## Simple (One Predictor) Linear Regression

Needed libraries

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.0 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.1 ✔ tibble 3.1.8  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.1   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 1.0.0 ──  
## ✔ broom 1.0.3 ✔ rsample 1.1.1  
## ✔ dials 1.1.0 ✔ tune 1.0.1  
## ✔ infer 1.0.4 ✔ workflows 1.1.2  
## ✔ modeldata 1.0.1 ✔ workflowsets 1.0.0  
## ✔ parsnip 1.0.3 ✔ yardstick 1.1.0  
## ✔ recipes 1.0.4   
## ── Conflicts ───────────────────────────────────────── tidymodels\_conflicts() ──  
## ✖ scales::discard() masks purrr::discard()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ recipes::fixed() masks stringr::fixed()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ yardstick::spec() masks readr::spec()  
## ✖ recipes::step() masks stats::step()  
## • Dig deeper into tidy modeling with R at https://www.tmwr.org

library(GGally) #ggcorr and ggpairs

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(ggcorrplot) #correlation plot alternative  
library(gridExtra) #create grids of plots

##   
## Attaching package: 'gridExtra'  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine

Read-in the data. Before doing this make sure that you have placed the CreditData.csv file (downloadable from Canvas) in your project’s working directory.

credit = read\_csv("CreditData.csv")

## Rows: 5000 Columns: 5  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (5): AnnualIncome, HouseholdSize, YrsEdAfterHS, HrWkTV, AnnualCharges  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

Examine the structure and summary of the dataset

str(credit) #all variables numeric

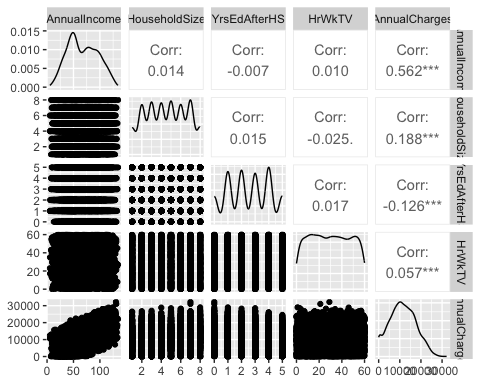
## spc\_tbl\_ [5,000 × 5] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)  
## $ AnnualIncome : num [1:5000] 21.8 65.5 54.2 73.7 110.4 ...  
## $ HouseholdSize: num [1:5000] 4 7 3 6 7 8 5 8 1 3 ...  
## $ YrsEdAfterHS : num [1:5000] 5 3 2 0 5 3 4 5 4 1 ...  
## $ HrWkTV : num [1:5000] 29 46 18 44 39 39 40 27 15 3 ...  
## $ AnnualCharges: num [1:5000] 10024 11249 6115 9786 12634 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. AnnualIncome = col\_double(),  
## .. HouseholdSize = col\_double(),  
## .. YrsEdAfterHS = col\_double(),  
## .. HrWkTV = col\_double(),  
## .. AnnualCharges = col\_double()  
## .. )  
## - attr(\*, "problems")=<externalptr>

summary(credit) #no missingness

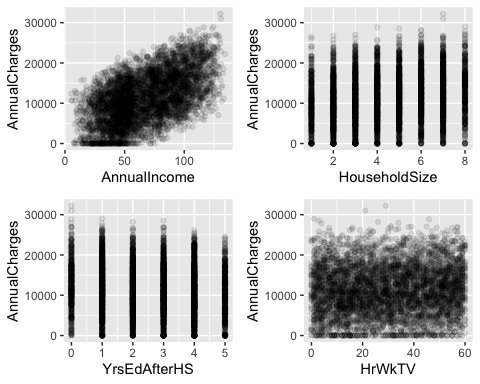
## AnnualIncome HouseholdSize YrsEdAfterHS HrWkTV   
## Min. : 5.40 Min. :1.000 Min. :0.000 Min. : 0.0   
## 1st Qu.: 45.30 1st Qu.:3.000 1st Qu.:1.000 1st Qu.:15.0   
## Median : 65.50 Median :5.000 Median :3.000 Median :30.0   
## Mean : 67.61 Mean :4.534 Mean :2.525 Mean :29.9   
## 3rd Qu.: 90.40 3rd Qu.:6.000 3rd Qu.:4.000 3rd Qu.:45.0   
## Max. :134.20 Max. :8.000 Max. :5.000 Max. :60.0   
## AnnualCharges   
## Min. : 0   
## 1st Qu.: 6926   
## Median :11168   
## Mean :11351   
## 3rd Qu.:15724   
## Max. :32204

Our Y (response) variable in this dataset is “AnnualCharges”. Let’s look at ggpairs plot for visualization and correlation.

ggpairs(credit)

 Alternatively:

p1 = ggplot(credit, aes(x=AnnualIncome,y=AnnualCharges)) + geom\_point(alpha=0.1) #changing alpha is helpful when many points may overlap  
p2 = ggplot(credit, aes(x=HouseholdSize,y=AnnualCharges)) + geom\_point(alpha=0.1)  
p3 = ggplot(credit, aes(x=YrsEdAfterHS,y=AnnualCharges)) + geom\_point(alpha=0.1)  
p4 = ggplot(credit, aes(x=HrWkTV,y=AnnualCharges)) + geom\_point(alpha=0.1)  
grid.arrange(p1,p2,p3,p4,ncol=2)

 The best variable (by correlation and confirmed by visualization) to predict AnnualCharges appears to be AnnualIncome (correlation = 0.562 and there is an intuitive increase in charges as income increases).

Build a regression model with AnnualIncome to predict AnnualCharges.

This is the non-Tidymodels approach

mod1 = lm(AnnualCharges ~ AnnualIncome, credit) #create linear regression model  
summary(mod1) #examine the model

##   
## Call:  
## lm(formula = AnnualCharges ~ AnnualIncome, data = credit)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -12284.4 -3938.1 14.4 3947.9 13232.5   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3146.361 185.193 16.99 <2e-16 \*\*\*  
## AnnualIncome 121.355 2.529 47.98 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5027 on 4998 degrees of freedom  
## Multiple R-squared: 0.3153, Adjusted R-squared: 0.3152   
## F-statistic: 2302 on 1 and 4998 DF, p-value: < 2.2e-16

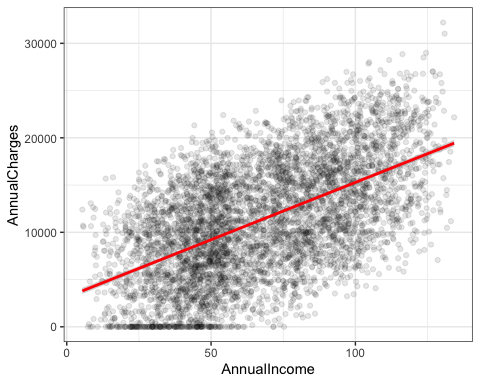
Is this a good model?

R-squared value is OK. The AnnualIncome variable is significant (p-value < 0.05) and has an intuitive (positive) coefficient sign.

Plot the model

ggplot(credit,aes(x=AnnualIncome,y=AnnualCharges)) + geom\_point(alpha=0.1) + geom\_smooth(method = "lm", color = "red") + theme\_bw()

## `geom\_smooth()` using formula = 'y ~ x'



Let’s see how we would do this same model with Tidymodels. We start by building a recipe.

credit\_simple = recipe(AnnualCharges ~ AnnualIncome, credit)  
credit\_simple

## Recipe  
##   
## Inputs:  
##   
## role #variables  
## outcome 1  
## predictor 1

Not too much to see here, but shows the roles that the variables will take in the model.

We’re not going to do any feature engineering at this point. We will also not worry about interaction terms.

Next we specify the type of model that we are building.

lm\_model = #give the model type a name   
 linear\_reg() %>% #specify that we are doing linear regression  
 set\_engine("lm") #specify the specify type of linear tool we want to use

Next we combine the recipe and the model with a workflow.

lm\_wflow =   
 workflow() %>%   
 add\_model(lm\_model) %>%   
 add\_recipe(credit\_simple)

Next we fit (execute) the workflow on our dataset.

lm\_fit = fit(lm\_wflow, credit)

summary(lm\_fit$fit$fit$fit) #three fits :), the actual fit is embedded deeply in the object

##   
## Call:  
## stats::lm(formula = ..y ~ ., data = data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -12284.4 -3938.1 14.4 3947.9 13232.5   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3146.361 185.193 16.99 <2e-16 \*\*\*  
## AnnualIncome 121.355 2.529 47.98 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5027 on 4998 degrees of freedom  
## Multiple R-squared: 0.3153, Adjusted R-squared: 0.3152   
## F-statistic: 2302 on 1 and 4998 DF, p-value: < 2.2e-16

In some ways, this seems harder than the simple line of code for the linear regression model before. However, this approach gives us a lot more flexibility with more complicated models in the future.

Build a regression model with next “best” variable HouseholdSize to predict AnnualCharges.

credit\_simple\_2 = recipe(AnnualCharges ~ HouseholdSize, credit) #recipe  
  
lm\_wflow\_2 = #change name   
 workflow() %>%   
 add\_model(lm\_model) %>% #can re-use the same lm\_model   
 add\_recipe(credit\_simple\_2) #change to new recipe name  
  
lm\_fit\_2 = fit(lm\_wflow\_2, credit)

summary(lm\_fit\_2$fit$fit$fit)

##   
## Call:  
## stats::lm(formula = ..y ~ ., data = data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -13260.4 -4374.8 -105.1 4144.9 19494.1   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 8853.15 202.92 43.63 <2e-16 \*\*\*  
## HouseholdSize 550.91 40.71 13.53 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5967 on 4998 degrees of freedom  
## Multiple R-squared: 0.03535, Adjusted R-squared: 0.03516   
## F-statistic: 183.2 on 1 and 4998 DF, p-value: < 2.2e-16

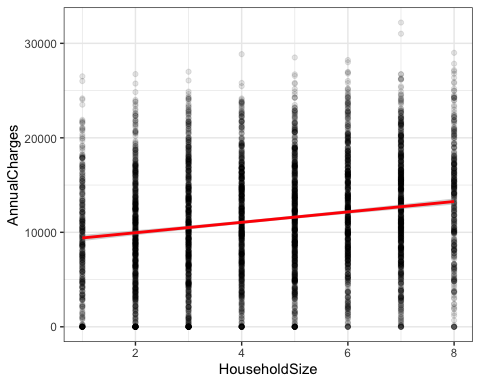
Is this a good model?

R-squared value is pretty poor. The HouseholdSize variable is significant (p-value < 0.05) and has an intuitive sign. Note: As datasets increase in size, it’s VERY easy for the predictor variable to be significant.

Plot the model

ggplot(credit,aes(x=HouseholdSize,y=AnnualCharges)) + geom\_point(alpha=0.1) + geom\_smooth(method = "lm", color = "red") + theme\_bw()

## `geom\_smooth()` using formula = 'y ~ x'

 While the slope is significant, is this really a good model?