LASSO Regression

Jaime Davila

4/6/2022

Previously on ADM

Let's recall the steps to set-up our ridge regression model

Dataset loading and training/testing split

We can load our dataset by doing:

```
library(ISLR2)
data(Credit)
credit.tbl <- as_tibble(Credit)
```

And create our training/testing datasets by:

```
library(tidymodels)
tidymodels_prefer()

set.seed(654321)
credit.split <- initial_split(credit.tbl, prop=0.8)
credit.train.tbl <- training(credit.split)
credit.test.tbl <- testing(credit.split)
```

Setting up the ridge regression model

First we set-up our ridge model below. Notice that `mixture=0` and that set-up our `penalty` (or λ) to be tuned later on:

```
ridge.model <-
  linear_reg(mixture = 0, penalty=tune()) %>% #tune()
  set_mode("regression") %>%
  set_engine("glmnet")
```

Below we create our recipe and our workflow. Notice that we need to use `step_normalize()` to make sure all the variables are standardized.

```
ridge.recipe <-
  recipe(formula = Balance ~ ., data = credit.train.tbl) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_normalize(all_predictors())

ridge.wf <- workflow() %>%
  add_recipe(ridge.recipe) %>%
  add_model(ridge.model)
```

Setting up the parameters for the optimization

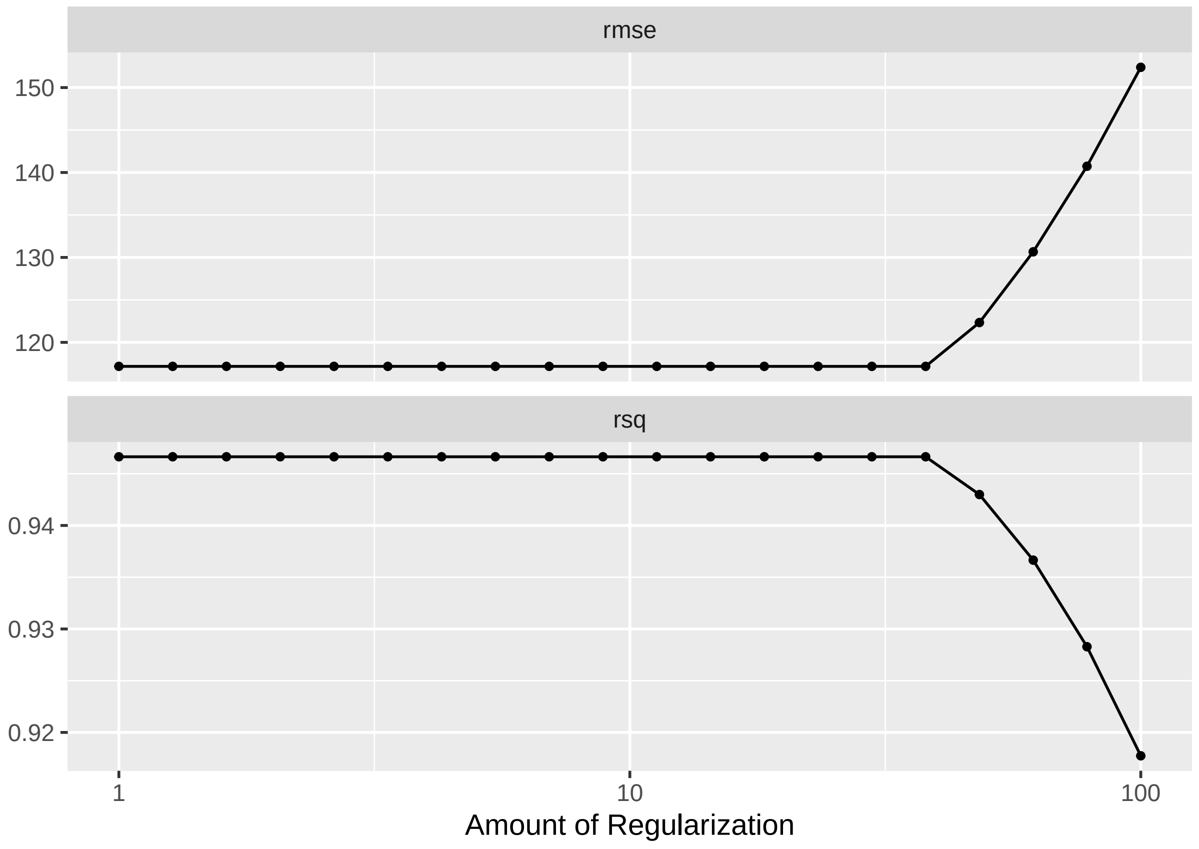
Below we create our 10-fold cross-validation dataset using `vfold_cv()` and we create a logarithmic grid using `penalty()`

```
credit.fold <- vfold_cv(credit.train.tbl, v = 10)

penalty.grid <-
  grid_regular(penalty(range = c(0, 2)), levels = 20)
```

Optimizing the penalty (λ)

The following code allows to optimize the penalty parameter

```
tune.res <- tune_grid(
  ridge.wf,
  resamples = credit.fold,
  grid = penalty.grid
)
autoplot(tune.res)
```



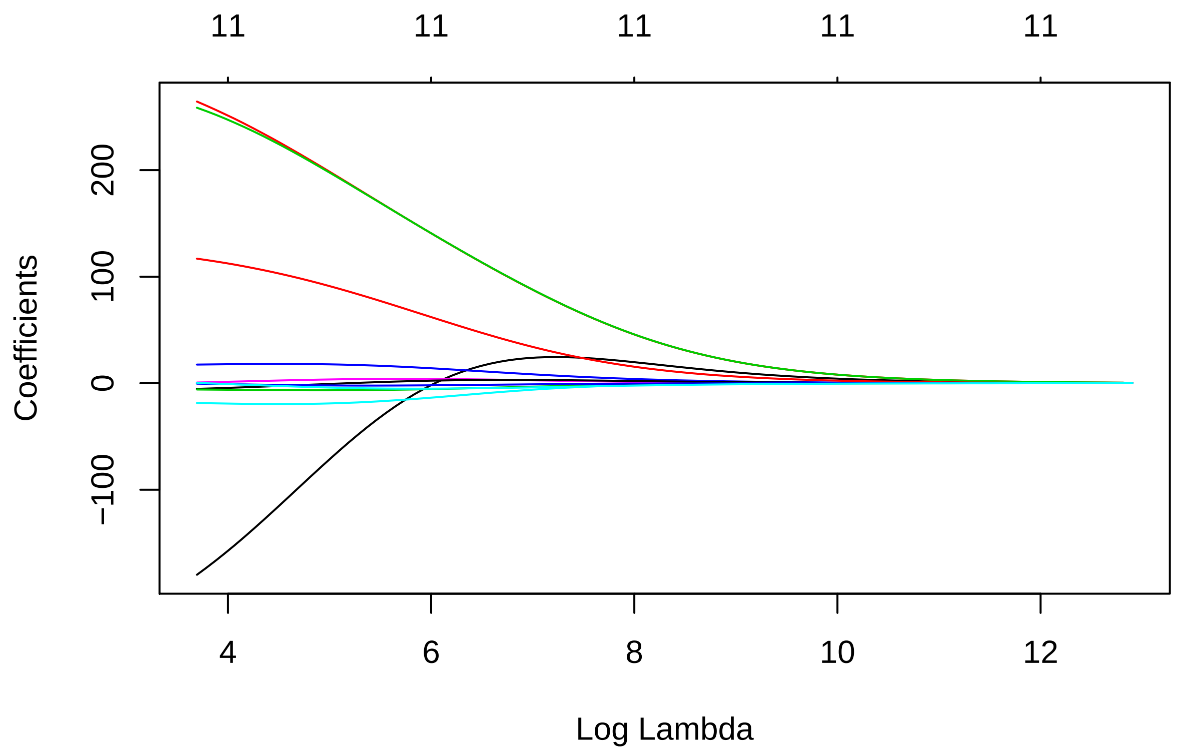
And finally we can:

- Select the best parameter
- Explore the influence of lambda on the coefficients
- Calculate our R^2 using the testing dataset
- Show the importance of the variables

```
(best.penalty <- select_best(tune.res, metric = "rsq"))
```

```
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <fct>
## 1       1 Preprocessor1_Model01
```

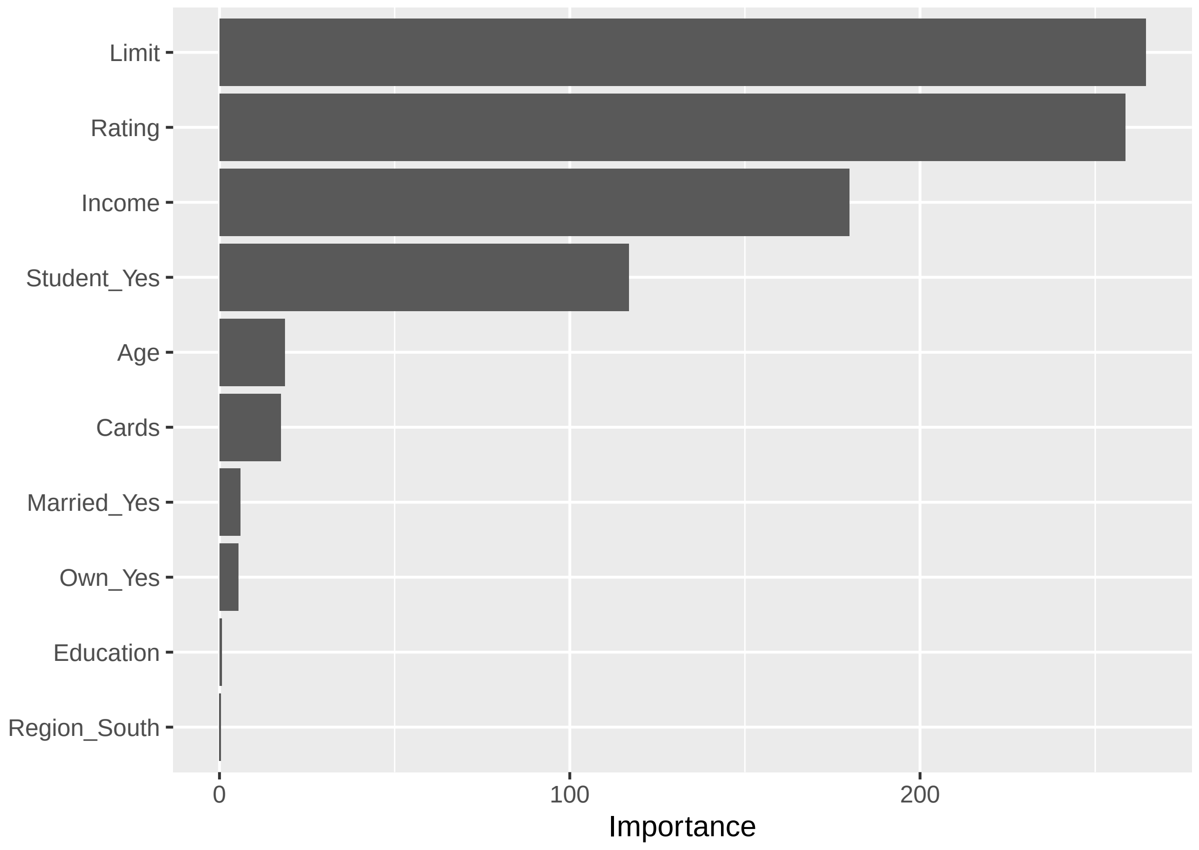
```
ridge.final.wf <- finalize_workflow(ridge.wf, best.penalty)
ridge.final.fit <- fit(ridge.final.wf, data = credit.train.tbl)

ridge.final.fit %>%
  extract_fit_engine() %>%
  plot(xvar = "lambda")
```



```
augment(ridge.final.fit, new_data = credit.test.tbl) %>%
  rsq(truth = Balance, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.932
```

```r
library(vip)
extract_fit_parsnip(ridge.final.fit) %>%
  vip()
```



note: we only calculate imporatnce of variables, not significance.

LASSO regression

When we were dealing with ridge regression we minimized the following function:

$$RSS(\beta_0, \ldots, \beta_p) + \lambda \sum_{i=1}^{p} \beta_j^2$$

The key thing to realize is that we are optimizing over the choice of our coefficients $\beta_0, \ldots, \beta_p$ and that λ is a fixed value which will be finding later on.

In LASSO we minimize a slightly different optimization function with profound consequences on our model. The function is:

$$RSS(\beta_0, \ldots, \beta_p) + \lambda \sum_{i=1}^{p} |\beta_j|$$

The following exercises will walk you through the setting up of the LASSO model and how the result differs from the ridge regression

1. Set up a LASSO regression model by using `tidymodels()` making sure you use the parameter `mixture=1`. For your λ use a value of 1. Calculate the R^2 and the order of importance of variables

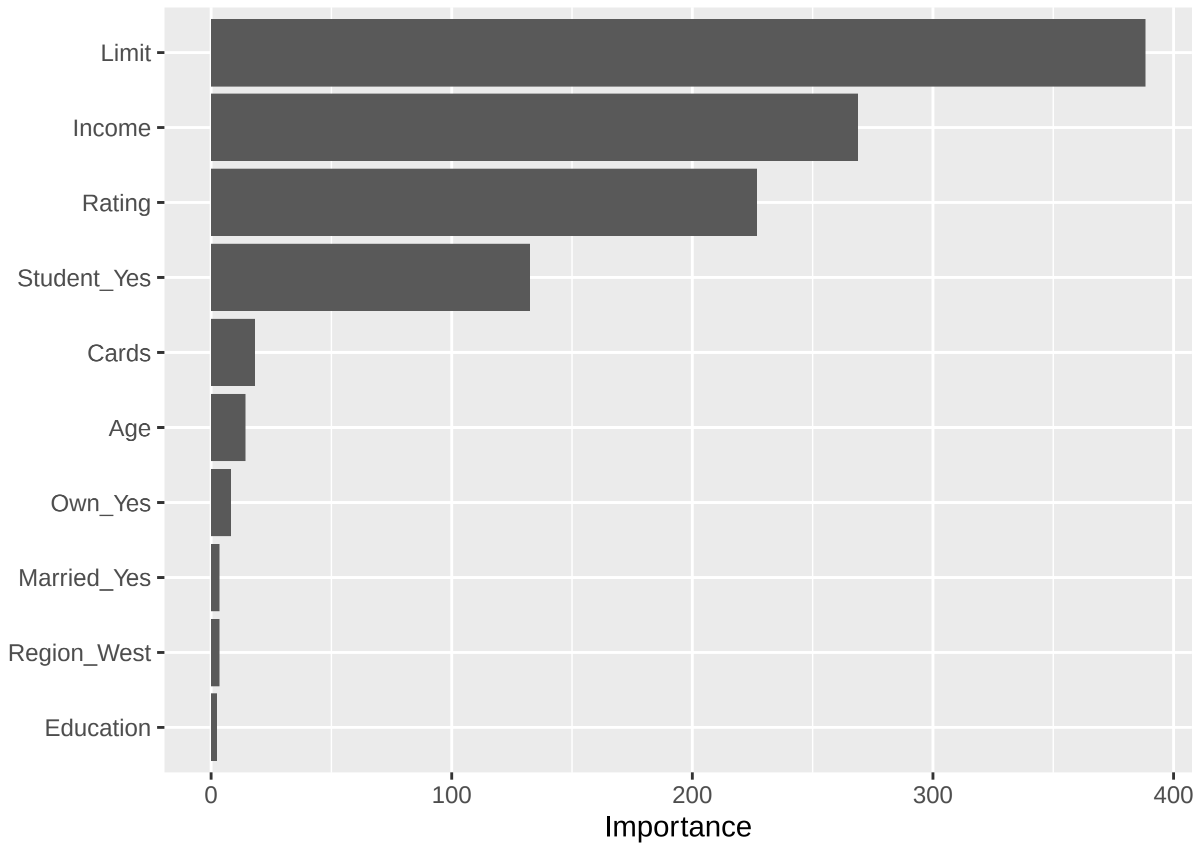
Note: mixture = 1 is lasso, mixture = 0 is ridge. The names are "ridge" but it should be lasso.

ridge: when variables are very correlated, it shrinks them by the same amount.

```
ridge.model <-
  linear_reg(mixture = 1, penalty=1) %>%
  set_mode("regression") %>%
  set_engine("glmnet")

ridge.recipe <-
  recipe(formula = Balance ~ ., data = credit.train.tbl) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_normalize(all_predictors())

ridge.wf <- workflow() %>%
  add_recipe(ridge.recipe) %>%
  add_model(ridge.model)

ridge.fit <- fit(ridge.wf, credit.train.tbl)
tidy(ridge.fit)
```

```
## # A tibble: 12 x 3
##    term         estimate penalty
##    <chr>           <dbl>   <dbl>
##  1 (Intercept)    529.         1
##  2 Income        -267.         1
##  3 Limit          381.         1
##  4 Rating         232.         1
##  5 Cards           17.5        1
##  6 Age            -14.0        1
##  7 Education       -1.85       1
##  8 Own_Yes         -7.78       1
##  9 Student_Yes    132.         1
## 10 Married_Yes     -3.05       1
## 11 Region_South     0          1
## 12 Region_West      2.68       1
```

```
extract_fit_parsnip(ridge.fit) %>%
  vip()
```



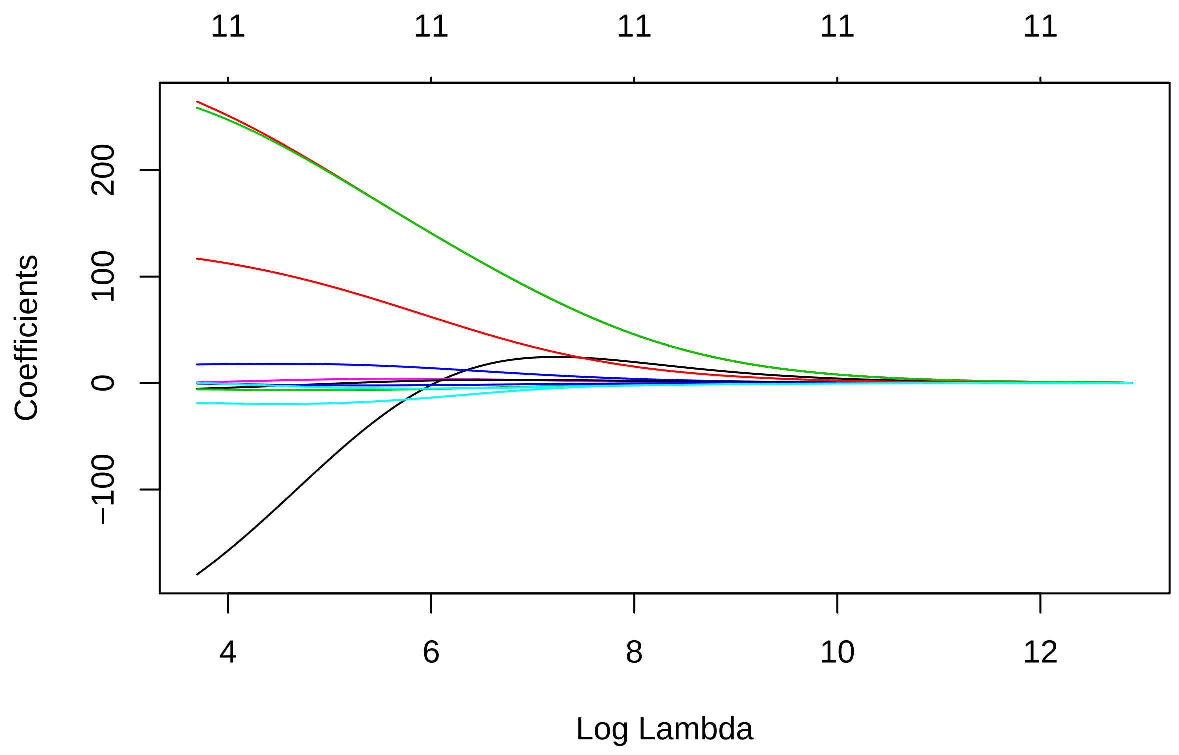
```
augment(ridge.fit, new_data = credit.test.tbl) %>%
  rsq(truth = Balance, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.946
```

r^2 = 0.946

2. Compare the following plot to fig 6.6 from ISLR (left panel). How do you interpret this plot? What do you think the numbers on the upper side of the plot mean (Notice they go like 10,9,7,5,4,2,0)? How is this plot different from the corresponding plot from the ridge regression?

```
ridge.final.fit %>%
  extract_fit_engine() %>%
  plot(xvar = "lambda")
```



There are 4 big coefficients. The other coefficients are relatively small. There is a point where all the small coefficients converge to 0. Lasso makes them all exactly equal to 0 whereas, with ridge regression, the coefficients aproach zero and all lines converge near the end of the x-axis.

The scale on x-axis is logarithmic. The top of the x-axis may represent the number of coefficients remaining. The plot shows how large remaining coefficient values are in addition to how many coefficients remain depending on the value of lambda.

3. Experiment with your penalty grid and use cross-validation optimize the parameter λ. Calculate your R^2 on your testing dataset and determine the importance of the features. Compare this to your results using ridge regression.

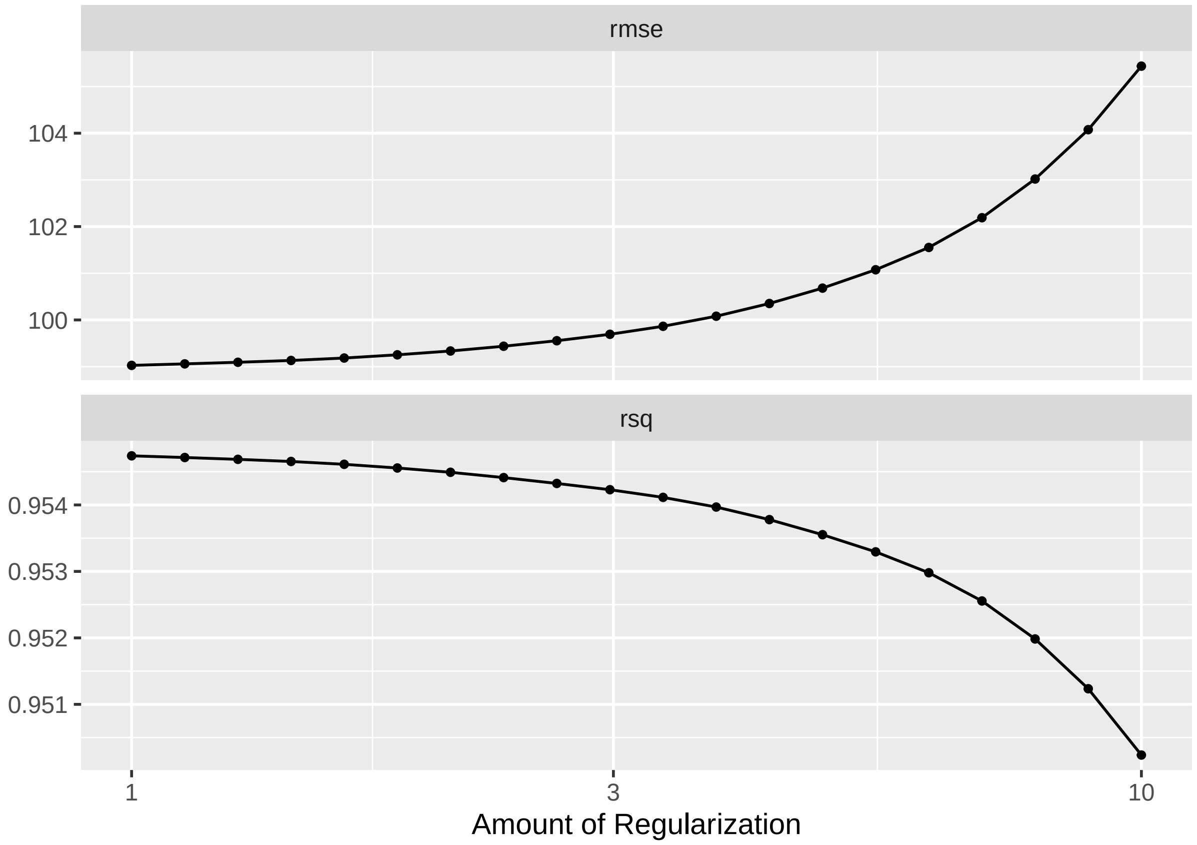
```
ridge.model <-
  linear_reg(mixture = 1, penalty=tune()) %>%
  set_mode("regression") %>%
  set_engine("glmnet")
```

```
ridge.recipe <-
  recipe(formula = Balance ~ ., data = credit.train.tbl) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_normalize(all_predictors())
ridge.wf <- workflow() %>%
  add_recipe(ridge.recipe) %>%
  add_model(ridge.model)
```

```
credit.fold <- vfold_cv(credit.train.tbl, v = 10)

penalty.grid <-
```

```
  grid_regular(penalty(range = c(0, 1)), levels = 20)
```

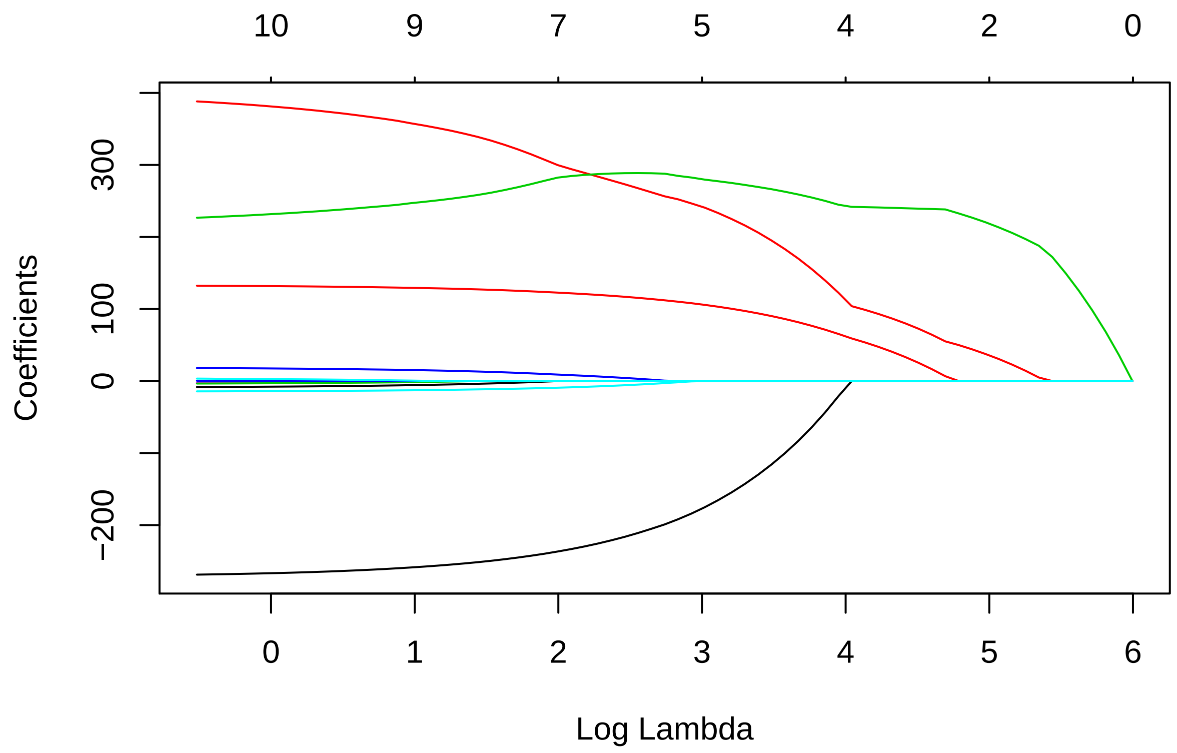
```
tune.res <- tune_grid(
  ridge.wf,
  resamples = credit.fold,
  grid = penalty.grid
)
autoplot(tune.res)
```



```
(best.penalty <- select_best(tune.res, metric = "rsq"))
```

```
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <fct>
## 1       1 Preprocessor1_Model01
```

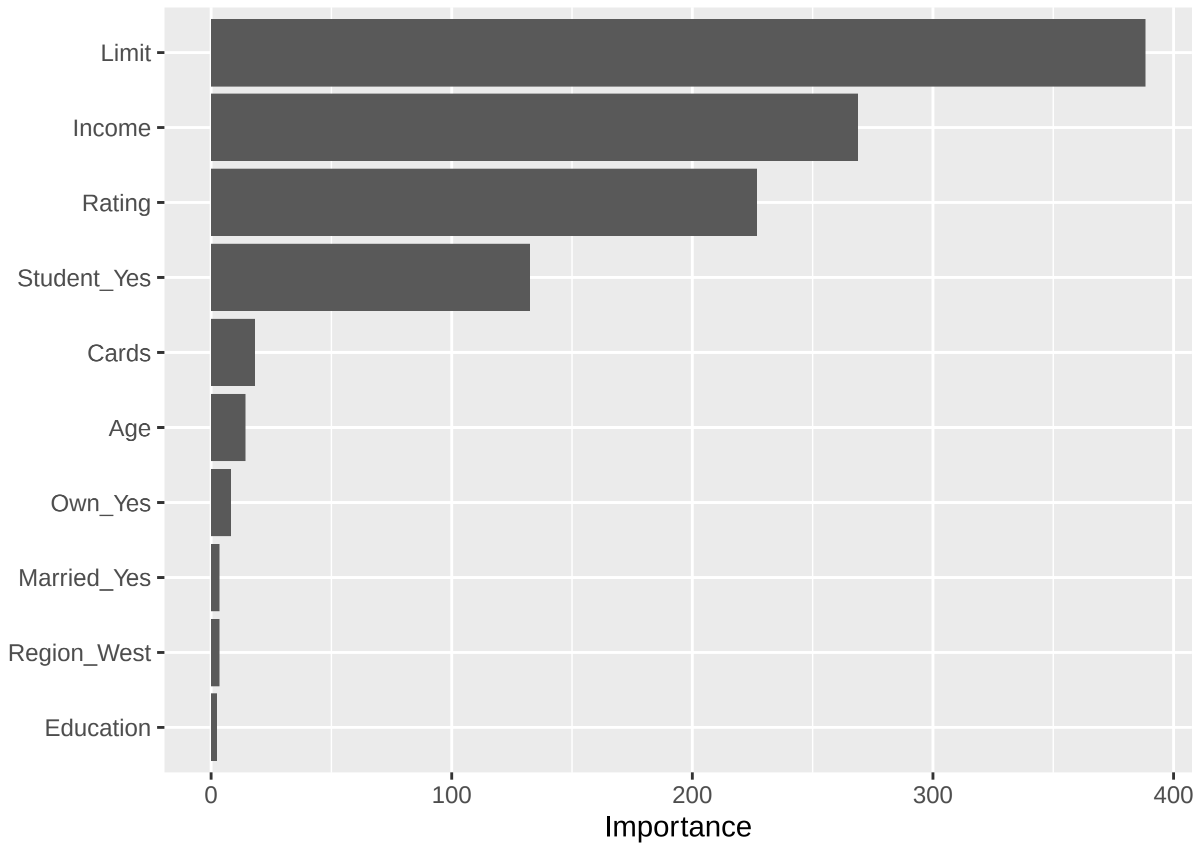
```
ridge.final.wf <- finalize_workflow(ridge.wf, best.penalty)
ridge.final.fit <- fit(ridge.final.wf, data = credit.train.tbl)

ridge.final.fit %>%
  extract_fit_engine() %>%
  plot(xvar = "lambda")
```



```
augment(ridge.final.fit, new_data = credit.test.tbl) %>%
  rsq(truth = Balance, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.946
```

```
library(vip)
extract_fit_parsnip(ridge.final.fit) %>%
  vip()
```



I modified the penalty grid from 0 to 2 into 0 to 5. It appears that smaller values of lambda work better. $R^2 = 0.946$.

The importance of features is, in order, Limit, Income, Rating, and Student-Yes. For the ridge regression model, this is Limit, Rating, Income, Student_Yes. The variables are similar, but Income and Rating are swapped.

Notes: Lagrange multivariables 'I am interested only in variables that are within a certain range of numbers' some formula where if s becoems very large, something becomes linear regression. If s approaches 0, then all coefficients approach 0. (y = mx + m1x + ... + mn)

suppose you are optimizing m and m1:

m^2 + m1^2 <= s this function looks like a circle on a plot of m, m1 where sum (mn^2) < s

|m| + |m1| <= s similar to circle but it's a diamond.

A word on the geometry of LASSO and ridge

Ridge and LASSO regression can be interpreted as minimizing $RSS(\beta_0, \ldots, \beta_p)$ subject to the constraint (in this case note that s is a fixed parameter)

- For ridge regression, $\sum\limits_{j=1}^{p} (\beta_j)^2 \leq s$
- For LASSO regression, $\sum\limits_{j=1}^{p} |\beta_j| \leq s$

One way to think about this is that we are allowed a budget of s that we are spending across the sum of the β_j^2 for ridge and the sum of $|\beta_j|$ for LASSO. Notice that if s tends to infinity we end up with just linear regression and that if s is very small we force all of the coefficients to go to zero (Implying that s and λ from our first formulation are inversely related)

Furthermore notice that if $p = 2$ the constraint for ridge consists of the unit disc with radius $\sqrt(s)$, while the constraint for LASSO is a diamond with diagonal $\frac{s}{2}$. Check figure 6.7 from ISLR from an illustration of how this observation implies that in LASSO we tend to choose values on the corners of the diamond, hence we are preferring solutions where the input variables are equal to zero.

---

look at video for this. I don't quite understand this paragraph above.

The College dataset

In the following set of exercises we will be exploring the `College` dataset from the `ISLR2` package. For more information on this dataset please consult `?College`. Let's start by loading our dataset and creating a training/testing dataset

```
data(College)
college.tbl <- as_tibble(College) %>%
  select(-c(Accept, Enroll))

set.seed(123456)
college.split <- initial_split(college.tbl, prop=0.8)
college.train.tbl <- training(college.split)
college.test.tbl <- testing(college.split)
```

We are interested in predicting the percent of acceptance (`pct.accept`) based on the other variables.

4. Create a linear model to predict `Apps` based on the other variables. What is the R^2 on the testing dataset? Plot the input variable importance

```
college.recipe <-
  recipe(formula=Apps ~ ., data=college.train.tbl) %>%
  step_dummy(all_nominal_predictors())
```

```
lm.model <- linear_reg() %>%
  set_engine("lm")

lm.wflow <- workflow() %>%
  add_recipe(college.recipe) %>%
  add_model(lm.model)

lm.fit <- fit(lm.wflow, college.train.tbl)
lm.fit %>%
  tidy()
```

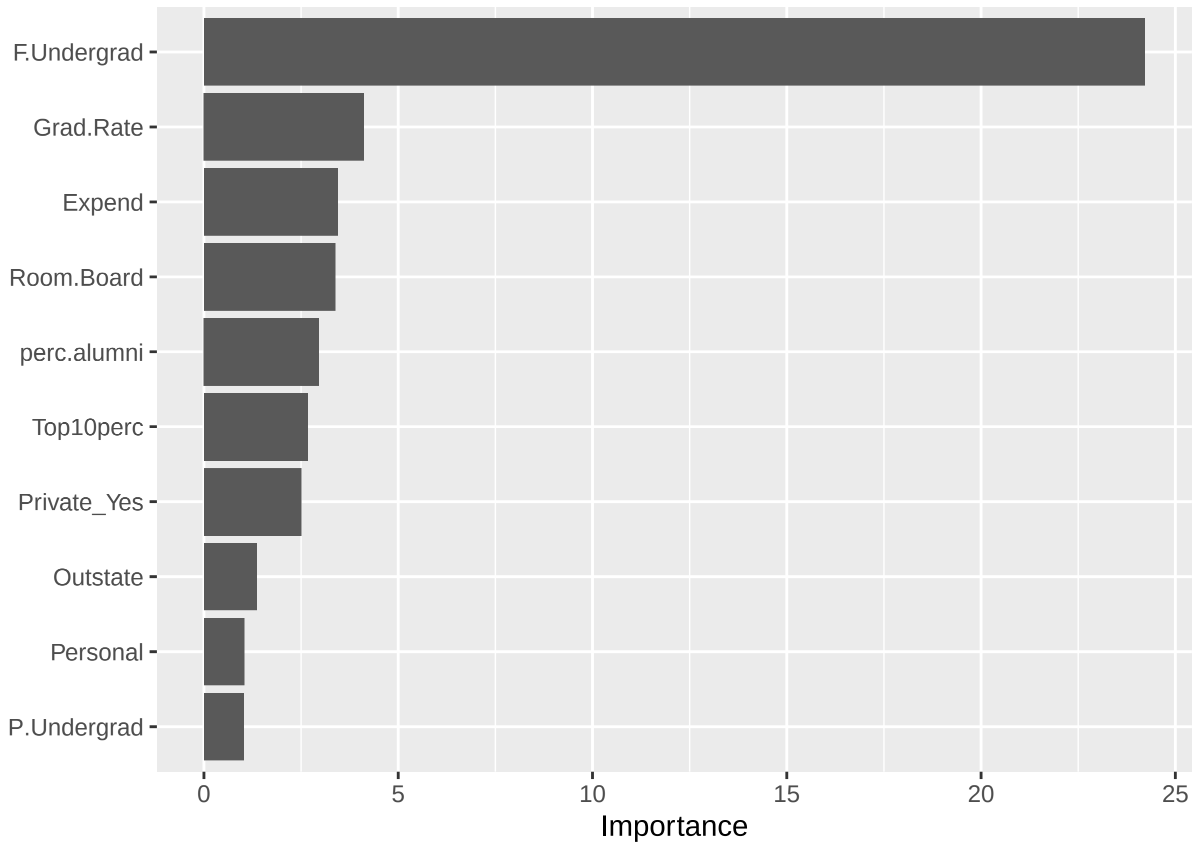
```
## # A tibble: 16 x 5
##    term         estimate std.error statistic  p.value
##    <chr>           <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept) -2738.      870.       -3.15   1.73e- 3
##  2 Top10perc      31.7      11.9       2.67   7.69e- 3
##  3 Top25perc      -7.60      9.72     -0.782  4.35e- 1
##  4 F.Undergrad     0.645     0.0266   24.2    4.40e-91
##  5 P.Undergrad    -0.0718    0.0699   -1.03   3.04e- 1
##  6 Outstate        0.0557    0.0409    1.36   1.74e- 1
##  7 Room.Board      0.350     0.104     3.38   7.67e- 4
##  8 Books           0.0456    0.533     0.0855 9.32e- 1
##  9 Personal       -0.143     0.138    -1.04   2.99e- 1
## 10 PhD            -4.64     10.0      -0.462  6.44e- 1
## 11 Terminal       -5.40     11.0      -0.491  6.24e- 1
## 12 S.F.Ratio      26.4      27.8       0.949  3.43e- 1
## 13 perc.alumni   -25.4       8.57     -2.96   3.18e- 3
## 14 Expend          0.0893    0.0259    3.45   6.05e- 4
## 15 Grad.Rate      26.1       6.34      4.12   4.34e- 5
## 16 Private_Yes  -761.      304.       -2.50   1.26e- 2
```

```
augment(lm.fit, new_data = college.test.tbl) %>%
  rsq(truth = Apps, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.839
```

$R^2 = 0.839$

```
extract_fit_parsnip(lm.fit) %>%
  vip()
```



5. Create a LASSO model to predict `Apps` based on the other variables. What is the R^2 on the testing dataset? Plot the input variable importance

Note: this is prediction (not classification.) note "regression" for set_mode()

```r
ridge.model <-
  linear_reg(mixture = 1, penalty=tune()) %>%
  set_mode("regression") %>%
  set_engine("glmnet")
```

```r
ridge.recipe <-
  recipe(formula = Apps ~ ., data = college.train.tbl) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_normalize(all_predictors())
ridge.wf <- workflow() %>%
  add_recipe(ridge.recipe) %>%
  add_model(ridge.model)
```

```r
college.fold <- vfold_cv(college.train.tbl, v = 10)

penalty.grid <-
  grid_regular(penalty(range = c(0, 1)), levels = 20)
```

```r
tune.res <- tune_grid(
  ridge.wf,
  resamples = college.fold,
  grid = penalty.grid
)
#autoplot(tune.res)
```

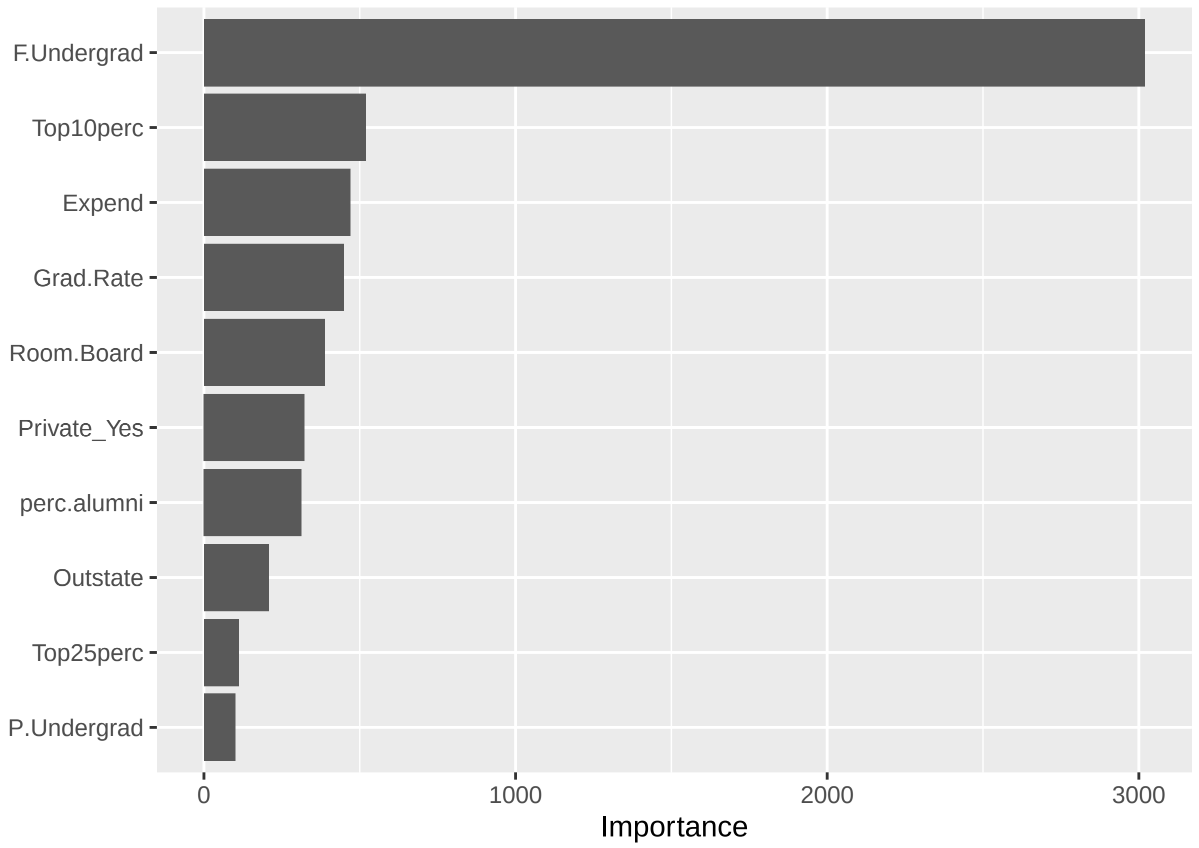
```r
(best.penalty <- select_best(tune.res, metric = "rsq"))
```

```
## # A tibble: 1 x 2
##   penalty .config
##     <dbl> <fct>
## 1      10 Preprocessor1_Model20
```

```r
ridge.final.wf <- finalize_workflow(ridge.wf, best.penalty)
ridge.final.fit <- fit(ridge.final.wf, data = college.train.tbl)

#ridge.final.fit %>%
#  extract_fit_engine() %>%
#  plot(xvar = "lambda")

augment(ridge.final.fit, new_data = college.test.tbl) %>%
  rsq(truth = Apps, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.840
```

```r
library(vip)
extract_fit_parsnip(ridge.final.fit) %>%
  vip()
```



$R^2 = 0.840$

6. Select the top-5 most important variables from LASSO and create a linear model based on only these features. How does the R^2 on the testing dataset compare with the full linear model?

Select top 5 most important variables from lasso:

```r
data(College)
college.tbl <- as_tibble(College) %>%
  select(-c(Accept, Enroll))
set.seed(123456)
college.split <- initial_split(college.tbl, prop=0.8)
college.train.tbl <- training(college.split)
college.test.tbl <- testing(college.split)

college.train.tbl <- college.train.tbl %>%
  select(Apps, F.Undergrad, Top10perc, Expend, Grad.Rate, Room.Board)
college.test.tbl <- college.test.tbl %>%
  select(Apps, F.Undergrad, Top10perc, Expend, Grad.Rate, Room.Board)
```

```r
college.recipe <-
  recipe(formula=Apps ~ ., data=college.train.tbl) %>%
  step_dummy(all_nominal_predictors())
```

```r
lm.model <- linear_reg() %>%
  set_engine("lm")

lm.wflow <- workflow() %>%
  add_recipe(college.recipe) %>%
  add_model(lm.model)

lm.fit <- fit(lm.wflow, college.train.tbl)
lm.fit %>%
  tidy()
```

```
## # A tibble: 6 x 5
##   term         estimate std.error statistic   p.value
##   <chr>           <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept) -3636.      407.        -8.93 5.01e- 18
## 2 F.Undergrad     0.672     0.0181     37.1  3.01e-159
## 3 Top10perc      19.3       6.70        2.88 4.13e-  3
## 4 Expend          0.0717    0.0218      3.28 1.08e-  3
## 5 Grad.Rate      21.2       5.79        3.66 2.73e-  4
## 6 Room.Board      0.358     0.0900      3.98 7.68e-  5
```

```
augment(lm.fit, new_data = college.test.tbl) %>%
  rsq(truth = Apps, estimate = .pred)
```

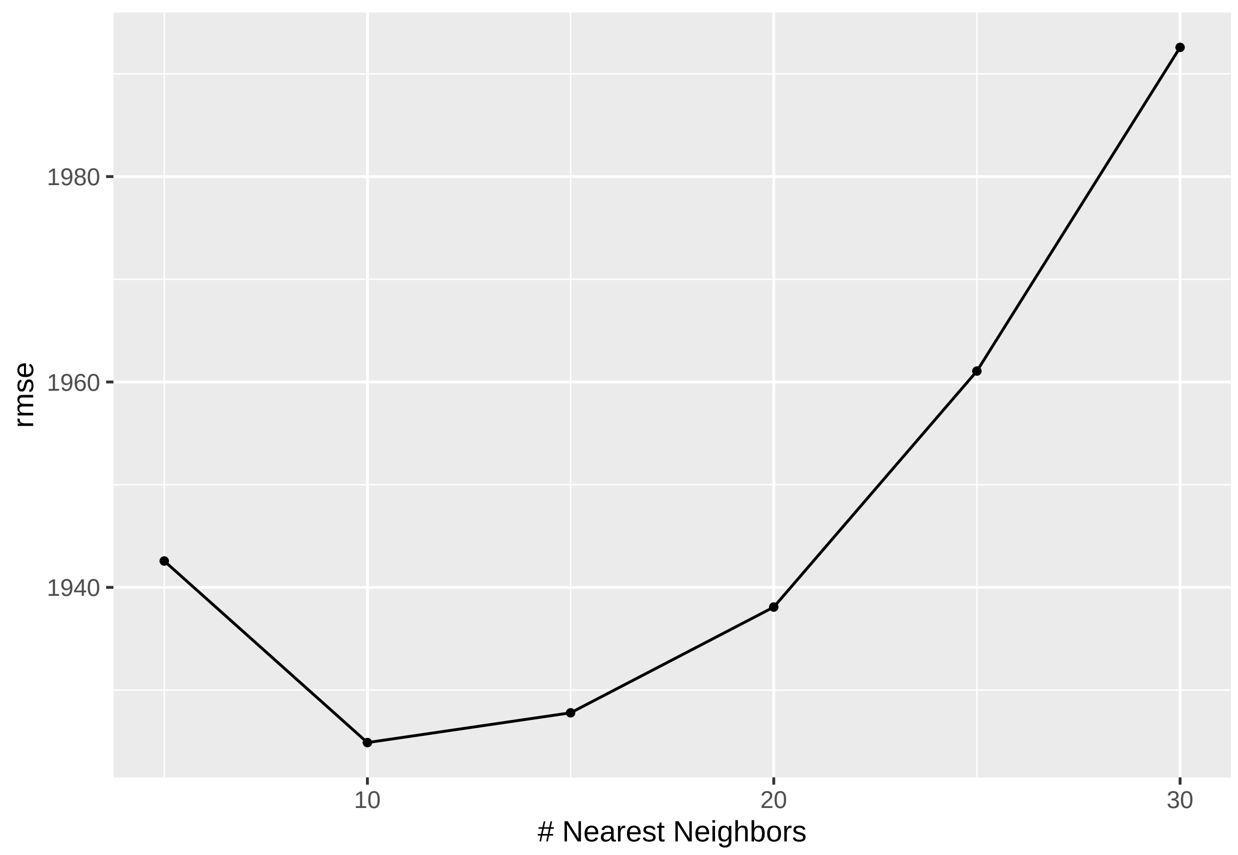
```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.835
```

r^2 = 0.835 for the testing dataset.

The r^2 for the full linear model is 0.839. The full linear model is only slightly better.

7. Construct a KNN model using the top-5 most important variables from LASSO. Based on the R^2 on the testing dataset, is this a better model than 6?

```
library(kknn)
set.seed(54321)
knn.model <- nearest_neighbor(neighbors = tune()) %>%
set_engine("kknn") %>%
set_mode("regression")
recipe <- recipe(Apps ~., data=college.train.tbl)
knn.wf <- workflow() %>%
add_recipe(recipe) %>%
add_model(knn.model)
# Neighbors optimization using CV
college.folds <- vfold_cv(college.train.tbl, v = 10)

#neighbors.grid.tbl <- tibble(neighbors = seq(5,30, by=5))
#neighbors.grid.tbl <- grid_regular(penalty(range = c(0, 2)), levels = 6)
neighbors.grid.tbl <- tibble(neighbors = seq(5,30, by=5))

tune.results <- tune_grid(object = knn.wf,
resamples = college.folds,
grid = neighbors.grid.tbl)
autoplot(tune.results, metric="rmse")
```



```r
show_best(tune.results, metric="rsq")
```

```
## # A tibble: 5 x 7
##   neighbors .metric .estimator  mean     n std_err .config
##       <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <fct>
## 1        25 rsq     standard   0.783    10  0.0253 Preprocessor1_Model5
## 2        20 rsq     standard   0.782    10  0.0248 Preprocessor1_Model4
## 3        30 rsq     standard   0.782    10  0.0262 Preprocessor1_Model6
## 4        15 rsq     standard   0.779    10  0.0254 Preprocessor1_Model3
## 5        10 rsq     standard   0.775    10  0.0267 Preprocessor1_Model2
```

```r
best.neighbor <- select_best(tune.results,
neighbors, metric = "rsq")
knn.final.wf <- finalize_workflow(knn.wf, best.neighbor)
knn.final.fit <- fit(knn.final.wf, college.train.tbl)

# Evaluation of model on the testing dataset
augment(knn.final.fit, college.test.tbl) %>%
rsq(truth = Apps, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.809
```

R^2 = 0.809

It is a worser model than #6. As the professor said, linear models seem to work surprisingly well, at least in this case.

8. What will be the topic for your Challenge 2 spotlight? Write a paragraph explaining the topic, dataset or question that you will be addressing.

I am guessing the project will be something with prediction since the last project was classification.

I need to take a look at what sorts of datasets I can find, but what I might do is: find a data science paper to go off of and find some additional questions to answer based on the data they use; predict post-college employment rates based on stuff like the number of accreditations the school has or etc. Or, combine a housing price dataset and population dataset for a single city; brainstorm some of the variables I want for this; think about how to join the dataets; then do a lasso regression to find the most important factors for increasing house prices - emphasis on increase. (proportion of houses owned by renters over time - maybe mutate variables that measure increase over time). (note to self: look at how the music half-life project was set up)