Using t-SNE to build useful features in predictive models

ADVANCED DIMENSIONALITY REDUCTION IN R

Federico Castanedo
Data Scientist at DataRobot

Benefits of using t-SNE in predictive models

Training predictive models with a good dimensionality reduction provide the following benefits:

1. Less correlation of input features
2. Reduction in computation time

Credit card fraud detection dataset

- European credit card transactions in September 2013
- Released by Andrea Dal Pozzolo, et al. and available in **Kaggle datasets**
- Highly unbalanced: 492 fraud cases out of 248,807 (0.172%)
- Anonymized numerical features which are the result of a PCA
- 30 features plus the Class (1 fraud, 0 not-fraud)
- We only know the meaning of two features: time and amount of the transaction

Credit card: data exploration

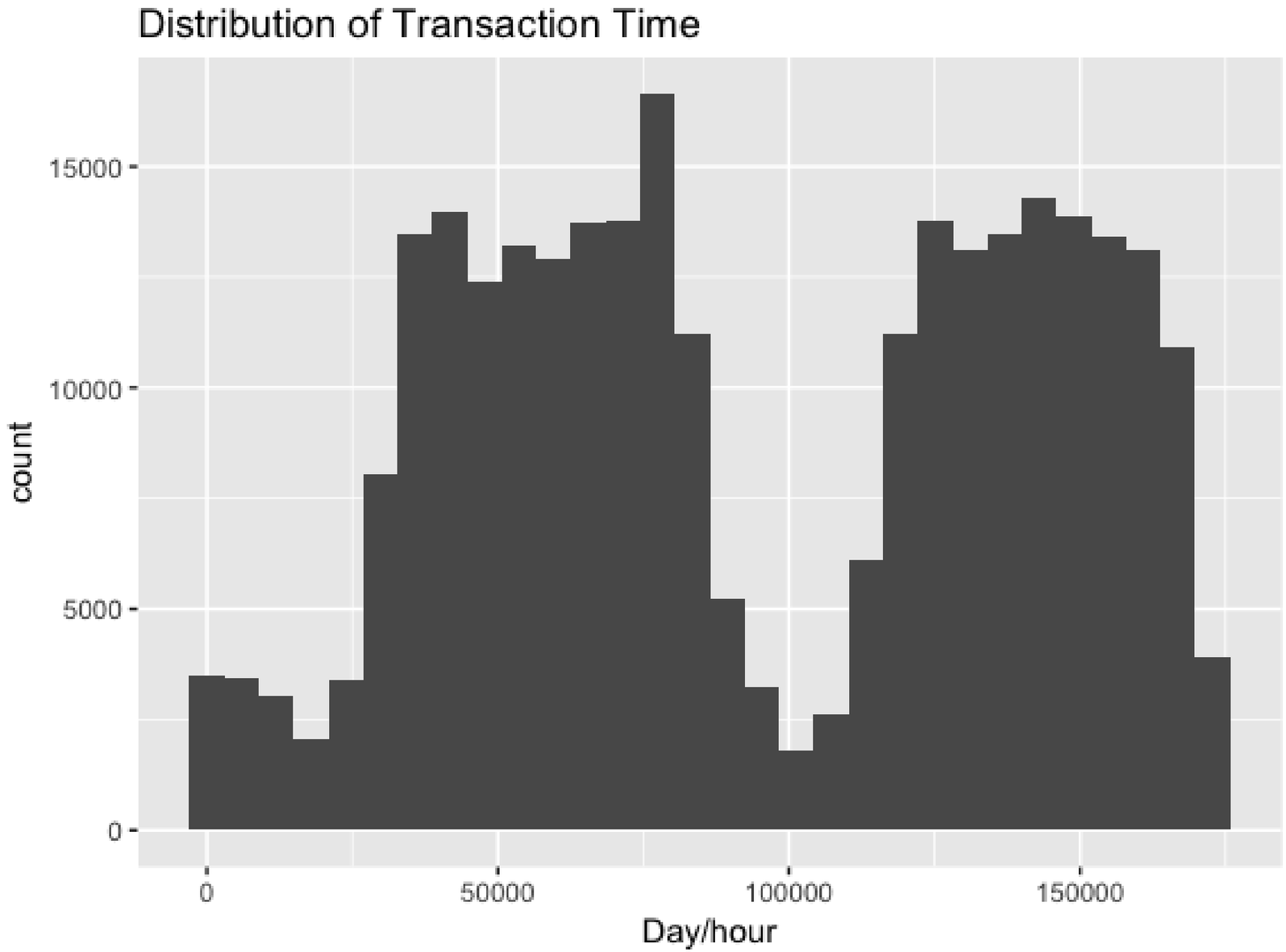
- We do not have unknown values
- The transaction amount is small with some outliers

```
summary(creditcard$Amount)
```

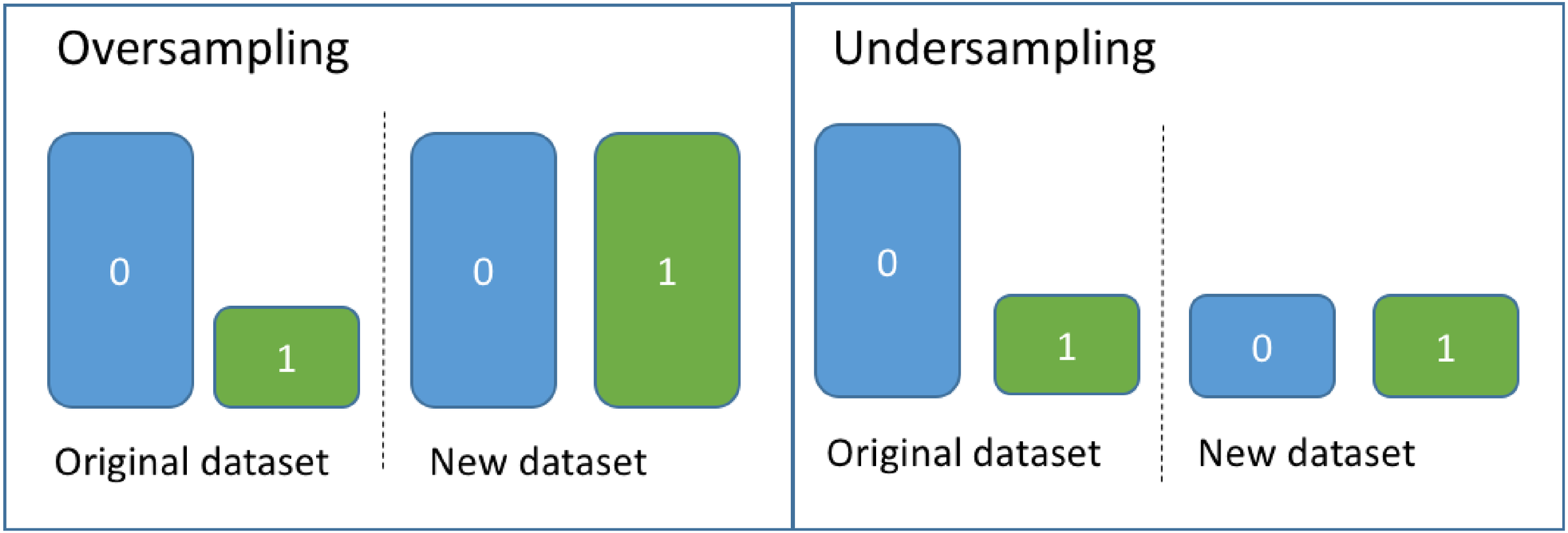
```
 Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
 0.00     5.60    22.00    88.35    77.17 25691.16
```

  - Distribution of transaction time

```
ggplot(creditcard, aes(x=Time)) +
  geom_histogram() +
  ggtitle("Distribution of Transaction Time")
```


Distribution of Transaction Time
15000
10000
5000
0
count
0
50000
100000
150000
Day/hour

Handling class imbalance



Data preparation

- Split data into train and test sets

```
set.seed(1234)
idx <- sample(1:nrow(creditcard), nrow(creditcard)*.20)
creditcard.test <- creditcard[idx]
creditcard.train <- creditcard[!idx]
```

Balance training set

Under-sample training set

```
creditcard.pos <- creditcard.train[Class==1]
creditcard.neg <- creditcard.train[Class==0]
creditcard.neg.bal <- creditcard.neg[sample(1:nrow(creditcard.neg),
                                            nrow(creditcard.pos))]
```

Balanced train set

```
creditcard.train <- rbind(creditcard.pos, creditcard.neg.bal)
```

Let's do some fraud detection!

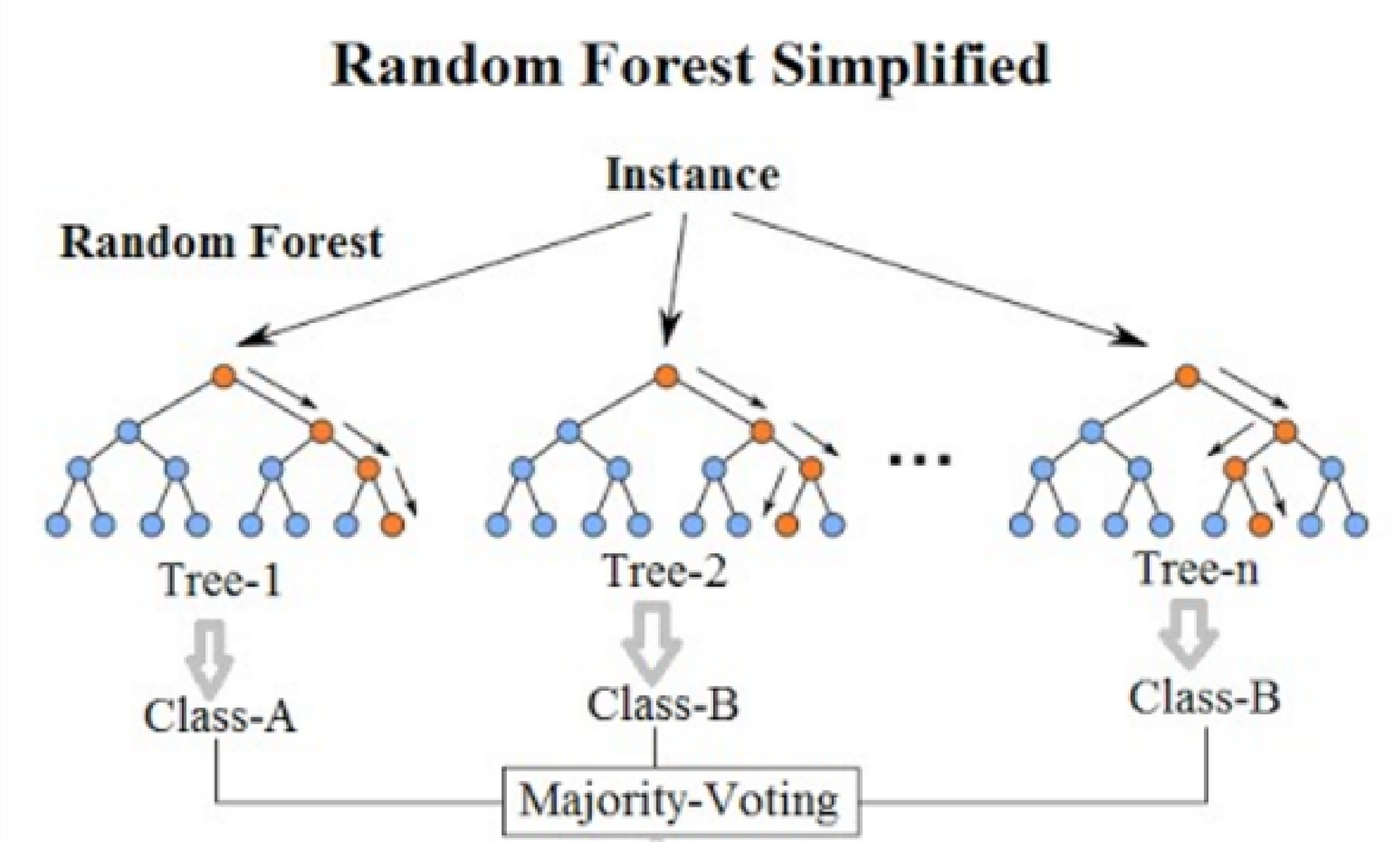
ADVANCED DIMENSIONALITY REDUCTION IN R

Training random forests models

ADVANCED DIMENSIONALITY REDUCTION IN R

Federico Castanedo
Data Scientist at DataRobot

Random forest



Training a random forest in R

- Several packages for implementing a random forest
- `randomForest` the most common

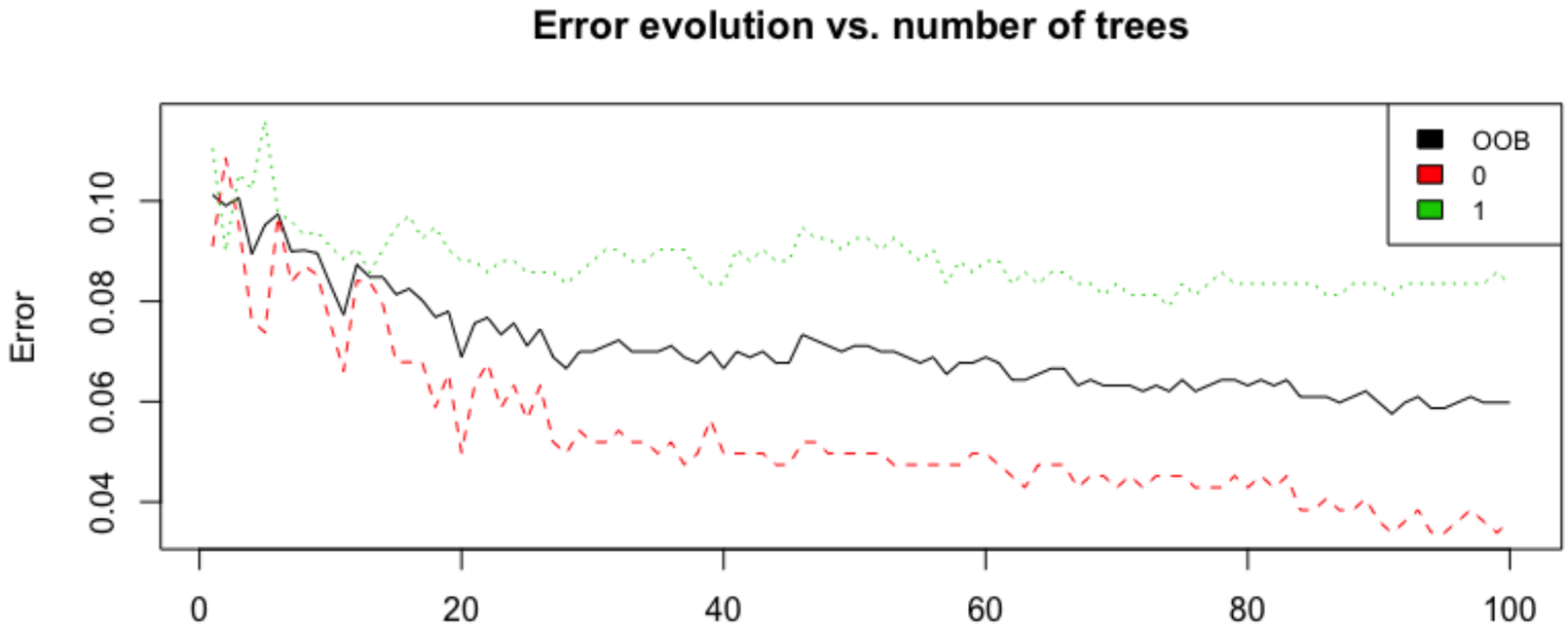
```
library(randomForest)
```

```
# Select the features and target
train_x <- creditcard_train[, -31]
train_y <- creditcard_train$Class
```

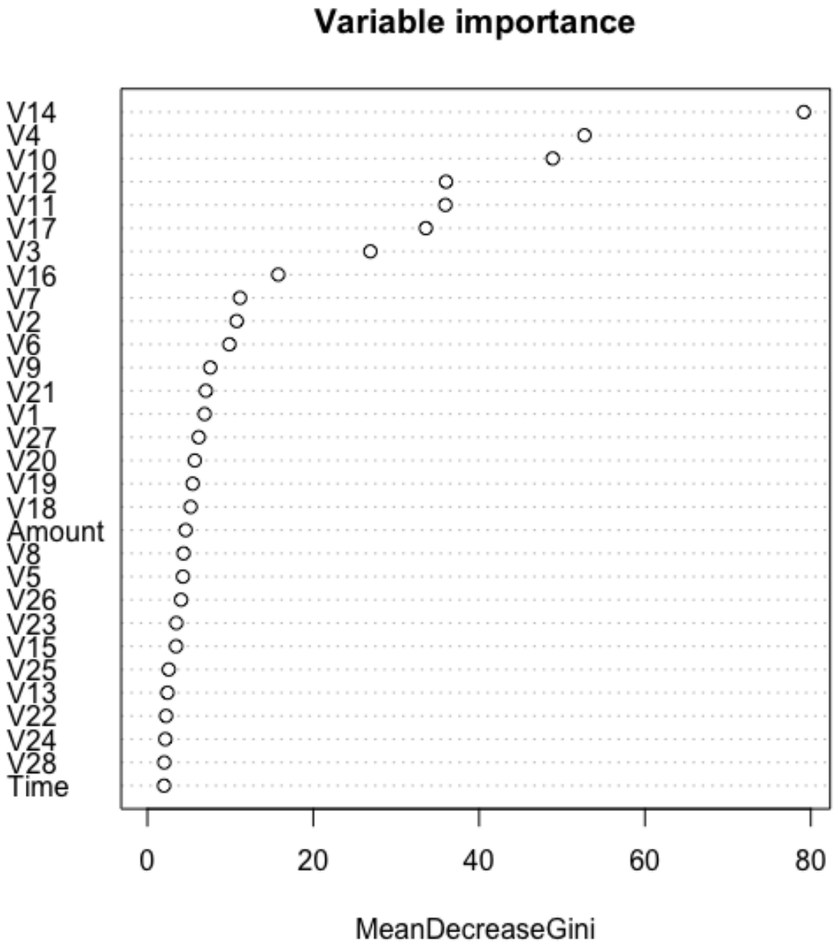
```
# Train the model
rf_model <- randomForest(x = train_x, y = train_y, ntree = 100)
```

Performance based on the number of trees

```
plot(rf_model, main = "Error evolution vs number of trees")
legend("topright", colnames(rf_model$err.rate),col=1:3,cex=0.8,fill=1:3)
```



```
varImpPlot(rf_model, main = "Variable importance")
```



Let's train some random forests!

ADVANCED DIMENSIONALITY REDUCTION IN R

Predicting data

ADVANCED DIMENSIONALITY REDUCTION IN R

Federico Castanedo
Data Scientist at DataRobot

Evaluate the model with test set

Evaluate the model using the test set (original distribution)

```
prop.table(table(creditcard_test$Class))
```

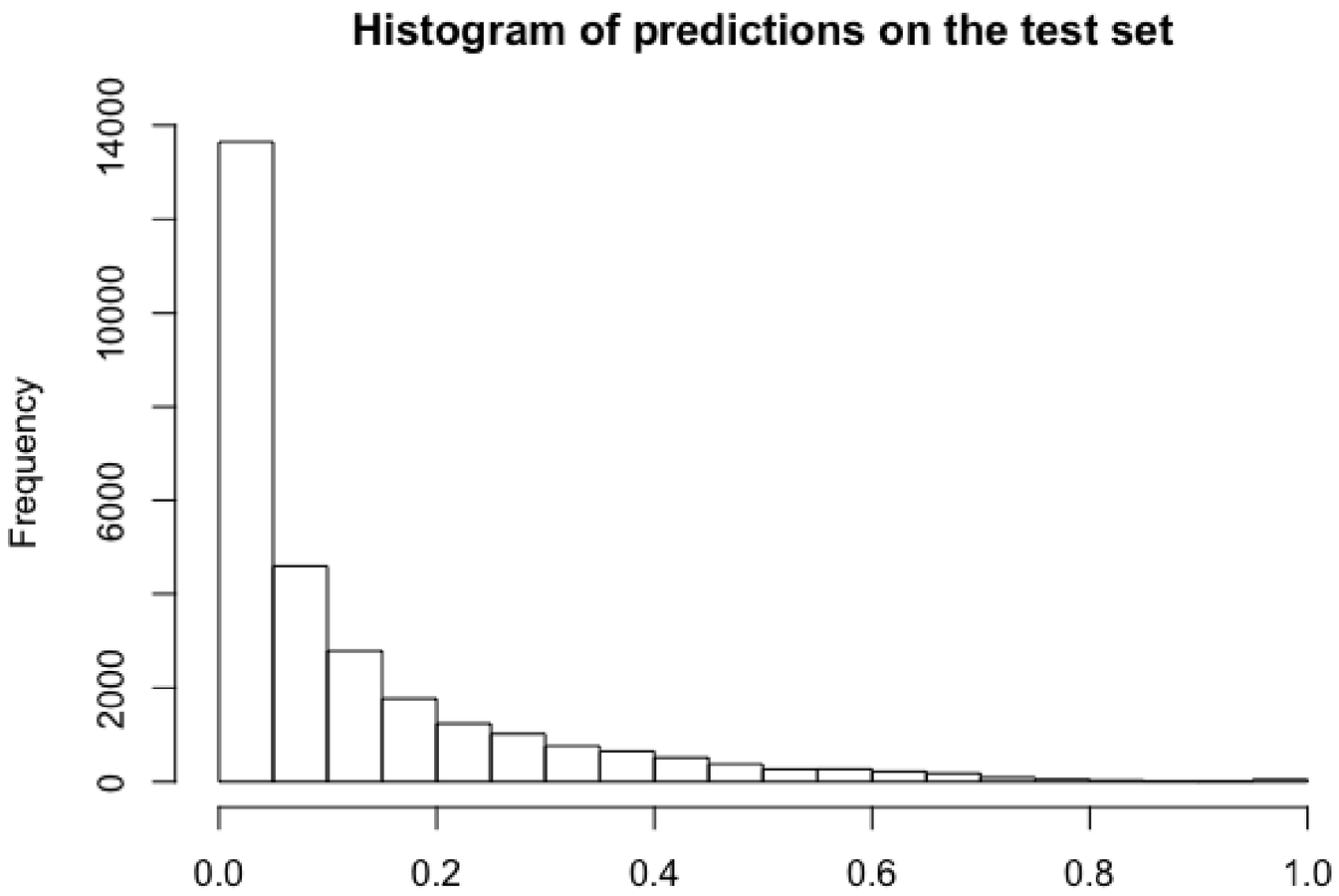
```
          0           1
0.998279494 0.001720506
```

Predictions using random forest

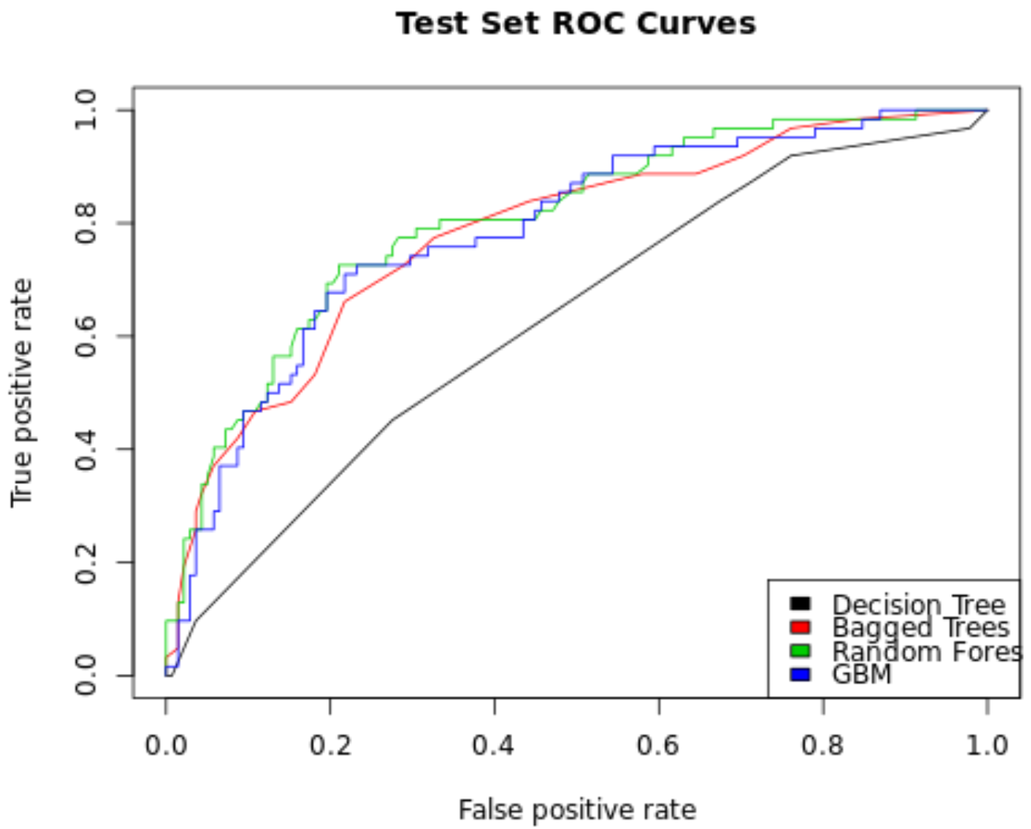
```
pred_rf <- predict(rf_model, creditcard_test, type = "prob")
head(pred_rf)
```

```
     0    1
1 0.33 0.67
2 0.97 0.03
3 1.00 0.00
4 0.89 0.11
5 1.00 0.00
6 0.91 0.09
```

```
hist(pred_rf[, 2], main = "Histogram of predictions on the test set",
     xlab = "prediction value")
```



Area under the ROC curve (AUC)



Source: ***Machine Learning with tree based models in R***. Data Camp course

AUC in R using ROCR package

Generate a prediction object used by `ROCR`

```
pred <- prediction(pred_rf[,2], creditcard_test$Class)
```

Compute the `auc` metric

```
perf <- performance(pred, measure = "auc")
```

Get the `y.values` slot of the object

```
perf@y.values
```

```
0.9801234
```

Let's practice!

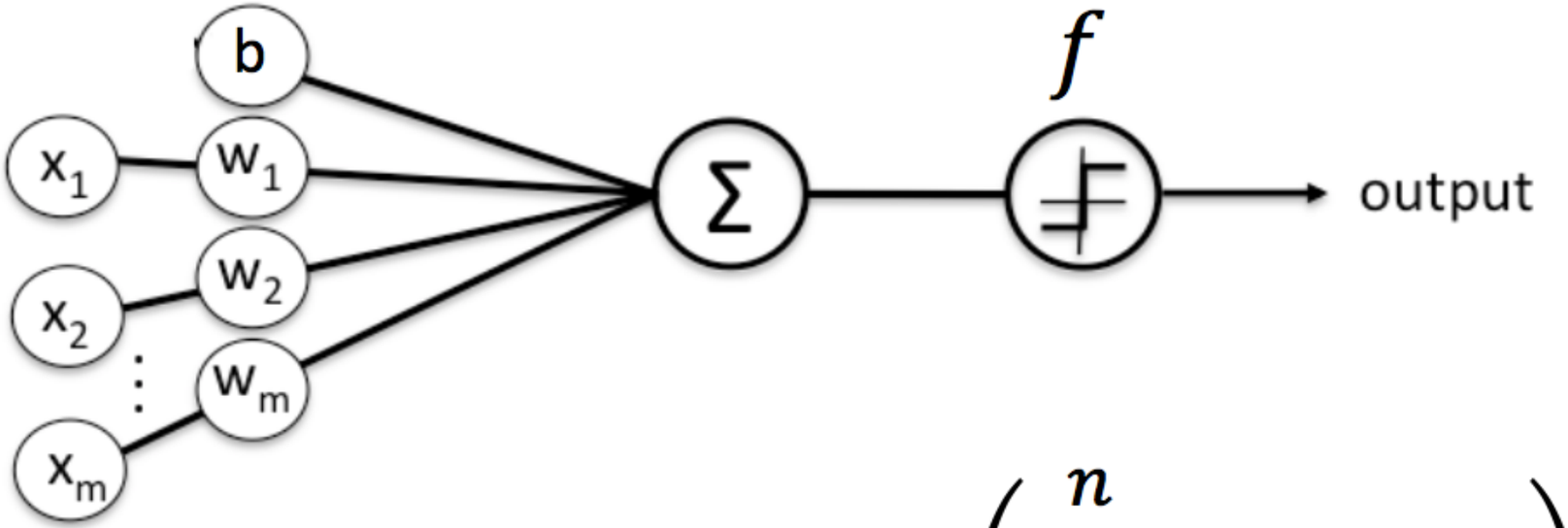
ADVANCED DIMENSIONALITY REDUCTION IN R

Visualizing neural networks layers

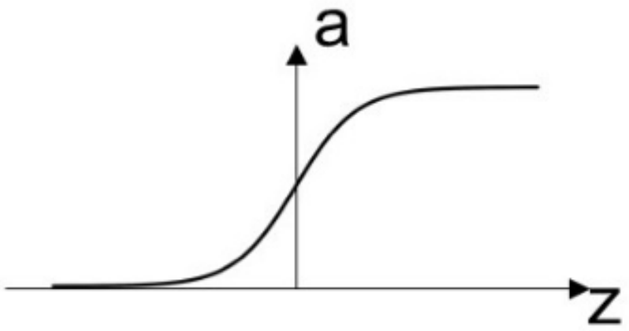
ADVANCED DIMENSIONALITY REDUCTION IN R

Federico Castanedo
Data Scientist at DataRobot

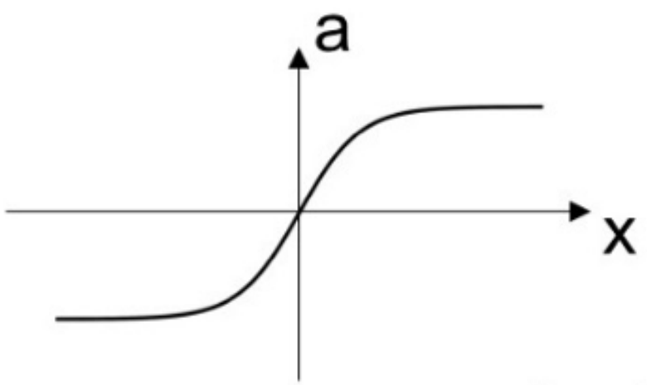
Neural networks: a neuron



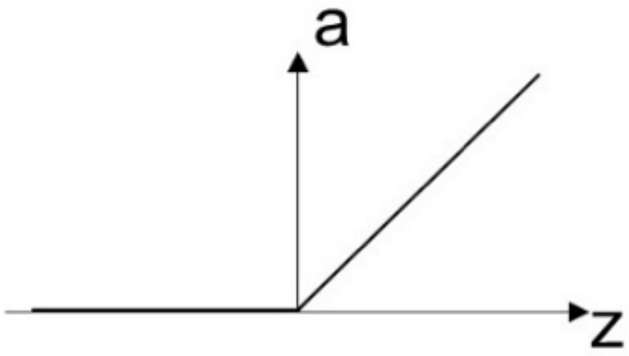
$$y(x) = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$



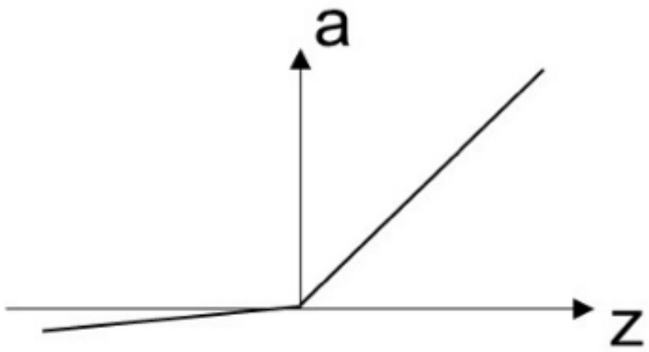
$sigmoid$ $a = \frac{1}{1 + e^{-z}}$



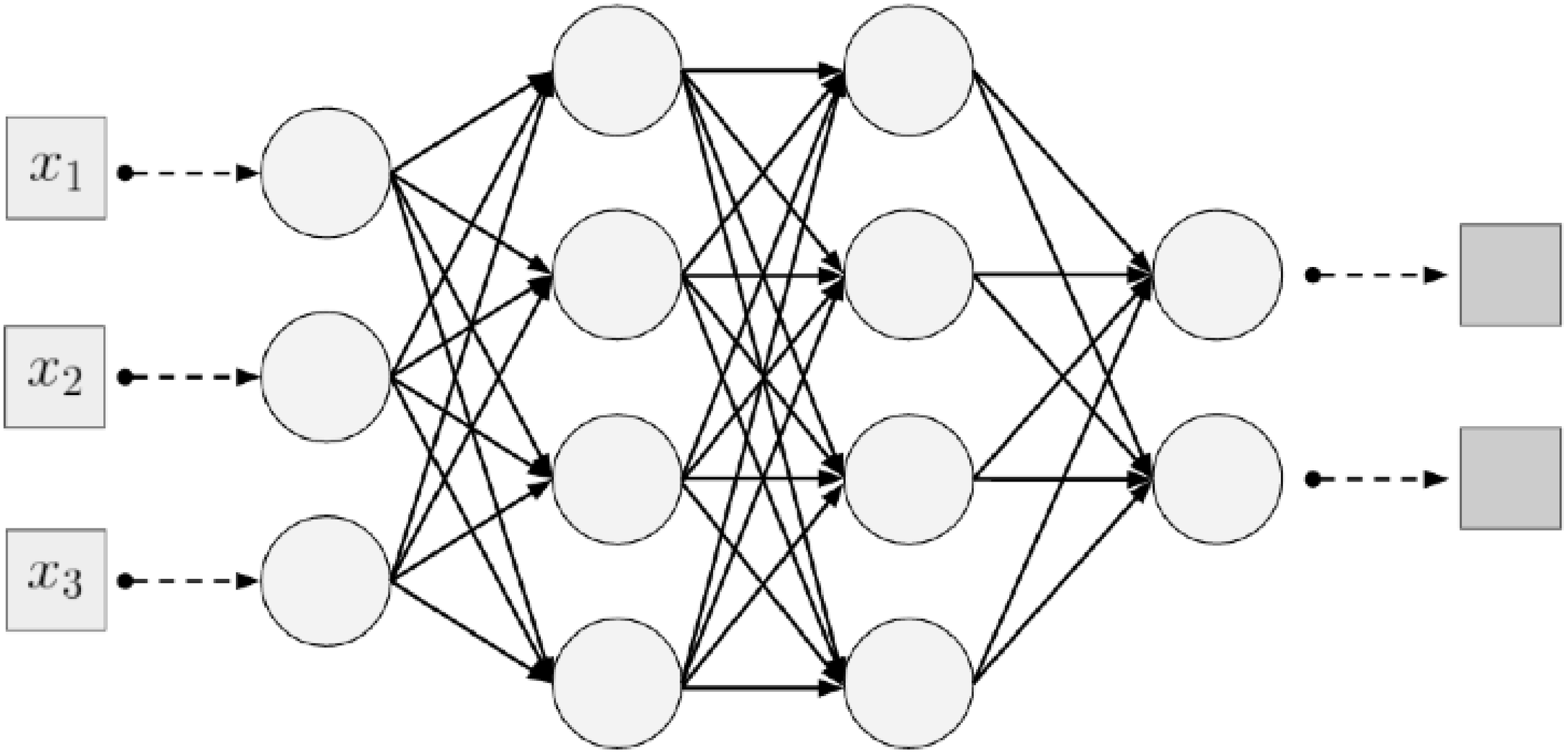
$tanh$: $a = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$

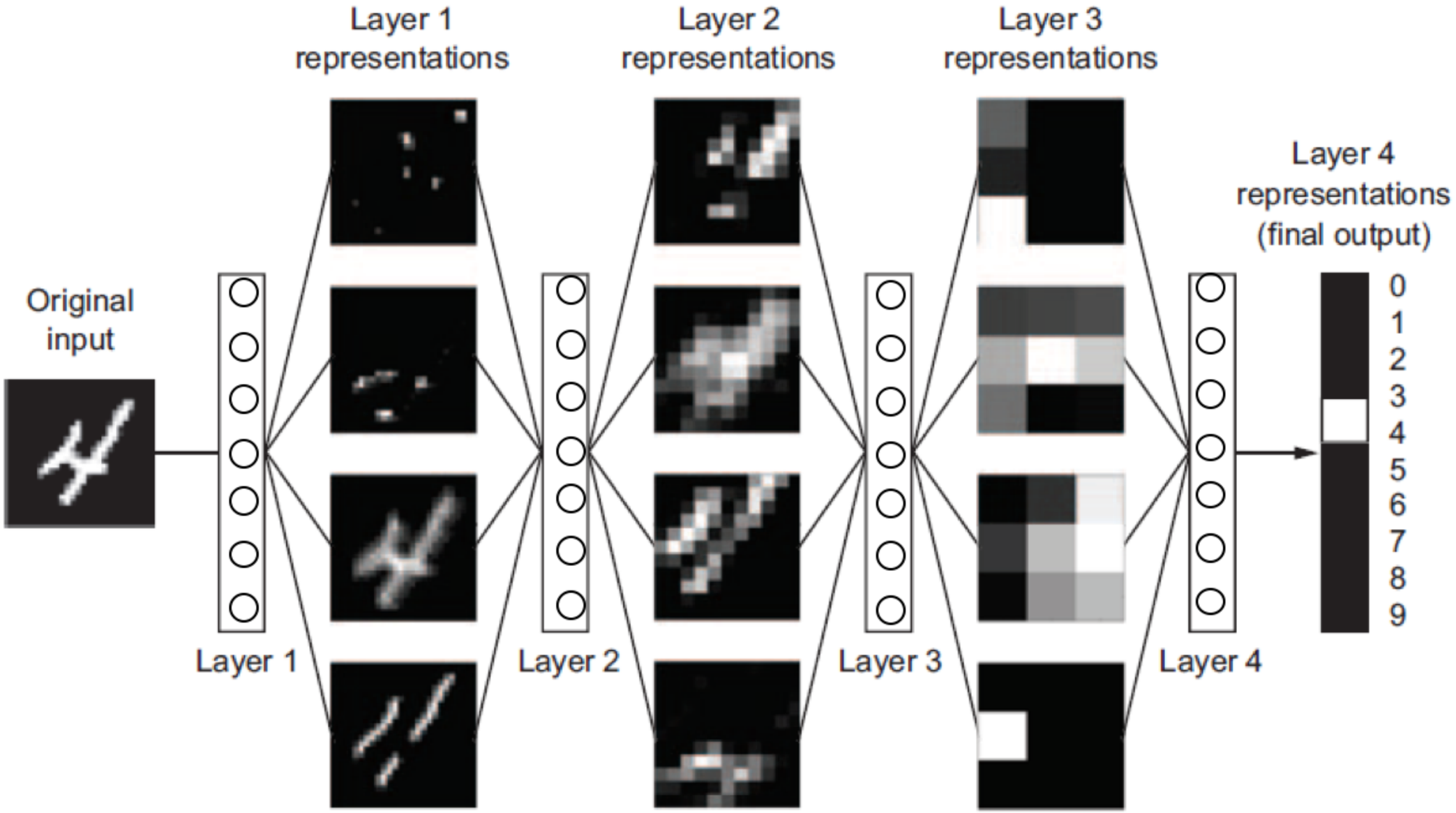


$ReLU$: $a = \max(0, z)$



$Leaky\ ReLU$: $a = \max(0.01z, z)$


x_1
x_2
x_3


Layer 1 representations
Layer 2 representations
Layer 3 representations
Layer 4 representations (final output)
Original input
Layer 1
Layer 2
Layer 3
Layer 4
0
1
2
3
4
5
6
7
8
9

Neural network outputs of each layer

- In the training phase weights are updated
- Extract neuron output values of each layer

```
head(layer_128_train[, 1:7])
```

```
   DF.L3.C1    DF.L3.C2 DF.L3.C3  DF.L3.C4 DF.L3.C5 DF.L3.C6 DF.L3.C7
1:        0 0.000000000 1.314435 1.4025972        0        0 1.928277
2:        0 0.008506777 1.605109 1.7618873        0        0 1.345420
3:        0 0.366096246 1.903230 1.3633492        0        0 1.171147
...
```

Neural network outputs of each layer

```
summary(layer_128_train[, 1:4])
```

```
    DF.L3.C1     DF.L3.C2          DF.L3.C3          DF.L3.C4
 Min.   :0   Min.   :0.0000   Min.   :0.06825   Min.   :0.0000
 1st Qu.:0   1st Qu.:0.0610   1st Qu.:1.22175   1st Qu.:0.8087
 Median :0   Median :0.2250   Median :1.38019   Median :1.0720
 Mean   :0   Mean   :0.2513   Mean   :1.39677   Mean   :1.1008
 3rd Qu.:0   3rd Qu.:0.4008   3rd Qu.:1.52888   3rd Qu.:1.3364
 Max.   :0   Max.   :1.1181   Max.   :3.57375   Max.   :2.4934
```

Visualising neural network layers

t-SNE of the neural network layer

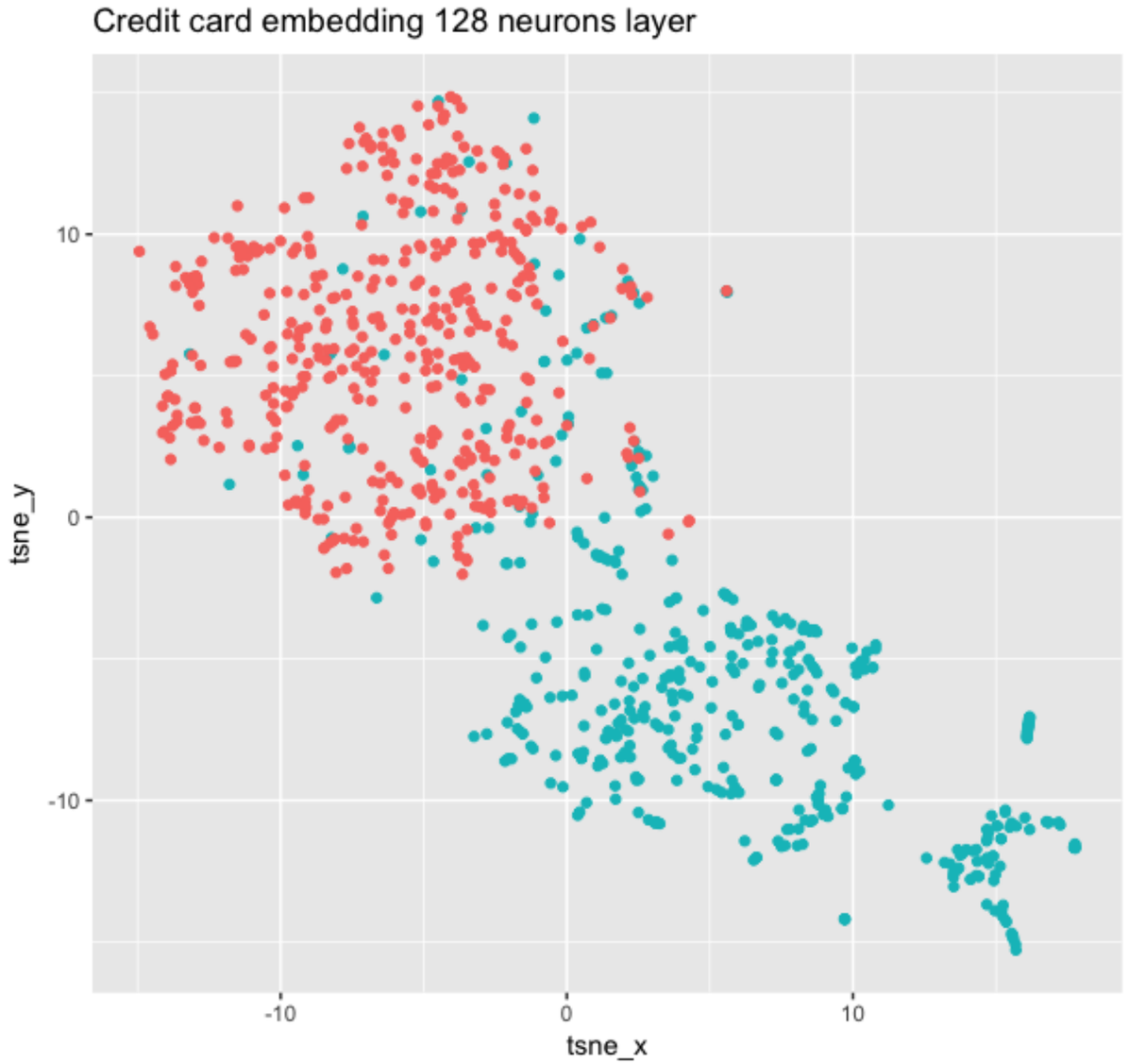
```
tsne_nn_layer_train <- Rtsne(as.matrix(layer_128_train), perplexity = 50,
                             max_iter = 400, check_duplicates = F,
                             dims = 2, verbose = T)
```

data frame with x and y coordinates and the Class

```
tsne_plot_train <- data.frame(tsne_x = tsne_nn_layer_train$Y[,1],
                              tsne_y = tsne_nn_layer_train$Y[,2],
                              y_col = creditcard_train$Class)
```

Generating the plot

```
ggplot(tsne_plot_train, aes(x = tsne_x, y = tsne_y, color = y_col)) +
   geom_point() +
   ggtitle("Credit card embedding 128 neurons layer") +
   theme(legend.position="none")
```


Credit card embedding 128 neurons layer
10
0
-10
tsne_y
-10
0
10
tsne_x

Let's practice!

ADVANCED DIMENSIONALITY REDUCTION IN R