Task B - Part 1

Statistical Analysis of Big Data

Amanda Magzal 207608647

Contents

1 Introduction

In this project, I study imbalanced classification and its impact on classification algorithms.

In part 1, I develop a statistical model for imbalanced classification, and use a Monte Carlo simulation in order to study via training-testing procedures the classification accuracy of logistic regression, taking into account the model I developed.

2 Data

The data includes information about clients who made credit card transactions and whether it turned out to be fraud.

Features:

- ID - Client Number
- GENDER - M: Male, F: Female
- CAR - Owns car
- REALITY - Owns a property
- NO_OF_CHILD - Number of children
- INCOME - Anual income
- EDUCATION_TYPE - Education level
- FAMILY_TYPE - Marital status
- HOUSE_TYPE - House type
- FLAG_MOBILE - Owns a mobile phone
- WORK_PHONE - Owns a work phone
- PHONE - Owns a phone
- OCCUPATION_TYPE - Occupation
- FAMILY_SIZE - Number of family members
- BEGIN_MONTH - The month of the extracted data
- YEARS_EMPLOYED - Years of employment
- Target - Fraud: 1, Not Fraud: 0

Table 1: Dataset

X	ID	GENDER	CAR	REALITY	NO_OF_CHILD	INCOME	INCOME_TYPE
0	5008806	M	Y	Y	0	112500	Working
1	5008808	F	N	Y	0	270000	Commercial associate
2	5008809	F	N	Y	0	270000	Commercial associate
3	5008810	F	N	Y	0	270000	Commercial associate
4	5008811	F	N	Y	0	270000	Commercial associate
5	5008815	M	Y	Y	0	270000	Working

The `X` column is just the index, the `ID` and `BEGIN_MONTH` columns are irrelevant and thus will be removed.

```
dat <- dat %>% dplyr::select(-X, -ID, -BEGIN_MONTH)
```

The `FLAG_MOBIL` has only one unique value (everyone owns a phone), and will also be removed.

```
unique(dat$FLAG_MOBIL)
```

```
## [1] 1
```

```
dat <- dat %>% dplyr::select(-FLAG_MOBIL)
```

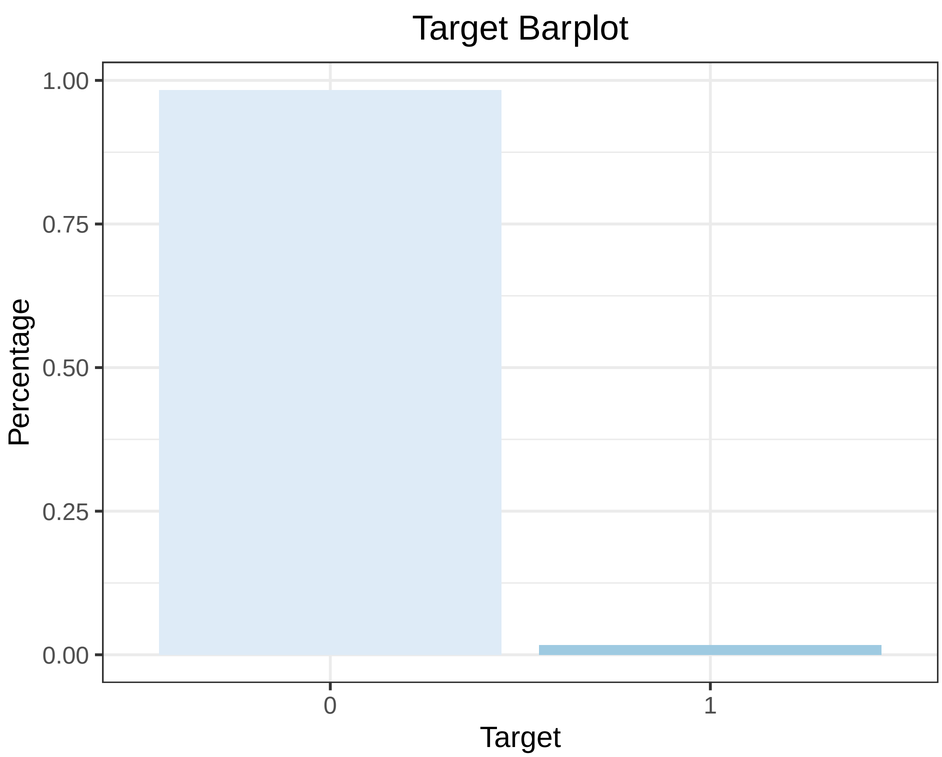
The `FAMILY.SIZE` and `NO_OF_CHILD` variables are highly correlated. Therefore, I decided to remove the `FAMILY.SIZE` variable.

```
cor(FAMILY.SIZE, NO_OF_CHILD)
```

```
## [1] 0.9022281
```

```
dat <- dat %>% dplyr::select(-FAMILY.SIZE)
```

The dataset is extremely imbalanced, with only 1.7% fraud transactions.



3 Statistical Model - Logistic Regression

To select the features to be included in the model, I fit a logistic regression model to the data.

```
full.model <- glm(TARGET ~ ., data = dat, family = 'binomial')
```

Next, I perform stepwise selection to find the best model.

```
step.model <- full.model %>% stepAIC(trace = FALSE)
summary(step.model)
```

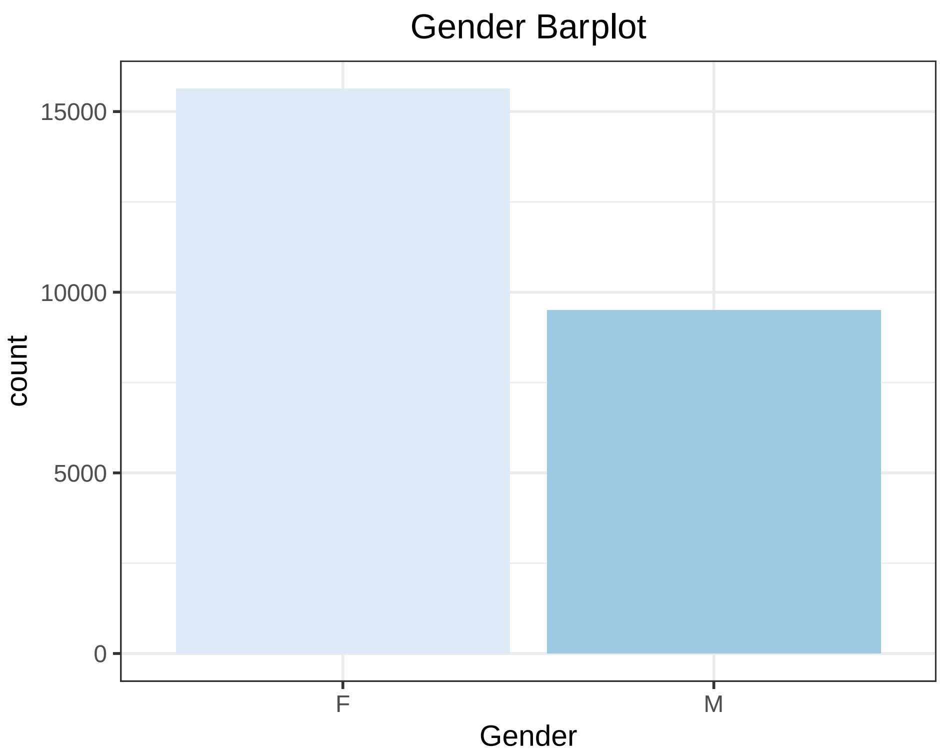
```
##
## Call:
## glm(formula = TARGET ~ GENDER + REALITY + INCOME_TYPE + FAMILY_TYPE +
##     YEARS_EMPLOYED, family = "binomial", data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.3409  -0.1995  -0.1757  -0.1515   3.3280
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -3.811e+00  2.130e-01 -17.894  < 2e-16 ***
## GENDERM                       2.556e-01  1.027e-01   2.489 0.012820 *
## REALITYY                     -3.429e-01  1.012e-01  -3.389 0.000702 ***
## INCOME_TYPEPensioner          1.875e+01  2.421e+02   0.077 0.938286
## INCOME_TYPEState servant     -2.069e-01  2.106e-01  -0.982 0.326066
## INCOME_TYPEStudent           -1.003e+01  2.788e+02  -0.036 0.971287
## INCOME_TYPEWorking            2.285e-04  1.123e-01   0.002 0.998376
## FAMILY_TYPEMarried            9.046e-02  1.919e-01   0.471 0.637428
## FAMILY_TYPESeparated         -2.763e-01  3.169e-01  -0.872 0.383342
## FAMILY_TYPESingle / not married  4.503e-01  2.142e-01   2.102 0.035537 *
## FAMILY_TYPEWidow              7.889e-01  3.213e-01   2.456 0.014065 *
## YEARS_EMPLOYED               -4.988e-02  1.034e-02  -4.824  1.4e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4286.3  on 25133  degrees of freedom
## Residual deviance: 4109.8  on 25122  degrees of freedom
## AIC: 4133.8
##
## Number of Fisher Scoring iterations: 13
```

The coefficients for the logistic regression model that I develop will be the ones estimated for the original data.

```
beta <- as.vector(coef(step.model))
```

3.1 Feature 1 - Gender

The gender feature's distribution is shown in the following plot.



I define the feature `Male` as follows:

$$\text{Male} \sim \text{Bernoulli}(0.3781)$$

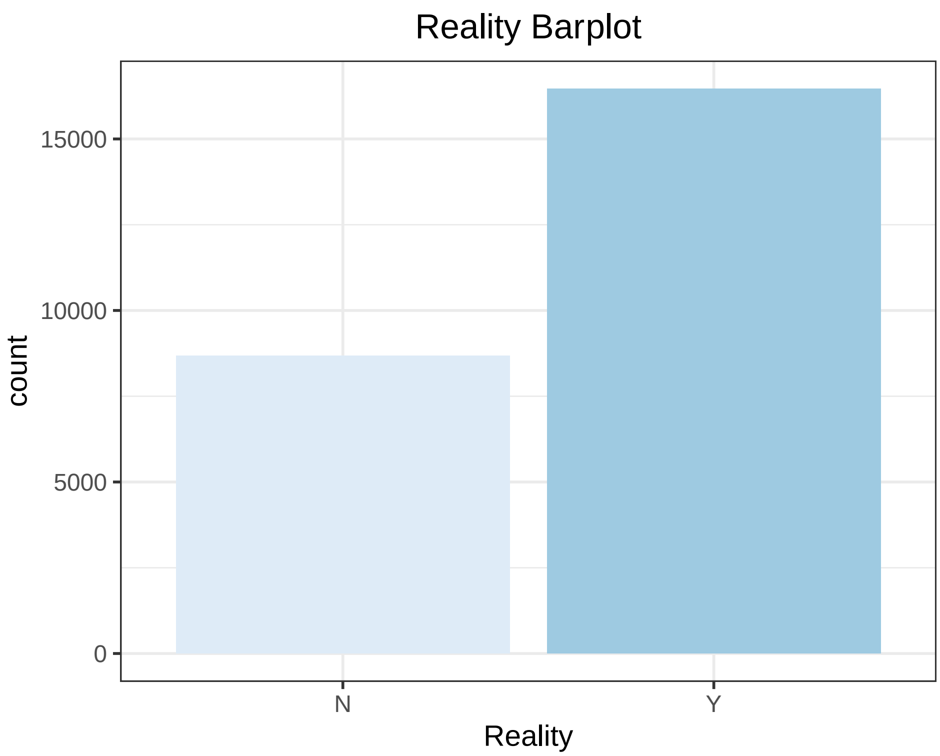
The value of p is the proportion of males in the dataset.

```
(p.male <- nrow(dat[GENDER == 'M',])/N)
```

```
## [1] 0.3781332
```

3.2 Feature 2 - Reality

The reality feature's distribution is shown in the following plot.



I define the feature `Reality` as follows:

$$\text{Reality} \sim \text{Bernoulli}(0.655)$$

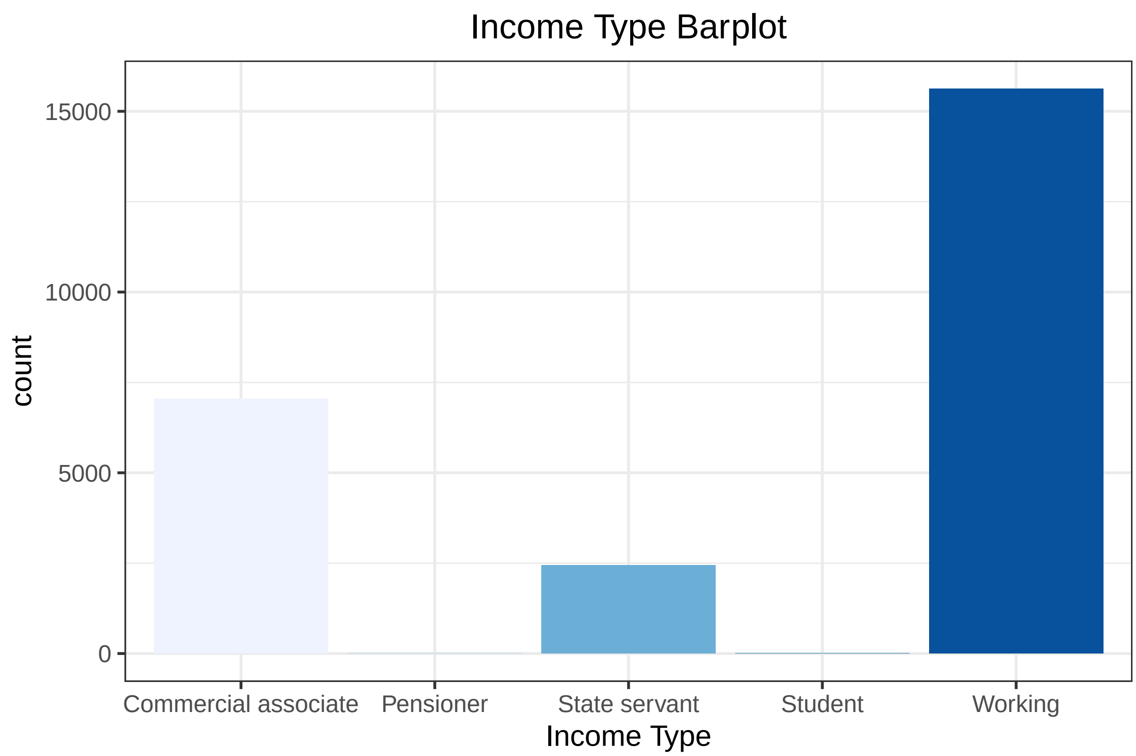
The value of p is the proportion of people who own property in the dataset.

```
(p.reality <- nrow(dat[REALITY == 'Y',])/N)
```

```
## [1] 0.6549296
```

3.3 Feature 3 - Income Type

The income type feature's distribution is shown in the following plot.



I define the feature `Income Type` as follows:

$$\text{Income Type} = \begin{cases} \text{Commercial associate} & 0.2806 \\ \text{Pensioner} & 0.0006 \\ \text{State servant} & 0.0969 \\ \text{Student} & 0.0004 \\ \text{Working} & 0.6215 \end{cases}$$

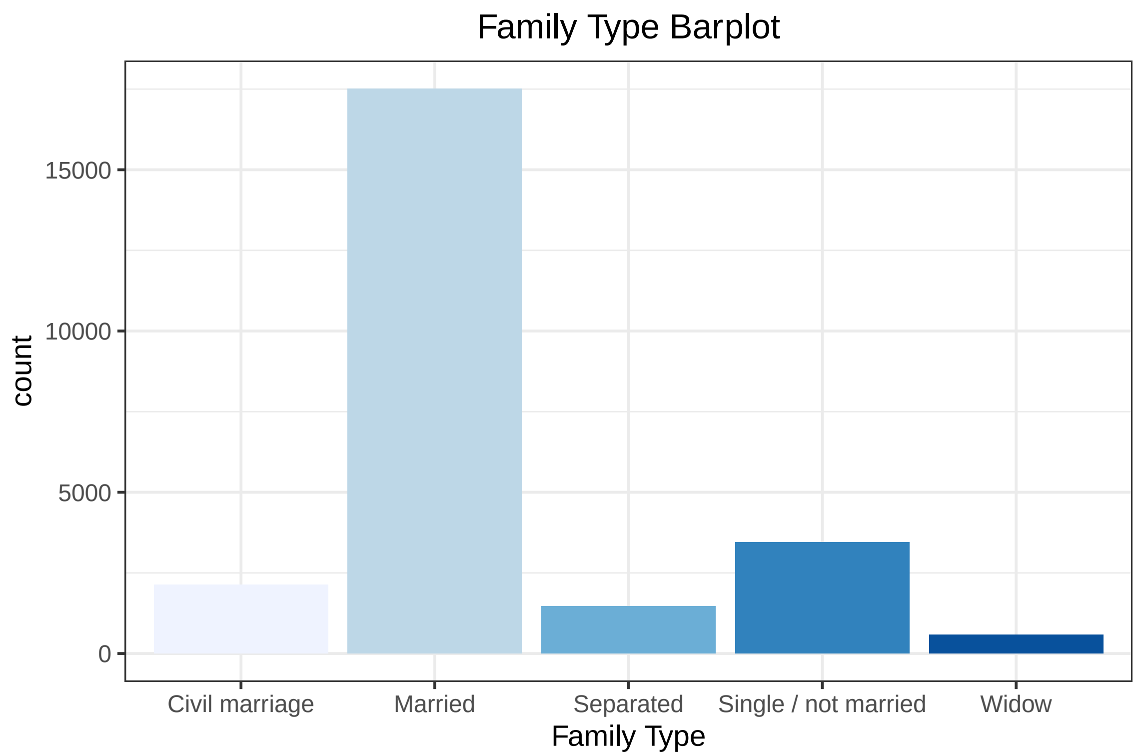
The value of p for each value is its corresponding proportion in the dataset.

```
(p.incomeType <- as.vector(table(dat$INCOME_TYPE)/N))
```

```
## [1] 0.2805761120 0.0005172277 0.0969602928 0.0003978674 0.6215485000
```

3.4 Feature 4 - Family Type

The family type feature's distribution is shown in the following plot.



I define the feature `Family Type` as follows:

$$\text{Family Type} = \begin{cases} \text{Civil marriage} & 0.0849 \\ \text{Married} & 0.6966 \\ \text{Separated} & 0.0584 \\ \text{Single / not married} & 0.1370 \\ \text{Widow} & 0.0231 \end{cases}$$

The value of p for each value is its corresponding proportion in the dataset.

```
(p.familyType <- as.vector(table(FAMILY_TYPE)/N))
```

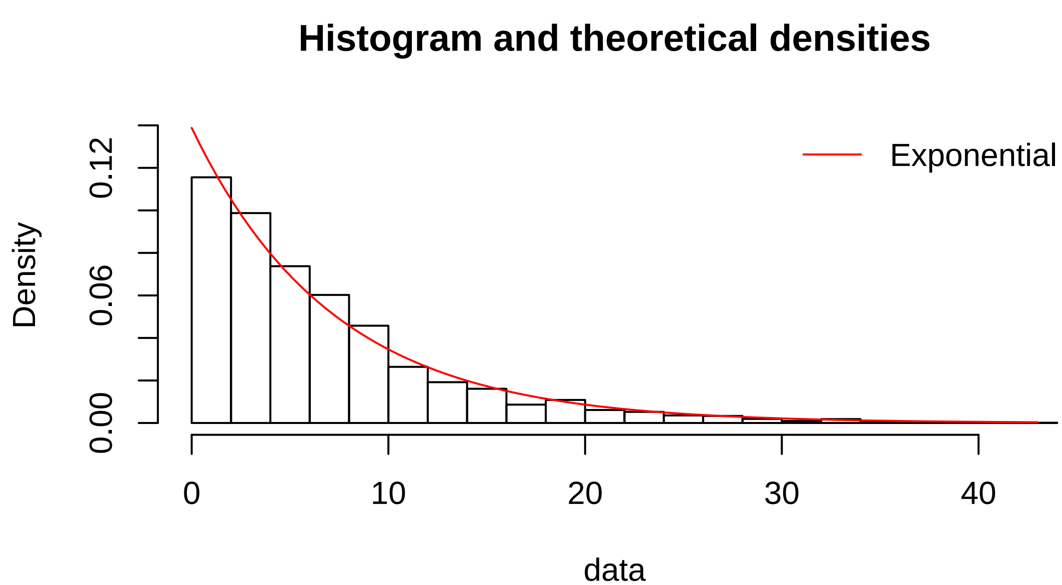
```
## [1] 0.08486512 0.69662608 0.05836715 0.13706533 0.02307631
```

3.5 Feature 5 - Years Employed

The years employed feature's distribution is shown in the following plot.

To fit the a suitable distribution for this feature, I used the `fitdist` method from the `MASS` package.

```
fe <- fitdist(YEARS_EMPLOYED, "exp")
```



Thus, I define the feature `Years Employed` as follows:

$$\text{Years Employed} \sim \text{Exp}(0.1388)$$

The distribution's parameter is the MLE estimate produced by the `fitdist` method.

```
(p.yearsEmployed <- as.vector(fe$estimate))
```

```
## [1] 0.1388097
```

4 Creating a Sample Dataset

To create a dataset from the model that I developed, I create four functions.

4.1 get.X

The function recieves the number of desired observations n and randomly samples n observations of each of the features, according to their distributions.

Note that the features are sampled independently, which may not be the most accurate procedure.

```
get.X <- function(n){

  x0 <- rep(1, n)
  gender <- rbernoulli(n, p.male)
  reality <- rbernoulli(n, p.reality)
  income.type <- sample(levels(INCOME_TYPE), n, replace = T, prob = p.incomeType)
  family.type <- sample(levels(FAMILY_TYPE), n, replace = T, prob = p.familyType)
  years.employed <- rexp(n, p.yearsEmployed)

  df <- cbind(x0, gender, reality, income.type, family.type, years.employed)
  return(as.data.frame(df))
}
```

4.2 get.dummies

The function recieves a dataset as input and creates dummy variables.

```
get.dummies <- function(df){

  # gender column
  df$gender_M <- ifelse(df$gender == T, 1, 0)

  # reality column
  df$reality_Y <- ifelse(df$reality == T, 1, 0)
```

```
  # income type columns
  df$income.type_Pensioner <- ifelse(df$income.type == 'Pensioner', 1, 0)
  df$income.type_StateServant <- ifelse(df$income.type == 'State servant', 1, 0)
  df$income.type_Student <- ifelse(df$income.type == 'Student', 1, 0)
  df$income.type_Working <- ifelse(df$income.type == 'Working', 1, 0)

  # family type columns
  df$family.type_Married <- ifelse(df$family.type == 'Married', 1, 0)
  df$family.type_Separated <- ifelse(df$family.type == 'Separated', 1, 0)
  df$family.type_Single <- ifelse(df$family.type == 'Single / not married', 1, 0)
  df$family.type_Widow <- ifelse(df$family.type == 'Widow', 1, 0)

  # rearange df
  df <- df %>% dplyr::select(-gender, -reality, -income.type, -family.type)
  df <- df[, c(1, 3:12, 2)]
  return(df)
}
```

4.3 sigmoid

The function recieves a vector of coefficients and an observation, and estimates $p(x)$ which is defined as:

$$p(x) = \frac{1}{1 + e^{-(\beta^T x)}}$$

```
sigmoid <- function(beta, X){
  z = sum(beta*X)
  return(1 / (1 + exp(-z)))
}
```

4.4 create.sample.df

The function uses the previous methods to create a sample dataset.

```
create.sample.df <- function(num.rows=10^4){

  # create random sample of size num.rows
  X <- get.X(num.rows)
  # get dummies for categorical variables
  df <- get.dummies(X)
  # create target column
  df$target <- NA
  # estimate target
  for(i in 1:nrow(df)){
    row <- as.numeric(df[i, 1:12])
    # calculate sigmoid
    p <- sigmoid(beta, row)
    # estimate target
```

```r
    df$target[i] <- sum(rbernoulli(1, p))
  }

  # remove x0 column
  df <- df[, -1]

  # create target=1 dataset and repeat rows
  target1 <- df[df$target == 1, ]
  target1 <- target1[rep(seq_len(nrow(target1)), each = 170), ]

  # add target rows to df
  df <- bind_rows(df, target1)

  # shuffle df rows
  rows <- sample(nrow(df))
  df <- df[rows, ]

  # change columns to numeric
  df$years.employed <- as.numeric(df$years.employed)

  # scale df
  df[,c(1:11)] <- lapply(df[,c(1:11)], function(x) (scale(x)))

  return(as.data.frame(df))
}
```

5 Logistic Model Accuracy

I create a dataset as described above and fit a logistic regression model.

```r
df <- create.sample.df()
mod <- glm(target ~ ., data = df, family = 'binomial')
summary(mod)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -4.144   0.000   0.000   0.000   0.174
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -3122.704   1990.904  -1.568   0.1168
## gender_M                   14.420   1070.270   0.013   0.9893
## reality_Y                 -10.972    691.883  -0.016   0.9873
## income.type_Pensioner       5.347    142.966   0.037   0.9702
## income.type_StateServant    1.764    676.008   0.003   0.9979
## income.type_Student        50.455   2966.744   0.017   0.9864
```

```
## income.type_Working        -15.110    844.749  -0.018   0.9857
## family.type_Married        -19.432    677.032  -0.029   0.9771
## family.type_Separated      -16.828   1644.301  -0.010   0.9918
## family.type_Single          -5.128    655.223  -0.008   0.9938
## family.type_Widow            3.707   1324.436   0.003   0.9978
## years.employed           -1845.136    771.272  -2.392   0.0167 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1736.361  on 10169  degrees of freedom
## Residual deviance:   29.096  on 10158  degrees of freedom
## AIC: 53.096
##
## Number of Fisher Scoring iterations: 25
```

Next, using CV, I estimate the accuracy of the model.

```
cv.error <- cv.glm(df, mod, K = 100)
```

Table 2: Model Accuracy

	Logistic Regression
Accuracy	0.9991

The model had almost perfect accuracy because it was able to perfectly predict the majority class (aka not fraud). However, that does not necessarily mean that it was able to correctly identify the minority class (aka fraud).

Thus, in part 2, I explore different sampling methods used to improve the performance of machine learning algorithms in identification of the minority class.