**Project Title: Breast Cancer Prediction and Visualization Using Power BI and Machine Learning**

**Brief Description:**

This project focuses on predicting **breast cancer** using **machine learning algorithms** and visualizing the results dynamically using **Power BI**. The integration of machine learning models with interactive dashboards provides an insightful and user-friendly system for data analysis and prediction.

The project leverages a dataset containing breast cancer-related features to train ML models, enabling accurate classification of tumors as **malignant** or **benign**. The results are visualized in Power BI to offer real-time insights and enhance decision-making.

**Key Features:**

1. **Machine Learning Models**:
   * Implemented ML algorithms such as **Logistic Regression**, **Random Forest**, and **Support Vector Machines (SVM)** for accurate prediction.
   * Models were evaluated based on metrics like **accuracy**, **precision**, and **recall**.
2. **Power BI Integration**:
   * Visualized the predictions and performance metrics of ML models using interactive dashboards.
   * Displayed key metrics such as:
     + **Number of malignant vs. benign cases**
     + **Model accuracy and comparison**
     + **Patient feature distribution**
   * Enabled **real-time updates** of visualizations when the input data changes.
3. **Dataset**:
   * Utilized a structured **CSV dataset** (e.g., Breast Cancer Wisconsin Dataset) containing features such as radius, texture, compactness, and smoothness of cells.
4. **Insights**:
   * Easy-to-understand Power BI dashboards that allow medical professionals to visualize and interpret predictions.
   * Combined the power of **ML algorithms** for accuracy and **Power BI** for presentation.

**Tools/Technologies Used:**

* **Python**: For building and training machine learning models
* **Scikit-Learn**: For implementing ML algorithms
* **Power BI**: For creating interactive dashboards and visualizations
* **Pandas**: For data preprocessing
* **Matplotlib/Seaborn**: For initial visualization of data

**Outcome:**

The project successfully predicts breast cancer cases and visualizes the results using an interactive Power BI dashboard. It enables medical professionals and data analysts to make data-driven decisions efficiently by combining machine learning with powerful data visualization

**Methodology**🡪

**1.Created a csv file on a drive.**

**2.Create a code in google colab using a python as a backend language and by using Machine Learning algorithm.**

**Code:**

from google.colab import drive

drive.mount('/content/drive') import pandas as pd

file\_path = '/content/drive/My Drive/SEER\_Breast\_Cancer\_Dataset .csv'

data = pd.read\_csv(file\_path)

file\_path\_obj = Path(file\_path)

if not file\_path\_obj.is\_file():

print(f"Error: File not found at {file\_path}")

else:

# Read the CSV file if it exists

data = pd.read\_csv(file\_path)

print("File loaded successfully!")

X = data.drop(columns=['Survival Months'])

y = data['Survival Months']

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42) # Added train\_test\_split

# Create a LabelEncoder object

label\_encoder = LabelEncoder()

# Iterate over each column in the DataFrame

for column in X\_train.columns:

# Check if the column contains string values

if X\_train[column].dtype == 'object':

# Fit and transform the column using LabelEncoder

X\_train[column] = label\_encoder.fit\_transform(X\_train[column])

X\_test[column] = label\_encoder.transform(X\_test[column])

**3. By using Random forest classifier algorithm.**

**Code:**

from sklearn.metrics import accuracy\_score

model = RandomForestClassifier()

model.fit(X\_train, y\_train)

predictions = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, predictions)

abel\_encoder = LabelEncoder()

# Iterate over each column in the DataFrame 'X' (not just X\_train)

for column in X.columns: # Changed to X instead of X\_train

# Check if the column contains string values

if X[column].dtype == 'object':

# Fit and transform the column using LabelEncoder

X[column] = label\_encoder.fit\_transform(X[column])

# Generate an Extra Column

data['predictions'] = model.predict(X)

data.to\_csv('/content/drive/My Drive/SEER\_Breast\_Cancer\_Dataset .csv', index=False)

output\_df = pd.DataFrame(X\_test) # Assuming X\_test contains the features

output\_df['Predictions'] = predictions

1. **Update the file.**

output\_file\_path = '/content/drive/My Drive/SEER\_Breast\_Cancer\_Dataset1.csv'

output\_df.to\_csv(output\_file\_path, index=False)

1. **Use a Power BI for visualization.**
2. **Connecting power Bi and google colab**

Mount Google Drive in Google Colab

You need to store your generated data (e.g., a CSV file) in Google Drive.

Step 1: Make the CSV File Publicly Accessible

To allow Power BI to access the CSV file, you need to make the file publicly accessible or share it with a specific link.

1. Go to Google Drive.

2. Locate the file (colab\_data.csv), right-click it, and select Share.

3. Change the sharing settings to Anyone with the link can view.

4. Copy the link to the file.

Now, the file is accessible via a direct URL, which you’ll use in Power BI.

Step 2: Import CSV into Power BI

1. Open Power BI Desktop.

2. Click on Get Data → Web.

3. Paste the link you copied from Google Drive but modify it slightly:

* + Replace the https://drive.google.com/file/d/FILE\_ID/view?usp=sharing link with https://drive.google.com/uc?id=FILE\_ID&export=download.

Step 3: Enable Scheduled Refresh in Power BI

To ensure your Power BI dashboard reflects real-time changes from the Colab program, enable the refresh option:

1. Upload to Power BI Service:

* After creating the Power BI report, save and publish it to Power BI Service (cloud).
* Click on File → Publish → Publish to Power BI Service.
* Name of our file saved

2. Configure Scheduled Refresh:

* Go to Power BI Service (<https://app.powerbi.com>).
* Navigate to Datasets.
* Find your dataset and click on the Schedule Refresh option.
* Set up the refresh frequency (e.g., every hour or once a day).

Power BI will now automatically pull updated data from the Google Drive CSV file at the scheduled intervals.

Step 4: Sync Google Colab Data Updates with Power BI

Whenever your Google Colab program modifies the CSV file, those updates will be reflected in Power BI during the next refresh cycle.

**Images of project “Breast Cancer Prediction and Visualization Using Power BI and Machine Learning”**



