Synthetic Minority Oversampling Technique (SMOTE) – Bank Marketing

Goal

In the dataset on Amazon Alexa reviews, we observed that imbalance in the dataset reduced the models' accuracy to predict minority class (1 star ratings). In this exercise, we will use SMOTE to deal with an unbalanced dataset – to improve our models' accuracy in predicting minority class

Data Source: https://www.kaggle.com/henriqueyamahata/bank-marketing

Summary

We have used a dataset on the results of marketing calls for term deposit, made by a Portuguese bank. The data has a total of 41188 rows, of which only 4640 are 1; thus, our dataset is unbalanced. Initially, we created decision tree and logistic regression models and evaluated their performances (accuracy, sensitivity, and specificity), and then we compared these results with those of the models built on SMOTEd data.

We used SMOTE to create a balanced dataset: we oversampled (synthetically generated more data points) the minority class and under-sampled the majority class.

Function

SMOTE(y ~ ., data, perc.over = 100, perc.under = 200)

The number of minority class rows are decided as follows.

New no. of minority class rows = Original no. of minority class rows X [1 + (perc.over/100)]

The number of majority class rows are decided as follows.

New no. of majority class rows = [New no. of minority class rows - Original no. of minority class rows] X (perc.under/100)

We were able to see improvement in the sensitivities of the models, which is a key metric for us – as we want to predict 1s with accuracy; however, usage of SMOTE resulted in a slight decline in the specificities and thus in the overall accuracy.

We apply SMOTE function on the training data; thus, data we need to split the data before using SMOTE.

Variable Description

INPUT

age: Client age
job: Job Type
marital: Marital status
education: Education level
default: Whether the client has defaulted
housing: Whether the client has housing loan
loan: Whether the client has a personal loan
contact: Type of communication
month: Last month of the contact
day_of_week: Last contact day of the week
duration: Duration of last contact (seconds)
campaign: Number of times client contacted in the campaign
pdays: Number of days since the client was last contacted

previous: Number of times the client was contacted in earlier campaigns
poutcome: Outcome of the previous marketinig campaign
emp.var.rate: Employment variation rate
cons.price.idx: Consumer price index
cons.conf.idx: Consumer confidence index
euribor3m: Euribor 3 month rate
nr.employed: Number of employees

TARGET
y: Whether the client subscribed for a term deposit

Analysis

1. Check the datatypes of the variables

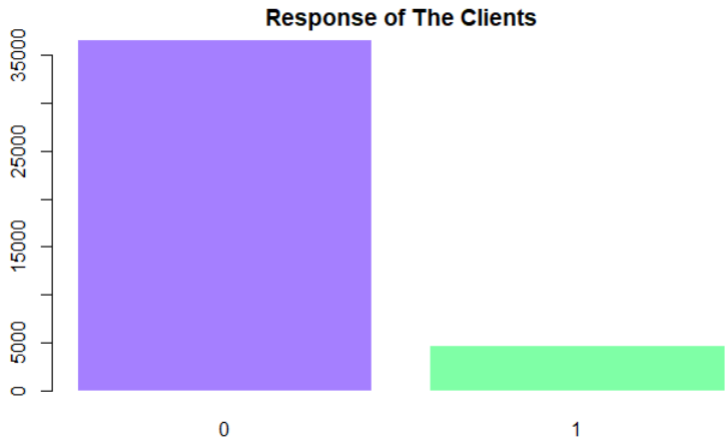
```
'data.frame':   41188 obs. of  21 variables:
 $ age            : int  44 53 28 39 55 30 37 39 36 27 ...
 $ job            : Factor w/ 12 levels "admin.","blue-collar",..: 2 10 5 8 6 5 2 2 1 2 ...
 $ marital        : Factor w/ 4 levels "divorced","married",..: 2 2 3 2 2 1 2 1 2 3 ...
 $ education      : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 8 7 4 1 1 1 3 7 1 ...
 $ default        : Factor w/ 3 levels "no","unknown",..: 2 1 1 1 1 1 1 1 1 1 ...
 $ housing        : Factor w/ 3 levels "no","unknown",..: 3 1 3 1 3 3 3 3 1 3 ...
 $ loan           : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ contact        : Factor w/ 2 levels "cellular","telephone": 1 1 1 1 1 1 1 1 1 1 ...
 $ month          : Factor w/ 10 levels "apr","aug","dec",..: 2 8 5 1 2 4 7 7 5 1 ...
 $ day_of_week    : Factor w/ 5 levels "fri","mon","thu",..: 3 1 3 1 1 4 3 1 2 3 ...
 $ duration       : int  210 138 339 185 137 68 204 191 174 191 ...
 $ campaign       : int  1 1 3 2 1 8 1 1 1 2 ...
 $ pdays          : int  999 999 6 999 3 999 999 999 3 999 ...
 $ previous       : int  0 0 2 0 1 0 0 0 1 1 ...
 $ poutcome       : Factor w/ 3 levels "failure","nonexistent",..: 2 2 3 2 3 2 2 2 3 1 ...
 $ emp_var_rate   : num  1.4 -0.1 -1.7 -1.8 -2.9 1.4 -1.8 -1.8 -2.9 -1.8 ...
 $ cons_price_idx: num  93.4 93.2 94.1 93.1 92.2 ...
 $ cons_conf_idx : num  -36.1 -42 -39.8 -47.1 -31.4 -42.7 -46.2 -46.2 -40.8 -47.1 ...
 $ euribor3m      : num  4.963 4.021 0.729 1.405 0.869 ...
 $ nr_employed    : num  5228 5196 4992 5099 5076 ...
 $ y              : int  0 0 1 0 1 0 0 0 1 0 ...
```

2. We looked for the missing values, but did not have any

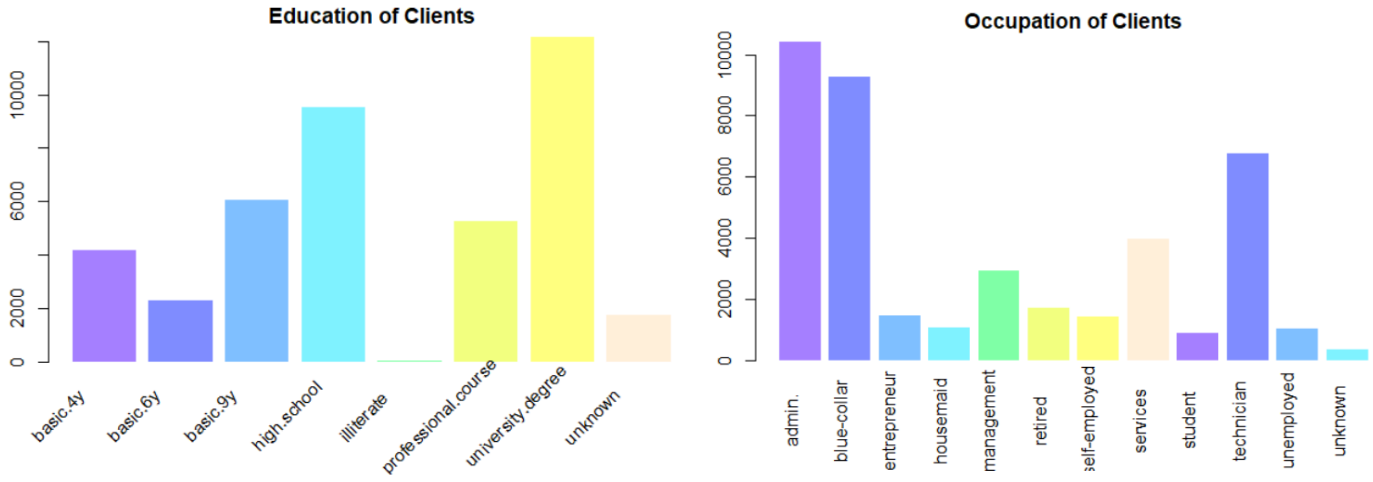
```
        age         job     marital   education     default     housing        loan     contact       month 
          0           0           0           0           0           0           0           0           0 
day_of_week    duration    campaign       pdays    previous    poutcome emp_var_rate cons_price_idx cons_conf_idx 
          0           0           0           0           0           0           0           0           0 
  euribor3m nr_employed           y 
          0           0           0 
```

3. ‘pdays’ was a numerical variable, and 999 in this column indicated that the client was never contacted. We converted this variable into a numerical one and changed 999 to ‘Not_contacted’.

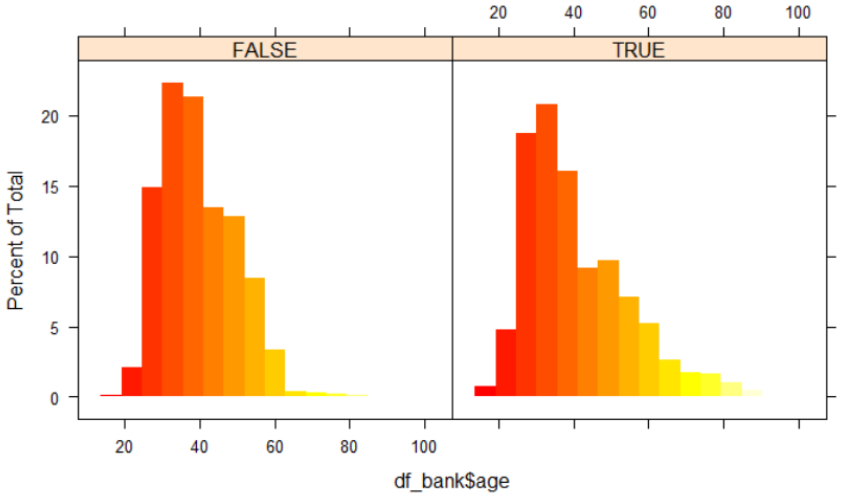
4. Only about 11.2% of the clients had responded positively to the calls.



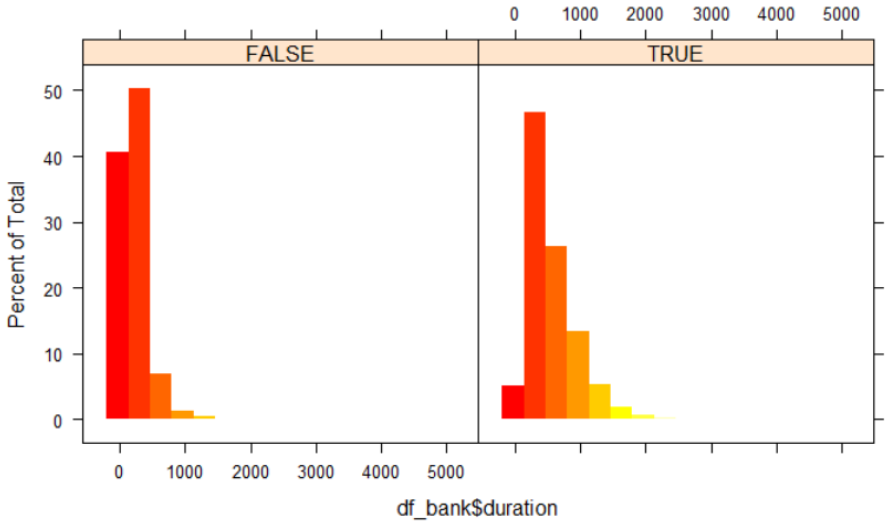
5. Then we looked into the education levels and the occupation of the clients.



6. The age range of people who responded negatively is comparatively more narrow; however, the difference appeared to be marginal.

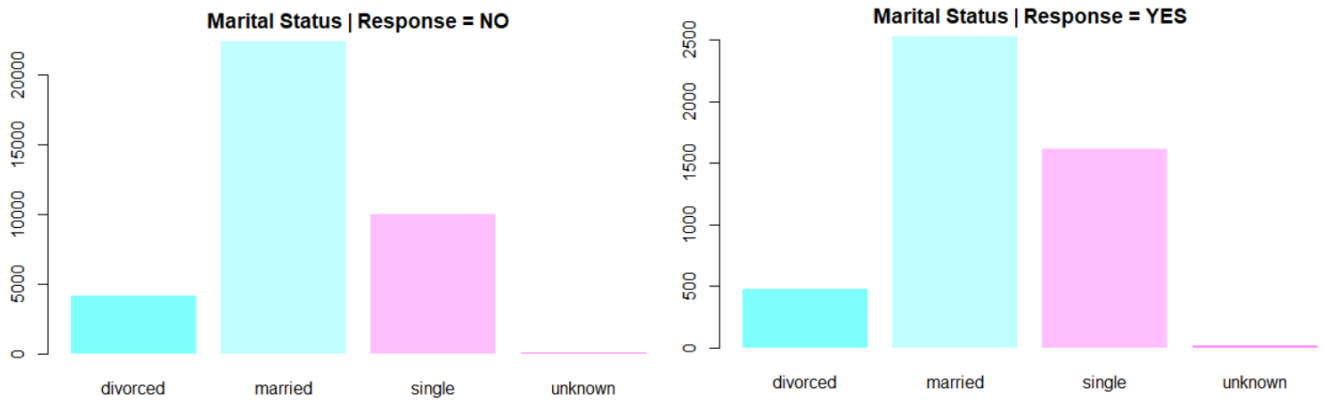


7. People responding positively tend to stay on the call for comparatively longer duration

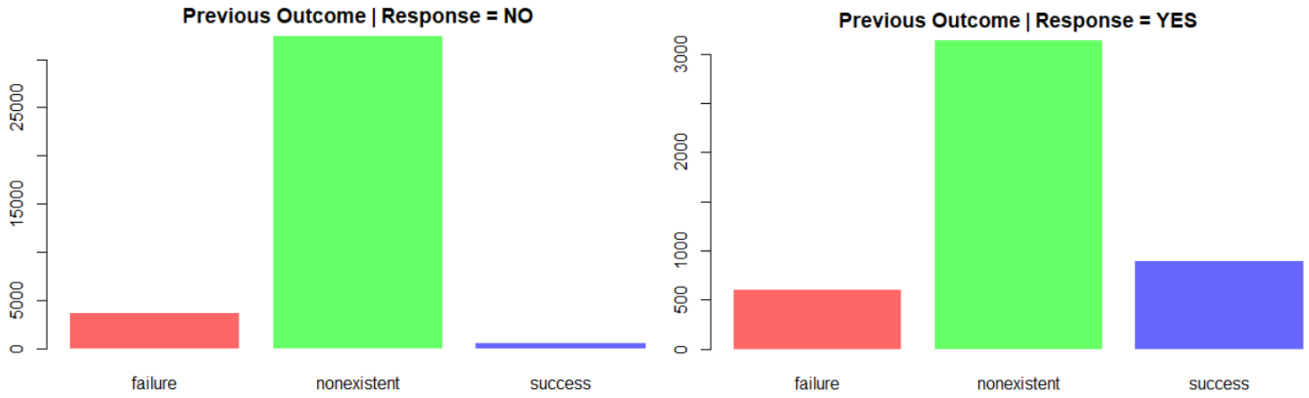


Because duration of a call is an aspect that cannot be decided before calling customers, we excluded this variable from the model; nonetheless, it might be interesting for the bank to look into whether longer duration has key elements that could be used for the training of sales teams.

8. Clients responding positively had greater proportion of single people



9. More proportion of the people who said yes in the previous campaign responded positively to this campaign



Machine Learning Models & Results

1. We created Logistic Regression and Decision Tree models on the unbalanced training data and calculated their accuracies, sensitivities, and specificities.

2. On the training data, we then used SMOTE to over-sample (artificially create) the minority class (1) and under-sample the majority class (0).

```r
# We will use SMOTE to make the dataset balanced
# AS the SMOTE function needs the target variable in factor format, we will convert 'y' into a factor

df_train_smote <- df_train

df_train_smote$y = as.factor(df_train_smote$y)

df_train_smote <- SMOTE(y~.,df_train_smote,perc.over = 100,perc.under = 200)
table(df_train_smote$y)
```

```
   0    1
6534 6534
```

3. We created the Logistic Regression and Decision Tree models on SMOTEd training data (balanced), and then again compared the results with the models created earlier on the unbalanced data. **The results are from the test data.**

Logistic Regression

Sensitivity improved by about 36 percentage points

Model on unbalanced training data

```
[1] "Accuracy: 90.22%"
[1] "Sensitivity: 23.45%"
[1] "Specificity: 98.56%"
[1] "Precision: 67.08%"
```

Model on balanced training data (SMOTEd)

```
[1] "Accuracy: 84.34%"
[1] "Sensitivity: 59.94%"
[1] "Specificity: 87.39%"
[1] "Precision: 37.27%"
```

Decision Tree

Sensitivity improved by about 29 percentage points

Model on unbalanced training data

```
[1] "Accuracy: 90.3%"
[1] "Sensitivity: 27.46%"
[1] "Specificity: 98.15%"
[1] "Precision: 65%"
```

Model on balanced training data (SMOTEd)

```
[1] "Accuracy: 87.12%"
[1] "Sensitivity: 56.96%"
[1] "Specificity: 90.9%"
[1] "Precision: 43.88%"
```