Practica II: Visualización de datos

Equipo número 5

Grupo: 03, Lunes

1595894 GONZALEZ CAMPOS EDSON ALI 1941521 LOPEZ DOMINGUEZ FRANCISCO EVERARDO 1663288 HERRERA RIVERA JENNIFER JACQUELINE 1981779 SEGOVIA GONZÁLEZ ANAKAREN

1) Titulo de la base de datos BankChurners

a) bankChuners

b) https://www.kaggle.com/salamatoto/bankchuners

In [41]:

```
import pandas as pd
import numpy as np
import json
import seaborn as sns
import matplotlib.pyplot as plt
```

Mandamos llamar nuesta base de datos

In [4]:

```
df = pd.read_csv("BankChurners.csv",header = 0,sep = ",")

df
```

Out[4]:

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Catego
0	768805383	Existing Customer	45	M	3	High School	Married	$60K-8$
1	818770008	Existing Customer	49	F	5	Graduate	Single	Less than $4
2	713982108	Existing Customer	51	M	3	Graduate	Married	$80K-12$
3	769911858	Existing Customer	40	F	4	High School	Unknown	Less than $4
4	709106358	Existing Customer	40	M	3	Uneducated	Married	$60K-8$
...	...	...	...	...	...	...	...	
10122	772366833	Existing Customer	50	M	2	Graduate	Single	$40K-6$
10123	710638233	Attrited Customer	41	M	2	Unknown	Divorced	$40K-6$
10124	716506083	Attrited Customer	44	F	1	High School	Married	Less than $4
10125	717406983	Attrited Customer	30	M	2	Graduate	Unknown	$40K-6$
10126	714337233	Attrited Customer	43	F	2	Graduate	Married	Less than $4

10127 rows × 23 columns

Eliminamos las columnas inecesarias en nuestra base de datos, son columnas que no se necesitaran pues son muy complicadas de definir

In [5]:

```
df = df.drop(['Attrition_Flag','Total_Amt_Chng_Q4_Q1','Total_Trans_Amt','Total_Trans_Ct'
,'Total_Ct_Chng_Q4_Q1','Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Coun
t_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_1'],axis=1)
```

Para facilitar la manipulación cambiamos los nombres de las columnas

In [6]:

```
df = df.rename(columns = {'CLIENTNUM':'NumeroDCliente'})
df = df.rename(columns = {'Customer_Age':'EdadDCliente'})
df = df.rename(columns = {'Gender':'Genero'})
df = df.rename(columns = {'Dependent_count':'CuentasDependientes'})
df = df.rename(columns = {'Education_Level':'NivelDEducacion'})
df = df.rename(columns = {'Marital_Status':'EstadoCivil'})
df = df.rename(columns = {'Income_Category':'IntervaloDIngresos'})
df = df.rename(columns = {'Card_Category':'NivelDTarjeta'})
df = df.rename(columns = {'Months_on_book':'MesesEnLibro'})
df = df.rename(columns = {'Total_Relationship_Count':'CuentasRelacionadas'})
df = df.rename(columns = {'Months_Inactive_12_mon':'MesesInactivoEnUnAnho'})
df = df.rename(columns = {'Contacts_Count_12_mon':'ContactoConLaCuenta'})
df = df.rename(columns = {'Credit_Limit':'LimiteCrediticio'})
df = df.rename(columns = {'Total_Revolving_Bal':'TotalDSaldoRotatorio'})
df = df.rename(columns = {'Avg_Open_To_Buy':'SaldoDisponible'})
df = df.rename(columns = {'Avg_Utilization_Ratio':'PorcentajeUtilizado'})
df = df.rename(columns = {'Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_
Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2':'ClasificacionNaiv
eBayes'})
print(df)
```

```
       NumeroDCliente  EdadDCliente Genero  CuentasDependientes  \
0           768805383            45      M                    3
1           818770008            49      F                    5
2           713982108            51      M                    3
3           769911858            40      F                    4
4           709106358            40      M                    3
...               ...           ...    ...                  ...
10122       772366833            50      M                    2
10123       710638233            41      M                    2
10124       716506083            44      F                    1
10125       717406983            30      M                    2
10126       714337233            43      F                    2

      NivelDEducacion EstadoCivil IntervaloDIngresos NivelDTarjeta  \
0         High School     Married        $60K - $80K          Blue
1            Graduate      Single     Less than $40K          Blue
2            Graduate     Married       $80K - $120K          Blue
3         High School     Unknown     Less than $40K          Blue
4          Uneducated     Married        $60K - $80K          Blue
...               ...         ...                 ...          ...
10122        Graduate      Single        $40K - $60K          Blue
10123         Unknown    Divorced        $40K - $60K          Blue
10124     High School     Married     Less than $40K          Blue
10125        Graduate     Unknown        $40K - $60K          Blue
10126        Graduate     Married     Less than $40K        Silver

       MesesEnLibro  CuentasRelacionadas  MesesInactivoEnUnAnho  \
0                39                   5                      1
1                44                   6                      1
2                36                   4                      1
3                34                   3                      4
4                21                   5                      1
...             ...                 ...                    ...
10122            40                   3                      2
```

```
10123            25                    4                       2
10124            36                    5                       3
10125            36                    4                       3
10126            25                    6                       2

       ContactoConLaCuenta  LimiteCrediticio  TotalDSaldoRotatorio  \
0                        3           12691.0                   777
1                        2            8256.0                   864
2                        0            3418.0                     0
3                        1            3313.0                  2517
4                        0            4716.0                     0
...                    ...               ...                   ...
10122                    3            4003.0                  1851
10123                    3            4277.0                  2186
10124                    4            5409.0                     0
10125                    3            5281.0                     0
10126                    4           10388.0                  1961

       SaldoDisponible  PorcentajeUtilizado  ClasificacionNaiveBayes
0              11914.0                0.061                 0.999910
1               7392.0                0.105                 0.999940
2               3418.0                0.000                 0.999980
3                796.0                0.760                 0.999870
4               4716.0                0.000                 0.999980
...                ...                  ...                      ...
10122           2152.0                0.462                 0.999810
10123           2091.0                0.511                 0.004729
10124           5409.0                0.000                 0.002118
10125           5281.0                0.000                 0.003294
10126           8427.0                0.189                 0.003377

[10127 rows x 17 columns]
```

Para saber si tenemos datos nulos en nuestra base de datos

In [7]:

```
df.isnull().any()
```

Out[7]:

```
NumeroDCliente             False
EdadDCliente               False
Genero                     False
CuentasDependientes        False
NivelDEducacion            False
EstadoCivil                False
IntervaloDIngresos         False
NivelDTarjeta              False
MesesEnLibro               False
CuentasRelacionadas        False
MesesInactivoEnUnAnho      False
ContactoConLaCuenta        False
LimiteCrediticio           False
TotalDSaldoRotatorio       False
SaldoDisponible            False
PorcentajeUtilizado        False
ClasificacionNaiveBayes    False
dtype: bool
```

Ejercicios de estadística básica

Para averiguar el total de contactos que han hecho nuestros clientes en el año con su cuenta

In [8]:

```
TotCon=df['ContactoConLaCuenta'].sum()
SumAcum=df['ContactoConLaCuenta'].cumsum()
```

```
print("La suma acumulada de contactos que nuestros clientes han hecho con sus cuentas en un año es:", SumAcum)
print("La cantidad total de contactos que nuestros clientes han hecho con sus cuentas en un año es:", TotCon)
```

```
La suma acumulada de contactos que nuestros clientes han hecho con sus cuentas en un año es: 0            3
1            5
2            5
3            6
4            6
         ...
10122    24851
10123    24854
10124    24858
10125    24861
10126    24865
Name: ContactoConLaCuenta, Length: 10127, dtype: int64
La cantidad total de contactos que nuestros clientes han hecho con sus cuentas en un año es: 24865
```

Usamos la columna de EdadDCliente para averiguar si nuestro banco es mas utilizado por jovenes, adultos o adultos mayores

In [9]:

```
prom = df['EdadDCliente'].mean()
EdadMin = df['EdadDCliente'].min()
EdadMax = df['EdadDCliente'].max()
Mediana = df['EdadDCliente'].median()


print("El promedio de edad de clientes que usan el banco es:", prom)
print("La edad minima de clientes que usan el banco es:", EdadMin)
print("La edad maxima de clientes que usan el banco es:", EdadMax)
print("La mediana de edad de los clientes que usan el banco es:", Mediana)
```

```
El promedio de edad de clientes que usan el banco es: 46.32596030413745
La edad minima de clientes que usan el banco es: 26
La edad maxima de clientes que usan el banco es: 73
La mediana de edad de los clientes que usan el banco es: 46.0
```

Para darnos una idea mas amplia de la columna SaldoDisponible usaremos la función describe

In [10]:

```
df["SaldoDisponible"].describe()
```

Out[10]:

```
count    10127.000000
mean      7469.139637
std       9090.685324
min          3.000000
25%       1324.500000
50%       3474.000000
75%       9859.000000
max      34516.000000
Name: SaldoDisponible, dtype: float64
```

Esto nos dice que la media de sldo disponible es de 7469.13 con un minimo de 3 y un maximo de 34516, siendo valores muy alejados uno del otro

Ahora buscaremos explicarla columna LimiteCrediticio

In [11]:

```
Mediana2 = df['LimiteCrediticio'].median()
Var = df['LimiteCrediticio'].var()
```

```
DesvEst = df['LimiteCrediticio'].std()
Asi = df['LimiteCrediticio'].skew()
Curt = df['LimiteCrediticio'].kurt()


print("La mediana es:", Mediana2)
print("La varianza es:", Var)
print("La desviación estándar es:", DesvEst)
print("El valor de asimetría de la columna es:", Asi)
print("El valor de curtosis es:", Curt)
```

```
La mediana es: 4549.0
La varianza es: 82605860.99764086
La desviación estándar es: 9088.776650223113
El valor de asimetría de la columna es: 1.666725807993647
El valor de curtosis es: 1.8089893357093434
```

EL valor de la curtosis nos dice que en la curva de la normal nuestros datos se encuentran muy dispersos, mientras que nuestra asimetria es positiva lo que nos indica que hay mas valores arriba de la media que abajo de esta.

Tabla de correlación de los datos

In [12]:

```
df.corr()
```

Out[12]:

	NumeroDCliente	EdadDCliente	CuentasDependientes	MesesEnLibro	CuentasRelacionadas	MesesIn
NumeroDCliente	1.000000	0.007613	0.006772	0.134588	0.006907	
EdadDCliente	0.007613	1.000000	-0.122254	0.788912	-0.010931	
CuentasDependientes	0.006772	-0.122254	1.000000	-0.103062	-0.039076	
MesesEnLibro	0.134588	0.788912	-0.103062	1.000000	-0.009203	
CuentasRelacionadas	0.006907	-0.010931	-0.039076	-0.009203	1.000000	
MesesInactivoEnUnAnho	0.005729	0.054361	-0.010768	0.074164	-0.003675	
ContactoConLaCuenta	0.005694	-0.018452	-0.040505	-0.010774	0.055203	
LimiteCrediticio	0.005708	0.002476	0.068065	0.007507	-0.071386	
TotalDSaldoRotatorio	0.000825	0.014780	-0.002688	0.008623	0.013726	
SaldoDisponible	0.005633	0.001151	0.068291	0.006732	-0.072601	
PorcentajeUtilizado	0.000266	0.007114	-0.037135	-0.007541	0.067663	
ClasificacionNaiveBayes	0.046410	-0.018189	-0.019189	-0.013694	0.149981	

In [13]:

```
df.cov()
```

Out[13]:

	NumeroDCliente	EdadDCliente	CuentasDependientes	MesesEnLibro	CuentasRelacionadas	MesesIn
NumeroDCliente	1.361889e+15	2.252209e+06	324600.732489	3.966693e+07	396204.467439	
EdadDCliente	2.252209e+06	6.426931e+01	-1.273041	5.051060e+01	-0.136216	
CuentasDependientes	3.246007e+05	-1.273041e+00	1.687163	-1.069129e+00	-0.078897	
MesesEnLibro	3.966693e+07	5.051060e+01	-1.069129	6.378285e+01	-0.114248	
CuentasRelacionadas	3.962045e+05	-1.362164e-01	-0.078897	-1.142484e-01	2.416184	

	NumeroDCliente	EdadDCliente	CuentasDependientes	MesesEnLibro	CuentasRelacionadas	MesesIn
MesesInactivoEnUnAnho	2.136500e+05	4.404313e-01	-0.014135	5.985924e-01	-0.005774	
ContactoConLaCuenta	2.324687e+05	-1.636385e-01	-0.058201	-9.519009e-02	0.094923	
LimiteCrediticio	1.914395e+09	1.804254e+02	803.535812	5.449094e+02	-1008.514979	
TotalDSaldoRotatorio	2.479788e+07	9.656595e+01	-2.845655	5.612435e+01	17.388217	
SaldoDisponible	1.889597e+09	8.385949e+01	806.381467	4.887851e+02	-1025.903197	
PorcentajeUtilizado	2.705892e+03	1.572362e-02	-0.013298	-1.660332e-02	0.028996	
ClasificacionNaiveBayes	6.256592e+05	-5.326626e-02	-0.009105	-3.995006e-02	0.085163	

In [ ]:

```
```

Graficar datos

Grafica de barras

Para los ejercicios de grafica de datos usare la base de datos previamente limpiada en la actividad pasada para que sea mas simple su uso

In [14]:

```
df1 = pd.read_csv("BankChurnersMod.csv",header = 0,sep = ",")

df1
```

Out[14]:

	Unnamed: 0	NumeroDCliente	EdadDCliente	Genero	CuentasDependientes	NivelDEducacion	EstadoCivil	IntervaloDIngr
0	0	768805383	45	M	3	High School	Married	$60K-$
1	1	818770008	49	F	5	Graduate	Single	Less than
2	2	713982108	51	M	3	Graduate	Married	$80K-$
3	3	769911858	40	F	4	High School	Unknown	Less than
4	4	709106358	40	M	3	Uneducated	Married	$60K-$
...	...	...	...	...	...	...	...	
10122	10122	772366833	50	M	2	Graduate	Single	$40K-$
10123	10123	710638233	41	M	2	Unknown	Divorced	$40K-$
10124	10124	716506083	44	F	1	High School	Married	Less than
10125	10125	717406983	30	M	2	Graduate	Unknown	$40K-$
10126	10126	714337233	43	F	2	Graduate	Married	Less than

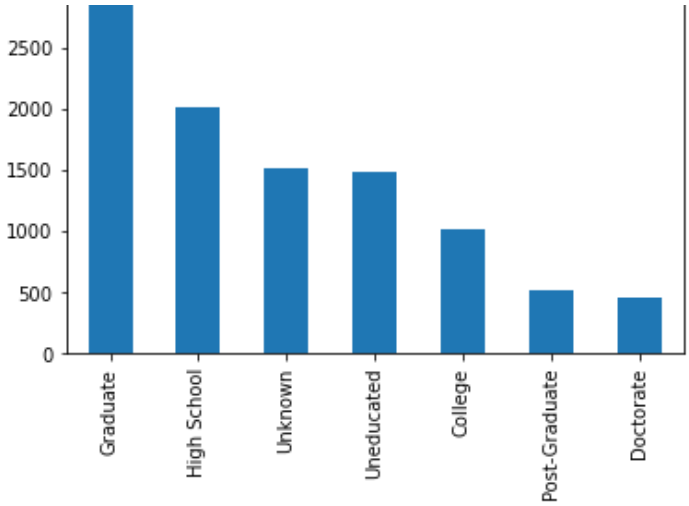
10127 rows × 18 columns

In [15]:

```
df1['NivelDEducacion'].value_counts().plot.bar()
```

Out[15]:

```
<AxesSubplot:>
```



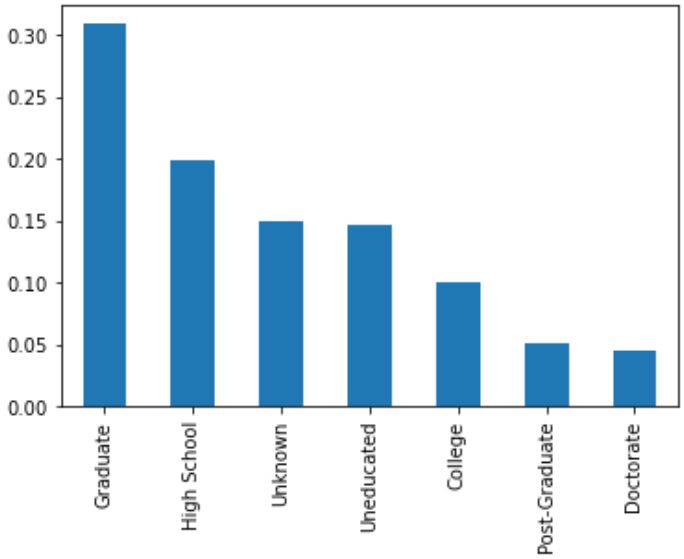
Apreciamos con claridad que la mayoria son graduados de la universidad superando por el doble de los que no recibieron educación o no se sabe la educación que tuvieron

In [16]:

```
(df1['NivelDEducacion'].value_counts() / len(df1)).plot.bar()
```

Out[16]:

<AxesSubplot:>



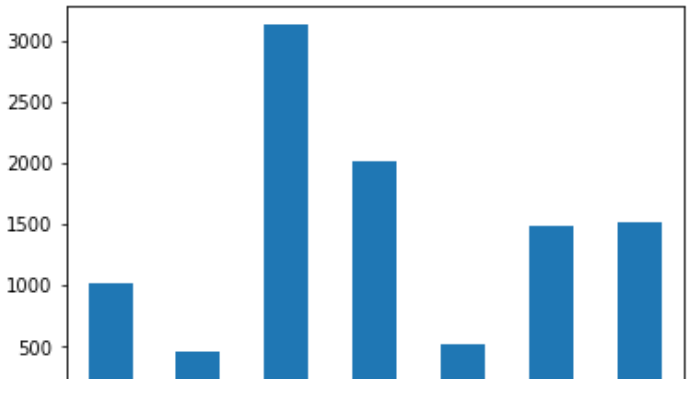
Aqui vemos claramente que los graduados son el 30% de nuestros clientes seguidos por los de high school con el 20%

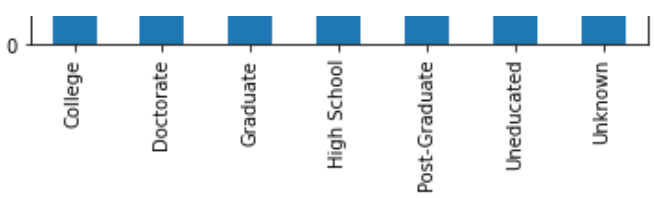
In [17]:

```
df1['NivelDEducacion'].value_counts().sort_index().plot.bar()
```

Out[17]:

<AxesSubplot:>





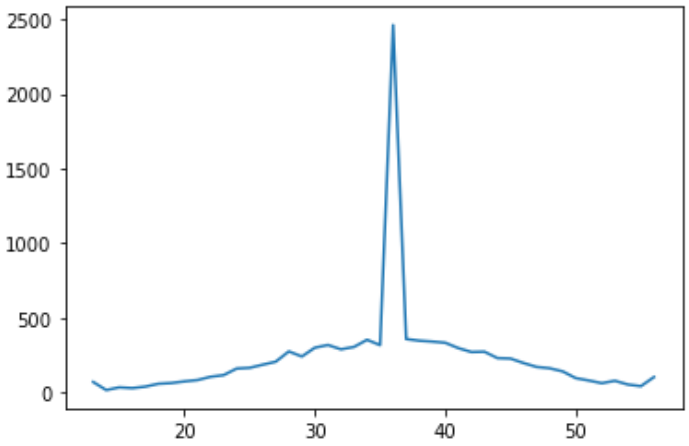
Grafica de lineas

In [18]:

```
df1['MesesEnLibro'].value_counts().sort_index().plot.line()
```

Out[18]:

<AxesSubplot:>



Aqui observamos que hay una clara mayoria de personas que han estado 35 meses en nuestro banco

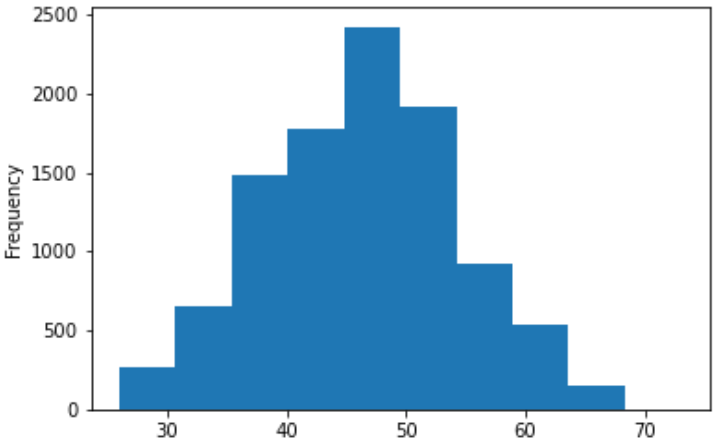
Histogramas

In [19]:

```
df1['EdadDCliente'].plot.hist()
```

Out[19]:

<AxesSubplot:ylabel='Frequency'>

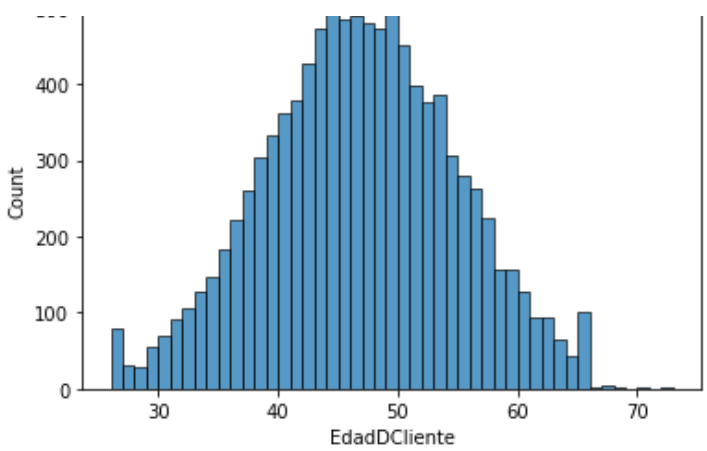


In [20]:

```
sns.histplot(df1["EdadDCliente"])
```

Out[20]:

<AxesSubplot:xlabel='EdadDCliente', ylabel='Count'>



En nuestros histogramas podemos notar que la edad media de nuestros clientes es de 45 años, como lo hav¿biamos averiguado anteriormente por los medios estadisticos

Graficar datos bi-variantes

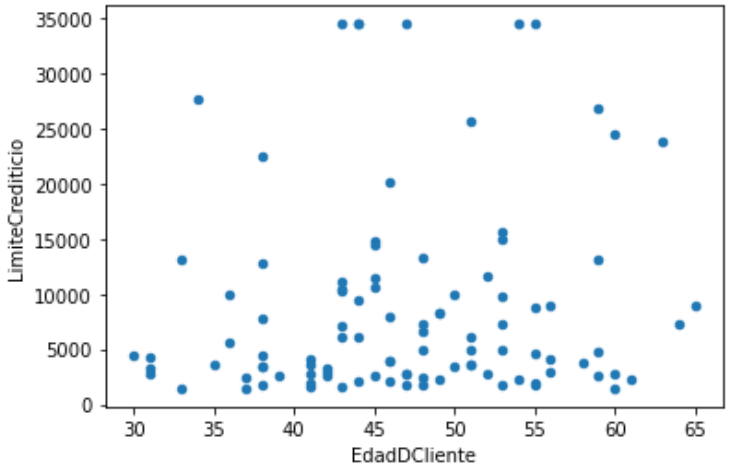
Scatter plot

In [21]:

```
df1[df1['EdadDCliente'] < 100].sample(100).plot.scatter(x='EdadDCliente', y='LimiteCrediticio')
```

Out[21]:

```
<AxesSubplot:xlabel='EdadDCliente', ylabel='LimiteCrediticio'>
```



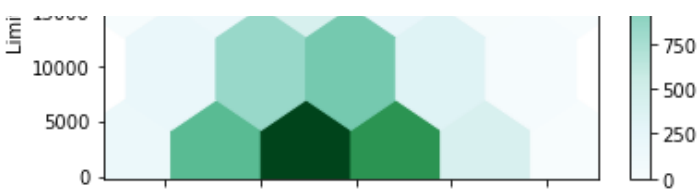
Esta grafica nos dice que cuando las personas estan en la mediana edad tienen un mayor limite crediticio, esto puede ser porque son trabajadores que pueden solventar los gastos.

In [22]:

```
df1[df1['EdadDCliente'] < 100].plot.hexbin(x='EdadDCliente', y='LimiteCrediticio', gridsize=5)
```

Out[22]:

```
<AxesSubplot:xlabel='EdadDCliente', ylabel='LimiteCrediticio'>
```



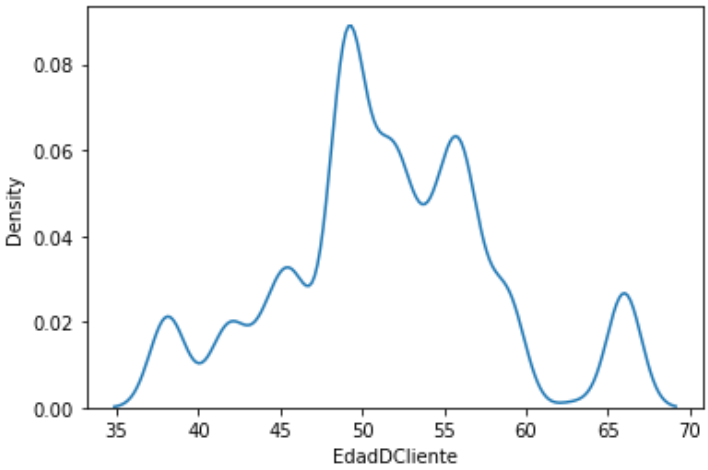
Stacked plots

In [25]:

```
sns.kdeplot(df1.query('EdadDCliente').EdadDCliente)
```

Out[25]:

```
<AxesSubplot:xlabel='EdadDCliente', ylabel='Density'>
```



Como podemos notan hay una mayoria de densidad de datos entre los 45 y los 50 pues es la edad predominante en unestra base de datos

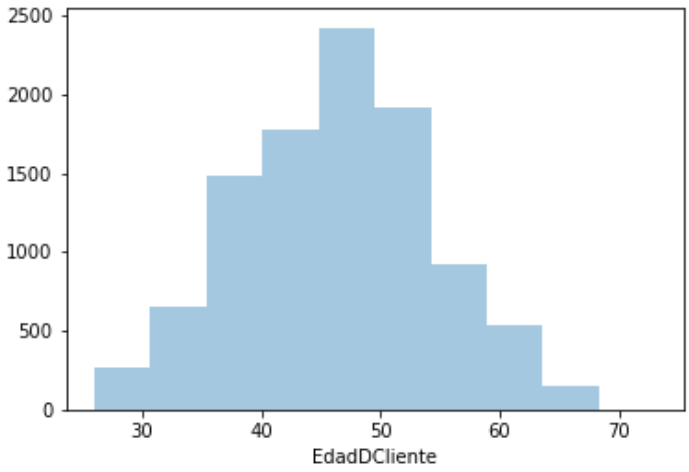
In [26]:

```
sns.distplot(df1['EdadDCliente'], bins=10, kde=False)
```

```
C:\Users\jenny\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[26]:

```
<AxesSubplot:xlabel='EdadDCliente'>
```



Esta grafica nos dice exactamente lo mismo que nuestro histograma

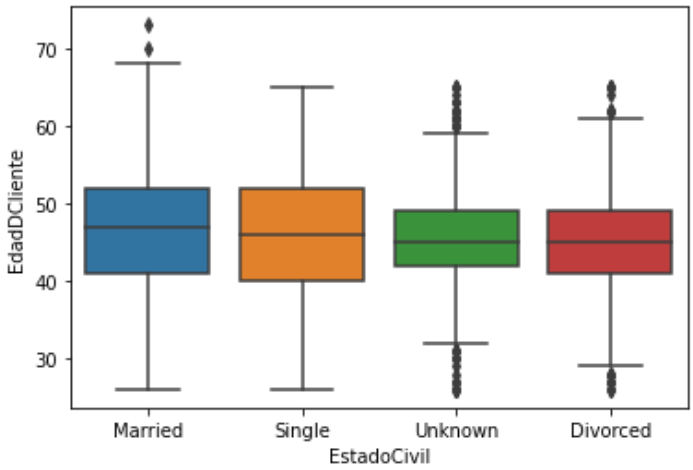
BoxPlots

In [48]:

```
sns.boxplot(
    x='EstadoCivil',
    y='EdadDCliente',
    data=df1
)
```

Out[48]:

```
<AxesSubplot:xlabel='EstadoCivil', ylabel='EdadDCliente'>
```



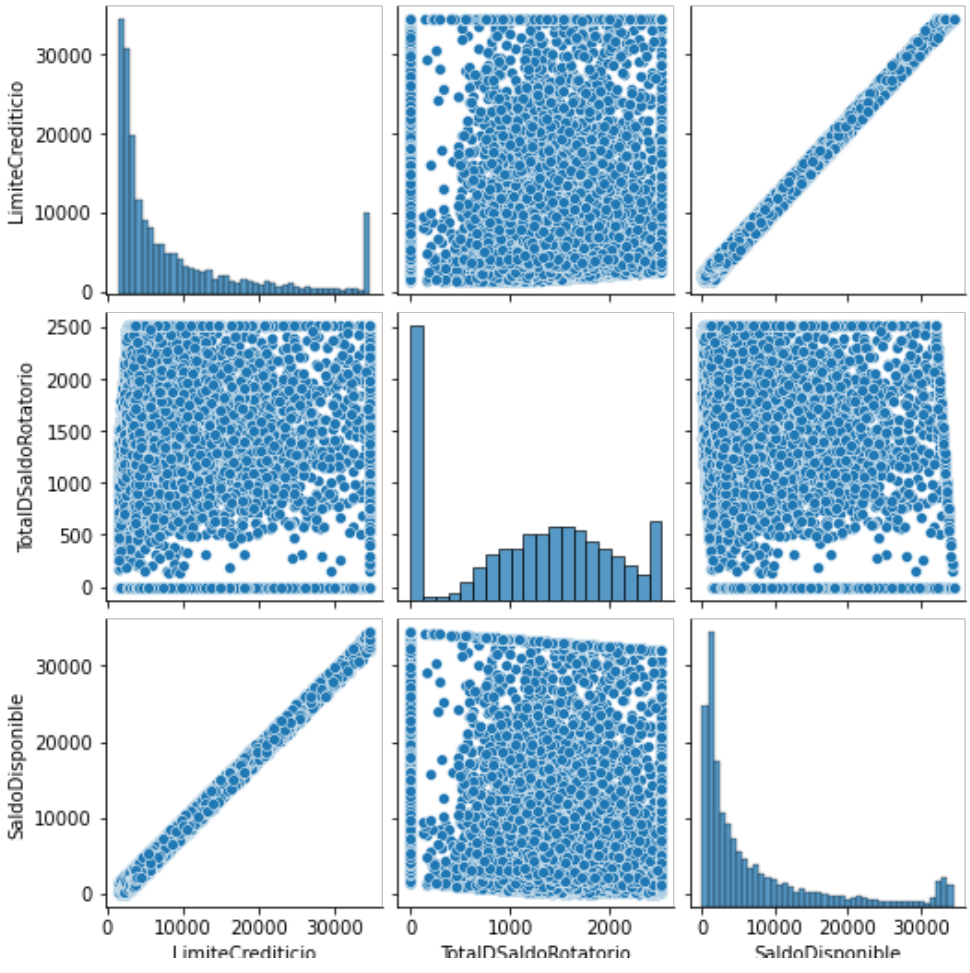
Como podemos ver hay una gran similitud entre personas casadas y solteras en referencia a su edad

In [31]:

```
sns.pairplot(df1[['LimiteCrediticio', 'TotalDSaldoRotatorio', 'SaldoDisponible']])
```

Out[31]:

```
<seaborn.axisgrid.PairGrid at 0x1e607148580>
```



In [ ]:

```
```

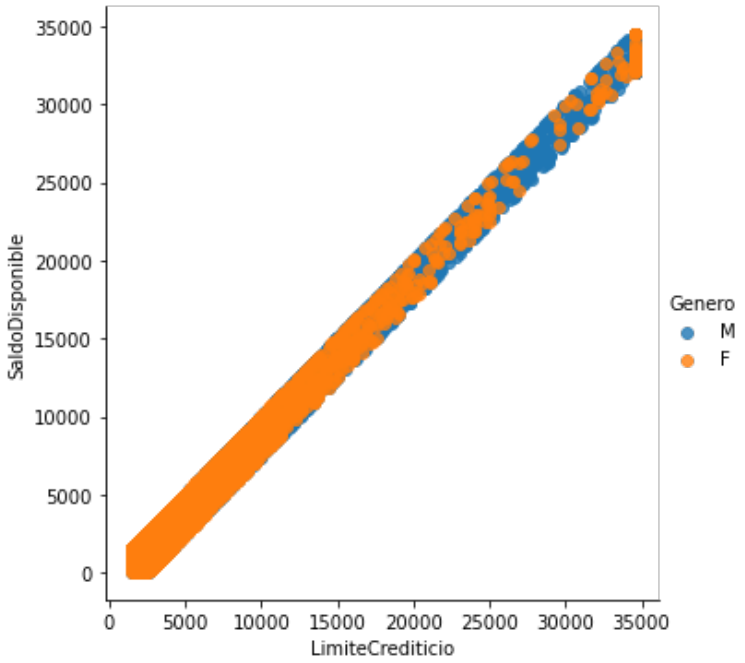
Graficas multi variantes

In [39]:

```
sns.lmplot(x='LimiteCrediticio', y='SaldoDisponible', hue='Genero',
           data=df1.loc[df1['Genero'].isin(['M', 'F'])],
           fit_reg=False)
```

Out[39]:

```
<seaborn.axisgrid.FacetGrid at 0x1e607713880>
```

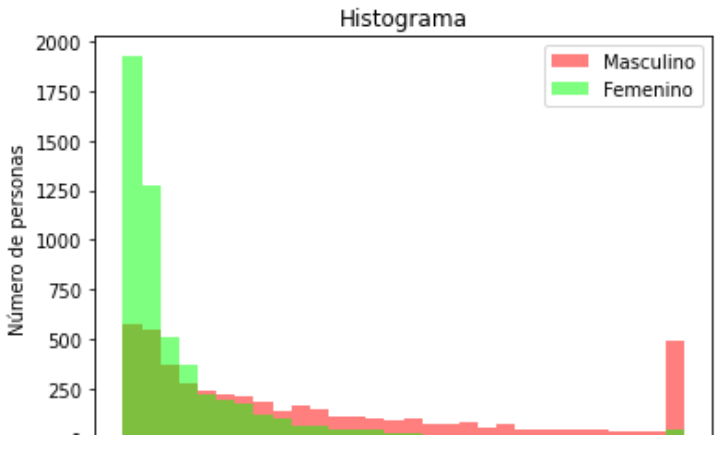


In [ ]:

```
```

In [43]:

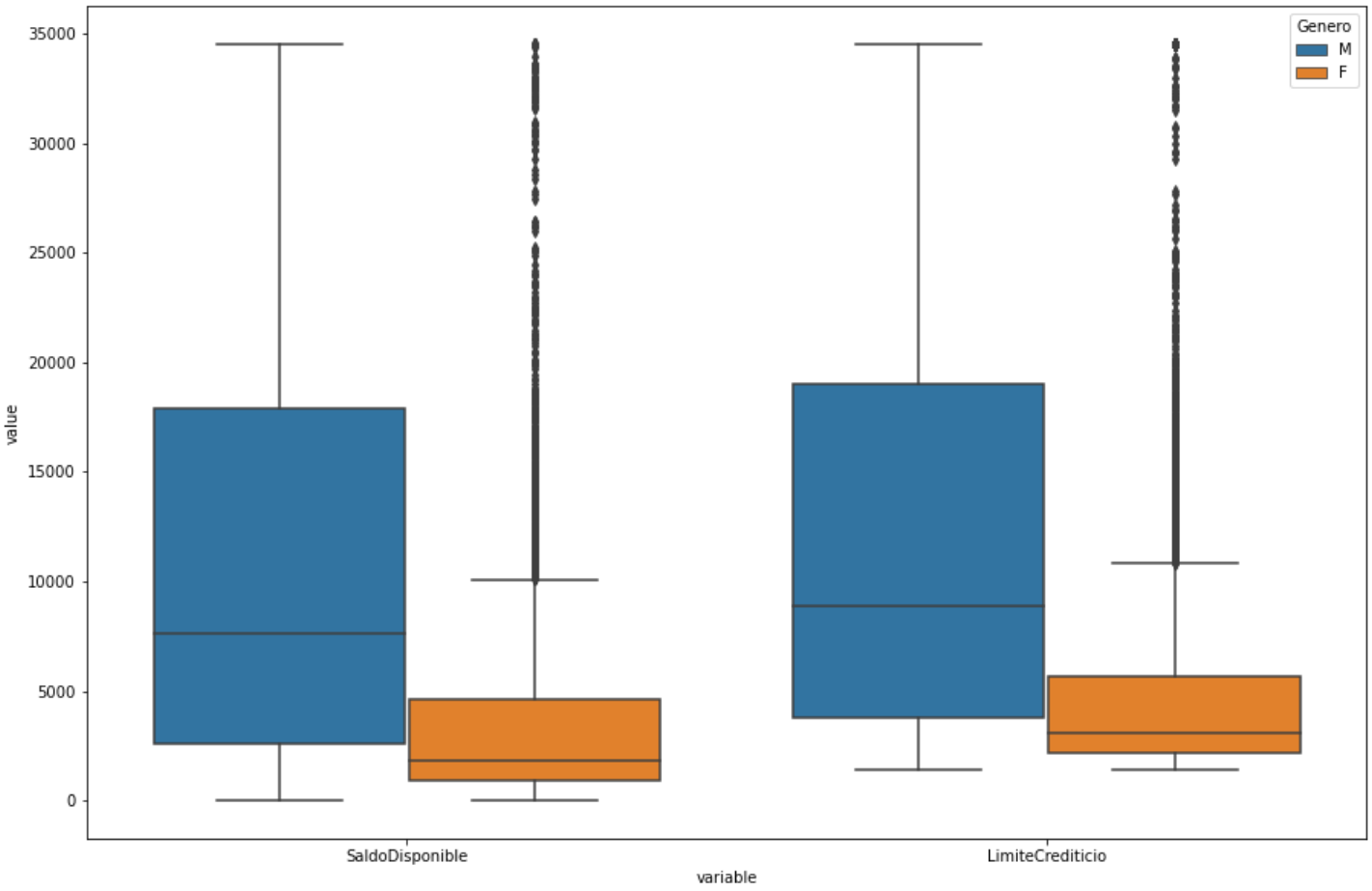
```
m = plt.hist(df1[df1["Genero"] == "M"].LimiteCrediticio,bins=30,fc = (1,0,0,0.5),label =
"Masculino")
f = plt.hist(df1[df1["Genero"] == "F"].LimiteCrediticio,bins=30,fc = (0,1,0,0.5),label =
"Femenino")
plt.legend()
plt.xlabel("Limite de Credito")
plt.ylabel("Número de personas")
plt.title("Histograma")
plt.show()
```



En esta grafica notamos que una mayoria femenina tiene un limite de credito relativamente bajo a comparación de su contraparte masculina que aunque muchos tienen un limite de credito bajo tambien vemos que ellos tienen una gran mayoria en el limite de credito mas alto a diferencia de las mujeres

In [44]:

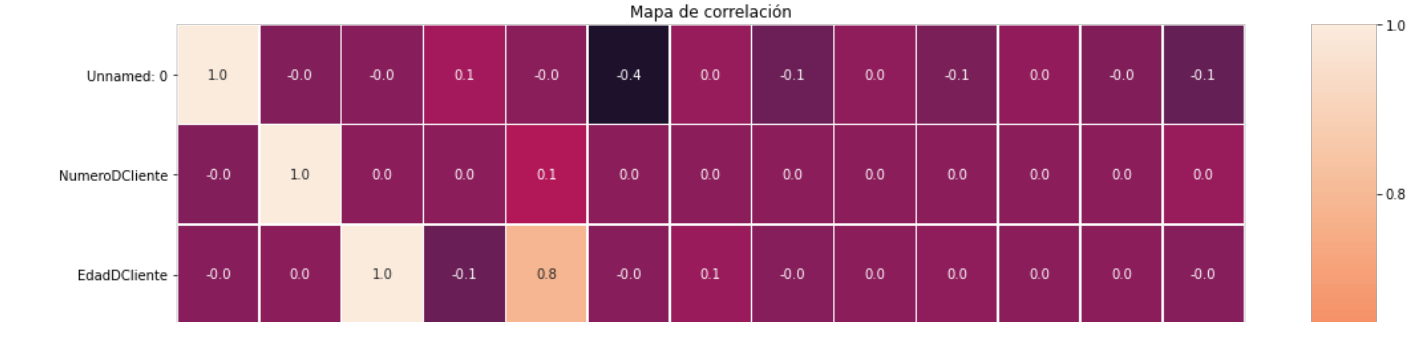
```
mdata = pd.melt(df,id_vars = "Genero",value_vars = ['SaldoDisponible', 'LimiteCrediticio
'])
plt.figure(figsize = (15,10))
sns.boxplot(x = "variable", y = "value", hue="Genero",data= mdata)
plt.show()
```

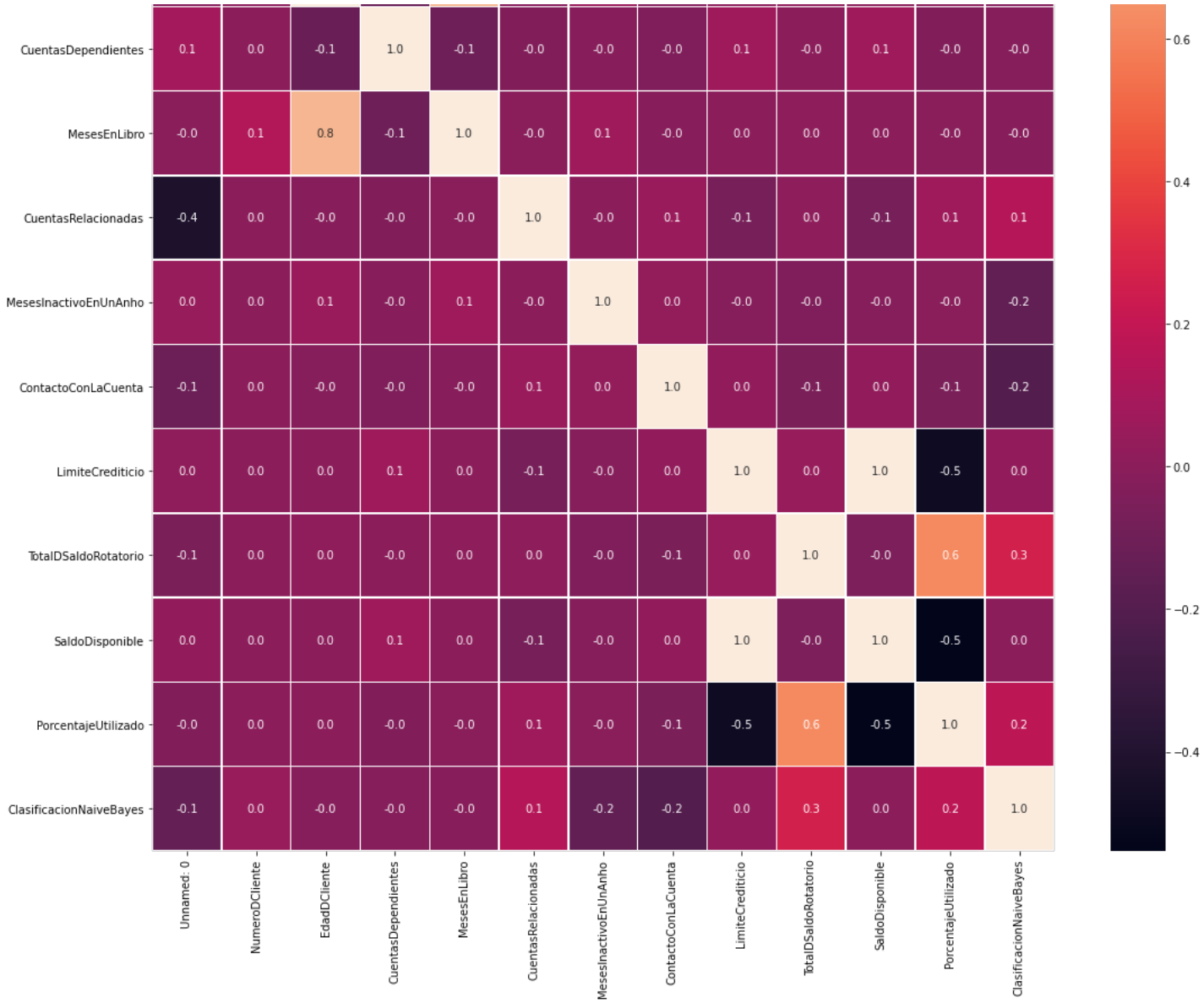


En esta grafica vemos claramente que mientras los hombres tienen un mayor saldo disponible tambien tienen un mayor limite de credito

In [45]:

```
f,ax=plt.subplots(figsize = (18,18))
sns.heatmap(df1.corr(),annot= True,linewidths=0.5,fmt = ".1f",ax=ax)
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.title('Mapa de correlación')
plt.savefig('graph.png')
plt.show()
```




CuentasDependientes 0.1 0.0 -0.1 1.0 -0.1 -0.0 -0.0 -0.0 0.1 -0.0 0.1 -0.0 -0.0
MesesEnLibro -0.0 0.1 0.8 -0.1 1.0 -0.0 0.1 -0.0 0.0 0.0 0.0 -0.0 -0.0
CuentasRelacionadas -0.4 0.0 -0.0 -0.0 -0.0 1.0 -0.0 0.1 -0.1 0.0 -0.1 0.1 0.1
MesesInactivoEnUnAnho 0.0 0.0 0.1 -0.0 0.1 -0.0 1.0 0.0 -0.0 -0.0 -0.0 -0.0 -0.2
ContactoConLaCuenta -0.1 0.0 -0.0 -0.0 -0.0 0.1 0.0 1.0 0.0 -0.1 0.0 -0.1 -0.2
LimiteCrediticio 0.0 0.0 0.0 0.1 0.0 -0.1 -0.0 0.0 1.0 0.0 1.0 -0.5 0.0
TotalDSaldoRotatorio -0.1 0.0 0.0 -0.0 0.0 0.0 -0.0 -0.1 0.0 1.0 -0.0 0.6 0.3
SaldoDisponible 0.0 0.0 0.0 0.1 0.0 -0.1 -0.0 0.0 1.0 -0.0 1.0 -0.5 0.0
PorcentajeUtilizado -0.0 0.0 0.0 -0.0 -0.0 0.1 -0.0 -0.1 -0.5 0.6 -0.5 1.0 0.2
ClasificacionNaiveBayes -0.1 0.0 -0.0 -0.0 -0.0 0.1 -0.2 -0.2 0.0 0.3 0.0 0.2 1.0
Unnamed: 0
NumeroDCliente
EdadDCliente
CuentasDependientes
MesesEnLibro
CuentasRelacionadas
MesesInactivoEnUnAnho
ContactoConLaCuenta
LimiteCrediticio
TotalDSaldoRotatorio
SaldoDisponible
PorcentajeUtilizado
ClasificacionNaiveBayes
0.6
0.4
0.2
0.0
−0.2
−0.4

In [ ]: