# Tarea 1

Matias Bajac

2024-10-16

## Ejercicio 1.1

Queremos probar que al estimar el error con los mismos datos en el cual el modelo fue estimado subestima el error.\

Para ello tenemos $(x_i, y_i)$ *con* $i = 1, ..., n$ una muestra de entrenamiento y $(x'_i, y'_i$ *con* $i = 1, ...., m$ ) una muestra de testeo.

Los coeficientes estimados de la regresión lineal $\hat{\beta} = (x^T x)^1 x^T y$.

Probar: $E[\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\beta^T} x_i)^2] \leq E[\frac{1}{m}\sum_{i=1}^{m}(y'_i - \hat{\beta^T} x'_i)^2$

Procedemos primero a probar que el error esperado de validación es el mismo si tenemos m observaciones o solamente 1. (a)

Por la regla de la esperanza iterada $E(Z) = E_t[E_v(Z|t)$, tenemos que:

$E(Z) = E_t[E_v(\frac{1}{m}\sum_{i=1}^{m}(y'_i - \hat{\beta^T} x'_i)^2|t] \overset{iid}{=} \frac{1}{\cancel{m}} E_v(\cancel{m}(y'_i - \hat{\beta^T} x'_i) \overset{por\ a)}{=} E_v(y'_1 - \hat{\beta_1}^T x'_1)$ dando por finalizada la prueba.

## Ejericico 1.2

Ahora consideramos dos variables aleatorias

$$A = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\beta^T} x_i)^2 \ y \ B = \frac{1}{n}\sum_{i=1}^{n}(y'_i - \tilde{\beta^T} x'_i)^2$$

donde $\tilde{\beta}$ denota el coeficiente de regresion lineal estimado en el conjunto de validacion. Argumenta que A y B tienen la misma distribucion, entonces E(A) = E(B)

Dado que ambos errores de estimacion surjen de la misma muestra y que m = n , tenemos que

$$E(A) = E(\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\beta^T} x_i)^2) = \frac{1}{n}E(\sum_{i=1}^{n}(y_i - \hat{\beta^T} x_i)^2) = \frac{\cancel{n}}{\cancel{n}}E(y_1 - \hat{\beta^T} x_1)^2$$

$$E(B) = E(\frac{1}{n}\sum_{i=1}^{n}(y'_i - \tilde{\beta^T} x_i)^2) = \frac{1}{n}E(\sum_{i=1}^{n}(y'_i - \tilde{\beta^T} x_i)^2) = \frac{\cancel{n}}{\cancel{n}}E(y'_1 - \tilde{\beta^T} x_1)^2$$

## Ejercicio 2

Para cada parte de la a) a la d) indica si debemos esperar en general que la performance de un método de aprendizaje estadístico **flexible** sea mejor o peor que un mètodo inflexible. Justifica tu respuesta.

a) El tamaño de muestra n es extremadamente grande y el numero de predictores p es pequeño.

b) En este caso lo mejor seria ajustar con un modelo flexible o complejo, ya que al tener un n grande y pocas variable explicativas, podriamos usar modelos no paramétricos, lo que implica que pequeños cambios en nuestros datos, no afectan el ajuste de los parametros.

c) El numero de predictores p es extremadamente grande y el numero de observaciones n es pequeño.

Al contrario del caso anterior, lo mejor seria ajustar con un modelo poco flexible (como por ej un modelo lineal), ya que si lo hacemos con un modelo flexible, corremos el riesgo de que haya sobreajuste.

c) La relacion entre los predictores y la respuesta en **no lineal**

El método flexible seria mejor en este caso que uno inflexible, por ejemplo podriamos usar splines.

d) La varianza del termino del error $\sigma^2 = V(\epsilon)$ es extremadamente alta

Al tener un error irreducible ($\sigma^2$) muy grande, corremos el riesgo que al usar un metodo flexible en el conjunto de entrenamiento, la f(x) estimada ajsuste bien a los datos pero de forma erronea, por lo que podria predecir mal afuera de la muestra.Por lo tanto, para evitar el sobreajuste, seria mejor un modelo poco flexible.

## Ejercicio 3

Cuales son las ventajas y desventajas de una aproximacion muy flexible vs una menos felxible para un problema de regresion o clasificacion? Bajo que circunstancias puede una aproximacion mas flexible ser preferida a una menos flexible? Cuando una aproximacion menos flexible es preferible?

Si nuestro objetivo es meramente interpretativo, y sin importar predecir puede ser preferible usar un metodo poco flexible, ahora bien, si nuestro objetivo es por ejemplo de prediccion quizas nos convenga mas usar metodos mas flexibles. Hay que tener en cuenta el trade -off entre sesgo y varianza, si contamos con un conjunto de datos en el cual tenemos muchas observaciones y pocos parametros para estimar, nos va a convenir usar un metodo flexible, en el cual seguramente tengamos una varianza grande, pero en general un sesgo mas chico que si usaramos un metodo inflexible. Tambien podriamos usar un método flexible si no conocemos a priori f(x) o la relacion entre variables es no lineal, en caso contrario si conocemos f, lo mejor seria ajustar con un método parametrico.

## Ejercicio 4

Cuidadosamente explique la diferencia entre un metodo de vecino mas cercano para clasificacion y regresion

En clasificacion el objetivo es predecir una nueva observacion en un grupo g, para ello podemos usar el algoritmo del vecino mas cercano y usamos el clasificador de bayes para predecir la clase. En la practica es dificil conocer la distribucion de Y/X por lo que el clasificador de Bayes no puede ser calculado. El vecino mas cercano estima la distribucion de Y/X y **asigna las observaciones a la clase con mayor probabilidad estimada**

Dado un $K \in N$ y una observacion $x_0$, KNN primero identifica los k vecinos mas cercanos. Esto forman el conjunto $N_0$. Despues se estima la probabilidad condicional para la clase j como la fraccion de puntos en $N_o$ cuya variable de respuesta es igual a j, esto es:

$$Pr(Y = j/X = x_0) = \frac{1}{K} \sum_{i \in N_0} I[y_i] = j$$

Cuando k =1 KNN es demasiado flexible y encuentra patrones en los datos que no se corresponden con el clasificdor de Bayes. El error de training es cero, pero el error en el grupo de test puede ser muy alto. A medida que K aumenta, el metodo se vuelve menos flexible y eventualmente, produce un clasificador que es lineal

Mientras que en regresion, en vez de encontrar individuos en un vecindario, condicionamos en un punto. Estimamos f con el **promedio local** de los K puntos mas cercanos a x

$\hat{f}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$

para calcular la distancia entre los puntos se puede usar la distancia euclidiana con cada uno de los k vecinos estimo los premedios.

# Ejercicio 5

Suponga que contamos con un conjunto de datos con 100 observaciones (n = 100) que contienen un único predictor y una respuesta cuantitativa. Se ajusta un modelo de regresión lineal para los datos así como un regresion cúbica $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$

a) Supone que la **verdadera** relacion entre X e Y es **lineal**. Considerar la suma de cuadrados de los residuos de entrenamiento (SCR) para la regresion lineal y la SCR para la regresion cubica. Esperarias que una sea menor que otra, que sean iguales o no hay suficiente informacion para concluir. Justifica tu respuesta

Para los datos de entrenamiento, si tenemos mayor grado de libertad (por ejemplo una regresion cubica), la $SCR = \sum_{i=1}^{n} (y_i - \hat{y_i})^2$ de una regresion cubica sera mas chica que la de una lineal. Lo podemos visualizar en el grafico 2.10 del ISL, donde se ve que a medida que el modelo es mas flexible, el ECM de entrenamiento disminuye.

b) Reporte el punto anterior utilizando el conjunto test en vez del entrenamiento para calcular la SCR

En este caso esperaria que la SCR test de una regresion lineal sea menor a la de una cúbica, ya que en este caso al conocer f, tendriamos sesgo 0 y varianza chica, mientras que para metodos mas flexibles (regresion cubica) aumentaria el sesgo y tambien la varianza.

c) Supone que la verdadera relacion entre X e Y **no es lineal** pero no sabemos que tan lejana está de la linealidad. Considerar la SCR de entrenamiento para la regresion lineal y tambien la SCR de entrenamiento para la regresion cubica. Esperariamos que una sea mayor que la otra, que sean iguales o no hay suficiente informacion para concluir? Justifica tu respuesta

Seguramente en esta caso tendremos una SCR mas chico para una regresion cúbica, ya que al ser la verdadera f no lineal, a medida que el modelo es mas flexible el ECM de entrenamiento dismiuye, aunque tendriamos que tener en cuenta los grados de libertad o splines para ver que realmente el modelo ajuste bien ya que podriamos tener sobreajuste.

d) Responder la anterior usando el conjunto test en vez de entrenamiento.

En este caso no queda tan claro cual va a ser menor, ya que a medida que aumenta la flexibilidad el error de test empieza a disminuir, pero llega un punto en donde vuelve a aumentar el error cuadratico medio. El MSE test empieza a aumentar en modelos flexibles debido a que los datos de testeo no generalizan bien los nuevos datos.

## Ejercico 6

Es un problema de regresion donde Y es la variable de respuesta y X la variable predictora, se cuenta con n = 20 observaciones que se presentan en la tabla

El objetivo de este ejercico es obtener el MSE por validacion cruzada usando 5 pliegues (k =5), en este problema la relacion entre Y y X es ajustada con un modelo de regresion lineal simple tal que $f(x) = \beta_0 + \beta_1 X$. En lo que sigue se presentan las IID de las observaciones seleccionadas aleatoriamente en cada pliegue.

Utiliza la informacion antes detallada para explicitar en cada pliuege los calculos que debes realizar para obtener el MSE 5- CV (estimacion del mse por validacion cruzada usando 5 pliegues). En este ejercicio es importante el proceso para llegar al resultado final no simplemente obtener el MSE 5 -CV.

El procedimiento para Cross validation k pliegues es el siguiente: se divide el conjunto de datos original en k folds (o pliegues) de igual tamaño en este caso 5 pliegues de tamaño 4 cada uno.

El pliegue 1 es usado como el conjunto de validacion, y el metodo consiste en ajustar los restantes 4 pliegues. Este proceso es iterativo, se repite 5 veces, cada vez, diferentes pliegues es tratado como un conjunto de validacion. El proceso resulta en 5 estimaciones de test de error, $MSE_1, MSE_2..., MSE_5$, el k - folds cross validation es estimado computando el promedio de esos valores.

$$CV_5 = \frac{1}{5}\sum_{i=1}^{5} ECM_{[i]}$$

Para empezar estimamos el intercepto $\beta_0$ y $\beta_1$ para el primer conjunto de entrenamiento.

Hallo las estimaciones el modelo lineal simple con lm() :

```
df = data.frame(id = seq(1:20), y = c(8,9,14,10,10,15,11,6,7,8,13,11,11,10,8,15,11,4,12,8), x  = c(6,8,
                                                                                                  ,6,5,
```

Por ejemplo la estimacion de $\beta_0$ y $\beta_1$ para el primer conjunto de entrenamiento es:

$\hat{y_i} = \hat{\beta_0} + \hat{\beta_1} x_i = 0.7394 + 0.9822 x_i$

Hacemos el promedio de los errores para las 4 observaciones que separamos de cada test.

Luego de calculado los 5 ECM, hacemos el prmedio de los 5 MSE , dando como reultado 1.870914

Calculos:

```
train1 = df[-c(4,3,19,16),]
train1 = train1$x


# Ajustamos modelo usando datos tr
fit <- lm(y ~ x, subset = train1,  data = df)

# Calculamos error MSE en los datos NO utilizados
 mse_1 = mean( (df$y - predict(fit, df))[-train1]^2 )

 #0.3679264

train2 = df[-c(2,15,7,18),]
train2 = train2$x


# Ajustamos modelo usando datos tr
fit2 <- lm(y ~ x, subset = train2,  data = df)

# Calculamos error MSE en los datos NO utilizados
mse_2 = mean( (df$y - predict(fit2, df))[-train2]^2 )

#0.5329727

train3 = df[-c(9,14,12,20),]
train3 = train3$x
```

```
fit3 <- lm(y ~ x, subset = train3,  data = df)


mse_3= mean( (df$y - predict(fit3, df))[-train3]^2 )

#0.5383798


train4 = df[-c(17,6,8,10),]
train4 = train4$x

fit4 <- lm(y ~ x, subset = train4,  data = df)


mse_4= mean( (df$y - predict(fit4, df))[-train4]^2 )

#0.3596957


train5 = df[-c(1,5,13,11),]
train5 = train5$x

fit5 <- lm(y ~ x, subset = train5,  data = df)


mse_5= mean( (df$y - predict(fit5, df))[-train5]^2 )


#0.3596957


cv = mse_1 + mse_2 + mse_3 + mse_4 + mse_5 /5
 cv
```

```
## [1] 1.870914
```

# Ejercicios ISLR

1) Capitulo 5, ejercicio 8

2) Capitulo 7 Ejercicios 1,2,3,4 y ejercicio 9 y 10 usando **tidymodels**

Capitulo 7, ejercicio 1

1) tenemos que el polinomio de grado 3 es de la forma : $f(x) = \beta_0 \ + \ \beta_1 x \ + \ \beta_2 x^2 \ + \beta_3 x^3 \ + \beta_4 (x - \xi)^3$ si $x > \xi$

a) $f_1(x) = a_1 + b1x + c_1 x^2 + d_1 x^3$

Entonces $a_1 = \beta_0, \ b_1 = \beta_1, \ c_1 = \beta_2, \ d_1 = \beta_3$

para el caso b) tenemos que $f_2(x) = a_2 + b_2 x + c2x^2 + d_2 x^3$

Si desarrollamos el trinomio de la parte de la funcion cubica $\beta_4 (x - x_i)^3$, entonces

$\beta_4 (x - \xi)^3 = \ \beta_4 (x^3 \ - 3x^2 \xi \ + 3x \xi^2 - \xi^3)$ , ordenando terminos llegamos a que:

$(\beta_0 - \beta_4\xi^3) + (\beta_1 + \beta_4 3\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3$

$a_2 = \beta_0 - \beta_4\xi^3$

$b_2 = \beta_1 + 3\beta_4\xi^2$

$c_2 = \beta_2 - 3\beta_3\xi$

$d_2 = \beta_4 + \beta_3$

c) Haciendo el limite por izquierda y por derecha se asegura la continuidad de f.

d) tenemos por c) que $f_1(\xi) = f_2(\xi)$, entonces $f_2 = (\beta_0 \cancel{-\beta_4\xi^3}) + (\beta_1 \cancel{+3\beta_4\xi^2})\xi + (\beta_2 \cancel{-3\beta_3\xi})\xi^2 + (\beta_3 \cancel{+\beta_4})\xi^3$

$f_2'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2$

por transitiva $f_1\prime(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2$

e) $f_2''(\xi) = +2\beta_2 + 6\beta_3\xi = f_1''(\xi)$

2) $\hat{g} = argmin(g)(\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda\int[g^{(m)}(x)]^2dx)$

a) $\lambda = \infty$ *y* $m = 0$, dado que el parametro de penalizacion lambda es grande y el grado m es 0, la estmiacion de $\hat{g} = 0$, minimizando la scr.

b) Ahora dado que el parametro de penalizcion sigue siendo infinito y el grado es igual a 1, la derivada de la estimacion de g es igual a una constante k.

c) $\lambda = \infty, m = 2$

Este es un caso de spline suavizada, la funcion es de la forma $\hat{g} = \hat{g} = argmin(g)(\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda\int[g^{(2)}(x)]^2dx)$

siendo lambda el parametro de ajuste que controla la flexibilidad de la spline puede estimarse mdiante cv (es decir queremos el optimo del valor de lambda que haga la scr lo mas chica posible)

en particular podemos notar que $\hat{g}$ es lineal, siendo $\hat{g} = a + bx$

d) Le agregamos un grado mas manteniendo la penalizacion, por lo que $\hat{g} = a + bx + cx^2$

e) $\lambda = 0, m = 3$ no hay penalizacion, por lo que $\hat{g}$ puede ser muy flexible y generar sobreajuste. Se debe estimar por minimos cuadrados.

Ejercicio 9 del capitulo 7:

This question uses the variables **dis**(the weighted mean of distances to five Boston employment) and **nox** (nitrogen oxides concentration in parts per 10 millon) from **Boston** data. We will treat **dis** as the predictor (X) and nox as the response (Y)

a) Use de poly() function to fit a cubic polynomial regression to predict **nox** using **dis**. Report the regression output, and plot the resulting data and polynomial fits

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 1.2.0 --
```

```
## v broom        1.0.5      v recipes      1.0.10
## v dials        1.2.1      v rsample      1.2.1
## v dplyr        1.1.4      v tibble       3.2.1
## v ggplot2      3.5.0      v tidyr        1.3.1
## v infer        1.0.7      v tune         1.2.1
## v modeldata    1.4.0      v workflows    1.1.4
## v parsnip      1.2.1      v workflowsets 1.1.0
## v purrr        1.0.2      v yardstick    1.3.1
```

```
## -- Conflicts ---------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Use tidymodels_prefer() to resolve common conflicts.
```

```
library(ISLR2)



Boston = as.tibble(Boston)
```

```
## Warning: `as.tibble()` was deprecated in tibble 2.0.0.
## i Please use `as_tibble()` instead.
## i The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
rec_poly <- recipe(nox ~ dis, data = Boston) %>%
  step_poly(dis, degree = 3, options = list(raw = TRUE))



lm_spec <- linear_reg() %>%
  set_mode("regression") %>%
  set_engine("lm")

poly_wf <- workflow() %>%
  add_model(lm_spec) %>%
  add_recipe(rec_poly)


poly_fit = fit(poly_wf, data = Boston)
tidy(poly_fit)
```

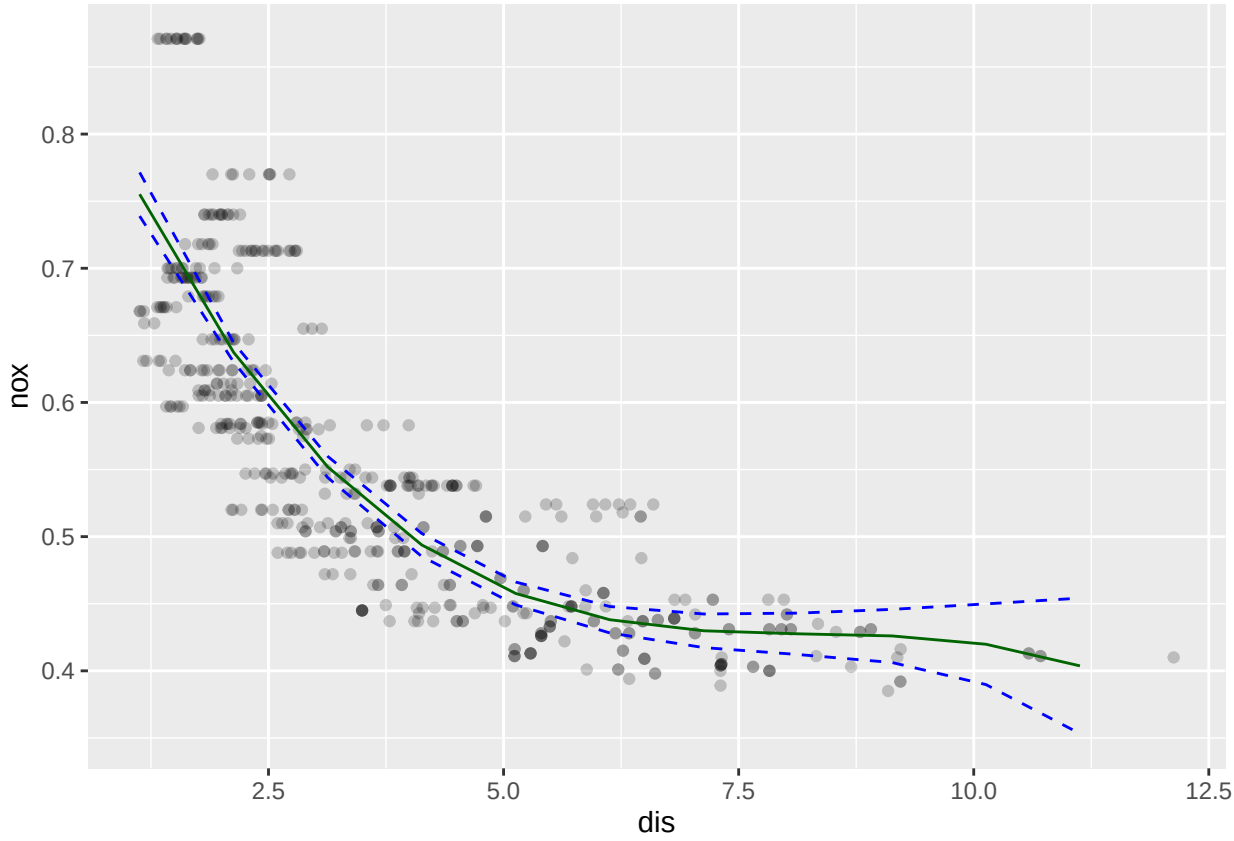
```
## # A tibble: 4 x 5
##   term         estimate std.error statistic   p.value
##   <chr>           <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)  0.934     0.0207       45.1  9.85e-179
## 2 dis_poly_1  -0.182     0.0147      -12.4  6.08e- 31
## 3 dis_poly_2   0.0219    0.00293       7.48 3.43e- 13
## 4 dis_poly_3  -0.000885  0.000173     -5.12 4.27e-  7
```

```
## nos creamos una grilla de valores para la variable  dis para la prediccion
dis_range <- tibble(dis = seq(min(Boston$dis), max(Boston$dis)))

regression_lines <- bind_cols(
  augment(poly_fit, new_data = dis_range),
  predict(poly_fit, new_data = dis_range, type = "conf_int"))


sum(regression_lines$.resid^2) ## scr
```

```
## Warning: Unknown or uninitialised column: `.resid`.
```

```
## [1] 0
```

```
Boston %>%
  ggplot(aes(dis, nox)) +
  geom_point(alpha = 0.2) +
  geom_line(aes(y = .pred), color = "darkgreen",
            data = regression_lines) +
  geom_line(aes(y = .pred_lower), data = regression_lines,
            linetype = "dashed", color = "blue") +
  geom_line(aes(y = .pred_upper), data = regression_lines,
            linetype = "dashed", color = "blue")
```

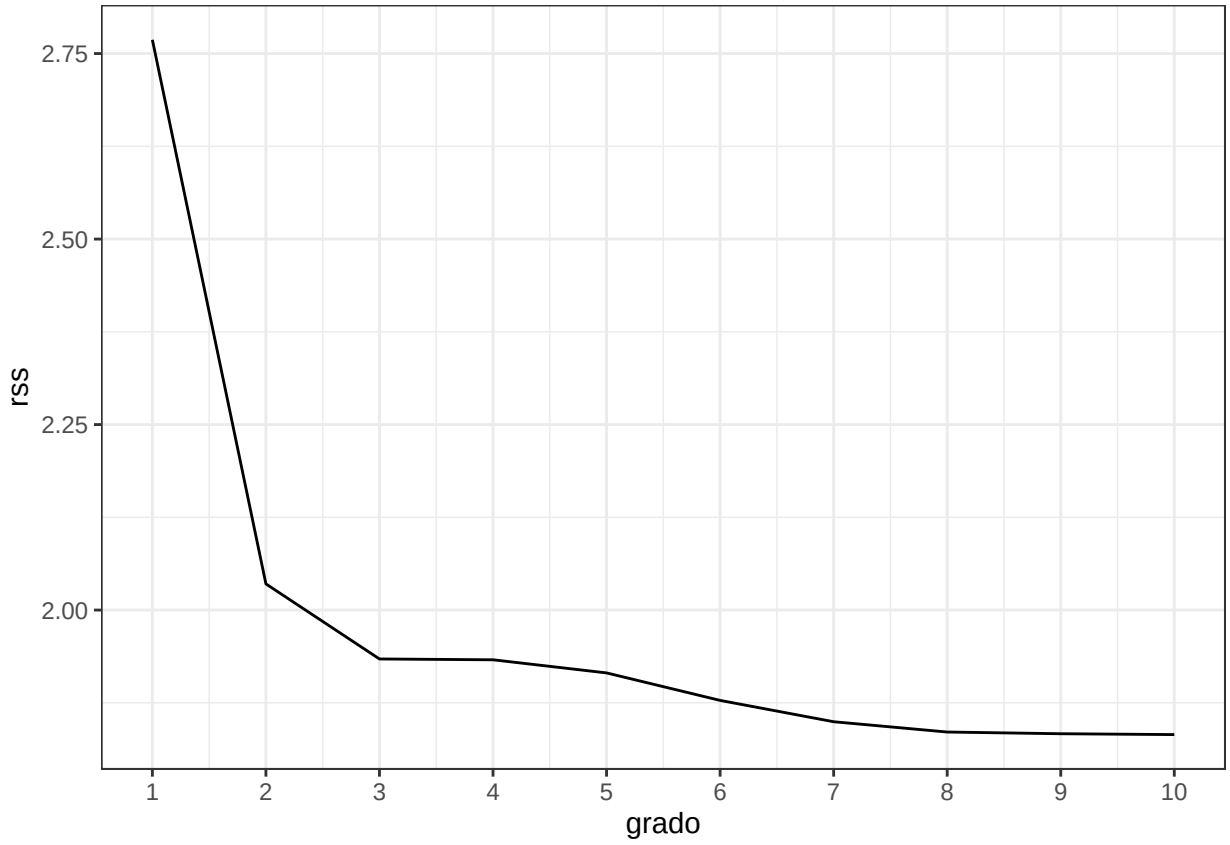



b) Plot the polynomial fits for a range od different polynomial degree( say, from 1 to 10),and report the associated residual sum of squares.

Me creo un vector de ceros para guardar la suma de cuadrados residuales $SCR = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ de los 10 grados del polinomio

```
 lm_spec <- linear_reg() %>%
  set_mode("regression") %>%
  set_engine("lm")

rss  =rep(0,10) ## me creo un vector de 0 para almancenar la suma de cuadrados residuales

for (i in 1:10) {
```

```
   ## creo la receta para los i grados
  rec_poly = recipe(nox ~ dis, data = Boston) %>%
    step_poly(dis,degree = i, options = list(raw=TRUE))

# flujo de trabajo, especifico el modelo
  poly_wf = workflow() %>%
    add_model(lm_spec) %>%
    add_recipe(rec_poly)


  # Ajusto el modelo con los datos de Boston, aqui se podria haber hecho con el train data pero no lo p
poly_fit  = fit(poly_wf, data = Boston)

  regresion_fit = augment(poly_fit, new_data =Boston )


  rss[i] = sum(regresion_fit$.resid^2)
}

data = data.frame(rss = rss, grado = c(1,2,3,4,5,6,7,8,9,10))

data %>%  ggplot(aes(x = grado,y = rss)) + geom_line() + scale_x_continuous(breaks = seq(1,10,1))  + th
```



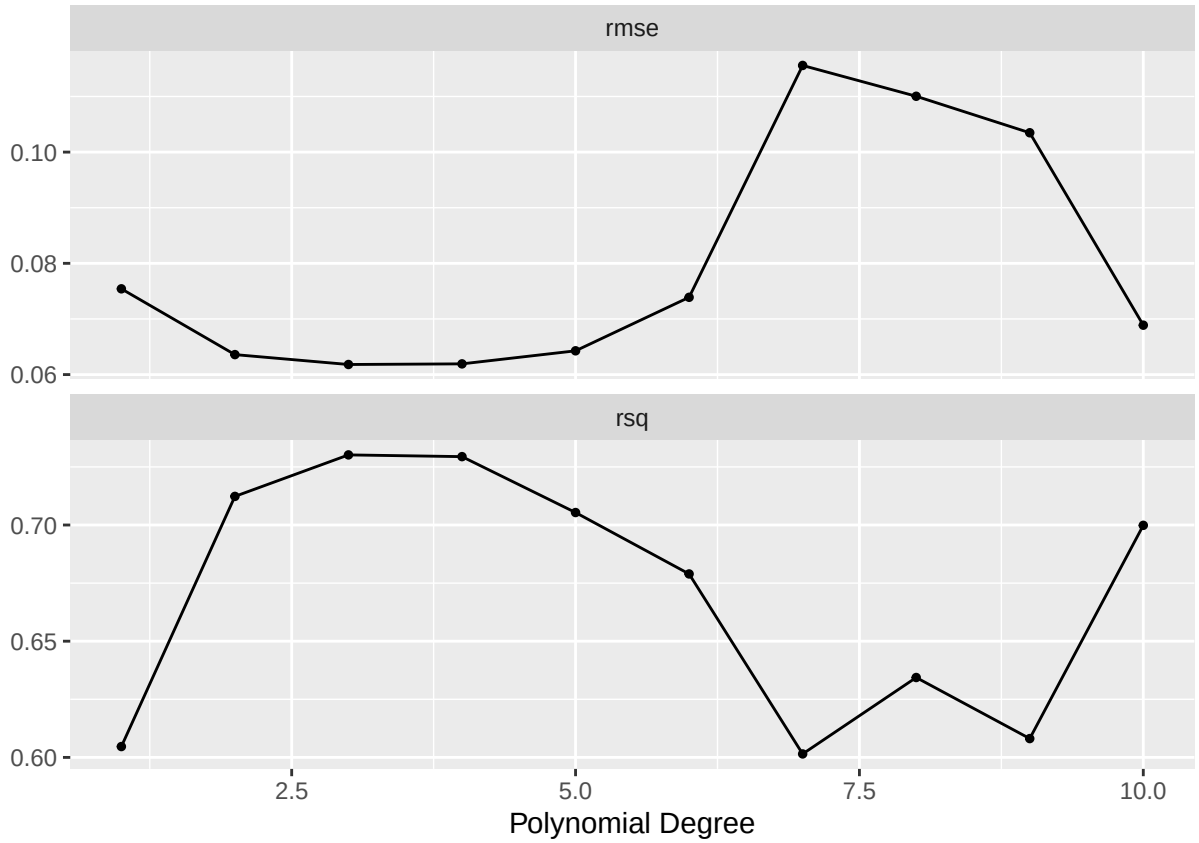

Podemos ver que el grado optimo es 3 cuando la suma de cuadrados residuales se empieza a establilizar

3) Perform cross - validation or another approach to select the optimal degree for the polynomial,and

explain your results.

Vemos graficamente que el grado 3 o 4 es optimo, pues tiene el menor error

```
split = initial_split(Boston,prop=3/4)

train_data = training(split)
test_data = testing(split)


poly_tuned_rec <- recipe(nox ~ dis, data = train_data) %>%
  step_poly(dis, degree = tune())

poly_tuned_wf <- workflow() %>%
  add_recipe(poly_tuned_rec) %>%
  add_model(lm_spec)

boston_folds = vfold_cv(train_data, v = 5)

degree_grid <- grid_regular(degree(range = c(1, 10)), levels = 10)



tune_res <- tune_grid(
  object = poly_tuned_wf,
  resamples = boston_folds,
  grid = degree_grid
)




 autoplot(tune_res)
```

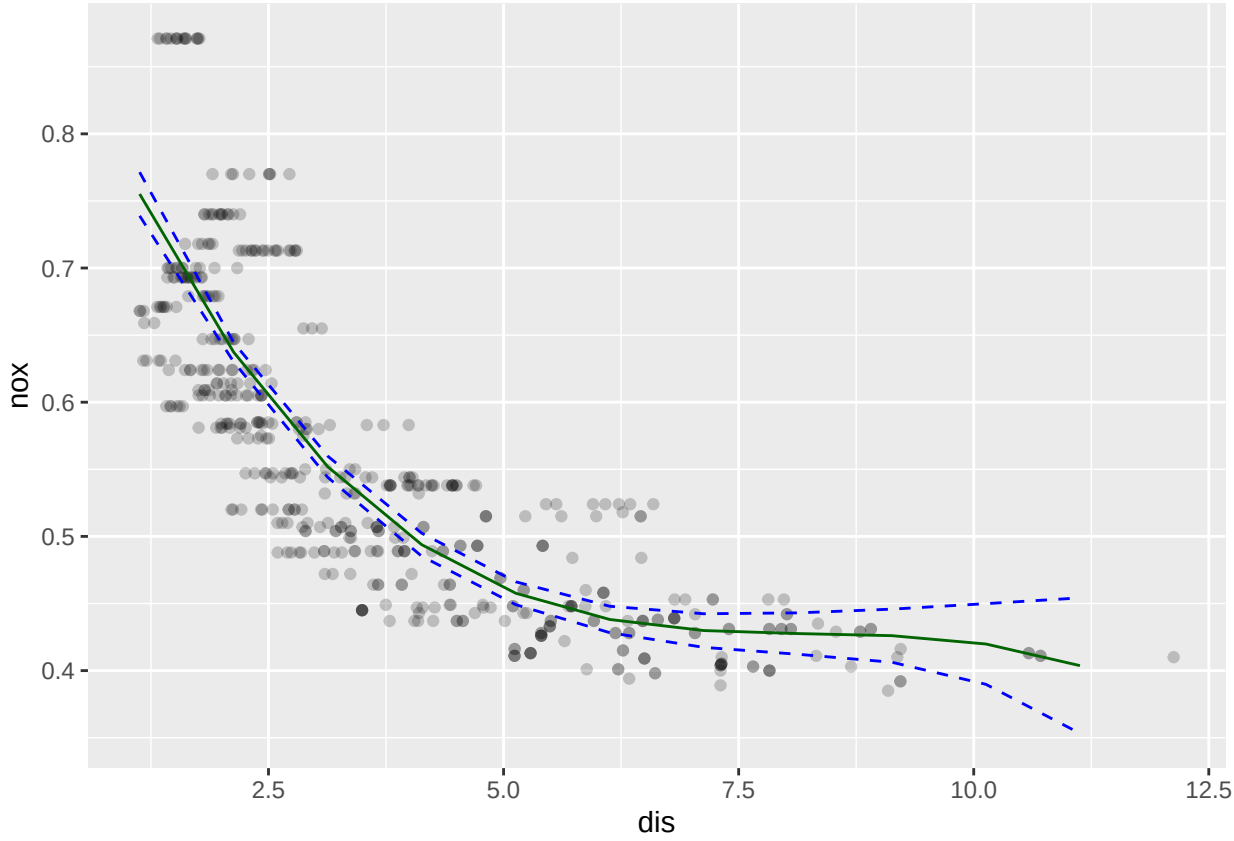



4) Use the bs() function to fit a **regression spline** to predict nox using dis. Report the output for the fit using 4 degrees of freedom. How did you choose the knots? Plot the resulting fit

```
rec_spline <- recipe(nox ~ dis, data = Boston) %>%
  step_bs(dis,  options = list(knots = 3, 5, 11))

spline_wf <- workflow() %>%
  add_model(lm_spec) %>%
  add_recipe(rec_spline)

spline_fit <- fit(spline_wf, data = Boston)
predict(spline_fit, new_data = dis_range)
```

```
## # A tibble: 11 x 1
##    .pred
##    <dbl>
##  1 0.755
##  2 0.637
##  3 0.552
##  4 0.494
##  5 0.458
##  6 0.438
##  7 0.430
##  8 0.428
##  9 0.426
## 10 0.420
## 11 0.404
```

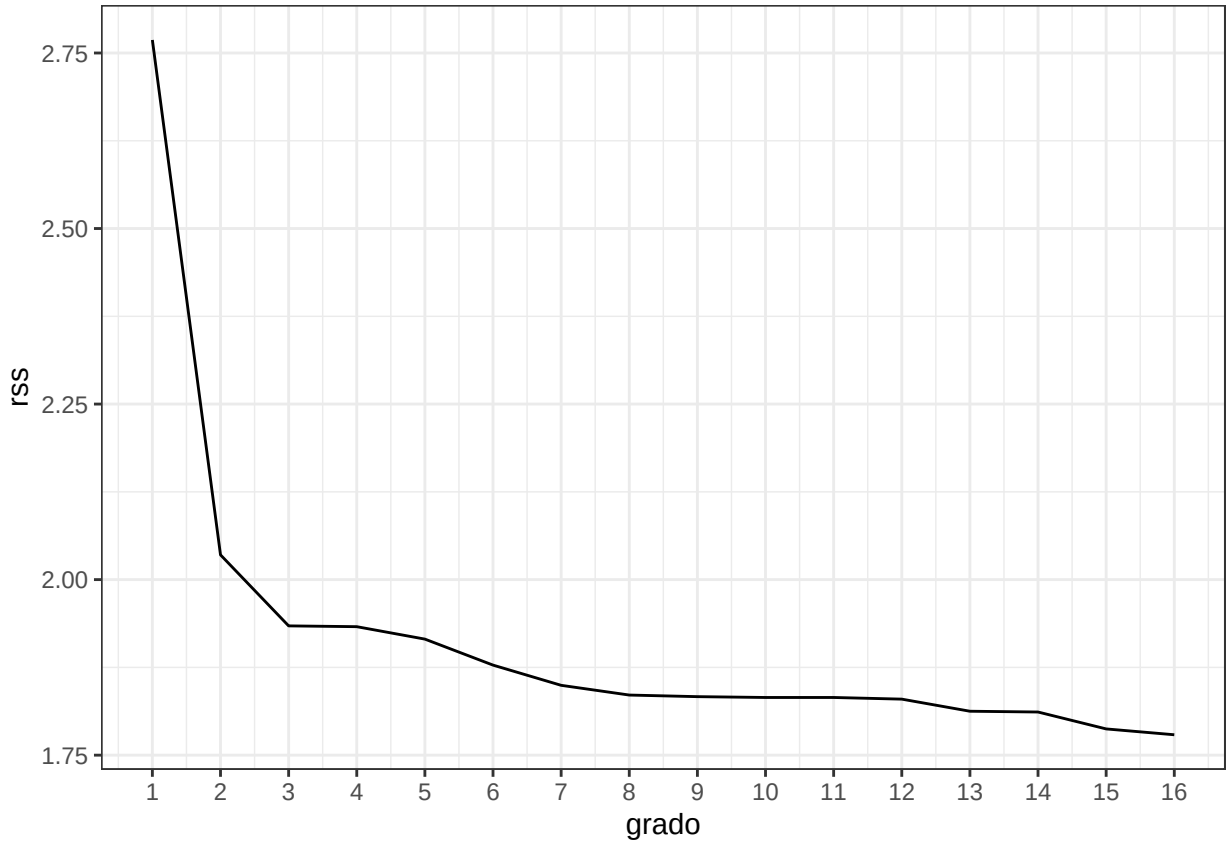
```
regression_lines <- bind_cols(
  augment(spline_fit, new_data = dis_range),
  predict(spline_fit, new_data = dis_range, type = "conf_int")
)
Boston %>%
  ggplot(aes(dis ,nox)) +
  geom_point(alpha = 0.2) +
  geom_line(aes(y = .pred), data = regression_lines, color = "darkgreen") +
  geom_line(aes(y = .pred_lower), data = regression_lines,
            linetype = "dashed", color = "blue") +
  geom_line(aes(y = .pred_upper), data = regression_lines,
            linetype = "dashed", color = "blue")
```




e) Now fit a regression spline for a range of degrees of freedom, and plot the resulting fits and report the resulting RSS. Describe the results obtained.

```
rss  =rep(0,16) ## me creo un vector de 0 para almancenar la suma de cuadrados residuales

for (i in 1:16) {

   ## creo la receta para los i grados de las splines
  rec_spline = recipe(nox ~ dis, data = Boston) %>%
    step_bs(dis, degree = i)

# flujo de trabajo, especifico el modelo
  spline_wf = workflow() %>%
```

```
    add_model(lm_spec) %>%
    add_recipe(rec_spline)


  # Ajusto el modelo con los datos de Boston, aqui se podria haber hecho con el train data pero no lo p
  spline_fit  = fit(spline_wf, data = Boston)

  regresion_fit = augment(spline_fit, new_data = Boston)


  rss[i] = sum(regresion_fit$.resid^2)

}

data = data.frame(rss = rss, grado = c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16))

data %>%  ggplot(aes(x = grado,y = rss))  + geom_line() + scale_x_continuous(breaks = seq(1,16,1))  + tl
```




f)

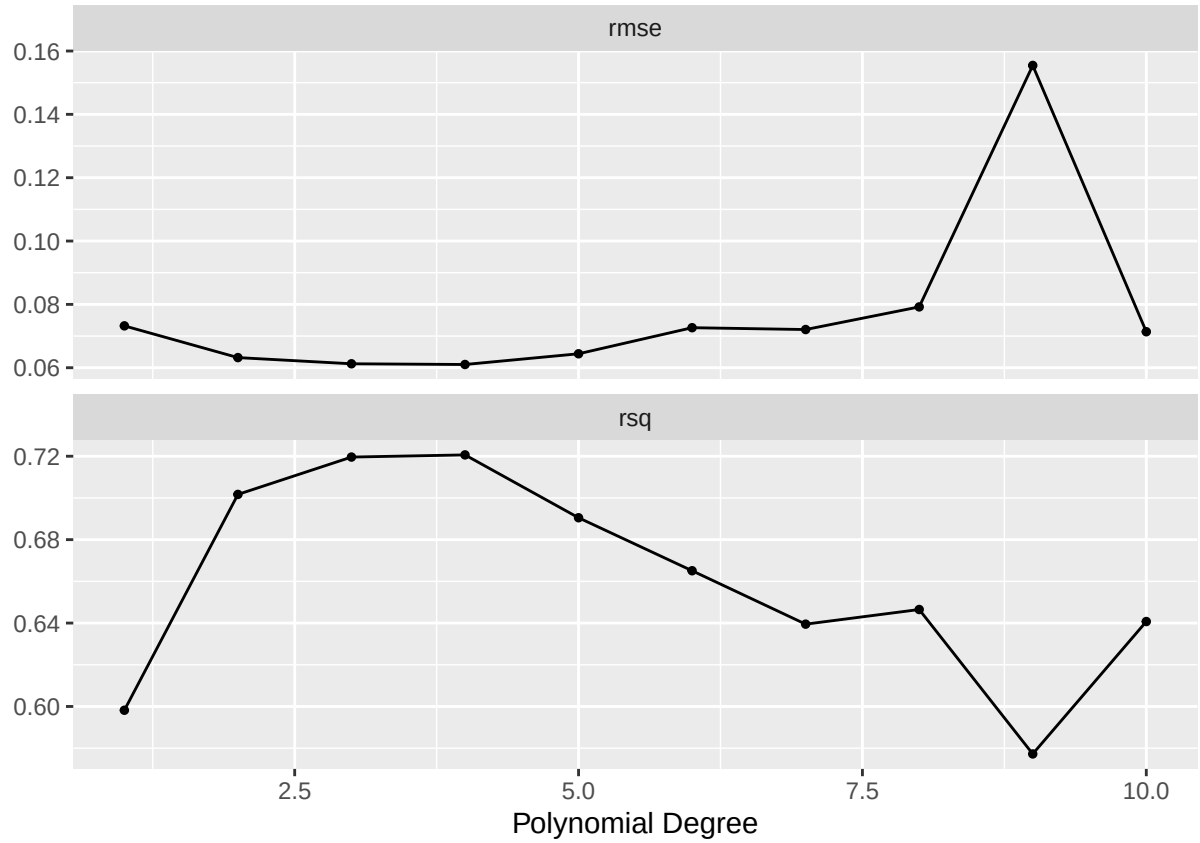
Perform cross validation or another approach in order to select the best degrees of freedom for **regression splines** on this data. Describe your results.

```
split = initial_split(Boston,prop=3/4)

train_data = training(split)
test_data = testing(split)
```

```
rec_spline <- recipe(nox ~ dis, data = train_data) %>%
   step_bs(dis, degree = tune())


  spline_wf = workflow() %>%
    add_model(lm_spec) %>%
    add_recipe(rec_spline)



boston_folds = vfold_cv(train_data, v = 5)

degree_grid <- grid_regular(degree(range = c(1, 10)), levels = 10)



tune_res <- tune_grid(
  object = spline_wf,
  resamples = boston_folds,
  grid = degree_grid
)
```

```
## > A | warning: some 'x' values beyond boundary knots may cause ill-conditioned bases

## There were issues with some computations   A: x1

## There were issues with some computations   A: x10

##
```

```
autoplot(tune_res)
```

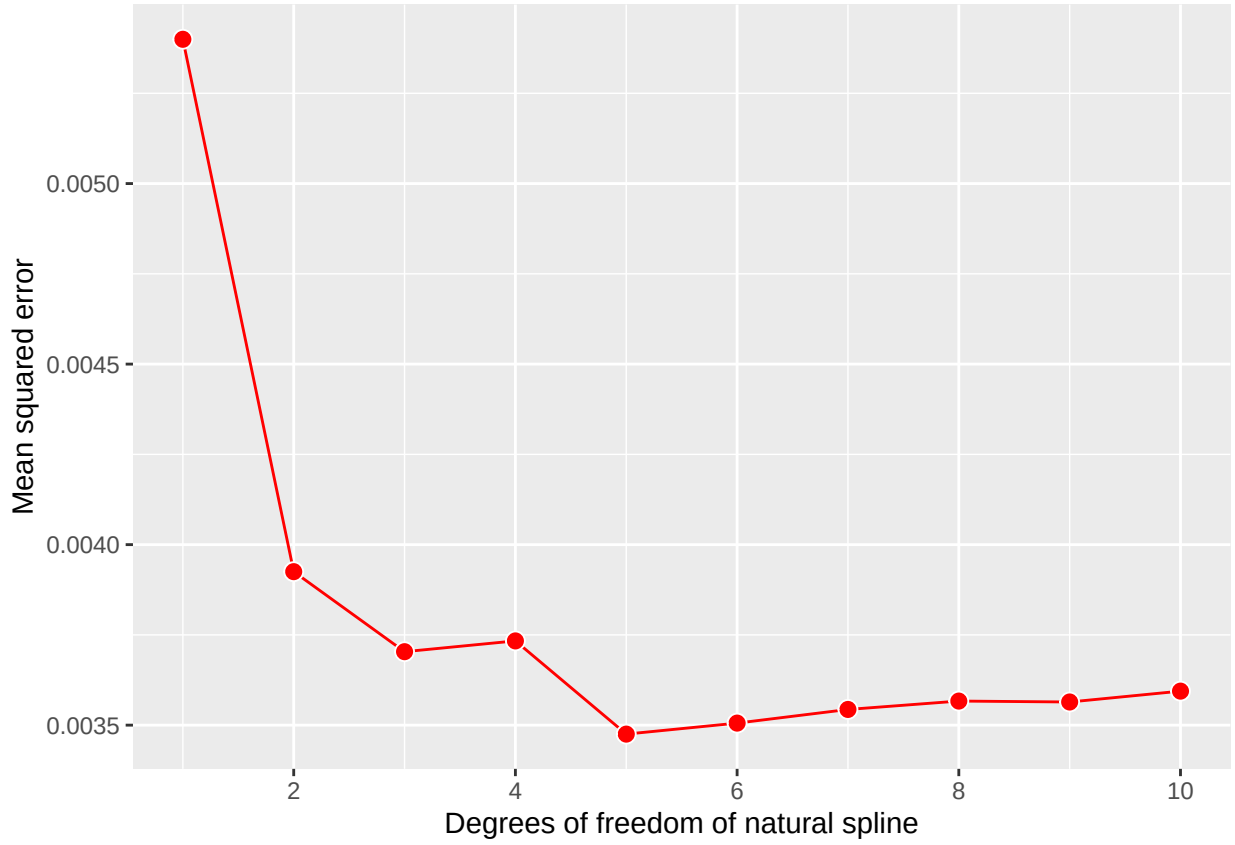



```
## tuneamos y corss validamos los parametros knots y grado


rec_spline <- recipe(nox ~ dis, data = train_data) %>%
   step_bs(dis, degree = tune(), deg_free = tune())


  spline_wf = workflow() %>%
    add_model(lm_spec) %>%
    add_recipe(rec_spline)



boston_folds = vfold_cv(train_data, v = 5)

bs_df_grid <- bind_rows(
  tibble(deg_free = 1:3, degree = 1:3),
  tibble(deg_free = 4:10, degree = 3)
)




tune_res <- tune_grid(
```

```
   spline_wf,
  resamples = boston_folds,
  grid = bs_df_grid)
```

```
## > A | warning: some 'x' values beyond boundary knots may cause ill-conditioned bases

## There were issues with some computations   A: x1

## There were issues with some computations   A: x10

##
```

```
p1 <- collect_metrics(  tune_res ) %>%
  filter(.metric == "rmse") %>%
  mutate(mse = mean^2) %>%
  ggplot(aes(x = deg_free, y = mse)) +
  geom_line(color = "red") +
  geom_point(fill = "red", color = "white", shape = 21, size = 3) +
  scale_x_continuous("Degrees of freedom of natural spline",
                     breaks = seq(2, 10, 2))  +
  labs(y = "Mean squared error")


p1
```




Se puede ver que en el grado 3 o 4 de la regresion de splines, el modelo ajusta bien ya que tiene un $R^2$ grande y un RSME chico.

10) This question relates to the College data set

a) Split the data into a **training** set and **test set**. Using out of state tuition as the response and the other variables as the predictors, perform **forward stepwise** selection on the training set in order to identify a satisfactory model that uses just a subset of the predictors.

10)

a) Separo los datos en conjunto de entrenamiento y testeo

```
College = tibble(College)
split = initial_split(College, prop = 0.75)

College_train = training(split)
college_test = testing(split)
```

Como no esta desarrollado stepwise en tidymodels, selecciono las variables segun Lasso ya que es un modelo de regularizacion y seleccion. Segun Lasso, nos quedamos con 4 variables, Private_Yes, Room board, Expend, Accept

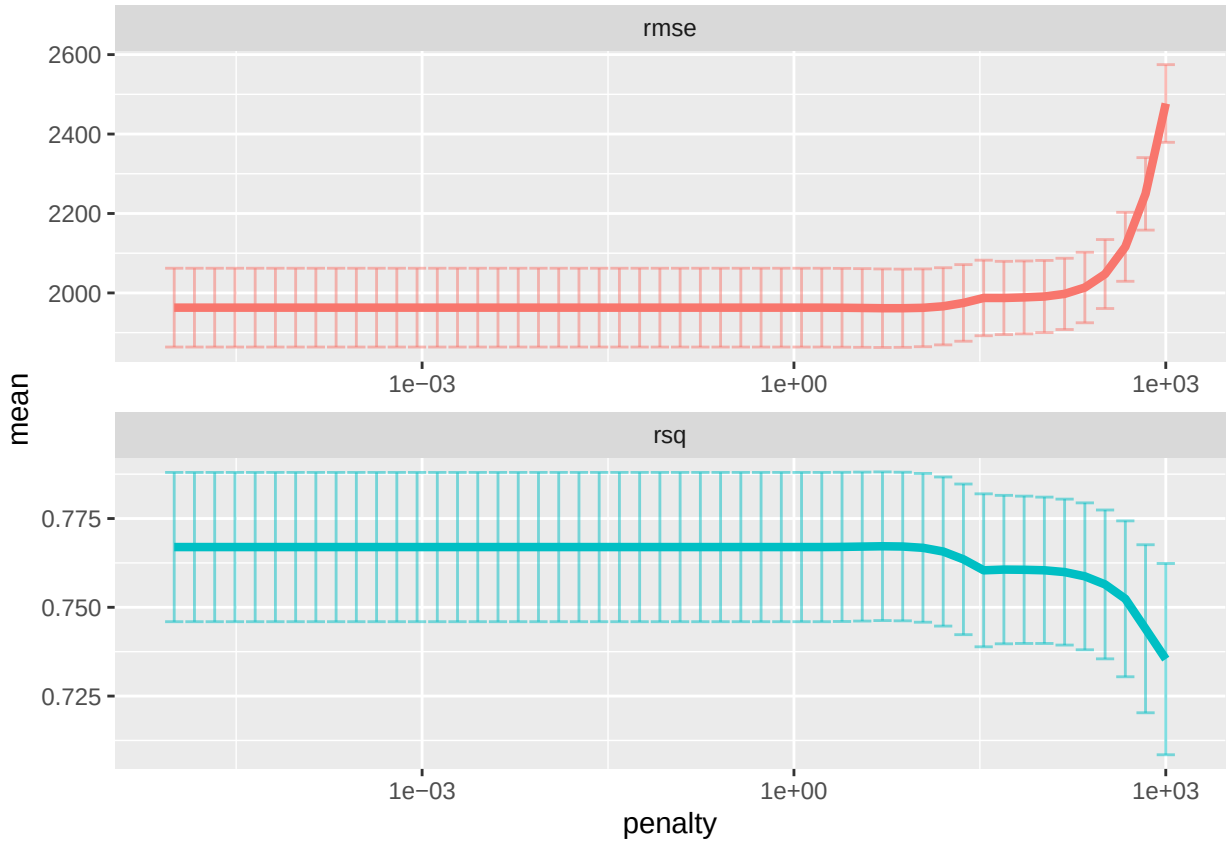
```
rec_mod <-  recipe(Outstate ~ ., data = College_train) %>% step_dummy(Private) %>%
  step_normalize(all_numeric(), -all_outcomes())




lm_spec <- linear_reg() %>%
  set_mode("regression") %>%
  set_engine("lm")



lasso_spec <- linear_reg(penalty = tune(), mixture = 1) |> ## lasso
  set_engine("glmnet")



lasso_workflow <- workflow() %>%

  add_recipe(rec_mod ) %>%
  add_model(lasso_spec)

wf <- workflow() %>%
  add_recipe(rec_mod )






lasso_grid <- grid_regular(penalty(range = c(-5, 3)), levels = 50)

college_cv <- rsample::vfold_cv(College_train, v = 5)
```

```
lasso_tune <- tune_grid(
  lasso_workflow,
  resamples = college_cv,
  grid = lasso_grid
)

lasso_tune |>
  collect_metrics()
```

```
## # A tibble: 100 x 7
##      penalty .metric .estimator     mean     n std_err .config
##        <dbl> <chr>   <chr>         <dbl> <int>   <dbl> <chr>
##  1 0.00001   rmse    standard   1963.        5 99.2    Preprocessor1_Model01
##  2 0.00001   rsq     standard      0.767     5  0.0210 Preprocessor1_Model01
##  3 0.0000146 rmse    standard   1963.        5 99.2    Preprocessor1_Model02
##  4 0.0000146 rsq     standard      0.767     5  0.0210 Preprocessor1_Model02
##  5 0.0000212 rmse    standard   1963.        5 99.2    Preprocessor1_Model03
##  6 0.0000212 rsq     standard      0.767     5  0.0210 Preprocessor1_Model03
##  7 0.0000309 rmse    standard   1963.        5 99.2    Preprocessor1_Model04
##  8 0.0000309 rsq     standard      0.767     5  0.0210 Preprocessor1_Model04
##  9 0.0000450 rmse    standard   1963.        5 99.2    Preprocessor1_Model05
## 10 0.0000450 rsq     standard      0.767     5  0.0210 Preprocessor1_Model05
## # i 90 more rows
```

```
lasso_tune |>
  collect_metrics() |>
  ggplot(aes(penalty, mean, color = .metric)) +
  geom_errorbar(aes(
    ymin = mean - std_err,
    ymax = mean + std_err
  ),
  alpha = 0.5
  ) +
  geom_line(size = 1.5) +
  facet_wrap(~.metric, scales = "free", nrow = 2) +
  scale_x_log10() +
  theme(legend.position = "none")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
lowest_rmse <- lasso_tune |>
  select_best(metric = "rmse")

final_lasso <- finalize_workflow(wf%>%  add_model(lasso_spec), lowest_rmse)

library(vip)
```

```
##
## Attaching package: 'vip'
```

```
## The following object is masked from 'package:utils':
##
##     vi
```

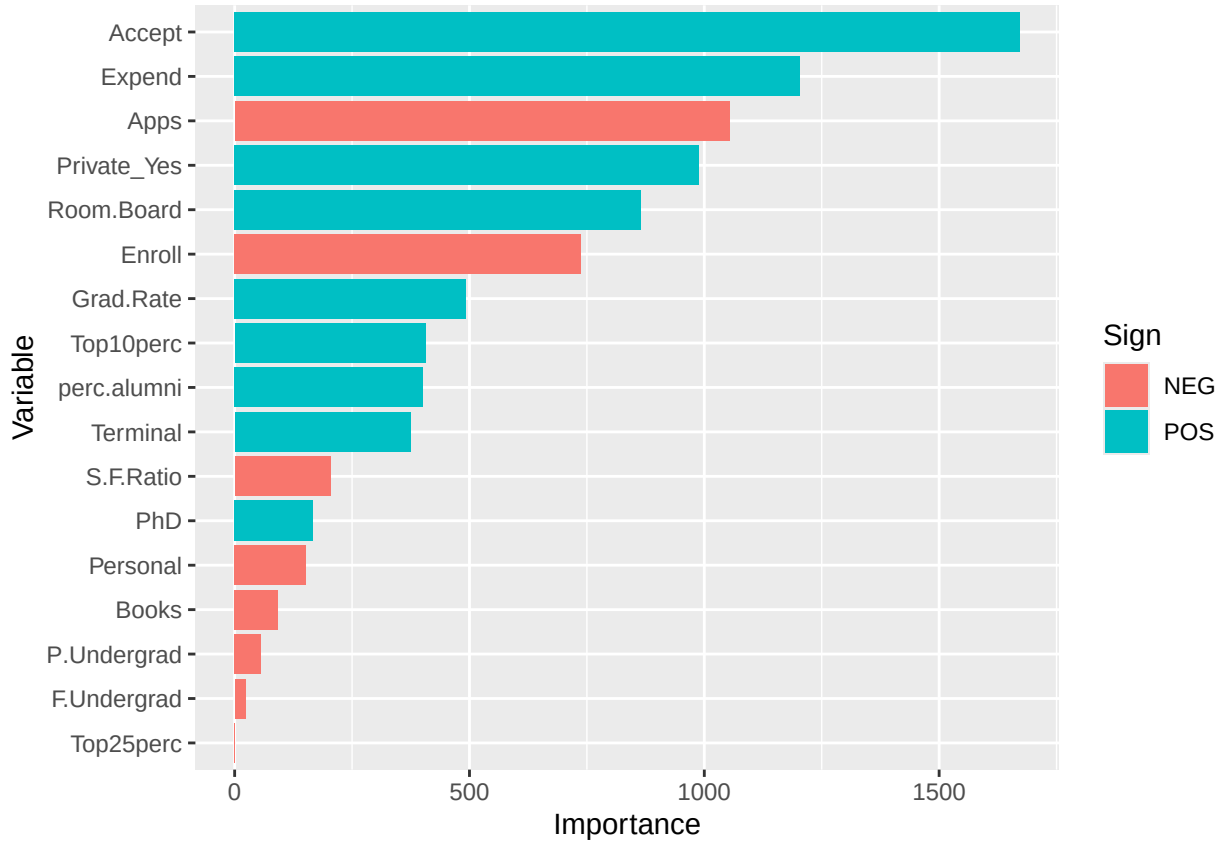
```
lasso_final_fit= fit(final_lasso, data = College_train)

augment(lasso_final_fit, new_data = college_test) %>%
  rsq(truth = Outstate, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.718
```

```
library(forcats)
final_lasso %>% fit(College_train) %>% pull_workflow_fit() %>%  vi(lambda = lowest_rmse$penalty) %>%  mu
```

```
## Warning: `pull_workflow_fit()` was deprecated in workflows 0.2.3.
```

```
## i Please use `extract_fit_parsnip()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




b) Fit a GAM on the training data, using out of state tuition as the response and the features selected in the previous step as the predictors. Plot the results, and explain your findings.

```
##( saco  private porque me lo sigue tirando como factor :'(
rec_gam_ns = recipe(Outstate ~ F.Undergrad +   Room.Board + Expend + Accept, data = College_train)  %>%
 step_ns(Room.Board, deg_free=3) %>%
 step_ns(Expend, deg_free=2) %>%
 step_ns(Accept, deg_free=2)   %>%
  step_ns(F.Undergrad, deg_free=2)



 gam_wf_ns <- workflow() %>%
  add_model(lm_spec) %>%
  add_recipe(rec_gam_ns)

gam_fit_ns <- fit(gam_wf_ns, data =College_train )

fit_gam <- extract_fit_engine(gam_fit_ns)
summary(fit_gam)
```

```
##
```

```
## Call:
## stats::lm(formula = ..y ~ ., data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6901.0 -1286.7    84.6  1416.2  8059.8
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3621.9      678.4   5.339 1.35e-07 ***
## Room.Board_ns_1     3292.4      493.3   6.674 5.87e-11 ***
## Room.Board_ns_2     6746.6     1543.7   4.371 1.47e-05 ***
## Room.Board_ns_3     4785.9      940.0   5.091 4.84e-07 ***
## Expend_ns_1        20361.5      923.4  22.052  < 2e-16 ***
## Expend_ns_2         6557.3     1101.3   5.954 4.57e-09 ***
## Accept_ns_1        10821.6     2060.6   5.252 2.13e-07 ***
## Accept_ns_2         -363.0     2083.1  -0.174    0.862
## F.Undergrad_ns_1 -13022.1     1477.7  -8.812  < 2e-16 ***
## F.Undergrad_ns_2  -5347.8     1239.3  -4.315 1.88e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2075 on 572 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7365
## F-statistic: 181.5 on 9 and 572 DF,  p-value: < 2.2e-16
```

Se puede observar que todas las covariables son individualmente significativas, por ejemplo podemos ver que se estima que ante un aumento en Full undergrad, el costo de la residencial (la variable dependiente Out of state), en promedio disminuye 14537, manteniendo todo lo demas constante

Tambien , se estima que el costo de la residencial aumente en promedio ante un aumento en Room Board (cuanto mas camas y costos tenga), manteniendo todo lo demas constante.

c) Evaluate the model obtained on the **test set**, and explain the results obtained.

Reporto el ECM en el conjunto de test

```
regression_lines <- bind_cols(
  augment(gam_fit_ns , new_data = college_test),
  predict(gam_fit_ns , new_data = college_test, type = "conf_int"))


mean((college_test$Outstate -regression_lines$.pred )^2)# MSE
```

```
## [1] 4812199
```

d) For which variables, if any, is there evidence of a non - linear relationship with the response?

```
library(gratia)
```

```
##
## Attaching package: 'gratia'
```

```
## The following object is masked from 'package:dials':
##
##     penalty
```

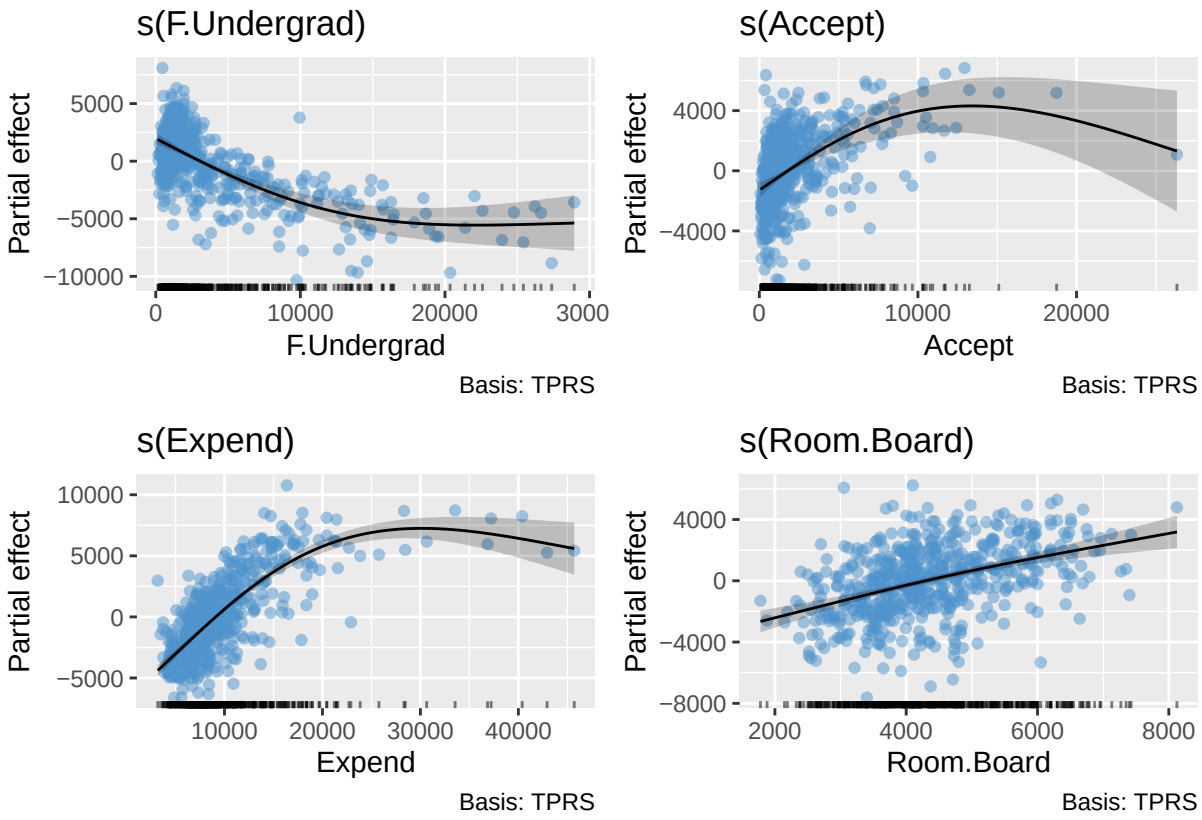
```
rec_gam_ns = recipe(Outstate ~ F.Undergrad +   Room.Board + Expend + Accept, data = College_train)  %>%
 step_ns(Room.Board, deg_free=3) %>%
```

```
 step_ns(Expend, deg_free=2) %>%
 step_ns(Accept, deg_free=2)   %>%
  step_ns(F.Undergrad, deg_free=2)
```

```
rec_gam2 = recipes::recipe(Outstate ~ F.Undergrad + Room.Board + Expend + Accept , data = College_train)

gam_spec = gen_additive_mod() %>%  set_engine("mgcv") %>%  set_mode("regression")

gam_wf = workflow() %>%  add_recipe(rec_gam2) %>%  add_model(gam_spec, formula = Outstate ~ s(F.Undergra

gam_fit = fit(gam_wf,data=College_train)
```

```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): basis dimension, k, increased
```

```
fit_gam = extract_fit_engine(gam_fit)

draw(fit_gam,residuals = T)
```




No parece haber linealdad entre la variable explicativa Expend y la variable dependiente outstate.