**Cancer Survival Prediction Application**

**Prepared by: McDonald Chibueze Madufor  
Project Title: Cancer Survival Prediction Using Machine Learning and Streamlit Deployment**

**1. Introduction**

**Cancer remains one of the most prevalent and deadly diseases globally, accounting for millions of deaths each year. As the complexity of medical data increases, healthcare professionals are turning to data driven approaches to assist with clinical decision making. One of the most promising avenues is machine learning, which allows for pattern recognition and prediction using large and complex datasets.**

**The early prediction of cancer patient survival is critical. It enables oncologists and medical teams to make better-informed decisions regarding treatment options, palliative care, and patient counseling. However, most machine learning models require technical expertise to operate, making them inaccessible to clinicians who are not data scientists.**

**Problem Statement**

**While various machine learning models exist for predicting survival rates, many of these tools are not deployed in a manner that allows for easy interaction by non-technical users. Furthermore, data preprocessing and model tuning often require extensive expertise and resources. Therefore, there is a need for a simplified, interactive system that can make these predictions accessible and interpretable to healthcare professionals.**

**Project Aim**

**This project aims to develop a predictive cancer survival model using machine learning and make it easily accessible via a Streamlit web application. The model uses patient-specific features to predict whether a cancer patient is likely to survive, enabling healthcare providers to receive real time predictions through an intuitive user interface.**

**Objectives**

* **Preprocess and analyze a structured dataset of cancer patient records.**
* **Train and compare various machine learning models to identify the most accurate and clinically useful option.**
* **Develop a Streamlit web application that allows users to input patient data and receive predictions.**
* **Deploy the model and application online using GitHub and Streamlit Cloud to make it publicly accessible.**

**2. Data Description and Preprocessing**

**The dataset used for this project contains structured clinical information on cancer patients. It includes both numerical and categorical features relevant to survival prediction. Some of the key variables present in the dataset include:**

* **Age: Age of the patient at the time of diagnosis. Originally provided as Date of Birth, it was later converted to age due to errors encountered during formatting.**
* **Tumor Size: A numerical feature representing the size of the tumor in millimeters.**
* **Gender: A categorical feature that distinguishes between male and female patients.**
* **Survival Prediction: The target variable representing whether the patient survived (1 = Yes, 0 = No).**

**Data Cleaning and Preparation**

**To ensure the model's accuracy, a series of preprocessing steps were performed:**

* **Handling Missing Values: Missing data was handled through imputation. Median values were used for numerical features, while mode was used for categorical variables. This strategy preserved as much data as possible while maintaining consistency.**
* **Outlier Treatment: Boxplots were used to identify extreme values. These outliers were not entirely removed but were adjusted (minorized) to minimize their skewing effect on the model.**
* **Categorical Encoding: The 'Gender' variable was encoded as a binary feature (e.g., 0 for male, 1 for female). Other categorical values would have been one-hot encoded if present.**
* **Feature Scaling: Numerical features were standardized using z-score normalization (mean = 0, standard deviation = 1). This step is crucial for algorithms sensitive to feature magnitude.**
* **Additional Cleaning: Minor but significant refinements were made to ensure consistent column formats and data types.**

**These preprocessing steps ensured that the final dataset was balanced, consistent, and ready for machine learning model training.**

**3. Exploratory Data Analysis (EDA)**

**Overview**

**EDA was performed to understand the distributions and relationships between features, particularly how they relate to the survival outcome. The goal was to identify patterns that could inform model selection and interpretation.**

**Visualizations and Insights**

**Although multiple plots were planned, technical issues in Jupyter Notebook prevented smooth rendering of all visualizations. However, a histogram of the numerical data particularly age was successfully generated. This visualization revealed that most patients in the dataset were elderly, primarily between the ages of 50 and 90.**

**Limitations**

**Due to memory issues and the notebook running slowly, only limited visualizations were rendered. Still, basic patterns were observable and informative enough to proceed with model development.**

**4. Methodology**

**A range of supervised classification algorithms were explored to build a reliable survival prediction model. The project experimented with both baseline models and advanced ensemble methods.**

**Models Considered**

* **Random Forest Classifier**
* **Decision Tree Classifier**
* **Ada Boost Classifier**
* **XGBoost Classifier**
* **Gradient Boosting Classifier (final selection)**

**In addition, the dataset exhibited class imbalance, which can negatively impact model performance. To address this, various oversampling and under sampling methods were used, such as:**

* **Random Over Sampler**
* **SMOTE (Synthetic Minority Oversampling Technique)**
* **ADASYN (Adaptive Synthetic Sampling)**
* **Random Under Sampler**

**Addressing Class Imbalance**

**Class imbalance was particularly critical because predicting the minority class (e.g., patients who did not survive) is more important than the majority class. False negatives where a non-survivor is misclassified as a survivor could have significant real-world consequences. Thus, sampling techniques were tested in conjunction with classifiers to determine the most effective combination.**

**5. Model Evaluation**

**The dataset was split into training and test sets using an 80/20 ratio. Cross-validation was not employed due to performance issues, and hyperparameter tuning methods like Grid Search CV and Randomized Search were avoided because they led to poor results and overfitting.**

**Metrics Used**

**Several metrics were used to evaluate model performance:**

* **Accuracy: Measures overall correctness but can be misleading on imbalanced datasets.**
* **Precision: Measures the proportion of correct positive predictions out of all predicted positives.**
* **Recall (Sensitivity): Measures the ability to correctly identify true positives, crucial in medical contexts.**
* **F1-Score: Balances precision and recall, especially helpful when dealing with uneven class distributions.**
* **Confusion Matrix: Visual tool used to understand true positives, true negatives, false positives, and false negatives.**

**Final Model Selection**

**After multiple iterations, the Gradient Boosting Classifier with ADASYN oversampling was selected. This model achieved:**

* **Recall: ~99%**
* **F1 Score: ~75%**
* **Accuracy: ~60%**
* **Precision: ~60%**

**The high recall was the deciding factor, as missing true cases (i.e., predicting a survivor when the patient actually did not survive) could be catastrophic in real-world scenarios. A sample .csv file was also created with patient IDs and survival prediction outputs for demonstration and testing.**

**6. Deployment**

**Streamlit App Development**

**The final model was serialized using Python’s pickle module and integrated into a user-friendly Streamlit web application (app.py). The app includes:**

* **Input widgets (sliders, text boxes, dropdowns) for user-friendly data entry**
* **A 'Predict' button to trigger the model inference**
* **An output area that clearly states whether the patient is likely to survive or not**

**Deployment Process**

**The app, model, and a requirements.txt file listing all dependencies were pushed to a public GitHub repository. Streamlit Cloud was then linked to this repository.**

**Despite several deployment errors including version mismatches and failed dependency loads these were successfully debugged, and the app was eventually deployed.**

**Upon pushing changes to GitHub, Streamlit Cloud automatically rebuilt the application and hosted it live at a public URL.**

**Public Access**

**This cloud-based deployment ensures that users can access the app from any device without needing to install anything locally. The URL can be shared with medical professionals for immediate use or future evaluation. This approach not only provides ease of access but also allows for easy iteration and updates.**

**7. Conclusion**

**This project demonstrates how machine learning and modern deployment tools can be leveraged to solve real-world problems in healthcare. By training a classification model to predict cancer patient survival and deploying it through an accessible web interface, the gap between data science and clinical practice was significantly reduced.**

**The Gradient Boosting Classifier, boosted with ADASYN sampling, proved effective in achieving high recall, a critical factor for clinical predictions. The Streamlit interface makes it easy for users to interact with the model, test outcomes, and apply the tool to hypothetical or real patient scenarios.**