# POLISCI 450c: Regularised Regression

Apoorva Lal

April 23, 2021

# On the agenda today:

**PCA regression Ridge, LASSO, Elastic Net**

**Naive Bayes**

# But First. . . .

**Questions?**

# Major Topics in Machine Learning

- Supervised Learning: "predict $Y$ using $X$"
  - e.g., predict conflict or election outcomes
  - Incomplete list: linear regression, GLMs, GAMs, random forests, **ridge regression, LASSO**, boosted trees, local linear regression (loess), SVMs, neural networks, ensembles, **naive bayes**
- Unsupervised Learning: "characterize $X$"
  - e.g., find similar individuals (clusters, topics), find a major dimension of variation ("common ideological structure")
  - Incomplete list: **principal components analysis** (PCA), clustering, topic models, mixture models, factor analysis, IRT models, wordfish, scaling, autoencoders, multinomial inverse regression, latent class models

# Regularized Regression - The Problem

- Familiar world: use $x_i$ to predict $y_i$
- Would like to solve:

$$(\hat{\alpha}, \hat{\beta}) = argmin_{\alpha,\beta} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, \alpha + x_i \beta) \right\}$$

- Sometimes $n \approx p$ or $n \leq p \implies$ *unbiased estimators* will **badly overfit to training data**
- **e.g., predict news source (foxnews vs. nytimes) from text**

# Regularized Regression - The Problem

- Modify problem:

$$(\hat{\alpha}, \hat{\beta})(\lambda) = argmin_{\alpha,\beta} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, \alpha + x_i\beta) + \lambda \, \text{Penalty}(\beta) \right\}$$

- **Ridge**: L2 Norm - $\text{Penalty}(\beta) = \|\beta\|_2^2 = \sum_{j=1}^{J} \beta_j^2$
- **LASSO**: L1 Norm - $\text{Penalty}(\beta) = \|\beta\|_1 = \sum_{j=1}^{J} |\beta_j|$
- **Elastic Net**: Combination - $\text{Penalty}(\beta) = \sum_{j=1}^{J} (1-\alpha)\frac{1}{2}\beta_j^2 + \alpha|\beta_j|$

# Regularisation in a picture

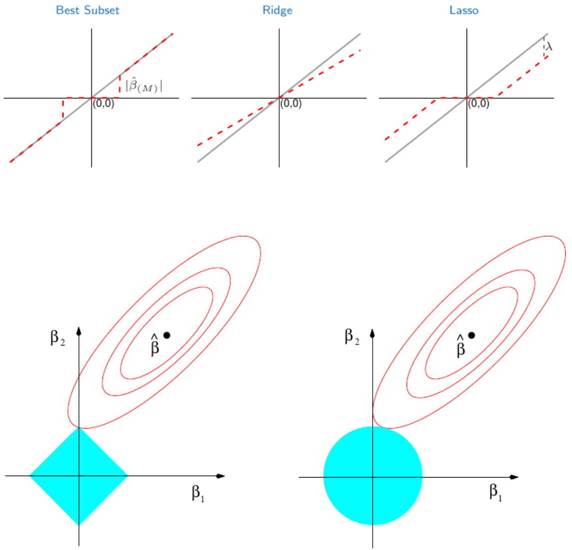



**FIGURE 3.11.** *Estimation picture for the lasso (left) and ridge regression*

# Benefits of Regularized Regression

- Modify problem:

$$(\hat{\alpha}, \hat{\beta})(\lambda) = argmin_{\alpha,\beta} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{Loss}(y_i, \alpha + x_i\beta) + \lambda \, \text{Penalty}(\beta) \right\}$$

- **Ridge & LASSO**: bias $\hat{\beta}(\lambda)$ towards zero to prevent overfitting
- **LASSO**: force many $\hat{\beta}(\lambda)_k$s to equal zero
- **LASSO** tends to find predictors that are most relevant (no guarantees though)
- Choose $\lambda$ via cross-validation

# Data for Today

- **Research Question**: Do less-developed countries create separate spaces for themselves in international relations?
- International Organization (IO) membership for 183 countries in 2000
- **Restated RQ:**: We can ask: is IO membership diagnostic of a country's income?
- **Challenge:** IO membership is very high-dimensional

# Data for Today

- Membership in IOs for 183 countries in 2000
- 125-dimensional vector $\{0, 1, ..., 1, 0\}$ for each country
- Dependent variable: `rich`: is GDP per capita above median?

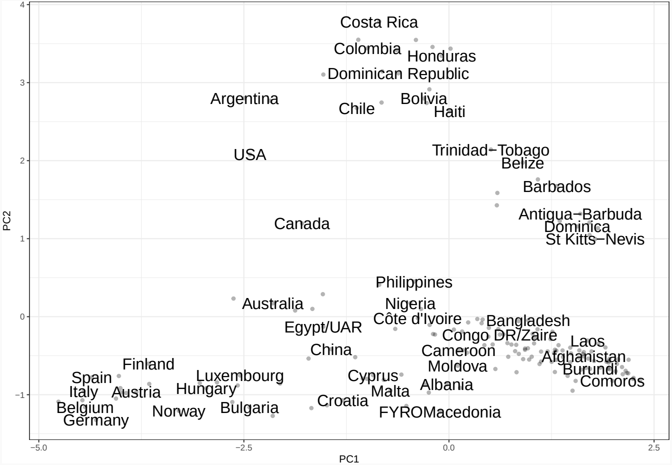
# Data for Today

```
# Load data on IO membership from Correlates of War
library(readstata13)
df <- read.dta13('ios.dta')
X <- df[,4:125] %>% as.matrix()
Y <- df %>% pull(rich)
# Take a look at the data
glimpse(df)
```

```
# Rows: 183
# Columns: 125
# $ country     <chr> "Afghanistan", "Angola", "Albania", "United Arab Emirates"~
# $ rich        <dbl> 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1~
# $ gdppc       <dbl> 1734.7, 5725.3, 12316.1, 66616.0, 18288.2, 9177.7, 23840.7~
# $ aalco       <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0~
# $ aaro        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0~
# $ acssrb      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0~
# $ afpu        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0~
# $ amco        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0~
# $ anzus       <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
# $ aopu        <int> 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0~
# $ apfic       <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0~
# $ apo         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0~
# $ arpu        <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0~
# $ asecna      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0~
# $ afdb        <int> 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0~
# $ bcsc        <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0~
# $ benelux     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0~
# $ bescc       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

# PCA Regression

```r
pcaout <- prcomp(X)
toplot <- tibble(country = df$country, PC1 = pcaout$x[,1], PC2 = pcaout$x[,2])
ggplot(toplot, aes(x = PC1, y = PC2))+geom_point(alpha = 0.3)+
  geom_text(aes(label = country),check_overlap=T,size = 6)+theme_bw()
```



# PCA Regression

```
pca_reg <- lm(df$rich~ pcaout$x[,1] + pcaout$x[,2])
summary(pca_reg)
```

```
#
# Call:
# lm(formula = df$rich ~ pcaout$x[, 1] + pcaout$x[, 2])
#
# Residuals:
#    Min     1Q Median     3Q    Max
# -0.701 -0.344 -0.190  0.416  0.814
#
# Coefficients:
#               Estimate Std. Error t value             Pr(>|t|)
# (Intercept)    0.47541    0.03246   14.65 < 0.0000000000000002 ***
# pcaout$x[, 1] -0.13587    0.01807   -7.52      0.0000000000025 ***
# pcaout$x[, 2] -0.00977    0.02600   -0.38                 0.71
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.439 on 180 degrees of freedom
# Multiple R-squared:  0.239,   Adjusted R-squared:  0.231
# F-statistic: 28.3 on 2 and 180 DF,  p-value: 0.00000000002
```

# PCA Regression

```
x_reg   <- pcaout$rotation[,1:2]%*%pca_reg$coef[2:3]
tibble(IOs = rownames(x_reg)[1:5], coefs = x_reg[1:5])
```

```
# # A tibble: 5 x 2
#   IOs         coefs
#   <chr>       <dbl>
# 1 aalco   -0.00157
# 2 aaro     0.0000329
# 3 acssrb   0.00584
# 4 afpu    -0.00163
# 5 amco    -0.00150
```

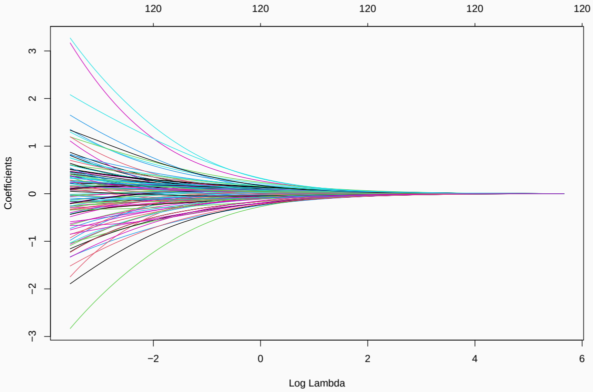
# PCA Regression - Performance

```r
dat <- toplot %>% mutate(rich = Y)
# get predicted values using cross validation
Foldsize <- ceiling(nrow(dat)/10)
# fold var
set.seed(12345)
dat$fold <- sample(rep(1:10,each=Foldsize),nrow(dat),replace=F)
dat$pred_pca <- NA
for(i in 1:10){
  # fit model
  temp_pca <- lm(data = dat[dat$fold!=i,],
                 rich~PC1+PC2)
  dat$pred_pca[dat$fold==i] <- predict(temp_pca,
                                       newdata=dat[dat$fold==i,],
                                       type = "response")
}
#Mean Squared Error
mses <- cbind(tibble(Measure=c('MSE')), PCA = c(round(mean((dat$pred_pca-Y)^2),4)))
knitr::kable(mses, digits = 3)
```

| Measure | PCA |
|---|---|
| MSE | 0.195 |

# Ridge

```
library(glmnet)
ridge <- glmnet(x = X, y = Y,
  family = "binomial", alpha = 0)
plot(ridge, xvar = "lambda")
```

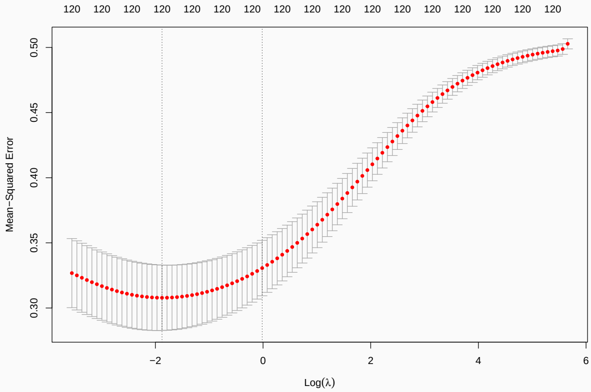


# Ridge - Cross-Validated $\lambda$

```
ridge <- cv.glmnet(x = X, y = Y,
  family = "binomial", alpha = 0,
  type.measure="mse")
ridge$lambda.min # Cross- Validation Selected lambda
```

```
# [1] 0.1531
```

```
plot(ridge)
```



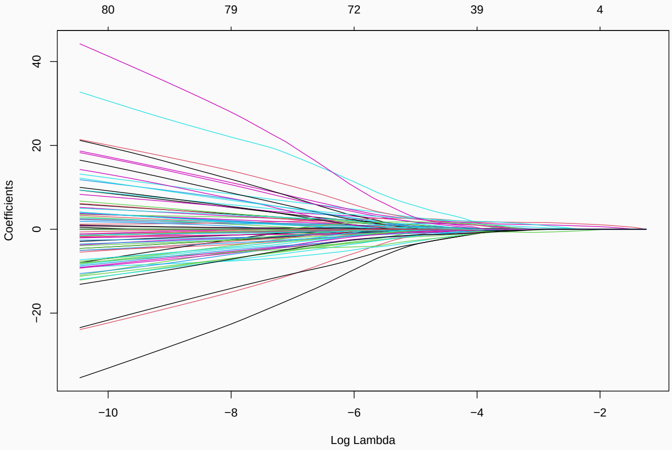
# Ridge - Cross-Validated $\lambda$

```r
#Mean Squared Error (i.e. Brier Score)
mses <- cbind(mses, Ridge = min(ridge$cvm))
knitr::kable(mses, digits = 3)
```

| Measure | PCA | Ridge |
|---|---|---|
| MSE | 0.195 | 0.308 |

# LASSO

```
lasso <- glmnet(x = X, y = Y,
  family = "binomial", alpha = 1)
plot(lasso, xvar = "lambda")
```

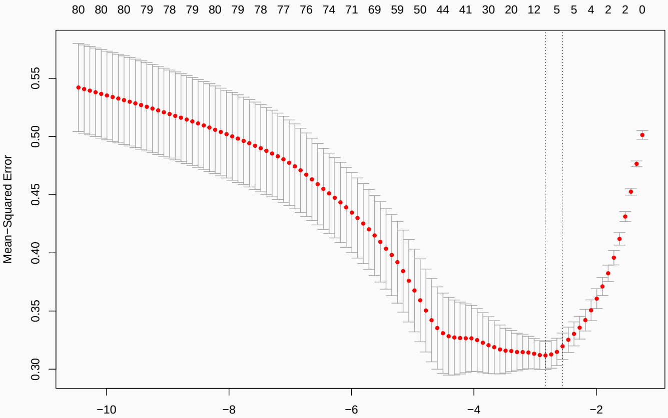


# LASSO - Cross-Validated $\lambda$

```
lasso <- cv.glmnet(x = X, y = Y,
  family = "binomial", alpha = 1,
  type.measure="mse")
lasso$lambda.min
```

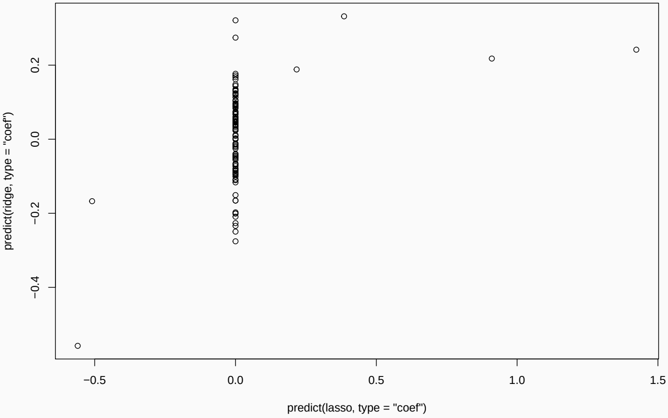
```
# [1] 0.059
```

```
plot(lasso)
```



# LASSO Shrinks Coefficients to zero

```
plot(predict(lasso, type = "coef"),
     predict(ridge, type = "coef"))
```



# LASSO - Cross-Validated $\lambda$

```
#Mean Squared Error (i.e. Brier Score)
mses <- cbind(mses, LASSO = min(lasso$cvm))
knitr::kable(mses, digits = 3)
```

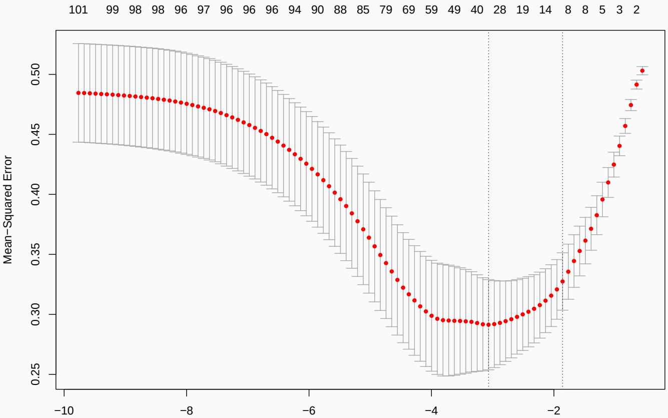
| Measure | PCA | Ridge | LASSO |
|---|---|---|---|
| MSE | 0.195 | 0.308 | 0.312 |

# Elastic Net

```r
elasticnet <- cv.glmnet(x = X, y = Y,
  family = "binomial", alpha = 0.5,
  type.measure="mse")
elasticnet$lambda.min
```

```
# [1] 0.04654
```

```r
plot(elasticnet)
```



# Elastic Net

```r
#Mean Squared Error (i.e. Brier Score)
mses <- cbind(mses, ElasticNet = min(elasticnet$cvm))
knitr::kable(mses, digits = 3)
```

| Measure | PCA | Ridge | LASSO | ElasticNet |
|---|---|---|---|---|
| MSE | 0.195 | 0.308 | 0.312 | 0.291 |

# Discrimination Ability

```r
# (e)
library('pROC'); library('plotROC')
# get predicted values using cross validation
Foldsize <- ceiling(nrow(dat)/10)
# fold var
fold       <- sample(rep(1:10,each=Foldsize), nrow(dat),replace=F)
yhat_pca   <- NA; yhat_ridge <- NA; yhat_lasso <- NA; yhat_enet  <- NA
```

# Discrimination Ability

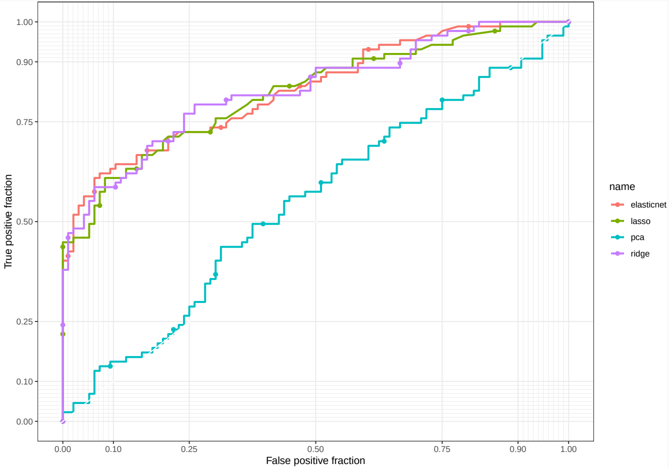
```r
for(i in 1:10){
  # Split
  y_train <- Y[fold!=i]; x_train <- X[fold!=i, ]; x_test <- X[fold==i, ]
  pca1_train <- pcaout$x[fold!=i,1];  pca2_train <- pcaout$x[fold!=i,2]
  pca1_test <- pcaout$x[fold==i,1];   pca2_test <- pcaout$x[fold==i,2]
  # Fit
  pca_reg <- lm(y_train~ pca1_train + pca2_train)
  ridge <- cv.glmnet(x = x_train, y = y_train,
                     family = "binomial", alpha = 0, type.measure="mse")
  lasso <- cv.glmnet(x = x_train, y = y_train,
                     family = "binomial", alpha = 1, type.measure="mse")
  elasticnet <- cv.glmnet(x = x_train, y = y_train,
                     family = "binomial", alpha = 0.5, type.measure="mse")
  # Predict
  yhat_pca[fold==i]   <- predict(pca_reg,newdata=cbind.data.frame(pca1_test,pca2_test))
  yhat_ridge[fold==i] <- predict(ridge, newx = x_test, s="lambda.min")
  yhat_lasso[fold==i] <- predict(lasso, newx = x_test, s="lambda.min")
  yhat_enet[fold==i]  <- predict(elasticnet, newx = x_test, s="lambda.min")

}

dd <- data.frame(y=rep(Y,4), pred=c(yhat_pca,yhat_ridge,yhat_lasso,yhat_enet),
        name=rep(c("pca","ridge",'lasso','elasticnet'), each=nrow(dat)))
```

# Discrimination Ability

```
ggroc <- ggplot(dd, aes(d = y, m = pred, color = name)) +
     geom_roc(labels = FALSE) + style_roc()
ggroc
```



# Discrimination Ability

```
calc_auc(ggroc)
```

```
#   PANEL group    AUC
# 1     1     1 0.8302
# 2     1     2 0.8221
# 3     1     3 0.5453
# 4     1     4 0.8300
```

# Caret

```
library(caret)
```

# Caret

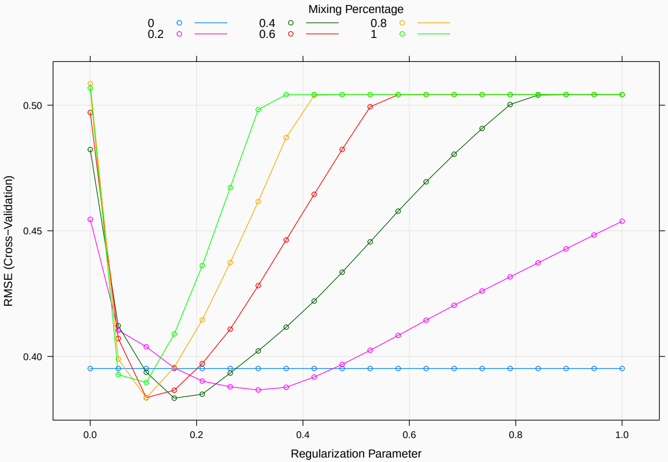
```r
# Separate train and test sets
inTraining <- createDataPartition(df$rich, p = .5,
                list = FALSE)
train <- df[inTraining, ]
test <- df[-inTraining, ]
# Can feed caret a grid of parameter values to
# cross validate over
myGrid <- expand.grid( alpha = seq(0,1,0.2),
                       lambda = seq(0.0001, 1, length = 20) )
```

# Caret

```
model <- train(
  x = train[,4:125], y=train$rich,
  method = "glmnet",
  tuneGrid = myGrid,
  summaryFunction = twoClassSummary,
  trControl = trainControl( method = "cv", number = 10, verboseIter = FALS
)
```

# Caret - plot of relative performance

```r
# Plot results
plot(model)
```



# Caret - best tuned parameters

```
#Best Chosen parameters
model$bestTune
```

```
#    alpha lambda
# 44   0.4  0.158
```

# Caret - best model's coefficients

```
coefs_caret <- coef(model$finalModel, model$bestTune$lambda)
coefs_caret[1:20]
```

```
#  [1] 0.32103 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.0
# [10] 0.00000 0.05512 0.00000 0.00000 0.00000 0.00000 0.00000 0.15203 0.2
# [19] 0.00000 0.00000
```

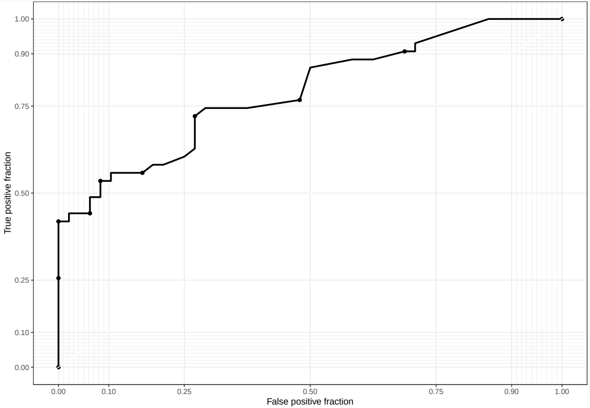
# Caret - ROC

```
# Predict probabilites, calculate AUC, and draw ROC
prediction_p <- predict(model, test[,c(2,4:125)])
dd <- data.frame(y=test$rich,
        pred=prediction_p,
        name=rep(c("final model"),
          each=nrow(test)))
ggroc <- ggplot(dd, aes(d = y, m = pred, label = name)) +
     geom_roc(labels = FALSE) + style_roc()
calc_auc(ggroc)
```

```
#   PANEL group    AUC
# 1     1    -1 0.7888
```

# Caret - ROC

```
ggroc
```



# Naive Bayes

- Can we apply this to our data?
- Predict “label” i.e. income level
- What assumption are we making about IO membership

# Naive Bayes

- Recall, we have two categories $C_i \in \{rich, notrich\}$
- $\mathbf{x}_i$ is a membership vector
- We want to predict, for each $i$:
  - $C_i = C_{max} = argmax_k p(C_k|\mathbf{x}_i)$
- Applying Bayes' Rule, we get:
  - $C_i = C_{max} = argmax_k \{\frac{p(C_k)p(\mathbf{x}_i|C_k)}{p(\mathbf{x}_i)}\}$
  - $C_i = C_{max} = argmax_k \{p(C_k)p(\mathbf{x}_i|C_k)\}$

# Naive Bayes

- Using our data, $p(C_i = rich) = \frac{No.RichCountries}{No.Countries} = 0.5$
- Under the independence assumption,
    - we can factor the likelihood for the $J$ international organizations:
    - $p(\mathbf{x}_i | rich) = \prod_{j=1}^{J} p(x_{ij} | rich)$
    - $= \prod_{j=1}^{J} \frac{\text{No. Countries that are rich AND have } x_j = x_{ij}}{No.RichCountries}$
    - $\approx \prod_{j=1}^{J} \frac{(\text{No. Countries that are rich AND have } x_j = x_{ij}) + 1}{No.RichCountries + 2}$

# Naive Bayes - Code

```
library(e1071)
model <- naiveBayes(rich ~ .,
                    data = train%>%dplyr::select(-c(gdppc)),
                    laplace = 3)

preds <- predict(model, test, 'raw')
preds <- as.numeric(preds[,2])

dd <- data.frame(y=test$rich,
        pred=preds,
        name=rep(c("Naive Bayes"),
          each=nrow(test)))
ggroc <- ggplot(dd, aes(d = y, m = preds, label = name)) +
     geom_roc(labels = FALSE) + style_roc()

# ROC from Naive Bayes
calc_auc(ggroc)
```

```
#   PANEL group    AUC
# 1     1    -1 0.7466
```

```
# Brier Score (MSE) from Naive Bayes
mean((test$rich - preds)^2)
```

```
# [1] 0.3175
```

# Naive Bayes

`ggroc`