BenchCouncil Transactions on Benchmarks, Standards and Evaluations 3 (2023) 100150

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/

Full length article

CloudAISim: A toolkit for modelling and simulation of modern applications in AI-driven cloud computing environments

Abhimanyu Bhowmik [a], Madhushree Sannigrahi [a], Deepraj Chowdhury [b,c], Ajoy Dey [d], Sukhpal Singh Gill [e,*]

[a] SeaTech School of Engineering, University of Toulon, Toulon, France
[b] Center for Application Research in India (CARIn), Carl Zeiss (Bangalore) India Pvt Ltd., Bangalore, India
[c] Dept of Electronics and Communication Engineering, IIIT Naya Raipur, Naya Raipur, India
[d] TCS Research and Innovation, Kolkata, India
[e] School of Electronic Engineering and Computer Science, Queen Mary University of London, London, UK

ARTICLE INFO

Keywords:
Artificial intelligence
Explainable AI
Cloud computing
Machine learning
Healthcare
CloudAISim
Simulation

ABSTRACT

There is a very significant knowledge gap between Artificial Intelligence (AI) and a multitude of industries that exist in today's modern world. This is primarily attributable to the limited availability of resources and technical expertise. However, a major obstacle is that AI needs to be flexible enough to work in many different applications, utilising a wide variety of datasets through cloud computing. As a result, we developed a benchmark toolkit called CloudAISim to make use of the power of AI and cloud computing in order to satisfy the requirements of modern applications. The goal of this study is to come up with a strategy for building a bridge so that AI can be utilised in order to assist those who are not very knowledgeable about technological advancements. In addition, we modelled a healthcare application as a case study in order to verify the scientific reliability of the CloudAISim toolkit and simulated it in a cloud computing environment using Google Cloud Functions to increase its real-time efficiency. A non-expert-friendly interface built with an interactive web app has also been developed. Any user without any technical knowledge can operate the entire model, which has a 98% accuracy rate. The proposed use case is designed to put AI to work in the healthcare industry, but CloudAISim would be useful and adaptable for other applications in the future.

1. Introduction

A set of practices aiming to base decisions on the analysis of data instead of intuitive insights can be used to define data-driven decision-making. When compared to conventional ones, businesses that implement data-driven decision-making processes are more financially beneficial and productive [1]. The outcomes of recent Artificial Intelligence (AI) research projects serve as the foundation for many decision-making tools [2]. The development of Machine Learning (ML) techniques is largely responsible for the success of AI-based tools [3]. The availability of sizable datasets on various real-world features as well as the rise in computational gains, which are typically attributable to the powerful Graphics Processing Unit (GPU) cards [4], are particularly encouraging in this regard.

The need to create sophisticated AI models with previously unheard-of performance levels has progressively given way to a rising interest in alternative design elements that would improve the usability of emerging products [5]. Complex AI models lose a part of their practical effectiveness in a wide range of application domains [6]. The main cause is that AI models are frequently created with a performance-focused approach, neglecting other significant – and occasionally crucial – aspects like accountability, transparency, and justice [7]. The AI models are typically "black boxes" since there is no explanation provided for the elements that are projected to perform well; as a result, they simply allow for the prominent display of input and output parameters while hiding the visibility of the intrinsic relationships between those parameters [8]. It is advantageous to have some explanations of individual predictions that are recognised using an AI system, more specifically in an automated environment, because these applications may include crucial decision-making [9].

This research aims to develop a transparent and self-explanatory system using AI, especially Automated Machine Learning (AutoML)

* Corresponding author.
E-mail addresses: abhimanyu-bhowmik@etud.univ-tln.fr (A. Bhowmik), madhushree-sannigrahi@etud.univ-tln.fr (M. Sannigrahi), deepraj.chowdhury@zeiss.com (D. Chowdhury), deyajoy80@gmail.com (A. Dey), s.s.gill@qmul.ac.uk (S.S. Gill).

https://doi.org/10.1016/j.tbench.2024.100150
Received 10 August 2023; Received in revised form 3 January 2024; Accepted 6 January 2024
Available online 9 January 2024

systems [10], that uses cloud computing, particularly serverless computing to propose the best machine-learning configuration for a particular issue and trace the reasoning behind a recommendation. It could also make it possible to interpretably and credibly examine the predicted outcomes.

1.1. Motivation

Even if AI/ML tools are open sources and widely available, creating a data processing pipeline and generating fine-tuned models for specific domains requires knowledge and skills in data science and AI, which are not always present in public sector industries like hospitals and nursing homes [11]. Therefore, we designed a toolkit called CloudAISim by utilising AI and cloud computing for the modelling and simulation of modern applications to bridge the gaps between non-professionals and AI expertise, starting with the healthcare application as a case study that would be useful and adaptable for other applications in the future.

The healthcare industry produces enormous amounts of data, using various sensors and health-monitoring devices to collect data [12]. The availability of data about the health of millions of patients may make it possible to create AI-based processes and models that give healthcare professionals useful insights [13]. This research also looks beyond the healthcare domain to design a benchmark model that can be useful for other applications from different domains. The framework and insights presented in this article can serve as a blueprint for extending AutoML and explainability to various other domains, from finance and retail to agriculture and manufacturing. The scalability and flexibility offered by cloud-based solutions make it feasible to democratise these AI technologies and accelerate innovation across industries [14]. This research mainly aims to provide the power of AI to non-experts without writing a single line of code. The proposed CloudAISim toolkit will do all the technical steps like Explanatory Data Analysis (EDA), feature engineering, choosing the best algorithm, and explaining the results predicted by the framework for better understanding by the user. In this paper, we have considered the applications/dataset of the healthcare domain, but it can be used for any domain where the dataset is in CSV format.

1.2. Contributions

The main contributions of this work are:

1. Proposing a toolkit called CloudAISim for efficient explainable machine learning technique modelling and implementation in the healthcare domain.
2. Finding the most accurate and responsive machine learning model for chronic as well as infectious diseases like diabetes, heart disease, breast cancer and COVID-19 in the healthcare domain.
3. Simulating a prototype web application for the validation of CloudAISim to provide a visual display for data, models and the explainability of results.
4. Implementing the CloudAISim in a cloud computing environment using Google Cloud Functions to increase real-time efficiency.
5. Highlighting the promising future directions.

The rest of the paper is structured as follows: The relevant related works regarding ML-based data analytics solutions and the requirement for transparency to build confidence in AI models are covered in Section 2. The proposed framework is described in Section 3 along with how its various elements work together to accomplish the desired outcomes. The results obtained using a few test cases are discussed in Section 4. Section 5 demonstrates the proposed application and Section 6 discusses the important findings of this research. Finally, Section 7 concludes the paper and offers future directions.

2. Related works

Various AutoML systems have been developed in recent years that offer partial or full ML automation, including systems like Auto-sklearn [15], tree-based pipeline optimisation tool (TPOT) [16], Auto-WEKA [17], and ATM [18] and commercialised software like Google AutoML,[1] RapidMiner,[2] and DarwinAI.[3] These methods include automatic feature engineering [19,20], automatic model selection [21,22], automatic hyperparameter tuning [23], and automatic data preparation [24,25]. Some methods make an effort to automatically select an ML algorithm while also optimising its hyper-parameters.

Many of the AutoML solutions, some of which are the results of contests from 2015 to 2018, were developed in the last few years. The ChaLearn AutoML Challenges[4] primarily focused on automating the solution of supervised machine learning tasks under certain computing restrictions. These computational restrictions varied significantly across tasks, but they were typically time (about 20 min for training and assessment) and memory consumption restrictions. Guyon et al. [26] review a thorough examination of the AutoML difficulties from 2015 to 2018. In essence, neural architecture may be considered a specific kind of indifferentiable hyperparameter. Hyperparameter optimisation is one of these activities that most directly relates to the approach we suggest in this study. Grid search and random search [27] are two of the simplest techniques to find a suitable configuration of hyperparameters from a list of options without considering past results. There are various sets of approaches that are often used in hyperparameter optimisation. Being one of the most well-known Sequential Model-based Optimisation (SMBO) [28] techniques that can take advantage of historical data, Bayesian optimisation [29] uses the Gaussian method for prototyping the surrogate function that roughly imitates the relationships between hyperparameters and their desired outputs. All of these techniques are, however, black-box optimised. The single study on AutoML for graph representation [30] employs the Gaussian Process to determine the performance of the hyperparameters, but it scarcely explains how individual hyperparameter affects the performance of the model or why a specific value is picked for a hyperparameter to execute the subsequent assessment trial.

AI systems that can give human-understandable explanations for their activities and output are referred to as Explainable AI (XAI) [6]. By their very nature, end users may be curious as to how and why systems reach any conclusion [31]. They are seen as “black boxes” when the sophistication of AI algorithms and systems increases [32]. Growing complexity may lead to a lack of openness that makes it difficult to comprehend these systems’ logic, which has a detrimental impact on users’ faith in them.

2.1. Critical analysis

Table 1 compares the proposed CloudAISim with existing frameworks based on important parameters. The model accuracy of the aforementioned studies is pretty high; however, the generalisation of the studies is limited. Only 3 of the studies have formalised feature extraction procedures that can be generalised. Two of the research have implemented explainability, which can be critical for many industries such as healthcare. None of the mentioned studies have implemented their framework in a serverless cloud environment. A novel CloudAISim framework that has high customisation and consists of a web interface that can be very easily used by non-technical users. This will enable the non-expert to unleash the power of AI and can be immensely helpful for any industry such as the healthcare industry. Moreover, the CloudAISim

[1] https://cloud.google.com/automl
[2] https://rapidminer.com
[3] https://darwinai.com/
[4] http://automl.chalearn.org/

Table 1
Comparison of proposed CloudAISim framework with existing solutions.

Related works	Dataset	Feature extraction	AutoML	Cloud computing (Serverless)	XAI
Ferreira et al. [33]	Pneumonia dataset	✗	WebDL	✗	✗
Shawi et al. [34]	Breast Cancer Wisconsin	✓	NNI	✗	✗
Alaa et al. [35]	Risk factors Cystic Fibrosis	✓	AutoPrognosis	✗	✓
Alnegheimish et al. [36]	MIMIC-III	✓	MLBlocks	✗	✗
Garouani et al. [13]	Big Industrial Data	✗	AMLBID	✗	✓
CloudAISim (Proposed work)	**Breast Cancer Wisconsin, Covid, Heart Disease Cleveland Dataset**	✓	AutoKeras	✓	✓

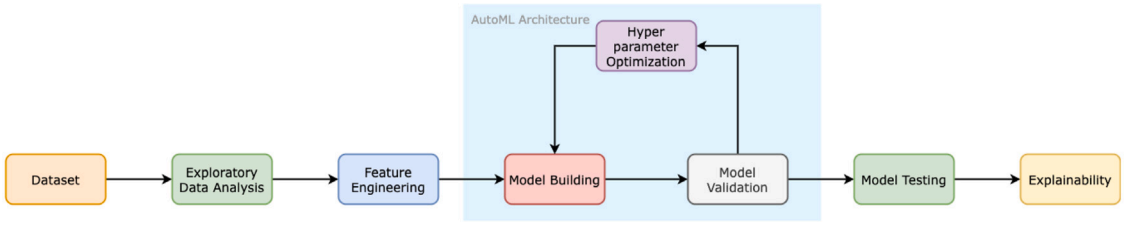

Fig. 1. The Abstract View of Proposed Framework.

framework enhances the scalability of the environment and can be incorporated into a large-scale scenario. To the best of our knowledge, the proposed solution is the first to use generic explanation techniques of Auto ML systems as decision support systems in a serverless setting.

3. CloudAISim framework

In this section, the description of the proposed approach used to achieve the objectives of the work has been discussed. Fig. 1 shows the abstract view or high-level view of the proposed CloudAISim framework. As shown in Fig. 1, the framework accepts the dataset from the user's device, like a smart phone or laptop. Then it is passed to the Automated EDA tool for explanatory data analysis, and then feature engineering is done on the dataset by the feature tool, the best machine learning model is generated, and hyperparameter tuning is done. After that, the explanation of the prediction is shown using lime. The entire execution is conducted on a serverless platform.

3.1. Architecture

Fig. 2 shows the main architecture of CloudAISim, which includes AutoML models, and the usage of Explainable AI (XAI) to demonstrate the working of the ML models, as well as the serverless architecture of the prototype. The main components of the proposed architecture are discussed further in Sections 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7.

3.2. Dataset

Firstly, the "Breast Cancer Wisconsin (Diagnostic) Data Set" by "UCI ML Repository" is implemented on the novel methodology for the paper [37]. The dataset contains tabular data with 32 features and over 569 data points. A fine needle aspirate (FNA) of a breast lump is used to generate the features from a digital image in 3-dimensional space as described by Bannett et al. [38]. They characterise the properties of all the observable cell nuclei in the image. Every data point is classified into either Benign (B) or Malignant (M) class. Secondly, the architecture is applied to the "Heart Disease Cleveland dataset" Dataset by "UCI ML Repository" [39]. The dataset constitutes over 300 patients' data with 75 attributes. However, only 14 of the features are taken into consideration for determining whether a patient has heart disease or not. Thirdly, the "Diabetes dataset", originally from the National Institute of Diabetes and Digestive and Kidney Diseases, is used in this work [40]. The goal is to determine if a patient has diabetes based on diagnostic parameters. The implemented Diabetes dataset is a subset of an enormous dataset with 10 attributes and 768 instances. All patients are Pima Indian females who are at least 21 years old. Finally the "Covid-19" is a dataset, used in the paper [41] which contains data from 800 people and 26 attributes such as their profession, health parameters and lifestyle parameter, and the risk factor of getting infection by covid is mentioned. The higher the risk factor the higher chance of getting infected by Covid. So we classified the person with a risk factor of more than 0.5 as high (1) and less than 0.5 as low (0).

3.3. Serverless cloud

Serverless computing is a method for providing backend services on an "as-needed" basis. The cloud provider controls the servers on behalf of their customers while expanding and maintaining the system as necessary [42]. This is the cloud computing execution paradigm. Since any device, regardless of its specifications, may access an application, it becomes a resource independently [43]. This allows for greater scalability and flexibility as the application can automatically scale up or down based on demand [44]. Additionally, serverless computing eliminates the need for developers to worry about server management and infrastructure maintenance, allowing them to focus solely on writing and deploying code.

The experimental architecture was developed on Google Cloud Platform (GCP), a serverless solution enabling efficient data storage and analysis. The proposed experiment was conducted using cloud functions in the Python 3.9 runtime. Google Cloud Storage is also used for storing the data and its respective output. GCP also allows seamless interactions with other Google Cloud services, like Cloud Storage events, which are used as triggers for the respective cloud functions. This architecture also offers built-in security features and automatic scaling capabilities, ensuring optimal performance and cost efficiency for the application.

In the proposed architecture, three subsequent cloud functions were used as shown in Fig. 2:

1. To perform automated feature selection and feature engineering using the open-source Feature tools;
2. To generate an AutoML model and predict the results using the Auto-Keras Python library; and
3. To explain the predicted results.

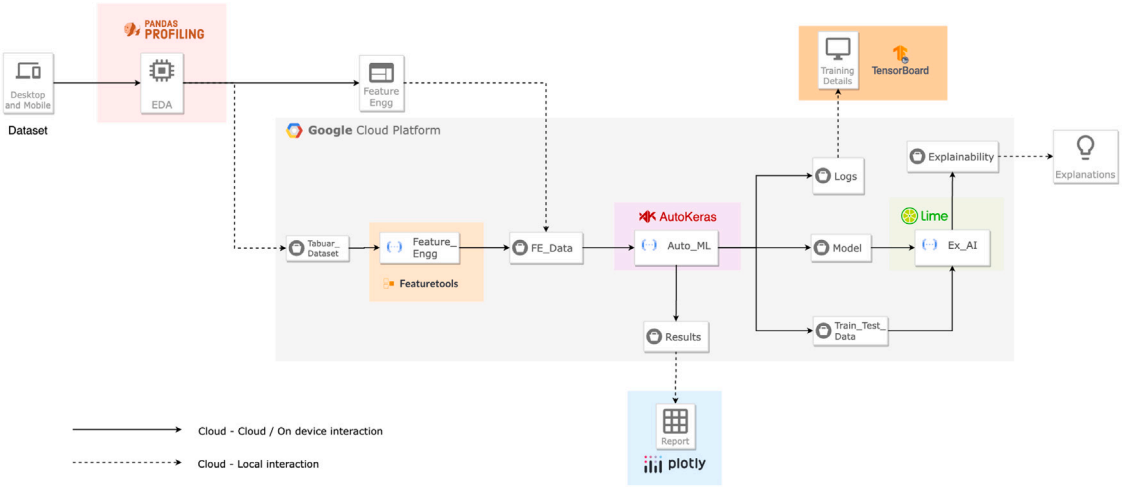

Fig. 2. CloudAISim Architecture.

The model explanation is completed using the Lime library. In place of standard Lime explanations, SP-Lime (Submodular Pick Locally Interpretable Model-Agnostic Explanations) is used to explain the model's global decision boundary over a sample set of observations.

3.4. Data preprocessing

Case-specific and automated data pre-processing was a considerable challenge while implementing the system in a cloud platform. This section discusses the implementation of Pandas Profiling[5] and Feature Tools[6] for automated and rapid EDA and Feature Engineering respectively.

3.4.1. Automated EDA using Pandas profiling

Exploratory data analysis (EDA) is a vital stage in constructing any impressive model. EDA involves finding outliers, spotting missing values, figuring out how skewed the datasets are, converting categorical variables, and overall understanding the underlying characteristics and ways to apply them in models.

Pandas Profiling[6] is a user-friendly open-source Python tool for automated exploratory data analysis. It generates a data frame report in a range of different formats. Although the Pandas' df.describe the operation can demonstrate basic information, it does not give a full data frame report. Pandas profiling was implemented in the system architecture for automated and rapid analysis of data.

3.4.2. Automated feature engineering using feature tools

Feature engineering is the process of creating and adding new features, or variables, to the dataset to enhance the effectiveness and precision of the machine learning model. Case-based knowledge and accessible data sources serve as the foundation for the most efficient feature engineering. Without requiring any human input, automated feature extraction employs deep networks or specialised algorithms to automatically extract characteristics from images or signals.

Featuretools[7] is a free and open-source Python architecture for automated feature engineering. It generates features automatically from relational and temporal information. Deep Feature Synthesis (DFS) is utilised for implementing automated feature engineering. Featuretools gives users the ability to perform feature selection by (1) removing null values; (2) removing single-value features; and (3) removing highly correlated attributes. For machine learning and predictive modelling, one can construct useful features by combining the raw data with information about the data.

3.5. AutoML

The time-consuming and iterative activities required in developing a machine learning model may be automated using Automated Machine Learning or AutoML. It provides a diverse range of approaches to help those with little background in machine learning access this technology. It attempts to minimise the requirement for experienced individuals to create the ML model. Additionally, it helps to increase productivity and promote machine learning research [13].

To properly comprehend automated machine learning, we must first understand the life cycle of a data science or machine learning project. A data science project's lifetime typically includes many stages, including data cleaning, feature engineering, model selection, parameter optimisation, and model validation. Even though technology has improved so much, all of these procedures still involve manual labour, which takes time and calls for several data scientists with the necessary skills. For non-ML professionals, it is quite challenging to do these jobs because of their intricacies. The demand for automating these activities has increased because of the rapid development of ML apps, which will make it easier for those without technical expertise to utilise them. Consequently, automated machine learning was developed in order to fully automate the process, from data cleansing to parameter optimisation. Not only does it save time, but it also performs fantastically.

In this paper, AutoML is considered the first mandatory cloud function for the framework. It is launched when the clean data is uploaded to the designated bucket. This will create five different models from the training set (which makes up 70% of the total data), train them for 100 iterations, and select the model with the highest accuracy for the given dataset. The performance matrices (Confusion Matrix, Classification Report) will be generated after the chosen model has been evaluated on test data (30% of the supplied data). The model itself, all training and testing data, performance matrices, and more will be exported into separate cloud buckets as in Fig. 1. The TensorFlow-Keras standard format for exporting ML models, *.h5*, is used to export the architecture. There are multiple models to perform AutoML operations. Some of them are given in Table 2.

3.5.1. AutoKeras

A Keras-based AutoML system is called AutoKerac.[7] It was created by Texas A & M University's DATA Lab. Making machine learning accessible to everyone is the aim of AutoKeras. It searches using Neural Architecture Search (NAS) algorithms to eventually eliminate the

[5] https://pandas-profiling.ydata.ai/docs/master/index.html
[6] https://www.featuretools.com
[7] https://autokeras.com/

requirement for deep learning engineers. AutoKeras takes advantage of Keras to conduct a potent neural network search for model parameters. It offers modular building pieces to carry out architectural searches as well as high-level end-to-end APIs like ImageClassifier or TextClassifier to address machine learning challenges in a few lines.

3.5.2. Auto-sklearn

The detailed description of the Auto-sklearn approach by Feurer et al. [15] demonstrates how it empowers the machine learning user from algorithm selection and hyperparameter setting. Meta-learning, Bayesian optimisation (BO), and ensemble construction make up its three main elements. Auto-sklearn uses traditional ML algorithms provided by the sklearn library. The optimal hyperparameters may be found using Bayesian optimisation, which is data-efficient. However, Auto-Keras performs better than Auto-sklearn when utilising data that is suitable for deep neural networks. Moreover, the recent version of Auto-Sklearn has compatibility issues with various Operating systems. Hence, it was not considered for this research work.

3.5.3. TPOT

The tree-based pipeline optimisation tool (TPOT) [16], an AutoML framework, is based on an evolutionary method that uses genetic programming to select a framework for implementation. To provide the optimal Python code, it can evaluate hundreds of pipelines. It has built-in classification and regression algorithms. It combines several pipeline operators to create flexible tree-based pipelines. ML models from the Sklearn library or different data transformation operators make up these operators. However, it cannot handle categorical data and language processing power. TPOT mainly employs ML pipelines but with complex datasets, deep learning implementation is required which is provided by AutoKeras.

3.5.4. AutoWEKA

AutoWEKA [17] is an automated machine learning tool, which has been designed to find the best machine learning algorithm automatically for any dataset. It uses the WEKA framework which is a meta-learning approach for finding the best algorithm for a specific problem by comparing the performance of a lot of machine learning algorithms on the given dataset. AutoWEKA also tunes the hyperparameters of the best algorithm selected to improve its performance. It is a user-friendly tool that requires very limited user input and is suitable for both experts and non-experts. AutoWEKA is an open-source software tool that can be used freely.

3.5.5. Google AutoML

Google AutoML[2] is an attempt by Google to empower professionals with limited knowledge in Machine Learning to generate models based on their specific needs. They use techniques like evolutionary algorithms, and neural architectures to build a deep learning model using data and prompts by the user. However, it is only restricted to Google Clouds and it is very difficult to transfer enormous amounts of data from existing systems to a cloud network.

3.5.6. DarwinAI

DarwinAI[4] is an AI solutions provider that provides a range of AI tools and technologies to develop and deploy Neural Network models. This allows users to quickly and easily develop highly optimised neural network architectures without requiring a deep understanding of neural network architecture design. The platform also includes a range of tools for data preparation, model training, and model deployment, making it a comprehensive end-to-end AI tool.

3.5.7. Lale

Lale [45] is a semi-automated approach using scikit-learn, for tuning hyperparameters and selecting algorithms. It mainly aims at users with some knowledge of data science and machine learning, providing a high-level interface to experiment with different neural structures with their data. Lale uses popular existing tools like GridSearchCV, XGBoost, etc. for automation and interoperability. Lale is available as an off-the-shelf tool in Python. Nevertheless, it is only limited to professionals and is difficult to extend it to other domains.

3.5.8. Auto Pytorch

Auto Pytorch [46] is an automl framework based Pytorch. It was developed by AutoML Groups Freiburg and Hannover in the year 2021 with close collaboration with the University of Freiburg. This automl architecture is specialised to solve tabular data and time series datasets. This framework combines neural architecture search with ML hyperparameter tuning. It gives a developer-friendly API to interact with the model similar to AUtoKeras. However it does not optimise for Image and Text data as input format, so it is less generalised compared to AutoKeras.

3.5.9. Online AutoML (OAML)

The Online AutoML framework (OMAL) is an extension of an open-source General Automated Machine Learning Assistant (GAMA) framework. It has been developed by Pieter Gijsbers [47] in 2019. This framework generates an ideal machine-learning model depending upon the dataset and resource limitation provided to the framework. GAMA uses several search processes to find some implacable machine-learning models and combine them into one ensemble pipeline. OMAL extends GAMA's ability to handle online learning.

3.6. Result visualisation

Once AutoKeras creates the best neural network based on the given data, it is very important for the user to know and evaluate its performance. Hence, it is very important to visualise the result. We have used Plotly to satisfy this requirement.

3.6.1. Plotly

Plotly[8] is a graphing tool used to communicate data with customised visualisation and interactive graphs. It is available as a free-for-use library available for many coding languages like Python and R languages. Plotly provides an extensive range of charts and graphs that can be easily embedded in web applications. Thus, Plotly was used for most of the visualisations in the application to make it user-friendly and interactive.

3.7. Explainable AI (XAI)

In the field of machine learning, "explanations" at different levels offer insights into various elements of the model, from knowledge of the learnt representations to the identification of various prediction techniques, general trends and patterns, as well as the evaluation of the general model behaviour [41]. The two types of model explainability are global explainability and local explainability. When a model is globally explainable, users may infer its meaning from its general organisation. Local explainability only takes into account a single input and seeks to understand why the model chooses a certain course of action.

[8] https://plotly.com

Table 2
Comparison of different AutoML models.

Tool	Framework	Auto feature extraction	User interface	Explainability
Auto-Sklearn [15]	Scikit-learn	Only Missing Values	✗	✗
AutoKeras[a]	Keras	✗	✗	✗
TPOT [16]	Scikit-learn	✗	✗	✗
AutoWeka [17]	Weka	✗	✗	✗
Google AutoML[b]	TensorFlow,SparkML	Only Missing Values	✓	✓
DarwinAI[c]	TensorFlow	✓	✓	✓
Lale [45]	Scikit-learn	✗	✗	✗
Auto-PyTorch [46]	PyTorch	✗	✗	✗
Online AutoML (OAML) [48]	GAMA	✗	✗	✗

[a] https://plotly.com
[b] https://rapidminer.com
[c] http://automl.chalearn.org/

3.7.1. Local interpretable model-agnostic explanations (LIME)

The LIME methodology proposed by Ribeiro et al. [49] generates local explanations of classifier f predictions by fitting a simpler, interpretable explanation model g locally around the data point x to be explained. To maintain interpretability in the generated explanations, LIME represents the data in a way that is comprehensible, locally accurate, and model-neutral. These explanations are simpler because they demonstrate a closer relationship between the input and prediction [50].

For instance, let the grey-scale value vector of pixels in an image be $x \in \mathbb{R}^d$. A comprehensible representation of the initial dataset is used to fit the XAI model. The presence or absence of pixels in the picture might therefore be represented by a binary value vector as an interpretable representation of $x' \in \{0,1\}^{d'}$. (absence refers to having the value of a background colour, e.g., white). As a result of resolving the optimisation issue, the LIME explanation $\hat{g}$ is generated (1).

$$\hat{g} = \underset{g \in G}{argmin} \mathscr{L}(f, g, \pi_x) + \Omega(g) \tag{1}$$

Here,
G = the explanation model family,
$\mathscr{L}$ = loss function,
π_x = the locality around x,
Ω = complexity penalty.

Practically, G is the collection of linear regression models, with Ω limiting the expanse of explanatory features that can possess regression weights other than zero (even if several explanatory models may be employed). The weighted L2 distance is assumed to represent the loss function as in Eqs. (2).

$$\mathscr{L}(f, g, \pi_x) = \sum_i \pi_x(z_i)(f(z_i) - g(z_i'))^2 \tag{2}$$

where the sum passes over a collection of selected perturbed points around $x, (z_i, z_i')i = 1, \ldots, m$, where,
z_i = a disturbed data point in the initial dataset,
z_i' = the corresponding explainable version;

Here, $\pi_x(z_i)$ assigns a weight to each sample according to how similar they are to the point x, which is used to explain the classification result.

The second cloud function receives the model along with the training and test data when they are generated and uploaded to their respective cloud storage. The function generates SP-Lime explanations. For generating the explanations, a random 20 test data samples were chosen, among them, 5 results were generated. The combined graphs were then uploaded to a cloud storage bucket as a single HTML file. Any client-side web or mobile application can use the Google Cloud SDK to retrieve data, upload predictions, and explain them.

4. Validation of CloudAISim toolkit: Modelling of healthcare application

We used a healthcare application as a case study in order to verify the scientific reliability of this proposed CloudAISim framework. So, this section discusses the experimental process, datasets, and data preparation with all the relevant details on several use cases as in Table 7. The results on different datasets are compared against each other visually and through various metrics. In the healthcare domain, we have considered four different datasets such as Breast Cancer Wisconsin Diagnosis, Heart Disease Cleveland Dataset, Diabetes Dataset and COVID-19 Dataset.

Table 3
Classification report for 75:25 train–validation ratio of Breast Cancer dataset.

Class	Accuracy	Precision	Recall	f1-score
0	0.97	0.99	0.98	0.98
1	0.99	0.96	0.98	0.97

Table 4
Classification report for 75:25 train–validation ratio for Heart Disease Cleveland dataset.

Classes	Accuracy	Precision	Recall	f1-score
0	0.95	0.98	0.94	0.96
1	0.95	0.92	0.98	0.95

4.1. Case I: Breast cancer Wisconsin diagnosis

The greatest cause of cancer-related death among women worldwide and the malignancy with the highest rate of diagnosis is breast cancer [51]. According to Mubarak's epidemiological research, breast cancer, a particularly deadly kind of cancer, can cause death and mortality in women if it is not recognised in its early stages [52]. It can be found using a variety of techniques, including X-ray mammography, 3-D ultrasound, computed tomography, positron emission tomography, magnetic resonance imaging (MRI), and breast temperature monitoring, although a pathology diagnosis is the most reliable [53]. Sex, age, oestrogen, family history, gene abnormalities, an unhealthy lifestyle, and other variables linked to the development of the illness are only a few of the many risk factors for breast cancer [54].

Our dataset contains tabular data with 32 features and over 569 data points. A fine needle aspirate (FNA) of a breast lump is used to generate the features from a digital image in 3-dimension. The model is trained-tested with a 75:25 ratio of this dataset as given in Table 3. With such specifications, The proposed application produced an accuracy of 98% which is much better than existing cloud-based models. Further, the Confusion matrix and the ROC curve are also shown in Fig. 3.

4.2. Case II: Heart Disease Cleveland Dataset

Several different cardiac disorders are referred to as "heart disease". Coronary Artery Disease (CAD), which disrupts the blood flow to the heart, is the most typical kind of heart disease in the United States. A heart attack may result from reduced blood flow. Heart illness can

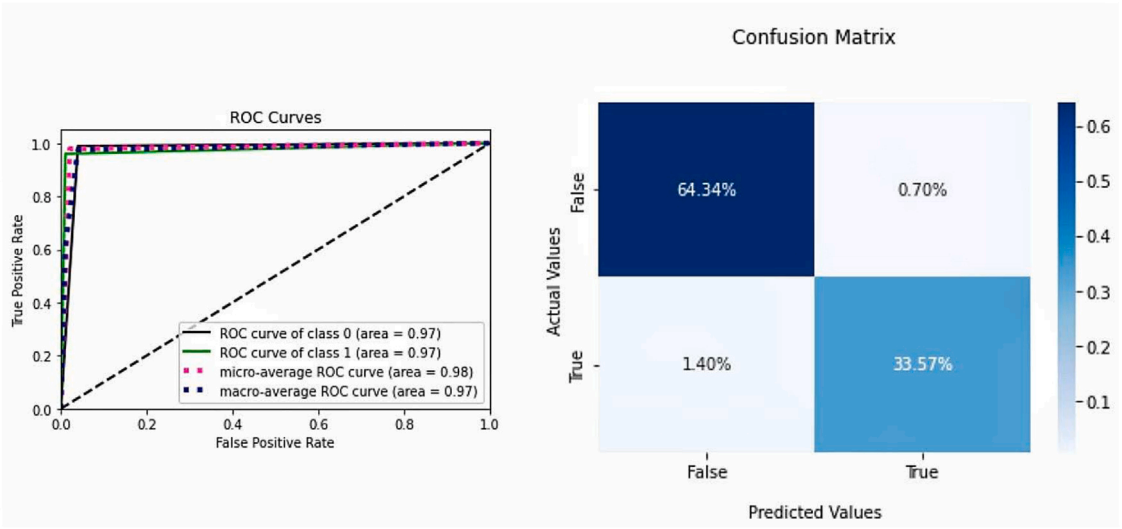

Fig. 3. ROC curve and Confusion matrix for Breast Cancer Dataset.

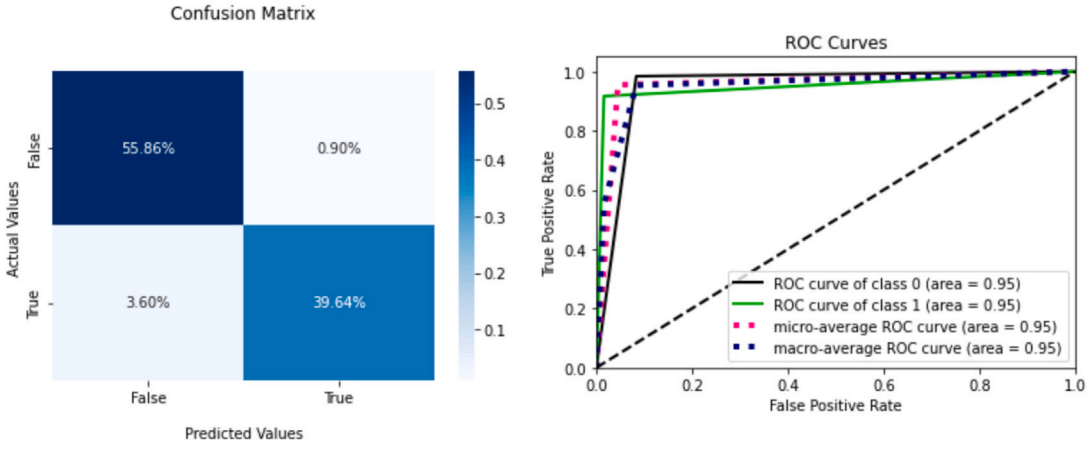

Fig. 4. ROC curve and Confusion matrix for Heart Disease Cleveland Dataset.

Table 5
Classification report for 75:25 train–validation ratio for Diabetes dataset.

Classes	Accuracy	Precision	Recall	f1-score
0	0.99	0.99	0.95	0.97
1	0.9	0.91	0.99	0.95

Table 6
Classification report for 75:25 train–validation ratio of Covid-19 dataset.

Classes	Accuracy	Precision	Recall	f1-score
0	0.96	0.97	0.98	0.97
1	0.94	0.93	0.92	0.93

sometimes go unnoticed until a person exhibits the early symptoms or signs of a cardiac arrest, heart failure, or an arrhythmia [55].

The dataset implemented in the proposed model constitutes over 300 patients' data with 75 attributes. With a ratio of 75:25 from this dataset, the model is trained and validated. With these conditions, the proposed application obtained an accuracy of 95% with a precision and recall of 0.95 and 0.96 respectively (as shown in Table 4). In addition, Fig. 4 also displays the ROC curve and the Confusion matrix.

4.3. Case III: Diabetes dataset

Diabetes is a chronic condition brought on by either insufficient insulin production by the pancreas or inefficient insulin use by the body. The hormone called insulin controls blood sugar levels. Uncontrolled diabetes frequently results in hyperglycemia, or elevated blood sugar, which over time causes substantial harm to many different bodily systems, including the neurons and blood vessels. A total of 1.5 million fatalities were directly related to diabetes in 2019, and 48% of these deaths occurred in those under the age of 70. Diabetes contributed to an additional 460,000 renal disease deaths, and high blood glucose is responsible for 20% of cardiovascular fatalities around the world [56].

To test the proposed application, the Diabetes dataset from the National Institute of Diabetes and Digestive and Kidney Disease is taken into consideration. The dataset constituted 10 characteristics and 768 instances with all patients being Pima Indian females who are at least 21 years old. The model is trained-tested on the 75:25 ratio of the dataset and achieved an overall accuracy of 96%. The Classification report is provided in Table 5. Further, the ROC-AUC curve and confusion matrix are given in Fig. 5.

4.4. Case IV: COVID-19 dataset

In general, human life and health have been profoundly damaged by the SARS-Cov2-led COVID-19 pandemic [57]. The majority

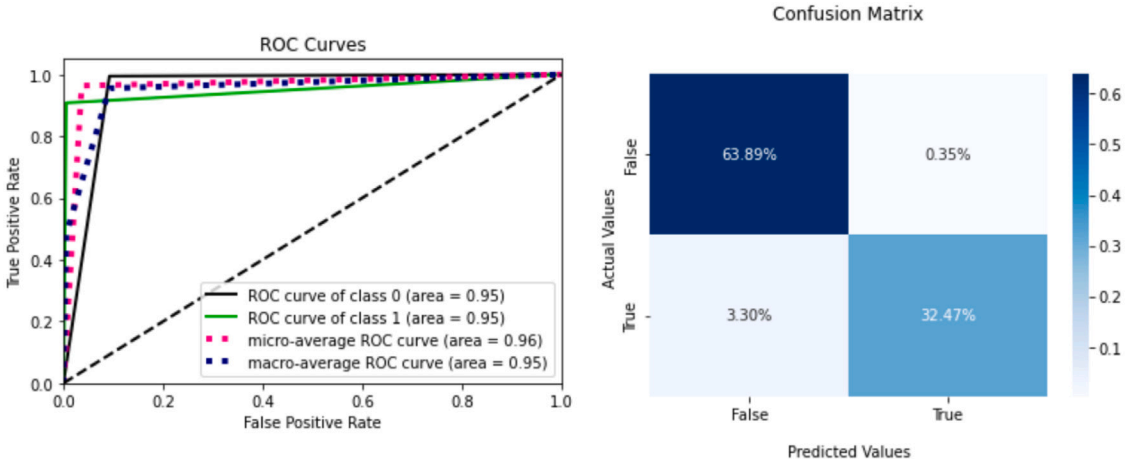

Fig. 5. ROC curve and Confusion matrix for Diabetes Dataset.

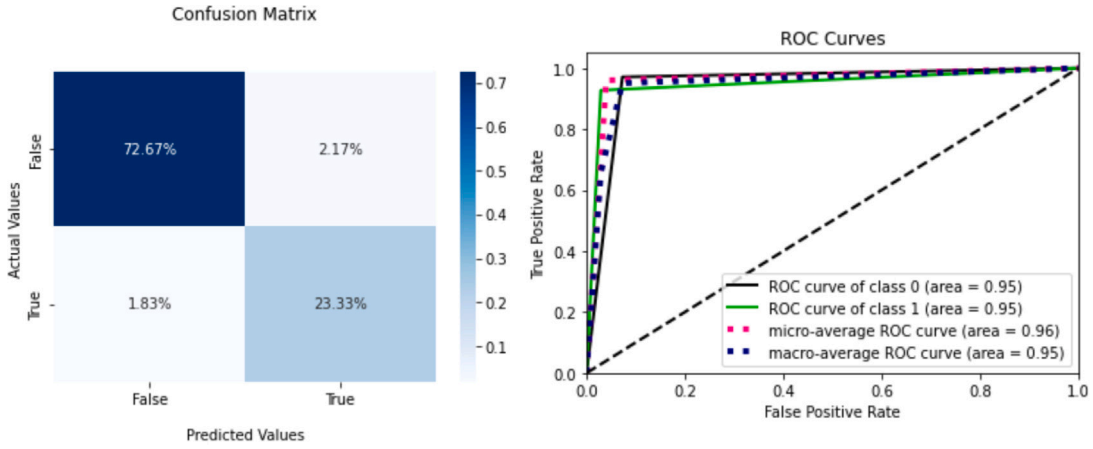

Fig. 6. ROC curve and Confusion matrix for COVID-19 Dataset.

of COVID-19 patients have mild to moderate symptoms. However, this devastating outbreak caused suffering and death in people. COVID-19 has the propensity to target and harm lung tissue [58]. This devastating outbreak caused death in 6,657,706 people around the world. Because of its fast-spreading ability, the World Health Organisation (WHO) designated COVID-19 a Public Health Emergency.

For Covid-19 dataset, tabular data is used with about 800 patient data. It contains 26 attributes such as age, heart conditions, smoking, pregnancy, etc. With a ratio of 75:25 from this dataset, the model is trained and validated. With these conditions, the proposed application has obtained an accuracy of 96% as shown in Table 6. In addition, Fig. 6 also displays the ROC curve and the Confusion matrix.

5. Simulation of proposed healthcare application

The aim of developing this application is to make AI usable for healthcare professionals and normal users, without any technical knowledge. An interactive web application is developed using StreamLit Framework to make it possible. The Application is deployed on StreamLit cloud and Google App Engine.

5.1. Implementation details

This section describes the working of the proposed model and its implementation details. The proposed model uses AutoML to automate the task of recognising and classifying different diseases and verifying the diagnosis.

The entire system is a hybrid architecture of cloud-based platforms and physical servers. The user end has a simple easy-to-read interface to access the proposed framework.

The dataset is uploaded by the user into a cloud storage bucket using the web interface as shown in Fig. 7. After the dataset is entered into the system by the user, the basic information of the dataset like datatype, class distribution, data correlation, etc. is available on the interface. The next Exploratory Data Analysis tab allows the user to select the data to be visualised and the attributes for analysis which will then generate a detailed summary with the help of Pandas Profiling, as shown in Fig. 8. All these computations are carried out on-device. For feature engineering, the user has the choice to make the entire process either manual or automated. In case of manual feature selection, all computations are carried out in the local device and will then be uploaded into the feature engineering cloud bucket (FE_Data) in GCP server. This gives the user freedom to select attributes, impute missing values, perform feature transformation, and remove outliers using different methods like Z-score, inter-quartile range, and so on. When opting for automated feature engineering, the tabular dataset is uploaded into a cloud bucket and the entire process is carried out in the serverless platform using Featuretools (Feature_Engg function) and is uploaded into the FE_Data bucket, as shown in Fig. 9.

As soon as any dataset is uploaded into the FE_Data bucket, the Auto_ML function automatically gets triggered in the cloud. This function has 3 main tasks (1) To generate the best possible model using AutoKeras (default hyperparameters: 200 epochs and 50 max trials); (2) To generate a Train–Test dataset for the upcoming explainability

Table 7
Comparison of proposed model with various train–test ratios for each case study.

Type of datasets	Train–Test ratio	Precision	Recall	Accuracy	f1-score
Case I: Breast Cancer Wisconsin Diagnosis	90:10	0.98	0.98	0.98	0.98
	80:20	0.97	0.99	0.98	0.98
	70:30	0.97	0.98	0.98	0.98
	60:40	0.97	0.96	0.97	0.97
Case II: Heart Disease Cleveland dataset	90:10	0.97	0.97	0.97	0.97
	80:20	0.95	0.96	0.95	0.96
	70:30	0.95	0.96	0.95	0.95
	60:40	0.94	0.94	0.94	0.94
Case III: Diabetes dataset	90:10	0.97	0.96	0.97	0.96
	80:20	0.96	0.96	0.97	0.96
	70:30	0.95	0.96	0.96	0.96
	60:40	0.91	0.92	0.92	0.92
Case IV: COVID-19 dataset	90:10	0.97	0.97	0.98	0.97
	80:20	0.97	0.97	0.98	0.97
	70:30	0.95	0.95	0.95	0.95
	60:40	0.94	0.95	0.96	0.95

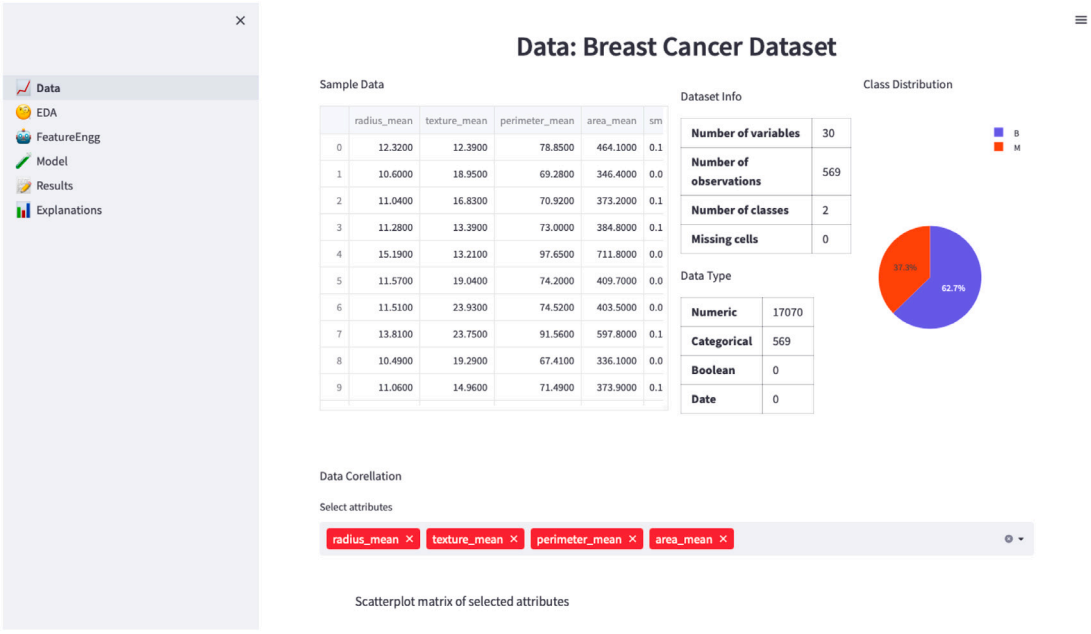

Fig. 7. Web Interface Showing Dataset Page with Breast Cancer Dataset.

function (default Train–Test split: 70-30); (3) To generate results in form of confusion matrics, ROC-AUC curve, PR-curve and classification reports which are plotted using Plotly to make the graphs interactive, as shown in Fig. 10; (4) To generate Tensorflow Logs of the AutoKeras model in the .zip format to the cloud bucket. This data can be accessed by the client-side application using TensorBoard as demonstrated in Fig. 11. Lastly, the LIME explainer cloud function(Ex_AI) is initiated which accesses this model file along with the Train–Test dataset file from the respective cloud bucket and generates 5 sample explanations which are then displayed on the user screen in an HTML format, as shown in Fig. 12. The Lime Explainer displays a feature value table and a plot showing which feature contributed to a particular decision and how much the contribution concerning other features. This means more the value in the feature table, the higher the impact of the feature in the predicted outcome by the model.

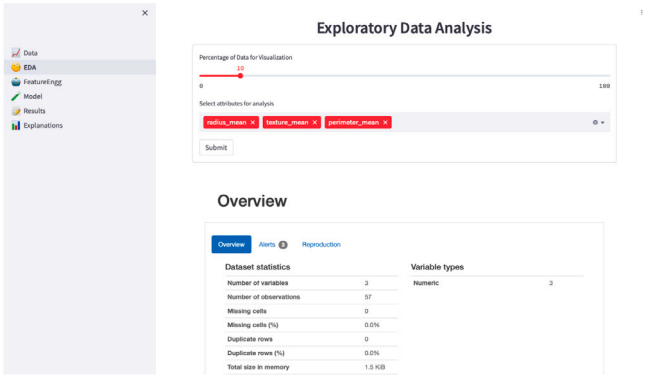

Fig. 8. The EDA page with Breast Cancer Dataset.

6. Discussion

By testing several instances of the function in different conditions, we have realised that the execution lasted almost 2 min on average. Although there are instances when the 'auto_ml' cloud function executes for up to 4 min, it is evident that the large dataset requires more memory and hardware capacity. On the other hand, we are also able to understand from Fig. 13 that many of the function instances have crashed due to several reasons. Two major reasons are the incompatibility of the dataset and the long execution time. As GCP has a limit on the

Fig. 9. The Feature Engineering page with Breast Cancer Dataset.

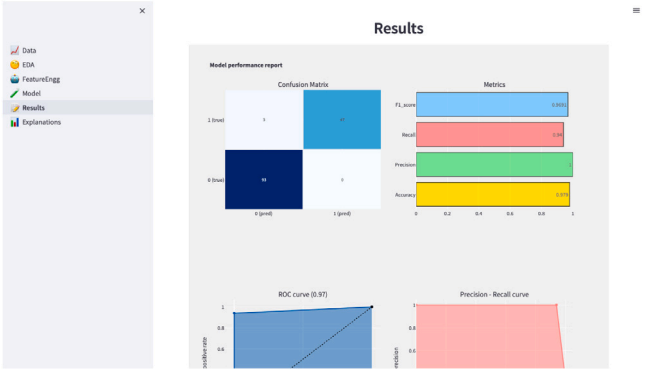

Fig. 10. The Results page with Breast Cancer Dataset.

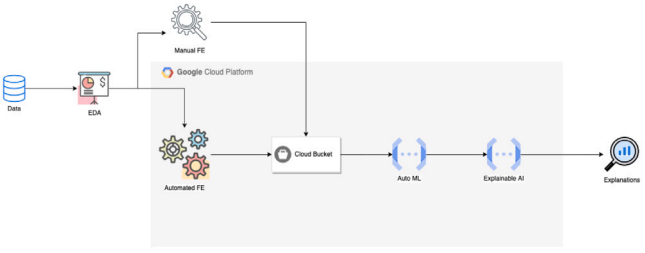

Fig. 11. The Model page with Breast Cancer Dataset.

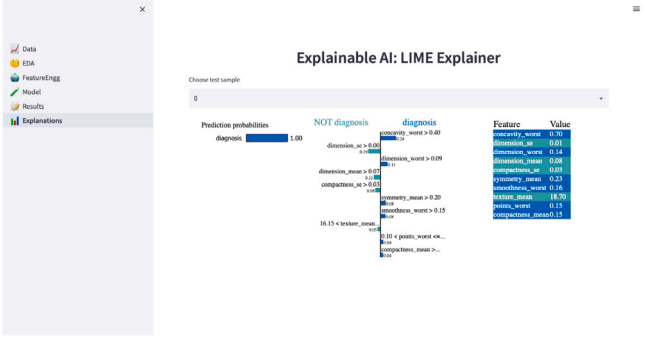

Fig. 12. The LIME Explainer page with Breast Cancer Dataset.

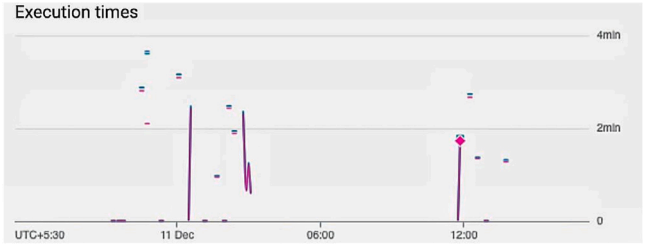

Fig. 13. Execution Time Graph of 'auto_ml' cloud function.

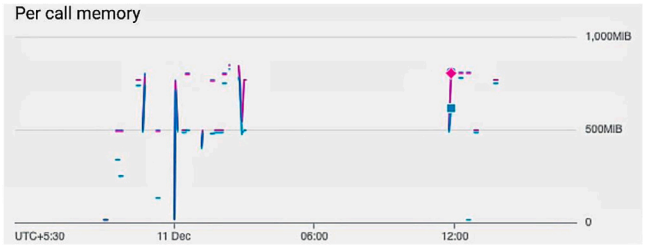

Fig. 14. Per call memory utilisation Graph of 'auto_ml' cloud function.

execution time of up to 5 min for cloud functions, the implementation of a large ML model with a long training time might be problematic. There are multiple ways to fix this issue, which include using multiple functions parallelly or consecutively for a long training period.

Not only does execution time matter for cloud functions, but the memory capacity of the functions also plays a very important role in a cloud environment. According to Fig. 14, most of the function execution took 500 MB to 1000 MB of memory bandwidth. The right balance of memory and processing capability is crucial for smooth function execution. As these are tabular datasets, this execution requires less amount of memory bandwidth in comparison to the image and time series datasets. That means while dealing with image and time-series data we must increase the function memory capacity. The memory bandwidth of the 'auto_ml' function is set at 4 GB at max so that any interruption can be prevented.

It has been concluded that the CloudAISim framework only needs its users to upload the dataset. After uploading the dataset, the CloudAISim framework performs tasks such as EDA, feature engineering, selection of the best machine learning/deep learning model, hyperparameter optimisation, result prediction, and explainability of the result. This makes the CloudAISim framework suitable for all non-experts and non-coders to use the power of AI without writing a single line of code. The CloudAISim framework has achieved a better accuracy of 98% in comparison with existing work [34], which has achieved 85.75% while considering the breast cancer Wisconsin dataset only.

7. Conclusions and future work

There has been substantial progress in globalizing the use of ML to non-experts in data analysis. However, these robust support systems behave like highly efficient black boxes since they do not offer comprehensive information on the recommendations and the internal workings of these models. Traditional ML methods do not always cater to the diverse nature of the datasets, and it is very difficult and tedious for a non-professional to design models specifically for specific datasets. Additionally, these powerful systems are mostly resource-intensive models, which is a big obstacle for the healthcare industry. Moreover, conventional machine learning approaches may not consistently address the varied characteristics of datasets, making it challenging and laborious for individuals without the expertise to create models tailored to particular datasets. In this paper, we have presented

a novel transparent serverless and self-explanatory AutoML framework called CloudAISim to overcome these issues. The proposed framework possesses the ability to autonomously select models in accordance with the given dataset and the task. Further, we designed a healthcare application as a case study in order to verify the scientific reliability of this proposed CloudAISim toolkit. The proposed healthcare application is promising for automated machine learning that has the potential to make AI accessible to non-technical individuals and healthcare professionals. An interactive web application that is user-friendly and effective has been created using the StreamLit Framework and deployed on both the Google App Engine and the StreamLit cloud. The model's maximum accuracy of 98% demonstrates that it is successful in achieving its objective. By providing a more effective and precise way to analyse medical data, the application has the potential to help both patients and healthcare professionals significantly.

7.1. Possible extensions of CloudAISim toolkit

In the future, the CloudAISim can be further extended in the following ways:

1. **Regression Model:** The model is primarily addressing the classification data problem as it is the most prominent use case in the healthcare domain. It can be extended into regression problems as well.
2. **IoT Applications:** The application of this model can be extended to further domains from agriculture to manufacturing and finance [44].
3. **Training:** The model can also be trained for incorporating different types of inputs like images, audio files, text data etc.
4. **Time-series Data:** StrutureDataClassifier and StrutureDataRegressor deal with tabular data and it does not count on the other forms of data such as time-series data [59]. Although time-series data are less frequent than image data in the context of health care, it is useful while measuring real-time patient activity.
5. **Data Variability**: Different forms of data like images and videos need more processing capability, which can be implemented using edge architecture and extended using a cloud model training schedule [60].
6. **Edge Computing**: Real-time disease detection using on-device model prediction can be implemented in edge-fog and cloud models.
7. **Privacy**: Federated learning [61] can be implemented for the privacy protection of the patients by which the learning model can improve from individual patients' model feedback.

Software availability

All the data, material and code involved in this research work are available for public use in Github at abhimanyubhowmik/CloudAISim.

CRediT authorship contribution statement

Abhimanyu Bhowmik: Writing – original draft, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Madhushree Sannigrahi:** Writing – original draft, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Deepraj Chowdhury:** Writing – original draft, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Ajoy Dey:** Writing – original draft, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Sukhpal Singh Gill:** Writing – original draft, Supervision, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] E. Brynjolfsson, L.M. Hitt, H.H. Kim, Strength in numbers: How does data-driven decisionmaking affect firm performance? 2011, Available at SSRN 1819486.
[2] W. Samek, K.-R. Müller, Towards explainable artificial intelligence, in: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Springer, 2019, pp. 5–22.
[3] Y. Bengio, Y. LeCun, et al., Scaling learning algorithms towards AI, Large-Scale Kernel Mach. 34 (5) (2007) 1–41.
[4] E. Lindholm, J. Nickolls, S. Oberman, J. Montrym, NVIDIA Tesla: A unified graphics and computing architecture, IEEE Micro 28 (2) (2008) 39–55.
[5] S.S. Gill, M. Xu, C. Ottaviani, P. Patros, R. Bahsoon, A. Shaghaghi, M. Golec, V. Stankovski, H. Wu, A. Abraham, et al., AI for next generation computing: Emerging trends and future directions, Internet Things 19 (2022) 100514.
[6] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, J. Zhu, Explainable AI: A brief survey on history, research areas, approaches and challenges, in: CCF International Conference on Natural Language Processing and Chinese Computing, Springer, 2019, pp. 563–574.
[7] R. Singh, et al., Edge AI: a survey, Internet Things Cyber-Phys. Syst. (2023).
[8] S. Iftikhar, et al., AI-based fog and edge computing: A systematic review, taxonomy and future directions, Internet Things (2022) 100674.
[9] G.K. Walia, et al., AI-empowered fog/edge resource management for IoT applications: A comprehensive review, research challenges and future perspectives, IEEE Commun. Surv. Tutor. (2023).
[10] A. Bhowmik, M. Sannigrahi, D. Chowdhury, A.D. Dwivedi, R.R. Mukkamala, DBNex: Deep belief network and explainable AI based financial fraud detection, in: 2022 IEEE International Conference on Big Data, Big Data, IEEE, 2022, pp. 3033–3042.
[11] S. Tuli, et al., Predicting the growth and trend of COVID-19 pandemic using machine learning and cloud computing, Internet Things 11 (2020) 100222.
[12] F. Desai, et al., HealthCloud: A system for monitoring health status of heart patients using machine learning and cloud computing, Internet Things 17 (2022) 100485.
[13] M. Garouani, A. Ahmad, M. Bouneffa, M. Hamlich, G. Bourguin, A. Lewandowski, Towards big industrial data mining through explainable automated machine learning, Int. J. Adv. Manuf. Technol. 120 (1) (2022) 1169–1188.
[14] M. Golec, et al., HealthFaaS: AI based smart healthcare system for heart patients using serverless computing, IEEE Internet Things J. (2023).
[15] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, F. Hutter, Auto-sklearn 2.0: Hands-free automl via meta-learning, 2020, arXiv:2007.04074 [cs.LG].
[16] R.S. Olson, J.H. Moore, TPOT: A tree-based pipeline optimization tool for automating machine learning, in: Workshop on Automatic Machine Learning, PMLR, 2016, pp. 66–74.
[17] L. Kotthoff, C. Thornton, H.H. Hoos, F. Hutter, K. Leyton-Brown, Auto-WEKA: Automatic model selection and hyperparameter optimization in WEKA, in: Automated Machine Learning, Springer, Cham, 2019, pp. 81–95.
[18] T. Swearingen, W. Drevo, B. Cyphers, A. Cuesta-Infante, A. Ross, K. Veeramachaneni, ATM: A distributed, collaborative, scalable system for automated machine learning, in: 2017 IEEE International Conference on Big Data, Big Data, IEEE, 2017, pp. 151–162.
[19] U. Khurana, H. Samulowitz, D. Turaga, Feature engineering for predictive modeling using reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, No. 1, 2018.
[20] F. Nargesian, H. Samulowitz, U. Khurana, E.B. Khalil, D.S. Turaga, Learning feature engineering for classification, in: Ijcai, 2017, pp. 2529–2535.
[21] M. Reif, F. Shafait, M. Goldstein, T. Breuel, A. Dengel, Automatic classifier selection for non-experts, Pattern Anal. Appl. 17 (1) (2014) 83–96.
[22] R. Vainshtein, A. Greenstein-Messica, G. Katz, B. Shapira, L. Rokach, A hybrid approach for automatic model recommendation, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1623–1626.
[23] M. Feurer, J. Springenberg, F. Hutter, Initializing bayesian hyperparameter optimization via meta-learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, No. 1, 2015.
[24] B. Bilalli, A. Abelló, T. Aluja-Banet, R. Wrembel, Automated data pre-processing via meta-learning, in: International Conference on Model and Data Engineering, Springer, 2016, pp. 194–208.
[25] B. Bilalli, A. Abelló, T. Aluja-Banet, R.F. Munir, R. Wrembel, Presistant: data pre-processing assistant, in: International Conference on Advanced Information Systems Engineering, Springer, 2018, pp. 57–65.
[26] I. Guyon, L. Sun-Hosoya, M. Boullé, H.J. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, M. Sebag, et al., Analysis of the automl challenge series, Autom. Mach. Learn. (2019) 177.

[27] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2) (2012).
[28] F. Hutter, H.H. Hoos, K. Leyton-Brown, Sequential model-based optimization for general algorithm configuration, in: International Conference on Learning and Intelligent Optimization, Springer, 2011, pp. 507–523.
[29] J. Snoek, H. Larochelle, R.P. Adams, Practical bayesian optimization of machine learning algorithms, Adv. Neural Inf. Process. Syst. 25 (2012).
[30] K. Tu, J. Ma, P. Cui, J. Pei, W. Zhu, Autone: Hyperparameter optimization for massive network embedding, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 216–225.
[31] B. Hall, D.D. Henningsen, Social facilitation and human–computer interaction, Comput. Hum. Behav. 24 (6) (2008) 2965–2971.
[32] M. Garouani, A. Ahmad, M. Bouneffa, A. Lewandowski, G. Bourguin, M. Hamlich, Towards the automation of industrial data science: A meta-learning based approach, in: ICEIS (1), 2021, pp. 709–716.
[33] J. Ferreira, C. Costa, Web platform for medical deep learning services, in: 2021 IEEE International Conference on Bioinformatics and Biomedicine, BIBM, IEEE, 2021, pp. 1727–1732.
[34] R.E. Shawi, K. Kilanava, S. Sakr, An interpretable semi-supervised framework for patch-based classification of breast cancer, Sci. Rep. 12 (1) (2022) 1–15.
[35] A.M. Alaa, M. van der Schaar, Prognostication and risk factors for cystic fibrosis via automated machine learning, Sci. Rep. 8 (1) (2018) 1–19.
[36] S. Alnegheimish, N. Alrashed, F. Aleissa, S. Althobaiti, D. Liu, M. Alsaleh, K. Veeramachaneni, Cardea: An open automated machine learning framework for electronic health records, in: 2020 IEEE 7th International Conference on Data Science and Advanced Analytics, DSAA, IEEE, 2020, pp. 536–545.
[37] Breast cancer wisconsin (diagnostic) data set, 1995, URL: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29.
[38] K.P. Bennett, O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, Optim. Methods Softw. 1 (1) (1992) 23–34.
[39] UCI Machine Learning Repository, Heart disease data set, 1998, URL: https://archive.ics.uci.edu/ml/datasets/heart+Disease.
[40] National Institute of Diabetes and Digestive and Kidney Diseases, Diabetes dataset, 2022, URL: https://www.kaggle.com/datasets/mathchi/diabetes-data-set.
[41] D. Chowdhury, S. Poddar, S. Banarjee, R. Pal, A. Gani, C. Ellis, R.C. Arya, S.S. Gill, S. Uhlig, CovidXAI: Explainable AI assisted web application for COVID-19 vaccine prioritisation, Internet Technol. Lett. (2022) e381.
[42] S.S. Gill, Quantum and blockchain based serverless edge computing: A vision, model, new trends and future directions, Internet Technol. Lett. (2021) e275.
[43] A. Bhowmik, M. Sannigrahi, D. Chowdhury, D. Das, RiceCloud: A cloud integrated ensemble learning based rice leaf diseases prediction system, in: 2022 IEEE 19th India Council International Conference, INDICON, IEEE, 2022, pp. 1–6.
[44] S.S. Gill, et al., Modern computing: Vision and challenges, Telematics Inform. Rep. (2024) 1–46.
[45] M. Hirzel, K. Kate, P. Ram, A. Shinnar, J. Tsay, Gradual automl using lale, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 4794–4795.
[46] L. Zimmer, M. Lindauer, F. Hutter, Auto-pytorch tabular: Multi-fidelity MetaLearning for efficient and robust AutoDLL. arXiv 2020, 2000, arXiv preprint arXiv:2006.13799.
[47] P. Gijsbers, J. Vanschoren, GAMA: Genetic automated machine learning assistant, J. Open Source Softw. 4 (33) (2019) 1132, http://dx.doi.org/10.21105/joss.01132.
[48] B. Celik, P. Singh, J. Vanschoren, Online automl: An adaptive automl framework for online learning, Mach. Learn. 112 (6) (2023) 1897–1921.
[49] M.T. Ribeiro, S. Singh, C. Guestrin, Model-agnostic interpretability of machine learning, 2016, arXiv preprint arXiv:1606.05386.
[50] T. Peltola, Local interpretable model-agnostic explanations of Bayesian predictive models via Kullback-Leibler projections, 2018, arXiv preprint arXiv:1810.02678.
[51] L. Wilkinson, T. Gathani, Understanding breast cancer as a global health concern, Br. J. Radiol. 95 (1130) (2022) 20211033.
[52] S. Mubarik, Y. Yu, F. Wang, S.S. Malik, X. Liu, M. Fawad, F. Shi, C. Yu, Epidemiological and sociodemographic transitions of female breast cancer incidence, death, case fatality and DALYs in 21 world regions and globally, from 1990 to 2017: an age-period-cohort analysis, J. Adv. Res. 37 (2022) 185–196.
[53] X. Zhou, C. Li, M.M. Rahaman, Y. Yao, S. Ai, C. Sun, Q. Wang, Y. Zhang, M. Li, X. Li, et al., A comprehensive review for breast histopathology image analysis using classical and deep neural networks, IEEE Access 8 (2020) 90931–90956.
[54] W. Majeed, B. Aslam, I. Javed, T. Khaliq, F. Muhammad, A. Ali, A. Raza, Breast cancer: major risk factors and recent developments in treatment, Asian Pac. J. Cancer Prev. 15 (8) (2014) 3353–3358.
[55] About heart disease, 2022, URL: https://www.cdc.gov/heartdisease/about.htm.
[56] Diabetes, 2022, URL: https://www.who.int/news-room/fact-sheets/detail/diabetes.
[57] M. Singh, et al., Quantifying COVID-19 enforced global changes in atmospheric pollutants using cloud computing based remote sensing, Remote Sens. Appl.: Soc. Environ. 22 (2021) 100489.
[58] D. Chowdhury, A. Das, A. Dey, S. Banerjee, M. Golec, D. Kollias, M. Kumar, G. Kaur, R. Kaur, R.C. Arya, et al., CoviDetector: A transfer learning-based semi supervised approach to detect Covid-19 using CXR images, BenchCouncil Trans. Benchmarks Stand. Eval. 3 (2) (2023) 100119.
[59] A. Bhowmik, et al., DYNAMITE: Dynamic aggregation of mutually-connected points based clustering algorithm for time series data, Internet Technol. Lett. (2022) e395.
[60] A. Bhowmik, M. Sannigrahi, P.K. Dutta, S. Bandyopadhyay, Using edge computing framework with the internet of things for intelligent vertical gardening, in: 2023 1st International Conference on Advanced Innovations in Smart Cities, ICAISC, IEEE, 2023, pp. 1–6.
[61] D. Chowdhury, S. Banerjee, M. Sannigrahi, A. Chakraborty, A. Das, A. Dey, A.D. Dwivedi, Federated learning based Covid-19 detection, Expert Syst. 40 (5) (2023) e13173.