

DML MINI PROJECT

AIM: Handwritten digit recognition using mnist dataset

What is MNIST?

1. Set of 70,000 small images of digits handwritten by high school students and employees of the US Census Bureau.
2. All images are labeled with the respective digit they represent.
3. MNIST is the hello world of machine learning. Every time a data scientist or machine learning engineer makes a new algorithm for classification, they would always first check its performance on the MNIST dataset.
4. There are 70,000 images and each image has $28 \times 28 = 784$ features.
5. Each image is 28×28 pixels and each feature simply represents one-pixel intensity from 0 to 255. If the intensity is 0, it means that the pixel is white and if it is 255, it means it is black.

CODE:

```
from sklearn.datasets import fetch_openml

import matplotlib

import matplotlib.pyplot as plt

import numpy as np

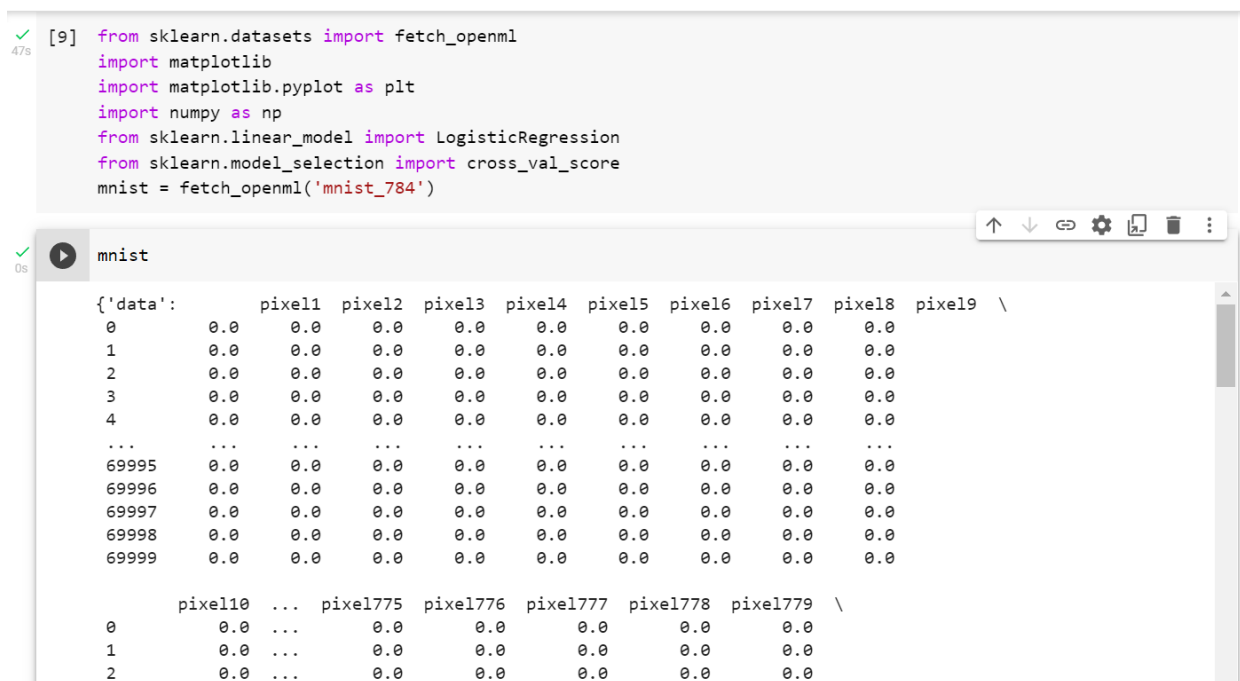
from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import cross_val_score

mnist = fetch_openml('mnist_784')

mnist
```

OUTPUT:



```
[9] from sklearn.datasets import fetch_openml
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
mnist = fetch_openml('mnist_784')
```

mnist

	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	\
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	
69995	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
69996	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
69997	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
69998	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
69999	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

	pixel10	...	pixel775	pixel776	pixel777	pixel778	pixel779	\
0	0.0	...	0.0	0.0	0.0	0.0	0.0	
1	0.0	...	0.0	0.0	0.0	0.0	0.0	
2	0.0	...	0.0	0.0	0.0	0.0	0.0	

CODE:

```
x, y = mnist['data'], mnist['target']
some_digit = x.to_numpy()[36001]

some_digit_image = some_digit.reshape(28, 28) # let's reshape to plot it
plt.imshow(some_digit_image, cmap=matplotlib.cm.binary,
            interpolation='nearest')
plt.axis("off")
plt.show()
```

OUTPUT:**CODE**

```
x_train, x_test = x[:60000], x[60000:70000]
y_train, y_test = y[:60000], y[60000:70000]
shuffle_index = np.random.permutation(60000)
x_train, y_train = x_train.iloc[shuffle_index], y_train.iloc[shuffle_index]

# Creating a 2-detector
y_train = y_train.astype(np.int8)
y_test = y_test.astype(np.int8)
y_train_2 = (y_train == 2)
y_test_2 = (y_test == 2)
```

```
clf = LogisticRegression(tol=0.1)
```

```
clf.fit(x_train, y_train_2)
```

OUTPUT

```
LogisticRegression(tol=0.1)
```

CODE:

```
a = cross_val_score(clf, x_train, y_train_2, cv=3, scoring="accuracy")  
print(a.mean())
```

In [15]:

OUTPUT:

The screenshot shows a Jupyter Notebook interface. At the top, there is a toolbar with 'Help' and 'Cannot save changes'. Below the toolbar, there are tabs for '+ Code' and '+ Text', and a 'Copy to Drive' button. On the right, there are status indicators for 'RAM' and 'Disk', both with green checkmarks. The main area displays a warning message: 'extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG, 0.9783666666666667'. Above the message is a blue hyperlink: 'https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression'. To the left of the message is a green checkmark and the text '17s'. To the right of the message are up and down arrow buttons.