INSTITUTO FEDERAL
Brasília
Campus Brasília

TECNOLOGIA EM SISTEMAS PARA INTERNET

Davi Pereira Speck Alves
Marcelo Maia Bentes
Müller Araújo do Vale
Prince Neres

RELATÓRIO DE PRÁTICA INTEGRADA
DE
CIÊNCIA DE DADOS E APRENDIZADO DE MÁQUINA

Brasília - DF

12/01/2022

Sumário

1. Objetivo geral

Desenvolver um programa capaz de realizar o estudo de um *Dataset* criado a partir de uma pulseira capaz de transmitir dados em forma de coordenadas (X, Y, Z) gerados por um acelerômetro acoplado. Dessa forma, sustentado nos conhecimentos aprendidos nas matérias de Introdução à Ciência de Dados e Introdução a Aprendizagem de Máquina, será realizado um estudo aprofundado no intuito de entender o que se encontra “escondido” por trás da superfície dos dados coletados.

1.1 Objetivos específicos

Para a proposta inicial, os objetivos específicos são:

- Organizar o Dataset;
- Explorar os dados coletados;
- Preparar o conjunto para uma análise mais profunda.

2. Descrição do problema

O estudo de uma extensa base de dados envolve diversas etapas no decorrer do caminho, como o planejamento, a organização e a resolução de problemas, por exemplo.

2.1 Introdução e coleta de dados

Inicialmente, o *Dataset* se encontra bagunçado e exige uma organização mais específica, buscando a definição de padrões para nomenclatura e inclusive de execução.

2.2 Exploração

Com os dados organizados e padrões definidos, é momento para explorar os dados coletados, buscando encontrar ligações entre os dados, através de gráficos, correlações e afins.

2.3 Preparação

Foi feito o estudo inicial por meio da exploração, agora é necessário preparar o ambiente para aprofundar as análises, abrindo ainda mais a possibilidade para a construção de fatos, ou seja, informações concretas geradas a partir dos dados coletados.

3. Desenvolvimento

Para desenvolver o projeto de forma que possa atender às necessidades de agora e futuras, foram definidas duas etapas:

- Análise da estrutura dos dados fornecidos para o projeto;
- Programação para que possam ser encontradas as informações fundamentais no projeto. Depois de encontrado, será gravado em um banco de dados do tipo SQLite para que se trabalhe adequadamente .

Foram utilizadas as seguintes tecnologias:

- Python: segundo o site Kenzie define python como " uma linguagem de programação de alto nível — ou High Level Language —, dinâmica, interpretada, modular, multiplataforma e orientada a objetos — uma forma específica de organizar softwares onde, a grosso modo, os procedimentos estão submetidos às classes, o que possibilita maior controle e estabilidade de códigos para projetos de grandes proporções. Nesse projeto a linguagem de programação Python deve o papel criar as funcionalidades necessárias para o desenvolvimento do projeto seja realizado com sucesso".

  As principais bibliotecas utilizadas no projetos:

  - Pandas: o site Medium define o pandas como "uma biblioteca para manipulação e análise de dados, escrita em Python. Essa é a biblioteca perfeita para iniciar suas análises exploratórias de dados, pois ela nos permite ler, manipular, agregar e plotar os dados em poucos passos";
  - *matplotlib.pyplot*: o site USP () define o matplotlib.pyplot como "uma biblioteca com recursos para a geração de gráficos 2D a partir de *arrays*. Gráficos comuns podem ser criados com alta qualidade a partir de comandos simples, inspirados nos comandos gráficos do MATLAB";
  - *seaborn*: o site minerando dados define como uma "atuante em cima do matplotlib e ajuda a melhorar o visual dos gráficos, dando

uma aparência mais bem acabada. Seguem alguns exemplos de como usar o *Seaborn* na geração de gráficos";

- *IPython.display*: o site Ulisboa define como "um ambiente interactivo para a utilização livre da linguagem Python como ferramenta de cálculo e visualização, bem como para o desenvolvimento de (pequenos) programas na linguagem Python e sua prototipagem rápida. A versatilidade do ambiente IPython facilita a integração com outros ambientes e linguagens de programação";

- SQLite: o site rockcontent define SQLite como "uma base de dados relacional de código aberto e que dispensa o uso de um servidor na sua atuação. Armazenando seus arquivos dentro de sua própria estrutura, ele é capaz de funcionar muito bem em aplicações diversas, principalmente, websites de tráfego médio e sistemas mobile. Foi analisado que nesse projeto um banco de dados é necessário pois, os metadados encontrados são difíceis de serem manipulados dentro de um array, lista, dicionário, variável entre outros. Sendo mais adequado um banco de dados onde não precise ser instalado para ser utilizado".

3.1 Código implementado

```
# Criar um menu para que o usuário possa selecionar as opções fornecidas
pelo programa

opt = 1

while (opt <= 7 or opt >= 1):

        # Menu de opções
        print('Digite uma das opcoes abaixo')
        print('1 - Analisa Diretorio HMP_Dataset')
        print('2 - Gerar Relatorios do tipo cvs')
        print('3 - Relatorio geral')
        print('4 - Correlacao entre as coordenadas')
        print('5 - Histograma revelando a distribuicao das medidas obtidas
nas coordenadas X')
```

```
        print('6 - Grafico com ocorrencias por tipo de movimento')
        print('7 - Finalizar programa')

        opt = int(input('Opcao escolhida:'))

        if opt == 1:
            analiseDiretorio()
        elif opt == 2:
            if if os.path.isfile('registro.db')
                GerarRelatorioCVS()
            else:
                print('Não existe o banco de dados')
                print('Por favor gere o banco de dados na opção 1')
                print('Tecle enter para volta ao menu.')
                input('')
        elif opt == 3:
            if os.path.isfile('registro.db'):

                df = pd.read_csv('Sprint_I\\relatorio\Relatoriogeral.csv')
                df = df.drop(columns=['Unnamed: 0'])
                display(df)
                print('Tecle enter para volta ao menu.')
                input('')
            else:
                print('Não existe o arquivo relatoriogeral.cvs')
                print('Por favor gere o arquivo na opção 2')
                print('Tecle enter para volta ao menu.')
                input('')
        elif opt == 4:
                                                                    if
os.path.isfile('Sprint_I/relatorio/tipomovimento_x_y_z.csv'):
                CorrelacaoCoordenada()
            else:
                print('Não existe o arquivo tipomovimento_x_y_z.cvs')
                print('Por favor gere o arquivo na opção 2')
                print('Tecle enter para volta ao menu.')
                input('')
        elif opt == 5:
            if os.path.isfile('Sprint_I/relatorio/Relatoriogeral.csv'):
                HistogramaMedidasX()
            else:
                print('Não existe o arquivo Relatoriogeral.cvs')
```

```
                print('Por favor gere o arquivo na opção 2')
                print('Tecle enter para volta ao menu.')
                input('')
        elif opt == 6:
            if os.path.isfile('Sprint_I/relatorio/Relatoriogeral.csv'):
                GraficoTipoMovimento()
            else:
                print('Não existe o arquivo relatoriogeral.cvs')
                print('Por favor gere o arquivo na opção 2')
                print('Tecle enter para volta ao menu.')
                input('')
        elif opt == 7:
            print('Saindo do programa!')
            return
        elif opt > 7 or opt < 1:
            print('Opcao inválida!')

        os.system('cls')

    os.system('cls')
```

```
# Função onde será criado o banco de dados para gravar os metadados encontrados na pasta 'Sprint_I\HMP_Dataset'. Todas as vezes que o programa é executado para analisar o diretório 'Sprint_I\HMP_Dataset'. O banco de dados é deletado para que possa ser gravado os dados de forma consistente.
        def analiseDiretorio():

            if os.path.isfile('registro.db'):
                os.remove("registro.db")

            banco = Connect()

            pasta = './Sprint_I/HMP_Dataset'
            tipomovimento = ''
            model = False
            data = ''
            hora = ''
            sexo = 0
            quantidade = 0
            idtipomovimento = 0
```

```
        idcoleta = 0
        x = 0
        y = 0
        z = 0

        for diretorio, subpastas, arquivos in os.walk(pasta):

            for subpasta in subpastas:

                tipomovimento = subpasta.replace('_MODEL', '')

                if subpasta.find('_MODEL') != -1:
                    model = True
                else:
                    model = False

                print(tipomovimento)

                banco.inserir_registro("""
                        INSERT INTO tipomovimento (descricao,model)
                        VALUES ('{}','{}');
                        """.format(tipomovimento, model))

                idtipomovimento = banco.ler_registro(
                                    'SELECT  MAX(idtipomovimento)  FROM
tipomovimento;')[0]

                        for  arquivo  in  os.listdir(os.path.join(pasta,
subpasta)):
                    for i, valor in enumerate(arquivo.replace('.txt',
'').split('-')):
                        if i > 0 and i < 4:
                                        data  =  valor  if  i  ==  1  else
'{}-{}'.format(data, valor)
                        elif i > 3 and i < 7:
                                        hora  =  valor  if  i  ==  4  else
'{}:{}'.format(hora, valor)
                        elif i == 8:
                            sexo = 'f' if valor[0] == 'f' else 'm'

                            quantidade = valor[1:].replace('_', '')
```

```
                    banco.inserir_registro("""
                                                INSERT INTO coleta
(idsexo,idtipomovimento,data,hora,quantidade)
                                VALUES ({},{},'{}','{}',{});
                                 """.format(1 if sexo == 'f' else 2,
idtipomovimento, data, hora, quantidade))

                    idcoleta = banco.ler_registro(
                        "SELECT MAX(idcoleta) FROM coleta;")[0]

                    dados = pd.read_table('{}'.format(os.path.join(
                                 pasta, subpasta, arquivo)), sep=" ",
header=None, names=["X", "Y", "Z"])

                    ini = time.time()
                    valor = ''

                    for i in range(len(dados)):
                        if valor:
                            valor = '{},({},{},{},{})'.format(
                                        valor, idcoleta, dados['X'][i],
dados['Y'][i], dados['Z'][i])
                        else:
                            valor = '({},{},{},{})'.format(
                                               idcoleta, dados['X'][i],
dados['Y'][i], dados['Z'][i])

                    banco.inserir_registro("""
                        INSERT INTO dados (idcoleta,x,y,z)
                        VALUES {};
                        """.format(valor))

                    fim = time.time()
                       print("Tipo de Movimento {} para o arquivo {}
levou {} para se executado: ".format(
                           tipomovimento, os.path.join(pasta, subpasta,
arquivo), fim-ini))
```

```
# Função para gerar os arquivos .cvs onde os dados são retirados do banco de dados SQLite para serem gravados. Hoje temos 25 tipos de relatórios para análise. Caso exista algo no diretório relatório, será deletado permanentemente.
def GerarRelatorioCVS():

    banco = Connect()

    # Definindo relatório geral
    resultado = banco.ler_registros("""
                        SELECT
                        tipomovimento.descricao,tipomovimento.model,
                        coleta.data,coleta.hora,coleta.quantidade,
                        sexo.sexo,sexo.sigla,
                             dados.x,dados.y,dados.z, (dados.x + dados.y + dados.z)/3
                        FROM coleta
                                                INNER JOIN tipomovimento ON (tipomovimento.idtipomovimento = coleta.idtipomovimento)
                        INNER JOIN sexo ON (sexo.idsexo = coleta.idsexo)
                                    INNER JOIN dados ON (dados.idcoleta = coleta.idcoleta)
                        WHERE tipomovimento.model == 'False';
                        """)

    df = pd.DataFrame(resultado, columns=[
                                    "TipoMovimento", "Model", "Data", "Hora", "Quantidade", "Sexo", "Sigla", "X", "Y", "Z", "Media"])

    df.to_csv('Sprint_I/relatorio/RelatorioGeral.csv')

    # Tipo de Movimento
    resultado = banco.ler_registros("""
                        SELECT
                        tipomovimento.descricao,tipomovimento.model
                        FROM
                        tipomovimento;
                        """)

    df = pd.DataFrame(resultado, columns=["TipoMovimento", "Model"])

    df.to_csv('Sprint_I/relatorio/tipomovimento.csv')
```

```
# Tipo de Movimento -->  model true
resultado = banco.ler_registros("""
                    SELECT
                    tipomovimento.descricao,tipomovimento.model
                    FROM
                    tipomovimento
                    WHERE
                    tipomovimento.model = 'True';
                    """)

df = pd.DataFrame(resultado, columns=["TipoMovimento", "Model"])

df.to_csv('Sprint_I/relatorio/tipomovimento_model_true.csv')

# Tipo de Movimento --> model False
resultado = banco.ler_registros("""
                    SELECT
                    tipomovimento.descricao,tipomovimento.model
                    FROM
                    tipomovimento
                    WHERE
                    tipomovimento.model = 'False';
                    """)

df = pd.DataFrame(resultado, columns=["TipoMovimento", "Model"])

df.to_csv('Sprint_I/relatorio/tipomovimento_model_false.csv')
```

Modelagem do banco de dados para gravar os metadados. Foi pensando em gravar todos os metadados encontrados no diretório, e havendo a necessidade poderá se construído as informações necessárias para se analisada.

Link para acessar o repositório no servidor GitHub https://github.com/infocbra/pratica-integrada-cd-e-am-2021-2-g6-dmpm.

4. Considerações finais

Esse trabalho pretendeu analisar o quanto foram desenvolvidas as habilidades dos alunos ao aplicar o conhecimento aprendido nas matérias de Introdução à Ciência de Dados e Introdução a Aprendizagem de Máquinas. Até o momento, apesar dos problemas encontrados no meio do caminho, todos os objetivos foram cumpridos com total satisfação. **(Continuar desenvolvimento a partir das próximas sprints.**

Referências

The pandas development team. **pandas documentation**. Pandas, 2008. Disponível em: https://pandas.pydata.org/docs/index.html. Acesso em: 09/01/2022.

RIBEIRO, Lucas. **Introdução a Biblioteca Pandas**. Medium, 2020. Disponível em: https://medium.com/tech-grupozap/introdução-a-biblioteca-pandas-89fa8ed4fa38. Acesso em: 27/12/2021.

ROVEDA, Ugo. **O que é Python, para que serve e por que aprender?** Kenzie, 2013. Disponível em: https://kenzie.com.br/blog/o-que-e-python/. Acesso em: 30/12/2021.

DE SOUZA, Ivan. **O que é SQLite, porque ele é usado, e o que o diferencia do MySQL?** Rockcontent, 2020. Disponível em: https://rockcontent.com/br/blog/sqlite/ Acesso em: 03/01/2022.

DCC IME USP. **Visualização de gráficos 2D usando matplotlib**. USP, 2019. Disponível em: https://panda.ime.usp.br/algoritmos/static/algoritmos/10-matplotlib.html#:~:text=O%20matplotlib%20é%20uma%20biblioteca,nos%20comandos%20gráficos%20do%20MATLAB. Acesso em: 26/01/2022.

Adminvooo. **Biblioteca Seaborn com o matplotlib**. Vooo, 2022. Disponível em: https://www.vooo.pro/insights/biblioteca-seaborn-com-o-matplotlib/. Acesso em: 07/01/2022.

CALEIRO, Carlos e RAMOS, Jaime. **Introdução à Programação em Python**. Ulisboa, 2019. Disponível em: https://www.math.tecnico.ulisboa.pt/~ccal/python/nb01.html. Acesso em: 20/12/2021.