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.

Breast Cancer

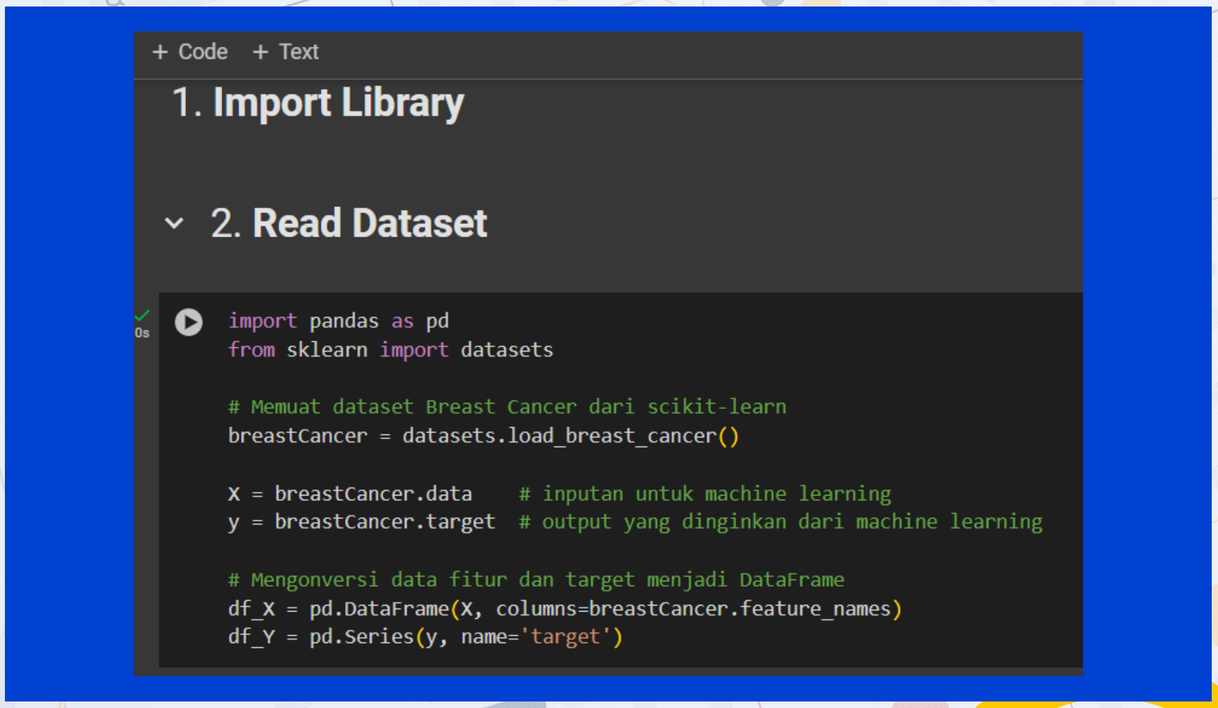
The Breast Cancer dataset from scikit-learn is used for binary classification and consists of medical data from breast cancer biopsies to classify tumors as malignant or benign. It contains 30 features related to cell properties measured from digital biopsy images and includes 569 patient records. This dataset is extensively employed in training and testing machine learning models to study tumor characteristics and assess model performance using various metrics.

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).



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.

+ Code + Text

RAM
Disk

[116] df_X

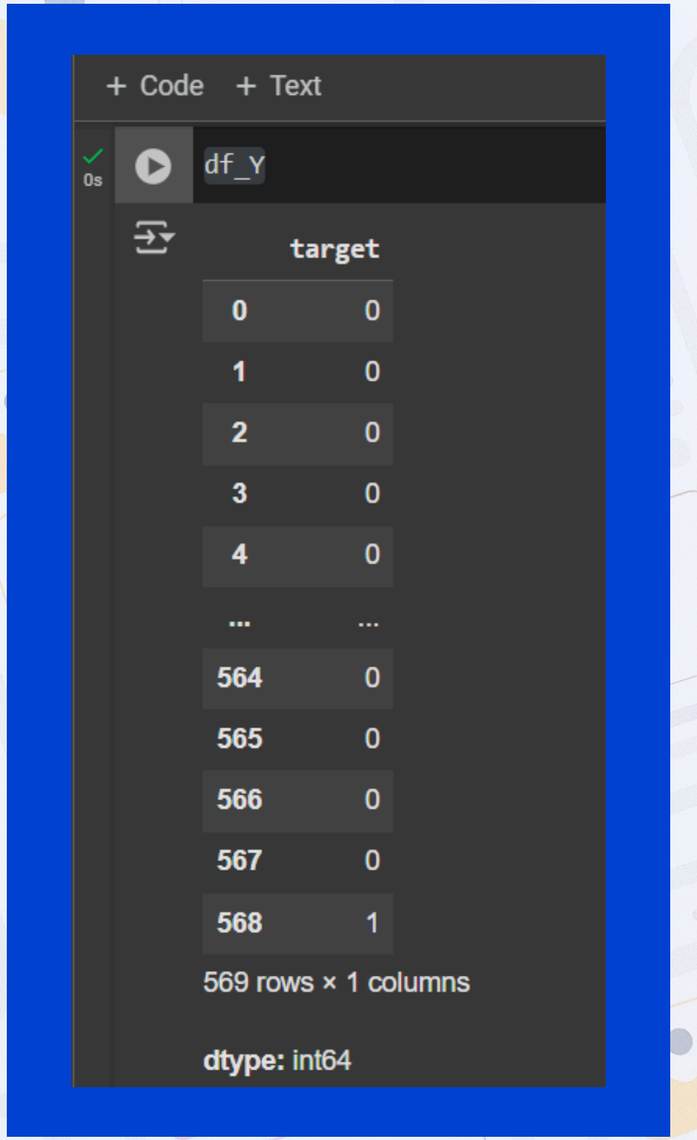
0s

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	w con po
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...	25.380	17.33	184.60	2019.0	0.16220	0.66560	0.7119	0.
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	...	24.990	23.41	158.80	1956.0	0.12380	0.18660	0.2416	0.
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	...	23.570	25.53	152.50	1709.0	0.14440	0.42450	0.4504	0.
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	...	14.910	26.50	98.87	567.7	0.20980	0.86630	0.6869	0.
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	...	22.540	16.67	152.20	1575.0	0.13740	0.20500	0.4000	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623	...	25.450	26.40	166.10	2027.0	0.14100	0.21130	0.4107	0.
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533	...	23.690	38.25	155.00	1731.0	0.11660	0.19220	0.3215	0.
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648	...	18.980	34.12	126.70	1124.0	0.11390	0.30940	0.3403	0.
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016	...	25.740	39.42	184.60	1821.0	0.16500	0.86810	0.9387	0.
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884	...	9.456	30.37	59.16	268.6	0.08996	0.06444	0.0000	0.

569 rows × 30 columns

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).



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

RAM
Disk

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

      df.head(15)
```

	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	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	sym
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.26540	(
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.18600	(
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.24300	(
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.25750	(
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.16250	(
5	12.45	15.70	82.57	477.1	0.12780	0.17000	0.15780	0.08089	0.2087	0.07613	...	23.75	103.40	741.6	0.1791	0.5249	0.5355	0.17410	(
6	18.25	19.98	119.60	1040.0	0.09463	0.10900	0.11270	0.07400	0.1794	0.05742	...	27.66	153.20	1606.0	0.1442	0.2576	0.3784	0.19320	(
7	13.71	20.83	90.20	577.9	0.11890	0.16450	0.09366	0.05985	0.2196	0.07451	...	28.14	110.60	897.0	0.1654	0.3682	0.2678	0.15560	(
8	13.00	21.82	87.50	519.8	0.12730	0.19320	0.18590	0.09353	0.2350	0.07389	...	30.73	106.20	739.3	0.1703	0.5401	0.5390	0.20600	(
9	12.46	24.04	83.97	475.9	0.11860	0.23960	0.22730	0.08543	0.2030	0.08243	...	40.68	97.65	711.4	0.1853	1.0580	1.1050	0.22100	(
10	16.02	23.24	102.70	797.8	0.08206	0.06669	0.03299	0.03323	0.1528	0.05697	...	33.88	123.80	1150.0	0.1181	0.1551	0.1459	0.09975	(
11	15.78	17.89	103.60	781.0	0.09710	0.12920	0.09954	0.06606	0.1842	0.06082	...	27.28	136.50	1299.0	0.1396	0.5609	0.3965	0.18100	(
12	19.17	24.80	132.40	1123.0	0.09740	0.24580	0.20650	0.11180	0.2397	0.07800	...	29.94	151.70	1332.0	0.1037	0.3903	0.3639	0.17670	(
13	15.85	23.95	103.70	782.7	0.08401	0.10020	0.09938	0.05364	0.1847	0.05338	...	27.66	112.00	876.5	0.1131	0.1924	0.2322	0.11190	(
14	13.73	22.61	93.60	578.3	0.11310	0.22930	0.21280	0.08025	0.2069	0.07682	...	32.01	108.80	697.7	0.1651	0.7725	0.6943	0.22080	(

Exploratory Data Analysis (EDA)

Some Information

```
+ Code  + Text

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

df['target'].unique()

array([0, 1])
```

Statistic

```
+ Code  + Text

[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	...	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	...	25.677223	107.261213
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060	...	6.146258	33.602542
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960	...	12.020000	50.410000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700	...	21.080000	84.110000
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540	...	25.410000	97.660000
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120	...	29.720000	125.400000
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440	...	49.540000	251.200000

8 rows × 31 columns

Split Data & Train The Model

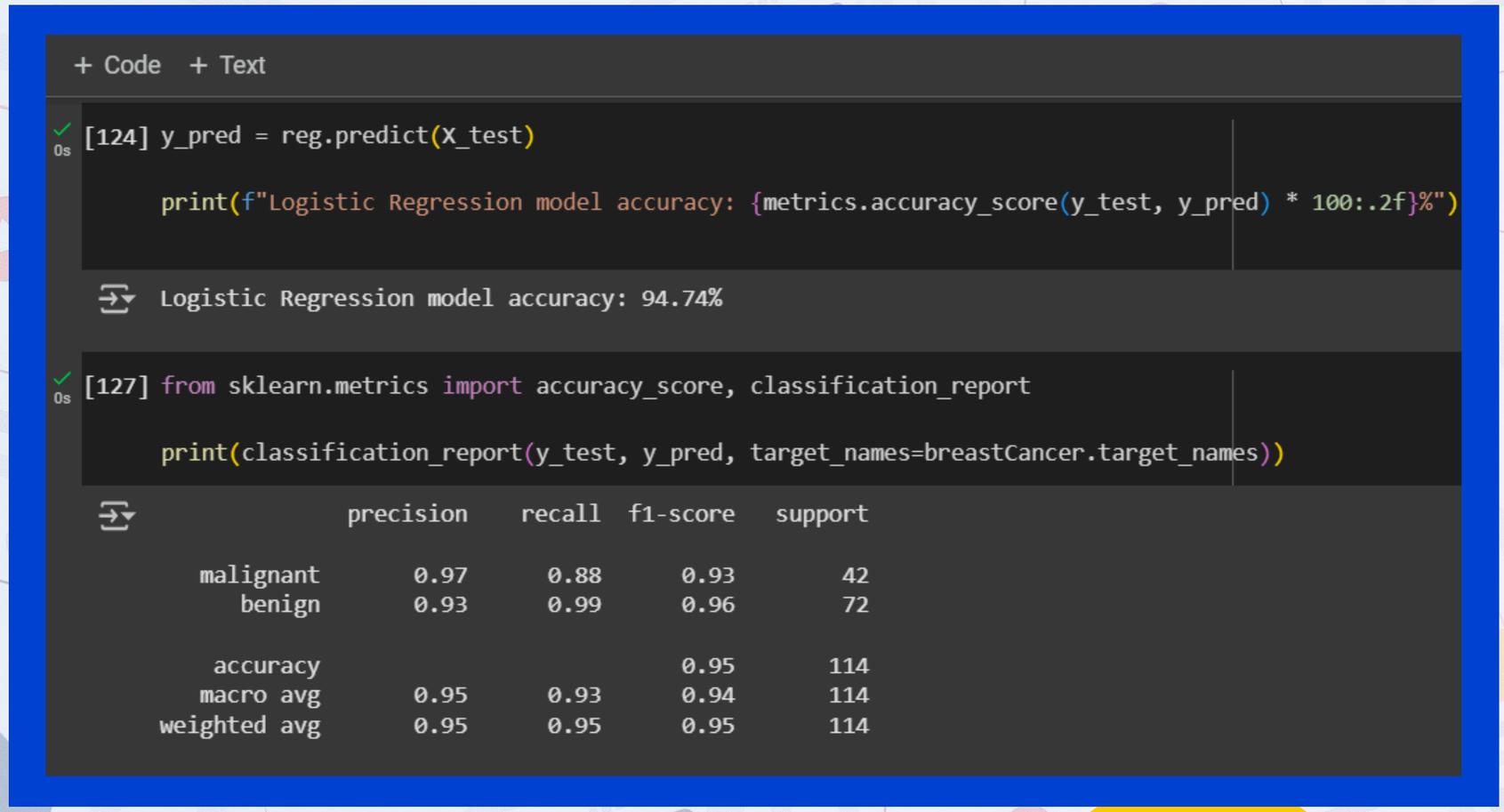
The main objectives of splitting the data and training a machine learning model are to ensure the model generalizes well to new data and to accurately evaluate its performance.

```
+ 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

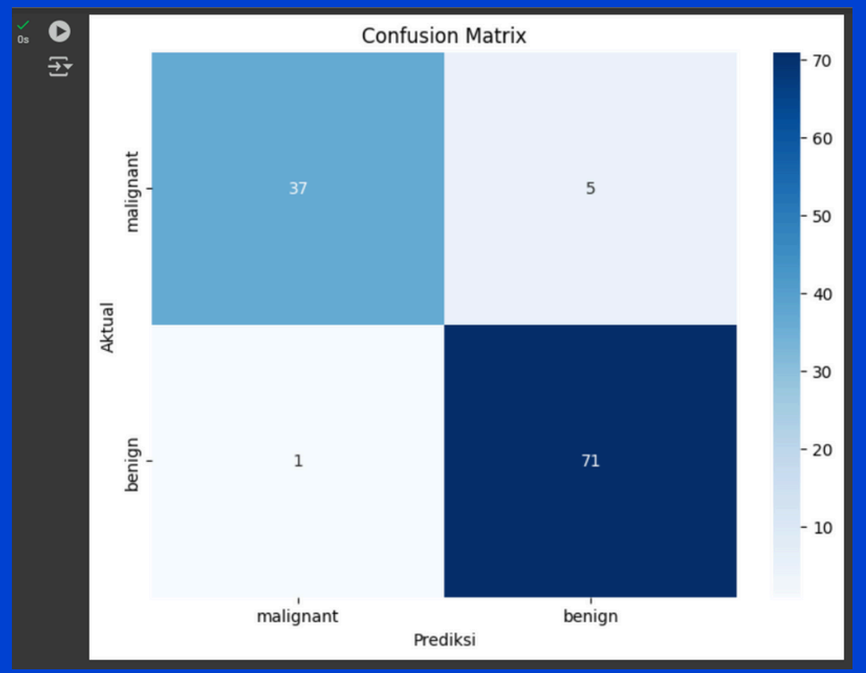
This process helps in assessing how well the Logistic Regression model performs on the test data, providing both an accuracy metric and a comprehensive classification report.



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