Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Departamento de Estatística

Relatório Atividade 2
Análise de Regressão

Grupo: Andrielle Couto - 770295
Crystiane Souza - 760955
Douglas Nestlehner - 752728
Eric Sato - 729739

Setembro, 2021

Sumário

Capítulo 1

Introdução

Este trabalho tem como objetivo abordar uma aplicação do modelo de regressão linear simples, o qual é dado por:

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i,$$

em que Y_i é a variável resposta do i-ésimo ensaio, β_0 é o valor esperado da variável resposta quando a variável independente vale zero, β_1 é o efeito aditivo da variável independente no valor esperado da variável resposta, x_i é o valor da i-ésima observação da variável independente e ϵ_i é o erro aleatório do i-ésimo indivíduo.

Para isso, utilizamos o conjunto de dados com $n = 1000000$ (10^6) observações e $l = 199$ covariáveis que geramos na Atividade 1 através do software de linguagem de programação Python, e para essa Atividade, consideramos apenas as observações da covariável x_3. Em seguida, construímos intervalos de confiança para β_0 e β_1 considerando primeiro toda a amostra e depois apenas parte dela. Também, geramos a Tabela ANOVA a fim de conduzirmos um Teste de Hipóteses para testarmos $H_0 : \beta_1 = 0$. Depois disso, construímos um intervalo de confiança para $\mathbb{E}[Y_0]$ considerando apenas parte da amostra e dois específicos x_0, um próximo de $\bar{x}$ e um longe de $\bar{x}$. Logo depois, considerando toda a amostra novamente, calculamos intervalos de predição para uma nova resposta e para a média de m novas respostas. Por fim, conduzimos um Teste Linear Geral para testar $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$.

Nesse sentido, no Capítulo 2 apresentamos os resultados obtidos pelo o que foi citado anteriormente e no Apêndice está o código que foi utilizado para a realização dessa atividade.

Capítulo 2

Resultados

Neste capítulo, apresentamos os resultados obtidos.

Antes, vale lembrar o começo do que foi feito na Atividade 1. Primeiro, geramos um conjunto de dados com $n = 10^6$ observações e $l = 199$ covariáveis, ou seja, $x_{i1}, x_{i2}, ..., x_{i199}, i = 1, ..., 10^6$, de diferentes distribuições (Distribuições Normal, Uniforme, Exponencial, Beta e Gamma), obtendo a Figura 2.1 abaixo:

	x1	x2	x3	...	x197	x198	x199
0	-0.306653	-0.785878	0.153280	...	17.670058	3.487106	5.211530
1	0.537779	0.421365	-0.944903	...	8.682123	1.831579	0.048897
2	0.509093	-0.009545	0.148196	...	12.636960	2.508360	0.078390
3	-0.655666	0.321823	0.617224	...	11.467218	5.855939	8.373159
4	-0.599789	1.180766	0.795135	...	12.625468	2.353158	16.160959
...	...	...	...	...	...	...	...
999995	0.051659	-0.358460	-1.217553	...	8.019396	2.779270	6.715539
999996	-0.565747	0.541477	-0.074514	...	9.001532	2.589704	1.340682
999997	-0.106465	-0.775124	-1.384023	...	4.142704	4.136626	0.480302
999998	-0.723440	0.606755	0.307154	...	10.501333	1.019886	5.380319
999999	0.670020	-0.899993	-0.723158	...	10.428158	2.791444	2.464120

[1000000 rows x 199 columns]

Figura 2.1: Base de dados das covariáveis.

Após isso, tomamos $\beta_0 = 10$ e geramos os outros 199 , ou seja, $\beta_1, \beta_2, ..., \beta_{199}$ por meio de uma Distribuição Normal(0,2), obtendo a Figura 2.2:

```
[-3.17814026  0.82174032  0.64914096 -1.04403025  1.67897664  0.03355381
  0.36548631 -0.53472134 -4.40452794 -3.72488118  2.22921507 -2.56674148
 -0.2218001  -0.30273885 -1.14344852 -1.4108977  -2.24284415 -0.34993769
 -0.29480631 -1.45472404 -1.30804015  1.21252752 -1.27263772  0.01834462
 -4.07926342 -0.02983682  2.1492547   0.54072849  3.36359461 -3.43577417
 -0.61956996 -1.65899187 -1.48100326 -0.82904771  2.49938852  0.95582614
  1.64313732  0.72736667 -0.21834761  0.56758481  1.68079489 -0.60776302
 -5.41942787  2.64512003 -1.2904296   3.71243567 -1.83064893 -1.72123276
  3.07093598 -0.22050032  1.01880253 -2.93309822  2.46228704 -1.42007188
 -1.3490867  -0.67000704  3.78614346  0.58582059  0.40857637 -1.99265093
 -1.34064886  0.87039185 -4.48876381  1.28099526 -4.94923955 -0.71667124
  3.34536237  3.66304458  0.06446188  2.1599863   1.40821561 -0.70820471
  0.15991261  1.8310001  -0.94333349  1.18312055 -0.60613177  0.14803882
 -0.034801    0.9503359   2.71121137  0.51514558  1.18204787 -3.07282414
 -0.49542716  3.39707191  0.42507642  3.16467326 -2.16076843 -3.48066186
  1.73087323  2.3443886  -1.85691189 -0.9167518   1.6385639  -0.23156519
 -1.58362555  1.39692614 -0.07161874  1.82576022 -0.64032545  2.31952762
 -3.94114947  1.59773054  0.22193928 -2.41061498  1.42927481  3.11419923
  0.13967969  0.62888182 -0.61918391  2.10651299  0.7852024   1.8853202
 -0.99024665 -0.51201348  1.4677751  -0.89813128 -0.50220247  1.25297738
  2.23353315 -1.86371397 -1.56774697  0.54172092  0.17889165 -0.9968165
  0.0181375  -0.52523938 -3.55193009  0.70593984  0.50482978 -0.42993725
 -2.97575416 -1.06684421 -2.66857689  0.10760858  2.54887705  1.08836041
  1.52770122 -0.52997377  4.60465302 -4.14096941 -0.3081288   2.29917931
  2.22362534 -2.21540078  0.19454106  0.34293316 -0.56450044 -2.53945319
  1.84909421 -0.17591821  2.24440917  0.50252927 -0.28893614 -1.9561467
 -2.32435807  0.0770403   1.09228597  2.71488892  2.33796042  2.17238092
  0.99685207  2.52247565 -1.6330807  -1.3042549  -0.19458548 -0.45940525
 -1.31746202 -0.79962706 -1.16531094 -1.39055513 -2.59285303  0.2597009
 -1.32777504 -0.75517182  0.3447409   0.27240809 -0.05282772 -0.18319746
 -0.89783316  1.1382691  -0.73467581 -1.52279924  1.84284837 -0.43858816
 -1.20277471 -0.80515865 -0.42935529  0.84206293  0.1525661   1.83328626
  4.93738707  0.02644062  1.87304475 -0.75532492 -1.09302609 -1.35194961
  0.35378042]
```

Figura 2.2: Valores de $\beta_1, \beta_2, ..., \beta_{199}$.

Também, geramos valores para os erros $\epsilon_i, i = 1, ..., 10^6$ usando uma Distribuição Normal(0,1):

```
[-0.75061472  1.31635732  1.24614003 ... -0.26060049 -1.77081153
  1.64005059]
```

Figura 2.3: Valores de ϵ_i.

Com isso, através da fórmula

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_{199} x_{i199} + \epsilon_i$$

chegamos nos seguintes valores de Y_i:

```
[-128.77452511   60.28502227   -6.50087476 ...  -38.03158261  126.04605317
   -4.63388594]
```

Figura 2.4: Valores de Y_i.

2.1 Intervalos de Confiança para β_0 e β_1 considerando toda a amostra

Para determinar o intervalo de confiança para β_0 e β_1 considerando toda a nossa amostra, escolhemos apenas uma covariável com o objetivo de ajustar uma regressão linear simples.

A covariável escolhida dentre as 199 da Figura 2.1 foi x_3, a qual pode ser observada abaixo:

```
[[ 0.15327959]
 [-0.94490268]
 [ 0.14819629]
 ...
 [-1.38402254]
 [ 0.30715368]
 [-0.72315764]]
```

Figura 2.5: Valores das 10^6 observações da covariável x_3.

Após isso, geramos valores para os erros $\epsilon_i, i = 1, ..., 10^6$, usando uma Distribuição Normal(0,1) e usamos os betas gerados na atividade anterior, ou seja, $\beta_0 = 10$ e $\beta_1 = -3.1781402630116435$ (correspondente ao valor de β_3 na Atividade 1, já que a covariável que estamos utilizando é x_3), para encontrar os valores da variável resposta $Y_i, i = 1, ..., 10^6$, chegando nos seguintes valores:

```
[ 8.76224125 14.31939057 10.77515144 ... 14.13801727  7.25301099
 13.93834699]
```

Figura 2.6: Valores das 10^6 variáveis resposta $Y_i, i = 1, ..., 10^6$, geradas.

Desse modo, tendo obtido os valores das variáveis respostas $Y_i, i = 1, ..., 10^6$, e o valor da variável preditora x_3 estimamos, através da regressão linear simples, os valores $\hat{\beta}_0$ e $\hat{\beta}_1$, sendo eles:

$$\hat{\beta}_0 = 9.999146868848243 \quad \text{e} \quad \hat{\beta}_1 = -3.1762718676089308$$

Calculamos em seguida os valores ajustados $\hat{Y}_i$ utilizando a fórmula $\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i3}$, $i = 1, ..., 10^6$:

```
[ 9.51228922 13.00041466  9.52843517 ... 14.39517872  9.02354327
 12.29609212]
```

Figura 2.7: Valores de $\hat{Y}_i, i = 1, ..., 10^6$.

Depois, calculamos os resíduos e_i, sendo o i-ésimo resíduo a diferença entre o valor observado e o valor correspondente ajustado $\hat{Y}_i$, obtendo como resultado:

```
[-0.75004797  1.31897591  1.24671627 ... -0.25716145 -1.77053228
  1.64225486]
```

Figura 2.8: Valores observados dos resíduos $e_i, i = 1, ..., 10^6$.

Com os valores anteriores, calculamos a soma de quadrados dos resíduos SSE e o quadrado médio dos resíduos MSE cujas fórmulas são:

$$SSE = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \quad \text{e} \quad MSE = \frac{SSE}{n-2}$$

Sabendo que, no caso da nossa atividade, $n = 10^6$, obtivemos assim $SSE = 998361.8862771506$ e $MSE = 0.9983638830049166$.

Os estimadores da variância de $\hat{\beta}_0$ e $\hat{\beta}_1$ foram obtidos através das fórmulas:

$$\hat{V}[\hat{\beta}_0] = MSE \cdot \left(\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right)$$

$$\hat{V}[\hat{\beta}_1] = \frac{MSE}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

e seus resultados foram, respectivamente, $9.98364391 \cdot 10^{-07}$ e $1.34002636 \cdot 10^{-06}$.

Desse modo, fomos capazes de determinar os intervalos de confiança para β_0 e β_1, utilizando uma confiança de 95%. O IC de β_0 é:

$$\begin{aligned} IC(\beta_0; 95\%) &= \left[\hat{\beta}_0 - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_0]} \; ; \; \hat{\beta}_0 + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_0]} \right] \\ &= \Big[9.99718850601654 \; ; \; 10.001105231679945 \Big] \end{aligned}$$

Analisando o resultado acima, temos 95% de confiança de que o intervalo [9.99718850601654; 10.001105231679945] contém o verdadeiro valor de β_0. Como sabemos que $\beta_0 = 10$, então concluímos que esse intervalo realmente contém o verdadeiro valor do intercepto.

Junto disso, o IC para β_1 é:

$$IC(\beta_1; 95\%) = \left[\hat{\beta}_1 - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_1]} \; ; \; \hat{\beta}_1 + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_1]} \right]$$
$$= \Big[-3.178540715010965 \; ; \; -3.1740030202068965 \Big]$$

Agora, temos 95% de confiança de que o intervalo $[-3.178540715010965; -3.1740030202068965]$ contém o verdadeiro valor de β_1. Como sabemos que $\beta_1 = -3.1781402630116435$, então concluímos que esse intervalo realmente contém o verdadeiro valor do slope.

2.2 Intervalos de Confiança para β_0 e β_1 considerando apenas parte da amostra

Nessa seção, faremos os mesmos passos da anterior, porém considerando agora uma amostra aleatória de tamanho 4800 retirada da mesma covariável x_3, a qual é mostrada abaixo:

```
[[-0.02464019]
 [-0.06923491]
 [ 1.09485779]
 ...
 [ 1.84945334]
 [-1.04604505]
 [-1.67696782]]
```

Figura 2.9: Valores das 4800 observações da covariável x_3.

Após isso, geramos valores para os erros $\epsilon_i, i = 1, ..., 4800$, usando uma Distribuição Normal(0,1) e usamos novamente $\beta_0 = 10$ e $\beta_1 = -3.1781402630116435$ para encontrar novos valores da variável resposta $Y_i, i = 1, ..., 4800$, chegando em:

```
[10.37115253  7.9589889   7.82161447 ...  3.9869818  12.57241937
 16.43704434]
```

Figura 2.10: Valores das 4800 variáveis resposta $Y_i, i = 1, ..., 4800$, geradas.

Desse modo, estimamos, através da regressão linear simples, os valores $\hat{\beta}_0$ e $\hat{\beta}_1$, sendo eles:

$$\hat{\beta}_0 = 10.002770288849836 \quad \text{e} \quad \hat{\beta}_1 = -3.1545368314263604$$

Calculamos em seguida os valores ajustados $\hat{Y}_i$:

```
[10.08049867 10.22117436  6.54900106 ...  4.1686016  13.30255791
 15.29282705]
```

Figura 2.11: Valores de $\hat{Y}_i, i = 1, ..., 4800$.

Depois, calculamos os resíduos e_i, sendo o i-ésimo resíduo a diferença entre o valor observado e o valor correspondente ajustado $\hat{Y}_i$, obtendo como resultado:

```
[ 0.29065386 -2.26218547  1.27261341 ... -0.1816198  -0.73013855
  1.14421729]
```

Figura 2.12: Valores observados dos resíduos $e_i, i = 1, ..., 4800$.

Com os valores anteriores, calculamos a soma de quadrados dos resíduos SSE e o quadrado médio dos resíduos MSE, obtendo $SSE = 4763.775615994951$ e $MSE = 0.992866947893904$. Além disso, os estimadores da variância de $\hat{\beta}_0$ e $\hat{\beta}_1$ têm como resultados, respectivamente, 0.00027615 e 0.00020691.

Desse modo, fomos capazes de determinar os intervalos de confiança para β_0 e β_1, utilizando uma confiança de 95%. O IC de β_0 é:

$$\begin{aligned} IC(\beta_0; 95\%) &= \left[\hat{\beta}_0 - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_0]} \; ; \; \hat{\beta}_0 + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_0]} \right] \\ &= \Big[9.974570259587507 \; ; \; 10.030970318112166 \Big] \end{aligned}$$

Analisando o resultado acima, temos 95% de confiança de que o intervalo [9.974570259587507; 10.030970318112166] contém o verdadeiro valor de β_0. Como sabemos que $\beta_0 = 10$, então concluímos que esse intervalo realmente contém o verdadeiro valor do intercepto.

Junto disso, o IC para β_1 é:

$$\begin{aligned} IC(\beta_1; 95\%) &= \left[\hat{\beta}_1 - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_1]} \; ; \; \hat{\beta}_1 + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{\beta}_1]} \right] \\ &= \Big[-3.187115395101897 \; ; \; -3.121958267750824 \Big] \end{aligned}$$

Agora, temos 95% de confiança de que o intervalo $[-3.187115395101897; -3.121958267750824]$ contém o verdadeiro valor de β_1. Como sabemos que $\beta_1 = -3.1781402630116435$, então concluímos que esse intervalo realmente contém o verdadeiro valor do slope.

Comparando essa seção com a anterior, ou seja, a construção dos intervalos de confiança para β_0 e β_1 considerando toda a amostra e depois uma parte dela de tamanho 4800, percebemos que quanto maior o tamanho amostral, menor é a amplitude dos intervalos, isto é, maior é a precisão deles.

2.3 Tabela ANOVA e Teste de Hipóteses

Nessa seção, continuaremos a considerar a amostra retirada anteriormente de tamanho 4800.

A fim de testarmos $H_0 : \beta_1 = 0$, ou seja, se há ou não uma associação linear entre x_3 e a variável resposta, geramos a Tabela ANOVA abaixo:

	df	sum_sq	mean_sq	F	PR(>F)
x	1.0	35777.799298	35777.799298	36034.837673	0.0
Residual	4798.0	4763.775616	0.992867	NaN	NaN

Figura 2.13: Tabela ANOVA.

Observando a Figura 2.13 acima e considerando uma significância de 1%, percebemos que o valor-p=0 $< \alpha = 0.01$, ou seja, rejeitamos $H_0 : \beta_1 = 0$. Desse modo, com valor-p=0, há evidências de que há uma associação linear entre x_3 e a variável resposta.

2.4 Intervalo de Confiança para $\mathbb{E}[Y_0]$

Nessa seção, assim como na anterior, continuaremos a considerar a amostra retirada anteriormente de tamanho 4800 e utilizando o mesmo ajuste e estimação dos parâmetros feitos na Seção 2.2 (que considerava apenas uma amostra de tamanho 4800).

Faremos agora o intervalo de confiança para $\mathbb{E}[Y_0]$, ou seja, para o valor predito da variável resposta, considerando dois específicos x_0, um próximo de $\bar{x}$ e um longe de $\bar{x}$. Como temos $\bar{x} = -0.015207517319944928$, consideramos $x_0^{\text{próximo}} = -0.014999$ e $x_0^{\text{longe}} = 1.555555$. Através dos valores apresentados agora, obtivemos as variáveis resposta ajustadas: $\hat{Y}_0^{\text{próximo}} = 10.04977288763809$ e $\hat{Y}_0^{\text{longe}} = 5.0958882475661325$.

Após isso, calculamos os estimadores da variância de $\hat{Y}_0^{\text{próximo}}$ e $\hat{Y}_0^{\text{longe}}$ através da fórmula:

$$\hat{V}[\hat{Y}_0] = MSE \cdot \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right)$$

chegando em $\hat{V}[\hat{Y}_0^{\text{próximo}}] = 0.00020684730692609572$ e $\hat{V}[\hat{Y}_0^{\text{longe}}] = 0.0008881486777383533$.

Na sequência, construímos os intervalos de confiança para ambos os $\mathbb{E}[Y_0]$, considerando 95% de confiança. Assim, o IC para $\mathbb{E}[Y_0^{\text{próximo}}]$ é:

$$\begin{aligned} IC(\mathbb{E}[Y_0^{\text{próximo}}]; 95\%) &= \left[\hat{Y}_0^{\text{próximo}} - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{Y}_0^{\text{próximo}}]} \; ; \; \hat{Y}_0^{\text{próximo}} + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{Y}_0^{\text{próximo}}]} \right] \\ &= \Big[10.021577209052666 \; ; \; 10.077968566223515 \Big] \end{aligned}$$

Analisando o resultado acima, temos 95% de confiança de que o intervalo [10.021577209052666; 10.077968566223515] contém o verdadeiro valor de $\mathbb{E}[Y_0^{\text{próximo}}]$. Como sabemos que $\mathbb{E}[Y_0^{\text{próximo}}] = 10.047354289918873$, então concluímos que esse intervalo realmente contém o verdadeiro valor de $\mathbb{E}[Y_0^{\text{próximo}}]$.

Junto disso, o IC para $\mathbb{E}[Y_0^{\text{longe}}]$ é:

$$\begin{aligned} IC(\mathbb{E}[Y_0^{\text{longe}}]; 95\%) &= \left[\hat{Y}_0^{\text{longe}} - t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{Y}_0^{\text{longe}}]} \; ; \; \hat{Y}_0^{\text{longe}} + t_{n-2;\frac{\alpha}{2}} \sqrt{\hat{V}[\hat{Y}_0^{\text{longe}}]} \right] \\ &= \Big[5.037463014632696 \; ; \; 5.154313480499569 \Big] \end{aligned}$$

Agora, temos 95% de confiança de que o intervalo [5.037463014632696; 5.154313480499569] contém o verdadeiro valor de $\mathbb{E}[Y_0^{\text{longe}}]$. Como sabemos que $\mathbb{E}[Y_0^{\text{longe}}] = 5.056402820885388$, então concluímos que esse intervalo realmente contém o verdadeiro valor de $\mathbb{E}[Y_0^{\text{longe}}]$.

Comparando esses dois intervalos, ou seja, a construção dos intervalos de confiança para $E[Y_0^{\text{próximo}}]$ e $\mathbb{E}[Y_0^{\text{longe}}]$, percebemos que quanto mais próximo for x_0 de $\bar{x}$, menor é a amplitude do intervalo, isto é, maior é a precisão dele. Concluímos isso devido à variabilidade de $\hat{Y}_0^{\text{próximo}}$ ser menor do que a de $\hat{Y}_0^{\text{longe}}$.

2.5 Intervalo de Predição para uma nova resposta

Nessa seção, voltaremos a considerar toda a amostra.

A fim de construirmos o intervalo de predição para uma nova resposta, geramos uma nova observação da covariável x_3, e denotamos essa observação como x_{novo}, sendo $x_{\text{novo}} = -0.08112442165992582$. Através desse valor e utilizando o mesmo ajuste e estimação dos parâmetros feitos na Seção 2.1 (que considerava toda a amostra), obtivemos Y_{novo} e $\hat{Y}_{\text{novo}}$, os quais são, respectivamente, 10.672226800235403 e 10.25682008714271.

Ainda, temos que o intervalo de predição com uma confiança de 95% para Y_{novo} é dado por:

$$IP(Y_{\text{novo}}; 95\%) = \left[\hat{Y}_{\text{novo}} - t_{n-2;\frac{\alpha}{2}} \sqrt{MSE \cdot \left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\right)} \; ; \; \hat{Y}_{\text{novo}} + t_{n-2;\frac{\alpha}{2}} \sqrt{MSE \cdot \left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\right)} \right]$$

Desse modo, temos:

$$IP(Y_{\text{novo}}; 95\%) = \Big[8.298456766118457 \; ; \; 12.215183408166963 \Big]$$

Analisando o resultado acima, temos 95% de confiança de que o intervalo [8.298456766118457; 12.215183408166963] contém o verdadeiro valor de Y_{novo}. Como sabemos que $Y_{\text{novo}} = 10.672226800235403$, então concluímos que esse intervalo realmente contém o verdadeiro valor de Y_{novo}.

2.6 Intervalo de Predição para a média de m novas respostas

Nessa seção, também consideraremos toda a amostra.

Com o objetivo de construirmos agora o intervalo de predição para a média de m novas respostas, consideramos $m = 150$ e o mesmo $x_{\text{novo}} = -0.08112442165992582$ considerado na seção anterior, e calculamos a média dessas m respostas, encontrando como valor $\bar{Y}_{\text{novo}} = 10.209972327215253$.

Ainda, temos que o intervalo de predição com uma confiança de 95% para m novas respostas é dado por:

$$IP(\bar{Y}_{\text{novo}}; 95\%) = \left[\hat{Y}_{\text{novo}} - t_{n-2;\frac{\alpha}{2}} \sqrt{MSE \cdot \left(\frac{1}{m} + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \right)} \; ; \; \hat{Y}_{\text{novo}} + t_{n-2;\frac{\alpha}{2}} \sqrt{MSE \cdot \left(\frac{1}{m} + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \right)} \right]$$

Desse modo, temos:

$$IP(\bar{Y}_{\text{novo}}; 95\%) = \Big[10.096908375859297 \; ; \; 10.416731798426124 \Big]$$

Portanto, temos 95% de confiança de que o intervalo [10.096908375859297; 10.416731798426124] contém o verdadeiro valor de $\bar{Y}_{\text{novo}}$. Como sabemos que $\bar{Y}_{\text{novo}} = 10.209972327215253$, então concluímos que esse intervalo realmente contém o verdadeiro valor de $\bar{Y}_{\text{novo}}$.

2.7 Teste Linear Geral

Para a realização do teste linear geral e considerando os dados da Figura 2.1, escolhemos 8 covariáveis, sendo elas: x_{60}, x_{79}, x_{120}, x_{127}, x_{145}, x_{160}, x_{175}, x_{194}, para ajustar o modelo de regressão linear múltipla considerando uma amostra escolhida aleatoriamente de tamanho 800. Além disso, dentre os β's que acompanham essas covariáveis, três são próximos de zero e os outros cinco são mais distantes de zero.

Após escolher as covariáveis e retirar a amostra, para construir o teste linear geral é necessário dar o primeiro passo que é ajustar o modelo completo. Nesse sentido, na Figura 2.14 temos o conjunto de dados das covariáveis que foram usadas para ajustar o modelo completo.

	x79	x127	x194	...	x160	x175
0	0.106861	0.818931	2.878963	...	0.317880	6.879663
1	0.787219	0.739447	7.864554	...	1.590331	4.839365
2	0.872126	0.593939	3.947550	...	0.287297	4.822773
3	0.862474	0.613155	12.531605	...	0.718214	1.801654
4	0.257793	0.822022	4.423280	...	0.748059	1.708784
..	...	...	...	...	...	...
795	0.352054	0.872846	12.265217	...	0.496957	12.478482
796	0.361209	0.797508	3.419382	...	0.385098	1.996428
797	0.680479	0.302043	3.224519	...	0.126229	8.378537
798	0.907782	0.679589	8.315152	...	0.909111	3.418992
799	0.817525	0.658900	5.165363	...	0.019627	9.910728

Figura 2.14: Base de dados das covariáveis para o modelo completo.

Nesse contexto, denotando $x_1 = x_{79},\ x_2 = x_{127},\ x_3 = x_{194},\ x_4 = x_{60},\ x_5 = x_{120},\ x_6 = x_{145},\ x_7 = x_{160}$ e $x_8 = x_{175}$, e gerando um erro aleatório usando uma Distribição Normal(0,1) temos o seguinte modelo de regressão:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + \beta_5 x_{i5} + \beta_6 x_{i6} + \beta_7 x_{i7} + \beta_8 x_{i8} + \epsilon_i,\ \text{i=1,...,800}$$

em que:

$$\beta_0 = 10\ ;\ \beta_1 = 0.018137500871328753\ ;\ \beta_2 = 0.018344615457445486\ ;$$

$$\beta_3 = 0.02644061506466454\ ;\ \beta_4 = -1.9926509283527098\ ;$$

$$\beta_5 = 1.2529773794892074\ ;\ \beta_6 = 2.223625336049015\ ;$$

$$\beta_7 = 2.7148889203997544\ ;\ \beta_8 = -1.3277750401412078.$$

e através dos quais podemos percebemos que os três β's próximos de zero são β_1, β_2 e β_3.

Sendo assim, ao realizar o ajuste do modelo completo através da regressão linear múltipla, temos os valores de Y_i representados na Figura 2.15.

```
[[ 2.62969791e+00]
 [ 8.52342699e+00]
 [ 8.00262183e+00]
 ...
 [ 1.27833355e+00]
 [ 8.72770633e+00]
 [-1.72173784e+00]]
```

Figura 2.15: Valores de $Y_i, i = 1, ..., 800$.

Além disso, temos que $\hat{\beta}_0 = 9.891565160052185$ e que os valores de $\hat{\beta}_1$ até $\hat{\beta}_8$ são os apresentados na Figura 2.16:

```
[-0.02838375  0.30001007  0.01047673 -1.9567567   1.06379156  2.1916815
  2.72519925 -1.31080136]
```

Figura 2.16: Valores de $\hat{\beta}_1, \hat{\beta}_2, \cdots, \hat{\beta}_8$.

Desse modo,

$$\begin{aligned}\hat{Y}_i &= \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + \hat{\beta}_3 x_{i3} + \hat{\beta}_4 x_{i4} + \hat{\beta}_5 x_{i5} + \hat{\beta}_6 x_{i6} + \hat{\beta}_7 x_{i7} + \hat{\beta}_8 x_{i8} \\ &= 9.891565160052185 - 0.02838375 x_{i1} + 0.30001007 x_{i2} + 0.01047673 x_{i3} - 1.9567567 x_{i4} \\ &+ 1.06379156 x_{i5} + 2.1916815 x_{i6} + 2.72519925 x_{i7} - 1.31080136 x_{i8}, \text{ i=1,...,800}\end{aligned}$$

Portanto, após isso, calculamos o SSE (error sum of squares) desse modelo, obtendo:

$$\begin{aligned}SSE_F &= \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \\ &= \sum_{i=1}^{n} e_i^2 \\ &= 743.0523238039088.\end{aligned}$$

Para continuar o teste linear geral, ajustamos o modelo reduzido. Nesse caso, a fim de testarmos $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$, das 8 covariáveis que tínhamos, passamos a considerar apenas as 5 que eram mais distantes de zero. Na Figura 2.17 temos o conjunto de dados das covariáveis que foram utilizadas para ajustar o modelo reduzido.

	x60	x120	x145	x160	x175
0	0.405620	0.825170	0.532528	0.317880	6.879663
1	0.425470	0.019710	0.508636	1.590331	4.839365
2	0.179142	0.772082	0.531151	0.287297	4.822773
3	0.562808	0.423557	0.458763	0.718214	1.801654
4	0.876407	0.831110	0.697260	0.748059	1.708784
..	...	...	...	...	...
795	0.102798	0.235132	0.706814	0.496957	12.478482
796	0.804806	0.416398	0.552890	0.385098	1.996428
797	0.370474	0.486421	0.708183	0.126229	8.378537
798	0.931451	0.740708	0.440223	0.909111	3.418992
799	0.303573	0.423201	0.842520	0.019627	9.910728

Figura 2.17: Base de dados das covariáveis para o modelo reduzido.

A partir disso, o modelo reduzido é,

$$Y_i = \beta_0 + \beta_4 x_{i4} + \beta_5 x_{i5} + \beta_6 x_{i6} + \beta_7 x_{i7} + \beta_8 x_{i8} + \epsilon_i \text{ , i = 1,...,800}$$

Com isso, após realizar o ajuste do modelo reduzido através da regressão linear múltipla, temos os valores de Y_i representados na Figura 2.18:

```
[[ 3.30623679e+00]
 [ 8.53766985e+00]
 [ 5.19794417e+00]
 ...
 [ 1.06534564e+00]
 [ 7.17271831e+00]
 [-1.83648375e+00]]
```

Figura 2.18: Valores de $Y_i, i = 1, ..., 800$.

Ademais, temos os valores de $\hat{\beta}_0 = 10.110555423040278$ e os valores de $\hat{\beta}_3$ até $\hat{\beta}_8$ apresentados na Figura 2.19 abaixo.

```
[-2.02501787  1.41503217  1.98297221  2.72350746 -1.33598369]
```

Figura 2.19: Valores ajustados $\hat{\beta}_3, \hat{\beta}_4, \cdots, \hat{\beta}_8$.

Desse modo,

$$\begin{aligned}\hat{Y}_i &= \hat{\beta}_0 + \hat{\beta}_4 x_{i4} + \hat{\beta}_5 x_{i5} + \hat{\beta}_6 x_{i6} + \hat{\beta}_7 x_{i7} + \hat{\beta}_8 x_{i8} \\ &= 10.110555423040278 - 2.02501787 x_{i4} + 1.41503217 x_{i5} + 1.98297221 x_{i6} \\ &+ 2.72350746 x_{i7} - 1.33598369 x_{i8}, \text{ i=1,...,800}\end{aligned}$$

Assim, após isso, vamos calcular novamente o SSE (error sum of squares), mas agora para o modelo reduzido. Logo, temos que,

$$\begin{aligned}SSE_R &= \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \\ &= \sum_{i=1}^{n} \epsilon_i^2 \\ &= 796.3711652776251.\end{aligned}$$

Depois de ambos os modelos serem ajustados, construímos uma estatística teste, a qual foi usada para comparar a soma de quadrados dos erros dos modelo completo e do modelo reduzido, ou seja, comparar SSE_F e SSE_R.

Com isso, devemos considerar as seguintes hipóteses:

$$\begin{cases} H_0 : \beta_1 = \beta_2 = \beta_3 = 0; \\ H_1 : \beta_i \neq 0, \text{para pelo menos um i} = 1, 2, 3. \end{cases}$$

Denotando gl_F como os graus de liberdade do modelo completo e gl_R como os graus de liberdade do modelo reduzido, temos que, sob H_0, a estatística teste é dada por:

$$F_0 = \frac{\frac{SSE_R - SSE_F}{gl_R - gl_F}}{\frac{SSE_F}{gl_F}} \sim F_{gl_R - gl_F, gl_F}$$

Já sabemos quem são SSE_F e SSE_R, e sabemos também que, como temos 800 observações, $gl_F = 800 - 9 = 791$ e $gl_R = 800 - 6 = 794$. Logo, considerando $\alpha = 0.05$, temos como regra de decisão:

$$F_0 < F_{3,791;0.05} = 2.616159809544512 \longrightarrow \text{concluo por } H_0$$
$$F_0 > F_{3,791;0.05} = 2.616159809544512 \longrightarrow \text{concluo por } H_1$$

Desse modo, calculando o valor observado da estatística de teste, obtivemos:

$$\begin{aligned} F_0 &= \frac{\frac{796.37-743.05}{794-791}}{\frac{743.05}{791}} \\ &= 18.919799792744058 \end{aligned}$$

Portanto, como $F_0 = 18.919799792744058 > F_{3,791;0.05} = 2.616159809544512$, concluímos pela hipótese alternativa H_1, ou seja, não há evidências de que x_1, x_2 e x_3 têm contribuição nula na explicação de Y_i simultaneamente.

Apêndice A

Código

Para essa atividade, o seguinte código feito no Python foi utilizado:

```
                    ## CODIGO - ANALISE DE REGRESSAO ##

                            #### ATIVIDADE 1 ####

# Bibliotecas utilizadas
import random
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
import seaborn as sns
import matplotlib.pyplot as plt
from random import sample
import scipy.stats
from scipy.stats import t
import math
import statsmodels.api as sm
from statsmodels.formula.api import ols

np.random.seed(44)
random.seed(44)

# Numero de observacoes
n = 1000000

# Gerando 199 covariaveis oriundas de distribuicoes Normal, Uniforme,
   Exponencial, Beta e Gamma.
dset = {}
for i in range(1,40):   # Distribuicao Normal
    chave = 'x' + str(i)
    valor = np.random.normal(0,random.random(),n)
    dset[chave] = valor

for i in range(40,80):   # Distribuicao Uniforme
```

```
    chave = 'x' + str(i)
    valor = np.random.rand(n)
    dset[chave] = valor

for i in range(80,120):   # Distribuicao Exponencial
    chave = 'x' + str(i)
    valor = np.random.exponential(random.randrange(1, 11),n)
    dset[chave] = valor

for i in range(120,160):    # Distribuicao Beta
    chave = 'x' + str(i)
    valor = np.random.beta(random.randrange(1, 5),random.randrange(1, 5)
    ,n)
    dset[chave] = valor

for i in range(160,200):    # Distribuicao Gamma
    chave = 'x' + str(i)
    valor = np.random.gamma(random.randrange(1, 5),random.randrange(1,
    5),n)
    dset[chave] = valor

# Base de dados das covariaveis
df = pd.DataFrame(dset)
print(df)

X = df.iloc[:,list(range(199))].values
type(X)

# Gerando betas por meio de uma Normal(0,2)
betas = []
for j in range(0,199):
  b = np.random.normal(0,2)
  betas.append(b)
print(betas)

soma = 0
for k in range(0,199):
  soma = soma + betas[k]*X[:,k]

# Gerando valores para os erros E_i
E_i = np.random.normal(0,1,n)

# Calculando os valores de Y_i
y = 10+soma+E_i
print(y)

# Regressao Linear Multipla
reg = LinearRegression()
reg.fit(X,y)
```

```
80
81 # Coeficientes betachapeu_1,...,betachapeu_199
82 print(reg.coef_)
83
84 # Intercepto betachapeu_0
85 print(reg.intercept_)
86
87 # Calculo dos valores ajustados
88 yi_ajustado = 0
89 for g in range(0,199):
90     yi_ajustado = yi_ajustado + reg.coef_[g]*X[:,g]
91
92 # Calculo dos residuos
93 residuos = y - reg.intercept_ - yi_ajustado
94 print(residuos)
95
96                         #### ATIVIDADE 2 ####
97
98       ## IC para beta0 e beta1 considerando toda a amostra ##
99
100 # Escolha da covariavel x3
101 X3 = np.array(X[:,2]).reshape((-1,1))
102 print(X3)
103
104 # Calculo dos valores de Y_i
105 y_novo = 10 + betas[1]*X[:,2] + np.random.normal(0,1,n)
106 y_novo = np.array(y_novo)
107 print(y_novo)
108
109 # Regressao Linear Simples
110 model = LinearRegression()
111 model.fit(X3,y_novo)
112 print(model.coef_[0]) # valor estimado de beta1
113 print(model.intercept_) # valor estimado de beta0
114
115 # Calculo dos valores ajustados de Y_i
116 yi_chapeu = model.intercept_ + model.coef_*X3[:,0]
117 print(yi_chapeu)
118
119 # Calculo dos residuos
120 residuos = y_novo - yi_chapeu
121 print(residuos)
122
123 # Calculo do SSE e do MSE
124 graus_liberdade= n-2
125 SSE= sum((residuos)**2)
126 print(SSE)
127 MSE= SSE/graus_liberdade
128 print(MSE)
```

```
129
130 x_media = sum(X3)/n
131 print(x_media)
132
133 # Calculo dos estimadores da variancia de b0 e b1
134 est_varb1 = MSE/sum((X3-x_media)**2)
135 print(est_varb1)
136 est_varb0 = MSE*(1/n + (x_media**2/sum((X3-x_media)**2)))
137 print(est_varb0)
138
139 t_student = t.ppf(1-0.05/2, graus_liberdade)
140
141 # Intervalo de Confianca para beta1
142 lim_inf_beta1 = model.coef_ - t_student*math.sqrt(est_varb1)
143 lim_sup_beta1 = model.coef_ + t_student*math.sqrt(est_varb1)
144 print(lim_inf_beta1[0],lim_sup_beta1[0])
145
146
147 # Intervalo de Confianca para beta0
148 lim_inf_beta0 = model.intercept_ - t_student*math.sqrt(est_varb0)
149 lim_sup_beta0 = model.intercept_ + t_student*math.sqrt(est_varb0)
150 print(lim_inf_beta0,lim_sup_beta0)
151
152     ## IC para beta0 e beta1 para uma amostra de tamanho 4800 ##
153
154 # Amostra aleatoria de tamanho 4800 retirada de x3
155 amostra = df['x3'].sample(n=4800,replace=False)
156 amostra = np.array(amostra).reshape(-1,1)
157 print(amostra)
158
159 # Calculo dos novos valores de Y_i
160 y_novo_amostra = 10 + betas[1]*amostra[:,0] + np.random.normal(0,1,4800)
161 print(y_novo_amostra)
162
163 # Regressao Linear Simples
164 modelo2 = LinearRegression()
165 modelo2.fit(amostra,y_novo_amostra)
166 print(modelo2.coef_[0]) # valor estimado de beta1
167 print(modelo2.intercept_) # valor estimado de beta0
168
169 # Calculo dos valores ajustados de Y_i
170 yi_chapeu2 = modelo2.intercept_ + modelo2.coef_*amostra[:,0]
171 print(yi_chapeu2)
172
173 # Calculo dos residuos
174 residuos2 = y_novo_amostra - yi_chapeu2
175 print(residuos2)
176
177 # Calculo do SSE e do MSE
```

```
graus_liberdade2= 4800-2
SSE_2= sum((residuos2)**2)
print(SSE_2)
MSE_2= SSE_2/graus_liberdade2
print(MSE_2)

x_media2 = sum(amostra)/4800
print(x_media2)

# Calculo dos estimadores da variancia de b0 e b1
est_varb1_2 = MSE_2/sum((amostra-x_media2)**2)
print(est_varb1_2)
est_varb0_2 = MSE_2*(1/4800 + (x_media2**2/sum((amostra-x_media2)**2)))
print(est_varb0_2)

t_student2 = t.ppf(1-0.05/2, 4800)

# Intervalo de Confianca para beta1
lim_inf_beta1_2 = modelo2.coef_ - t_student2*math.sqrt(est_varb1_2)
lim_sup_beta1_2 = modelo2.coef_ + t_student2*math.sqrt(est_varb1_2)
print(lim_inf_beta1_2[0],lim_sup_beta1_2[0])

# Intervalo de Confianca para beta0
lim_inf_beta0_2 = modelo2.intercept_ - t_student2*math.sqrt(est_varb0_2)
lim_sup_beta0_2 = modelo2.intercept_ + t_student2*math.sqrt(est_varb0_2)
print(lim_inf_beta0_2,lim_sup_beta0_2)

                        ## Tabela ANOVA ##

base_amostra = np.hstack((np.array(amostra),np.array(y_novo_amostra).
    reshape(-1,1)))
print(base_amostra)
df2 = pd.DataFrame(base_amostra,columns=['x', 'y'])
print(df2)

mod = ols('y ~ x', data=df2).fit()
print(mod.summary())
aov_table = sm.stats.anova_lm(mod, typ=1)
print(aov_table)

                        ## IC para E[Y_0] ##

# Calculo do valor da media de x
print(x_media2[0])

# Escolha de um x proximo da media e um longe
x0_proximo_media = -0.0149
x0_longe_media = 1.5555

```

```
# Calculo dos valores ajustados de Y_i proximo e longe
y0_proximo_media = modelo2.intercept_ + modelo2.coef_*x0_proximo_media
print(y0_proximo_media[0])
y0_longe_media = modelo2.intercept_ + modelo2.coef_*x0_longe_media
print(y0_longe_media[0])

# Calculo dos estimadores da variancia de Y_0chapeu proximo e longe
est_var_y0_proximo = MSE_2*(1/4800 + (x0_proximo_media - x_media2)**2/
    sum((amostra-x_media2)**2))
print(est_var_y0_proximo[0])
est_var_y0_longe = MSE_2*(1/4800 + (x0_longe_media - x_media2)**2/sum((
    amostra-x_media2)**2))
print(est_var_y0_longe[0])

# IC para E[Y_0 proximo]
lim_inf_y0_proximo = y0_proximo_media - t_student2*math.sqrt(
    est_var_y0_proximo)
lim_sup_y0_proximo = y0_proximo_media + t_student2*math.sqrt(
    est_var_y0_proximo)
print(lim_inf_y0_proximo[0],lim_sup_y0_proximo[0])

# IC para E[Y_0 longe]
lim_inf_y0_longe = y0_longe_media - t_student2*math.sqrt(
    est_var_y0_longe)
lim_sup_y0_longe = y0_longe_media + t_student2*math.sqrt(
    est_var_y0_longe)
print(lim_inf_y0_longe[0],lim_sup_y0_longe[0])

# Calculo de E[Y_0 proximo]
esperanca_y0_proximo = 10 + betas[1]*x0_proximo_media
print(esperanca_y0_proximo)

# Calculo de E[Y_0 longe]
esperanca_y0_longe = 10 + betas[1]*x0_longe_media
print(esperanca_y0_longe)


                    ## IP para uma nova resposta ##

# Gerando uma nova observacao da covariavel x3
xnovogerado = np.random.normal(0,1)
print(xnovogerado)

# Calculo de Ynovo
ynovo = 10 + betas[1]*xnovogerado + np.random.normal(0,1)
print(ynovo)

# Calculo de Ynovo_chapeu
ychapeu_novo_gerado = model.intercept_+ model.coef_*xnovogerado
```

```
print(ychapeu_novo_gerado[0])

# IP para uma nova resposta
lim_inf_ychapeu_novo_g = ychapeu_novo_gerado - t_student*math.sqrt(MSE
    *(1+1/n + (xnovogerado-x_media)**2/sum((X3-x_media)**2)))
lim_sup_ychapeu_novo_g = ychapeu_novo_gerado + t_student*math.sqrt(MSE
    *(1+1/n + (xnovogerado-x_media)**2/sum((X3-x_media)**2)))
print(lim_inf_ychapeu_novo_g[0],lim_sup_ychapeu_novo_g[0])

            ## IP para a media de m novas respostas ##

# Escolha de um m
m = 150

y_m = []
for w in range(0,m):
    y_m.append(10 + betas[1]*xnovogerado + np.random.normal(0,1))

# Calculo da media das m respostas
media_y_m = sum(y_m)/m
print(media_y_m)

# IP para a media de m novas respostas
lim_inf_ychapeu_novo_m = ychapeu_novo_gerado  - t_student*math.sqrt(MSE
    *(1/m+1/n + (xnovogerado-x_media)**2/sum((X3-x_media)**2)))
lim_sup_ychapeu_novo_m = ychapeu_novo_gerado  + t_student*math.sqrt(MSE
    *(1/m+1/n + (xnovogerado-x_media)**2/sum((X3-x_media)**2)))
print(lim_inf_ychapeu_novo_m[0],lim_sup_ychapeu_novo_m[0])

                    ## Teste Linear Geral ##

np.random.seed(91)
random.seed(91)

# Escolhendo 3 betas proximos de 0
b = sorted(betas)
ordenado = [i for i in b if i > 0]
beta1 = ordenado[0]
beta2 = ordenado[1]
beta3 = ordenado[2]

# Pegando uma amostra aleatoria de 800 observacoes de 8 covariaveis
    escolhidas, sendo x79, x127 e x194 as que acompanham os betas
    proximos de 0
x79 = np.array(df['x79'].sample(n=800,replace=False)).reshape(-1,1)
x127 = np.array(df['x127'].sample(n=800,replace=False)).reshape(-1,1)
x194 = np.array(df['x194'].sample(n=800,replace=False)).reshape(-1,1)
x60 = np.array(df['x60'].sample(n=800,replace=False)).reshape(-1,1)
x120 = np.array(df['x120'].sample(n=800,replace=False)).reshape(-1,1)
```

```
x145 = np.array(df['x145'].sample(n=800,replace=False)).reshape(-1,1)
x160 = np.array(df['x160'].sample(n=800,replace=False)).reshape(-1,1)
x175 = np.array(df['x175'].sample(n=800,replace=False)).reshape(-1,1)

### MODELO COMPLETO

# Juntando os dados dessas covariaveis
amostra_comp = np.hstack((x79,x127,x194,x60,x120,x145,x160,x175))
amostra_comp = pd.DataFrame(amostra_comp)
amostra_comp = amostra_comp.iloc[:,list(range(8))].values
print(amostra_comp)

# Calculo dos Y_i
soma2 = 0
betas_comp = [beta1,beta2,beta3,betas[60],betas[120],betas[145],betas
    [160],betas[175]]
for q in range(0,8):
     soma2 = soma2 + betas_comp[q]*amostra_comp[:,q]

Y_F = 10+soma2+np.random.normal(0,1,800)
print(Y_F)

# Modelo de Regressao Linear Multipla
modelo_completo = LinearRegression()
modelo_completo.fit(amostra_comp,Y_F)
print(modelo_completo.coef_) # valor estimado de beta1,...,beta8
print(modelo_completo.intercept_) # valor estimado de beta0

# Calculo dos valores ajustados
yi_estimado_comp = 0
for gi in range(0,8):
    yi_estimado_comp = yi_estimado_comp + modelo_completo.coef_[gi]*
    amostra_comp[:,gi]
print(yi_estimado_comp)

# Calculo dos residuos
residuos_comp = Y_F - modelo_completo.intercept_ - yi_estimado_comp
print(residuos_comp)

# Calculo do SSE_F
SSE_F = sum((residuos_comp)**2)
print(SSE_F)

### MODELO REDUZIDO

# Juntando os dados das 5 covariaveis que nao acompanham os 3 betas
    proximos de 0
amostra_red = np.hstack((x60,x120,x145,x160,x175))
amostra_red = pd.DataFrame(amostra_red)
```

```
amostra_red = amostra_red.iloc[:,list(range(5))].values
print(amostra_red)

# Calculo dos Y_i
soma3 = 0
betas_red = [betas[60],betas[120],betas[145],betas[160],betas[175]]
for qi in range(0,5):
     soma3 = soma3 + betas_red[qi]*amostra_red[:,qi]

Y_R = 10+soma3+np.random.normal(0,1,800)
print(Y_R)

# Modelo de Regressao Linear Multipla
modelo_reduzido = LinearRegression()
modelo_reduzido.fit(amostra_red,Y_R)
print(modelo_reduzido.coef_) # valor estimado de beta4,...,beta8
print(modelo_reduzido.intercept_) # valor estimado de beta0

# Calculo dos valores ajustados
yi_estimado_red = 0
for gj in range(0,5):
    yi_estimado_red = yi_estimado_red + modelo_reduzido.coef_[gj]*
   amostra_red[:,gj]
print(yi_estimado_red)

# Calculo dos residuos
residuos_red = Y_R - modelo_reduzido.intercept_ - yi_estimado_red
print(residuos_red)

# Calculo do SSE_R
SSE_R = sum((residuos_red)**2)
print(SSE_R)

# Graus de liberdade do modelo completo e do reduzido, respectivamente
glF = 800-9
glR = 800-6

# Calculo de SSE_R - SSE_F
diferenca = SSE_R - SSE_F
print(diferenca)

# Calculo do valor observado da estatistica teste (F_0)
estat_teste = ((diferenca)/(glR - glF))/(SSE_F/glF)
print(estat_teste)

# Calculo da F
F = scipy.stats.f.ppf(0.95,glR-glF,glF,scale=1)
print(F)
```