Project 2 University Enrollment

Xuhui Ying

10/05/2022

Logistic Models, the Tidymodel way.

Load Libraries

```
options(warn = -1)
library(tidyverse)
library(tidymodels)
library(janitor)
library(skimr)
library(kableExtra)
library(GGally)
library(kableExtra) # -- make nice looking resutls when we knitt
library(vip)        # --  tidymodels variable importance
library(fastshap)   # -- shapley values for variable importance
library(MASS)
```

Stage w. Readr

Import your data with read_csv()

```
inq05_samp <- read_csv("inq05_samp.csv") %>%
  clean_names()

df <- inq05_samp %>% dplyr::select(-academic_interest_1, -academic_interest_2, -contact_code1, -contact_date
1, -ethn_code, -irschool, -level_year, -satscore, -sex, -telecq, -referral_cntcts, -recr_code)

nrow(df)
```

```
## [1] 56237
```

```
head(df)
```

enroll <dbl>	total_contacts <dbl>	self_init_cntcts <dbl>	travel_init_cntcts <dbl>	solicited_cntcts <dbl>	campus_visit <dbl>	ma <dl
1	1	1	0	0	0	
1	8	7	1	0	0	
1	1	1	0	0	0	
1	6	6	0	0	0	
1	1	1	0	0	0	
1	3	3	0	0	0	

6 rows | 1-8 of 17 columns

Profile w. Skimr

skimr has lots of options and supports groupby - check it out.

```
df %>%
  skim()
```

Data summary

Name	Piped data
Number of rows	56237
Number of columns	17
________________________	
Column type frequency:	

character	1
numeric	16
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
instate	0	1	1	1	0	2	0

Variable type: numeric

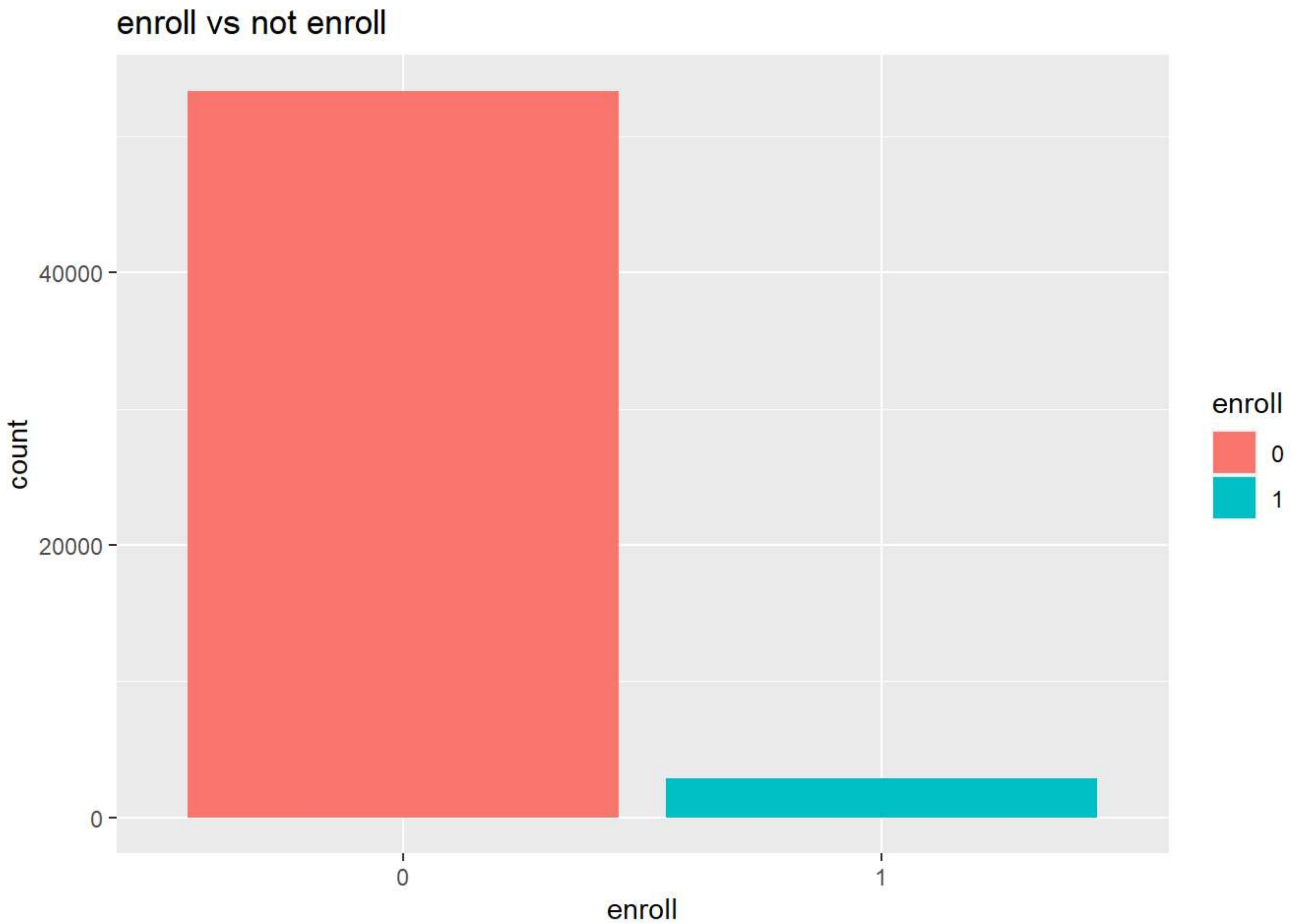
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
enroll	0	1.00	0.05	0.22	0.00	0.00	0.00	0.00	1.00	
total_contacts	0	1.00	2.26	2.01	1.00	1.00	2.00	3.00	58.00	
self_init_cntcts	0	1.00	1.31	1.81	0.00	0.00	1.00	2.00	56.00	
travel_init_cntcts	0	1.00	0.37	0.56	0.00	0.00	0.00	1.00	5.00	
solicited_cntcts	0	1.00	0.55	0.65	0.00	0.00	0.00	1.00	9.00	
campus_visit	0	1.00	0.04	0.20	0.00	0.00	0.00	0.00	2.00	
mailq	0	1.00	4.08	1.45	1.00	3.00	5.00	5.00	5.00	
premiere	0	1.00	0.04	0.19	0.00	0.00	0.00	0.00	1.00	
interest	0	1.00	0.06	0.25	0.00	0.00	0.00	0.00	3.00	
stuemail	0	1.00	0.50	0.50	0.00	0.00	0.00	1.00	1.00	
init_span	0	1.00	19.64	8.71	-216.00	13.00	19.00	25.00	91.00	
int1rat	0	1.00	0.04	0.02	0.00	0.02	0.04	0.05	1.00	
int2rat	0	1.00	0.04	0.03	0.00	0.02	0.06	0.06	1.00	
hscrat	0	1.00	0.04	0.06	0.00	0.00	0.04	0.05	1.00	
avg_income	12801	0.77	47605.93	20799.66	7117.00	32228.00	42606.00	57710.00	200001.00	
distance	11905	0.79	378.17	397.23	0.42	112.39	181.63	538.43	3901.07	

```
# address missing values (drop columns with more than 20% missing values)

data <- df %>% dplyr::select(-avg_income, -distance)

#data %>% write_csv("data.csv")
```

Explore target

what's the frequency of responders? remember the count function of summarize is n() with no parameters.

```
data$enroll <- as.factor(data$enroll)

data %>%
  ggplot(aes(x=enroll, fill=enroll)) +
  geom_histogram(stat="count") +
  labs(title = "enroll vs not enroll")
```




```
data %>%
  group_by(enroll) %>%
  summarize(n=n()) %>%
  ungroup() %>%
  mutate(pct = n/sum(n))
```

enroll <fct>	n <int>	pct <dbl>
0	53369	0.94900155
1	2868	0.05099845

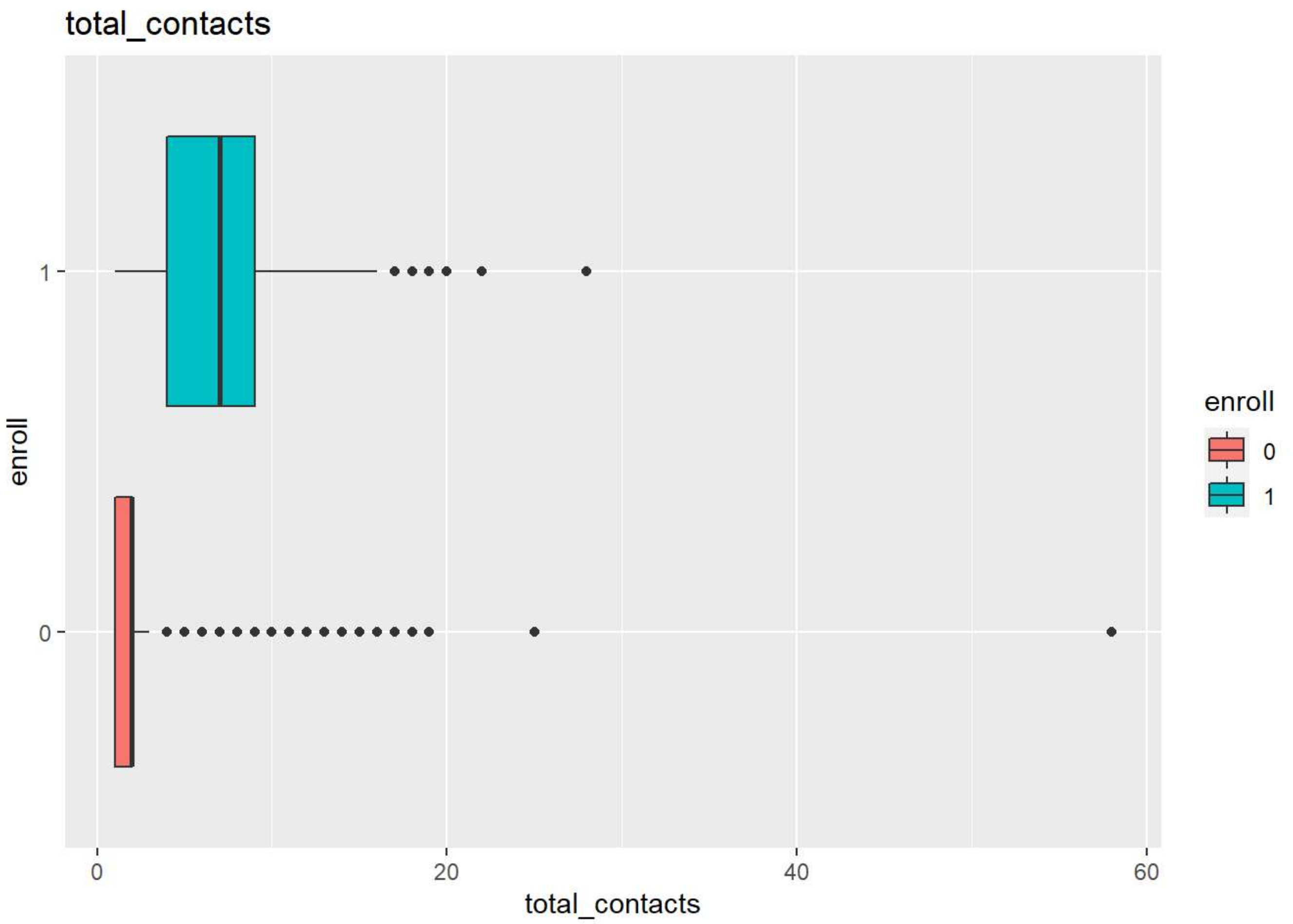
2 rows

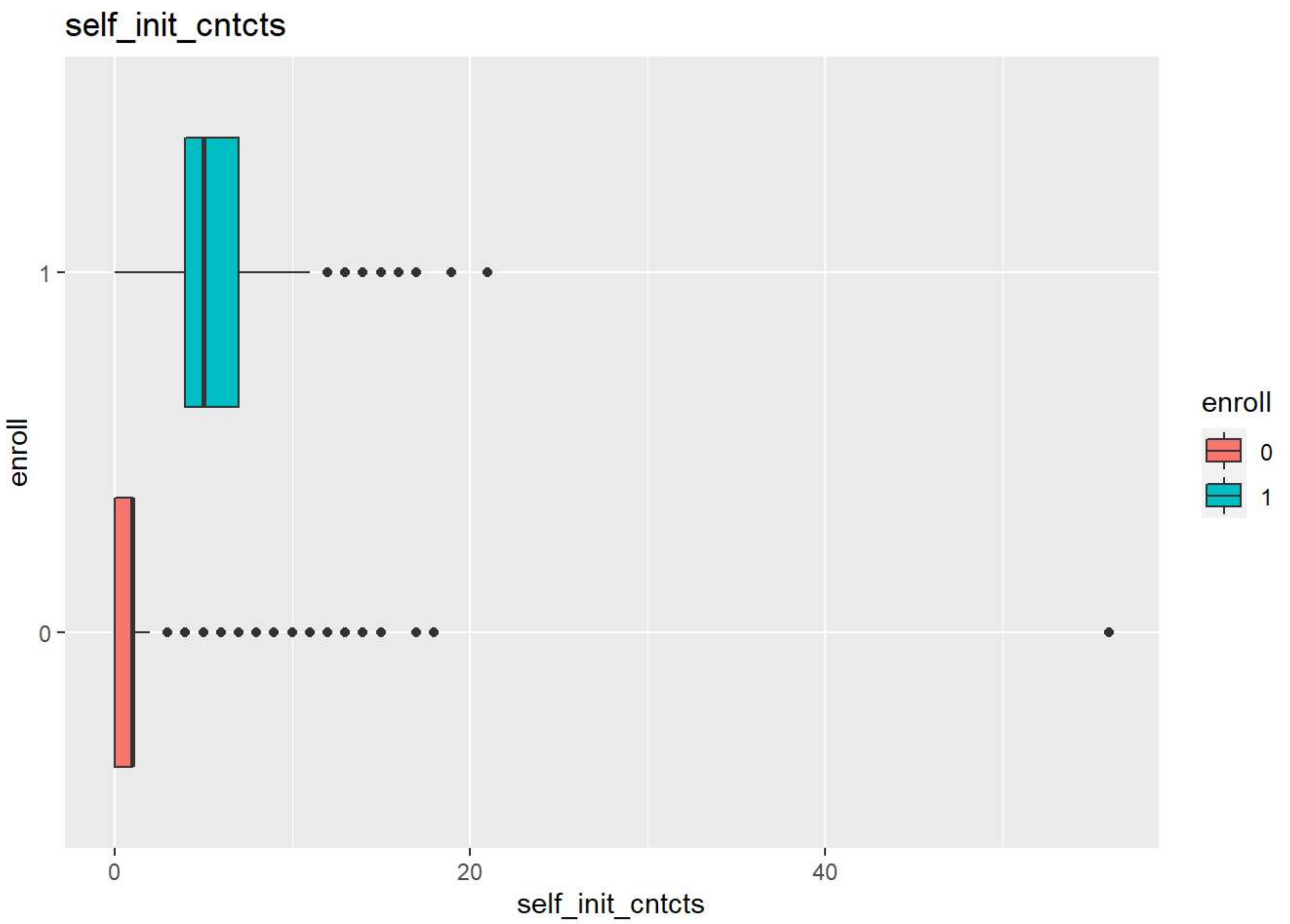
Explore numerics

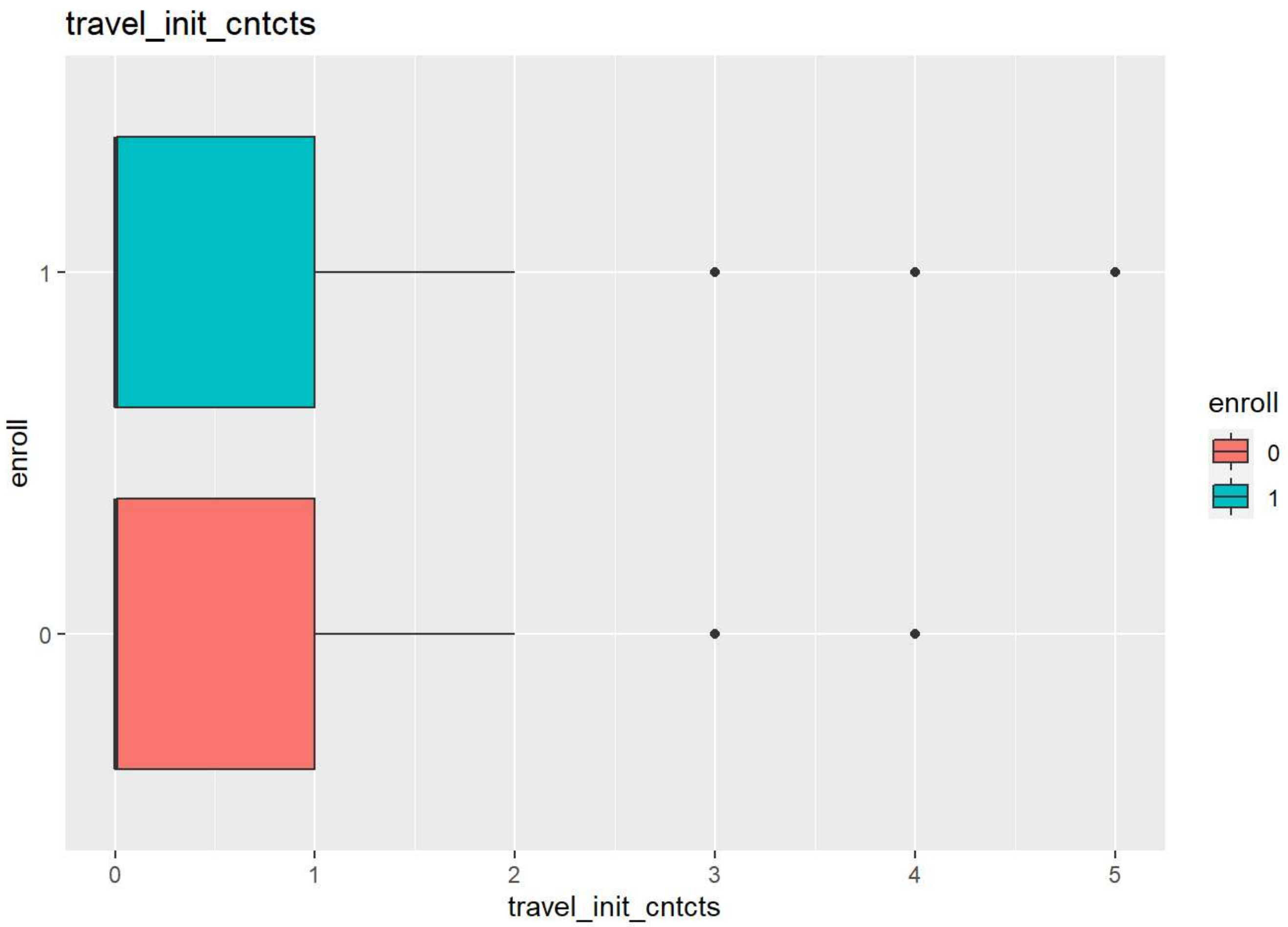
Compare numeric variables by comparing histograms of respond vs non-respond of course you can follow up with descriptive statistics too…

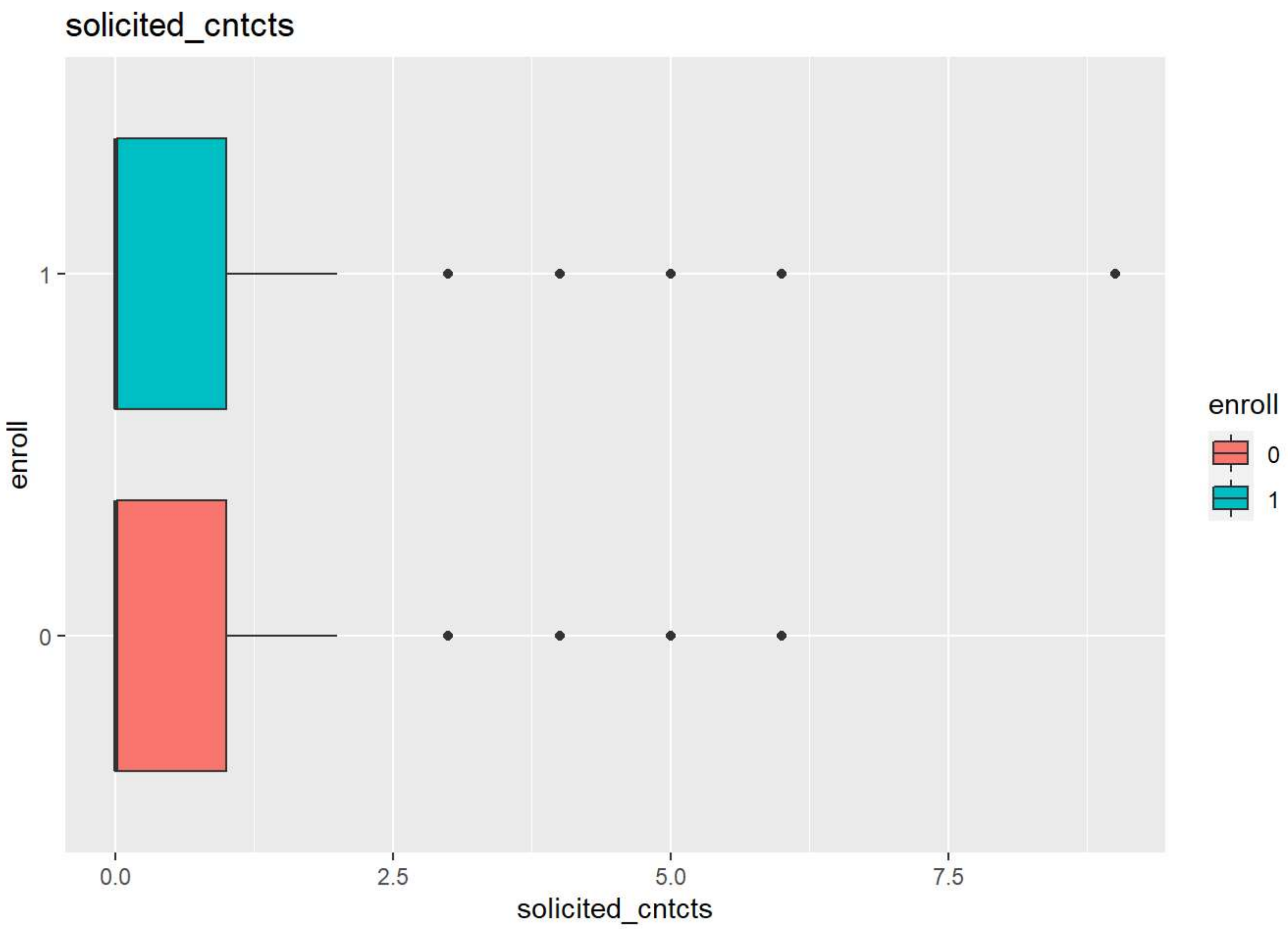
numeric variables: total_contacts, self_init_cntcts, travel_init_cntcts, solicited_cntcts, referral_cntcts, mailq, interest, init_span, int1rat, int2rat, hscrat

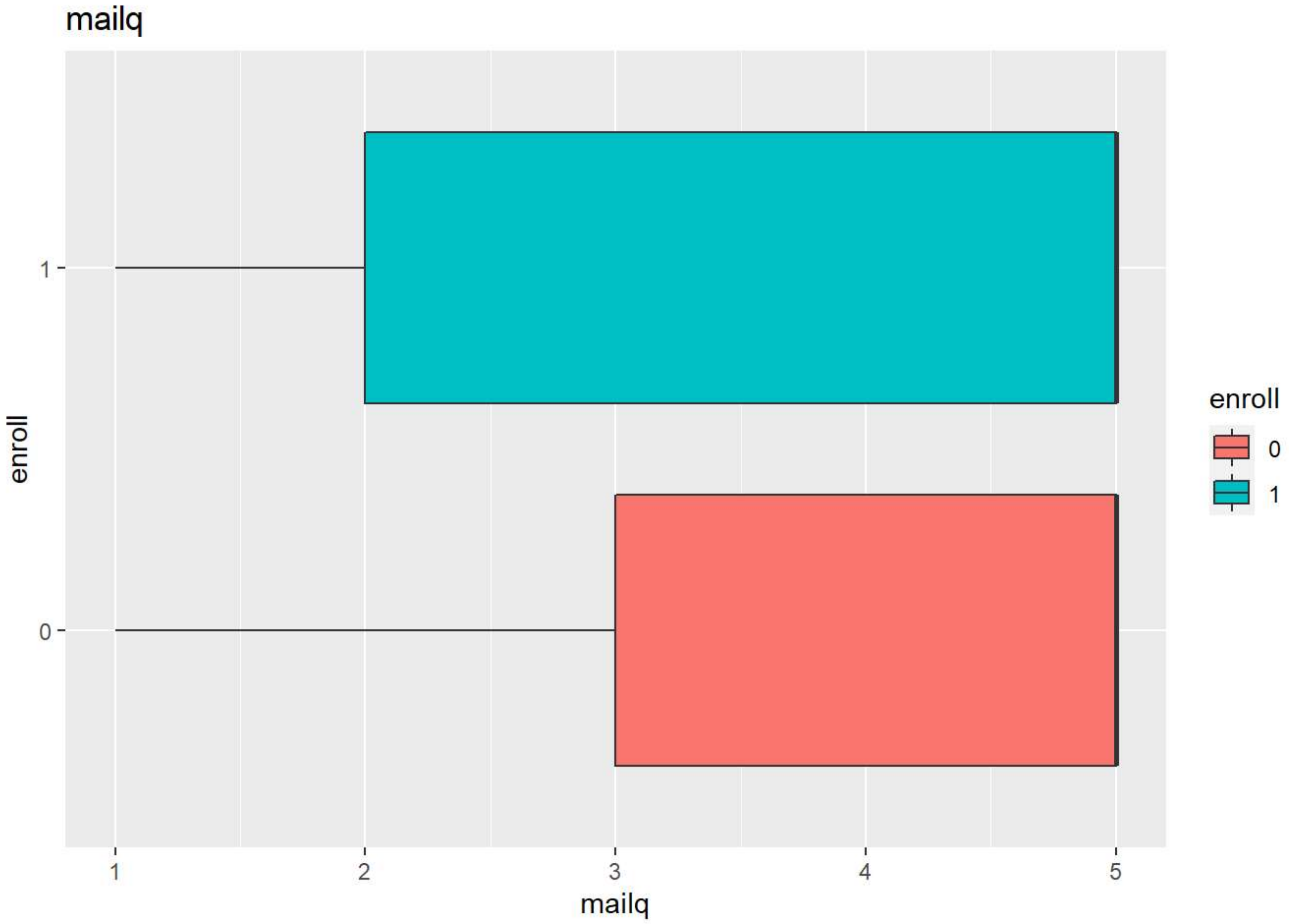
```
# -- comparative boxplots

boxplot <- function(m){
    ggplot(data, aes(x=!!as.name(m), y=enroll, fill=enroll)) +
    geom_boxplot() +
    labs(title = as.character(m))
}

numerics <- c("total_contacts", "self_init_cntcts", "travel_init_cntcts", "solicited_cntcts", "mailq", "inte
         rest", "init_span", "int1rat", "int2rat", "hscrat")

for (c in numerics){
    print(boxplot(c))
}
```

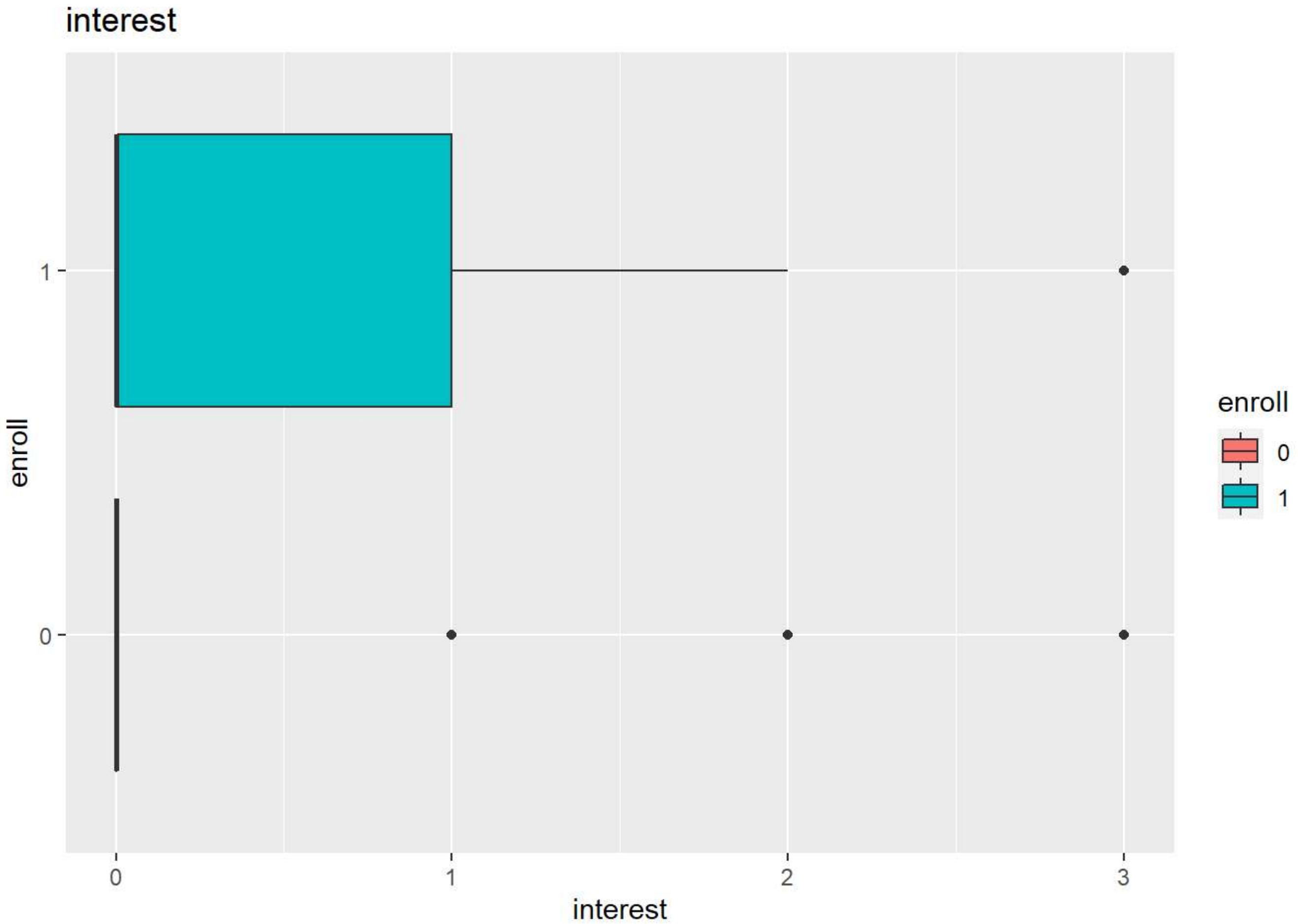

total_contacts
enroll
1
0
0
20
40
60
total_contacts
enroll
0
1

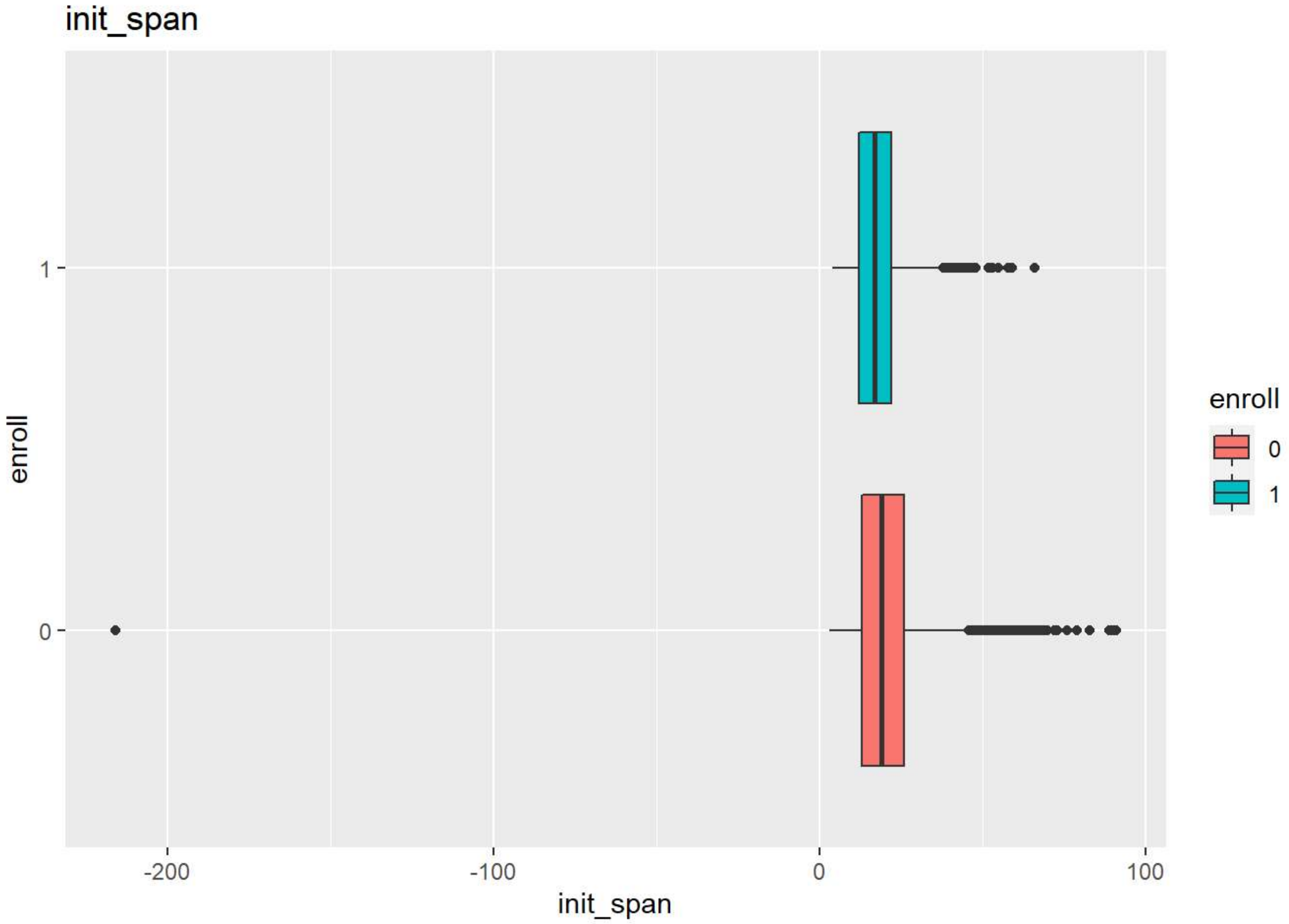

self_init_cntcts
enroll
1
0
0
20
40
self_init_cntcts
enroll
0
1

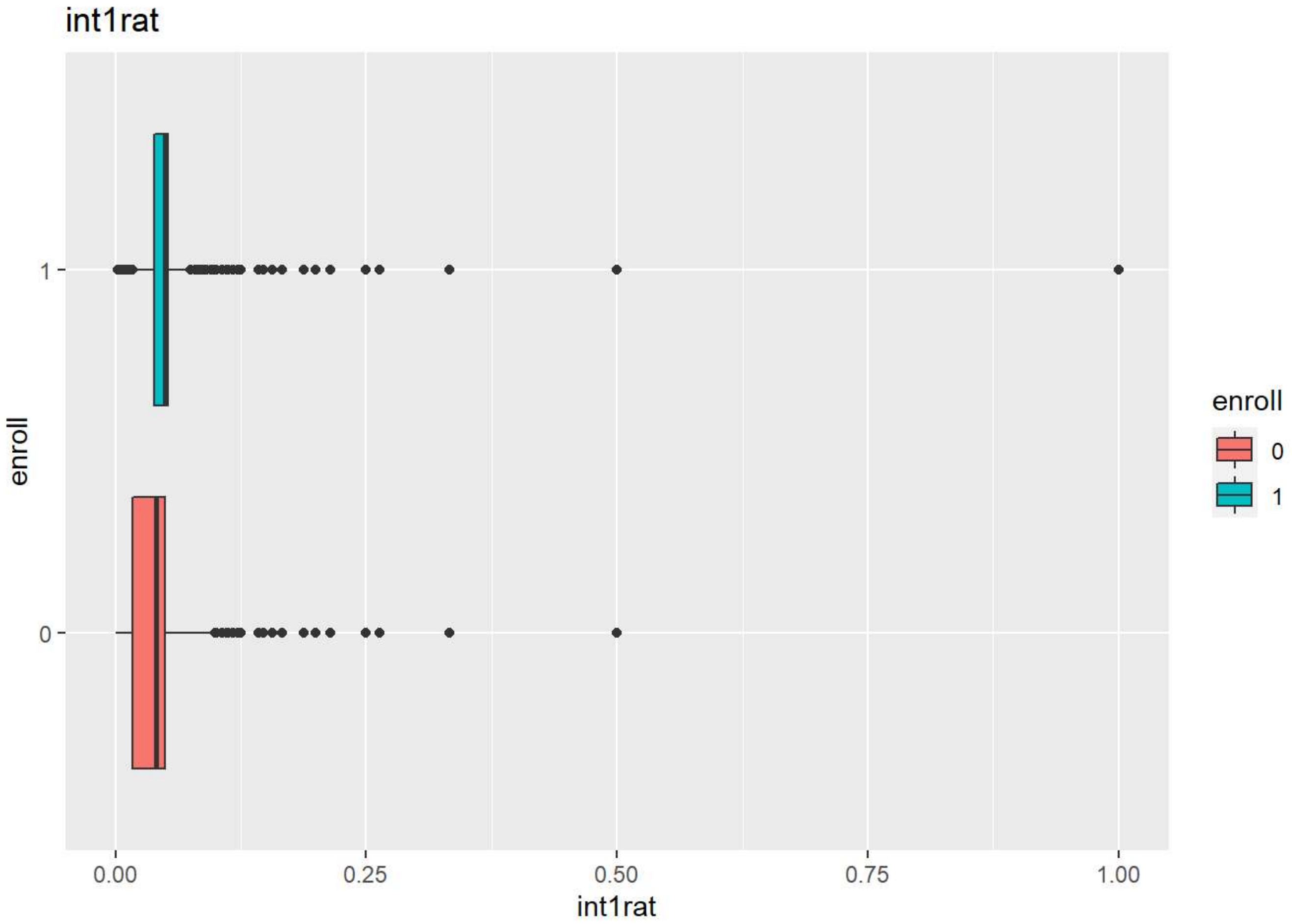

travel_init_cntcts
enroll
1
0
0
1
2
3
4
5
travel_init_cntcts
enroll
0
1

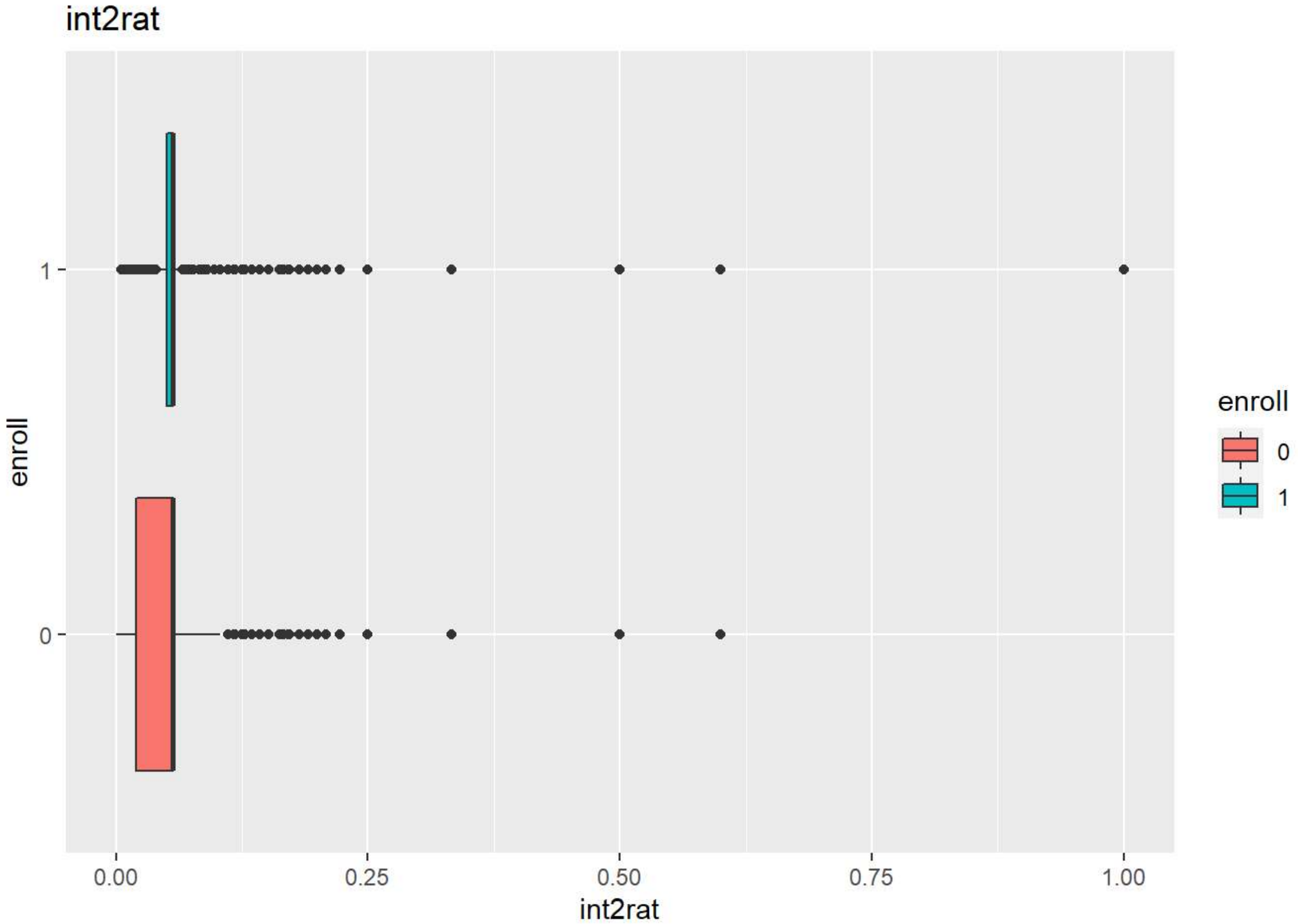

solicited_cntcts
enroll
1
0
0.0
2.5
5.0
7.5
solicited_cntcts
enroll
0
1

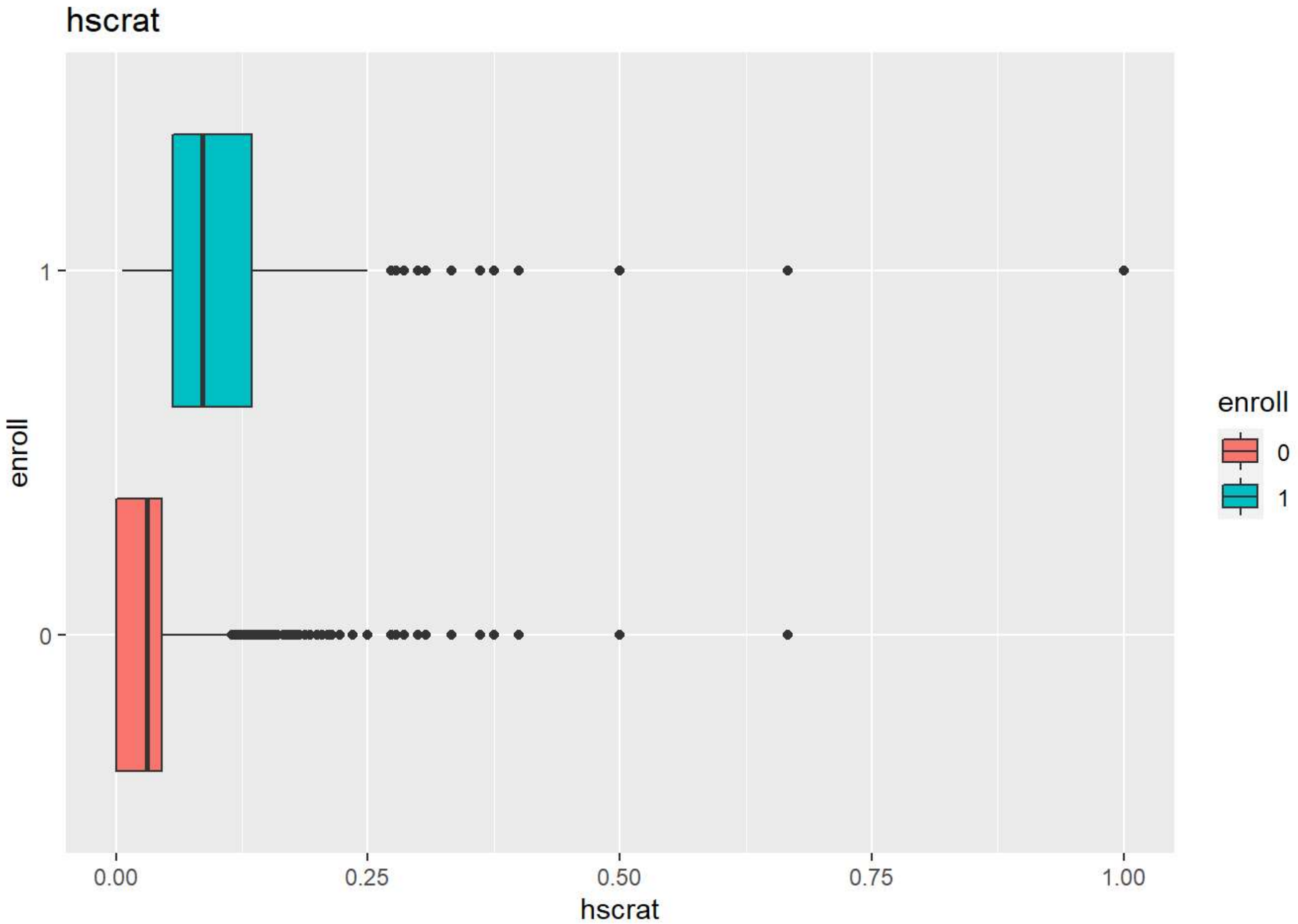

mailq
enroll
1
0
1
2
3
4
5
mailq
enroll
0
1


interest
enroll
1
0
0
1
2
3
interest
enroll
0
1


init_span
enroll
1
0
-200
-100
0
100
init_span
enroll
0
1


int1rat
enroll
1
0
0.00
0.25
0.50
0.75
1.00
int1rat
enroll
0
1


int2rat
enroll
1
0
0.00
0.25
0.50
0.75
1.00
int2rat
enroll
0
1

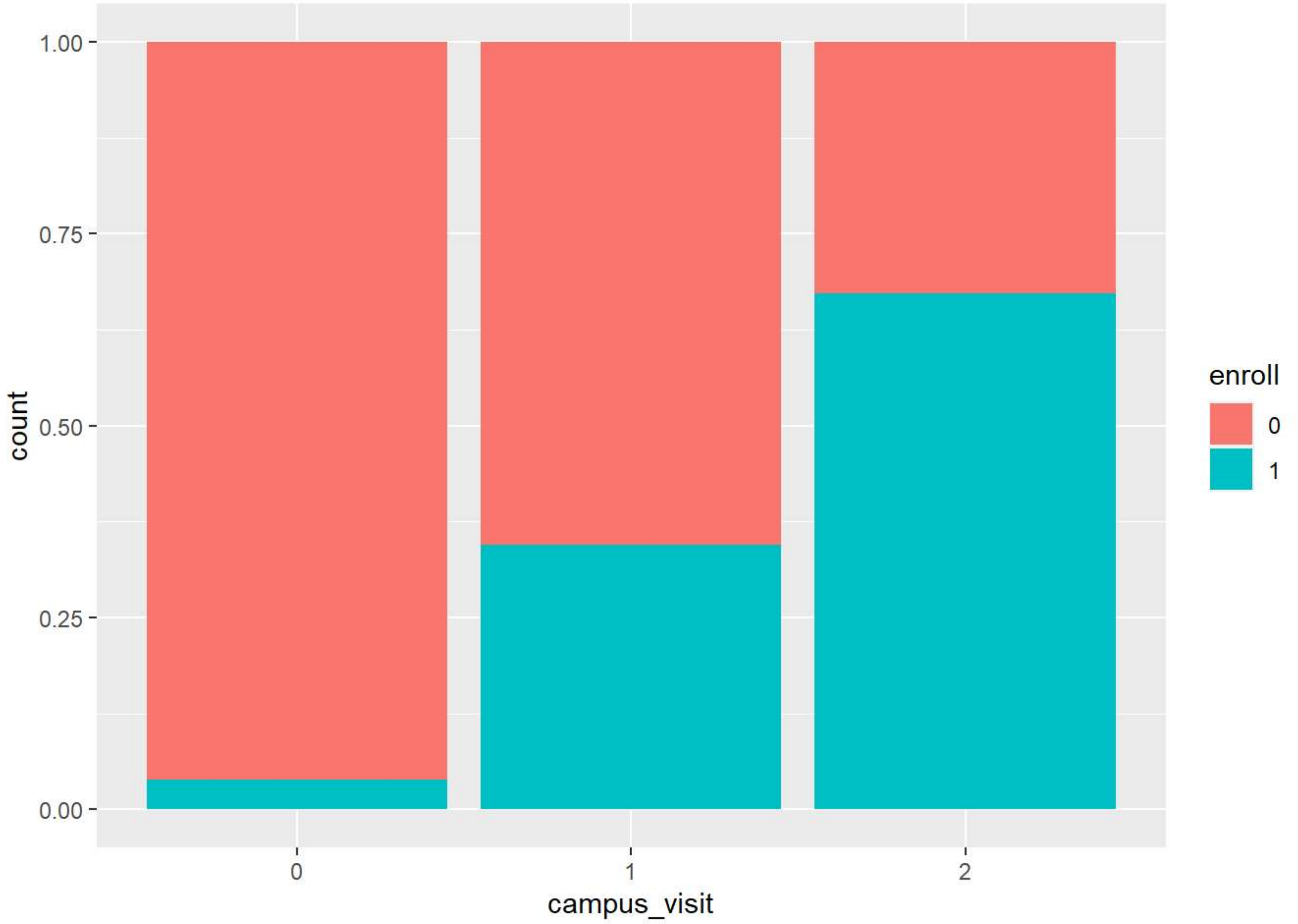

hscrat
enroll
1
0
0.00
0.25
0.50
0.75
1.00
hscrat
enroll
0
1

Explore character variables

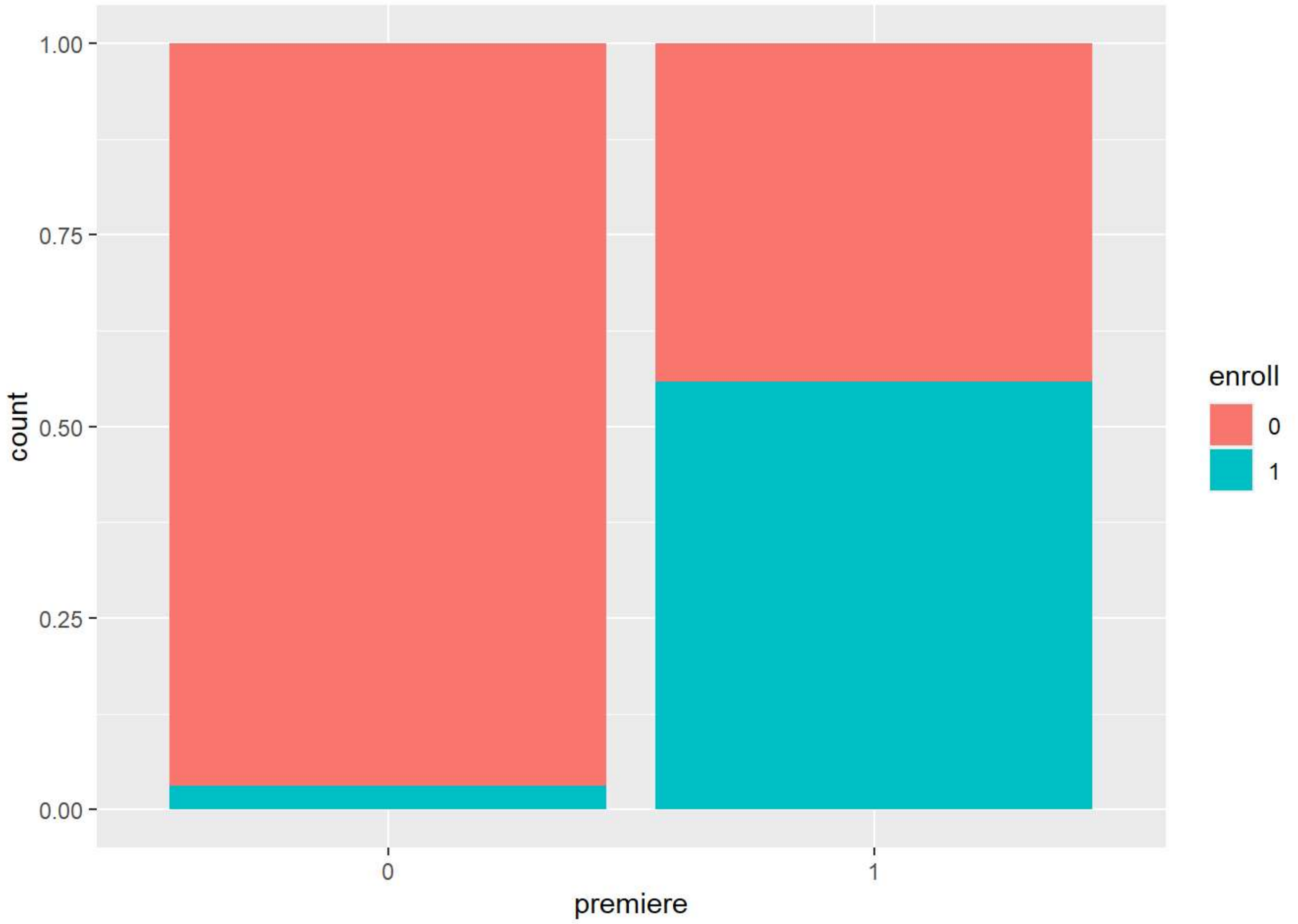
cycle through each character column and look for separation for respond vs non-respond

categorical variables: recr_code, campus_visit, premiere, stuemail, instate

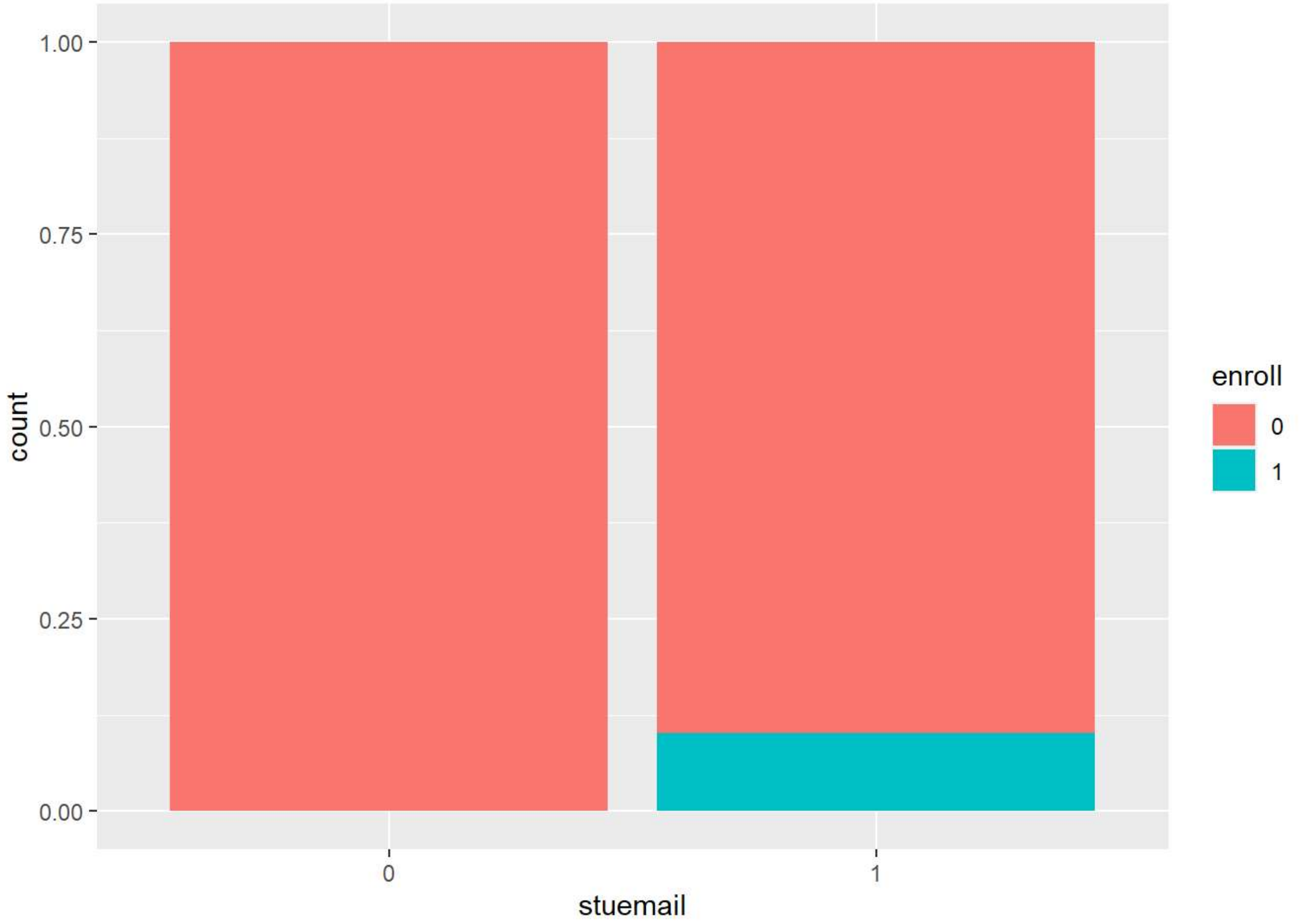
```
char_explore <- function(col){
    data %>%
    ggplot(aes(!!as.name(col))) +
    geom_bar(aes(fill = enroll), position = "fill")
}

data$campus_visit <- as.character(data$campus_visit)
data$premiere <- as.character(data$premiere)
data$stuemail <- as.character(data$stuemail)

# -- for each character column, create a chart
for (column in names(data %>% select_if (is_character))){
    chrt <- char_explore(column)
    print(chrt)
}
```

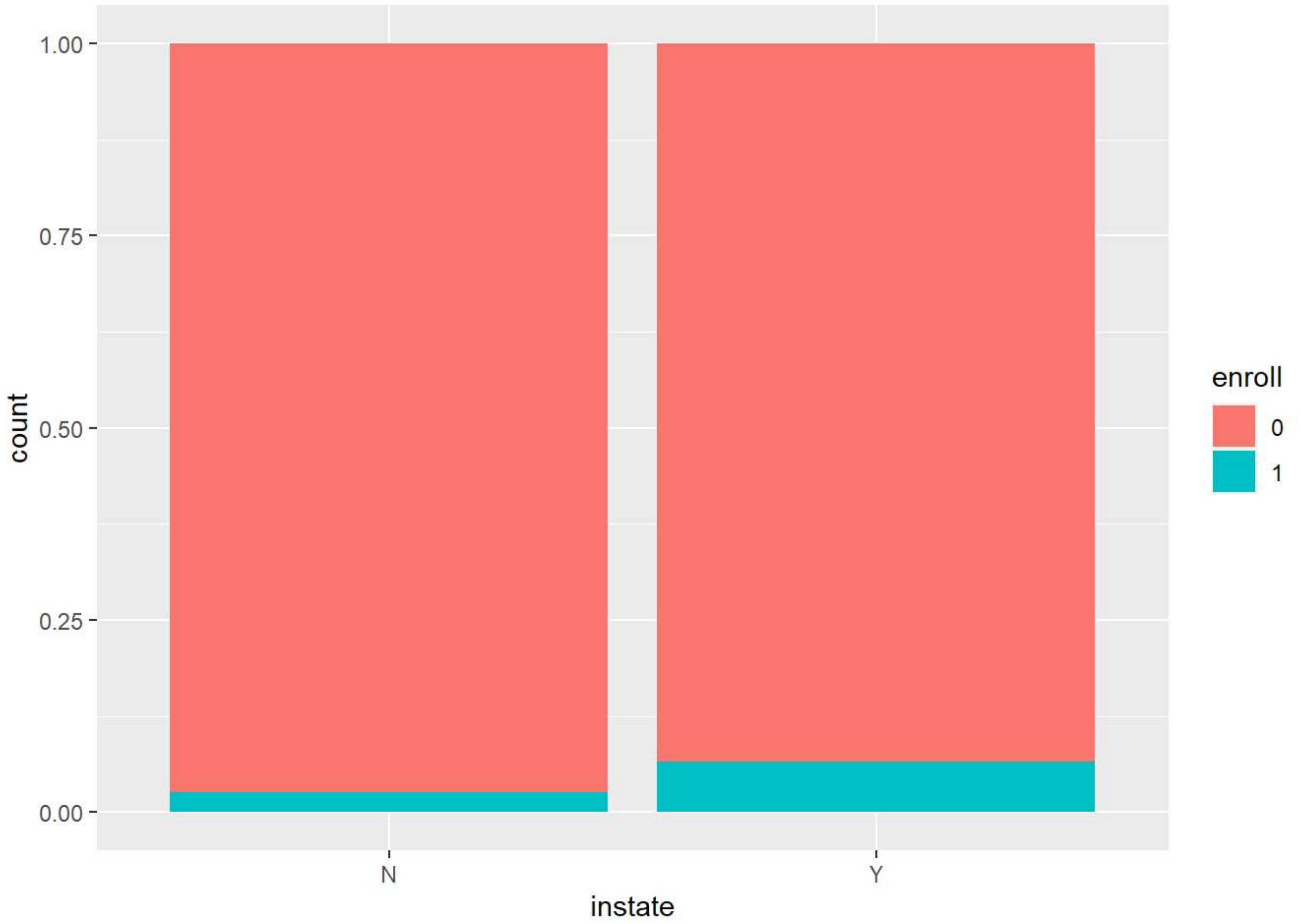


1.00
0.75
0.50
0.25
0.00
count
0
1
2
campus_visit
enroll
0
1



1.00
0.75
0.50
0.25
0.00
count
0
1
premiere
enroll
0
1


count
1.00
0.75
0.50
0.25
0.00
0
1
stuemail
enroll
0
1


count
1.00
0.75
0.50
0.25
0.00
N
Y
instate
enroll
0
1

0. Make Factors!

The next step for us is to create a dataset for modeling. Let's include a set of all of the columns we are interested in, and convert all the **character columns** to **factors** as well as any "nominal" or low frequency numeric columns likely to be a factor. This is done for the modeling functions coming later.

```
data %>%
    mutate_if(is.character, factor)  -> data_prep

head(data_prep)
```

enroll <fct>	total_contacts <dbl>	self_init_cntcts <dbl>	travel_init_cntcts <dbl>	solicited_cntcts <dbl>	campus_visit <fct>	ma <dl
1	1	1	0	0	0	
1	8	7	1	0	0	
1	1	1	0	0	0	
1	6	6	0	0	0	
1	1	1	0	0	0	
1	3	3	0	0	0	

6 rows | 1-8 of 15 columns

1. Partition my data 70/30 (train / test split)

```
# -- set a random seed for repeatablity
set.seed(1234)

# -- performs our train / test split
data_split <- initial_split(data_prep, prop = 0.7)

# -- extract the training data
data_train <- training(data_split)
# -- extract the test data
data_test <- testing(data_split)

sprintf("Train PCT : %1.2f%%", nrow(data_train)/ nrow(data) * 100)
```

```
## [1] "Train PCT : 70.00%"
```

```
sprintf("Test  PCT : %1.2f%%", nrow(data_test)/ nrow(data) * 100)
```

```
## [1] "Test  PCT : 30.00%"
```

2. Recipe

```
# -- create our recipe --
data_recipe <- recipe(enroll ~ ., data = data_train) %>%
#  step_rm(duration) %>%
  step_impute_mode(all_nominal(), -all_outcomes()) %>%
  step_impute_median(all_numeric()) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  prep()

data_recipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         14
##
## Training data contained 39365 data points and no missing data.
##
## Operations:
##
## Mode imputation for campus_visit, premiere, stuemail, instate [trained]
## Median imputation for total_contacts, self_init_cntcts, travel_init_c... [trained]
## Dummy variables from campus_visit, premiere, stuemail, instate [trained]
```

3. Bake

```
# -- apply the recipe
bake_train <- bake(data_recipe, new_data = data_train)
bake_test  <- bake(data_recipe, new_data = data_test)
```

4. Fit

```
## logistic code is here for reference and comparison

logistic_glm <-logistic_reg(mode = "classification") %>%
                set_engine("glm") %>%
                fit(enroll ~ ., data = bake_train)

## -- check out your parameter estimates ...
tidy(logistic_glm) %>%
  mutate_at(c("estimate", "std.error", "statistic", "p.value"),round, 4)
```

term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	-24.0851	190.3719	-0.1265	0.8993
total_contacts	0.3869	0.1152	3.3578	0.0008
self_init_cntcts	0.1239	0.1174	1.0552	0.2913
travel_init_cntcts	-0.0280	0.1276	-0.2195	0.8263
solicited_cntcts	-0.4336	0.1216	-3.5669	0.0004
mailq	0.1871	0.0252	7.4171	0.0000
interest	0.5605	0.0746	7.5124	0.0000
init_span	-0.0587	0.0051	-11.4447	0.0000
int1rat	5.2312	1.2619	4.1455	0.0000
int2rat	5.6289	1.1038	5.0997	0.0000

1-10 of 16 rows Previous **1** 2 Next

5. Prep for Evaluation

We want to attach both the Predicted Probabilities (.pred_No, .pred_Yes) and the Predicted Class (.pred_class) to the dataset so we can deep dive into where out model is performing well and where it's not. We do this to both the Training and the Test set.

```
# -- training
predict(logistic_glm, bake_train, type = "prob") %>%
  bind_cols(.,predict(logistic_glm, bake_train)) %>%
  bind_cols(.,bake_train) -> scored_train_glm

head(scored_train_glm)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	total_contacts <dbl>	self_init_cntcts <dbl>	travel_init_cntcts <dbl>
0.9541034	4.589657e-02	0	2	1	1
1.0000000	1.245570e-10	0	1	0	0
0.9132568	8.674321e-02	0	7	4	1
0.9984921	1.507886e-03	0	1	0	0
1.0000000	9.869545e-11	0	1	0	0
1.0000000	1.005288e-10	0	1	0	1

6 rows | 1-8 of 19 columns

```
# -- testing
predict(logistic_glm, bake_test, type = "prob") %>%
  bind_cols(.,predict(logistic_glm, bake_test)) %>%
  bind_cols(.,bake_test) -> scored_test_glm

head(scored_test_glm)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	total_contacts <dbl>	self_init_cntcts <dbl>	travel_init_cntcts <dbl>
8.917703e-01	0.10822973	0	1	1	0
9.602403e-01	0.03975973	0	1	1	0
3.535963e-01	0.64640367	1	6	6	0
1.807514e-04	0.99981925	1	3	3	0
9.537229e-01	0.04627712	0	2	2	0
5.065245e-06	0.99999493	1	10	10	0

6 rows | 1-8 of 19 columns

6. Evaluate

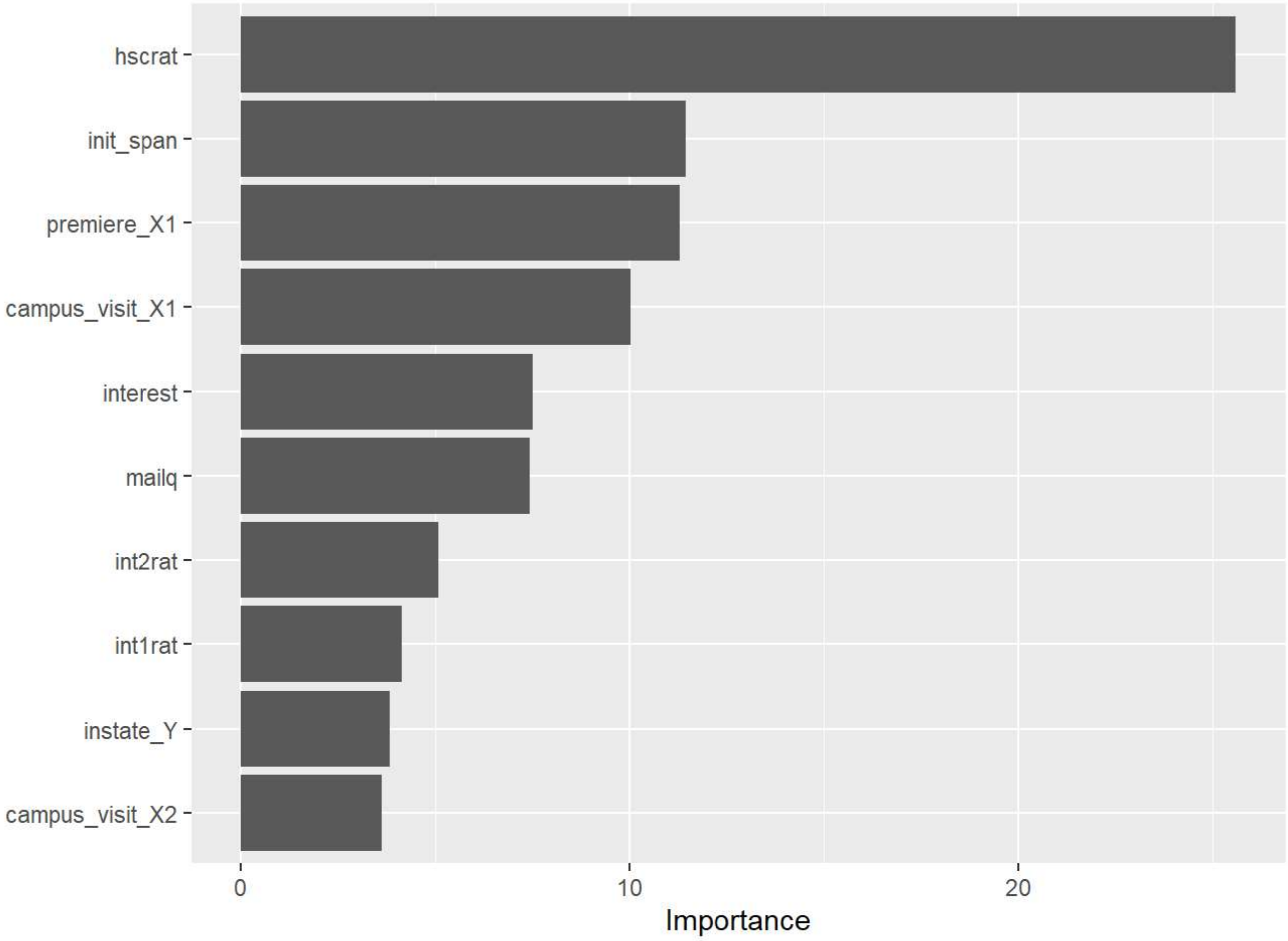
We want to check our model’s performance and take a look at which features were most important.

```
options(yardstick.event_first = FALSE)


# -- AUC: Train and Test
scored_train_glm %>%
  metrics(enroll, .pred_1, estimate = .pred_class) %>%
  mutate(part="training") %>%
  bind_rows( scored_test_glm %>%
               metrics(enroll, .pred_1, estimate = .pred_class) %>%
               mutate(part="testing")
  ) %>%
  filter(.metric %in% c("accuracy", "roc_auc"))
```

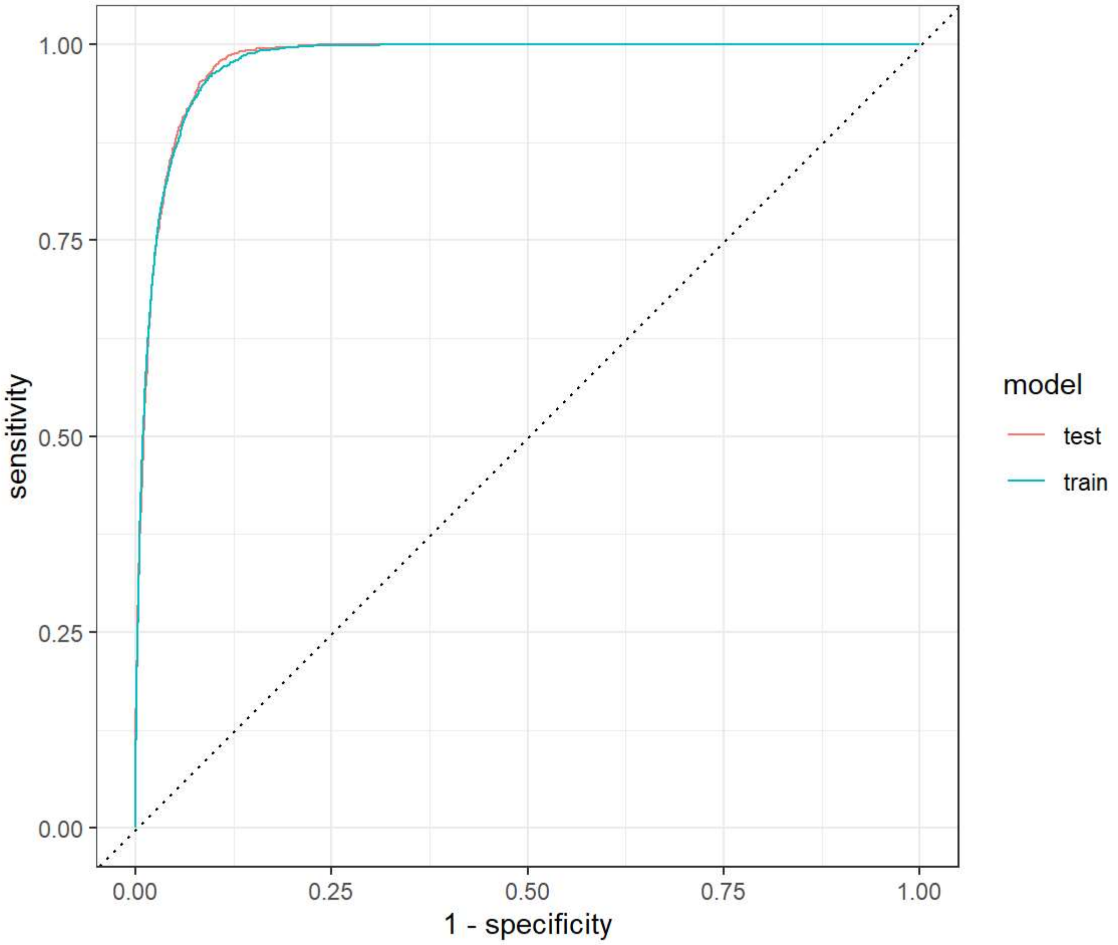
.metric <chr>	.estimator <chr>	.estimate <dbl>	part <chr>
accuracy	binary	0.9655532	training
roc_auc	binary	0.9779126	training
accuracy	binary	0.9650308	testing
roc_auc	binary	0.9786638	testing

4 rows

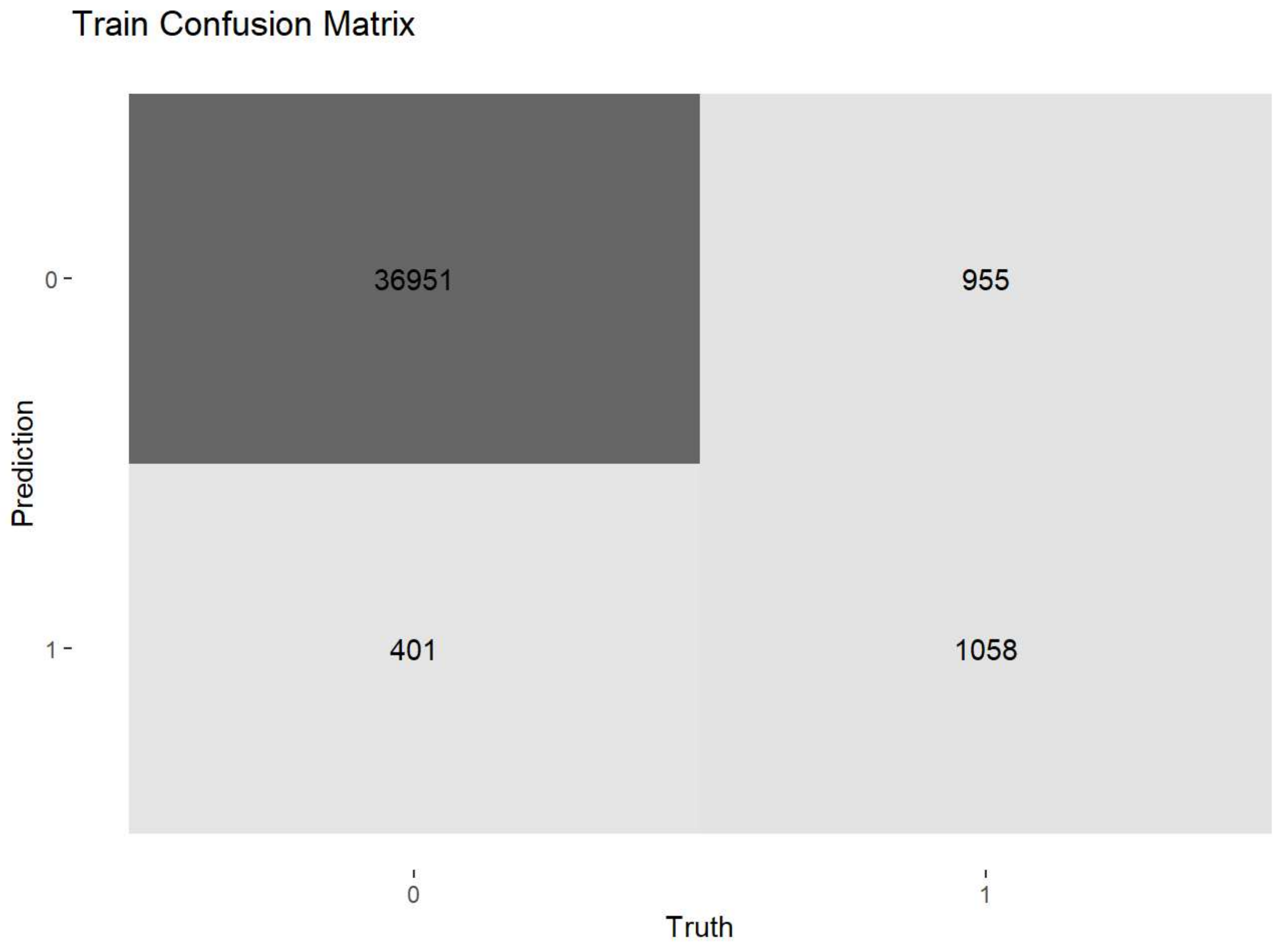
```
# -- Variable Importance top 10 features
logistic_glm %>%
  vip(num_features = 10)
```



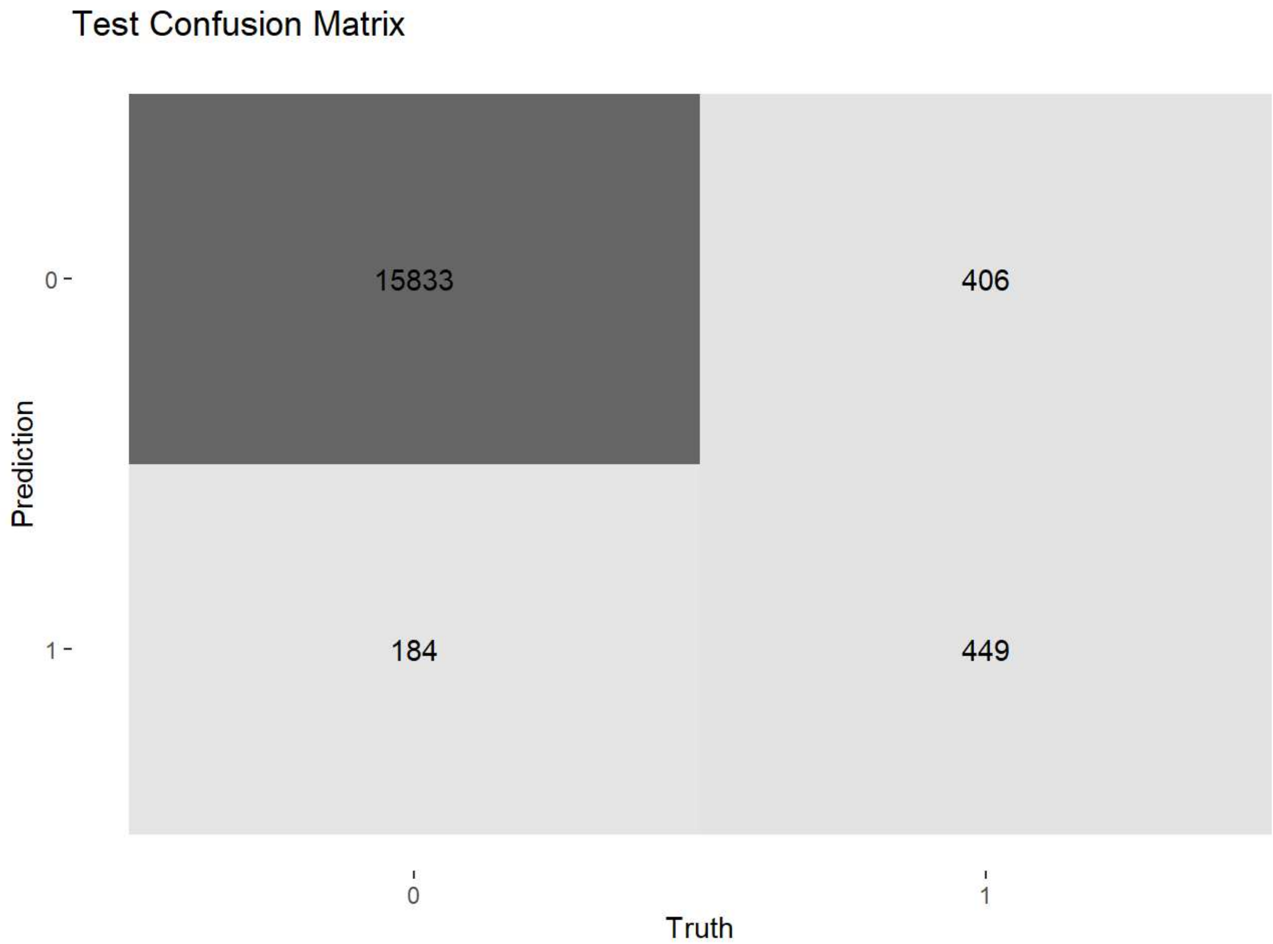

```
# -- ROC Charts
scored_train_glm %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_glm %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(enroll, .pred_1) %>%
  autoplot()
```




```
# -- Confustion Matricies
scored_train_glm %>%
  conf_mat(enroll, .pred_class) %>%
  autoplot( type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_glm %>%
  conf_mat(enroll, .pred_class) %>%
  autoplot( type = "heatmap") +
  labs(title="Test Confusion Matrix")
```




```
## -- Use stepwise selection to reduce the model

steplog <- glm(enroll ~ ., data = bake_train, family=binomial(link="logit"))
step <- stepAIC(steplog, direction="both")
```

```
## Start:  AIC=6577.81
## enroll ~ total_contacts + self_init_cntcts + travel_init_cntcts +
##     solicited_cntcts + mailq + interest + init_span + int1rat +
##     int2rat + hscrat + campus_visit_X1 + campus_visit_X2 + premiere_X1 +
##     stuemail_X1 + instate_Y
##
##                      Df Deviance    AIC
## - travel_init_cntcts  1   6545.9 6575.9
## - self_init_cntcts    1   6546.9 6576.9
## <none>                    6545.8 6577.8
## - total_contacts      1   6556.9 6586.9
## - solicited_cntcts    1   6558.3 6588.3
## - instate_Y           1   6560.7 6590.7
## - campus_visit_X2     1   6561.5 6591.5
## - int1rat             1   6563.3 6593.3
## - int2rat             1   6569.9 6599.9
## - interest            1   6600.5 6630.5
## - mailq               1   6602.4 6632.4
## - campus_visit_X1     1   6643.9 6673.9
## - premiere_X1         1   6672.1 6702.1
## - init_span           1   6691.3 6721.3
## - stuemail_X1         1   7362.8 7392.8
## - hscrat              1   7625.7 7655.7
##
## Step:  AIC=6575.85
## enroll ~ total_contacts + self_init_cntcts + solicited_cntcts +
##     mailq + interest + init_span + int1rat + int2rat + hscrat +
##     campus_visit_X1 + campus_visit_X2 + premiere_X1 + stuemail_X1 +
##     instate_Y
##
##                      Df Deviance    AIC
## <none>                    6545.9 6575.9
## + travel_init_cntcts  1   6545.8 6577.8
## - self_init_cntcts    1   6552.3 6580.3
## - instate_Y           1   6560.7 6588.7
## - campus_visit_X2     1   6561.6 6589.6
## - int1rat             1   6563.3 6591.3
## - int2rat             1   6569.9 6597.9
## - solicited_cntcts    1   6583.9 6611.9
## - total_contacts      1   6589.4 6617.4
## - interest            1   6600.6 6628.6
## - mailq               1   6606.4 6634.4
## - campus_visit_X1     1   6644.3 6672.3
## - premiere_X1         1   6672.1 6700.1
## - init_span           1   6691.3 6719.3
```

```
## - stuemail_X1         1   7363.6 7391.6
## - hscrat              1   7629.1 7657.1
```

```
summary(step)
```

```
##
## Call:
## glm(formula = enroll ~ total_contacts + self_init_cntcts + solicited_cntcts +
##     mailq + interest + init_span + int1rat + int2rat + hscrat +
##     campus_visit_X1 + campus_visit_X2 + premiere_X1 + stuemail_X1 +
##     instate_Y, family = binomial(link = "logit"), data = bake_train)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -6.889  -0.157   0.000   0.000   3.085
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -24.091477 190.372752  -0.127 0.899297
## total_contacts     0.364734   0.055218   6.605 3.97e-11 ***
## self_init_cntcts   0.146319   0.057927   2.526 0.011540 *
## solicited_cntcts  -0.411363   0.066946  -6.145 8.01e-10 ***
## mailq              0.188336   0.024579   7.663 1.82e-14 ***
## interest           0.560530   0.074607   7.513 5.78e-14 ***
## init_span         -0.058650   0.005125 -11.443  < 2e-16 ***
## int1rat            5.231269   1.261863   4.146 3.39e-05 ***
## int2rat            5.622590   1.103444   5.095 3.48e-07 ***
## hscrat            12.264189   0.478629  25.624  < 2e-16 ***
## campus_visit_X1    0.970875   0.096717  10.038  < 2e-16 ***
## campus_visit_X2    2.119043   0.585656   3.618 0.000297 ***
## premiere_X1        1.044916   0.092643  11.279  < 2e-16 ***
## stuemail_X1       18.414758 190.372655   0.097 0.922941
## instate_Y          0.399986   0.104936   3.812 0.000138 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15891.6  on 39364  degrees of freedom
## Residual deviance:  6545.9  on 39350  degrees of freedom
## AIC: 6575.9
##
## Number of Fisher Scoring iterations: 20
```

```
## -- remove insignificant variable (stuemail)

model_1 <- glm(enroll ~ total_contacts + self_init_cntcts + solicited_cntcts + mailq + interest + init_span
         + int1rat + int2rat + hscrat + campus_visit + premiere + instate, data = data_train, family=binomi
         al(link="logit"))


model_1
```

```
##
## Call:  glm(formula = enroll ~ total_contacts + self_init_cntcts + solicited_cntcts +
##     mailq + interest + init_span + int1rat + int2rat + hscrat +
##     campus_visit + premiere + instate, family = binomial(link = "logit"),
##     data = data_train)
##
## Coefficients:
##      (Intercept)    total_contacts  self_init_cntcts  solicited_cntcts
##         -6.29381           0.43612           0.17495          -0.37087
##            mailq          interest         init_span           int1rat
##          0.17161           0.67259          -0.06979           6.08524
##          int2rat            hscrat     campus_visit1     campus_visit2
##          6.36424          11.84376           1.02076           2.30075
##        premiere1          instateY
##          0.99552           0.33732
##
## Degrees of Freedom: 39364 Total (i.e. Null);  39351 Residual
## Null Deviance:       15890
## Residual Deviance: 7364  AIC: 7392
```

```
summary(model_1)
```

```
##
## Call:
## glm(formula = enroll ~ total_contacts + self_init_cntcts + solicited_cntcts +
##     mailq + interest + init_span + int1rat + int2rat + hscrat +
##     campus_visit + premiere + instate, family = binomial(link = "logit"),
##     data = data_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -7.5931  -0.1710  -0.1081  -0.0689   3.3958
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -6.29381    0.17880 -35.201  < 2e-16 ***
## total_contacts    0.43612    0.05528   7.889 3.05e-15 ***
## self_init_cntcts  0.17495    0.05771   3.031 0.002435 **
## solicited_cntcts -0.37087    0.06741  -5.502 3.76e-08 ***
## mailq             0.17161    0.02466   6.959 3.44e-12 ***
## interest          0.67259    0.07500   8.967  < 2e-16 ***
## init_span        -0.06979    0.00525 -13.294  < 2e-16 ***
## int1rat           6.08524    1.26853   4.797 1.61e-06 ***
## int2rat           6.36424    1.13283   5.618 1.93e-08 ***
## hscrat           11.84376    0.42333  27.977  < 2e-16 ***
## campus_visit1     1.02076    0.09660  10.567  < 2e-16 ***
## campus_visit2     2.30075    0.61208   3.759 0.000171 ***
## premiere1         0.99552    0.09209  10.811  < 2e-16 ***
## instateY          0.33732    0.10491   3.215 0.001303 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15891.6  on 39364  degrees of freedom
## Residual deviance:  7363.6  on 39351  degrees of freedom
## AIC: 7391.6
##
## Number of Fisher Scoring iterations: 25
```

```
## -- Use tidymodel framework to fit and evaulate reduced model

uni_steprecipe <- recipe(enroll ~ total_contacts + self_init_cntcts + solicited_cntcts + mailq + interest +
          init_span + int1rat + int2rat + hscrat + campus_visit + premiere + instate, data = data_train) %>%
    step_impute_median(all_numeric()) %>%
    prep()

uni_steprecipe
```

```
## Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         12
##
## Training data contained 39365 data points and no missing data.
##
## Operations:
##
## Median imputation for total_contacts, self_init_cntcts, solicited_cnt... [trained]
```

```
# -- apply new recipe
bake_steptrain <- bake(uni_steprecipe, new_data = data_train)
bake_steptest  <- bake(uni_steprecipe, new_data = data_test)

logistic_step1 <- logistic_reg(mode = "classification") %>%
                  set_engine("glm") %>%
                  fit(enroll ~ ., data = bake_steptrain)



## -- check out your parameter estimates ...
tidy(logistic_step1) %>%
  mutate_at(c("estimate", "std.error", "statistic", "p.value"),round, 4)
```

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-6.2938	0.1788	-35.2005	0.0000
total_contacts	0.4361	0.0553	7.8887	0.0000
self_init_cntcts	0.1749	0.0577	3.0313	0.0024
solicited_cntcts	-0.3709	0.0674	-5.5018	0.0000

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
mailq	0.1716	0.0247	6.9585	0.0000
interest	0.6726	0.0750	8.9674	0.0000
init_span	-0.0698	0.0053	-13.2936	0.0000
int1rat	6.0852	1.2685	4.7971	0.0000
int2rat	6.3642	1.1328	5.6180	0.0000
hscrat	11.8438	0.4233	27.9773	0.0000

1-10 of 14 rows Previous **1** 2 Next

```
# -- training predictions from stepwise model
predict(logistic_step1, bake_steptrain, type = "prob") %>%
  bind_cols(.,predict(logistic_step1, bake_steptrain)) %>%
  bind_cols(.,bake_steptrain) -> scored_train_step1

head(scored_train_step1)
```

.pred_0	.pred_1	.pred_class	total_contacts	self_init_cntcts	solicited_cntcts	ma...
<dbl>	<dbl>	<fct>	<dbl>	<dbl>	<dbl>	<dbl>
0.9751860	0.0248140497	0	2	1	0	1
0.9937606	0.0062394078	0	1	0	1	5
0.9302196	0.0697803886	0	7	4	1	1
0.9993977	0.0006023133	0	1	0	1	5
0.9960810	0.0039189841	0	1	0	1	5
0.9949554	0.0050446364	0	1	0	0	2

6 rows | 1-9 of 16 columns

```
# -- testing predictions from stepwise model
predict(logistic_step1, bake_steptest, type = "prob") %>%
  bind_cols(.,predict(logistic_step1, bake_steptest)) %>%
  bind_cols(.,bake_steptest) -> scored_test_step1

head(scored_test_step1)
```

.pred_0 <dbl>	.pred_1 <dbl>	.pred_class <fct>	total_contacts <dbl>	self_init_cntcts <dbl>	solicited_cntcts <dbl>	ma... <dbl>
9.396685e-01	0.06033154	0	1	1	0	5
9.802095e-01	0.01979047	0	1	1	0	5
4.404356e-01	0.55956439	1	6	6	0	5
4.491213e-04	0.99955088	1	3	3	0	5
9.748635e-01	0.02513652	0	2	2	0	5
6.156524e-06	0.99999384	1	10	10	0	2

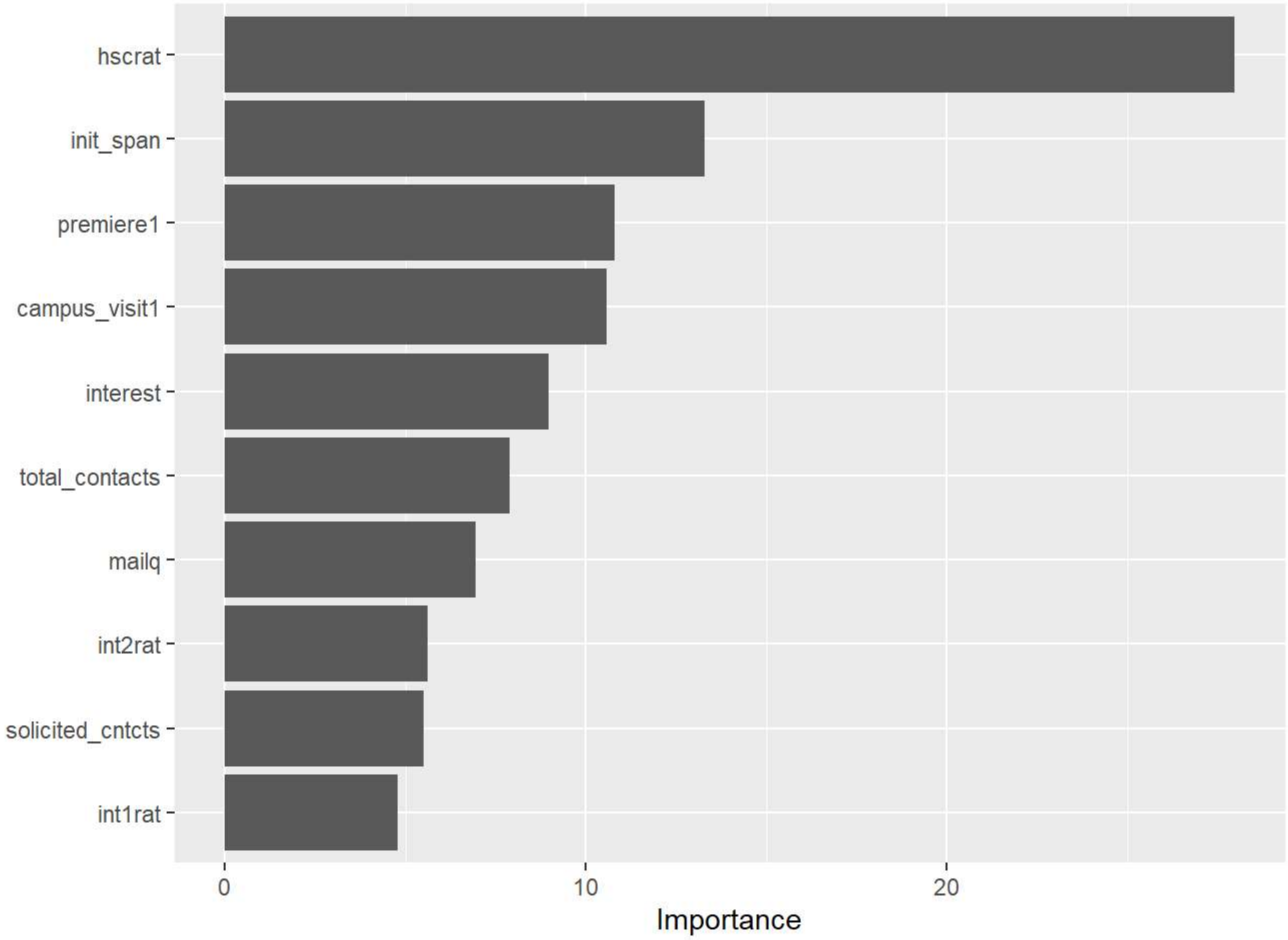
6 rows | 1-9 of 16 columns

```
# -- Evaluate Stepwise Model
# -- AUC: Train and Test
options(yardstick.event_first = FALSE)

scored_train_step1 %>%
  metrics(enroll, .pred_1, estimate = .pred_class) %>%
  mutate(part="training") %>%
  bind_rows( scored_test_step1 %>%
               metrics(enroll, .pred_1, estimate = .pred_class) %>%
               mutate(part="testing")
  ) %>%
  filter(.metric %in% c("accuracy", "roc_auc"))
```

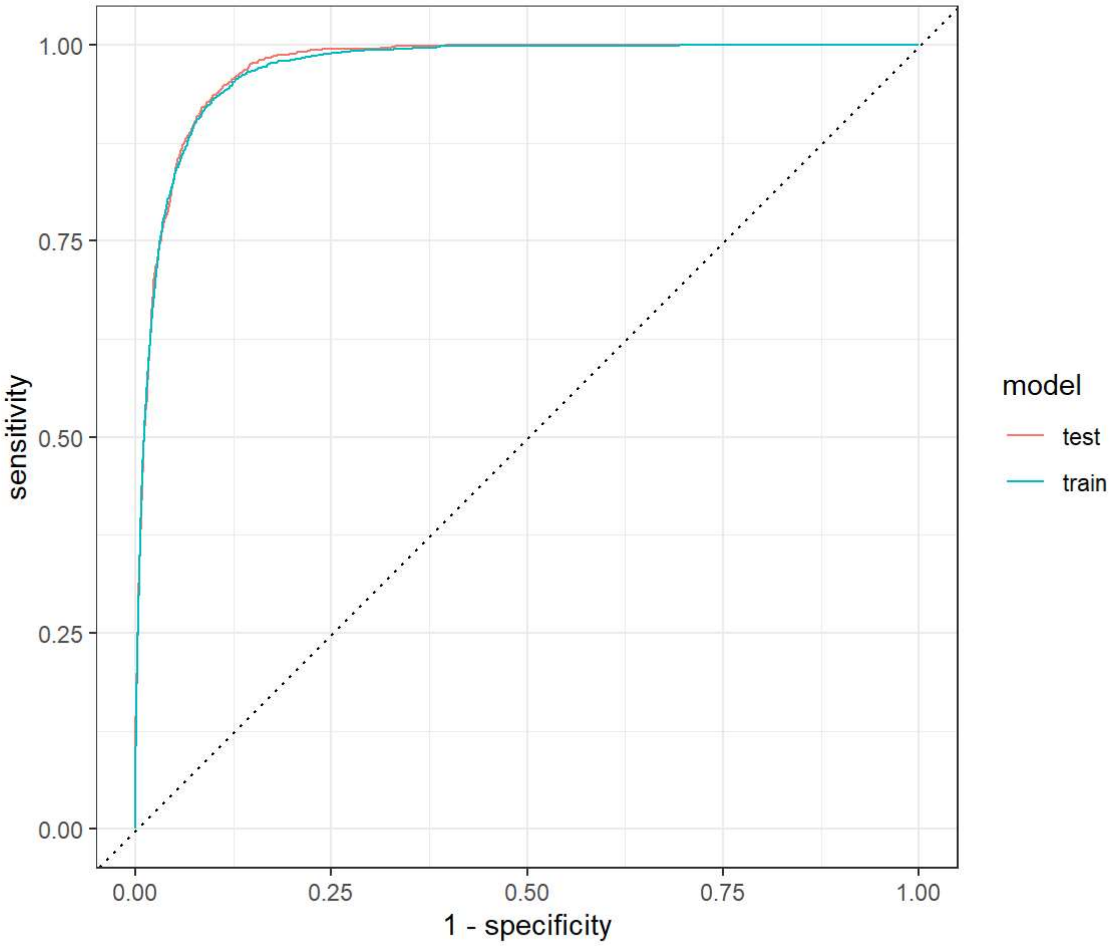
.metric <chr>	.estimator <chr>	.estimate <dbl>	part <chr>
accuracy	binary	0.9640290	training
roc_auc	binary	0.9705458	training
accuracy	binary	0.9640825	testing
roc_auc	binary	0.9725269	testing

4 rows

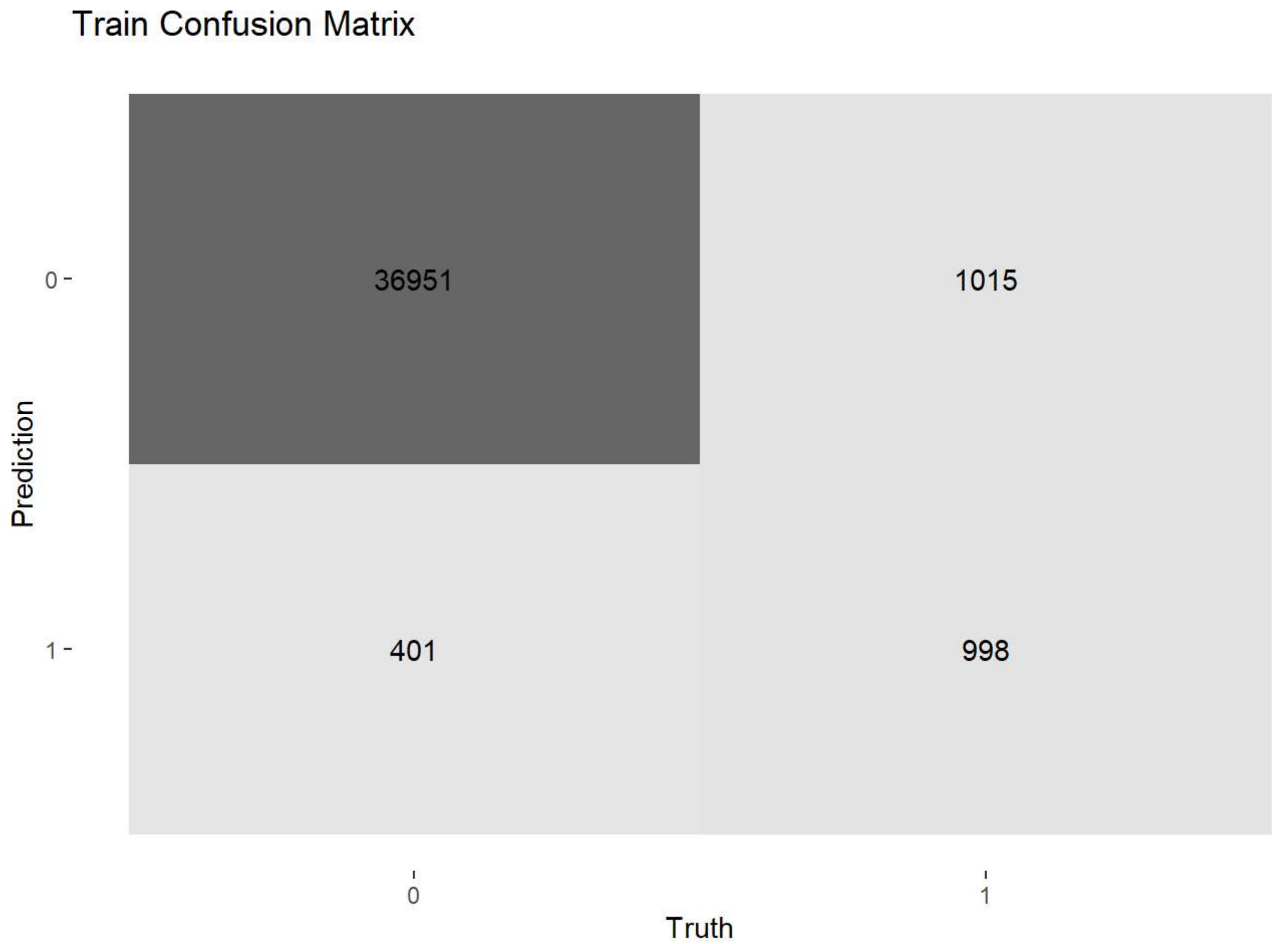
```
# -- Variable Importance top 10 features
model_1 %>%
  vip(num_features = 10)
```



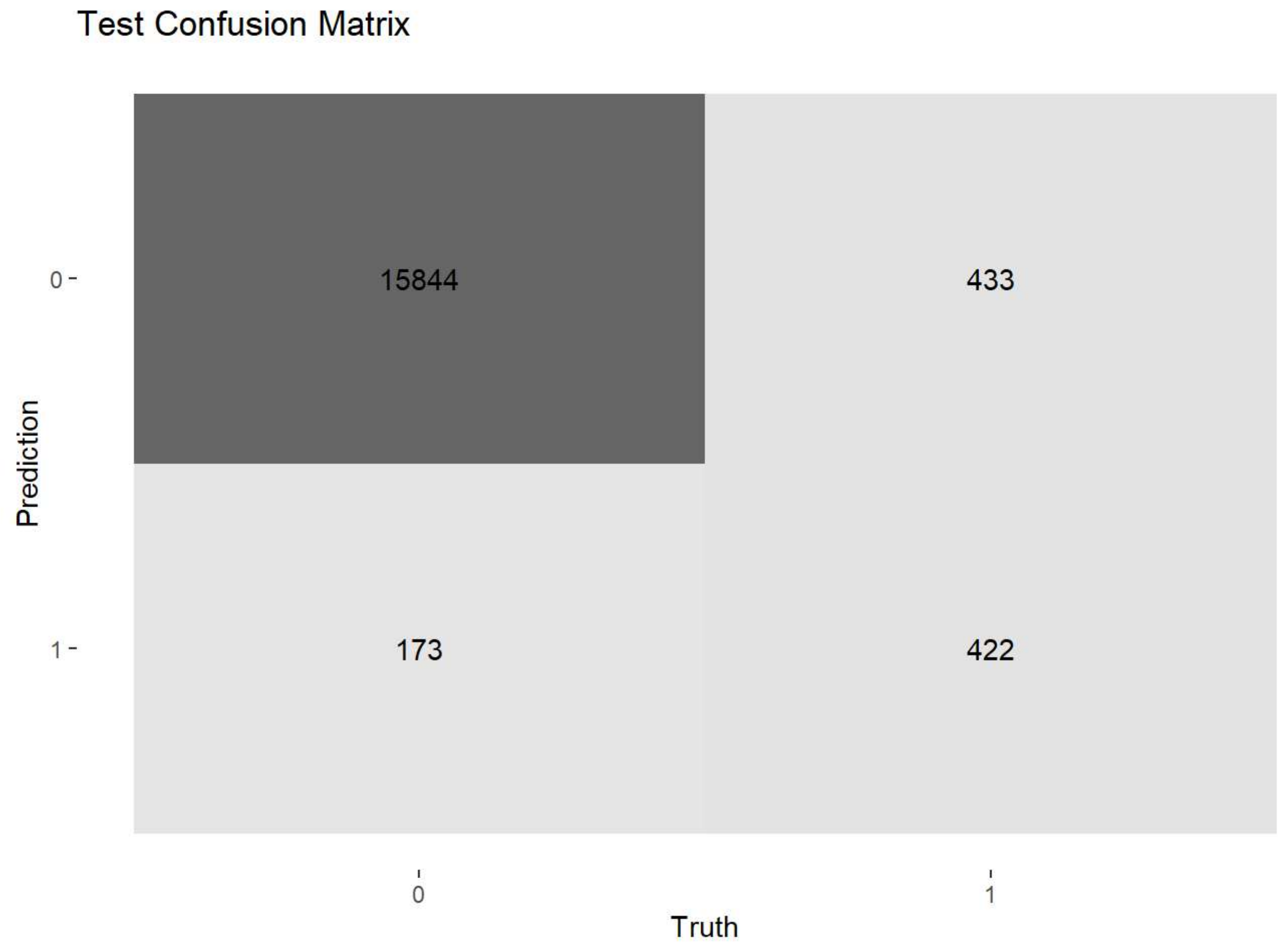

```
# -- ROC Charts
scored_train_step1 %>%
  mutate(model = "train") %>%
  bind_rows(scored_test_step1 %>%
              mutate(model="test")) %>%
  group_by(model) %>%
  roc_curve(enroll, .pred_1) %>%
  autoplot()
```




```
# -- Confustion Matricies
scored_train_step1 %>%
  conf_mat(enroll, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Train Confusion Matrix")
```




```
scored_test_step1 %>%
  conf_mat(enroll, .pred_class) %>%
  autoplot(type = "heatmap") +
  labs(title="Test Confusion Matrix")
```


Test Confusion Matrix
Prediction
0
1
15844
433
173
422
0
1
Truth