Proyecto 3 Estadística - Modelos de regresión

Nicolas Vargas Flores 1001368855

Jader Stalyn Chingal Atis 1085948736

Universidad de Antioquia

Primera parte - Análisis exploratorio de datos (AED)

Se carga la base de datos "`datos_state.csv`".

```
BD = readtable("datos_state.csv");
```

Para calculas las estadísticas descriptivas pertinentes al proyecto se utiliza la herramienta "*Compute by group*".

```
% Compute group summary
AED_stats = groupsummary(BD,"Life_Exp",1,["mean","median","mode","max","min", ...
    "std","var","nummissing"],true(1,9))
```

```
AED_stats = 1×74 table
```

...

	disc_Life_Exp	GroupCount	mean_Population	median_Population
1	[65, 73.6]	50	4.246420000000000e+03	2.838500000000000e+03

Con la información recopilada en "AED_stats" se procede a presentar las estadísticas de los datos.

```
varNames = AED_stats.Properties.VariableNames;

% La información se almacena en una tabla
statsTable = table('Size', [9 8], 'VariableTypes', repmat({'double'}, 1, 8), ...
    'VariableNames', {'Means', 'Medians', 'Modes', 'Maxs', 'Mins', 'Stds', 'Vars',
...
    'Nummissings'}, 'RowNames', {'Population', 'Illiteracy', 'Income', 'Life_Exp',
...
    'Murder', 'HS_Grad', 'Frost', 'Area', 'Density_pob'});

% Cada fila es una estadística de cada variable
for i = 1:length(varNames)
    if startsWith(varNames{i}, 'mean_')
        statsTable.('Means')(varNames{i}(6:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'median_')
        statsTable.('Medians')(varNames{i}(8:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'mode_')
        statsTable.('Modes')(varNames{i}(6:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'max_')
        statsTable.('Maxs')(varNames{i}(5:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'min_')
        statsTable.('Mins')(varNames{i}(5:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'std_')
```

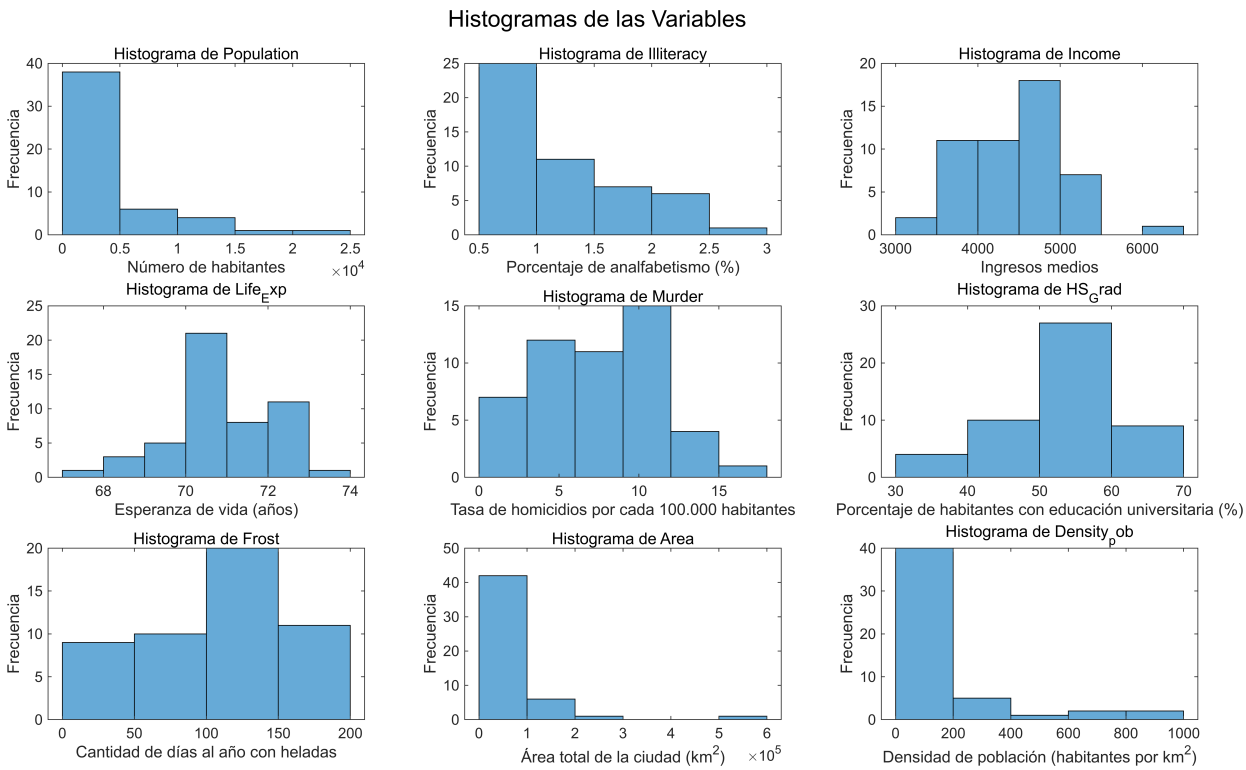
```
        statsTable.('Stds')(varNames{i}(5:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'var_')
        statsTable.('Vars')(varNames{i}(5:end)) = AED_stats.(varNames{i});
    elseif startsWith(varNames{i}, 'nummissing_')
        statsTable.('Nummissings')(varNames{i}(12:end)) = AED_stats.(varNames{i});
    end
end

disp(statsTable)
```

	Means	Medians	Modes	Maxs	Mins	Stds
Population	4246.42	2838.5	365	21198	365	4464.49143338584
Illiteracy	1.17	0.95	0.6	2.8	0.5	0.609533110048091
Income	4435.8	4519	3098	6315	3098	614.469939152803
Life_Exp	70.8786	70.675	70.55	73.6	67.96	1.34239355219682
Murder	7.378	6.85	2.3	15.1	1.4	3.69153969315277
HS_Grad	53.108	53.25	38.5	67.3	37.8	8.07699782577357
Frost	104.46	114.5	20	188	0	51.9808481214819
Area	70735.88	54277	1049	566432	1049	85327.2996223509
Density_pob	149.22447334164	73.01543341	0.644384498	975.003324	0.644384498	221.006340294913

Para tener una vista general de los datos se realizan histogramas para cada variable.

```
% Histogramas
variables = {'Population', 'Illiteracy', 'Income', 'Life_Exp', 'Murder', ...
    'HS_Grad', 'Frost', 'Area', 'Density_pob'};

figure;
for i = 1:length(variables)
    subplot(3, 3, i);
    histogram(BD.(variables{i}));
    title(['Histograma de ' variables{i}]);
    switch variables{i}
        case 'Population'
            xlabel('Número de habitantes');
        case 'Illiteracy'
            xlabel('Porcentaje de analfabetismo (%)');
        case 'Income'
            xlabel('Ingresos medios');
        case 'Life_Exp'
            xlabel('Esperanza de vida (años)');
        case 'Murder'
            xlabel('Tasa de homicidios por cada 100.000 habitantes');
        case 'HS_Grad'
            xlabel('Porcentaje de habitantes con educación universitaria (%)');
        case 'Frost'
            xlabel('Cantidad de días al año con heladas');
        case 'Area'
            xlabel('Área total de la ciudad (km^2)');
        case 'Density_pob'
```

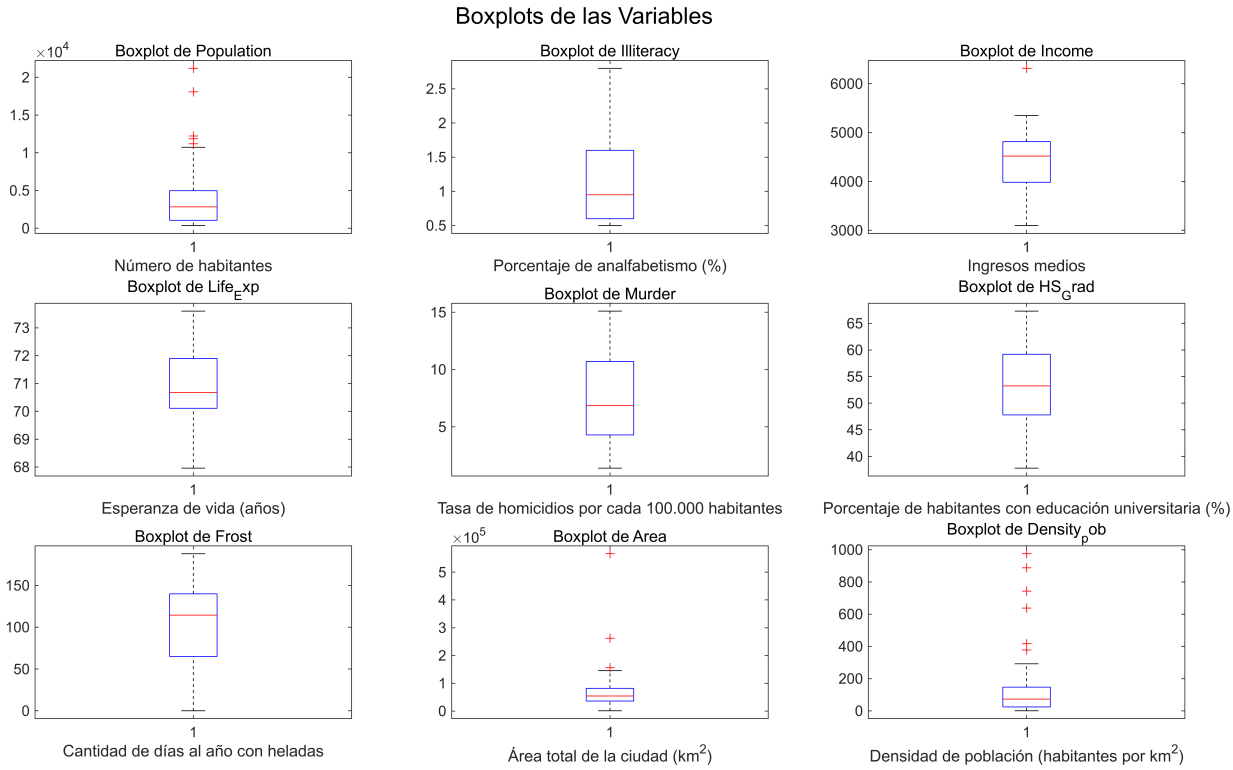
```
            xlabel('Densidad de población (habitantes por km^2)');
    end
    ylabel('Frecuencia');
end
sgtitle('Histogramas de las Variables');
set(gcf, 'Position', get(0, 'Screensize')); % Maximiza la figura
```



Los siguientes diagramas de cajas y bigotes serán útiles para visualizar la distribución de la información, y ser conscientes de los posibles outliers que hay en cada variable.

```
% Boxplots
figure;
for i = 1:length(variables)
    subplot(3, 3, i);
    boxplot(BD.(variables{i}));
    title(['Boxplot de ' variables{i}]);
    switch variables{i}
        case 'Population'
            xlabel('Número de habitantes');
        case 'Illiteracy'
            xlabel('Porcentaje de analfabetismo (%)');
        case 'Income'
            xlabel('Ingresos medios');
        case 'Life_Exp'
            xlabel('Esperanza de vida (años)');
        case 'Murder'
```

```
            xlabel('Tasa de homicidios por cada 100.000 habitantes');
        case 'HS_Grad'
            xlabel('Porcentaje de habitantes con educación universitaria (%)');
        case 'Frost'
            xlabel('Cantidad de días al año con heladas');
        case 'Area'
            xlabel('Área total de la ciudad (km^2)');
        case 'Density_pob'
            xlabel('Densidad de población (habitantes por km^2)');
    end
end
sgtitle('Boxplots de las Variables');
set(gcf, 'Position', get(0, 'Screensize')); % Maximiza la figura
```

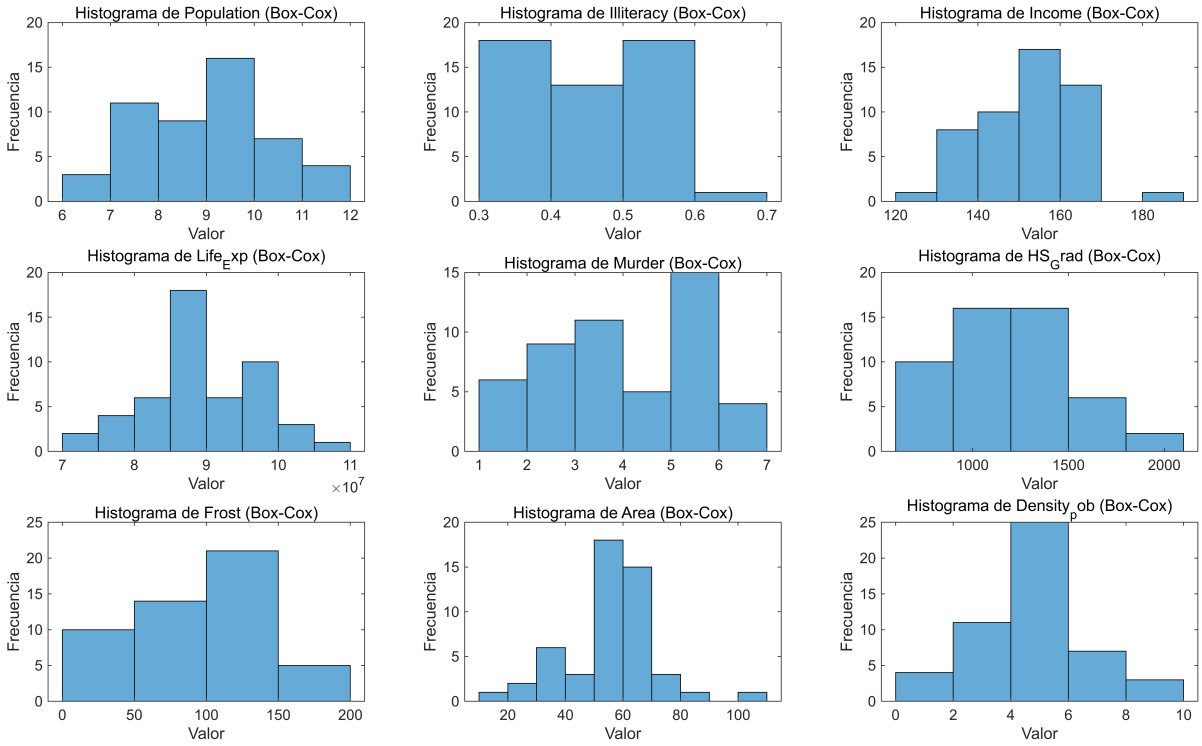


Distribuciones

Notese que en los diagramas anteriores se ve que existen muchos datos outliers que aumentan la varianza significativamente, lo cual hace que la normalidad de los datos se vea gravemente afectada. Con el propósito de encontrar una distribución estadística que nos sirva para realizar los posteriores análisis; se transforman los datos para atenuar el peso de los outliers dentro del desenlace estadístico.

```
% Librería de estadísticas
import statistics.*

figure;
% Box-Cox transformation
```

```
for i = 1:length(variables)
    % Aplicar la transformación de Box-Cox
    [transformed, lambda] = boxcox(BD.(variables{i})+1);

    % Se almacenan los datos en una tabla
    BD.([variables{i} '_boxcox']) = transformed;

    % Histogramas con los datos transformados
    subplot(3, 3, i);
    histogram(transformed);
    title(['Histograma de ' variables{i} ' (Box-Cox)']);
    xlabel('Valor');
    ylabel('Frecuencia');
end
set(gcf, 'Position', get(0, 'Screensize'));
```



La transformación de Box-Cox es una técnica estadística que se utiliza para transformar datos no normales en una forma que se aproxime a la normalidad. La transformación de Box-Cox se define como:

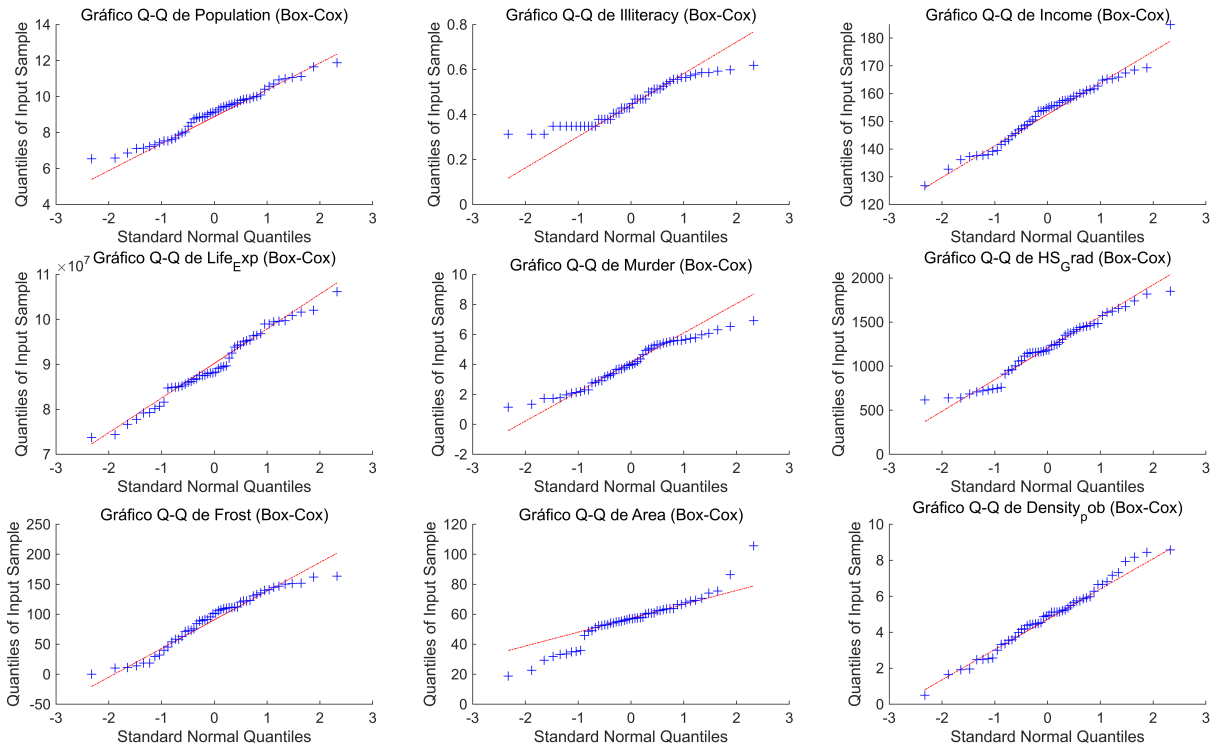
$$y(\lambda) = \begin{cases} \dfrac{y^{\lambda} - 1}{\lambda} & \text{si } \lambda \neq 0, \\ \ln(y) & \text{si } \lambda = 0, \end{cases}$$

donde y es la variable que se desea transformar y λ es el parámetro de transformación. El valor de λ se elige de manera que maximice la verosimilitud de los datos transformados.

Para este estudio la transformación de Box-Cox es útil porque ayuda a estabilizar la varianza, hace que los datos se comporten más como una distribución normal, mejora la validez de las medidas de asociación (como la correlación) y hace que los patrones en los datos y los residuos sean más fáciles de identificar. Box-Cox es especialmente útil para manejar los valores atípicos.

Una herramienta útil para visualizar la normalidad de un conjunto de datos son los gráficos Q-Q, presentados a continuación.

```
figure;
% Gráficos Q-Q para cada variable
for i = 1:length(variables)
    subplot(3, 3, i);
    qqplot(BD.([variables{i} '_boxcox']));
    title(['Gráfico Q-Q de ' variables{i} ' (Box-Cox)']);
end
set(gcf, 'Position', get(0, 'Screensize'));
```



Contrastando la normalidad

Es importante conocer las distribuciones de los datos; especialmente conocer cuales de ellos tienen una distribución Gaussiana. Esta sección realizará una prueba de Shapiro-Wilk para cada variable para evaluar si los datos siguen una distribución normal.

```
results = table('Size', [length(variables) 4], ...
    'VariableTypes', {'string', 'logical', 'double', 'string'}, ...
```

```
    'VariableNames', {'Variable', 'H0_Rejected', 'P_Value', 'Distribution'});

% Se descargó prueba swtest.m en File Exchange MatLab
for i = 1:length(variables)
    % Aplicar la prueba de Shapiro-Wilk
    [h, p] = swtest(BD.([variables{i} '_boxcox']));

    % Determinar si los datos siguen una distribución normal
    if h == 0
        distribution = 'Gaussiana';
    else
        distribution = 'No Gaussiana';
    end

    results(i, :) = {variables{i}, h, p, distribution};
end

disp(results)
```

Variable	H0_Rejected	P_Value	Distribution
"Population"	false	0.388168726841706	"Gaussiana"
"Illiteracy"	true	0.00378857579766434	"No Gaussiana"
"Income"	false	0.337746272996109	"Gaussiana"
"Life_Exp"	false	0.500640020842799	"Gaussiana"
"Murder"	false	0.0639329661711463	"Gaussiana"
"HS_Grad"	false	0.133140146531281	"Gaussiana"
"Frost"	true	0.0455866554185235	"No Gaussiana"
"Area"	true	0.00748834749528549	"No Gaussiana"
"Density_pob"	false	0.871219126976328	"Gaussiana"

Segunda parte - Estadística inferencial

Modelos univariados

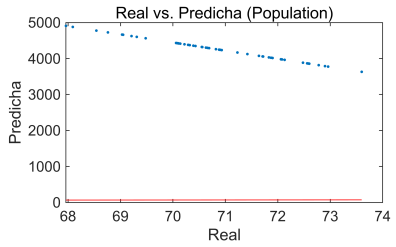
A continuación se van a realizar múltiples modelos de regresión lineal univariada teniendo únicamente en cuenta aquellas variables que sigan una distribución Gaussiana.

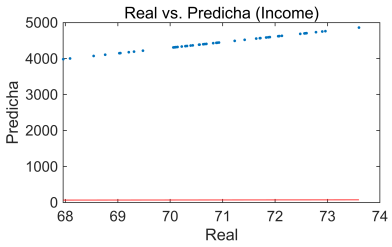
La normalidad de las variables es una suposición importante que subyace en muchos modelos y técnicas. Los modelos de regresión lineal, por ejemplo, asumen que los errores (es decir, las diferencias entre las respuestas observadas y las predichas) siguen una distribución normal. Esta suposición permite realizar inferencias estadísticas válidas a partir del modelo. Si una variable no sigue una distribución normal, puede violar esta suposición y hacer que las inferencias del modelo sean menos precisas o incluso inválidas.

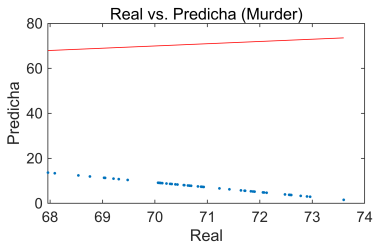
Lo anterior es para argumentar la necesidad de considerar solo las variables que siguen una distribución normal. Eso garantiza que las inferencias basadas en el modelo de regresión lineal sean válidas y precisas.

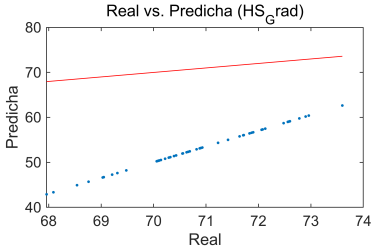
```
% Lista de variables predictoras
predictors = {'Population', 'Illiteracy', 'Income', 'Murder', ...
    'HS_Grad', 'Frost', 'Area', 'Density_pob'};

% Variable de salida
```

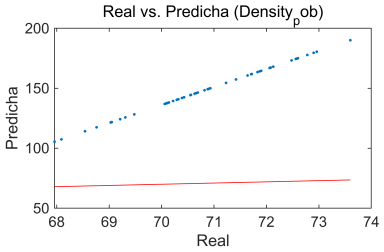
```
response = 'Life_Exp';

results = table('Size', [length(predictors) 4], ...
    'VariableTypes', {'string', 'double', 'double', 'double'}, ...
    'VariableNames', {'Variable', 'R_Squared', 'RMSE', 'P_Value'});

figure;
for i = 1:length(predictors)
    % Aplicar la prueba de Shapiro-Wilk
    [h, p] = swtest(BD.([predictors{i} '_boxcox']));

    % Si los datos siguen una distribución normal, se consideran en el modelo de
regresión lineal
    if h == 0
        lm = fitlm(BD, [predictors{i} ' ~ ' response]);

        % Almacenar los resultados en la tabla
        results(i, :) = {predictors{i}, lm.Rsquared.Ordinary, lm.RMSE,
lm.Coefficients.pValue(2)};

        % Obtener las predicciones del modelo
        predictions = predict(lm, BD);

        % Gráfico de las respuestas reales Vs las predicciones
        subplot(3, 3, i);
        plot(BD.(response), predictions, '.');
        hold on;
        plot([min(BD.(response)), max(BD.(response))], [min(BD.(response)), max(BD.
(response))], 'r');
        hold off;
        title(['Real vs. Predicha (' predictors{i} ')']);
        xlabel('Real');
        ylabel('Predicha');
    end
end
set(gcf, 'Position', get(0, 'Screensize'));
```












```
disp(results)
```

Variable	R_Squared	RMSE	P_Value
"Population"	0.00463106818249603	4500.29989704977	0.638659368647134
<missing>	0	0	0
"Income"	0.115773695674702	583.794202558109	0.0156172816016912
"Murder"	0.609720088771197	2.33009196683398	2.26007028627032e-11
"HS_Grad"	0.338975708088246	6.63492620799347	9.19609565380663e-06
<missing>	0	0	0
<missing>	0	0	0
"Density_pob"	0.00829224459453315	222.368880232051	0.529397156750111

Modelo lineal multivariado

Teniendo en cuenta las variables predictoras seleccionadas en la sección anterior, se propone el siguiente modelo de regresión para predecir la esperanza de vida de una población.

```
% Lista de variables predictoras
predictors = {'Population', 'Income', 'Murder', 'HS_Grad', 'Density_pob'};

% Variable de salida
response = 'Life_Exp';

% Fórmula del modelo
formula = [response ' ~ ' strjoin(predictors, ' + ')];
```

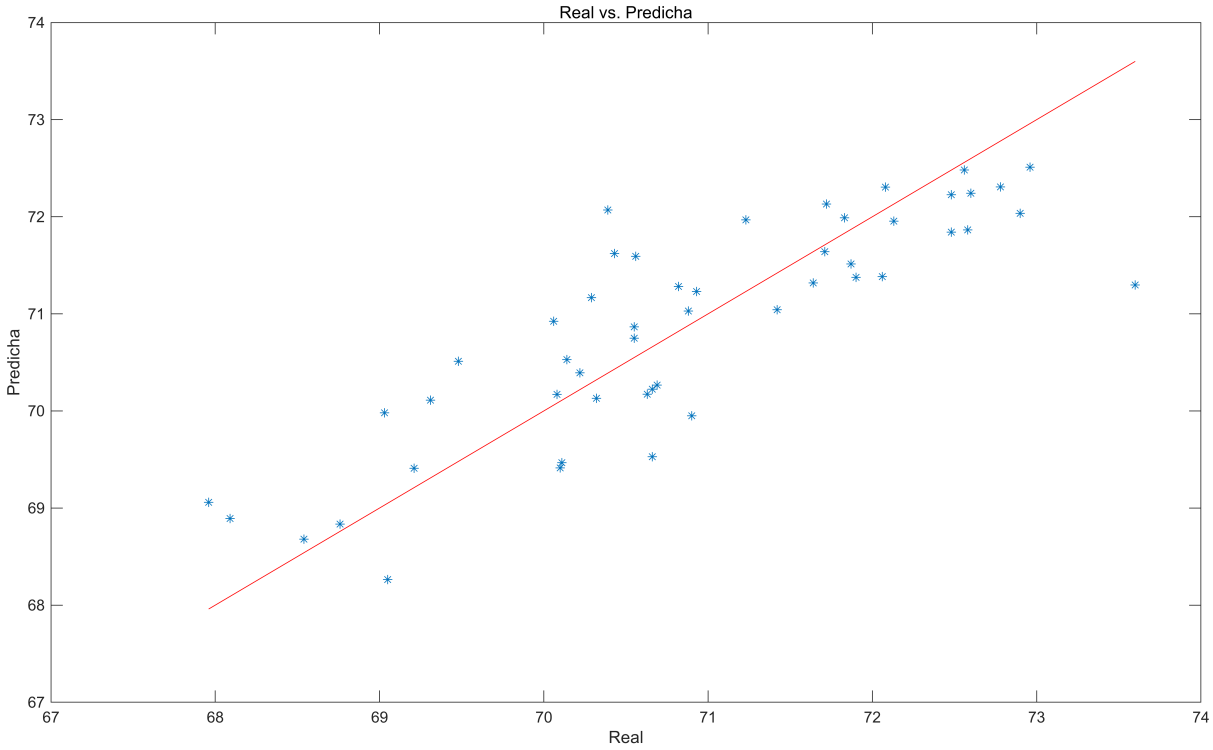
```
% Modelo de regresión lineal multivariado
lm = fitlm(BD, formula);

% Mostrar las métricas de evaluación del modelo
disp(lm)
```

```
Linear regression model:
    Life_Exp ~ 1 + Population + Income + Murder + HS_Grad + Density_pob

Estimated Coefficients:
                       Estimate                SE                tStat                pValue
                    ____________________    ____________________    ____________________    ____________________

    (Intercept)          70.7539783154793        1.07824713543101        65.6194447362912    1.54613405364646e-45
    Population       7.23063812602683e-05    2.85269867444416e-05        2.53466592556808      0.0148877778570757
    Income           2.94006778925043e-05    0.000275654419027494       0.106657741951787       0.915545330356118
    Murder             -0.282103599092049      0.0397057548990406       -7.10485419076785    8.02709996875579e-09
    HS_Grad            0.0349653067303268      0.0224647005560794         1.55645549973131       0.126764187459146
    Density_pob      -0.000592491698843036    0.000630077384500953      -0.940347508762458       0.352173410798024


Number of observations: 50, Error degrees of freedom: 44
Root Mean Squared Error: 0.766
R-squared: 0.708,  Adjusted R-Squared: 0.674
F-statistic vs. constant model: 21.3, p-value = 9.21e-11
```

```
% Predicciones del modelo
predictions = predict(lm, BD);
```

También se realiza un gráfico de dispersión que muestre la relación entre las variables predichas con las reales.

```
% Gráfico de las respuestas reales vs. las predicciones
figure;
plot(BD.(response), predictions, '*');
hold on;
plot([min(BD.(response)), max(BD.(response))], [min(BD.(response)), max(BD.
(response))], 'r');
hold off;
title('Real vs. Predicha');
xlabel('Real');
ylabel('Predicha');

set(gcf, 'Position', get(0, 'Screensize'));
```



Purga de variables

Se va realizar otro modelo multivariado de regresión lineal pero únicamente considerando las variables que sean verdaderamente predictoras. Los supuestos de normalidad de confirmaron utilizando la prueba de Shapiro-Wilk.

Ahora resta realizar una prueba de homogenidad de varianzas y un test de correlación para verificar los supuestos del modelo de regresión lineal multivariado. En esta sección se realizará la prueba de Levene para la homogenidad de varianzas.

```
% Lista de variables predictoras
predictors = {'Population', 'Income', 'Murder', 'HS_Grad', 'Density_pob'};

leveneResults = table('Size', [length(predictors) 3], ...
    'VariableTypes', {'string', 'double', 'string'}, ...
    'VariableNames', {'Variable', 'P_Value', 'Homogeneity'});

for i = 1:length(predictors)
    % Verificar si la variable existe en la tabla
    if ~any(strcmp(BD.Properties.VariableNames, predictors{i}))
        fprintf('La variable %s no existe en la tabla BD.\n', predictors{i});
        continue;
    end

    % Verificar el tipo de datos
    if ~isa(BD.(predictors{i}), 'double')
```

```
        fprintf('La variable %s no es de tipo double.\n', predictors{i});
        continue;
    end

    % Limpiar los datos
    data = BD.(predictors{i});
    data = data(~isnan(data) & ~isinf(data));  % Eliminar NaNs e Infs

    % Verificar si los datos son constantes
    if range(data) == 0
        fprintf('La variable %s es constante y se omitirá de la prueba de
Levene.\n', predictors{i});
        continue;
    end

    % Dividir los datos en dos grupos
    medianValue = median(data);
    group1 = data(data <= medianValue);
    group2 = data(data > medianValue);

    % Prueba de Levene
    p = vartestn([group1; group2], [ones(size(group1)); 2*ones(size(group2))], ...
                 'TestType', 'LeveneAbsolute', 'Display', 'off');

    % Determinar si los datos pasan la verificación de supuestos
    if p > 0.05
        homogeneity = 'Pass';
    else
        homogeneity = 'Fail';
    end

    leveneResults(i, :) = {predictors{i}, p, homogeneity};
end

disp(leveneResults)
```

Variable	P_Value	Homogeneity
"Population"	9.17936038618155e-06	"Fail"
"Income"	0.484303246290379	"Pass"
"Murder"	0.768150794332981	"Pass"
"HS_Grad"	0.00256092404202033	"Fail"
"Density_pob"	4.7588443015055e-06	"Fail"

Interpretación de la prueba Levene

La prueba de Levene se realizó para verificar la homogeneidad de las varianzas entre dos grupos para cada variable predictora. Los grupos se formaron dividiendo los datos de cada variable en dos basándose en la mediana. Esta es una forma común de dividir los datos cuando no se dispone de una categorización natural.

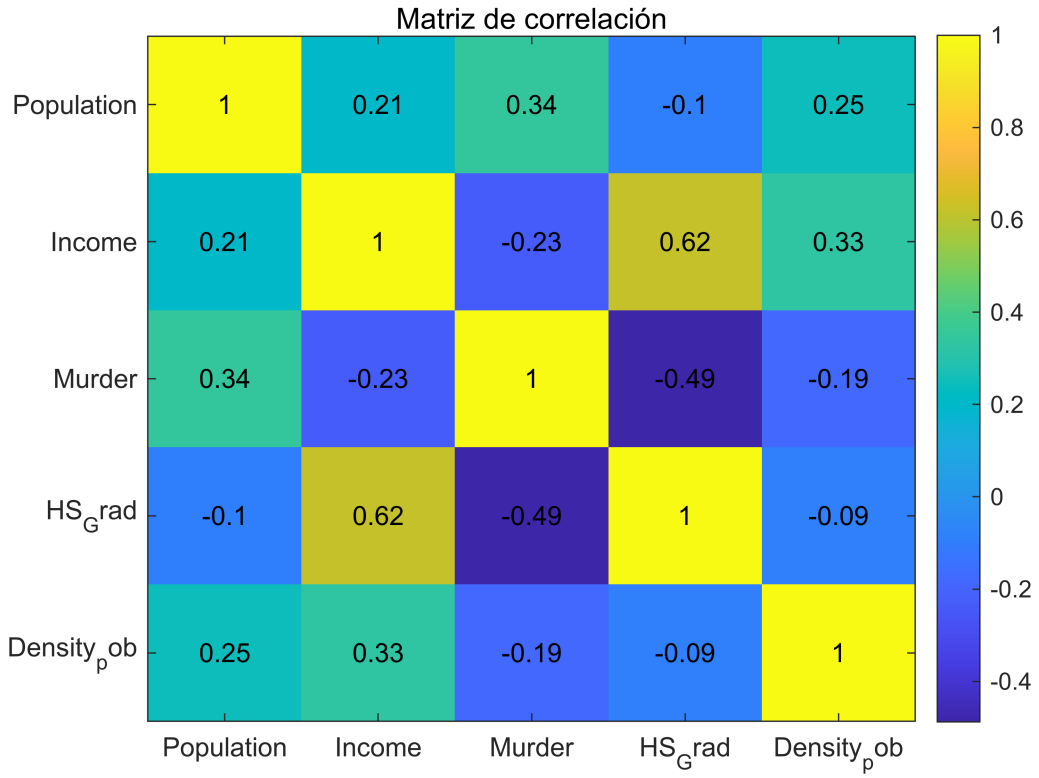
Los resultados de la prueba de Levene se interpretan de la siguiente manera: un valor p menor a 0.05 indica que se rechaza la hipótesis nula de igualdad de varianzas, lo que significa que las varianzas de los dos grupos son significativamente diferentes. Por otro lado, un valor p mayor a 0.05 indica que no hay suficiente evidencia para rechazar la hipótesis nula, lo que sugiere que las varianzas de los dos grupos son iguales.

En este caso, las variables `Population`, `HS_Grad` y `Density_pob` tienen valores p muy pequeños (menores a 0.05), lo que indica que las varianzas de los dos grupos para estas variables son significativamente diferentes. Esto sugiere que la distribución de estos datos está sesgada y tiene valores atípicos. Por otro lado, las variables `Income` y `Murder` tienen valores p mayores a 0.05, lo que indica que las varianzas de los dos grupos para estas variables no son significativamente diferentes, lo que sugiere que estos datos son homogéneos.

Matriz de correlación

Esta sección ayudará a verificar qué relación tienen las variables entre sí. Es un proceso importante para elegir finalmente cuáles serán las variables predictoras en el modelo de regresión final.

```
predictors = {'Population', 'Income', 'Murder', 'HS_Grad', 'Density_pob'};

% Matriz de correlación
corrMatrix = corr(BD{:, predictors});

% Mapa de calor de la matriz de correlación
figure;
imagesc(corrMatrix);
colorbar;
title('Matriz de correlación');
xticks(1:length(predictors));
yticks(1:length(predictors));
xticklabels(predictors);
yticklabels(predictors);

% Coeficientes de correlación al mapa de calor
for i = 1:length(predictors)
    for j = 1:length(predictors)
        text(j, i, num2str(round(corrMatrix(i, j), 2)), 'HorizontalAlignment',
'center');
    end
end
```



Modelo de regresión final

Se pueden seleccionar las variables predictoras para el modelo multivariado basándose en los resultados de la prueba de Levene y la matriz de correlación. En este caso, las variables `Income` y `Murder` pasaron la prueba de Levene, lo que indica que sus varianzas son homogéneas. Además, según la matriz de correlación, estas dos variables no están fuertemente correlacionadas entre sí, lo que es bueno porque evita el problema de la multicolinealidad.

Por lo tanto, se usa `Income` y `Murder` como las variables predictoras en el siguiente modelo multivariado.

```
% Lista de variables predictoras finales
predictors = {'Income', 'Murder'};

% Variable de salida
response = 'Life_Exp';

% Fórmula del modelo
formula = [response ' ~ ' strjoin(predictors, ' + ')];

% Modelo de regresión lineal multivariado
lm = fitlm(BD, formula);

% Métricas de evaluación del modelo
disp(lm)
```

```
Linear regression model:
    Life_Exp ~ 1 + Income + Murder

Estimated Coefficients:
```

	Estimate	SE	tStat	pValue
(Intercept)	71.225581467592	0.967395161335469	73.6261502169046	3.31942949898273e-50
Income	0.000370463882283167	0.000197297362787839	1.87769302665003	0.0666361919095996
Murder	-0.269759441125457	0.0328408492348385	-8.21414328224158	1.22248947421689e-10

```
Number of observations: 50, Error degrees of freedom: 47
Root Mean Squared Error: 0.826
R-squared: 0.637,  Adjusted R-Squared: 0.622
F-statistic vs. constant model: 41.2, p-value = 4.56e-11
```

Conclusiones finales

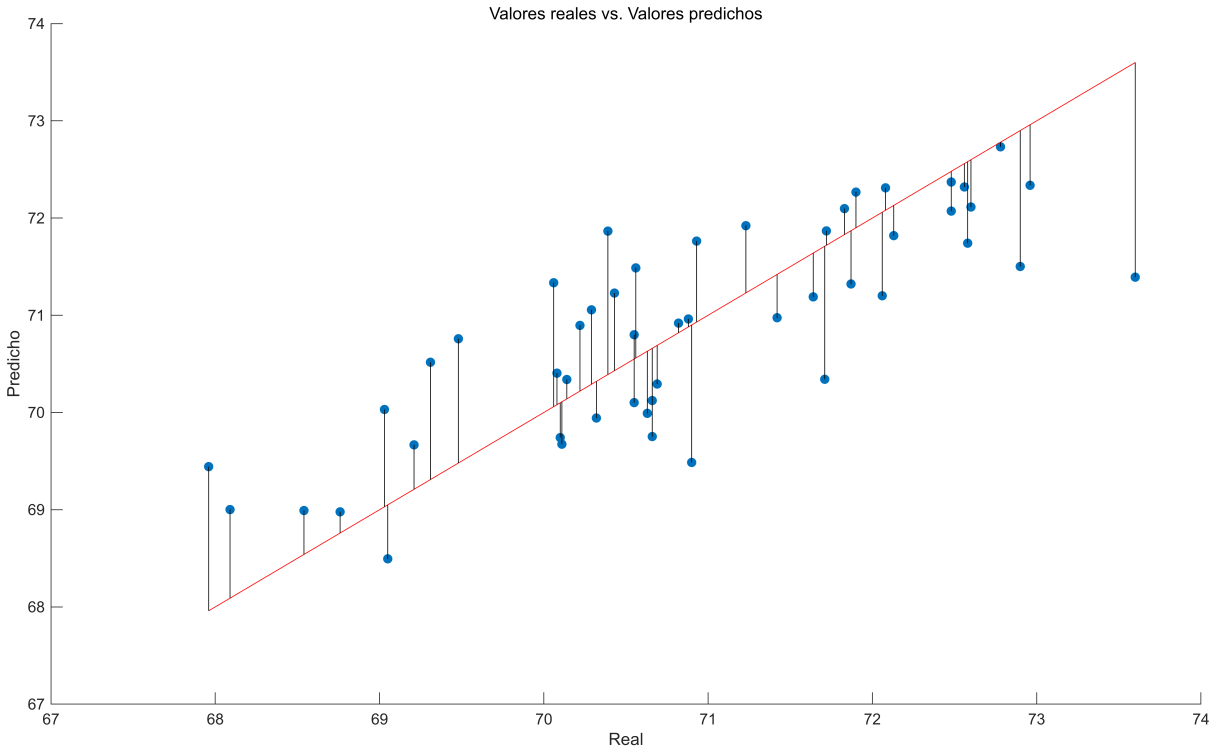
El modelo de regresión lineal multivariado que se ajustó tiene como variables predictoras `Income` (Ingresos) y `Murder` (Asesinatos), y como variable de respuesta `Life_Exp` (Esperanza de Vida). Observando las métricas de evaluación del modelo se puede concluir lo siguiente:

1. **Intercepto (`Intercept`)**: El intercepto del modelo es aproximadamente 71.23. Esto significa que, cuando los ingresos y la tasa de asesinatos son cero, la esperanza de vida media estimada es de alrededor de 71.23 años.
2. **Ingresos (`Income`)**: El coeficiente para los ingresos es aproximadamente 0.00037. Esto significa que, manteniendo constante la tasa de asesinatos, un aumento de una unidad en los ingresos se asocia con un aumento de 0.00037 unidades en la esperanza de vida. Sin embargo, el valor p para los ingresos es aproximadamente 0.067, que es mayor que el umbral comúnmente utilizado de 0.05. Esto sugiere que los ingresos no son significativamente predictivos de la esperanza de vida en este modelo.
3. **Asesinatos (`Murder`)**: El coeficiente para los asesinatos es aproximadamente -0.27. Esto significa que, manteniendo constante los ingresos, un aumento de una unidad en la tasa de asesinatos se asocia con una disminución de 0.27 unidades en la esperanza de vida. El valor p para los asesinatos es muy pequeño (aproximadamente 1.22e-10), lo que indica que los asesinatos son un predictor significativo de la esperanza de vida en este modelo.
4. **R-cuadrado y R-cuadrado ajustado**: El R-cuadrado del modelo es aproximadamente 0.637, lo que indica que alrededor del 63.7% de la variabilidad en la esperanza de vida se puede explicar por los ingresos y la tasa de asesinatos. El R-cuadrado ajustado, que tiene en cuenta el número de predictores en el modelo, es aproximadamente 0.622.
5. **Valor-p del modelo (`p-value`)**: el valor p del modelo es extremadamente pequeño (4.56e-11), lo que indica que el modelo es significativo.

A continuación podemos ver gráficamente los valores del modelo final Vs los valores reales, agregando el error entre dichos valores.

```
% Valores predichos
predicted = predict(lm, BD);
```

```
% Valores reales
real = BD.Life_Exp;

% Gráfico de dispersión
figure;
scatter(real, predicted, 'filled');  % Los puntos son grandes y están rellenos
hold on;

% Línea de identidad (valores predichos = valores reales)
plot([min(real), max(real)], [min(real), max(real)], 'r');

% Líneas de error (residuos)
for i = 1:length(real)
    line([real(i), real(i)], [real(i), predicted(i)], 'Color', 'k');
end

xlabel('Real');
ylabel('Predicho');
title('Valores reales vs. Valores predichos');

hold off;
set(gcf, 'Position', get(0, 'Screensize'));
```



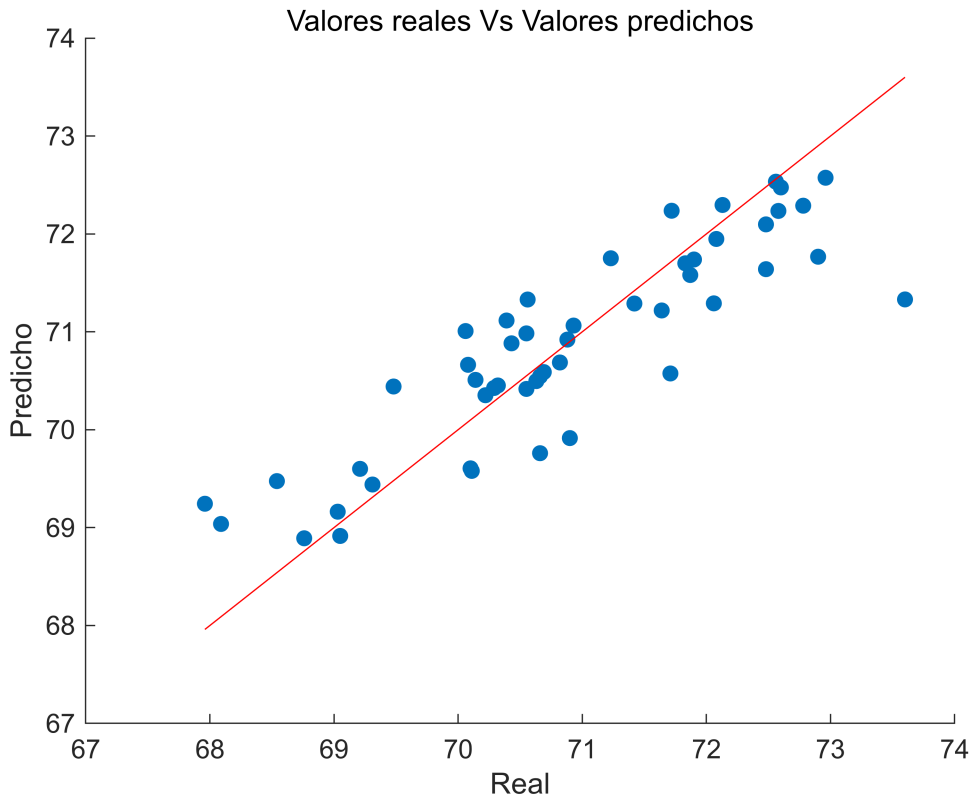
Aplicación de MatLab

Para este análisis, se utilizó la herramienta de **`MATLAB Regression Learner`** para entrenar varios modelos de regresión y evaluar su rendimiento. Entre todos los modelos probados, el que demostró tener el mejor rendimiento fue el **Medium Gaussian SVM**. Este modelo se destacó por tener el Error Absoluto Medio (MAE) más bajo, que fue de 0.71921, y el coeficiente de determinación (R cuadrado) más alto, que fue de 0.58.

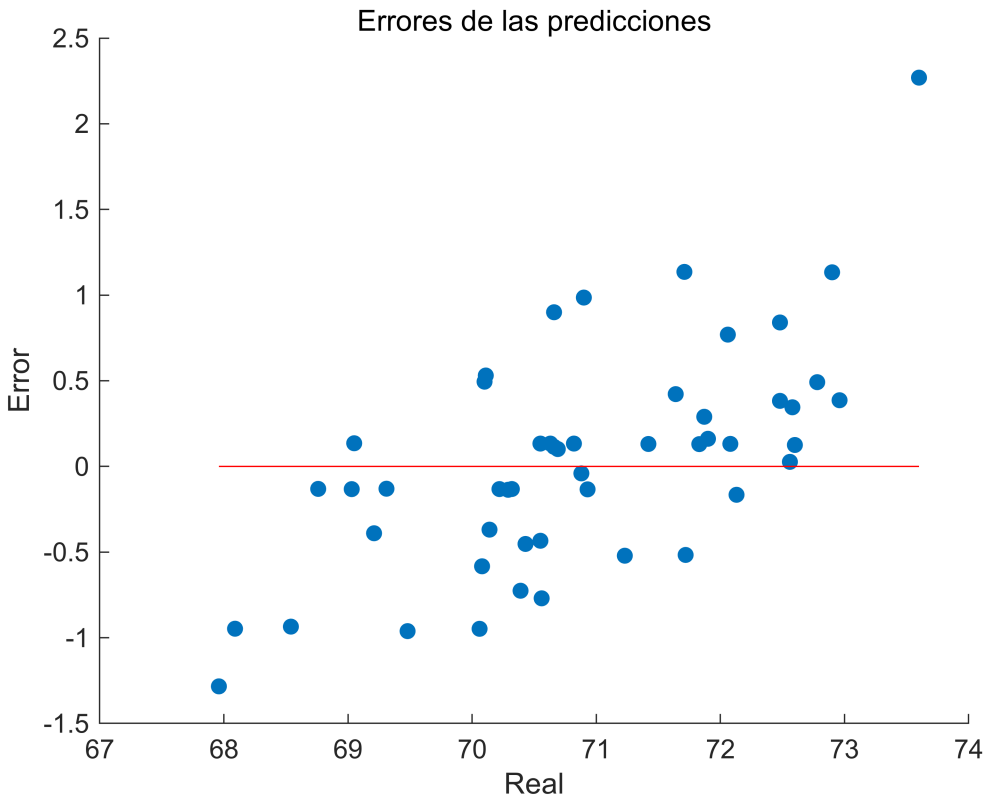
El MAE bajo indica que las predicciones del modelo tienen un error promedio pequeño, lo que sugiere que el modelo es preciso. Por otro lado, el alto valor de R cuadrado indica que una gran proporción de la variabilidad en la esperanza de vida puede ser explicada por las variables predictoras seleccionadas en el modelo.

Estos resultados sugieren que el modelo Medium Gaussian SVM es una opción sólida para predecir la esperanza de vida en este contexto, en comparación con los otros modelos probados.

```
% Valores predichos
predicted = trainedModel.predictFcn(BD);

% Valores reales
real = BD.Life_Exp;

% Grafico de dispersión de los valores reales Vs los valores predichos
figure;
scatter(real, predicted, 'filled');
hold on;

% Línea de identidad (valores predichos = valores reales)
plot([min(real), max(real)], [min(real), max(real)], 'r');
title('Valores reales Vs Valores predichos');
xlabel('Real');
ylabel('Predicho');
```




```
% Errores de las predicciones (residuos)
errors = real - predicted;

% Gráfico de los errores
figure;
scatter(real, errors, 'filled');  % Los puntos son grandes y están rellenos
hold on;

% Línea en y = 0 para comparar
plot([min(real), max(real)], [0, 0], 'r');
title('Errores de las predicciones');
xlabel('Real');
ylabel('Error');

hold off;
```


Errores de las predicciones
Error
Real
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
67
68
69
70
71
72
73
74