

Final

Ethan, Sylvester, Navid

```
library(tidyverse)
library(tidymodels)
library(scales)
library(corrplot)
library(olsrr)
library(caret)
library(gridExtra)
library(glmnet)
library(dials)
```

```
data <- read_csv(choose.files())
```

```
glimpse(data)
```

Rows: 205

Columns: 26

```
$ car_ID          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16~
$ symboling      <dbl> 3, 3, 1, 2, 2, 2, 1, 1, 1, 0, 2, 0, 0, 0, 1, 0, 0, 0,~
$ CarName        <chr> "alfa-romero giulia", "alfa-romero stelvio", "alfa-ro~
$ fueltype       <chr> "gas", "gas", "gas", "gas", "gas", "gas", "gas", "gas~
$ aspiration     <chr> "std", "std", "std", "std", "std", "std", "std", "std~
$ doornumber     <chr> "two", "two", "two", "four", "four", "two", "four", "~
$ carbody        <chr> "convertible", "convertible", "hatchback", "sedan", "~
$ drivewheel     <chr> "rwd", "rwd", "rwd", "fwd", "4wd", "fwd", "fwd", "fwd~
$ enginelocation <chr> "front", "front", "front", "front", "front", "front",~
$ wheelbase      <dbl> 88.6, 88.6, 94.5, 99.8, 99.4, 99.8, 105.8, 105.8, 105~
$ carlength      <dbl> 168.8, 168.8, 171.2, 176.6, 176.6, 177.3, 192.7, 192.~
$ carwidth       <dbl> 64.1, 64.1, 65.5, 66.2, 66.4, 66.3, 71.4, 71.4, 71.4,~
$ carheight      <dbl> 48.8, 48.8, 52.4, 54.3, 54.3, 53.1, 55.7, 55.7, 55.9,~
$ curbweight     <dbl> 2548, 2548, 2823, 2337, 2824, 2507, 2844, 2954, 3086,~
```

```

$ enginetype      <chr> "dohc", "dohc", "ohcv", "ohc", "ohc", "ohc", "ohc", "~
$ cylindernumber  <chr> "four", "four", "six", "four", "five", "five", "five"~
$ enginesize       <dbl> 130, 130, 152, 109, 136, 136, 136, 136, 131, 131, 108~
$ fuelsystem      <chr> "mpfi", "mpfi", "mpfi", "mpfi", "mpfi", "mpfi", "mpfi~
$ boreratio       <dbl> 3.47, 3.47, 2.68, 3.19, 3.19, 3.19, 3.19, 3.19, 3.13,~
$ stroke          <dbl> 2.68, 2.68, 3.47, 3.40, 3.40, 3.40, 3.40, 3.40, 3.40,~
$ compressionratio <dbl> 9.00, 9.00, 9.00, 10.00, 8.00, 8.50, 8.50, 8.50, 8.30~
$ horsepower      <dbl> 111, 111, 154, 102, 115, 110, 110, 110, 140, 160, 101~
$ peakrpm         <dbl> 5000, 5000, 5000, 5500, 5500, 5500, 5500, 5500, 5500,~
$ citympg         <dbl> 21, 21, 19, 24, 18, 19, 19, 19, 17, 16, 23, 23, 21, 2~
$ highwaympg      <dbl> 27, 27, 26, 30, 22, 25, 25, 25, 20, 22, 29, 29, 28, 2~
$ price           <dbl> 13495.00, 16500.00, 16500.00, 13950.00, 17450.00, 152~

```

We'll work with these 4 categorical variables. Also, there were no Missing Values.

```

data$fueltype      <- as.factor(data$fueltype)
data$doornumber    <- as.factor(data$doornumber)
data$carbody       <- as.factor(data$carbody)
data$drivewheel    <- as.factor(data$drivewheel)

data <- data |>
  mutate(model = word(CarName, 1)) |> # extract first token
  mutate(model = case_when(
    model %in% c("maxda") ~ "mazda",
    model %in% c("mazda") ~ "mazda",
    model %in% c("nissan") ~ "nissan",
    model %in% c("porcshce", "porsche") ~ "porsche",
    model %in% c("toyouta", "toyota") ~ "toyota",
    model %in% c("vokswagen", "vw", "volkswagen") ~ "volkswagen",
    TRUE ~ model
  ))

```

The inflation price which we had talked about. I called it `new_price`

```

data <- data |> select(-car_ID, -CarName, -symboling)
data <- data |>
  mutate(new_price = 2.8 * price)

brand_avg_price <- data |>
  group_by(model) |>
  summarise(mean_new_price = mean(new_price, na.rm = TRUE)) |>

```

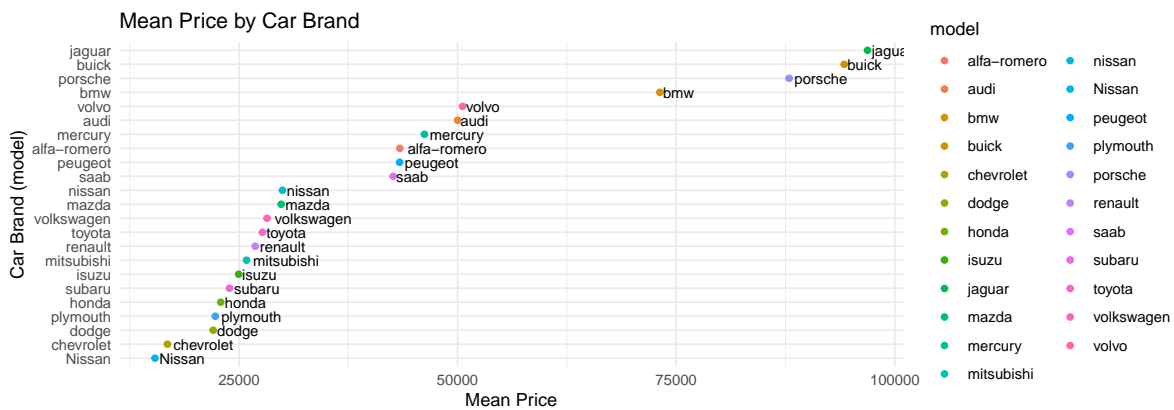
```

arrange(mean_new_price)

p_brand_mean <- ggplot(brand_avg_price,
                      aes(x = reorder(model, mean_new_price), y = mean_new_price)) +
  geom_point(aes(color = model)) +
  geom_text(aes(label = model), hjust = -0.1, size = 3) +
  coord_flip() +
  labs(title = "Mean Price by Car Brand",
       x = "Car Brand (model)",
       y = "Mean Price") +
  theme_minimal()

print(p_brand_mean)

```



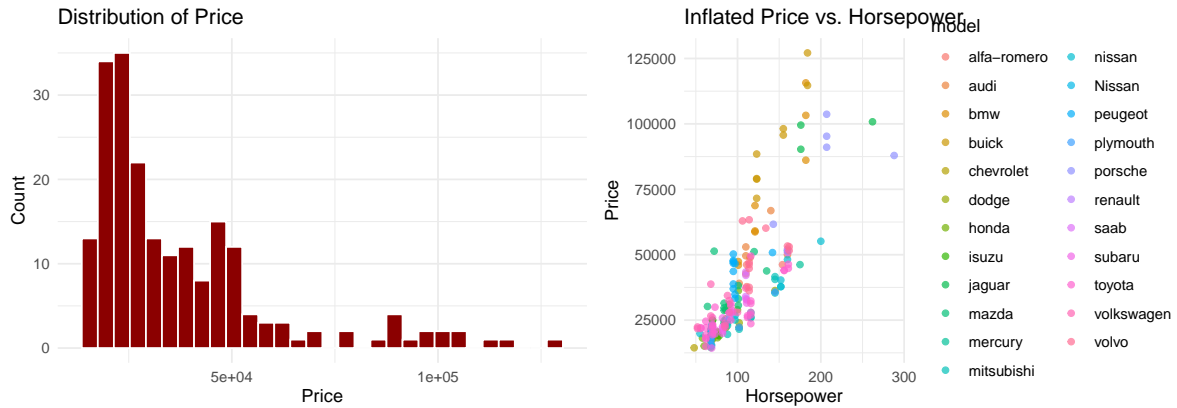
```

p_cont2 <- ggplot(data, aes(x = new_price)) +
  geom_histogram(bins = 30, color = "white", fill = "red4") +
  labs(title = "Distribution of Price",
       x = "Price", y = "Count") +
  theme_minimal()

# 3c. Scatter: new_price vs. horsepower
p_cont3 <- ggplot(data, aes(x = horsepower, y = new_price, color = model)) +
  geom_point(alpha = 0.7) +
  labs(title = "Inflated Price vs. Horsepower",
       x = "Horsepower", y = "Price") +
  theme_minimal()

grid.arrange(p_cont2, p_cont3, nrow = 1)

```

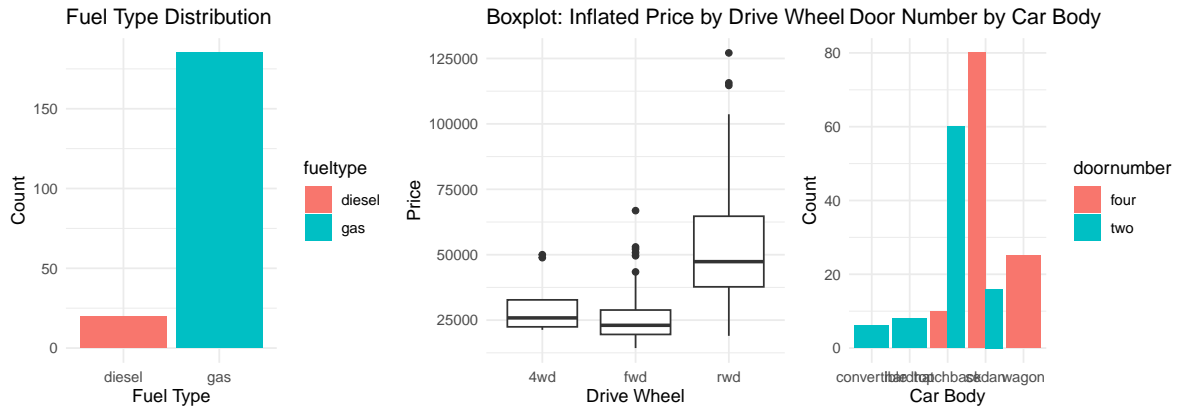


```
# 4a. Bar chart of fueltype
p_cat1 <- ggplot(data, aes(x = fueltype)) +
  geom_bar(aes(fill = fueltype)) +
  labs(title = "Fuel Type Distribution",
        x = "Fuel Type", y = "Count") +
  theme_minimal()

# 4b. Boxplot of new_price by drivewheel
p_cat2 <- ggplot(data, aes(x = drivewheel, y = new_price)) +
  geom_boxplot() +
  labs(title = "Boxplot: Inflated Price by Drive Wheel",
        x = "Drive Wheel", y = "Price") +
  theme_minimal()

# 4c. Bar chart: doornumber by carbody
p_cat3 <- ggplot(data, aes(x = carbody, fill = doornumber)) +
  geom_bar(position = "dodge") +
  labs(title = "Door Number by Car Body",
        x = "Car Body", y = "Count") +
  theme_minimal()

grid.arrange(p_cat1, p_cat2, p_cat3, nrow = 1)
```



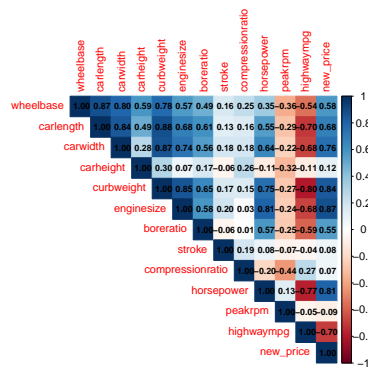
```
cont_vars <- c("wheelbase", "carlength", "carwidth", "carheight",
              "curbweight", "enginesize", "boreratio", "stroke",
              "compressionratio", "horsepower", "peakrpm", "highwaympg",
              "new_price")
```

```
cat_vars <- c("fueltype", "aspiration", "doornumber",
             "carbody", "drivewheel", "enginelocation", "enginetype",
             "fuelsystem", "model")
```

```
corr_mat <- cor(data[cont_vars], use = "complete.obs")
```

```
p_cont1 <- corrplot(corr_mat,
                   method = "color",
                   type = "upper",
                   addCoef.col = "black",
                   number.cex = 0.7,
                   tl.cex = 0.8,
                   title = "Correlation Matrix (Continuous)",
                   mar = c(0,0,2,0))
```

Correlation Matrix (Continuous)



```
cat("Average Price by Fuel Type:\n")
```

Average Price by Fuel Type:

```
print(data %>% group_by(fueltype) %>% summarise(mean_price = mean(new_price)) %>% arrange(
```

```
# A tibble: 2 x 2
  fueltype mean_price
  <fct>      <dbl>
1 gas        36399.
2 diesel     44347.
```

```
cat("Average Price by Aspiration:\n")
```

Average Price by Aspiration:

```
print(data %>% group_by(aspiration) %>% summarise(mean_price = mean(new_price)) %>% arrang
```

```
# A tibble: 2 x 2
  aspiration mean_price
  <chr>      <dbl>
1 std        35312.
2 turbo      45635.
```

```
cat("Average Price by Carbody:\n")
```

Average Price by Carbody:

```
print(data %>% group_by(carbody) %>% summarise(mean_price = mean(new_price)) %>% arrange(m
```

```
# A tibble: 5 x 2
  carbody    mean_price
  <fct>      <dbl>
1 hatchback 29055.
2 wagon    34641.
3 sedan    40164.
4 convertible 61293.
5 hardtop  62184.
```

```
cat("Average Price by Drivewheel:\n")
```

Average Price by Drivewheel:

```
print(data %>% group_by(drivewheel) %>% summarise(mean_price = mean(new_price)) %>% arrang
```

```
# A tibble: 3 x 2
  drivewheel mean_price
  <fct>      <dbl>
1 fwd       25870.
2 4wd       31045.
3 rwd       55750.
```

```
cat("Average Price by Enginelocation:\n")
```

Average Price by Enginelocation:

```
print(data %>% group_by(enginelocation) %>% summarise(mean_price = mean(new_price)) %>% ar
```

```
# A tibble: 2 x 2
  enginelocation mean_price
  <chr>           <dbl>
1 front           36291.
2 rear            96678.
```

```
cat("Average Price by Enginetype:\n")
```

Average Price by Enginetype:

```
print(data %>% group_by(enginetype) %>% summarise(mean_price = mean(new_price)) %>% arrang
```

```
# A tibble: 7 x 2
  enginetype mean_price
  <chr>       <dbl>
1 ohc        32407.
2 rotor      36456.
3 ohcf       38468.
4 l          40957.
5 dohc       50726.
6 ohcv       70275.
7 dohcv      87921.
```

```
cat("Average Price by Fuelsystem:\n")
```

Average Price by Fuelsystem:

```
print(data %>% group_by(fuelsystem) %>% summarise(mean_price = mean(new_price)) %>% arrang
```

```
# A tibble: 8 x 2
  fuelsystem mean_price
  <chr>       <dbl>
1 2bbl       20939.
2 1bbl       21156.
3 spdi       30773.
4 spfi       30934.
5 4bbl       34006.
```

```
6 mfi          36299.
7 idi          44347.
8 mpfi         49713.
```

```
cat("Average Price by Model:\n")
```

Average Price by Model:

```
print(data %>% group_by(model) %>% summarise(mean_price = mean(new_price)) %>% arrange(desc(mean_price)))
```

```
# A tibble: 23 x 2
  model      mean_price
  <chr>      <dbl>
1 jaguar      96880
2 buick       94212.
3 porsche     87921.
4 bmw         73132.
5 volvo       50577.
6 audi        50006.
7 mercury     46208.
8 alfa-romero 43395.
9 peugeot     43369.
10 saab       42625.
# i 13 more rows
```

Modeling

```
tidymodels_prefer() # optional, to prefer tidymodels functions
set.seed(1372)
```

Just Continuous variables.

```
# Just Continuous
data1 <- data[cont_vars]

data_split <- initial_split(data1, prop = 0.8)
train_data <- training(data_split)
test_data <- testing(data_split)
```

```

model_full <- lm(new_price ~ ., data = train_data)
# Use olsrr's function which can handle aliased coefficients
vif_values <- ols_vif_tol(model_full)
cat("VIF values for full model:\n")

```

VIF values for full model:

```
print(vif_values)
```

	Variables	Tolerance	VIF
1	wheelbase	0.14380190	6.954011
2	carlength	0.11683808	8.558853
3	carwidth	0.16663830	6.001022
4	carheight	0.43668239	2.289994
5	curbweight	0.04331503	23.086677
6	enginesize	0.12803537	7.810342
7	boreratio	0.48008907	2.082947
8	stroke	0.86149090	1.160778
9	compressionratio	0.50941347	1.963042
10	horsepower	0.13143902	7.608091
11	peakrpm	0.52624950	1.900239
12	highwaympg	0.17579580	5.688418

VIFs.

```

# VIF more than 10
vif_values |>
  filter(VIF > 10)

```

	Variables	Tolerance	VIF
1	curbweight	0.04331503	23.08668

Simple multiple regression model.

```

## just curbweight
train_data_without_vif_10 <- train_data |>
  select(!curbweight)

model_full_without_vif_10 <- lm(new_price ~ ., data = train_data_without_vif_10)

```

```
summary(model_full_without_vif_10)
```

Call:

```
lm(formula = new_price ~ ., data = train_data_without_vif_10)
```

Residuals:

Min	1Q	Median	3Q	Max
-29064.6	-4468.6	-527.2	3985.8	28284.7

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-1.196e+05	4.389e+04	-2.725	0.00718	**
wheelbase	2.591e+02	2.782e+02	0.931	0.35323	
carlength	-1.935e+02	1.518e+02	-1.274	0.20446	
carwidth	1.295e+03	7.135e+02	1.815	0.07148	.
carheight	5.974e+02	4.013e+02	1.489	0.13865	
engineize	3.219e+02	4.105e+01	7.841	7.28e-13	***
boreratio	-2.748e+03	3.426e+03	-0.802	0.42378	
stroke	-7.518e+03	2.046e+03	-3.675	0.00033	***
compressionratio	9.163e+02	2.193e+02	4.178	4.94e-05	***
horsepower	1.248e+02	4.351e+01	2.869	0.00470	**
peakrpm	5.802e+00	1.882e+00	3.083	0.00243	**
highwaympg	-3.863e+02	1.949e+02	-1.982	0.04931	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8177 on 152 degrees of freedom

Multiple R-squared: 0.8566, Adjusted R-squared: 0.8463

F-statistic: 82.57 on 11 and 152 DF, p-value: < 2.2e-16

```
pred_baseline <- predict(model_full_without_vif_10, newdata = test_data)
rmse_baseline <- sqrt(mean((pred_baseline - test_data$new_price)^2))
r2_baseline <- cor(pred_baseline, test_data$new_price)^2
cat("Baseline Linear Regression RMSE:", round(rmse_baseline, 2), "\n")
```

Baseline Linear Regression RMSE: 11623.55

```
cat("Baseline Linear Regression R2:", round(r2_baseline, 2), "\n")
```

Baseline Linear Regression R2: 0.82

```
# ###
# # VIF more than 5
#
# vif_values |>
#   filter(VIF > 5)
#
# ## just curbweight
#
# train_data_without_vif_5 <- train_data |>
#   select(-c(wheelbase, carlength, carwidth, curbweight, enginesize, horsepower, highway))
#
# model_full <- lm(new_price ~ ., data = train_data_without_vif_5)
# summary(model_full)
#
#
# pred_baseline <- predict(model_full, newdata = test_data)
# rmse_baseline <- sqrt(mean((pred_baseline - test_data$new_price)^2))
# r2_baseline <- cor(pred_baseline, test_data$new_price)^2
# cat("Baseline Linear Regression RMSE:", round(rmse_baseline, 2), "\n")
# cat("Baseline Linear Regression R2:", round(r2_baseline, 2), "\n")
#
# ## It seems that VIF more than 10 works much better than VIF more than 5
#
```

Ridge

```
## RIDGE
car_fold <- vfold_cv(train_data, v = 10)

ridge_recipe_continuous <-
  recipe(formula = new_price ~ ., data = train_data) |>
  step_novel(all_nominal_predictors()) |>
  step_dummy(all_nominal_predictors()) |>
  step_zv(all_predictors()) |>
  step_normalize(all_predictors())

ridge_spec_continuous <-
  linear_reg(penalty = tune(), mixture = 0) |>
  set_mode("regression") |>
  set_engine("glmnet")
```

```

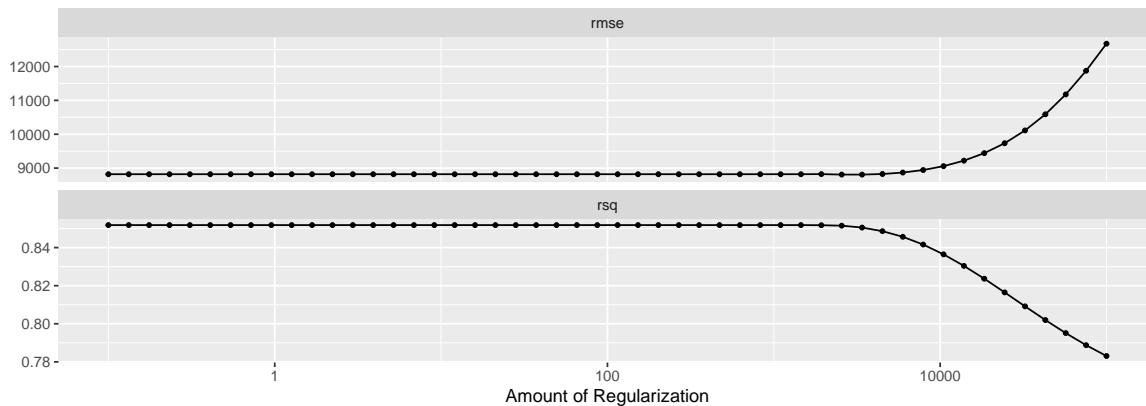
ridge_workflow_continuous <- workflow() |>
  add_recipe(ridge_recipe_continuous) |>
  add_model(ridge_spec_continuous)

ridge_penalty_grid_continuous <- grid_regular(dials::penalty(range = c(-1, 5)), levels = 5)
#penalty_grid

ridge_tune_res_continuous <- tune_grid(
  ridge_workflow_continuous,
  resamples = car_fold,
  grid = ridge_penalty_grid_continuous
)

autoplot(ridge_tune_res_continuous)

```



Ridge Metrics

```

collect_metrics(ridge_tune_res_continuous) |>
  tail()

```

A tibble: 6 x 7

	penalty	.metric	.estimator	mean	n	std_err	.config
	<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>	<chr>
1	56899.	rmse	standard	11176.	10	1507.	Preprocessor1_Model48
2	56899.	rsq	standard	0.795	10	0.0340	Preprocessor1_Model48
3	75431.	rmse	standard	11875.	10	1557.	Preprocessor1_Model49
4	75431.	rsq	standard	0.789	10	0.0351	Preprocessor1_Model49
5	100000	rmse	standard	12675.	10	1599.	Preprocessor1_Model50

```
6 100000 rsq standard 0.783 10 0.0360 Preprocessor1_Model150
```

```
ridge_best_penalty_continuous <- select_best(ridge_tune_res_continuous, metric = "rmse")  
ridge_best_penalty_continuous
```

```
# A tibble: 1 x 2  
  penalty .config  
  <dbl> <chr>  
1 3393. Preprocessor1_Model138
```

```
ridge_final_continuous <- finalize_workflow(ridge_workflow_continuous, ridge_best_penalty_<br>ridge_final_fit_continuous <- fit(ridge_final_continuous, data = train_data)
```

```
augment(ridge_final_fit_continuous, new_data = test_data) |>  
  rsq(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3  
  .metric .estimator .estimate  
  <chr> <chr> <dbl>  
1 rsq standard 0.815
```

```
augment(ridge_final_fit_continuous, new_data = test_data) |>  
  rmse(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3  
  .metric .estimator .estimate  
  <chr> <chr> <dbl>  
1 rmse standard 12575.
```

Lasso

```
### LASSO  
lasso_recipe_continuous <-  
  recipe(formula = new_price ~ ., data = train_data) |>  
  step_novel(all_nominal_predictors()) |>  
  step_dummy(all_nominal_predictors()) |>  
  step_zv(all_predictors()) |>
```

```

step_normalize(all_predictors())

lasso_spec_continuous <-
  linear_reg(penalty = tune(), mixture = 1) |>
  set_mode("regression") |>
  set_engine("glmnet")

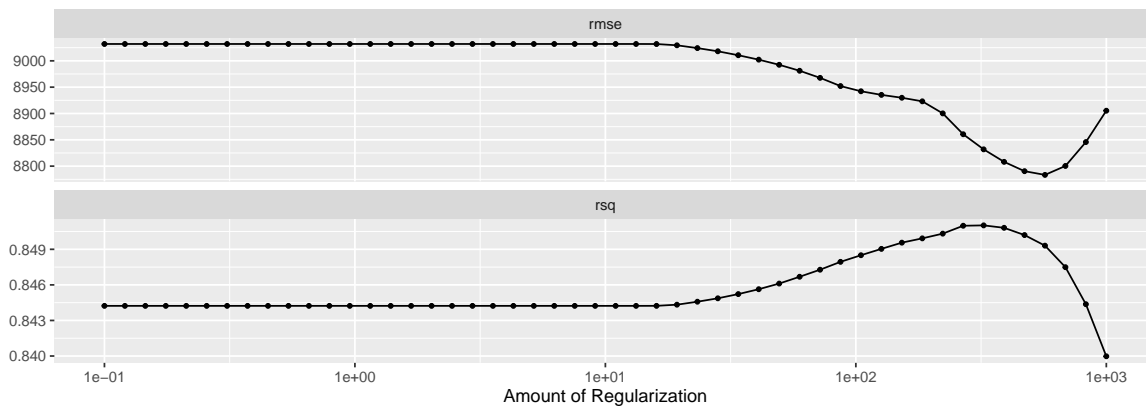
lasso_workflow_continuous <- workflow() |>
  add_recipe(lasso_recipe_continuous) |>
  add_model(lasso_spec_continuous)

lasso_penalty_grid_continuous <- grid_regular(penalty(range = c(-1, 3)), levels = 50)

lasso_tune_res_continuous <- tune_grid(
  lasso_workflow_continuous,
  resamples = car_fold,
  grid = lasso_penalty_grid_continuous
)

autoplot(lasso_tune_res_continuous)

```



Lasso Metrics

```

lasso_best_penalty_continuous <- select_best(lasso_tune_res_continuous, metric = "rmse")

lasso_final_continuous <- finalize_workflow(lasso_workflow_continuous, lasso_best_penalty_

lasso_final_fit_continuous <- fit(lasso_final_continuous, data = train_data)

```

```
augment(lasso_final_fit_continuous, new_data = test_data) |>
  rsq(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 rsq     standard         0.813
```

```
augment(lasso_final_fit_continuous, new_data = test_data) |>
  rmse(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 rmse   standard     12359.
```

```
tidy(ridge_final_fit_continuous)
```

```
# A tibble: 13 x 3
  term          estimate penalty
  <chr>         <dbl>   <dbl>
1 (Intercept)  36415.  3393.
2 wheelbase    349.    3393.
3 carlength   -282.    3393.
4 carwidth    2146.    3393.
5 carheight    323.    3393.
6 curbweight  4238.    3393.
7 enginesize   7289.    3393.
8 boreratio   -190.    3393.
9 stroke     -1780.    3393.
10 compressionratio 2401.    3393.
11 horsepower   5106.    3393.
12 peakrpm     1809.    3393.
13 highwaympg -1435.    3393.
```

```
tidy(lasso_final_fit_continuous)
```

```
# A tibble: 13 x 3
  term          estimate penalty
  <chr>          <dbl>   <dbl>
1 (Intercept)  36415.   569.
2 wheelbase      0       569.
3 carlength      0       569.
4 carwidth     1000.   569.
5 carheight     26.1    569.
6 curbweight   5715.   569.
7 enginesize    8900.   569.
8 boreratio      0       569.
9 stroke     -1667.   569.
10 compressionratio 1737.   569.
11 horsepower   4429.   569.
12 peakrpm     1811.   569.
13 highwaympg    0       569.
```

Adding 4 categorical variables. [fueltype, doornumber, carbody, drivewheel]

```
data2 <- data1 |>
  bind_cols(data[cat_vars][c(1, 3, 4, 5)])

data_split_2 <- initial_split(data2, prop = 0.8)
train_data_2 <- training(data_split_2)
test_data_2 <- testing(data_split_2)

model_full_2 <- lm(new_price ~ ., data = train_data_2)
# Use olsrr's function which can handle aliased coefficients
vif_values_2 <- ols_vif_tol(model_full_2)
cat("VIF values for full model:\n")
```

VIF values for full model:

```
print(vif_values_2)
```

	Variables	Tolerance	VIF
1	wheelbase	0.10584039	9.448189
2	carlength	0.08109906	12.330599
3	carwidth	0.16113389	6.206019

4	carheight	0.35739561	2.798020
5	curbweight	0.02954480	33.846900
6	enginesize	0.10281097	9.726589
7	boreratio	0.40700031	2.457001
8	stroke	0.62090518	1.610552
9	compressionratio	0.01593290	62.763225
10	horsepower	0.11778886	8.489767
11	peakrpm	0.42839379	2.334301
12	highwaympg	0.15164860	6.594192
13	fueltypegas	0.01492706	66.992431
14	doornumbertwo	0.34539557	2.895231
15	carbodyhardtop	0.32494701	3.077425
16	carbodyhatchback	0.09157985	10.919433
17	carbodysedan	0.06997934	14.289931
18	carbodywagon	0.14411081	6.939105
19	drivewheel fwd	0.08205005	12.187683
20	drivewheel rwd	0.08197961	12.198155

Ridge for Continuous and Categoricals

```

car_fold_2 <- vfold_cv(train_data_2, v = 10)

ridge_recipe_2 <-
  recipe(formula = new_price ~ ., data = train_data_2) |>
  step_novel(all_nominal_predictors()) |>
  step_dummy(all_nominal_predictors()) |>
  step_zv(all_predictors()) |>
  step_normalize(all_predictors())

ridge_spec_2 <-
  linear_reg(penalty = tune(), mixture = 0) |>
  set_mode("regression") |>
  set_engine("glmnet")

ridge_workflow_2 <- workflow() |>
  add_recipe(ridge_recipe_2) |>
  add_model(ridge_spec_2)

penalty_grid_2 <- grid_regular(dials::penalty(range = c(-1, 5)), levels = 50)

ridge_tune_res_2 <- tune_grid(
  ridge_workflow_2,

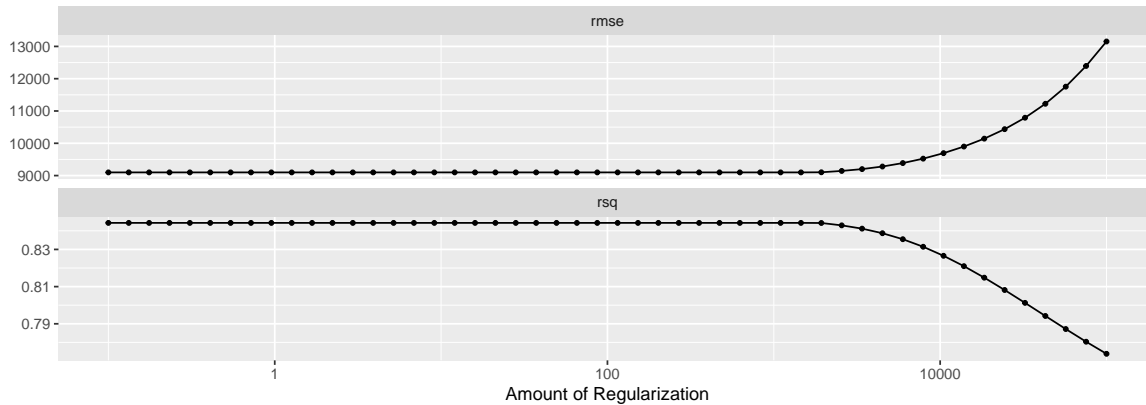
```

```

  resamples = car_fold_2,
  grid = penalty_grid_2
)

autoplot(ridge_tune_res_2)

```



Best Penalty

```

best_penalty_2 <- select_best(ridge_tune_res_2, metric = "rmse")
best_penalty_2

```

```

# A tibble: 1 x 2
  penalty .config
  <dbl> <chr>
1 0.1 Preprocessor1_Model01

```

Ridge Metrics

```

ridge_final_2 <- finalize_workflow(ridge_workflow_2, best_penalty_2)
ridge_final_fit_2 <- fit(ridge_final_2, data = train_data_2)

augment(ridge_final_fit_2, new_data = test_data_2) |>
  rsq(truth = new_price, estimate = .pred)

```

```

# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr> <chr> <dbl>
1 rsq standard 0.834

```

```
augment(ridge_final_fit_2, new_data = test_data_2) |>
  rmse(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 rmse   standard       8947.
```

Lasso for Continuous and Categoricals

```
### LASSO
lasso_recipe_2 <-
  recipe(formula = new_price ~ ., data = train_data_2) |>
  step_novel(all_nominal_predictors()) |>
  step_dummy(all_nominal_predictors()) |>
  step_zv(all_predictors()) |>
  step_normalize(all_predictors())

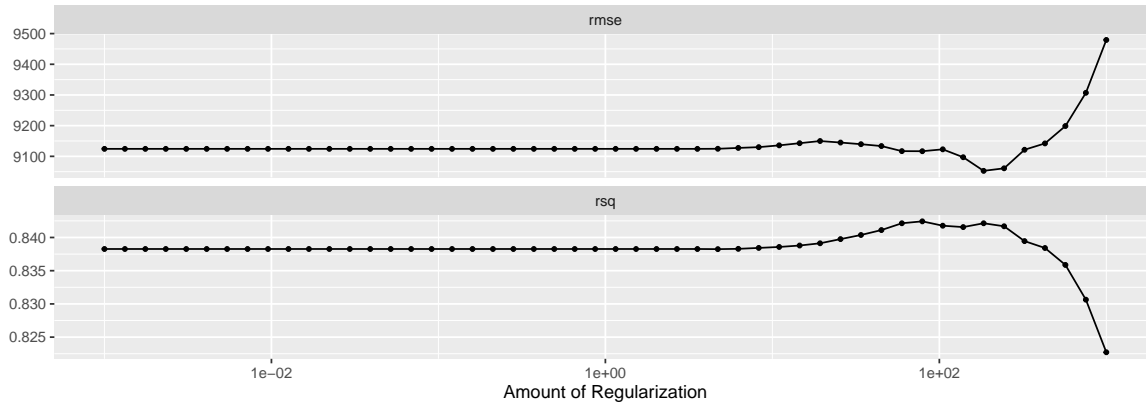
lasso_spec_2 <-
  linear_reg(penalty = tune(), mixture = 1) |>
  set_mode("regression") |>
  set_engine("glmnet")

lasso_workflow_2 <- workflow() |>
  add_recipe(lasso_recipe_2) |>
  add_model(lasso_spec_2)

penalty_grid_2 <- grid_regular(penalty(range = c(-3, 3)), levels = 50)

lasso_tune_res_2 <- tune_grid(
  lasso_workflow_2,
  resamples = car_fold_2,
  grid = penalty_grid_2
)

autoplot(lasso_tune_res_2)
```



Lasso Penalty

```
best_penalty_2 <- select_best(lasso_tune_res_2, metric = "rmse")
best_penalty_2
```

```
# A tibble: 1 x 2
  penalty .config
  <dbl> <chr>
1 184. Preprocessor1_Model144
```

Lasso Metrics

```
lasso_final_2 <- finalize_workflow(lasso_workflow_2, best_penalty_2)

lasso_final_fit_2 <- fit(lasso_final_2, data = train_data_2)

augment(lasso_final_fit_2, new_data = test_data_2) |>
  rsq(truth = new_price, estimate = .pred)
```

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

```
augment(lasso_final_fit_2, new_data = test_data_2) |>
  rmse(truth = new_price, estimate = .pred)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 rmse   standard       8960.
```

```
tidy(lasso_final_fit_2)
```

```
# A tibble: 21 x 3
  term          estimate penalty
  <chr>         <dbl>   <dbl>
1 (Intercept)  37935.   184.
2 wheelbase     0         184.
3 carlength    -0.129   184.
4 carwidth     4039.   184.
5 carheight    1644.   184.
6 curbweight    0         184.
7 enginesize   12419.   184.
8 boreratio    -1010.   184.
9 stroke      -1954.   184.
10 compressionratio  0         184.
# i 11 more rows
```

```
tidy(ridge_final_fit_2)
```

```
# A tibble: 21 x 3
  term          estimate penalty
  <chr>         <dbl>   <dbl>
1 (Intercept)  37935.    0.1
2 wheelbase     638.    0.1
3 carlength   -1155.    0.1
4 carwidth     3756.    0.1
5 carheight    1325.    0.1
6 curbweight    2566.    0.1
7 enginesize    9206.    0.1
8 boreratio   -1096.    0.1
9 stroke     -1776.    0.1
10 compressionratio  474.    0.1
# i 11 more rows
```

Prediction of civic 2025

```
civic2025 <- data.frame(
  fueltype = "gas",
  carbody = "sedan",
  drivewheel = "fwd",
  doornumber = "four",
  wheelbase = 107.7,
  carlength = 184.8,
  carwidth = 70.9,
  carheight = 55.7,
  curbweight = 2875,
  enginesize = 122,
  boreratio = 3.2,
  stroke = 3.8,
  compressionratio = 13,
  horsepower = 150,
  citympg = 32,
  highwaympg = 41,
  peakrpm = 6500
)
```

```
predict(ridge_final_fit_continuous, new_data = civic2025)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 52419.
```

```
predict(lasso_final_fit_continuous, new_data = civic2025)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 50858.
```

```
predict(ridge_final_fit_2, new_data = civic2025)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 54539.
```

```
predict(lasso_final_fit_2, new_data = civic2025)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 53931.
```

Prediction of civic 2005

```
civic2005 <- data.frame(fueltype = "gas",
                        carbody = "sedan",
                        drivewheel = "fwd",
                        doornumber = "four",
                        wheelbase = 103.1,
                        carlength = 175.4,
                        carwidth = 67.5,
                        carheight = 55.1,
                        curbweight = 2449,
                        enginesize = 103.7,
                        boreratio = 2.95,
                        stroke = 3.74,
                        compressionratio = 9.5,
                        horsepower = 115,
                        citympg = 27,
                        highwaympg = 34,
                        peakrpm = 6100)
```

```
predict(ridge_final_fit_continuous, new_data = civic2005)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 35511.
```

```
predict(lasso_final_fit_continuous, new_data = civic2005)
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1 33450.
```

```
predict(ridge_final_fit_2, new_data = civic2005)
```

```
# A tibble: 1 x 1  
  .pred  
  <dbl>  
1 37979.
```

```
predict(lasso_final_fit_2, new_data = civic2005)
```

```
# A tibble: 1 x 1  
  .pred  
  <dbl>  
1 37104.
```