Actividad: Regresión Lineal 2

Por si alguna grafica, linea de codigo, o paso se ve distorsionado en el PDF, aquí está el link:

https://colab.research.google.com/drive/1OucxKJJIS1wCBG0OF-9Fxt51kaQA1xXn?usp=sharing

```
import numpy as np
import pandas as pd
from scipy.stats import t
import scipy.stats as stats
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.metrics import r2_score
import statsmodels.formula.api as smf
from statsmodels.formula.api import ols
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
df = pd.read_csv('/content/drive/MyDrive/7mo Semestre/Colab Notebooks/DataSources/breast_cancer.csv')
df.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280

5 rows × 32 columns

1.- Base de datos completa. No se observan valores faltantes. En caso de haberlos se realiza imputación simple

```
df.columns
```

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')
```

```
df.isnull().sum()
```

```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
```

```
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```

```
df['diagnosis'].unique()
```

```
array(['M', 'B'], dtype=object)
```

```
dummies_diagnosis = pd.get_dummies(df['diagnosis'], prefix = 'diagnosis')
dummies_diagnosis
```

	diagnosis_B	diagnosis_M
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1
...	...	...
564	0	1
565	0	1
566	0	1
567	0	1
568	1	0

569 rows × 2 columns

```
df = pd.concat([df, dummies_diagnosis], axis = 1)
df.drop('diagnosis', axis = 1, inplace = True)
df = df.rename(columns={'concave points_mean': 'concave_points_mean'})
df = df.rename(columns={'concave points_se': 'concave_points_se'})
df = df.rename(columns={'concave points_worst': 'concave_points_worst'})
```

2.- Mostrar que las variables regresoras son independientes. En caso de no serlo realizar el procedimiento correspondiente.

```
corr = df.corr()
corr
```

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compact
id	1.000000	0.074626	0.099770	0.073159	0.096893	-0.012968	
radius_mean	0.074626	1.000000	0.323782	0.997855	0.987357	0.170581	
texture_mean	0.099770	0.323782	1.000000	0.329533	0.321086	-0.023389	
perimeter_mean	0.073159	0.997855	0.329533	1.000000	0.986507	0.207278	
area_mean	0.096893	0.987357	0.321086	0.986507	1.000000	0.177028	
smoothness_mean	-0.012968	0.170581	-0.023389	0.207278	0.177028	1.000000	
compactness_mean	0.000096	0.506124	0.236702	0.556936	0.498502	0.659123	
concavity_mean	0.050080	0.676764	0.302418	0.716136	0.685983	0.521984	
concave_points_mean	0.044158	0.822529	0.293464	0.850977	0.823269	0.553695	
symmetry_mean	-0.022114	0.147741	0.071401	0.183027	0.151293	0.557775	
fractal_dimension_mean	-0.052511	-0.311631	-0.076437	-0.261477	-0.283110	0.584792	
radius_se	0.143048	0.679090	0.275869	0.691765	0.732562	0.301467	
texture_se	-0.007526	-0.097317	0.386358	-0.086761	-0.066280	0.068406	
perimeter_se	0.137331	0.674172	0.281673	0.693135	0.726628	0.296092	
area_se	0.177742	0.735864	0.259845	0.744983	0.800086	0.246552	
smoothness_se	0.096781	-0.222600	0.006614	-0.202694	-0.166777	0.332375	
compactness_se	0.033961	0.206000	0.191975	0.250744	0.212583	0.318943	
concavity_se	0.055239	0.194204	0.143293	0.228082	0.207660	0.248396	
concave_points_se	0.078768	0.376169	0.163851	0.407217	0.372320	0.380676	
symmetry_se	-0.017306	-0.104321	0.009127	-0.081629	-0.072497	0.200774	
fractal_dimension_se	0.025725	-0.042641	0.054458	-0.005523	-0.019887	0.283607	
radius_worst	0.082405	0.969539	0.352573	0.969476	0.962746	0.213120	
texture_worst	0.064720	0.297008	0.912045	0.303038	0.287489	0.036072	
perimeter_worst	0.079986	0.965137	0.358040	0.970387	0.959120	0.238853	
area_worst	0.107187	0.941082	0.343546	0.941550	0.959213	0.206718	
smoothness_worst	0.010338	0.119616	0.077503	0.150549	0.123523	0.805324	
compactness_worst	-0.002968	0.413463	0.277830	0.455774	0.390410	0.472468	
concavity_worst	0.023203	0.526911	0.301025	0.563879	0.512606	0.434926	
concave_points_worst	0.035174	0.744214	0.295316	0.771241	0.722017	0.503053	
symmetry_worst	-0.044224	0.163953	0.105008	0.189115	0.143570	0.394309	
fractal_dimension_worst	-0.029866	0.007066	0.119205	0.051019	0.003738	0.499316	
diagnosis_B	-0.039769	-0.730029	-0.415185	-0.742636	-0.708984	-0.358560	
diagnosis_M	0.039769	0.730029	0.415185	0.742636	0.708984	0.358560	

```
alta_corr = np.where((corr > 0.95) & (corr < 1))
baja_corr = np.where((corr < -0.95) & (corr > -1))
print('Alta correlación: ', alta_corr)
print('Baja correlación: ', baja_corr)
```

```
Alta correlación:  (array([ 1,  1,  1,  1,  3,  3,  3,  3,  4,  4,  4,  4,  4, 11, 11, 13, 14,
       21, 21, 21, 21, 21, 23, 23, 23, 23, 23, 24, 24, 24]), array([ 3,  4, 21, 23,  1,  4, 21, 23,  1,  3, 21,
        1,  3,  4, 23, 24,  1,  3,  4, 21, 24,  4, 21, 23]))
Baja correlación:  (array([], dtype=int64), array([], dtype=int64))
```

Como podemos observar, hay bastantes columnas que se encuentran correlacionadas, asi que las vamos a eliminar.

```
df.drop('perimeter_mean', axis = 1, inplace = True)
df.drop('area_mean', axis = 1, inplace = True)
df.drop('perimeter_worst', axis = 1, inplace = True)
df.drop('radius_se', axis = 1, inplace = True)
df.drop('radius_worst', axis = 1, inplace = True)


corr = df.corr()
alta_corr = np.where((corr > 0.95) & (corr < 1))
baja_corr = np.where((corr < -0.95) & (corr > -1))
print('Alta correlación: ', alta_corr)
print('Baja correlación: ', baja_corr)
```

```
Alta correlación:  (array([], dtype=int64), array([], dtype=int64))
Baja correlación:  (array([], dtype=int64), array([], dtype=int64))
```

Y las variables regresoras que terminaremos utilizando son las siguientes:

```
df.columns
```

```
Index(['id', 'radius_mean', 'texture_mean', 'smoothness_mean',
       'compactness_mean', 'concavity_mean', 'concave_points_mean',
       'symmetry_mean', 'fractal_dimension_mean', 'texture_se', 'perimeter_se',
       'area_se', 'smoothness_se', 'compactness_se', 'concavity_se',
       'concave_points_se', 'symmetry_se', 'fractal_dimension_se',
       'texture_worst', 'area_worst', 'smoothness_worst', 'compactness_worst',
       'concavity_worst', 'concave_points_worst', 'symmetry_worst',
       'fractal_dimension_worst', 'diagnosis_B', 'diagnosis_M'],
      dtype='object')
```

3.- Hipótesis nula de los coeficientes de regresión. Estadístico de prueba, distribución del estadístico de prueba.

Estandarizamos los datos:

```
scaler = StandardScaler()

df_estandar = scaler.fit_transform(df)

df_estandar = pd.DataFrame(df_estandar, columns = df.columns)

df_estandar
```

	id	radius_mean	texture_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mear
0	-0.236405	1.097064	-2.073335	1.568466	3.283515	2.652874	2.532475
1	-0.236403	1.829821	-0.353632	-0.826962	-0.487072	-0.023846	0.548144
2	0.431741	1.579888	0.456187	0.942210	1.052926	1.363478	2.037231
3	0.432121	-0.768909	0.253732	3.283553	3.402909	1.915897	1.451707
4	0.432201	1.750297	-1.151816	0.280372	0.539340	1.371011	1.428493
...	...	...	...	...	...	...	..
564	-0.235732	2.110995	0.721473	1.041842	0.219060	1.947285	2.320965
565	-0.235730	1.704854	2.085134	0.102458	-0.017833	0.693043	1.263669
566	-0.235727	0.702284	2.045574	-0.840484	-0.038680	0.046588	0.105777
567	-0.235725	1.838341	2.336457	1.525767	3.272144	3.296944	2.658866
568	-0.242406	-1.808401	1.221792	-3.112085	-1.150752	-1.114873	-1.261820

569 rows × 28 columns

Entrenamiento:

```
entrenamiento, prueba = train_test_split(df_estandar, test_size=0.20, random_state=42)
```

```
df.columns
```

```
Index(['id', 'radius_mean', 'texture_mean', 'smoothness_mean',
       'compactness_mean', 'concavity_mean', 'concave_points_mean',
       'symmetry_mean', 'fractal_dimension_mean', 'texture_se', 'perimeter_se',
       'area_se', 'smoothness_se', 'compactness_se', 'concavity_se',
       'concave_points_se', 'symmetry_se', 'fractal_dimension_se',
       'texture_worst', 'area_worst', 'smoothness_worst', 'compactness_worst',
       'concavity_worst', 'concave_points_worst', 'symmetry_worst',
       'fractal_dimension_worst', 'diagnosis_B', 'diagnosis_M'],
      dtype='object')
```

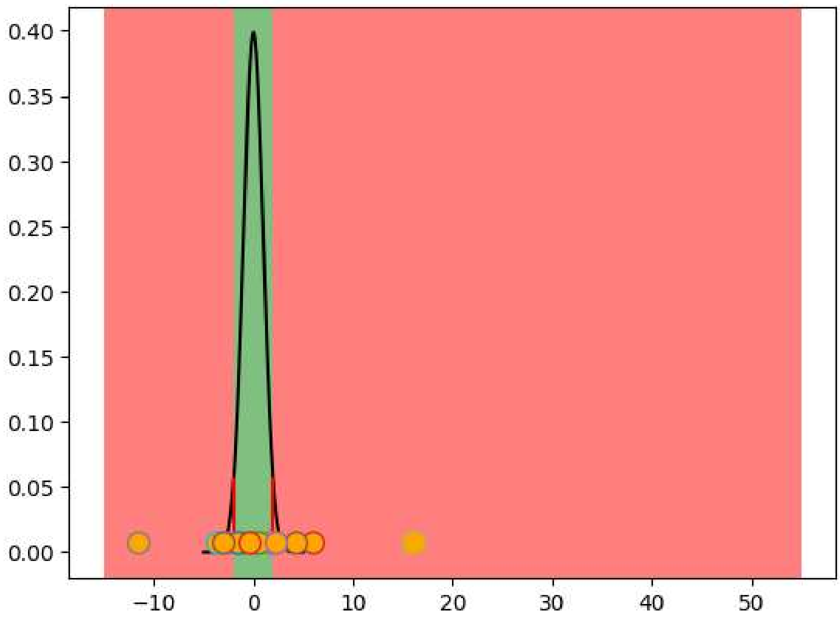
```
modelo = smf.ols(formula = 'radius_mean~id+texture_mean+smoothness_mean+compactness_mean+concavity_mean+concave_point:

modelo = modelo.fit()

print(modelo.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            radius_mean   R-squared:                       0.954
Model:                            OLS   Adj. R-squared:                  0.952
Method:                 Least Squares   F-statistic:                     344.7
Date:                Wed, 06 Sep 2023   Prob (F-statistic):          1.50e-268
Time:                        06:33:44   Log-Likelihood:                 55.550
No. Observations:                 455   AIC:                            -57.10
Df Residuals:                     428   BIC:                             54.15
Df Model:                          26
Covariance Type:            nonrobust
===========================================================================================
                              coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------------
Intercept                   0.0048      0.010      0.466      0.641      -0.016       0.025
id                         -0.0068      0.010     -0.655      0.513      -0.027       0.014
texture_mean                0.0854      0.035      2.412      0.016       0.016       0.155
smoothness_mean             0.0463      0.029      1.601      0.110      -0.011       0.103
compactness_mean            0.3412      0.057      5.953      0.000       0.229       0.454
concavity_mean             -0.1055      0.083     -1.277      0.202      -0.268       0.057
concave_points_mean         0.3414      0.081      4.218      0.000       0.182       0.501
symmetry_mean              -0.0367      0.021     -1.766      0.078      -0.077       0.004
fractal_dimension_mean     -0.4009      0.034    -11.627      0.000      -0.469      -0.333
texture_se                  0.0163      0.021      0.766      0.444      -0.026       0.058
perimeter_se               -0.0706      0.041     -1.716      0.087      -0.151       0.010
area_se                    -0.0006      0.043     -0.015      0.988      -0.085       0.083
smoothness_se              -0.0052      0.020     -0.265      0.791      -0.044       0.034
compactness_se             -0.0439      0.038     -1.153      0.250      -0.119       0.031
concavity_se                0.0657      0.038      1.732      0.084      -0.009       0.140
concave_points_se          -0.0550      0.034     -1.621      0.106      -0.122       0.012
symmetry_se                 0.0082      0.024      0.350      0.727      -0.038       0.054
fractal_dimension_se        0.0568      0.029      1.962      0.050      -0.000       0.114
texture_worst              -0.1192      0.044     -2.701      0.007      -0.206      -0.032
area_worst                  0.5639      0.035     16.050      0.000       0.495       0.633
smoothness_worst           -0.1198      0.033     -3.659      0.000      -0.184      -0.055
compactness_worst          -0.1018      0.061     -1.679      0.094      -0.221       0.017
concavity_worst            -0.0266      0.057     -0.464      0.643      -0.139       0.086
concave_points_worst        0.0376      0.063      0.599      0.549      -0.086       0.161
symmetry_worst             -0.0119      0.031     -0.388      0.699      -0.072       0.049
fractal_dimension_worst     0.1011      0.045      2.259      0.024       0.013       0.189
diagnosis_B                -0.0619      0.021     -2.989      0.003      -0.103      -0.021
==============================================================================
Omnibus:                       39.226   Durbin-Watson:                   1.982
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              182.226
Skew:                          -0.109   Prob(JB):                     2.69e-40
Kurtosis:                       6.093   Cond. No.                         38.1
==============================================================================

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

Nuestra R^2 es bastante buena. Si queremos simplificar el modelo, eliminamos las β que su p-valor sea mayor a 0.05

Para un 95% de confianza realiza un diagrama en donde se muestre la distribución del estadístico de prueba, la zona de aceptación y la zona de rechazo.

```
indice_columna = 0
dof = df_estandar.shape[0] - 1
alpha = 0.05
ntails = 2

tcrit = abs(stats.t.ppf(alpha/ntails, dof))

xs = np.linspace(-5,5,1000)

plt.plot(xs, stats.t.pdf(xs,dof), 'k')
plt.vlines([-tcrit, tcrit], 0.0, stats.t.pdf(tcrit,dof), colors='r')

for i in range(1, len(modelo.tvalues)):
    t = modelo.tvalues[i]
    plt.plot(t, 0.008, marker="o", markersize=10, markerfacecolor="orange")

plt.axvspan(-tcrit, tcrit, facecolor='green', alpha=0.5, label = 'Aceptación')
plt.axvspan(-15, -tcrit, facecolor='red', alpha=0.5, label = 'Rechazo')
plt.axvspan(tcrit, 55, facecolor='red', alpha=0.5)
plt.show()
```



Si observamos, hay muchos valores que se encuentran en la región de rechazo, así que esta prueba nos indica que aun quedan variables regresoras que podemos eliminar para hacer mas simple nuestro modelo.

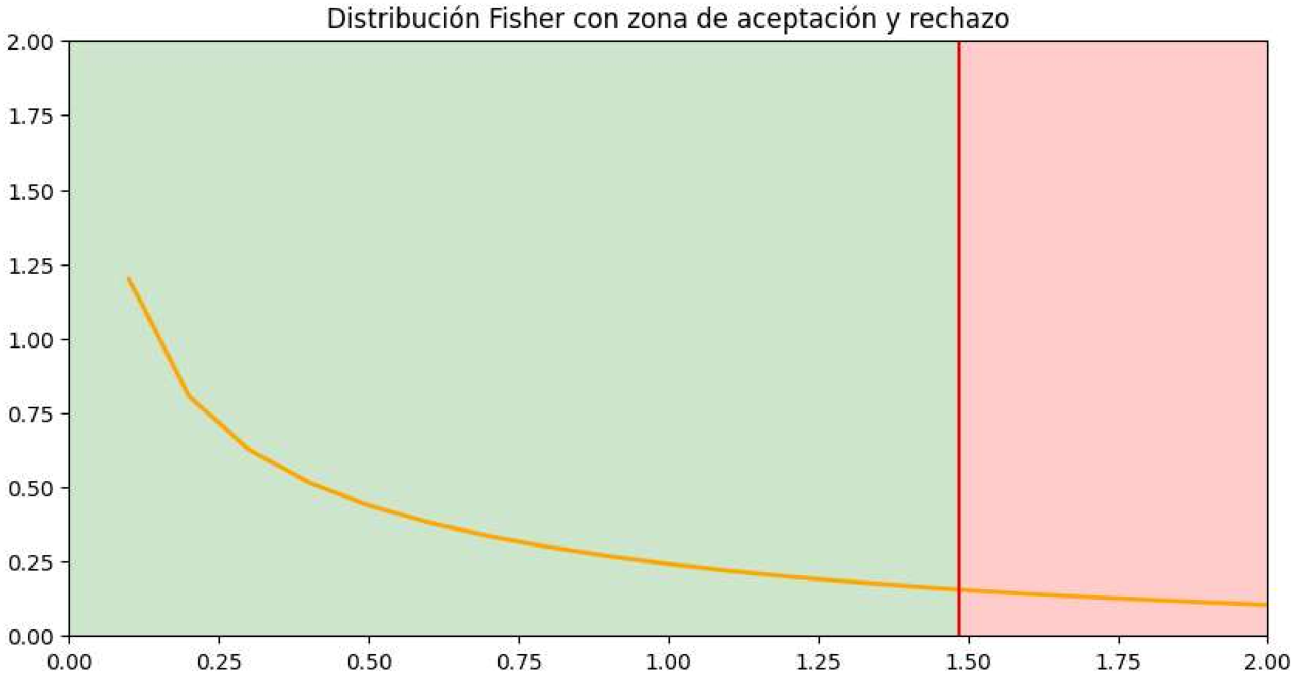
4.- Hipótesis nula de la significancia del modelo (prueba F-Fisher). Menciona que distribución tiene el estadístico de prueba con qué número de grados de libertad. Para un 95% de confianza realiza un diagrama en donde se muestre la distribución del estadístico de prueba, la zona de aceptación y la zona de rechazo.

```
fTest = np.identity(len(modelo.params))
fTest = fTest[1:,:]
modelo.f_test(fTest)
```

```
<class 'statsmodels.stats.contrast.ContrastResults'>
<F test: F=344.692004902787, p=1.501640435420423e-268, df_denom=428, df_num=26>
```

```
rand_f_samples = stats.f.rvs(dfn=30, dfd=424, size=100000)

plt.figure(figsize=(10, 5))

plt.plot(
    np.arange(0, 4, 0.1),
    stats.f.pdf(np.arange(0, 4, 0.1), dfn=1, dfd=453),
    "-",
    linewidth=2,
    color="orange",
)

plt.xlim(0, 2)
plt.ylim(0, 2)

plt.title("Distribución Fisher con zona de aceptación y rechazo")

critical_value = stats.f.ppf(q=1-.05, dfn=30, dfd=424)
print('Valor critico', critical_value)

plt.vlines([-critical_value, critical_value],0,4, colors='r', label = 'valores criticos')
plt.axvspan(-critical_value, critical_value, facecolor='green', alpha=0.2)
plt.axvspan(critical_value, 10, facecolor='red', alpha=0.2)


plt.show()
```

```
Valor critico 1.4861716526654176
```



Si vemos nuestra F en el test de Fisher, nos da como resultado, $F = 344.692004902787$, valor que entra en la región de rechazo (que la marca el valor critico = 1.4861716526654176) de la hipotesis nula, infiriendo que el modelo sí es significativo.

5.- Realiza un modelo de regresión hacia atrás (backward). Explica el criterio para ir eliminando variables del modelo.

Bueno, ahora sabemos que el modelo tiene espacio para mejorar, así que vamos a mejorar el modelo analizando el p-valor de nuestas variables regresoras, y si son mayores a 0.05, los eliminamos porque se acepta la hipotesis nula que implica que la variable es igual a cero.

```
modelo = smf.ols(formula = 'radius_mean~texture_mean+compactness_mean+concave_points_mean+fractal_dimension_mean+frac

modelo = modelo.fit()

print(modelo.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:            radius_mean   R-squared:                       0.948
Model:                            OLS   Adj. R-squared:                  0.947
Method:                 Least Squares   F-statistic:                     812.5
Date:                Wed, 06 Sep 2023   Prob (F-statistic):          3.46e-278
Time:                        06:33:45   Log-Likelihood:                 26.391
No. Observations:                 455   AIC:                            -30.78
Df Residuals:                     444   BIC:                             14.54
Df Model:                          10                                         
Covariance Type:            nonrobust                                         
=============================================================================================
                                coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------------
Intercept                     0.0023      0.011      0.215      0.830      -0.019       0.024
texture_mean                  0.0658      0.030      2.211      0.028       0.007       0.124
compactness_mean              0.1482      0.034      4.337      0.000       0.081       0.215
concave_points_mean           0.3274      0.040      8.218      0.000       0.249       0.406
fractal_dimension_mean       -0.3467      0.027    -12.892      0.000      -0.400      -0.294
fractal_dimension_se          0.0272      0.016      1.660      0.098      -0.005       0.059
texture_worst                -0.1034      0.031     -3.321      0.001      -0.165      -0.042
area_worst                    0.5270      0.025     20.919      0.000       0.477       0.576
smoothness_worst             -0.0792      0.017     -4.563      0.000      -0.113      -0.045
fractal_dimension_worst       0.0410      0.025      1.659      0.098      -0.008       0.089
diagnosis_B                  -0.0399      0.020     -1.955      0.051      -0.080       0.000
==============================================================================
Omnibus:                       34.774   Durbin-Watson:                   1.984
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               70.536
Skew:                          -0.445   Prob(JB):                     4.82e-16
Kurtosis:                       4.712   Cond. No.                         10.0
==============================================================================

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

Empeoró ligeramente nuestra R^2, pero disminuimos considerablemente el numero de variables regresoras. Ahora este modelo nos dió otros p-valores que son mayores que 0.05, así que vamos a ver que tanto cambia el modelo si los eliminamos.

```
modelo = smf.ols(formula = 'radius_mean~texture_mean+compactness_mean+concave_points_mean+fractal_dimension_mean+textu

modelo = modelo.fit()

print(modelo.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:            radius_mean   R-squared:                       0.947
Model:                            OLS   Adj. R-squared:                  0.946
Method:                 Least Squares   F-statistic:                     1133.
Date:                Wed, 06 Sep 2023   Prob (F-statistic):          6.33e-280
Time:                        06:33:45   Log-Likelihood:                 19.774
No. Observations:                 455   AIC:                            -23.55
Df Residuals:                     447   BIC:                             9.414
Df Model:                           7                                         
Covariance Type:            nonrobust                                         
============================================================================================
                               coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------------
Intercept                    0.0035      0.011      0.318      0.751      -0.018       0.025
texture_mean                 0.0585      0.029      2.017      0.044       0.002       0.116
compactness_mean             0.1730      0.032      5.394      0.000       0.110       0.236
concave_points_mean          0.3350      0.036      9.280      0.000       0.264       0.406
fractal_dimension_mean      -0.3084      0.021    -14.792      0.000      -0.349      -0.267
texture_worst               -0.0824      0.030     -2.759      0.006      -0.141      -0.024
area_worst                   0.5432      0.025     21.734      0.000       0.494       0.592
```

```
smoothness_worst          -0.0774      0.016     -4.988      0.000      -0.108      -0.047
==============================================================================
Omnibus:                       46.278   Durbin-Watson:                   1.982
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               96.896
Skew:                          -0.573   Prob(JB):                     9.11e-22
Kurtosis:                       4.949   Cond. No.                         7.71
==============================================================================

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

Nuevamente, nuestra R^2 empeoró ligeramente, pero personalmente considero que es decremento minimo, en comparación al numero de variables al que se pudo minimizar gracias a este decremento. Y ya no quedan p-valores que se puedan eliminar.

Asi que vamos a hacer las predicciones con este modelo.

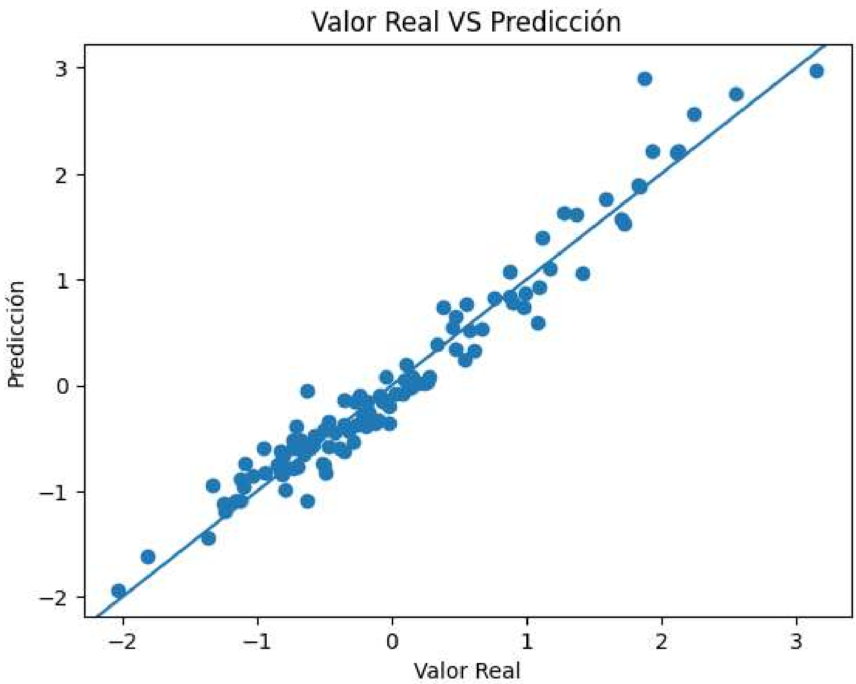
6.- Comparación entre datos reales y predicción. Análisis de los resultados.

```
y_pred = modelo.predict(prueba)
y_test = prueba["radius_mean"]
```

```
print("MAE = ", mean_absolute_error(y_test, y_pred))
print("MSE = ", mean_squared_error(y_test, y_pred))
print("R^2 = ", r2_score(y_test, y_pred))
```

```
MAE =  0.16672250002141548
MSE =  0.0468402226218193
R^2 =  0.9519550705600349
```

```
plt.scatter(y_test, y_pred)
plt.title("Valor Real VS Predicción")
plt.xlabel("Valor Real")
plt.ylabel("Predicción")
plt.axline([0, 0], [1, 1])
plt.show()
```



Como podemos ver nuestro modelo es bastante bueno porque mayormente se alinean los resultados verdaderos con las predicciones de manera lineal.

Conclusión

Al darnos la libertad de analizar con diferentes pruebas la eficiencia del modelo, podemos elegir de una manera mas optima las variables que valen la pena eliminar para mejorar el modelo. Además, podemos simplificar el modelo y evitar que se sobreajuste al eliminar variables regresoras.