PSTAT 131 HW 5

Luke Todd

5/15/2022

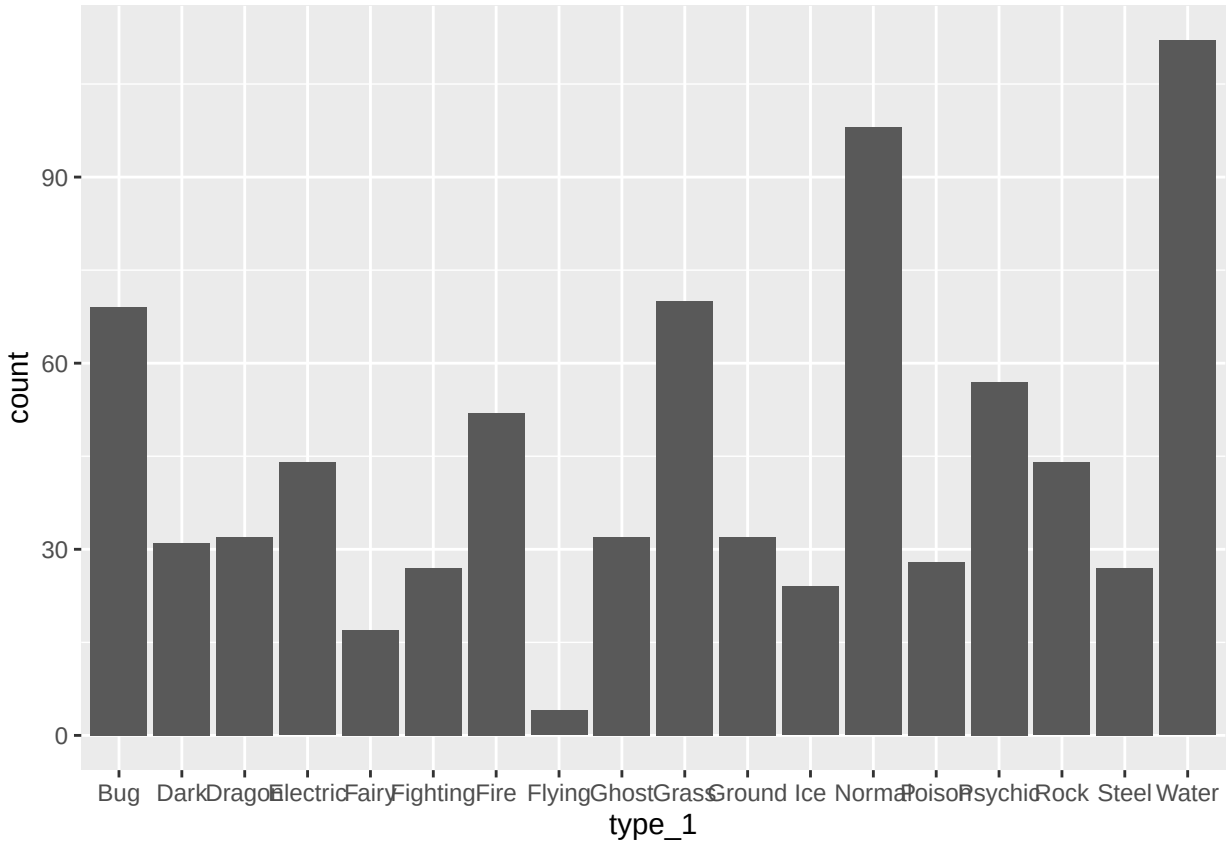
Exercise 1

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

clean_names() is a useful function because it provides a standard naming system for the whole dataset. This way, the data is more predictable than it would be packaged raw.

Exercise 2

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



Based on the plot, there are 18 different types. Flying and fairy have a small amount of pokemon when compared to other types.

```
pokemon <- sqldf('SELECT * FROM pokemon WHERE type_1 IN ("Normal", "Bug", "Grass", "Fire", "Water", "Psy

pokemon <- pokemon %>%
  mutate(type_1 = factor(type_1),
         legendary = factor(legendary),
         generation = factor(generation))
```

Exercise 3

```
pokemon_split <- initial_split(pokemon, prop = 0.8,
                               strata = type_1)

pokemon_train <- training(pokemon_split)
pokemon_test <- testing(pokemon_split)

pokemon_fold <- vfold_cv(pokemon_train, v = 5,
                         strata = type_1)
```

Exercise 4

```
pokemon_recipe <- recipe(type_1 ~ legendary + generation + sp_atk +
                           attack + speed + defense + hp + sp_def, data = pokemon) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_normalize(all_predictors())
```

Exercise 5

```
multinom_model <- multinom_reg(mixture = tune(), penalty = tune()) %>%
  set_mode("classification") %>%
  set_engine("glmnet")

multinom_workflow <- workflow() %>%
  add_recipe(pokemon_recipe) %>%
  add_model(multinom_model)

pokemon_grid <- grid_regular(penalty(range = c(-5, 5)), mixture(range = c(0,1)), levels = 10)
pokemon_grid
```

```
## # A tibble: 100 x 2
##          penalty mixture
##            <dbl>   <dbl>
##  1      0.00001         0
##  2      0.000129        0
##  3      0.00167         0
##  4      0.0215          0
##  5      0.278           0
##  6      3.59            0
```

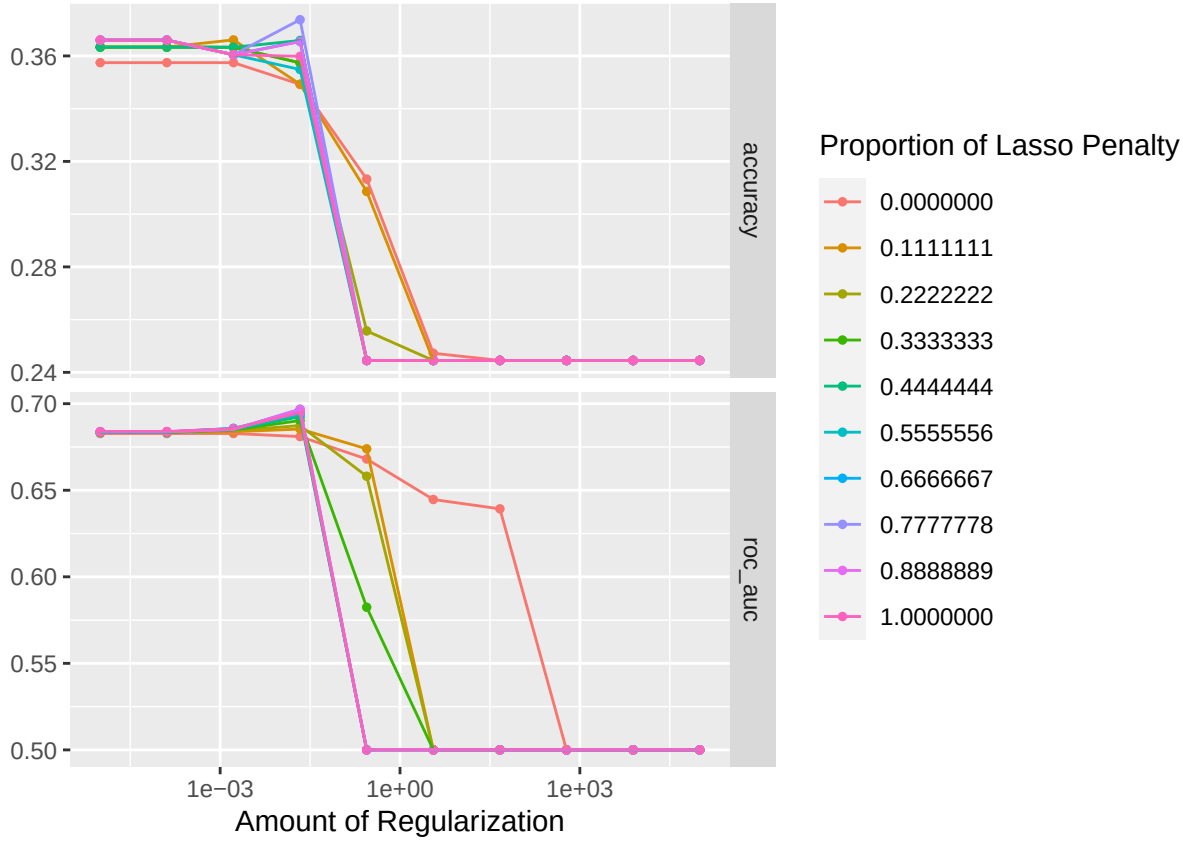
```
##  7     46.4            0
##  8    599.             0
##  9   7743.             0
## 10 100000              0
## # ... with 90 more rows
```

When we fit these models to the folded data, we will be fitting a total of 500 models.

Exercise 6

```
tune_res <- tune_grid(multinom_workflow,
  resamples = pokemon_fold,
  grid = pokemon_grid)

autoplot(tune_res)
```



Smaller values of penalty and mixture produce better accuracy and ROC AUC.

Exercise 7

```
best <- select_best(tune_res)
```

```
## Warning: No value of ‘metric‘ was given; metric ’roc_auc’ will be used.
```

```
multinom_final <- finalize_workflow(multinom_workflow, best)

multinom_final_fit <-  fit(multinom_final, data = pokemon_train)
augment(multinom_final_fit, new_data = pokemon_test) %>%
  accuracy(truth = type_1, estimate = .pred_class)
```

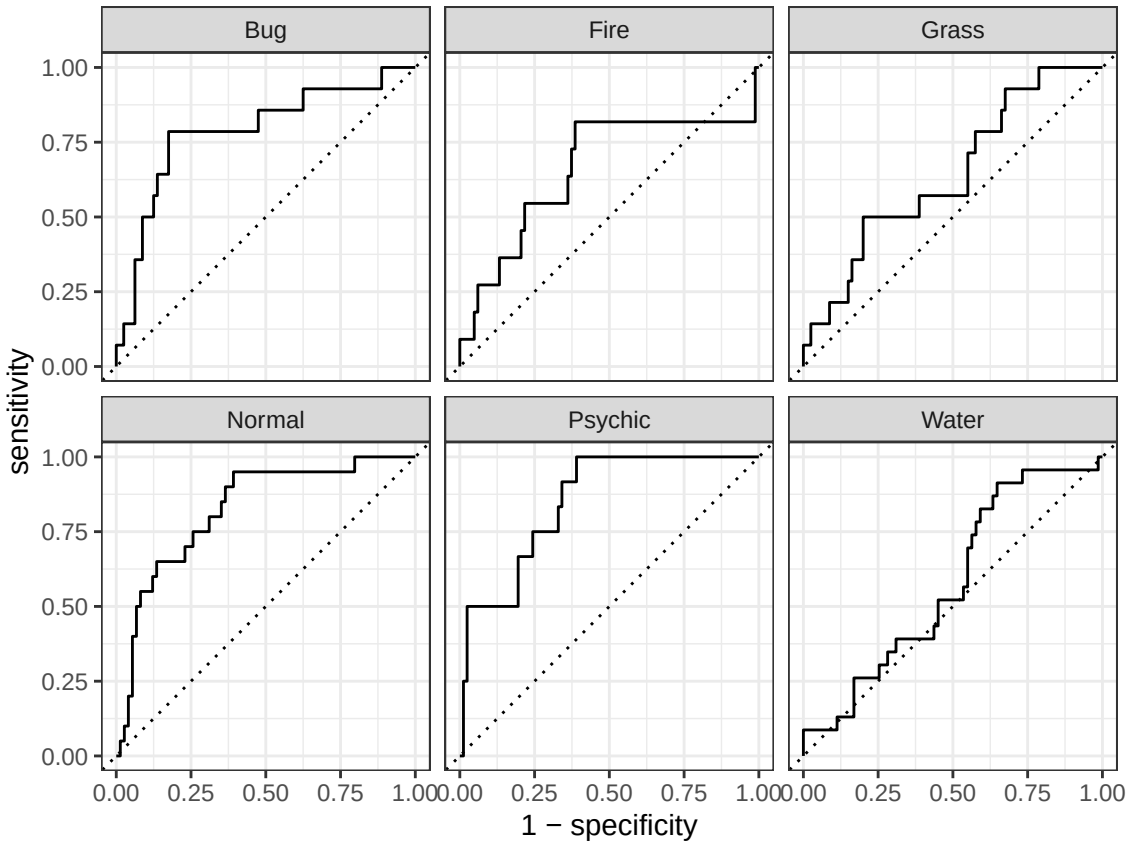
```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy multiclass     0.362
```

Exercise 8

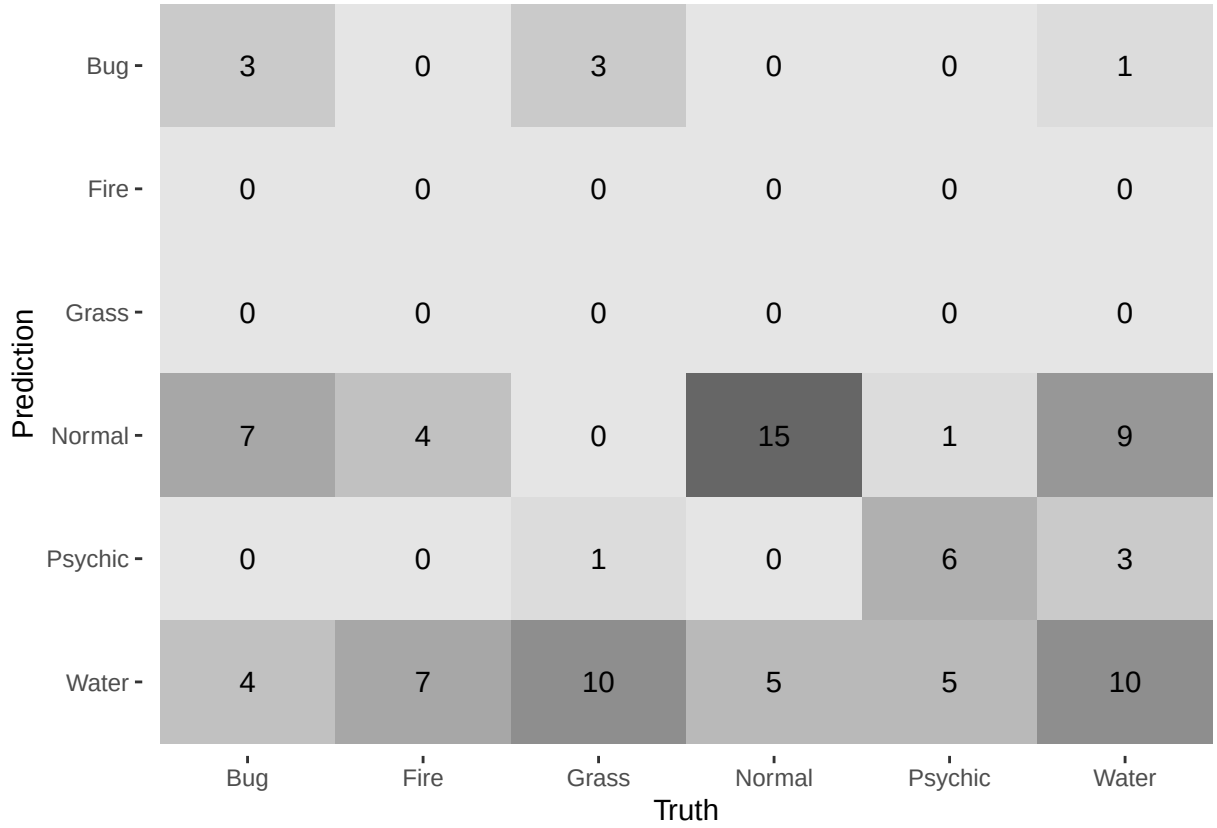
```
# table
augment(multinom_final_fit, new_data = pokemon_test, type = 'prob')
```

```
## # A tibble: 94 x 20
##    number name      type_1 type_2 total    hp attack defense sp_atk sp_def speed
##     <dbl> <chr>     <fct>  <chr>  <dbl> <dbl>  <dbl>   <dbl>  <dbl>  <dbl> <dbl>
##  1     10 Caterpie  Bug    <NA>     195    45     30      35     20     20    45
##  2     11 Metapod   Bug    <NA>     205    50     20      55     25     25    30
##  3     13 Weedle    Bug    Poison   195    40     35      30     20     20    50
##  4     20 Raticate  Normal <NA>     413    55     81      60     50     70    97
##  5     21 Spearow   Normal Flying   262    40     60      30     31     31    70
##  6     43 Oddish    Grass  Poison   320    45     50      55     75     65    30
##  7     48 Venonat   Bug    Poison   305    60     55      50     40     55    45
##  8     59 Arcanine  Fire   <NA>     555    90    110      80    100     80    95
##  9     61 Poliwhirl Water  <NA>     385    65     65      65     50     50    90
## 10     62 Poliwrath Water  Fight~   510    90     95      95     70     90    70
## # ... with 84 more rows, and 9 more variables: generation <fct>,
## #   legendary <fct>, .pred_class <fct>, .pred_Bug <dbl>, .pred_Fire <dbl>,
## #   .pred_Grass <dbl>, .pred_Normal <dbl>, .pred_Psychic <dbl>,
## #   .pred_Water <dbl>
```

```
# plots
augment(multinom_final_fit, new_data = pokemon_test, type = 'prob') %>%
  roc_curve(type_1, estimate = c(.pred_Bug,
                                 .pred_Fire,
                                 .pred_Grass,
                                 .pred_Normal,
                                 .pred_Psychic,
                                 .pred_Water)) %>%
  autoplot()
```




```
# heatmap of confusion matrix
augment(multinom_final_fit, new_data = pokemon_test) %>%
  conf_mat(truth = type_1, estimate = .pred_class) %>%
  autoplot(type = 'heatmap')
```



The models did relatively well. Based on these plots, we can see that our models did best at predicting normal, water, and psychic. For the rest of the types, the models were insufficient. My best guess for the models not fitting too well is just based on the fact that many of these pokemon have more than one type. We may be trying to guess the type_1 of the pokemon based on all of our observations, but in reality, these may be a result of the second type of the pokemon.