HW2 - Joe Risi - PLSC 597: Machine Learning

Introduction and Replication - Question 1

Harvard Dataverse Link - https://doi.org/10.7910/DVN/OSSD8O
JSTOR Stable Link to Paper - https://www.jstor.org/stable/42940606
Link to github repository - https://github.com/jrisi256/schoolwork/tree/main/plsc597/hw2/src

Introduction - Is Immigration a Racial Issue? Anglo Attitudes on Immigration Policies in a Border County.

- The purpose of the study is to assess the association between Anglo aversion to Latinos and a variety of other factors including: physical proximity to Latinos, contact with ethnic minorities, and expressed preferences for immigration policies.
- The underlying hypothesis argues immigration policy preferences are strongly influenced by racial resentment rather than other considerations like economic anxiety.
- Other hypotheses argue Anglos living in neighborhoods with larger pro portions of Latinos will harbor more restrictive attitudes on immigration, but Anglos who interact more frequently with minorities will harbor less restrictive attitudes on immigration.

Data and Methods

- Data was collected through a telephone survey using random-digit-dial procedures in San Diego County, California in 2005 - 2006. Data were weighted in all regressions to represent San Diego County demographic characteristics for Anglos based on US Census estimates. No significant differences appeared in conclusions when analyses were replicated using unweighted values.
- **Dependent Variable**: The dependent variable of interest was survey respondents' answers to the amnesty question: "As you may know, in 1986 the US Congress passed the Immigration Reform and Control Act, which granted amnesty to nearly 2 million persons who had lived continuously in this country for four or more years without proper documentation. This amnesty law allowed these immigrants to remain here as permanent residents and to apply for US citizenship. At this time, do you think repeating this amnesty program would be a good thing?" 0 = bad idea, 1 = good idea.
- **Independent Variables**
  - Respondent aversion to Latinos (attitude about Latinos): Measured using the Bogardus scale which is a composite index to detect racial attitudes. Recoded as a dummy variable where 1 means aversion was detected and 0 means no aversion was detected.
  - Latino context (concentration in the same Census tract): The natural log of the percent of Latino residents within each respondent's Census tract.
  - Reported contact with minorities: Composite scale summarizing interactions respondents had with Latinos.
- **Controls**
  - Personal financial situation; "In terms of your personal economic situation, would you say that it has improved, remained the same, or gotten worse over the past 12 months?"
  - Family Income.
  - Age.
  - Education.
  - Gender.
  - Political Ideology: "Would you consider yourself conservative, moderate, or liberal?" Liberals were coded as low and conservatives as high.

- **Codebook**
  - Amnesty = amnesty
  - Latino Aversion = dishis2
  - Latino context = pcthis2
  - Latino contact = contact
  - Personal financial situation = retecon
  - Political ideology = idea
  - Education = edu2
  - Famliy Income = income
  - Age = age
  - Sex = male
- **Results**
  - Age, gender, and personal financial situation are not statistically significant. In contrast to the their hypothesis, Latino aversion was not statistically significant either.
  - Increased minority contact was positively associated with amnesty however it was only marginally significant.
  - Decreased income was negatively associated with amnesty however it was only marginally significant.
  - Latino context was negatively and statistically significantly associated with amnesty meaning the more Latinos living in your Census tract the more likely you were to view amnesty as a bad thing.
  - Increased education was positively and statistically significantly associated with amnesty.
  - As one became more conservative, the probability of supporting amnesty decreased. Highly statistically significant.

Packages

```
library(mlr)
library(here)
library(dplyr)
library(purrr)
library(tidyr)
library(caret)
library(foreign)
library(parallel)
library(neuralnet)
library(parallelMap)
```

Read in the data

```
ssq <- foreign::read.dta(here("hw1/data/ssq.dta"))
```

Light data cleaning

Select only relevant columns for the regression and remove all rows with a missing value on any of the variables. Turn all variables into numeric values (to match the results from the paper).

```
X2 <-
    ssq %>%
    select(amnesty, dishis2, pcthis2, contact, retecon, idea, edu2, income2,
           age, male, wteth) %>%
    filter(across(everything(), ~!is.na(.x))) %>%
    mutate(across(everything(), as.numeric)) %>%
    mutate(amnesty = as.factor(amnesty))
```

Estimate regression

```
# Make the learning task
amnestyTaskPaper <- makeClassifTask(data = select(X2, -wteth),
                                    target = "amnesty",
                                    weights = X2$wteth)

# Create a logistic regression learner
logRegLearner <- makeLearner("classif.logreg", predict.type = "prob")

# Train or fit the model on the whole dataset
amnestyModelPaper <- mlr::train(logRegLearner, amnestyTaskPaper)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
summary(amnestyModelPaper$learner.model)
```

```
##
## Call:
## stats::glm(formula = f, family = "binomial", data = getTaskData(.task,
##     .subset), weights = .weights, model = FALSE)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4768  -0.9430  -0.4715   0.9180   2.6897
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.816576   0.732098   2.481   0.0131 *
## dishis2     -0.549234   0.403517  -1.361   0.1735
## pcthis2     -2.359947   0.973049  -2.425   0.0153 *
## contact      0.083985   0.044441   1.890   0.0588 .
## retecon     -0.233547   0.158242  -1.476   0.1400
## idea        -0.707595   0.163012  -4.341 1.42e-05 ***
## edu2         0.290774   0.131775   2.207   0.0273 *
## income2     -0.128457   0.070815  -1.814   0.0697 .
## age          0.008602   0.006670   1.290   0.1972
## male        -0.345788   0.225834  -1.531   0.1257
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 533.15  on 391  degrees of freedom
## Residual deviance: 484.58  on 382  degrees of freedom
## AIC: 446.5
##
## Number of Fisher Scoring iterations: 4
```

Question 2

Create test and train

We scale our variables for use with the neural network and SVM (SVM does it automatically, but it's OK we do it manually). Results don't need to be scaled for the random forest, but it doesn't change the substantive findings.

TABLE 3

Ordered and Binary Logistic Regressions of Immigration Policy Preferences on Respondents' Latino Aversion and Selected Predictors Among Anglos, 2005–2006[a]

Predictor	Legal Immigration	Mexican Immigration	Amnesty
Latino aversion	−0.955**	−1.734**	−0.549
	(0.357)	(0.355)	(0.403)
Latino context	−1.671#	−1.332	−2.359*
	(0.890)	(0.855)	(0.973)
Minority contact	0.057	−0.021	0.084#
	(0.043)	(0.039)	(0.044)
Economic evaluation	−0.202	−0.319*	−0.233
	(0.147)	(0.141)	(0.158)
Political ideology	−0.535**	−0.547**	−0.707**
	(0.155)	(0.145)	(0.163)
Education	0.443**	0.249*	0.290*
	(0.127)	(0.117)	(0.131)
Income	−0.026	0.025	−0.128#
	(0.069)	(0.063)	(0.070)
Age	0.003	0.011#	0.008
	(0.006)	(0.006)	(0.006)
Male	0.176	0.019	−0.345
	(0.215)	(0.199)	(0.225)
Cutpoint 1	−1.701	−1.761	1.470
Cutpoint 2	0.819	0.682	
Chi-square	52.138	60.649	48.567
Cox & Snell R^2	0.140	0.140	0.116
N	347	402	392

[a]Numbers in cells are ordered (legal and Mexican immigration) and binary logistic (amnesty) regression coefficients, associated standard errors, and two-tailed probabilities.

#$p<0.10$; * = $p<0.05$; ** = $p<0.01$.

NOTE: Legal and Mexican immigration are derived from policy preferences for increased (3), current levels (2), or decreased immigration (1). Amnesty is derived from preferences for repeating the 1986 amnesty program, a good thing (1) or bad thing (0). List-wise deletion was used for analysis.

Figure 1: Regression Results from Paper

```
set.seed(420)
split <- createDataPartition(X2$amnesty, p = 0.8, list = F, times = 1)

train = X2[split,]
test = X2[-split,]

scaleTrain <- train %>% mutate(across(c(-wteth, -amnesty), ~as.numeric(scale(.x))))
scaleTest <- test %>% mutate(across(c(-wteth, -amnesty), ~as.numeric(scale(.x))))

# save for use in hyperparameter tuning
save(scaleTrain, file = here("hw2", "output", "scaleTrain"))
```

Compare the cross-validation performances of the 3 algorithms

```
# Make the training task
amnestyTaskTrain <- makeClassifTask(data = select(scaleTrain, -wteth),
                                    target = "amnesty")

# Create our learners
svm <- makeLearner("classif.svm", predict.type = "prob")
randforest <- makeLearner("classif.randomForest",
                          importance = T,
                          predict.type = "prob")
neuralnet <- makeLearner("classif.neuralnet", predict.type = "prob")
```

It should be noted these algorithms do not use survey weights in contrast to the logistic regression estimated in the paper. Moving forward, we will not be using the survey weights (even for the logistic regression).

Additionally the hyperparameter tuning through cross-validation occurs in a different R script because the process is quite lengthy, and it wastes times to have to do it every time just to generate a new report.

Looking at accuracy, the false positive rate, and the false negative rate:

- SVM and Random Forest have very similar accuracy ratings with Random Forest being a bit more accurate. However SVM is a bit more balanced in being able to tell apart positive and negative cases. Random Forest does not do a very good job at picking out negatives, but it does a comparatively good job at picking out positives.
- The neural network performs worse than both SVM and Random Forest but not by much. It appears as if once one starts adding more than 3 hidden layers, predictive accuracy decreases.
- Overall it's much harder for the models to pick out negative cases than it is for them to pick out positive cases.
- The split in the train set between positive and negative cases is 0.5509554, 0.4490446. According to the cross-validation, our models are performing about 8 - 11 percentage points better than a naive baseline which guesses 0 for every case.

```
load(here("hw2", "output", "tunedSvm"))
load(here("hw2", "output", "tunedRandForest"))
load(here("hw2", "output", "tunedNeuralNet"))
tunedSvm
```

```
## Tune result:
## Op. pars: kernel=polynomial; degree=1; cost=3.69; gamma=4.01
## acc.test.mean=0.6528226,fpr.test.mean=0.4900614,fnr.test.mean=0.2396790
```

```
tunedRandForest
```

```
## Tune result:
```

```
## Op. pars: ntree=200; mtry=6; nodesize=1; maxnodes=5
## acc.test.mean=0.6689516,fpr.test.mean=0.5836474,fnr.test.mean=0.1394017
```

```
tunedNeuralNet
```

```
## [[1]]
## Tune result:
## Op. pars: threshold=0.8; algorithm=rprop+; act.fct=logistic; hidden=2
## acc.test.mean=0.6348589,fpr.test.mean=0.4737202,fnr.test.mean=0.2761957
##
## [[2]]
## Tune result:
## Op. pars: threshold=0.8; algorithm=rprop+; act.fct=logistic; hidden=1,4
## acc.test.mean=0.6360282,fpr.test.mean=0.5465697,fnr.test.mean=0.2126477
##
## [[3]]
## Tune result:
## Op. pars: threshold=0.8; algorithm=rprop+; act.fct=logistic; hidden=2,4,4
## acc.test.mean=0.6397379,fpr.test.mean=0.5582722,fnr.test.mean=0.2058516
##
## [[4]]
## Tune result:
## Op. pars: threshold=0.8; algorithm=rprop+; act.fct=logistic; hidden=3,4,4,4
## acc.test.mean=0.6216734,fpr.test.mean=0.6071947,fnr.test.mean=0.1946713
##
## [[5]]
## Tune result:
## Op. pars: threshold=0.8; algorithm=rprop+; act.fct=logistic; hidden=4,4,3,4,4
## acc.test.mean=0.5922782,fpr.test.mean=0.7225935,fnr.test.mean=0.1414872
```

Question 3

Train our models

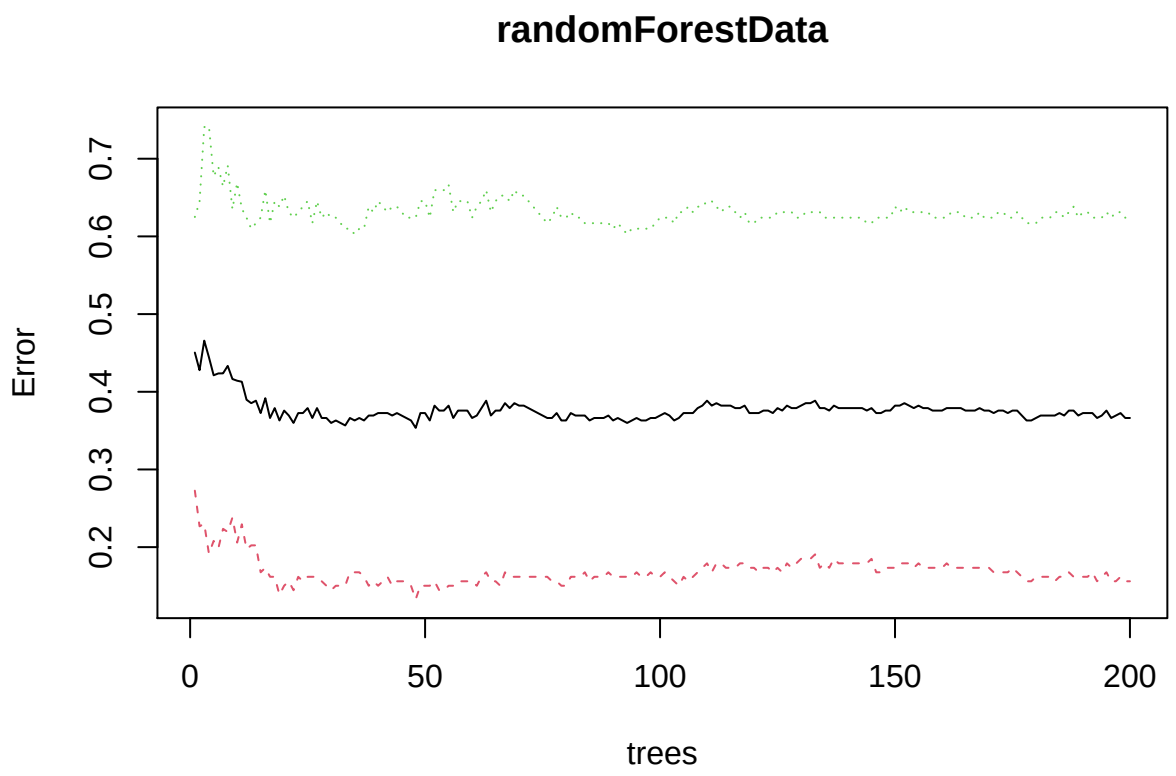
```
# train our Logistic regression
trainedLogit <- mlr::train(logRegLearner, amnestyTaskTrain)

# set hyperparameter for SVM
tunedSvmPars <- setHyperPars(svm, par.vals = tunedSvm$x)
trainedSvm <- mlr::train(tunedSvmPars, amnestyTaskTrain)

# set hyperparameters and train the random forest
tunedRandomForestPars <- setHyperPars(randforest, par.vals = tunedRandForest$x)
trainedRandomForest <- mlr::train(tunedRandomForestPars, amnestyTaskTrain)

# set hyperparameters and train neural net
accuracyLayer <- unlist(map(tunedNeuralNet, function(x) {x$y[["acc.test.mean"]]}))
maxAccLayer <- which(accuracyLayer == max(accuracyLayer))
tunedNeuralNetPars <- setHyperPars(neuralnet, par.vals = tunedNeuralNet[[maxAccLayer]]$x)
trainedNn <- mlr::train(tunedNeuralNetPars, amnestyTaskTrain)

# Check to see if our mean out-of-bag error converges for random forest
randomForestData <- trainedRandomForest$learner.model
plot(randomForestData)
```



Run Predictions

```
# Run predictions for logistic regression
predictLogit <- predict(trainedLogit,
                        newdata = select(scaleTest, -amnesty, -wteth))
predictLogit$data["truth"] <- scaleTest[["amnesty"]]

# Run predictions for Support Vector Machine
predictSvm <- predict(trainedSvm, newdata = select(scaleTest, -amnesty, -wteth))
predictSvm$data["truth"] <- scaleTest[["amnesty"]]

# Run predictions for random forest
predictRf <- predict(trainedRandomForest,
                     newdata = select(scaleTest, -amnesty, -wteth))
predictRf$data["truth"] <- scaleTest[["amnesty"]]

# Run predictions for neural net
predictNn <- predict(trainedNn, newdata = select(scaleTest, -amnesty, -wteth))
predictNn$data[["truth"]] <- scaleTest[["amnesty"]]
```

Plot ROC Curves and calculate Area under the ROC Curve (AUC)

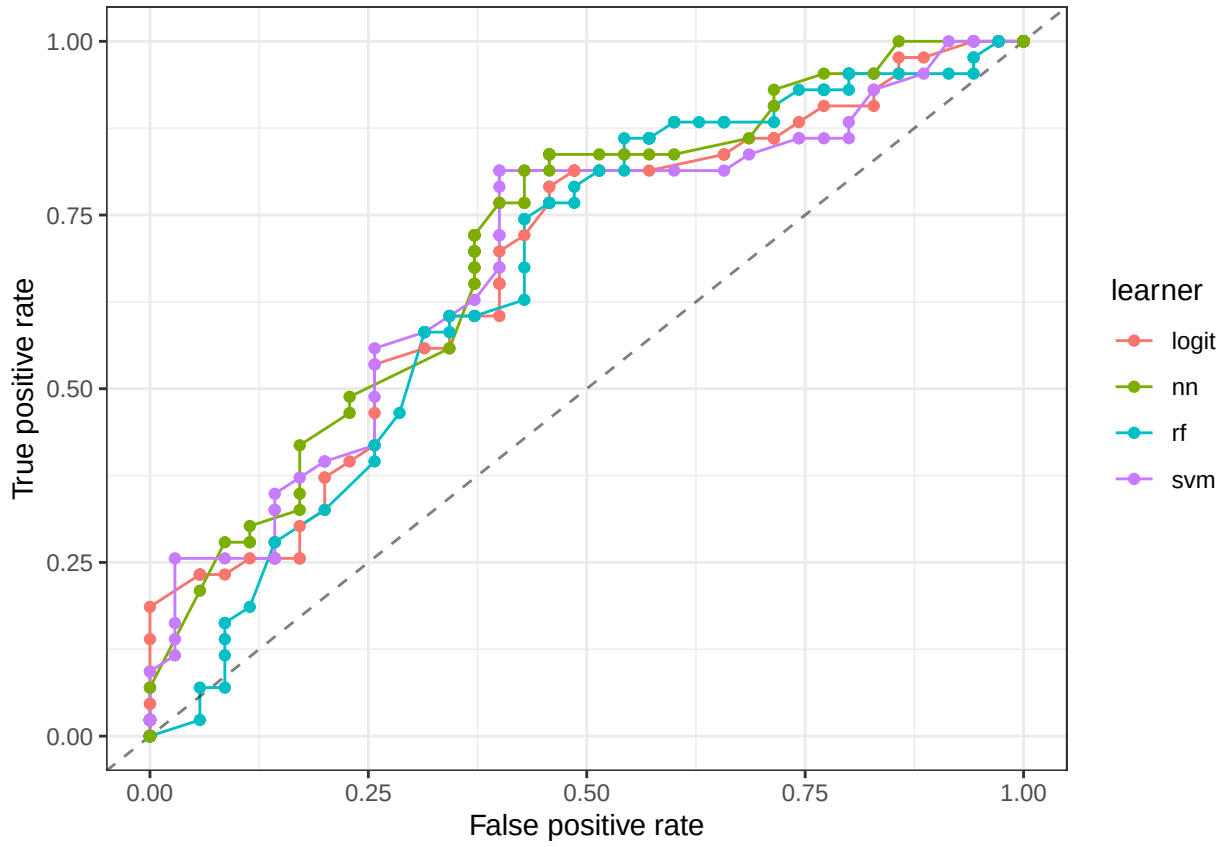
- The Support Vector Machine model performs 2nd best. It does have the highest maximum accuracy though.
- Logistic Regression has nearly identical performance to SVM but is still slightly a little worse.
- The neural network has the best AUC and has the 2nd best maximum accuracy.

- The random forest model performed the worst but not by a wide margin.
- Overall our cross-validation procedures seemed to capture the out-of-sample performance well. Our models performed 8 - 10 percentage points better than the naive baseline. However the models were all very similar in terms of performance and none really outpeformed the relatively simple logistic regression by a large margin.

```
predictions <- list(logit = predictLogit, svm = predictSvm, rf = predictRf,
                    nn = predictNn)

# Generate dataframe of measures
performance <- generateThreshVsPerfData(predictions,
                                        measures = list(fpr, tpr, acc))

# Plot ROC Curves
plotROCCurves(performance) + theme_bw() + geom_point()
```




```
# Calculate AUC for each model
auc <- map(predictions, mlr::performance, measures = mlr::auc)
auc
```

```
## $logit
##       auc
## 0.6797342
##
## $svm
##       auc
## 0.6857143
```

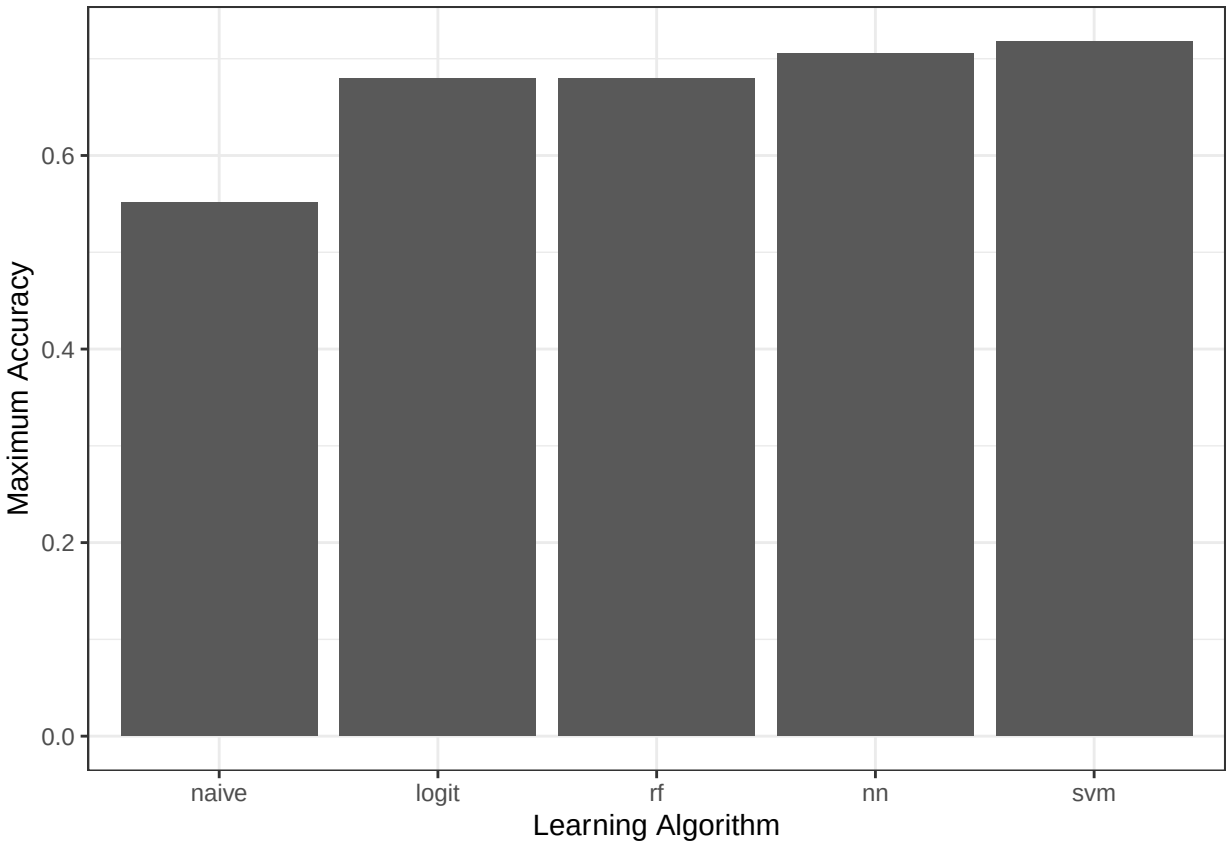
```
##
## $rf
##      auc
## 0.661794
##
## $nn
##       auc
## 0.7069767
```

```
aucDf <-
    auc %>%
    as_tibble() %>%
    pivot_longer(everything(), names_to = "learner", values_to = "auc")

# Calculate naive accuracy
naiveAcc <-  prop.table(table(scaleTest$amnesty))[["0"]]

# Find maximum accuracy
maxAcc <-
    performance$data %>%
    group_by(learner) %>%
    summarise(acc = max(acc)) %>%
    add_row(learner = "naive", acc = naiveAcc) %>%
    full_join(aucDf, by = "learner")

ggplot(maxAcc, aes(x = reorder(learner, acc), y = acc)) +
    geom_bar(stat = "identity") +
    theme_bw() +
    labs(y = "Maximum Accuracy", x = "Learning Algorithm")
```



Question 4

Train our models selectively holding out one of the features

```
# combine train and test
scaleFull <- bind_rows(scaleTrain, scaleTest) %>% select(-wteth)

# Create our list of features to hold out one-by-one (as well as full features)
features <- as.list(c(colnames(select(scaleFull, -amnesty)), "total"))
names(features) <- features

# Create new task for each new feature set
featureTasks <- map(features, function(missingFeat) {
    if(missingFeat != "total")
        makeClassifTask(data = select(scaleFull, -missingFeat), target = "amnesty")
    else
        makeClassifTask(data = scaleFull, target = "amnesty")
})
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(missingFeat)` instead of `missingFeat` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
# For each learning algorithm, for each task, train a model (4 * 10) = 40 models
trainedLogits <- map(featureTasks,
                     function(task, model) {
```

```
                        mlr::train(model, task)},
                    model = logRegLearner)

trainedSvms <- map(featureTasks,
                   function(task, model) {
                       mlr::train(model, task)},
                   model = tunedSvmPars)

trainedRfs <- map(featureTasks,
                  function(task, model) {
                      mlr::train(model, task)},
                  model = tunedRandomForestPars)

trainedNns <- map(featureTasks,
                  function(task, model) {
                      mlr::train(model, task)},
                  model = tunedNeuralNetPars)
```

Run predictions on our newly trained models

```
# Run predictions for each of the 36 models
PredictModify <- function(model, missingFeat) {
    if(missingFeat != "total")
        p <- predict(model, newdata = select(scaleFull, -amnesty, -missingFeat))
    else
        p <- predict(model, newdata = select(scaleFull, -amnesty))

    p$data["truth"] <- scaleFull[["amnesty"]]
    p
}

predictLogits <- pmap(list(trainedLogits, names(trainedLogits)), PredictModify)
predictSvms <- pmap(list(trainedSvms, names(trainedSvms)), PredictModify)
predictRfs <- pmap(list(trainedRfs, names(trainedRfs)), PredictModify)
predictNns <- pmap(list(trainedNns, names(trainedNns)), PredictModify)
```

Analyse results to get a sense of how holding out each variables worsens (or improves) predictive peformance

```
# Calculates AUC for set of models with the missing feature
BuildPredictionDf <- function(logit, svm, rf, nn, missingFeat) {
    predictList <- list(logit = logit, svm = svm, rf = rf, nn = nn)
    performanceDf <- generateThreshVsPerfData(predictList,
                                              measures = list(fpr, tpr, acc))

    # Grab AUC
    auc <-
        map(predictList, mlr::performance, measures = mlr::auc) %>%
        as_tibble() %>%
        pivot_longer(everything(), names_to = "learner", values_to = "auc") %>%
        mutate(run = missingFeat)
}
```

```r
# Create dataframe of each model's performance when missing a feature
featureImportance <-
    pmap(list(predictLogits, predictSvms, predictRfs, predictNns, features),
         BuildPredictionDf) %>%
    bind_rows()

# Extract model trained on all features
total <-
    featureImportance %>%
    filter(run == "total") %>%
    rename(total_auc = auc) %>%
    select(-run)

# Find improvement or decline in performance when excluding each feature
diff <-
    featureImportance %>%
    filter(run != "total") %>%
    full_join(total, by = "learner") %>%
    mutate(aucDiff = auc - total_auc)

ggplot(diff, aes(x = rep(1, nrow(diff)), y = aucDiff)) +
    geom_bar(stat = "identity", position = "dodge", aes(fill = learner)) +
    theme_bw() +
    theme(axis.title.x = element_blank(), axis.text.x = element_blank()) +
    labs(y = "Change in AUC") +
    facet_wrap(~run)
```

Taking each of the variables in turn:

- Holding out latino aversion (dishis2) caused model performance to decline moderately with the biggest declines happening for the random forest and the neural network.
- Holding out latino context (pcthis2) caused model performance to decline pretty uniformly across models.
- Holding out latino contact (contact) caused model performance to slightly decline. The effect is slightly pronounced for the random forest.
- Holding out personal financial situation (retecon) caused very small declines in model performance. Slightly pronounced for the random forest and neural network.
- Holding out political ideology (idea) degrades model performance the most for nearly all models. The one exception being random forest which saw very little decline.
- Holding out education (edu2) caused slight declines in model performance.
- Holding out famliy income (income) did very little to change the predictive accuracy of the models and in some cases improved it (the neural network).
- Holding out age (age) caused no change for the logistic regression or the SVM. It decreased the accuracy of the random forest but increased the performance of the neural network.
- Holding out sex (male) caused slight decreases in model performance with slightly pronounced effects for the random forest and the neural network.

The ranking of variable importance in each model differs somewhat but the amount of change they produced in predictive accuracy is largely the same across models (with the exception of age and income). The random forest and neural network seemed most sensitive to variables being removed. It's possible they are the most sensitive to the hyperparameter tuning process.

The findings from the paper seem to largely hold up when it comes to the interpretation of which variables and associated coefficients were most important. Both ideology (idea) and latino context (pcthis2) produced the largest average decreases in accuracy when held out. These variables were identified as statistically

significant in the paper. However education (edu2) did not really produce substantive changes in predictive accuracy even as it achieved statistical significance in the paper. Sex (sex) and latino aversion (dishis2) produced had some pronounced negative effects on predictive power, but the drop in performance was not uniform across models.

Contribution to predictive power isn't the same statistical significance, of course. Nevertheless it's interesting to compare the substantive findings from this predictive exercise to the largely explanatory exercise in the paper.