Conceptos de estadística avanzada

Presenta: Juliho David Castillo Colmenares

¡Bienvenido a la actividad práctica del módulo!

Antes de empezar

En este archivo pondrás en práctica algunos de los conceptos aprendidos de estadística avanzada, como pruebas de normalidad, algunas medidas estadísticas importantes como varianza y desviación estándar, así como el concepto de linealidad aplicado a un ejemplo sobre como predecir el precio de venta de casas de acuerdo a sus diferentes características. La descripción de cada característica la podrás encontrar en el archivo de data_description.txt

Paso a paso:

- Descarga el archivo House Pricing.csv. En la sección de Anexos.
- Verifica por medio de pruebas analíticas y visuales como es la distribución de las diferentes variables númericas que se encuentran en el dataset como “Salesprice”, “GrLivArea”, “'2ndFlrSF”. Calculando algunas métricas importantes como la media, la desviación estándar y sus cuártiles.

- Calcula las correlaciones entre estás variables y visualiza el resultado en un heatmap, ¿Qué variables correlacionan más con el Precio de Venta?
- Configura un modelo de regresión con todas las variables numéricas por medio de la función OLS de la librería *sm*
- Imprime el reporte de regresión y revisa que variables son estadísticamente significativas para nuestro modelo.
- Por último por medio de la libería *variance_inflation_factor* revisa que variables presentan multicolinealidad
- Con todo el análisis anterior construye el modelo de regresión final excluyendo aquellas variables que consideres necesarias y mide el nivel de precisión del modelo por medio del error cuadratico medio y la R cuadrada. ¿Qué tan preciso fue tu modelo?

Archivo final

¿Cómo presentar su entrega?

Un Jupyter Notebook (archivo de extensión .ipynb), archivo PDF y capturas de pantalla en el espacio de respuesta que muestren tanto el código desarrollado como la solución al problema planteado, incluyendo los comentarios que sean pertinentes a las preguntas que se plantean. Entregas sin estos elementos no serán calificadas.

Tiempo estimado de resolución: 90 minutos

In [ ]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.feature_selection import RFE
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import mean_squared_error, r2_score
```

In [ ]:
```
TARGET = 'SalePrice'
```

In [ ]:
```
df = pd.read_csv('House Pricing.csv')
df.head()
```

Out[ ]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	F
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	

5 rows × 81 columns

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          91 non-null     object
 7   LotShape       1460 non-null   object
 8   LandContour    1460 non-null   object
 9   Utilities      1460 non-null   object
 10  LotConfig      1460 non-null   object
 11  LandSlope      1460 non-null   object
 12  Neighborhood   1460 non-null   object
 13  Condition1     1460 non-null   object
 14  Condition2     1460 non-null   object
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
 21  RoofStyle      1460 non-null   object
 22  RoofMatl       1460 non-null   object
 23  Exterior1st    1460 non-null   object
 24  Exterior2nd    1460 non-null   object
 25  MasVnrType     588 non-null    object
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1423 non-null   object
 31  BsmtCond       1423 non-null   object
 32  BsmtExposure   1422 non-null   object
 33  BsmtFinType1   1423 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1422 non-null   object
 36  BsmtFinSF2     1460 non-null   int64
```

```
37  BsmtUnfSF      1460 non-null   int64
38  TotalBsmtSF    1460 non-null   int64
39  Heating        1460 non-null   object
40  HeatingQC      1460 non-null   object
41  CentralAir     1460 non-null   object
42  Electrical     1459 non-null   object
43  1stFlrSF       1460 non-null   int64
44  2ndFlrSF       1460 non-null   int64
45  LowQualFinSF   1460 non-null   int64
46  GrLivArea      1460 non-null   int64
47  BsmtFullBath   1460 non-null   int64
48  BsmtHalfBath   1460 non-null   int64
49  FullBath       1460 non-null   int64
50  HalfBath       1460 non-null   int64
51  BedroomAbvGr   1460 non-null   int64
52  KitchenAbvGr   1460 non-null   int64
53  KitchenQual    1460 non-null   object
54  TotRmsAbvGrd   1460 non-null   int64
55  Functional     1460 non-null   object
56  Fireplaces     1460 non-null   int64
57  FireplaceQu    770 non-null    object
58  GarageType     1379 non-null   object
59  GarageYrBlt    1379 non-null   float64
60  GarageFinish   1379 non-null   object
61  GarageCars     1460 non-null   int64
62  GarageArea     1460 non-null   int64
63  GarageQual     1379 non-null   object
64  GarageCond     1379 non-null   object
65  PavedDrive     1460 non-null   object
66  WoodDeckSF     1460 non-null   int64
67  OpenPorchSF    1460 non-null   int64
68  EnclosedPorch  1460 non-null   int64
69  3SsnPorch      1460 non-null   int64
70  ScreenPorch    1460 non-null   int64
71  PoolArea       1460 non-null   int64
72  PoolQC         7 non-null      object
73  Fence          281 non-null    object
74  MiscFeature    54 non-null     object
75  MiscVal        1460 non-null   int64
76  MoSold         1460 non-null   int64
77  YrSold         1460 non-null   int64
78  SaleType       1460 non-null   object
```

```
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

In [ ]:
```
numerical_df = df.select_dtypes(include='number').drop(columns='Id')
numerical_df.head()
```

Out[ ]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinS
0	60	65.0	8450	7	5	2003	2003	196.0	706	
1	20	80.0	9600	6	8	1976	1976	0.0	978	
2	60	68.0	11250	7	5	2001	2002	162.0	486	
3	70	60.0	9550	7	5	1915	1970	0.0	216	
4	60	84.0	14260	8	5	2000	2000	350.0	655	

5 rows × 37 columns

In [ ]:
```
# Calcular métricas estadísticas
metrics = numerical_df.describe().T
metrics['variance'] = numerical_df.var()
metrics['skewness'] = numerical_df.skew()
metrics['kurtosis'] = numerical_df.kurtosis()
```

In [ ]:
```
# Mostrar las métricas
print(metrics)
```

	count	mean	std	min	25%	\
MSSubClass	1460.0	56.897260	42.300571	20.0	20.00	
LotFrontage	1201.0	70.049958	24.284752	21.0	59.00	
LotArea	1460.0	10516.828082	9981.264932	1300.0	7553.50	
OverallQual	1460.0	6.099315	1.382997	1.0	5.00	
OverallCond	1460.0	5.575342	1.112799	1.0	5.00	
YearBuilt	1460.0	1971.267808	30.202904	1872.0	1954.00	
YearRemodAdd	1460.0	1984.865753	20.645407	1950.0	1967.00	
MasVnrArea	1452.0	103.685262	181.066207	0.0	0.00	
BsmtFinSF1	1460.0	443.639726	456.098091	0.0	0.00	
BsmtFinSF2	1460.0	46.549315	161.319273	0.0	0.00	
BsmtUnfSF	1460.0	567.240411	441.866955	0.0	223.00	
TotalBsmtSF	1460.0	1057.429452	438.705324	0.0	795.75	
1stFlrSF	1460.0	1162.626712	386.587738	334.0	882.00	
2ndFlrSF	1460.0	346.992466	436.528436	0.0	0.00	
LowQualFinSF	1460.0	5.844521	48.623081	0.0	0.00	
GrLivArea	1460.0	1515.463699	525.480383	334.0	1129.50	
BsmtFullBath	1460.0	0.425342	0.518911	0.0	0.00	
BsmtHalfBath	1460.0	0.057534	0.238753	0.0	0.00	
FullBath	1460.0	1.565068	0.550916	0.0	1.00	
HalfBath	1460.0	0.382877	0.502885	0.0	0.00	
BedroomAbvGr	1460.0	2.866438	0.815778	0.0	2.00	
KitchenAbvGr	1460.0	1.046575	0.220338	0.0	1.00	
TotRmsAbvGrd	1460.0	6.517808	1.625393	2.0	5.00	
Fireplaces	1460.0	0.613014	0.644666	0.0	0.00	
GarageYrBlt	1379.0	1978.506164	24.689725	1900.0	1961.00	
GarageCars	1460.0	1.767123	0.747315	0.0	1.00	
GarageArea	1460.0	472.980137	213.804841	0.0	334.50	
WoodDeckSF	1460.0	94.244521	125.338794	0.0	0.00	
OpenPorchSF	1460.0	46.660274	66.256028	0.0	0.00	
EnclosedPorch	1460.0	21.954110	61.119149	0.0	0.00	
3SsnPorch	1460.0	3.409589	29.317331	0.0	0.00	
ScreenPorch	1460.0	15.060959	55.757415	0.0	0.00	
PoolArea	1460.0	2.758904	40.177307	0.0	0.00	
MiscVal	1460.0	43.489041	496.123024	0.0	0.00	
MoSold	1460.0	6.321918	2.703626	1.0	5.00	
YrSold	1460.0	2007.815753	1.328095	2006.0	2007.00	
SalePrice	1460.0	180921.195890	79442.502883	34900.0	129975.00	

	50%	75%	max	variance	skewness	\
MSSubClass	50.0	70.00	190.0	1.789338e+03	1.407657	
LotFrontage	69.0	80.00	313.0	5.897492e+02	2.163569	

LotArea	9478.5	11601.50	215245.0	9.962565e+07	12.207688
OverallQual	6.0	7.00	10.0	1.912679e+00	0.216944
OverallCond	5.0	6.00	9.0	1.238322e+00	0.693067
YearBuilt	1973.0	2000.00	2010.0	9.122154e+02	-0.613461
YearRemodAdd	1994.0	2004.00	2010.0	4.262328e+02	-0.503562
MasVnrArea	0.0	166.00	1600.0	3.278497e+04	2.669084
BsmtFinSF1	383.5	712.25	5644.0	2.080255e+05	1.685503
BsmtFinSF2	0.0	0.00	1474.0	2.602391e+04	4.255261
BsmtUnfSF	477.5	808.00	2336.0	1.952464e+05	0.920268
TotalBsmtSF	991.5	1298.25	6110.0	1.924624e+05	1.524255
1stFlrSF	1087.0	1391.25	4692.0	1.494501e+05	1.376757
2ndFlrSF	0.0	728.00	2065.0	1.905571e+05	0.813030
LowQualFinSF	0.0	0.00	572.0	2.364204e+03	9.011341
GrLivArea	1464.0	1776.75	5642.0	2.761296e+05	1.366560
BsmtFullBath	0.0	1.00	3.0	2.692682e-01	0.596067
BsmtHalfBath	0.0	0.00	2.0	5.700283e-02	4.103403
FullBath	2.0	2.00	3.0	3.035082e-01	0.036562
HalfBath	0.0	1.00	2.0	2.528937e-01	0.675897
BedroomAbvGr	3.0	3.00	8.0	6.654938e-01	0.211790
KitchenAbvGr	1.0	1.00	3.0	4.854892e-02	4.488397
TotRmsAbvGrd	6.0	7.00	14.0	2.641903e+00	0.676341
Fireplaces	1.0	1.00	3.0	4.155947e-01	0.649565
GarageYrBlt	1980.0	2002.00	2010.0	6.095825e+02	-0.649415
GarageCars	2.0	2.00	4.0	5.584797e-01	-0.342549
GarageArea	480.0	576.00	1418.0	4.571251e+04	0.179981
WoodDeckSF	0.0	168.00	857.0	1.570981e+04	1.541376
OpenPorchSF	25.0	68.00	547.0	4.389861e+03	2.364342
EnclosedPorch	0.0	0.00	552.0	3.735550e+03	3.089872
3SsnPorch	0.0	0.00	508.0	8.595059e+02	10.304342
ScreenPorch	0.0	0.00	480.0	3.108889e+03	4.122214
PoolArea	0.0	0.00	738.0	1.614216e+03	14.828374
MiscVal	0.0	0.00	15500.0	2.461381e+05	24.476794
MoSold	6.0	8.00	12.0	7.309595e+00	0.212053
YrSold	2008.0	2009.00	2010.0	1.763837e+00	0.096269
SalePrice	163000.0	214000.00	755000.0	6.311111e+09	1.882876

	kurtosis
MSSubClass	1.580188
LotFrontage	17.452867
LotArea	203.243271
OverallQual	0.096293
OverallCond	1.106413

```
YearBuilt       -0.439552
YearRemodAdd    -1.272245
MasVnrArea      10.082417
BsmtFinSF1      11.118236
BsmtFinSF2      20.113338
BsmtUnfSF        0.474994
TotalBsmtSF     13.250483
1stFlrSF         5.745841
2ndFlrSF        -0.553464
LowQualFinSF    83.234817
GrLivArea        4.895121
BsmtFullBath    -0.839098
BsmtHalfBath    16.396642
FullBath        -0.857043
HalfBath        -1.076927
BedroomAbvGr     2.230875
KitchenAbvGr    21.532404
TotRmsAbvGrd     0.880762
Fireplaces      -0.217237
GarageYrBlt     -0.418341
GarageCars       0.220998
GarageArea       0.917067
WoodDeckSF       2.992951
OpenPorchSF      8.490336
EnclosedPorch   10.430766
3SsnPorch      123.662379
ScreenPorch     18.439068
PoolArea       223.268499
MiscVal        701.003342
MoSold          -0.404109
YrSold          -1.190601
SalePrice        6.536282
```

In [ ]:
```
# Separar la variable objetivo (price) de los predictores
X = numerical_df.drop(columns=[TARGET])
y = numerical_df[TARGET]
```

In [ ]:
```
X.loc[:, X.isna().any()].isna().mean()
```

Out[ ]:
```
LotFrontage    0.177397
MasVnrArea     0.005479
GarageYrBlt    0.055479
dtype: float64
```

In [ ]:
```
X.fillna(X.median(), inplace=True)
```

In [ ]:
```
X.loc[:, X.isna().any()].isna().mean()
```

Out[ ]:
```
Series([], dtype: float64)
```

In [ ]:
```
# Crear el modelo de regresión lineal
model = LinearRegression()

# Usar RFE para seleccionar las mejores variables predictoras
selector = RFE(model, n_features_to_select=5, step=1)
selector = selector.fit(X, y)

# Obtener las columnas seleccionadas
selected_columns = X.columns[selector.support_]

print("Mejores variables predictoras para el precio:")
print(selected_columns)
```

```
Mejores variables predictoras para el precio:
Index(['OverallQual', 'BsmtFullBath', 'FullBath', 'Fireplaces', 'GarageCars'], dtype='object')
```

In [ ]:
```
X = X[selected_columns]
X.head()
```

Out[ ]:

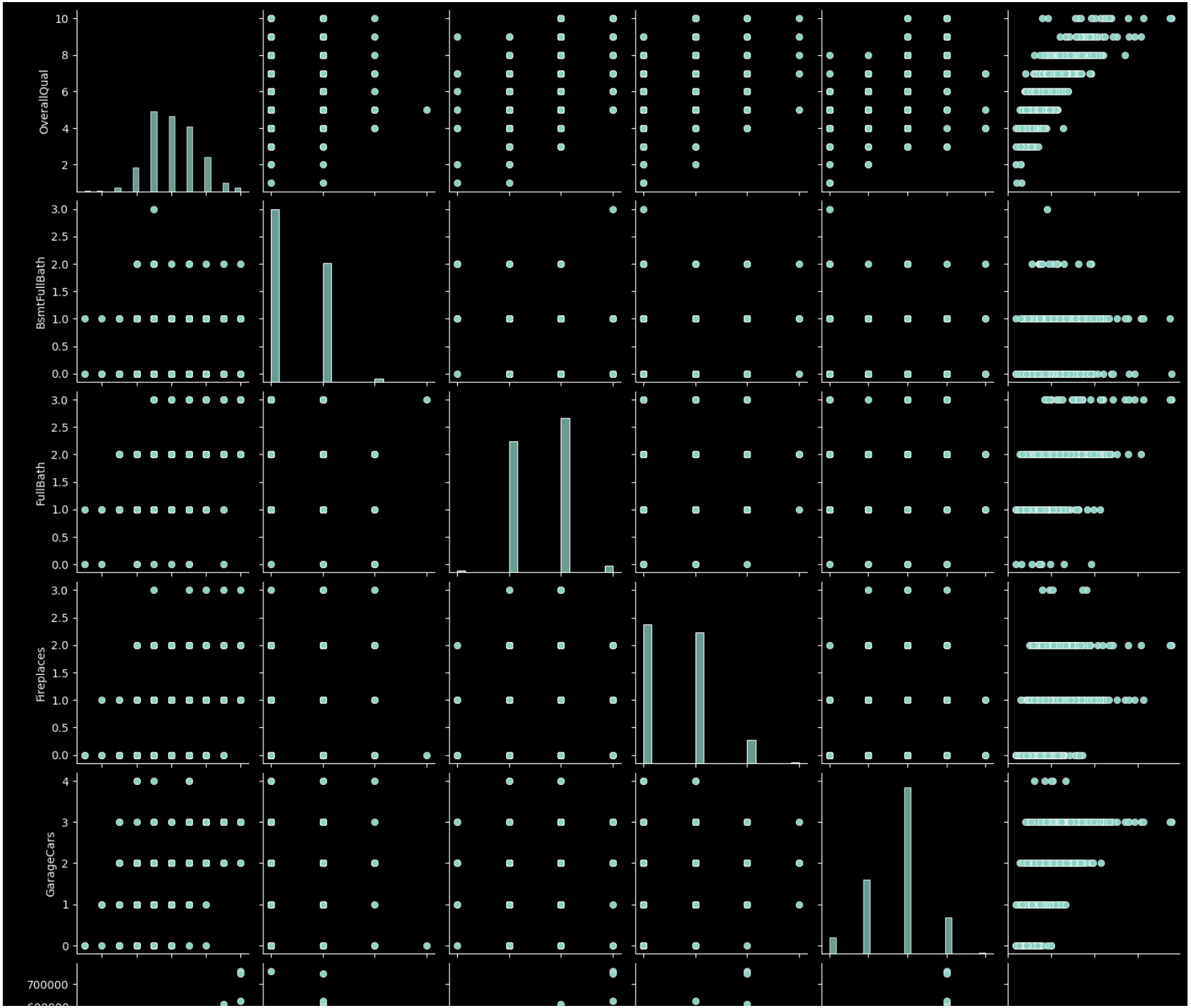
	OverallQual	BsmtFullBath	FullBath	Fireplaces	GarageCars
0	7	1	2	0	2
1	6	0	2	1	2
2	7	1	2	1	2
3	7	1	1	1	3
4	8	1	2	1	3

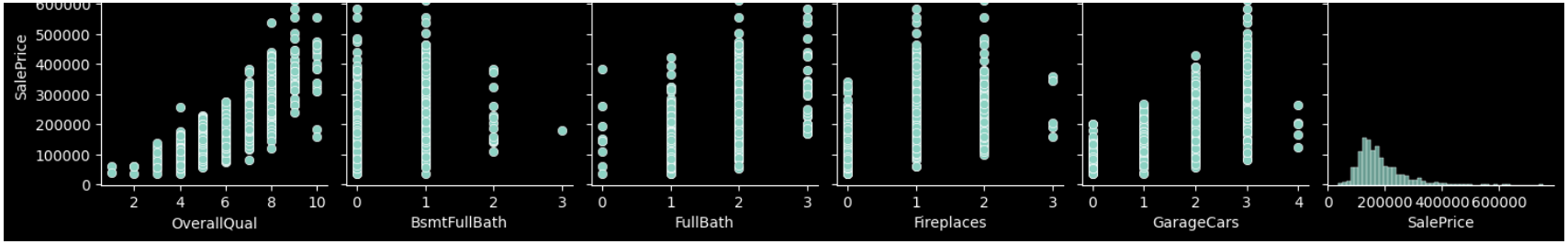
In [ ]:
```
extended_X = pd.concat([X, y], axis=1)
extended_X.head()
```

Out[ ]:

	OverallQual	BsmtFullBath	FullBath	Fireplaces	GarageCars	SalePrice
0	7	1	2	0	2	208500
1	6	0	2	1	2	181500
2	7	1	2	1	2	223500
3	7	1	1	1	3	140000
4	8	1	2	1	3	250000

In [ ]:
```
# Visualizar la distribución de las variables numéricas usando pairplot
sns.pairplot(extended_X)
plt.show()
```


OverallQual
BsmtFullBath
FullBath
Fireplaces
GarageCars
700000

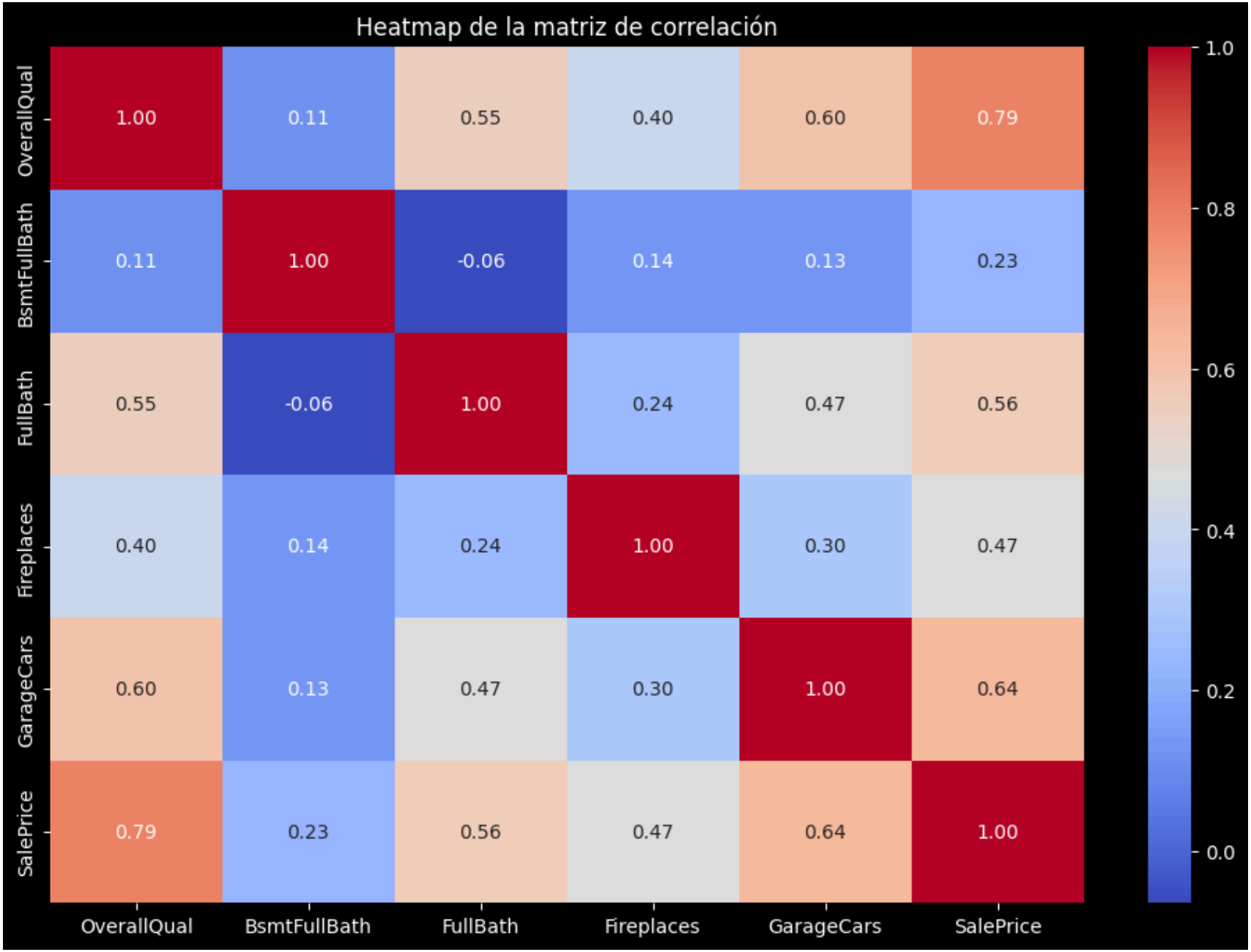


In [ ]:

```python
# Calcular la matriz de correlación
correlation_matrix = extended_X.corr()

# Visualizar la matriz de correlación usando un heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Heatmap de la matriz de correlación')
plt.show()

# Identificar las variables que tienen la mayor correlación con SalePrice
correlation_with_target = correlation_matrix['SalePrice'].sort_values(ascending=False)
print("Variables que más correlacionan con SalePrice:")
print(correlation_with_target)
```


Heatmap de la matriz de correlación
OverallQual
BsmtFullBath
FullBath
Fireplaces
GarageCars
SalePrice
1.00 0.11 0.55 0.40 0.60 0.79
0.11 1.00 -0.06 0.14 0.13 0.23
0.55 -0.06 1.00 0.24 0.47 0.56
0.40 0.14 0.24 1.00 0.30 0.47
0.60 0.13 0.47 0.30 1.00 0.64
0.79 0.23 0.56 0.47 0.64 1.00
OverallQual
BsmtFullBath
FullBath
Fireplaces
GarageCars
SalePrice
1.0
0.8
0.6
0.4
0.2
0.0

```
Variables que más correlacionan con SalePrice:
SalePrice       1.000000
OverallQual     0.790982
GarageCars      0.640409
FullBath        0.560664
Fireplaces      0.466929
BsmtFullBath    0.227122
Name: SalePrice, dtype: float64
```

In [ ]:
```
X = sm.add_constant(X)
model = sm.OLS(y, X).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              SalePrice   R-squared:                       0.719
Model:                            OLS   Adj. R-squared:                  0.718
Method:                 Least Squares   F-statistic:                     745.0
Date:                Thu, 15 Aug 2024   Prob (F-statistic):               0.00
Time:                        16:06:02   Log-Likelihood:                -17617.
No. Observations:                1460   AIC:                         3.525e+04
Df Residuals:                    1454   BIC:                         3.528e+04
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -9.117e+04   5172.036    -17.627      0.000   -1.01e+05    -8.1e+04
OverallQual   2.942e+04   1120.611     26.252      0.000    2.72e+04    3.16e+04
BsmtFullBath  2.057e+04   2191.822      9.384      0.000    1.63e+04    2.49e+04
FullBath      2.299e+04   2493.010      9.223      0.000    1.81e+04    2.79e+04
Fireplaces    1.817e+04   1882.068      9.654      0.000    1.45e+04    2.19e+04
GarageCars    2.082e+04   1906.570     10.919      0.000    1.71e+04    2.46e+04
==============================================================================
Omnibus:                      708.496   Durbin-Watson:                   1.987
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            11225.955
Skew:                           1.858   Prob(JB):                         0.00
Kurtosis:                      16.066   Cond. No.                         32.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Para determinar la significancia estadística de cada variable, se observó el valor p (P>|t|). Una variable se considera estadísticamente significativa si su valor p es menor que 0.05.

Todas las variables en el modelo (`const` , `OverallQual` , `BsmtFullBath` , `FullBath` , `Fireplaces` , `GarageCars`) son estadísticamente significativas, ya que todos los valores p son menores a 0.05.

In [ ]:
```
# Calcular el VIF para cada variable
vif_data = pd.DataFrame()
vif_data["Variable"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

# Mostrar las variables con alta multicolinealidad (VIF > 10)
print(vif_data[vif_data["VIF"] > 10])
```

```
  Variable        VIF
0    const  21.966881
```

In [ ]:
```
# Identificar y excluir las variables con alta multicolinealidad (VIF > 10)
high_vif_vars = vif_data[vif_data["VIF"] > 10]["Variable"].tolist()
X_final = X.drop(columns=high_vif_vars)

# Construir el modelo de regresión final
model_final = sm.OLS(y, X_final).fit()

# Imprimir el resumen del modelo
print(model_final.summary())
```

```
                                OLS Regression Results
==============================================================================
Dep. Variable:              SalePrice   R-squared (uncentered):                   0.945
Model:                            OLS   Adj. R-squared (uncentered):              0.945
Method:                 Least Squares   F-statistic:                              4995.
Date:                Thu, 15 Aug 2024   Prob (F-statistic):                        0.00
Time:                        16:06:02   Log-Likelihood:                         -17758.
No. Observations:                1460   AIC:                                  3.553e+04
Df Residuals:                    1455   BIC:                                  3.555e+04
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
OverallQual  1.552e+04    877.016     17.697      0.000    1.38e+04    1.72e+04
BsmtFullBath 1.596e+04   2396.644      6.660      0.000    1.13e+04    2.07e+04
FullBath     1.659e+04   2716.227      6.107      0.000    1.13e+04    2.19e+04
Fireplaces   2.457e+04   2033.753     12.082      0.000    2.06e+04    2.86e+04
GarageCars    2.41e+04   2089.645     11.535      0.000       2e+04    2.82e+04
==============================================================================
Omnibus:                      927.576   Durbin-Watson:                   1.980
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16982.362
Skew:                           2.646   Prob(JB):                         0.00
Kurtosis:                      18.848   Cond. No.                         16.2
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [ ]:
```python
# Predicciones del modelo
y_pred = model_final.predict(X_final)

# Calcular el Error Cuadrático Medio (MSE)
mse = mean_squared_error(y, y_pred)
print(f"Error Cuadrático Medio (MSE): {mse}")
```

```
Error Cuadrático Medio (MSE): 2148962435.637613
```

In [ ]:
```python
# Calcular el coeficiente de determinación (R²)
r2 = r2_score(y, y_pred)
print(f"Coeficiente de determinación (R²): {r2}")
```

```
Coeficiente de determinación (R²): 0.6592620148026425
```

Análisis de Precisión del Modelo

El coeficiente de determinación (R^2) obtenido es 0.6592620148026425.

Interpretación

- **R^2 = 0.659**: Esto indica que aproximadamente el 65.9% de la variabilidad en la variable dependiente (`SalePrice`) puede ser explicada por las variables independientes en el modelo.
- **Precisión Moderada**: Un R^2 de 0.659 sugiere que el modelo tiene una precisión moderada. No es extremadamente preciso, pero aún proporciona una buena cantidad de información sobre la relación entre las variables independientes y la variable dependiente.

Conclusión

El modelo de regresión final tiene una precisión moderada, lo que significa que puede ser útil para hacer predicciones, pero hay espacio para mejorar. Podríamos considerar:

- Incluir más variables relevantes.
- Probar transformaciones de variables.
- Utilizar técnicas de selección de características.
- Probar otros tipos de modelos de regresión o técnicas de machine learning.

Esto ayudará a mejorar la precisión del modelo y obtener mejores predicciones.