REGRESION POLINOMIAL

Contenido

Datos.

Una marca de coches quiere generar un modelo de regresión que permita predecir el consumo de combustible (mpg) en función de la potencia del motor (horsepower).

```
#Regresion polinomial
library(ISLR)
attach(Auto)
```

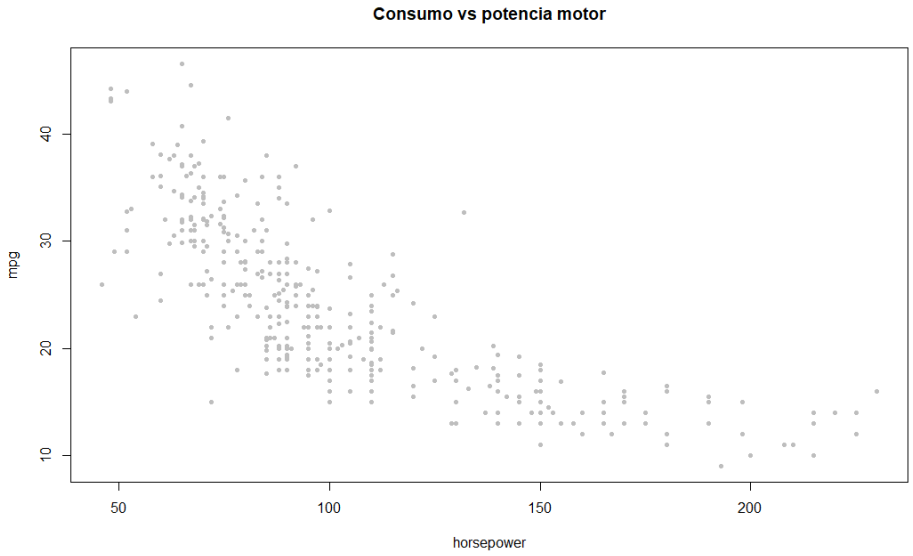
```
head (Auto)
```

```
##   mpg cylinders displacement horsepower weight acceleration year origi
n
## 1  18         8          307        130   3504         12.0   70
1
## 2  15         8          350        165   3693         11.5   70
1
## 3  18         8          318        150   3436         11.0   70
1
## 4  16         8          304        150   3433         12.0   70
1
## 5  17         8          302        140   3449         10.5   70
1
## 6  15         8          429        198   4341         10.0   70
1
##                       name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

Tomaremos las variables mpg (y) y horsepower (x)

La potencia del motor influye en las millas por galón

```
plot(x = horsepower, y = mpg, main = "Consumo vs potencia motor", pch = 20,
    col = "grey")
```



La representación gráfica de los datos muestra una fuerte asociación entre el consumo y la potencia del motor. La distribución de las observaciones apunta a que la relación entre ambas variables tiene cierta curvatura, por lo que un modelo lineal no puede captarla por completo. A pesar de ello ajustemos a un modelo lineal para observar el comportamiento del modelo.

Modelo lineal.

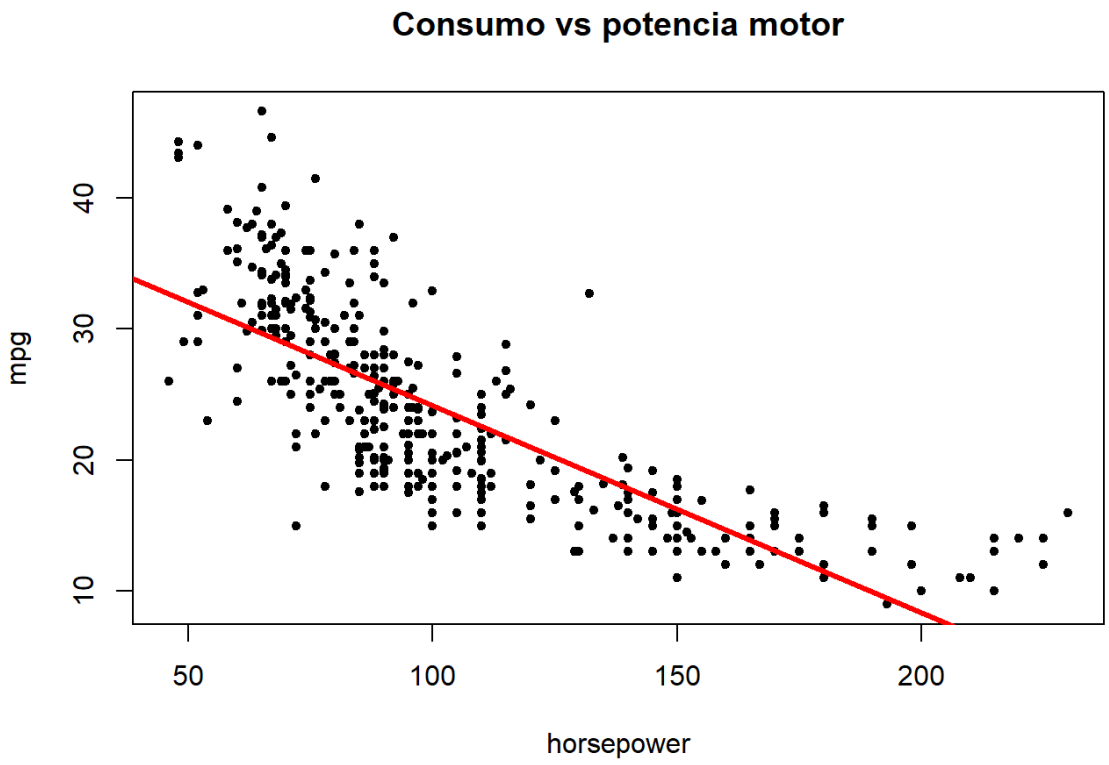
```
modelo_lineal <- lm(formula = mpg ~ horsepower, data = Auto)
summary(modelo_lineal)
```

```
## Call:
## lm(formula = mpg ~ horsepower, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate   Std. Error    t value Pr(>|t|)
## (Intercept)  39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4.906 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

como puede observarse, tanto el intercepto como la variable horsepower presentan significancia, a su vez el coeficiente de determinación esta al rededor del 61% y el modelo en su conjunto utilizando la técnica del análisis de varianza es bueno $p(0.000) < \alpha(0.05)$ esto, sin considerar los supuestos del modelo lineal. Graficando.

```
plot(x = horsepower, y = mpg, main = "Consumo vs potencia motor", pch = 20,
    col = "grey")
abline(modelo_lineal, lwd = 3, col = "red")
```



Puede mejorarse el ajuste, debido a que se observa una tendencia de los datos no lineal.

```
# predecir valores para X
```

```
predict_value <- predict(modelo_lineal,Auto)
head(predict_value)
```

```
##         1         2         3         4         5         6
## 19.416046 13.891480 16.259151 16.259151 17.837598  8.682604
```

```
# comparando
b = cbind(Auto$mpg, predict_value)
head(b)
```

```
##      predict_value
## 1 18     19.416046
## 2 15     13.891480
## 3 18     16.259151
## 4 16     16.259151
## 5 17     17.837598
## 6 15      8.682604
```

```
# Cálculo del error cuadrático medio RMSE:
# RMSE: raiz del error cuadratico medio
error = predict_value - Auto$horsepower
head(error)
```

```
##         1         2         3         4         5         6
## -110.5840 -151.1085 -133.7408 -133.7408 -122.1624 -189.3174
```

```
# Cálculo del error cuadrático medio RMSE:
sqrt(mean(error^2))
```

```
## [1] 92.44422
```

```
MSE <- mean(modelo_lineal$residuals^2)
MSE
```

```
## [1] 23.94366
```

```
RMSE <- sqrt(MSE)
RMSE
```

```
## [1] 4.893226
```

```
# prediccion de nuevos valores
y_pred_lin = predict(modelo_lineal, newdata = data.frame(horsepower=6.5))
y_pred_lin
```

```
##        1
## 38.90987
```

Modelo cuadrático

Una forma de incorporar asociaciones no lineales a un modelo lineal es mediante transformaciones de los predictores incluidos en el modelo, por ejemplo, elevándolos a distintas potencias. En este caso, el tipo de curvatura es de tipo cuadrática, por lo que un polinomio de segundo grado podría mejorar el modelo.

En R se pueden generar modelos de regresión polinómica de diferentes formas:

- Identificando cada elemento del polinomio: modelo_pol2 <- lm(formula = mpg ~ horsepower + I(horsepower^2), data = Auto) El uso de I() es necesario ya que el símbolo ^ tiene otra función dentro de las formula de R.

- Con la función poly(): lm(formula = mpg ~ poly(horsepower, 2), data = Auto)

```
modelo_cuadratico <- lm(formula = mpg ~ poly(horsepower, 2), data = Auto)
summary(modelo_cuadratico)
```

```
## Call:
## lm(formula = mpg ~ poly(horsepower, 2), data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.7135  -2.5943  -0.0859   2.2868  15.8961
##
## Coefficients:
##                               Estimate   Std. Error  t value     Pr(>|t|)
## (Intercept)                   23.4459     0.2209      106.13   <2e-16 ***
## poly(horsepower, 2)1 -120.1377    4.3739      -27.47   <2e-16 ***
## poly(horsepower, 2)2   44.0895    4.3739       10.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.374 on 389 degrees of freedom
## Multiple R-squared:  0.6876, Adjusted R-squared:  0.686
## F-statistic:   428 on 2 and 389 DF,  p-value: < 2.2e-16
```

El valor R^2 del modelo cuadrático (0.6876) es mayor que el obtenido con el modelo lineal simple (0.6059) y el *p-value* del término cuadrático es altamente significativo. Se puede concluir que el modelo cuadrático recoge mejor la verdadera relación entre el consumo de los vehículos y la potencia de su motor.

modelo: $Y = \beta_o + \beta_o x_1 + \beta_o x_1^2$

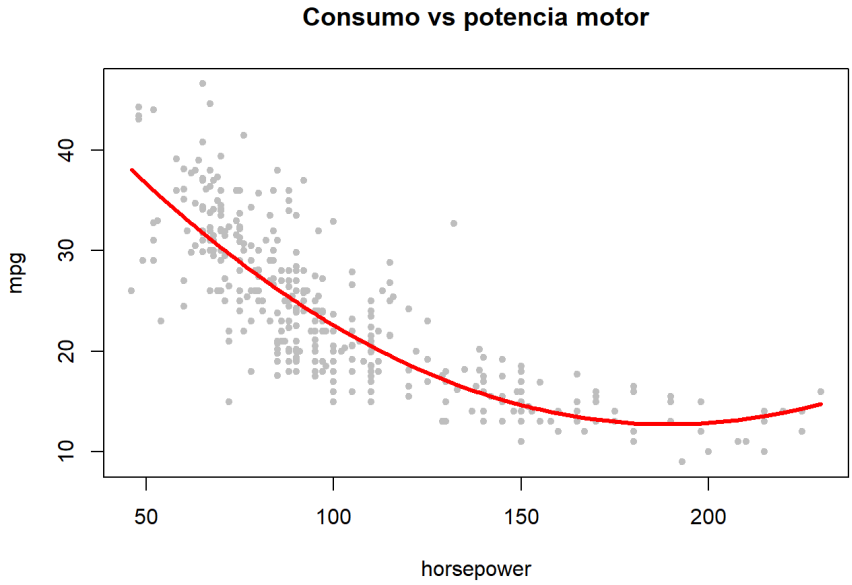
$$Y = 23.4459 - 120.1377x_1 + 44.0895x_1^2$$

Ajuste del modelo (Gráficamente)

```
par(mfrow = c(1, 1))
```

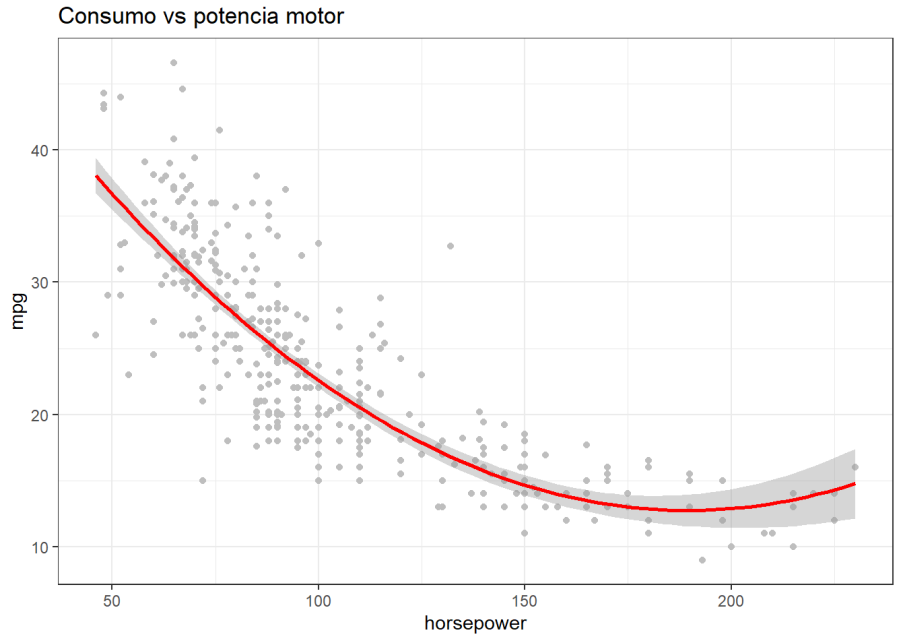
```
plot(x = horsepower, y = mpg, main = "Consumo vs potencia motor", pch = 20,
    col = "grey")
puntos_interpolados <- seq(from = min(horsepower), to = max(horsepower), by = 1)
prediccion <- predict(object = modelo_cuadratico,
                      newdata = data.frame(horsepower = horsepower))
```

```
lines(sort(horsepower), prediccion[order(horsepower)], col = "red", lwd = 3)
```



Se observa un mejor ajuste.

```
ggplot(Auto, aes(x = horsepower, y = mpg)) +
  geom_point(colour = "grey") +
  stat_smooth(method = "lm", formula = y ~ poly(x, 2), colour = "red") +
  labs(title = "Consumo vs potencia motor") +
  theme_bw()
```



```r
# predecir valores para X
predict_valuec <- predict(modelo_cuadratico,Auto)
head(predict_valuec)
```

```
##        1        2        3        4        5        6
## 17.09151 13.48016 14.65872 14.65872 15.75206 12.83649
```

```r
# Cálculo del error cuadrático medio RMSE:
# RMSE: raiz del error cuadratico medio
error = predict_valuec - Auto$horsepower
head(error)
```

```
##         1         2         3         4         5         6
## -112.9085 -151.5198 -135.3413 -135.3413 -124.2479 -185.1635
```

```r
# Cálculo del error cuadrático medio RMSE:
sqrt(mean(error^2))
```

```
## [1] 92.47104
```

```r
MSE <- mean(modelo_cuadratico$residuals^2)
MSE
```

```
## [1] 18.98477
```

```r
RMSE <- sqrt(MSE)
RMSE
```

```
## [1] 4.357151
```

Prediciendo nuevos valores

```
y_pred_cuad = predict(modelo_cuadratico, newdata = data.frame(horsepower=
6.5))

y_pred_cuad
```

```
##        1
## 53.92186
```

Al tratarse de modelos anidados, es posible emplear un ANOVA para contrastar la hipótesis nula de que ambos modelos se ajustan a los datos igual de bien.

```
anova(modelo_lineal, modelo_cuadratico)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ horsepower
## Model 2: mpg ~ poly(horsepower, 2)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    390 9385.9
## 2    389 7442.0  1    1943.9 101.61 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

existe diferencia significativa $p(0.000) < \alpha(0.05)$. La elección del grado del polinomio influye directamente en la flexibilidad del modelo. Cuanto mayor es el grado del polinomio más se ajusta el modelo a las observaciones, un polinomio de grado $n^o observaciones - 1$ pasa por todos los puntos. Por lo tanto, es importante no excederse en el grado del polinomio para no causar problemas de *overfitting* (no suelen recomendarse grados superiores a 3-4).

Existen varias estrategias para identificar el grado óptimo del polinomio:

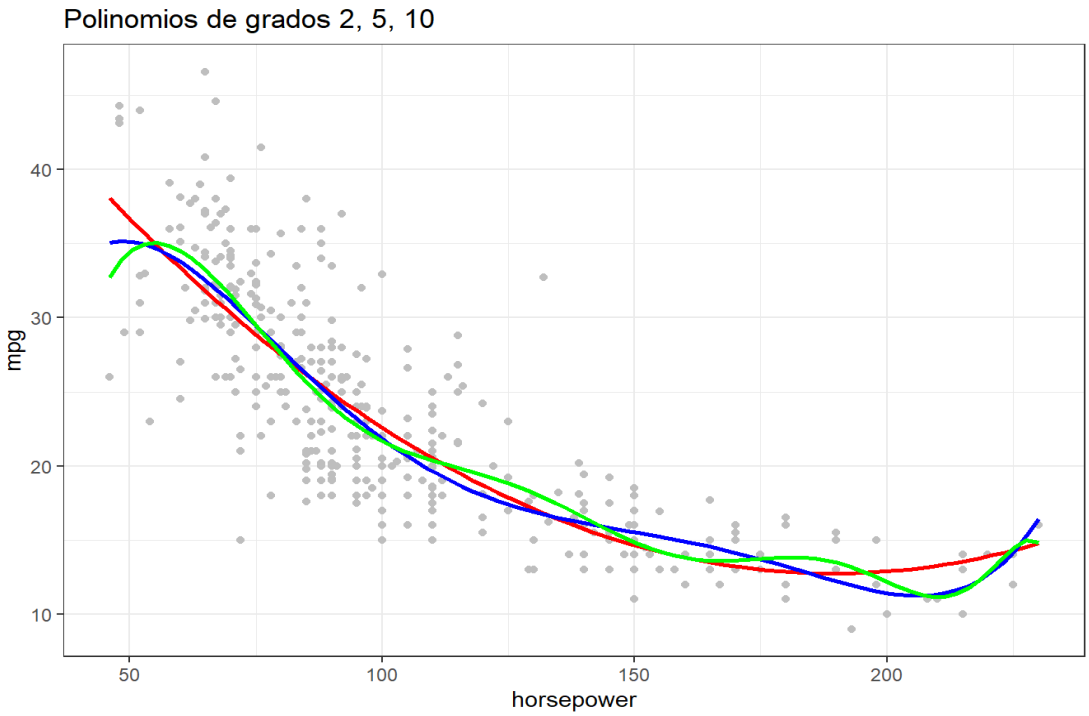
- Incrementar secuencialmente el orden del polinomio hasta que la nueva incorporación no sea significativa.

- Iniciar el proceso con un polinomio de grado alto e ir eliminando secuencialmente, de mayor a menor, los términos no significativos.
- Emplear un ANOVA para comparar cada nuevo modelo con el de orden inferior y determinar si la mejora es significativa. Este proceso es equivalente al primero. Ver ejemplo en Regresión Polinomial: incorporar no-linealidad a los modelos lineales.
- Emplear *cross-validation*. En el capítulo Validación de modelos de regresión: Cross-validation, OneLeaveOut, Bootstrap se explica cómo emplear la validación-cruzada para identificar el grado adecuado.

Como norma general, si se incluye en el modelo un término polinómico, se tienen que incluir también como predictores los órdenes inferiores de esas variables, aunque no sean significativos.

Observando polinomios de grado 2, 5 y 10

```
library(ggplot2)
ggplot(Auto, aes(x = horsepower, y = mpg)) +
geom_point(colour = "grey") + stat_smooth(method = "lm", formula = y ~ poly(x, 2),
colour = "red", se = FALSE) + stat_smooth(method = "lm", formula = y ~ poly(x, 5),
colour = "blue", se = FALSE) + stat_smooth(method = "lm",formula = y ~ poly(x, 10),
colour = "green", se= FALSE) + labs(title = "Polinomios de grados 2, 5, 10") +
theme_bw()
```



Polinomio de grado 5.

```
# modelo grado 5
modelo5 <- lm(formula = mpg ~ poly(horsepower, 5), data = Auto)
summary(modelo5)
```

```
##
## Call:
## lm(formula = mpg ~ poly(horsepower, 5), data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.4326  -2.5285  -0.2925   2.1750  15.9730
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              23.4459     0.2185 107.308  < 2e-16 ***
## poly(horsepower, 5)1 -120.1377     4.3259 -27.772  < 2e-16 ***
## poly(horsepower, 5)2   44.0895     4.3259  10.192  < 2e-16 ***
## poly(horsepower, 5)3   -3.9488     4.3259  -0.913  0.36190
## poly(horsepower, 5)4   -5.1878     4.3259  -1.199  0.23117
## poly(horsepower, 5)5   13.2722     4.3259   3.068  0.00231 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.326 on 386 degrees of freedom
## Multiple R-squared:  0.6967, Adjusted R-squared:  0.6928
## F-statistic: 177.4 on 5 and 386 DF,  p-value: < 2.2e-16
```

```
# predecir valores para 5
predict_value5 <- predict(modelo5,Auto)
head(predict_value5)
```

```
##        1        2        3        4        5        6
## 16.90200 14.54761 15.52854 15.52854 16.13579 11.55326
```

```
# Cálculo del error cuadrático medio RMSE:
# RMSE: raiz del error cuadratico medio
error = predict_value5 - Auto$mpg
head(error)
```

```
##          1          2          3          4          5          6
## -1.0979951 -0.4523907 -2.4714623 -0.4714623 -0.8642136 -3.4467396
```

```
# Cálculo del error cuadrático medio RMSE:
sqrt(mean(error^2))
```

```
## [1] 4.292665
```

Polinomio de grado 10

```
# modelo grado 10
modelo10 <- lm(formula = mpg ~ poly(horsepower, 10), data = Auto)
summary(modelo10)
```

```
## Call:
## lm(formula = mpg ~ poly(horsepower, 10), data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.7081  -2.5904  -0.1922   2.2859  14.8338
##
## Coefficients:
```

```
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               23.4459     0.2174 107.840   <2e-16 ***
## poly(horsepower, 10)1   -120.1377     4.3046 -27.909   <2e-16 ***
## poly(horsepower, 10)2     44.0895     4.3046  10.242   <2e-16 ***
## poly(horsepower, 10)3     -3.9488     4.3046  -0.917   0.3595
## poly(horsepower, 10)4     -5.1878     4.3046  -1.205   0.2289
## poly(horsepower, 10)5     13.2722     4.3046   3.083   0.0022 **
## poly(horsepower, 10)6     -8.5462     4.3046  -1.985   0.0478 *
## poly(horsepower, 10)7      7.9806     4.3046   1.854   0.0645 .
## poly(horsepower, 10)8      2.1727     4.3046   0.505   0.6140
## poly(horsepower, 10)9     -3.9182     4.3046  -0.910   0.3633
## poly(horsepower, 10)10    -2.6146     4.3046  -0.607   0.5440
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.305 on 381 degrees of freedom
## Multiple R-squared:  0.7036, Adjusted R-squared:  0.6958
## F-statistic: 90.45 on 10 and 381 DF,  p-value: < 2.2e-16
```

```r
# predecir valores para 10
predict_value10 <- predict(modelo10,Auto)
head(predict_value10)
```

```
##        1        2        3        4        5        6
## 18.16751 13.61005 14.86473 14.86473 16.52326 12.49179
```

```r
# Cálculo del error cuadrático medio RMSE:
# RMSE: raiz del error cuadratico medio

error = predict_value10 - Auto$mpg
head(error)
```

```
##           1           2           3           4           5           6
##  0.1675140 -1.3899507 -3.1352679 -1.1352679 -0.4767431 -2.5082121
```

```
# Cálculo del error cuadrático medio RMSE:
sqrt(mean(error^2))
```

```
## [1] 4.243763
```

Comparando resultados

Modelo	R2	RMSE		Pred (6.5)
Lineal	0.6059 (4)	4.893226 (4)	Todos sig	38.90987
Cuadrático	0.6876 (3)	4.357151 (3)	Todos sig	53.92186
Grado 5	0.6967 (2)	4.292665 (2)	2 NS	
Grado 10	0.7036 (1)	4.243763 (1)	6 NS	

A pesar que los polinomios de Grado 10 y 5 tiene los R2 mas altos y RMSE no son los mejores, debido a que tienen valores no significativos. El modelo mas mesurado es el modelo cuadrático debido a que tiene mayor R2 y menor RSME que le lineal.