CIND119 - Group Project

10/04/2021

## Memebers

Patrick Litte, [patrick.little@ryerson.ca](mailto:patrick.little@ryerson.ca) Manjola Chiappetta, [m1chiappetta@ryerson.ca](mailto:m1chiappetta@ryerson.ca)

## Summary

This section is a summary of the project.

instructions: Write an abstract (a kind of a summary) to describe your project. The abstract must be within 175 to 250 words (inclusive). To write the abstract, first state the problem you are addressing. For example, if your project is on Churn analysis, then give a brief explanation of it. Second, write the summary of your classification results (e.g., accuracy). Third, state key points about the post-predictive analysis and fourth, summarize your recommendations to the organization.

## Workload Distribution

|  |  |
| --- | --- |
| Member Name | List of Tasks Preformed |
| Patrick Little | - some tasks |
| Manjola Chiappetta | - some tasks |

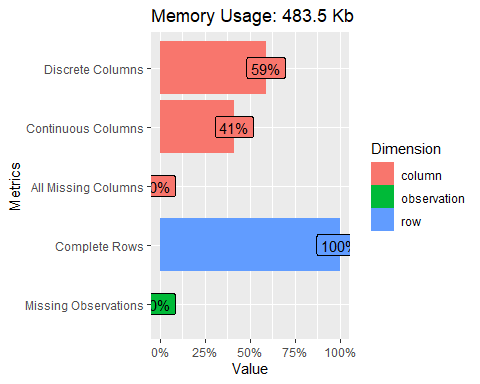
## Exploratory Data Analysis

In this section we will: - Look at the attribute types in the dataset - Find and missing values - Find max,min,mean and standard deviation of the atttributes - Determine any outlier values for the attributes under consideration - Analyze the distribution of numeric attributes

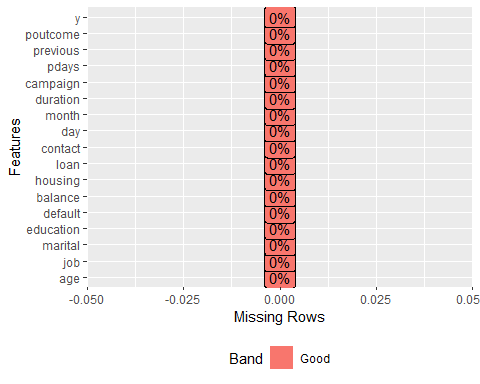
bank<-read.csv("https://raw.githubusercontent.com/PatLittle/CIND119-group-project/main/bank\_marketing/bank.csv")  
  
introduce(bank)

## rows columns discrete\_columns continuous\_columns all\_missing\_columns  
## 1 4521 17 10 7 0  
## total\_missing\_values complete\_rows total\_observations memory\_usage  
## 1 0 4521 76857 495152

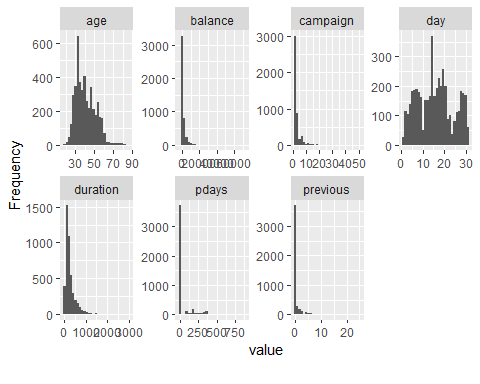
plot\_intro(bank)



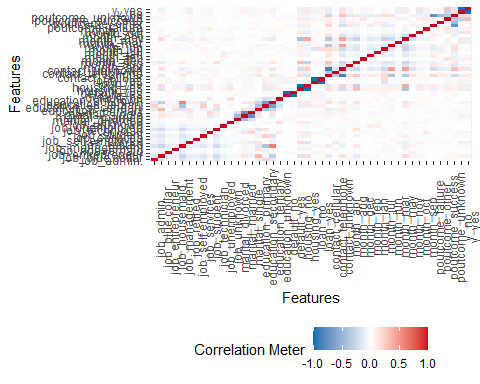
plot\_missing(bank)



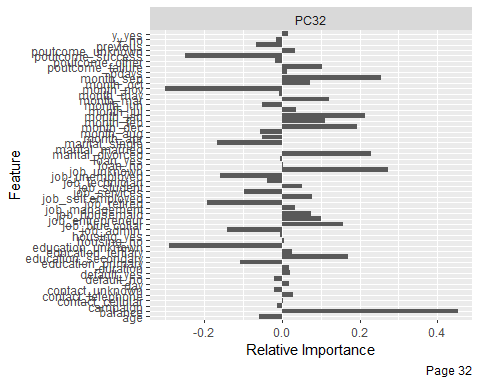
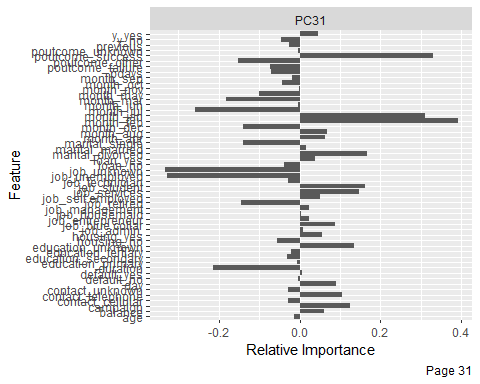
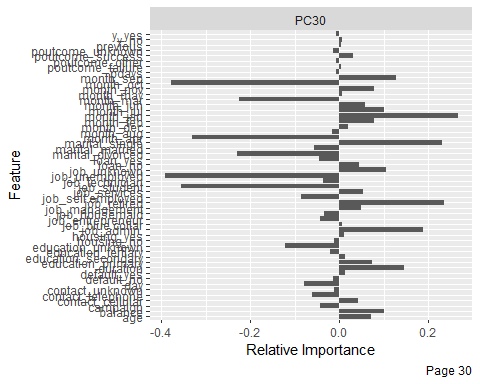
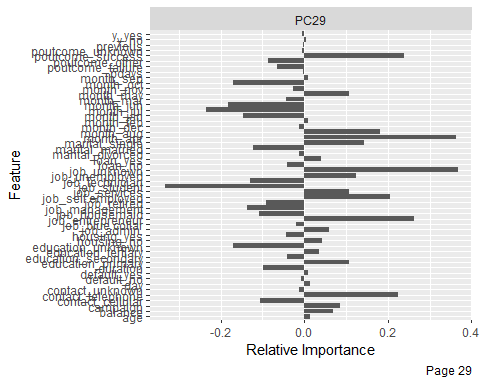
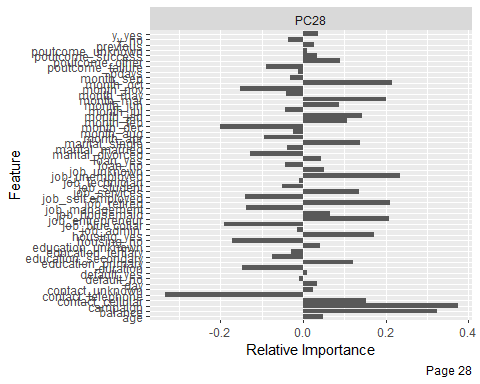
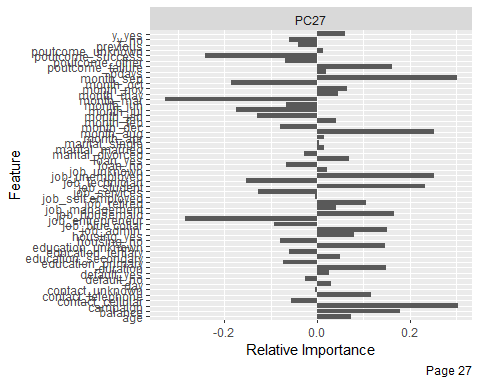
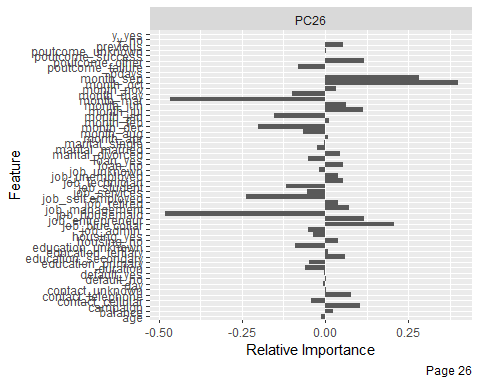
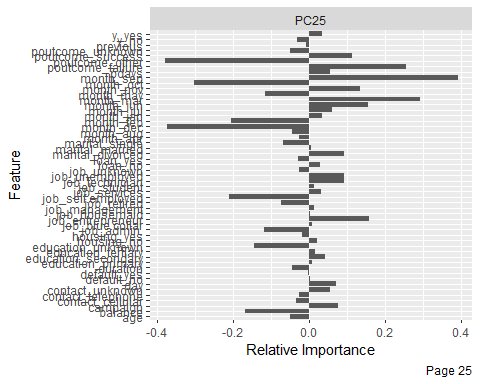
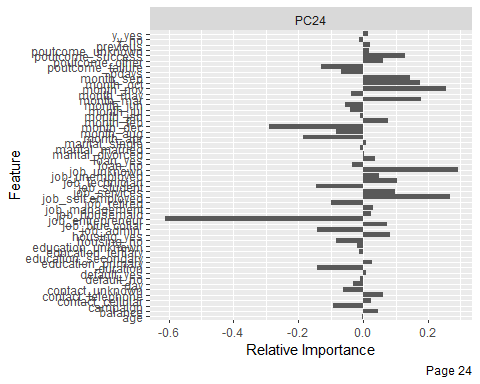
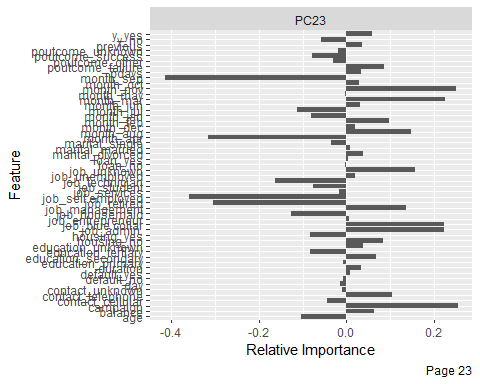
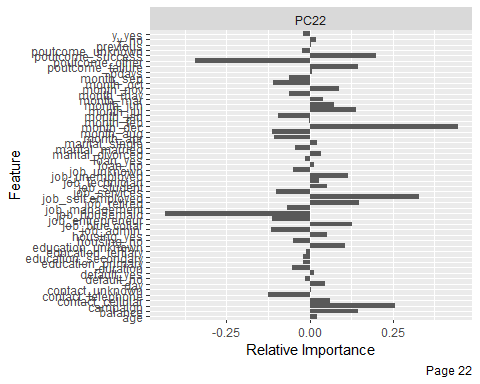
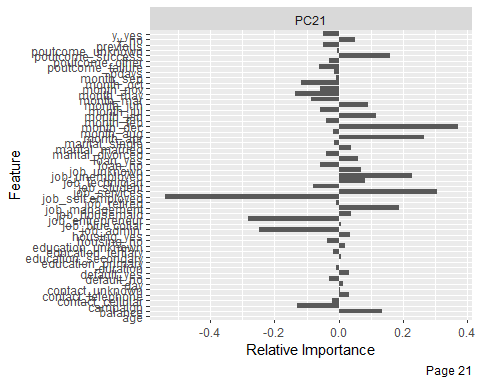
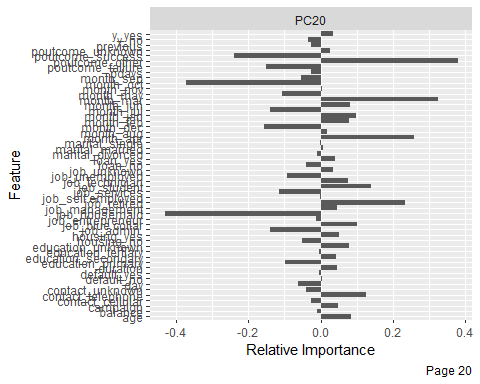
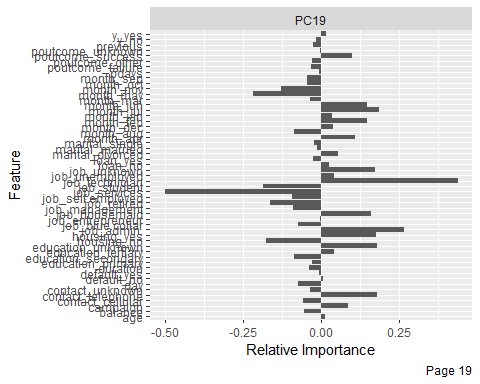
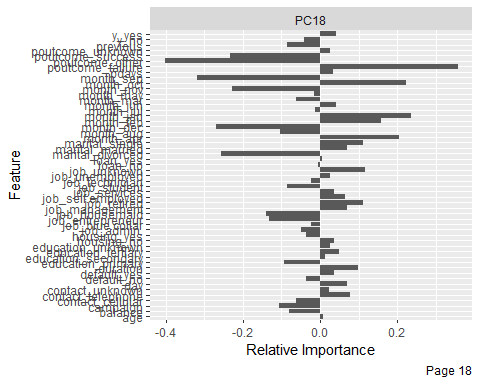
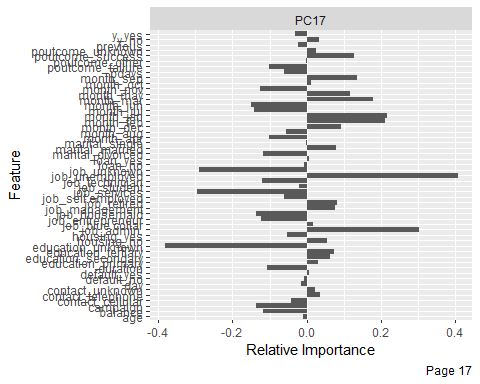
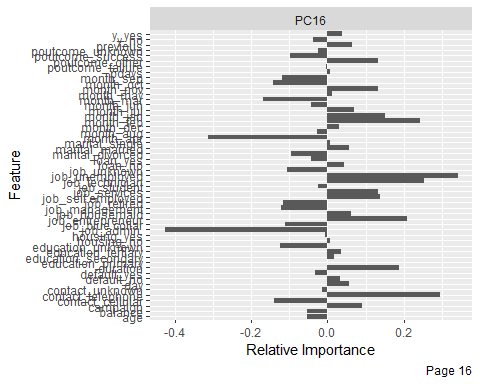
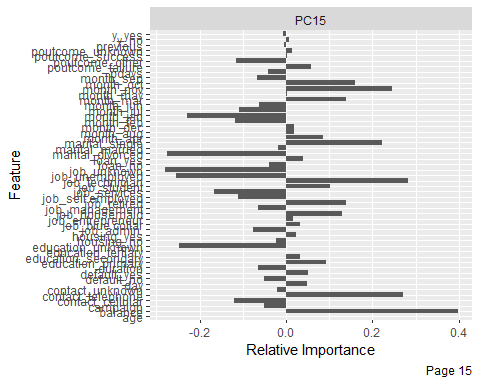
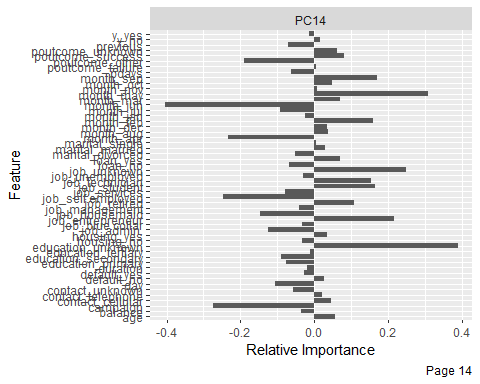
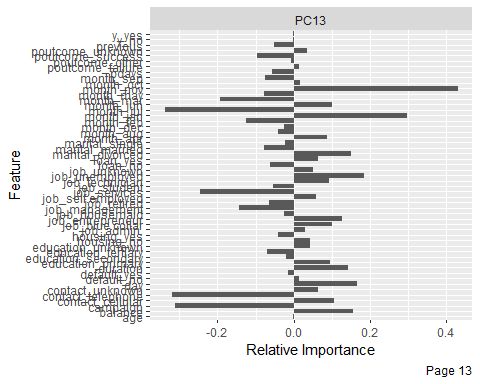
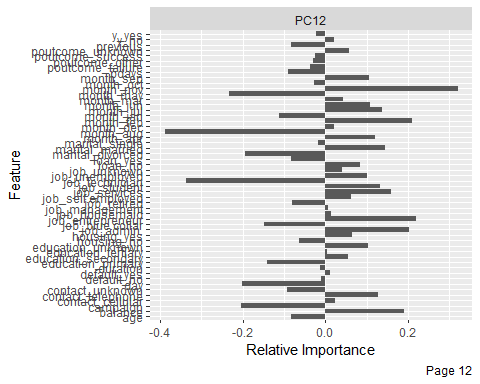
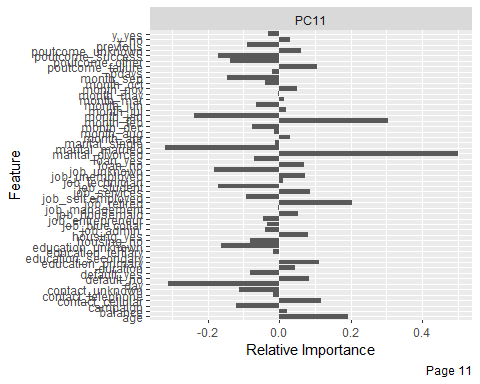
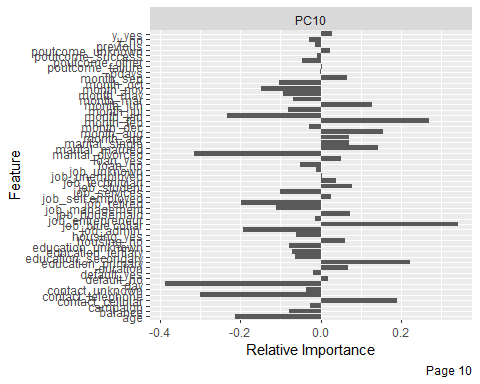
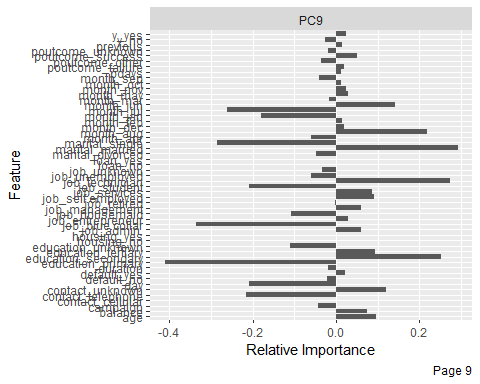
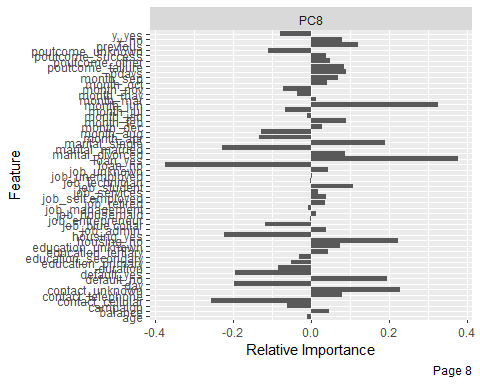
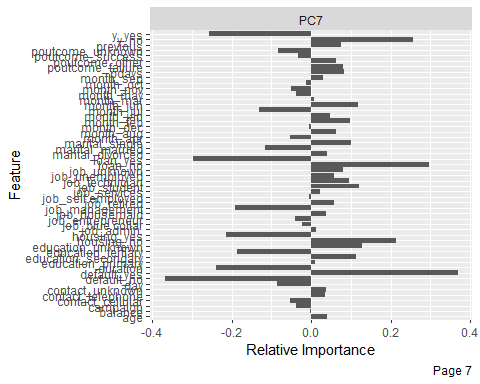
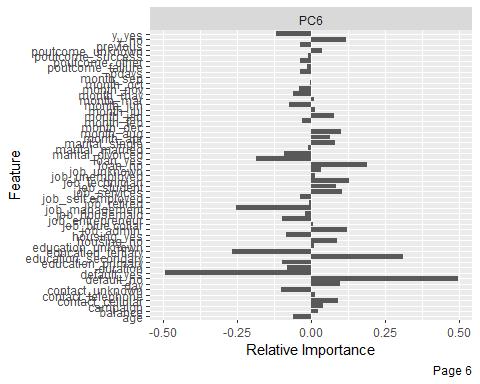
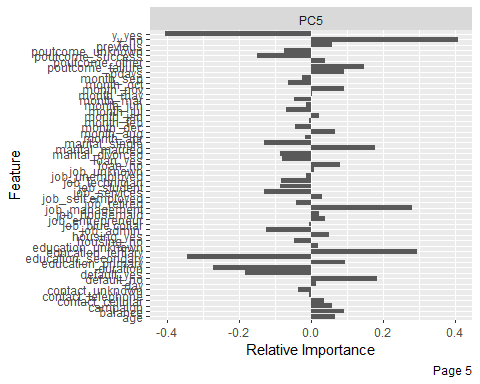
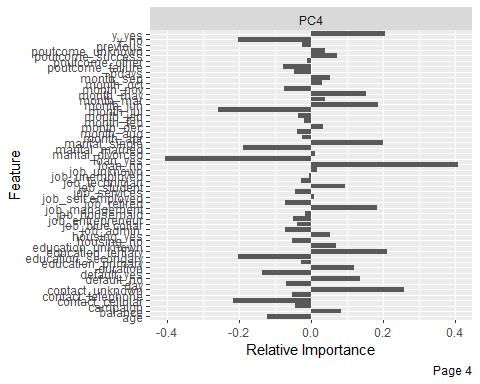
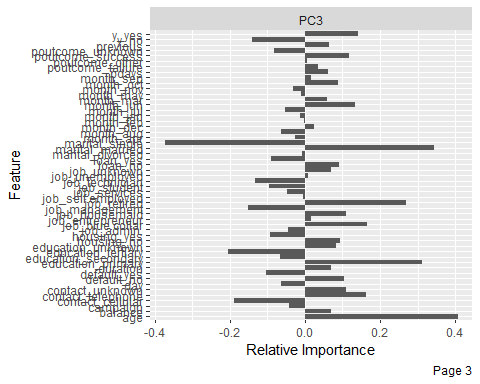
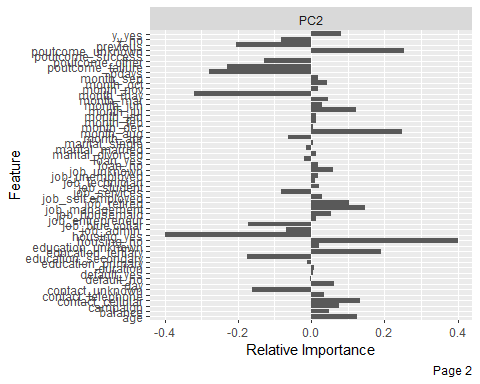
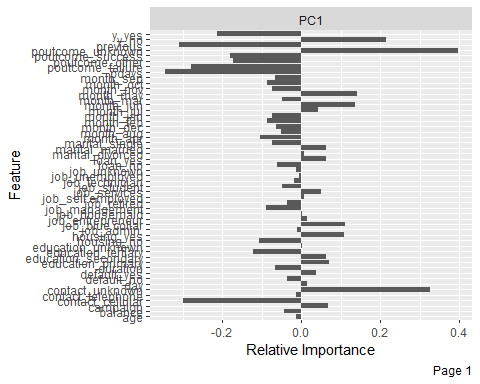
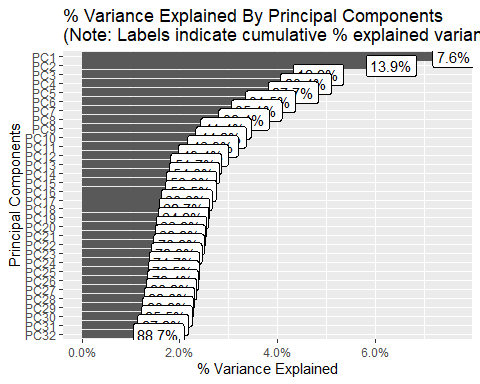
#plot\_bar(bank, by = "y")  
plot\_histogram(bank)



plot\_correlation(na.omit(bank), type = "d")



plot\_prcomp(bank, variance\_cap = 0.9, ncol =1L, nrow=1L)



str(bank)

## 'data.frame': 4521 obs. of 17 variables:  
## $ age : int 30 33 35 30 59 35 36 39 41 43 ...  
## $ job : chr "unemployed" "services" "management" "management" ...  
## $ marital : chr "married" "married" "single" "married" ...  
## $ education: chr "primary" "secondary" "tertiary" "tertiary" ...  
## $ default : chr "no" "no" "no" "no" ...  
## $ balance : int 1787 4789 1350 1476 0 747 307 147 221 -88 ...  
## $ housing : chr "no" "yes" "yes" "yes" ...  
## $ loan : chr "no" "yes" "no" "yes" ...  
## $ contact : chr "cellular" "cellular" "cellular" "unknown" ...  
## $ day : int 19 11 16 3 5 23 14 6 14 17 ...  
## $ month : chr "oct" "may" "apr" "jun" ...  
## $ duration : int 79 220 185 199 226 141 341 151 57 313 ...  
## $ campaign : int 1 1 1 4 1 2 1 2 2 1 ...  
## $ pdays : int -1 339 330 -1 -1 176 330 -1 -1 147 ...  
## $ previous : int 0 4 1 0 0 3 2 0 0 2 ...  
## $ poutcome : chr "unknown" "failure" "failure" "unknown" ...  
## $ y : chr "no" "no" "no" "no" ...

colSums(is.na(bank))

## age job marital education default balance housing loan   
## 0 0 0 0 0 0 0 0   
## contact day month duration campaign pdays previous poutcome   
## 0 0 0 0 0 0 0 0   
## y   
## 0

## Predictive Modeling / Classification

### Decision Tree

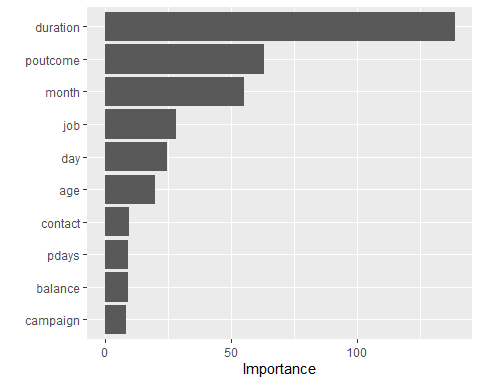
###Decision Tree  
  
  
  
bank\_clean<- bank %>% mutate\_if(is.character, factor)  
  
set.seed(888)  
bank\_split <- initial\_split(bank\_clean, prop = 0.75,   
 strata = y)  
  
bank\_training <- bank\_split %>% training()  
bank\_test <- bank\_split %>% testing()  
bank\_folds <- vfold\_cv(bank\_training, v = 10)  
  
  
  
bank\_recipe <- recipe(y ~ ., data = bank\_training)   
   
  
  
bank\_clean\_baked<-bank\_recipe %>%   
 prep() %>%   
 bake(new\_data = bank\_training)  
  
tree\_model <- decision\_tree(cost\_complexity = tune(),  
 tree\_depth = tune(),  
 min\_n = tune()) %>%   
 set\_engine('rpart') %>%   
 set\_mode('classification')  
  
tree\_workflow <- workflow() %>%   
 add\_model(tree\_model) %>%   
 add\_recipe(bank\_recipe)  
  
tree\_grid <- grid\_latin\_hypercube(cost\_complexity(),  
 tree\_depth(),  
 min\_n(),   
 size = 60)  
  
set.seed(888)  
  
tree\_tuning <- tree\_workflow %>%   
 tune\_grid(resamples = bank\_folds,  
 grid = tree\_grid)

## Warning: package 'vctrs' was built under R version 4.0.5

tree\_tuning %>% show\_best('roc\_auc')

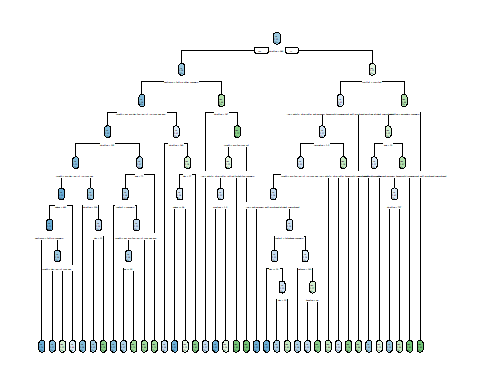
## # A tibble: 5 x 9  
## cost\_complexity tree\_depth min\_n .metric .estimator mean n std\_err  
## <dbl> <int> <int> <chr> <chr> <dbl> <int> <dbl>  
## 1 0.000000000688 10 20 roc\_auc binary 0.863 10 0.00653  
## 2 0.00000448 9 28 roc\_auc binary 0.862 10 0.00615  
## 3 0.00000322 9 27 roc\_auc binary 0.862 10 0.00623  
## 4 0.00000000641 11 24 roc\_auc binary 0.860 10 0.00646  
## 5 0.00000874 11 23 roc\_auc binary 0.860 10 0.00644  
## # ... with 1 more variable: .config <chr>

best\_tree <- tree\_tuning %>%   
 select\_best(metric = 'roc\_auc')  
  
  
final\_tree\_workflow <- tree\_workflow %>%   
 finalize\_workflow(best\_tree)  
  
  
tree\_wf\_fit <- final\_tree\_workflow %>%   
 fit(data = bank\_training)  
  
tree\_fit <- tree\_wf\_fit %>%   
 pull\_workflow\_fit()  
  
vip(tree\_fit)



rpart.plot(tree\_fit$fit, roundint = FALSE)

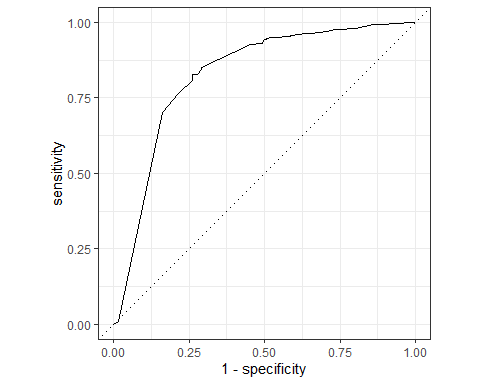
## Warning: labs do not fit even at cex 0.15, there may be some overplotting



tree\_last\_fit <- final\_tree\_workflow %>%   
 last\_fit(bank\_split)  
  
tree\_last\_fit %>% collect\_metrics()

## # A tibble: 2 x 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.892 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.827 Preprocessor1\_Model1

tree\_last\_fit %>% collect\_predictions() %>%   
 roc\_curve(truth = y, estimate = .pred\_no) %>%   
 autoplot()



tree\_predictions <- tree\_last\_fit %>% collect\_predictions()  
  
conf\_mat(tree\_predictions, truth = y, estimate = .pred\_class)

## Truth  
## Prediction no yes  
## no 949 71  
## yes 51 59

predict(tree\_last\_fit$.workflow[[1]],bank\_test[15,])

## # A tibble: 1 x 1  
## .pred\_class  
## <fct>   
## 1 no

saveRDS(tree\_last\_fit$.workflow[[1]],"./saved\_model.Rds")  
  
trained\_model<-readRDS("saved\_model.Rds")

### Naive Bayes

set.seed(888)  
nb\_split <- initial\_split(bank\_clean, prop = 0.75,   
 strata = y)  
  
nb\_training <- nb\_split %>% training()  
nb\_test <- nb\_split %>% testing()  
nb\_folds <- vfold\_cv(nb\_training, v = 10)  
  
nb\_recipe <- recipe(y ~ ., data = nb\_training)  
   
  
  
nb\_wf <- workflow() %>%  
 add\_recipe(nb\_recipe)  
  
library(discrim)

##   
## Attaching package: 'discrim'

## The following object is masked from 'package:dials':  
##   
## smoothness

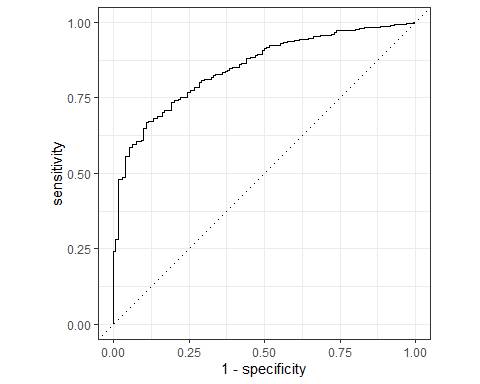
nb\_spec <- naive\_Bayes() %>%  
 set\_mode("classification") %>%  
 set\_engine("naivebayes")  
  
nb\_spec

## Naive Bayes Model Specification (classification)  
##   
## Computational engine: naivebayes

nb\_fit <- nb\_wf %>%  
 add\_model(nb\_spec) %>%  
 fit(data = nb\_training)  
  
nb\_wf\_final <- workflow() %>%  
 add\_recipe(nb\_recipe) %>%  
 add\_model(nb\_spec)  
  
nb\_rs <- fit\_resamples(  
 nb\_wf\_final,  
 nb\_folds,  
 control = control\_resamples(save\_pred = TRUE)  
)  
  
  
nb\_last\_fit <- nb\_wf\_final %>%   
 last\_fit(nb\_split)  
  
nb\_last\_fit %>% collect\_metrics()

## # A tibble: 2 x 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.881 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.849 Preprocessor1\_Model1

nb\_last\_fit %>% collect\_predictions() %>%   
 roc\_curve(truth = y, estimate = .pred\_no) %>%   
 autoplot()



nb\_predictions <- nb\_last\_fit %>% collect\_predictions()  
conf\_mat(nb\_predictions, truth = y, estimate = .pred\_class)

## Truth  
## Prediction no yes  
## no 956 91  
## yes 44 39

## XGBoost

as.data.table(bank\_clean)

## age job marital education default balance housing loan contact  
## 1: 30 unemployed married primary no 1787 no no cellular  
## 2: 33 services married secondary no 4789 yes yes cellular  
## 3: 35 management single tertiary no 1350 yes no cellular  
## 4: 30 management married tertiary no 1476 yes yes unknown  
## 5: 59 blue-collar married secondary no 0 yes no unknown  
## ---   
## 4517: 33 services married secondary no -333 yes no cellular  
## 4518: 57 self-employed married tertiary yes -3313 yes yes unknown  
## 4519: 57 technician married secondary no 295 no no cellular  
## 4520: 28 blue-collar married secondary no 1137 no no cellular  
## 4521: 44 entrepreneur single tertiary no 1136 yes yes cellular  
## day month duration campaign pdays previous poutcome y  
## 1: 19 oct 79 1 -1 0 unknown no  
## 2: 11 may 220 1 339 4 failure no  
## 3: 16 apr 185 1 330 1 failure no  
## 4: 3 jun 199 4 -1 0 unknown no  
## 5: 5 may 226 1 -1 0 unknown no  
## ---   
## 4517: 30 jul 329 5 -1 0 unknown no  
## 4518: 9 may 153 1 -1 0 unknown no  
## 4519: 19 aug 151 11 -1 0 unknown no  
## 4520: 6 feb 129 4 211 3 other no  
## 4521: 3 apr 345 2 249 7 other no

set.seed(888)  
xg\_split<- initial\_split(bank\_clean)  
xg\_train<-training(xg\_split)  
xg\_test<-testing(xg\_split)  
  
set.seed(888)  
xg\_folds<-vfold\_cv(xg\_train,v=10)  
  
xgb\_spec <- boost\_tree(  
 trees = 1000,   
 tree\_depth = tune(), min\_n = tune(),   
 loss\_reduction = tune(),   
 sample\_size = tune(), mtry = tune(),   
 learn\_rate = tune()   
) %>%   
 set\_engine("xgboost") %>%   
 set\_mode("classification")  
  
xgb\_spec

## Boosted Tree Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = 1000  
## min\_n = tune()  
## tree\_depth = tune()  
## learn\_rate = tune()  
## loss\_reduction = tune()  
## sample\_size = tune()  
##   
## Computational engine: xgboost

xgb\_grid <- grid\_latin\_hypercube(  
 tree\_depth(),  
 min\_n(),  
 loss\_reduction(),  
 sample\_size = sample\_prop(),  
 finalize(mtry(), xg\_train),  
 learn\_rate(),  
 size = 60  
)  
  
xgb\_grid

## # A tibble: 60 x 6  
## tree\_depth min\_n loss\_reduction sample\_size mtry learn\_rate  
## <int> <int> <dbl> <dbl> <int> <dbl>  
## 1 12 22 3.13e- 6 0.150 4 0.0000283   
## 2 6 8 1.95e-10 0.883 3 0.00000810   
## 3 6 36 1.33e+ 1 0.140 6 0.000316   
## 4 14 27 5.43e- 6 0.280 5 0.00568   
## 5 5 21 5.33e+ 0 0.390 7 0.000829   
## 6 10 32 6.16e- 7 0.496 9 0.000000195   
## 7 7 38 2.16e- 3 0.967 12 0.000000757   
## 8 5 37 3.53e-10 0.307 4 0.00830   
## 9 3 9 2.98e- 9 0.668 4 0.0208   
## 10 7 29 2.83e- 4 0.446 14 0.0000000100  
## # ... with 50 more rows

xgb\_recipe <- recipe(y ~ ., data = xg\_train) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes())  
   
  
  
tic()  
xgb\_recipe %>%  
 prep() %>%  
 bake(new\_data = xg\_train)

## # A tibble: 3,391 x 43  
## age balance day duration campaign pdays previous y job\_blue.collar  
## <int> <int> <int> <int> <int> <int> <int> <fct> <dbl>  
## 1 30 1787 19 79 1 -1 0 no 0  
## 2 33 4789 11 220 1 339 4 no 0  
## 3 35 747 23 141 2 176 3 no 0  
## 4 36 307 14 341 1 330 2 no 0  
## 5 43 -88 17 313 1 147 2 no 0  
## 6 39 9374 20 273 1 -1 0 no 0  
## 7 43 264 17 113 2 -1 0 no 0  
## 8 20 502 30 261 1 -1 0 yes 0  
## 9 31 360 29 89 1 241 1 no 1  
## 10 40 194 29 189 2 -1 0 no 0  
## # ... with 3,381 more rows, and 34 more variables: job\_entrepreneur <dbl>,  
## # job\_housemaid <dbl>, job\_management <dbl>, job\_retired <dbl>,  
## # job\_self.employed <dbl>, job\_services <dbl>, job\_student <dbl>,  
## # job\_technician <dbl>, job\_unemployed <dbl>, job\_unknown <dbl>,  
## # marital\_married <dbl>, marital\_single <dbl>, education\_secondary <dbl>,  
## # education\_tertiary <dbl>, education\_unknown <dbl>, default\_yes <dbl>,  
## # housing\_yes <dbl>, loan\_yes <dbl>, contact\_telephone <dbl>,  
## # contact\_unknown <dbl>, month\_aug <dbl>, month\_dec <dbl>, month\_feb <dbl>,  
## # month\_jan <dbl>, month\_jul <dbl>, month\_jun <dbl>, month\_mar <dbl>,  
## # month\_may <dbl>, month\_nov <dbl>, month\_oct <dbl>, month\_sep <dbl>,  
## # poutcome\_other <dbl>, poutcome\_success <dbl>, poutcome\_unknown <dbl>

toc()

## 0.11 sec elapsed

xgb\_wf <- workflow() %>%  
 add\_model(xgb\_spec) %>%  
 add\_recipe(xgb\_recipe)  
  
xgb\_wf

## == Workflow ====================================================================  
## Preprocessor: Recipe  
## Model: boost\_tree()  
##   
## -- Preprocessor ----------------------------------------------------------------  
## 1 Recipe Step  
##   
## \* step\_dummy()  
##   
## -- Model -----------------------------------------------------------------------  
## Boosted Tree Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = 1000  
## min\_n = tune()  
## tree\_depth = tune()  
## learn\_rate = tune()  
## loss\_reduction = tune()  
## sample\_size = tune()  
##   
## Computational engine: xgboost

library(doParallel)

## Loading required package: foreach

##   
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':  
##   
## accumulate, when

## Loading required package: iterators

## Loading required package: parallel

cores<-detectCores()  
cl<- makeCluster(cores[1]-4)  
registerDoParallel(cl)  
  
tic()  
set.seed(888)  
xgb\_res <- tune\_grid(  
 xgb\_wf,  
 resamples = xg\_folds,  
 grid = xgb\_grid,  
 control = control\_grid(save\_pred = TRUE))  
toc()

## 758.47 sec elapsed

best\_auc <- select\_best(xgb\_res, "roc\_auc")  
best\_auc

## # A tibble: 1 x 7  
## mtry min\_n tree\_depth learn\_rate loss\_reduction sample\_size .config   
## <int> <int> <int> <dbl> <dbl> <dbl> <chr>   
## 1 4 9 3 0.0208 0.00000000298 0.668 Preprocessor1\_Mo~

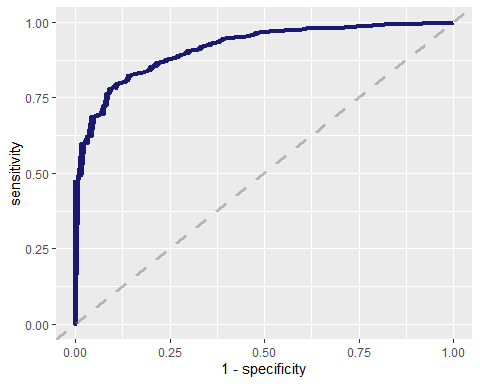
final\_xgb <- finalize\_workflow(  
 xgb\_wf,  
 best\_auc  
)  
  
tic()  
final\_res <- last\_fit(final\_xgb, xg\_split)  
collect\_metrics(final\_res)

## # A tibble: 2 x 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.909 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.915 Preprocessor1\_Model1

toc()

## 2.74 sec elapsed

final\_res %>%  
 collect\_predictions() %>%  
 roc\_curve(y, .pred\_no) %>%  
 ggplot(aes(x = 1 - specificity, y = sensitivity)) +  
 geom\_line(size = 1.5, color = "midnightblue") +  
 geom\_abline(  
 lty = 2, alpha = 0.5,  
 color = "gray50",  
 size = 1.2  
 )

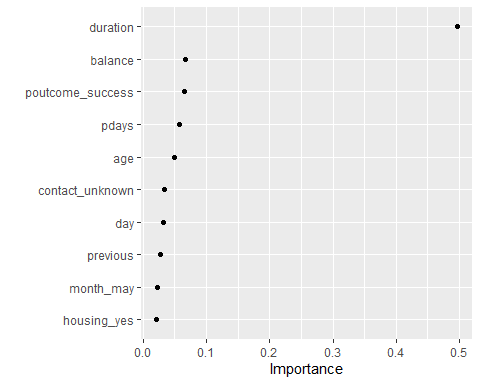


final\_res %>%  
 collect\_predictions() %>%   
 conf\_mat(truth = y, estimate = .pred\_class)

## Truth  
## Prediction no yes  
## no 992 86  
## yes 17 35

library(vip)  
final\_xgb %>%  
 fit(data = xg\_train) %>%  
 pull\_workflow\_fit() %>%  
 vip(geom = "point")

## [01:02:01] WARNING: amalgamation/../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval\_metric if you'd like to restore the old behavior.



final\_res %>% collect\_metrics()

## # A tibble: 2 x 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.909 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.915 Preprocessor1\_Model1

## Conclusions and Recommendations

Some text wrapping up the report