PROYECTO 1: Score de Compra

Eduardo Tomás Leyva Díaz

Datos y Librerías

Cargamos la base de datos y se activan las librerías necesarias para realizar los cálculos.

```
# Librerías
library(readxl)
library(dplyr)
library(ggplot2)
library(caret)    # matriz de confusión
library(pROC)     # curva ROC
library(knitr)
library(corrplot)

# Datos
setwd("C:/Users/Eduardo Leyva/Documents/Octavo Semestre/Administración Integral de Riesgos/Riesgo de Cr
datos<-read_excel("Proyecto 1 Datos.xlsm",sheet="E12")
kable(head(datos,20),align = "cccccc")
```

User ID	Gender	Age	EstimatedSalary USD	Purchased	Sample
15624510	Male	19	19000	0	Train
15810944	Male	35	20000	0	Train
15668575	Female	26	43000	0	Train
15603246	Female	27	57000	0	Train
15804002	Male	19	76000	0	Test
15728773	Male	27	58000	0	Train
15598044	Female	27	84000	0	Train
15694829	Female	32	150000	1	Train
15600575	Male	25	33000	0	Train
15727311	Female	35	65000	0	Train
15570769	Female	26	80000	0	Train
15606274	Female	26	52000	0	Train
15746139	Male	20	86000	0	Train
15704987	Male	32	18000	0	Train
15628972	Male	18	82000	0	Test
15697686	Male	29	80000	0	Train
15733883	Male	47	25000	1	Train
15617482	Male	45	26000	1	Train
15704583	Male	46	28000	1	Train
15621083	Female	48	29000	1	Train

El conjunto de datos está dividido en dos tipos, los datos de **entrenamiento** que se utilizarán para crear el modelo de score y los datos de **prueba** para verificar si predice correctamente, así que lo primero es realizar un filtrado de la base de datos para separarlos.

```
# Datos de entrenamiento
datos_train<-datos %>%
              filter(Sample=="Train")

# Datos de prueba
datos_test<-datos %>%
              filter(Sample=="Test")

kable(head(datos_train,15),align = "cccccc")
```

User ID	Gender	Age	EstimatedSalary USD	Purchased	Sample
15624510	Male	19	19000	0	Train
15810944	Male	35	20000	0	Train
15668575	Female	26	43000	0	Train
15603246	Female	27	57000	0	Train
15728773	Male	27	58000	0	Train
15598044	Female	27	84000	0	Train
15694829	Female	32	150000	1	Train
15600575	Male	25	33000	0	Train
15727311	Female	35	65000	0	Train
15570769	Female	26	80000	0	Train
15606274	Female	26	52000	0	Train
15746139	Male	20	86000	0	Train
15704987	Male	32	18000	0	Train
15697686	Male	29	80000	0	Train
15733883	Male	47	25000	1	Train

Análisis Descriptivo

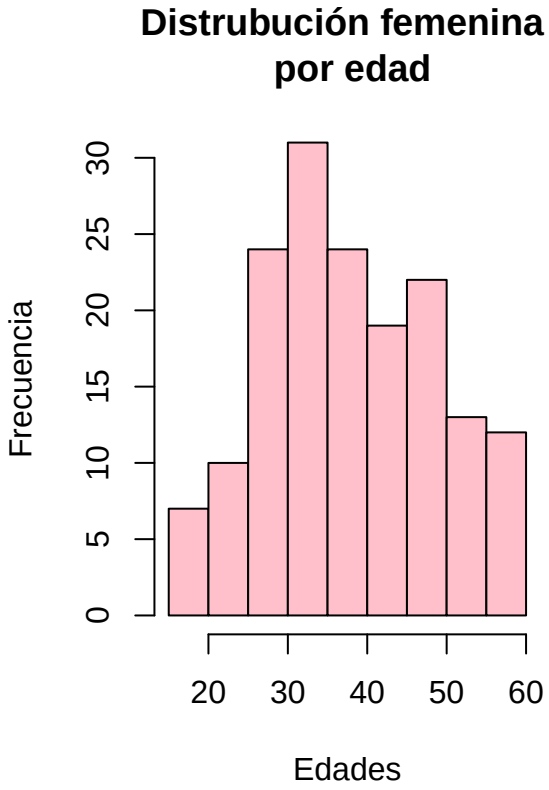
Como primer paso se realizará un análisis preliminar de los datos de entrenamiento, para esto se dividirá en dos grupos dependiendo la variable del **Género**. De aquí se presentará un pequeño resumen de las medidas de tendencia central para poder observar el comportamiento que tiene con la **Edad** y el **Salario** registrado.

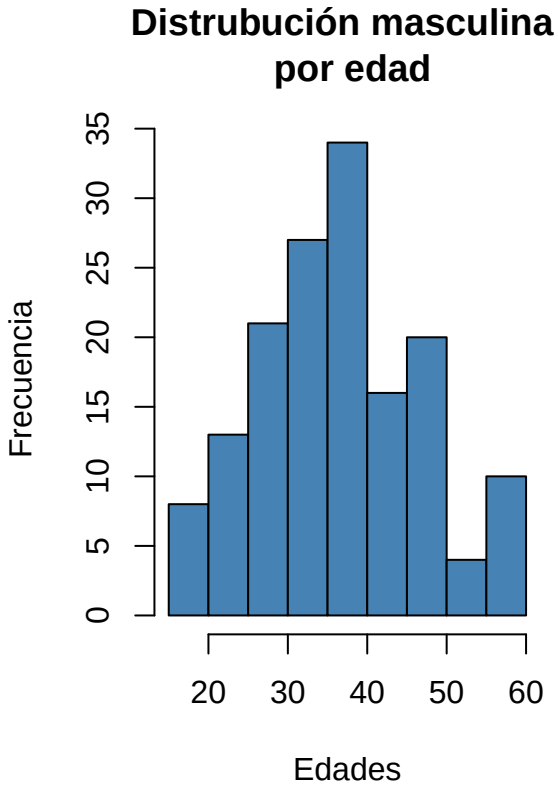
```
# Filtrar por Género
fem_train <- filter(datos_train,Gender=="Female")
male_train <- filter(datos_train,Gender=="Male")

# Edad y Género
GeneroEdad <-datos_train %>%
  group_by(Gender) %>%
  summarise(Variable="Edad",Media=mean(Age,na.rm=T),
 Mediana=median(Age,na.rm=T),
 Varianza=var(Age,na.rm=T),
 SD=sd(Age,na.rm=T),
 Min=min(Age),
 Max=max(Age))
```

```
kable(GeneroEdad,align ="cccccccc",digits=2)
```

Gender	Variable	Media	Mediana	Varianza	SD	Min	Max
Female	Edad	38.28	37	115.57	10.75	18	60
Male	Edad	37.03	37	100.62	10.03	18	60

```
par(mfrow=c(1,2))
hist(fem_train$Age,xlab = "Edades",ylab="Frecuencia",
     main="Distrubución femenina \n por edad",col="pink")
hist(male_train$Age,xlab = "Edades",ylab="Frecuencia",
     main="Distrubución masculina \n por edad",col="steelblue")
```

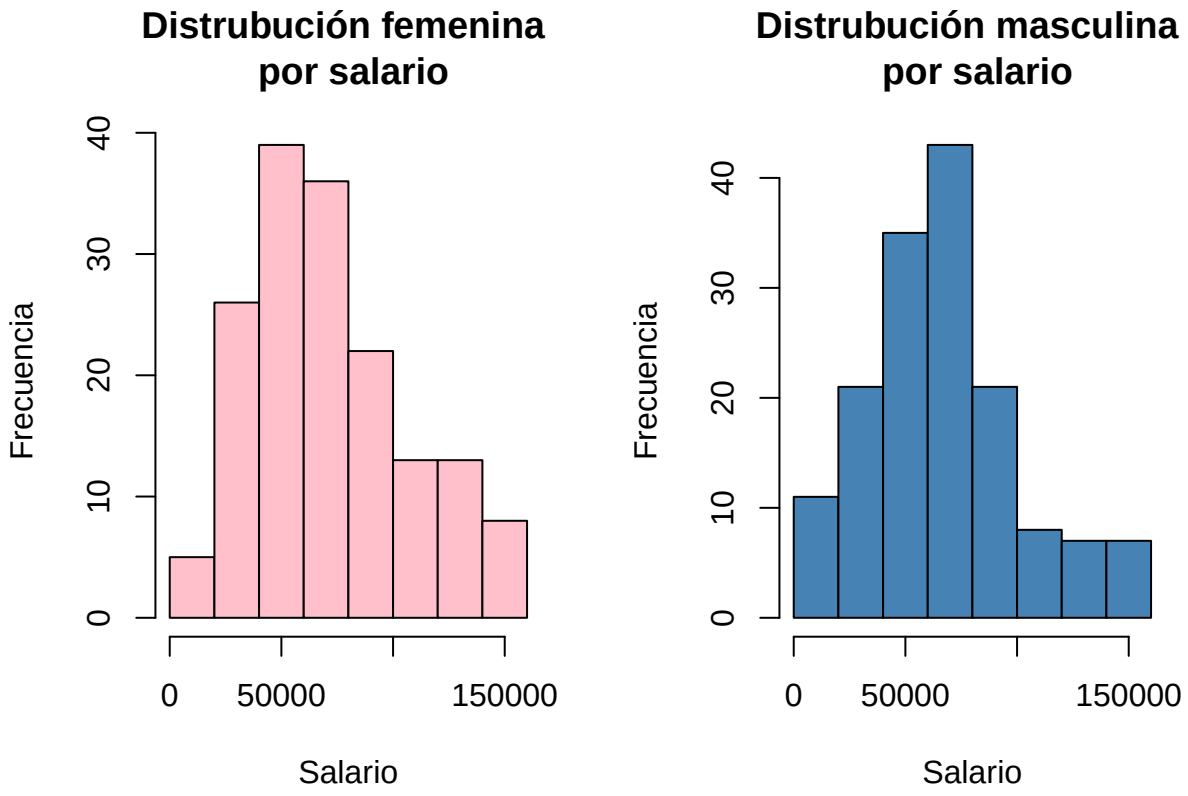





```
# Salario y Género
GeneroSalario <-datos_train %>%
  group_by(Gender) %>%
  summarise(Variable="Salario",Media=mean(`EstimatedSalary USD`,na.rm=T),
 Mediana=median(`EstimatedSalary USD`,na.rm=T),
 Varianza=var(`EstimatedSalary USD`,na.rm=T),
 SD=sd(`EstimatedSalary USD`,na.rm=T),
 Min=min(`EstimatedSalary USD`),
 Max=max(`EstimatedSalary USD`))
kable(GeneroSalario,align ="cccccccc",digits=2)
```

Gender	Variable	Media	Mediana	Varianza	SD	Min	Max
Female	Salario	72222.22	71000	1237987578	35185.05	15000	150000
Male	Salario	68019.61	65000	1100861455	33179.23	16000	150000

```
par(mfrow=c(1,2))
hist(fem_train$`EstimatedSalary USD`,
     xlab = "Salario",ylab="Frecuencia",
     main="Distrubución femenina \n por salario",col="pink")
hist(male_train$`EstimatedSalary USD`,
     xlab="Salario",ylab="Frecuencia",
     main="Distrubución masculina \n por salario",col="steelblue")
```



Lo que se puede rescatar de las gráficas y tablas anteriores es información como las edades mínimas y máximas tanto de hombres como mujeres, siendo de 18 y 60 años para ambos sexos, la media de su salario y qué tanto podría variar. Respecto a las mujeres, la media de su edad fue de 38 años y la muestra se concentró en mujeres entre los 25 y 40 años de edad, donde su sueldo promedio era de aproximadamente $\$72,000$ con una desviación estándar de diferencia ($\$35,185$). Esto se refleja en el histograma y la amplia gama de resultados obtenidos, concentrándose entre los $\$65,000$ y $\$85,000$, con un salario máximo de $\$150,000$.

Por el lado de los hombres, la media fue de 37 años, a pesar de que la muestra se concentró más entre los 30 y 40 años, en general se ve una distribución más acorde respecto al contexto planteado, ya que se trata de anuncios de una empresa de automóviles; respecto al salario de los hombres, hay más diferencias respecto a las mujeres, con un promedio salarial de aprox. $\$68,000$ a pesar de reflejar un salario mínimo mayor, esto se debe de igual forma a como se distribuyeron los datos obtenidos, hay alrededor de 27 hombres que perciben más de $\$90,000$ de sueldo, los demás se concentran entre los $\$20,000$ hasta los $\$80,000$, razón por la que su desviación estándar es menor($\$33,179$).

Construcción del Modelo de Score

También es necesario **estandarizar** las variables para evitar problemas de escala con los datos numéricos (edad y el salario), es decir, se les va a restar su media y dividirlos entre su desviación estándar.

$$\mathbf{Z} = \frac{\mathbf{X} - \mu}{\sigma_{\mathbf{X}}}$$

Por otro lado, la variable del Género es categórica, así que se trabajará con el **WoE (weight of evidence)** que es un valor continuo en lugar de su valor original, por lo que hay que hallar la distribución de los clientes que si compran (1) y la distribución de los que no (0).

$$\mathbf{WoE} = \ln(\frac{\mathbf{Distr(1)}}{\mathbf{Distr(0)}})$$

```
# Peso de Evidencia
genero<-datos_train %>%
            group_by(Gender) %>%
            summarise(Compran=sum(Purchased==1),
                      NoCompran=sum(Purchased==0)) %>%
            mutate(Distr_Compran=Compran/sum(Compran)) %>%
            mutate(Distr_NoCompran=NoCompran/sum(NoCompran)) %>%
            mutate(WoE = log(Distr_Compran/Distr_NoCompran))
WoE_genero<-genero$WoE
```

```
# Cambiar el nombre a la columna del Salario
colnames(datos_train)[4]<-c("Salary")

# Estandarizar las Variables y crear una columna que asigna el
# valor del WoE para el género
datos_train<-datos_train %>%
mutate(Age_Z = (Age - mean(Age))/(sd(Age))) %>%
mutate(Salary_Z = (Salary - mean(Salary))/(sd(Salary))) %>%
mutate(Genero=ifelse(Gender=="Male",WoE_genero[2]
,WoE_genero[1]))

kable(head(datos_train,10),digits = 4,align = "ccccccccccc")
```

User ID	Gender	Age	Salary	Purchased	Sample	Age_Z	Salary_Z	Genero
15624510	Male	19	19000	0	Train	-1.7941	-1.4950	-0.0566
15810944	Male	35	20000	0	Train	-0.2571	-1.4657	-0.0566
15668575	Female	26	43000	0	Train	-1.1217	-0.7939	0.0526
15603246	Female	27	57000	0	Train	-1.0256	-0.3850	0.0526
15728773	Male	27	58000	0	Train	-1.0256	-0.3558	-0.0566
15598044	Female	27	84000	0	Train	-1.0256	0.4036	0.0526
15694829	Female	32	150000	1	Train	-0.5453	2.3314	0.0526
15600575	Male	25	33000	0	Train	-1.2177	-1.0860	-0.0566
15727311	Female	35	65000	0	Train	-0.2571	-0.1513	0.0526
15570769	Female	26	80000	0	Train	-1.1217	0.2868	0.0526

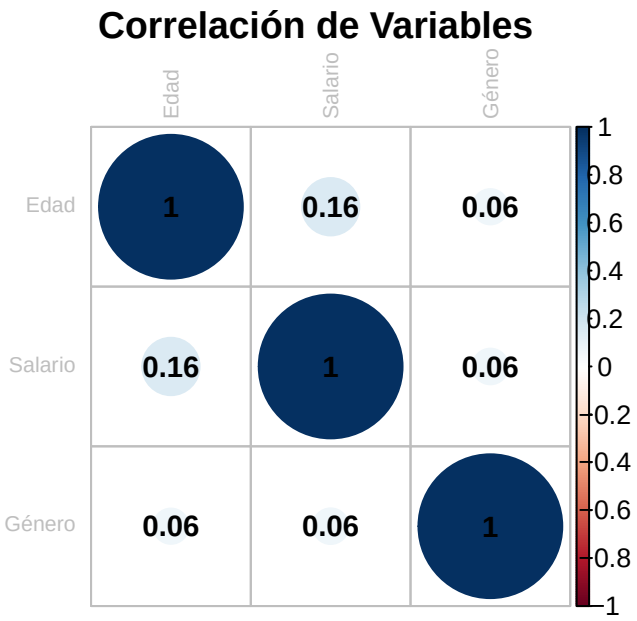
Modelo Lineal Generalizado

Antes de proponer algún modelo hay que verificar el grado de correlación que tienen las variables, esto se hará mediante un análisis de la matriz de correlación y de un gráfico con la librería **corrplot**.

```
# Matriz de Correlación
matriz_correlacion<-cor(datos_train[,7:9])
colnames(matriz_correlacion) <- c("Edad", "Salario", "Género")
rownames(matriz_correlacion) <- c("Edad", "Salario", "Género")
matriz_correlacion
```

```
##                Edad    Salario     Género
## Edad    1.00000000 0.15916773 0.06017282
## Salario 0.15916773 1.00000000 0.06145001
## Género  0.06017282 0.06145001 1.00000000
```

```
# Gráfica de Correlación de Variables
corrplot(matriz_correlacion,
         addCoef.col = "black",    # Muestra coeficientes en negro
         number.cex = 0.9,         # Tamaño de los coeficientes
         tl.col = "grey",          # Color de etiquetas
         tl.cex = 0.70,             # Tamaño de etiquetas
         cl.pos = "r",             # Ubicación de la leyenda
         title = "Correlación de Variables",
         mar=c(0,0,1,0)
)
```



Como se pudo observar tanto en la matriz como en la gráfica, los valores de la correlación de las variables de Edad y Salario con el Género son muy cercanos a cero, lo cual indica que su relación es casi nula. Además, el Salario y la Edad si presentan un relación positiva (de forma proporcional) , pero es muy débil, por lo que se puede continuar con la proposición del modelo.

Usando la función **glm** se hallarán los coeficientes de las variables y el intercepto (betas) que servirán para encontrar el Score.

$$\mathbf{Score} = \beta_\mathbf{0} + \beta_\mathbf{1}\mathbf{X_1} + ... + \beta_\mathbf{k}\mathbf{X_k}$$

```
# Primer Modelo Considerando las tres variables
mod1<-glm(datos_train$Purchased~datos_train$Age_Z+
                                datos_train$Salary_Z+
                                datos_train$Genero,
                                data=datos_train,
                                family="binomial")
summary(mod1)
```

```
##
## Call:
## glm(formula = datos_train$Purchased ~ datos_train$Age_Z + datos_train$Salary_Z +
##     datos_train$Genero, family = "binomial", data = datos_train)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.2296     0.1932  -6.363 1.98e-10 ***
## datos_train$Age_Z      2.3406     0.2918   8.023 1.03e-15 ***
## datos_train$Salary_Z   1.1986     0.2018   5.941 2.84e-09 ***
## datos_train$Genero    -3.2928     3.1374  -1.050    0.294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 406.32  on 314  degrees of freedom
## Residual deviance: 219.68  on 311  degrees of freedom
## AIC: 227.68
##
## Number of Fisher Scoring iterations: 6
```

El primer modelo arroja que la variable del Género no es significativa (el valor-p es muy alto), entonces se va a crear un segundo modelo para observar qué sucede.

```
mod<-glm(datos_train$Purchased~datos_train$Age_Z+
                               datos_train$Salary_Z,
                               data=datos_train,
                               family="binomial")
summary(mod)
```

```
##
## Call:
## glm(formula = datos_train$Purchased ~ datos_train$Age_Z + datos_train$Salary_Z,
##     family = "binomial", data = datos_train)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.2126     0.1914  -6.337 2.34e-10 ***
## datos_train$Age_Z      2.3061     0.2865   8.049 8.35e-16 ***
## datos_train$Salary_Z   1.1812     0.2004   5.895 3.75e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 406.32  on 314  degrees of freedom
## Residual deviance: 220.79  on 312  degrees of freedom
## AIC: 226.79
##
## Number of Fisher Scoring iterations: 6
```

Podemos concluir que las variables de edad y salario tienen un impacto en el modelo y son predictores significativos en la variable respuesta, por lo que el modelo se ajusta bien, entonces se quitará la variable de género y se procederá a extraer los coeficientes que arrojó la función.

```
# Coeficientes del modelo
coef<-coef(mod)
coef
```

```
##          (Intercept)    datos_train$Age_Z datos_train$Salary_Z
##            -1.212615             2.306135             1.181198
```

Respecto a los coeficientes: El intercepto tiene un valor estimado de -1.2126, lo que significa que cuando todas las variables predictoras son cero, el logaritmo de la probabilidad de que ocurra el evento es -1.2126. La edad tiene un coeficiente de 2.3061, lo que indica que un aumento de una unidad en esta variable aumenta el logaritmo de la probabilidad de que ocurra el evento en 2.3061. Los salarios tienen un coeficiente de 1.1812, lo que sugiere que un aumento de una unidad en esta variable aumenta el logaritmo de la probabilidad de que ocurra el evento en 1.1812.

Regresión Logística

Lo que sigue es calcular el score con las betas obtenidas por el modelo y usar la regresión logística para mapear a la probabilidad con valores entre 0 y 1.

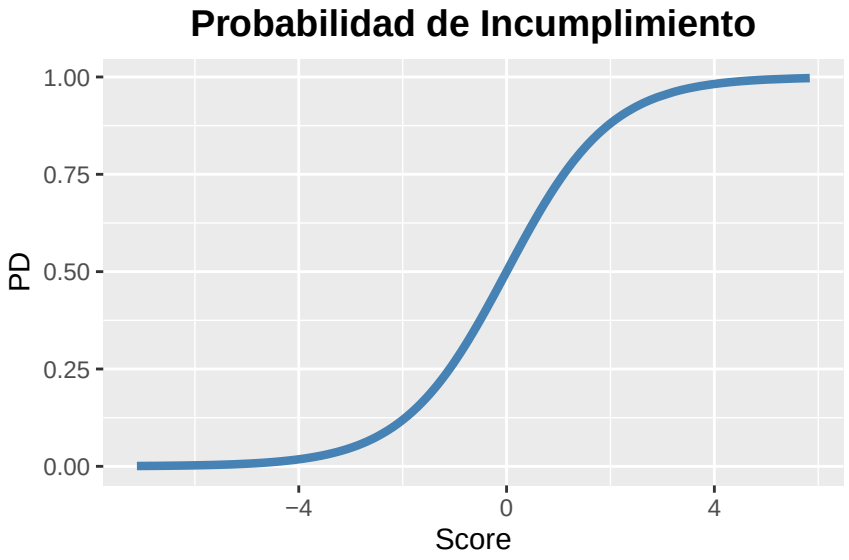
$$\mathbf{y} = \frac{\mathbf{1}}{\mathbf{1} + \mathbf{e}^{-\mathbf{score}}}$$

```
# Cambiar nombre a columna de los ID
colnames(datos_train)[1]<-c("UserID")

# Calcular el Score y la PD
datos_train<-datos_train %>%
mutate(Score=coef[1]+coef[2]*Age_Z+coef[3]*Salary_Z) %>%
arrange(UserID) %>%
mutate(PD=1/(1+exp(-1*Score)))

kable(head(datos_train,5),digits = 4,align = "ccccccccccc")
```

UserID	Gender	Age	Salary	Purchased	Sample	Age_Z	Salary_Z	Genero	Score	PD
15566689	Female	35	57000	0	Train	-0.2571	-0.3850	0.0526	-2.2603	0.0945
15570769	Female	26	80000	0	Train	-1.1217	0.2868	0.0526	-3.4606	0.0305
15570932	Male	34	115000	0	Train	-0.3532	1.3091	-0.0566	-0.4807	0.3821
15571059	Female	33	41000	0	Train	-0.4492	-0.8524	0.0526	-3.2554	0.0371
15573452	Female	21	16000	0	Train	-1.6020	-1.5826	0.0526	-6.7764	0.0011

Graficando el Score obtenido en el eje x y la Probabilidad de Incumplimiento en el eje y se llega a una función *sigmoide* en donde:"A mayor score, mayor probabilidad de que compre".

```
ggplot(data=datos_train,aes(x=Score,y=PD)) +
  geom_line(col="steelblue",lwd=1.5) +
  labs(title = "Probabilidad de Incumplimiento", x="Score", y="PD")+
theme(plot.title = element_text(hjust = 0.5, face = "bold",size = 14))
```



Score Agrupado

Se crea un nuevo data frame para hallar los intervalos de la agrupación del score, pero antes de eso se va a transformar el Score calculado en el paso anterior.

```
# Nuevo Data Frame
score_agrup<-datos_train %>%
             select(UserID,Purchased,Score)
```

El proceso a realizar es multiplicar el Score por una cosntante negativa, sin embargo, debido al contexto de problema, cambiará la interpretación de los valores de la siguiente manera: "Un menor score, indica una probailidad más alta de que el cliente haga la compra". El valor elegido como **Constante de Escalado** fue $\mathbf{a = -8}$.

```
# Score Escalado
a<- -8
score_agrup<-score_agrup %>%
             mutate(Score_S = Score * a)
```

Sin embargo, aún se observa que hay valores del Score con signo negativo. Para corregir esto, se van a desplazar sumando el mínimo del Score Escalado de modo que la puntuación de los clientes inicie en cero.

```
desplazamiento<-min(score_agrup$Score_S) # es un valor negativo

# Score Positivo
score_agrup<-score_agrup %>%
             mutate(Score_Positivo = Score_S + abs(desplazamiento))
```

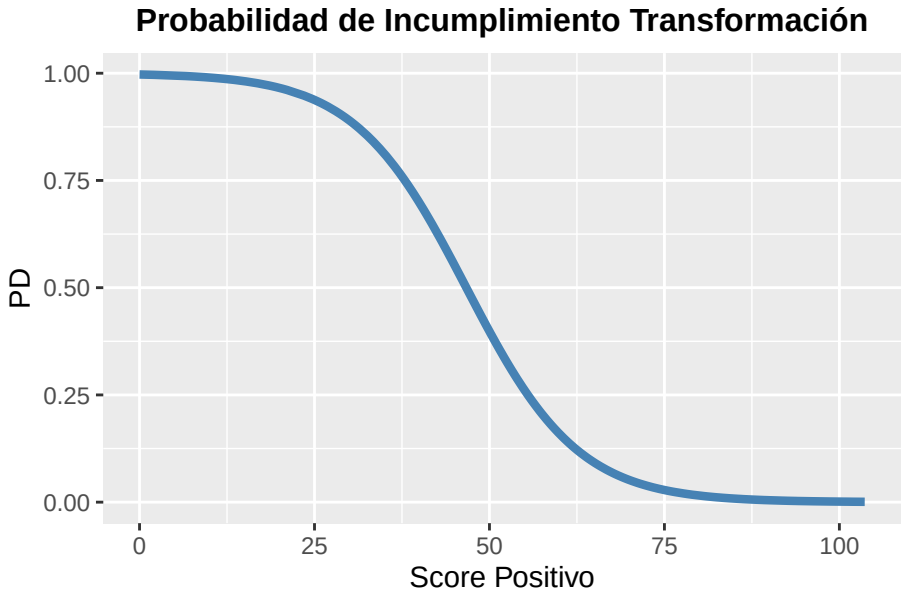
Para verificar que usando estos nuevos valores del Score no se altera la Probabilidad de Incumplimiento, se volverá a calcular en otra columna usando la regresión logística.

```
# PD
score_agrup<-score_agrup %>%
             mutate(PD =1/(1+ exp(-1*(Score_Positivo-abs(desplazamiento))/a)))
# Primero se le resta el desplazamiento y después se divide entre
# la Constante de Escalado antes de hacer la regresión logística

kable(head(score_agrup,8),align = "cccccc")
```

UserID	Purchased	Score	Score_S	Score_Positivo	PD
15566689	0	-2.2602634	18.08211	64.77648	0.0944678
15570769	0	-3.4605748	27.68460	74.37897	0.0304551
15570932	0	-0.4806975	3.84558	50.53995	0.3820874
15571059	0	-3.2553709	26.04297	72.73734	0.0371344
15573452	0	-6.7763871	54.21110	100.90547	0.0011391
15574305	0	-2.3982707	19.18617	65.88054	0.0833047
15574372	1	2.4901209	-19.92097	26.77341	0.9234463
15575002	0	-2.1567580	17.25406	63.94844	0.1037014

Si se realiza la misma gráfica mostrada anteriormente, ahora se puede notar que un Score cercano a cero indica una alta Probabilidad de Compra debido a la transformación realizada.

```
ggplot(data=score_agrup,aes(x=Score_Positivo,y=PD)) +
  geom_line(col="steelblue",lwd=1.5) +
  labs(
    title="Probabilidad de Incumplimiento Transformación", # TÍTULO
    x="Score Positivo",                                    # EJE X
    y="PD") +                                              # EJE Y
  theme(
    plot.title = element_text(hjust = 0.5,
                              face = "bold",size = 12))
```



Intervalos de Agrupación

El siguiente paso es agrupar en ciertos intervalos todos los scores positivos que se obtuvieron, para este proceso se estableció una cantidad de 12 intervalos que tienen la misma longitud y se le asignó a cada valor del Score Positivo el extremo en donde cae.

```
# Obtener los intervalos
minimo<-min(score_agrup$Score_Positivo)
maximo<-max(score_agrup$Score_Positivo)
num_intervalos<-12

intervalos<-seq(minimo,maximo,length.out=num_intervalos)
intervalos
```

```
##  [1]   0.000000   9.420187  18.840374  28.260561  37.680747  47.100934
##  [7]  56.521121  65.941308  75.361495  84.781682  94.201869 103.622056
```

Luego se cuenta el número de valores que tiene cada intervalo, mejor dicho, hallar su distribución.Se van a usar las funciones **cut() y count()** que hacen lo mismo que la función **FRECUENCIA** de excel.

```
# Es necesario Agregar el -Infinito para contar valores menores o
# iguales que el primer valor de los intervalos planteados
intervalos_nuevos<-c(-Inf,intervalos)

distribucion<-score_agrup %>%
              mutate(Rango=cut(Score_Positivo,breaks=intervalos_nuevos,right = TRUE)) %>%
              count(Rango)
kable(distribucion,align = "cc")
```

Rango	n
(-Inf,0]	1
(0,9.42]	6
(9.42,18.8]	12
(18.8,28.3]	19
(28.3,37.7]	23
(37.7,47.1]	34
(47.1,56.5]	69
(56.5,65.9]	47
(65.9,75.4]	38
(75.4,84.8]	33
(84.8,94.2]	23
(94.2,104]	10

Finalmente, se va a hacer uso de la función **findInterval**, la cual es quivalente a un **BUSCARV** de excel para asignar el Score Agrupado dependiendo el lugar en donde cae de los intervalos creados.

```
# Score Agrupado
score_agrup<-score_agrup %>%
             mutate(Score_Agrupado = intervalos[findInterval(Score_Positivo,intervalos)])

kable(head(score_agrup,10),align ="ccccccc",digits = 6)
```

UserID	Purchased	Score	Score_S	Score_Positivo	PD	Score_Agrupado
15566689	0	-2.260263	18.08211	64.77648	0.094468	56.52112
15570769	0	-3.460575	27.68460	74.37897	0.030455	65.94131
15570932	0	-0.480697	3.84558	50.53995	0.382087	47.10093
15571059	0	-3.255371	26.04297	72.73734	0.037134	65.94131
15573452	0	-6.776387	54.21110	100.90547	0.001139	94.20187
15574305	0	-2.398271	19.18616	65.88054	0.083305	56.52112
15574372	1	2.490121	-19.92097	26.77341	0.923446	18.84037
15575002	0	-2.156758	17.25406	63.94844	0.103701	56.52112
15575694	0	-1.708234	13.66587	60.36025	0.153393	56.52112
15576219	0	-1.345050	10.76040	57.45477	0.206681	56.52112

Para ver si con esos intervalos propuestos se obtienen valores tanto de los clientes que si compran como de los que no, se realiza un conteo por Score Agrupado y tipo de cliente

```
# Conteo
conteo<- score_agrup %>%
         group_by(Score_Agrupado) %>%
         summarise(Compran = sum(Purchased==0),
         No_Compran=sum(Purchased==1))
kable(conteo,align = "ccc")
```

Score_Agrupado	Compran	No_Compran
0.000000	0	7
9.420187	3	9
18.840374	2	17
28.260561	2	21
37.680748	10	24
47.100934	41	28
56.521121	45	2
65.941308	37	1
75.361495	33	0
84.781682	23	0
94.201869	9	0
103.622056	1	0

Existen zonas donde hay cero clientes que no compran o cero clientes que si compran, por lo que se va a reducir el número de intervalos (de 12 a 7) juntando los valores de los extremos con otro intervarlo para arreglar esto, mejor dicho:

```
# Modificación de Intervalos
intervalos<-intervalos[-c(1,9:12)]
intervalos
```

```
## [1]  9.420187 18.840374 28.260561 37.680747 47.100934 56.521121 65.941308
```

Esta modificación ocasiona que se vuelvan a asignar el Score Agrupado.

```
# Score Agrupado
score_agrup<-score_agrup %>%
             mutate(Score_Agrupado=ifelse(
               findInterval(Score_Positivo,intervalos)==0,1,
               findInterval(Score_Positivo,intervalos))) %>%
            mutate(Score_Agrupado=intervalos[Score_Agrupado])
kable(head(score_agrup,4),align ="ccccccc",digits = 4)
```

UserID	Purchased	Score	Score_S	Score_Positivo	PD	Score_Agrupado
15566689	0	-2.2603	18.0821	64.7765	0.0945	56.5211
15570769	0	-3.4606	27.6846	74.3790	0.0305	65.9413
15570932	0	-0.4807	3.8456	50.5400	0.3821	47.1009
15571059	0	-3.2554	26.0430	72.7373	0.0371	65.9413

Una vez aplicadas las correcciones se puede observar que en todas las agrupaciones del Score se captan tanto clientes que compran como los que no.

```
conteo<- score_agrup %>%
         group_by(Score_Agrupado) %>%
         summarise(Compran = sum(Purchased==1),
         No_Compran=sum(Purchased==0),
         Total = Compran + No_Compran)
kable(conteo,align ="cccc")
```

Score_Agrupado	Compran	No_Compran	Total
9.420187	16	3	19
18.840374	17	2	19
28.260561	21	2	23
37.680748	24	10	34
47.100934	28	41	69
56.521121	2	45	47
65.941308	1	103	104

Reporte Bad Rate

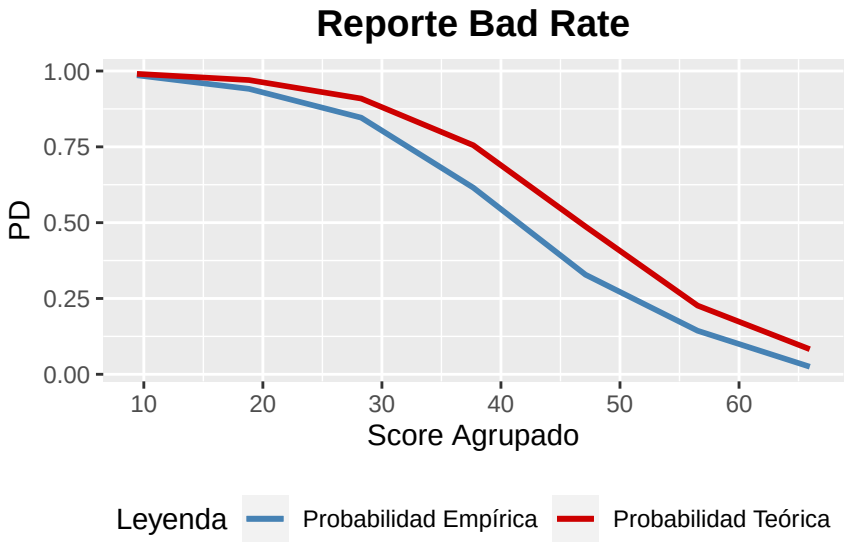
El objetivo es comparar el promedio de las Probabilidades **Empíricas** (obtenidas con un promedio de la PD por intervalos de agrupación) con las calculadas de manera **Teórica** (regresión logística del Score Agrupado) y observar la distancia que hay entre ambas curvas.

```
# PD Empírica y Teórica
bad_rate<-score_agrup %>%
          group_by(Score_Agrupado) %>%
          summarise(PD_Empirica=mean(PD,na.rm = TRUE)) %>%
          mutate(PD_Teorica=1/(1+ exp(-1*(Score_Agrupado-abs(desplazamiento))/a)))
kable(bad_rate,align ="ccc",digits =6)
```

Score_Agrupado	PD_Empirica	PD_Teorica
9.420187	0.985983	0.990616
18.840374	0.941151	0.970164
28.260561	0.846143	0.909226
37.680747	0.615260	0.755230
47.100934	0.328400	0.487298
56.521121	0.143595	0.226472
65.941308	0.024959	0.082726

Como resultado se genera una gráfica donde es posible identificar que la distancia de separación entre las dos probabilidades.

```
# Gráfica de PD Empírica vs Teórica
ggplot(data=bad_rate,aes(x=Score_Agrupado))+
geom_line(aes(y=PD_Empirica,color="Probabilidad Empírica"),lwd=1)+
geom_line(aes(y=PD_Teorica,color="Probabilidad Teórica"),lwd=1) +
  labs(title="Reporte Bad Rate",x="Score Agrupado", y="PD", color="Leyenda") +
scale_color_manual(values=c("Probabilidad Empírica"="steelblue",
                            "Probabilidad Teórica" = "red3"))+
  theme(plot.title = element_text(hjust = 0.5, face = "bold",size = 14),
legend.position = "bottom")
```



Validación del Modelo

La siguiente parte es validar el modelo, por lo que es necesario crear un data frame que contenga el número de clientes que si compran y no compran por cada Score Agrupado con el objetivo de construir la **Densidad** de ambos, así como su **Distribución Acumulada** y el **Peso de Evidencia**.

```
# Data Frame
validacion<-score_agrup %>%
# Agrupar
group_by(Score_Agrupado) %>%
# Contar el número de clientes que si compran o no por score
summarise(Compran=sum(Purchased==1),NoCompran=sum(Purchased==0)) %>%
# Columma del Total
mutate(Total = Compran + NoCompran) %>%
# Tasa de los clientes que No Compran
mutate(Tasa_NoCompran = NoCompran / Total) %>%
# Densidad de los clientes que Compran
mutate(Distr_Compran= Compran / sum(Compran)) %>%
# Densidad de los clientes que No Compran
mutate(Distr_NoCompran= NoCompran / sum(NoCompran)) %>%
# Densidad del Total
mutate(Distr_Total= Total / sum(Total)) %>%
# Distribución Acumulada de que Compran
mutate(Acum_Compran=cumsum(Distr_Compran)) %>%
# Distribución Acumulada de los que No Compran
mutate(Acum_NoCompran=cumsum(Distr_NoCompran)) %>%
# Distribución Acumulada del Total
mutate(Acum_Total=cumsum(Distr_Total)) %>%
# WoE: peso de evidencia
mutate(WoE = log(Distr_Compran / Distr_NoCompran))

kable(head(validacion[,-c(2,3,4,5,8,11)],8),align ="cccccc" ,digits =4)
```

Score_Agrupado	Distr_Compran	Distr_NoCompran	Acum_Compran	Acum_NoCompran	WoE
9.4202	0.1468	0.0146	0.1468	0.0146	2.3105
18.8404	0.1560	0.0097	0.3028	0.0243	2.7766
28.2606	0.1927	0.0097	0.4954	0.0340	2.9879
37.6807	0.2202	0.0485	0.7156	0.0825	1.5120
47.1009	0.2569	0.1990	0.9725	0.2816	0.2552
56.5211	0.0183	0.2184	0.9908	0.5000	-2.4770
65.9413	0.0092	0.5000	1.0000	1.0000	-3.9982

Se realizarán cuatro pruebas para revisar la calidad del modelo utilizando los datos de la tabla creada en el paso anterior.

Prueba de KS

Es un estadístico cuyo objetivo es medir el poder predictivo de los sistemas de clasificación, en este caso ayudará a determinar si las dos distribuciones acumuladas de los clientes difieren.

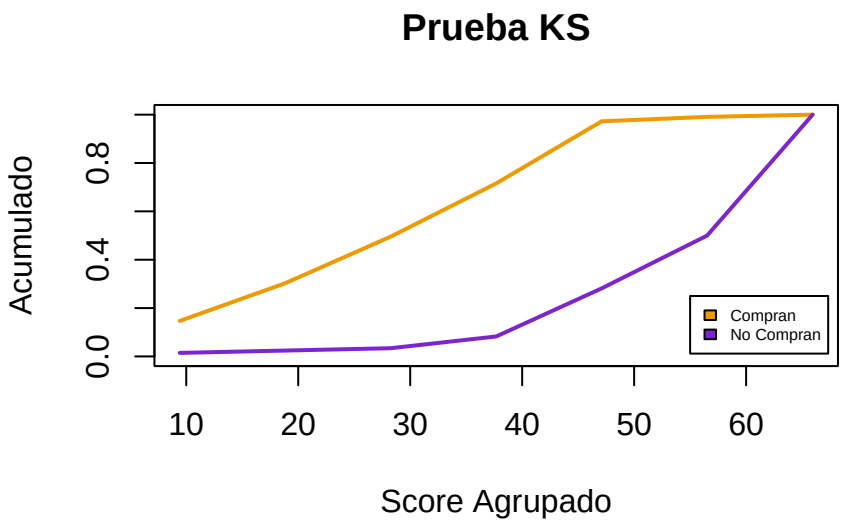
$$\mathbf{KS} = \mathbf{max}\{|\mathbf{cp}(\mathbf{Y}) - \mathbf{cp}(\mathbf{X})|\}$$

```
# Prueba Kolmogoru-Smirnov
pruebaKS<-validacion %>%
    select(Score_Agrupado,Acum_Compran,Acum_NoCompran) %>%
    mutate(Delta = abs(Acum_Compran - Acum_NoCompran))
kable(head(pruebaKS,9),align="cccc",digits=6)
```

Score_Agrupado	Acum_Compran	Acum_NoCompran	Delta
9.420187	0.146789	0.014563	0.132226
18.840374	0.302752	0.024272	0.278480
28.260561	0.495413	0.033981	0.461432
37.680747	0.715596	0.082524	0.633072
47.100934	0.972477	0.281553	0.690924
56.521121	0.990826	0.500000	0.490826
65.941308	1.000000	1.000000	0.000000

La siguiente gráfica muestra las dos distribuciones acumuladas

```
# Gráfica
plot(pruebaKS$Score_Agrupado,pruebaKS$Acum_Compran,type = "l",
     lwd=2,col="orange2",xlab="Score Agrupado",ylab ="Acumulado",
     main="Prueba KS",ylim=c(0,1))
lines(pruebaKS$Score_Agrupado,pruebaKS$Acum_NoCompran,type="l",
      lwd=2,col="purple3")
  legend(55,0.25,c("Compran","No Compran"),cex=0.5,
         fill=c("orange2","purple3"))
```



Una vez establecida la prueba, se puede sacar el **Punto de Corte del Score** como el Score Agrupado de la máxima de las distancias entre las acumuladas (delta)

```
# Identificar el máximo
delta_max<-pruebaKS %>%
             filter(Delta==max(Delta))
kable(delta_max,align="cccc",digits =5)
```

Score_Agrupado	Acum_Compran	Acum_NoCompran	Delta
47.10093	0.97248	0.28155	0.69092

```
# Score Agrupado correspondiente
punto_corte<-delta_max$Score_Agrupado
punto_corte
```

```
## [1] 47.10093
```

La interpretación que se le da a este valor es que si un cliente obtiene una puntuación menor al punto de corte, significa que el modelo lo va a clasificar como un Cliente que realizará la compra.

Índice de Gini

Este índice calcula el área entre la curva y la diagonal de la curva de Lorenz, cuando más alto sea el valor de este coeficiente más robusta será el modelo; uno sin discriminación tendría un Gini de Cero, mientras que uno perfecto tendría un Gini de 100.

El coeficiente de Gini se calcula como:

$$\mathbf{Gini} = \frac{\mathbf{A_T} - \mathbf{A_g}}{\mathbf{A_T}}$$

```
# Se hace la suma del valor actual más el anterior para los que si
# compran y la resta del valor actual menos el anterior para los que no
gini<-validacion %>%
select(Score_Agrupado,Acum_Compran, Acum_NoCompran) %>%
mutate(DeltaCompran=NA,DeltaNoCompran=NA,A1=NA)

for(k in 1:length(gini$Score_Agrupado)){
  if(k==1){
    gini$DeltaCompran[k]<-0
    gini$DeltaNoCompran[k]<-0
    gini$A1[k]<-0
  }else{
    gini$DeltaCompran[k]<-gini$Acum_Compran[k-1]+ gini$Acum_Compran[k]
    gini$DeltaNoCompran[k]<-gini$Acum_NoCompran[k]- gini$Acum_NoCompran[k-1]
    gini$A1[k]<-0.5 * gini$DeltaCompran[k] * gini$DeltaNoCompran[k]
    # Es el promedio de los dos resultados anteriores
  }}
kable(head(gini),align ="cccccc",digits =5)
```

Score_Agrupado	Acum_Compran	Acum_NoCompran	DeltaCompran	DeltaNoCompran	A1
9.42019	0.14679	0.01456	0.00000	0.00000	0.00000
18.84037	0.30275	0.02427	0.44954	0.00971	0.00218
28.26056	0.49541	0.03398	0.79817	0.00971	0.00387
37.68075	0.71560	0.08252	1.21101	0.04854	0.02939
47.10093	0.97248	0.28155	1.68807	0.19903	0.16799
56.52112	0.99083	0.50000	1.96330	0.21845	0.21444

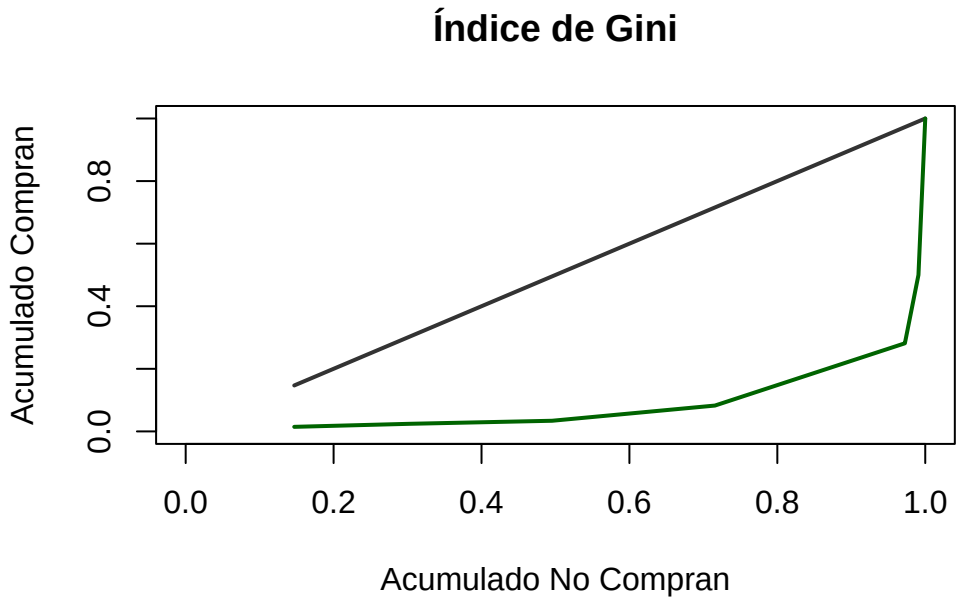
```
# Cálculo Índice de Gini
At<-0.5
Ag<-sum(gini$A1)  # Es la suma de los valores de la columna
indice_gini<-(Ag - At) / At
indice_gini
```

```
## [1] 0.8311659
```

Nota: El Ag salió mayor que el At (0.91), así que en la resta se invirtieron de orden para cambiar el signo del indicador. El valor obtenido es alto y muestra que el modelo puede llegar a funcionar correctamente (está por encima del 30% donde se establece que los modelos son buenos).

Visualmente se genera una gráfica en donde se puede observar la diagonal y el área inferior que se calculó con las fórmulas presentadas.

```
# Gráfica
plot(gini$Acum_Compran,gini$Acum_Compran,type = "l",
     lwd=2,col="grey20",xlab="Acumulado No Compran",
     ylab="Acumulado Compran",main="Índice de Gini",xlim=c(0,1),ylim =c(0,1))
lines(gini$Acum_Compran,gini$Acum_NoCompran,type="l",lwd=2,col="darkgreen")
```



Information Value (IV)

Mide el área entre dos distribuciones (buenos y malos), sus valores siempre serán positivos y se calcula como:

$$\mathbf{IV} = \sum_{\mathbf{i=1}}^{\mathbf{N}} [\frac{\mathbf{N_i}}{\sum \mathbf{N}} - \frac{\mathbf{P_i}}{\sum \mathbf{P}} * \mathbf{WoE_i}]$$

```
# Obtener el IV
information_value<-validacion %>%
select(Score_Agrupado,Acum_Compran,Acum_NoCompran,WoE) %>%
mutate(IV = WoE * (Acum_Compran - Acum_NoCompran))

kable(head(information_value,15),align ="ccccc",digits =6)
```

Score_Agrupado	Acum_Compran	Acum_NoCompran	WoE	IV
9.420187	0.146789	0.014563	2.310505	0.305509
18.840374	0.302752	0.024272	2.776594	0.773227
28.260561	0.495413	0.033981	2.987904	1.378715
37.680747	0.715596	0.082524	1.511997	0.957203
47.100934	0.972477	0.281553	0.255161	0.176297
56.521121	0.990826	0.500000	-2.476987	-1.215769
65.941308	1.000000	1.000000	-3.998201	0.000000

Solo queda sumar la columna del IV para obtener el valor final

```
IV<-sum(information_value$IV)
IV
```

```
## [1] 2.375182
```

Este valor se mueve entre 0 y 4, si se acerca a 4 indica que el modelo es bueno. Por otro lado, valores cercanos a 0 es que el modelo es malo. En este caso, se obtuvo como resultado 2.375, lo cual nos puede indicar que el modelo realizado es aceptable y es candidato a ser elegido como modelo de predicción.

Área bajo la curva ROC (AUROC)

Una curva ROC (curva de característica operativa del receptor) es un gráfico que muestra el rendimiento de un modelo de clasificación en todos los umbrales de clasificación. Esta curva representa dos parámetros: Tasa de Verdaderos Positivos y Tasa de Falsos Positivos.

AUROC significa "área bajo la curva ROC, se puede interpretar como la probabilidad de que el modelo clasifique un ejemplo positivo aleatorio más alto que un ejemplo negativo aleatorio, se calcula mediante:

$$\mathbf{AUROC} = \mathbf{0.5} * (\mathbf{Gini} + \mathbf{1})$$

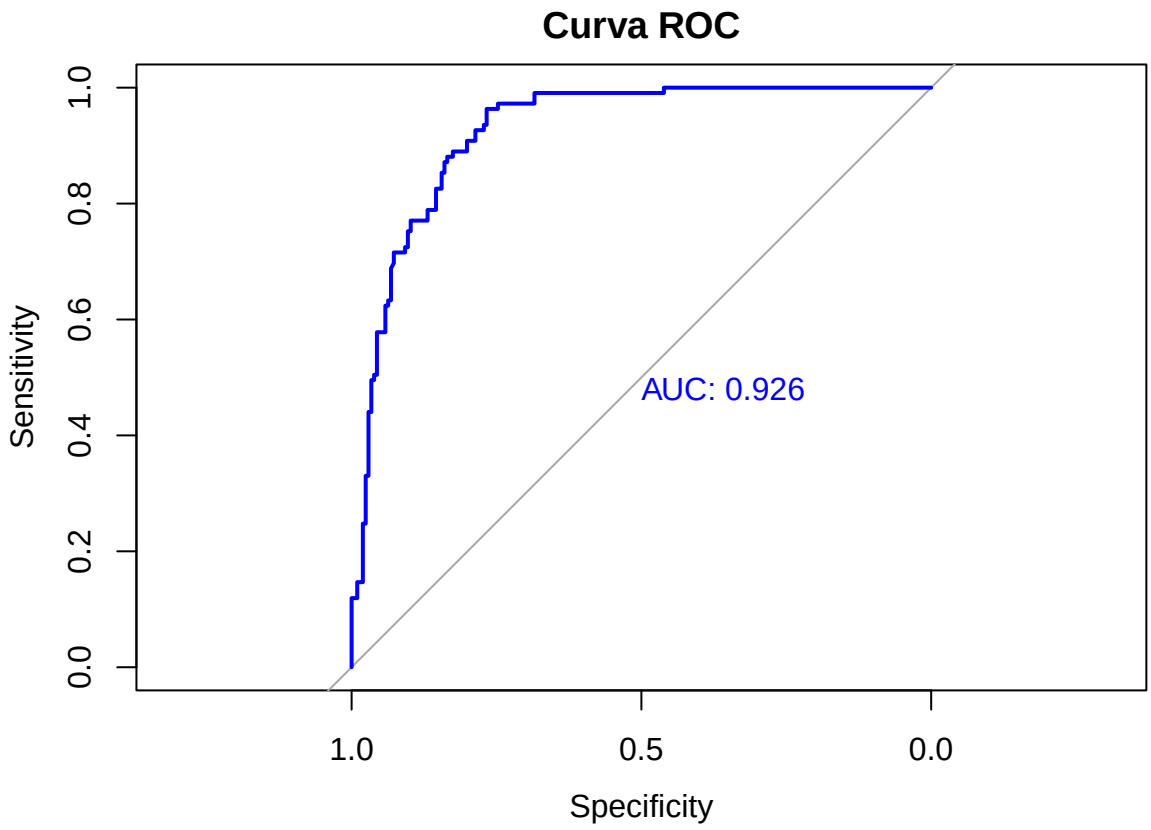
```
AUROC<-0.5*(indice_gini+1)
AUROC
```

```
## [1] 0.915583
```

El resultado es un valor que supera el 0.9, por lo que el modelo entra en la categoría de "bueno" si se toma en cuenta este parámetro

En R existe la librería **pROC**, la cual obtiene la curva ROC introduciendo los valores reales de Compra (variable Purchased) y las probabilidades calculadas con los coeficientes del modelo.

```
# Curva ROC
roc_curve <- roc(datos_train$Purchased,datos_train$PD)
plot(roc_curve, col = "blue", main = "Curva ROC", print.auc = TRUE)
```




```
# Graficar la curva ROC

auroc_valor <- auc(roc_curve)
auroc_valor
```

```
## Area under the curve: 0.9256
```

Se observa que este resultado no se aleja de lo que se obtuvo con la metodología mencionada con anterioridad, al contrario, la distancia ente los dos AUROC calculados es pequeña.

Conociendo este valor es posible calcular el Índice de Gini despejando de la fórmula, es decir:

```
gini_index <- (2 * auroc_valor) - 1
gini_index
```

```
## [1] 0.8511178
```

El resultado se parece mucho al que se obtuvo utilizándo el otro método.

Matriz de Confusión

El modelo estructurado en todos los pasos previos será utilizado nuevamente pero esta vez con los datos de prueba para construir una matriz de confusión, la cual indicará el desempeño del modelo comparando las predicciones con los valores reales. Lo primero es estandarizar las variables de edad y salario, y calcular el Score con los coeficientes establecidos.

```
# Cambiar el nombre a la columna del Salario
colnames(datos_test)[4]<-c("Salary")

# Estandarizar las Variables y crear una columna que asigna el valor del género

datos_test<-datos_test %>%
mutate(Age_Z = (Age - mean(Age))/(sd(Age))) %>%
mutate(Salary_Z = (Salary - mean(Salary))/(sd(Salary))) %>%
mutate(Genero=ifelse(Gender=="Male",WoE_genero[2]
,WoE_genero[1]))


# Cambiar nombre a columna de los ID
colnames(datos_test)[1]<-c("UserID")

# Calcular el Score
datos_test<-datos_test %>%
mutate(Score=coef[1]+coef[2]*Age_Z+coef[3]*Salary_Z) %>%
arrange(UserID)

kable(head(datos_test,10),align="ccccccccccc",digits=3)
```

UserID	Gender	Age	Salary	Purchased	Sample	Age_Z	Salary_Z	Genero	Score
15569641	Female	58	95000	1	Test	1.889	0.797	0.053	4.085
15573926	Male	40	71000	1	Test	0.224	0.085	-0.057	-0.595
15575247	Female	48	131000	1	Test	0.964	1.864	0.053	3.213
15577806	Male	41	87000	1	Test	0.317	0.560	-0.057	0.179
15578006	Male	23	63000	0	Test	-1.348	-0.152	-0.057	-4.501
15579212	Male	39	77000	0	Test	0.132	0.263	-0.057	-0.598
15582066	Male	40	78000	0	Test	0.224	0.293	-0.057	-0.350
15582492	Male	28	123000	1	Test	-0.886	1.627	-0.057	-1.333
15584320	Female	37	146000	1	Test	-0.053	2.309	0.053	1.392
15586996	Female	41	72000	0	Test	0.317	0.115	0.053	-0.346

Con el propósito de clasificar a los clientes se va a comparar el Score Positivo (multiplicado por la constante y desplazado) con el punto de corte obtenido en la **Prueba KS**, en caso de que lo superen significa que el cliente No Compra y se le asigna un 0, de lo contrario quiere decir que el Score del cliente es menor y Compra, por lo que un 1 es asignado en la columna de "Predicción",

```
# Clasificación según el Punto de Corte
datos_test<-datos_test %>%
            mutate(Score_Escalado = Score * a,
            Score_Positivo = Score_Escalado+abs(desplazamiento),
            Predicción = ifelse(Score_Positivo<punto_corte,1,0))
```

```r
kable(head(datos_test[,-c(2,3,4,6,9)],10),align="cccccccc",digits=3)
```

UserID	Purchased	Age_Z	Salary_Z	Score	Score_Escalado	Score_Positivo	Predicción
15569641	1	1.889	0.797	4.085	-32.678	14.016	1
15573926	1	0.224	0.085	-0.595	4.758	51.453	0
15575247	1	0.964	1.864	3.213	-25.702	20.992	1
15577806	1	0.317	0.560	0.179	-1.431	45.264	1
15578006	0	-1.348	-0.152	-4.501	36.006	82.700	0
15579212	0	0.132	0.263	-0.598	4.783	51.478	0
15582066	0	0.224	0.293	-0.350	2.797	49.491	0
15582492	1	-0.886	1.627	-1.333	10.664	57.359	0
15584320	1	-0.053	2.309	1.392	-11.136	35.559	1
15586996	0	0.317	0.115	-0.346	2.772	49.466	0

Sigue hallar la matriz de Confusión, por lo que hay que calcular 4 parámetros mediante un ciclo for, estos son:

Verdaderos Positivos: La predicción del modelo y la clasificación real coinciden en que el cliente Compra (se obtiene un 1 en ambos casos).

Verdaderos Negativos: La predicción del modelo y la clasificación real coinciden en que el cliente No Compra (se obtiene un 0 en ambos casos).

Falsos Positivos: La predicción del modelo es que si Compra y la clasificación real es que el cliente No Compra (Error Tipo I).

Falsos Negativos: La predicción del modelo es que No Compra y la clasificación real es que el cliente Compra (Error Tipo II).

```r
# Contador de los Valores (inician todos en cero)
VP<-0
VN<-0
FP<-0
FN<-0

# Ciclo For para contar todos los casos
for(k in 1:length(datos_test$UserID)){
  if(datos_test$Predicción[k]==1 & datos_test$Purchased[k]==1){
    VP<-VP + 1
  }else if(datos_test$Predicción[k]==0&datos_test$Purchased[k]==0){
    VN<-VN + 1
  }else if(datos_test$Predicción[k]==1&datos_test$Purchased[k]==0){
    FP<-FP+1
  }else{
    FN<-FN+1
  }
}
```

La matriz se contruye acomodando los valores obtenidos en el paso anterior

```
matriz_confusion<-matrix(data=NA,nrow=2,ncol = 2)
matriz_confusion[1,1]<-VP
matriz_confusion[1,2]<-FP
matriz_confusion[2,1]<-FN
matriz_confusion[2,2]<-VN

rownames(matriz_confusion)<-c("Predicción Compran",
                              "Predicción No Compran")
colnames(matriz_confusion)<-c("Compran","No Compran")
matriz_confusion
```

```
##                         Compran No Compran
## Predicción Compran           21          1
## Predicción No Compran        13         50
```

Finalmente, se calculan algunas métricas que evalúan el desempeño del modelo, las cuales son:

```
# Accuracy (Exactitud)
accuracy <- (VP + VN) / sum(matriz_confusion)
# Tasa de Verdaderos Positivos (Sensibilidad)
tasa_VP <- VP / (VP + FN)
# Tasa de Verdaderos Negativos (Especificidad)
tasa_VN <- VN / (VN + FP)
# Tasa de Falsos Positivos
tasa_FP <- FP / (FP + VN)
# Tasa de Falsos Negativos
tasa_FN <- FN / (FN + VP)
# Error
error <- 1 - accuracy
# Random Accuracy
Random_Accuracy <- ((VP + FN) * (VP + FP) + (VN + FP) * (VN + FN)) / (sum(matriz_confusion)^2)
# Acurracy Ratio
AR <- (accuracy - Random_Accuracy) / (1 - Random_Accuracy)

# Mostrar resultados
resultados <- data.frame(
  Métricas = c("Accuracy", "Tasa VP", "Tasa VN", "Tasa FP", "Tasa FN", "Error","Accuracy Ratio"),
  Valores = c(accuracy, tasa_VP, tasa_VN, tasa_FP, tasa_FN, error,AR))

kable(resultados,align ="cc",digits =6)
```

Métricas	Valores
Accuracy	0.835294
Tasa VP	0.617647
Tasa VN	0.980392
Tasa FP	0.019608
Tasa FN	0.382353
Error	0.164706
Accuracy Ratio	0.635417

La exactitud del modelo indica que las predicciones son correctas el 83.52 % de las veces **(accuracy)** manteniendo un **error** de 16.47%. Otras medidas como la especificidad indican un alto porcentaje de 98%.04% para poder clasificar adecuadamente los casos donde los clientes No Compran **(tasa VN)**. En adición a esto, se obtuvo un valor adecuado de 0.0196 para la **tasa FN**, la cual representa los casos reales donde el Cliente No Compró, pero el modelo lo clasificó incorrectamente.

Sin embargo, hay que tener en cuenta otros dos resultados de la matriz donde el modelo presenta más fallas, el primero es la sensibilidad de 61.76% que mide el porcentaje de casos reales donde el cliente Compró y fue clasificado de igual manera **(tasa VP)**, este sería bueno que fuera un poco más alto para identificar mejor los Clientes que compran. El segundo valor es la **tasa FP** donde están los casos de clientes que si compraron pero fueron clasificados erróneamente como negativos, como ajuste podría buscarse que este valor disminuya.

Finalmente el Accuracy Ratio fue de 63.54%, esta es una medida del desempeño del modelo en comparación con un modelo aleatorio, el valor obtenido se acerca más a 1 que a 0, por lo que sugiere que el modelo es mejor que una clasificación aleatoria.

Como otra alternativa, con la librería **caret** de R se llega al mismo resultado para calcular la Matriz de Confusión.

```
# Se establece al 1 como la clase positiva
matriz_confusion<- confusionMatrix(
factor(datos_test$Predicción,levels = c(1, 0)), factor(datos_test$Purchased,levels = c(1, 0)))
rownames(matriz_confusion$table)<-c("Predicción Compran",
                              "Predicción No Compran")
colnames(matriz_confusion$table)<-c("Compran","No Compran")
matriz_confusion
```

```
## Confusion Matrix and Statistics
##
##                       Reference
## Prediction              Compran No Compran
##   Predicción Compran         21          1
##   Predicción No Compran      13         50
##
##                Accuracy : 0.8353
##                  95% CI : (0.7391, 0.9069)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 2.52e-06
##
##                   Kappa : 0.6354
##
##  Mcnemar's Test P-Value : 0.003283
##
##             Sensitivity : 0.6176
##             Specificity : 0.9804
##          Pos Pred Value : 0.9545
##          Neg Pred Value : 0.7937
##              Prevalence : 0.4000
##          Detection Rate : 0.2471
##    Detection Prevalence : 0.2588
##       Balanced Accuracy : 0.7990
##
##        'Positive' Class : 1
##
```

Conclusión

El presente análisis permitió desarrollar un modelo de regresión logística para predecir la compra de clientes en función de variables como edad, salario y género. Con base en los resultados obtenidos, se pueden destacar las siguientes conclusiones:

En primer lugar, se determinó que las variables edad y salario son altamente significativas en la predicción de compras, mientras que la variable género no presentó un impacto estadísticamente relevante, motivo por el cual fue excluida en la versión final del modelo.

El poder predictivo del modelo fue evaluado mediante diversos indicadores. Se obtuvo un índice de Gini de 83.1% y un área bajo la curva ROC de 91.5 %, lo que demuestra una alta capacidad de discriminación. Asimismo, la prueba de Kolmogorov-Smirnov permitió determinar un punto de corte óptimo de 47.1, contribuyendo a mejorar la segmentación entre clientes compradores y no compradores.

Durante la validación, se observó que el modelo clasificó correctamente 21 clientes que compraron (Verdaderos Positivos) y 50 clientes que no compraron (Verdaderos Negativos), mientras que se identificaron 13 falsos negativos y solo 1 falso positivo. Estos resultados confirman que el modelo tiene un buen desempeño en la clasificación de clientes y un bajo margen de error.

Enfocando más en su desempeño, el modelo logró un porcentaje de precisión del 83.5 %, lo que refleja una adecuada capacidad de clasificación. Asimismo, la tasa de verdaderos positivos fue del 61.76 %, lo que implica que el modelo identifica correctamente una proporción considerable de los clientes que efectúan una compra. Por otro lado, la tasa de falsos positivos fue del 1.96 %, minimizando así la probabilidad de clasificar erróneamente a clientes que no realizarían una compra.

En términos de aprendizaje, durante el desarrollo de este proyecto permitió reforzar conocimientos clave en la construcción y evaluación de modelos de clasificación. Se comprendió la importancia de la estandarización de datos para garantizar la estabilidad del modelo y evitar problemas de escala. Adicionalmente, se aplicaron técnicas de transformación de variables categóricas, como el uso del Weight of Evidence (WoE), lo que facilitó la inclusión de la variable género en el análisis.

En la etapa de validación del modelo, se aplicaron métricas avanzadas como la prueba de Kolmogorov-Smirnov, el índice de Gini, el valor de información (IV) y el área bajo la curva ROC, lo que permitió evaluar de manera integral su efectividad. Asimismo, la construcción de la matriz de confusión fue fundamental para identificar errores de clasificación y ajustar el umbral de decisión con el objetivo de mejorar el desempeño general del modelo.

En conclusión, un modelo de clasificación efectivo no solo debe lograr una alta precisión, sino también optimizar el balance entre falsos positivos y falsos negativos para maximizar su utilidad en la toma de decisiones. A través de este ejercicio, se fortaleció la comprensión de la regresión logística y su aplicación en escenarios reales, permitiendo una mejor interpretación de los resultados y una toma de decisiones más informada.