PROCESAMIENTO DE DATOS

INFORME PROYECTO 01

Fecha de elaboración: 18-01-2021 **Fecha de entrega:** 21-01-2021

Integrantes:

- **Aconda Gustavo**
- **Vásquez Carlos**

1. **TEMA:** DESARROLLO DE UN MODELO DE ESTIMACIÓN DE DATOS

2. **OBJETIVOS**

   - Identificar las principales características, tendencias y estadísticas del conjunto de datos seleccionado.
   - Crear modelos de regresión lineal y polinomial que predigan el total de ingresos de una persona en base a sus características laborales, usando las técnicas vistas en clase para mejorar el modelo y su evaluación.
   - Evaluar los modelos y determinar cuál de ellos tienen mejores métricas y, por ende, mejor predicen la etiqueta de salida.

3. **DESARROLLO DE LA PRÁCTICA**

- Código de Programación

```
#Se importan las librerías necesarias para trabajar con dataframes y vectores.
import pandas as pd import numpy as np

#Se lee el dataset y se lo guarda como dataframe
df=pd.read_csv("Datasetproyecto.csv")
df=df.drop([0],axis=0);

#Se muestran los 5 primeros valores para identificar las variables
df.head()

#Se muestra el numero de registros y variables
df.shape

#Se crea la cabecera del dataframe
df.columns=["PuestoInstitucional","Gradojerarquicooescalaalqueperteneceelpuesto","Remuneracionmensualunificada","Remuneracionunificadaanual","DecimoTerceraRemuneracion","DecimaCuartaRemuneracion","Horassuplementariasyextraordinarias","Encargosysubrogaciones","Totalingresosadicionales"]
   df.head()
```

```
#Se muestra el tipo de dato de las variables
df.dtypes


#Se cambia el tipo de dato a numerico
df["Gradojerarquicooescalaalqueperteneceelpuesto"]=df["Gradojerarquicooescalaalqueperteneceelpuesto"].apply(pd.to_numeric)


#Se cambia todos los datos numericos a float
df["Remuneracionmensualunificada"]=df["Remuneracionmensualunificada"].astype(float)
df["Remuneracionunificadaanual"]=df["Remuneracionunificadaanual"].astype(float)
df["DecimoTerceraRemuneracion"]=df["DecimoTerceraRemuneracion"].astype(float)
df["DecimaCuartaRemuneracion"]=df["DecimaCuartaRemuneracion"].astype(float)
df["Horassuplementariasyextraordinarias"]=df["Horassuplementariasyextraordinarias"].astype(float)
df["Encargosysubrogaciones"]=df["Encargosysubrogaciones"].astype(float)
df["Totalingresosadicionales"]=df["Totalingresosadicionales"].astype(float)


#Se muestran los 20 primeros registros
df.head(20)


#Se muestra la información estadística
df.describe()


#Se muestra las variables con su coeficiente de correlación con las demás variables
df.corr()


# Importamaos las bibliotecas necesarias para poder utilizar la biblioteca de regresión lineal
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split


# Definimos las variables con el más bajo valor de correlación para poder apreciar cual es el comportamiento del dataset con # estos valores y lo compararemos si utilizamos un solo valor con un alto valor de correlación
x=np.asarray(df[["Totalingresosadicionales","Gradojerarquicooescalaalqueperteneceelpuesto","Horassuplementariasyextraordinarias"]])
y=np.asarray(df["Remuneracionunificadaanual"])


#Definimos los conjuntos de entrenamiento y prueba
xtrain,xtest,ytrain,ytest=train_test_split(x,y,train_size=0.70,test_size=0.30,random_state=101)
# Imprimimos el número de elementos del conjunto de entrenamiento y prueba
print("Conjunto Entrenamiento :", xtrain.shape,ytrain.shape)
print("conjunto Prueba: ",xtest.shape,ytest.shape)
```

```
# Almacenamos en una variable el metodo para regresión lineal
lm=LinearRegression()


# Entrenamos el modelo con el conjunto de entrenamiento
lm.fit(xtrain,ytrain)


# Definimos la hipotesis
h=lm.predict(xtest)


# Importamos las bibliotecas para el error medio cuadrático
from sklearn.metrics import mean_squared_error


# utilizamos la biblioteca para calcular el error medio en función de la hipótesis y el co
njunto de prueba
mse2 = mean_squared_error(ytest,h)
print("el error cuadratico medio del precio y  hipotesis es :",mse2)


import seaborn as sns
import matplotlib.pyplot as plt


grafica=sns.distplot(ytrain,hist=False, color="r",label="Salida-Real")#precio real
sns.distplot(h,hist=False,color="b",label="Valores Estimados",ax=grafica)#hipotesis ax
para que esten sobrelapadas

## si colocamos tRUE dibujamos barras

#Detalles de etiquetado
plt.title("Valores reales ajustados a valores estimados")
plt.xlabel("Pago unificado anual")
plt.ylabel("Cantidad de personas")
plt.show()


## Valor más alto se utilizó la característica con el valor de correlación más alto debid
o a un erro se debio utilizar
# la variable del valor mas bajo de corelación
xa=np.asarray(df[["DecimoTerceraRemuneracion","DecimaCuartaRemuneracion"]])
ya=np.asarray(df["Remuneracionunificadaanual"])



# dividimos el conjunto de datos en entrenamiento y prueba
xtraina,xtesta,ytraina,ytesta=train_test_split(xa,ya,train_size=0.70,test_size=0.30,rando
m_state=101)
# definimos en una variable el modelo de regresión lineal
lma=LinearRegression()
#entrenamos el modelo en base a las características seleccionadas
lma.fit(xa,ya)


# Calculamos los valores de la hipotesis
```

```
ha=lma.predict(xtesta)


# Calculamos
msea = mean_squared_error(ytesta,ha)
print("el error cuadratico medio del precio y  hipotesis es :",msea)


grafica=sns.distplot(ytraina,hist=False, color="r",label="Salida-Real")#precio real
sns.distplot(ha,hist=False,color="b",label="Valores Estimados",ax=grafica)#hipotesis
ax para que esten sobrelapadas ## si colocamos tRUE dibujamos barras #Detalles de
etiquetado
plt.title("Valores reales ajustados a valores estimados")
plt.xlabel("precio")
plt.ylabel("porcion de carros") plt.show()


#Se miran las correlaciones para seleccionar variables de entrada
df.corr()


#Se hace la grafica de dispersión y residual de las 3 variables más relacionadas a la eti
queta para ver si se puede realizar un sobreajuste si la tendencia de los puntos tiende a
no ser lineal


#grafica dispersion Encargosysubrogaciones,Totalingresosadicionales
sns.regplot(x="Encargosysubrogaciones",y="Totalingresosadicionales",data=df)


#grafica residual Encargosysubrogaciones,Totalingresosadicionales
import seaborn as sns
sns.residplot(df["Encargosysubrogaciones"],df["Totalingresosadicionales"])
plt.show()


#grafica dispersión Gradojerarquicooescalaalqueperteneceelpuesto,Totalingresosadici
onales
sns.regplot(x="Gradojerarquicooescalaalqueperteneceelpuesto",y="Totalingresosadicio
nales",data=df)



#grafica residual
Gradojerarquicooescalaalqueperteneceelpuesto,Totalingresosadicionales
sns.residplot(df["Gradojerarquicooescalaalqueperteneceelpuesto"],df["Totalingresosadi
cionales"]) plt.show()


#grafica dispersión Horassuplementariasyextraordinarias,Totalingresosadicionales
sns.regplot(x="Horassuplementariasyextraordinarias",y="Totalingresosadicionales",dat
a=df)


#grafica residual Horassuplementariasyextraordinarias,Totalingresosadicionales
```

```
sns.residplot(df["Horassuplementariasyextraordinarias"],df["Totalingresosadicionales"])
plt.show()


#Para el modelo de regresión polinomial se hará uso de las 3 variables de las gráficas
ya que presentan buenas graficas de dispersión y residual. Además, se
#hara uso de aumento de características con la función polinomial, esto con el fin de m
ejorar el modelo.
#Por otro lado se utilizará validación cruzada para evaluar el rendimiento del modelo.
#Tambien se utilizará regularización Ridge para disminuir los valores de theta y mejor
ar el modelo


#Se normalizan la variables del dataset
from sklearn.preprocessing import StandardScaler
SCALE=StandardScaler()
v_n=df[["Encargosysubrogaciones","Gradojerarquicooescalaalqueperteneceelpuesto","
Horassuplementariasyextraordinarias"]]


SCALE.fit(v_n)


#Se obtienen los nuevos valores normalizados
x_normalizado=SCALE.transform(v_n)
y_normalizado=(df["Totalingresosadicionales"])


#Dividir el dataset from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x_normalizado,y_normalizado,train_size=0.70,t
est_size=0.30,random_state=101) print("Numero de muestras de Prueba=
",xtest.shape[0]) print("Numero de muestras de Entrenamiento= ",xtrain.shape[0])


#Dimensiones de xtest
xtest.shape


#Dimensiones de ytest
ytest.shape


#se crea el modelo de regresión lineal empleando función polinomial de grado 3 from
sklearn.preprocessing import PolynomialFeatures pr=PolynomialFeatures(degree=3)
xtrain_pr=pr.fit_transform(xtrain) xtest_pr=pr.fit_transform(xtest)


#Dimensiones o numero de variables de entrada
print("Dimensiones de entrenamiento",xtrain_pr.shape)
print("Dimensiones de prueba",xtest_pr.shape)


#Entrenar el algoritmo y definir la hipótesis
from sklearn.linear_model import LinearRegression
poly=LinearRegression().fit(xtrain_pr,ytrain)
poly.fit(xtrain_pr,ytrain)
```

```
h=poly.predict(xtest_pr)


#Evaluar el modelo con MSE
from sklearn.metrics import mean_squared_error,r2_score
mse=mean_squared_error(ytest,h)
print("El error cuadratico medio es: ",mse)


#Evaluar el modelo con R^2
r2=r2_score(ytest,h)
print("El factor R^2 es: ",r2)


#Se hace la gráfica de distribución para ver si el modelo predecirá bien
import seaborn as sns
#Graficar la salida para la regresion lineal para engine-size
grafica=sns.distplot(ytest,hist=False,color="r",label="Datos reales")#False para que no se vean las lineas
sns.distplot(h,hist=False,color="b",label="Datos estimados",ax=grafica)#ax para ver si estan sobrelapadas con la grafica

#detalles de etiquetado
plt.title("Valores reales ajustados a valores estimados")
plt.xlabel("Variables de entrada")
plt.ylabel("Total de ingresos adicionales")
plt.show()


#Como se puede ver en la gráfica de distribución, el modelo predice bien con respecto a los datos reales.


#Se aplica regularización para disminuir los parámetros theta con el fin de disminuir el error o aumentar R^2
from sklearn.linear_model import Ridge
#Crear un lazo para evaluar el valor de R2_score
R2_val=[]
FACTOR_LAMBDA=[0.01,0.1,0.5,0.8,1,3,5,10,40]
for factor in FACTOR_LAMBDA:
    RidgeR1=Ridge(alpha=factor)
    RidgeR1.fit(xtrain_pr,ytrain)
    hR1=RidgeR1.predict(xtest_pr)
    R2_val.append(r2_score(ytest,hR1))


#se grafica el factor lambda para que el factor R^2 aumente
import matplotlib.pyplot as plt
width=12
height=10
plt.figure(figsize=(width,height))
plt.plot(FACTOR_LAMBDA,R2_val,label="Datos de Prueba")
plt.xlabel("Lambda")
plt.ylabel("R^2")
plt.legend()
```

#Mediante la gráfica se puede observar que mientras más aumenta el valor de lambda más disminuye el rendimiento, es decir que no hace falta aplicar regularización.

```
from sklearn.model_selection import cross_val_score,cross_val_predict
Lr=LinearRegression() #VC es una matriz que contiene los valores de evaluación de cada iteración VC=cross_val_score(Lr,xtrain,ytrain,cv=8)#se divide en 4 segmentos y hace 4 iteraciones


#Se imprime los valores calculados en cada iteracion VC


#Se imprime el promedio
print("El promedio del rendimiento de las iteraciones es: ",VC.mean())


#Mediante la validación cruzada se tuvo un mayor rendimiento que cuando se utilizó el conjunto de entrenamiento y prueba para validar,
#Esto quiere decir que el porcentaje de rendimiento mostrado es más confiable que con el conjunto train y test
```

- Tablas

	Puesto Institucional	Grado jerarquico o escala al que pertenece el puesto	Remuneracion mensual unificada	Remuneracion unificada (anual)	Decimo Tercera Remuneracion	Decima Cuarta Remuneracion	Horas suplementarias y extraordinarias	Encargos y subrogaciones	Total ingresos adicionales
1	SUPERINTENDENTE DE MANTENIMIENTO	6	5250.0	63000.0	437.50	31.25	0.0	0.0	468.75
2	SUPERVISOR RCRS	5	3300.0	39600.0	275.00	31.25	0.0	0.0	306.25
3	MENSAJERO	1	962.0	11544.0	80.17	31.25	0.0	0.0	111.42
4	AUXILIAR DE CAFETERIA	1	779.0	9348.0	64.92	31.25	0.0	0.0	96.17
5	AUXILIAR DE CAFETERIA	1	784.0	9408.0	65.33	31.25	0.0	0.0	96.58

Tabla 1. Dataframe con 5 registros.

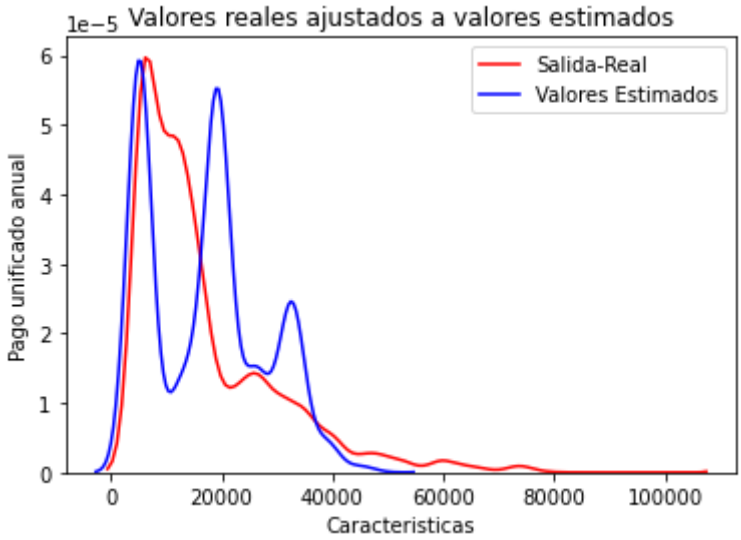
	GradojerarquicooescalaalqueperteneceelpuestO	RemuneracionmensualunificadA	RemuneracionunificadaanuaL	DecimoTerceraRemuneracion
count	10278.000000	10278.000000	10278.000000	10278.000000
mean	2.781280	1402.857275	16834.287297	120.844457
std	1.596566	1072.107958	12865.295495	91.988766
min	1.000000	465.000000	5580.000000	37.460000
25%	1.000000	574.000000	6888.000000	48.420000
50%	3.000000	1087.000000	13044.000000	91.370000
75%	4.000000	1886.150000	22633.800000	164.375000
max	8.000000	8437.500000	101250.000000	703.120000

Tabla 2. Resumen estadístico

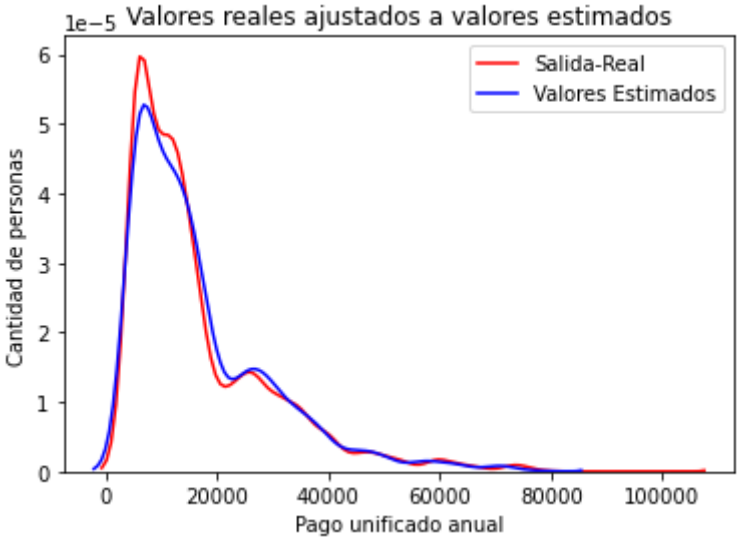
DecimoTerceraRemuneracion	DecimaCuartaRemuneracion	HorassuplementariasyextraordinariaS	EncargosysubrogacioneS	TotalingresosadicionalES
0.792869	-0.014162	0.144326	0.337541	0.562197
0.989889	-0.009767	0.126538	0.002480	0.327985
0.989889	-0.009767	0.126538	0.002480	0.327985
1.000000	0.009501	0.258905	-0.003309	0.384587
0.009501	1.000000	0.009773	0.009917	0.016238
0.258905	0.009773	1.000000	-0.040982	0.479533
-0.003309	0.009917	-0.040982	1.000000	0.816734
0.384587	0.016238	0.479533	0.816734	1.000000

Tabla 3. Correlaciones entre variables

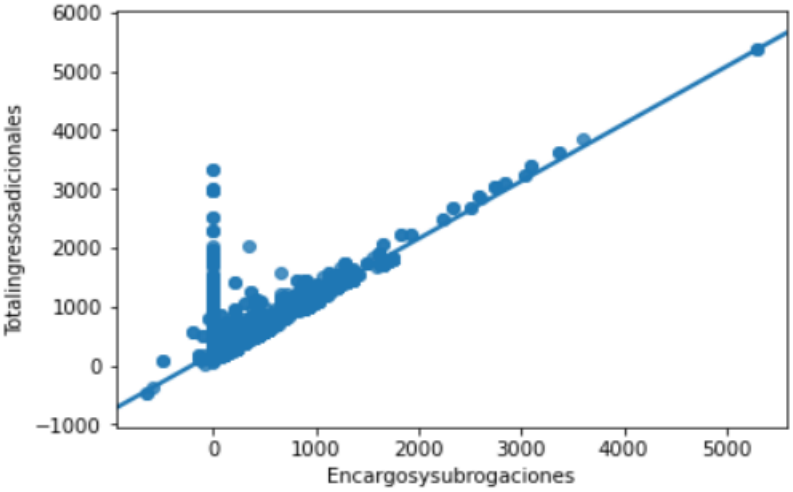
- Gráficas



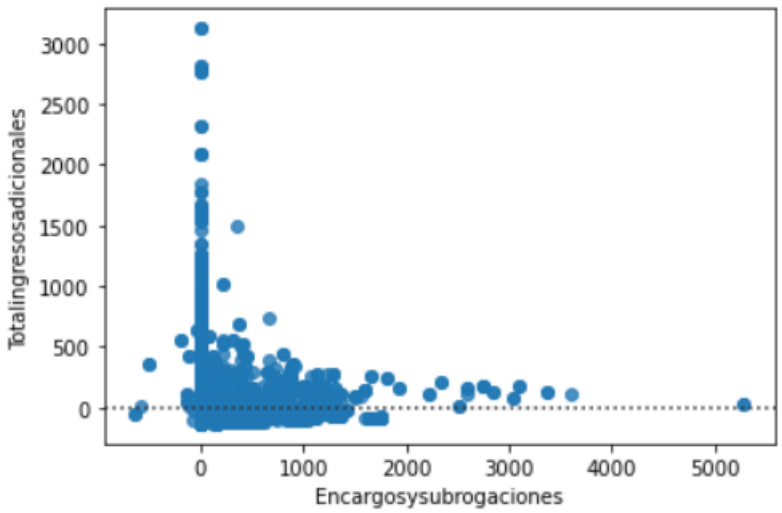
Gráfica 1. Diagrama de distribución con regresión lineal multivariable.



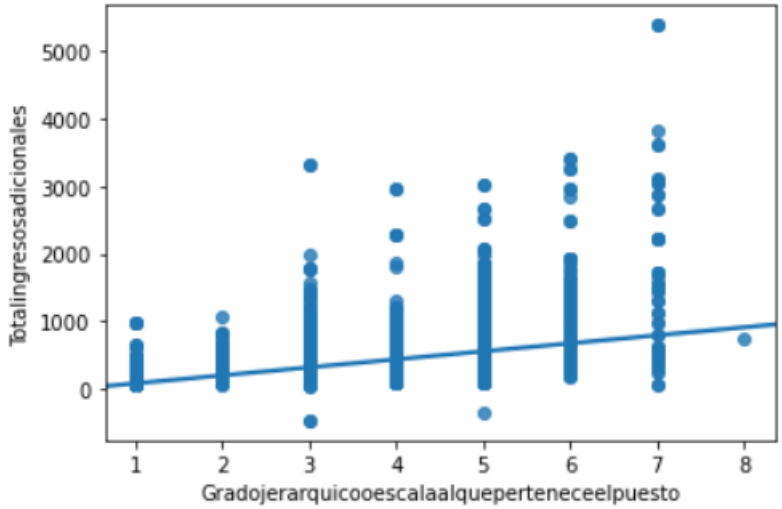
Gráfica 2. Diagrama de distribución con regresión lineal con una variable.



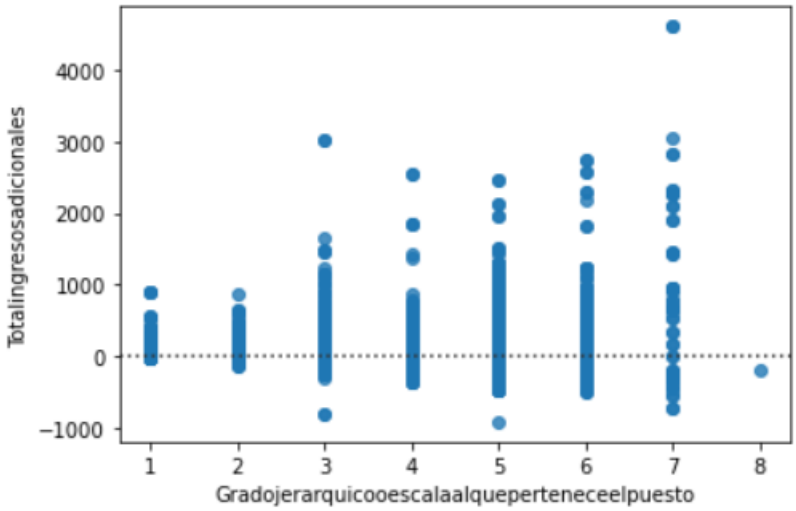
Grafica 3. Diagrama de dispersión Encargosysubrogaciones vs Totalingresosadicionales.



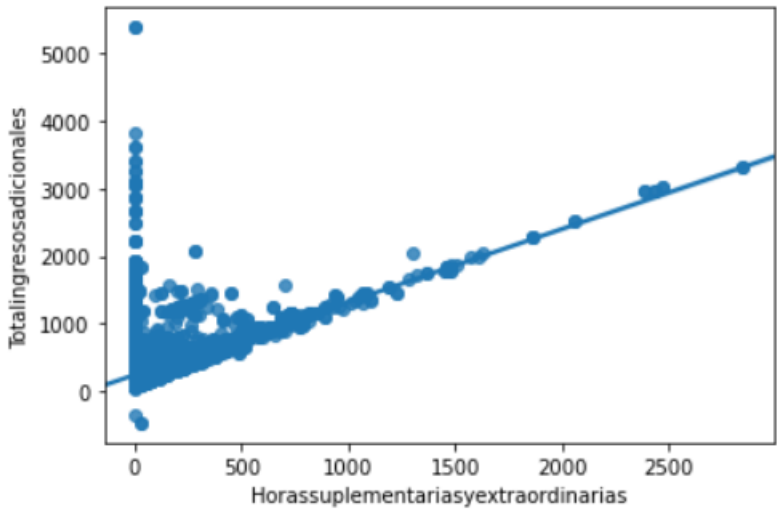
Grafica 4. Diagrama residual Encargosysubrogaciones vs Totalingresosadicionales.



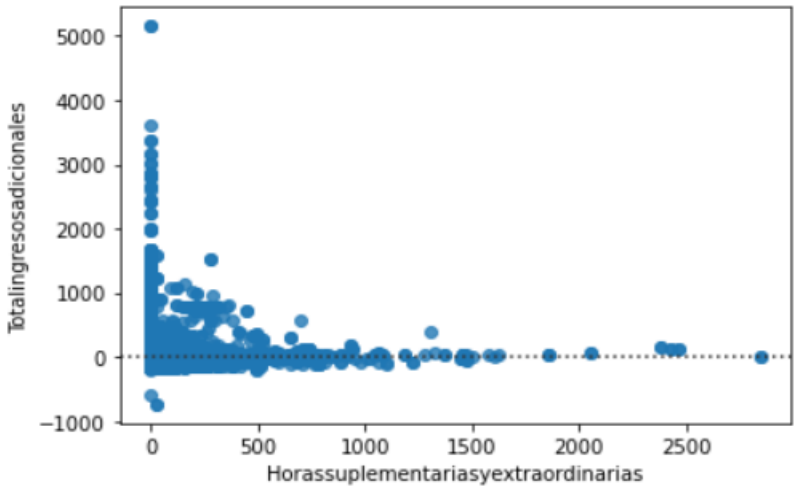
Grafica 5. Diagrama de dispersión Gradojerarquicooescalaalqueperteneceelpuesto vs Totalingresosadicionales.



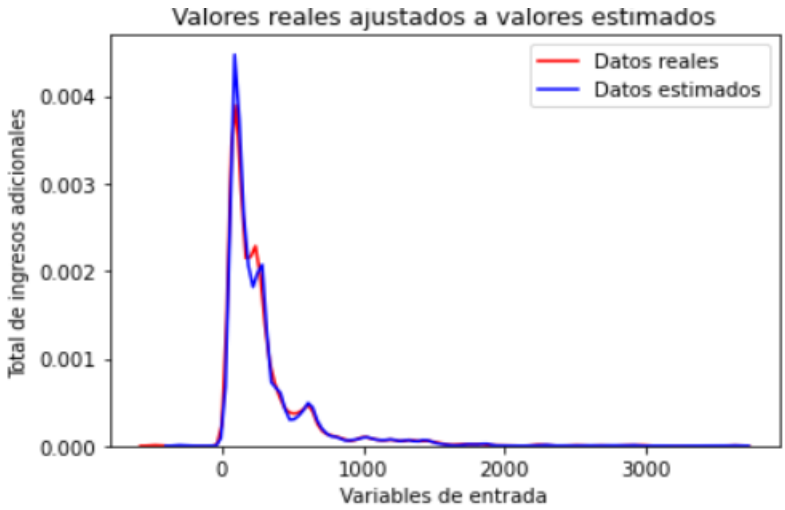
Grafica 6. Diagrama residual Gradojerarquicooescalaalqueperteneceelpuesto vs Totalingresosadicionales.



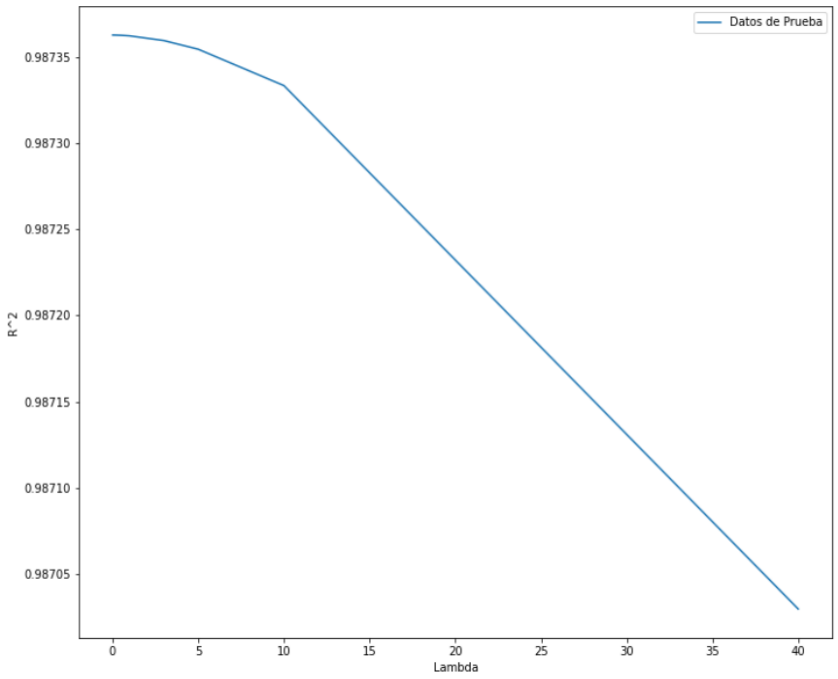
Grafica 7. Diagrama de dispersión Horassuplementariasyextraordinarias vs Totalingresosadicionales.



Grafica 8. Diagrama residual Horassuplementariasyextraordinarias vs Totalingresosadicionales.



Gráfica 9. Gráfica de distribución en modelo de regresión polinomial.



Gráfica 10. Gráfica de factor lambda vs R^2 para modelo de regresión polinomial.

- Salidas de consulta de datos

```
PuestoInstitucional                          object
GradojerarquicooescalaalqueperteneceelpuestO   int64
Remuneracionmensualunificada                float64
Remuneracionunificadaanual                  float64
DecimoTerceraRemuneracion                   float64
DecimaCuartaRemuneracion                    float64
Horassuplementariasyextraordinarias         float64
Encargosysubrogaciones                      float64
Totalingresosadicionales                    float64
dtype: object
```

Salida 1. Tipo de datos del dataset.

```
Numero de muestras de Prueba=  3084
Numero de muestras de Entrenamiento=  7194
```

Salida 2. Muestras del conjunto de prueba y entrenamiento en Regresión Polinomial.

```
(3084, 3)
```

Salida 3. Dimensiones de xtest en Regresión Polinomial

```
(3084,)
```

Salida 4. Dimensiones de ytest en Regresión Polinomial

```
Dimensiones de entrenamiento (7194, 20)
Dimensiones de prueba (3084, 20)
```

Salida 5. Dimensiones de xtrain y xtest aplicando aumento de características.

```
El error cuadratico medio es:  1321.4284332512125
```

Salida 6. MSE en modelo de regresión Polinomial con conjunto train y test.

```
El factor R^2 es:  0.9873631174848144
```

Salida 7 Factor R^2 en modelo de regresión Polinomial con conjunto train y test.

```
array([0.97164744, 0.98117402, 0.98494903, 0.98040755, 0.97920627,
       0.9794348 , 0.9801217 , 0.97983   ])
```

Salida 8. Vector de valores de validación cruzada para regresión polinomial.

```
El promedio del rendimiento de las iteraciones es:  0.979596351668256
```

Salida 9. Promedio de validación cruzada para regresión polinomial.

4. ANÁLISIS DE RESULTADOS

El análisis de datos hecho al dataset propuesto constaba en su mayoría de datos numéricos como se puede observar en la tabla 1, esto es porque el dataset describía características laborales de una persona. Entre los datos estadísticos importantes se pudo observar que en promedio las personas ganaban 1402.85 dólares al mes, lo cual es un valor alto. El mínimo salario que se tenía era de 465 $, mientras que el máximo bordeaba los 8500 $. Analizando las correlaciones se obtuvo que 1 presenta un alto coeficiente, mientras que otras 3 presentan un valor aceptable.

De todos los modelos realizados, el que mejor predecía fue el de regresión polinomial. Para realizarlo se visualizaron las variables que más correlación presentaban con respecto a la variable de etiqueta “Totalingresosadicionales”, estas eran Encargosysubrogaciones, Gradojerarquicooescalaalqueperteneceelpuesto y Horassuplementariasyextraordinarias. El gran aporte que iban a brindar estas variables al modelo se podía evidenciar también en las gráficas 3,4,5,6,7 y 8, pero al presentar un diagrama de dispersión lineal, no se pudo usar la función polinomial para aumentar el grado ya que iba a producir sobreajuste, en vez de eso, se hizo un aumento de características con las 3 variables de entrada escogidas, dando como resultado 20 variables de entrada, como se indica en la salida 5**.** Paso siguiente se quiso probar sobreajuste para reducir los parámetros theta del modelo y para esto se usó una gráfica de cómo se comportaba el factor R^2 mientras aumentaba lambda, gráfica que se muestra en la figura 10, pero al presentar el valor máximo en lambda=0, se decidió no usar regularización porque esto produciría una disminución de rendimiento al modelo. Finalmente, se utilizó validación cruzada para mejorar la evaluación del modelo, cuando se usaron el conjunto de entrenamiento y prueba se obtuvo un MSE de 1321.42$ y un R^2 de 98.74% (ver salidas 6 y 7), pero con la validación cruzada con 8 iteraciones se obtuvo un factor de rendimiento de 97.96% (ver salida 9), resultado que es más confiable que cuando se validaba el modelo usando conjunto de entrenamiento y prueba.

5. CONCLUSIONES

- El modelo de regresión polinomial sirvió para aumentar el número de variables características de 3 a 20, esto beneficio al modelo de predicción excelentemente, esto se puede evidenciar cuando se evaluó el modelo ya que se presentó un bajo error cuadrático medio de 1321.42 dólares y un factor de rendimiento de 97.96% usando

el método de validación cruzada para evaluar el modelo, además gracias a la gráfica de rendimiento vs factor lambda, se determinó que no es factible utilizar regularización porque produciría sobreajuste.

- Al aplicar la regresión lineal a entradas con bajo valor de correlación se puedo aprecias que el modelo obtenido tenía un gran margen de error, pues como se observó en la gráfica entre el conjunto de prueba y entrenamiento el error era alto, además el número de variables no ayudo a un mejoramiento del modelo, por otro lado, tenemos regresión lineal aplicada al valor más alto de correlación con el cual se tuvo una gran aproximación a la salida esperada.
- Con el análisis para dos casos siendo estos un valor alto de correlación y un valor bajo de correlación se pudo observar que mientras más cercano sea a uno la correlación mejor rendimiento y precisión tenga nuestro modelo, en cambio para un valor bajo es necesario utilizar distintas características para obtener una mejor aproximación al modelo esperado.

6. RECOMENDACIONES

- Se recomienda utilizar los diagramas residuales y de dispersión para analizar si es conveniente utilizar modelados lineales o no-lineales con las variables que tengan un buen índice de correlación, una vez realizado el modelo se recomienda evaluar con validación cruzada para una mejor evaluación del modelo.
- Para los dataset es necesario utilizar un formato utf8 para que nos pueda compilar él .csv, para esto es útil utilizar programas web como lo es las hojas de cálculo de Google.
- Otro dato importante al trabajar con datasets es saber que no reconoce la coma como un carácter valido para separar como decimales, por lo cual se debe cambiar la coma por el punto para evitar posibles errores.

7. BIBLIOGRAFÍA

[1] A. R. Hoyos. [En línea]. Available: file:///C:/Users/hp/Desktop/vasquez/ProcesamientoDeDatos/DiaposMateria/Python_Parte_1.pdf. [Último acceso: 03 diciembre 2020].

[2] A. R. Hoyos. [En línea]. Available: file:///C:/Users/hp/Desktop/vasquez/ProcesamientoDeDatos/DiaposMateria/Python_Parte_2.pdf. [Último acceso: 03 diciembre 2020].

[3] pandas development team, [En línea]. Available: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.groupby.html. [Último acceso: 03 diciembre 2020].

[4] Open Knowledge Fundation, «Datos Abiertos,» [En línea]. Available: http://catalogo.datosabiertos.gob.ec/en/dataset. [Último acceso: 22 12 2020].