Instituto Tecnológico y de Estudios Superiores de Monterrey

Materia:

Analítica de datos y herramientas de inteligencia artificial II

(Gpo 501)

Entregable:

Actividad 6

Regresión Lineal Múltiple Y No Lineal

Alumnos:

Andrés Salmerón García A01731809

Julen Ugartechea Repetto A01735646

Yael Mojica Pérez A01735620

Fecha de entrega:

12 de Octubre del 2023

Introducción

Para esta actividad, se analizaron los datos de un grupo de individuos de la India, tomando en cuenta categorías como la edad, estado civil y profesión para después realizar diferentes análisis, gráficas y modelos para interpretar todo en información más entendible, además de poder buscar y conocer algunos datos específicos entre columnas, como lo fue la relación y correlación que existía entre las columnas, además de la búsqueda de las mejores relaciones en columnas, buscando el mejor porcentaje de correlación, haciendo uso de diferentes metodologías como lo fue la regresión lineal simple, la regresión lineal múltiple y la regresión no lineal.

Cabe mencionar que para todos los análisis, se hizo primero un preprocesamiento, en el que eliminaron valores nulos y outliers.

4. Extracción de características

En este punto, se realizó la extracción de características de las siguientes columnas categóricas: Age, Experience, Married/Single, House_Ownership, Profession, CITY, CURRENT_JOB_YRS, CURRENT_HOUSE_YRS y Risk_Flag, con el propósito de generar un análisis descriptivo de los hallazgos obtenidos empleando tablas y gráficos.

Lo primero que se realizó, fue pasar los valores de las columnas mencionadas anteriormente para que tuviera valores categóricos, como se muestra en el siguiente código:

```
[ ]  1 #Primero pasamos algunos valores de la columna a tipo categorico
     2 df['Age']=df['Age'].astype(str)
     3 df['Experience']=df['Experience'].astype(str)
     4 df['CURRENT_JOB_YRS']=df['CURRENT_JOB_YRS'].astype(str)
     5 df['CURRENT_HOUSE_YRS']=df['CURRENT_HOUSE_YRS'].astype(str)
     6 df['Risk_Flag']=df['Risk_Flag'].astype(str)
```

Posteriormente, se realizó el código para el análisis univariado para cada una de estas variables, empleando filtros, ajustes y la tabla de frecuencias de cada una. Esto se puede observar en las siguientes líneas de código:

```
#Obtengo un ánalisis univariado de las variables categóricas en especifico
table1=freq_tbl(df['Age'])
#Obtengo un filtro de los valores más reelevantes de la variables categórica seleccionada
Filtro1=table1[table1['frequency']>1]
#Ajusto el indice de mi dataframe
Filtro_index=Filtro1.set_index('Age')
#Realizamos grafico de barras del dataframe filtrado
Filtro_index.plot(kind = 'bar', width=1.2, figsize=(14,4))
```

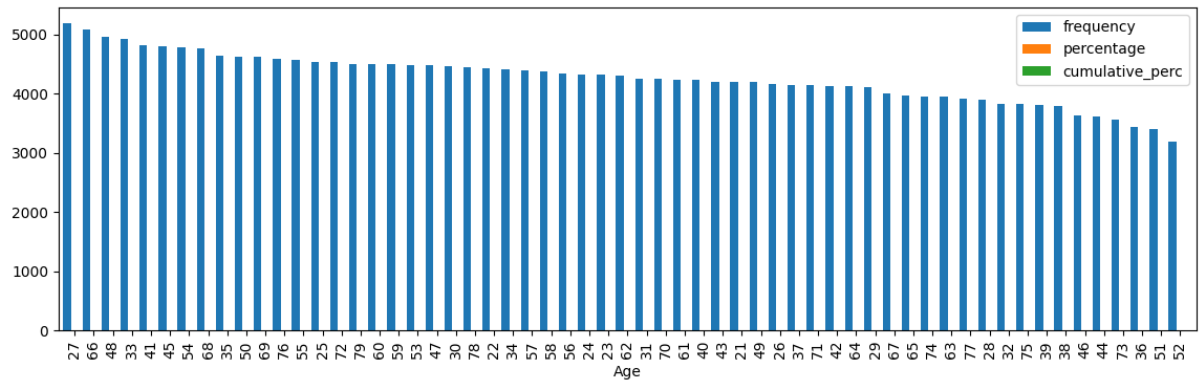
Con esto, obtuvimos la tabla de frecuencias de la variable ‘Age’, la cual nos mostró, que las cinco edades más comunes son las siguientes:

1. 27 años
2. 66 años
3. 48 años
4. 33 años
5. 41 años

Mientras que las menos comunes son:

1. 52 años
2. 51 años
3. 36 años
4. 73 años
5. 44 años

Se pueden observar estas y todas las demás edades con su respectiva cantidad, en la tabla de frecuencia que nos arrojó el código:



Utilizamos el mismo código, para obtener la tabla de frecuencias de las demás variables, empezando por la variables ‘Experience’, en la cual, se obtuvo que los cinco valores más comunes son:

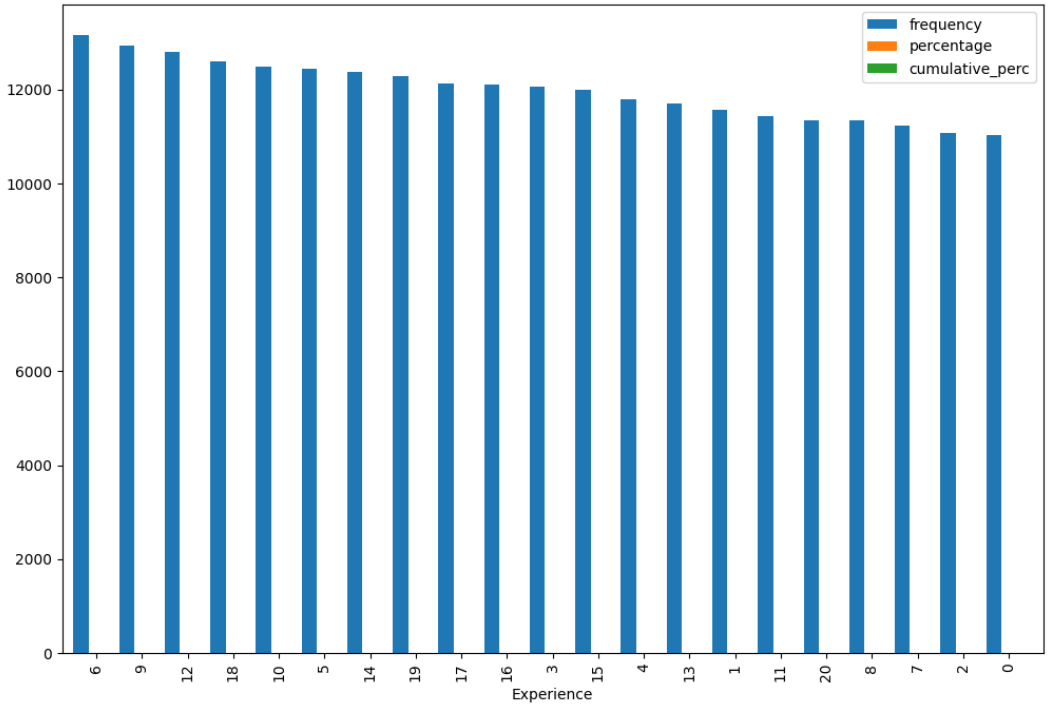
1. 6

2. 9
3. 12
4. 18
5. 10

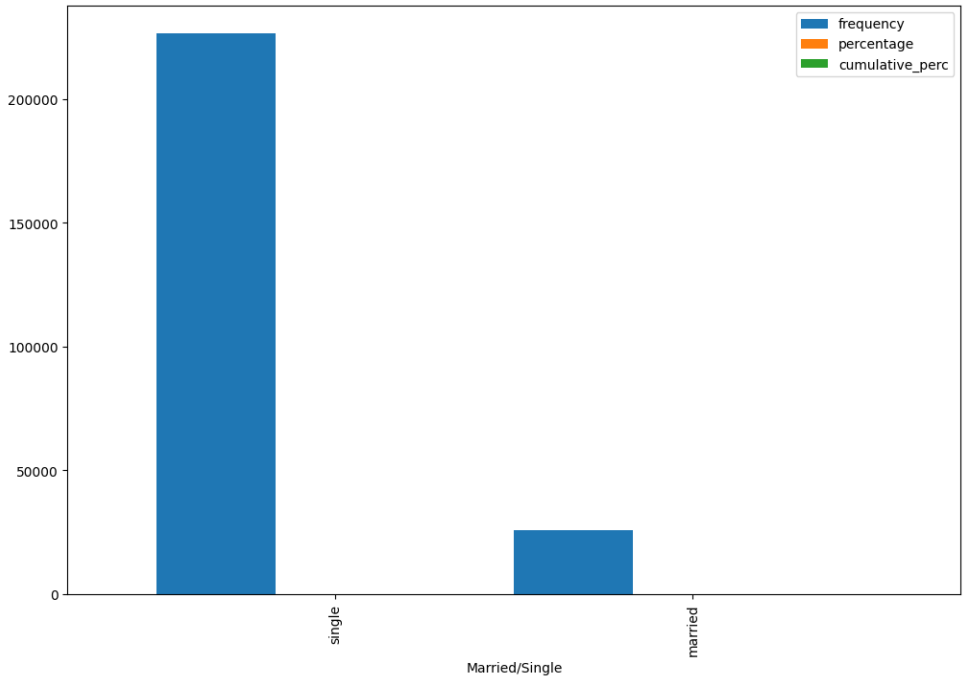
Mientras que los menos comunes son:

1. 0
2. 2
3. 7
4. 8
5. 20

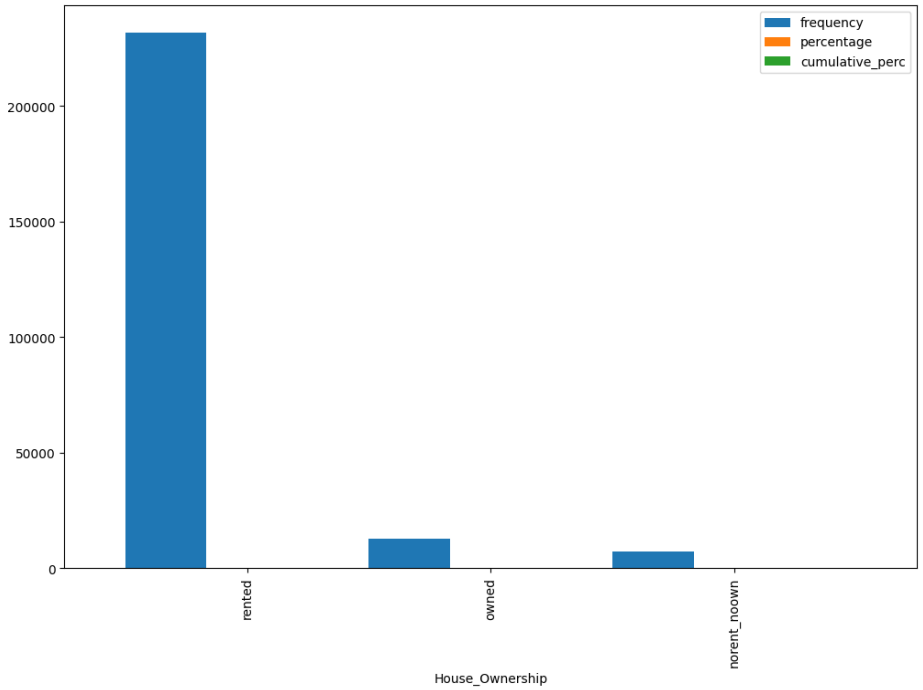
Se pueden observar estos y todos los demás valores de la columna ‘Experience’ con su respectiva cantidad, en la tabla de frecuencia que nos arrojó el código:



Para la variable ‘Married/Single’, solo se tienen dos valores, por lo que comparar los datos de esta variable es mucho más fácil. Los resultados que obtuvimos fueron que existen muchos más solteros (más de 20,000) que casados (menos de 50,000), como se puede observar en la tabla de frecuencia que nos arrojó el código:



Para la variable de ‘House_ownership’, tenemos tres valores posibles, los cuales son rented, owned y norent_noown. Los resultados de la tabla, nos mostraron que en su mayoría, los valores de esta variable se encuentran dentro de la categoría rented (más de 20,000), seguido por owned (menos de 2,500) y norent_noown (menos de 1,000):



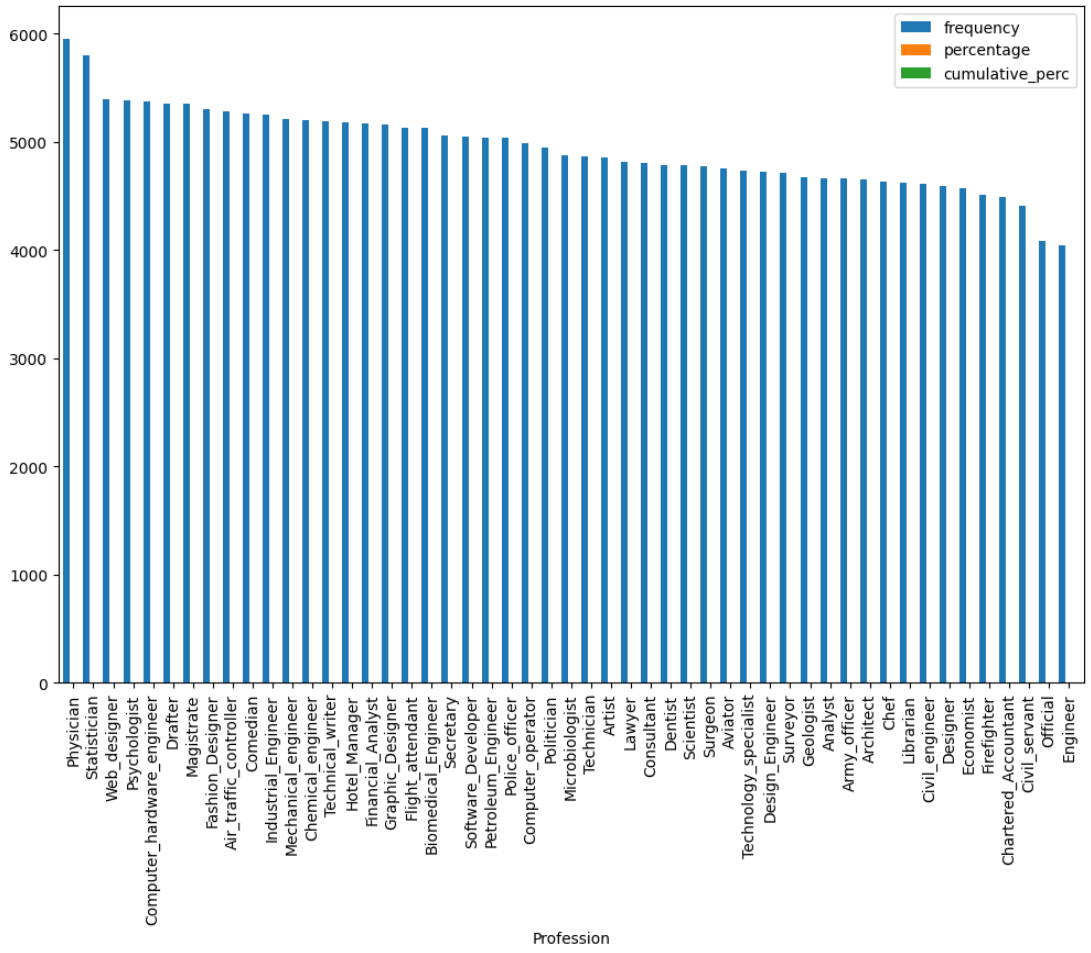
Para la variable ‘Profession’, existen muchos posibles valores, sin embargo, los resultados de la tabla nos mostraron que los cinco más comunes son:

1. Physician
2. Statistician
3. Web_designer
4. Psychologist
5. Computer_hardware_engineer

Mientras que los cinco resultados menos comunes son:

1. Engineer
2. Official
3. Civil_servant
4. Chartered_accountant
5. Firefighter

Todos estos resultados, y los demás, se pueden observar en la tabla de frecuencia que nos arrojó el código:



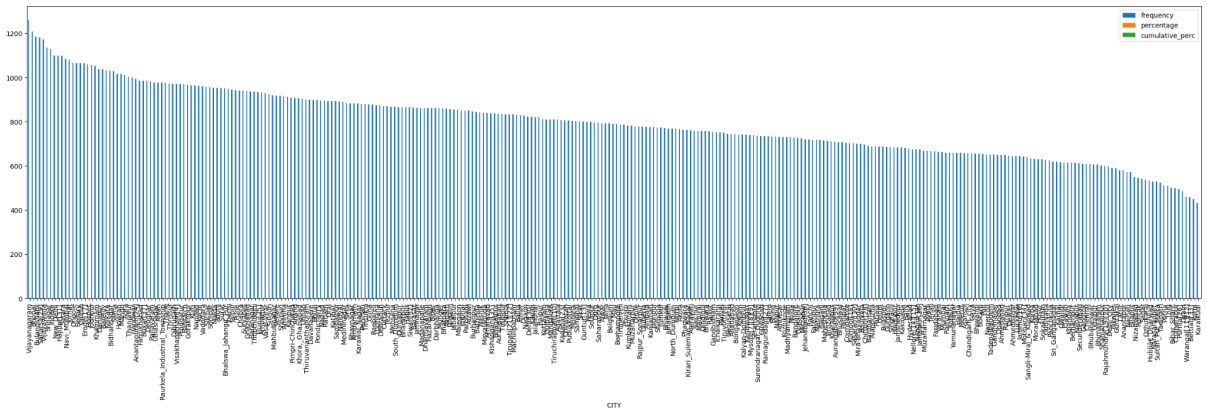
Para la variable ‘CITY’, los cinco resultados más comunes son los siguientes:

1. Vijayanagaram
2. Bhopal
3. Bulandshanl
4. Saharsal
5. Vijayawada

Mientras que los resultados menos comunes son:

1. Karalkudi
2. Katni
3. Bettiah
4. Warangall
5. Purnla

Se pueden observar todos estos resultados y los demás, en la tabla de frecuencia que nos arrojó el código:



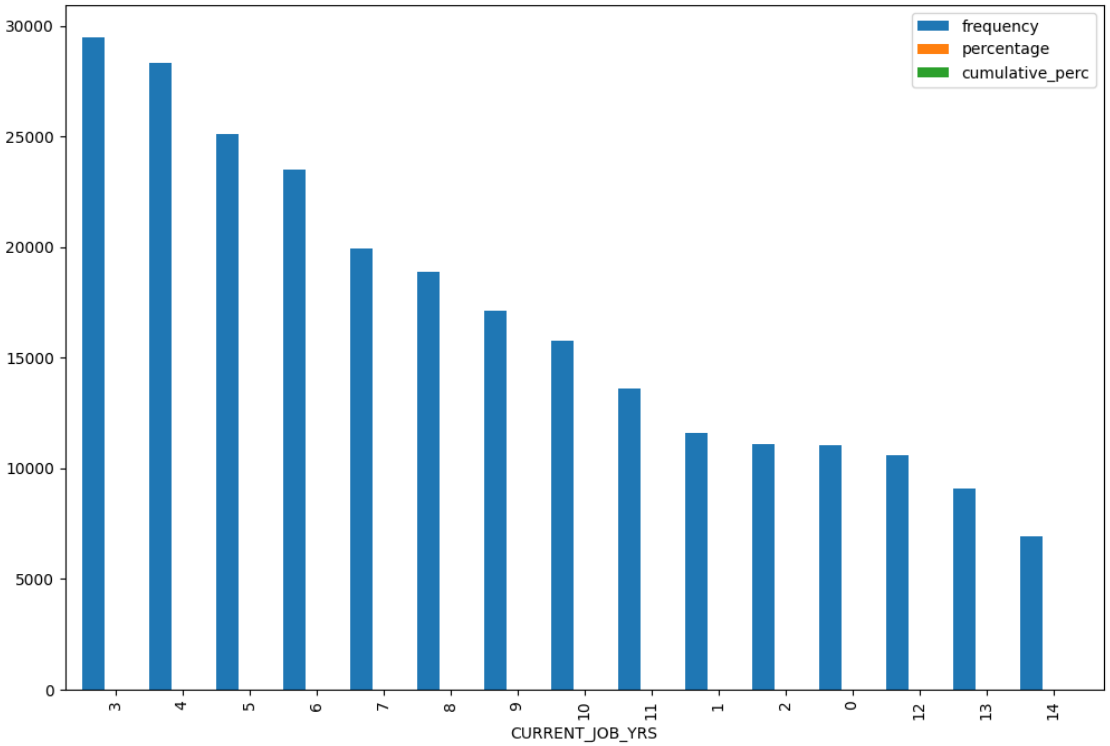
Para la variable ‘CURRENT_JOB_YRS’, los cinco resultados más comunes fueron:

1. 3
2. 4
3. 5
4. 6
5. 7

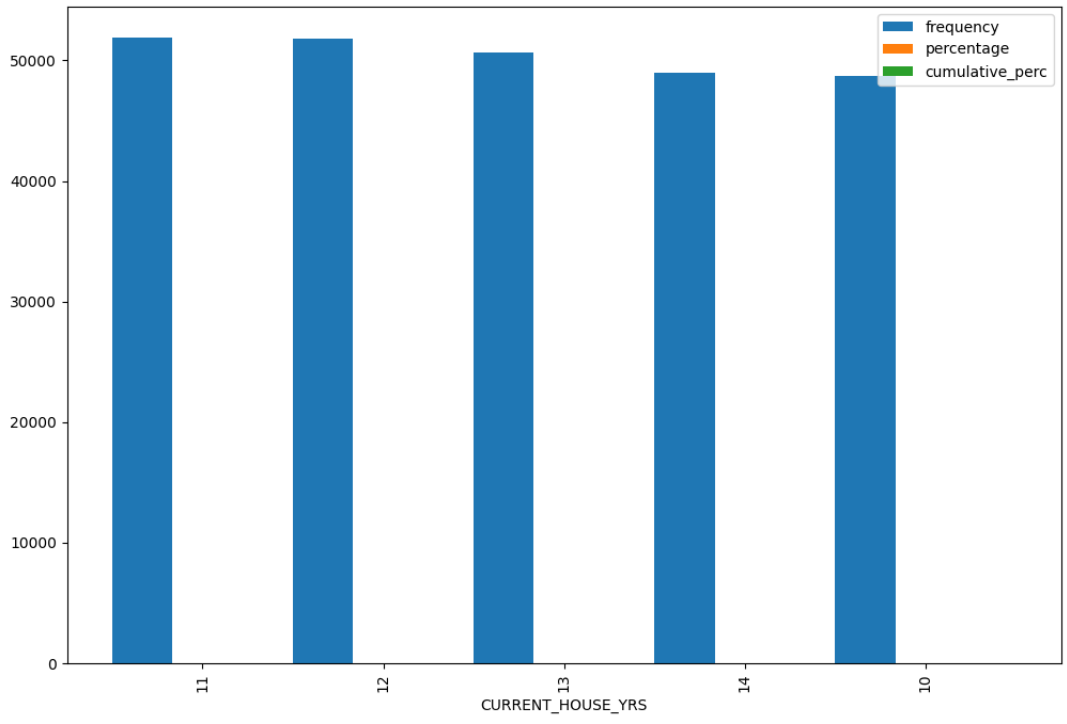
Mientras que los menos comunes son:

1. 14
2. 13
3. 12
4. 0
5. 2

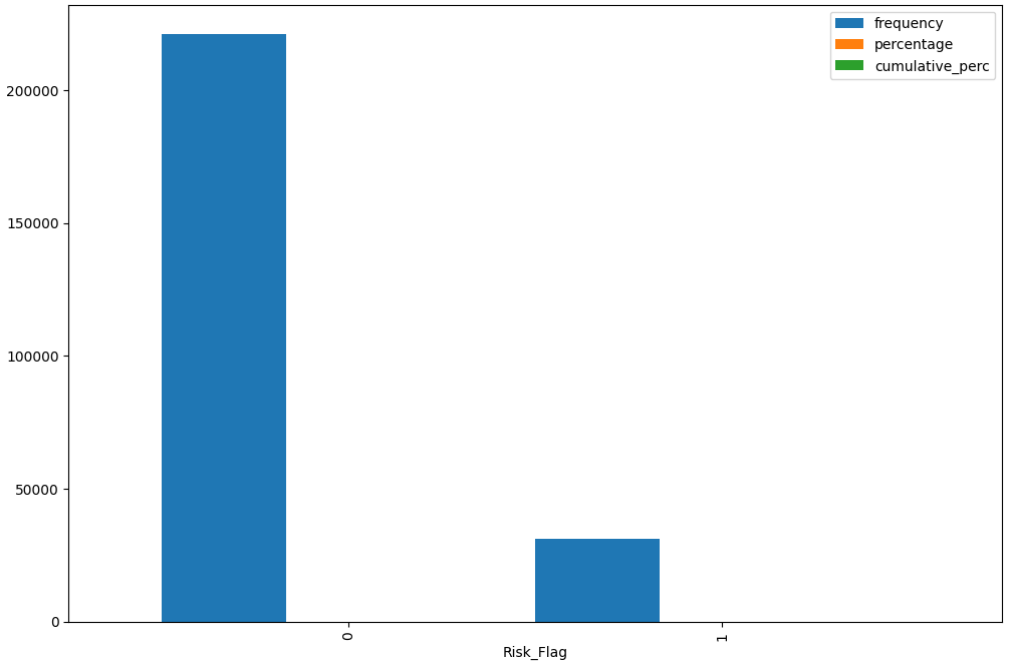
Se pueden observar estos resultados, y los demás de la variable, dentro de la tabla de frecuencia que nos arrojó el código:



En la variable ‘CURRENT_HOUSE_YRS’, sólo existen cinco valores, de los cuales destacan 11, como el valor más común dentro de la variable, y 10 como el valor menos común. Se pueden observar los resultados en la siguiente tabla de frecuencia que nos arrojó el código:



Por último, la variable ‘Risk_flag’, contiene solo dos posibles valores, de los cuales, los resultados nos mostraron que es mucho más común el 0 (más de 20,000) que el 1 (menos de 5,000):



5. Correlación entre variables

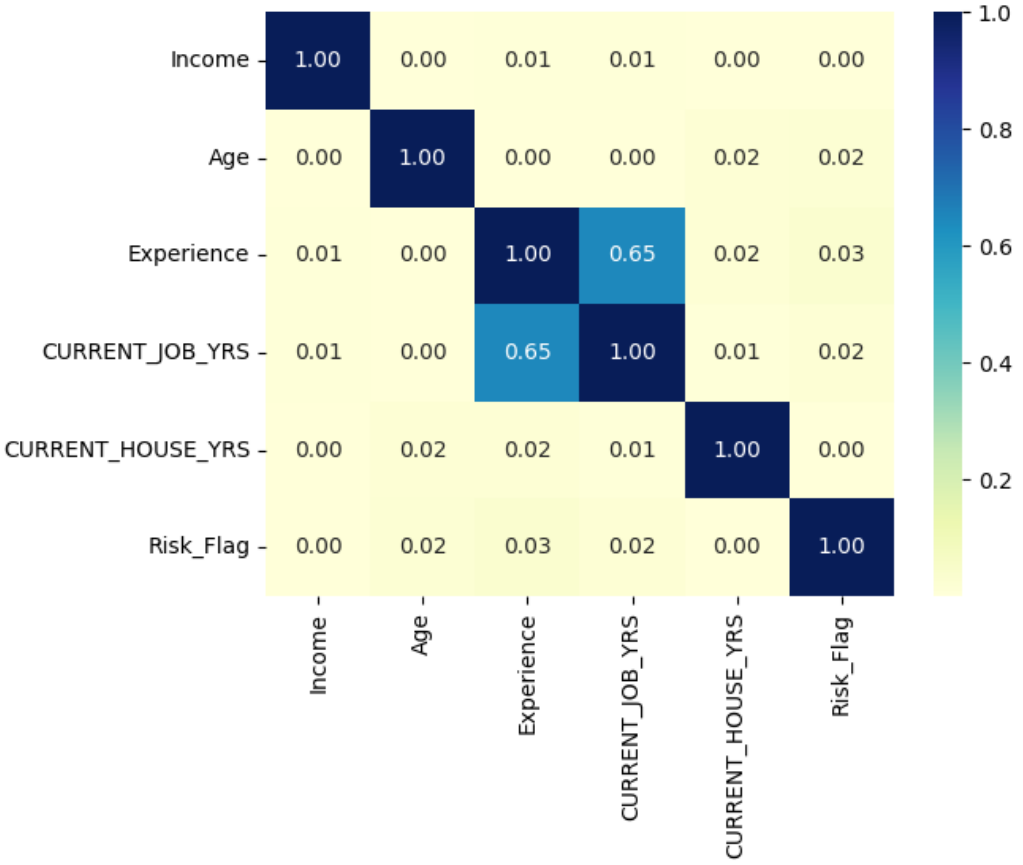
En este punto, lo primero que se realizó, fue la identificación de las variables predictoras, las cuales son 'Income', 'Age', 'Experience', 'CURRENT_JOB_YRS', 'CURRENT_HOUSE_YRS' y por último 'Risk_Flag', y a todas estas variables, se les cambió a datos de tipo entero (int) con el siguiente código:

```
#Primero pasamos algunos valores de la columna a tipo número
G['Income']=G['Income'].astype(int)
G['Age']=G['Age'].astype(int)
G['Experience']=G['Experience'].astype(int)
G['CURRENT_JOB_YRS']=G['CURRENT_JOB_YRS'].astype(int)
G['CURRENT_HOUSE_YRS']=G['CURRENT_HOUSE_YRS'].astype(int)
G['Risk_Flag']=G['Risk_Flag'].astype(int)
```

Posteriormente, se utilizó la función ‘.corr’, con el objetivo de ver qué tan fuerte es la correlación entre cada una de estas variables predictoras, y se obtuvieron las siguientes tablas de correlaciones:

	Income	Age	Experience	CURRENT_JOB_YRS	CURRENT_HOUSE_YRS	Risk_Flag
Income	1.000000	-0.000652	0.006422	0.007045	-0.002397	-0.003091
Age	-0.000652	1.000000	-0.001118	0.002154	-0.020134	-0.021809
Experience	0.006422	-0.001118	1.000000	0.646098	0.019309	-0.034523
CURRENT_JOB_YRS	0.007045	0.002154	0.646098	1.000000	0.005372	-0.016942
CURRENT_HOUSE_YRS	-0.002397	-0.020134	0.019309	0.005372	1.000000	-0.004375
Risk_Flag	-0.003091	-0.021809	-0.034523	-0.016942	-0.004375	1.000000

Después de tener esto, se realizó también un heat map, con la función sns.heatmap de la librería seaborn, y el código nos arrojó el siguiente mapa:



Al analizar el mapa de calor y la tabla de correlaciones, se puede concluir que hay una sola relación entre variables que destaca, ya que los demás presentan un correlación muy baja, por lo que la única a la que le podemos aplicar el análisis de regresión lineal, es las columnas 'Experience' y 'Current_Job_YRS'. Después de concluir esto con las correlaciones de las variables, se declaró como la variable dependiente a la variable ‘Current_Job_YRS’ y la independiente ‘Experience’. Además de esto, se definió model, como la función de regresión lineal, se verificó la función relacionada al modelo y se ajustó a las variables declaradas como independiente y dependiente. Todo esto, se realizó con las siguientes líneas de código:

```
#Declaramos las variables dependientes e independientes para la regresión lineal
#Vars_Indep= df[['alcohol', 'speeding']]
Vars_Indep= G[['Experience']]
Var_Dep= G['CURRENT_JOB_YRS']
```

```
#Se define model como la función de regresión lineal
from sklearn.linear_model import LinearRegression
model= LinearRegression()
```

```
#Verificamos la función relacionada al modelo
type(model)
```

```
#Ajustamos el modelo con las variables antes declaradas
model.fit(X=Vars_Indep, y=Var_Dep)
```

Por último, se verificaron los coeficientes obtenidos por el modelo ajustado, se utilizó la función '.predict' para predecir los valores totales e insertamos la columna de predicciones al dataframe.

```
#Verificamos los coeficientes obtenidos para el modelo ajustado
model.__dict__
```

```
#Predecimos los valores de total de accidentes a partir de la variable "alcohol"
#y_pred= model.predict(X=df[['alcohol', 'speeding']])
y_pred= model.predict(X=G[['Experience']])
y_pred
```

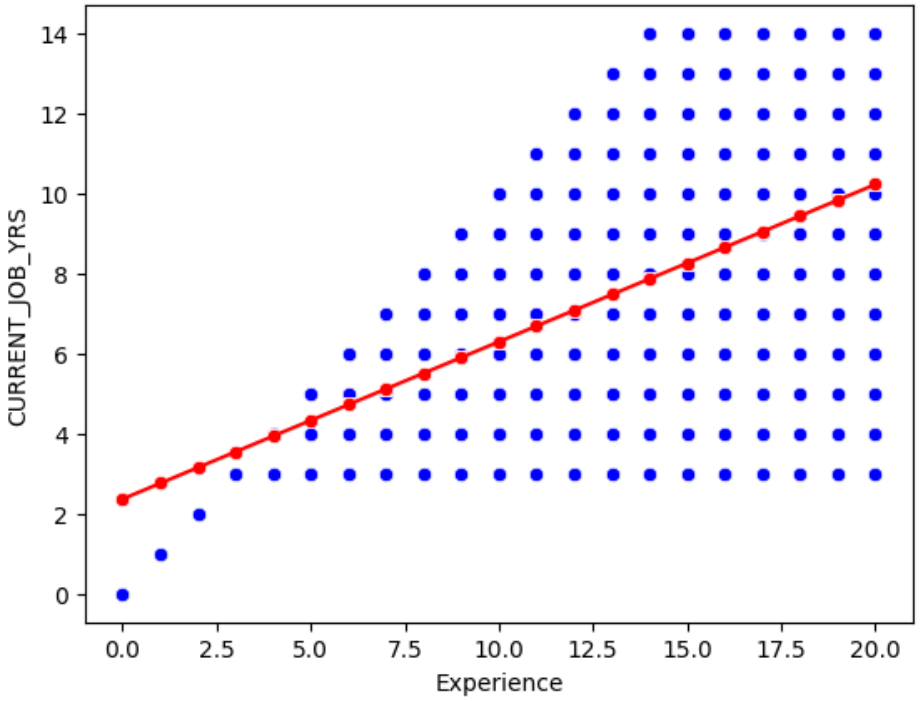
```
#Insertamos la columna de predicciones en el DataFrame
G.insert(0, 'Predicciones', y_pred)
G
```

	Predicciones	Income	Age	Experience	CURRENT_JOB_YRS	CURRENT_HOUSE_YRS	Risk_Flag
0	3.552840	1303834	23	3	3	13	0
1	6.300731	7574516	40	10	9	13	0
2	3.945396	3991815	66	4	4	10	0
3	3.160284	6256451	41	2	2	12	1
4	6.693287	5768871	47	11	3	14	1
...	...	...	...	...	...	...	...
251995	7.478399	8154883	43	13	6	11	0
251996	6.300731	2843572	26	10	6	11	0
251997	5.123063	4522448	46	7	7	12	0
251998	2.375172	6507128	45	0	0	10	0
251999	9.048622	9070230	70	17	7	11	0

Posteriormente, se realizó el diagrama de dispersión, utilizando estas dos variables con el siguiente código:

```
sns.scatterplot(x='Experience', y='CURRENT_JOB_YRS', color="blue", data=G)
sns.scatterplot(x='Experience', y='Predicciones', color="red", data=G)
sns.lineplot(x='Experience', y='Predicciones', color="red", data=G)
```

Y se obtuvo la siguiente gráfica de dispersión:



Por último, se realizó el siguiente código, para corroborar el coeficiente de determinación de nuestro modelo:

```
1 #Corroboramos cual es el coeficiente de Determinación de nuestro modelo
2 coef_Deter=model.score(X=Vars_Indep, y=Var_Dep)
3 coef_Deter
```

```
1 #Corroboramos cual es el coeficiente de Correlación de nuestro modelo
2 coef_Correl=np.sqrt(coef_Deter)
3 coef_Correl
```

```
0.6460975168038419
```

Cabe destacar que para este modelo el nivel de correlación fue de 64.61%, además de tener un coeficiente de determinación de 41.74%.

6. Modelo de regresión múltiple

En este punto, lo primero que se hizo fue definir las variables independientes y la variable dependiente. Para el primer caso, se definió a la variable ‘Age’ como la única dependiente y como las variables independientes a 'Income', 'Experience', 'CURRENT_JOB_YRS', 'CURRENT_HOUSE_YRS', y 'Risk_Flag'. Se volvió a definir model como la función de regresión lineal, se ajustó el modelo, se verificaron los coeficientes obtenidos y se evaluó la eficiencia del modelo.

```
#Declaramos las variables dependientes e independientes para la regresión lineal
Vars_Indep= df[['Income', 'Experience', 'CURRENT_JOB_YRS', 'CURRENT_HOUSE_YRS', 'Risk_Flag']]
Var_Dep= df['Age']
```

```
#Se define model como la función de regresión lineal
from sklearn.linear_model import LinearRegression
model= LinearRegression()
```

```
#Ajustamos el modelo con las variables antes declaradas
model.fit(X=Vars_Indep, y=Var_Dep)
```

```
#Verificamos los coeficientes obtenidos para el modelo ajustado
model.__dict__
```

```
#Evaluamos la eficiencia del modelo obtenido por medio del coeficiente R Determinación
model.score(Vars_Indep,Var_Dep)
```

```
0.0009016476659863271
```

Después se aplicó la función ‘.predict’ para todas las variables independientes que se definieron anteriormente, y se añadió la variables ‘Predicciones_Age’ al dataframe.

```
#Predecimos los valores de total de accidentes a partir de las variables: "alcohol", "speeding" y "no_previous"
y_pred= model.predict(X=df[['Income', 'Experience', 'CURRENT_JOB_YRS', 'CURRENT_HOUSE_YRS', 'Risk_Flag']])
y_pred
```

```
array([49.8820715 , 49.89838223, 50.61745948, ..., 50.15215512,
       50.56710831, 50.24522988])
```

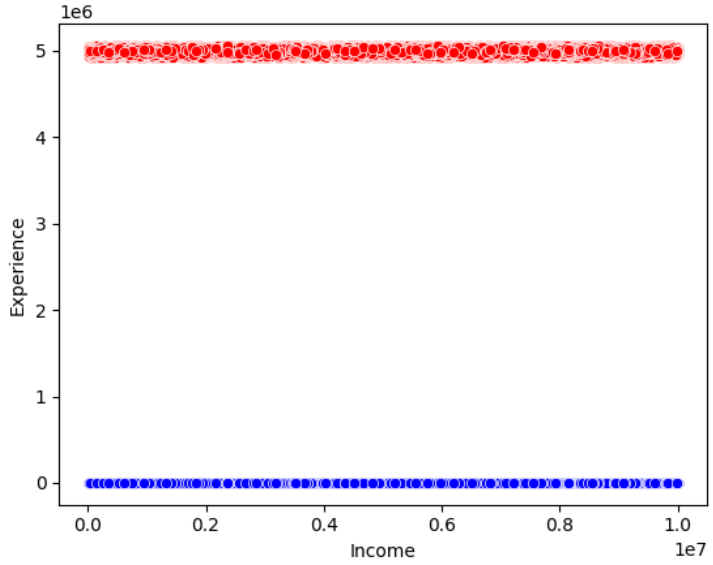
```
#Insertamos la columna de predicciones en el DataFrame
df.insert(0, 'Predicciones_Age', y_pred)
df
```

	Predicciones_Age	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	Profession	CITY	STATE	CURRENT_JOB_YRS	CURRENT_HOUSE_YRS	Risk_Flag
0	49.882071	1	1303834	23	3	single	rented	no	Mechanical_engineer	Rewa	Madhya_Pradesh	3	13	0
1	49.898382	2	7574516	40	10	single	rented	no	Software_Developer	Parbhani	Maharashtra	9	13	0
2	50.617459	3	3991815	66	4	married	rented	no	Technical_writer	Alappuzha	Kerala	4	10	0
3	48.953852	4	6256451	41	2	single	rented	yes	Software_Developer	Bhubaneswar	Odisha	2	12	1
4	48.368124	5	5768871	47	11	single	rented	no	Civil_servant	Tiruchirappalli[10]	Tamil_Nadu	3	14	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
251995	50.279343	251996	8154883	43	13	single	rented	no	Surgeon	Kolkata	West_Bengal	6	11	0
251996	50.343300	251997	2843572	26	10	single	rented	no	Army_officer	Rewa	Madhya_Pradesh	6	11	0
251997	50.152155	251998	4522448	46	7	single	rented	no	Design_Engineer	Kalyan-Dombivli	Maharashtra	7	12	0
251998	50.567108	251999	6507128	45	0	single	rented	no	Graphic_Designer	Pondicherry	Puducherry	0	10	0
251999	50.245230	252000	9070230	70	17	single	rented	no	Statistician	Avadi	Tamil_Nadu	7	11	0

Ya con esto, se comenzaron a sacar las gráficas de dispersión, modificando las variables en cada caso. Es importante saber, que para cada gráfica, se tienen que repetir los códigos anteriores del punto 6, cambiando la variable dependiente, también cambiando el nombre de la variable añadida al dataframe por el nombre ‘Predicciones_[**Variable dependiente de este**

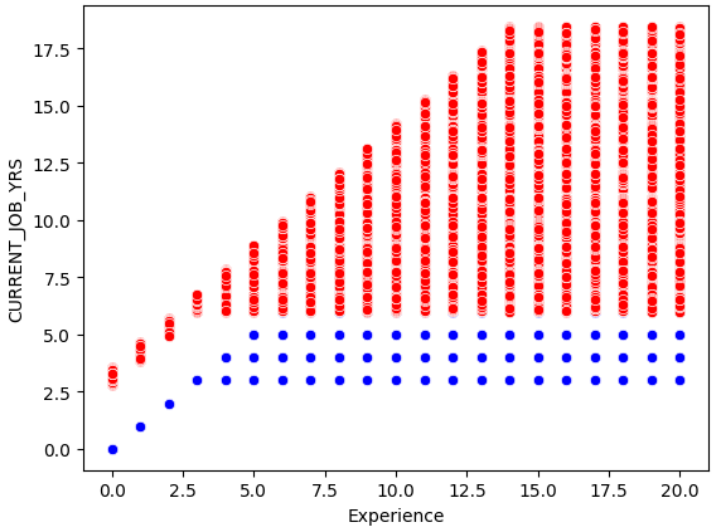
caso]', y por último cambiando las variables a las correctas al momento de realizar la gráfica. Con esto se obtuvieron los siguientes resultados:

1. x = 'Income', y = 'Experience'.
    - Coeficiente de determinación = 7.086061455907622e-05.
    - Coeficiente de correlación = 0.008417874705593818.



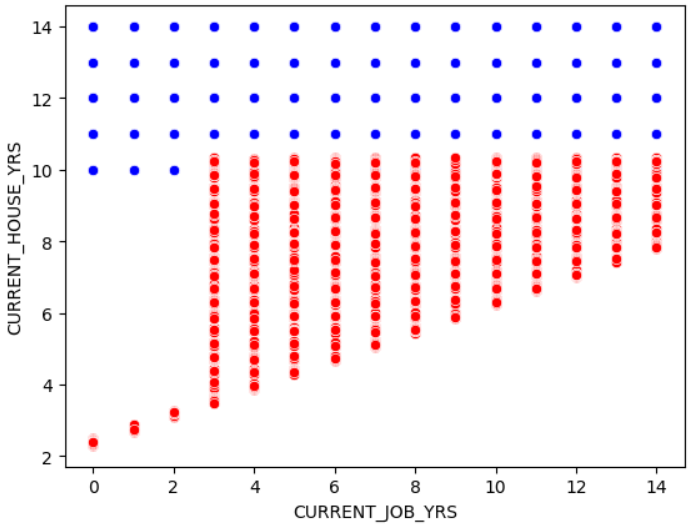
2. x = 'Experience', y = 'CURRENT_JOB_YRS'

- Coeficiente de determinación = 0.41825638424757683.
- Coeficiente de correlación = 0.646727442009056.



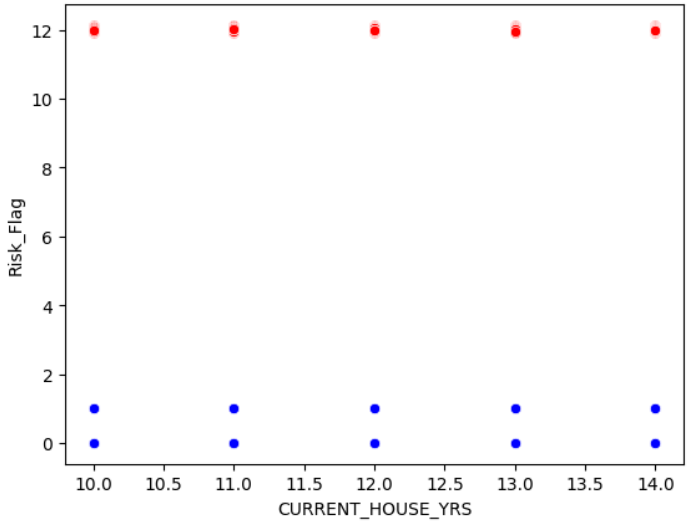
3. x = 'CURRENT_JOB_YRS', y = 'CURRENT_HOUSE_YRS'

- Coeficiente de determinación = 0.41753751840360886.
- Coeficiente de correlación = 0.6461714311261438.



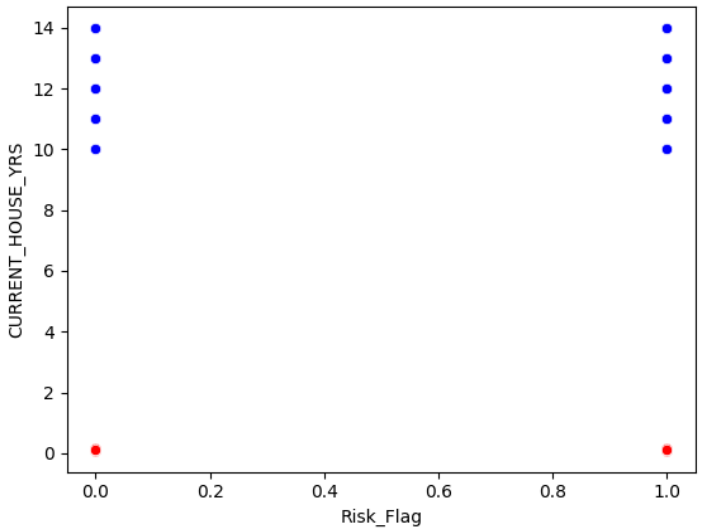
4. x = ‘CURRENT_HOUSE_YRS’, y = ‘Risk_Flag’

- Coeficiente de determinación = 0.0008855691515357034.
- Coeficiente de correlación = 0.02975851393359056.



5. x = ‘Risk_Flag’, y = ‘CURRENT_HOUSE_YRS’

- Coeficiente de determinación = 0.0017448954344835288.
- Coeficiente de correlación = 0.04177194554343296.



7. Regresión lineal

En este punto, se realizaron regresiones no lineales entre muchas combinaciones de las variables, y se encontraron sus coeficientes de correlación y de determinación para comparar los resultados con los de las regresiones lineales y regresiones múltiples realizadas en los puntos anteriores. Para realizar las regresiones no lineales, se utilizaron los siguientes códigos:

Regresión No Lineal con "Income" y "CURRENT_JOB_YRS"

```
[ ]  #Declaramos las variables dependientes e independientes para la regresión No lineal
     Vars_Indep= df[['CURRENT_JOB_YRS']]
     Var_Dep= df['Income']
```

```
[ ]  #Redefinimos las variables
     x= Vars_Indep
     y= Var_Dep
```

```
[ ]  def  func1 (x, a, b, c):
         return  (a*x**2 + b)/ c*x
```

```
[ ]  #Ajustamos los parámetros de la función curve_fit
     parametros, covs= curve_fit(func1, df['CURRENT_JOB_YRS'], df['Income'])
```

```
     #Obtenemos los coeficientes del modelo de regresión no lineal
     parametros
```

```
[ ]  #Creamos el modelo de predicción con los parámetros obtenidos
     parametros, _ = curve_fit(func1, df['CURRENT_JOB_YRS'], df['Income'])
     a, b, c = parametros[ 0 ], parametros[ 1 ], parametros[ 2 ]
     yfit1 = (a*x**2 + b)/ c*x
```

```
[ ]  #Calculamos las predicciones y reestructuramos el vector de predicciones
     #x= x.reshape(51, 1)
     #y = y.reshape(51, 1)
     #yfit1 = yfit1.reshape(51, 1)
     yfit1
```

```
#Graficamos las predicciones y los datos originales para realizar la comparación
plt.plot(x, y, 'bo', label="y-original")
plt.plot(x, yfit1, label="(a*x**2 + b)/ c*x")
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc='best', fancybox=True, shadow=True)
plt.grid(True)
plt.show()
```

```
#Calculamos el coeficiente de determinación del modelo

R2 = r2_score(y, yfit1)
R2
```

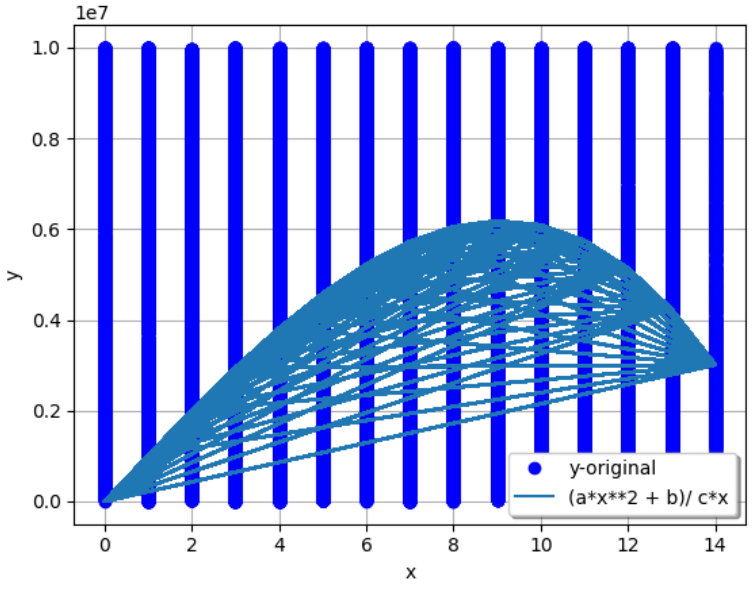
```
-0.41053748853059746
```

Con estos, se definieron la variable dependiente e independiente, se aplicó una función, se ajustaron los parámetros, se creó el modelo de predicción, se graficó, y se calcularon los coeficientes de determinación y correlación.

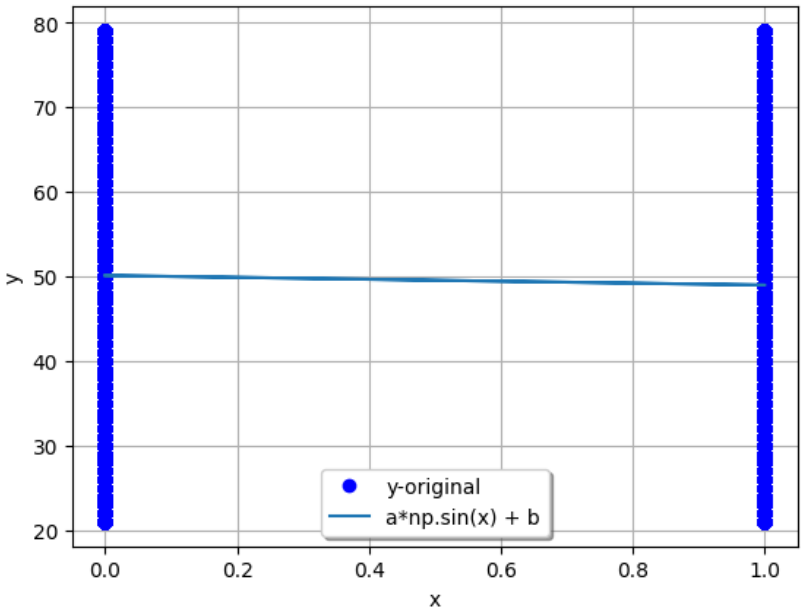
Estos mismos códigos se repiten para las demás variables, haciendo sus respectivos ajustes, como definiendo bien la variable dependiente e independiente de cada caso, y modificando la función del modelo, en busca de los mejores resultados de los coeficientes de determinación y correlación entre las variables.

Los resultados de las gráficas obtenidas fueron los siguientes:

1. ‘CURRENT_JOBS_YRS’ vs ‘Income’



2. ‘Risk_Flag’ vs ‘Age’



3.