STAT167 Lab #7 - Spring 2025

Ethan Choi

2025/5/16

Contents

Discussion/Lab #7 instructions

This week, we will review example code of multiple linear regression , polynomial regression, and model evaluation metrics for regressions.

- First, download the `rmd` file from Canvas.
- Open this `rmd` file in RStudio and click `Knit -> Knit to PDF` to render it to PDF format. You need to have `LaTex` installed on the computer to render it to PDF format. If not, you can also render it to HTML format.
- Read this `rmd` file and the rendered `pdf/html` file side-by-side, to see how this document was generated!

- Be sure to play with this document! Change it. Break it. Fix it. The best way to learn R Markdown (or really almost anything) is to try, fail, then find out what you did wrong.
- Read over the example code and the output. If you have any questions about certain functions or parameters, it is the time to ask!
- There are some exercises through out this document. Replace **INSERT_YOUR_ANSWER** with your own answers. Knit the file, and check your results.

Please comment your R code thoroughly, and follow the R coding style guideline (https://google.github.io/styleguide/Rguide.xml). Partial credit will be deducted for insufficient commenting or poor coding styles.

Lab submission guideline

- After you completed all exercises, save your file to `FirstnameLastname-SID-lab7.rmd` and save the rendered pdf file to `FirstnameLastname-SID-lab7.pdf`. If you can not knit it to pdf, knit it to html first and then print/save it to pdf format.
- Submit **BOTH your source `rmd` file and the knitted `pdf` file** to **GradeScope**. Do NOT create a zip file.
- You can submit multiple times, you last submission will be graded.

Install necessary packages

Note that you only need to install each package once. Then you can comment out the following installation lines.

```
#install.packages("MASS")
```

Load necessary package

```
library(MASS) # for the `Boston` data set
library(tidyverse) # for `ggplot2`, `dplyr`, `tibble`, and more
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
library(boot) # for cv.glm()
```

Set the random seed

```r
# set the random seed so that the analysis is reproducible
set.seed(167)
```

The Boston data set

The Boston data set (included in the MASS library) contains housing values in 500+ suburbs of Boston.

```r
?Boston # full documentation
## starting httpd help server ... done
dim(Boston)
## [1] 506  14
glimpse(Boston)
## Rows: 506
## Columns: 14
## $ crim    <dbl> 0.00632, 0.02731, 0.02729, 0.03237, 0.06905, 0.02985, 0.08829,~
## $ zn      <dbl> 18.0, 0.0, 0.0, 0.0, 0.0, 0.0, 12.5, 12.5, 12.5, 12.5, 12.5, 1~
## $ indus   <dbl> 2.31, 7.07, 7.07, 2.18, 2.18, 2.18, 7.87, 7.87, 7.87, 7.87, 7.~
## $ chas    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ nox     <dbl> 0.538, 0.469, 0.469, 0.458, 0.458, 0.458, 0.524, 0.524, 0.524,~
## $ rm      <dbl> 6.575, 6.421, 7.185, 6.998, 7.147, 6.430, 6.012, 6.172, 5.631,~
## $ age     <dbl> 65.2, 78.9, 61.1, 45.8, 54.2, 58.7, 66.6, 96.1, 100.0, 85.9, 9~
## $ dis     <dbl> 4.0900, 4.9671, 4.9671, 6.0622, 6.0622, 6.0622, 5.5605, 5.9505~
## $ rad     <int> 1, 2, 2, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4,~
## $ tax     <dbl> 296, 242, 242, 222, 222, 222, 311, 311, 311, 311, 311, 311, 31~
## $ ptratio <dbl> 15.3, 17.8, 17.8, 18.7, 18.7, 18.7, 15.2, 15.2, 15.2, 15.2, 15~
## $ black   <dbl> 396.90, 396.90, 392.83, 394.63, 396.90, 394.12, 395.60, 396.90~
## $ lstat   <dbl> 4.98, 9.14, 4.03, 2.94, 5.33, 5.21, 12.43, 19.15, 29.93, 17.10~
## $ medv    <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9, 15~
```

Lecture Review - Multiple linear regression

First, let's build a multiple linear regression which uses the **full set** of predictors to estimate the median housing values medv.

```r
mlr <- lm(medv ~ ., data = Boston)
summary(mlr)
##
## Call:
## lm(formula = medv ~ ., data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.595  -2.730  -0.518   1.777  26.199
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288
## chas         2.687e+00  8.616e-01   3.118 0.001925 **
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

Exercise #1

Which predictor is the most statistically significant (contributed the most to medv)? Explain your answer.

I'd say either rm or lstat are the most statistically significant, since their p values are both $<$ 2e-16. This means that they are strongly negatively related to medv. For future questions, lets say lstat is the most significant.

Interpret the coefficient of the most significant predictor.

The estimated coefficient for lstat = -0.5248 (or, -5.248e-01). This means that for every one percent increase in the percentage of lstat, medv decreases by around 524.80 dollars. This affirms our previous answer that there is a strong negative relationship between lstat and housing prices.

Which predictor is the least statistically significant (contributed the least to `medv`)? Explain your answer.

The least statistically significant predictor is age, with a larger p value of 0.958229. This high of a p value means that we fail to reject the null hypothesis for age, meaning it has no impact on predicting medv. ***

Exercise #2

Explain why the R^2 value 0.7406427 is different from the adjusted R^2 value 0.7337897?

R^2 measures how well a model can explain the variance in data, but it will always increase when there are more predictors added. Adjusted R^2 corrects for the number of predictors in the model by reducing, even penalizing, the inclusion of unwanted predictors. The adjusted value only increases when a new variable improves the model.

Lecture Review - Polynomial Regression

Suppose we start with a simple linear regression to predict median housing values `medv` using only one predictor `lstat` – percent of lower status population.

```
lm.fit <- lm(medv ~ lstat, Boston)
summary(lm.fit)
##
## Call:
## lm(formula = medv ~ lstat, data = Boston)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.168  -3.990  -1.318   2.034  24.500
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.55384    0.56263   61.41   <2e-16 ***
## lstat       -0.95005    0.03873  -24.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432
## F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

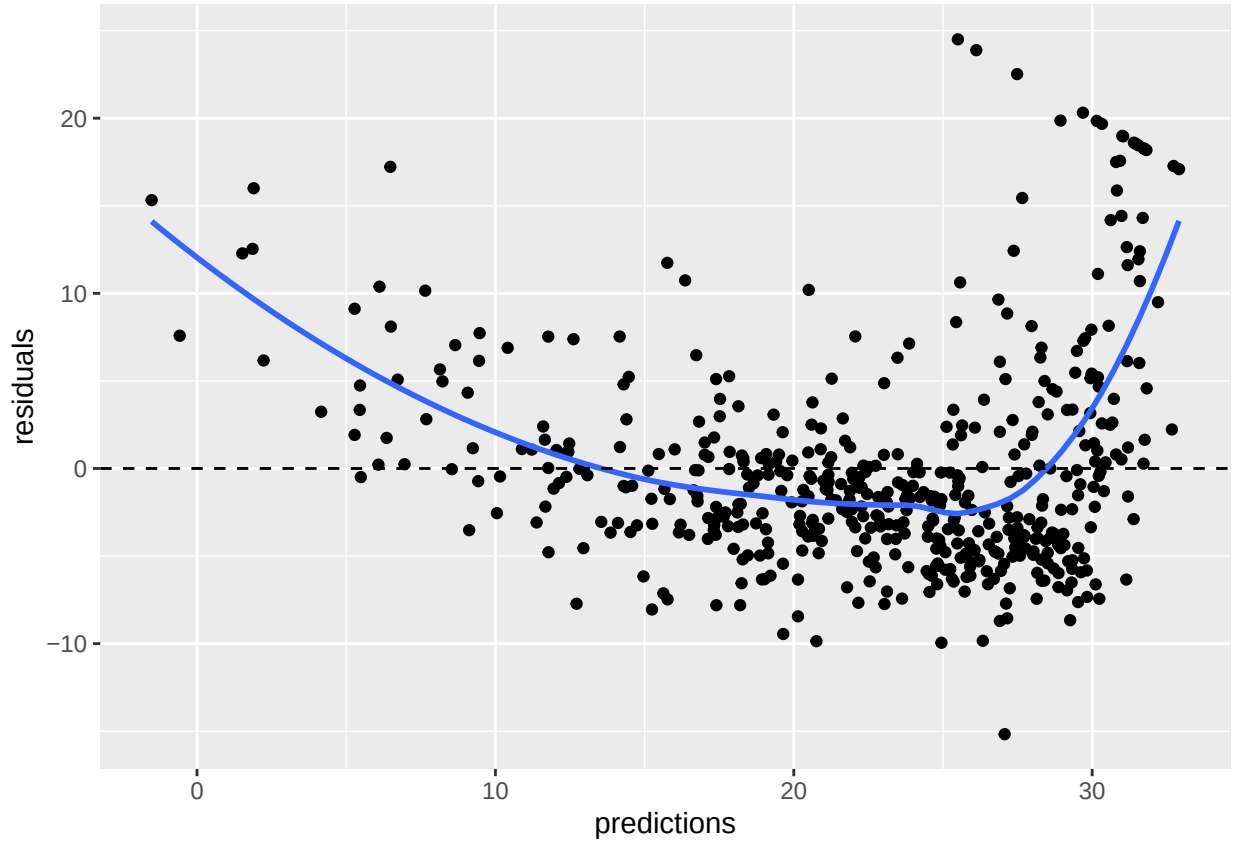
Using the `lm.fit` results, we can get the predicted values from our least squares regression fit and generate the following residual plot.

```
# extract the predictions / fitted values
predictions <- lm.fit$fitted.values
# or call the predict() function
predictions <- predict(lm.fit)

# extract the residuals
residuals <- lm.fit$residuals
# or calculate it yourself
residuals <- Boston$medv - lm.fit$fitted.values
residuals <- Boston$medv - predict(lm.fit)

# draw the residual plot
diagnostics <- tibble(predictions = lm.fit$fitted.values,
                      residuals = lm.fit$residuals)

ggplot(diagnostics, aes(x = predictions, y = residuals)) +
  geom_point() +
  geom_smooth(se = F) +
  geom_hline(yintercept = 0, linetype = 2)
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



The residual plot suggests that there is some non-linearity in the data.

Therefore, we propose to use a polynomial regression model (degree of 2).

```
lm.fit2 <- lm(medv ~ lstat + I(lstat^2), Boston)
summary(lm.fit2)
##
## Call:
## lm(formula = medv ~ lstat + I(lstat^2), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.862007   0.872084   49.15   <2e-16 ***
## lstat       -2.332821   0.123803  -18.84   <2e-16 ***
## I(lstat^2)   0.043547   0.003745   11.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared:  0.6407, Adjusted R-squared:  0.6393
## F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

Alternatively, we can use the `poly()` function to build a polynomial regression model.

```
lm.fit2poly <- lm(medv ~ poly(lstat, 2), Boston)
summary(lm.fit2poly)
##
## Call:
## lm(formula = medv ~ poly(lstat, 2), data = Boston)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       22.5328     0.2456   91.76   <2e-16 ***
## poly(lstat, 2)1 -152.4595     5.5237  -27.60   <2e-16 ***
## poly(lstat, 2)2   64.2272     5.5237   11.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared:  0.6407, Adjusted R-squared:  0.6393
## F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

Exercise #3

(a) Why are the coefficients estimated by `lm.fit2` and `lm.fit2poly` different?

lm.fit2 uses raw polynomial terms that are not orthogonal. lm.fit2poly uses the poly() function, and creates orthogonal polynomial terms that improves stability

(b) Compare the `summary(lm.fit)` output and the `summary(lm.fit2poly)` output. Which model is better?
Explain your answer. What statistic did you use to draw your conclusion?

Both models R squared values and significant statistics are the same, but i would still pick the second one for numerical stability.

Lecture Review - Regression model metrics

Model metrics are score functions which aim to quantify the quality of predictions (either regressions or classifications).

For regression models, the model metrics we have learned so far include:

- **residual sum of squares (RSS)**

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- **residual standard error (RSE)**

$$\begin{aligned}\text{RSE} = \hat{\sigma} &= \sqrt{\frac{1}{n-2}\text{RSS}} \\ &= \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}\end{aligned}$$

- R^2

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}}$$

where

$$\text{TSS} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

- **adjusted R^2**

$$R^2_{\text{adjusted}} = 1 - \frac{(n-1)(1-R^2)}{n-p-1}$$

where n is the number of observations, p is the number of predictors.

- **mean squared error (MSE)**

$$\begin{aligned}\text{MSE} &= E\left[(y - \hat{y}_i)^2\right] \\ &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\end{aligned}$$

Exercise #4

Recall in Homework 5 Question 2(c), you have written your own code to calculate RSE and R^2 statistic.

```
calculateRSE <- function(y, yhat) {
  n <- length(y)
  rss <- sum((y-yhat)^2)
  rse <- sqrt(rss/(n-2))
  return(rse)
}

calculateR2 <- function(y, yhat) {
  tss <- mean((y-mean(y))^2)
  rss <- mean((y-yhat)^2)
  r2 <- 1 - rss/tss
  return(r2)
}
```

Now write your own functions to calculate the adjusted R^2 statistic and MSE.

INSERT_YOUR_ANSWER

```r
calculateR2adj <- function(y, yhat, p) {
  n <- length(y)
  r2 <- 1 - sum((y - yhat)^2) / sum((y - mean(y))^2)
  r2adj <- 1 - ((1 - r2) * (n - 1)) / (n - p - 1)
  return(r2adj)
}
```

```r
# Calculates mean squared error (MSE)
calculateMSE <- function(y, yhat) {
  mse <- mean((y - yhat)^2)
  return(mse)
}
```
