Fundamentos do Azure Machine Learning

Princípios Básicos

O Azure Machine Learning oferece diversos recursos para facilitar o desenvolvimento, treinamento e gerenciamento de modelos de inteligência artificial. Esses recursos permitem desde a criação de um ambiente centralizado para experimentação até a disponibilização de infraestrutura escalável para execução de modelos.

Neste desafio, você deve associar corretamente cada recurso do Azure Machine Learning à sua funcionalidade correspondente.

Entrada.

A entrada consistirá no nome de um dos recursos disponíveis do Azure Machine Learning. Os seguintes valores são considerados válidos para este desafio de código:

1. “Azure ML Worksspace”
2. “Azure ML Compute”
3. “Azure ML Datasets”
4. “Azure ML Pipelines”

Saída

A saída esperada é a descrição associada ao recursos fornecido como entrada. Segue as saídas possíveis:

1. “Ambiente centralizado para gerenciar experimentos de Machine Learning”
2. “Recurso para provisionar Vms escaláveis para treinar modelos”
3. “Conjunto de dados organizados para uso nos experimentos de ML”
4. “Automação de fluxo de ML, organizando preparação e modelagem”

Exemplos

A tabela abaixo apresenta exemplos com alguns dados de entrada e suas respectivas saídas esperadas. Certifique-se de testar seu programa com esses exemplos e com outros casos possíveis.

|  |  |
| --- | --- |
| Entrada | Saída |
| Azure ML Worksapce | Ambiente centralizado para gerenciar experimentos de Machine Learning |
| Azure ML Compute | Recurso para provisionar Vms escaláveis para treinar modelos |
| Azure ML Datasets | Conjunto de dados organizados para uso nos experimentos de ML |

entrada = input()

def descrever\_recurso(recurso):

if recurso == "Azure ML Workspace":

return "Ambiente centralizado para gerenciar experimentos de Machine Learning"

elif recurso == "Azure ML Compute":

return "Recurso para provisionar VMs escaláveis para treinar modelos"

elif recurso == "Azure ML Datasets":

return "Conjunto de dados organizados para uso nos experimentos de ML"

elif recurso == "Azure ML Pipelines":

return "Automação do fluxo de ML, organizando preparação e modelagem"

else:

return "Recurso inválido"

print(descrever\_recurso(entrada))

Todas as entradas e saída dos algoritmos são utilizados o STDIN e STDOUT de cada linguagem, abaixo tem algumas dicas de como utilizar cada STDIN e STDOUT de cada linguagem.

**JavaScript**

Em JavaScript as funções de STDIN e STDOUT respectivamente são **gets** e **console.log**, a função gets é implementada internamente para auxiliar a entrada dos dados.

**Exemplo:**

let line = gets(); // Lê a linha de entrada

console.log(line); // Imprime o dado

**Java**

**Em Java existe várias formas de implementar o STDIN e STDOUT recomendamos utilizar BufferedReader para o STDIN e o System.out.println para o STDOUT.**

**Exemplo:**

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StringTokenizer st = new StringTokenizer(br.readLine()); // Lê a linha de entrada

int a = Integer.parseInt(st.nextToken());

System.out.println(a); // Imprime o dado

**Python**

**Em Python existe várias formas de implementar o STDIN e STDOUT recomendamos utilizar sys.stdin.readline para o STDIN e o print para o STDOUT.**

**Exemplo:**

import sys

a = int(sys.stdin.readline()) // Lê a linha de entrada

print(a); // Imprime o dado

**Descrição**

O ambiente de execução no Azure Machine Learning precisa ser configurado corretamente para garantir compatibilidade com bibliotecas e requisitos do modelo. Esses ambientes pode ser baseados em Conda, Docker ou máquinas virtuais, garantindo consistência e replicabilidade dos experimentos.

Neste desafio, você deve associar corretamente os conceitos de configuração de ambiente com suas respectivas funcionalidades.

**Entrada**

A entrada consistirá no nome de um dos conceitos de configuração de ambiente do Azure ML. Os seguintes valores são considerados válidos para este desafio de código:

1. “Ambiente Conda”
2. “Docker”
3. “Compute Instance”
4. “Inference Cluster”

**Saída**

A saída esperada é a descrição associada ao conceito fornecido como entrada. Seguem as saídas possíveis:

1. “Espaço isolado para armazenar pacites e dependências do projeto”
2. “Containerização para garantir replicabilidade do ambiente”
3. “Máquina virtual dedicada para experimentação e desenvolvimento”
4. “Infrastrutura escalável para implantação de modelos em produção”

**Exemplos**

A tabela abaixo apresenta exemplos com alguns dados de entrada e suas respectivas saídas esperadas. Certifique-se de testar seu program com esses exmplos e com outros casos possíveis.

|  |  |
| --- | --- |
| Entrada | Saída |
| Ambiente Conda | Espaço isolado que armazena dependências e versões de pacotes |
| Docker | Containerização para garantir replicabilidade do ambiente |
| Compute Instance | Máquina virtual dedicada para experimentação e desenvolvimento |

entrada = input()

def descrever\_recurso(recurso):

if recurso == "Ambiente Conda":

return "Espaço isolado para armazenar pacotes e dependências do projeto"

elif recurso == "Docker":

return "Containerização para garantir replicabilidade do ambiente"

elif recurso == "Compute Instance":

return "Máquina virtual dedicada para experimentação e desenvolvimento"

elif recurso == "Inference Cluster":

return "Infraestrutura escalável para implantação de modelos em produção"

else:

return "Recurso inválido"

print(descrever\_recurso(entrada))