Práctica 1: Regresión lineal con R (1ra parte)

Maximiliano Voca Montejano
364897
maximiliano.voca@uabc.edu.mx

Ejercicio 1

(a) Genera un data frame compuesto de la informaci´on de un examen para 10 alumnos (hipotéticos):

- Nombre.
- Calificación.
- Número de veces que ha presentado el examen (Intentos).
- Aprobado o no.

- definimos un vector para cada columna de lo que será el dataframe

In [155…
```
Nombre <- c("Max", "Samuel", "Michel", "Alondra", "Asael",
            "Valeria", "Raul", "Paola", "Carlos", "Sergio")
```

In [156…
```
Calificacion <- c(47, 92, 85, 75, 52,
                  85, 55, 95, 97, 65)
```

In [157…
```
Intentos <- c(1, 2, 2, 3, 1,
              1, 2, 3, 1, 1)
```

- hacemos uso de un condicional para simplificar el arreglo de aprobados

In [158…
```
Aprobado <- c(Calificacion>60)

print(Aprobado)
```

```
[1] FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

- definimos el dataframe, instertando como parametros los vectores columna anteriormente definidos, especificando que no convierta automaticamente los strings a factores, por los nombres

In [159…
```
df <- data.frame(Nombre, Calificacion, Intentos, Aprobado
                 ,stringsAsFactors=FALSE)
                 #,stringsAsFactors=FALSE

df
```

Nombre	Calificacion	Intentos	Aprobado
Max	47	1	FALSE
Samuel	92	2	TRUE
Michel	85	2	TRUE
Alondra	75	3	TRUE
Asael	52	1	FALSE
Valeria	85	1	TRUE
Raul	55	2	FALSE
Paola	95	3	TRUE
Carlos	97	1	TRUE
Sergio	65	1	TRUE

In [160…
```
str(df)
```

```
'data.frame':   10 obs. of  4 variables:
 $ Nombre      : chr  "Max" "Samuel" "Michel" "Alondra" ...
 $ Calificacion: num  47 92 85 75 52 85 55 95 97 65
 $ Intentos    : num  1 2 2 3 1 1 2 3 1 1
 $ Aprobado    : logi  FALSE TRUE TRUE TRUE FALSE TRUE ...
```

In [161…
```
# Convertir la columna 'Aprobado' a factores
df$Aprobado <- as.factor(df$Aprobado)
```

In [162…
```
str(df)
```

```
'data.frame':   10 obs. of  4 variables:
 $ Nombre      : chr  "Max" "Samuel" "Michel" "Alondra" ...
 $ Calificacion: num  47 92 85 75 52 85 55 95 97 65
 $ Intentos    : num  1 2 2 3 1 1 2 3 1 1
 $ Aprobado    : Factor w/ 2 levels "FALSE","TRUE": 1 2 2 2 1 2 1 2 2 2
```

(b) Imprima el renglón 3 y 5 con la información de la primera y tercer columna.

In [163…
```
df[c(3,5), c(1,3)]
```

	Nombre	Intentos
3	Michel	2
5	Asael	1

(c) Añada dos nuevos renglones con la información de dos alumnos.

definimos un dataframe provisional, para despues añadirlo al original

In [164…
```
nuevos_renglones <- data.frame(
    Nombre= c("Fernando", "Isaac"),
    Calificacion= c(65, 78),
    Intentos= c(2, 1),
    Aprobado= c(TRUE, TRUE)
)

df <- rbind(df, nuevos_renglones)

df
```

Nombre	Calificacion	Intentos	Aprobado
Max	47	1	FALSE
Samuel	92	2	TRUE
Michel	85	2	TRUE
Alondra	75	3	TRUE
Asael	52	1	FALSE
Valeria	85	1	TRUE
Raul	55	2	FALSE
Paola	95	3	TRUE
Carlos	97	1	TRUE
Sergio	65	1	TRUE
Fernando	65	2	TRUE
Isaac	78	1	TRUE

(d) Guarde la información de la hoja de datos en un archivo de datos con extensión .csv

el parametro row.names=FALSE es importante porque si no toma la primer columna como los nombres de las filas

In [165…
```
write.csv(df, file = "practica1.csv", row.names=FALSE)
```

(e) Abra el archivo y asigne la información a una nueva variable.

In [166…
```
df_2 <- read.csv(file = "practica1.csv", header = TRUE,  sep = ",", row.names = NULL,  stringsAsFactors = FALSE)

df_2
```

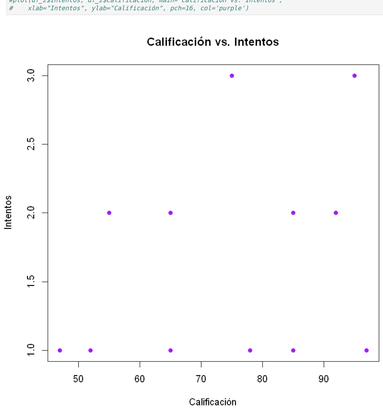
Nombre	Calificacion	Intentos	Aprobado
Max	47	1	FALSE
Samuel	92	2	TRUE
Michel	85	2	TRUE
Alondra	75	3	TRUE
Asael	52	1	FALSE
Valeria	85	1	TRUE
Raul	55	2	FALSE
Paola	95	3	TRUE
Carlos	97	1	TRUE
Sergio	65	1	TRUE
Fernando	65	2	TRUE
Isaac	78	1	TRUE

In [167…
```
df_2$Calificacion
```

```
 1. 47
 2. 92
 3. 85
 4. 75
 5. 52
 6. 85
 7. 55
 8. 95
 9. 97
10. 65
11. 65
12. 78
```

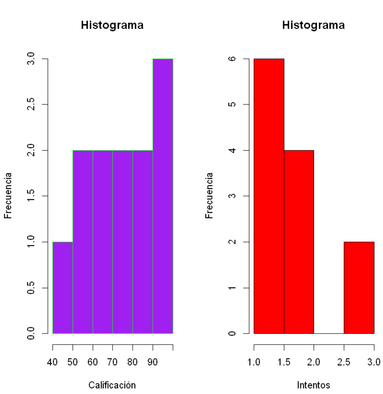
(f) Grafique Calificación vs. Intentos.

In [222…
```
plot(df_2$Calificacion, df_2$Intentos, main="Calificación vs. Intentos",
     xlab="Calificación", ylab="Intentos", pch=16, col='purple')

#plot(df_2$Intentos, df_2$Calificacion, main="Calificación vs. Intentos",
#     xlab="Intentos", ylab="Calificación", pch=16, col='purple')
```



(g) Grafique el histograma de Calificación e Intentos en un lienzo con dos subfiguras.

In [216…
```
# Configurar el diseño del lienzo
par(mfrow = c(1, 2))

hist(df_2$Calificacion, main = "Histograma", xlab = "Calificación", ylab = "Frecuencia", col = "purple", border = "green")
hist(df_2$Intentos, main = "Histograma", xlab = "Intentos", ylab = "Frecuencia", col = "red", border = "black")

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



(h) Modifique los valores por defecto de los gráficos (colores, tipo de línea, dimensiones, etc.)

In [ ]:
```
```

Práctica 1: Regresión lineal con R (2da parte)

Maximiliano Vaca Montejano
364897
maximiliano.vaca@uabc.edu.mx

Ejercicio 2

Deseamos investigar la relación entre la distancia de frenado de un auto y la velocidad al momento en que el conductor se encuentra con un señalamiento de alto. Se supone que el tiempo de reacción para que el conductor aplique los frenos es aproximadamente fijo, por lo que el auto viajará una distancia proporcional a su velocidad antes de comenzar a parar. Por otro lado, la energía cinética es proporcional al cuadrado de su velocidad, pero los frenos disipan esa energía y **bajan la velocidad a una razón aproximadamente constante por unidad de distancia recorrida.** Por lo que esperamos que una vez que se aplquen los frenos, el auto viaje una distancia proporcional al cuadrado de su velocidad inicial antes de parar completamente.
Dada la información anterior, un modelo inicial es

$$dist_i = \beta_0 + \beta_1 speed_i + \beta_2 speed_i^2 + \epsilon$$

(a) Ajuste un modelo comenzando con esta propuesta inicial y seleccione el que considere más adecuado.
(b) Para su modelo óptimo estime el tiempo promedio que le toma al conductor aplicar los frenos (hay 5280 pies en una milla).
(c) Grafique los datos y la estimación de cada modelo en una sola figura
(d) Nota: los datos se pueden acceder a través del comando: *data(cars)*

- investigamos el sistema en el que estan medidos los datos, para despues pasarlos al sistema internacional y que sean mas comprensibles

In [221]:
```
data(cars)

# añadimos una columna con las velocidades en metros/segundos
cars$speed_mps <- cars$speed * 1609.34 / 3600

# añadimos una columna con las distancias en metros
cars$dist_m <- cars$dist * 0.3048

head(cars)
```

A data.frame: 6 × 4

	speed<dbl>	dist<dbl>	speed_mps<dbl>	dist_m<dbl>
1	4	2	1.788156	0.6096
2	4	10	1.788156	3.0480
3	7	4	3.129272	1.2192
4	7	22	3.129272	6.7056
5	8	16	3.576311	4.8768
6	9	10	4.023350	3.0480

- graficamos los datos para observar su forma y hacernos una idea del modelo adecuado

In [222]:
```
x = cars$speed_mps ; y = cars$dist_m
plot(x, y)
```

(a) Ajuste un modelo comenzando con esta propuesta inicial y seleccione el que considere más adecuado.

In [223]:
```
library(gamair)
```

- primero intentamos con el modelo cuadratico propuesto al principio

In [224]:
```
cars.mod_1 <- lm(dist_m ~ speed_mps + I(speed_mps^2), data=cars)

summary(cars.mod_1)
```

```
Call:
lm(formula = dist_m ~ speed_mps + I(speed_mps^2), data = cars)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.7537 -2.7993  0.9717  1.4187 11.7623 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.7529     4.5163   0.167   0.868
speed_mps      0.6227     1.3078   0.449   0.656
I(speed_mps^2)  0.1525     0.1006   1.515   0.136

Residual standard error: 4.426 on 47 degrees of freedom
Multiple R-squared:  0.6673,    Adjusted R-squared:  0.6532 
F-statistic: 47.14 on 2 and 47 DF,  p-value: 5.852e-12
```

- podemos acceder así a los coeficientes, lo usaremos mas adelante:

In [225]:
```
cars.mod_1$coefficients[1]
```

(Intercept): 0.752897996888211

se obtienen los valores:
Multiple R-squared: 0.6673, Adjusted R-squared: 0.6532
los cuales no son muy buenos

- Como se observo en la grafica, tiene sentido suponer que la intersección del modelo pase por el origen. Es por esto que optamos por fijar la intersección en el origen con ese parametro "-1" a ver si mejora nuestro modelo

In [226]:
```
cars.mod_2 <- lm(dist_m ~ speed_mps + I(speed_mps^2) -1, data=cars)

summary(cars.mod_2)
```

```
Call:
lm(formula = dist_m ~ speed_mps + I(speed_mps^2) - 1, data = cars)

Residuals:
    Min      1Q  Median      3Q     Max 
-8.7892 -2.7648  0.9608  1.3929 11.7118 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
speed_mps     0.04480    0.30180   2.213  0.03171 *
I(speed_mps^2)  0.13748    0.04482   3.067  0.00355 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.579 on 48 degrees of freedom
Multiple R-squared:  0.9133,    Adjusted R-squared:  0.9097 
F-statistic: 252.8 on 2 and 48 DF,  p-value: < 2.2e-16
```

In [227]:
```
cars.mod_2$coefficients[1]
```

speed_mps: 0.0447952518890453

se obtienen los valores:
Multiple R-squared: 0.9133, Adjusted R-squared: 0.9097
presentando una mejora significativa respecto al modelo anterior

- Los resultados anteriores parecian bastante satisfactorios, pero nos preguntamos que pasaría si quitamos el termino cuadratico del modelo, dado que a simple vista los datos no parecen necesitarlo

In [228]:
```
cars.mod_3 <- lm(dist_m ~ speed_mps -1, data=cars)

summary(cars.mod_3)
```

```
Call:
lm(formula = dist_m ~ speed_mps - 1, data = cars)

Residuals:
   Min     1Q Median     3Q    Max 
-7.981 -1.852 -1.463  1.399 15.295 

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
speed_mps  1.98350    0.09639   20.58   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.956 on 49 degrees of freedom
Multiple R-squared:  0.8963,    Adjusted R-squared:  0.8942 
F-statistic: 423.5 on 1 and 49 DF,  p-value: < 2.2e-16
```

se obtienen los valores:
Multiple R-squared: 0.8963, Adjusted R-squared: 0.8942
presentando una pequeña disminución respecto al modelo anterior, pero aun asi superando bastante al primero

un modelo mas adecuado es el de **cars.mod_2**, con: *Multiple R-squared: 0.9133 y Adjusted R-squared: 0.9097*

$$dist_i = \beta_0 speed_i + \beta_1 speed_i^2 + \epsilon_i$$

Donde:

- dist_i es la distancia de frenado del i-ésimo auto.
- speed_i5 es la velocidad del i-ésimo auto al momento de aplicar los frenos.
- beta_1,beta_2 son los coeficientes del modelo.
- epsilon_i es el error aleatorio asociado con el i-ésimo auto.

definimos beta 0 y beta 1 como nos indica el modelo

In [229]:
```
b_0 = cars.mod_2$coefficients[1]
print(b_0)

b_1 = cars.mod_2$coefficients[2]
print(b_1)
```

```
speed_mps 
0.0447953 
I(speed_mps^2) 
     0.1374789 
```

$$tiempo_{frenado} = \frac{distancia}{velocidad}$$

(b) Para su modelo óptimo estime el tiempo promedio que le toma al conductor aplicar los frenos (hay 5280 pies en una milla).

- Sacamos la media a la distancia de frenado, solo para explorar

In [230]:
```
dist_media = mean(cars$dist_m)

dist_media
```

13.100304

- aplicamos nuestro modelo a los arreglos de datos, podemos cambiar al sistema internacional ya que se conserva la relacion entre variables

In [231]:
```
modelo_dist_arr = b_0 * cars$speed_mps + b_1 * (cars$speed_mps)^2
```

- sacamos la media de dicho arreglo

In [232]:
```
mean_modelo_dist = mean(modelo_dist_arr)

mean_modelo_dist
```

13.0845077292659

se obtiene un valor similar a la media de los datos originales, lo cual es un buen indicio ya que nuestro modelo no incluye dicha distancia, solo la infiere de las velocidades

- obtenemos el tiempo promedio de frenado

In [233]:
```
tiempo_prom = mean_modelo_dist / mean(cars$speed_mps)

tiempo_prom
```

1.90060278906611

el tiempo promedio de frenado es de **1.90060278906611** segundos

(c) Grafique los datos y la estimación de cada modelo en una sola figura

- graficamos el 1er modelo (el peor segun los parametros R) trazando la parabola con los coeficientes de regresion obtenidos del modelo

In [249]:
```
cars.mod_1$coefficients[1:3]
```

(Intercept): 0.752897996888211 **speed_mps:** 0.622697648325463 **I(speed_mps^2):** 0.152457069582459

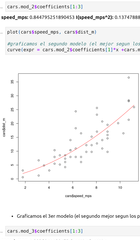
In [251]:
```
plot(cars$speed_mps, cars$dist_m)

#graficamos el 1er modelo (el peor segun los parametros R) trazando la parabola con los coeficientes de regresion obtenidos del modelo
curve(expr = cars.mod_1$coefficients[1]+ cars.mod_1$coefficients[2]*x
              +cars.mod_1$coefficients[3]*x^2,
              from = min(cars$speed_mps), to = max(cars$speed_mps), add = TRUE, col = "red")
```

- Graficamos el segundo modelo (el mejor segun los parametros R^2) trazando la parabola con los coeficientes de regresion obtenidos del modelo

In [236]:
```
cars.mod_2$coefficients[1:3]
```

speed_mps: 0.0447952518890453 **I(speed_mps^2):** 0.137478882066519 **3:** <NA>

In [237]:
```
plot(cars$speed_mps, cars$dist_m)

#graficamos el segundo modelo (el mejor segun los parametros R) trazando la parabola con los coeficientes de regresion obtenidos del modelo
curve(expr = cars.mod_2$coefficients[1]*x +cars.mod_2$coefficients[2]*x^2, from = min(cars$speed_mps), to = max(cars$speed_mps), add = TRUE, col = "red")
```

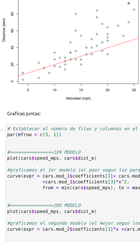


- Graficamos el 3er modelo (el segundo mejor segun los parametros R^2) trazando la recta con los coeficientes de regresion obtenidos del modelo

In [238]:
```
cars.mod_3$coefficients[1:3]
```

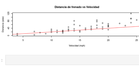
speed_mps: 1.98350411190173 **2:** <NA> **3:** <NA>

In [239]:
```
plot(cars$speed, cars$dist, xlab = "Velocidad (mph)", ylab = "Distancia (pies)", main = "Distancia de frenado vs Velocidad")

#graficamos el tercer modelo con esta funcion, al parecer solo funciona para lineas rectas
abline(cars.mod_3, col= "red")
```



Graficas juntas:

In [253]:
```
# Establecer el número de filas y columnas en el lienzo
par(mfrow = c(3, 1))


#################SER MODELO
plot(cars$speed_mps, cars$dist_m)

#graficamos el 1er modelo (el peor segun los parametros R) trazando la parabola con los coeficientes de regresion obtenidos del modelo
curve(expr = cars.mod_1$coefficients[1]+ cars.mod_1$coefficients[2]*x
              +cars.mod_1$coefficients[3]*x^2,
              from = min(cars$speed_mps), to = max(cars$speed_mps), add = TRUE, col = "red")


#################DOS MODELO
plot(cars$speed_mps, cars$dist_m)

#graficamos el segundo modelo (el mejor segun los parametros R) trazando la parabola con los coeficientes de regresion obtenidos del modelo
curve(expr = cars.mod_2$coefficients[1]*x +cars.mod_2$coefficients[2]*x^2, from = min(cars$speed_mps), to = max(cars$speed_mps), add = TRUE, col = "red")


#################TER MODELO
plot(cars$speed, cars$dist, xlab = "Velocidad (mph)", ylab = "Distancia (pies)", main = "Distancia de frenado vs Velocidad")

#graficamos el tercer modelo con esta funcion, al parecer solo funciona para lineas rectas
abline(cars.mod_3, col= "red")
```



In [ ]:
```

```