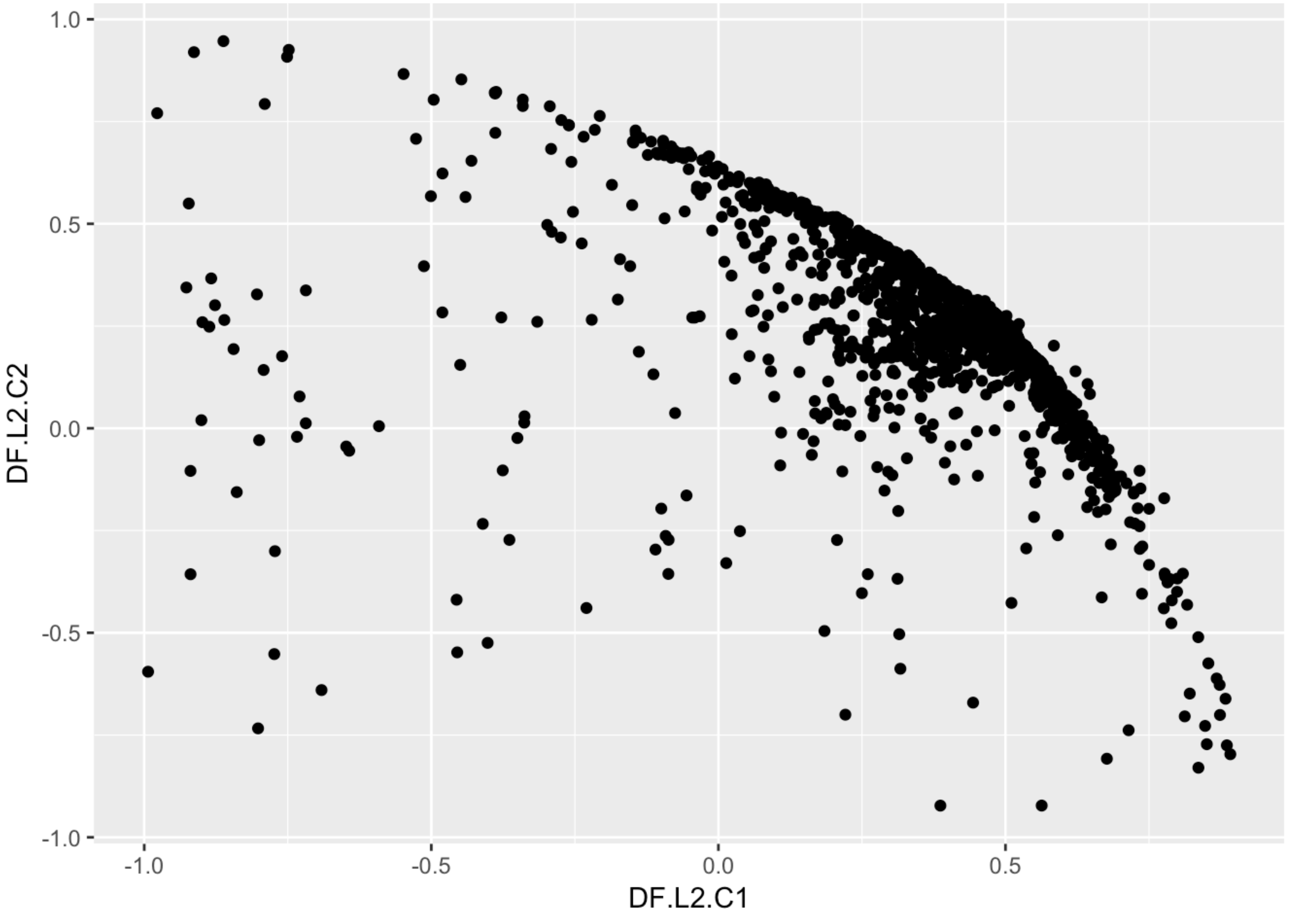
```
#view model details
model
```

```
## Model Details:
## ==============
##
## H2OAutoEncoderModel: deeplearning
## Model ID:  model
## Status of Neuron Layers: auto-encoder, gaussian distribution, Quadratic loss, 1,070 weights/biases, 22.6 KB, 4
36,800 training samples, mini-batch size 1
##   layer units  type dropout       l1       l2 mean_rate rate_rms momentum
## 1     1    48 Input  0.00 %
## 2     2    10  Tanh  0.00 % 0.000000 0.000000  0.348282 0.362665 0.000000
## 3     3     2  Tanh  0.00 % 0.000000 0.000000  0.009478 0.012374 0.000000
## 4     4    10  Tanh  0.00 % 0.000000 0.000000  0.013723 0.010302 0.000000
## 5     5    48  Tanh         0.000000 0.000000  0.010195 0.016526 0.000000
##   mean_weight weight_rms mean_bias bias_rms
## 1
## 2   -0.009823   0.266514 -0.469928 0.919526
## 3   -0.062443   0.555147 -0.341334 0.106308
## 4   -0.172732   0.815202  0.384186 0.789907
## 5    0.018930   0.136326 -0.076747 0.119782
##
##
## H2OAutoEncoderMetrics: deeplearning
## ** Reported on training data. **
##
## Training Set Metrics:
## =====================
##
## MSE: (Extract with `h2o.mse`) 0.002539306
## RMSE: (Extract with `h2o.rmse`) 0.05039153
```

```
#Test1
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test1_col <-
h2o.deepfeatures(model, test1, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                 |   0%
  |
  |================================                                 |  50%
  |
  |=================================================================| 100%
```

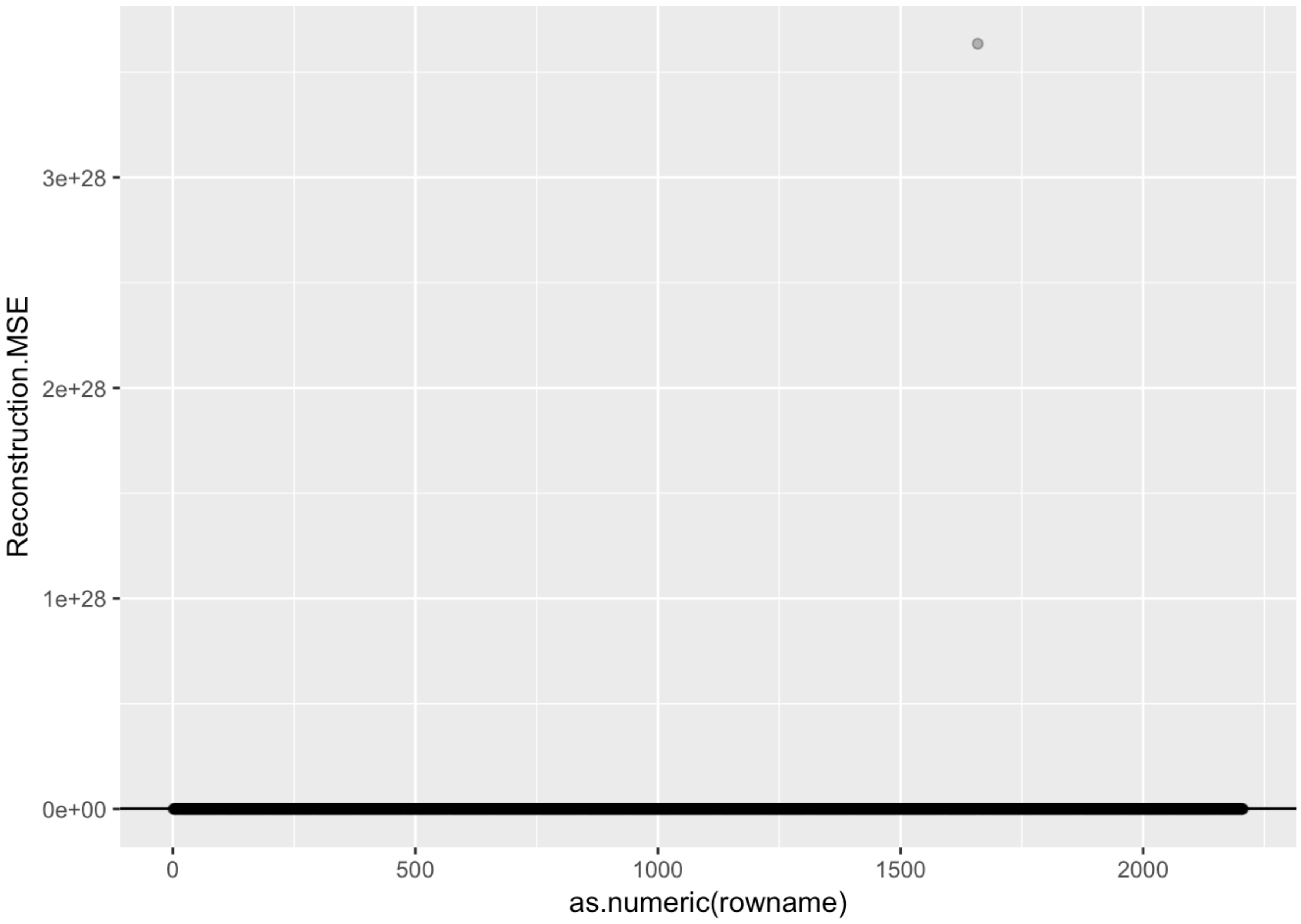
```
ggplot(test1_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly detection function
```



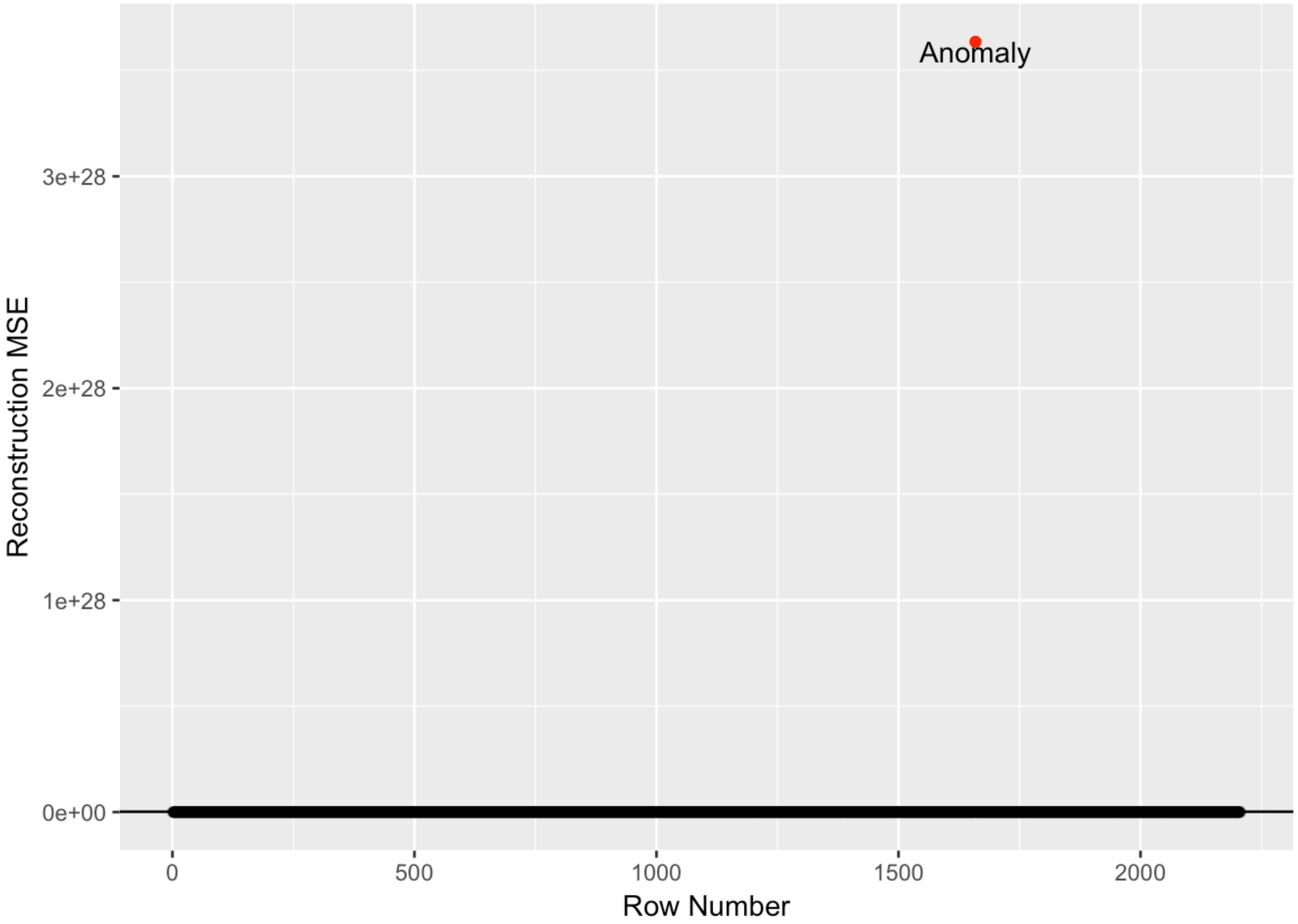
```
#anomaly detection
anomaly <- h2o.anomaly(model, test1) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #1.647652e+25
```

```
##           mean
## 1 1.647652e+25
```

```
#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



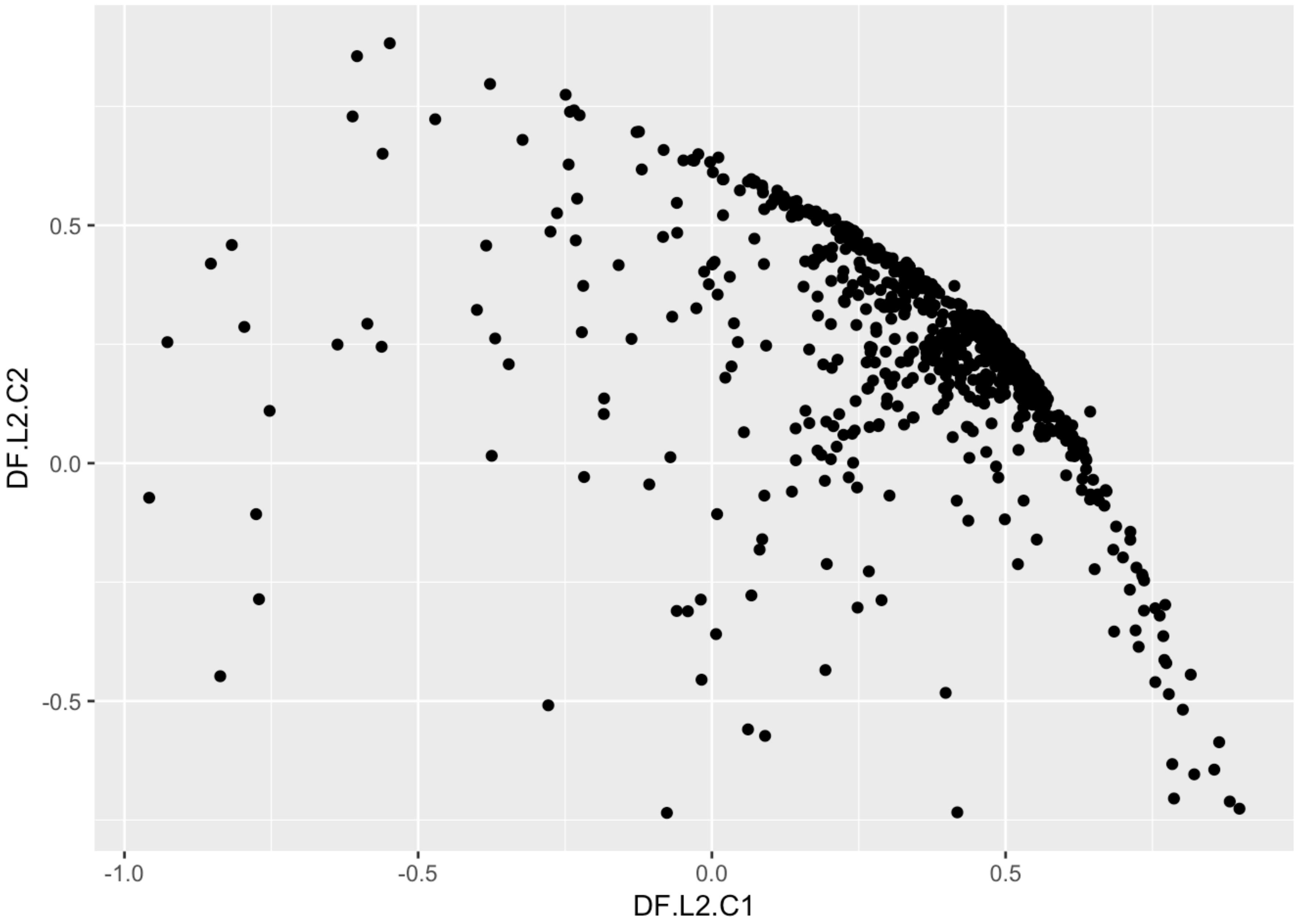
```
#add outlier vs no outlier to anomaly df
anomaly <- anomaly %>%
mutate(outlier = ifelse(Reconstruction.MSE > 1.647652e+25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier <-
anomaly[which(anomaly[, 2] > 1.647652e+25), ] #row 1659 looks to be an anomaly

#plot
ggplot(anomaly, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier, colour =
"red") +
geom_text(data = outlier,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions row 1659 of test1 is equal to row 4165 of the model_df
fraud_auto11 <- model_df[4165, ]

#Test2
#dimensionality reduction to explore feature space
#extract this hidden feature and plot it to show the reduced #representation of the input data.
test2_col <-
h2o.deepfeatures(model, test2, layer = 2) %>% as.data.frame()
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

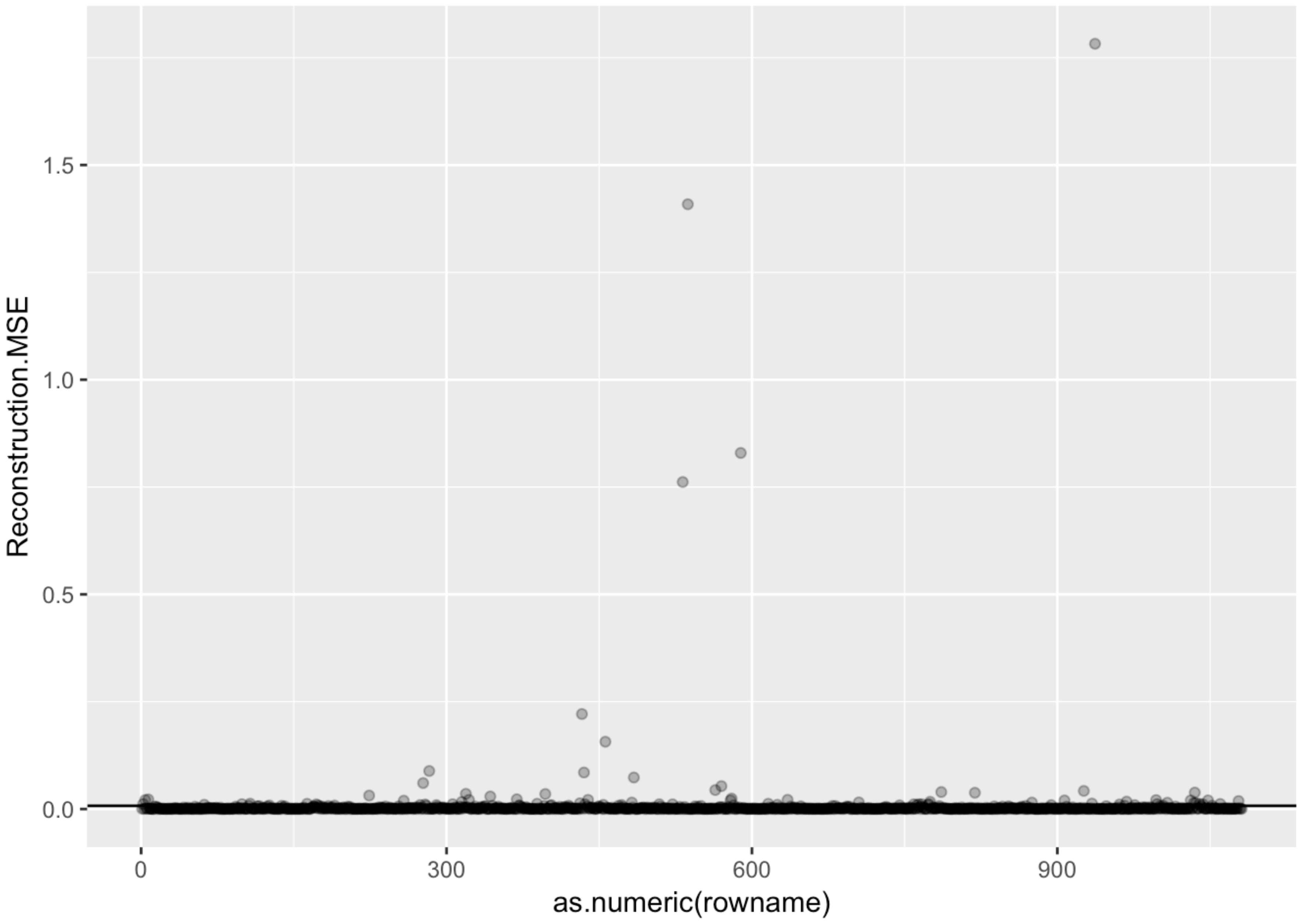
```
ggplot(test2_col, aes(x = DF.L2.C1, y = DF.L2.C2)) +
geom_point() #difficult to see outliers, use anomaly detection function
```



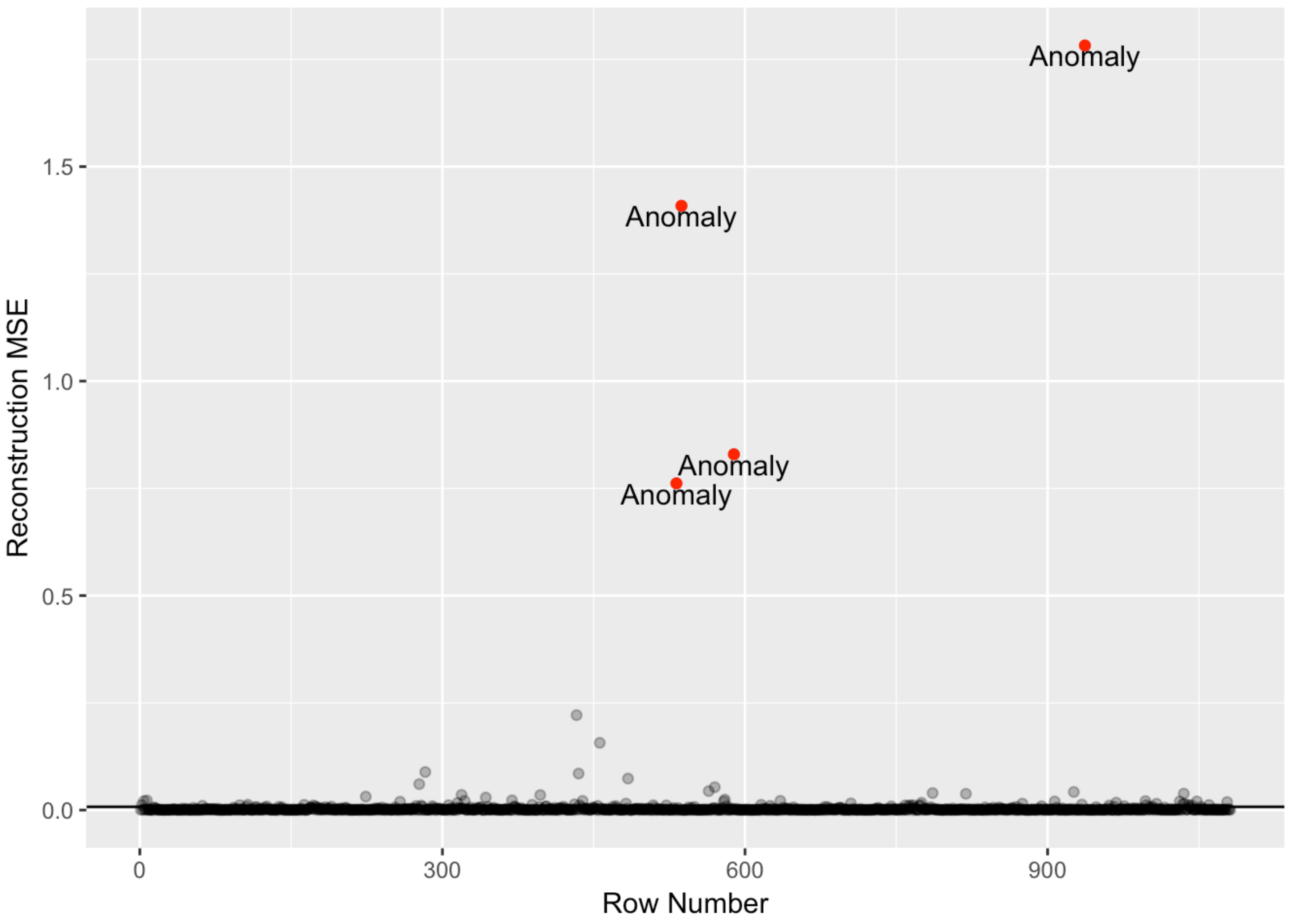
```
#anomaly detection
anomaly2 <- h2o.anomaly(model, test2) %>%
as.data.frame() %>%
tibble::rownames_to_column()

mean_mse <- anomaly2 %>%
summarise(mean = mean(Reconstruction.MSE))
mean_mse #0.007465285
```

```
##          mean
## 1 0.007538041
```

```
#plot
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1")
```



```
#add outlier vs no outlier to anomaly2 df
anomaly2 <- anomaly2 %>%
mutate(outlier = ifelse(Reconstruction.MSE > 0.25, "outlier", "no_outlier"))

#subset which rows greater than mean mse
outlier2 <-
anomaly2[which(anomaly2[, 2] > 0.25), ] #4 possible anomalies

#plot, possibly anomaly points are colored red
ggplot(anomaly2, aes(x = as.numeric(rowname), y = Reconstruction.MSE)) +
geom_point(alpha = 0.3) +
geom_hline(data = mean_mse, aes(yintercept = mean)) +
scale_color_brewer(palette = "Set1") +
labs(x = "Row Number", y = "Reconstruction MSE") + geom_point(data = outlier2, colour =
"red") + geom_text(data = outlier2,
label = "Anomaly",
vjust = 1)
```



```
#create DF that include possible fraud transactions
#row 532 of test2 is equal to row 2687 of the model_df
#row 537 of test2 is equal to row 2713 of the model_df
#row 589 of test2 is equal to row 573 of the model_df
#row 937 of test2 is equal to row 4752 of the model_df
fraud_auto12 <- model_df[c(2687, 2713, 573, 4752), ]
fraud_auto11 <- rbind(fraud_auto11, fraud_auto12)
```

2.3 Anomalies

```
#all versions of tuned hyperparameters resulted in the same outliers
#view fraud_auto dataframe
kable(fraud_auto) %>% kable_styling(latex_options = "scale_down")
```

	Agency_Name	Merchant_Category	Max_201307	Max_201308	Max_201309	Max_201310	Max_201311	Max_201312	Max_201401	Ma
4165	REDLANDS COMMUNITY COLLEGE	LUXOR HOTEL AND CASINO	0.000000	0.000000	0.000000	0.000000	-2.421811e+16	0.0000000	0	
2687	MENTAL HEALTH AND SUBSTANCE ABUSE SERV.	RENAISSANCE HOTELS	1.773986	1.656732	0.000000	0.000000	7.434916e-01	2.9160310	0	
2713	N. E. OKLA. A & M COLLEGE	AUTOMATED FUEL DISPENSER	0.000000	0.000000	1.234737	1.349614	0.000000e+00	0.4567655	0	
573	DEPARTMENT OF AGRICULTURE	HOLIDAY INNS	0.000000	0.000000	1.657086	2.485629	1.181337e+00	3.7576390	0	
4752	STATE ELECTION BOARD	STATIONERY,OFFICE AND SCHOOL SUPPLY STORES	0.000000	0.000000	0.000000	0.000000	0.000000e+00	4.9158030	0	

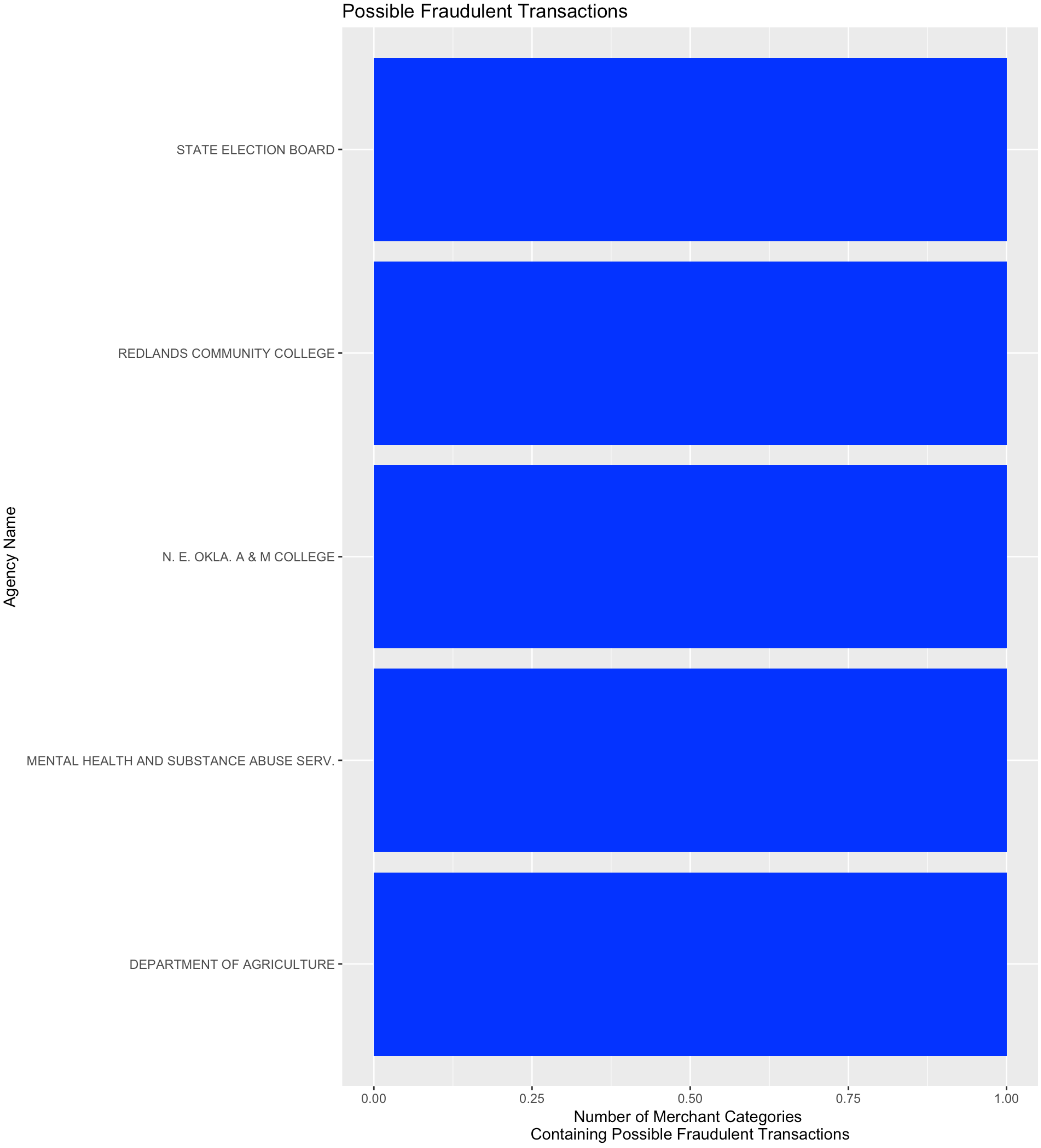
2.4 Business Insight

Agency transactions that occurred within the merchant category listed in the fraud_auto data frame could possibly be fraud based on my Autoencoder analysis. Transactions that occurred within these merchant categories at these agencies require further analysis to determine if fraud actually occurred.

Of the five agency transactions flagged as possible anomalies using Autoencoder, all but one was also flagged with my MeanShift model. The one data point that was flagged by Autoencoder but not MeanShift was the Holiday Inns category of the Department of Agriculture.

Overall, I found using Autoencoder more challenging for anomaly detection than MeanShift as the model did not cluster data points.

```
#Plot number of merchant categories that contain
#possible fraudulent transaactions by agency
ggplot(data = fraud_auto, mapping = aes(Agency_Name)) + geom_bar(fill = 'blue', stat = "count") + xlab("Agency Na
me") + ylab("Number of Merchant Categories \nContaining Possible Fraudulent Transactions") + ggtitle("Possible Fr
audulent Transactions") + coord_flip()
```


Possible Fraudulent Transactions
Agency Name
STATE ELECTION BOARD
REDLANDS COMMUNITY COLLEGE
N. E. OKLA. A & M COLLEGE
MENTAL HEALTH AND SUBSTANCE ABUSE SERV.
DEPARTMENT OF AGRICULTURE
Number of Merchant Categories
Containing Possible Fraudulent Transactions