Subscribe Or Not Subscribe, That's The Question

Yizhou Zhang

January 20, 2018

Abstract

I used the Bank Marketing Data Set from UC Irvine Machine Learning Repository during May 2008 to November 2010 to develop a supervised learning model of term deposit subscription, in the hope of helping the bank's future campaigns. Models such as logistic regression, random forest, and stochastic gradient boosting were considered. A random forest model is found to be most accurate with the highest prediction accuracy.

Data

```
##UPLOAD DATA##
bank.big= read.table("data/bank-additional-full.csv", sep = ";", header = TRUE)
```

Summary

I used the full sample with additional information dataset, with 41188 observations and 20 features. The time span was from May 2008 to November 2010. The features include:

Variable Name	Description
age	age of customer
job	type of job
marital	marital status
education	educational attainment
default	has credit in default?
housing	has housing loan?
loan	has personal loan?
contact	contact communication type
month	last contact month of year
day_of_week	last contact day of the week
campaign	number of contacts performed during this campaign and for this client
pdays	number of days that passed by after the client was last contacted from a previous campaign
previous	number of contacts performed before this campaign and for this client
poutcome	outcome of the previous marketing campaign
emp.var.rate	employment variation rate - quarterly indicator
cons.price.idx	consumer price index - monthly indicator
cons.conf.idx	consumer confidence index - monthly indicator
euribor3m	euribor 3 month rate - daily indicator
nr.employed	number of employees - quarterly indicator

Drop Unwanted Variable

I drop the variable `duration` , as suggested in the data documentation.

```
#drop the variable duration, as suggested in the documentation
bank = bank.big[, !names(bank.big) %in% c("duration")]
remove(bank.big)
```

Dealing with Topcoded Variables

The variable `pdays` is topcoded. A `999` indicates that the customer has never been contacted, while a `0` means the customer has been contacted in the same day. Therefore, I recode `999` as zero, and `0` as 0.5, such that previous call would be more accurately described.

```
#first find out how many entries are topcoded
length(which(bank$pdays == 999))
```

```
## [1] 39673
```

```
#first replace all  entries with zero pday as 0.5,
#indicating that it was same day  re-contact

bank$pdays[bank$pdays == 0] = 0.5

#then replace all  entries with topcode as 0,
#indicating that the customer was never contacted before

bank$pdays[bank$pdays == 999] = 0
```

Factorization

I Factorize all categorical variables, or columns 2-7, in the dataset.They are job, marital, education, default, housing, and loan.

```
for (i in 2:7) {
bank[, i] = as.factor(bank[, i])

}
```

Unknown Features & Missing Values

First I calculate the total number of "unknown" in each factor predictor. They respectively have the following number of entries with unknown values:

```
#first to know how many unknowns in each column
del = rep(0, 6)
for (i in 1:6) {
del[i] = length(which(bank[, i + 1] == "unknown"))

}

uknow = data.frame(var = names(bank)[2:7], count = del)

uknow
```

```
##         var count
## 1       job   330
## 2   marital    80
## 3 education  1731
```

```
## 4   default  8597
## 5   housing   990
## 6      loan   990
```

Reclassify unknown default status

The results above show that the variable education has the most number of unknown entries. Instead of just dropping all observations where `default == unknown`, I replace the "unknown" with a new value "mysterious" for two reasons. First, there are too many observations with unknown default status; Second, a simple OLS with `y~ all` shows that `default.unknown` is a significant predictor. In other words, the customers that did not report credit default status might be systematically different than those that report, and this difference puts an impact on new term subscription. Therefore, I reclassify unknown default status as "mysterious" to maintain this potential effect.

```
levels(bank$default)[2] = "mysterious"
```

Omit other unknowns

I reclassify all unknowns in other features with NA, and omit those observations. After removing observations with unknown features, there are 41188 observations, and there are 19 variables.

```
bank[bank == "unknown"] = NA

bank = na.omit(bank)
```

Train-test Split

```
set.seed(12345)

##Test/Train Split##
split_idx = createDataPartition(y = bank$y, p = .8)
trn_data = bank[split_idx$Resample1,]
tst_data = bank[-split_idx$Resample1,]
```

Use the `createDataPartition` function to create a 80-20 split. The training set has 30597 observations, while the test set has 7648.

Calculate Cook Distance and drop potential outliers

The code below calculates the cook distance in a glm model. I dispose of the 50 points with the biggest cook distance, as they are the most likely outliers.

```
a = glm(y ~ ., family = binomial, data = trn_data)

cooksd = cooks.distance(a)

sorted = sort(cooksd, decreasing = TRUE)

outlier50 = as.numeric(names(sorted)[1:50])


trn_data = trn_data[-outlier50, ]
```

Models

Generalized Linear Model

```
set.seed(12345)

fit_glm = train(y ~ .,
data = trn_data,
method = "glm",
preProcess = c("range"))

save(fit_glm, file = "./models/glm.rda")
```

First I fit an additive logistic regression model, as the baseline model. The training data is standardized meaning that they are transformed to have zero mean and unit variance.

The Random Forest Model

```
rf_grid = expand.grid(mtry = c(4:5))
```

```
set.seed(12345)


fit_rf = train(
y ~ .,
data = trn_data,
method = "rf",
ntree = 500,
preProcess = c("center", "scale"),
tuneGrid = rf_grid,
trControl = trainControl(method = "oob")
)

save(fit_rf, file = "./models/rf.rda")
```

I choose to run my random forest with ntree = 500 and with mtry values at 4 or 5. I tuned the random forest using out-of-bagging instead of cross-validation, because this greatly saves computational time. Meanwhile, mtry = 4 or 5 because the rule-of-thumb is to set mtry equal the square root of the total number of predictors. Since there are 19 predictors, the mtry could be either 4 or 5. The final model chooses four over five.

Stochastic Gradient Boosting

For GBM, I used 5-fold cross-validation to tune over the following tuning grid:

```
cv_5 = trainControl(method = "cv", number = 5)
gbm_grid = expand.grid(
interaction.depth = c(1, 2, 3),
n.trees = (1:30) * 100,
shrinkage = c(0.1, 0.3),
n.minobsinnode = 20
)
```

These parameters are chosen by rule-of-thumb. For example, it is common to consider interaction depth at one or two, representing very small trees with only two leaves. This is actually the core advantage of gbm as a gradual learning process through many small trees. However, unlike random forest, `gbm` could overfit the model if there are two many trees. To avoid this, `ntree` is chosen through cross-validation over 100 to 3000 trees.

```
set.seed(12345)

fit_gbm = train(
y ~ .,
trControl = cv_5,
data = trn_data,
method =  "gbm",
verbose = FALSE,
tuneGrid = gbm_grid
)

save(fit_gbm, file = "./models/gbm.rda")
```

Results

Model Comparison

Table 2 compares the prediction accuracy of three models. The results show that a random forest model performs the best.

```
calc_acc = function(actual, predicted) {
  mean(actual == predicted)

}


model_all = list(fit_gbm, fit_glm, fit_rf)

models = c("Stochastic Gradient Boosting",
"Generalized Linear Model",
" Random Forest")

Accuracy = rep(NA, length(model_all))

for (i in seq_along(model_all)) {
pred = predict(model_all[[i]], newdata = tst_data)

Accuracy[i] = calc_acc(predicted = pred, actual = tst_data$y)
}

Accuracy = round(Accuracy, 4)

results = data.frame(Models = models,
coefficient = Accuracy)


knitr::kable(
```

Table 2: These are the Prediction Accuracy for the Three Chosen Models

Models	coefficient
Stochastic Gradient Boosting	0.8991
Generalized Linear Model	0.9018
Random Forest	0.9022

Table 3: This table shows the Coefficients of the Most Important Variables

	Coefficient
euribor3m	0.653
nr.employed	2.004
age	-0.018
pdays	0.880
cons.conf.idx	0.805
cons.price.idx	5.510
emp.var.rate	-7.080
poutcomesuccess	1.586
campaign	-1.757
previous	-0.157

```
results,
format = "latex",
col.names = gsub("_", " ", names(results)),
caption = "These are the Prediction Accuracy for the Three Chosen Models",
booktabs = TRUE
) %>%
kable_styling(latex_options = c("striped"))
```

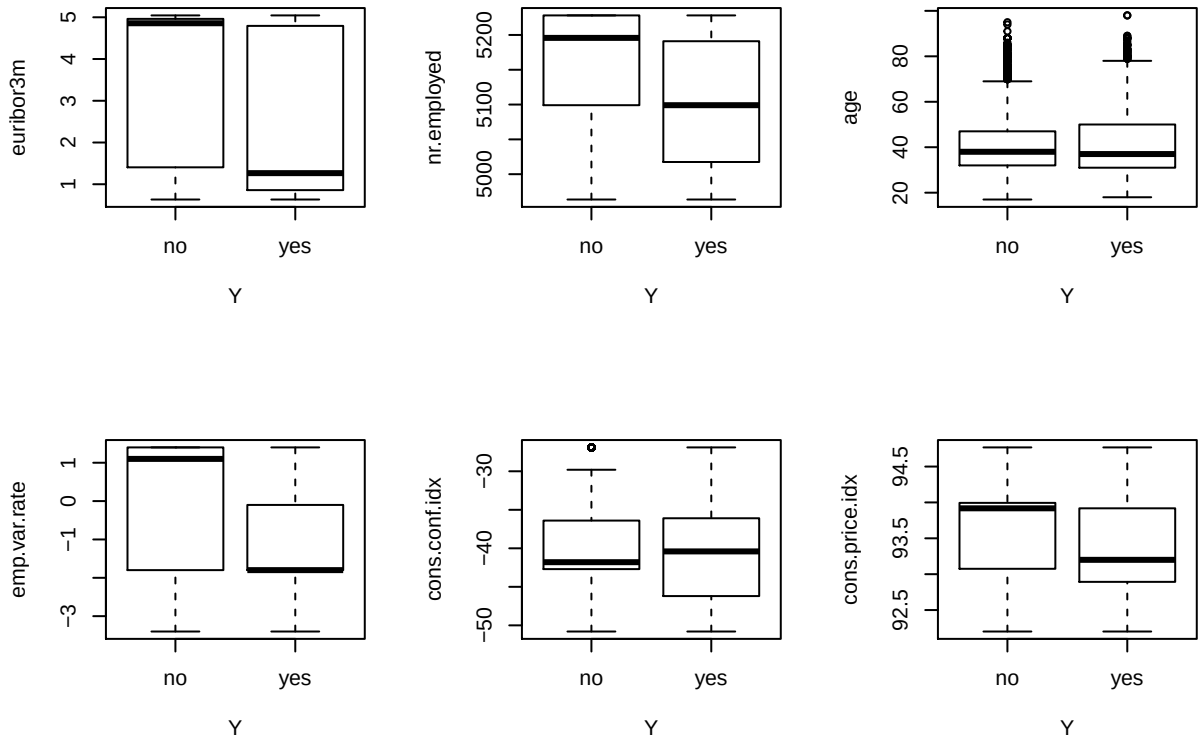
Variable Importance

The `importance` function determines the top 10 most important variables, in the sense that they explain the largest shares of the variance in the original dataset. Table 3 further shows the magnitudes and the direction of the effects of these ten variables.

```
imp = names(importance(fit_rf$finalModel)[order(importance(fit_rf$finalModel),
decreasing = TRUE),][1:10])

top_coef = fit_glm$finalModel$coefficients[imp]

coef = data.frame(Coefficient = round(top_coef, 3))

##Print Top 10##
knitr::kable(
  coef,
  format = "latex",
  col.names = gsub("_", " ", names(coef)),
  caption = "This table shows the Coefficients of the Most Important Variables",
  booktabs = TRUE)
```

Graphical Analysis

```
names = c(
  "euribor3m",
  "nr.employed",
  "age",
  "emp.var.rate",
  "cons.conf.idx",
  "cons.price.idx"
  )

  par(mfrow = c(2, 3))
  for (i in seq_along(names)) {
  plot(trn_data$y, trn_data[, names[i]], xlab = "Y", ylab = names[i])


  }
```



In all three classification models, the positive class is "no". A positive coefficient contributes to the probability of positive class, while a negative coefficient contributes to a negative class ("yes"). The above plots further confirm the findings in Table 3. For example, `euribor3m` is obviously higher for nonsubscribers.

The variable `emp.var.rate` deserves extra attention. It has negative coefficient indicating that higher employment variance rate should help subscription. However, this does not seem to be the case in the plot. The reason is because `emp.var.rate` is highly correlated with `nr.employed` and `euribor3m` with respective correlation at 0.91 and 0.97. Thus, the plot of `emp.var.rate` is not accurate, as the effect is confounded by

collinearity. In contrast, the negative coefficient is the true marginal effect after holding every other predictor fixed.

Discussion

Indicators of the macro-economic environment are the most important predictors of outcome. Specifically, the greater the interest rate `euribor3m`, the less likely of a term subscription; The lower the consumer price index `cons.price.idx`, the more likely of subscription. If the labor market is underperforming with small number of employees `nr.employed`, the more likely that a customer would subscribe. On the contrary, greater uncertainty in the job market, represented by `emp.var.rate` ,would lead to more subscriptions. Greater consumer confidence, `cons.conf.idx` leads to lower subscription probability. To summarize, term deposit seems more attractive to customers under bad macroeconomy with a low interest rate, low inflation, stagnant and risky job market, and low consumer confidence.

Table 3 also shows that the results of past campaigns are important predictors of the current campaign. Specifically, a successful previous campaign would not help the current campaign. A longer gap between the previous and current campaign,`pdays` , would not help either. However, a customer is more likely to subscribe if the previous campaigner contacted multiple times.

Regarding personal characteristics, `age` has a samll, significantly positive effect: the likelihood of subscription increases with age. However, as `age` has high importance in the tree model it should also have big coefficients in the linear regression model with standardized predictors. This discrepancy might due to a possible nonlinear effect of age (e.g polynomial form, exponential), which should be investigated in future studies.

Conclusion

This study analyzes a full sample of customer term deposits between May 2008 and Nov 2010. Three classification methods were performed, and a random forest model was found with the highest predictive power. The most important predictors are macro-economic indicators, while the outcomes and characteristics of previous campaigns also matter.