Análise Comparativa de Plataformas em Nuvem para Análise de Dados Turísticos por Não-Desenvolvedores

Comparative Analysis of Cloud Platforms for Tourism Data Analysis by Non-Developers

Jefferson Santana Filho
Universidade Federal do Paraná
Curitiba - PR
jefferson.santana@ufpr.br

João Eugenio Marynowski
Universidade Federal do Paraná
Curitiba - PR
jeugenio@ufpr.br

José Elmar Feger
Universidade Federal do Paraná
Curitiba - PR
elmar@ufpr.br

RESUMO

Este trabalho resulta de uma abordagem interdisciplinar para empregar uma aplicação computacional na avaliação dos comentários de frequentadores de atrativos naturais e culturais em redes sociais. Investigou-se uma forma para facilitar a execução e o compartilhamento dessa aplicação para usuários não desenvolvedores. A revisão da literatura abordou o processo de análise textual e mineração de dados, bem como os principais modelos de serviços em nuvem, elencando as plataformas comerciais disponíveis conforme os modelos. As plataformas foram testadas por meio da criação e execução de *scripts* na linguagem R. Foram analisadas as plataformas baseadas em Notebook Jupyter de três diferentes empresas: Amazon, Microsoft e Google. Assim, o trabalho apresenta uma análise exploratória e experimental das principais plataformas comerciais de serviços em nuvem, relatando a experiência na utilização de cada uma delas para auxiliar na escolha da plataforma conforme a necessidade do usuário. Facilidade de uso, barreiras de entrada, custos e funcionalidades de colaboração estão entre os critérios avaliados. O Google Kaggle se destacou por permitir a persistência de arquivos. Já o Google Colab apresentou uma melhor gestão para o desenvolvimento compartilhado.

CCS Concepts

• **Applied computing** → **Document management and text processing**

• **Computer systems organization** → **Cloud computing**

Palavras-chave

Computação em Nuvem; Modelos Serviço em Nuvem; Análise de Dados em Nuvem; Jupyter Notebook.

ABSTRACT

This work results from an interdisciplinary approach to employ a computational application for evaluating comments from visitors to natural and cultural attractions on social media. It investigated ways to facilitate the execution and sharing of this application for non-developer users. The literature review addresses the process of data mining and textual analysis, as well as the main cloud service models, listing the available commercial platforms according to these models. The platforms were tested by creating and executing scripts in the R language. Platforms based on Jupyter Notebook from three different companies, Amazon, Microsoft, and Google were analyzed. Thus, the work presents an exploratory and experimental analysis of the main commercial cloud service platforms, reporting the experience of using each one to assist in selecting the most suitable platform based on user's needs. Ease of use, entry barriers, costs, and collaboration features were among the evaluated criteria. Google Kaggle stood out for allowing file persistence, while Google Colab offered better support for collaborative development.

Keywords

Cloud Computing; Cloud Services Models; Cloud Data Analysis; Jupyter Notebook.

1. INTRODUÇÃO

Com o crescimento exponencial de dados, o uso de plataformas em nuvem se torna uma opção preferível para muitas empresas e instituições. Isso impulsionou a oferta de plataformas que oferecem soluções em nuvem [7]. Com a existência de diversas soluções fornecidas por várias empresas, torna-se difícil a escolha da mais adequada, tanto da solução quanto da empresa. Embora existam trabalhos que avaliam plataformas considerando o desempenho computacional, falta uma avaliação empírica focada na usabilidade e na curva de aprendizado para usuários sem formação técnica em computação.

Este projeto nasce de uma relação interdisciplinar, a qual busca investigar soluções em nuvem para a implementação de sistemas computacionais para a análise de comentários de atrativos turísticos. O sistema de extração dos comentários é dinâmico e precisa ser atualizado frequentemente devido à natureza dos sites e redes sociais que também são frequentemente atualizados [4,14,2]. A complexidade aumenta ao citar a necessidade do processamento massivo de dados turísticos e, então, a aplicação de soluções de Big Data [26].

A importância deste trabalho reside no encontro das áreas de Big Data, onde é necessário o processamento de uma grande quantidade de dados, e a área do Turismo, que, com a ascensão da Internet e das Redes Sociais, tem acesso a uma diversidade de dados sobre as opiniões e experiências dos turistas.

O objetivo principal deste estudo é democratizar, através do uso de plataformas em nuvem, o acesso e a utilização de aplicações para análise de dados, abstraindo a necessidade de configuração de ambiente e tornando-as mais acessíveis e amigáveis para uma ampla gama de usuários, independentemente de sua experiência em desenvolvimento de software. A computação em nuvem oferece vantagens como a virtualização do ambiente de desenvolvimento, que facilita o compartilhamento e a atualização do sistema. Este trabalho explora principalmente o uso de plataformas derivadas do Notebook Jupyter, que simplificam o ambiente de desenvolvimento, abstraindo as camadas abaixo da aplicação. Também apresenta uma revisão da literatura sobre mineração de dados e análise textual, bem como dos principais modelos de serviços de computação em nuvem, fornecendo um panorama geral de seu funcionamento.

2. REVISÃO DE LITERATURA

2.1. MINERAÇÃO DE DADOS E ANÁLISE TEXTUAL

Mineração de dados, segundo [23], envolve encontrar padrões e tendências em grandes conjuntos de dados para orientar decisões. Sendo uma área relacionada à estatística e podendo estar relacionada à inteligência artificial e aprendizado de máquina, a mineração de dados visa gerar conhecimento. Os passos anteriores à mineração de dados passam pela seleção dos dados brutos nos quais há algum interesse; a limpeza desses dados, onde é feita a remoção do que não será utilizado; e o processamento e padronização dos dados selecionados em novos dados para facilitar a análise. Na análise de texto, ou mineração de texto (*Text Mining*), após o processo de limpeza e padronização dos dados, diversas técnicas podem ser aplicadas para a obtenção de conhecimento a partir dos dados, como o uso de aprendizado de máquina para a análise textual e de sentimentos [29, 15].

Podem ser utilizadas ainda abordagens como a tokenização, a formação de n-gramas e o algoritmo Apriori. A tokenização é um processo de quebra de frases em sequência de palavras que são utilizadas para o processamento de linguagem natural [17]. Já o algoritmo Apriori tem a função de facilitar a análise de grandes quantidades de texto, agrupando conjuntos frequentes [25]. Assim, é possível quantificar ”n-gramas”, que no contexto do processamento de linguagem natural, representam sequências de palavras de tamanho ”n”[5]. Por exemplo, a frase ”A cama está desarrumada”, transformada em unigramas (1-gram) resulta no conjunto: [“A”, “cama”, “está”, “desarrumada”]. Transformada em bigramas (2-gramas) resulta em: [“A cama”, “cama está”, “está desarrumada”]. Utilizando-se da contagem da repetição de vezes que os n-gramas aparecem, é possível extrair características importantes do objeto em análise de forma contextualizada.

Essas etapas e procedimentos são geralmente implementados da forma habitual, onde o desenvolvedor usa sua própria máquina para desenvolver e tem controle total sobre as variáveis, como o hardware, sistema operacional e conexão à rede. Assim, o desenvolvedor pode gerenciar todas as camadas, tanto a aplicação quanto as abaixo dela, caso necessário, o que não ocorre utilizando o modelo de desenvolvimento em computação em nuvem.

2.2. COMPUTAÇÃO EM NUVEM E SEUS MODELOS DE SERVIÇO

A computação em nuvem (Cloud Computing) é uma modalidade de computação dada em servidores de forma remota, permitindo utilizar a capacidade computacional desses servidores para executar aplicações. [22] argumentam que existem algumas vantagens da computação em nuvem em relação à execução feita da forma habitual. A possibilidade de escalar a aplicação, já que as grandes empresas que lidam com ambiente em nuvem possuem diversos servidores. A alta disponibilidade e confiabilidade, já que múltiplas cópias dos dados estão salvas em ambiente em nuvem, tornam menos provável a perda desses dados. A virtualização, que permite a aplicação ser acessada de qualquer lugar do mundo. E por último a versatilidade, já que o ambiente em nuvem oferece vários serviços com diferentes funcionalidades.

No modelo de computação em nuvem adotado por empresas como a Google, Microsoft, Amazon e IBM, se fornece acesso aos computadores da empresa mediante virtualização, variando o grau de abstração[1] sobre as camadas de software contidas acima dos componentes físicos (hardware), efetivamente. Por meio da virtualização de um sistema operacional, no Windows, por exemplo, o usuário interage como se estivesse acessando um computador remotamente, com todas as funcionalidades que um computador normal teria. No entanto, há serviços em que o usuário tem controle apenas sobre funcionalidades específicas, como escrever código para execução ou administrar um banco de dados. Nesses casos, o acesso é mais restrito, assemelhando-se mais a interagir com um programa, aplicativo ou funcionalidade específica do que a ter acesso completo a um computador remotamente. Quando se usam serviços onde o acesso é mais específico a certa funcionalidade, o usuário não precisa se preocupar com detalhes, como quais são as especificações da máquina ou qual sistema operacional está executando a aplicação; todos esses detalhes são administrados pela empresa (provedor). Ou seja, todos os detalhes das camadas abaixo do serviço que está sendo executado são abstratos para o usuário.

Os principais modelos de serviço oferecidos são: *Infrastructure as a Service (IaaS)*, *Containers as a Service (CaaS), Platform as a Service (PaaS), Function as a Service (FaaS), Backend as a Service (BaaS)* e *Software as a Service (SaaS)*. A Figura 1 apresenta os modelos e as camadas de acordo com os níveis de abstração possíveis e quem faz o gerenciamento, se é o usuário ou provedor [19, 13, 12, 27].

Os serviços FaaS e BaaS são agrupados em Serverless, pois abstraem as camadas inferiores à aplicação e rodam código sob demanda, onde a cobrança só é feita se houver uso [30]. Os modelos CaaS, PaaS e FaaS estão destacados com um colchete, pois abstraem as camadas de software e hardware abaixo de uma

[1]Forma de ocultar os detalhes de trabalho de um sistema sobre o outro, permitindo facilitar a interoperabilidade e a independência do sistema até o hardware.

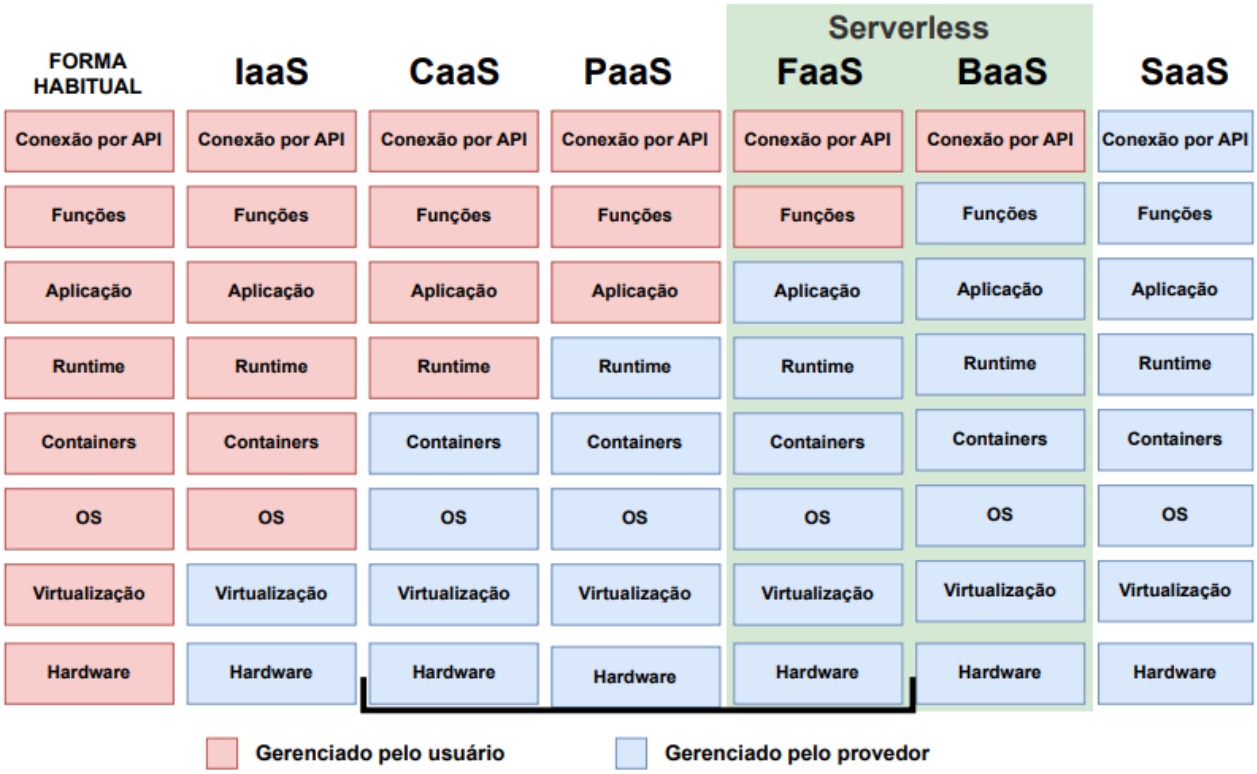

Figura 1: Modelos de serviço. Baseado em [19]

aplicação. Na forma habitual, é necessário possuir uma máquina com capacidade de processamento suficiente para executar a aplicação, ficando a cargo do desenvolvedor a responsabilidade sobre o *hardware*, conexão com a Internet e rotas de rede, se necessário, além de configuração da máquina, instalação de bibliotecas e *softwares* necessários para executar a aplicação. Somente após isso,a aplicação poderia ser executada.

No modelo IaaS toda a parte de administração de hardware fica a serviço da plataforma em nuvem que ofereceria, mediante virtualização, uma máquina virtual para ser utilizada, mas toda a configuração referente ao software (configuração da máquina, instalação de bibliotecas, programas necessários para executar a aplicação) ainda estaria nas mãos do desenvolvedor [18]. Nesse modelo, para a implantação da aplicação de análise de dados, seria necessária a configuração do sistema operacional, variáveis de ambiente, a instalação da linguagem de programação R e a instalação das bibliotecas necessárias para a aplicação.

O modelo CaaS é utilizado para se referir a serviços de orquestração de contêineres. A principal funcionalidade desses orquestradores é administrar todas as variáveis do ciclo de vida de um *container*[2], incluindo implantação, escalonamento, alocação de recursos, acesso à rede, entre outros aspectos [11].

O CaaS pode ser compreendido como uma camada mais abstrata posicionada acima do IaaS (*Infrastructure as a Service*). Nesse modelo, o desenvolvedor é responsável por realizar a conteinerização de sua aplicação, juntamente com as bibliotecas necessárias, e posteriormente implantá-la em uma plataforma na nuvem. Tudo isso sem a necessidade de se preocupar com o *hardware* ou com partes do *software* que não fazem parte da aplicação em si.

Com o uso do CaaS, é possível controlar a escalabilidade e a distribuição da carga de processamento dos *containers* — como ocorre com o uso do Kubernetes, que realiza essa administração. Alternativamente, a própria plataforma pode assumir de forma autônoma a tarefa de escalar os *containers* conforme a demanda.

No modelo PaaS o desenvolvedor se utiliza de uma plataforma onde é feita a implantação da aplicação e das bibliotecas, e o serviço administra a infraestrutura abaixo. A implantação da aplicação dependerá do serviço que for utilizado; porém, se o Google Colab fosse escolhido, seria necessária a instalação das bibliotecas utilizadas e a transferência do código para a plataforma.

No modelo SaaS, o provedor do serviço disponibiliza uma interface gráfica onde o usuário final pode interagir com a aplicação final, e todas as camadas abaixo são administradas pelo provedor do serviço. Os modelos BaaS e FaaS são ditos "*serverless*", porque são serviços que abstraem a parte da administração de um servidor, permitindo que desenvolvedores implementem e executem suas aplicações sem se preocupar com a infraestrutura, conforme [10, 24, 19, 30, 28]. A cobrança é efetuada apenas pelos recursos utilizados. Estes modelos são considerados uma ramificação do PaaS, sendo a principal diferença a despreocupação sobre a escalabilidade da aplicação.

[2] *Container*: é um pacote de *software* executável que contém tudo o que é necessário para sua execução: o código do usuário, o ambiente de execução (runtime), bibliotecas, ferramentas e configurações.

O modelo BaaS fornece uma terceirização de funções de backend, como autenticação de usuários, serviço de notificação, comunicação com banco de dados, APIs, entre outros. FaaS é um modelo que funciona com requisições e respostas a um evento, onde o desenvolvedor escreve a aplicação, executada conforme a demanda das requisições. No FaaS não se guarda estado das informações, tem um ciclo de execução curto (geralmente em milissegundos).

Também têm-se trabalhos que apresentam técnicas de divisão de carga de processamento para um paralelismo em ambiente em nuvem, como propõem [8] e [20]. Técnicas essas que podem utilizar mais de um modelo de serviço, dependendo da abordagem. Os trabalhos relacionados com a análise de dados em ambiente em nuvem tendem a focar em soluções que utilizam plataformas com maior controle das camadas abaixo da camada de aplicação, como a AWS Lambda, citada por [6]. E se utilizando de frameworks especializados, como o Apache Hadoop e Apache Spark, citados no trabalho de [31].

Com o crescimento da demanda por processar e armazenar uma abundância de dados, a computação em nuvem tornou-se uma opção popular para empresas que optam por não precisar administrar a parte de *hardware*. Grandes empresas de tecnologia ingressaram nesse mercado e buscam expandir sua influência e oferecer esses serviços, mas escolher dentre todas as opções dos diferentes modelos de serviço se torna uma tarefa difícil e pouco explorada cientificamente.

3. METODOLOGIA

Os métodos de pesquisa escolhidos foram o exploratório e o qualitativo, visando proporcionar maior familiaridade com o objeto de estudo [21]. Foram explorados os serviços pertencentes às três empresas líderes de mercado: Google Cloud, Amazon Web Services e Microsoft Azure. O objetivo é testar empiricamente como seria a experiência de implantar uma aplicação simples que se relacionasse com o conceito de turismo inteligente, avaliando a dificuldade para agentes da área do turismo que não são da área da tecnologia da informação. A pesquisa foi feita qualitativamente, para avaliar as características de cada plataforma e serviço, como gratuidade, facilidade de uso do serviço por não desenvolvedores e o processo necessário para executar a aplicação.

A avaliação foi feita considerando a implementação de um estudo de caso, utilizando um *script* de mineração de texto. Para auxiliar a implementação e implantação do *script*, também foi disponibilizada uma breve introdução à plataforma Kaggle, identificada como “Notebook de introdução à plataforma Kaggle” e disponível em: https://www.kaggle.com/code/jeffersonufpr/introdu-o-ao-kaggle.

O *script* utilizado para a implementação da aplicação é escrito na linguagem R e está disponível no Kaggle em: https://www.kaggle.com/code/jeffersonufpr/1-minera-o-de-coment-rios. Esse *script* visa a análise de comentários coletados de redes sociais, envolvendo mineração de dados, por meio de classificação e clusterização. Os dados foram obtidos a partir de comentários da plataforma Tripadvisor, que é uma das maiores redes internacionais sobre atrativos turísticos, naturais e culturais [3]. O *script* envolve tarefas que um analista de turismo realiza frequentemente, mesmo com objetivos diversos dos apresentados na referência indicada, por isso o seu emprego neste trabalho.

Na parte de mineração de texto do *script*, os procedimentos utilizados no pré-processamento são semelhantes aos utilizados por Schmitz (2015) [25], envolvendo diversos processos. A "tokenização", onde é feita uma “limpeza” no texto, retirando símbolos e caracteres especiais e transformando frases em vetores de palavras. A remoção de “stopwords”, isto é, palavras que devem ser desconsideradas na análise, como artigos, preposições e palavras identificadas por especialistas. A transformação morfológica, que reduz as palavras ao seu radical, elimina sufixos, plurais e inflexões. O cálculo de relevância, que identifica quais palavras têm maior relevância no texto. E a identificação de termos compostos com duas (bigramas) e até 5 palavras (pentagramas), sendo os termos obtidos que possuem sentidos diferentes quando estão em conjunto, por exemplo “data final” e “data inicial” para pertencer à mesma posição de palavra no vetor formado e na distribuição de relevância dos termos.

Para a classificação, foram utilizados quadrigramas (termos com 4 palavras) classificados manualmente por especialistas em cada domínio da experiência. Os quadrigramas foram gerados a partir dos comentários feitos por usuários na plataforma TripAdvisor, e são utilizados para treinar e avaliar os desempenhos de diversos modelos de aprendizado de máquina. O *script* para a classificação e clusterização foi identificado como “Notebook de classificação e clusterização” e está disponível em: https://www.kaggle.com/code/jeffersonufpr/2-class-e-clust-de-coment-rios.

Inicialmente, o estudo exploratório envolveu identificar e avaliar o desenvolvimento de uma aplicação usando a virtualização em nuvem de forma análoga ao desenvolvimento local. Posteriormente, os diferentes modelos de serviço em nuvem foram identificados e avaliados para a abstração de parte das camadas de *hardware* e *software* abaixo da aplicação. São avaliados a gratuidade de uso, suporte ao desenvolvimento compartilhado, possibilidade de salvar a sessão, dificuldade para acesso ao sistema, acesso a conjunto de dados e fóruns, integração com outros serviços em nuvem, configuração prévia necessária, dificuldade de configuração, escopo de uso, quantidade de passos no primeiro uso e após, e tempo para execução.

4. RESULTADOS

A Tabela 1 apresenta a instanciação dos diferentes modelos de serviço, disponibilizados por três grandes empresas da área de tecnologia, sendo elas a Google, principalmente com a Google Cloud; Amazon, com a AWS; e a Microsoft, com a Azure.

O funcionamento e a abstração dos modelos de serviço comerciais nas diferentes empresas foram analisados e considerados equivalentes; assim, foram experimentados os serviços instanciados na plataforma da Google conforme os modelos. Foram explorados os modelos IaaS, CaaS, PaaS, FaaS, BaaS e SaaS disponibilizados, mas a maioria se apresentou desnecessariamente complexa para o uso de não desenvolvedores. Assim, buscou-se identificar modelos com interfaces onde não era necessário configurar parte do ambiente de desenvolvimento, ou seja, onde as camadas abaixo da aplicação fossem abstraídas.

Os serviços da Google Cloud Run, App Engine e Cloud Functions, pertencendo respectivamente aos modelos de serviço CaaS, PaaS e FaaS, destacados na Tabela 1, foram objetos de estudo em uma exploração inicial. Posteriormente, serviços baseados em Jupyter Notebook, que pertencem ao modelo PaaS, e são disponibilizados

Table 1: Serviços comerciais conforme os modelos.

Modelos de Serviço	Descrição	Google	Amazon	Microsoft
IaaS(Infraestructure as a Service)	Se utiliza de VMs para Processamento	Compute Engine	Amazon Elastic Compute Cloud	Máquinas Virtuais do Azure
CaaS (Container as a service)	Auxilia o gerenciamento e implantação de aplicações usando containers	Google Kubernetes Engine, Cloud Run	Amazon Elastic Kubernetes Service e Amazon Elastic Container Service, AWS Fargate, AWS Lambda, AWS SAM	Azure Kubernetes Service, Instâncias de containers do Azure
PaaS (Plataform as a Service)	Permite a criação, execução e gerenciamento de scripts sem se preocupar com infraestrutura, código é executado enquanto a aplicação existir	App Engine, Google Colab, Kaggle	AWS Elastic Beanstalk, Amazon SageMaker	Serviço de aplicativo do Azure, Azure Notebooks
FaaS (Function as a Service)	Permite a criação, execução e gerenciamento de scripts sem se preocupar com infraestrutura, código é executado só quando necessário	Cloud Functions	AWS Lambda	Azure Functions
BaaS(Backend as a Service)	fornece uma terceirização de funções de backend, como autenticação de usuários, serviço de notificação, comunicação com banco de dados, APIs, entre outros	Google Firebase	AWS Amplify	Azure App Service
SaaS (Software as Service)	Serviço completo com interface gráfica utilizada por usuários finais	Google Drive, Gmail, Google Documentos e Planilhas	Amazon (loja), Twitch, Amazon Prime Video	Word online, Outlook e Onedrive

como interface de utilização pelas principais provedoras de serviço em nuvem, também foram analisados e testados.

Notebooks Jupyter são amplamente utilizados por analistas e pesquisadores na área de ciência de dados [16], o que os torna objeto de estudo pertinente. Assim, os serviços baseados em Jupyter foram avaliados nas plataformas da Microsoft, Amazon e Google.

4.1 EXPLORAÇÃO EXPERIMENTAL DE TRÊS MODELOS NA PLATAFORMA GOOGLE CLOUD

A primeira abordagem foi realizada na plataforma em nuvem da Google, Google Cloud Platform (GCP), já que ela fornece uma cota de uso gratuita na maioria dos modelos de serviços citados anteriormente. A plataforma possui mais de cem serviços que abrangem: IA, aprendizado de máquina, gerenciamento de APIs, computação, contêineres, análise de dados, banco de dados, ferramentas para desenvolvedores, internet das coisas (IoT), serviços de rede, e servidores para jogos [9].

Para utilizar os serviços da GCP é necessário possuir uma conta do Google e fornecer um cartão de crédito para eventuais cobranças, caso a cota de uso gratuito seja ultrapassada. Após o login na plataforma, acessa-se a página do console, onde é possível criar um projeto. Após a criação do projeto, acessa-se a barra lateral da página e seleciona-se qual o serviço que será utilizado, então o usuário pode adicionar, configurar e utilizar o serviço escolhido. Os serviços analisados são o Cloud Run, App Engine e Cloud Functions, pertencendo, respectivamente, aos modelos de serviço CaaS, PaaS e FaaS. Esses serviços abstraem a configuração de infraestrutura de hardware, são escaláveis sob demanda e a cobrança é realizada apenas pelos recursos utilizados. Para fazer a implantação (*deploy*) de aplicações no Cloud Run e no App Engine é necessário possuir um aplicativo para acesso ao GCP, por exemplo a interface de linha de comandos do Google Cloud (CLI gcloud). Esse procedimento pode ser feito diretamente de um computador local ou a partir do navegador, utilizando o Cloud Shell, que possui uma interface para desenvolvimento (*Integrated Development Environment* - IDE) e um terminal de comandos do GCP, conforme apresentado na Figura 2.

O Cloud Run permite fazer *deploy* (implantação) de imagens de contêineres, que podem ter um tempo de execução de até uma hora. Para implantar um container no Cloud Run é necessário fazer o envio (*upload*) da imagem do container em algum repositório e fazer posteriormente a obtenção (*pull*) da imagem utilizando o terminal. O Cloud Run é configurado para receber até oitenta requisições na mesma instância, antes que uma nova seja criada.

O Cloud Functions pertence ao FaaS e permite fazer implantação de código, que é geralmente executado por um curto período, de no máximo nove minutos normalmente e uma hora para funções HTTP. Pode-se utilizar as linguagens de programação: Node.js,

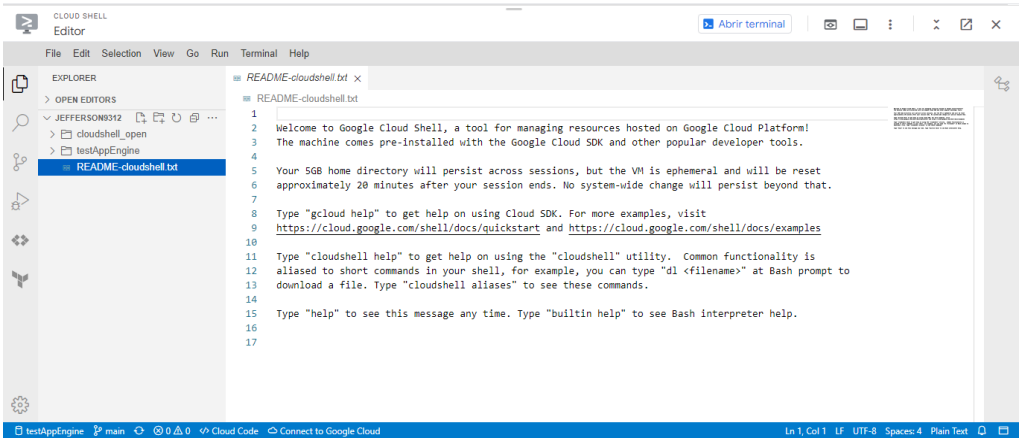

Figura 2: Interface da Google Cloud Shell IDE.

Python, Go, Java, C#, Ruby e PHP. O Cloud Functions permite apenas uma requisição para cada instância, e uma nova instância é criada para cada requisição paralela efetuada. A implantação de funções no Cloud Functions pode ser feita utilizando o editor de código no navegador ou através do envio de um arquivo compactado contendo o código; ou ainda, por meio de um repositório do Cloud Source Repositories (https://cloud.google.com/source-repositories).

O App Engine oferece diferentes opções de máquinas virtuais que irão executar uma aplicação web. Possui suporte a várias linguagens de programação como: Node.js, Java, Ruby, .NET, Go, Python e PHP. A configuração é feita principalmente por meio de um arquivo YAML (como o app.yaml). A App Engine faz todo o processo de escalabilidade da aplicação e administração da infraestrutura abaixo, onde é possível visualizar logs, efetuar monitoramento de recursos e debug da aplicação pelo Google Stackdriver (https://cloud.google.com/stackdriver/). Para fazer a implantação de um projeto no App Engine, é necessário fazer um clone de um projeto em um repositório pelo terminal. Assim, o projeto é clonado para o ambiente de desenvolvimento na nuvem, onde o editor do Cloud Shell funciona como uma IDE e permite a implantação da aplicação.

4.2 JUPYTER NOTEBOOKS

O Jupyter (https://jupyter.org) é um ambiente de desenvolvimento baseado na utilização de navegador web e de código aberto. É uma aplicação criada com o foco em ciência de dados, onde é possível ter anotações e execuções interativas, intercalando texto e código executável no mesmo lugar. A possibilidade de ter texto explicando o código pode auxiliar o entendimento por usuários não desenvolvedores.

O Jupyter funciona instanciando um núcleo para desenvolvimento *kernel* onde existe um ambiente de execução que provê suporte a alguma linguagem de programação. Ele pode ser instalado de forma offline, em um computador, bem como disponibilizado por ambientes de desenvolvimento remoto e por plataformas em nuvem. Geralmente, o *kernel* padrão é na linguagem Python, mas é possível alterar para a execução de outras linguagens, como o R.

Um notebook Jupyter funciona com o conceito de células, de texto ou código. Cada célula de código é executada individualmente e de forma sequencial, e toda saída, seja de texto ou gráfica, será impressa abaixo da célula de código correspondente. É empregado o conceito de "sessão" em cada notebook, ou seja, apesar das células de texto e código continuarem salvas, todas as alterações feitas pelo usuário no ambiente de execução serão descartadas ao fim da sessão.

As plataformas em nuvem trazem diversas bibliotecas por padrão, para facilitar o processo de configuração e implantação de scripts em diversas linguagens. Por exemplo, são disponibilizadas as bibliotecas em R: "*tidyverse*", "*tibble*" e "*ggplot2*". E em Python:"*beautifulsoup4*", "*pandas*" e "*tensorflow*". Para visualizar quais bibliotecas estão instaladas é possível executar os comandos conforme as linguagens. Caso esteja desenvolvendo em R, por exemplo, seria: "*installed.packages()*"; e caso esteja sendo desenvolvido em Python: "*!pip freeze*". Também é possível o acesso às pastas e arquivos no ambiente de execução temporário disponibilizado, assim como o terminal Shell, ambos em Linux. Um notebook Jupyter pode ser compartilhado como um arquivo qualquer e possui a extensão ".ipynb".

Jupyter é base de plataformas de desenvolvimento compartilhado de diversas empresas, como no Google o Colab e o Kaggle, na Amazon o SageMaker e na Microsoft o Azure Notebooks. Assim, essas plataformas serão objeto de estudo mais detalhado a seguir.

4.3 GOOGLE COLAB E KAGGLE

O Colab (https://colab.google/) e o Kaggle (https://kaggle.com/) se utilizam do Jupyter, mas com algumas alterações de interface gráfica. As duas plataformas se diferenciam principalmente nas suas propostas. Enquanto o Colab se limita aos notebooks, o Kaggle tem foco no desenvolvimento de comunidades e compartilhamento de bases de dados.

O Kaggle possui fóruns, cursos de ciência de dados, conjuntos de dados compartilhados pelos próprios usuários e competições que podem até oferecer prêmios em dinheiro. Ele fornece um espaço de 20 GB para cada notebook do usuário, onde é possível salvar arquivos de aplicação e saídas (logs). Também fornece 107 GB, disponíveis para o armazenamento de conjuntos de dados (datasets). Por sua vez, o Colab necessita de plataformas externas para o armazenamento, como o GitHub (https://github.com/) ou Google Drive (https://drive.google.com/drive/).

O Colab conta com três modalidades de oferta, uma gratuita e duas pagas. A versão gratuita conta com 12 GB de RAM e pode ficar em execução 12 horas seguidas e é a única modalidade que não dá acesso ao terminal Linux. A modalidade Pro conta com mais memória RAM, até 24 GB, e pode ficar em execução por 24 horas.

Já a modalidade Pro+ possui até 48 GB de memória RAM e a execução pode ser feita em segundo plano, sem a necessidade de manter o navegador aberto. O Kaggle é inteiramente gratuito e possui um limite de 16 GB de RAM, podendo ficar em execução por 12 horas seguidas, inclusive em segundo plano.

A Figura 3 apresenta a interface de desenvolvimento do Colab. Para alternar para o kernel da linguagem R, pois o padrão é utilizar a linguagem em Python, deve-se clicar na flecha para baixo presente no canto superior direito, selecionar “Ambiente de Execução” e depois “Alterar o Tipo de Ambiente de Execução” e, então, selecionar a linguagem R.

A Figura 4 apresenta a interface de desenvolvimento do Kaggle. Para alternar para o kernel da linguagem R, deve-se clicar na flecha para a esquerda presente no canto superior direito, ir em “Settings” e selecionar a linguagem R.

O Kaggle vem com o acesso do notebook à Internet desativado por padrão. Para instalar os pacotes, foi necessário verificar um número de celular e depois habilitar a internet nas opções do notebook.

4.4 Microsoft Azure Notebooks

Para utilizar Jupyter na plataforma da Microsoft Azure é necessário ter uma conta, assim como registrar um cartão de crédito, o que pode resultar em fatura. Atualmente, é fornecido um período de avaliação gratuito, onde é fornecido um crédito de duzentos dólares. É necessário criar um *workspace* da Azure Machine Learning Studio e criar uma instância de computação, onde é possível selecionar a configuração da máquina virtual que executará o notebook. Só então é possível criar um notebook usando a instância de computação e a Figura 5 apresenta a interface de desenvolvimento da Azure.

A máquina virtual utilizada para o teste na plataforma pertencia a um cluster de servidores da Azure em São Paulo. A instância foi a

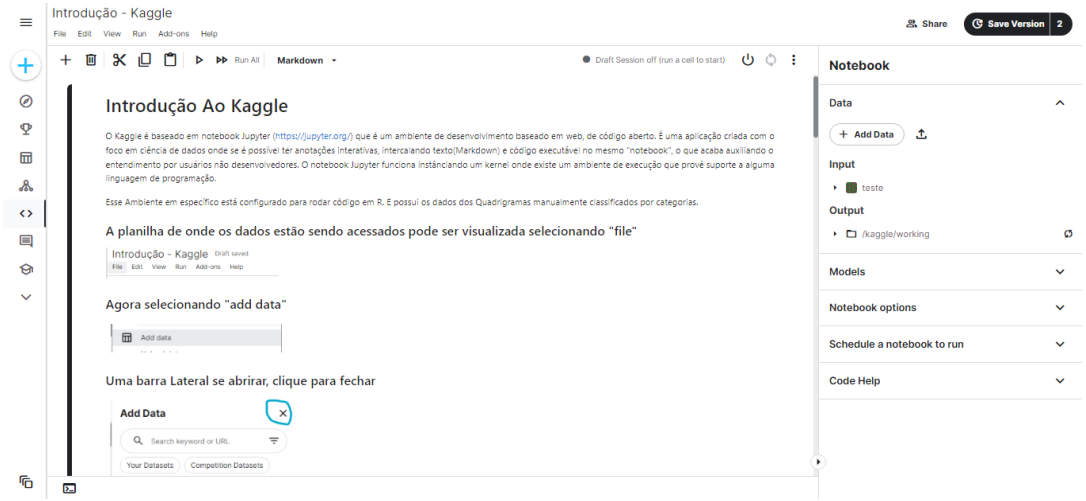

Figura 3: Interface de desenvolvimento do Kaggle.

Figura 4: Interface de desenvolvimento do Google Colab.

“D2 v3” com uma CPU não especificada, de 2 núcleos com 8 gigabytes de memória RAM, com um preço de R$0,855 por hora.

A princípio, não seria possível instalar pacotes porque a linguagem R não possuía acesso à pasta onde os pacotes são instalados, o que já é um problema conhecido pela própria Azure, que traz uma possível solução (https://azure.github.io/azureml-sdk-for-r/articles/troubleshooting.html). A solução se trata de acessar o terminal e a pasta na qual os pacotes são instalados e utilizar o comando “chmod” do Linux para dar permissão de escrita, leitura e execução na pasta. Na Azure, para alternar para o kernel R, deve-se direcionar a opção presente no canto superior direito e alterar para a linguagem R. A persistência de arquivos na Azure Notebooks é feita no ambiente da máquina virtual e não do Notebook Jupyter, o que permite que o arquivo exista enquanto a instância da máquina virtual existir, esteja ela ativa ou não.

4.5 AWS SAGEMAKER NOTEBOOK INSTANCES

Para utilizar o Notebook Jupyter na plataforma AWS é necessário utilizar o SageMaker (Figura 6), ter uma conta na AWS, registrar um cartão de crédito e configurar o domínio do Amazon SageMaker Studio. Existem duas formas de executar código R na AWS SageMaker: pelo servidor RStudio, sendo necessário possuir uma licença do RStudio Workbench; ou pela instância de notebooks. Para acessar a instância de notebooks, é necessário buscar na aba lateral do SageMaker Studio por uma marcação

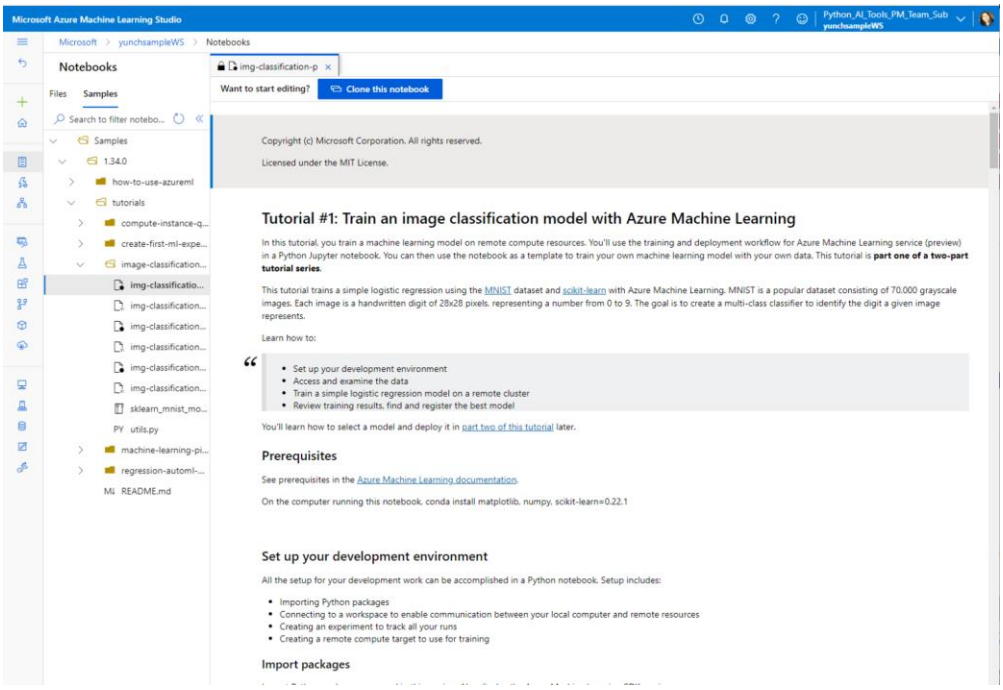

Figura 5: Ambiente de desenvolvimento do Azure Notebooks.

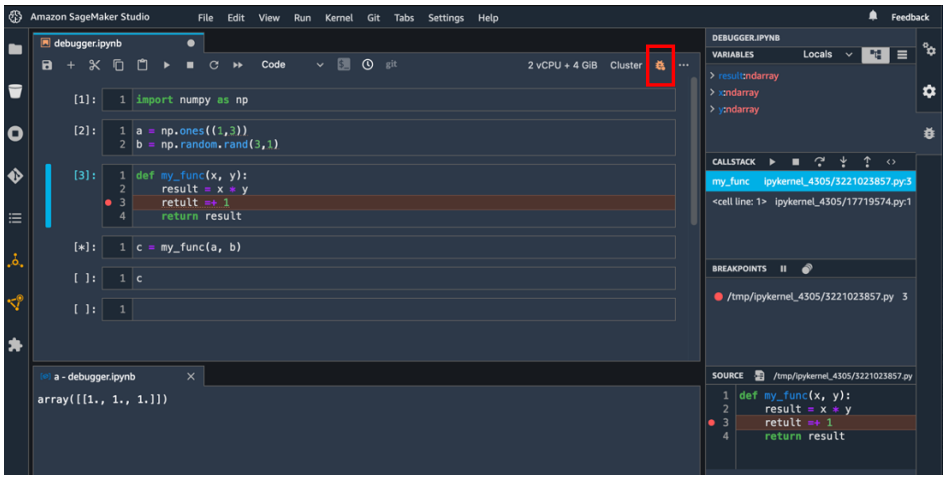

Figura 6: Ambiente de desenvolvimento da AWS SageMaker Notebook.

escrita “bloco de anotações” que, quando clicada, mostra a possibilidade de acessar as “instâncias de bloco de anotações”, onde é possível criar instâncias de Notebook Jupyter. Criando uma instância, é possível selecionar a máquina virtual na qual o notebook irá executar, assim como outras configurações adicionais. Após isso, é possível selecionar o kernel que possui a linguagem R, e assim o ambiente de desenvolvimento está pronto para uso. A máquina virtual utilizada para o teste, disponibilizada pela AWS, pertencia a um cluster de servidores no norte da Virgínia, nos Estados Unidos. Utilizou-se a instância “ml.t3.medium”, com uma CPU não especificada, de 2 núcleos com 4 gigabytes de memória RAM, e possui um custo aproximado de USD 0,05 (R$0,27) por hora. Na exploração da aplicação, por acidente, a instância acabou ficando em atividade por cerca de 35 horas, o que acabou resultando num custo de USD 1,78 (R$9,48). A persistência (salvamento) dos arquivos na AWS SageMaker só pode ser feita com serviços externos, como Amazon Simple Storage Service (Amazon S3) ou GitHub. O arquivo depende da sessão ativa do notebook, ao fim da sessão todos os arquivos adicionados pelo usuário ou gerados durante a execução serão descartados.

4.6 DESENVOLVIMENTO COMPARTILHADO

Desenvolvimento compartilhado se refere à prática onde várias pessoas podem trabalhar simultaneamente no mesmo notebook. No Colab, o desenvolvimento compartilhado preza pela atomicidade. Se um notebook estiver sendo editado por dois usuários ao mesmo tempo, e um usuário salvar o notebook, o outro usuário não conseguirá salvar até resolver de alguma forma os conflitos entre o seu notebook e o notebook salvo anteriormente. No Kaggle, o controle de versões é administrado pelos usuários que possuem permissão de editar o notebook. Se os dois usuários estiverem editando o mesmo notebook ao mesmo tempo, e um deles salvar, a versão será salva com sucesso; porém, se, posteriormente a isso, o outro usuário salvar, isso irá sobrescrever o salvamento anterior, o que merece uma atenção especial no desenvolvimento compartilhado.

Nos notebooks da Azure é informado que o desenvolvimento compartilhado funciona de forma simultânea [32]. Assim como nos notebooks da AWS SageMaker, com o uso de ”Shared Spaces”[1]. Uma solução comum é usar uma plataforma externa de versionamento, como o GitHub, para manter um histórico das versões. Essas operações não foram avaliadas experimentalmente devido à restrição financeira para as execuções, mas, de acordo com comentários em seus fóruns, os serviços funcionam adequadamente.

4.7 DISCUSSÃO

A Tabela 2 apresenta o comparativo entre as plataformas de notebooks em nuvem avaliadas. São apresentados os critérios de gratuidade de uso, suporte ao desenvolvimento compartilhado, possibilidade de salvar a sessão, dificuldade para acesso ao sistema, acesso a conjuntos de dados e fóruns, integração com outros serviços em nuvem, configuração prévia necessária, dificuldade de configuração, escopo de uso, quantidade de passos no primeiro uso, quantidade de passos após o primeiro uso, e tempo para execução.

O Colab e Kaggle podem ser considerados os mais econômicos, pois proporcionam acesso gratuito. Possuem entrada e configurações facilitadas, já que é possível acessar as duas plataformas com uma conta do Google e as configurações para o setup do notebook são mínimas, geralmente para escolher qual a linguagem de programação (Python ou R) e habilitar o acesso do notebook à internet (no caso do Kaggle) o que torna a barreira de entrada baixa para não desenvolvedores. Para um profissional de turismo, a necessidade de registrar um cartão de crédito no Azure e na AWS representa uma barreira de entrada significativamente maior, mesmo que haja um período gratuito.

O Kaggle se destaca por seus conjuntos de dados públicos e fóruns acessíveis. Por outro lado, Azure Notebooks e AWS Sagemaker são voltados para aplicações de grande escala que utilizam grandes volumes de dados e processamento, com infraestrutura em nuvem mais complexa. Exigem pré-requisitos técnicos, como requisitos de configuração. Assim, a escolha dessas plataformas deve refletir a experiência do usuário em ambiente em nuvem e o escopo do projeto, nesses casos menos indicada para não especialistas.

A integração com serviços em nuvem é possível e facilitada para as soluções de cada empresa. Neste caso, tanto o Kaggle quanto o Colab integram com o Google Drive, entretanto, o Kaggle tem uma

Table 2: Comparativo entre plataformas

Critério	Kaggle	Colab	Azure Notebooks	AWS Sagemaker
Uso Gratuito	Sim	Sim	Não	Não
Desenvolvimento Compartilhado	Sim	Sim	Sim	Sim
Salvar Sessão	Sim	Não	Não	Não
Configurações iniciais	Facilitada	Facilitada	Complexa	Complexa
Acesso a Datasets Públicos e Fóruns	Sim	Não	Não	Não
Integração com serviços em Nuvem	Google	Google Drive	Azure	AWS
Configuração Prévia Necessária	Não	Não	Sim	Sim
Dificuldade na Configuração	Baixa	Baixa	Média	Alta
Escopo de Uso	Pequeno e médio	Pesqueno	Grande	Grande
Quantidade de Passos no primeiro uso	4	3	9	11
Quantidade de Passos após o primeiro uso	4	3	3	7
Tempo para execução	9.624s	11.775s	10.315s (Std. D2 v3)	12.211s (ml.t3.large)

integração mais intrínseca com o ecossistema Google Cloud para conjuntos de dados, enquanto o Colab se integra mais apenas para o armazenamento via Google Drive.

A quantidade de passos para executar o notebook Jupyter é maior na Azure e AWS porque, no primeiro acesso, é necessário registrar dados relacionados ao setup de ambiente de trabalho (workspace), cobrança e pagamento. Após, com exceção da sessão AWS, é necessário somente criar o notebook, escrever o código e executar o notebook. O Kaggle tem um passo extra de selecionar a linguagem e habilitar a internet na aba lateral do notebook. A quantidade de passos da AWS é maior porque, a cada notebook, é necessário especificar as configurações da instância em nuvem, permissões de acesso ao notebook e configurações de ambiente. Na Azure, esses detalhes podem ser configurados durante o primeiro acesso, na criação de um workspace.

Quanto ao tempo de execução, este é uma limitação da análise pois foi considerado um processamento simplificado, uma vez que o custo inviabilizou executar a análise dos dados turísticos em todas as plataformas. Então, foi executado um script com uma repetição que incrementa uma variável, iniciada em zero, até o valor de cem milhões.O tempo de execução é representativamente aproximado, já que outros fatores, como velocidade de internet e localização geográfica do servidor que hospeda a plataforma, afetam esse quantificador. Além disso, é possível alterar as configurações de hardware disponíveis em cada instância, o que pode impactar profundamente o desempenho em diferentes contextos.

5. CONSIDERAÇÕES FINAIS

A utilização de plataformas hospedadas em nuvem apresenta grande potencial para ser empregada por aplicações que demandam processamento, compartilhamento e análise de dados. Elas proporcionam benefícios como a abstração de camadas de hardware e software subjacentes à aplicação, o que é essencial para a democratização do uso por usuários não desenvolvedores. Mas, com o grande número de modelos de serviços existentes, pode-se tornar uma tarefa difícil escolher a opção mais adequada para a utilização.O uso de plataformas baseadas em notebooks Jupyter permite intercalar código do programa e texto, possibilitando unificar o ambiente de desenvolvimento e a documentação. Assim, se mostra uma boa opção para usuários que não possuem experiência na área de desenvolvimento de programas, facilitando o acompanhamento passo a passo e a atualização ou personalização do programa. A importância de se utilizar essas plataformas em nuvem se dá pela conveniência que oferece, permitindo o desenvolvimento de algoritmos sem se preocupar com instalações e capacidade de *software* e *hardware* disponíveis localmente.

Este trabalho então cumpre com o objetivo de apresentar uma análise exploratória e experimental das principais plataformas em nuvem disponíveis e indicadas para a implantação de programas que analisam comentários de atrativos turísticos, que podem ser usados numa estratégia de turismo inteligente. Em suas limitações estão: a financeira, para testes mais aprofundados nas plataformas não gratuitas, as possíveis atualizações das plataformas, e o emprego de programas mais robustos que demandem mais recursos computacionais e exijam distribuição e paralelização do processamento, além do escopo da aplicação atual.

Como trabalho futuro, um novo programa poderia ser desenvolvido. Integrando a raspagem de dados de sites relacionados ao turismo, com uma ferramenta de processamento e análise desses dados, gerando *insights* sobre algum atrativo, tudo isso feito através de uma interface gráfica onde o usuário precisaria somente selecionar a atração desejada.

6. REFERENCES

[1] AWS. Share and use an amazon sagemaker studio classic notebook. https://docs.aws.amazon.com/sagemaker/latest/dg/notebooks-sharing.html, 2023. Acesso em: 9 jul. 2025.

[2] S. Bunese, J. E. Feger, J. E. Marynowski, R. D. F. R. Garcia, and M. F. A. Caracristi. Experiência turística no atrativo serra do espírito santo situado no parque estadual do jalapão, Tocantins. Proc. of ANPTUR - *Seminário Anual da Associação Nacional de Pesquisa e Pós-Graduação em Turismo*, 2022.

[3] M. d. F. d. A. Caracristi, J. E. Feger, S. Minasi, and J. E. Marynowski. A demanda turística do parque estadual do jalapão (pej, tocantins, brasil) baseada em comentários de redes sociais. *Revista Brasileira de Ecoturismo (RBEcotur)*, 14(3), ago. 2021.

[4] M. F. A. Caracristi, J. E. Feger, J. E. Marynowski, and S. M. Minasi. A demanda turística do parque estadual do jalapão (pej, to, brasil) baseada em comentários de redes sociais. *Revista Brasileira de Ecoturismo*, 14:291–314, 2021.

[5] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[6] L. G. R. De Jesus, A. Oliveira, and M. M. Teixeira. Mineração de dados em rede social baseada em uma arquitetura em nuvem. *In Anais da 7ª Escola Regional de Computação do Ceará, Maranhão e Piauí (ERCEMAPI)*, pages 111–118, São Luís/MA, 2019. Sociedade Brasileira de Computação.

[7] M. R. DeLisi. Gartner forecasts worldwide public cloud end-user spending to reach nearly 600 billion in 2023, 10 2022. Acesso em: 02/09/2023.

[8] X. Geng and Z. Yang. Data mining in cloud computing. In *Anais do 1º International Conference on Information Science and Computer Applications*, pages 1–6, Jiangsu Zhenjiang, 2013. Atlantis Press. Acesso em: 19 jul. 2024.

[9] Google. Produtos do google cloud. https://cloud.google.com/products?hl=pt-br, 2023. Acesso em: 19 jul. 2024.

[10] C. S. W. Group. Cncf wg-serverless whitepaper v1.0. https://github.com/cncf/wg-serverless/raw/master/whitepapers/serverless-overview/cncf_serverless_whitepaper_v1.0.pdf, 2018. Acesso em: 19 jul. 2024.

[11] R. Hat. O que é orquestração de containers? https://www.redhat.com/pt-br/topics/ containers/what-is-container-orchestration, 2019. Acesso em: 19 jul. 2024.

[12] IBM Cloud Education. Iaas paas saas cloud service models. https://www.ibm.com/cloud/learn/iaas-paas-saas, 2021. Acesso em: 19 jul. 2024.

[13] K. Jamsa. *Cloud Computing: SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security and More*. Jones & Bartlett Publishers, Burlington, MA, 2013.

[14] E. F. Kaizer, M. F. A. . Caracristi, J. E. . Feger, J. E. . Marynowski, and T. M. Silva. Análise da experiência relatada pelos turistas ao visitar o parque estadual do jalapão (pej) – to, Brasil. *Ateliê do Turismo*, 5:183–204, 2021.

[15] D. P. Kansãon, M. A. Brandão, and S. A. d. P. Pinto. Analysis of classification algorithms for emotion detection in brazilian portuguese tweets. *iSys - Brazilian Journal of Information Systems*, 12(3):116–138, Sep. 2019.

[16] T. Kluyver et al. Jupyter notebooks - a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas, 20th International Conference on Electronic Publishing,* Nieuwe Hemweg 6B 1013 BG Amsterdam, The Netherlands, 2016. IOS Press.

[17] S. J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Gallé, A. Raja, C. Si, W. Y. Lee, B. Sagot, and S. Tan. Between words and characters: A brief history of open-vocabulary modeling and tokenization in nlp, 12 2021.

[18] C. J. Miguel, J. A. Santos, and P. C. da Silva. Comparação de serviços para cloud computing. *Revista de Sistemas e Computação (RSC)*, 6:111–119, 12 2016.

[19] O. Paakkunainen. Serverless computing and faas platform as a web application backend. Master's thesis, Aalto University, Espoo, 2019.

[20] D. S. Pranav et al. Data mining in cloud computing. *In Anais eletrônicos do 5º International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1–8, Erode, 2021. IEEE. Acesso em: 19 jul. 2024.

[21] C. C. Prodanov and E. C. de Freitas. *Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico*. Editora Feevale, Av. Dr. Maurício Cardoso, 510, Bairro Hamburgo Velho, Novo Hamburgo, RS - CEP 93510-235, 2013.

[22] K. Rajamani and D. Sheela. Data mining techniques and algorithms in cloud environment-a review. *International Journal of Pure and Applied Mathematics*, 119:599–602, 2018. Acesso em: 19 jul. 2024.

[23] R. Ramakrishnan and J. Gehrke. *Sistemas de Gerenciamento de Bancos de Dados*. AMGH, Porto Alegre, RS, 3a edi¸c˜ao edition, 2008.

[24] M. Roberts. Serverless architectures. https: //martinfowler.com/articles/serverless.html, 2018. Acesso em: 19 jul. 2024.

[25] F. E. B. Schmitz and P. F. Brito. aplicação da técnica de text mining para comentários relacionados ao contexto do turismo. In ENCOINFO - *Congresso de Computação e Tecnologias da Informação*, 18., pages 43–51, Palmas - TO, 2016. CEULP/ULBRA. Acesso em: 29 jul. 2024.

[26] N. R. d. Silva and E. S. Koga. Evolução das pesquisas sobre big data e turismo: *Revisão sistemática da literatura nacional. Revista Brasileira de Iniciação Científica*, 10:e023043, nov. 2023.

[27] T. B. Soares and R. T. S. Araújo. Concepção e implementação de computação em nuvem aplicada ao cenário virtual. *Conexões - Ciência e Tecnologia*, 10(3):71–77, jun. 2016.

[28] F. R. C. Sousa. Computação serverless e gerenciamento de dados. *In Anais do 35º Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 199–204, Porto Alegre, 2020. Sociedade Brasileira de Computação.

[29] C. A. Sá and R. S. Moura. Estudo sobre métricas para definir reputação do autor de comentários em sites de vendas de produtos. *iSys - Brazilian Journal of Information Systems*, 12:6–23, 9 2019.

[30] A. G. Vieira et al. Computação serverless: Conceitos, aplicações e desafios. In A. G. Vieira et al., editors, *Minicursos do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, pages 190–236. SBC, Rio de Janeiro, RJ, Brasil, 2020. Acesso em: 19 jul. 2024.

[31] R. Vrbíc. Data mining and cloud computing. *JITA - Journal of Information Technology and Applications*, 4(1):1–18, jan 2012. Acesso em: 19 jul. 2024.

[32] S. Xu and Abeomor. Improving collaboration and productivity in azure machine learning. "https://techcommunity.microsoft.com/blog/machine-learningblog/improving-collaboration-\and-productivity-in-azure-machine-learning/2160906", 2021. Acesso em: 9 jul. 2025.