MethodsX 12 (2024) 102696

Contents lists available at ScienceDirect

MethodsX

journal homepage: www.elsevier.com/locate/methodsx

An explainable AI-assisted web application in cancer drug value prediction ☆

Sonali Kothari [a,*], Shivanandana Sharma [a], Sanskruti Shejwal [a], Aqsa Kazi [a], Michela D'Silva [a], M. Karthikeyan [b]

[a] Symbiosis Institute of Technology – Pune Campus, Symbiosis International (Deemed University), Pune, India
[b] Senior Principal Scientist, Chemical Engineering and Process Development, NCL-CSIR, Pune, India

ARTICLE INFO

Method name:
CancerXAI is an explainable AI-based web application that explains the details of chemical entities and drugs for a dataset on lung cancer

Keywords:
Explainable Artificial Intelligence (XAI)
Machine learning
Healthcare
Cancer AI
Ensemble Learning

ABSTRACT

In recent years, there has been an increase in the interest in adopting Explainable Artificial Intelligence (XAI) for healthcare. The proposed system includes

- An XAI model for cancer drug value prediction. The model provides data that is easy to understand and explain, which is critical for medical decision-making. It also produces accurate projections.
- A model outperformed existing models due to extensive training and evaluation on a large cancer medication chemical compounds dataset.
- Insights into the causation and correlation between the dependent and independent actors in the chemical composition of the cancer cell.

While the model is evaluated on Lung Cancer data, the architecture offered in the proposed solution is cancer agnostic. It may be scaled out to other cancer cell data if the properties are similar. The work presents a viable route for customizing treatments and improving patient outcomes in oncology by combining XAI with a large dataset. This research attempts to create a framework where a user can upload a test case and receive forecasts with explanations, all in a portable PDF report.

Specifications table

Subject area:	Computer Science
More specific subject area:	*Artificial Intelligence for Drug Discovery*
Name of your method:	CancerXAI is an explainable AI-based web application that explains the details of chemical entities and drugs for a dataset on lung cancer.
Name and reference of the original method:	*NA*
Resource availability:	*NA*

☆ Related research article: NA

* Corresponding author.
E-mail address: sonali.kothari@sitpune.edu.in (S. Kothari).

https://doi.org/10.1016/j.mex.2024.102696
Received 11 February 2024; Accepted 2 April 2024
Available online 3 April 2024

Method details

Introduction

Cancer is one of the world's most significant causes of death, and early detection is critical for successful treatment. Medical imaging methods such as mammography and computed tomography (CT) have been widely employed for cancer detection. However, image interpretation can be subjective and error-prone, resulting in missing or misdiagnosed instances. AI models have demonstrated promising results in automating cancer detection from medical imaging, but their lack of interpretability has hampered their broad use in clinical use. Explainability is a quality of an AI-powered system that allows a user to retrace how a specific AI arrived at the predictions that were produced [1].

Traditional AI is incapable of answering "why" questions. This need for explanation gave rise to XAI, a new discipline of AI research. It focuses on providing accurate predictions and insights into the decision-making process, boosting the model's credibility and transparency. As a result, XAI research has become a key priority in academia and industry [2]. This is especially essential in medical applications, where misdiagnosis can have profound implications. XAI models are designed to provide more transparent and accessible findings so that users can understand how an AI system arrived at a specific conclusion. XAI aims to bridge the gap between machine decision-making and human understanding, prioritizing clarity and credibility in AI models.

This project intends to make XAI accessible to medical researchers and doctors who are unfamiliar with XAI or who, as one should, evaluate the results provided by AI models with skepticism. What makes the model's prediction credible in the absence of an explanation? The project will attempt to design a framework that is ambiguous to the dataset or the models and more focused on giving the users an experience that encourages them to explore the application of AI in their domain, with the model taking responsibility for its predictions.

Literature review

The literature review conducted for the project was extensive, ranging from searching for keywords such as "XAI in healthcare," "LIME Explainer," "Explainable AI in Cancer," "AI for Healthcare," "AI for cancer," and similar searches. The Literature Review led to the formulation of the path of solution for the problem at hand and project structuring to achieve the same with the resources at the disposal of the researchers.

The authors of [3] discuss the importance of explainability in clinical decision support systems (CDSS) powered by artificial intelligence (AI). It highlights the need for first- and second-level explainability in AI-based CDSS to ensure patient safety and informed decision-making. The paper acknowledges the trade-off between performance and explainability in AI models, highlighting challenges for developers. It concludes that overlooking explainability in CDSS poses ethical risks and advocates for multidisciplinary collaboration. Achieving explainability in AI-based CDSS is vital for discerning critical features and patterns and ensuring patient safety. While clinical validation is a primary requirement, explainability, often considered secondary, plays a pivotal role. In this context, although inherent explainability is preferred, modern algorithms like artificial neural networks (ANNs) are justified for their superior performance.

Wang et al. [4] discusses the concept of XAI in the healthcare domain. XAI techniques are proposed to address the lack of transparency and trust in AI systems. The paper highlights the benefits of using XAI, including increased transparency, result tracing, and model improvement. The proposed approach suggests leveraging existing XAI models with clinical knowledge to obtain more benefits in AI-based systems. The strategy involves capturing health information, using trained AI models to predict abnormalities or diseases, generating explanations using XAI methods, analyzing these explanations with the help of clinicians, and using the insights gained to improve predictions and recommendations. The paper also mentions the challenges of XAI techniques, like the development of user interfaces and the increased computational cost. Overall, the paper highlights the importance of XAI in healthcare and its potential to enhance decision-making, accountability, and trust in AI-based systems.

The authors of [5] discuss the importance of explainability in AI algorithms, particularly in the healthcare domain. It presents various methods for assessing explainability, including functional, operational, usability, safety, validation, simulatability, decomposability, algorithmic transparency, correctness, comprehensiveness, coherence, performance, model comprehensibility, explanation, and user category. Additionally, the paper highlights the balance between explainability and accuracy/performance, noting that the need for explainability can sometimes lead to a reduction in model performance. The authors suggest creating open and honest education to end users on the strengths and weaknesses of AI algorithms to establish trust. Overall, the paper provides valuable insights into the challenges and considerations surrounding explainability in AI algorithms, particularly in healthcare.

The experimental results of [6] showed that the revised papers with improved explanations received higher average scores for each AI application. This indicates that using common expressions and color management techniques effectively improved the explainability of the clinic recommendation method. However, these findings may not establish the theoretical superiority of these XAI tools. Simple, domain-agnostic XAI tools like color management and common expressions were practical across all AI applications, with the most significant improvement seen in explaining Fuzzy Linear Regression-Fuzzy Intersection (FLR-FI) attributed to the traceable aggregation technique. Applying common expressions and color management to explain Artificial Neural Network (ANN) / Deep Neural Networks (DNN) architecture and example propagation proved effective. In conclusion, incorporating common expressions, color management, and traceable aggregation, the proposed methodology enhances AI application explainability in healthcare.

The study in [7] focuses on XAI in the healthcare domain. The study's primary goals include identifying and categorizing XAI methods, conducting a literature review focusing on healthcare, and ascertaining the challenges and problems of XAI in healthcare.

The study utilizes the Prisma Model to include and exclude records and explores various academic database search engines to retrieve relevant articles. The paper emphasizes the importance of model transparency in safety-critical applications such as the medical domain. It highlights the need for further research to understand the reasons behind the decisions made by Machine Learning (ML) models.

The research in [8] discusses the importance of XAI in the healthcare industry. It stresses the importance of interpretability and transparency in AI models for effective decision-making. The paper shows case studies, including forecasting heart stroke risk and improving decision-making in colorectal cancer diagnosis using XAI. Toolkits like LIME, SHAP, and Yellowbrick are explored to evaluate model explainability. Overall, the research emphasizes the necessity of transparent AI models in healthcare to instill confidence in professionals and stakeholders. It highlights XAI's accuracy performance and justifies predictions through critical input data identification. The study concludes by proposing that XAI can enhance AI adoption in healthcare by addressing accountability, clarity, and trust challenges in decision-making.

The discussion in [9] focuses on applying XAI methods in the healthcare system, specifically older adults's healthcare. The paper notes challenges in lacking a robust XAI framework, hindering clear conclusions on method performance. The authors stress the need for more detailed evaluation criteria, as current ones are insufficient. The paper highlights the reliability and usability of visualization models for brain tumor diagnosis but acknowledges limitations like imbalanced datasets in medical image detection using XAI. Overall, it underscores the significance of XAI in healthcare, urging further research to overcome identified challenges, enhance evaluation criteria, and address dataset issues for more effective models.

The authors of [10] present a study on predicting invasive ductal carcinoma (IDC) in breast cancer patients using machine learning algorithms. The authors used a consensus procedure and four classifiers to evaluate the IDC prediction. XGBoost was the top-performing classifier, achieving high accuracy in 5- and 10-year predictions. They additionally carried out an explainable analysis to ascertain the classifications of individual patients and comprehend how features influenced the model's decision. The study's results indicate that while age and the number of eradicated lymph nodes show a negative contribution to the occurrence of IDC, certain features such as ER, Ki67, lymph node status, tumor diameter, grading, and multiplicity show a positive contribution.

The proposed XAI method in [11] addresses the lack of explainability in genomic medicine by combining high accuracy and explainability. It uses a knowledge graph and deep tensor to generate physician-friendly explanations, focusing on clinically relevant variants. The results indicate strong performance, comparable to or better than decision trees. Deep tensor shows consistent performance and superior generalization compared to random forests. The paper also addresses challenges in evaluating decision tree explanations, emphasizing the efficiency of the XAI approach in addressing these issues.

The research conducted in [12] discusses the importance of choosing the most appropriate model for AI solutions to maximize the chances of success. It emphasizes the need to define and characterize the issues that must be addressed before implementing AI. The research also highlights the challenge of unbalanced data distribution in classification algorithms and suggests resampling methods like random over-sampling and under-sampling. It goes over the importance of feature selection and the different approaches that can be used. Cross-validation is mentioned as a validation technique to evaluate AI models on limited samples. The research also touches upon the use of databases and knowledge bases for comprehensive information in the field of oncology. It mentions the role of precision oncology decision support teams in manually reviewing literature.

The study in [13] investigated the spatial variability of risk factors contributing to lung and bronchus cancer (LBC) mortality rates in the United States using a stack-ensemble machine learning model and XAI techniques. The most significant risk factor for LBC mortality, according to the model, is the prevalence of smoking, especially in counties in the Mississippi Valley, the Appalachian region, and the southern states. The results of this investigation show how effective XAI methods and stack-ensemble ML models can be in measuring and illustrating the geographical variability of risk factors influencing LBC death rates. These observations can assist in informing public health initiatives and policies intended to lower the incidence of lung and bronchus cancer in various US regions.

The authors of [14] describe a novel method for retinoblastoma detection that combines XAI techniques with deep learning models. They use LIME and SHAP to interpret the model on a large dataset, demonstrating superior performance compared to accuracy, precision, recall, and F1 score benchmarks. SHAP helps identify key features that distinguish retinoblastoma from normal tissue in fundus images. The study offers a promising method for retinoblastoma detection, enhancing model interpretability. Further research could contribute to developing more precise diagnostic tools for retinoblastoma identification.

The problem of model interpretability for black-box machine learning models is discussed in [15]. It draws attention to the increasing efforts to create techniques for deciphering these models, primarily through LIME and other post hoc explanations based on perturbations. The instability issue in LIME is addressed with S-LIME. The study underscores the importance of reproducible and stable model explanations for user trust and understanding. S-LIME is introduced to address the stability requirement, highlighting the variation in explanations for the same instance across different model types and urging practitioners to be mindful of this dependency.

XAI is crucial for oncology research, according to the authors of [16]. XAI provides interpretability and transparency to machine learning models regarding cancer treatment, aiding comprehension and decision-making. It facilitates feature construction, selection, and determination of prognostic thresholds, enhancing overall model confidence. The authors highlight the potential to improve patient outcomes and influence policy by making ML models more understandable for non-technical users. They recommend utilizing high-performance ML techniques with interpretability tools such as SHAP and LIME. They advocate for inherently interpretable methods like regression models and decision trees, emphasizing the importance of interpretability and predictive accuracy in oncology research.

These research papers employ advanced deep neural networks to explore challenging medical phenomena, from alanine dipeptide isomerization to predicting drug activity against breast cancer cells. They utilize cutting-edge techniques like Support Vector Regression and Convolutional Neural Networks (CNNs) to offer transparency and insight into AI decision-making. Emphasizing transparency,

trustworthiness, and interpretability, these studies underscore the crucial role of XAI in transforming healthcare. XAI provides innovative solutions to longstanding challenges, paving the way for a more efficient, transparent, and accountable AI-driven healthcare ecosystem.

Methodology

Research hypothesis

The research hypothesis lays the foundation for thoroughly examining the efficacy of an XAI solution for cancer chemical composition regression prediction. It emphasizes the qualitative dimensions of interpretability, ethical considerations, and the quantitative aspects of prediction accuracy. By shedding light on the potential advantages of open and understandable AI models for cancer research and clinical practice, the study intends to contribute to the expanding field of AI in healthcare.

Null Hypothesis (H0): There is no significant difference in the predictive performance of an XAI solution for cancer chemical composition regression compared to traditional machine learning models.

Alternative Hypothesis (H1): The XAI solution for cancer chemical composition regression will demonstrate significantly improved predictive accuracy, transparency, and interpretability compared to traditional machine learning models.

This research attempts to examine the hypothesis and further elaborate on them as follows:

(i) Predictive Performance Comparison: The study seeks to determine whether an XAI model created for cancer chemical composition regression prediction could outperform conventional machine learning models. The null hypothesis claims no discernible difference exists between the XAI model's and conventional models' forecasting abilities.
(ii) Transparency and Interpretability: In contrast to conventional models, the XAI model is predicted to have more transparent predictions and easier to understand. According to this, the XAI model will offer more precise insights into the chemical makeup that affects cancer outcomes.
(iii) Higher Predictive Accuracy: The XAI model will produce more accurate predictions according to the alternative hypothesis. This includes the capacity to anticipate cancer chemical compositions more accurately, improving decision-support for researchers and healthcare practitioners.
(iv) Interpretable Features: The XAI solution is anticipated to pinpoint and emphasize the significance of chemical characteristics that influence cancer outcomes. According to the alternative theory, these interpretable characteristics will offer significant new information on how the chemical composition affects cancer development.
(v) Real-World Applicability: According to the research hypothesis, the XAI system can make clinical decisions and advance cancer research. It assumes that the model's outputs will be usable, allowing researchers and healthcare professionals to make wise choices.
(vi) Ethical Considerations: The study will investigate the ethical implications of AI in healthcare as part of the alternative hypothesis. It will examine if implementing an XAI solution adheres to the moral principles and recommendations for such applications.

Functional implementation design

The prototype's block diagram (Fig. 1) illustrates a comprehensive, well-structured system. The front end, powered by Streamlit, is integrated with the back end, ensuring a smooth and efficient data flow. A user-friendly interface on Webpage 1 enables users to download a CSV template, view it, and upload data to the backend. The uploaded data is stored in the tempDir directory and is ready for backend processing. Dynamic model selection, which enables users to modify ML models, is introduced on webpage 2. Users can receive a PDF with anticipated outcomes and LIME graphs for transparency from webpage 3. This block diagram illustrates a user-focused, technically sound, and attractive user interface.

System backend development

The backend of the prototype required a modular structure and a plug-and-play architecture that would cater to the maintenance and update of this ever-evolving field of medical research. The architecture of the backend has been brainstormed upon throughout the build process, and the following points were emphasized in the brainstorming sessions and implemented upon -

(i) Implementing OOP - Each functionality was modularized into methods, and methods of similar use cases were encapsulated into appropriate classes. The classes inherited methods and attributes from each other, making the source code faster and much more efficient to develop.
(ii) Documentation of The Code - Each class and method has been well documented according to the SciPi convention, making it ideal for open-source distribution and use. The params included in the docstrings are {Description, Parameters, Returns, Example, Notes}.
(iii) Pipelining the Internal Process - Automating the internal processes requires intricate pipelines to pipe the result of each thread to its following thread without manual testing and intervention. The output of each module defined further in this section has been piped into the input of the next module.

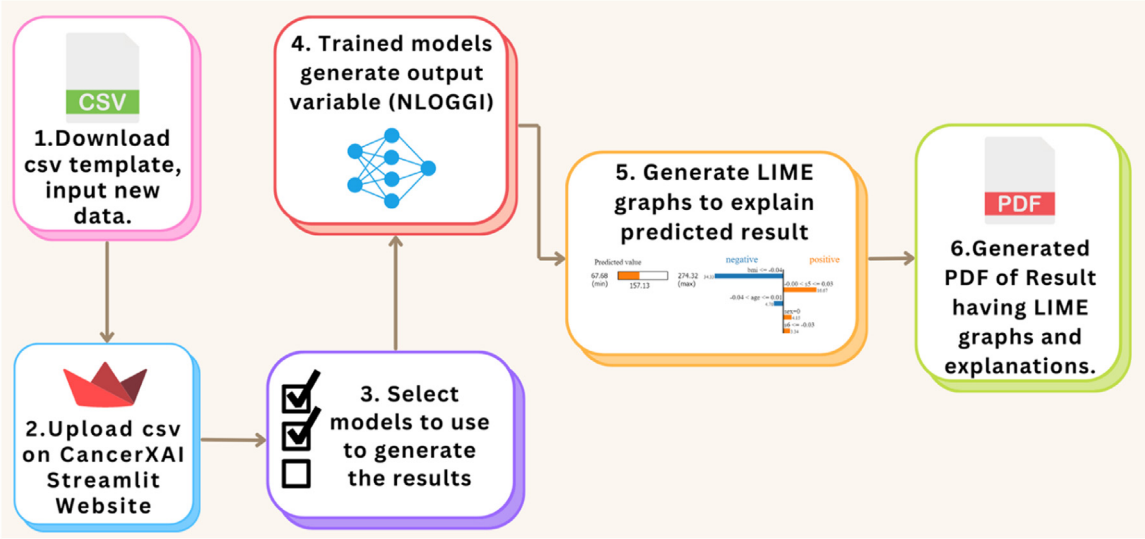


Fig. 1. Flow diagram of proposed methodology.

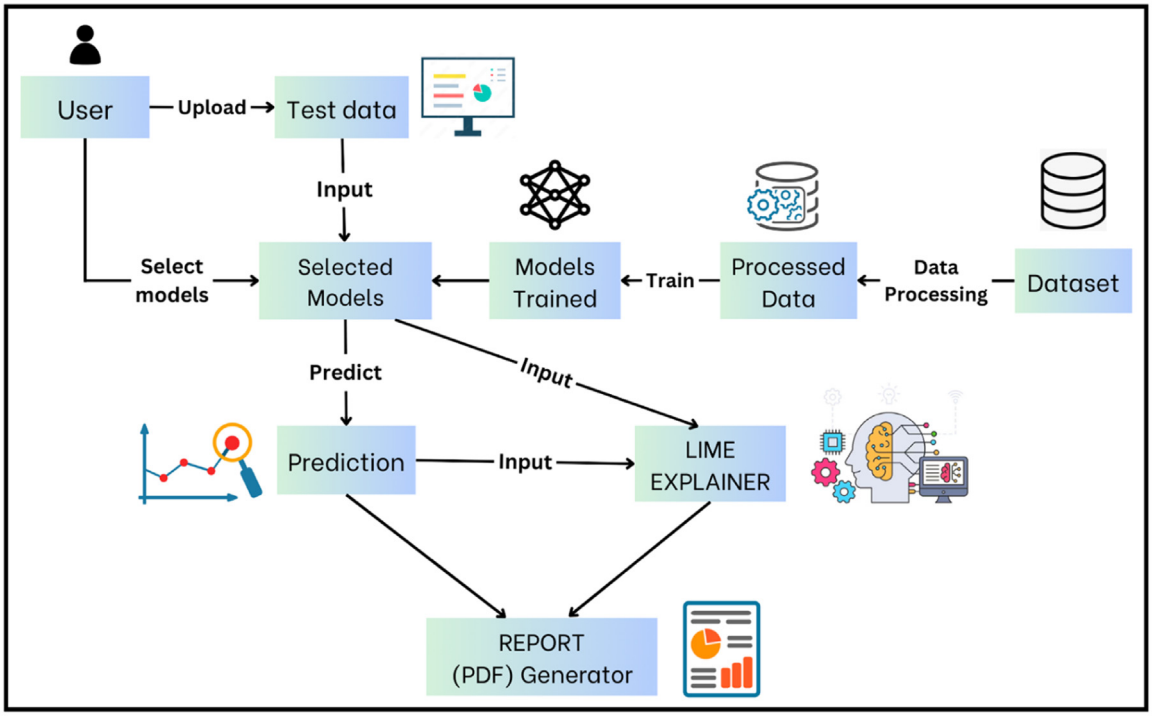


Fig. 2. Data flow diagram of the backend.

The backend processes can be broadly divided into three submodules - Data ELT (Extract, Load, Transform), Machine Learning - Model Training, and Final PDF Report Generation (Fig. 2).

1. Data ELT (Extract, Load, Transform)

CSIR-NCL provided the original training dataset in collaboration with Symbiosis Institute of Technology, Pune. The dataset initially contained 1,86,491 scientific observations that the scientists at CSIR-NCL recorded. The values that were categorized under 'Null,' 'NaN', and 'Infinite' were removed under standard machine learning practice as the models can't train on said values. After initial data cleaning, the dataset consists of 186,375 entries (rows) and 202 attributes (columns). The attributes convey vital information about the drug combination used to treat the site at which the cancerous cells are metastasizing, including, but not limited to – the chemical

composition of the individual drug in the combinatorial drug therapy (density, diameter, chiral, pH), the site of cancer, the cell type affected, etc. EDA was conducted on the dataset and with an ensemble feature selection method, which included weighted scores for the Pearson Correlation Test, Chi-Squared Test, Recursive Feature Elimination(RFE) with Logistic Regression, RFE with Random Forest, RFE with Lasso Regression, and RFE with LightGBM. The most influential features were extracted and noted for model training. The features obtained were converted to columns of a blank 'CSV' file stored in the backend, which could be downloaded by the user on the front end. The blank CSV file would act as a template, and users can fill in their test data to obtain the prediction and explanation for the projections in further steps.

The uploaded CSV file by the user is then sent to the backend for validation and preprocessing. The CSV file must be in the predefined format; if any modifications are done to the column headings, such as the user adding new columns by themselves or removing specific columns without express notification to the admin, the CSV file is to be rejected. This is done so that the models pre-trained to the particular data format aren't subjected to an unknown one. This ensures the integrity of the models trained and prevents any unexpected errors during runtime.

The uploaded data is also preprocessed. The cells containing NaN or Infinite are handled as defined by the admin; the preprocessing pipeline supports handling the said values with the mean, median, or mode of the same column by the original subset of data used for model training. Categorical data is checked against the original training set's possible values; valid values undergo OneHotEncoding or Ordinal Encoding. To improve the space complexity of the preprocessing pipeline, instead of creating multiple data frames for each stage, only two data frames were initialized at the start of the preprocessing pipeline and reused in each step of the pipeline. This reduced the runtime space complexity by at least 45 % in multiple testing iterations. At the end of the preprocessing pipeline, the preprocessed data is saved as a CSV file in the temp runtime folder for further implementation.

2. Machine Learning - Model Training

In the model training and development phase, a select number of regression models were trained on the original training dataset. Due to the target feature being continuous data of the numerical type, the application of regression models was the most suited to the training dataset. The regression models that were trained are listed below.

(i) Linear Regression
(ii) Random Forest Regressor
(iii) XGB Regressor
(iv) Lasso Regression
(v) Gradient Boosting Regression Model
(vi) Light GBM Regression Model
(vii) Stacked Ensemble Regression Model

The listed models' base models were trained, and the results were noted. A hashmap containing the various ranges of individual parameters was defined, and the models were hyperparameter-tuning using the GridSearchCV algorithm.

(i) Linear Regression

Linear regression is a fundamental statistical and machine learning method that models the relationship between one or more independent variables and a dependent variable using a linear equation. It is widely applied for predictions and inferences due to its simplicity and ease of interpretation. In various fields, linear regression is a foundational tool in machine learning and regression analysis. Linear regression is characterized by a linear equation, commonly known as the regression equation. The mathematical notation for the same is as follows.

$$Y = \beta_0 + \beta_1 X + \varepsilon \tag{1}$$

Where,

Y stands for the dependent variable, representing the predicted or explained outcome.

X represents the independent variable, also referred to as the predictor or feature.

β_n is the intercept, indicating the Y value when X is zero.

β_1 is the regression coefficient, showing the change in Y for a one-unit change in X.

ε is the error term, accounting for the unexplained variability in Y in the linear relationship with X.

Linear regression, specifically Ordinary Least Squares (OLS), aims to minimize the sum of squared differences between predicted and observed values. Coefficients 'a' and 'b' are determined to describe the relationship, enabling predictions and further analysis. Various regression methods, such as multiple linear and polynomial regression, build upon this foundation in regression analysis and machine learning. The base model of linear regression was first trained with default input parameters, which yielded a good fit. Subsequently, the tuned model was trained on the same training data using the following parameters.

The params grid specified for grid search cv are {'positive': [True, False], 'fit_intercept': [True, False], 'copy_X': [True, False]}

Best Params for LR Model = {'copy_X': True, 'fit_intercept': True, 'positive': False}

(ii) Random Forest Regressor

The Random Forest Regressor is a powerful ensemble learning technique for regression tasks, leveraging multiple decision trees for accurate predictions. It excels in reducing overfitting, enhancing accuracy, and handling complex relationships in datasets. It employs two key strategies to maximize efficacy: bagging (Bootstrap Aggregating) and feature selection. Bagging involves creating

diverse subsets (bags) from the dataset and training trees on each with random sampling. Feature selection uses different feature subsets at each node, contributing to a robust model with varied trees.

The formula for the Random Forest Regressor is a weighted average of the predictions from individual decision trees. Given 'N' decision trees in the forest, each tree provides a prediction. $f\frac{1}{N}\sum_{i=1}^{N} Y_i(X)$

In this equation:

Y_{rf} (X) represents the predicted value for input X by the Random Forest Regressor.

N is the total number of decision trees in the forest.

Y_i (X)signifies the prediction made by the i^{th} decision tree.

In the implementation, the base model of the random forest regressor (100 estimators) was trained, and the model gave the best results compared to other base models. The tuned model had different variations of estimators, namely 200, 500, and 1200 estimators, out of which the 1200 estimators version gave the best results.

The best hyperparameters of the Random Forest Regressor were chosen using Halving Grid Search CV. The n_estimators parameter was incrementally set in the range of 100 to 1200 by 100 as part of the scope of the experiment. The increase in the n_estimators is a tradeoff between accuracy and computational power.

The decrease in the error rate upon increasing n_estimators had plateaued around 1000 estimators. The model's global minima of error in its error vs. n_estimators plot may correspond to a higher value of n_estimators and have a different value to its local minima. Hence, the number of n_estimators was capped at the error rate plateau.

(iii) XGB Regression Model

XGBoost is a boosting algorithm that sequentially builds decision trees to correct errors made by earlier trees. It uses gradient boosting to minimize a loss function by adjusting predictions. Performance enhancements include regularization, sparse awareness, and handling missing values. XGBoost is known for its scalability, speed, adjustable hyperparameters, and pre-configured model tuning options. The prediction formula for XGBoost Regression can be defined as:

$$XGB(X = \phi(X) + \sum_{i=1}^{N} f_i(X)$$

In this equation:

Y_{XGB}(X) represents the predicted value for input X by the XGBoost Regression model.

ϕ(X) denotes the model's initial prediction for X.

N is the total number of decision trees in the ensemble.

$the \sum_{i=1}^{N} f_i(X)$ signifies the prediction made by the ith decision tree.

The XGB base model was trained with default parameters and ranked 3rd in the base models. The following ranges were defined for the parameters of the XGB model in the parameter grid.

Parameter grid defined for XGB Regression Model = {

'min_child_weight': [1,5,10],

'gamma': [0.5, 1, 1.5, 2, 5],

'subsample': [0.6, 0.8, 1.0],

'colsample_bytree': [0.6, 0.8, 1.0],

'max_depth': [3–5],

'learning_rate': [0.1, 0.01, 0.001]

}

The best parameters selected by Grid Search CV from the defined parameter grid are as follows.

Parameters for XGB Tuned model = {

'colsample_bytree': 0.6,

'gamma': 5,

'learning_rate': 0.1,

'max_depth': 5,

'min_child_weight': 1,

'subsample': 1.0

}

(iv) Lasso Regression Model

Lasso Regression, short for Least Absolute Shrinkage and Selection Operator, is a fundamental machine learning technique for regression analysis, especially when feature sparsity is expected or overfitting must be prevented. Fundamentally, a regularization term, typically the Mean Squared Error (MSE) loss, is included in the linear regression cost function to avoid overfitting. This encourages the model to shrink less significant feature coefficients to zero, effectively performing feature selection and preventing overfitting.

The Lasso Regression formula can be defined as follows:

Cost Function = $1/2m \sum\_(i = 1)^{m}(y_i - \hat{y}_i)^2 + \lambda \sum\_(j = 1)^{n}|\theta_j|$

Where:

Cost Function is the objective to minimize. m represents the number of samples in the training data. n is the number of features i

y_i stands for the actual target value of the ith sample.

$\hat{y}_i$ is the predicted value for the ith sample.

θ_j is the coefficient associated with the jth feature.

λ is the regularization hyperparameter that controls the strength of regularization. A higher value of λ encourages sparsity in the model.

Lasso Regression performs a trade-off between fitting the data well and keeping the model simple by regularizing and selecting a subset of features.

Parameter grid defined for lasso regression model = {
'lasso_alpha': [0.1, 0.5, 1.0, 2.0, 5.0],
'lasso_max_iter': [1000],
'lasso_fit_intercept': [True, False],
'lasso_selection': ['cyclic', 'random'],
'lasso_tol': [1e-3, 1e-4, 1e-5]
}

The best fit tuned model that was obtained from the parameter grid by Grid Search CV was given by
Lasso(lasso_alpha=0.1, lasso_max_iter=1000, lasso_fit_intercept=True, lasso_selection='random',
'lasso_tol': 1e-5)

(v) Gradient Boosting Regression Model

Gradient Boosting Regressor is a robust machine learning algorithm that solves regression problems through an ensemble of decision trees. Fundamentally, it combines several weak learners—usually in the form of decision trees—to create a robust predictive model using the ensemble technique of boosting. By iteratively optimizing the model based on the mistakes made in prior iterations, the Gradient Boosting Regressor aims to minimize a particular loss function [24]. GBR works by initializing with a basic model, often the mean of the target variable. It calculates the gradient of the loss function to determine the error direction and size. A new decision tree is then constructed to predict the negative gradient, and its contribution is added to the existing prediction, scaled by a learning rate. This process is iteratively repeated to minimize residual error.

The formula for the Gradient Boosting Regressor's prediction at each stage (after the mth iteration) can be represented as

$\hat{y}^{(m)}(x) = \hat{y}^{(m-1)}(x) + \eta.h^{(m)}(x)$

Where:

$\hat{y}^{(m)}(x)$ is the prediction after the m-th iteration.

$\hat{y}^{(m-1)}(x)$ is the prediction from the previous iteration.

η is the learning rate, controlling the step size in each iteration. $h^{(m)}(x)$ represents the contribution of the m-th decision tree

In essence, the Gradient Boosting Regressor iteratively improves its predictions by adding decision trees to reduce the errors made in previous iterations. The model adapts to the data, gradually learning complex relationships and delivering highly accurate predictions.

Parameter grid defined for Gradient Boosting Regression Model = {
'learning_rate': [0.01,0.02,0.03,0.04],
'subsample' : [0.9, 0.5, 0.2, 0.1],
'n_estimators' : [100,500,1000, 1500],
'max_depth' : [4,6,8,10]
}

The best fit tuned model that was obtained from the parameter grid by Grid Search CV was given as:
GradientBoostingRegressor(learning_rate=0.01, max_depth=8, n_estimators=500, subsample=0.2)

(i) Light GBM Regression Model

Light GBM is a high-performance gradient-boosting framework designed for large datasets with diverse features. It prioritizes speed and scalability using a histogram-based method for optimal splits during training. Unlike traditional approaches, it emphasizes maximizing leaf growth over tree levels, enhancing model effectiveness, and reducing computing costs. Exclusive Feature Bundling (EFB) and Gradient-based One-Side Sampling (GOSS) improve performance by reducing redundancy and overfitting. Light GBM excels in regression tasks, outperforming alternative methods, thanks to its adaptability to large, high-dimensional datasets, distributed computing compatibility, parallel processing, GPU support, and seamless integration with popular ML frameworks.

The formula for Light GBM's regression model is based on its objective function, typically Mean Squared Error (MSE) for regression tasks:

Objective Function = Σ [2.w (y - $\hat{y}$) - w.g.l - w.h. (l^2 + c)] + $\Omega(T)$ + $\Omega(\Theta)$

Where:

Σ represents the summation of all data points. w is the weight of a data point. y is the actual target value.

$\hat{y}$ is the predicted target value. g and h are the first and second-order gradients, respectively. l is the current leaf value. c is a constant term.

$\Omega(T)$ is the regularization term for the leaf values.

Ω(Θ) is the regularization term for the split values.

(i) Stacked Ensemble Model

A stacked ensemble, or stacked generalization, is an advanced machine learning method that enhances predictive models by combining the strengths of multiple base models. It employs a meta-model to integrate predictions from different base models efficiently, improving overall performance by capturing complementary information. There are other ways to sum up how a stacked ensemble operates:

Basic Model Training: A variety of primary models are trained on the same training dataset in the first phase. These foundational models can be any machine learning approach, including decision trees, random forests, regression models, etc.

Base model predictions are made using the validation dataset (out-of-fold predictions) after the base models have been trained. The results of each base model's predictions are then pooled to produce a new dataset, each representing a feature.

The formula for the stacked ensemble's final prediction is essentially a linear combination of the projections from the base models:

$\hat{y}_{\text{stacked}} = \sum\_(i = 1)^{N} \alpha_i . \hat{y}_i$

Where:

$\hat{y}_{\text{stacked}}$ is the final prediction made by the stacked ensemble.

N is the number of base models.

$\hat{y}_{\text{i}}$ is the prediction from the i^{th} base model.

α_{i} is the weight assigned to the i^{th} base model's prediction

The αi values are determined during the meta-model training to combine base models' predictions effectively. The second one proved the most effective of the two iterations of stacked ensemble model training. The first iteration used all individual models and the RF Model (1200 estimators) as the meta-learner. The second iteration featured the stack's RF Model (1200 estimators) as the meta-learner with the GBR Model, RF Model (200 Estimators), and XGB-tuned model.

3. Model Packaging & Generating Predictions

Models in the section were pickled for production, with file paths stored in a hashmap using model names as keys. This enables easy instantiation based on selected models on the front end. Preprocessed data is used for predictions when unpacking pickled models. LIME Explainer is then applied to provide explanations for forecasts. LIME Explainer doesn't directly decipher black-box models; instead, it creates a simplified, locally faithful linear model to approximate the black box's behavior for a specific data point. It selects a data point, tweaks it, observes the black box's responses, and builds a dataset. Then, it fits a simple linear model to this dataset, revealing local rules. LIME also assigns weights to features in the model, indicating the significance of each input variable for a specific prediction. LIME unveils local insights into the black-box model's decision-making process. LIME can also employ regularization and sampling to improve generalization and stability in its predictions when the underlying model has a high dimensional space.

The LIME Explainer processes preprocessed test data to generate predictions and analysis, including a LIME bar chart. The chart is saved as a 'PNG' image in a temp folder, identified by the dataset row number and model used. Feature ranges and weights for each feature in the test case are stored in a nested hashmap, saved as a temp JSON file, or in a NoSQL database like MongoDB for convenient retrieval. The challenge with LIME Explainer is the inability to adjust chart size, leading to paper readability issues. This was addressed by employing runtime polymorphism (method overloading). By inheriting the 'Explainer' class and overriding the 'as_pyplot_figure' method, a new class, derived from 'LimeTabularExplainer,' was created. This allows customization of the chart size, enhancing readability for papers with numerous features.

Pseudocode for the same would be - from lime.lime_tabular import LimeTabularExplainer from lime.lime_tabular import TableDomainMapper from lime.lime_tabular importExplanation import matplotlib.pyplot as plt class CustomExplanation(Explanation): def as_pyplot_figure(self, figsize, dpi, **kwargs):

rest of the code as it is fig = plt.figure(figsize=figsize, dpi=dpi)

Adjust the figsize

rest of the code as it is return fig class CustomTabularExplainer(LimeTabularExplainer): def explain_instance(self, data_row, predict_fn, num_features, **kwargs): explanation=super().explain_instance(data_row, predict_fn, num_features, **kwargs); explanation.as_pyplot_figure= CustomExplanation.as_pyplot_figure # replacing default method return explanation

This is done as the LimeTabularExplainer calls the external method instead of inheriting the class containing the method. An alternative method to this would be to suggest a code commit in the source codebase that would allow users to directly pass the figsize and dpi of the LIME chart as a parameter to the method instead of manually overriding it to preserve reproducibility of code.

Final report generation

After LIME Explainer has diligently worked its magic by approximating a black-box model's behavior at a local level, it proceeds to deliver its findings in the form of the LIME chart images and the range explanations stored as JSON files. To provide the end user with the images and ranges in a human-readable format, all the results are organized into a pdf file and displayed to the users in the front, using FPDF2 in Python, giving the option to view it or download the pdf for future reading.

FPDF2, a Python library for PDF generation, was chosen for fine-grain control over dynamic PDF content. An ambiguous template was created for runtime generation, displaying user-chosen models and test instance search instructions on the first page. Each chart, representing data from a CSV file, occupied three-quarters of a page with explanations following. Predictions were placed at the top, beneath the title, specifying the Model Name and row number. The template maintained design integrity while accommodating multiple explanations. PDF generation time could be optimized with distributed computing or a higher-spec CPU. The final LIME

report, carefully crafted and assembled using FPDF2, bridges black-box models and the human domain. It renders the machine's decision-making process understandable and justifiable, making it an indispensable tool for bridging the gap between data science and decision-makers.

System frontend development

A GUI is provided using a web interface developed with Streamlit. Streamlit is an open-source Python library used to make web pages, making integration with the Machine Learning backend smooth.

1. Interactivity and Functionality

In this project, a crucial aspect was to create a user interface that offers a seamless user experience. The frontend provides the user with the following functionalities over multiple web pages:

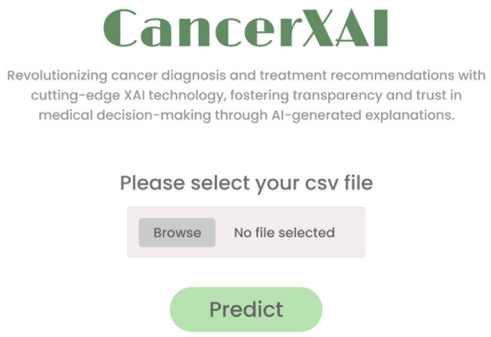
(i) Download a CSV file template to input multiple new data rows and upload this updated CSV to the backend. This feature would enable users to download a CSV file template to input new data values and then upload the updated CSV to the backend.
(ii) View the uploaded dataset on the front end so users can verify that they have uploaded the correct dataset with the correct values.
(iii) Customize the selection of models so the user can choose which Machine Learning model(s) is to be used to generate drug value prediction results to increase the system's flexibility.
(iv) View and download a PDF file containing predicted results and their explanation in LIME graphs and feature ranges for each inputted row of new data.

User interface designing and planning

Before embarking on the frontend implementation, designing and planning is essential in the user interface (UI). This includes the creation of wireframes, using Figma, that served as visual blueprints for the project. These visual blueprints outlined webpage structure and element placement and facilitated early-stage design discussions. As a guide during development, the wireframes ensured the final UI was intuitive, user-friendly, and aligned with project goals.

The wireframes provided below provide a clear vision of how the UI designing and planning process took place:

For the first webpage, a feature enabling users to upload a CSV file, which includes the row of data values to be used for the prediction, was designed for the backend through this webpage. The user would click the "predict" button to get the results. This feature allows for efficient data input and management. The webpage is optimized for scalability to handle a growing number of input rows simultaneously while maintaining high performance and response times.



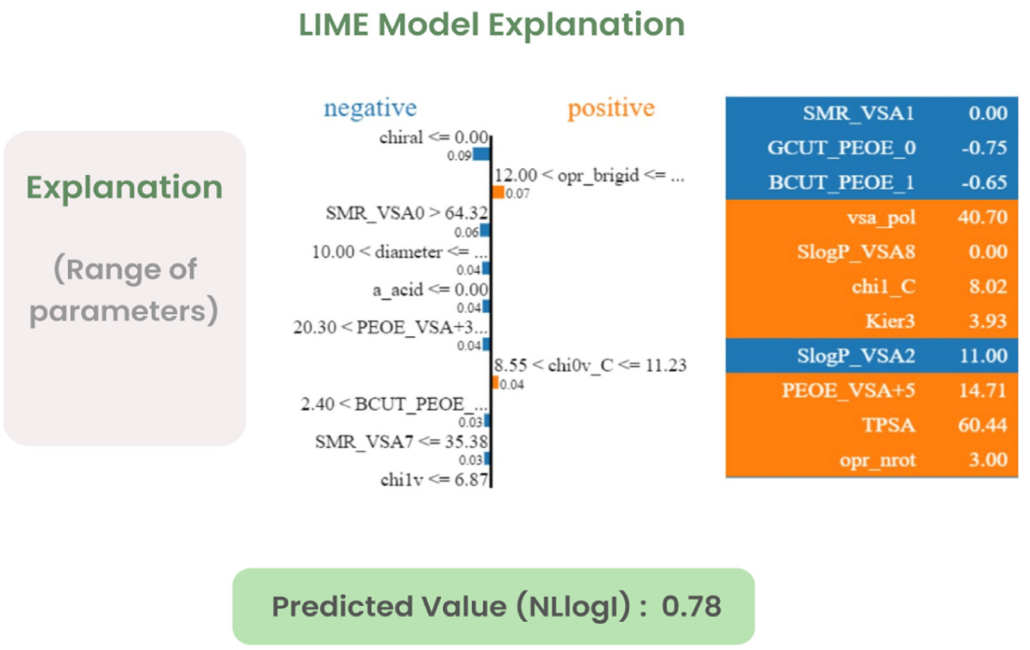
Screenshot 1: Wireframe of Home Page

Webpage 2 would provide the functionality to view the predicted results and LIME graphs, as shown below. This adds value by providing clear explanations for each inputted data point. Achieving this high level of interactivity and functionality required meticulous planning and coding.

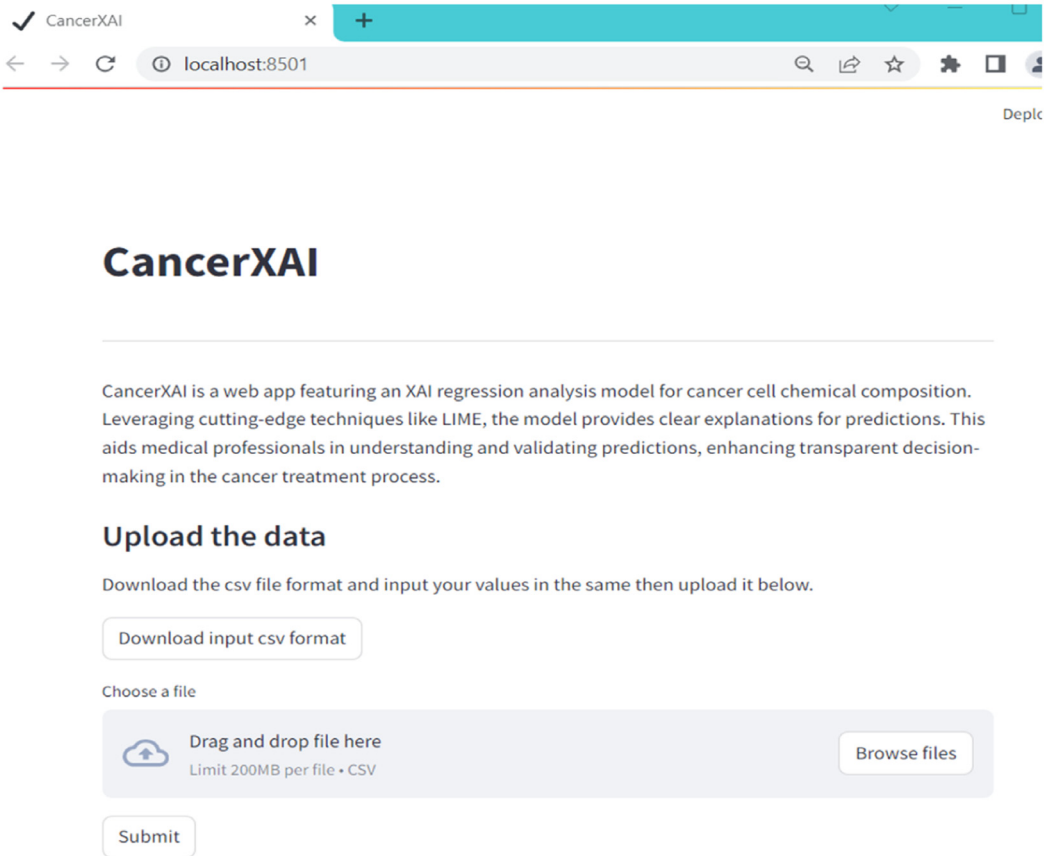
Frontend implementation

The crucial frontend phase in the project involved translating wireframes into functional webpages using Streamlit, a Python library known for quick interactive web app development. Each web page's functionalities were coded to enable user-friendly system interaction. The frontend seamlessly integrates with the backend for smooth data flow. Additional features like downloading CSV templates, viewing uploaded files, model selection, and a PDF LIME graph viewer were added based on user and model requirements.

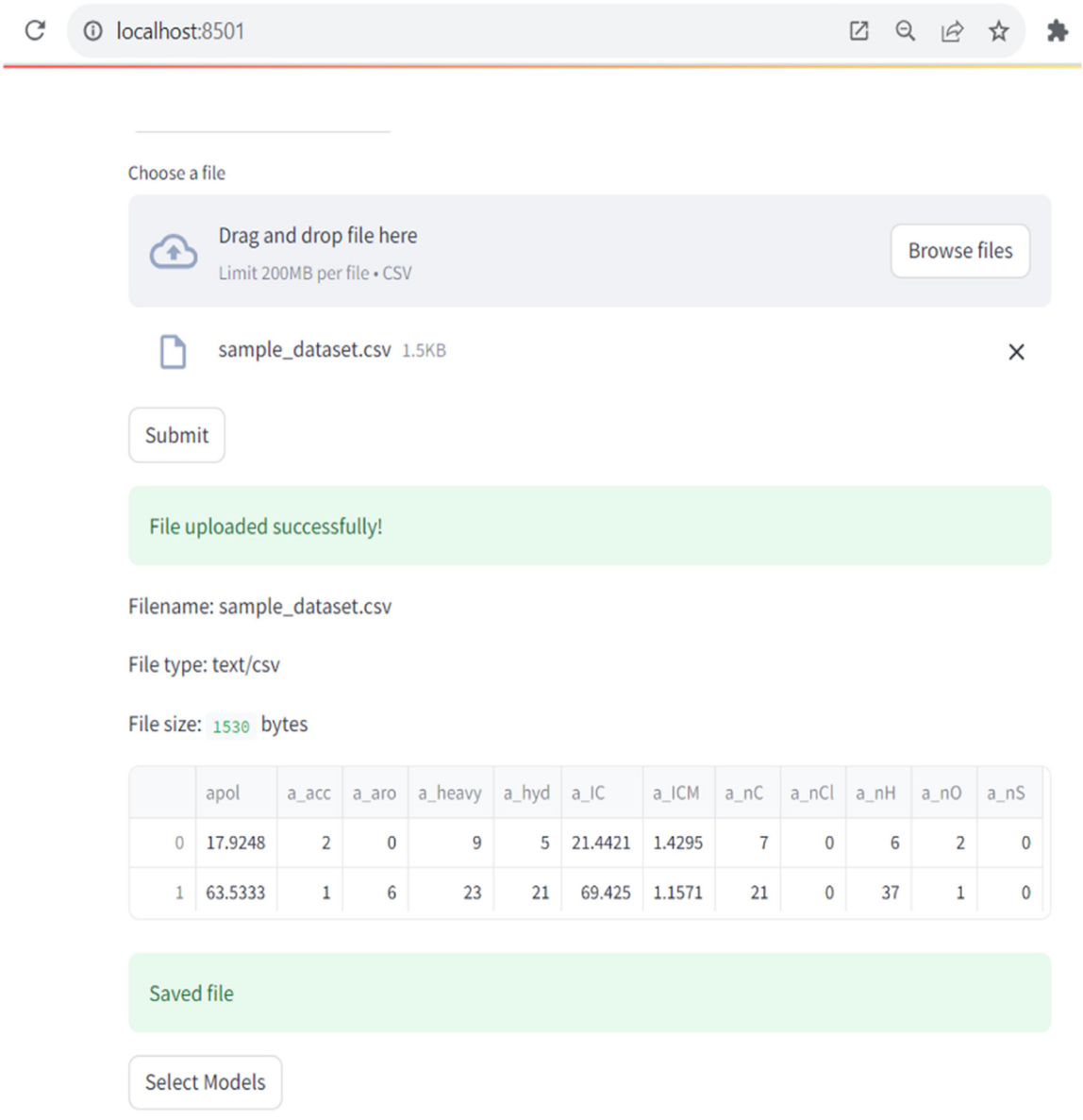
In "Webpage 1 - Home Page", a system for downloading a CSV file template, viewing it, and uploading data to the backend was developed, featuring user-friendly interfaces for these tasks. Upon data upload to the backend, a robust mechanism was employed to store the CSV file within the frontend/tempDir directory, guaranteeing backend accessibility for further processing and analysis.



Screenshot 2: Wireframe of Output Page



Screenshot 3: Home Page of Website



	apol	a_acc	a_aro	a_heavy	a_hyd	a_IC	a_ICM	a_nC	a_nCl	a_nH	a_nO	a_nS
0	17.9248	2	0	9	5	21.4421	1.4295	7	0	6	2	0
1	63.5333	1	6	23	21	69.425	1.1571	21	0	37	1	0

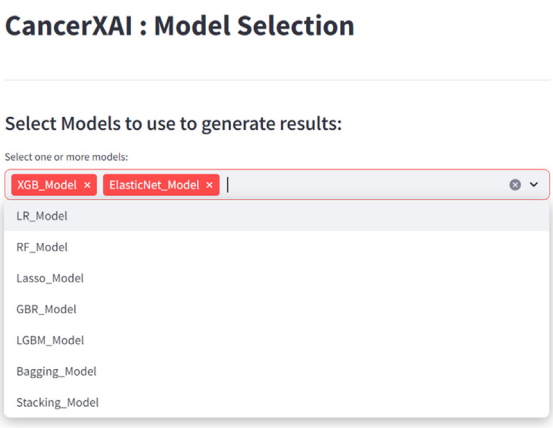
Screenshot 4: Home Page of the Website after uploading the CSV file

"Webpage 2 - Model Selection" included a multi-select model selection component, empowering users to choose and customize the Machine Learning (ML) models employed in the prediction process. This dynamic model selection mechanism is a crucial element of our user interface, enabling users to fine-tune their predictions according to their specific needs and preferences.

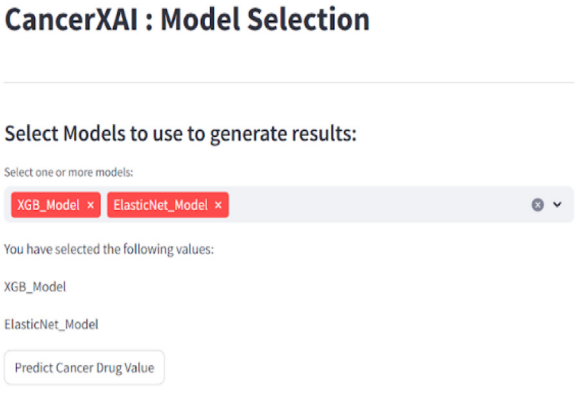
Lastly, "webpage 3 - Output Page" was designed to enable users to download a PDF file containing predicted results. This PDF document includes the expected results and LIME (Local Interpretable Model-agnostic Explanations) graphs that shed light on the decision-making process of our ML models. This feature provides valuable transparency to users, allowing them to understand the factors and features that influenced the predictions, which is pivotal for building trust in the system.

The pdf report is dynamically generated to accommodate the varying number of elements and pages in the pdf. This was done by setting the y-coordinate of each class of elements and then estimating the x-coordinate according to the relative positioning of the current component to the previous element. A page break is added to separate each result. A naming convention shows the row number and the model used, specified at the beginning of the pdf.

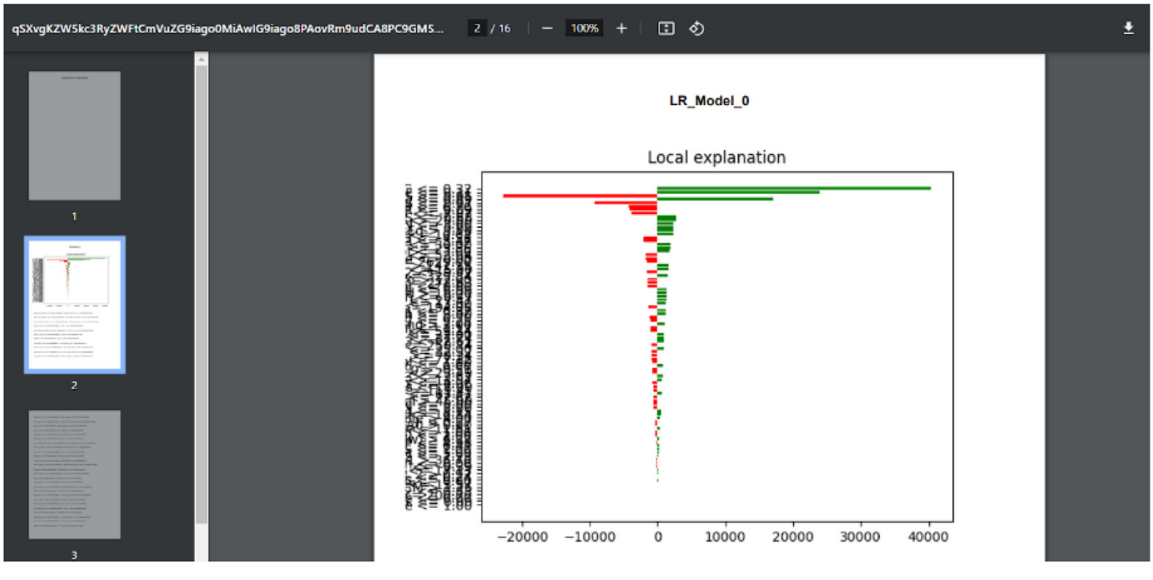
The clarity of the graphs generated in the pdf has a low pixel count yet higher dpi, allowing for smaller image generation of graphs while maintaining high clarity when zoomed in, as seen in 'Screenshot 8'. This solves the problem of fitting 90+ features in the y-axis of the graph. The user can view the pdf or download it based on the requirement. The PDF generated also contains the



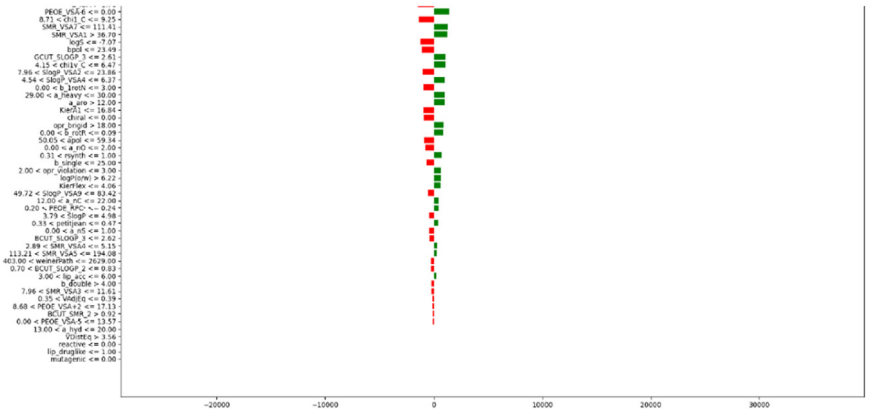
Screenshot 5: Model Selection Page of the Website



Screenshot 6: Model Selection Page of the Website after Model Selection



Screenshot 7: Output Page of the Website - PDF viewer



('PEOE_VSA_FNEG <= 0.32', 36542.37390191805) ('PEOE_VSA_FPNEG <= 0.05', -25725.374416059654)

('PEOE_VSA_FPOL <= 0.11', 22253.40559161703) ('0.88 < PEOE_VSA_FHYD <= 0.89', 20103.82756534518)

('0.00 < PEOE_VSA_FPPOS <= 0.07', -6779.0121914259) ('4.03 < Kier2 <= 9.47', 5398.8584854500705)

('chi1v <= 10.62', -4140.387421971915) ('302.90 < PEOE_VSA_HYD <= 329.84', 4109.379106597462)

Screenshot 8: A closer look at the LIME graphs and value ranges in the PDF Report

range of values in which the values of the features of the test case lie, which explains why the base statistical model generated the value of the final output feature.

Throughout the implementation phase, the development team conducted exhaustive testing and debugging to ensure our user interface was robust, error-free, and capable of withstanding various usage scenarios. The implementation phase also involved adhering to best practices in terms of code structure.

Results and discussions

This project methodically completed a comprehensive set of objectives, greatly expanding the healthcare field using Machine Learning and Explainable Artificial Intelligence (XAI). The effort began with collecting a complicated cancer medication dataset with 186,375 items and 202 characteristics. Outlier identification and handling, normalization, and categorical variable encoding were the data preparation operations that prepared the dataset for further modeling. The most relevant aspects were exposed through feature selection and engineering, while a meticulous algorithm selection procedure ensured optimal machine learning model selections, prioritizing forecast accuracy and generalization. Following the successful training and validation of these models, reliable predictions of cancer medication values and treatment efficacy were obtained. The integration of the XAI model and LIME enhanced forecasting transparency. A user-friendly web interface using Streamlit allows healthcare practitioners to input data and receive real-time predictions for patient cases. The project prioritized scalability and performance for handling many input rows. Comprehensive documentation ensures sustainability and research findings were published in peer-reviewed journals, contributing significantly to the healthcare industry. The project's outcomes include a vital tool that not only aids in cancer detection and treatment but also shows the potential of ML and XAI to alter healthcare practices.

Conclusion

The CancerXAI initiative represents a groundbreaking leap in cancer treatment. Initial tests show the model's impressive predictive performance with low RMSE and high R2 scores. What makes it unique is the incorporation of XAI techniques, providing valuable explanations for its decisions.

The project boasts an intuitive web interface with exceptional performance, offering promising contributions to cancer care. Its transparency, facilitated by XAI methods, provides clear predictive explanations, instilling confidence in patients and healthcare professionals. However, challenges include addressing bias due to duplicated entries in the NCL-provided dataset and demanding rigorous efforts for equitable and accurate performance across diverse cancer cells and chemical properties. A versatile framework for model training and prediction, irrespective of model or data, is crucial for widespread use. The framework's application in predicting cancer chemical composition is a significant milestone for XAI, fostering trust among medical researchers and doctors collaborating with AI prediction systems.

In conclusion, the CancerXAI project stands as a potential game-changer in cancer diagnosis and treatment by augmenting the accuracy of cancer diagnoses and delivering clear rationales for its predictions. As the project proceeds through the validation process,

it carries the potential for a profound impact on the landscape of cancer therapy. This, in turn, can lead to improved patient outcomes and further advancements in the healthcare sector.

Future scope

To enhance cancer treatment predictions, machine learning models need ongoing improvement. This involves using diverse datasets, refining feature selection, and optimizing algorithm performance. Continuous model enhancement empowers healthcare professionals to make better decisions, leading to personalized therapy suggestions based on individual genetic and clinical profiles. This customized approach holds promise for more accurate and effective cancer treatments, minimizing side effects and improving the likelihood of successful outcomes.

Integrating multimodal data is one of the most critical components of thorough decision support. Healthcare providers can develop a comprehensive understanding of each patient's condition by merging genomes, radiological imaging, electronic health data, and patient histories. This data integration improves diagnostic precision and provides information on treatment plans, resulting in better-informed choices. Enhancements to explainability are essential for winning the respect of medical practitioners. Thanks to advancements in XAI approaches, the system can now explain model predictions in a way that is simpler and easier to comprehend.

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.

CRediT authorship contribution statement

Sonali Kothari: Project administration, Funding acquisition, Supervision, Writing – review & editing, Conceptualization. **Shivanandana Sharma:** Data curation, Methodology, Software. **Sanskruti Shejwal:** Writing – original draft, Software, Visualization. **Aqsa Kazi:** Software, Methodology, Visualization. **Michela D'Silva:** Writing – original draft, Software. **M. Karthikeyan:** Writing – review & editing, Supervision, Validation.

Data availability

The data that has been used is confidential.

Ethics statements

Not applicable.

Acknowledgments

Not applicable.
This research received no specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] J. Amann, A. Blasimme, E. Vayena, D. Frey, V.I. Madai, Explainability for artificial intelligence in healthcare: a multidisciplinary perspective, BMC Med. Inform. Decis. Mak. 20 (1) (2020) Springer Science and Business Media LLC, doi:10.1186/s12911-020-01332-6.
[2] U. Pawar, D. O'Shea, S. Rea, R. O'Reilly, Explainable AI in healthcare, in: Proceedings of the 2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), Dublin, Ireland, 2020, pp. 1–2, doi:10.1109/CyberSA49311.2020.9139655.
[3] T. Hulsen, "Explainable Artificial Intelligence (XAI): concepts and Challenges in Healthcare," AI, vol. 4, no. 3. MDPI AG, pp. 652–666, Aug. 10, 2023.
[4] Y.C. Wang, T.C.T. Chen, M.C. Chiu, An improved explainable artificial intelligence tool in healthcare for hospital recommendation, Healthc. Anal. 3 (2023) 100147, doi:10.1016/j.health.2023.100147.
[5] Z. Sadeghi et al., "A brief review of explainable artificial intelligence in healthcare." arXiv, 2023.
[6] P.N. Srinivasu, N. Sandhya, R.H. Jhaveri, R. Raut, From blackbox to explainable AI in healthcare: existing tools and case studies, Mob. Inf. Syst. 2022 (2022) 1–20 Jun. 13, doi:10.1155/2022/8167821.
[7] S.S. Band, et al., Application of explainable artificial intelligence in medical health: a systematic review of interpretability methods, Inform. Med. Unlocked. 40 (2023) 101286 Elsevier BV, doi:10.1016/j.imu.2023.101286.
[8] R. Massafra, et al., Analyzing breast cancer invasive disease event classification through explainable artificial intelligence, Front. Med. 10 (2023) (Lausanne)Frontiers Media SA, Feb. 02, doi:10.3389/fmed.2023.1116354.
[9] S. Abe, et al., Explainable AI for estimating pathogenicity of genetic variants using large-scale knowledge graphs, Cancers 15 (4) (2023) 1118 (Basel)MDPI AG, doi:10.3390/cancers15041118.
[10] D. Shao, et al., Artificial intelligence in clinical research of cancers, Brief Bioinform. 23 (1) (2021) Oxford University Press (OUP)Dec. 21, doi:10.1093/bib/bbab523.
[11] Z.U. Ahmed, K. Sun, M. Shelly, L. Mu, Explainable artificial intelligence (XAI) for exploring spatial variability of lung and bronchus cancer (LBC) mortality rates in the contiguous USA, Sci. Rep. 11 (1) (2021) Springer Science and Business Media LLC, Dec. 16, doi:10.1038/s41598-021-03198-8.
[12] B. Aldughayfiq, F. Ashfaq, N.Z. Jhanjhi, M. Humayun, Explainable AI for retinoblastoma diagnosis: interpreting deep learning models with LIME and SHAP, Diagnostics 13 (11) (2023) 1932 MDPI AG, doi:10.3390/diagnostics13111932.
[13] Z. Zhou, G. Hooker, F. Wang, S-LIME, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, ACM, 2021 Aug. 14, doi:10.1145/3447548.3467274.

[14] C. Ladbury, et al., Utilization of model-agnostic explainable artificial intelligence frameworks in oncology: a narrative review, Transl. Cancer Res. 11 (10) (2022) 3853–3868 AME Publishing Company, doi:10.21037/tcr-22-1626.
[15] F. Stieler, F. Rabe, B. Bauer, Towards domain-specific explainable AI: model interpretation of a skin image classifier using a human approach, in: Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, 2021, doi:10.1109/cvprw53098.2021.00199.
[16] Streamlit official documentation, 2024 https://docs.streamlit.io.