Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Departamento de Estatística

Relatório Atividade 1 Análise de Regressão

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

Agosto, 2021

Sumário

Capítulo 1

Introdução

Este trabalho tem como objetivo abordar uma aplicação do modelo de regressão linear múltipla. A diferença entre a regressão linear simples e a múltipla é que na múltipla são tratadas duas ou mais covariáveis, permitindo a comparação entre a combinação de diversos fatores do modelo, o qual pode ser representado pela seguinte equação:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_l x_{il} + \epsilon_i, \quad i = 1, .., n \ ,$$

sendo l o número de covariáveis.

Assim, trazendo para o problema da atividade, geramos um conjunto de dados com $n = 1000000$ (10^6) observações e $l = 199$ covariáveis através do software de linguagem de programação Python. Após isso, foi realizada a estimação do modelo de regressão linear múltipla, além de uma comparação entre os parâmetros verdadeiros e os estimados.

Nesse sentido, no Capítulo 2 apresentamos os resultados obtidos na estimação do modelo de regressão linear múltipla e uma abordagem também do modelo na forma matricial. Ademais, fizemos uma análise gráfica desse modelo através de um mapa de calor e de matrizes de gráficos de dispersão. Por fim, no Apêndice está o código que foi utilizado para a realização dessa atividade.

Capítulo 2

Resultados

Neste capítulo, apresentamos os resultados obtidos.

2.1 Modelo de Regressão Linear Múltipla

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 covariávies.

Conseguimos ter uma melhor visualização da base ao olhar para apenas uma coluna dela, como exemplificado pela Figura 2.2:

```
0         0.990220
1         0.406758
2         0.631710
3         0.476006
4         0.327398
            ...
999995    0.980769
999996    0.968079
999997    0.491758
999998    0.386370
999999    0.999209
Name: x120, Length: 1000000, dtype: float64
```

Figura 2.2: Coluna “x120”da base de dados.

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 2.3:

```
[-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 2.3: 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 2.4: 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 2.5: Valores de Y_i.

Desse modo, tendo obtido os valores das variáveis respostas $Y_i, i = 1, ..., 10^6$ e os valores das variáveis preditoras $x_{ij}, i = 1, .., 10^6$ e $j = 1, ..., 199$, estimamos, através da regressão linear múltipla, os valores $\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_{199}$, sendo o intercepto:

$$\hat{\beta}_0 = 9.999999999999424$$

e os coeficientes $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_{199}$:

```
[-0.73037484  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 2.6: Valores de $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_{199}$.

Para analisarmos a diferença entre os parâmetros verdadeiros e os estimados, fizemos $\beta_i - \hat{\beta}_i$. Lembrando que os valores de β_i podem ser encontrados na Figura 2.3 e os valores de $\hat{\beta}_i$ podem ser encontrados na Figura 2.6. Assim:

```
[ 5.75539616e-13 -2.44776542e+00  3.58602037e-14  1.99840144e-15
  3.88578059e-14  2.55351296e-14 -1.12174159e-13 -4.36317649e-14
 -2.88791213e-12 -1.68753900e-14  5.99520433e-14 -5.32907052e-15
 -5.95079541e-14  1.38777878e-14 -4.32431868e-14 -3.39728246e-14
 -7.90478794e-14  3.18411963e-13 -3.27515792e-15  2.26485497e-14
  7.21644966e-14 -4.44089210e-16 -8.43769499e-15 -4.66293670e-15
 -5.45050116e-14 -1.31450406e-13  4.30558367e-15  6.54587495e-13
  1.86517468e-14  1.15463195e-14 -5.77315973e-15  9.65894031e-15
  3.75255382e-14 -8.37108161e-14  6.07625061e-13  1.50990331e-14
  1.08801856e-14 -1.44328993e-14 -1.70974346e-14  1.45994328e-14
 -1.77635684e-15 -1.97619698e-14  5.36237721e-14  4.97379915e-14
  2.53130850e-14 -3.90798505e-14 -2.39808173e-14 -1.33226763e-14
  1.79856130e-14 -7.10542736e-14 -9.82547377e-15 -5.50670620e-14
  1.59872116e-14  2.79776202e-14  1.99840144e-15  9.99200722e-15
 -2.70894418e-14  3.99680289e-15  2.30926389e-14 -1.94289029e-14
  2.88657986e-15 -5.99520433e-15  5.88418203e-15  2.75335310e-14
 -2.13162821e-14  2.04281037e-14 -3.99680289e-15 -2.79776202e-14
 -6.66133815e-15  2.07056594e-14 -7.99360578e-15  4.66293670e-15
  4.32986980e-15 -1.52933222e-14  3.57491814e-14  2.27595720e-14
 -4.61852778e-14 -1.73194792e-14  7.77156117e-16  1.35447209e-14
 -1.33226763e-15 -5.32907052e-15  3.33066907e-16  1.11022302e-15
 -6.66133815e-15  2.60902411e-15 -3.55271368e-15  6.32827124e-15
  6.21724894e-15  4.44089210e-16  2.66453526e-15 -2.44249065e-15
 -7.54951657e-15  6.66133815e-16  3.44169138e-15 -2.39808173e-14
  3.02535774e-15 -1.31006317e-14 -3.33066907e-15  5.35682609e-15
  1.17683641e-14 -5.55111512e-16  3.10862447e-15  4.44089210e-16
  1.33226763e-15  1.97064587e-15  0.00000000e+00 -2.02060590e-14
 -3.10862447e-15  5.55111512e-15  4.10782519e-15 -5.55111512e-15
 -3.99680289e-15 -2.34257058e-14  2.44249065e-15 -2.05391260e-14
 -4.44089210e-15 -3.33066907e-15  3.21964677e-15  1.22124533e-15
 -8.50430837e-14 -2.93098879e-14  6.66133815e-16 -9.88098492e-14
 -1.77635684e-15 -2.01505479e-14 -1.34336986e-14 -2.18505769e-14
 -2.88657986e-15  2.53130850e-14 -5.10702591e-15  9.22595333e-14
 -1.09912079e-14 -8.88178420e-15  1.57651669e-14 -7.68274333e-14
 -6.08679773e-14  1.77635684e-15 -1.19904087e-14  6.66133815e-16
 -8.99280650e-15 -1.77635684e-15 -1.15463195e-14 -1.99285033e-14
 -1.55431223e-14 -3.10862447e-15 -1.44328993e-13 -5.88418203e-15
 -7.77156117e-14 -1.48769885e-14 -5.32907052e-15  2.04281037e-14
  2.33979502e-14  6.66133815e-15  2.75335310e-14  1.08246745e-14
 -2.26485497e-14 -5.28466160e-14  2.77555756e-17  9.76996262e-15
 -2.35367281e-14  6.21724894e-15  4.44089210e-16  4.21884749e-15
 -2.22044605e-15 -4.44089210e-16 -2.66453526e-15  8.40993941e-15
 -2.66453526e-15 -5.55111512e-15 -3.99680289e-15  7.99360578e-15
 -1.77635684e-15 -5.77315973e-15 -2.77555756e-15 -8.88178420e-16
 -1.55431223e-15  4.88498131e-15  4.44089210e-16 -6.29357677e-15
 -3.33066907e-16 -3.33066907e-15  0.00000000e+00  1.66533454e-15
  5.32907052e-15 -3.55271368e-15  2.44249065e-15 -4.44089210e-16
 -1.44328993e-15  4.49640325e-15  2.22044605e-16 -1.38777878e-15
 -4.44089210e-16 -1.33226763e-14 -1.78676518e-15 -7.32747196e-15
  1.33226763e-15  2.88657986e-15 -1.02140518e-14 -8.10462808e-15]
```

Figura 2.7: Valores de $\beta_i - \hat{\beta}_i, i = 0, ..., 199$.

Com base na Figura 2.7 acima, notamos que a grande maioria dos valores são extremamente próximos de 0, ou seja, concluímos que o modelo parece ser bem ajustado.

A fim de uma melhor análise, calculamos também os resíduos, sendo o i-ésimo resíduo a diferença entre o valor observado e o valor correspondente ajustado $\hat{Y}_i$, ou seja,

$$e_i = Y_i - \hat{Y}_i, \ i = 1, ..., 10^6$$

e, para isso, antes encontramos os valores correspondentes ajustados $\hat{Y}_i$, obtendo:

```
[-138.77452511   50.28502227  -16.50087476 ...  -48.03158261  116.04605317
  -14.63388594]
```

Figura 2.8: Valores de $\hat{Y}_i, i = 1, ..., 10^6$.

Portanto, temos como resultado dos resíduos e_i:

```
[-2.27373675e-13  1.13686838e-13  2.30926389e-13 ...  9.23705556e-14
  6.25277607e-13 -5.36459765e-13]
```

Figura 2.9: Valores dos resíduos $e_i, i = 1, ..., 10^6$.

Do mesmo modo que a análise anterior, percebemos que os resíduos também possuem valores muito próximos de 0, reforçando a ideia de que o modelo é bem ajustado.

2.2 Modelo na Forma Matricial

Para fins de comparação, faremos o mesmo modelo agora na forma matricial, e encontraremos os estimadores de $\beta_0, \beta_1, ..., \beta_{199}$ através da fórmula:

$$(X^tX)^{-1}X^tY = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_{199} \end{bmatrix}$$

Assim, sabendo que

```
      [[ 1.00000000e+00 -3.06653043e-01 -7.85878063e-01 ...  1.76700582e+01
         3.48710556e+00  5.21153033e+00]
       [ 1.00000000e+00  5.37779197e-01  4.21365464e-01 ...  8.68212260e+00
         1.83157889e+00  4.88965087e-02]
       [ 1.00000000e+00  5.09092912e-01 -9.54494322e-03 ...  1.26369599e+01
         2.50835990e+00  7.83896634e-02]
X =     ...
       [ 1.00000000e+00 -1.06464648e-01 -7.75124452e-01 ...  4.14270369e+00
         4.13662612e+00  4.80301749e-01]
       [ 1.00000000e+00 -7.23440046e-01  6.06754984e-01 ...  1.05013332e+01
         1.01988618e+00  5.38031914e+00]
       [ 1.00000000e+00  6.70019508e-01 -8.99992928e-01 ...  1.04281577e+01
         2.79144401e+00  2.46412004e+00]]
```

Figura 2.10: Matriz X.

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

Figura 2.11: Matriz Y.

temos:

```
            [[ 5.84718188e-04 -7.41562526e-08  4.15776569e-09 ... -2.49399060e-07
              -1.03716244e-06 -3.34594210e-07]
             [-7.41562526e-08  6.00255940e-06 -6.84040509e-09 ...  2.86570316e-10
              -9.01147741e-10 -3.58470719e-10]
             [ 4.15776569e-09 -6.84040509e-09  3.40530798e-06 ...  1.33471032e-10
              -1.14370639e-10 -3.19420865e-11]
(Xt X)^-1 =   ...
             [-2.49399060e-07  2.86570316e-10  1.33471032e-10 ...  2.07851222e-08
              -7.11492603e-11 -5.13870518e-12]
             [-1.03716244e-06 -9.01147741e-10 -1.14370639e-10 ... -7.11492603e-11
               2.49842323e-07 -7.49643744e-12]
             [-3.34594211e-07 -3.58470719e-10 -3.19420865e-11 ... -5.13870518e-12
              -7.49643744e-12  1.11492226e-07]]
```

Figura 2.12: Matriz $(X^t X)^{-1}$.

```
        [[ 5.88104326e+07 ]
         [-1.52610620e+05]
         [ 1.14790326e+05]
         [ 3.70460514e+05]
XtY =    ...
         [ 3.39287217e+08]
         [ 6.53466882e+08]
         [ 2.29905074e+08]
         [ 1.79106872e+08 ]]
```

Figura 2.13: Matriz $X^t Y$.

Desse modo,

$\hat{\beta}$ =

```
[ [ 10.          ]
  [ -0.73037484 ]
  [ 0.82174032  ]
  [ 0.64914096  ]
  ...
  [ -0.75532492 ]
  [ -1.09302609 ]
  [ -1.35194961 ]
  [ 0.35378042  ] ]
```

Figura 2.14: Valores estimados de $\beta_0, \beta_1, ..., \beta_{199}$.

Ao observamos o resultado da Figura 2.14 acima, e comparando com o valor de $\hat{\beta}_0 = 9.999999999999424$ e com os valores de $\hat{\beta}_1, \hat{\beta}_2, ..., \hat{\beta}_{199}$ encontrados e apresentados na Figura 2.6, percebemos que os valores são basicamente iguais, o que já era esperado.

A fim de uma melhor visualização, fizemos a diferença entre o β estimado anteriormente e o encontrado através da multiplicação de matrizes $(X^tX)^{-1}X^tY$:

```
[ [ -1.95875316e-10 ]
  [  2.10942375e-14 ]
  [  1.33226763e-15 ]
  [ -2.55351296e-14 ]
  ...
  [  1.93955962e-13 ]
  [ -2.58459920e-13 ]
  [ -2.67119660e-13 ]
  [ -8.63753513e-14 ] ]
```

Figura 2.15: Valores da diferença entre o β estimado anteriormente e o encontrado através da multiplicação de matrizes $(X^tX)^{-1}X^tY$.

Desse modo, percebemos que essa diferença é quase 0, o que nos mostra que ambos os métodos são eficazes e possuem o mesmo resultado para a estimação dos parâmetros $\beta_0, \beta_1, ..., \beta_{199}$.

2.3 Análise Gráfica

2.3.1 HeatMap

Para uma melhor análise e para facilitar a identificação de variáveis correlacionadas, plotamos um heatmap (mapa de calor), que utiliza os coeficientes de correlação de todas as variáveis e mapeia os valores em uma matriz.

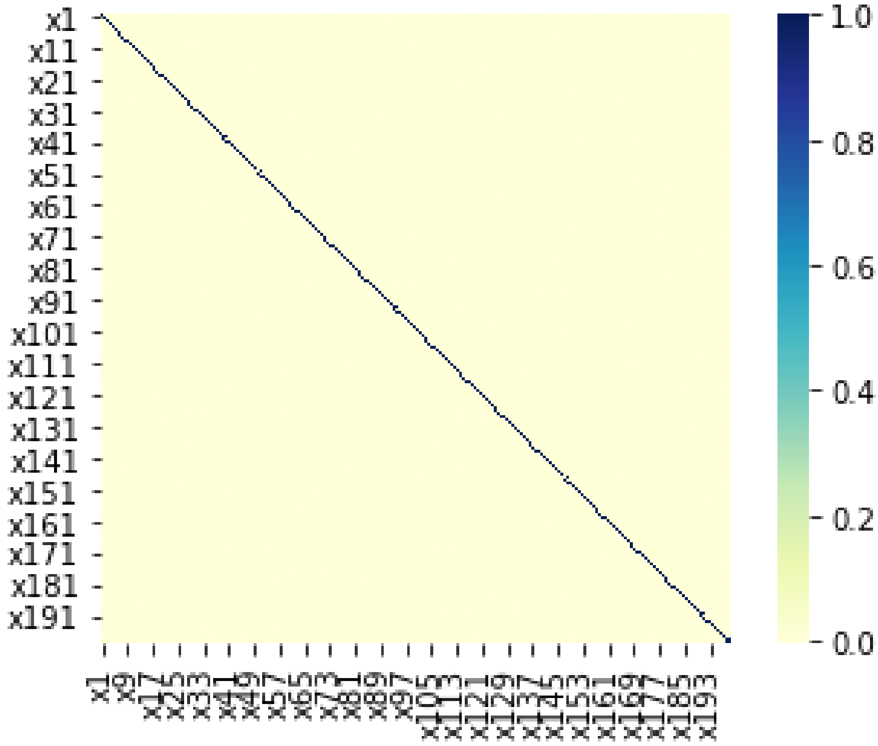

Figura 2.16: Heatmap (mapa de calor).

Através da Figura 2.16 percebemos que há independência, ou seja, não temos nenhuma relação entre as covariáveis, o que pode ser exemplificado pelo fato de apenas a diagonal principal estar em azul, já que a correlação entre uma covariável e ela mesma é 1, e as outras correlações estarem todas em um amarelo bem claro, próximas de 0.

2.3.2 Matriz de Dispersão

Por fim, para demonstrar se existem correlações lineares entre as variáveis, foi plotado uma matriz de gráficos de dispersão (scatterplot matrix). Para montar essa matriz, realizamos uma amostragem aleatória simples sem reposição e sorteamos 10 covariáveis das 199 presentes no modelo.

Essa escolha foi realizada pelo fato de termos um grande número de observações e covariáveis. Nesse sentido, seria necessário fazer um uso muito alto da memória RAM da máquina para conseguir plotar essa matriz, contudo, nenhum dos membros do grupo possui uma máquina com memória RAM suficiente para o mesmo.

As covariáveis sorteadas para a realização da matriz foram: x_4, x_{17}, x_{45}, x_{58}, x_{89}, x_{91}, x_{146}, x_{169}, x_{172} e x_{196}. Na Figura 2.17 temos a representação da matriz de gráficos de dispersão com as covariáveis sorteadas.

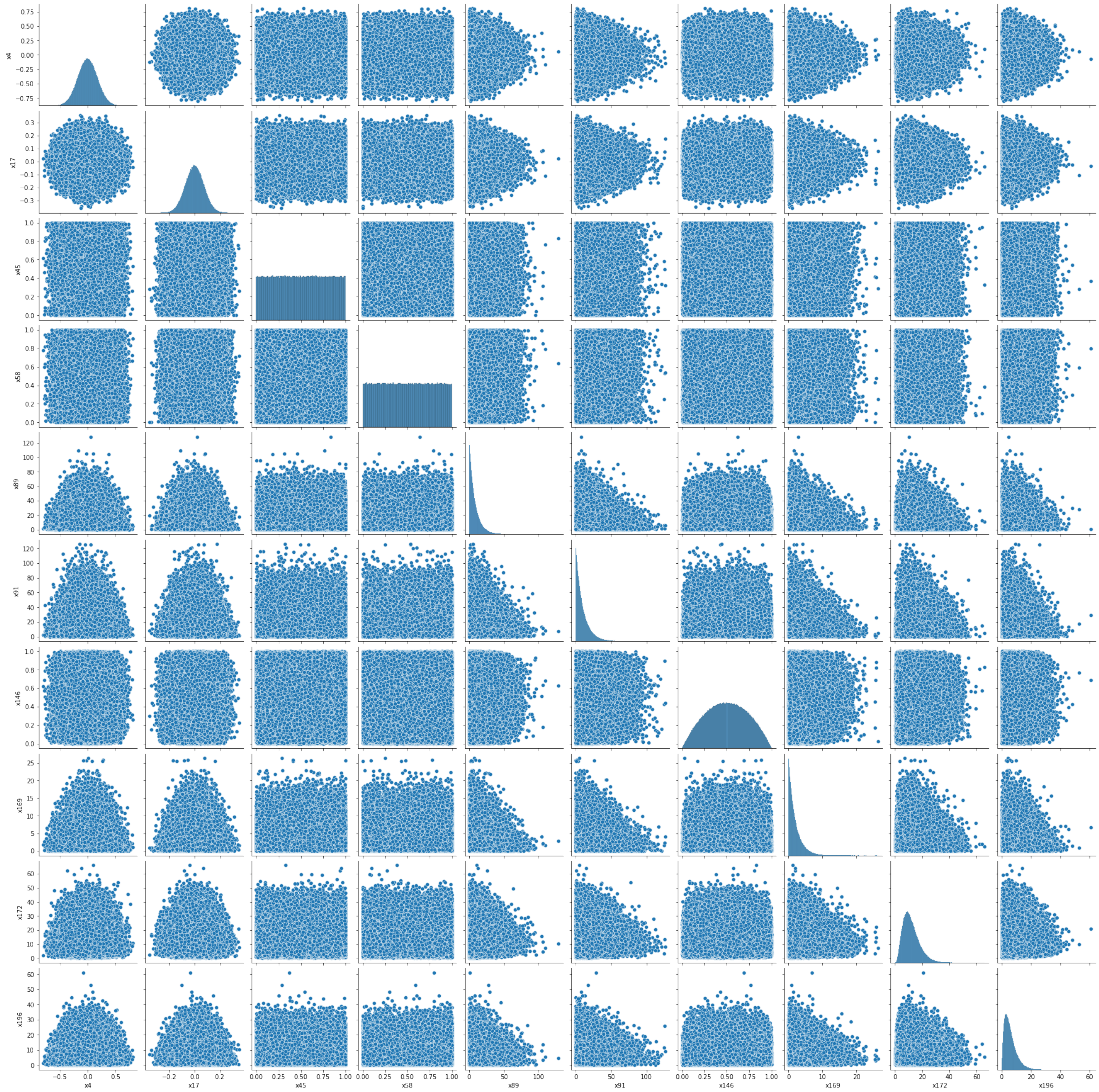

Figura 2.17: Scatterplot matrix (matriz de gráficos de dispersão).

Ao analisar a Figura 2.17, percebemos na diagonal principal os gráficos das distribuições de cada covariável, sendo as distribuições: Normal (x_4 e x_{17}), Uniforme (x_{45} e x_{58}), Exponencial (x_{89} e x_{91}), Beta (x_{146}) e Gamma (x_{169}, x_{172} e x_{198}). Além disso, os gráficos de dispersão entre duas covariáveis de distribuição Uniforme e os gráficos de dispersão entre uma covariável de distribuição Uniforme e outra de distribuição Beta possuem a forma de um quadrado, já que estas distribuições só assumem valores entre 0 e 1.

Ademais, é notável que na maioria das relações das covariáveis não há uma dispersão muito considerável nos dados, pois estes estão muito concentrados.

Para verificar se existe algum padrão de correlação entre as covariáveis, realizamos um novo sorteio em que as covariáveis sorteadas foram: x_{14}, x_{20}, x_{32}, x_{43}, x_{81}, x_{100}, x_{114}, x_{124}, x_{173} e x_{179}. Na Figura 2.18 temos a representação da nova matriz.

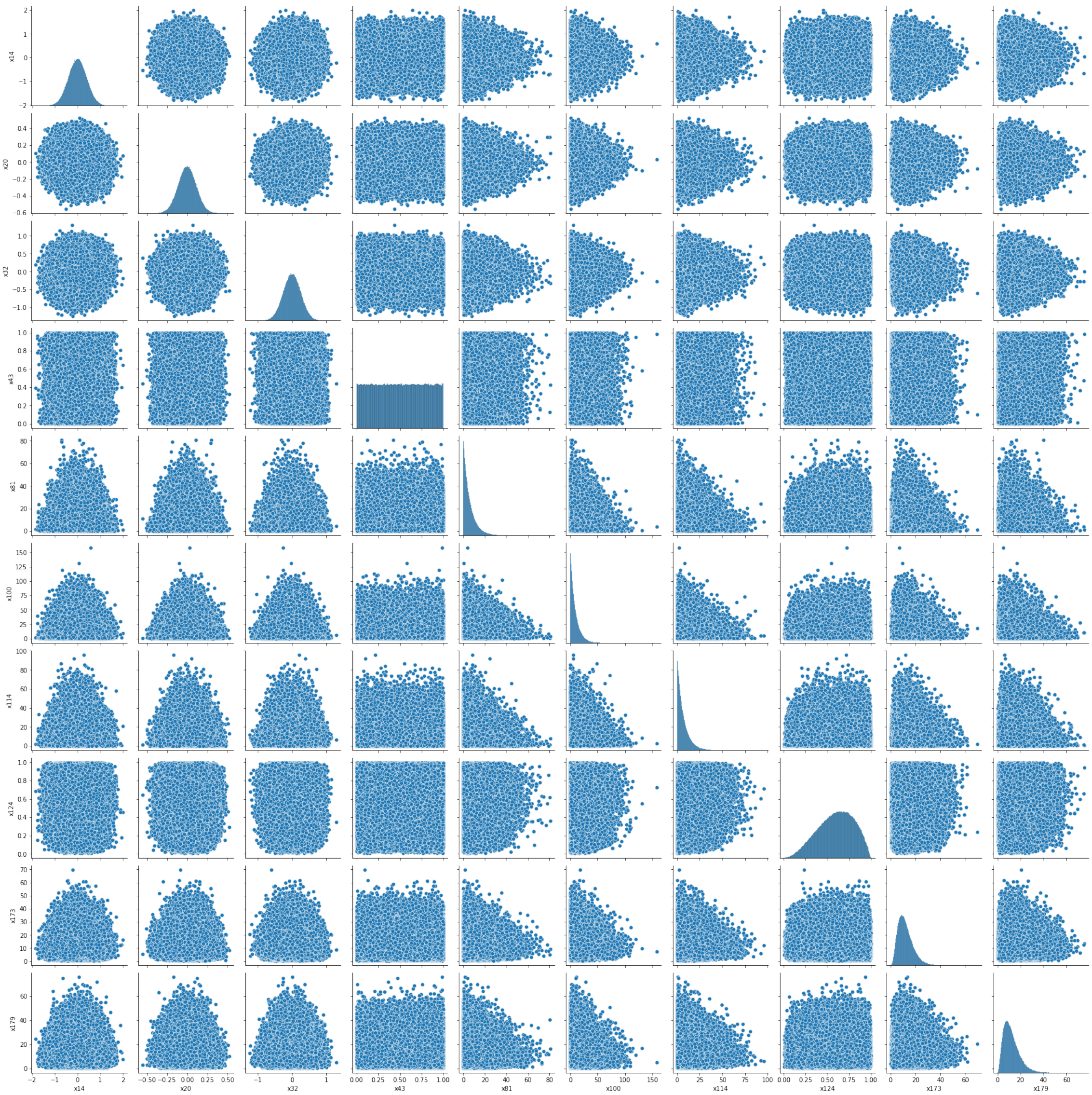

Figura 2.18: Scatterplot matrix (matriz de gráficos de dispersão).

Notamos através da Figura 2.18 que, dessa vez, temos na diagonal principal os gráficos das distribuições: Normal (x_{14}, x_{20} e x_{32}), Uniforme (x_{43}), Exponencial (x_{81}, x_{100} e x_{114}), Beta (x_{124}) e Gamma ($x_{173},$ e x_{179}). Do mesmo modo, os gráficos de dispersão entre duas covariáveis de distribuição Uniforme e os gráficos de dispersão entre uma covariável de distribuição Uniforme e outra de distribuição Beta possuem a forma de um quadrado, já que estas distribuições só assumem valores entre 0 e 1.

Percebemos ainda que este gráfico possui muitas semelhanças com o da Figura 2.17 feito anteriormente, o que nos leva a pensar que pode existir certo padrão de correlação entre as variáveis.

Para melhores conclusões, seria necessário relacionar todas as 199 covariáveis, o que foi inviável computacionalmente, além de que a análise do gráfico seria confusa, já que teríamos 39601 gráficos de dispersão juntos. De qualquer maneira, o código para isto está presente no Apêndice.

Apêndice A

Código

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

```
                ## CODIGO ATIVIDADE 1 - ANALISE DE REGRESSAO ##

# 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

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)

# Exemplo da coluna 120 dessa base
print(df["x120"])

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)

# Coeficientes betachapeu_1,...,betachapeu_199
print(reg.coef_)

```

```
# Intercepto betachapeu_0
print(reg.intercept_)

# Analise da diferenca entre os betas verdadeiros e os estimados
betas.insert(0,10)
betas2 = np.array(betas)
betasestimados = np.hstack((reg.intercept_,reg.coef_))

diferenca = betas2 - betasestimados
print(diferenca)

# Calculo dos valores ajustados
yi_ajustado = 0
for g in range(0,199):
    yi_ajustado = yi_ajustado + reg.coef_[g]*X[:,g]

# Calculo dos residuos
residuos = y - reg.intercept_ - yi_ajustado
print(residuos)

                    # MODELO NA FORMA MATRICIAL #

# Calculo da matriz X
x = np.insert(X,0,1,axis=1)
x = np.matrix(x)
print(x)

# Calculo da matriz Xtransposta
x_transp = x.T
x_transp = np.matrix(x_transp)

# Calculo da matriz Y
Y = np.matrix(y)
Y = Y.T
print(Y)

xtransp_x = np.matmul(x_transp,x) # X^t * X
xtransp_x_inversa = np.linalg.inv(xtransp_x) # (X^t * X)^(-1)
xtransp_y = np.matmul(x_transp,Y) # X^t * Y

# Calculo de (Xtrasnposta X)^(-1)(Xtransposta Y)
betachapeu = np.matmul(xtransp_x_inversa,xtransp_y)
print(betachapeu)

# Calculo da diferenca entre os betas estimados anteiormente o os
   encontrados atraves da multiplicacao de matrizes
betasestimados_1 = np.matrix(betasestimados)
betasestimados_1 = betasestimados_1.T

```

```
diferenca2 = betachapeu - betasestimados_1
print(diferenca2)

                          ## GRAFICOS ##

# Heatmap
sns.heatmap(df.corr(), square = True, cmap = "YlGnBu")

# Scatterplot matrix 10x10
def sorteio_amostras():
    return sorted(sample(range(1, 200), 10))

print(sorteio_amostras())

sns.pairplot(df.iloc[:,sorteio_amostras()])

## pd.plotting.scatter_matrix(df) # Scatterplot matrix 199x199
```