Sistemas Operacionais e Computação em Nuvem

Implementação de um algoritmo de IA

Guilherme Alves de Oliveira e Oliveira

João Pedro Lima Paulo

Lara Marina de Oliveira

Giulia Nogueira Lopes de Sá

Beatriz Castilho Ferreira

Sistema de Monitoramento de Recursos

Implementação

O sistema de monitoramento foi implementado em Python utilizando as seguintes bibliotecas:

- Psutil -> Para a coleta de dados de CPU e memória
- Pandas -> para armazenamento e manipulação dos Dados
- Matplotlib -> para a visualização dos dados

Criando uma instância EC2 no AWS, via comando touch, foram criados os arquivos .py dentro da instância, de modo que usando o comando vim, pudemos editar os arquivos.

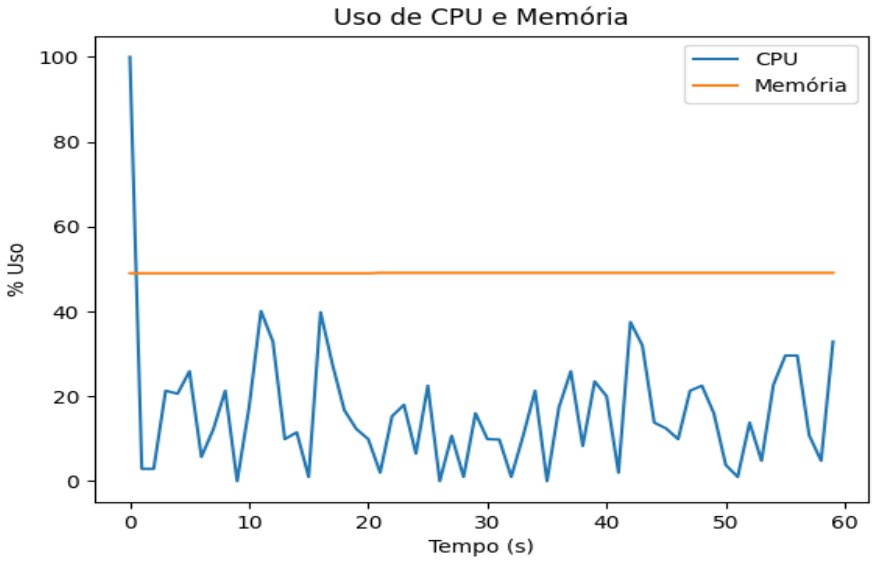
Codigo para a coleta dos dados de monitoramento

```
data = []
for _ in range(60):
    cpu = psutil.cpu_percent()
    mem = psutil.virtual_memory().percent
    timestamp = datetime.now().strftime("%H:%M:%S")
    data.append([timestamp, cpu, mem])
    time.sleep(1)

# Salvando dados em CSV
df = pd.DataFrame(data, columns=["Time", "CPU", "Memory"])
df.to_csv("monitoramento.csv", index=False)
```

Resultados do Monitoramento

O gráfico a seguir mostra o resultado do Exemplo 1, onde temos uma base fictícia de precificação de projetos, podemos fazer a predição de preços usando LinearSVC( ).

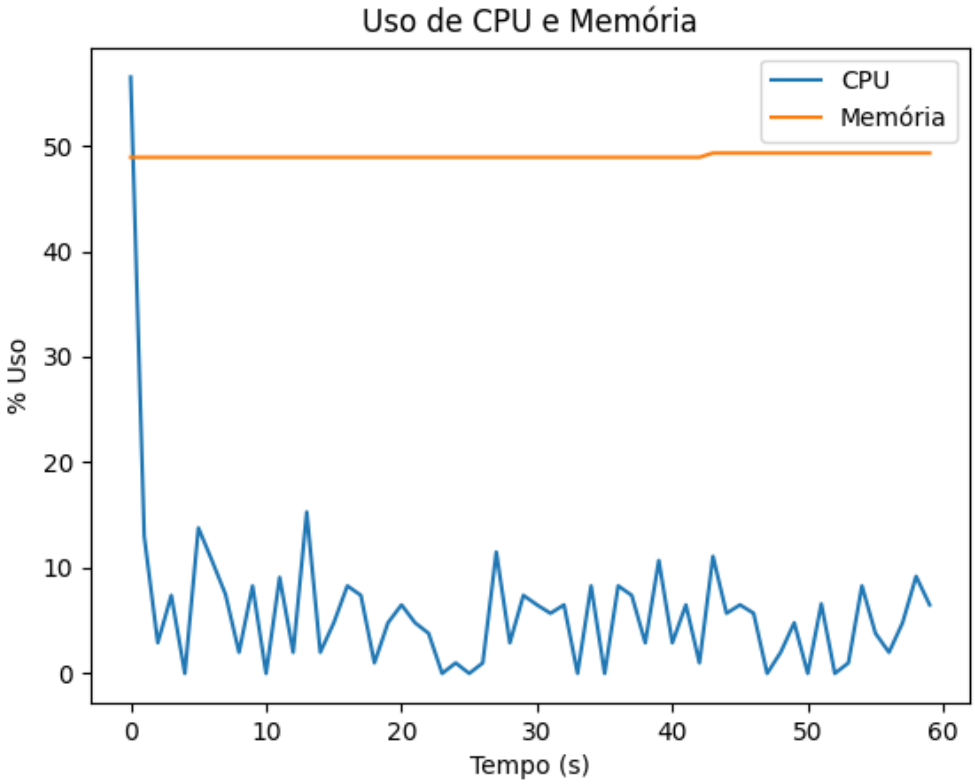


E pelo comando Htop, vemos o uso em tempo real da aplicação.

```
51620 ec2-user    20   0  622M  169M 64368 S   0.0 17.9  0:01.57 python exemplo1.py
```

Observamos um pico de uso de CPU no início do código, e ao longo de 60 segundos, houveram variações de acordo com a leitura da base e geração de métricas.

O gráfico seguinte mostra o resultado do Exemplo 2, onde temos uma base fictícia sobre venda de carro online, aqui usamos um modelo não linear, temos o teste de acurácia de vários modelos e o DecisionTreeClassifier se mostrou o mais eficaz.

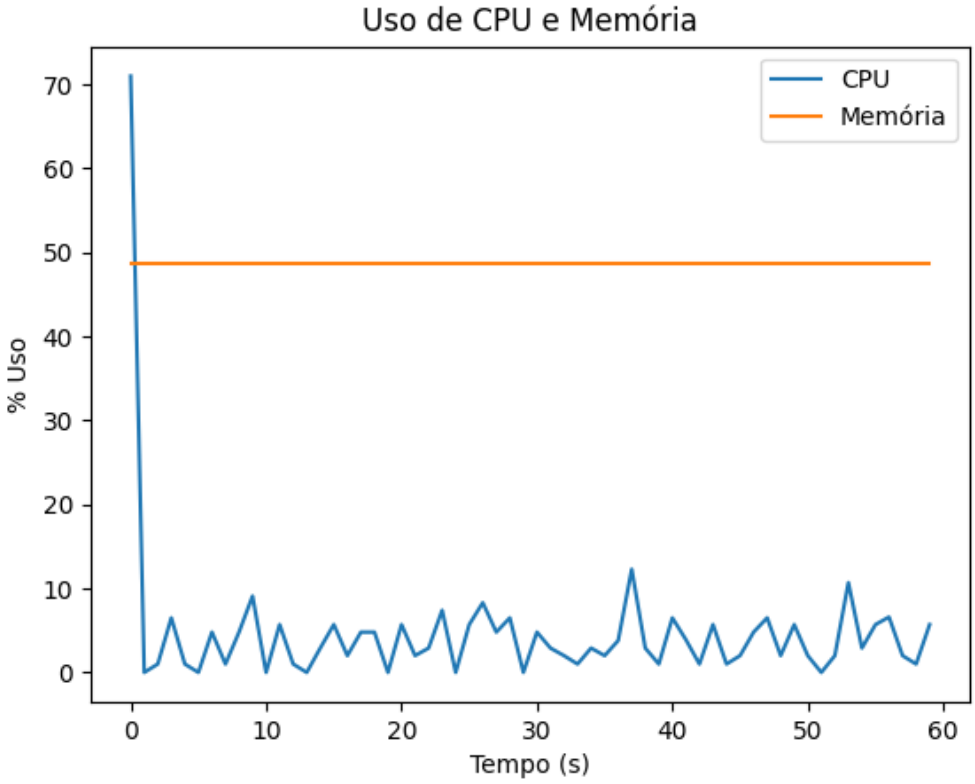


Aqui vemos o pico de uso ao iniciar

```
  PID USER      PRI  NI  VIRT   RES   SHR S  CPU%  MEM%   TIME+  Command
52048 ec2-user   20   0  655M  201M 65708 S   0.0 21.2  0:05.12 python exemplo_2.py
```

Vemos um Pico de CPU, mas logo abaixa, talvez mostrando que o uso do modelo de Árvores de decisão não pese tano para o sistema.

Por fim temos o gráfico 3, que temos uma base fictícia sobre inadimplência bancária, também usado para classificar modelos, temos como teste LogisticRegression( ), KNeighborsClassifier( ), DecisionTreeClassifier( ). Novamente a Árvore de Decisão é a escolhida para o uso devido à alta acurácia.



Conclusões

O sistema de monitoramento implementado foi eficaz na coleta e visualização dos dados de recursos da nuvem.

Entre os modelos testados, o Random Forest apresentou o melhor desempenho geral na tarefa de previsão de inadimplência, demonstrando a vantagem de algoritmos ensemble para problemas complexos.