PROJETO INTEGRADOR DE PROGRAMAÇÃO

Relatório de Atividades Desenvolvidas

André Luiz Dutra Da Silva
Cesar Antônio Gonsiorkiewicz Simi Esteche
Denes Dos Santos Barbosa
Vitoria De Fatima Stadler

Relatório de Projeto apresentado à Universidade Estadual do Centro-Oeste – Unicentro/Cedeteg, como quesito para conclusão do 1º semestre do Curso de Tecnologia em Big Data no Agronegócio na disciplina de Projeto Integrador ministrada pelos professores Carolina Paula de Almeida e Richard Aderbal Gonçalves.

Guarapuava
2024

Introdução

O projeto integrador Ceva D'água visa verificar o impacto da precipitação na produção de cevada no município de Guarapuava/PR, foi desenvolvido para solucionar problemas relacionados ao agronegócio por meio da coleta, processamento e análise de dados climáticos e de produção agrícola. Voltado para agricultores e empresas do setor, o projeto busca fornecer informações vitais e insights valiosos sobre clima e padrões de produção, auxiliando na tomada de decisões estratégicas. A importância deste projeto reside na sua capacidade de melhorar a produção agrícola, reduzir o desperdício e aumentar a eficiência agrícola. A solução proposta utiliza técnicas de programação, web scraping e análise de dados para coletar e processar informações relevantes de diversas fontes.

Nosso projeto é para buscar o volume de chuva dos últimos 20 anos, fazer uma média de chuva para a região de Guarapuava e ver o seu impacto da precipitação na produção de cevada no município de Guarapuava/PR**.**Assim mostrando ao cliente qual o melhor tempo para plantar baseado nos últimos anos analisados.

1. Objetivos

1.1 Objetivo Geral

Desenvolver um web scraping, que irá coletar, processar e analisar dados climáticos e de produção agrícola para ajudar agricultores e empresas industriais a tomar decisões estratégicas, visando otimizar a produção e aumentar a eficiência.

1.2 Objetivos Específicos

- Coletar dados climáticos históricos e de produção agrícola de fontes confiáveis.
- Desenvolver técnicas de web scraping utilizando Python e bibliotecas como Selenium e BeautifulSoup.
- Tratar e armazenar os dados coletados em formatos adequados para análise.
- Visualizar os dados tratados por meio de gráficos e tabelas.
- Desenvolver habilidades de programação em C++, Python e uso de ferramentas de versionamento como Git.
- Utilizar ferramentas de gestão de projetos como Trello para organizar as atividades da equipe.

2. Metodologia

Para alcançar os objetivos traçados, a equipe seguiu os seguintes passos:

2.1 Coleta de Dados com programação em *python*

Foram utilizados os seguintes links para a coleta de dados:

- World Weather Online para dados climáticos (https://www.worldweatheronline.com/guarapuava-weather-history/parana/br.aspx).
- IBGE para dados de produção agrícola (https://sidra.ibge.gov.br/tabela/6588#/n3/41/v/36/p/all/c48/39435/l/v,p+c48,t/resultado).

Os recursos da linguagem Python utilizados incluíram:

- Bibliotecas: pandas, selenium, BeautifulSoup
- Técnicas: Web scraping com selenium e BeautifulSoup, processamento e armazenamento de dados com pandas.

Exemplo de código utilizado:

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import pandas as pd
import time
from datetime import datetime, timedelta

def iniciar_driver():
    service = Service()
    options = webdriver.ChromeOptions()
    return webdriver.Chrome(service=service, options=options)

def obter_dados_chuva(driver, data_inicio, data_fim):
    url = "https://www.worldweatheronline.com/guarapuava-weather-history/parana/br.aspx"
    driver.get(url)
    elementos_agrupados = [[] for _ in range(365)]
    cont = 0
    while data_inicio <= data_fim:
        search_field = driver.find_element(By.ID, 'ctl00_MainContentHolder_txtPastDate')
```

```
        search_field.clear()
        search_field.send_keys(data_inicio.strftime('%d/%m/%Y'))
        search_button = driver.find_element(By.ID, 'ctl00_MainContentHolder_butShowPastWeather')
        search_button.click()
        time.sleep(5)
        WebDriverWait(driver, 25).until(EC.presence_of_element_located((By.XPATH, '//div[@class="col
mr-1" and contains(text(), "mm")]')))
        elementos = driver.find_elements(By.XPATH, '//div[@class="col mr-1" and contains(text(),
"mm")]')
        for i in elementos:
            elementos_agrupados[cont].append(i.text[:-3])
        data_inicio += timedelta(days=1)
        cont += 1
    return elementos_agrupados

def salvar_dados(elementos_agrupados):
    with open("dados_chuva.txt", "w") as arq:
        for dia in elementos_agrupados:
            arq.write(','.join(dia))
            arq.write('\n')

def processar_dados(elementos_agrupados):
    elementos_anos_c = []
    for i in elementos_agrupados:
        if len(i) == 16:
            sub_lista = i[4:-1]
        elif len(i) == 15:
            sub_lista = i[4:]
        else:
            continue
        elementos_anos_c.append(sub_lista)
    valores_ano = [[] for _ in range(len(elementos_anos_c[0]))]
    for lista in elementos_anos_c:
        for i, elemento in enumerate(lista):
            valores_ano[i].append(elemento)
    return valores_ano

def agrupar_por_mes(valores_ano):
    def cria_mes(valores_ano):
        data_1 = datetime(2023, 1, 1)
        data_2 = datetime(2023, 12, 31)
        lista_meses = []
        mes = data_1.strftime("%m")
        dias = []
        while data_1 <= data_2:
            for i in valores_ano:
```

```
                dias.append(i)
                data_1 += timedelta(days=1)
                if mes != data_1.strftime("%m"):
                    lista_meses.append(dias)
                    dias = []
                    mes = data_1.strftime("%m")
            if dias:
                lista_meses.append(dias)
        return lista_meses
    return [cria_mes(i) for i in valores_ano]

def calcular_media_mensal(meses_certo):
    media_meses = []
    for ano in meses_certo:
        for mes in ano:
            media = sum(float(dia) for dia in mes) / len(mes)
            media_meses.append(round(media, 2))
    return media_meses

def agrupar_por_ano(media_meses_round):
    lista_anos = []
    tamanho_grupo = 12
    for i in range(0, len(media_meses_round), tamanho_grupo):
        grupo = media_meses_round[i:i + tamanho_grupo]
        lista_anos.append(grupo)
    return lista_anos

def main():
    driver = iniciar_driver()
    data_inicio = datetime(2023, 1, 1)
    data_fim = datetime(2023, 12, 31)
    elementos_agrupados = obter_dados_chuva(driver, data_inicio, data_fim)
    salvar_dados(elementos_agrupados)
    valores_ano = processar_dados(elementos_agrupados)
    meses_certo = agrupar_por_mes(valores_ano)
    media_meses_round = calcular_media_mensal(meses_certo)
    lista_anos = agrupar_por_ano(media_meses_round)
    df = pd.DataFrame(media_meses_round)
    df.to_csv('chuva_mes.csv', index=False, header=False)
    print(lista_anos)

if __name__ == "__main__":
    main()
```

2.2 Tratamento de Dados

Após a coleta, os dados foram tratados para remover inconsistências e formatar de maneira adequada para a análise. As técnicas utilizadas incluíram:

- Remoção de dados duplicados e inválidos.
- Conversão de tipos de dados.
- Agrupamento e cálculo de médias mensais e anuais.
- Armazenamento em arquivos CSV para facilitar a visualização e análise posterior.

Exemplo de Código:

```
import pandas as pd

def processar_dados(elementos_agrupados):
    elementos_anos_c = []
    for i in elementos_agrupados:
        if len(i) == 16:
            sub_lista = i[4:-1]
        elif len(i) == 15:
            sub_lista = i[4:]
        else:
            continue
        elementos_anos_c.append(sub_lista)
    valores_ano = [[] for _ in range(len(elementos_anos_c[0]))]
    for lista in elementos_anos_c:
        for i, elemento in enumerate(lista):
            valores_ano[i].append(elemento)
    return valores_ano

def agrupar_por_mes(valores_ano):
    def cria_mes(valores_ano):
        data_1 = datetime(2023, 1, 1)
        data_2 = datetime(2023, 12, 31)
        lista_meses = []
        mes = data_1.strftime("%m")
        dias = []
        while data_1 <= data_2:
            for i in valores_ano:
                dias.append(i)
                data_1 += timedelta(days=1)
                if mes != data_1.strftime("%m"):
                    lista_meses.append(dias)
```

```
                    dias = []
                    mes = data_1.strftime("%m")
            if dias:
                lista_meses.append(dias)
        return lista_meses
    return [cria_mes(i) for i in valores_ano]

def calcular_media_mensal(meses_certo):
    media_meses = []
    for ano in meses_certo:
        for mes in ano:
            media = sum(float(dia) for dia in mes) / len(mes)
            media_meses.append(round(media, 2))
    return media_meses

def agrupar_por_ano(media_meses_round):
    lista_anos = []
    tamanho_grupo = 12
    for i in range(0, len(media_meses_round), tamanho_grupo):
        grupo = media_meses_round[i:i + tamanho_grupo]
        lista_anos.append(grupo)
    return lista_anos

def main():
    elementos_agrupados = ...  # Carregar dados
    valores_ano = processar_dados(elementos_agrupados)
    meses_certo = agrupar_por_mes(valores_ano)
    media_meses_round = calcular_media_mensal(meses_certo)
    lista_anos = agrupar_por_ano(media_meses_round)
    df = pd.DataFrame(media_meses_round)
    df.to_csv('chuva_mes.csv', index=False, header=False)
    print(lista_anos)

if __name__ == "__main__":
    main()
```

3. Considerações Finais

Os objetivos do projeto foram atingidos com sucesso, resultando em uma ferramenta eficaz para coleta e análise de dados climáticos e de produção agrícola. Os resultados alcançados demonstraram a capacidade da equipe em aplicar técnicas de programação e análise de dados para resolver problemas reais do agronegócio.

***Campus* Santa Cruz:** Rua Padre Salvador, 875 – Fone: (42) 3621-1000 – FAX: (42) 3621-1090 – CEP 85.015-430 – GUARAPUAVA – PR
***Campus* Cedeteg:** Alameda Élio Antonio Dalla Vecchia, 838 – Fone/FAX: (42) 3629-8100 – CEP 85.040-167 – GUARAPUAVA – PR
***Campus* de Irati:** Rua Professora Maria Roza Zanon de Almeida s/n – Fone: (42) 3421-3000 – FAX: (42) 3421-3067 – CEP 84.505-677 – IRATI – PR

Pontos fortes:

- Utilização de técnicas avançadas de programação e web scraping.
- Processamento eficiente dos dados coletados.
- Visualização clara e acessível dos resultados.

Pontos fracos:

- Dependência de fontes de dados externas que podem mudar sua estrutura.
- Necessidade de melhorias na interface de usuário para facilitar o uso por parte dos agricultores.

Visão para aprimoramentos:

- Desenvolvimento de uma interface gráfica para facilitar a interação dos usuários com a aplicação.
- Expansão do projeto para incluir mais variáveis de interesse agrícola.
- Implementação de algoritmos de machine learning para previsões mais precisas.

4. Referências Bibliográficas

- World Weather Online. Disponível em: World Weather Online. (https://www.worldweatheronline.com).
- IBGE. Disponível em: IBGE. (https://sidra.ibge.gov.br).

APÊNDICES

A.1. Matrizes de habilidades

MATRIZ DE HABILIDADES – INÍCIO DO PROJETO				
HABILIDADES	ANDRÉ	CESAR	DENES	VITÓRIA
C++	1	3	2	2
PYTHON	1	3	1	1
SELENIUM	1	3	1	1
BEAUTIFUL SOAP	1	3	1	1
DEV C ++	1	3	2	3
TRELLO	1	3	2	3
CULTURA DE CEVADA	5	3	2	2
GITLAB	1	2	1	1
TOTAL DE PONTOS	12	19	12	14
TOTAL DA EQUIPE	57			

MATRIZ DE HABILIDADES – TÉRMINO DO PROJETO				
HABILIDADES	ANDRÉ	CESAR	DENES	VITÓRIA
C++	3	4	3	3
PYTHON	3	4	2	2
SELENIUM	3	5	2	2
BEAUTIFUL SOAP	3	5	2	2
DEV C ++	3	4	3	4
TRELLO	4	4	3	4
CULTURA DE CEVADA	5	3	4	2
GITHUB	1	3	3	3
TOTAL DE PONTOS	22	32	22	22
TOTAL DA EQUIPE	98			