## Model Validation Assignment

# Lea Shipley

library(tidyverse)

## -- Attaching packages ------------------------------------------ tidyverse 1.2.1 --

## v ggplot2 3.2.1 v purrr 0.3.3  
## v tibble 2.1.3 v dplyr 0.8.3  
## v tidyr 1.0.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.4.0

## -- Conflicts --------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

library(caret)

## Warning: package 'caret' was built under R version 3.6.2

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

bike = read\_csv("hour.csv")

## Parsed with column specification:  
## cols(  
## instant = col\_double(),  
## dteday = col\_date(format = ""),  
## season = col\_double(),  
## yr = col\_double(),  
## mnth = col\_double(),  
## hr = col\_double(),  
## holiday = col\_double(),  
## weekday = col\_double(),  
## workingday = col\_double(),  
## weathersit = col\_double(),  
## temp = col\_double(),  
## atemp = col\_double(),  
## hum = col\_double(),  
## windspeed = col\_double(),  
## casual = col\_double(),  
## registered = col\_double(),  
## count = col\_double()  
## )

bike = bike %>% mutate(season = as\_factor(as.character(season))) %>%  
mutate(season = fct\_recode(season,"Spring" = "1","Summer" = "2","Fall" = "3","Winter" = "4")) %>%  
 mutate(yr = as\_factor(as.numeric(yr))) %>%  
 mutate(mnth = as\_factor(as.numeric(mnth))) %>%  
 mutate(hr = as\_factor(as.numeric(hr))) %>%  
 mutate(holiday = as\_factor(as.numeric(holiday))) %>%  
 mutate(holiday = fct\_recode(holiday, "NotHoliday" = "0", "Holiday" = "1")) %>%  
 mutate(workingday = as.factor(as.numeric(workingday))) %>%   
 mutate(workingday = fct\_recode(workingday, "NotWorkingDay" = "0", "WorkingDay" = "1")) %>%  
 mutate(weathersit = as.factor(as.numeric(weathersit))) %>%  
 mutate(weathersit = fct\_recode(weathersit, "NoPrecip" = "1", "Misty" = "2", "LightPrecip" = "3" , "HeavyPrecip" = "4")) %>%  
 mutate(weekday = as\_factor(as.numeric(weekday))) %>%  
 mutate(weekday = fct\_recode(weekday, "Monday" = "1", "Tuesday" = "2", "Wednesday" = "3", "Thursday" = "4", "Friday" = "5", "Saturday" = "6", "Sunday" = "0"))

**Task 1**

set.seed(1234)   
train.rows = createDataPartition(y = bike$count, p=0.7, list = FALSE) #70% in training  
train = bike[train.rows,]   
test = bike[-train.rows,]

**Task 2**

The testing set has 5,212 rows, and the training set has 12,167 rows.

**Task 3**

mod1 = lm(count ~ season + mnth + hr + holiday + weekday + temp + weathersit, train) #create linear regression model  
summary(mod1) #examine the model

##   
## Call:  
## lm(formula = count ~ season + mnth + hr + holiday + weekday +   
## temp + weathersit, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -419.31 -61.93 -9.98 52.57 504.24   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -95.3295 6.8961 -13.824 < 2e-16 \*\*\*  
## seasonSummer 28.8486 6.4074 4.502 6.78e-06 \*\*\*  
## seasonFall 19.7865 7.6029 2.602 0.009266 \*\*   
## seasonWinter 62.0339 6.4333 9.643 < 2e-16 \*\*\*  
## mnth2 -0.8013 5.1396 -0.156 0.876114   
## mnth3 2.5584 5.7973 0.441 0.659003   
## mnth4 -1.2250 8.6334 -0.142 0.887166   
## mnth5 -1.5879 9.2279 -0.172 0.863382   
## mnth6 -15.3992 9.4846 -1.624 0.104485   
## mnth7 -38.8277 10.6085 -3.660 0.000253 \*\*\*  
## mnth8 -16.8557 10.3542 -1.628 0.103569   
## mnth9 5.4060 9.2152 0.587 0.557459   
## mnth10 -2.7341 8.5079 -0.321 0.747943   
## mnth11 -12.8043 8.2169 -1.558 0.119193   
## mnth12 -15.3615 6.5409 -2.349 0.018864 \*   
## hr1 -19.7855 6.9722 -2.838 0.004550 \*\*   
## hr2 -28.2440 6.9696 -4.052 5.10e-05 \*\*\*  
## hr3 -40.3146 7.0910 -5.685 1.34e-08 \*\*\*  
## hr4 -40.5469 7.0249 -5.772 8.03e-09 \*\*\*  
## hr5 -26.7454 6.9592 -3.843 0.000122 \*\*\*  
## hr6 32.8518 7.0435 4.664 3.13e-06 \*\*\*  
## hr7 161.3872 6.9925 23.080 < 2e-16 \*\*\*  
## hr8 312.2263 6.9502 44.923 < 2e-16 \*\*\*  
## hr9 164.2556 7.0163 23.411 < 2e-16 \*\*\*  
## hr10 107.1856 6.9552 15.411 < 2e-16 \*\*\*  
## hr11 139.6256 7.0057 19.930 < 2e-16 \*\*\*  
## hr12 179.7448 6.9778 25.760 < 2e-16 \*\*\*  
## hr13 178.6812 7.0201 25.453 < 2e-16 \*\*\*  
## hr14 156.2811 7.0628 22.127 < 2e-16 \*\*\*  
## hr15 168.7543 7.0939 23.788 < 2e-16 \*\*\*  
## hr16 228.1106 7.0881 32.182 < 2e-16 \*\*\*  
## hr17 377.6085 7.0185 53.802 < 2e-16 \*\*\*  
## hr18 347.7287 6.9806 49.813 < 2e-16 \*\*\*  
## hr19 238.7339 7.0128 34.043 < 2e-16 \*\*\*  
## hr20 159.7394 7.0231 22.745 < 2e-16 \*\*\*  
## hr21 108.1070 6.9494 15.556 < 2e-16 \*\*\*  
## hr22 72.3808 6.9874 10.359 < 2e-16 \*\*\*  
## hr23 32.5734 6.9996 4.654 3.30e-06 \*\*\*  
## holidayHoliday -29.0249 6.4088 -4.529 5.98e-06 \*\*\*  
## weekdayMonday 7.5047 3.8928 1.928 0.053894 .   
## weekdayTuesday 6.7559 3.8314 1.763 0.077878 .   
## weekdayWednesday 10.7642 3.7993 2.833 0.004617 \*\*   
## weekdayThursday 12.3082 3.8052 3.235 0.001221 \*\*   
## weekdayFriday 15.3600 3.7730 4.071 4.71e-05 \*\*\*  
## weekdaySaturday 14.0349 3.7638 3.729 0.000193 \*\*\*  
## temp 288.1743 12.1860 23.648 < 2e-16 \*\*\*  
## weathersitMisty -19.6696 2.3717 -8.293 < 2e-16 \*\*\*  
## weathersitLightPrecip -94.1331 3.8166 -24.664 < 2e-16 \*\*\*  
## weathersitHeavyPrecip -80.2490 64.7672 -1.239 0.215356   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 111.9 on 12118 degrees of freedom  
## Multiple R-squared: 0.6217, Adjusted R-squared: 0.6202   
## F-statistic: 414.8 on 48 and 12118 DF, p-value: < 2.2e-16

The model has a good R-squared value of 0.6217, and is statistcally significant with a p-value less than 0.05. The model also matches intuition/common sense, as the variables in the model are likely to influence a person’s decision to rent a bike. The Adjusted R-squared value, while slightly lower than the R-squared, is still good.

**Task 4**

predict\_train = predict(mod1,train)  
head(predict\_train)

## 1 2 3 4 5 6   
## -37.68169 -46.14026 -52.44730 -52.67962 -58.54772 14.95557

There is only one positive value in the first six predictions on the training set, with the other five predictions being -37 and below. Since the number of bike rentals can not be negative, this predictions would equate to no bikes being rented.

**Task 5**

predict\_test = predict(mod1,test)  
head(predict\_test)

## 1 2 3 4 5 6   
## -12.13272 137.72755 174.04493 17.56108 -22.20993 168.48847

The first six predictions on the testing set look more realistic than the predictions from the training set. Only two predictions are negative, and the other values look more like what we’ve seen in the bike dataset.

**Task 6**

SSE = sum((test$count - predict\_test)^2) #sum of squared residuals from model  
SST = sum((test$count - mean(test$count))^2) #sum of squared residuals from a "naive" model  
1 - SSE/SST #definition of R squared

## [1] 0.6289223

The R-squared value for the testing set (0.6289) is just slightly higher than the R-squared value for the training set (0.6217). As there was no severe degradation, it does not appear that the model overfit the training set. The model would be likely to perform well on new data.

**Task 7**

Instead of splitting the data into two separate sets (one for training and one for testing), k-fold cross validation provides ample data for training, while leaving ample data for testing by splitting the data into *k* partitions. A model is built for each partition that is held out. Typically, *k* is set to 3, 5, or 10. k-fold cross-validation allows you to test multiple times, and see how model performance might differ across different partitions.