BREAST-CANCER-DIAGNOSIS

Submitted by team **DATACRUSHERS**

Team Members:

1. Aman Narang
2. Lakshay Garg
3. Gracy Dhamija
4. Daxh Khatreja

Contents

About Breast Cancer

Data analytics plays a pivotal role in modern healthcare, particularly in the domain of breast cancer diagnosis. By leveraging advanced analytical techniques on vast datasets of patient information, medical researchers and practitioners can gain valuable insights into the disease's characteristics, risk factors, and potential treatment strategies. Through data analytics, patterns and trends within patient data, such as mammography images, genetic profiles, tumor characteristics, and patient demographics, can be identified and analyzed to develop more accurate diagnostic methods and personalized treatment plans. Machine learning algorithms, for instance, can be trained on large datasets of breast cancer cases to predict the likelihood of malignancy based on various features, aiding in early detection and intervention. Furthermore, data analytics facilitates the integration of diverse data sources, including electronic health records, imaging data, and genomic data, enabling a comprehensive and holistic approach to breast cancer diagnosis and management. By harnessing the power of data analytics, healthcare professionals can enhance the accuracy of diagnosis, optimize treatment strategies, and ultimately improve patient outcomes in the fight against breast cancer.

INTRODUCTION

Data analytics plays a pivotal role in modern healthcare, particularly in the domain of breast cancer diagnosis. By leveraging advanced analytical techniques on vast datasets of patient information, medical researchers and practitioners can gain valuable insights into the disease's characteristics, risk factors, and potential treatment strategies. Through data analytics, patterns and trends within patient data, such as mammography images, genetic profiles, tumor characteristics, and patient demographics, can be identified and analyzed to develop more accurate diagnostic methods and personalized treatment plans. Machine learning algorithms, for instance, can be trained on large datasets of breast cancer cases to predict the likelihood of malignancy based on various features, aiding in early detection and intervention. Furthermore, data analytics facilitates the integration of diverse data sources, including electronic health records, imaging data, and genomic data, enabling a comprehensive and holistic approach to breast cancer diagnosis and management. By harnessing the power of data analytics, healthcare professionals can enhance the accuracy of diagnosis, optimize treatment strategies, and ultimately improve patient outcomes in the fight against breast cancer.

Python and its libraries used:

In our breast cancer diagnosis project, we utilized Python along with several key libraries to facilitate data analysis, model development, and visualization. Here's a brief overview of the Python libraries we employed:

NumPy: This library was essential for efficient handling of numerical data, performing array operations, conducting linear algebra operations, and executing various mathematical functions.

Matplotlib: Matplotlib allowed us to create diverse visualizations, including line plots, histograms, and scatter plots. We used it extensively to visualize data distributions, explore relationships between variables, and present our findings visually.

Seaborn: Leveraging Seaborn, we created more advanced statistical graphics such as heatmaps, pair plots, and categorical plots. This library, built on top of Matplotlib, provided a high-level interface for generating informative and visually appealing visualizations, aiding us in gaining deeper insights into our data.

Scikit-learn: Scikit-learn played a crucial role in building and evaluating machine learning models for breast cancer diagnosis. With its wide range of tools, we implemented classification algorithms like Logistic Regression, Decision Trees, Random Forests, and Support Vector Machines, allowing us to develop accurate predictive models.

Pandas: Pandas facilitated efficient data manipulation and analysis, offering data structures and functions for working with structured data. We relied on Pandas for tasks such as loading data from various sources, handling missing values, and transforming datasets into suitable formats for modeling.

Breast Cancer Wisconsin (Diagnostic) Data Set

This dataset contains features computed from images of fine needle aspirates of breast masses, describing characteristics of cell nuclei. It consists of 10 real-valued features, including radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension.

texture3	perimeter3	area3	smoothness3	compactness3	concavity3	concave_points3	symmetry3	fractal_dimension3	Diagnosis
17.33	184.60	2019.0	0.16220	0.66560	0.7119	0.2654	0.4601	0.11890	1
23.41	158.80	1956.0	0.12380	0.18660	0.2416	0.1860	0.2750	0.08902	1
25.53	152.50	1709.0	0.14440	0.42450	0.4504	0.2430	0.3613	0.08758	1
26.50	98.87	567.7	0.20980	0.86630	0.6869	0.2575	0.6638	0.17300	1
16.67	152.20	1575.0	0.13740	0.20500	0.4000	0.1625	0.2364	0.07678	1
...	...	...	...	...	...	...	...	...	...
26.40	166.10	2027.0	0.14100	0.21130	0.4107	0.2216	0.2060	0.07115	1
38.25	155.00	1731.0	0.11660	0.19220	0.3215	0.1628	0.2572	0.06637	1
34.12	126.70	1124.0	0.11390	0.30940	0.3403	0.1418	0.2218	0.07820	1
39.42	184.60	1821.0	0.16500	0.86810	0.9387	0.2650	0.4087	0.12400	1
30.37	59.16	268.6	0.08996	0.06444	0.0000	0.0000	0.2871	0.07039	0

Details regarding the Dataset

Radius characteristics (radius_mean, radius_se, and radius_worst): These characteristics show different tumor-related measures. They offer details regarding the dimensions and morphology of the tumor cells.

Texture characteristics: These characteristics (texture_mean, texture_se, and texture_worst) characterise the image's fluctuation in grayscale intensities. They offer details regarding the tumor cells' roughness or smoothness.

Perimeter characteristics (perimeter_mean, perimeter_se, and perimeter_worst): These characteristics show measurements pertaining to the tumor's perimeter or its length. They offer more details regarding the tumor's dimensions and morphology.

Area Features (area_mean, area_se, and area_worst): These features show measurements of the overall area (or pixels) that the tumor has taken up. They offer details regarding the tumor's overall size.

Smoothness Features (smoothness_mean, smoothness_se, smoothness_worst): These features provide information about the uniformity of cell size and shape by describing the variation in the local smoothness of the tumor boundary.

Compactness Features (compactness_mean, compactness_se, compactness_worst): These features describe how closely the tumor cells are packed together, which is known as the tumor's compactness.

Concavity Features (concavity_mean, concavity_se, concavity_worst): These features describe the severity of concave portions of the tumor boundary and provide information about the irregularity of the tumor shape.

Concave Points characteristics: The number of concave sections of the tumor border is represented by these characteristics (concave_points_mean, concave_points_se, and concave_points_worst). They offer more details regarding the uneven morphology of the tumor.

Symmetry Features: (symmetry_mean, symmetry_se, symmetry_worst) These features define the tumor cells' symmetry. They offer details regarding the consistency of cell size and shape throughout the tumor's various sections.

Fractal Dimension Features: The complexity of the tumor boundary is described by these features (fractal_dimension_mean, fractal_dimension_se, and fractal_dimension_worst). They offer details regarding the tumor's uneven and rough shape.

Machine learning process:

Data Gathering:

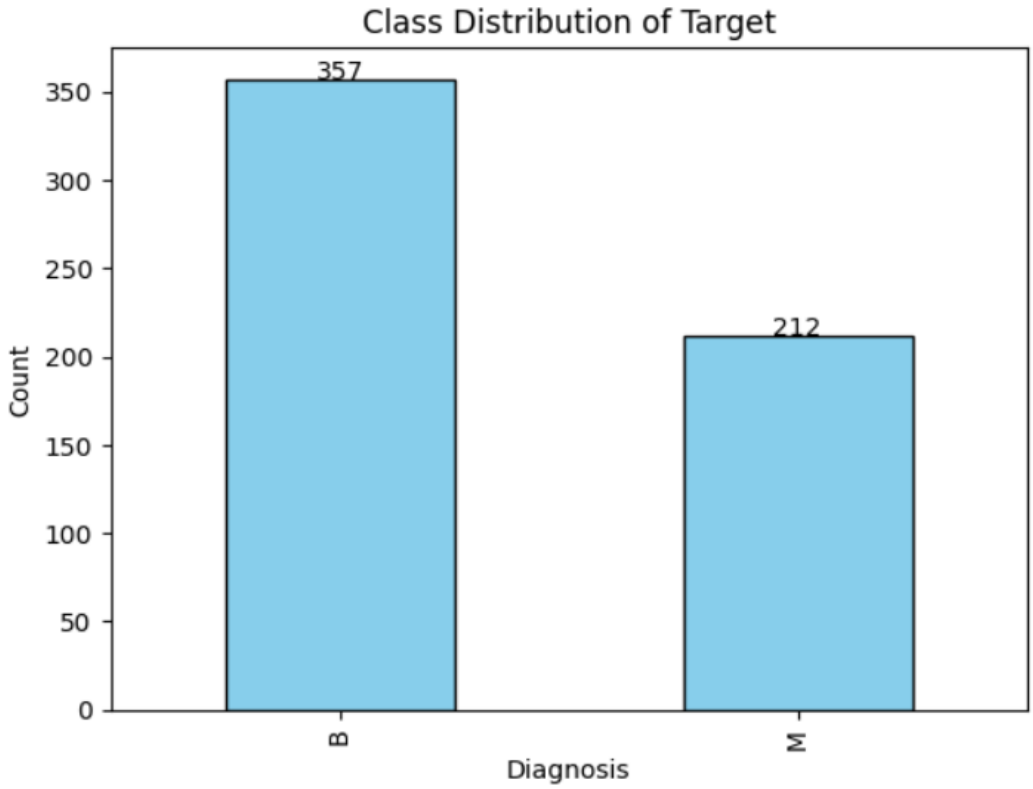
- The Breast Cancer Wisconsin Diagnostic dataset was obtained from the UCI Machine Learning Repository. This dataset consists of features extracted from digitized images of breast mass obtained from fine needle aspirates.
- Each instance in the dataset represents a sample of breast cells, and the features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass.
- The dataset contains a total of 569 instances and 30 features, including the target variable.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   radius1               569 non-null    float64
 1   texture1              569 non-null    float64
 2   perimeter1            569 non-null    float64
 3   area1                 569 non-null    float64
 4   smoothness1           569 non-null    float64
 5   compactness1          569 non-null    float64
 6   concavity1            569 non-null    float64
 7   concave_points1       569 non-null    float64
 8   symmetry1             569 non-null    float64
 9   fractal_dimension1    569 non-null    float64
 10  radius2               569 non-null    float64
 11  texture2              569 non-null    float64
 12  perimeter2            569 non-null    float64
 13  area2                 569 non-null    float64
 14  smoothness2           569 non-null    float64
 15  compactness2          569 non-null    float64
 16  concavity2            569 non-null    float64
 17  concave_points2       569 non-null    float64
 18  symmetry2             569 non-null    float64
 19  fractal_dimension2    569 non-null    float64
 20  radius3               569 non-null    float64
 21  texture3              569 non-null    float64
 22  perimeter3            569 non-null    float64
 23  area3                 569 non-null    float64
 24  smoothness3           569 non-null    float64
 25  compactness3          569 non-null    float64
 26  concavity3            569 non-null    float64
 27  concave_points3       569 non-null    float64
 28  symmetry3             569 non-null    float64
 29  fractal_dimension3    569 non-null    float64
 30  Diagnosis             569 non-null    object 
dtypes: float64(30), object(1)
memory usage: 137.9+ KB
((569, 31), None)
```

Data Exploration:

Visualizing Class Distribution:

- We began by exploring the distribution of the target variable, which indicates whether a tumor is benign or malignant.
- This was visualized using a bar plot, with the two classes (benign and malignant) on the x-axis and the count of instances on the y-axis.
- This step provided insights into the balance or imbalance of classes in the dataset, which is crucial for model training and evaluation.



Analyzing Feature Correlations:

- We computed the correlation matrix between the features to understand the relationships among them.
- This was visualized using a heatmap, where each cell represents the correlation coefficient between two features.
- High correlations between features can indicate redundant information, which may affect the performance of certain models.

Descriptive Statistics:

- We computed descriptive statistics such as mean, standard deviation, minimum, maximum, and quartiles for each feature.
- This provided insights into the range and distribution of values for each feature, helping to identify outliers or anomalies in the data.

Feature Visualization:

- We visualized the distribution of individual features using histograms or box plots.
- This allowed us to identify any skewness or outliers in the feature distributions, which may require further preprocessing.

Data Preprocessing:

Encoding the Target Variable ('Diagnosis') using LabelEncoder:

- In the Breast Cancer Wisconsin Diagnostic dataset, the target variable 'Diagnosis' indicates whether a tumor is benign or malignant.
- Machine learning algorithms typically require numerical input for model training and prediction.
- Therefore, we encoded the target variable into numerical values using LabelEncoder from the scikit-learn library.
- LabelEncoder assigns a unique numerical label to each class in the target variable, transforming categorical data into a format suitable for model training.

Splitting the Data into Training and Testing Sets:

- To evaluate the performance of our machine learning models effectively, we split the dataset into training and testing sets.
- The training set is used to train the models, while the testing set is used to assess their performance on unseen data.
- We used the train_test_split function from sci-kit-learn to randomly split the dataset into training and testing sets, typically with a specified ratio (e.g., 80% training, 20% testing).

Model Building:

In this phase of the project, we trained several machine learning models for breast cancer diagnosis using the Breast Cancer Wisconsin Diagnostic dataset. Each model was selected based on its suitability for binary classification tasks and its potential to achieve high accuracy in diagnosing breast cancer. The following models were trained and evaluated:

1. Logistic Regression:

1.0.3 LogisticRegression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix

clf1 = LogisticRegression()
clf1.fit(X_train, y_train)
```

- Logistic Regression is a simple yet effective linear model for binary classification tasks.
- It models the probability that an instance belongs to a particular class using a logistic function.
- Despite its simplicity, logistic regression can perform well when the relationship between features and the target variable is linear or can be approximated by a linear function.

2. Support Vector Classifier (SVC):

1.0.4 SVC

```
from sklearn.svm import SVC
clf2 = SVC(kernel='linear')  # You can choose different kernels such as␣
 ↪'linear', 'rbf', 'poly', etc.

# Training the model
clf2.fit(X_train, y_train)
```

- Support Vector Classifier, or SVC, is a powerful model for binary classification that works by finding the hyperplane that best separates the classes.
- It can handle non-linear decision boundaries through the use of kernel functions.
- SVC aims to maximize the margin between the classes while minimizing classification errors.

3. K-Nearest Neighbors (KNN):

1.0.5 KNeighborsClassifier

```
from sklearn.neighbors import KNeighborsClassifier

clf3 = KNeighborsClassifier(n_neighbors=5)
clf3.fit(X_train, y_train)
```

- K-Nearest Neighbors is a non-parametric algorithm that classifies instances based on the majority class of their k nearest neighbors in feature space.
- It is simple to understand and implement, making it a popular choice for classification tasks.
- KNN's performance can be influenced by the choice of distance metric and the value of k.

4. Random Forest:

1.0.6 RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier

clf4 = RandomForestClassifier(n_estimators=100, random_state=42)
clf4.fit(X_train, y_train)
```

- Random Forest is an ensemble learning method that builds multiple decision trees during training and combines their predictions through averaging or voting.
- It is robust to overfitting and can handle high-dimensional data with complex decision boundaries.

- Random Forest can provide insights into feature importance through the analysis of tree structures.

5. XGBoost:

1.0.7 XGBoost

```
import xgboost as xgb

clf5 = xgb.XGBClassifier(objective='binary:logistic', eval_metric='logloss',␣
 ↪random_state=42)
clf5.fit(X_train, y_train)
```

- XGBoost (Extreme Gradient Boosting) is a gradient boosting algorithm known for its speed and performance.
- It builds a sequence of decision trees, where each tree corrects the errors made by the previous ones.
- XGBoost employs regularization techniques to prevent overfitting and can handle missing values and categorical features.

6. CatBoost:

1.0.8 CatBoostClassifier

```
from catboost import CatBoostClassifier

clf6 = CatBoostClassifier(iterations=500, depth=6, learning_rate=0.1,␣
 ↪loss_function='Logloss', random_state=42)
clf6.fit(X_train, y_train, cat_features=None)
```

- CatBoost is another gradient boosting algorithm designed for categorical data.
- It handles categorical features efficiently without requiring one-hot encoding or preprocessing.
- CatBoost incorporates robust handling of categorical variables, making it suitable for datasets with mixed data types.

7. LightGBM:

1.1 LGBMClassifier

```
: import lightgbm as lgb

clf7 = lgb.LGBMClassifier()
clf7.fit(X_train, y_train)
```

- LightGBM is a gradient boosting framework that focuses on speed and efficiency.
- It uses a histogram-based algorithm to partition data and grow decision trees, resulting in faster training times.
- LightGBM is suitable for large-scale datasets and can handle categorical features directly.

8. AdaBoost:

1.2 AdaBoostClassifier

```
from sklearn.ensemble import AdaBoostClassifier

clf8 = AdaBoostClassifier()
clf8.fit(X_train, y_train)
```

- AdaBoost (Adaptive Boosting) is an ensemble learning method that combines multiple weak learners to create a strong classifier.
- It assigns higher weights to misclassified instances during training, focusing on the most challenging instances.
- AdaBoost iteratively improves the model's performance by adjusting the weights of incorrectly classified instances.
- Each model was trained on the preprocessed dataset and evaluated using various performance metrics such as accuracy, precision, recall, and F1-score. The performance

of each model was compared to identify the best-performing algorithm for breast cancer diagnosis.

Model Evaluation:

In this phase, we assessed the performance of each trained machine-learning model for breast cancer diagnosis using the following metrics:

Accuracy:

It indicates the proportion of correctly classified instances out of the total dataset, giving a broad measure of the model's overall predictive capability.

Confusion Matrix:

This matrix displays counts of true positive, true negative, false positive, and false negative predictions.

It offers insights into the model's ability to classify instances accurately and identify any misclassifications across different classes.

Classification Report:

This report includes precision, recall, and F1-score metrics for each class in the dataset.

Precision measures the accuracy of positive predictions, recall assesses the model's ability to capture all positive instances, and F1 score provides a balanced measure of precision and recall.

```
Accuracy: 0.9824561403508771
Confusion Matrix:
[[70  1]
 [ 1 42]]
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99        71
           1       0.98      0.98      0.98        43

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114
```

Based on the evaluation results, the accuracy achieved by each model is as follows:

Logistic Regression: 96.5%

Support Vector Classifier (SVC): 96.5%

K-Nearest Neighbors (KNN): 95.6%

Random Forest: 96.5%

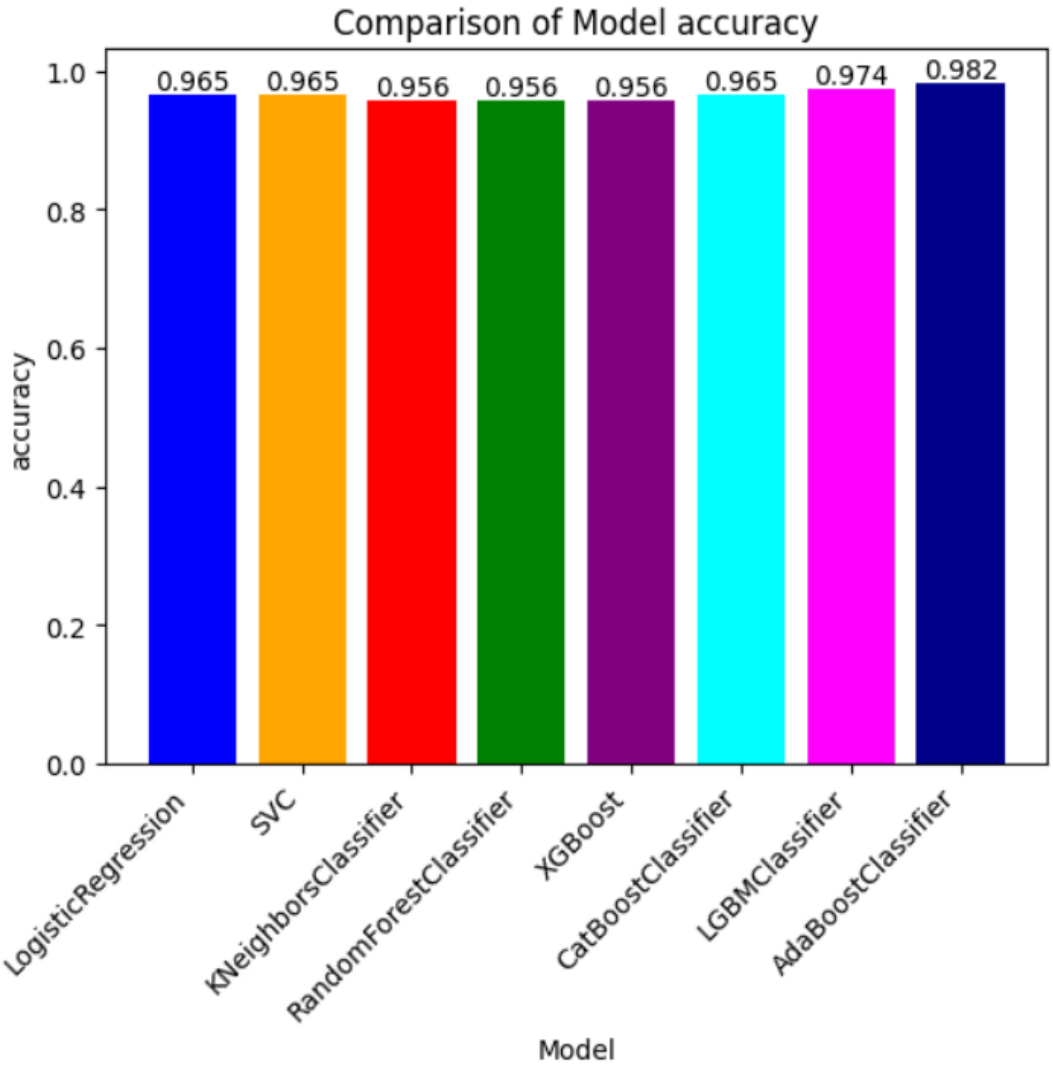
XGBoost: 96.5%

CatBoost: 96.5%

LightGBM: 97.4%

AdaBoost: 98.2%

Among the models tested, the AdaBoostClassifier demonstrated the highest accuracy of 98.2%, making it the best-performing model for breast cancer diagnosis in this study


Comparison of Model accuracy
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
0.965
0.965
0.956
0.956
0.956
0.965
0.974
0.982
LogisticRegression
SVC
KNeighborsClassifier
RandomForestClassifier
XGBoost
CatBoostClassifier
LGBMClassifier
AdaBoostClassifier
Model