Homwork 5

```
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 0.2.0 --

## v broom        0.8.0     v recipes      0.2.0
## v dials        0.1.1     v rsample      0.1.1
## v dplyr        1.0.9     v tibble       3.1.7
## v ggplot2      3.3.6     v tidyr        1.2.0
## v infer        1.0.0     v tune         0.2.0
## v modeldata    0.1.1     v workflows    0.2.6
## v parsnip      0.2.1     v workflowsets 0.2.1
## v purrr        0.3.4     v yardstick    0.0.9

## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/
```

```
library(ISLR)
library(ISLR2)
```

```
##
## Attaching package: 'ISLR2'

## The following objects are masked from 'package:ISLR':
##
##     Auto, Credit
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v readr   2.1.2     v forcats 0.5.1
## v stringr 1.4.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## x dplyr::filter()     masks stats::filter()
## x stringr::fixed()    masks recipes::fixed()
## x dplyr::lag()        masks stats::lag()
## x readr::spec()       masks yardstick::spec()
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-4
```

```
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
tidymodels_prefer()
```

```
poke <- read.csv('Pokemon.csv')
```

```
pokemon <- clean_names(poke)
```

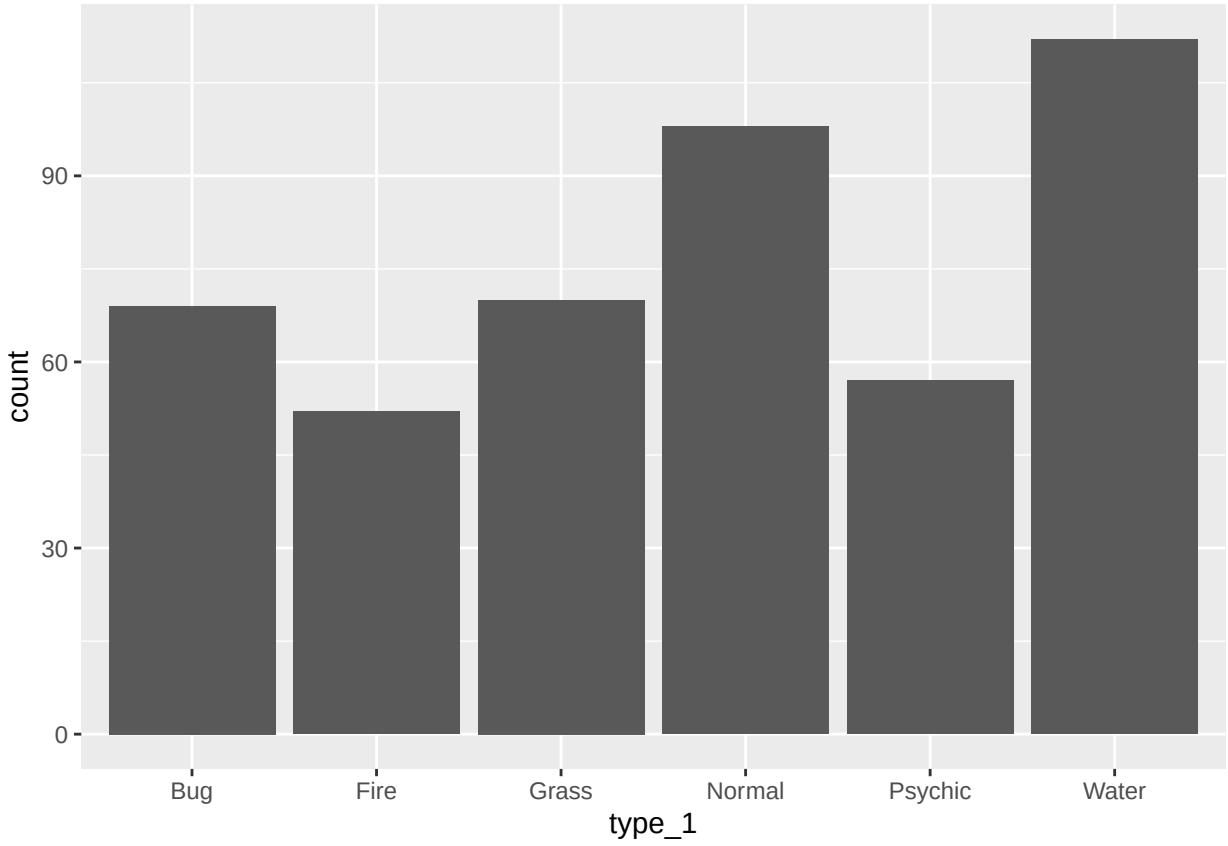
Clean names made all the predictor names lowercase and replaced periods with underscores. This is useful because it allopws us to write code to access thes predictors, such as pokemon$type_1 without the periods and spaces condusing R.

```
pokemon <- pokemon%>%filter(type_1 == 'Bug' | type_1== 'Fire' | type_1== 'Grass' | type_1== 'Normal' | t

pokemon$legendary <- as.factor(pokemon$legendary)

pokemon$type_1 <- as.factor(pokemon$type_1)
pokemon$generation <- as.factor(pokemon$generation)

ggplot(pokemon, aes(x=type_1)) + geom_bar()
```




```
set.seed(608)
pokeSplit <- initial_split(pokemon, prop = 0.80,
                                strata = type_1)
poke_train <- training(pokeSplit)
poke_test <- testing(pokeSplit)
```

Training set has 364,Test had 94.

```
poke_fold <- vfold_cv(poke_train, v = 5, strata = type_1)
```

We want to stratify the folds so that we don't end up with folds consisting mostly of one type or having very little of one type.

```
pokeRecipe <- recipe(type_1 ~ legendary + generation + sp_atk + attack + speed  + defense + hp + sp_def
summary(pokeRecipe)
```

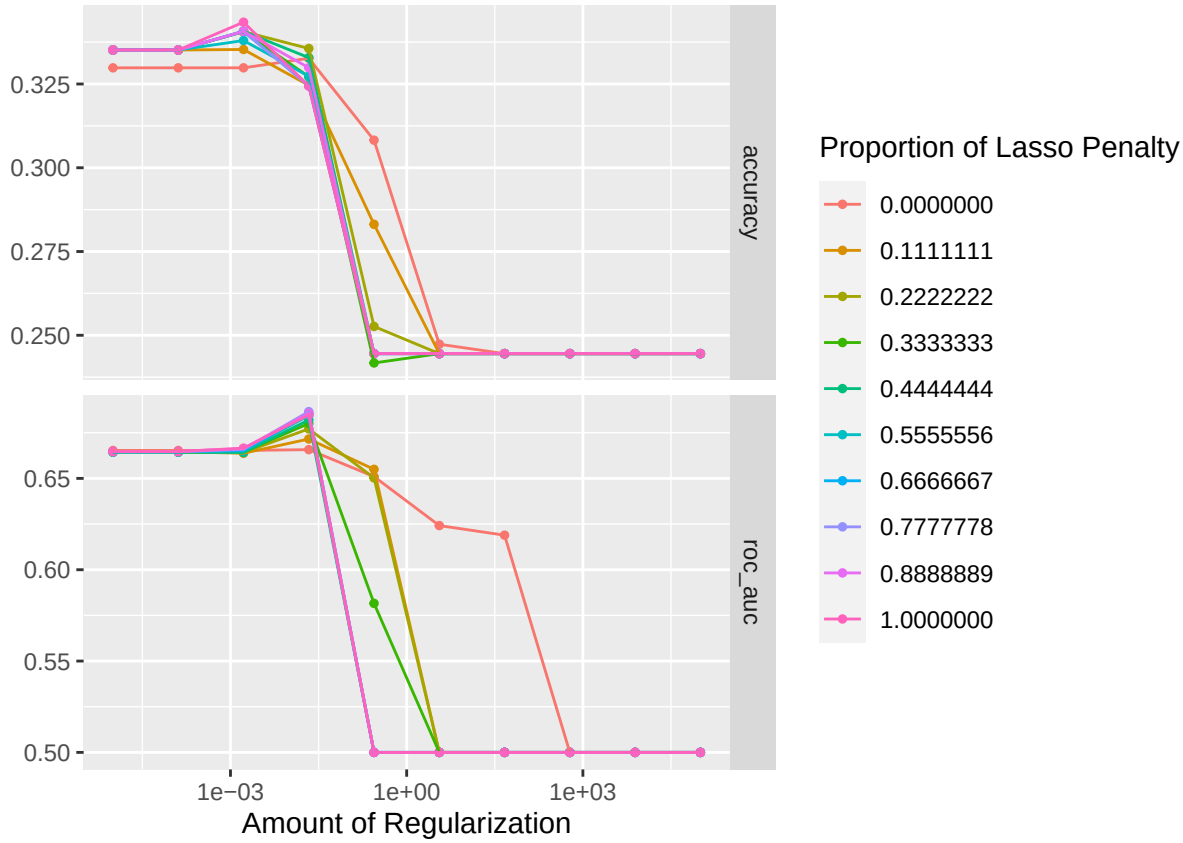
```
## # A tibble: 9 x 4
##   variable   type    role      source
##   <chr>      <chr>   <chr>     <chr>
## 1 legendary  nominal predictor original
## 2 generation nominal predictor original
## 3 sp_atk     numeric predictor original
## 4 attack     numeric predictor original
## 5 speed      numeric predictor original
```

```
## 6 defense    numeric predictor original
## 7 hp         numeric predictor original
## 8 sp_def     numeric predictor original
## 9 type_1     nominal outcome   original
```

```
penMix <- grid_regular(penalty(range=c(-5,5)), mixture(range= c(0,1)), levels = 10)
mReg <- multinom_reg(penalty = tune(), mixture = tune()) %>% set_engine('glmnet') %>% set_mode('classif
mWkflow <- workflow() %>% add_model(mReg) %>% add_recipe(pokeRecipe)
```

I will be fitting 50 models total

```
tuneFit <- tune_grid(mWkflow, resamples = poke_fold, grid = penMix)
autoplot(tuneFit)
```



smaller values of mixture and penalty produce better AUC and accuracy.

```
selected <- select_best(tuneFit, metric = 'roc_auc')
finalwkflw <- finalize_workflow(mWkflow, selected)

final_fit <- fit(finalwkflw, data = poke_train)

augment(final_fit, new_data = poke_test) %>% accuracy(truth = type_1, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.404
```

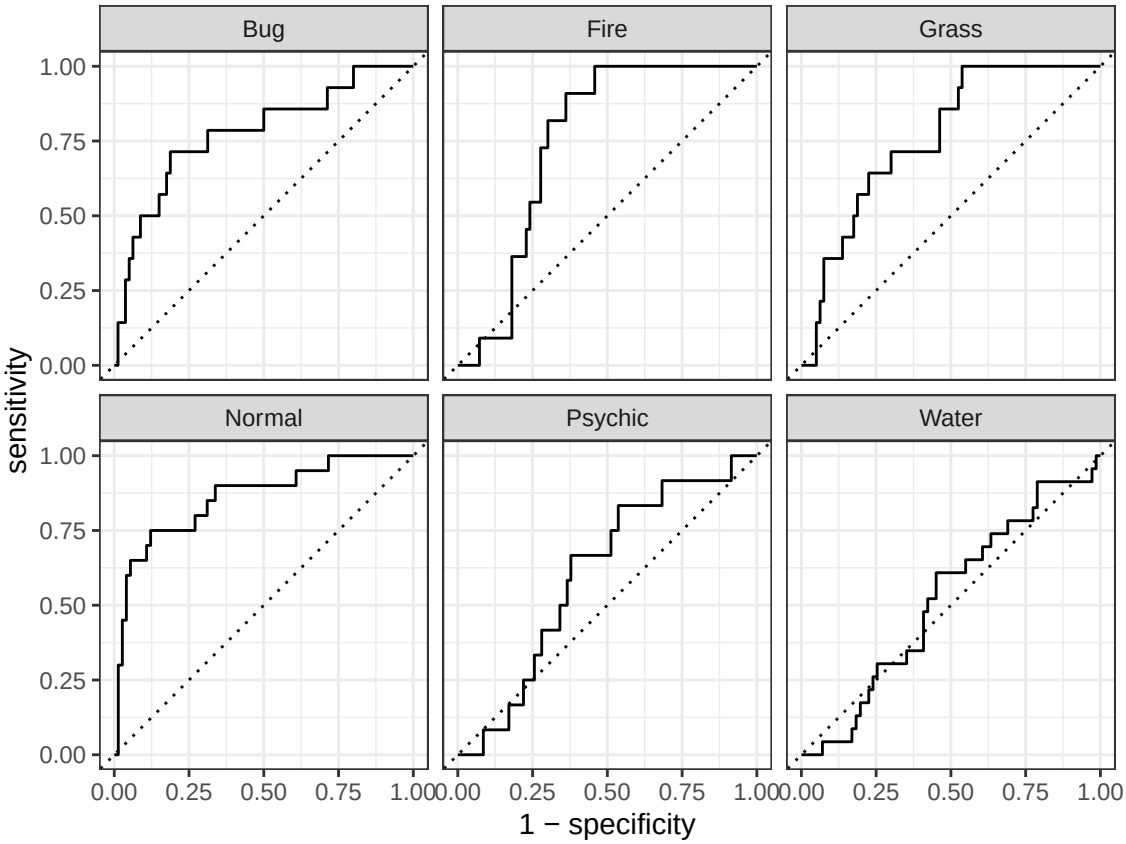
low accuracy: 0.4042553

```
augment(final_fit, new_data = poke_test) %>% roc_auc(type_1, estimate = c(.pred_Bug, .pred_Fire, .pred_
```

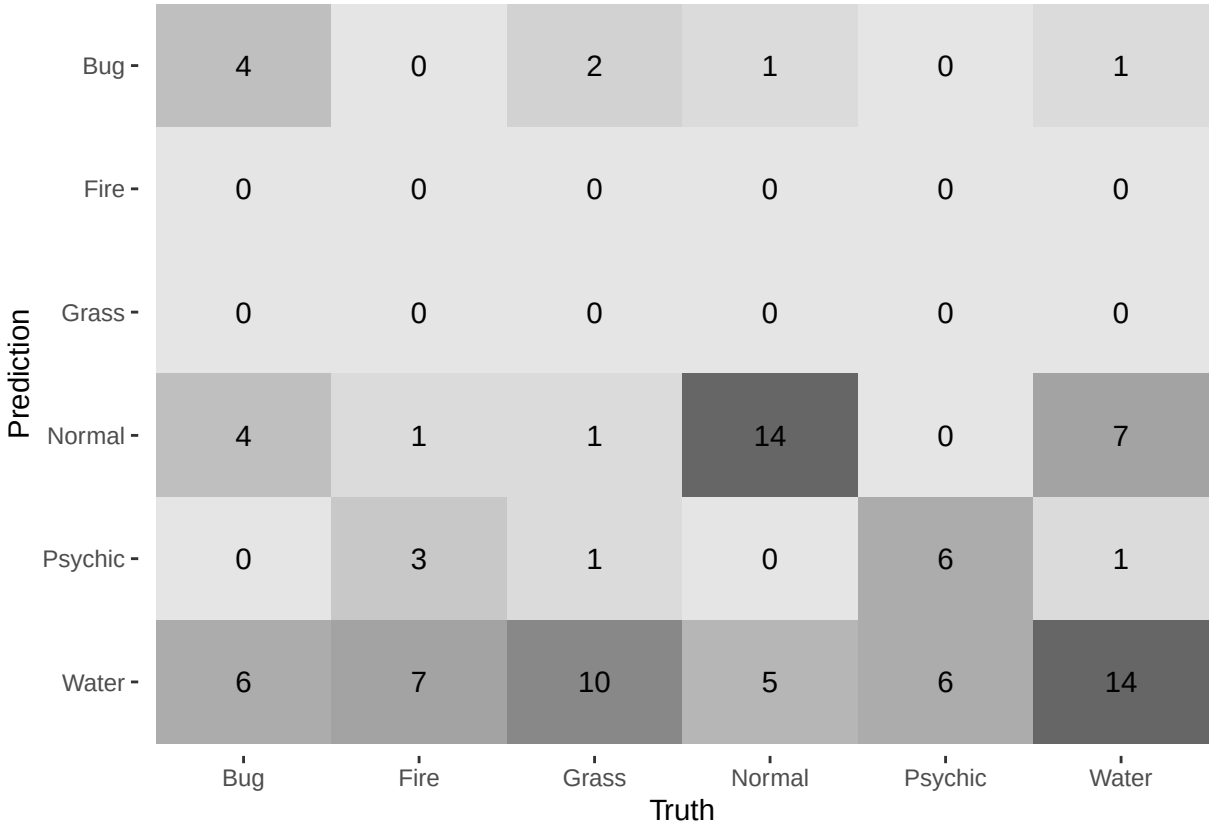
```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc hand_till      0.705
```

AUC = 0.7050415

```
augment(final_fit, new_data = poke_test) %>% roc_curve(type_1, estimate = c(.pred_Bug, .pred_Fire, .pred
```




```
augment(final_fit, new_data = poke_test) %>% conf_mat(truth = type_1, estimate = .pred_class) %>% autopl
```


Prediction
Bug
Fire
Grass
Normal
Psychic
Water
4 0 2 1 0 1
0 0 0 0 0 0
0 0 0 0 0 0
4 1 1 14 0 7
0 3 1 0 6 1
6 7 10 5 6 14
Bug
Fire
Grass
Normal
Psychic
Water
Truth