



Project Digital Skill Fair 36.0

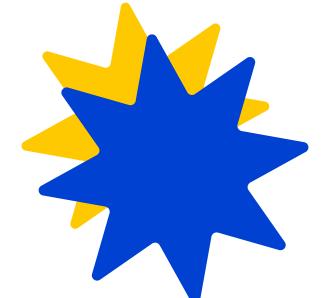
(Machine Learning Classification)

Luthfi Wulandari



Hello , I am Luthfi Wulandari

I am a second-year student studying Information Systems with a keen interest in data management and analysis. Currently, I am improving my skills in data processing, data analysis, and data visualization. I want to improve my skills in cleaning and preparing data, identifying patterns, creating visual representations and predicting trends using tools such as Excel, Tableau, SQL, and Python.



Data Science



Data science is the practice of using analytical methods and tools to uncover insights and knowledge from data. It integrates multiple disciplines including statistics, coding, machine learning, and data visualization. The primary aim of data science is to transform raw data into valuable information for informed decision-making.



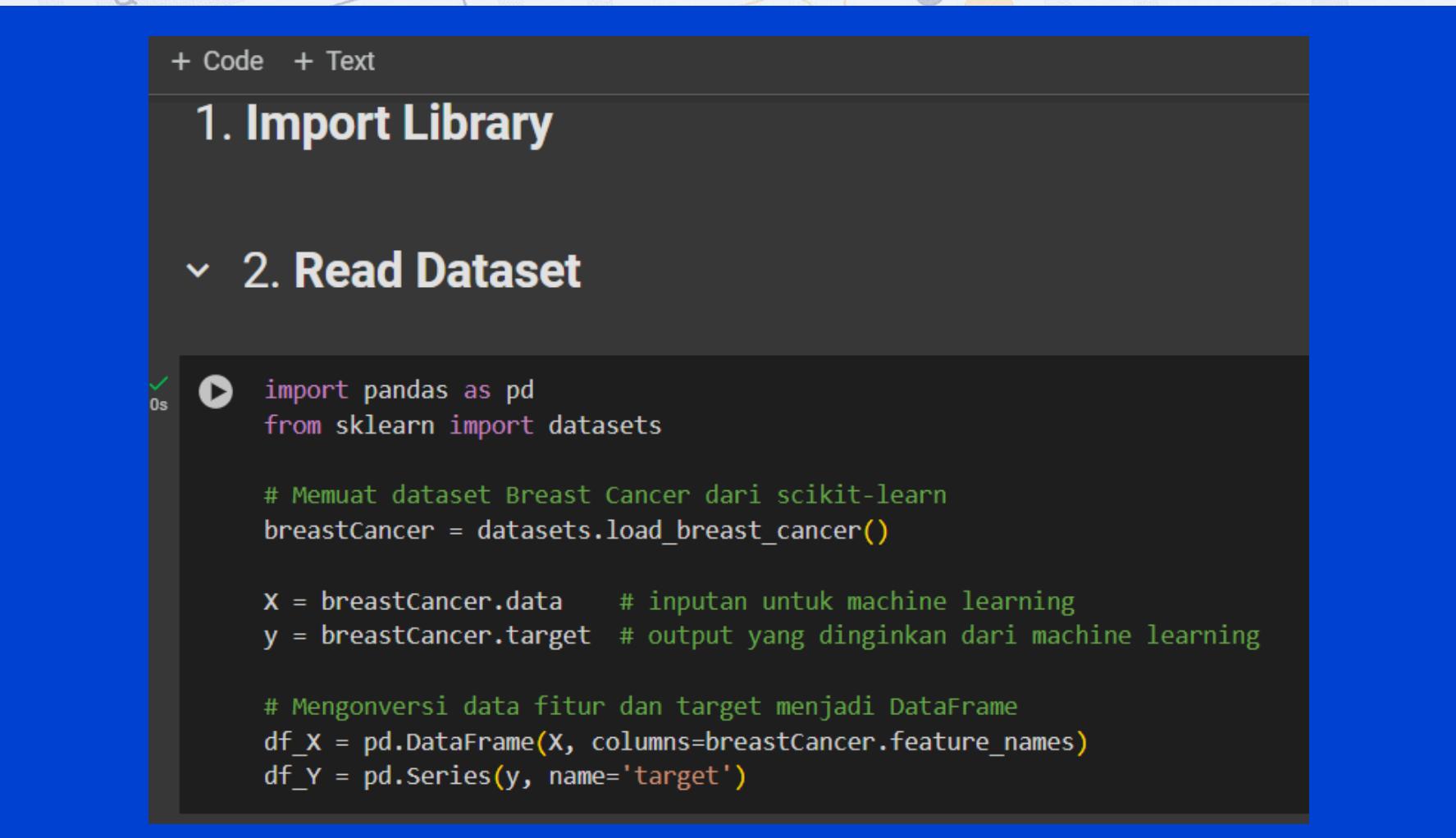
Tools Used



Machine Learning Application on Breast Cancer Dataset

Import and Read Breast Cancer dataset From Scikit-learn

The breast cancer dataset from Scikit-learn is a collection of data used for the classification of breast cancer. It contains medical information collected from biopsies of patients and is used to determine whether a tumor is malignant (cancerous) or benign (non-cancerous).



```
+ Code + Text
1. Import Library
2. Read Dataset

import pandas as pd
from sklearn import datasets

# Memuat dataset Breast Cancer dari scikit-learn
breastCancer = datasets.load_breast_cancer()

X = breastCancer.data      # inputan untuk machine learning
y = breastCancer.target    # output yang dinginkan dari machine learning

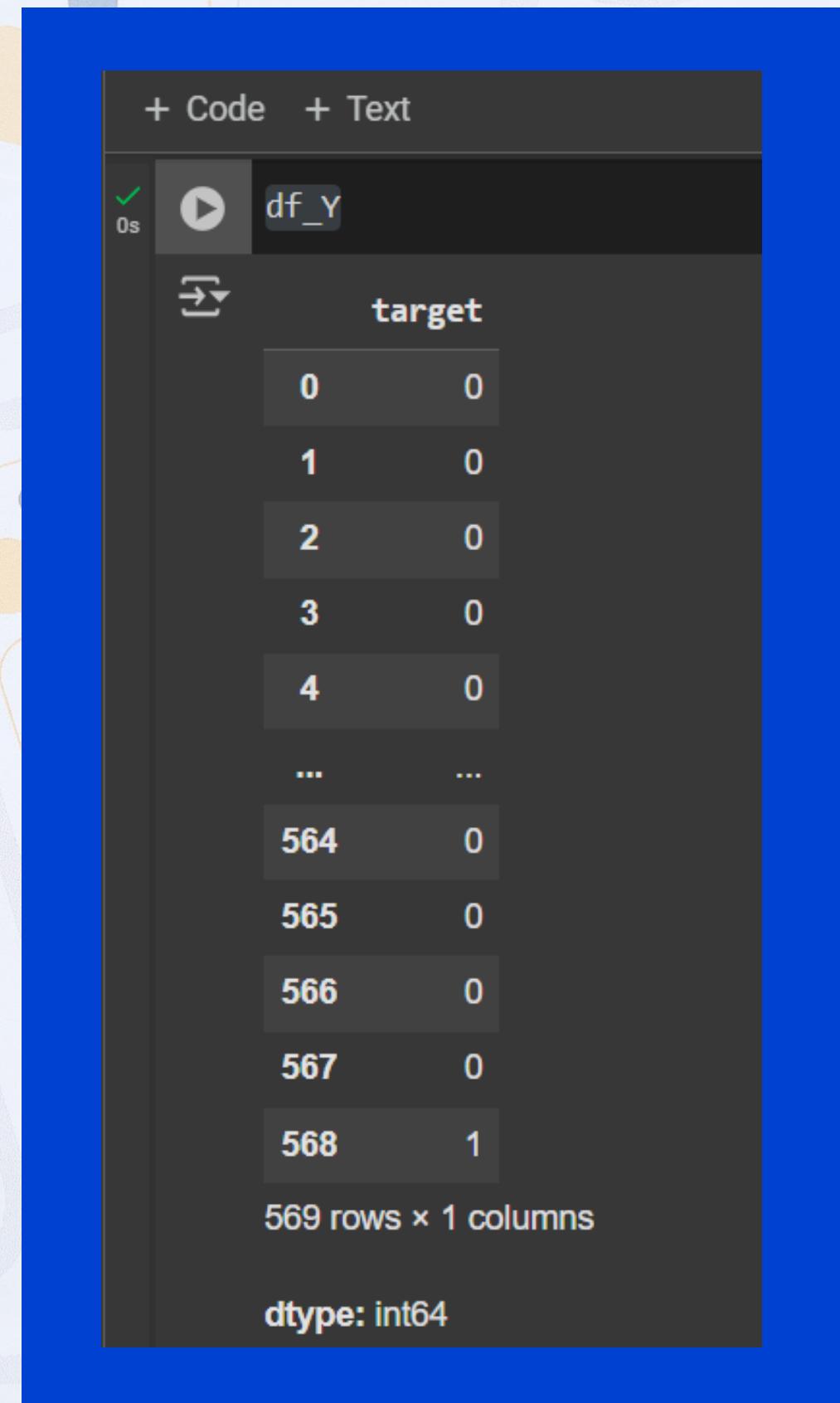
# Mengonversi data fitur dan target menjadi DataFrame
df_X = pd.DataFrame(X, columns=breastCancer.feature_names)
df_Y = pd.Series(y, name='target')
```

Pandas Dataframe

Each row in df_X represents an individual sample (patient), and each column corresponds to one of the 30 different features (attributes) measured from the digital biopsy images, such as cell nucleus size, shape, and texture.

Pandas Series

df_Y is a pandas series that holds the target data from the breast cancer dataset. It contains the classification labels indicating whether each tumor is benign (0) or malignant (1).



A screenshot of a Jupyter Notebook cell. The cell title is 'df_Y'. The content shows a pandas Series named 'target' with 569 rows. The values are mostly 0 (benign) with one value of 1 (malignant) at index 568. The output ends with the series dimensions and type.

	target
0	0
1	0
2	0
3	0
4	0
...	...
564	0
565	0
566	0
567	0
568	1

569 rows × 1 columns
dtype: int64

Combine Feature & Target

The purpose of combining `df_X` and `df_Y` into one DataFrame `df` is to make data manipulation and analysis more convenient. The line `df.head(15)` displays the first 15 rows of this combined DataFrame.

```
+ Code + Text
✓ [128] # Gabungkan fitur dan target dalam satu DataFrame
          df = pds.concat([df_X, df_Y], axis=1)

          df.head(15)

mean mean mean mean mean mean mean
radius texture perimeter area smoothness compactness concavity
0 17.99 10.38 122.80 1001.0 0.11840 0.27760 0.30010
1 20.57 17.77 132.90 1326.0 0.08474 0.07864 0.08690
2 19.69 21.25 130.00 1203.0 0.10960 0.15990 0.19740
3 11.42 20.38 77.58 386.1 0.14250 0.28390 0.24140
4 20.29 14.34 135.10 1297.0 0.10030 0.13280 0.19800
5 12.45 15.70 82.57 477.1 0.12780 0.17000 0.15780
6 18.25 19.98 119.60 1040.0 0.09463 0.10900 0.11270
7 13.71 20.83 90.20 577.9 0.11890 0.16450 0.09366
8 13.00 21.82 87.50 519.8 0.12730 0.19320 0.18590
9 12.46 24.04 83.97 475.9 0.11860 0.23960 0.22730
10 16.02 23.24 102.70 797.8 0.08206 0.06669 0.03299
11 15.78 17.89 103.60 781.0 0.09710 0.12920 0.09954
12 19.17 24.80 132.40 1123.0 0.09740 0.24580 0.20650
13 15.85 23.95 103.70 782.7 0.08401 0.10020 0.09938
14 13.73 22.61 93.60 578.3 0.11310 0.22930 0.21280
```

Exploratory Data Analysis (EDA)

Some Information

```
+ Code + Text
0s df.info()

→ <class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   mean radius      569 non-null    float64
 1   mean texture     569 non-null    float64
 2   mean perimeter   569 non-null    float64
 3   mean area        569 non-null    float64
 4   mean smoothness  569 non-null    float64
 5   mean compactness 569 non-null    float64
 6   mean concavity   569 non-null    float64
 7   mean concave points 569 non-null  float64
 8   mean symmetry    569 non-null    float64
 9   mean fractal dimension 569 non-null  float64
 10  radius error    569 non-null    float64
 11  texture error   569 non-null    float64
 12  perimeter error 569 non-null    float64
 13  area error      569 non-null    float64
 14  smoothness error 569 non-null    float64
 15  compactness error 569 non-null    float64
 16  concavity error  569 non-null    float64
 17  concave points error 569 non-null  float64
 18  symmetry error   569 non-null    float64
 19  fractal dimension error 569 non-null  float64
 20  worst radius     569 non-null    float64
 21  worst texture    569 non-null    float64
 22  worst perimeter   569 non-null    float64
 23  worst area       569 non-null    float64
 24  worst smoothness 569 non-null    float64
 25  worst compactness 569 non-null    float64
 26  worst concavity   569 non-null    float64
 27  worst concave points 569 non-null  float64
```

Identify Unique Values

```
+ Code + Text
0s df['target'].unique()

→ array([0, 1])
```

Statistic

```
+ Code + Text
0s [121] df.describe()

→
   mean radius      mean texture      mean perimeter      mean area      mean smoothness      mean compactness      mean concavity      mean concave points      mean symmetry      mean fractal dimension      ...      worst texture      worst perimeter
count  569.000000  569.000000  569.000000  569.000000  569.000000  569.000000  569.000000  569.000000  569.000000  569.000000  ...
mean   14.127292  19.289649  91.969033  654.889104  0.096360  0.104341  0.088799  0.048919  0.181162  0.062798  ...
std    3.524049  4.301036  24.298981  351.914129  0.014064  0.052813  0.079720  0.038803  0.027414  0.007060  ...
min    6.981000  9.710000  43.790000  143.500000  0.052630  0.019380  0.000000  0.000000  0.106000  0.049960  ...
25%   11.700000  16.170000  75.170000  420.300000  0.086370  0.064920  0.029560  0.020310  0.161900  0.057700  ...
50%   13.370000  18.840000  86.240000  551.100000  0.095870  0.092630  0.061540  0.033500  0.179200  0.061540  ...
75%   15.780000  21.800000  104.100000  782.700000  0.105300  0.130400  0.130700  0.074000  0.195700  0.066120  ...
max   28.110000  39.280000  188.500000  2501.000000  0.163400  0.345400  0.426800  0.201200  0.304000  0.097440  ...
8 rows × 31 columns
```

Split Data & Train The Model

The main objectives of splitting the data and training a machine learning model are to ensure the model generalizes well to new data and to accurately evaluate its performance.

```
+ Code + Text  
✓ [122] from sklearn.model_selection import train_test_split  
  
# Membagi data menjadi train dan test  
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
```

```
+ Code + Text  
✓ [123] from sklearn import datasets, linear_model, metrics  
  
digits = datasets.load_digits()  
  
reg = linear_model.LogisticRegression(max_iter=10000, random_state=1)  
reg.fit(x_train, y_train)  
  
LogisticRegression  
LogisticRegression(max_iter=10000, random_state=1)
```

Predict & Evaluate

This process helps in assessing how well the Logistic Regression model performs on the test data, providing both an accuracy metric and a comprehensive classification report.

```
+ Code + Text
[124] y_pred = reg.predict(x_test)

print(f"Logistic Regression model accuracy: {metrics.accuracy_score(y_test, y_pred) * 100:.2f}%")

→ Logistic Regression model accuracy: 94.74%


[127] from sklearn.metrics import accuracy_score, classification_report

print(classification_report(y_test, y_pred, target_names=breastCancer.target_names))

→
```

	precision	recall	f1-score	support
malignant	0.97	0.88	0.93	42
benign	0.93	0.99	0.96	72
accuracy			0.95	114
macro avg	0.95	0.93	0.94	114
weighted avg	0.95	0.95	0.95	114

Confusion Matrix

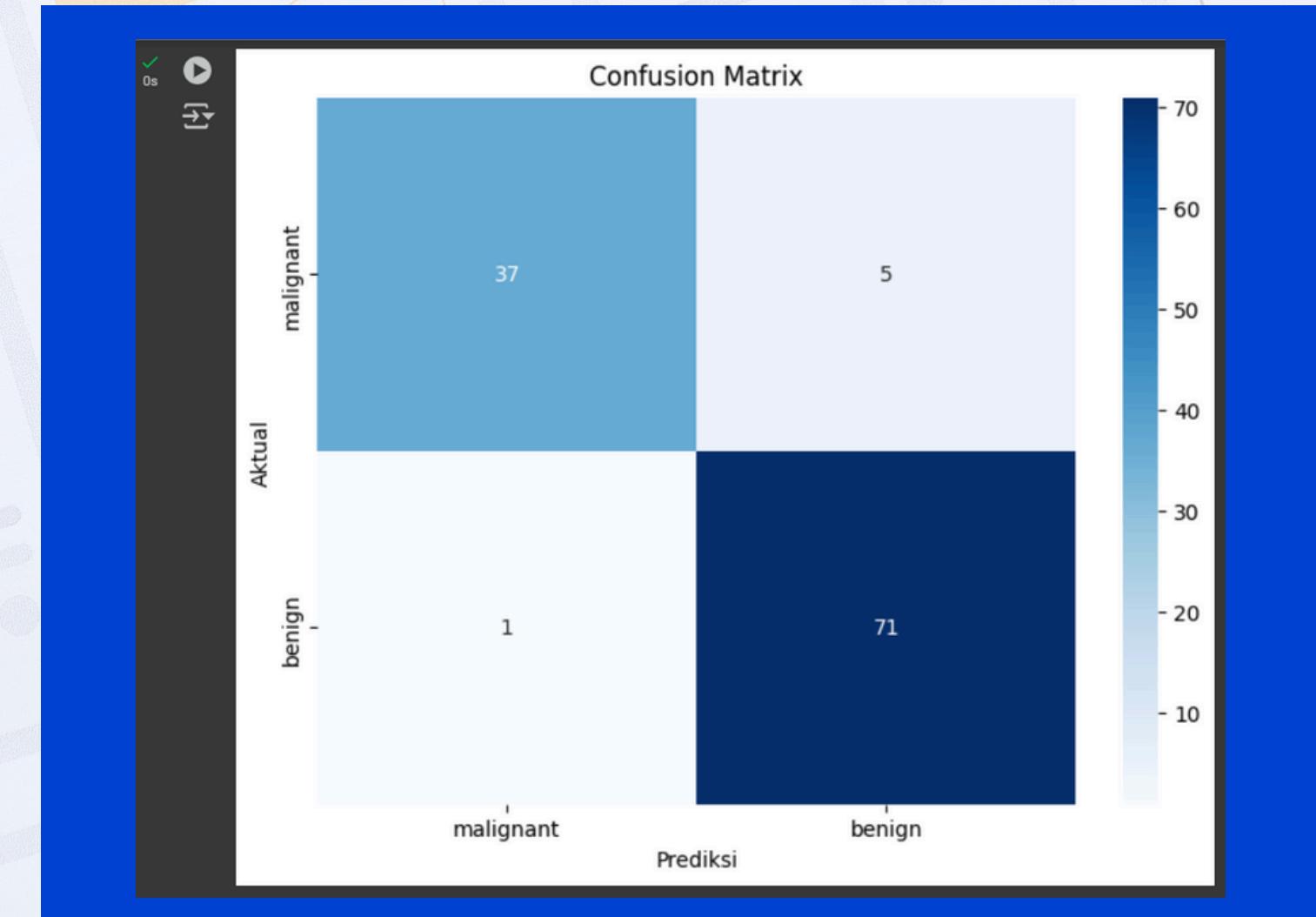
A confusion matrix is used to evaluate the performance of a classification model by comparing the predicted labels with the actual labels.

It provides a detailed breakdown of the model's performance across different classes.

```
+ Code + Text
0s
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix # Import the confusion_matrix function

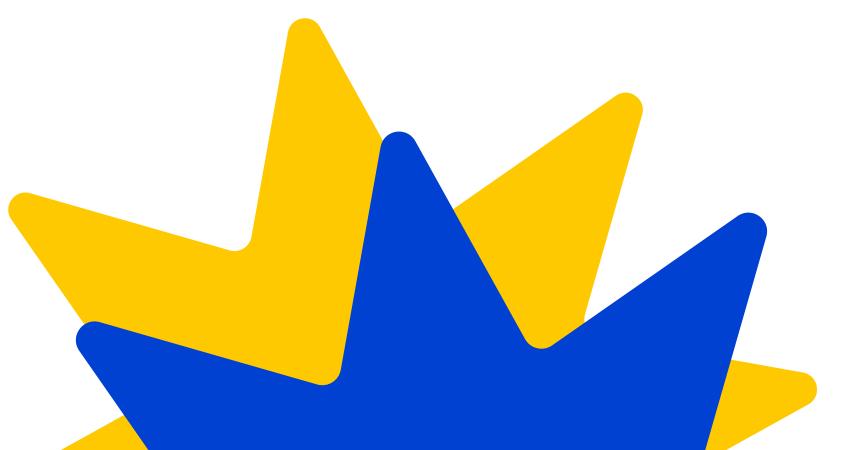
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

# Plot Confusion Matrix using seaborn
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=breastCancer.target_names, yticklabels=breastCancer.target_names)
plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix")
plt.show()
```



Conclusion

The application of machine learning, specifically through the use of logistic regression, provides a valuable and precise method for the early detection and diagnosis of breast cancer. This technique is adept at classifying tumors as either malignant or benign by analyzing characteristics such as size, shape, and texture derived from medical assessments.



Contact Information



luthfiwulandari153@gmail.com



(+62) 81362955309