Lab1

July 28, 2021

Laboratorio 1 - Data Science 1

Karina Alejandra Valladares Díaz

Carné 18005

José Eduardo López Gómez

Carné 181045

[222]:
```
#importamos las librerías necesarias para trabajar
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from factor_analyzer import FactorAnalyzer
from sklearn.decomposition import PCA
from sklearn import preprocessing
from pandas_profiling import *
from apyori import apriori
```

[223]:
```
#Utilizar únicamente el train

data = pd.read_csv("train.csv") #usamos el Id como índice
data #mostramos la data (1460 registros con 80 variables )
```

[223]:
```
         Id  MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0         1          60       RL         65.0     8450   Pave   NaN      Reg
1         2          20       RL         80.0     9600   Pave   NaN      Reg
2         3          60       RL         68.0    11250   Pave   NaN      IR1
3         4          70       RL         60.0     9550   Pave   NaN      IR1
4         5          60       RL         84.0    14260   Pave   NaN      IR1
…     …          …       …           …       …    …    …       …
1455   1456          60       RL         62.0     7917   Pave   NaN      Reg
1456   1457          20       RL         85.0    13175   Pave   NaN      Reg
1457   1458          70       RL         66.0     9042   Pave   NaN      Reg
1458   1459          20       RL         68.0     9717   Pave   NaN      Reg
1459   1460          20       RL         75.0     9937   Pave   NaN      Reg
```

```
     LandContour Utilities  … PoolArea PoolQC  Fence MiscFeature MiscVal  \
0            Lvl    AllPub  …        0    NaN    NaN         NaN       0
1            Lvl    AllPub  …        0    NaN    NaN         NaN       0
2            Lvl    AllPub  …        0    NaN    NaN         NaN       0
3            Lvl    AllPub  …        0    NaN    NaN         NaN       0
4            Lvl    AllPub  …        0    NaN    NaN         NaN       0
…           …       …  …      …    …    …          …     …
1455         Lvl    AllPub  …        0    NaN    NaN         NaN       0
1456         Lvl    AllPub  …        0    NaN  MnPrv         NaN       0
1457         Lvl    AllPub  …        0    NaN  GdPrv        Shed    2500
1458         Lvl    AllPub  …        0    NaN    NaN         NaN       0
1459         Lvl    AllPub  …        0    NaN    NaN         NaN       0

     MoSold YrSold  SaleType  SaleCondition  SalePrice
0         2   2008        WD         Normal     208500
1         5   2007        WD         Normal     181500
2         9   2008        WD         Normal     223500
3         2   2006        WD        Abnorml     140000
4        12   2008        WD         Normal     250000
…     …    …       …             …        …
1455      8   2007        WD         Normal     175000
1456      2   2010        WD         Normal     210000
1457      5   2010        WD         Normal     266500
1458      4   2010        WD         Normal     142125
1459      6   2008        WD         Normal     147500

[1460 rows x 81 columns]
```

Procederemos a quitar la variable "Id" dado que no nos es de utilidad porque pandas tiene su propia indexación.

[224]:
```
data = data.drop( columns= "Id") #removemos la varaible "id"
```

[225]:
```
data #vemos que se removiera con éxito
```

[225]:
```
      MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
0             60       RL         65.0     8450   Pave   NaN      Reg
1             20       RL         80.0     9600   Pave   NaN      Reg
2             60       RL         68.0    11250   Pave   NaN      IR1
3             70       RL         60.0     9550   Pave   NaN      IR1
4             60       RL         84.0    14260   Pave   NaN      IR1
…           …       …           …       …     …   …       …
1455          60       RL         62.0     7917   Pave   NaN      Reg
1456          20       RL         85.0    13175   Pave   NaN      Reg
1457          70       RL         66.0     9042   Pave   NaN      Reg
1458          20       RL         68.0     9717   Pave   NaN      Reg
1459          20       RL         75.0     9937   Pave   NaN      Reg
```

```
     LandContour Utilities LotConfig  … PoolArea PoolQC  Fence MiscFeature  \
0            Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1            Lvl    AllPub       FR2  …        0    NaN    NaN         NaN
2            Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
3            Lvl    AllPub    Corner  …        0    NaN    NaN         NaN
4            Lvl    AllPub       FR2  …        0    NaN    NaN         NaN
…          …       …       …  …       …    …    …         …
1455         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1456         Lvl    AllPub    Inside  …        0    NaN  MnPrv         NaN
1457         Lvl    AllPub    Inside  …        0    NaN  GdPrv        Shed
1458         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1459         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN

     MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
0          0      2    2008        WD         Normal     208500
1          0      5    2007        WD         Normal     181500
2          0      9    2008        WD         Normal     223500
3          0      2    2006        WD        Abnorml     140000
4          0     12    2008        WD         Normal     250000
…      …    …      …       …            …         …
1455       0      8    2007        WD         Normal     175000
1456       0      2    2010        WD         Normal     210000
1457    2500      5    2010        WD         Normal     266500
1458       0      4    2010        WD         Normal     142125
1459       0      6    2008        WD         Normal     147500

[1460 rows x 80 columns]
```

0.0.1 Correr las lineas de abajo únicamente si se desea obtener el documento de Pandas Profiling

[226]:
```
#profile = ProfileReport(data, title = "Reporte Laboratorio 1", exploratives =␣
↪True, correlations={"cramers": {"calculate": False}}) #generamos el␣
↪Codebook, le ponemos un titulo y que sea interactivo
```

[227]:
```
#profile.to_file("ReporteLab1.html") #imprimimos el HTML
```

0.0.2 Mostramos un poco de estadística descriptiva para las variables que tenemos (solo las numéricas)

[228]:
```
data.describe() #estadística básica
```

[228]:
```
        MSSubClass  LotFrontage        LotArea  OverallQual  OverallCond  \
count  1460.000000  1201.000000    1460.000000  1460.000000  1460.000000
mean     56.897260    70.049958   10516.828082     6.099315     5.575342
std      42.300571    24.284752    9981.264932     1.382997     1.112799
min      20.000000    21.000000    1300.000000     1.000000     1.000000
```

```
25%      20.000000    59.000000    7553.500000     5.000000     5.000000
50%      50.000000    69.000000    9478.500000     6.000000     5.000000
75%      70.000000    80.000000   11601.500000     7.000000     6.000000
max     190.000000   313.000000  215245.000000    10.000000     9.000000

         YearBuilt  YearRemodAdd   MasVnrArea   BsmtFinSF1   BsmtFinSF2  …  \
count  1460.000000   1460.000000  1452.000000  1460.000000  1460.000000  …
mean   1971.267808   1984.865753   103.685262   443.639726    46.549315  …
std      30.202904     20.645407   181.066207   456.098091   161.319273  …
min    1872.000000   1950.000000     0.000000     0.000000     0.000000  …
25%    1954.000000   1967.000000     0.000000     0.000000     0.000000  …
50%    1973.000000   1994.000000     0.000000   383.500000     0.000000  …
75%    2000.000000   2004.000000   166.000000   712.250000     0.000000  …
max    2010.000000   2010.000000  1600.000000  5644.000000  1474.000000  …

        WoodDeckSF  OpenPorchSF  EnclosedPorch    3SsnPorch  ScreenPorch  \
count  1460.000000  1460.000000    1460.000000  1460.000000  1460.000000
mean     94.244521    46.660274      21.954110     3.409589    15.060959
std     125.338794    66.256028      61.119149    29.317331    55.757415
min       0.000000     0.000000       0.000000     0.000000     0.000000
25%       0.000000     0.000000       0.000000     0.000000     0.000000
50%       0.000000    25.000000       0.000000     0.000000     0.000000
75%     168.000000    68.000000       0.000000     0.000000     0.000000
max     857.000000   547.000000     552.000000   508.000000   480.000000

          PoolArea       MiscVal       MoSold       YrSold      SalePrice
count  1460.000000   1460.000000  1460.000000  1460.000000    1460.000000
mean      2.758904     43.489041     6.321918  2007.815753  180921.195890
std      40.177307    496.123024     2.703626     1.328095   79442.502883
min       0.000000      0.000000     1.000000  2006.000000   34900.000000
25%       0.000000      0.000000     5.000000  2007.000000  129975.000000
50%       0.000000      0.000000     6.000000  2008.000000  163000.000000
75%       0.000000      0.000000     8.000000  2009.000000  214000.000000
max     738.000000  15500.000000    12.000000  2010.000000  755000.000000

[8 rows x 37 columns]
```

mostramos información un poco más detallada de las variables trabajadas en la celda anterior

[229]:
```
data.describe
```

[229]:
```
<bound method NDFrame.describe of       MSSubClass MSZoning  LotFrontage
LotArea Street Alley LotShape  \
0             60       RL         65.0     8450   Pave   NaN      Reg
1             20       RL         80.0     9600   Pave   NaN      Reg
2             60       RL         68.0    11250   Pave   NaN      IR1
3             70       RL         60.0     9550   Pave   NaN      IR1
```

```
4             60       RL         84.0    14260   Pave   NaN      IR1
…           …      …           …       …     …   …      …
1455          60       RL         62.0     7917   Pave   NaN      Reg
1456          20       RL         85.0    13175   Pave   NaN      Reg
1457          70       RL         66.0     9042   Pave   NaN      Reg
1458          20       RL         68.0     9717   Pave   NaN      Reg
1459          20       RL         75.0     9937   Pave   NaN      Reg

     LandContour Utilities LotConfig  … PoolArea PoolQC  Fence MiscFeature  \
0            Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1            Lvl    AllPub       FR2  …        0    NaN    NaN         NaN
2            Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
3            Lvl    AllPub    Corner  …        0    NaN    NaN         NaN
4            Lvl    AllPub       FR2  …        0    NaN    NaN         NaN
…          …        …        …  …       …    …    …          …
1455         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1456         Lvl    AllPub    Inside  …        0    NaN  MnPrv         NaN
1457         Lvl    AllPub    Inside  …        0    NaN  GdPrv        Shed
1458         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN
1459         Lvl    AllPub    Inside  …        0    NaN    NaN         NaN

     MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
0          0      2    2008        WD         Normal     208500
1          0      5    2007        WD         Normal     181500
2          0      9    2008        WD         Normal     223500
3          0      2    2006        WD        Abnorml     140000
4          0     12    2008        WD         Normal     250000
…       …    …      …       …             …         …
1455       0      8    2007        WD         Normal     175000
1456       0      2    2010        WD         Normal     210000
1457    2500      5    2010        WD         Normal     266500
1458       0      4    2010        WD         Normal     142125
1459       0      6    2008        WD         Normal     147500

[1460 rows x 80 columns]>
```

Hacemos una matriz de correlación para observar la correlación entre las variables (Solo los variables numéricas)

[230]:
```
sns.heatmap(data.corr()) #mostramos la correlación entre las variables
```

[230]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21d23afb730>
```

0.0.3 Conclusión

Podemos observar que hay variables que están bastantes correlacionadas por lo que deberemos proceder a removerlas a la hora de hacer las pruebas de esfericidad de Barlett y las pruebas χ^2 y la prueba del valor p. Más adelante tendremos que hacer una limpieza más profunda de los datos, para correr las pruebas y el PCA, el análisis de factores y el algoritmo apriori.

0.0.4 Observamos gráficos entre variables para ver su comportamiento, para ellos utilizamos un pairplot (incluye, gráficos de densidad, de dispersión, histogramas, entre otros).

[231]:
```
sns.pairplot(data)
```

[231]: <seaborn.axisgrid.PairGrid at 0x21d2473eac0>

0.0.5 Definimos un par de funciones para limpiar la data

[232]:
```
def generar_id(df): #función generadora de índices
    return list(range(0, len(df)))

def limpiar(df, categoria): #función que limpia los datos (numéricos) con el␣
 ↪promedio de una categoría dada
    prom = df[categoria].mean()
    for i in range(0, len(df)):
        if  str(df[categoria][i]) == "nan" or str(df[categoria][i]) == "NAN":
            df[categoria][i] = float(prom)
```

1 Variables con Datos faltantes

- LotFrontage: 259 datos faltantes (rellenar con el promedio)
- MasVnrArea: 8 datos faltantes (eliminar las filas asosicadas a los dantos faltantes porque no representa más del 1% de la data total)
- GarageYrBlt: 81 faltantes (rellenar con el promedio)

[233]:
```
data_limpia = data.dropna(0, subset = ["MasVnrArea"]) #removemos estos renglones
data_limpia.describe()
```

[233]:
```
        MSSubClass  LotFrontage        LotArea  OverallQual  OverallCond  \
count  1452.000000  1195.000000    1452.000000  1452.000000  1452.000000
mean     56.949036    70.030126   10507.276171     6.092975     5.579201
std      42.340097    24.289276    9989.563592     1.381289     1.113136
min      20.000000    21.000000    1300.000000     1.000000     1.000000
25%      20.000000    59.000000    7538.750000     5.000000     5.000000
50%      50.000000    69.000000    9478.500000     6.000000     5.000000
75%      70.000000    80.000000   11600.000000     7.000000     6.000000
max     190.000000   313.000000  215245.000000    10.000000     9.000000

         YearBuilt  YearRemodAdd   MasVnrArea   BsmtFinSF1   BsmtFinSF2  …  \
count  1452.000000   1452.000000  1452.000000  1452.000000  1452.000000  …
mean   1971.116391   1984.775482   103.685262   441.970386    46.805785  …
std      30.193761     20.652466   181.066207   455.360279   161.726239  …
min    1872.000000   1950.000000     0.000000     0.000000     0.000000  …
25%    1954.000000   1966.000000     0.000000     0.000000     0.000000  …
50%    1972.000000   1993.000000     0.000000   381.000000     0.000000  …
75%    2000.000000   2004.000000   166.000000   706.500000     0.000000  …
max    2010.000000   2010.000000  1600.000000  5644.000000  1474.000000  …

        WoodDeckSF  OpenPorchSF  EnclosedPorch    3SsnPorch  ScreenPorch  \
count  1452.000000  1452.000000    1452.000000  1452.000000  1452.000000
mean     94.416667    46.390496      21.937328     3.428375    15.143939
std     125.393706    66.048619      61.089879    29.396943    55.899665
min       0.000000     0.000000       0.000000     0.000000     0.000000
25%       0.000000     0.000000       0.000000     0.000000     0.000000
50%       0.000000    24.000000       0.000000     0.000000     0.000000
75%     168.000000    68.000000       0.000000     0.000000     0.000000
max     857.000000   547.000000     552.000000   508.000000   480.000000

          PoolArea       MiscVal       MoSold       YrSold      SalePrice
count  1452.000000   1452.000000  1452.000000  1452.000000    1452.000000
mean      2.774105     43.728650     6.318182  2007.815427  180615.063361
std      40.287389    497.478278     2.699644     1.328927   79285.541485
min       0.000000      0.000000     1.000000  2006.000000   34900.000000
25%       0.000000      0.000000     5.000000  2007.000000  129900.000000
50%       0.000000      0.000000     6.000000  2008.000000  162700.000000
75%       0.000000      0.000000     8.000000  2009.000000  214000.000000
```

```
max     738.000000  15500.000000    12.000000  2010.000000  755000.000000

[8 rows x 37 columns]
```

[234]:
```
data_limpia.index = generar_id(data_limpia)
```

[235]:
```
limpiar(data_limpia,"LotFrontage") #limpiamos las variables que tienen los␣
↪datos faltantes
limpiar(data_limpia,"GarageYrBlt")
```

```
<ipython-input-232-d7c9faf13977>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df[categoria][i] = float(prom)
<ipython-input-235-9ca0f393a865>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  limpiar(data_limpia,"LotFrontage") #limpiamos las variables que tienen los
datos faltantes
<ipython-input-235-9ca0f393a865>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  limpiar(data_limpia,"GarageYrBlt")
```

1.0.1 Separamos las variables cuantitativas para comenzar con la visualizaciñon de dichas variables.

[236]:
```
datos_cuantitativos = data.describe().columns
datos_cuantitativos
```

[236]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

1.1 Boxplots

[237]:
```
#mostramos la data proyectada con ISOMAP
fig = plt.figure(figsize=(20,10))
ax = fig.add_subplot(151)
data.boxplot("MSSubClass")
ax.set_title("Boxplot 1")
ax = fig.add_subplot(152)
data.boxplot("LotFrontage")
plt.title('Boxplot 2')
ax = fig.add_subplot(153)
data.boxplot("LotArea")
ax.set_title("Boxplot 3")
ax = fig.add_subplot(154)
data.boxplot("OverallQual")
plt.title('Boxplot 4')
ax = fig.add_subplot(155)
data.boxplot("OverallCond")
plt.axis('tight')
plt.title('Boxplot 5')
plt.show()
```

Podemos que notar que hay variables que tienen una gran cantidad de datos atípicos en algunas variables, por ejemplo "LotFrontage" y "LotArea". Estos datos atípicos de alguna forma afectaran el análisis de factores y el PCA, por lo que es necesario remover dichas variables para poder obtener mejores resultados.

1.2 Histogramas y Gráficos de Frecuencia

[238]:
```
fig1 = plt.figure(figsize=(20,10))
for k in datos_cuantitativos:
    data.hist(k, bins = 8, color = "crimson")
    plt.title("Histograma variable " + str(k))
```

```
C:\Users\jiio2\anaconda3\lib\site-
packages\pandas\plotting\_matplotlib\tools.py:184: RuntimeWarning: More than 20
figures have been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(**fig_kw)
```

```
<Figure size 1440x720 with 0 Axes>
```

Histograma variable LotFrontage
800
700
600
500
400
300
200
100
0
50
100
150
200
250
300

Histograma variable LotArea
1400
1200
1000
800
600
400
200
0
0
50000
100000
150000
200000

Histograma variable OverallQual
400
350
300
250
200
150
100
50
0
2
4
6
8
10

Histograma variable OverallCond
800
700
600
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9

Histograma variable YearBuilt
500
400
300
200
100
0
1880
1900
1920
1940
1960
1980
2000

Histograma variable YearRemodAdd
400
350
300
250
200
150
100
50
0
1950
1960
1970
1980
1990
2000
2010

Histograma variable MasVnrArea
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
1400
1600

Histograma variable BsmtFinSF1
1000
800
600
400
200
0
0
1000
2000
3000
4000
5000

Histograma variable BsmtFinSF2
1400
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
1400

Histograma variable BsmtUnfSF
400
300
200
100
0
0
500
1000
1500
2000

Histograma variable TotalBsmtSF
1000
800
600
400
200
0
0
1000
2000
3000
4000
5000
6000

Histograma variable 1stFlrSF
700
600
500
400
300
200
100
0
1000
2000
3000
4000

Histograma variable 2ndFlrSF
800
700
600
500
400
300
200
100
0
0
500
1000
1500
2000

Histograma variable LowQualFinSF
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600

Histograma variable GrLivArea
700
600
500
400
300
200
100
0
1000
2000
3000
4000
5000

Histograma variable BsmtFullBath
800
700
600
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Histograma variable BsmtHalfBath
1400
1200
1000
800
600
400
200
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

Histograma variable FullBath
800
700
600
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Histograma variable HalfBath
800
600
400
200
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

Histograma variable BedroomAbvGr
800
700
600
500
400
300
200
100
0
0
1
2
3
4
5
6
7
8

Histograma variable KitchenAbvGr
1400
1200
1000
800
600
400
200
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Histograma variable TotRmsAbvGrd
700
600
500
400
300
200
100
0
2
4
6
8
10
12
14

Histograma variable Fireplaces
700
600
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Histograma variable GarageYrBlt
500
400
300
200
100
0
1900
1920
1940
1960
1980
2000

Histograma variable GarageCars
800
700
600
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Histograma variable GarageArea
500
400
300
200
100
0
0
200
400
600
800
1000
1200
1400

Histograma variable WoodDeckSF
800
600
400
200
0
0
200
400
600
800

Histograma variable OpenPorchSF
1000
800
600
400
200
0
0
100
200
300
400
500

Histograma variable EnclosedPorch
1200
1000
800
600
400
200
0
0
100
200
300
400
500

Histograma variable 3SsnPorch
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500

Histograma variable ScreenPorch
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500

Histograma variable PoolArea
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
700

Histograma variable MiscVal
1400
1200
1000
800
600
400
200
0
0
2000
4000
6000
8000
10000
12000
14000
16000

Histograma variable MoSold
350
300
250
200
150
100
50
0
2
4
6
8
10
12

Podemos observar que algunas variables numéricas se comportan como variables categoóricas por lo que se puede proceder aa eliminarlas para las pruebas a realizar

posteriormente. Esto es con la finalidad de evitar el sesgo de los datos, con esto podremos obtener mejores resultaados. Además, podemos observar que la mayoría de histogramas presentan sesgos hacia la izquierda o hacia al derecha, por lo que se recomienda hacer varias corridas de las pruebas para alcanzar una que sea válida.

1.2.1 Pruebas

- Seleccionas los valores numéricos para analizar la correlación entre los datos

[239]:
```
datos_pca = data[datos_cuantitativos]
datos_pca
```

[239]:
```
      MSSubClass  LotFrontage  LotArea  OverallQual  OverallCond  YearBuilt  \
0             60         65.0     8450            7            5       2003
1             20         80.0     9600            6            8       1976
2             60         68.0    11250            7            5       2001
3             70         60.0     9550            7            5       1915
4             60         84.0    14260            8            5       2000
…            …           …        …            …            …          …
1455          60         62.0     7917            6            5       1999
1456          20         85.0    13175            6            6       1978
1457          70         66.0     9042            7            9       1941
1458          20         68.0     9717            5            6       1950
1459          20         75.0     9937            5            6       1965

      YearRemodAdd  MasVnrArea  BsmtFinSF1  BsmtFinSF2  …  WoodDeckSF  \
0             2003       196.0         706           0  …           0
1             1976         0.0         978           0  …         298
2             2002       162.0         486           0  …           0
3             1970         0.0         216           0  …           0
4             2000       350.0         655           0  …         192
…            …           …          …           …  …           …
1455          2000         0.0           0           0  …           0
1456          1988       119.0         790         163  …         349
1457          2006         0.0         275           0  …           0
1458          1996         0.0          49        1029  …         366
1459          1965         0.0         830         290  …         736

      OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea  MiscVal  \
0              61              0          0            0         0        0
1               0              0          0            0         0        0
2              42              0          0            0         0        0
3              35            272          0            0         0        0
4              84              0          0            0         0        0
…             …              …          …            …         …        …
1455           40              0          0            0         0        0
1456            0              0          0            0         0        0
1457           60              0          0            0         0     2500
```

```
1458            0            112           0             0          0         0
1459           68              0           0             0          0         0

      MoSold  YrSold  SalePrice
0          2    2008     208500
1          5    2007     181500
2          9    2008     223500
3          2    2006     140000
4         12    2008     250000
…       …     …         …
1455       8    2007     175000
1456       2    2010     210000
1457       5    2010     266500
1458       4    2010     142125
1459       6    2008     147500

[1460 rows x 37 columns]
```

1.2.2 Revisamos que no existan valores nulos, si los hay procedemos a limpiarlos.

[240]:
```
pca_nulos = datos_pca.isnull().sum().to_frame()
pca_nulos
```

[240]:
```
                 0
MSSubClass       0
LotFrontage    259
LotArea          0
OverallQual      0
OverallCond      0
YearBuilt        0
YearRemodAdd     0
MasVnrArea       8
BsmtFinSF1       0
BsmtFinSF2       0
BsmtUnfSF        0
TotalBsmtSF      0
1stFlrSF         0
2ndFlrSF         0
LowQualFinSF     0
GrLivArea        0
BsmtFullBath     0
BsmtHalfBath     0
FullBath         0
HalfBath         0
BedroomAbvGr     0
KitchenAbvGr     0
TotRmsAbvGrd     0
```

```
Fireplaces       0
GarageYrBlt     81
GarageCars       0
GarageArea       0
WoodDeckSF       0
OpenPorchSF      0
EnclosedPorch    0
3SsnPorch        0
ScreenPorch      0
PoolArea         0
MiscVal          0
MoSold           0
YrSold           0
SalePrice        0
```

[241]:
```
pca_na = datos_pca.isna().sum().to_frame()
pca_na
```

[241]:
```
                 0
MSSubClass       0
LotFrontage    259
LotArea          0
OverallQual      0
OverallCond      0
YearBuilt        0
YearRemodAdd     0
MasVnrArea       8
BsmtFinSF1       0
BsmtFinSF2       0
BsmtUnfSF        0
TotalBsmtSF      0
1stFlrSF         0
2ndFlrSF         0
LowQualFinSF     0
GrLivArea        0
BsmtFullBath     0
BsmtHalfBath     0
FullBath         0
HalfBath         0
BedroomAbvGr     0
KitchenAbvGr     0
TotRmsAbvGrd     0
Fireplaces       0
GarageYrBlt     81
GarageCars       0
GarageArea       0
WoodDeckSF       0
```

```
OpenPorchSF      0
EnclosedPorch    0
3SsnPorch        0
ScreenPorch      0
PoolArea         0
MiscVal          0
MoSold           0
YrSold           0
SalePrice        0
```

1.2.3 Desplegamos nuevamente una matriz de correlación para ver las variables que tienen correlación 0. Dichas variables deben ser eliminadas.

[242]:
```
sns.heatmap(datos_pca.corr())
```

[242]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21db4f66250>
```

1.2.4 Variables eliminadas

- "PoolArea"
- "3SsnPorch"

- “MoSold”
- “ScreenPorch”
- “OpenPorchSF”
- “EnclosedPorch”
- “MiscVal”
- “BsmtFinSF2”
- “WoodDeckSF”
- “OverallCond”
- “HalfBath”
- “YrSold”
- “GarageYrBlt”
- “YearRemodAdd”
- “YearBuilt”
- “BsmtHalfBath”
- “BsmtFullBath”
- “LowQualFinSF”
- “MasVnrArea”
- “LotFrontage”

Estas variables fueron elimidas por tener un coeficiente de corelación de0, l ocual tendrá implicaciones significativas sobre el dataset a la hora de hacer las pruebas de esfericidad de Barllet, la χ^2 y la prueba de KMO

[243]:
```
data_final = datos_pca.drop(columns = ["PoolArea", "3SsnPorch", "MoSold",␣
↪"ScreenPorch", "OpenPorchSF", "EnclosedPorch", "MiscVal", "BsmtFinSF2",
                                        "WoodDeckSF", "OverallCond", "HalfBath",␣
↪"YrSold", "GarageYrBlt", "YearRemodAdd", "YearBuilt", "BsmtHalfBath",␣
↪"BsmtFullBath",
                                        "LowQualFinSF", "MasVnrArea",␣
↪"LotFrontage"])
```

[244]:
```
data_final #mostramos la data con la que trabajeremos
```

[244]:
```
      MSSubClass  LotArea  OverallQual  BsmtFinSF1  BsmtUnfSF  TotalBsmtSF  \
0             60     8450            7         706        150          856
1             20     9600            6         978        284         1262
2             60    11250            7         486        434          920
3             70     9550            7         216        540          756
4             60    14260            8         655        490         1145
…          …       …           …           …          …           …
1455          60     7917            6           0        953          953
1456          20    13175            6         790        589         1542
1457          70     9042            7         275        877         1152
1458          20     9717            5          49          0         1078
1459          20     9937            5         830        136         1256

      1stFlrSF  2ndFlrSF  GrLivArea  FullBath  BedroomAbvGr  KitchenAbvGr  \
0          856       854       1710         2             3             1
```

```
1          1262          0        1262          2              3              1
2           920        866        1786          2              3              1
3           961        756        1717          1              3              1
4          1145       1053        2198          2              4              1
…           …          …           …            …              …              …
1455        953        694        1647          2              3              1
1456       2073          0        2073          2              3              1
1457       1188       1152        2340          2              4              1
1458       1078          0        1078          1              2              1
1459       1256          0        1256          1              3              1

      TotRmsAbvGrd  Fireplaces  GarageCars  GarageArea  SalePrice
0                8           0           2         548     208500
1                6           1           2         460     181500
2                6           1           2         608     223500
3                7           1           3         642     140000
4                9           1           3         836     250000
…              …           …           …           …         …
1455             7           1           2         460     175000
1456             7           2           2         500     210000
1457             9           2           1         252     266500
1458             5           0           1         240     142125
1459             6           0           1         276     147500

[1460 rows x 17 columns]
```

[245]:
```
data_final.dtypes
```

[245]:
```
MSSubClass      int64
LotArea         int64
OverallQual     int64
BsmtFinSF1      int64
BsmtUnfSF       int64
TotalBsmtSF     int64
1stFlrSF        int64
2ndFlrSF        int64
GrLivArea       int64
FullBath        int64
BedroomAbvGr    int64
KitchenAbvGr    int64
TotRmsAbvGrd    int64
Fireplaces      int64
GarageCars      int64
GarageArea      int64
SalePrice       int64
dtype: object
```

1.2.5 Luego de la limpieza obtenemos esta matriz de correlación

[246]:
```
sns.heatmap(data_final.corr())
```

[246]: <matplotlib.axes._subplots.AxesSubplot at 0x21da3c11670>

podemos observar que son pocas las variables que tienen correlación cero, por lo que se espera que los algoritmos converjan.

[247]:
```
np.linalg.det(np.array(data_final.corr())) #calculamos el determinante, vemos␣
↪que es pequeño
```

[247]: 7.31570603020266e-08

[248]:
```
from factor_analyzer.factor_analyzer import calculate_kmo
from factor_analyzer.factor_analyzer import calculate_bartlett_sphericity
chi_square_value,p_value=calculate_bartlett_sphericity(data_final)
chi_square_value, p_value
```

[248]: (23882.448043075692, 0.0)

podemos observar que si es válido hacer un análisis de factores dada la pruebas de chi cuadrado y la prueba del valor p.

[249]:
```
kmo_all,kmo_model=calculate_kmo(data_final)
kmo_model
```

[249]:
```
0.7247625945369006
```

podemos observar que si es válido hacer un PCA dada la prueba KMO

[250]:
```
# hacemos el análisis de factores con 8 factores
fa = FactorAnalyzer(n_factors = 8, rotation="varimax")
fa.fit(data_final)
# Check Eigenvalues
ev, v = fa.get_eigenvalues()
ev
```

[250]:
```
array([6.17244280e+00, 2.61317506e+00, 1.55515992e+00, 1.37110772e+00,
       1.17036985e+00, 8.67149802e-01, 8.05924386e-01, 6.18076189e-01,
       4.52620710e-01, 3.99584853e-01, 2.89717240e-01, 2.36234714e-01,
       1.55614865e-01, 1.46178369e-01, 1.02659349e-01, 4.03794185e-02,
       3.60475887e-03])
```

[251]:
```
plt.scatter(range(1,data_final.shape[1]+1),ev)
plt.plot(range(1,data_final.shape[1]+1),ev)
plt.title('Resultados Análisis de Factores')
plt.xlabel('Factores')
plt.ylabel('Eigenvalores')
plt.grid()
plt.show()
```

[252]:
```
fa.loadings_
```

[252]:
```
array([[ 0.04690629, -0.02803027, -0.1098893 , -0.11886128, -0.06795309,
         0.07522058,  0.67850896,  0.2217428 ],
       [ 0.11189792,  0.08631312,  0.11615699,  0.38888279, -0.06650707,
        -0.03800526, -0.11505301, -0.00523085],
       [ 0.21308496,  0.44000823,  0.26674056,  0.17530607,  0.10728588,
         0.69168373,  0.13488189, -0.2443296 ],
       [-0.06957524,  0.16339159,  0.58282045,  0.24050379, -0.72172251,
         0.08209586, -0.02483521, -0.03917994],
       [ 0.12455071,  0.1293406 ,  0.24230224, -0.00247661,  0.91411404,
         0.11536564, -0.09678993,  0.01670225],
       [ 0.02660948,  0.28258337,  0.83022427,  0.32091732,  0.14652974,
         0.18756215, -0.16333708, -0.01290624],
       [ 0.06276868,  0.27599687,  0.55782955,  0.54018323,  0.09128692,
         0.27860899, -0.27291353,  0.26583526],
       [ 0.80953878,  0.10731313, -0.2057859 ,  0.0729494 , -0.00985549,
        -0.0069173 ,  0.47294707, -0.25312624],
       [ 0.71388425,  0.2795611 ,  0.23153652,  0.46935788,  0.06162301,
         0.22268546,  0.17077715,  0.01675366],
       [ 0.45042836,  0.33671571,  0.14271023,  0.13802942,  0.13889626,
         0.34744955,  0.15214007,  0.11342183],
       [ 0.75797286, -0.02001146, -0.01499673,  0.03528778,  0.06983629,
```

```
 -0.0076454 , -0.13822516,  0.15995507],
[ 0.17580523, -0.0402704 ,  0.01890846, -0.07128986,  0.02675904,
 -0.09359624,  0.19227308,  0.651721  ],
[ 0.8183442 ,  0.19434069,  0.09244685,  0.25293083,  0.07611802,
  0.20090185, -0.01557582,  0.21765264],
[ 0.14836131,  0.14614355,  0.10483172,  0.554704  , -0.03679276,
  0.19340833,  0.03254005, -0.1270769 ],
[ 0.12282609,  0.89505682,  0.09907366,  0.16372128,  0.04021368,
  0.17694237, -0.00164877, -0.01358964],
[ 0.10719464,  0.90105926,  0.18232852,  0.17002307, -0.00387499,
  0.08656213, -0.05862452, -0.02609841],
[ 0.30399503,  0.47554218,  0.31345029,  0.39375175, -0.00902962,
  0.46430925,  0.02165793, -0.17696193]])
```

[253]:
```
fa.get_factor_variance()
```

[253]:
```
(array([2.86911613, 2.5046676 , 1.77330693, 1.48214561, 1.4444844 ,
        1.12970999, 0.94022619, 0.80478969]),
 array([0.16877154, 0.14733339, 0.10431217, 0.08718504, 0.08496967,
        0.06645353, 0.05530742, 0.04734057]),
 array([0.16877154, 0.31610493, 0.4204171 , 0.50760213, 0.5925718 ,
        0.65902533, 0.71433276, 0.76167333]))
```

[254]:
```
factores = list(range(1, len(fa.get_factor_variance()[2]) + 1))
factores
```

[254]:
```
[1, 2, 3, 4, 5, 6, 7, 8]
```

[255]:
```
plt.scatter(factores, fa.get_factor_variance()[2])
plt.plot(factores, fa.get_factor_variance()[2])
plt.xlabel('Número de Factores')
plt.ylabel('Varianza acumulada')
plt.grid()
```

[256]:
```
print("Varianza acumulada: ", fa.get_factor_variance()[2])
```

```
Varianza acumulada:  [0.16877154 0.31610493 0.4204171  0.50760213 0.5925718
0.65902533
 0.71433276 0.76167333]
```

Podemos observar que con 8 factores es posible representar el 76% de los datos por lo que es viable hacer una análisis de factores.

[257]:
```
data_final_norm = (data_final - data_final.mean())/data_final.std()␣
↪#normalizamos la data para el PCA
```

[258]:
```
pca = PCA(n_components = 9) # usamos 9 componentes
model = pca.fit_transform(data_final_norm)
```

[259]:
```
pca.components_
```

[259]:
```
array([[-2.65714332e-02,  1.32931201e-01,  3.16012158e-01,
         1.40040753e-01,  1.37112099e-01,  2.81090450e-01,
         2.93692468e-01,  1.60560330e-01,  3.49774101e-01,
         2.80431122e-01,  1.54274691e-01,  8.22980652e-04,
         2.92304986e-01,  2.17725338e-01,  2.92843665e-01,
         2.89580656e-01,  3.53632456e-01],
       [ 2.78515562e-01, -9.23526432e-02, -5.45563420e-02,
        -3.24260546e-01,  5.09812276e-02, -3.08606110e-01,
        -2.54082784e-01,  4.70673442e-01,  2.11673959e-01,
```

```
  1.86251337e-01,  3.77324674e-01,  2.39095292e-01,
  3.15713768e-01, -6.79783494e-02, -9.99649707e-02,
 -1.47658627e-01, -8.19890795e-02],
[ 2.79064695e-01,  1.42336052e-01, -1.83121677e-02,
  5.15512695e-01, -7.03207054e-01, -1.52275765e-01,
 -1.26668154e-01,  2.14831920e-01,  8.41314934e-02,
 -5.82920670e-02, -6.36543003e-02, -2.69191317e-02,
 -6.33712947e-03,  1.86888019e-01, -1.46853045e-02,
  4.42688153e-04,  7.38464492e-02],
[-2.72654464e-01,  3.83793344e-01, -3.10611855e-01,
  1.72769331e-01, -3.92627796e-02,  1.79381710e-01,
  2.94843263e-01, -1.49068821e-01,  9.98213005e-02,
 -1.04599026e-01,  3.38185040e-01,  3.14212511e-01,
  2.10290168e-01,  9.04144123e-02, -3.49176001e-01,
 -2.91574657e-01, -1.25775855e-01],
[ 4.42774066e-01, -2.11406141e-01, -4.51488360e-02,
  1.51441978e-01, -8.60817377e-04,  1.39571531e-01,
  1.87543426e-01, -2.34787515e-01, -6.06834072e-02,
  1.33812407e-01, -1.53584858e-01,  6.88959683e-01,
 -2.16204456e-02, -2.62379294e-01,  1.21276483e-01,
  1.25055848e-01, -7.07639330e-02],
[-4.15378623e-01,  1.45738631e-01, -2.26304515e-01,
  1.95950047e-02, -2.13182272e-01, -1.86483223e-01,
 -1.17539419e-01,  1.27084831e-02, -8.05278693e-02,
 -4.79806457e-02,  2.45518386e-01,  9.68839526e-02,
  7.78438162e-02, -4.32744872e-01,  4.11399344e-01,
  4.62260948e-01, -7.55540416e-02],
[ 2.55846863e-01,  7.72181473e-01, -6.74612958e-02,
 -2.91630380e-01,  1.48310572e-01, -1.35002370e-01,
 -8.53925531e-02, -7.95911189e-03, -7.34202376e-02,
 -3.20089524e-03, -2.50393816e-01,  1.42206309e-01,
 -1.35965414e-01,  2.21837972e-01,  1.65578446e-01,
  1.23447362e-01, -4.92678682e-02],
[ 1.51421946e-01,  3.60871802e-01,  1.71200405e-01,
  1.40658801e-01,  3.60329434e-02,  1.71302212e-01,
 -4.45525473e-02,  5.77921153e-02,  1.95075567e-02,
  2.88862380e-01,  2.60821858e-03, -2.54005695e-01,
 -8.61949037e-02, -7.14819987e-01, -2.14832393e-01,
 -1.85667092e-01,  1.20044432e-01],
[ 3.61817684e-01,  1.95156024e-02, -4.70823817e-02,
  5.48125975e-02,  1.95892131e-01,  2.52795637e-01,
  1.57803842e-02,  1.38508736e-01,  1.38547654e-01,
 -7.77537898e-01,  1.39201697e-01, -1.58956994e-01,
  1.15996235e-01, -1.66641854e-01,  4.55700499e-02,
  1.63326305e-01, -5.82328308e-02]])
```

[260]:
```
df_pca = pd.DataFrame(pca.components_)
df_pca
```

[260]:
```
          0         1         2         3         4         5         6  \
0 -0.026571  0.132931  0.316012  0.140041  0.137112  0.281090  0.293692
1  0.278516 -0.092353 -0.054556 -0.324261  0.050981 -0.308606 -0.254083
2  0.279065  0.142336 -0.018312  0.515513 -0.703207 -0.152276 -0.126668
3 -0.272654  0.383793 -0.310612  0.172769 -0.039263  0.179382  0.294843
4  0.442774 -0.211406 -0.045149  0.151442 -0.000861  0.139572  0.187543
5 -0.415379  0.145739 -0.226305  0.019595 -0.213182 -0.186483 -0.117539
6  0.255847  0.772181 -0.067461 -0.291630  0.148311 -0.135002 -0.085393
7  0.151422  0.360872  0.171200  0.140659  0.036033  0.171302 -0.044553
8  0.361818  0.019516 -0.047082  0.054813  0.195892  0.252796  0.015780

          7         8         9        10        11        12        13  \
0  0.160560  0.349774  0.280431  0.154275  0.000823  0.292305  0.217725
1  0.470673  0.211674  0.186251  0.377325  0.239095  0.315714 -0.067978
2  0.214832  0.084131 -0.058292 -0.063654 -0.026919 -0.006337  0.186888
3 -0.149069  0.099821 -0.104599  0.338185  0.314213  0.210290  0.090414
4 -0.234788 -0.060683  0.133812 -0.153585  0.688960 -0.021620 -0.262379
5  0.012708 -0.080528 -0.047981  0.245518  0.096884  0.077844 -0.432745
6 -0.007959 -0.073420 -0.003201 -0.250394  0.142206 -0.135965  0.221838
7  0.057792  0.019508  0.288862  0.002608 -0.254006 -0.086195 -0.714820
8  0.138509  0.138548 -0.777538  0.139202 -0.158957  0.115996 -0.166642

         14        15        16
0  0.292844  0.289581  0.353632
1 -0.099965 -0.147659 -0.081989
2 -0.014685  0.000443  0.073846
3 -0.349176 -0.291575 -0.125776
4  0.121276  0.125056 -0.070764
5  0.411399  0.462261 -0.075554
6  0.165578  0.123447 -0.049268
7 -0.214832 -0.185667  0.120044
8  0.045570  0.163326 -0.058233
```

1.2.6 Los coeficientes principales están representando las variables más importantes relacionadas con el resto de varaibles que fueron aboservidas con el fin de reducir la cantidad utilizada.

[261]:
```
componentes = list(range(1, len(pca.explained_variance_ratio_) + 1))
```

[262]:
```
pca.explained_variance_ratio_
```

[262]:
```
array([0.36308487, 0.15371618, 0.09148   , 0.0806534 , 0.06884529,
       0.05100881, 0.04740732, 0.03635742, 0.02662475])
```

[263]:
```
pca.explained_variance_
```

[263]:
```
array([6.1724428 , 2.61317506, 1.55515992, 1.37110772, 1.17036985,
       0.8671498 , 0.80592439, 0.61807619, 0.45262071])
```

[264]:
```
plt.scatter(componentes,pca.explained_variance_)
plt.plot(componentes, pca.explained_variance_)
plt.xlabel('Número de componentes' )
plt.ylabel('Varianza explicada')
plt.grid()
```

[265]:
```
plt.scatter(componentes, np.cumsum(pca.explained_variance_ratio_))
plt.plot(componentes, np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Número de componentes')
plt.ylabel('Varianza acumulada')
plt.grid()
plt.show()
```

1.2.7 Podemos observar que los datos del PCA se comportan de manera similar a los de análisis de factores, pero en el PCA podemos explicar con 9 componentes más del 90% de los datos, por lo que es viable

1.2.8 Reglas de Asignación

[266]:
```
data_cualitativa = data[data.columns[~data.columns.isin(datos_cuantitativos )]]
```

[267]:
```
data_cualitativa
```

[267]:
```
     MSZoning Street Alley LotShape LandContour Utilities LotConfig LandSlope  \
0          RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl
1          RL   Pave   NaN      Reg         Lvl    AllPub       FR2       Gtl
2          RL   Pave   NaN      IR1         Lvl    AllPub    Inside       Gtl
3          RL   Pave   NaN      IR1         Lvl    AllPub    Corner       Gtl
4          RL   Pave   NaN      IR1         Lvl    AllPub       FR2       Gtl
…        …    …    …       …          …        …        …        …
1455       RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl
1456       RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl
1457       RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl
1458       RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl
1459       RL   Pave   NaN      Reg         Lvl    AllPub    Inside       Gtl

     Neighborhood Condition1  … GarageType GarageFinish GarageQual  \
0         CollgCr       Norm  …     Attchd          RFn         TA
```

```
1          Veenker      Feedr  …     Attchd          RFn         TA
2          CollgCr       Norm  …     Attchd          RFn         TA
3          Crawfor       Norm  …     Detchd          Unf         TA
4          NoRidge       Norm  …     Attchd          RFn         TA
…            …        …  …        …          …         …
1455       Gilbert       Norm  …     Attchd          RFn         TA
1456        NWAmes       Norm  …     Attchd          Unf         TA
1457       Crawfor       Norm  …     Attchd          RFn         TA
1458         NAmes       Norm  …     Attchd          Unf         TA
1459       Edwards       Norm  …     Attchd          Fin         TA

     GarageCond PavedDrive PoolQC  Fence MiscFeature SaleType SaleCondition
0            TA          Y    NaN    NaN         NaN       WD        Normal
1            TA          Y    NaN    NaN         NaN       WD        Normal
2            TA          Y    NaN    NaN         NaN       WD        Normal
3            TA          Y    NaN    NaN         NaN       WD       Abnorml
4            TA          Y    NaN    NaN         NaN       WD        Normal
…          …         …    …    …          …       …            …
1455         TA          Y    NaN    NaN         NaN       WD        Normal
1456         TA          Y    NaN  MnPrv         NaN       WD        Normal
1457         TA          Y    NaN  GdPrv        Shed       WD        Normal
1458         TA          Y    NaN    NaN         NaN       WD        Normal
1459         TA          Y    NaN    NaN         NaN       WD        Normal

[1460 rows x 43 columns]
```

[268]:
```
data_cualitativa.groupby("GarageQual").count()
```

[268]:
```
            MSZoning  Street  Alley  LotShape  LandContour  Utilities  \
GarageQual
Ex                 3       3      0         3            3          3
Fa                48      48     10        48           48         48
Gd                14      14      2        14           14         14
Po                 3       3      1         3            3          3
TA              1311    1311     69      1311         1311       1311

            LotConfig  LandSlope  Neighborhood  Condition1  …  FireplaceQu  \
GarageQual                                                  …
Ex                  3          3             3           3  …            2
Fa                 48         48            48          48  …           16
Gd                 14         14            14          14  …           13
Po                  3          3             3           3  …            0
TA               1311       1311          1311        1311  …          730

            GarageType  GarageFinish  GarageCond  PavedDrive  PoolQC  Fence  \
GarageQual
Ex                   3             3           3           3       0      1
```

```
Fa                   48            48           48           48        0      10
Gd                   14            14           14           14        1       3
Po                    3             3            3            3        0       1
TA                 1311          1311         1311         1311        6     250

            MiscFeature  SaleType  SaleCondition
GarageQual
Ex                    1         3              3
Fa                    4        48             48
Gd                    0        14             14
Po                    0         3              3
TA                   46      1311           1311

[5 rows x 42 columns]
```

[269]:
```
def label_function(val):
    return f'{val / 100 * len(data_cualitativa):.0f}\n{val:.0f}%'
data_cualitativa.groupby('GarageQual').size().plot(kind='pie',␣
↪autopct=label_function, textprops={'fontsize': 20},
                                colors=['tomato', 'gold', 'skyblue',␣
↪"crimson"])
```

[269]: <matplotlib.axes._subplots.AxesSubplot at 0x21db5290af0>

[270]:
```
data_cualitativa.groupby('Utilities').size().plot(kind='pie',␣
↪autopct=label_function, textprops={'fontsize': 10},
```

```
                                    colors=['tomato', 'gold', 'skyblue',␣
↪"crimson"])
```

[270]: <matplotlib.axes._subplots.AxesSubplot at 0x21db52d6af0>

```
[271]: data_cualitativa.groupby('Foundation').size().plot(kind='pie',␣
↪autopct=label_function, textprops={'fontsize': 10},
                                    colors=['tomato', 'gold', 'skyblue',␣
↪"crimson"])
```

[271]: <matplotlib.axes._subplots.AxesSubplot at 0x21db5318b50>

[272]:
```
data_cualitativa.groupby('Heating').size().plot(kind='pie',␣
↪autopct=label_function, textprops={'fontsize': 10},
                                    colors=['tomato', 'gold', 'skyblue',␣
↪"crimson"])
```

[272]: <matplotlib.axes._subplots.AxesSubplot at 0x21db53756d0>

[273]:
```
data_cualitativa.groupby('SaleType').size().plot(kind='pie',␣
↪autopct=label_function, textprops={'fontsize': 10},
                                colors=['tomato', 'gold', 'skyblue',␣
↪"crimson"])
```

[273]: <matplotlib.axes._subplots.AxesSubplot at 0x21db53ce040>

Podemos ver que en los gráficos de pie tenemos datos que ocasionaran overfitting por lo que procederemos a removerlos en su mayoría, por lo tanto se quitarán las siguientes variables por dicha razón:

1.2.9 Variables a Remover

- "GarageCond"
- "Fence"
- "PoolQC"
- "GarageQual"
- "KitchenQual"
- "Condition1"
- "LandSlope"
- "MiscFeature"
- "FireplaceQu"
- "HeatingQC"
- "LandContour"
- "Alley"
- "Condition2"
- "BsmtCond"

- “BsmtQual”
- “BsmtFinType2”
- “BsmtFinType1”
- “ExterQual”
- “ExterCond”
- “GarageFinish”
- “Exterior1st”
- “Exterior2nd”
- “BldgType”
- “LotShape”
- “HouseStyle”
- “RoofMatl”
- “BsmtExposure”
- “MasVnrType”
- “GarageType”
- “CentralAir”
- “MSZoning”
- “PavedDrive”
- “RoofStyle”
- “LotConfig”
- “Street”
- “Utilities”
- “Heating”

[274]:
```
data_priori = data_cualitativa.drop(columns = ["GarageCond", "Fence", "PoolQC",␣
↪"GarageQual", "KitchenQual", "Condition1", "LandSlope", "MiscFeature",
                                               "FireplaceQu", "HeatingQC",␣
↪"LandContour", "Alley", "Condition2", "BsmtCond", "BsmtQual", "BsmtFinType2",
                                               "BsmtFinType1", "ExterQual",␣
↪"ExterCond", "GarageFinish", "Exterior1st", "Exterior2nd", "BldgType",
                                               "LotShape", "HouseStyle",␣
↪"RoofMatl", "BsmtExposure", "MasVnrType", "GarageType", "CentralAir",␣
↪"MSZoning",
                                               "PavedDrive", "RoofStyle",␣
↪"LotConfig", "Street", "Utilities", "Heating"])
data_priori
```

[274]:

	Neighborhood	Foundation	Electrical	Functional	SaleType	SaleCondition
0	CollgCr	PConc	SBrkr	Typ	WD	Normal
1	Veenker	CBlock	SBrkr	Typ	WD	Normal
2	CollgCr	PConc	SBrkr	Typ	WD	Normal
3	Crawfor	BrkTil	SBrkr	Typ	WD	Abnorml
4	NoRidge	PConc	SBrkr	Typ	WD	Normal
…	…	…	…	…	…	…
1455	Gilbert	PConc	SBrkr	Typ	WD	Normal
1456	NWAmes	CBlock	SBrkr	Min1	WD	Normal
1457	Crawfor	Stone	SBrkr	Typ	WD	Normal

```
1458        NAmes     CBlock      FuseA        Typ       WD        Normal
1459      Edwards     CBlock      SBrkr        Typ       WD        Normal

[1460 rows x 6 columns]
```

[275]:
```python
data_priori.columns
```

[275]:
```
Index(['Neighborhood', 'Foundation', 'Electrical', 'Functional', 'SaleType',
       'SaleCondition'],
      dtype='object')
```

[276]:
```python
records = []
for i in range(0, len(data_priori)):
    records.append([str(data_priori.values[i,j]) for j in range(0, 6)])
```

[277]:
```python
association_rules = apriori(records)
association_results = list(association_rules)
```

[278]:
```python
for i in range(0, 5):
    print(association_results[5*i],  "\n")
```

```
RelationRecord(items=frozenset({'BrkTil'}), support=0.1,
ordered_statistics=[OrderedStatistic(items_base=frozenset(),
items_add=frozenset({'BrkTil'}), confidence=0.1, lift=1.0)])

RelationRecord(items=frozenset({'PConc'}), support=0.44315068493150683,
ordered_statistics=[OrderedStatistic(items_base=frozenset(),
items_add=frozenset({'PConc'}), confidence=0.44315068493150683, lift=1.0)])

RelationRecord(items=frozenset({'Normal', 'CBlock'}),
support=0.38082191780821917,
ordered_statistics=[OrderedStatistic(items_base=frozenset(),
items_add=frozenset({'CBlock', 'Normal'}), confidence=0.38082191780821917,
lift=1.0), OrderedStatistic(items_base=frozenset({'CBlock'}),
items_add=frozenset({'Normal'}), confidence=0.8769716088328076,
lift=1.0687633964072614), OrderedStatistic(items_base=frozenset({'Normal'}),
items_add=frozenset({'CBlock'}), confidence=0.46410684474123537,
lift=1.0687633964072614)])

RelationRecord(items=frozenset({'CollgCr', 'Typ'}), support=0.10273972602739725,
ordered_statistics=[OrderedStatistic(items_base=frozenset(),
items_add=frozenset({'CollgCr', 'Typ'}), confidence=0.10273972602739725,
lift=1.0), OrderedStatistic(items_base=frozenset({'CollgCr'}),
items_add=frozenset({'Typ'}), confidence=1.0, lift=1.0735294117647058),
OrderedStatistic(items_base=frozenset({'Typ'}),
items_add=frozenset({'CollgCr'}), confidence=0.11029411764705882,
lift=1.0735294117647058)])
```

```
RelationRecord(items=frozenset({'PConc', 'Normal'}),
support=0.33356164383561643,
ordered_statistics=[OrderedStatistic(items_base=frozenset(),
items_add=frozenset({'PConc', 'Normal'}), confidence=0.33356164383561643,
lift=1.0), OrderedStatistic(items_base=frozenset({'Normal'}),
items_add=frozenset({'PConc'}), confidence=0.4065108514190317,
lift=0.917319695628727), OrderedStatistic(items_base=frozenset({'PConc'}),
items_add=frozenset({'Normal'}), confidence=0.7527047913446677,
lift=0.917319695628727)])
```

1.2.10 Podemos ver que algunas palabras si tienen un suporte alto, pero en general el soporte no es mayor a 0.5. Por lo que no es recomedable hacer este tipo de asosiaciones por que no se obtendran buenos resultados.

2 Conclusiones

- Es posible efectuar una análisis de factores con dado que las pruebas de hipótesis de chi cuadrado y el valor p lo permiten. Se pueden obtener bueno resultados.
- Es posible hacer un PCA, dado que la prueba de KMO es válida, además, se obtienen buenos resultados al ahcerlo.
- No es recomendable hacer reglas de asosiación porque no se obtienen buenos resultados con la data.