

covCorr

December 14, 2021

1 Calculando a variância e covariância de uma matriz de dados

```
[2]: import numpy as np
```

1.1 1.Dados

```
[3]: Data=np.array([[2.5, 2.4],
                    [0.5, 0.7],
                    [2.2, 2.9],
                    [1.9, 2.2],
                    [3.1, 3.0],
                    [2.3, 2.7],
                    [2.0, 1.6],
                    [1.1, 1.0],
                    [1.5, 1.6],
                    [1.1, 0.9]],dtype=float)
(n,d) = np.shape(Data)
```

1.2 2.Média

```
[40]: u=np.mean(Data, axis=0) #media de cada coluna
u
```

```
[40]: array([1.82, 1.9 ])
```

1.3 3.Desvio médio

```
[42]: Dm = Data-u
Dm
```

```
[42]: array([[ 0.68,  0.5 ],
            [-1.32, -1.2 ],
            [ 0.38,  1.  ],
            [ 0.08,  0.3 ],
            [ 1.28,  1.1 ],
            [ 0.48,  0.8 ],
            [ 0.18, -0.3 ])
```

```
[-0.72, -0.9 ],  
[-0.32, -0.3 ],  
[-0.72, -1.  ]])
```

1.4 4. Matriz de Covariância

Sendo,

σ_{ij} : covariância entre a variável X_i e a variável X_j .

σ_{ii} é a variância

σ_i é o desvio padrão $\sigma_i = \sqrt{\sigma_{ii}}$

1.4.1 Os Dados que usamos tem duas dimensões (duas variáveis ou campos)

1.4.2 Calculando

```
[46]: Cv = (Dm.T@Dm)/(n-1)  
Cv
```

```
[46]: array([[0.59955556, 0.61444444],  
           [0.61444444, 0.73555556]])
```

1.5 5. Variância, os elementos da diagonal principal

```
[47]: Var=np.diag(Cv)  
Var
```

```
[47]: array([0.59955556, 0.73555556])
```

1.6 6. Desvio padrão, a raiz quadrada da variância

```
[48]: dp=np.sqrt(Var)  
dp
```

```
[48]: array([0.77430973, 0.85764536])
```

1.7 7. Utilizando o desvio padrão como uma matriz diagonal

```
[49]: diagDp=np.diag(dp)  
diagDp
```

```
[49]: array([[0.77430973, 0.          ],  
           [0.          , 0.85764536]])
```

1.8 8.O método reshape para transformar um vetor em uma matriz.

Para um array provindo de uma lista simples, o este array é um vetor com uma dimensão. Mas um vetor unidimensional não é uma matriz. O método shape retorna uma tupla do tipo (d,_), indicando que não sabe quantos elementos tem na segunda dimensão. Se usarmos o método ndim, teremos a dimensão 1 e o método size nos dará o número de elementos. Com isso podemos transformar este array em um vetor (vetor coluna) ou vetor linha, com o método reshape. A transformação é necessária para operações matriciais. **Lembrando que algebricamente um vetor é uma matriz com n linha e uma coluna.**

```
[50]: l=dp.ndim
      c=dp.size
      dp=dp.reshape(1,c)
      print("Vetor de desvios padrões em formato matricial (vetor linha)")
      print(dp)
      print("Formato do vetor")
      print(dp.shape)
```

```
Vetor de desvios padrões em formato matricial (vetor linha)
[[0.77430973 0.85764536]]
Formato do vetor
(1, 2)
```

1.9 9.Repetindo o vetor de desvios padrões para operações matriciais

```
[52]: multDp = np.ones((c,1))@dp
      print("Matriz com duas linhas de desvio padrão")
      print(multDp)
```

```
Matriz com duas linhas de desvio padrão
[[0.77430973 0.85764536]
 [0.77430973 0.85764536]]
```

1.10 10.Criado a matriz de copadrões

Multiplica a diagonal de padrões com as linhas de padrões

```
[53]: Cp=diagDp@multDp
      print("Matriz de copadrão")
      print(Cp)
```

```
Matriz de copadrão
[[0.59955556 0.66408314]
 [0.66408314 0.73555556]]
```

1.11 11.Encontrando a matriz de correlação

Onde,

ρ_{ij} : é a correlação entre a variável i e j

σ_{ij} : é a covariância entre a variável i e j

σ_i : é o desvio padrão da variável i , a raiz quadrada da variância σ_{ii}

```
[54]: Corr=Cv/Cp  
print("Matriz de correlação")  
print(Corr)
```

Matriz de correlação

```
[[1.          0.92525228]  
 [0.92525228 1.          ]]
```

Repare que a correlação de ρ_{ii} é sempre 1. Na matriz de covariância são as diagonais da matriz.