Measures of Classification Performance

We have a data set from our text book with "5822 real customer records. Each record consists of 86 variables, containing sociodemographic data (variables 1-43) and product ownership (variables 44-86). The sociodemographic data is derived from zip codes. All customers living in areas with the same zip code have the same sociodemographic attributes. Variable 86 (Purchase) indicates whether the customer purchased a caravan insurance policy."

```
head(Caravan)
```

```
## # A tibble: 6 x 86
##   MOSTYPE MAANTHUI MGEMOMV MGEMLEEF MOSHOOFD MGODRK MGODPR MGODOV MGODGE
##     <dbl>    <dbl>   <dbl>    <dbl>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1      33        1       3        2        8      0      5      1      3
## 2      37        1       2        2        8      1      4      1      4
## 3      37        1       2        2        8      0      4      2      4
## 4       9        1       3        3        3      2      3      2      4
## 5      40        1       4        2       10      1      4      1      4
## 6      23        1       2        1        5      0      5      0      5
## # ... with 77 more variables: MRELGE <dbl>, MRELSA <dbl>, MRELOV <dbl>,
## #   MFALLEEN <dbl>, MFGEKIND <dbl>, MFWEKIND <dbl>, MOPLHOOG <dbl>,
## #   MOPLMIDD <dbl>, MOPLLAAG <dbl>, MBERHOOG <dbl>, MBERZELF <dbl>,
## #   MBERBOER <dbl>, MBERMIDD <dbl>, MBERARBG <dbl>, MBERARBO <dbl>,
## #   MSKA <dbl>, MSKB1 <dbl>, MSKB2 <dbl>, MSKC <dbl>, MSKD <dbl>,
## #   MHHUUR <dbl>, MHKOOP <dbl>, MAUT1 <dbl>, MAUT2 <dbl>, MAUT0 <dbl>,
## #   MZFONDS <dbl>, MZPART <dbl>, MINKM30 <dbl>, MINK3045 <dbl>,
## #   MINK4575 <dbl>, MINK7512 <dbl>, MINK123M <dbl>, MINKGEM <dbl>,
## #   MKOOPKLA <dbl>, PWAPART <dbl>, PWABEDR <dbl>, PWALAND <dbl>,
## #   PPERSAUT <dbl>, PBESAUT <dbl>, PMOTSCO <dbl>, PVRAAUT <dbl>,
## #   PAANHANG <dbl>, PTRACTOR <dbl>, PWERKT <dbl>, PBROM <dbl>,
## #   PLEVEN <dbl>, PPERSONG <dbl>, PGEZONG <dbl>, PWAOREG <dbl>,
## #   PBRAND <dbl>, PZEILPL <dbl>, PPLEZIER <dbl>, PFIETS <dbl>,
## #   PINBOED <dbl>, PBYSTAND <dbl>, AWAPART <dbl>, AWABEDR <dbl>,
## #   AWALAND <dbl>, APERSAUT <dbl>, ABESAUT <dbl>, AMOTSCO <dbl>,
## #   AVRAAUT <dbl>, AAANHANG <dbl>, ATRACTOR <dbl>, AWERKT <dbl>,
## #   ABROM <dbl>, ALEVEN <dbl>, APERSONG <dbl>, AGEZONG <dbl>,
## #   AWAOREG <dbl>, ABRAND <dbl>, AZEILPL <dbl>, APLEZIER <dbl>,
## #   AFIETS <dbl>, AINBOED <dbl>, ABYSTAND <dbl>, Purchase <fct>
```

```
Caravan %>%
  count(Purchase)
```

```
## # A tibble: 2 x 2
##   Purchase     n
##   <fct>    <int>
## 1 No        5474
## 2 Yes        348
```

```
348/5474
```

```
## [1] 0.06357326
```

Only 6% of people in the data set purchased an insurance policy.

Logistic Regression Model

Let's try predicting Purchase status using all other explanatory variables. For today, we will look at training set performance; ordinarily, you'd look at performance on a test set.

```
library(caret)
```

```
## Loading required package: lattice
```

```
# The . in the right hand side of the formula says to use all other variables in the
# data set as predictors
fit <- train(
  form = Purchase ~ .,
  data = Caravan,
  family = "binomial", # this is an argument to glm; response is 0 or 1, binomial
  method = "glm", # method for fit; "generalized linear model"
  trControl = trainControl(method = "none")
)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(fit)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7047  -0.3711  -0.2450  -0.1588   3.2916
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.542e+02  1.116e+04   0.023  0.98183
## MOSTYPE      6.580e-02  4.624e-02   1.423  0.15468
## MAANTHUI    -1.832e-01  1.927e-01  -0.951  0.34157
## MGEMOMV     -2.696e-02  1.399e-01  -0.193  0.84723
## MGEMLEEF     2.096e-01  1.016e-01   2.063  0.03911 *
## MOSHOOFD    -2.767e-01  2.076e-01  -1.333  0.18247
## MGODRK      -1.142e-01  1.069e-01  -1.068  0.28535
## MGODPR      -1.910e-02  1.177e-01  -0.162  0.87112
## MGODOV      -1.618e-02  1.055e-01  -0.153  0.87818
## MGODGE      -6.817e-02  1.113e-01  -0.612  0.54024
## MRELGE       2.310e-01  1.566e-01   1.475  0.14031
## MRELSA       8.509e-02  1.466e-01   0.580  0.56169
## MRELOV       1.467e-01  1.562e-01   0.939  0.34759
## MFALLEEN    -8.291e-02  1.311e-01  -0.633  0.52702
## MFGEKIND    -1.154e-01  1.337e-01  -0.863  0.38813
## MFWEKIND    -8.140e-02  1.417e-01  -0.575  0.56561
## MOPLHOOG     9.717e-04  1.311e-01   0.007  0.99408
## MOPLMIDD    -9.077e-02  1.365e-01  -0.665  0.50605
## MOPLLAAG    -1.994e-01  1.376e-01  -1.449  0.14740
## MBERHOOG     8.883e-02  9.349e-02   0.950  0.34204
## MBERZELF     3.918e-02  9.897e-02   0.396  0.69219
## MBERBOER    -1.169e-01  1.104e-01  -1.059  0.28951
## MBERMIDD     1.353e-01  9.191e-02   1.472  0.14106
```

```
## MBERARBG     3.976e-02  9.067e-02   0.438  0.66104
## MBERARBO     9.954e-02  9.143e-02   1.089  0.27628
## MSKA         2.690e-02  1.035e-01   0.260  0.79502
## MSKB1       -8.801e-03  1.011e-01  -0.087  0.93064
## MSKB2        1.200e-02  9.081e-02   0.132  0.89485
## MSKC         9.016e-02  9.958e-02   0.905  0.36527
## MSKD        -2.468e-02  9.724e-02  -0.254  0.79967
## MHHUUR      -1.472e+01  8.140e+02  -0.018  0.98557
## MHKOOP      -1.469e+01  8.140e+02  -0.018  0.98561
## MAUT1        1.819e-01  1.514e-01   1.202  0.22953
## MAUT2        1.507e-01  1.371e-01   1.099  0.27162
## MAUT0        9.325e-02  1.436e-01   0.649  0.51603
## MZFONDS     -1.445e+01  9.359e+02  -0.015  0.98768
## MZPART      -1.451e+01  9.359e+02  -0.016  0.98763
## MINKM30      1.181e-01  1.006e-01   1.174  0.24039
## MINK3045     1.366e-01  9.650e-02   1.415  0.15694
## MINK4575     1.009e-01  9.667e-02   1.043  0.29678
## MINK7512     1.144e-01  1.027e-01   1.114  0.26513
## MINK123M    -1.607e-01  1.449e-01  -1.109  0.26738
## MINKGEM      9.214e-02  9.945e-02   0.927  0.35417
## MKOOPKLA     6.856e-02  4.642e-02   1.477  0.13966
## PWAPART      5.954e-01  3.901e-01   1.526  0.12693
## PWABEDR     -2.757e-01  4.635e-01  -0.595  0.55196
## PWALAND     -4.405e-01  1.035e+00  -0.425  0.67052
## PPERSAUT     2.306e-01  4.199e-02   5.491 4.01e-08 ***
## PBESAUT      1.215e+01  4.029e+02   0.030  0.97595
## PMOTSCO     -8.101e-02  1.147e-01  -0.706  0.48006
## PVRAAUT     -2.106e+00  2.557e+03  -0.001  0.99934
## PAANHANG     1.014e+00  9.371e-01   1.082  0.27917
## PTRACTOR     7.229e-01  4.278e-01   1.690  0.09107 .
## PWERKT      -5.525e+00  4.805e+03  -0.001  0.99908
## PBROM        2.170e-01  4.865e-01   0.446  0.65559
## PLEVEN      -2.382e-01  1.170e-01  -2.036  0.04173 *
## PPERSONG    -4.523e-01  2.094e+00  -0.216  0.82901
## PGEZONG      1.444e+00  1.029e+00   1.404  0.16033
## PWAOREG      8.239e-01  5.943e-01   1.386  0.16565
## PBRAND       2.401e-01  7.714e-02   3.113  0.00185 **
## PZEILPL     -8.658e+00  3.261e+03  -0.003  0.99788
## PPLEZIER    -1.886e-01  3.259e-01  -0.579  0.56289
## PFIETS       3.664e-01  8.325e-01   0.440  0.65985
## PINBOED     -1.068e+00  8.764e-01  -1.219  0.22301
## PBYSTAND    -1.676e-01  3.321e-01  -0.505  0.61373
## AWAPART     -9.293e-01  7.802e-01  -1.191  0.23364
## AWABEDR      4.197e-01  1.082e+00   0.388  0.69824
## AWALAND      2.762e-01  3.528e+00   0.078  0.93758
## APERSAUT    -3.902e-02  1.772e-01  -0.220  0.82566
## ABESAUT     -7.298e+01  2.417e+03  -0.030  0.97591
## AMOTSCO      2.418e-01  3.772e-01   0.641  0.52142
## AVRAAUT     -4.490e+00  1.078e+04   0.000  0.99967
## AAANHANG    -1.351e+00  1.687e+00  -0.801  0.42322
## ATRACTOR    -2.376e+00  1.524e+00  -1.559  0.11899
## AWERKT      -8.749e-01  9.682e+03   0.000  0.99993
## ABROM       -1.060e+00  1.549e+00  -0.684  0.49367
## ALEVEN       4.789e-01  2.245e-01   2.133  0.03291 *
```

```
## APERSONG     3.997e-01  4.329e+00   0.092  0.92644
## AGEZONG     -3.163e+00  2.706e+00  -1.169  0.24247
## AWAOREG     -3.212e+00  3.433e+00  -0.936  0.34939
## ABRAND      -4.118e-01  2.787e-01  -1.477  0.13956
## AZEILPL      1.047e+01  3.261e+03   0.003  0.99744
## APLEZIER     2.516e+00  1.010e+00   2.490  0.01276 *
## AFIETS       2.318e-01  5.699e-01   0.407  0.68420
## AINBOED      1.947e+00  1.412e+00   1.378  0.16812
## ABYSTAND     1.078e+00  1.103e+00   0.977  0.32870
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2635.5  on 5821  degrees of freedom
## Residual deviance: 2243.5  on 5736  degrees of freedom
## AIC: 2415.5
##
## Number of Fisher Scoring iterations: 17
```

Confusion Matrix

```
preds <- predict(fit) # if you were doing test set predictions, you'd have to specify newdata here!

table(Caravan$Purchase, preds)
```

```
##      preds
##         No  Yes
##   No  5466    8
##   Yes  341    7
```

- Since I provided the observed response `Caravan$Purchase` as the first argument to `table` and `preds` as the second argument to `table`:
  - The rows have observed values
  - The columns have predicted values
  - For example, there were 8 people who were predicted to make a purchase who did not actually make a purchase

There is a bewildering array of different summaries of classification performance that are often calculated based on the confusion matrix. Here are some:

Classification Accuracy: Out of all predictions, what proportion were correct?

- We prefer higher classification accuracy
- How would this be calculated based on the confusion matrix above?

Classification Error Rate: Out of all predictions, what proportion were incorrect?

- We prefer lower classification error rate
- How would this be calculated based on the confusion matrix above?

False Positive Rate: Out of the "negative" cases, how many did we incorrectly predict to be "positive"?

- We prefer lower false positive rate
- Synonyms: Type I Error, 1 - Specificity
- How would this be calculated based on the confusion matrix above?

True Positive Rate: Out of the "positive" cases, how many did we correctly predict to be "positive"?

- We prefer higher true positive rate
- Synonyms: 1 - Type II Error, Power, Sensitivity, Recall
- How would this be calculated based on the confusion matrix above?

Positive Predictive Value: Out of our "positive" predictions, how many were correct?

- We prefer higher positive predictive value
- Synonyms: Precision, 1 - False Discovery Proportion
- How would this be calculated based on the confusion matrix above?

Negative Predictive Value: Out of our "negative" predictions, how many were correct?

- We prefer higher negative predictive value
- How would this be calculated based on the confusion matrix above?

Different classification threshold changes Accuracy/etc.

- Maybe the company wants to use predictions to follow up with any potential customers.
- They might be more interested in ensuring that most potential customers are predicted to Purchase, even if this means they place more unnecessary calls to people who don't end up purchasing.
- We can achieve this for example by predicting "Yes" if the model's estimated probability of purchasing is at least 0.05:

```
# if you were doing test set predictions, you'd have to specify newdata here!
preds_prob <- predict(fit, type = "prob")[["Yes"]]
head(preds_prob)
```

```
## [1] 0.09404792 0.01289249 0.06427098 0.07555249 0.01749071 0.01919232
```

```
preds <- ifelse(preds_prob > 0.05, "Yes", "No")
head(preds)
```

```
## [1] "Yes" "No"  "Yes" "Yes" "No"  "No"
```

```
table(Caravan$Purchase, preds)
```

```
##      preds
##         No  Yes
##   No  3514 1960
##   Yes   81  267
```

With this threshold, what is our true positive rate?

With this threshold, what is our false positive rate?

ROC and AUC

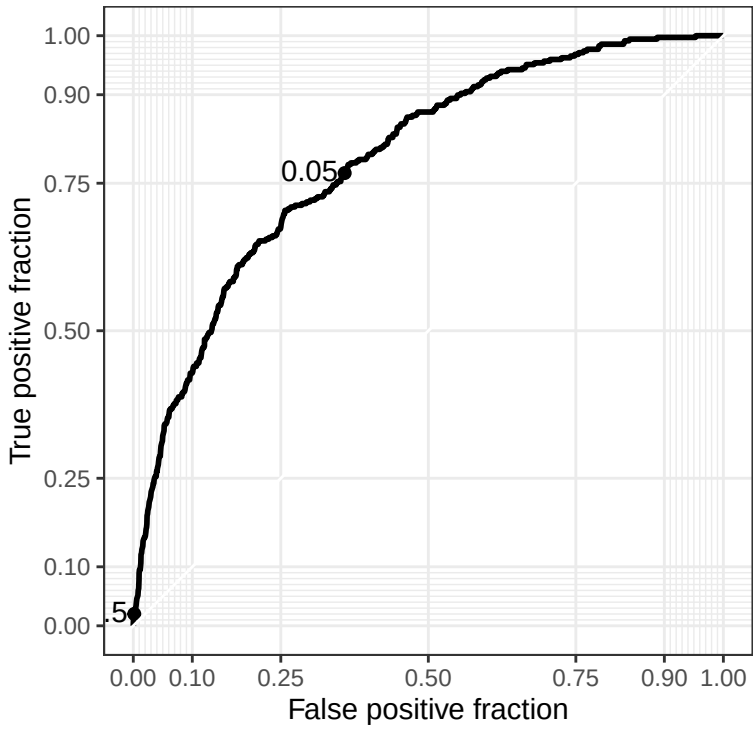
- As we vary our probability threshold for classification, how do our false positive fraction and true positive fraction change?

Code from https://stackoverflow.com/questions/31138751/roc-curve-from-training-data-in-caret

```
library(plotROC)

# add two variables to the data frame: predicted probability of being in class 1
# and a 0/1 version of the response variable (required by geom_roc)
Caravan <- Caravan %>%
  mutate(
    f1_hat = predict(fit, type = "prob")[["Yes"]],
    Purchase_01 = ifelse(Purchase == "Yes", 1, 0)
  )

p <- ggplot(data = Caravan, mapping = aes(m = f1_hat, d = Purchase_01)) +
  geom_roc(cutoffs.at = c(0.05, 0.5), cutoff.labels = c("0.05", "0.5")) +
  coord_equal() +
  style_roc()
p
```



The Area Under the ROC Curve (AUC) is between 0 and 1. Is it better if it's closer to 0 or closer to 1?

```
calc_auc(p)
```

```
##   PANEL group       AUC
## 1     1    -1 0.7902837
```

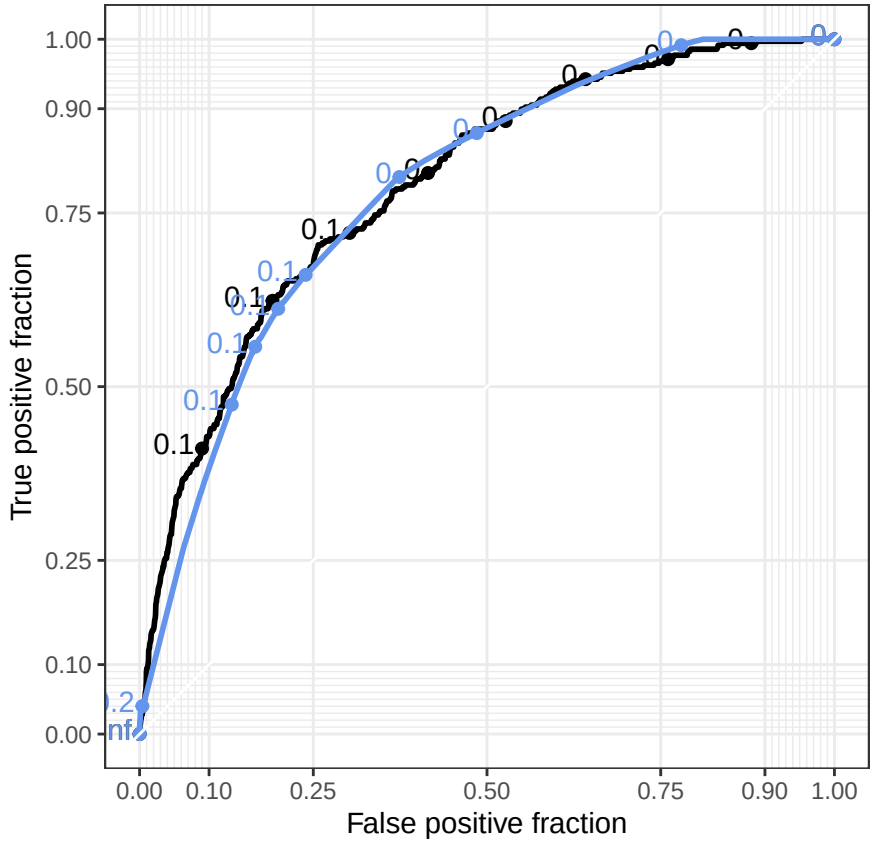
Adding on ROC for a KNN fit using just a few variables:

```
knn_fit <- train(
  form = Purchase ~ MGEMLEEF + PPERSAUT + PBRAND + APLEZIER,
  data = Caravan,
  preProcess = "scale",
  method = "knn",
  trControl = trainControl(method = "none")
)

Caravan <- Caravan %>%
  mutate(
    knn_f1_hat = predict(knn_fit, type = "prob")[["Yes"]]
  )

p <- ggplot(data = Caravan) +
  geom_roc(mapping = aes(m = f1_hat, d = Purchase_01)) +
  geom_roc(mapping = aes(m = knn_f1_hat, d = Purchase_01), color = "cornflowerblue") +
  coord_equal() +
  style_roc()

p
```



```r
# auc for each model separately
p_logistic_only <- ggplot(data = Caravan) +
    geom_roc(mapping = aes(m = f1_hat, d = Purchase_01))
p_knn_only <- ggplot(data = Caravan) +
    geom_roc(mapping = aes(m = knn_f1_hat, d = Purchase_01))
calc_auc(p_logistic_only)
```

```
##   PANEL group       AUC
## 1     1    -1 0.7902837
```

```r
calc_auc(p_knn_only)
```

```
##   PANEL group       AUC
## 1     1    -1 0.7836284
```

Log Score

- The log score is the log of the probability assigned do the observed data by the model:

$$\log\{P(Y_1 = y_1, Y_2 = y_2, \ldots, Y_n = y_n | x_1, \ldots, x_n)\} = \log\left\{\prod_{i=1}^{n} P(Y_i = y_i | x_i)\right\}$$
$$= \sum_{i=1}^{n} \log\left\{P(Y_i = y_i | x_i)\right\}$$

- Since log is an increasing function, a higher probability of observed data gives a higher log score.

```
Caravan <- Caravan %>%
  mutate(
    est_f_obs = ifelse(Purchase == "Yes", f1_hat, 1 - f1_hat),
    knn_est_f_obs = ifelse(Purchase == "Yes", knn_f1_hat, 1 - knn_f1_hat)
  )

sum(log(Caravan$est_f_obs))
```

```
## [1] -1121.743
```

```
sum(log(Caravan$knn_est_f_obs))
```

```
## [1] -1123.29
```