Classification of High and Low Income based on Environmental Climate Factors

Keenan Smith

2022-04-26

Introduction

When beginning this project, I knew I wanted to start with global data from the World Bank. I was unsure exactly where to begin, but I knew with the wealth of information within their archives, I would be able to find something to model. In the end, I settled on examining the indicators deemed important for climate change (World Bank Group, 2022) and examing whether they are useful indicators for determining whether a country is "High Income" or "Low-Middle" income.

Methodology & Process

"Built with R Version 4.2.0"

The first step in this project is to gather the data. This was done using the "WDI" package/API within R. Several steps were required to pull all 88 variables into R and make them useful. The first step was to write an R script to pull the data in and output it into a useful CSV to pull back into an R-Markdown document. Then the data were examined and determined which indicators were simply lacking in terms of usable data (lots of NA values or indicators like others.) Initially, two data sets were created since the data comes in terms of countries and years: One for the year 2010 and one for the year 2018. Unfortunately, 2018 had NA values in every single country so it could not be used for classification within the construct used to classify the model. The only full set of data are 2010 for 58 countries of 45 predictors. This original data set has n greater than p but not by much. Once this dataset was selected, I closely followed the methods used and recommended in Kirenz's" Classification with Tidymodels, Workflows and Recipes" which follows the Tidymodels framework in which I prefer to do my R coding. This process begins with exploratory data analysis and the printing of several histograms and boxplots to examine the data and examine how they react to the different income levels. Thankfully, the Tidymodels framework allows for preprocessing the data before they are modeled and analyzed. The steps chosen here were to transform the right skewed data, then normalize the data, then remove any variables and lastly, remove highly correlated values using the Pearson method. The model is then fit and validated using 10-fold cross validation with the following: Sensitivity, Precision, F Measure, Accuracy, Kappa, Roc Auc, and Specificity.

Limitations

- There are only a handful of countries with full datasets for the variables in question
- The data originally contains High, Upper Middle, Lower Middle and Low Income countries, due the constraints of what was learned in ISE537, multinomial classification was not used in this project
- This is a class project for ISE537. There is more that could be done here to continue to look into this data and determine ways to classify nations based on World Bank indicators

```
vis_dat(env_wdi_2010)
```

```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## Please use `gather()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```


```
# vis_miss(env_wdi_2010)

#is.na(env_wdi_2010) |>
#  colSums()
```

```
env_wdi_2010 %>%
  count(income_status, # count observations
        name ="total") %>%  # name the new variable
  mutate(percent = total/sum(total))
```

income_status	total	percent
High	26	0.3823529
Middle_Low	42	0.6176471

Data Budget Spending

```
set.seed(100)

split_2010 <- initial_split(env_wdi_2010, prop = .8, strata = income_status)

# 2010 Data Split
train_2010 <- training(split_2010)
test_2010 <- testing(split_2010)

# Exploration Datasets to isolate Dataframes from Tampering
explore_2010 <- train_2010

# k-fold Validation Set Creation
cv_folds_2010 <-vfold_cv(train_2010, v = 10, strata = income_status)
```

Data Pre-processing

```
env_2010_rec <-
  recipe(income_status ~ ., data = train_2010) |>
  step_log(agrid_land_km, arable_per, forest_per, pop_tot, mort_rate, coal_power_per, hydro_power_per, gas_power
           nuke_power_per, oil_power_per, renew_power_per, renew_non_hydro_per, renew_consum_per, elec_consum_kw
           co2_gaseous_kt, co2_liquid_kt, co2_emmision_mt, co2_solid_kt, total_greenhouse_mt, hfc_emissions_mt,
           methane_mt, no_emission_mt, agri_forest_fish_gdp_per) |>
  step_naomit(all_predictors(), skip = TRUE) |>
  step_normalize(all_numeric_predictors()) |>
  step_dummy(all_nominal_predictors()) |>
  step_zv(all_numeric_predictors()) |>
  step_corr(all_predictors(), threshold = .7, method = "pearson")

prepped_2010 <-
  env_2010_rec |>
  prep() |>
  juice()
```

Data Prep

```
glimpse(prepped_2010)
```

```
## Rows: 53
## Columns: 15
## $ agrid_land_per           <dbl> -0.14622778, -1.42773615, 0.62826882, 0.97109~
## $ arable_per               <dbl> -0.19707839, -0.33227438, 1.18896620, 1.51747~
## $ forest_per               <dbl> 0.22633598, -0.16142791, 0.29910529, -0.39006~
## $ cereal_yield             <dbl> 1.4177063, -0.8457892, 0.6450062, 1.2972223, ~
## $ pop_growth               <dbl> -0.061975424, 0.147974197, -0.639350770, -0.5~
## $ urban_pop_per            <dbl> 0.61613802, 0.29193396, 0.59729393, 1.3221491~
## $ for_invest_per           <dbl> -0.125516065, 6.915160089, -0.009733056, -0.5~
## $ school_enroll            <dbl> -0.035623892, 0.375459266, 0.311727061, 0.368~
## $ school_complete          <dbl> 0.23428105, 0.44958066, 0.94217018, 0.4360202~
## $ co2_emission_2017        <dbl> -0.89670589, 0.15204475, 0.67288548, -0.34656~
## $ other_greenhouse_mt      <dbl> 0.01598315, 0.05469239, -0.38079731, 0.135141~
## $ total_greenhouse_mt      <dbl> -0.263320290, -1.760964925, 0.502632696, -0.1~
## $ no_emission_per          <dbl> -0.2771243, 0.7814981, -0.5505021, -0.7235977~
## $ agri_forest_fish_gdp_per <dbl> -2.11830245, -1.00723560, -1.29113309, -1.532~
## $ income_status            <fct> High, High, High, High, High, High, High, Hig~
```

After the data is preprocessed and correlation is checked and removed from the system, the model is left with 14 indicators to classify income status. This is drastically reduced from the API pull from the World Bank data logs which had 88 variables and then that was reduced down to 37 based on data availability. The indicators left have a good variety to them in selecting a wealth of information pertaining to climate change. It takes into account the use of the land, the production of the land, the population, investment from foreign governments, indicators based on schooling and then the larger talking points of climate change which are based on greenhouse gases and other emissions. It is also left with the value added to the GDP due to agriculture, forestry, and fishing.

```
# Logistic Regression Model Spec
log_spec <-
  logistic_reg() |>
  set_engine("glm") |>
  set_mode("classification")

# Lasso Logistic Regression Using Glmnet
lasso_spec <-
```

```
  logistic_reg(penalty = tune(), mixture = 1) |>
  set_mode("classification") |>
  set_engine("glmnet")

# Elastic Net Logistic Regression Using Glmnet
elastic_spec <-
  logistic_reg(penalty = tune(), mixture = tune()) |>
  set_mode("classification") |>
  set_engine("glmnet")
```

GLM Fit & Visuals

This dataset presented a lot of challenges within the analysis. First, the data is mainly incomplete for many countries cataloged by the World Bank. This led to a very limited dataset when analyzing the classification fit. The first method used was a Tidymodels set up using the base R "glm" engine. An issue occurred within the glm.fit function within R as a result of a complete separation within the dataset resulting in a fit that was ok, but not desired. The conclusion at this point was that the indicators in question did not adequately provide a good classification for whether a nation was "High Income" or "Low Middle".

```
log_res |>
  extract_fit_engine() |>
  summary()
```

```
##
## Call:
## stats::glm(formula = ..y ~ ., family = stats::binomial, data = data)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -2.118e-05  -2.110e-08   2.110e-08   2.110e-08   2.210e-05
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               3.863e+01  6.050e+04   0.001    0.999
## agrid_land_per           -1.546e+01  7.434e+04   0.000    1.000
## arable_per                1.855e+01  1.135e+05   0.000    1.000
## forest_per                4.538e+00  1.784e+05   0.000    1.000
## cereal_yield             -1.382e+01  8.883e+04   0.000    1.000
## pop_growth                4.318e+01  1.902e+05   0.000    1.000
## urban_pop_per            -1.926e+01  8.155e+04   0.000    1.000
## for_invest_per           -1.156e+00  4.852e+04   0.000    1.000
## school_enroll            -4.932e-01  1.279e+05   0.000    1.000
## school_complete           3.775e+01  1.325e+05   0.000    1.000
## co2_emission_2017        -1.155e+01  7.732e+04   0.000    1.000
## other_greenhouse_mt       1.072e+00  2.895e+04   0.000    1.000
## total_greenhouse_mt       9.464e+00  3.393e+04   0.000    1.000
## no_emission_per          -1.360e+01  1.572e+05   0.000    1.000
## agri_forest_fish_gdp_per  7.137e+01  1.180e+05   0.001    1.000
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7.0252e+01  on 52  degrees of freedom
## Residual deviance: 2.5719e-09  on 38  degrees of freedom
## AIC: 30
##
## Number of Fisher Scoring iterations: 25
```

```
log_res_2010$.extracts[[1]][[1]]
```

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

```
## # A tibble: 14 x 5
##    term                estimate std.error statistic p.value
##    <chr>                  <dbl>     <dbl>     <dbl>   <dbl>
##  1 (Intercept)            274.     74548.  0.00368    0.997
##  2 agrid_land_per          42.3    50208.  0.000843   0.999
##  3 arable_per             -58.7    35391. -0.00166    0.999
##  4 forest_per            -152.    110524. -0.00137    0.999
##  5 cereal_yield          -200.     99034. -0.00202    0.998
##  6 pop_growth            -108.     80567. -0.00134    0.999
##  7 urban_pop_per         -187.     64547. -0.00290    0.998
##  8 for_invest_per         -70.8    22232. -0.00319    0.997
##  9 school_enroll         -214.     82304. -0.00260    0.998
## 10 school_complete        -74.1   139194. -0.000532   1.00
## 11 co2_emission_2017       43.3    32118.  0.00135    0.999
## 12 other_greenhouse_mt     17.4    27174.  0.000639   0.999
## 13 total_greenhouse_mt     23.8    50238.  0.000473   1.00
## 14 no_emission_per        305.    123641.  0.00247    0.998
```

```
all_coef <- map_dfr(log_res_2010$.extracts, ~ .x[[1]][[1]])

glm_metrics <- log_res_2010 |>  collect_metrics(summarize = TRUE)

glm_metrics
```

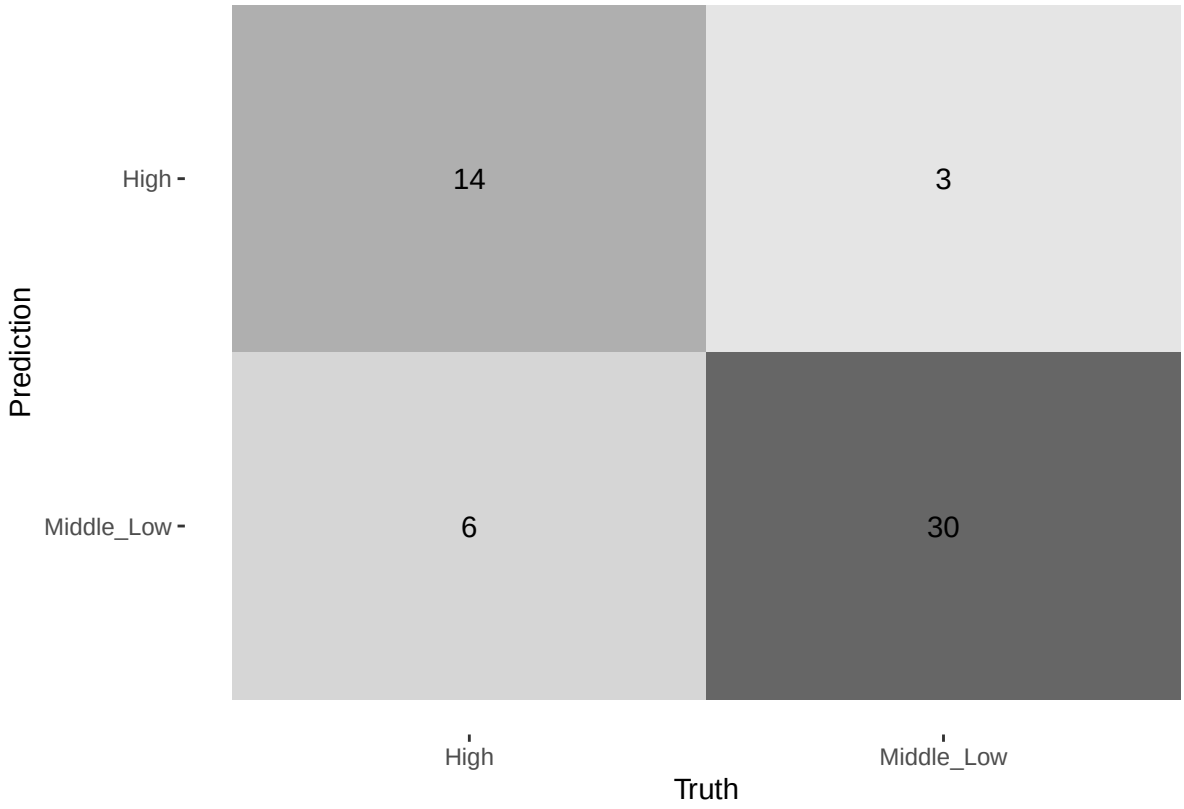
.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.8266667	10	0.0695311	Preprocessor1_Model1
f_meas	binary	0.7740741	9	0.1143659	Preprocessor1_Model1
kap	binary	0.6041126	10	0.1592156	Preprocessor1_Model1
precision	binary	0.7962963	9	0.1171214	Preprocessor1_Model1
recall	binary	0.7000000	10	0.1333333	Preprocessor1_Model1
roc_auc	binary	0.8354167	10	0.0750803	Preprocessor1_Model1
sens	binary	0.7000000	10	0.1333333	Preprocessor1_Model1
spec	binary	0.9083333	10	0.0472222	Preprocessor1_Model1

As we can see, after validation, the accuracy of the model is ok, providing an accurate classification about 71 percent of the time. The precision of the model is also ok providing a very similar value accuracy with it providing a 71.6% ability to label the positives correctly. The sensitivity is where the model really starts to fall off with the true positive rate only being 55%. The F Score is another performance metric that utilizes precision and sensitivity to determine the accuracy of the model and in this case, it gives the glm logistic regression a score of 59.6%.

```
log_pred <-
  log_res_2010 |>
  collect_predictions()

log_pred |>
  conf_mat(income_status, .pred_class)
```

```
##             Truth
## Prediction   High Middle_Low
##   High         14          3
##   Middle_Low    6         30
```

```
log_pred |>
  conf_mat(income_status, .pred_class) |>
  autoplot(type = "heatmap")
```



GLM Conclusion

We can conclude with this model that there is more work to be done to possibly get these climate change indicators to classify whether or not a country is part of the high income or low-middle income classification. Originally, when I had first run this model, I had thought that the indicators are inadequate indicators of a nations status as a high income nation, however, I did some research and discovered that both LASSO and Elastic Net logistic regression can be also be used to classify and have the ability to overcome the issues present in the R GLM function of complete separation by their ability to apply penalties.

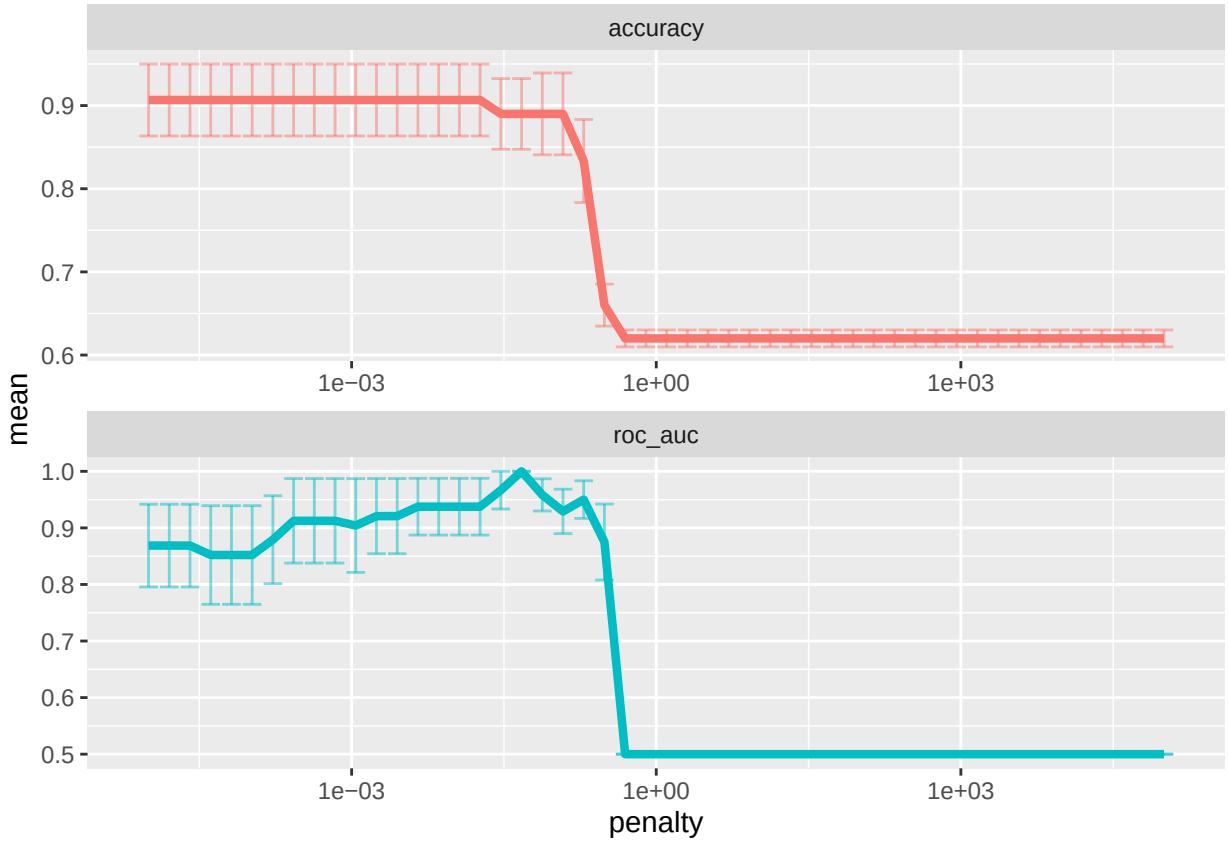
LASSO Logistic Regression

The LASSO is particularly useful because it provides variable selection as well as shrinking the indicators to the best possible level at the expense of a slight increase in bias within the data. This is done by tuning the lambda "tuning" value to an optimal position. In this case, the lambda is selected by running a tuning grid using the dials library in R. Then the best lambda is selected based on accuracy.

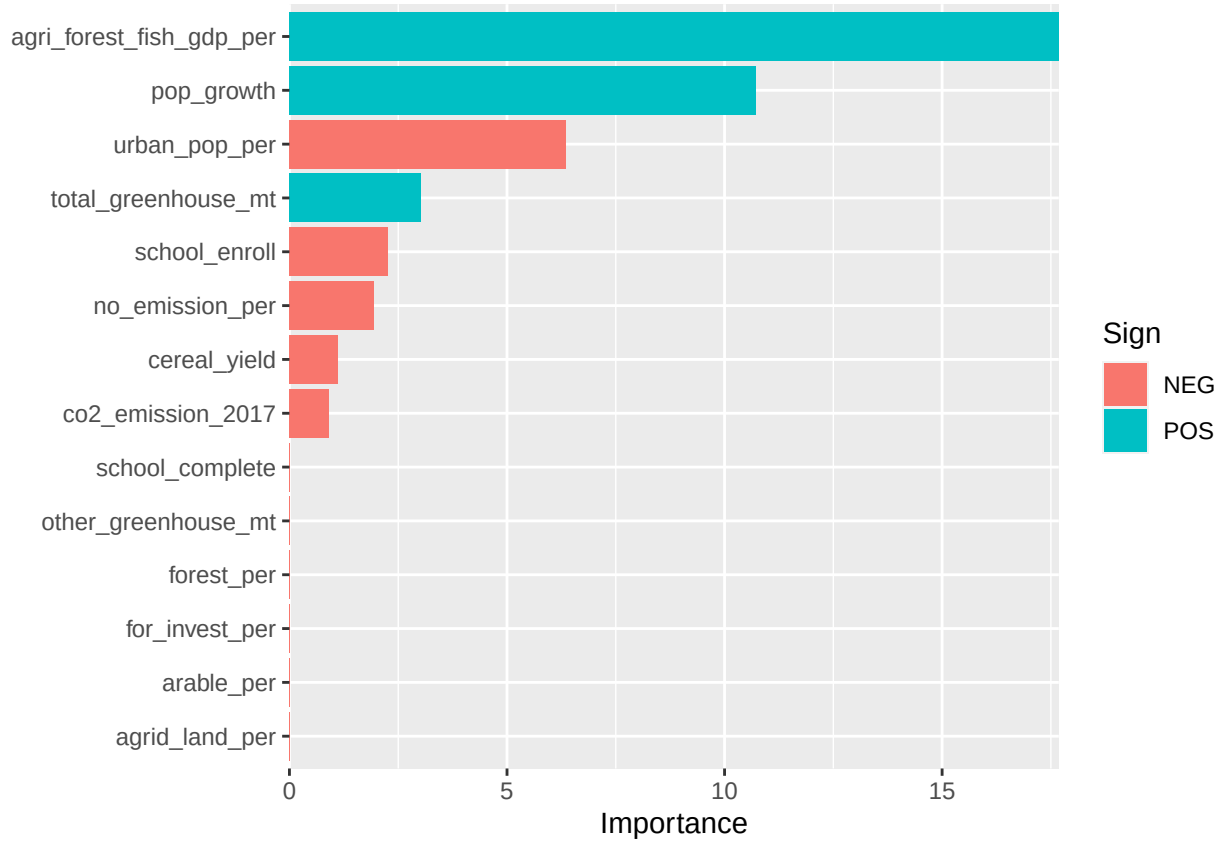
LASSO Fit & Visuals

```
# A very Cool Graph Based on Silge Work
lasso_grid |>
  collect_metrics() |>
  ggplot(aes(penalty, mean, color = .metric)) +
  geom_errorbar(aes(
    ymin = mean - std_err,
    ymax = mean + std_err
  ),
  alpha = 0.5
  ) +
  geom_line(size = 1.5) +
  facet_wrap(~.metric, scales = "free", nrow = 2) +
```

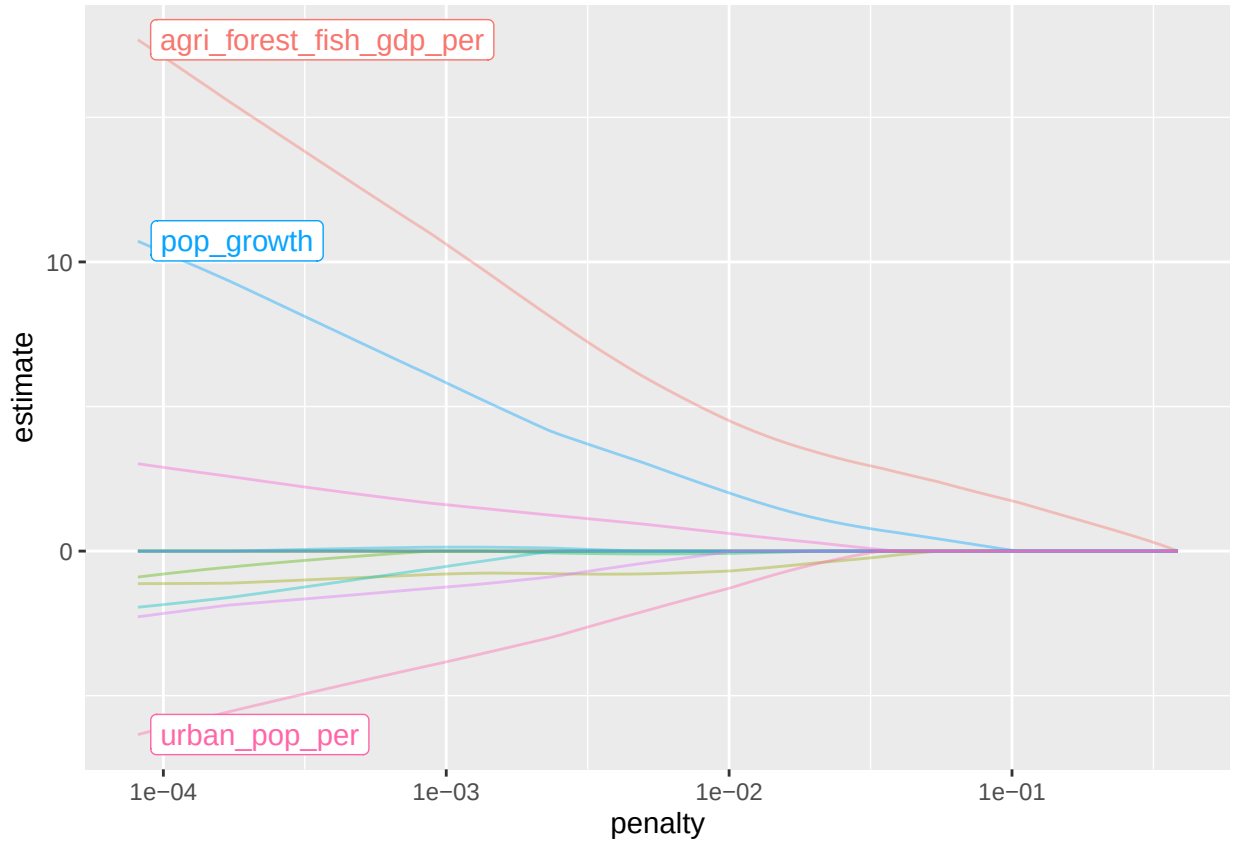
```
  scale_x_log10() +
  theme(legend.position = "none")
```




```
# A very nice graph that shows the predictors in Columns
final_lasso |>
  fit(train_2010) |>
  extract_fit_parsnip() |>
  vi(lambda = best_lasso_accuracy$penalty) |>
  mutate(
    Importance = abs(Importance),
    Variable = fct_reorder(Variable, Importance)
  ) |>
  ggplot(aes(x = Importance, y = Variable, fill = Sign)) +
  geom_col() +
  scale_x_continuous(expand = c(0,0)) +
  labs(y = NULL)
```




```
# Regression Coefficient Path
final_lasso |>
  fit(data = train_2010) |>
  extract_fit_engine() |>
  autoplot()
```



```
final_lasso |>
  fit(data = train_2010) |>
  extract_fit_parsnip() |>
  tidy()
```

term	estimate	penalty
(Intercept)	15.0604950	1e-05
agrid_land_per	0.0000000	1e-05
arable_per	0.0000000	1e-05
forest_per	0.0000000	1e-05
cereal_yield	-1.1261141	1e-05
pop_growth	10.7154812	1e-05
urban_pop_per	-6.3448554	1e-05
for_invest_per	0.0000000	1e-05
school_enroll	-2.2723698	1e-05
school_complete	0.0000000	1e-05
co2_emission_2017	-0.8983436	1e-05
other_greenhouse_mt	0.0000000	1e-05
total_greenhouse_mt	3.0195716	1e-05
no_emission_per	-1.9404782	1e-05
agri_forest_fish_gdp_per	17.6826277	1e-05

LASSO Conclusion

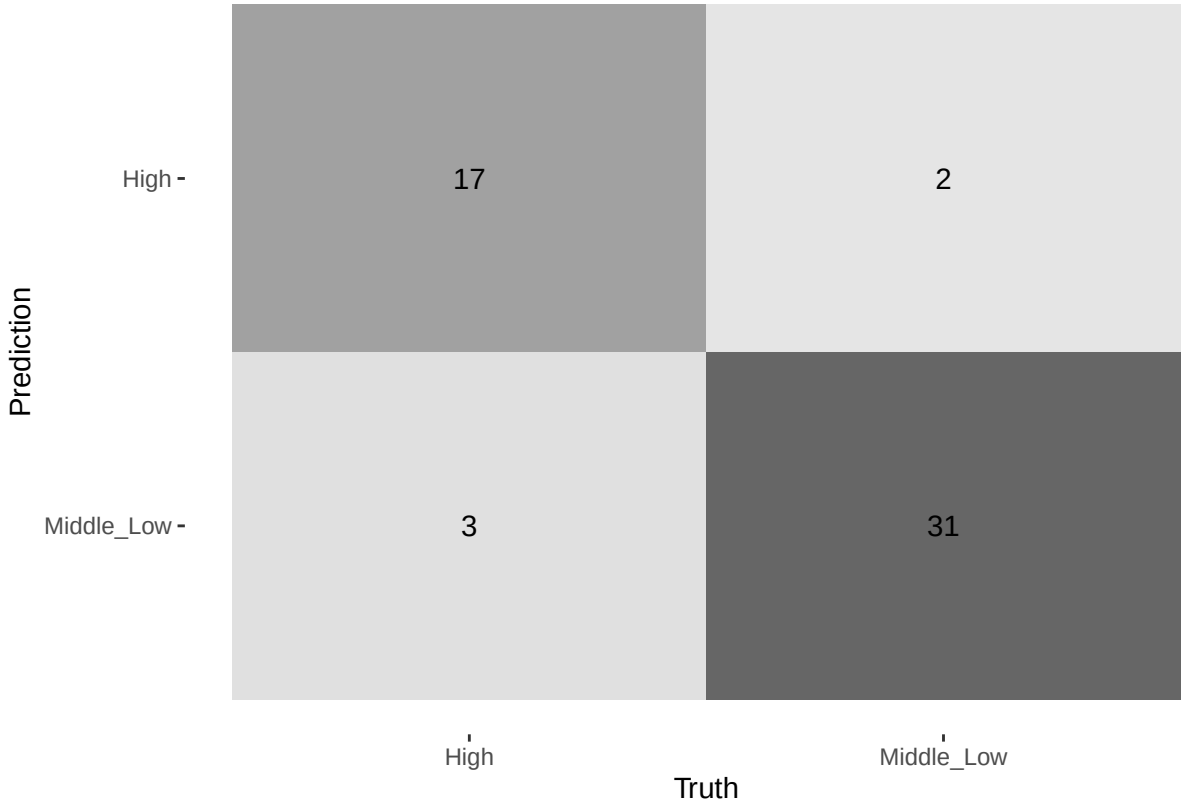
```
log_lasso_2010$.extracts[[1]][[1]]
```

```
## [[1]]
## # A tibble: 14 x 3
##    term                estimate penalty
##    <chr>                  <dbl>   <dbl>
##  1 (Intercept)            63.7  0.00001
##  2 agrid_land_per          4.62 0.00001
##  3 arable_per            -14.5  0.00001
##  4 forest_per            -42.3  0.00001
##  5 cereal_yield          -41.6  0.00001
##  6 pop_growth            -17.4  0.00001
##  7 urban_pop_per         -38.6  0.00001
##  8 for_invest_per        -18.1  0.00001
##  9 school_enroll         -47.3  0.00001
## 10 school_complete       -13.1  0.00001
## 11 co2_emission_2017      11.5  0.00001
## 12 other_greenhouse_mt     4.78 0.00001
## 13 total_greenhouse_mt     1.90 0.00001
## 14 no_emission_per        60.9  0.00001
```

```
all_coef <- map_dfr(log_res_2010$.extracts, ~ .x[[1]][[1]])

lasso_metrics <- log_lasso_2010 |>  collect_metrics(summarize = TRUE)

lasso_metrics
```

.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.9066667	10	0.0432620	Preprocessor1_Model1
f_meas	binary	0.9185185	9	0.0427132	Preprocessor1_Model1
kap	binary	0.7853480	10	0.1044541	Preprocessor1_Model1
precision	binary	0.9259259	9	0.0489954	Preprocessor1_Model1
recall	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
roc_auc	binary	0.8687500	10	0.0731016	Preprocessor1_Model1
sens	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
spec	binary	0.9416667	10	0.0393818	Preprocessor1_Model1

```
log_pred_lasso <-
  log_lasso_2010 |>
  collect_predictions()

log_pred_lasso |>
  conf_mat(income_status, .pred_class) |>
  autoplot(type = "heatmap")
```



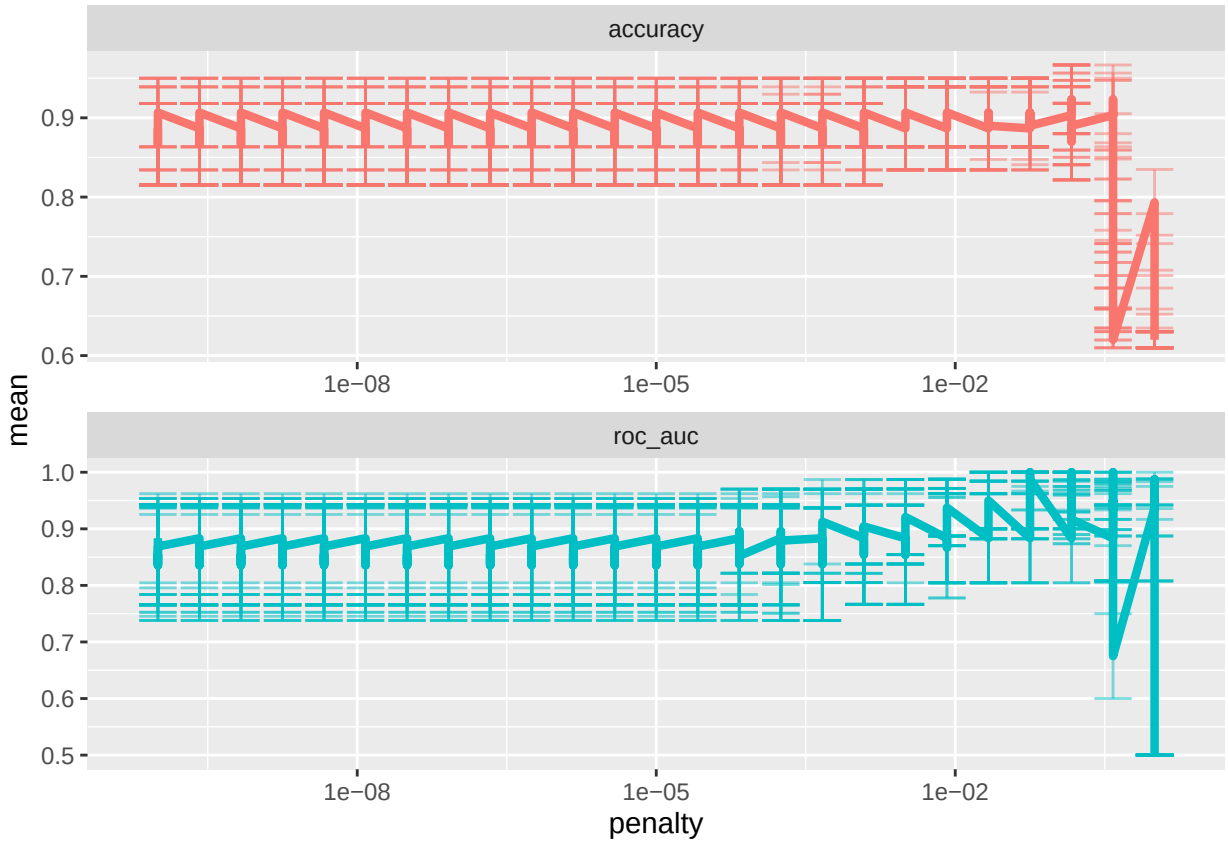
The effects of utilizing LASSO in this classification model is tremendous. It increasing the accuracy to 90%. The sensitivity is now 85%. This model is now much better at predicting the income level of a country based on climate change indicators. The model is not perfect, but it does now start to drive some conclusions that these indicators can classify what income status a country is.

Elastic Net Logistic Regression

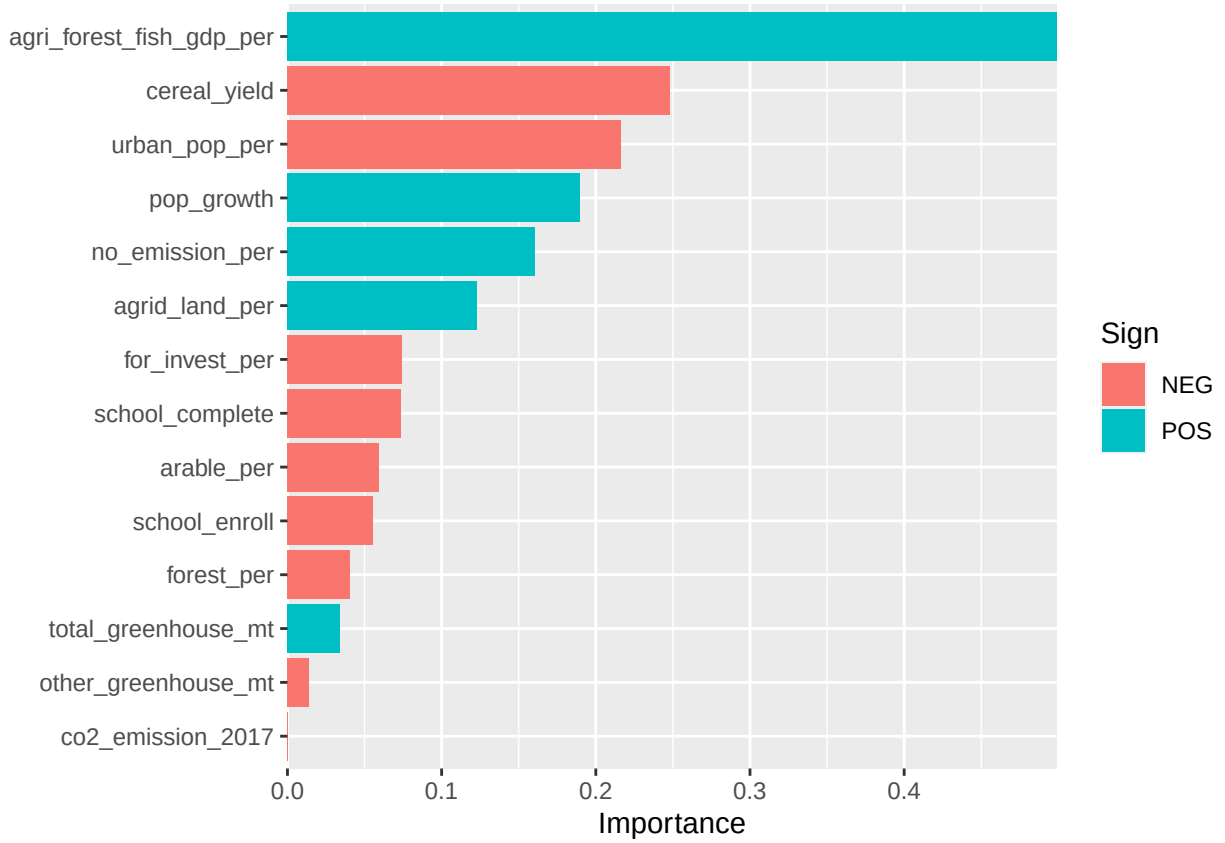
The last model utilized in this study is the Elastic Net Model. This is used when it is required to utilize regularization methods but there is a trade-off between variable selection and smaller coefficients. With the LASSO performing well, I decided to run Elastic Net as the final model over Ridge Regression since the LASSO eliminated 6 of the 14 indicators and had good metrics so I thought I would try to bring that compromise into the model.

Elastic Net Fit & Visuals

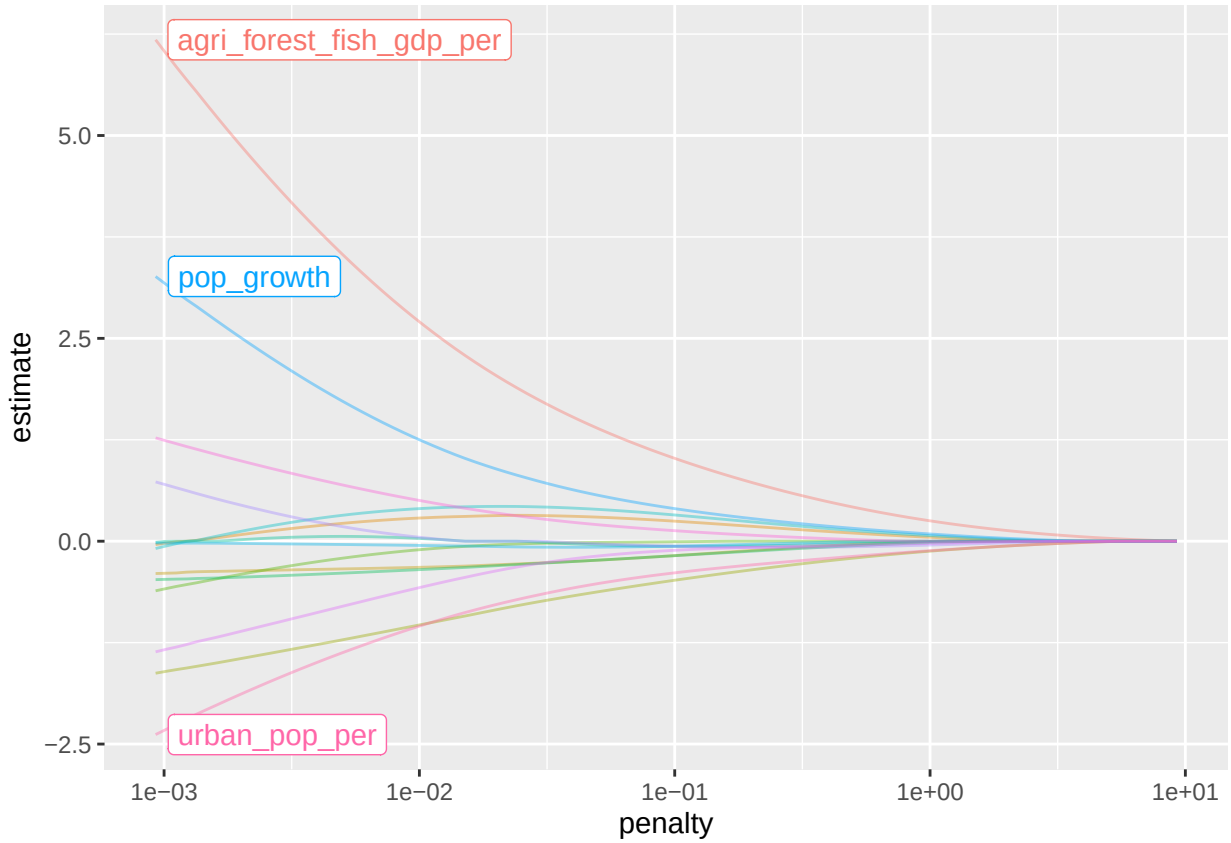
```
# A very Cool Graph Based on Silge Work
elastic_grid |>
  collect_metrics() |>
  ggplot(aes(penalty, mean, color = .metric)) +
  geom_errorbar(aes(
    ymin = mean - std_err,
    ymax = mean + std_err
  ),
  alpha = 0.5
  ) +
  geom_line(size = 1.5) +
  facet_wrap(~.metric, scales = "free", nrow = 2) +
  scale_x_log10() +
  theme(legend.position = "none")
```




```
# A very nice graph that shows the predictors in Columns
final_elastic |>
  fit(train_2010) |>
  extract_fit_parsnip() |>
  vi(lambda = best_elastic_accuracy$penalty) |>
  mutate(
    Importance = abs(Importance),
    Variable = fct_reorder(Variable, Importance)
  ) |>
  ggplot(aes(x = Importance, y = Variable, fill = Sign)) +
  geom_col() +
  scale_x_continuous(expand = c(0,0)) +
  labs(y = NULL)
```




```
# Regression Coefficient Path
final_elastic |>
  fit(data = train_2010) |>
  extract_fit_engine() |>
  autoplot()
```



```
final_elastic |>
  fit(data = train_2010) |>
  extract_fit_parsnip() |>
  tidy()
```

term	estimate	penalty
(Intercept)	0.6336905	0.3831187
agrid_land_per	0.1229218	0.3831187
arable_per	-0.0592898	0.3831187
forest_per	-0.0404739	0.3831187
cereal_yield	-0.2478799	0.3831187
pop_growth	0.1898463	0.3831187
urban_pop_per	-0.2162546	0.3831187
for_invest_per	-0.0742449	0.3831187
school_enroll	-0.0553430	0.3831187
school_complete	-0.0739073	0.3831187
co2_emission_2017	0.0000000	0.3831187
other_greenhouse_mt	-0.0142635	0.3831187
total_greenhouse_mt	0.0338404	0.3831187
no_emission_per	0.1605797	0.3831187
agri_forest_fish_gdp_per	0.4990210	0.3831187

Elastic Net Conclusion

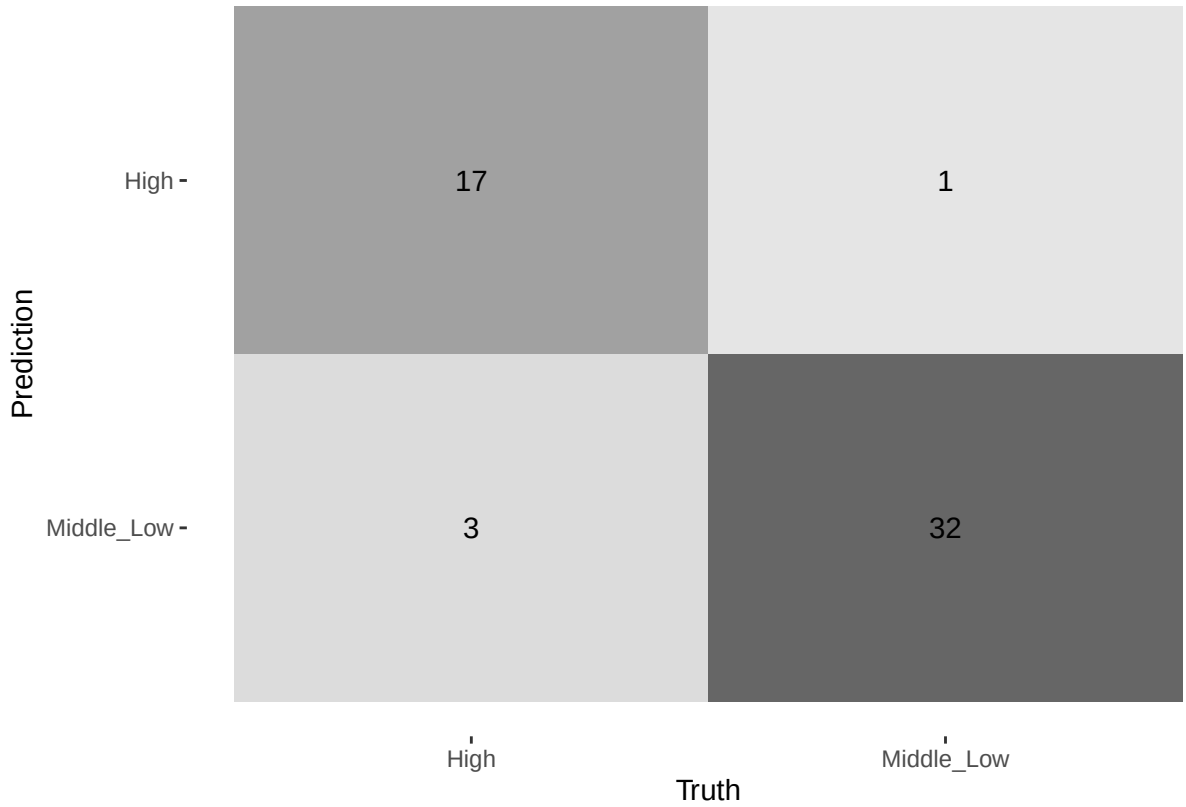
```
log_elastic_2010$.extracts[[1]][[1]]
```

```
## [[1]]
## # A tibble: 14 x 3
##    term                estimate penalty
##    <chr>                  <dbl>   <dbl>
##  1 (Intercept)           0.568     0.383
##  2 agrid_land_per        0.137     0.383
##  3 arable_per           -0.0500    0.383
##  4 forest_per           -0.0399    0.383
##  5 cereal_yield         -0.317     0.383
##  6 pop_growth            0.179     0.383
##  7 urban_pop_per        -0.299     0.383
##  8 for_invest_per       -0.0988    0.383
##  9 school_enroll        -0.0790    0.383
## 10 school_complete      -0.141     0.383
## 11 co2_emission_2017    -0.0169    0.383
## 12 other_greenhouse_mt -0.00530    0.383
## 13 total_greenhouse_mt   0.0366    0.383
## 14 no_emission_per       0.201     0.383
```

```
all_coef <- map_dfr(log_elastic_2010$.extracts, ~ .x[[1]][[1]])

elastic_metrics <- log_elastic_2010 |>  collect_metrics(summarize = TRUE)

elastic_metrics
```

.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.9233333	10	0.0433333	Preprocessor1_Model1
f_meas	binary	0.9407407	9	0.0407407	Preprocessor1_Model1
kap	binary	0.8186813	10	0.1055587	Preprocessor1_Model1
precision	binary	0.9629630	9	0.0370370	Preprocessor1_Model1
recall	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
roc_auc	binary	0.9208333	10	0.0506326	Preprocessor1_Model1
sens	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
spec	binary	0.9666667	10	0.0333333	Preprocessor1_Model1

```
log_pred_elastic <-
  log_elastic_2010 |>
  collect_predictions()

log_pred_elastic |>
  conf_mat(income_status, .pred_class) |>
  autoplot(type = "heatmap")
```



The Elastic Net chose a penalty of .38 and yielded the best classification metrics of any of the models shown here. The accuracy is 92% and the sensitivity is 85%. The heatmap of the confusion matrix also shows that it has an improvement over those of the GLM and LASSO fit.

```
testing_2010
```

.pred_class...1	.pred_class...2	.pred_class...3	income_status
High	High	Middle_Low	Middle_Low
High	High	High	High
Middle_Low	Middle_Low	Middle_Low	Middle_Low
High	High	High	High
Middle_Low	Middle_Low	Middle_Low	Middle_Low
Middle_Low	Middle_Low	Middle_Low	Middle_Low
Middle_Low	Middle_Low	Middle_Low	Middle_Low
Middle_Low	Middle_Low	Middle_Low	Middle_Low
High	High	High	High
Middle_Low	Middle_Low	Middle_Low	Middle_Low
Middle_Low	Middle_Low	Middle_Low	Middle_Low
High	High	High	High
High	High	High	High
High	High	High	High
Middle_Low	Middle_Low	Middle_Low	Middle_Low

```
glm_metrics
```

.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.8266667	10	0.0695311	Preprocessor1_Model1
f_meas	binary	0.7740741	9	0.1143659	Preprocessor1_Model1
kap	binary	0.6041126	10	0.1592156	Preprocessor1_Model1
precision	binary	0.7962963	9	0.1171214	Preprocessor1_Model1
recall	binary	0.7000000	10	0.1333333	Preprocessor1_Model1
roc_auc	binary	0.8354167	10	0.0750803	Preprocessor1_Model1
sens	binary	0.7000000	10	0.1333333	Preprocessor1_Model1
spec	binary	0.9083333	10	0.0472222	Preprocessor1_Model1

```
lasso_metrics
```

.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.9066667	10	0.0432620	Preprocessor1_Model1
f_meas	binary	0.9185185	9	0.0427132	Preprocessor1_Model1
kap	binary	0.7853480	10	0.1044541	Preprocessor1_Model1
precision	binary	0.9259259	9	0.0489954	Preprocessor1_Model1
recall	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
roc_auc	binary	0.8687500	10	0.0731016	Preprocessor1_Model1
sens	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
spec	binary	0.9416667	10	0.0393818	Preprocessor1_Model1

```
elastic_metrics
```

.metric	.estimator	mean	n	std_err	.config
accuracy	binary	0.9233333	10	0.0433333	Preprocessor1_Model1
f_meas	binary	0.9407407	9	0.0407407	Preprocessor1_Model1
kap	binary	0.8186813	10	0.1055587	Preprocessor1_Model1
precision	binary	0.9629630	9	0.0370370	Preprocessor1_Model1
recall	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
roc_auc	binary	0.9208333	10	0.0506326	Preprocessor1_Model1
sens	binary	0.8500000	10	0.1067187	Preprocessor1_Model1
spec	binary	0.9666667	10	0.0333333	Preprocessor1_Model1

Conclusion

After a disheartening beginning to this project, it actually yielded quite exciting results. The results of the GLM could have drawn the conclusion that a countries income status was not distinguishable from its climate change indicators, but the LASSO and Elastic Net models were able to show that these indicators can show the income status of a country. I think this is a great starting point for continued analysis into world development indicators and their ability to classify. The next steps I would

take with this model is to take these indicators and check them against the rest of the data and see if the 14 indicators can be replicated throughout the years. That was a major hurdle in this analysis was simply getting enough complete data to look at the entire dataset. If this can be further validated against other years, then it could uncover the true difference between high income and low income nations when it comes to climate change. I would also dig deeper into the data and determine other variables and try to pull them out, specifically bigger indicators around power usage. The original model had these values within them, but I think more time put into the model by eliminated variation could show other interesting insights. It may be possible to isolate types of indicators and run models based on those types instead of such a wide swath of climate indicators. In closing, this was a very interesting project that taught me new modeling techniques and answered some of the basic questions around climate indicators as well as opened the door for more research and modeling to look into other indicators or different manifestions of the climate indicators from the World Bank.

Bibliography

Kirenz, J. (2021, February 16). Classification with Tidymodels, Workflows and Recipes. Retrieved from Jan Kirenz: https://www.kirenz.com/post/2021-02-17-r-classification-tidymodels/

Kuhn, M., & Silge, J. (2022, April 15). Tidy Modeling with R. Retrieved from tmwr: https://www.tmwr.org/

World Bank Group. (2020, April 4). Accessing International Debt Statistics (IDS) through the World Bank Data API. Retrieved from github: https://worldbank.github.io/debt-data/api-guide/ids-api-guide-r-2.html

World Bank Group. (2022, January 1). Climate Change Knowledge Portal. Retrieved from Climate Change Knowledge Portal: https://climateknowledgeportal.worldbank.org/