

Ex-2
31/7/25

Implement Classifier Using Open Source Dataset

AIM: To implement a classifier using open source dataset

Algorithm:

- * Load IRIS dataset
- * Split dataset into training and testing
- * Preprocess data if necessary
- * Train a K-NN classifier on training data
- * Predict class labels on test data
- * Evaluate classifier

Objective:

- ⇒ Load IRIS dataset
- ⇒ Split the dataset
- ⇒ Preprocess using Standard Scaler
- ⇒ Train a KNN
- ⇒ Predict class labels
- ⇒ Evaluate the classifier

PSEUDO CODE:

⇒ BEGIN

LOAD IRIS DATA SET

split the dataset

Initialize K-NN with K neighbors

Train classifier using training set

Predict class labels on test data

Compute accuracy of models

Print accuracy

END.



Code:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import
    train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

iris = load_iris()
x = iris.data
y = iris.target

x_train, x_test, y_train, y_test =
    train_test_split(x, y, test_size=0.2,
                    random_state=42)

scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)

knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(x_train, y_train)
y_pred = knn.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

Output:

Accuracy: 1.0


```
[22]: from sklearn.metrics import accuracy_score

[23]: accuracy_score(y_test,y_pred)

[23]: 0.9694444444444444

[24]: from sklearn import metrics

[25]: confusion_matrix=metrics.confusion_matrix(y_test,y_pred)
      confusion_matrix

[25]: array([[33,  0,  0,  0,  0,  0,  0,  0,  0,  0],
          [ 0, 28,  0,  0,  0,  0,  0,  0,  0,  0],
          [ 0,  0, 33,  0,  0,  0,  0,  0,  0,  0],
          [ 0,  0,  0, 33,  0,  1,  0,  0,  0,  0],
          [ 0,  1,  0,  0, 44,  0,  1,  0,  0,  0],
          [ 0,  0,  1,  0,  0, 44,  1,  0,  0,  1],
          [ 0,  0,  0,  0,  0,  1, 34,  0,  0,  0],
          [ 0,  0,  0,  0,  0,  1,  0, 33,  0,  0],
          [ 0,  0,  0,  0,  0,  1,  0,  0, 29,  0],
          [ 0,  0,  0,  1,  0,  0,  0,  0,  1, 38]])

[26]: cm_display=metrics.ConfusionMatrixDisplay(confusion_matrix=confusion_matrix,display_labels=[0,1])
      cm_display

[26]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x720a307ade40>

[27]: pip install matplotlib
```

L2.ipynb (4) - JupyterLab

10.1.38.19/user/ra2311047010019/lab/tree/DEEP%20LEARNING/L2.ipynb

Verify it's you

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/ DEEP LEARNING /

Name	Last Modified
datasets	6 days ago
EX1.ipynb	20 days ago
L_3.ipynb	next year
L2.ipynb	next year
L3.ipynb	next year
Logistic reg...	13 days ago

Logistic regression w

L_3.ipynb

L2.ipynb

EX1.ipynb

breast_cancer_bd.csv

jupyter-ra231104701

Code

Notebook

Python 3 (ipykernel)

[1]:

from sklearn.datasets import load_digits

[2]:

d=load_digits()

[34]:

x=d.data

y=d.target

[35]:

x

[35]:

array([[0., 0., 5., ..., 0., 0., 0.],
 [0., 0., 0., ..., 10., 0., 0.],
 [0., 0., 0., ..., 16., 9., 0.],
 ...,
 [0., 0., 1., ..., 6., 0., 0.],
 [0., 0., 2., ..., 12., 0., 0.],
 [0., 0., 10., ..., 12., 1., 0.]], shape=(1797, 64))

[36]:

y

[36]:

array([0, 1, 2, ..., 8, 9, 8], shape=(1797,))

[37]:

from sklearn.model_selection import train_test_split

[38]:

x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2,random_state=42)

[39]:

from sklearn.neighbors import KNeighborsClassifier

Simple

1

4

Python 3 (ipykernel) | Idle

Mem: 283.78 MB

Mode: Command

Ln 1, Col 1

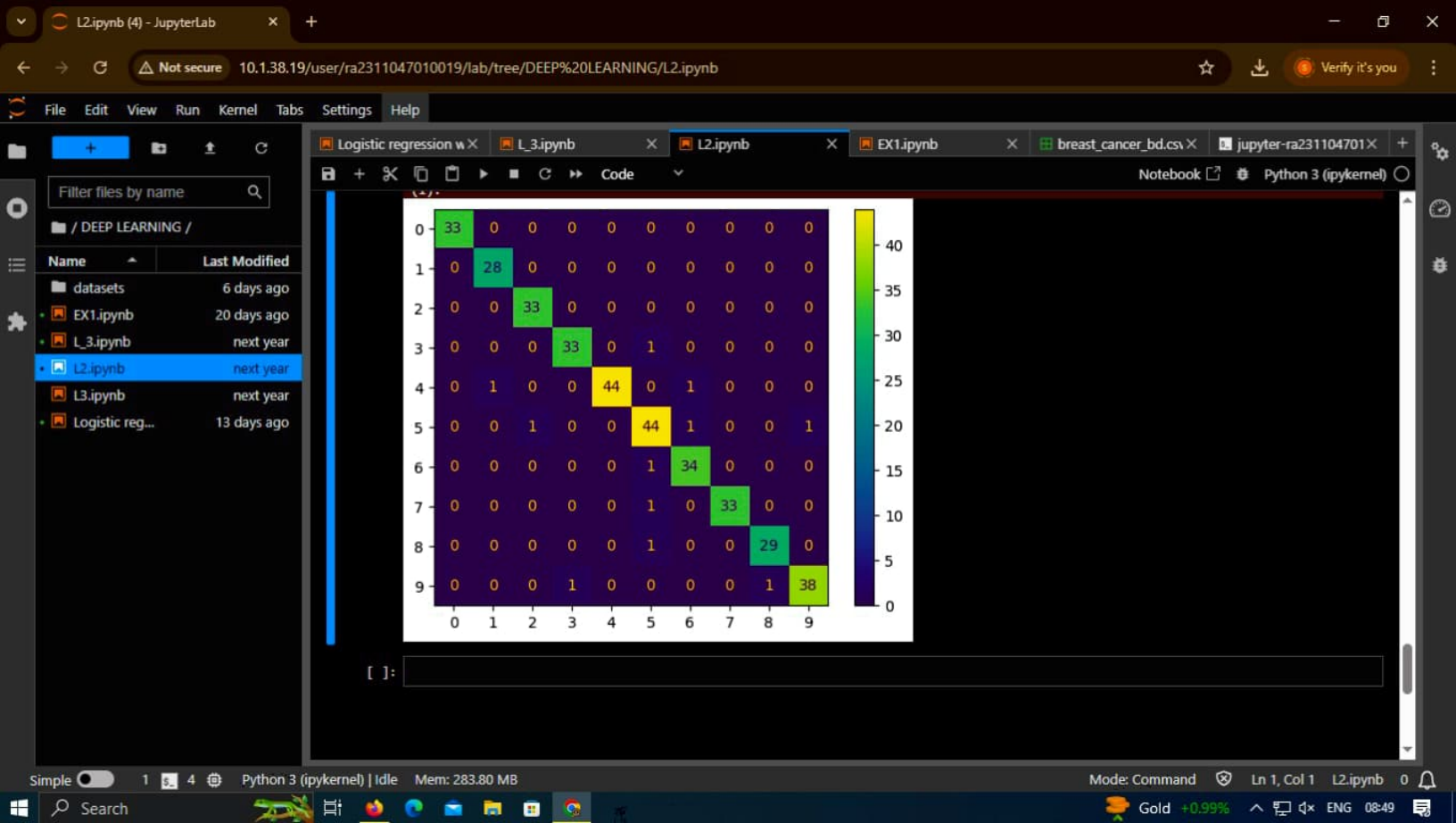
L2.ipynb

0

Nifty midcap -0.96%

ENG

08:47



File Edit View Run Kernel Tabs Settings Help

Filter files by name

/ DEEP LEARNING /

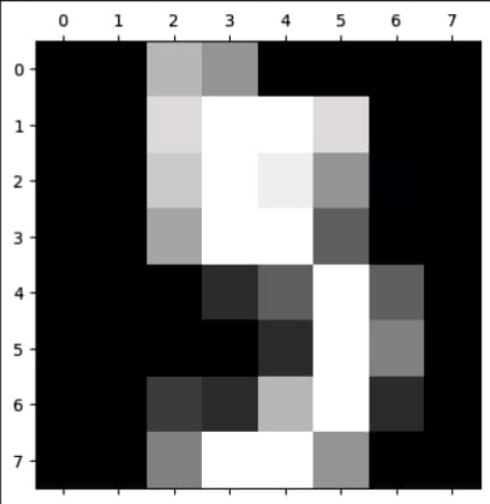
Name	Last Modified
datasets	6 days ago
EX1.ipynb	20 days ago
L_3.ipynb	next year
L2.ipynb	next year
L3.ipynb	next year
Logistic reg...	13 days ago

Logistic regression w x L_3.ipynb L2.ipynb EX1.ipynb breast_cancer_bd.csv jupyter-ra231104701

Notebook Python 3 (ipykernel)

[32]: plt.matshow(d.images[5], cmap="gray")

[32]: <matplotlib.image.AxesImage at 0x720a94ef9ba0>



[33]: cm_display.plot()
plt.show()

Simple 1 4 Python 3 (ipykernel) | Idle Mem: 283.80 MB Mode: Command Ln 1, Col 1 L2.ipynb 0

L2.ipynb (4) - JupyterLab

10.1.38.19/user/ra2311047010019/lab/tree/DEEP%20LEARNING/L2.ipynb

Verify it's you

File Edit View Run Kernel Tabs Settings Help

Filter files by name

/ DEEP LEARNING /

Name	Last Modified
datasets	6 days ago
EX1.ipynb	20 days ago
L_3.ipynb	next year
L2.ipynb	next year
L3.ipynb	next year
Logistic reg...	13 days ago

Logistic regression w X L_3.ipynb L2.ipynb EX1.ipynb breast_cancer_bd.csv jupyter-ra231104701

Code

Python 3 (ipykernel)

```
[40]: knn = KNeighborsClassifier()

[41]: from sklearn.linear_model import LogisticRegression

[42]: clf = LogisticRegression()

[20]: clf.fit(x_train, y_train)

/home/jupyter-ra2311047010011/.local/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:473: ConvergenceWarning: lb
fgs failed to converge after 100 iteration(s) (status=1):
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT

Increase the number of iterations to improve the convergence (max_iter=100).
You might also want to scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
LogisticRegression()
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

[21]: y_pred=clf.predict(x_test)
y_pred

[21]: array([6. 9. 3. 7. 2. 1. 5. 2. 5. 2. 1. 9. 4. 0. 4. 2. 3. 7. 8. 8. 4. 3.]
```

Simple 1 4 Python 3 (ipykernel) | Idle Mem: 283.80 MB

Mode: Command Ln 1, Col 1 L2.ipynb 0

NIFTY -0.55%

08:48