CLASSIFYING THE BREAST CANCER DATASET (SCIKIT-LEARN) USING THE K-NEAREST NEIGHBORS (KNN) ALGORITHM

Presented by Nabila Aulia Rahmah

Breast Cancer Wisconsin (Diagnostic) Dataset offers an opportunity to apply machine learning techniques in this research. A crucial step before using any machine learning algorithm is conducting Exploratory Data Analysis (EDA) to understand the dataset's characteristics. This dataset was obtained by analyzing the cell nuclei characteristics of 569 images obtained by Fine Needle Aspiration of the breast mass. Each of the images are classified(diagnosed) as being “Benign” or “Malignant”.

DATASET OVERVIEW

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	worst symmetry	worst fractal dimension	target
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	0.2654	0.4601	0.11890	0
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	0.1860	0.2750	0.08902	0
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	0.2430	0.3613	0.08758	0
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	0.2575	0.6638	0.17300	0
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	0.1625	0.2364	0.07678	0
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	0.1741	0.3985	0.12440	0
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	0.1932	0.3063	0.08368	0
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	0.1556	0.3196	0.11510	0
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	0.2060	0.4378	0.10720	0
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	0.2210	0.4366	0.20750	0

Source : Scikit-learn Breast Cencer Wisconsin Dataset
IInstance : 569 sample
Features : 30 numeric attributes
Classes : Malignant & Benign

Tools :
Programming : Python (Google Colab)
Libraries : Scikit-learn, Pandas, Numpy, Matplotlib, Seaborn

EXPLORATION DATA AND ANALYSIS (EDA)

UNDERSTANDING THE DATASET & DATA TYPES

DISPLAYING INFORMATION FROM THE BREAST CANCER DATASET OBTAINED FROM SCIKIT-LEARN.

```
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
 28  worst symmetry          569 non-null    float64
 29  worst fractal dimension 569 non-null    float64
 30  target                  569 non-null    int64
dtypes: float64(30), int64(1)
memory usage: 137.9 KB
```

CHECKING MISSING VALUE & CLASS DISTRIBUTION

```
[ ]  #Check for null values
     null_values = df.isnull().values.any()
     if null_values == True:
         print("There are some missing values in data")
     else:
         print("There are no missing values in the dataset")
```

```
There are no missing values in the dataset
```

THERE WERE NO MISSING VALUES FOUND ON THIS DATASET.

UNIQUE VALUES

DISPLAYING THE UNIQUE VALUES OF THE TARGET VARIABLE IN THE BREAST CANCER DATASET.

```
df['target'].unique()
```

```
array([0, 1])
```

```
print("The unique number of data values are")
df.nunique()
```

```
The unique number of data values are

id                         569
diagnosis                    2
radius_mean                456
texture_mean               479
perimeter_mean             522
area_mean                  539
smoothness_mean            474
compactness_mean           537
concavity_mean             537
concave points_mean        542
symmetry_mean              432
fractal_dimension_mean     499
radius_se                  540
texture_se                 519
perimeter_se               533
area_se                    528
smoothness_se              547
compactness_se             541
concavity_se               533
concave points_se          507
symmetry_se                498
fractal_dimension_se       545
radius_worst               457
texture_worst              511
perimeter_worst            514
area_worst                 544
smoothness_worst           411
compactness_worst          529
concavity_worst            539
concave points_worst       492
symmetry_worst             500
fractal_dimension_worst    535
dtype: int64
```

SUMMARY STATISTIC

```
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	worst area	worst smoothness
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	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	880.583128	0.132369
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	569.356993	0.022832
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	185.200000	0.071170
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	515.300000	0.116600
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	686.500000	0.131300
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	1084.000000	0.146000
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	4254.000000	0.222600

8 rows × 31 columns

DISPLAYING THE STATISTICS OF THE BREAST CANCER DATASET

DATA SPLIT

```
[15] from sklearn.model_selection import train_test_split

     #Split the data into train and test
     X_train, X_test, y_train, y_test = train_test_split(df_X, df_y, test_size=0.2, random_state=42)
```

IT FUNCTIONS TO SPLIT THE DATA INTO TESTING AND TRAINING SETS, WITH 20% ALLOCATED FOR TESTING AND 80% FOR TRAINING.

MODEL BUILDING & PREDICTION

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=10)
model.fit(X_train, y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=10)
```

```
[17] from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

     y_pred = model.predict(X_test)

     knnaccuracy = accuracy_score(y_test, y_pred)

     print("Laporan Klasifikasi:")
     print('\nModel Accuracy: {0:.2f}%'.format(knnaccuracy * 100))
```

```
Laporan Klasifikasi:

Model Accuracy: 97.37%
```

MODEL BUILDING IS CREATED WITH K-NEIREST NEIGHBOARS (KNN)
PREDICTION IS USED TO PREDICT THE TEST DATA

CLASSIFICATION REPORT

```
[18] class_report = classification_report(y_test, y_pred, target_names=breast_cancer.target_names)

     print("Classification Report: ")
     print(class_report)
```

```
Classification Report:
                precision    recall  f1-score   support

     malignant       0.98      0.95      0.96        43
        benign       0.97      0.99      0.98        71

      accuracy                           0.97       114
     macro avg       0.97      0.97      0.97       114
  weighted avg       0.97      0.97      0.97       114
```

SHOW THE MAIN EVALUATION MATRICS

1. PRECISION : POSITIVE PREDICTION ACCURACY
2. RECALL : ABILITY OF THE MODEL TO DETECT THE TRUE CLASS
3. FI-SCORE : BALANCE BETWEEN PRECISION AND RECALL

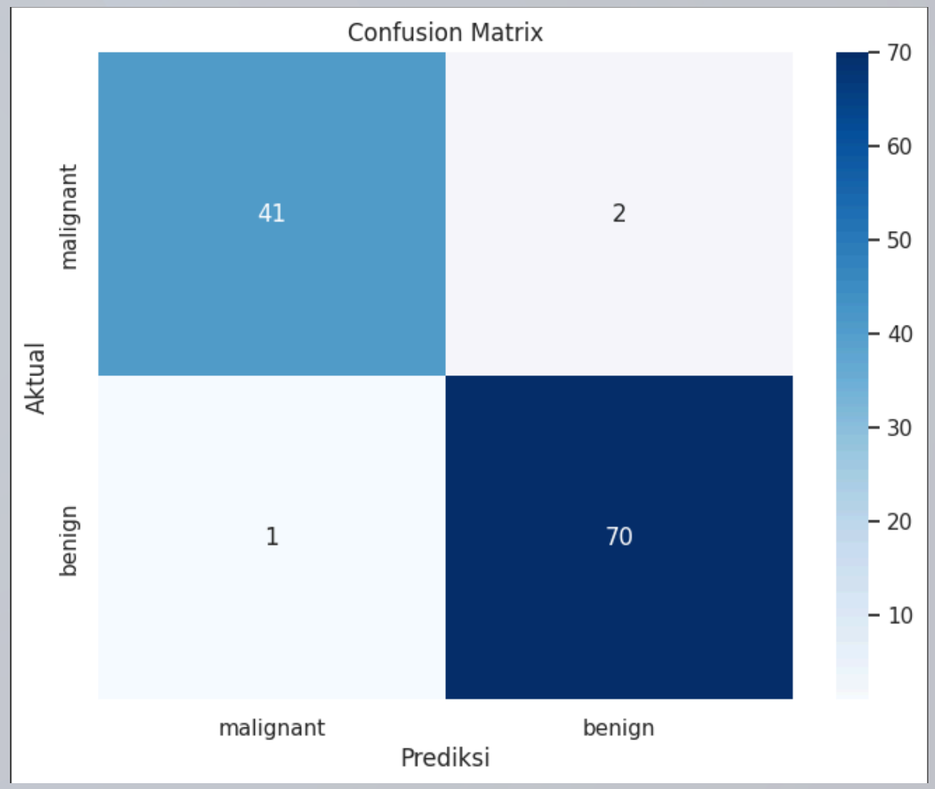
DATA VISUALIZATION

CONFUSION MATRICS

```python
import seaborn as sns
import matplotlib.pyplot as plt

# 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=breast_cancer.target_names,
            yticklabels=breast_cancer.target_names)
plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix")
plt.show()
```



DATA VISUALIZATION

DATA SPREAD

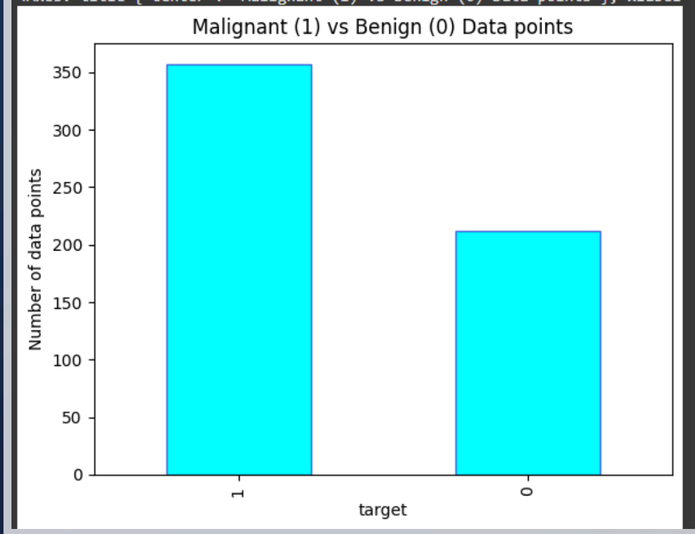
```python
# Count the occurrences of each target class
print(df["target"].value_counts())

# Print the mean to get the proportion of malignant (target = 1) samples
print(df["target"].mean())

# Display the total number of data points and the percentage of each target type
print("Total number of data points = ", len(df))
print("Malignant (target = 1) = {}%".format(round(df["target"].mean(), 3) * 100))
print("Benign (target = 0) = {}%".format((1 - round(df["target"].mean(), 3)) * 100))

# Plot the number of data points in each class (Malignant vs Benign)
df["target"].value_counts().plot.bar(ylabel="Number of data points",
                                     title="Malignant (1) vs Benign (0) Data points",
                                     color='cyan', edgecolor="royalblue")
```

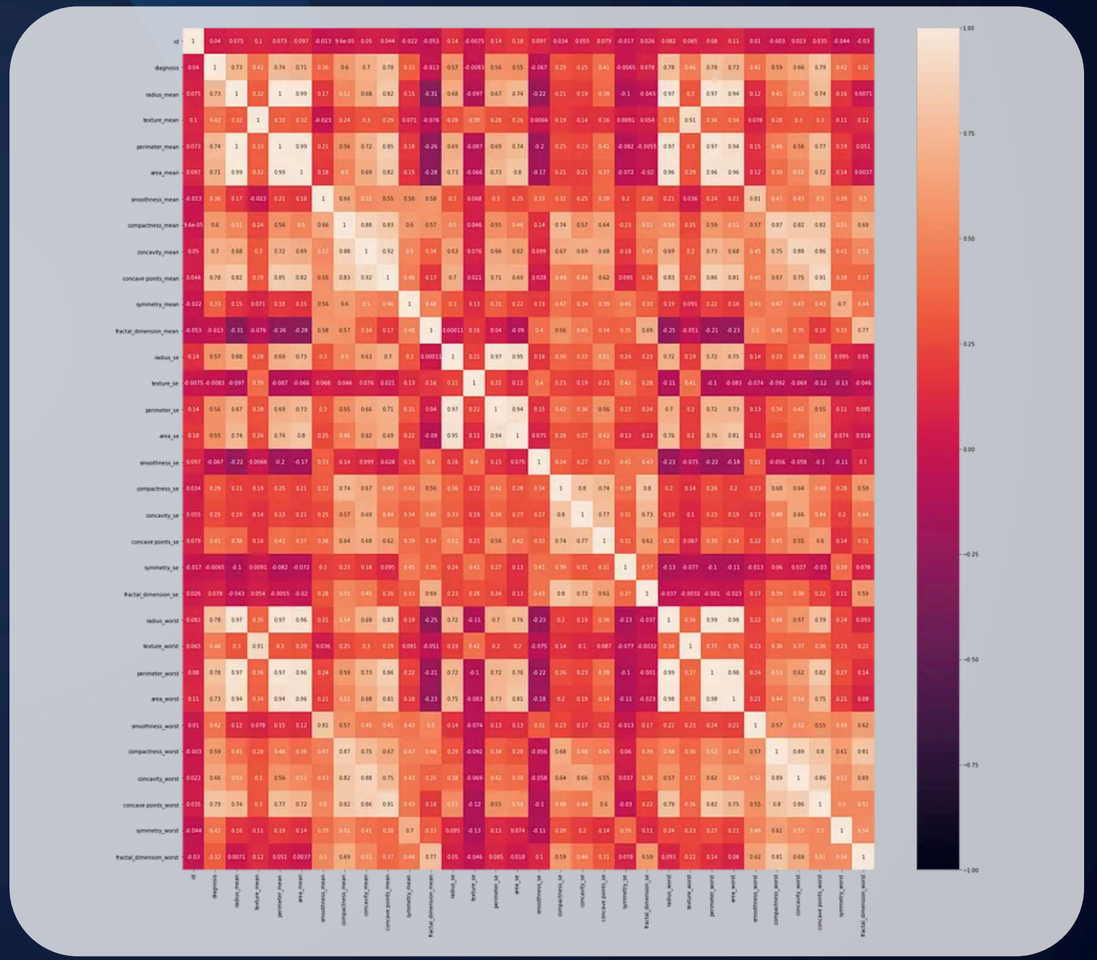
```
target
1    357
0    212
Name: count, dtype: int64
0.6274165202108963
Total number of data points =  569
Malignant (target = 1) = 62.7%
Benign (target = 0) = 37.3%
<Axes: title={'center': 'Malignant (1) vs Benign (0) Data points'}, xlabel='target', ylabel='Number of data points'>
```



DATA VISUALIZATION

HEATMAP

```
#HEATMAP
import seaborn as sns
import matplotlib
import matplotlib.pylab as plt
plt.figure(figsize =(30,30))
sns.heatmap(df.corr(), vmin=-1, vmax=1, annot=True)
plt.show()
```



THANK YOU

Ig : nabilarhmaa
Email : nabilaauliarhmaa