Regressão Linear

Análise de Dados em Informática

Licenciatura em Engenharia Informática

Instituto Superior de Engenharia do Porto

Ano letivo 2023/2024

- A Análise de Regressão é usada para explicar, ou modelar, a relação entre uma variável Y (aleatória) e as variáveis $X_1, \ldots, X_p$, $p \geq 1$ (não aleatórias).
- A variável Y diz-se dependente ou resposta.
- As variáveis $X_1, \ldots, X_p$, dizem-se independentes, preditoras ou explanatórias.
- Quando se tem apenas uma variável independente ($p = 1$) diz-se que a regressão é simples. Quando se tem mais que uma variável independente ($p > 1$) a regressão diz-se múltipla.
- A variável Y é uma variável contínua e as variáveis $X_1, \ldots, X_p$ podem ser contínuas, discretas ou categóricas.
- A escolha do modelo de regressão depende do conjunto de dados empíricos e deverá obedecer a alguns critérios estatísticos e práticos.

Modelo de Regressão Linear Simples

- Usa-se este modelo quando existe uma relação linear entre a variável dependente e a variável independente.

$$Y = \beta_0 + \beta_1 X + \epsilon$$

1. Os coeficientes β_0 (ordenada na origem) e β_1 (declive da recta) chamam-se coeficientes de regressão.
2. A variável ϵ representa as flutuações aleatórias causadas por erros nas medições dos dados ou por outros factores externos, e é designada por erro ou resíduo.
3. Para validar este modelo devemos verificar alguns pressupostos sobre a distribuição dos erros e independência dos valores observados da variável dependente.

[0] Na realidade, a definição de modelo linear é mais vasta. Por exemplo o modelo $Y = \beta_0 + \beta_1 X + \beta_2 X^2$ também é linear, mas o modelo $Y = \beta_0 + e^{\beta_1 X}$ não é linear. Aqui, o termo linear refere-se à expressão ser linear relativamente aos coeficientes do modelo e não à variável independente.

- Podemos escrever o modelo de regressão linear, para n observações, da forma,

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \quad i = 1, \ldots n.$$

- E o modelo deve satisfazer as seguintes condições suplementares:
  1. ϵ_i seguem uma distribuição normal e $E(\epsilon_i) = 0$, para todo o $i = 1, \ldots, n$ (equivalente a $E(y_i) = \beta_0 + \beta_1 x_i$).
  2. $Var(\epsilon_i) = \sigma^2$, para todo o $i = 1, \ldots, n$ (equivalente a $Var(y_i) = \sigma^2$)
  3. $Cov(\epsilon_i, \epsilon_j) = 0$, para todo o $i \neq j$ (equivalente a $Cov(y_i, y_j) = 0$)

Notas:

- A condição 1, implica que as variáveis y (e ϵ) são independentes, para todo o $i \in \{1, \ldots, n\}$ y_i apenas depende de x_i e que toda a restante variação de y_i é aleatória.
- A condição 2, afirma que a variância de ϵ não depende dos valores de x_i (é conhecida por homocedasticidade, variância homogénea ou variância constante).
- A condição 3 estabelece que as variáveis do resíduo (ou que as variáveis dependentes) são independentes.

- Seja uma amostra aleatória de n observações $y_1, y_2, \ldots, y_n$ e os respetivos valores $x_1, x_2, \ldots, x_n$ que os acompanham. As estimativas $\hat{\beta}_0$ e $\hat{\beta}_1$ são determinadas de modo a minimizar a norma dos erros (ou resíduos) $||\epsilon|| = \sqrt{\sum_{i=1}^{n} \epsilon_i^2}$, onde $\epsilon_i = y_i - (\beta_0 + \beta_1 x_1)$.
- Obtendo-se, deste modo, a Reta de Regressão

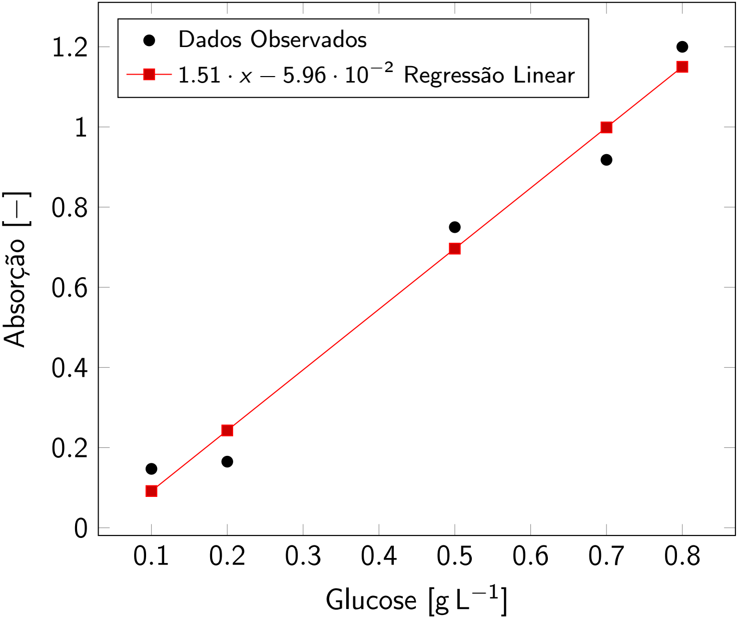
$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X.$$

onde,

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad (1)$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}. \quad (2)$$

Nota: Os três pressupostos não são necessários para o cálculo da reta de regressão. Contudo se se verificarem as três condições $\hat{\beta}_\mathbf{0}$ e $\hat{\beta}_\mathbf{1}$ são não enviesados e possuem a propriedade da variância mínima (entre todos os estimadores não enviesados).



Dados Observados
$1.51 \cdot x - 5.96 \cdot 10^{-2}$ Regressão Linear
1.2
1
0.8
0.6
0.4
0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Absorção [—]
Glucose [g L^{-1}]

- Para se verificar se o valor esperado da variável Y varia de forma linear com a variável X pode-se efetuar um teste de hipóteses

$$H_0: \ \beta_1 = 0 \ \ \text{(Y não varia linearmente com X)}$$
$$vs$$
$$H_1: \beta_1 \neq 0 \ \ \text{(Y varia linearmente com X)}$$

usando a estatística,

$$T_1 = \frac{\hat{\beta}_1 - \beta_1}{\sqrt{\frac{\hat{\sigma}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}} \sim T_{(n-2)}$$

- A estatística T_1 também é útil para construirmos intervalos de confiança para β_1.
- De forma análoga, usando a estatística $T_0 = \frac{\hat{\beta}_0 - \beta_0}{\sqrt{\frac{\hat{\sigma}^2}{n}}} \sim T_{(n-2)}$ determinamos intervalos de confiança e efetuamos testes para o coeficiente β_0.

- O coeficiente de determinação, R^2, é uma medida do poder explicativo do modelo utilizado. Dá a proporção da variável Y que é explicada em termos lineares pela variável independente X.

$$R^2 = \frac{\text{variação explicada}}{\text{variação total}} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

- Tem-se $0 \leq R^2 \leq 1$. Se $R^2 \cong 0$ o modelo não é adequado.
- $1 - R^2$ é a proporção da variação de Y que não é explicada pela variável X, resultante de fatores não incluídos no modelo.
- Alguns estatísticos preferem usar o coeficiente de determinação ajustado, R^2_{aj}, para modelos múltiplos, definindo para um modelo com p coeficientes

$$R^2_{\text{aj}} = 1 - \left(\frac{n-1}{n-p}\right)\left(1 - R^2\right)$$

porque a inclusão de variáveis com pouco poder explicativo aumenta o valor de R^2.

Exemplo:

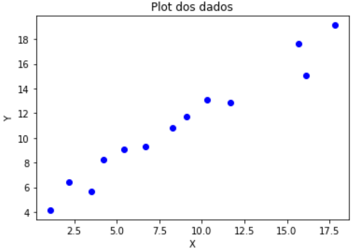
Considere o conjunto de dados guardados na *Data.Frame* dados, onde X é a variável independente e y é a variável dependente:

```
 import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
X=np.array([1.1,2.2,3.5,4.2,5.4,6.7,8.3,9.1,10.3,11.7,15.7,16.1,17.8])
n=len(X)
np.random.seed(55)
y=np.round(4+0.7*X+np.random.normal(1,1,n),2)
dados=pd.DataFrame({'X':X,'y':y})
print(dados)
```

```
       X       y
0    1.1    4.15
1    2.2    6.44
2    3.5    5.64
3    4.2    8.20
4    5.4    9.04
5    6.7    9.31
6    8.3   10.81
7    9.1   11.71
8   10.3   13.11
9   11.7   12.83
10  15.7   17.65
11  16.1   15.08
12  17.8   19.13
```

Plot dos dados:

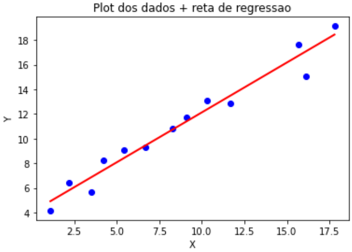
```
 # continuacao do codigo do slide anterior....
# plot dos pontos
plt.scatter(dados['X'], dados['y'], color='blue')
plt.title('Plot␣dos␣dados')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



Analisando os dados podemos observar que é apropriado usar uma regressão linear (os pontos estão localizados "ao longo de uma reta")

Regressão linear e plot com a reta de regressão:

```
Xc=sm.add_constant(dados['X'])
y=dados['y']
modelo=sm.OLS(y,Xc)
resultados=modelo.fit()
y_pred = resultados.predict(Xc)
plt.scatter(dados['X'], dados['y'], color='blue')
plt.plot(dados['X'], y_pred, color='red', linewidth=2, label='Regression Line')
plt.title('Plot dos dados + reta de regressao')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



Extracção da informação (excluindo a inferência estatística) dos resultados:

- **Resíduos:**

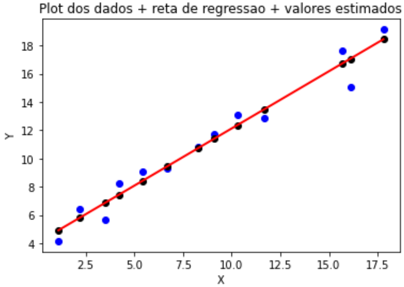
```
print('residuos:',resultados.resid)

residuos: 0    -0.765615
1      0.633621
2     -1.219100
3      0.774051
4      0.642308
5     -0.140413
6      0.063931
7      0.316102
8      0.744360
9     -0.669339
10     0.911519
11    -1.982395
12     0.690970
dtype: float64
```

- **Coeficientes da reta de regressão $\beta_0 + \beta_1 X$**

```
coeficientes = resultados.params
print("beta0:", coeficientes[0])
print("beta1:", coeficientes[1])
beta0: 4.024851066280948
beta1: 0.8097853357568929
```

Extracção dos valores estimados (ordenadas dos pontos pretos) e respectivo plot:

```
fitval=resultados.fittedvalues
plt.scatter(dados['X'], dados['y'], color='blue')
plt.plot(dados['X'], y_pred, color='red', linewidth=2)
plt.scatter(dados['X'], fitval, color='black')
plt.title('Plot␣dos␣dados␣+␣reta␣de␣regressao␣+␣valores␣estimados')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



Estimar valores:

- Estimar os valores de y para $X = 6.5$ e $X = 16.1$

```
Xp=np.array([6.5,16.1])
Xp=sm.add_constant(Xp)
yp=resultados.predict(Xp)
print('valores␣previstos␣para␣y␣do␣modelo␣X=6.5␣e␣X=16.1:' , yp)
valores previstos para y do modelo X=6.5 e X=16.1: [ 9.28845575 17.06239497]
```

- Extrair os coeficientes de determinação (R^2 e R^2_{ad}):

```
R2=resultados.rsquared
print('Coef␣de␣determinacao␣R2:',R2)
R2ajustado=resultados.rsquared_adj
print('Coef␣de␣determinacao␣R2adj:',R2ajustado)
Coef de determinacao R2: 0.960008192979036
Coef de determinacao R2adj: 0.9563725741589483
```

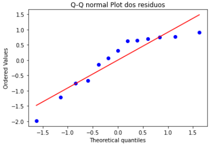
Ou seja, a variável X explica cerca de 96% da variabilidade da variável resposta y.

Antes de efetuarmos inferência estatística, é necessário verificarmos as condições sobre os resíduos:

(1) Os resíduos seguem uma distribuição normal com média zero:

- Processo gráfico (qq-normal-plot)

```
residuos = resultados.resid
import scipy.stats as stats
stats.probplot(residuos, dist="norm", plot=plt)
plt.title('Q-Q normal Plot dos residuos')
plt.show()
```



- Teste de Shapiro

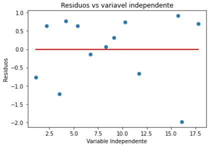
```
_, pvalue =  stats.shapiro(residuos)
print('shapiro - pavalue:',pvalue)
shapiro - pavalue: 0.05173667520284653
```

Verifica-se a condição de normalidade. O p-value é (ligeiramente) superior a $\alpha = 0.05$, logo podemos considerar que os resíduos verificam a condição de normalidade.

(2) A variância dos resíduos é constante (homocedasticidade).

- Método gráfico, plot dos resíduos *vs* valores da variável independente

```
plt.scatter(dados['X'], residuos)
plt.xlabel('Variable Independente')
plt.ylabel('Residuos')
plt.title('Residuos vs variavel independente')
plt.plot(dados['X'], 0*dados['X'], color='gray', linewidth=1)
plt.show()
```



Deve haver simetria relativamente à reta $y = 0$ e não deve existir tendência.

Podemos verificar que a condição de homocedasticidade não é verificada!

(3) Os resíduos são independentes.

- Efetua-se um teste de Durbin-Watson

```
from statsmodels.stats.stattools import durbin_watson
durbinWatson = durbin_watson(residuos)
print('Estatistica␣de␣Durbin-Watson:', durbinWatson)
if durbinWatson < 1.5:
        print('Sinais␣de␣autocorrela  o␣positiva', '\n')
        print('Condicao␣nao␣verificada')
elif durbinWatson > 2.5:
        print('Condicao␣nao␣verificada', '\n')
        print('Condicao␣nao␣verificada')
else:
        print('Sem␣autocorrela  o␣ou␣pequenos␣sinais', '\n')
        print('Condicao␣verificada')
Estatistica de Durbin-Watson: 3.076391305949389
Condicao nao verificada
```

Consideramos que os resíduos não são independentes

Conclusão: Não podemos efetuar inferência estatística. **No que se segue iremos assumir todas as condições são verificadas.**

Para analisarmos os resultados relativos dos parâmetros estimados pela regressão teremos de interpretar o output do sumário *resultados.summary()*.

1. **Dep.Variable: Variável dependente**
2. Coeficientes de ajuste do modelo:
   - R-squared: Mede quanto o modelo explica a variância da variável dependente
   - Adj. R-squared: Mede quanto o modelo explica a variância da variável dependente (adequado a modelos com mais do que uma variável independente)
3. Coeficientes de regressão
   - Intercept (const): valor estimado do termo intercept
   - Coeficientes (coef) valor estimado do coeficiente de cada variável independente
   - Erro standard (std err) desvio padrão do coeficiente estimado
   - t-value (t): Valor da estatística teste para testar se os coeficientes são significativamente diferentes de zero
   - P-value (P>|t|): A probabilidade de que o coeficiente não é diferente de zero
   - Intervalo de confiança (95% CI) Intervalo de confiança com nível de 95% para cada coeficiente
4. **Análise dos Resíduos**
   - Residuals: Resíduos
   - Erro quadrático médio (MSE)
   - Durbin-Watson Statistic: Teste para a independência dos resíduos
5. F-teste
   - T-statistic: Teste geral à significância do modelo
   - Prob(F-statistic): Probabilidade que o modelo tem de não ser significativo
6. AIC e BIC: Critérios de informação usados para a redução de modelos de regressão múltipla

- Encontrar intervalos de confiança para previsões, p. ex. : encontrar os intervalos de confiânça a 95% para os valores da variável y quando a variável X toma os valores: 2.6, 10.1 e 17.5

```
Xp2=np.array([2.6,10.1,17.5])
Xp2=sm.add_constant(Xp2)
yp2=resultados.predict(Xp2)
print('Valores␣previstos:',yp2)
prediction = resultados.get_prediction(Xp2)
conf_intervals = prediction.conf_int(alpha=0.05)
print('Intervalos␣de␣confianca␣a␣95\%',conf_intervals)
```

```
Valores previstos: [ 6.13029294 12.20368296 18.19609444]
Intervalos de confianca a 95% [[ 5.2527405   7.00784537]
 [11.60375764 12.80360827]
 [17.06399217 19.32819672]]
```

- A RLM permite estudar a relação entre uma variável dependente Y e um conjunto de variáveis independentes $X_1, X_2, \ldots X_p$, $(p > 1)$

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \epsilon$$

- Um modelo de RLM adequado deve satisfazer as seguintes condições:
  - Normalidade; Os erros (resíduos), ϵ têm uma distribuição normal $N(0, \sigma^2)$
  - Homocedasticidade; A variância é constante para todos os níveis das variáveis independentes.
  - Autocorrelação nula; Os erros são mutuamente independentes.
  - Multicolinearidade; As variáveis $X_1, \ldots, X_p$ devem ser linearmente independentes.
- A selecção das variáveis independentes $X_1, \ldots, X_p$ que entram no modelo de RLM devem ser relevantes.
- Deve-se usar o coeficiente de determinação ajustado, R^2_{ad}, para obter o poder explicativo do modelo de RLM.

- Dado um conjunto de n observações $\{Y_i,\ i = 1, \ldots, n\}$, $\{X_{i,j},\ i = 1, \ldots, n\ \wedge\ j = 1, \ldots, p\}$ tem-se,

$$Y_i = \beta_0 + \beta_1 X_{i,1} + \ldots + \beta_p X_{i,p} + \epsilon_i.$$

- Para se testar se a selecção das variáveis independentes é significativa efectua-se o TESTE GLOBAL

$$H_0 :\ \beta_0 = \beta_1 =, \ldots, = \beta_p = 0\ vs\ H_0 :\ \exists i\ ,\ \beta_i \neq 0.$$

  A relação global entre a variável Y e as variáveis $X_1, \ldots, X_p$ será significativa quando se rejeita H_0.

- Para verificarmos se existe diminuição na qualidade do modelo quando suprimimos, do modelo, uma variável X_k, $1 \leq k \leq p$, efectuamos um TESTE MARGINAL

$$H_0 :\ \beta_k = 0\ vs\ H_0 : \beta_k \neq 0.$$

  Se não rejeitarmos H_0 é porque existe diminuição na qualidade do modelo.

- O diagnóstico da Multicolinariedade pode ser feito usando vários métodos:
  - Analisando os valores próprios da matriz $C^T C$, onde C é a matriz das covariâncias (valores próprios pequenos, indiciam a presença de multicolinariedade).
  - Usando o factor de inflação de variância, VIF.
    Na prática, iremos considerar ausência de multicolinariedade quando VIF < 5.
- Exemplo do cálculo do VIF no Python: Supôr que o conjunto das variáveis independentes de um modelo de RLM é: x1, x2, x3, x4.

```
import pandas as pd
import numpy as np
from statsmodels.stats.outliers_influence import variance_inflation_factor
seed_value = 42
x1=np.random.normal(40,3,100)
x4=4*x1+77 # x4 e x1 est o correlacionadas
x2=np.random.uniform(0,20,100)
x3=np.random.normal(-25,1,100)
VarInd=pd.DataFrame({'x1':x1,'x2':x2,'x3':x3,'x4':x4})
# # # VIF dataframe
vif_res = pd.DataFrame()
vif_res["variaveis"] = VarInd.columns
vif_res["VIF"] = [variance_inflation_factor(VarInd.values, i)
                                     for i in range(len(VarInd.columns))]
print(vif_res)

   variaveis          VIF
0           x1  5849.466271
1           x2     1.044481
2           x3     1.078982
3           x4  9917.276727
```

- A presença de multicolinearidade, num modelo, implica que o cálculo dos coeficientes de regressão, $\hat{\beta}_i$, $i = 0, \ldots, p$ seja numericamente instável (pequenas variações nos dados provocam grandes variações dos $\hat{\beta}_i$).
- Uma variável *dummy* é uma variável categórica.
- Para incluirmos uma variável *dummy* ordinal num modelo atribuímos valores numéricos às categorias. Por exemplo se uma variável tomar valores no conjunto $\{\text{insuficiente},\ \text{suficiente},\ \text{bom}\}$, atribuímos os valores numéricos $\{\text{insuficiente} = 1,\ \text{suficiente} = 2,\ \text{mau} = 3\}$.
- Se existir uma variável nominal (não ordinal) com $k \geq 2$ categorias usamos $k - 1$ variáveis indicadoras $X_1, \ldots, X_{k-1}$, e codificamos

$$\begin{array}{ll} X_i = 1,\ \text{Se pertencer à i-ésima categoria} & \\ & i = 1, \ldots, k-1 \\ X_i = 0,\ \text{Caso contrário} & \end{array}$$

Não se inclui k variáveis para evitar a presença de colinearidade no modelo.
Modelos com variáveis Dummy chamam-se Modelos Lineares Generalizados.

Exemplo: Iremos usar um conjunto de dados (http://www.randomservices.org/random/data/Galton.txt) baseado no estudo da relação entre a altura dos filhos e a altura dos pais efectuado por Francis Galton em 1885. Além das variáveis "altura do pai" e da "altura da mãe" (altura medida em polegadas) iremos acrescentar a variável categórica "género do filho".

- Importação e tratamento dos dados dos dados;

```
import pandas as pd
Galton=pd.read_csv('Galton.txt',sep='\s+')
Galton['Gender']=Galton['Gender'].map({'M':1, 'F':0})
print(Galton.head(3))
  Family  Father  Mother  Gender  Height  Kids
0      1    78.5    67.0       1    73.2     4
1      1    78.5    67.0       0    69.2     4
2      1    78.5    67.0       0    69.0     4
```

Note que a variável Dummy "género do filho" foi codificada de forma a ser 0 (F) e 1 (M).

```
import statsmodels.api as sm
X=Galton[['Father','Mother','Gender']] # var. independentes
Xc=sm.add_constant(X)
y=Galton['Height']  # var. dependente
modelo=sm.OLS(y,Xc)
resultados=modelo.fit()
```

Informação extraída dos resultados:

```
p_values = resultados.pvalues
print("P-values␣para␣os␣coeficientes:")
print(p_values)
P-values para os coeficientes:
const       3.082284e-08
Father      6.525648e-40
Mother      1.701771e-23
Gender     5.786616e-178
```

- Os quatro p-values são pequenos logo, todos os coeficientes β_i, $i = 0, \ldots .3$ são significativamente diferentes de zero..

```
f_statistic_p_value = resultados.f_pvalue
print("P-value␣da␣estatistica-F:", f_statistic_p_value)
P-value da estatistica-F: 1.3288884053767563e-197
```

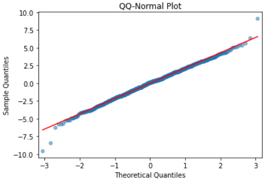
- O p-value da estatística-F é na prática 0 logo os valores de R^2 e de R^2_{aj} são estatisticamente significantes.

```
r_quadrado_ajustado = resultados.rsquared_adj
print("R-quadrado␣ajustado:", r_quadrado_ajustado)
R-quadrado ajustado: 0.6384661008338706
```

- O valor do coeficiente de determinação R^2_{aj} indica que o modelo explica 64% da variabilidade de y.

Normalidade dos resíduos:

```
import matplotlib.pyplot as plt
residuos = resultados.resid
sm.qqplot(residuos, line='s', markersize=5, alpha=0.5)
plt.title('QQ-Normal␣Plot')
plt.show()
```



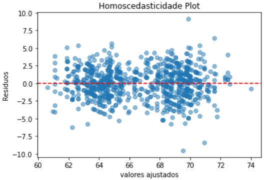
Teste de Shapiro:

```
from scipy.stats import shapiro
_, pvalue=shapiro (residuos)
print('p-value␣T-shapiro',pvalue)
p-value T-shapiro 0.013027232140302658
```

Tomando $\alpha = 0.05$ concluímos que a condição não é verificada. A análise do qq-normal plot leva a uma conclusão semelhante (os pontos extremos afastam-se da reta).
A figura da esquerda apresenta um padrão aleatório o que indica a independência e a homocedasticidade dos resíduos.

Homocedasticidade:

```
plt.scatter(resultados.fittedvalues, residuos, alpha=0.5)
plt.axhline(y=0, color='r', linestyle='--')
plt.title('Homoscedasticidade Plot')
plt.xlabel('valores ajustados')
plt.ylabel('Residuos')
plt.show()
```



Da análise do gráfico resulta que a condição de homocedacidade é verificada.

Independência dos resíduos:

```
from statsmodels.stats.stattools import durbin_watson
durbinWatson = durbin_watson(residuos)
print('valor da estat stica DW :',durbinWatson)
valor da estat stica DW : 1.5603949017849403
```

O valor da estatística DW está entre 1.5 e 2.5 logo a condição de independência é verificada.

Multicolinearidade:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif_res = pd.DataFrame()
vif_res["variaveis"] = X.columns
vif_res["VIF"]=[variance_inflation_factor(X.values , i)
                        for i in range(len(X.columns))]
print(vif_res)
  variaveis         VIF
0    Father  421.882892
1    Mother  421.600352
2    Gender    2.067652
```

O valor do factor de inflação de variância das variáveis *Father* e *Mother* é superior a 5 logo existe multicolinearidade.

Coeficientes de regressão e Intervalos de confiança:

```
coefs = resultados.params
print("Coeficientes:")
print(coefs)
Coeficientes:
const      15.344760
Father      0.405978
Mother      0.321495
Gender      5.225951


intervalos_conf95 = resultados.conf_int(alpha=0.05)
print("Intervalos␣de␣confianca:")
print(intervalos_conf95)
Intervalos de confianca:
               0          1
const   9.953516  20.736004
Father  0.348656   0.463300
Mother  0.260101   0.382889
Gender  4.943318   5.508584
```

Nota: Aqui os intervalos de confiança não são estatísticamente relevantes dados que os resíduos não satisfazem a condição de normalidade. Esta nota também é válida para os intervalos de confiança para as previsões.

Previsões com intervalo de confiança:

- Previsão da altura de um filho e de uma filha de um casal em que o pai mede $70''$ e a mãe $64''$.

```
predfilho = resultados.predict([[1,70,64,1]])
print("Previsao␣para␣a␣altura␣do␣filho:")
print(predfilho[0])
pred = resultados.get_prediction([[1,70,64,1]])
conf_intervals = pred.conf_int(alpha=0.05)
print('Intervalo␣de␣confianca␣a␣95␣para␣a␣altura␣do␣filho',conf_intervals[0])
Previsao para a altura do filho:
69.56486174593209
Intervalo de confianca a 95 para a altura do filho [69.36306691 69.76665658]
```

```
predfilha = resultados.predict([1,70,64,0])
print("Previs o␣para␣a␣altura␣da␣filha:")
print(predfilha[0])
pred = resultados.get_prediction([[1,70,64,0]])
conf_intervals2 = pred.conf_int(alpha=0.05)
print('Intervalo␣de␣confianca␣a␣95␣para␣a␣altura␣da␣filha',conf_intervals2[0])
Previs o para a altura da filha:
64.33891043539117
Intervalo de confianca a 95 para a altura da filha [64.13141387 64.546407 ]
```