UNIVERSIDADE FEDERAL DE SANTA MARIA
CACHOEIRA DO SUL
CURSO SUPERIOR DE BACHARELADO EM CIÊNCIA DE DADOS E INTELIGÊNCIA ARTIFICIAL

Thomáz França

LEVANTAMENTO DE DADOS UTILIZANDO NUMPY

Santa Maria, RS

Thomáz França

LEVANTAMENTO DE DADOS UTILIZANDO NUMPY

ORIENTADOR: Prof. Adriano Quiliao

Santa Maria, RS

Thomáz França

LEVANTAMENTO DE DADOS UTILIZANDO NUMPY

Aprovado em de de :

Adriano Quiliao, MSc. (UFSM)
(Presidente/Orientador)

Santa Maria, RS

LISTA DE FIGURAS

Figura 2.1 – Código Principal 5
Figura 2.2 – Código de Classe 6
Figura 3.1 – Declaração da função 7
Figura 3.2 – Declaração da função 7
Figura 3.3 – Declaração da função 8
Figura 3.4 – Declaração da função 8
Figura 3.5 – Cálculo para normalização 9
Figura 3.6 – Declaração da função 9
Figura 3.7 – Declaração da função 9
Figura 3.8 – Declaração da função 10
Figura 3.9 – Declaração da função 10

SUMÁRIO

1	**INTRODUÇÃO**	4
2	**MÓDULOS DO SOFTWARE**	5
2.1	CÓDIGO PRINCIPAL	5
2.2	CLASSE	5
3	**ESTATÍSTICAS DESCRITIVAS**	7
3.1	PRODUÇÃO TOTAL ANUAL DE CADA FAZENDA	7
3.2	MÉDIA DE PRODUÇÃO DE CADA MÊS	7
3.3	FAZENDA MAIS PRODUTIVA	8
3.4	MÊS COM MAIOR PRODUÇÃO TOTAL ACUMULADA	8
3.5	NORMALIZAÇÃO PARA INTERVALO DE 0 ATÉ 1	8
3.6	FAZENDAS COM PRODUÇÃO ACIMA DA MÉDIA ANUAL	9
3.7	VARIAÇÃO PERCENTUAL	9
3.8	DESVIO PADRÃO	9
4	**CONCLUSÃO**	11

1 INTRODUÇÃO

O objetivo deste trabalho é realizar uma análise estátistica de dados sintéticos utilizando de funções da biblioteca NUMPY em Python. Foram simulados valores que representam a produção mensal de 50 fazendas. Serão documentados seus resultados e feitos os apontamentos necessários.

2 MÓDULOS DO SOFTWARE

2.1 CÓDIGO PRINCIPAL

Figura 2.1 – Código Principal

```python
import numpy as np
import Classe as cl
def gerar_matriz_producao(matricula=202513298, fazendas=50, meses=12):
    np.random.seed(matricula)
    matriz = np.zeros((fazendas, meses))
    for fazenda in range(50):
        base = 80 + (matricula % 100) + fazenda * 2
        variacao = np.random.normal(0, 10, 12)
        matriz[fazenda, :] = np.maximum(base + variacao, 20)
    return matriz

arr = cl.SistemaProducao(gerar_matriz_producao())
print(arr.producao_total())
print(arr.media_total())
print(arr.fazenda_mais_produtiva())
print(arr.mes_mais_produtivo())
print(arr.normatizacao())
print(arr.filtro_maior_media())
print(arr.variacao())
print(arr.desvio())
```

Fonte: Autor.

2.2 CLASSE

Figura 2.2 – Código de Classe

```python
import numpy as np
import time
class SistemaProducao:
    def __init__(self, matriz):
        self.__matriz = matriz

    @property
    def get_matriz(self):
        return self.__matriz

    def producao_total(self):
        return np.sum(self.__matriz, axis=1)

    def media_total(self):
        return np.mean(self.__matriz, axis=0)

    def fazenda_mais_produtiva(self):
        return np.argmax(self.producao_total())

    def mes_mais_produtivo(self):
        return np.argmax(self.media_total())

    def normatizacao(self):
        return (self.__matriz-np.min(self.__matriz))/(np.max(self.__matriz)-np.min(self.__matriz))

    def filtro_maior_media(self):
        return self.producao_total()[self.producao_total() > (np.mean(self.producao_total()))]

    def variacao(self):
        return ((self.__matriz[:, -1] - self.__matriz[:, 0]) / self.__matriz[:, 0]) * 100

    def desvio(self):
        desvio = np.std(self.__matriz, axis=1)
        return np.argmin(desvio), desvio[np.argmin(desvio)]
```

Fonte: Autor.

3 ESTATÍSTICAS DESCRITIVAS

No presente capítulo serão expostos os testes e analises realizadas.

3.1 PRODUÇÃO TOTAL ANUAL DE CADA FAZENDA

Para declaração do método que retorna a produção total anual de cada fazenda, foi utilizado a função np.sum que faz a soma de todos os elementos do array. Nessa situação foi necessário o complemento axis = 1, que faz com que a soma seja realizada percorrendo a dimensão que representa as produção de cada fazenda individualmente, retornando uma matriz 50x1.

Figura 3.1 – Declaração da função

```
def producao_total(self):
    return np.sum(self.__matriz, axis=1)
```

Fonte: Autor.

3.2 MÉDIA DE PRODUÇÃO DE CADA MÊS

Para declaração do método que retorna a média de produção de cada mês, foi utilizado a função np.mean que obtém a média array. Nessa situação foi necessário o complemento axis = 0, que faz com que as médias sejam realizadas percorrendo a dimensão que representa cada mês com suas respectivas médias, retornando uma matriz 12x1 com valores médios de cada mês.

Figura 3.2 – Declaração da função

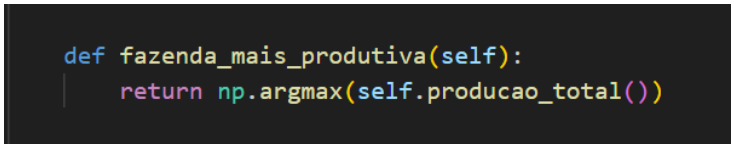
```
def media_total(self):
    return np.mean(self.__matriz, axis=0)
```

Fonte: Autor.

3.3 FAZENDA MAIS PRODUTIVA

Para declaração do método que retorna a fazenda mais produtiva do ano, foi utilizado a função np.argmax que obtém o valor máximo de determinada array. Nessa situação ela recebe a própria função de produção total anual de cada fazenda que retorna a soma de todos os meses de cada fazenda individualmente, tendo a função de apenas selecionar o maior valor entre elas e retornar seu indice.

Figura 3.3 – Declaração da função

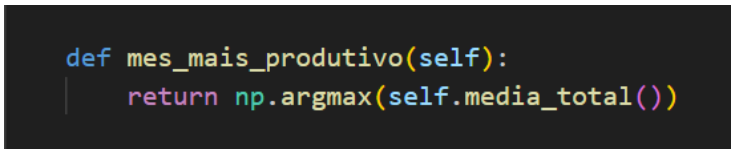


Fonte: Autor.

3.4 MÊS COM MAIOR PRODUÇÃO TOTAL ACUMULADA

Para declaração do método que retorna a fazenda mais produtiva do ano, foi utilizado a função np.argmax como anteriormente, porém nessa situação ela recebe a função de media total mensal, retornado apenas o indice do mês de maior produção total.

Figura 3.4 – Declaração da função



Fonte: Autor.

3.5 NORMALIZAÇÃO PARA INTERVALO DE 0 ATÉ 1

Para realizar o calculo de normalização foi utilizado a seguinte equação, sendo 'X' a matriz principal, com todas as produções de todas as fazendas, 'min' o menor valor dessa matriz, obtido através do método np.min() e 'max' o maior valor dessa matriz, sendo obtido através do método np.max(). Esses retornam uma matriz normalizadas com valores proporcionais aos originais, mas que variam de 0 a 1.

Figura 3.5 – Cálculo para normalização

$$norm = (x - min)/(max - min).$$

Fonte: Autor.

Figura 3.6 – Declaração da função

```
def normatizacao(self):
    return (self.__matriz-np.min(self.__matriz))/(np.max(self.__matriz)-np.min(self.__matriz))
```

Fonte: Autor.

3.6 FAZENDAS COM PRODUÇÃO ACIMA DA MÉDIA ANUAL

Para declaração da função que retorna uma matriz filtrada apenas com as fazendo que detem valores de produção acima da média geral, foi utilizado uma comparação de 'maior que' entre os valores de produção total anual de cada fazenda e a média dessa mesma produção, retornando uma mátriz de true e false, que posteriormente serve de filtro para a matriz dos valores de produção total anual, retornando assim as fazendas acima da média.

Figura 3.7 – Declaração da função

```
def filtro_maior_media(self):
    return self.producao_total()[self.producao_total() > (np.mean(self.producao_total()))]
```

Fonte: Autor.

3.7 VARIAÇÃO PERCENTUAL

Para declaração da função que retorna a variação percentual de cada fazenda, foi utilizado o cálculo padrão para esse tipo de análise, que consiste na subtração do primeiro elemento ([:,0]) com o último elemento ([:,-1]), e na divisão dessa subtração pelo primeiro elemento novamente. Após isso, basta multiplicar por 100 para obter o valor percentual.

3.8 DESVIO PADRÃO

Para declaração da função que retorna o indice do menor desvio padrão e seu respectivo desvio, foi utilizada a função np.std() de numpy, que retorna o desvio padrão,

Figura 3.8 – Declaração da função

```
def variacao(self):
    return ((self.__matriz[:, -1] - self.__matriz[:, 0]) / self.__matriz[:, 0]) * 100
```

Fonte: Autor.

que por sua vez é armazenado na variável 'desvio', pois é utilizado 3 vezes durantes a função. Após isso, recorremos a função np.argmin(), que retorna o indice do valor minimo de determinada array e aplicamos novamente np.std(), agora ja na variável, para obter também o desvio mínimo.

Figura 3.9 – Declaração da função

```
def desvio(self):
    desvio = np.std(self.__matriz, axis=1)
    return np.argmin(desvio), desvio[np.argmin(desvio)]
```

Fonte: Autor.

4 CONCLUSÃO

A biblioteca Numpy se destaca pela sua ampla gama de possibilidades em análise estátistica e descritiva, tratamento e manipulação de dados, sendo uma ferramenta essencial e indispensável na graduação em Ciência de Dados e Inteligência Artifícial.