TIME TO GET UNDERWAY...

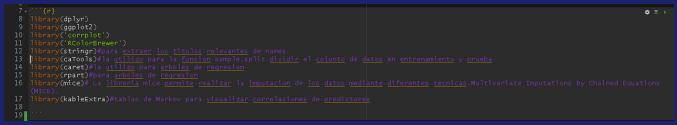
SOBREVIVIENDO AL TITANIC

PROYECTO FINAL ANALISIS DE DATOS
GUATEMALA DE LA ASUNCION 7 DE MAYO 2023

JULIO ANTHONY ENGELS RUIZ
COTO - 1284719

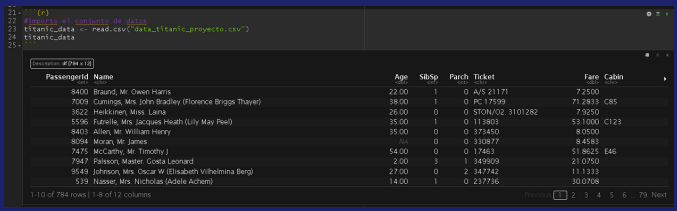
LIBRERIAS

PARA PODER RESOLVER ESTE PROBLEMA FUE NECESARIO INSTALAR LAS SIGUIENTES LIBRERÍAS PARA EXTRAER TEXTO, DIVIDIR EL CONJUNTO DE DATOS, IMPUTACIONES, Y ARBOLES DE REGRESIÓN.



DATOS

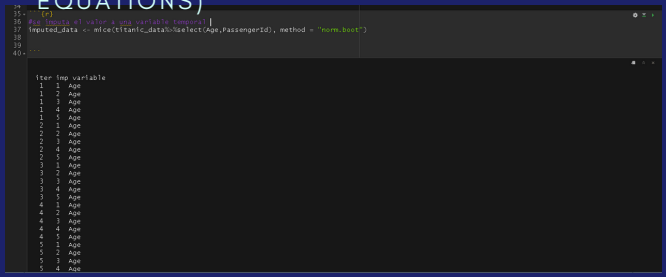
LUEGO SE NOS BRINDO EL DATASET LLAMADO DATA_TITANIC_PROYECTO.CSV ESTE DATASET ES PARA ENTRENAMIENTO Y VALIDACIÓN.



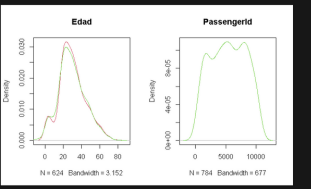
IMPUTACIÓN

CON EL MÉTODO COLSUMS SE VERIFICA QUE EN LA COLUMNA DE AGE HACEN FALTA DATOS.

PARA IMPUTAR LOS VALORES FALTANTES SE HACE USO DE UNA REGRESION LINEAL LA CUAL VIENE DADA MICE. (MULTIVARIATE IMPUTATIONS BY CHAINED EQUATIONS)



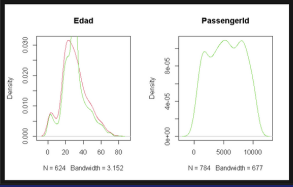
ACÁ SE PUEDE OBSERVAR A LA IZQUIERDA UNA GRÁFICA DE COLOR ROJO QUE ES EL VALOR OBTENIDO DE UNA REGRESION LINEAL DE LOS VALORES IMPUTADOS Y LA GRAFICA DE COLOR VERDE QUE ESTA INTERCALADA ES LA FORMA EN LA QUE SE DISTRIBUYEN LOS DATOS CONOCIDOS.

```
```{r}
#Se utiliza la funcion de regresion lineal de la libreria mice para imputar valores
Data_Impu_Boot <- mice::complete(Imputed_Data)

par(mfrow=c(1,2))

plot(density(titanic_data$Age ,na.rm = T),col=2,main="Edad")
lines(density(Data_Impu_Boot$Age ),col=3)

plot(density(titanic_data$PassengerId,na.rm = T),col=2,main="PassengerId")
lines(density(Data_Impu_Boot$PassengerId),col=3)
```
```



SE PUEDE DECIR QUE SE OBTIENE UN VALOR DE IMPUTACION DE LA EDAD BASTANTE ACEPTABLE.

EN LA GRÁFICA DE LA DERECHA SE PUEDE OBSERVAR LA VARIABLE PASSENGERID Y COMO ESTA VARIABLE NO TIENE NINGUN DATO FALTANTE NO IMPUTA NINGÚN VALOR.

```
```{r}
#Se utiliza para imputar valores el metodo por valor medio
imputed_data <- mice(titanic_data%>%select(Age,PassengerId), method = "mean")
```
```

```
iter imp variable
  1   1  Age
  1   2  Age
  1   3  Age
  1   4  Age
  1   5  Age
  2   1  Age
  2   2  Age
  2   3  Age
  2   4  Age
  2   5  Age
  3   1  Age
  3   2  Age
  3   3  Age
  3   4  Age
  3   5  Age
  4   1  Age
  4   2  Age
  4   3  Age
  4   4  Age
  4   5  Age
  5   1  Age
  5   2  Age
  5   3  Age
  5   4  Age
  5   5  Age
```

AHORA SE COMPARA CON EL MÉTODO PARA IMPUTAR DE VALOR PROMEDIO DE LOS VALORES EXISTENTES

```r
#Se muestra graficamente el resultado de la imputacion en color rojo como se puede ver no satisface como el metodo anterior
Data_Impu_media <- mice::complete(Imputed_data)

par(mfrow=c(1,2))

plot(density(titanic_data$Age ,na.rm = T),col=2,main="Edad")
lines(density(Data_Impu_media$Age ),col=3)

plot(density(titanic_data$PassengerId,na.rm = T),col=2,main="PassengerId")
lines(density(Data_Impu_media$PassengerId),col=3)
```



COMO SE PUEDE OBSERVAR CON EL VALOR PROMEDIO NO SE OBTIENE UNA BUENA DISTRIBUCIÓN DE DATOS IMPUTADOS.

```r
#De acuerdo al mejor desempeño en imputacion se eligio el metodo de regresion lineal
titanic_data<-titanic_data%>mutate(Age = Data_Impu_bootlAge, PassengerId = Data_Impu_bootlPassengerId)
titanic_data
```

Descripción: df [784 × 12]

PassengerId	Name	Age	SibSp	Parch	Ticket	Fare	Cabin
8400	Braund, Mr. Owen Harris	22.00000000	1	0	A/5 21171	7.2500	
7009	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	38.00000000	1	0	PC 17599	71.2833	C85
3622	Heikkinen, Miss. Laina	26.00000000	0	0	STON/O2. 3101282	7.9250	
5536	Futrelle, Mrs. Jacques Heath (Lily May Peel)	35.00000000	1	0	113803	53.1000	C123
8403	Allen, Mr. William Henry	35.00000000	0	0	373450	8.0500	
8094	Moran, Mr. James	52.96998782	0	0	330877	8.4583	
7475	McCarthy, Mr. Timothy J	54.00000000	0	0	17463	51.8625	E46
7947	Palsson, Master. Gosta Leonard	2.00000000	3	1	349909	21.0750	
9549	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	27.00000000	0	2	347742	11.1333	
539	Nasser, Mrs. Nicholas (Adele Achem)	14.00000000	1	0	237736	30.0708	

1-10 of 784 rows | 1-8 of 12 columns

Previous 1 2 3 4 5 6 ... 79 Next

DE ACUERDO AL MEJOR DESEMPEÑO EN IMPUTACIÓN SE ELIGIÓ EL MÉTODO DE REGRESIÓN LINEAL.

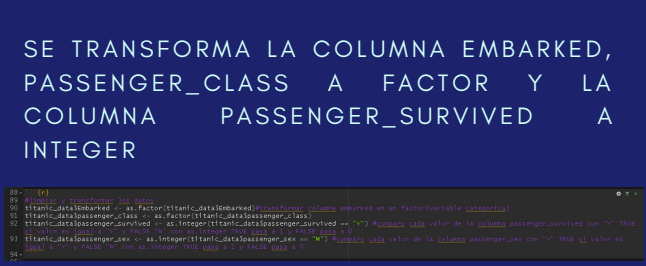
```r
colSums(is.na(titanic_data))
```

```
PassengerId        Name         Age       SibSp       Parch      Ticket        Fare 
          0           0           0           0           0           0           0 
      Cabin    Embarked passenger_c lass passenger_sex passenger_survived 
          0           0           0           0           0 
```

CON ESTO SE COMPRUEBA QUE YA TODAS LAS COLUMNAS NO TIENEN VALORES NAS

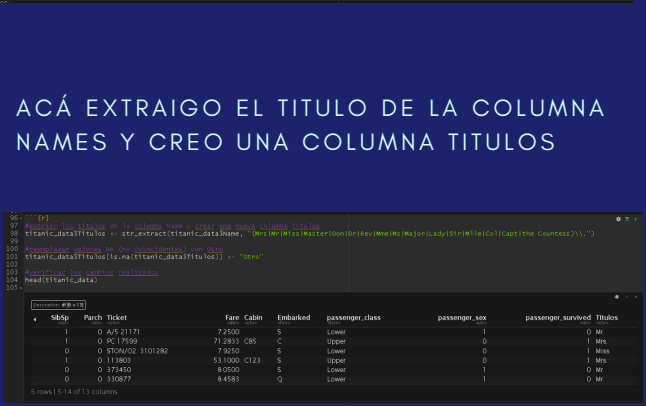
FEATURE ENGINEERING

SE TRANSFORMA LA COLUMNA EMBARKED, PASSENGER_CLASS A FACTOR Y LA COLUMNA PASSENGER_SURVIVED A INTEGER



```r
```{r}
#Limpiar y transformar los datos
titanic_data$Embarked <- as.factor(titanic_data$Embarked)#transformar columna embarked en un factor(variable categorica)
titanic_data$passenger_class <- as.factor(titanic_data$passenger_class)
titanic_data$passenger_survived <- as.integer(titanic_data$passenger_survived == "Y") #comparo cada valor de la columna passenger_survived con "Y" TRUE si valor es igual a "Y" y FALSE si no es igual(retorna TRUE para 1 y FALSE para a 0)
titanic_data$passenger_sex <- as.integer(titanic_data$passenger_sex == "M") #comparo cada valor de la columna passenger_sex con "Y" TRUE si valor es igual a "Y" y FALSE "N" con as.integer TRUE para a 1 y FALSE para a 0
```
```

ACÁ EXTRAIGO EL TITULO DE LA COLUMNA NAMES Y CREO UNA COLUMNA TITULOS



```r
```{r}
#Extraer los títulos de la columna Name y crear una nueva columna Títulos
titanic_data$Titulos <- str_extract(titanic_data$Name, "(Mrs|Mr|Ms|Master|Don|Dr|Rev|Mme|Ms|Major|Lady|Sir|Mlle|Col|Capt|the Countess)\\.")

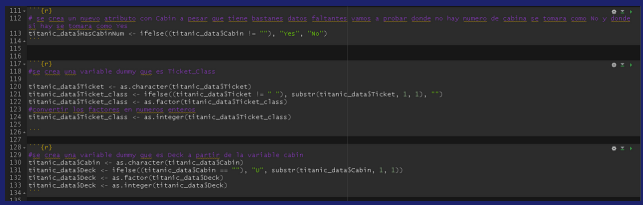
#reemplazar valores NA (no coincidentes) con otro
titanic_data$Titulos[is.na(titanic_data$Titulos)] <- "Otro"

#verificar los cambios realizados
head(titanic_data)
```
```

Descending: ↓ 0 × 13

	SibSp	Parch	Ticket	Fare	Cabin	Embarked	passenger_class	passenger_sex	passenger_survived	Titulos
	<dbl>	<dbl>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<chr>
1	1	0	A/5 21171	7.2500		S	Lower	1	0	Mr.
2	1	0	PC 17599	71.2833	C85	C	Upper	0	1	Mrs.
3	0	0	STON/O2. 3101282	7.9250		S	Lower	0	1	Miss.
4	1	0	113803	53.1000	C123	S	Upper	0	1	Mrs.
5	0	0	373450	8.0500		S	Lower	1	0	Mr.
6	0	0	330877	8.4583		Q	Lower	1	0	Mr.

6 rows | 5-14 of 13 columns



```r
# se crea un nuevo atributo con Cabin a pesar que tiene bastantes datos faltantes vamos a probar donde no hay numero de cabina se tomara como No y donde si hay se tomara como Yes
titanic_data$HasCabinNum <- ifelse(!(titanic_data$Cabin != ""), "Yes", "No")
```

```r
#se crea una variable dummy que es Ticket_Class

titanic_data$Ticket <- as.character(titanic_data$Ticket)
titanic_data$Ticket_class <- ifelse(!(titanic_data$Ticket != " "), substr(titanic_data$Ticket, 1, 1), "")
titanic_data$Ticket_class <- as.factor(titanic_data$Ticket_class)
#convertir las factores en caracter nuevamente
titanic_data$Ticket_class <- as.integer(titanic_data$Ticket_class)
```

```r
#se crea una variable dummy que es Deck a partir de la variable cabin
titanic_data$Cabin <- as.character(titanic_data$Cabin)
titanic_data$Deck <- ifelse(!(titanic_data$Cabin == ""), "N", substr(titanic_data$Cabin, 1, 1))
titanic_data$Deck <- as.factor(titanic_data$Deck)
titanic_data$Deck <- as.integer(titanic_data$Deck)
```

SE CREAN VARIABLES DUMMIES HASCABINNUM (RELACIONA CON LA VARIABLE CABIN), TICKET_CLASS(RELACIONA CON TICKET), DECK(SE RELACIONA CON CABIN).

```r
# crea un conjunto de datos sin columnas innecesarias.
titanic_data_clean <- titanic_data %>% select(-Ticket)
titanic_data_clean
```

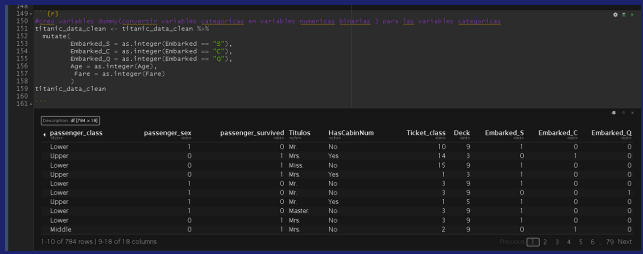
Description: df [784 × 15]

PassengerId	Name	Age	SibSp	Parch	Fare	Cabin	Embarked
8400	Braund, Mr. Owen Harris	22.00000000	1	0	7.2500		S
7009	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	38.00000000	1	0	71.2833	C85	C
1622	Heikkinen, Miss. Laina	26.00000000	0	0	7.9250		S
5936	Futrelle, Mrs. Jacques Heath (Lily May Peel)	35.00000000	1	0	53.1000	C123	S
8403	Allen, Mr. William Henry	35.00000000	0	0	8.0500		S
8094	Moran, Mr. James	52.96998782	0	0	8.4583		Q
7475	McCarthy, Mr. Timothy J	54.00000000	0	0	51.8625	E46	S
7947	Palsson, Master. Gosta Leonard	2.00000000	3	1	21.0750		S
9549	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	27.00000000	0	2	11.1333		S
539	Nasser, Mrs. Nicholas (Adele Achem)	14.00000000	1	0	30.0708		C

1-10 of 784 rows | 1-8 of 15 columns

Previous 1 2 3 4 5 6 ... 79 Next

ELIMINO LAS COLUMNAS INNECESARIAS



```r
# transformar dummy/categoric variables categoricas en variables numericas binarias ) para las variables categoricas
titanic_data_clean <- titanic_data_clean %>%
  mutate(
         Embarked_S = as.integer(Embarked == "S"),
         Embarked_C = as.integer(Embarked == "C"),
         Embarked_Q = as.integer(Embarked == "Q"),
         Age = as.integer(Age),
          Fare = as.integer(Fare)
  )

titanic_data_clean
```

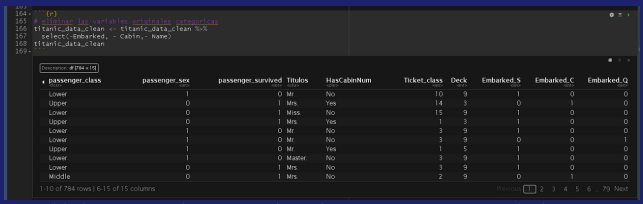
Description: df [784 × 18]

passenger_class	passenger_sex	passenger_survived	Titulos	HasCabinNum	Ticket_class	Deck	Embarked_S	Embarked_C	Embarked_Q
Lower	1	0	Mr	No	10	9	1	0	0
Upper	0	1	Mrs	Yes	14	3	0	1	0
Lower	0	1	Miss	No	15	9	1	0	0
Upper	0	1	Mrs	Yes	1	3	1	0	0
Lower	1	0	Mr	No	3	9	1	0	0
Lower	1	0	Mr	No	3	9	0	0	1
Upper	1	0	Mr	Yes	1	5	1	0	0
Lower	1	0	Master	No	3	9	1	0	0
Lower	0	1	Mrs	No	3	9	1	0	0
Middle	0	1	Mrs	No	2	9	0	1	0

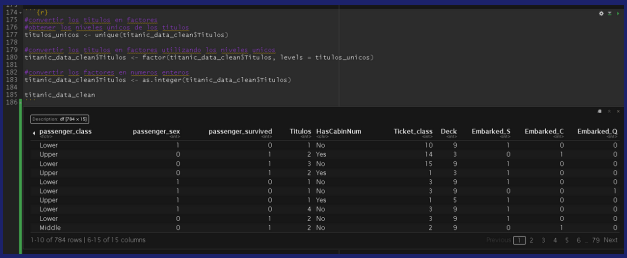
1-10 of 784 rows | 9-18 of 18 columns

Previous 1 2 3 4 5 6 ... 79 Next

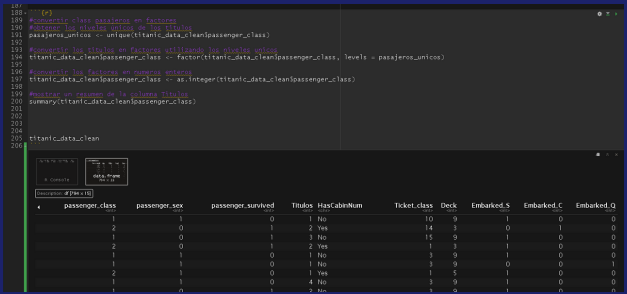
CREO VARIABLES DUMMY DE LA COLUMNA EMBARKED(EMBARKED_S, EMBARKED_C, EMBARKED_Q)



ELIMINO LAS VARIABLES ORIGINALES CATEGÓRICAS



CONVIERTO LA COLUMNA DE TITULOS EN CATEGÓRICAS



CONVIERTO LA COLUMNA DE PASSENGER_CLASS EN CATEGÓRICAS

```r
#Se crea una nueva variable dummy llamada HasCabinNum
#Convertir HasCabinNum en factores
#Obtener los niveles únicos de las cabinas
cabinas_unicas <- unique(titanic_data_clean$HasCabinNum)

#Convertir los niveles en factores, utilizando los niveles únicos
titanic_data_clean$HasCabinNum <- factor(titanic_data_clean$HasCabinNum, levels = cabinas_unicas)

#Convertir los factores en numeros enteros
titanic_data_clean$HasCabinNum <- as.integer(titanic_data_clean$HasCabinNum)

titanic_data_clean
```

Data Frame: 891 rows x 15 columns

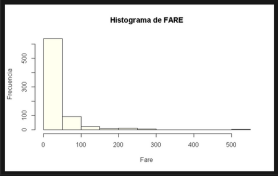
PassengerId	Age	SibSp	Parch	Fare	passenger_class	passenger_sex	passenger_survived	Titulos	HasCabinNum
8400	22	1	0	7	1	1	0	1	1
7009	38	1	0	71	2	0	1	2	2
3622	26	0	0	7	1	0	1	3	1
5596	35	1	0	53	2	0	1	2	2
8403	35	0	0	8	1	1	0	1	1
8094	52	0	0	8	1	1	0	1	1
7475	54	0	0	51	2	1	0	1	2
7947	2	3	1	21	1	1	0	4	1
9549	27	0	2	11	1	0	1	2	1
539	14	1	0	30	3	0	1	2	1

1-10 of 784 rows | 1-10 of 15 columns

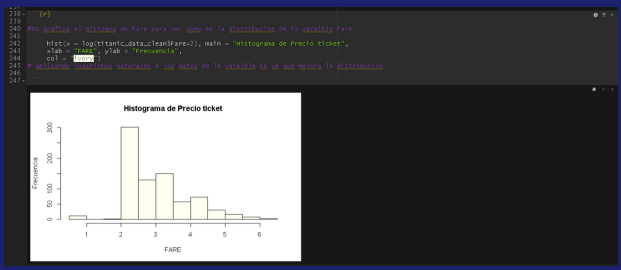
Previous 1 2 3 4 5 6 ... 79 Next

CONVIERTO LA COLUMNA DE HASCABINNUM EN CATEGÓRICAS

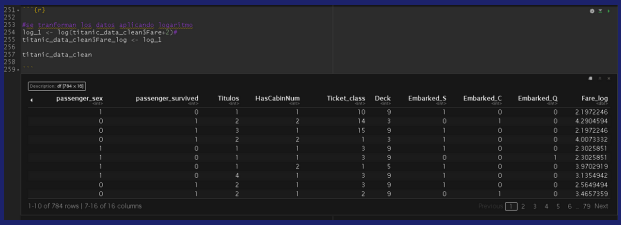
```r
#Se grafica el histogra de Fare para ver como es la distribucion de la varaible Fare

    hist(x = titanic_data_clean$Fare, main = "Histograma de FARE",
    xlab = "Fare", ylab = "Frecuencia",
    col = "green")
#Como se puede apreciar esta variable tiene mucho sesgo a la derecha
```



SE GRAFICA EL HISTOGRA DE FARE PARA VER COMO ES LA DISTRIBUCION DE LA VARAIBLE FARE, COMO SE PUEDE APRECIAR ESTA VARIABLE TIENE MUCHO SESGO A LA DERECHA



APLICANDO LOGARITMOS NATURALES A LOS DATOS DE LA VARIABLE SE VE QUE MEJORA LA DISTRIBUCIÓN



passenger_sex	passenger_survived	Titulos	HasCabinNum	Ticket_class	Deck	Embarked_S	Embarked_C	Embarked_Q	Fare_log
1	0	1	1	10	9	1	0	0	2.1972246
0	1	2	2	14	3	0	1	0	4.2904594
0	1	3	1	15	9	1	0	0	2.1972246
0	1	2	2	1	3	1	0	0	4.0073332
1	0	1	1	3	9	1	0	0	2.3025851
1	0	1	1	3	9	0	0	1	2.3025851
1	0	1	2	1	5	1	0	0	3.9702919
1	0	4	1	3	9	1	0	0	3.1354942
0	1	2	1	3	9	1	0	0	2.5649494
0	1	2	1	2	9	0	1	0	3.4657359

ACÁ SE SUMA A LA COLUMA FARE UN 2 PARA EVITAR VALORES INDEFINIDOS DE LOGARITMO NATURAL CUANDO FARE TENGA UN VALOR 0 Y SE CREA LA COLUMNA FARE_LOG CON ESTE CALCULO

```r
titanic_data_clean <- titanic_data_clean %>%
  select(-Fare)
titanic_data_clean
```

Descripción: df [784 x 15]

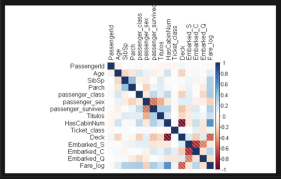
PassengerId	Age	SibSp	Parch	passenger_class	passenger_sex	passenger_survived	Titulos	HasCabinNum	Ticket_class
8400	22	1	0	1	1	0	1	1	10
7009	38	1	0	2	0	1	2	2	14
3622	26	0	0	1	0	1	3	1	15
5196	35	1	0	2	0	1	2	2	1
8403	35	0	0	1	1	0	1	1	3
8094	52	0	0	1	1	0	1	1	3
7475	54	0	0	2	1	0	1	2	1
7947	2	3	1	1	1	0	4	1	3
9549	27	0	2	1	0	1	2	1	3
539	14	1	0	3	0	1	2	1	2

1-10 of 784 rows | 1-10 of 15 columns

Previous 1 2 3 4 5 6 ... 79 Next

ELIMINO LA COLUMNA FARE

```r
#vamos a correlacionar la passenger_survived con el resto de predictores
corrplot(cor(titanic_data_clean), # Matriz de correlación
         method = "shade", # Método para el gráfico de correlación
         type = "full",    # Estilo del gráfico (también "upper" y "lower")
         diag = TRUE,      # Si TRUE (por defecto), añade la diagonal
         tl.col = "black", # Color de las etiquetas
         bg = "white",     # Color de fondo
         title = "",       # Título
         col = NULL)       # Paleta de colores
```



SE GRÁFICA LA MATRIZ DE CORRELACIONES PARA LA VARIABLE PASSENGER_SURVIVED DONDE SE PUEDE APRECIAR CON QUE OTRAS VARIABLES TIENE CORRELACIÓN SEA POSITIVA O NEGATIVA ESTO SERVIRÁ PARA DEFINIR LOS PREDICTORES QUE SERVIRÁN PARA ENTRENAR EL MODELO

EVALUACIÓN Y PREDICCIÓN

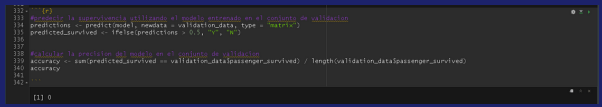
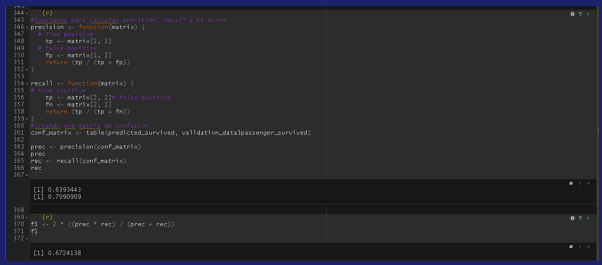
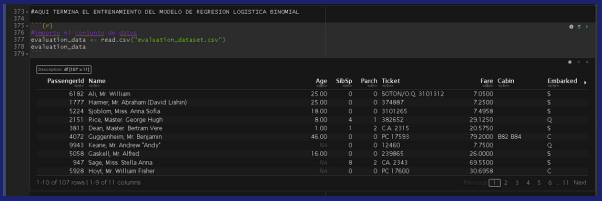
```
```{r}
# dividir los datos en conjuntos de entrenamiento y prueba (80% - 20%)
set.seed(1234) #establece una semilla para que la division sea reproducible

sample <- sample.split(titanic_data_clean$passenger_survived, SplitRatio = 0.8)
train_data  <- subset(titanic_data_clean, sample == TRUE)
validation_data   <- subset(titanic_data_clean, sample == FALSE)

#se muestra las dimensiones del conjunto de entrenamiento

dim(train_data)

#se muestra las dimensiones del conjunto de prueba
dim(validation_data)

```

[1] 627 15
[1] 157 15
```

APARTIR DEL DATASET TITANIC_DATA_CLEAN SE CREA UNA DIVISIÓN DE DATOS ALEATORIA CON EL MÉTODO SPLIT 80%-20% PARA OBTENER LOS DATASETS DE ENTRENAMIENTO Y VALIDACIÓN

```
```{r}

# entrenando el modelo de regresión logistica con los predictores que tienen mas correlacion
model <- rpart(passenger_survived ~ passenger_class + passenger_sex + HasCabinNum + Fare_log + Embarked_S + Embarked_C + Titulos + Deck, data =
train_data)

summary(model)
```

Call:
rpart(formula = passenger_survived ~ passenger_class + passenger_sex +
    HasCabinNum + Fare_log + Embarked_S + Embarked_C + Titulos +
    Deck, data = train_data)
  n= 627

          CP nsplit rel error   xerror   xstd
1 0.33318666   0 1.0000000 1.0037465 0.01853315
2 0.06617680   1 0.6668133 0.6963193 0.04073448
3 0.04069405   2 0.6006365 0.6332260 0.03955848
4 0.03264045   3 0.5599425 0.6106819 0.04167778
5 0.02740623   4 0.5273020 0.5897468 0.04049836
6 0.01000000   5 0.4998958 0.5486798 0.03781447

Variable importance
        Titulos   passenger_sex   Fare_log   Deck   HasCabinNum passenger_class   Embarked_S
             36   29   12   8   7   6   2

Node number 1: 627 observations,   complexity param=0.3331867
  mean=0.3875598, MSE=0.2373572
  left son=2 (373 obs) right son=3 (254 obs)
  Primary splits:
      Titulos   < 1.5   to the left, improve=0.33318670, (0 missing)
      passenger_sex   < 0.5   to the right, improve=0.32579950, (0 missing)
      passenger_class < 1.5   to the left, improve=0.11064860, (0 missing)
      Deck   < 6.5   to the right, improve=0.10290620, (0 missing)
      HasCabinNum   < 1.5   to the left, improve=0.09833606, (0 missing)
  Surrogate splits:
      passenger_sex < 0.5   to the right, agree=0.944, adj=0.862, (0 split)
      Fare_log   < 2.740319 to the left, agree=0.641, adj=0.114, (0 split)
      Deck   < 3.5   to the right, agree=0.619, adj=0.059, (0 split)
```

PARA ENTRENAR EL MODELO DE ARBOLES DE REGRESIÓN LOGÍSTICA SE HACE USO DEL DATASET TRAIN_DATA Y ADEMAS POR CORRELACION SE DEFINEN LOS PREDICTORES QUE SERVIRÁN PARA PREDECIR A LA VARIABLE OBJETIVO PASSENGER_SURVIVED



UNA VEZ EL MODELO ENTRENADO SE PROCEDE A PREDECIR LOS VALORES DE LA VARIABLE OBJETIVO PASSENGER_SURVIVED, USANDO EL METODO PREDICT Y PARA ELLO SE USA EL DATASET VALIDATION_DATA.



CON ESTAS MÉTRICAS SIRVEN PARA MEDIR QUE TAN BIEN ESTA PREDICIENDO EL MODELO CON DATOS DE ENTRENAMIENTO.

ESTAS METRICAS SE OBTUVIERON DE LOS DIFERENTES INTENTOS Y MODELOS DE MACHINE LEARNING DE APRENDIZAJE SUPERVISADO.

	F1 SCORE	PRECISION	RECALL
Entrenamiento fallido	-	-	-
Entrenamiento fallido	-	-	-
Entrenamiento fallido	-	-	-
Entrenamiento fallido	-	-	-
Entrenamiento y validación 60% - 40 %	[1] 0.7420814	[1] 0.7130435	[1] 0.7735849
Entrenamiento y validación 60% - 40 %	[1] 0.7420814	[1] 0.7130435	[1] 0.7735849
Entrenamiento y validación 80% - 20 %	[1] 0.733945	[1] 0.7017544	[1] 0.7692308
Entrenamiento y validación 90% - 10 %	[1] 0.7666667	[1] 0.71875	[1] 0.8214286
Quitando Embarked_S, Embarked_C, Embarked_Q y agregando columna Embarked entrenamiento y validación 90% - 10%	[1] 0.7868852	[1] 0.75	[1] 0.8275862
Normalizando la columna Fare_log 90% - 10%	[1] 0.7868852	[1] 0.75	[1] 0.8275862
Aplicando Arboles de regresión logística 90% - 10%	[1] 0.7741935	[1] 0.75	[1] 0.8
Aplicando arboles de decisión con variables Dummies	[1] 0.8	[1] 0.7333333	[1] 0.88
Aplicando regresión logística	[1] 0.7857143	[1] 0.7333333	[1] 0.8461538
Aplicando regresión logística	[1] 0.7857143	[1] 0.7333333	[1] 0.8461538
Aplicando la normalización 80% - 20%	[1] 0.789916	[1] 0.7704918	[1] 0.8103448
Aplicando arboles de decisión 90% - 10%	[1] 0.7586207	[1] 0.6875	[1] 0.8461538
Aplicando RandomForest	[1] 0.75	[1] 0.7	[1] 0.8076923
Aplicando regresión logística	[1] 0.7333333	[1] 0.6875	[1] 0.7857143
Aplicando regresión logística	[1] 0.6896552	[1] 0.6666667	[1] 0.7142857
Aplicando regresión logística	[1] 0.7142857	[1] 0.6666667	[1] 0.7692308
Aplicando regresión logística	[1] 0.7213115	[1] 0.7333333	[1] 0.7096774
Aplicando regresión logística	[1] 0.6785714	[1] 0.6333333	[1] 0.7307692
Aplicando regresión logística	[1] 0.6785714	[1] 0.6333333	[1] 0.7307692
Aplicando arboles de regresión logística	[1] 0.6785714	[1] 0.6333333	[1] 0.7307692
Aplicando arboles de regresión logística	[1] 0.6724138	[1] 0.6393443	[1] 0.7090909
Aplicando arboles de regresión logística	[1] 0.6724138	[1] 0.6393443	[1] 0.7090909
Aplicando arboles de regresión logística	[1] 0.6724138	[1] 0.6393443	[1] 0.7090909

EL INTENTO FINAL CON MAYOR SCORE EN KAGGLE

PRUEBAS DEL MODELO DE APRENDIZAJE CON DATOS DE EVALUACIÓN

LUEGO SE NOS BRINDO EL DATASET LLAMADO EVALUATION_DATASET.CSV ESTE DATASET NO CONTO CON LA COLUMNA DE PASSENGER_SURVIVED



```
#AQUI TERMINA EL ENTRENAMIENTO DEL MODELO DE REGRESION LOGISTICA BINOMIAL

(fr)
#Cargando el conjunto de datos
evaluation_data <- read.csv("evaluation_dataset.csv")
evaluation_data
```

data.frame: 107 x 11

PassengerId	Name	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
6182	Ali, Mr. William	25.00	0	0	SOTON/O.Q. 3101312	7.0500		S
1777	Hamer, Mr. Abraham (David Lishin)	25.00	0	0	374887	7.2500		S
1224	Sjoblom, Miss. Anna Sofia	18.00	0	0	3101265	7.4958		S
2151	Rice, Master. George Hugh	8.00	4	1	382652	29.1250		Q
3813	Dean, Master. Bertram Vere	1.00	1	2	C.A. 2315	20.5750		S
4072	Guggenheim, Mr. Benjamin	46.00	0	0	PC 17593	79.2000	B82 B84	C
9943	Keane, Mr. Andrew "Andy"	NA	0	0	12460	7.7500		Q
5058	Caldwell, Mr. Alfred	16.00	0	0	239865	26.0000		S
947	Sage, Miss. Stella Anna	NA	8	2	CA. 2343	69.5500		S
5929	Hoyt, Mr. William Fisher	NA	0	0	PC 17600	30.6958		C

1-10 of 107 rows | 1-9 of 11 columns

Previous 1 2 3 4 5 6 ... 11 Next

LUEGO DE IMPORTAR ESTE DATASET DE EVALUATION SE APLICO EL MISMO PROCEDIMIENTO DE ANALISIS DE DATOS, LIMPIEZA DE DATOS, IMPUTACION DE DATOS PARA DEJARLO DE LA MISMA FORMA QUE EL DATASET DE ENTRENAMIENTO Y CON ELLO SE BUSCA HOMOGENISAR LOS DATOS EN AMBOS ARCHIVOS. CON ESTE DATASET DE EVALUATION SE TIENE COMO OBJETIVO EVALUAR EL MODELO CON DATOS TOTALMENTE AGENOS A LOS QUE SIRVIERON EN EL ENTRENAMIENTO DEL MODELO DE APRENDIZAJE SUPERVISADO.

SE APLICA LAS PREDICCIONES CON EL DATASET DE EVALUATION_DATA_CLEAN PARA DETERMINAR LAS PREDICCIONES OBTENIDAS SON PROBABILIDADES DE OCURRENCIA DEL EVENTO POR LO QUE SE USA LA FUNCION PARA CLASIFICAR TODA PROBABILIDAD > 0.5 SE CONSIDERA COMO VERDADERO Y TODA PROBABILIDAD < 0.5 SE CONSIDERA COMO FALSO.

````
```{r}
PassengerId <- evaluation_data_clean$PassengerId
predicciones_data <- cbind(PassengerId , passenger_survived )
predicciones_data

```
````

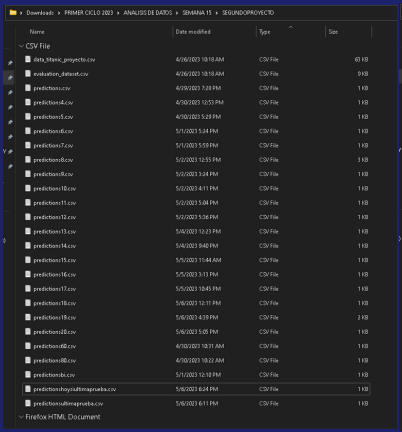
```
      PassengerId passenger_survived
 [1,]        6182                  0
 [2,]        1777                  0
 [3,]        5224                  1
 [4,]        2151                  0
 [5,]        3813                  1
 [6,]        4072                  0
 [7,]        9943                  0
 [8,]        5058                  0
 [9,]         947                  0
[10,]        5928                  0
[11,]        9617                  0
[12,]        3930                  0
[13,]        5842                  1
[14,]        9216                  1
[15,]        7517                  0
[16,]       10362                  1
[17,]        3915                  0
[18,]        9733                  0
[19,]        1915                  1
[20,]        7398                  1
[21,]        7475                  0
[22,]        5766                  0
[23,]        8660                  0
[24,]        1628                  1
[25,]        5937                  0
[26,]        7419                  1
[27,]        6217                  0
[28,]        4992                  0
[29,]       10674                  0
[30,]        8950                  0
```

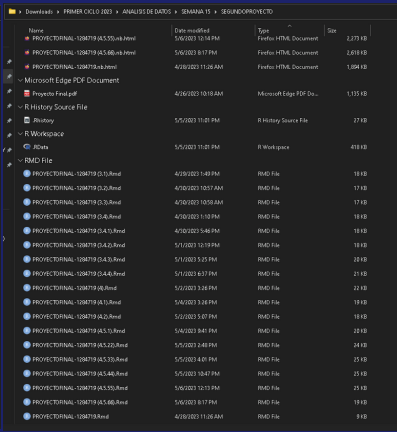
SE CONSTRUYE UN DATASET CON EL FORMATO PARA QUE KAGGLE PUEDA EVALUAR LA PRUEBA.

```
 [82,]        5377                  0
 [83,]        9818                  1
 [84,]        8889                  0
 [85,]        4054                  0
 [86,]       10650                  1
 [87,]        6133                  0
 [88,]        3275                  1
 [89,]        3221                  0
 [90,]        9817                  0
 [91,]        2762                  0
 [92,]        2212                  1
 [93,]        4919                  0
 [94,]        6389                  0
 [95,]        6139                  0
 [96,]        9769                  1
 [97,]        1318                  0
 [98,]        2228                  0
 [99,]        8486                  1
[100,]       10206                  0
[101,]        9601                  0
[102,]        6608                  0
[103,]         983                  0
[104,]       10484                  1
[105,]        2769                  1
[106,]        1934                  0
[107,]        1016                  0
```

SIEMPRE VERIFICANDO LA CANTIDAD DE FILAS CON LOS DATASET ORIGINALES PARA NO TENER MAS O MENOS DATOS.

PARA FINALIZAR SE GUARDA ESTE DATASET DE LAS PREDICCIONES OBTENIDAS POR EL MODELO EN UN FORMATO DE CSV SEPARADO POR COMA





DEJO EVIDENCIA DE TODAS LAS PRUEBAS DE LOS MODELOS QUE UTILICE PARA ESTA SOLUCION AL PROBLEMA.

COMENTARIO FINAL

CON MI MODELO DE APRENDIZAJE AUTOMATICO SUPERVISADO DE ARBOLES DE REGRESION LOGISTICA PUEDE OBSERVAR QUE LAS METRICAS OBTENIDAS EN EL ENTRENAMIENTO ERAN MAYORES A LAS OBTENIDAS CUANDO SE ENTRENA EL MODELO CON DATOS NUEVOS ESTO ME HACE PENSAR EN QUE EXISTE UN OVERFITTING CONSIDERO QUE ESO HACE QUE EL RESULTADO OBTENIDO EN KAGGLE NO SEA DEL TODO ACEPTABLE YA QUE EL MODELO NO APRENDE LO SUFICIENTEMENTE BIEN PARA ELLO HACE FALTA MAYOR ANALISIS DE DATOS POR QUE ESTO PUEDE CAUSAR DICHOS EFECTOS.