Exp-3

AIM

To analyze the performance of a classifier using Sastical evaluation metrics.

OBJECTIVE

↳ To compute statistical performance measurement such as Precision, recall, F1-score, and confusion matrix.

↳ To interpret the classifier's behavior on test data

PSEUDOCODE.

```
#USE THE TRAINED MODEL FROM EXP-2
y_pred = model.predict(x_test)
from sklearn.metrics import confusion_matrix,
    classification_report.
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)
report = classification_report(y_test, y_pred)
print("Classification Report\n", report).
```

OBSERVATION

Accuracy

Precision

Recall

F1-Score

Classification Report -- Logistic Regression --

	precision	recall	f1-score	suppt
setosa	1.00	1.00	1.00	19
versicolor	1.00	0.62	0.76	13
virginica	0.72	1.00	0.84	13
accuracy			0.89	45
macro avg	0.91	0.87	0.87	45
weighted avg	0.92	0.89	0.88	45

Classification Report -- Linear Regression (Round

	precision	recall	f1score	suppt
setosa	0.92	1.00	0.84	13
versicolor	1.00	1.00	0.76	13
virginica	1.00	0.62	1.00	19
accuracy			0.83	45
macro avg	0.92	0.89	0.87	45
weighted avg	0.91	0.87	0.87	45

RESULT

The classifier was evaluated using statistical parameters. It showed excellent precision recall, f1-score, confirming its reliability for multiclass classification tasks.

File Edit Selection View Go Run Terminal Help

Search

lab_3.ipynb | LAB_2.ipynb

C: > Users > ASUS > Downloads > lab_3.ipynb > M↓ LAB_3

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | ...

Python 3.10.11

LAB_3

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, accuracy_score
# ------------------------------
# Load dataset
# ------------------------------
data = load_breast_cancer()
X, y = data.data, data.target

# ------------------------------
# Split into train and test
# ------------------------------
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# ------------------------------
# Scale features (important for KNN and SVM)
# ------------------------------
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# ------------------------------
#  KNN Classifier
# ------------------------------
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
knn_preds = knn.predict(X_test_scaled)
```

[1]

⊗0 △0 Live Share

Cell 1 of 2

File Edit Selection View Go Run Terminal Help

lab_3.ipynb | LAB_2.ipynb

C: > Users > ASUS > Downloads > lab_3.ipynb > LAB_3

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | Python 3.10.11

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# ----------------------------
# Scale features (important for KNN and SVM)
# ----------------------------
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# ----------------------------
#  KNN Classifier
# ----------------------------
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
knn_preds = knn.predict(X_test_scaled)

# ----------------------------
#  SVM Classifier
# ----------------------------
svm = SVC(kernel='rbf')  # or 'linear'
svm.fit(X_train_scaled, y_train)
svm_preds = svm.predict(X_test_scaled)

# ----------------------------
#  Decision Tree Classifier
# ----------------------------

dt = DecisionTreeClassifier(random_state=42)
dt.fit(X_train, y_train)  # no scaling needed for trees
dt_preds = dt.predict(X_test)

# ----------------------------
# Results
# ----------------------------

print("\n--- KNN Classifier ---")
```

[1]

0 0 Live Share

Cell 1 of 2

File Edit Selection View Go Run Terminal Help

lab_3.ipynb | LAB_2.ipynb

C: > Users > ASUS > Downloads > lab_3.ipynb > LAB_3

Generate | Code | Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline | Python 3.10.11

[1] ✓ 5.5s Python

```
...
    --- KNN Classifier ---
    Accuracy: 0.9590643274853801
    Classification Report:
                  precision    recall  f1-score   support

               0       0.95      0.94      0.94        63
               1       0.96      0.97      0.97       108

        accuracy                           0.96       171
       macro avg       0.96      0.95      0.96       171
    weighted avg       0.96      0.96      0.96       171


    --- SVM Classifier ---
    Accuracy: 0.9766081871345029
    Classification Report:
                  precision    recall  f1-score   support

               0       0.97      0.97      0.97        63
               1       0.98      0.98      0.98       108

        accuracy                           0.98       171
       macro avg       0.97      0.97      0.97       171
    weighted avg       0.98      0.98      0.98       171
    ...
        accuracy                           0.94       171
       macro avg       0.93      0.94      0.94       171
    weighted avg       0.94      0.94      0.94       171
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

⊗ 0 ⚠ 0 Live Share Cell 1 of 2

Upcoming Earnings | Search | ENG IN | 9:41 PM 8/13/2025