

Tópicos Avançados em Aprendizagem de Máquina - Trabalho 1 - Regressão Linear Bayesiana

Claudemir Woche Vasconcelos Carvalho

5 de janeiro de 2021

1 Importação de bibliotecas

```
[8]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import rcParams
rcParams['figure.figsize'] = (10,10)
plt.style.use("seaborn-whitegrid")
np.set_printoptions(precision=5,suppress=True)
```

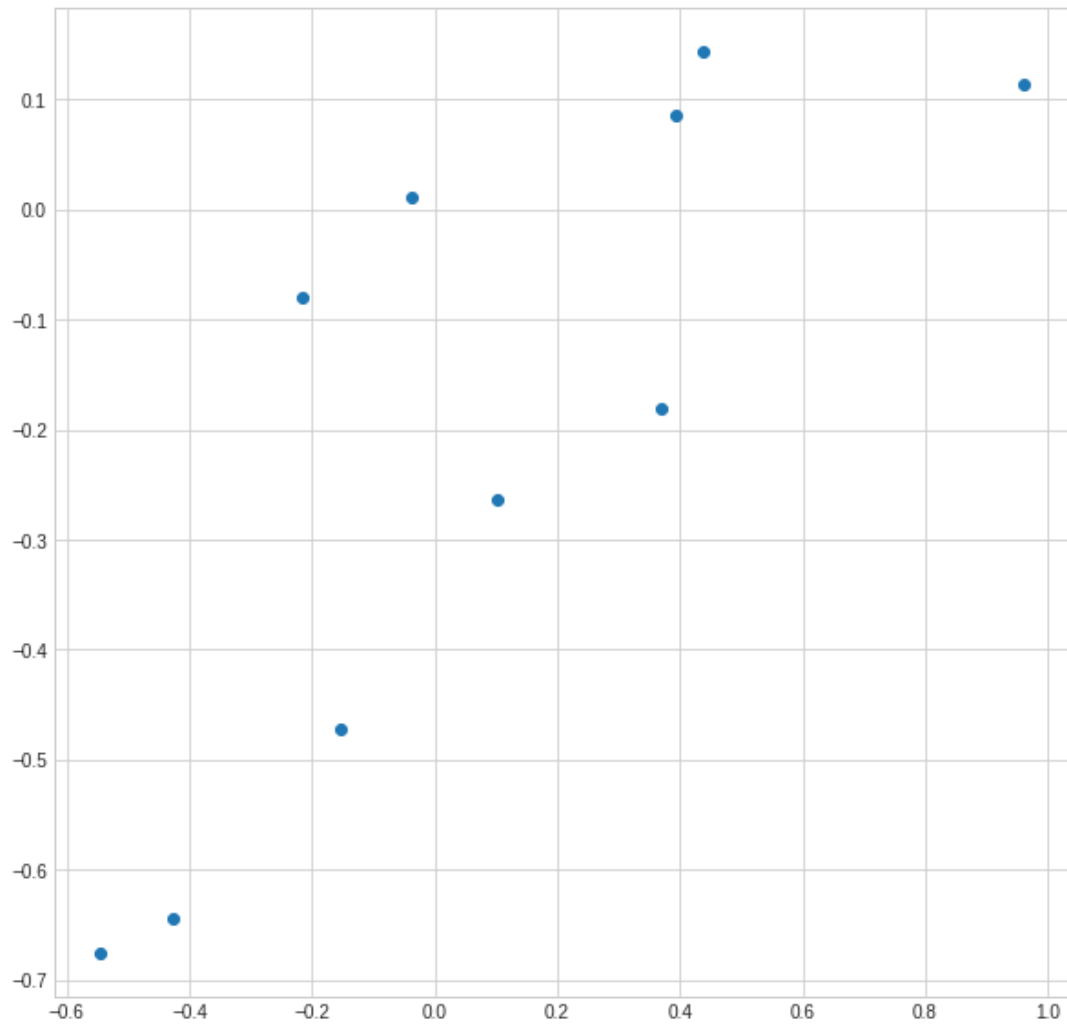
2 Importação e visualização dos dados

```
[3]: dataset = pd.read_csv('linear_regression_data.csv',header=None).
    ↪rename(columns={0:'x',1:'y'})
dataset
```

```
[3]:
```

	x	y
0	0.392938	0.086360
1	-0.427721	-0.643872
2	-0.546297	-0.674981
3	0.102630	-0.262892
4	0.438938	0.143177
5	-0.153787	-0.472729
6	0.961528	0.114167
7	0.369659	-0.180323
8	-0.038136	0.011821
9	-0.215765	-0.079865

```
[4]: plt.scatter(dataset['x'],dataset['y'])
plt.show()
```



3 Pré-processamento dos dados

```
[5]: p = 2
x = dataset['x'].values.reshape(-1,1)
y = dataset['y'].values.reshape(-1,1)
X = np.hstack([np.ones(dataset.shape[0]).reshape(-1,1),dataset['x'].
↪values.reshape(-1,1)])
X
```

```
[5]: array([[ 1.      ,  0.39294],
           [ 1.      , -0.42772],
           [ 1.      , -0.5463 ],
           [ 1.      ,  0.10263],
           [ 1.      ,  0.43894],
           [ 1.      , -0.15379],
           [ 1.      ,  0.96153],
```

```
[ 1.      ,  0.36966],
[ 1.      , -0.03814],
[ 1.      , -0.21576]])
```

4 Passo de estimação

4.1 Definições a partir de conhecimentos/experimentos anteriores

Os momentos da priori $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0)$.

→ Definirei $\mathbf{m}_0 = [0 \ 0]$ e $\mathbf{S}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

A variância do ruído $p(\epsilon) = \mathcal{N}(\epsilon|0, \sigma^2)$

→ Usarei $\sigma^2 = \{1, 5e-2, 1e-2, 1e-4, 1e-6, 1e-8\}$

```
[6]: m0 = np.zeros(p).reshape(-1,1)
      S0 = np.eye((p))
      sigma_ruidos = [1,5E-2,1E-2,1E-4,1E-6,1E-8]
```

4.2 Cálculo da posteriori de \mathbf{w} e da distribuição preditiva

Posteriori de \mathbf{w}

→ $p(\mathbf{w}|\mathcal{D}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$

$\boldsymbol{\mu} = \mathbf{m}_0 + (\mathbf{S}_0\mathbf{X}^T\mathbf{X} + \sigma^2\mathbf{I})^{-1} \mathbf{S}_0\mathbf{X}^T(\mathbf{y} - \mathbf{X}\mathbf{m}_0)$,

$\boldsymbol{\Sigma} = \mathbf{S}_0 - (\mathbf{S}_0\mathbf{X}^T\mathbf{X} + \sigma^2\mathbf{I})^{-1} \mathbf{S}_0\mathbf{X}^T\mathbf{X}\mathbf{S}_0$

Distribuição preditiva

→ $p(\mathbf{y}_*|\mathbf{X}_*) = \mathcal{N}(\mathbf{y}_*|\mathbf{X}_*\boldsymbol{\mu}, \mathbf{X}_*\boldsymbol{\Sigma}\mathbf{X}_*^T + \sigma^2\mathbf{I})$

```
[9]: f, axes = plt.subplots(3, 2)
      for i in range(3):
          for j in range(2):
              sigma_ruido = sigma_ruidos[i*2+j]

              u_1 = np.linalg.inv(S0.dot(X.T).dot(X) + np.eye(p)*sigma_ruido)
              u_2 = S0.dot(X.T).dot(y - X.dot(m0))
              u = m0 + u_1.dot(u_2)

              sigma_1 = u_1
              sigma_2 = S0.dot(X.T).dot(X).dot(S0)
              sigma = S0 - (sigma_1.dot(sigma_2))

              u_final = X.dot(u).flatten()
```

```

sigma_final = np.diag(X.dot(sigma).dot(X.T) + np.
↳eye(10)*sigma_ruido )

banda = 2 * np.sqrt(sigma_final).reshape(-1,1).flatten()
banda_mais = u_final + banda
banda_menos = u_final - banda

axes[i,j].set_title(f"sigma_ruído = {sigma_ruido}", fontsize=14)
sns.
↳scatterplot(dataset['x'],dataset['y'],ax=axes[i,j],color='r')
sns.lineplot(dataset['x'],banda_mais,ax=axes[i,j],label='$\mu+2\sigma$')
↳\sigma$')
sns.lineplot(dataset['x'],u_final,ax=axes[i,j],label='$\mu$')
sns.
↳lineplot(dataset['x'],banda_menos,ax=axes[i,j],label='$\mu-2\sigma$')
↳\sigma$')
axes[i,j].legend(bbox_to_anchor=(1.05, 1),
↳loc=2,shadow=True,frameon=True)

plt.tight_layout()

```

