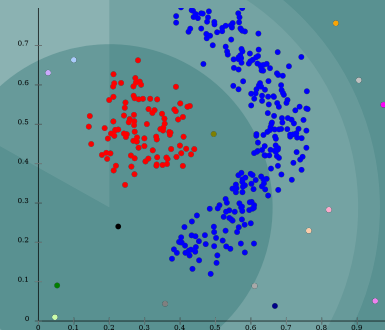


CS550 - Machine Learning and Business Intelligence



Classification on Colab using MNIST dataset

By: Maranata Muluneh

Instructor: Dr. Chang, Henry

2023



Table of Content

- Introduction
- Theory
- Implementation
- Conclusion
- References



Introduction

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data.

Examples of classification problems include:

- Given an example, classify if it is spam or not.
- Given a handwritten character, classify it as one of the known characters.
- Given recent user behavior, classify as churn or not.



Introduction

From a modeling perspective, classification requires a training dataset with many examples of inputs and outputs from which to learn.

A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative



Theory

Type of Classification

- **Binary Classification:** refers to those classification tasks that have two class labels.

Examples include:

- Email spam detection (spam or not).
- Churn prediction (churn or not).
- Conversion prediction (buy or not).



Theory

Multi-Class Classification: refers to those classification tasks that have more than two class labels.

Examples include:

- Face classification.
- Plant species classification.
- Optical character recognition.



Theory

Multi-Label Classification: refers to those classification tasks that have two or more class labels, where one or more class labels may be predicted for each example.

Consider the example of photo classification, where a given photo may have multiple objects in the scene and a model may predict the presence of multiple known objects in the photo, such as “bicycle,” “apple,” “person,” etc.



Implementation

- Here's a step-by-step guide on how to perform image classification on the MNIST dataset using Google Colab:

Step 1: Open Google Colab in your browser and create a new notebook.

Step 2: Import the necessary libraries for the project:

```
import tensorflow as tf
from tensorflow.keras.datasets import mnist
import matplotlib.pyplot as plt
```




Implementation

Step 3: Load the MNIST dataset:

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

Step 4: Check the shape of the dataset:

```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```



Implementation

Step 5: Preprocess the data by scaling the pixel values to be between 0 and 1:

```
x_train = x_train / 255.0  
x_test = x_test / 255.0
```

Step 6: Define the model:

```
model = tf.keras.models.Sequential([  
    tf.keras.layers.Flatten(input_shape=(28, 28)),  
    tf.keras.layers.Dense(128, activation='relu'),  
    tf.keras.layers.Dense(10, activation='softmax')  
])
```



Implementation

Step 9: Plot the accuracy and loss curves:

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```



Implementation

Step 10 : Evaluate the model:

```
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test Loss:', test_loss)
print('Test Accuracy:', test_acc)
```

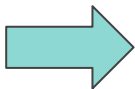


Implementation (Using Colab)

Environment : Colab, Tensorflow 2

Programming Language: Python

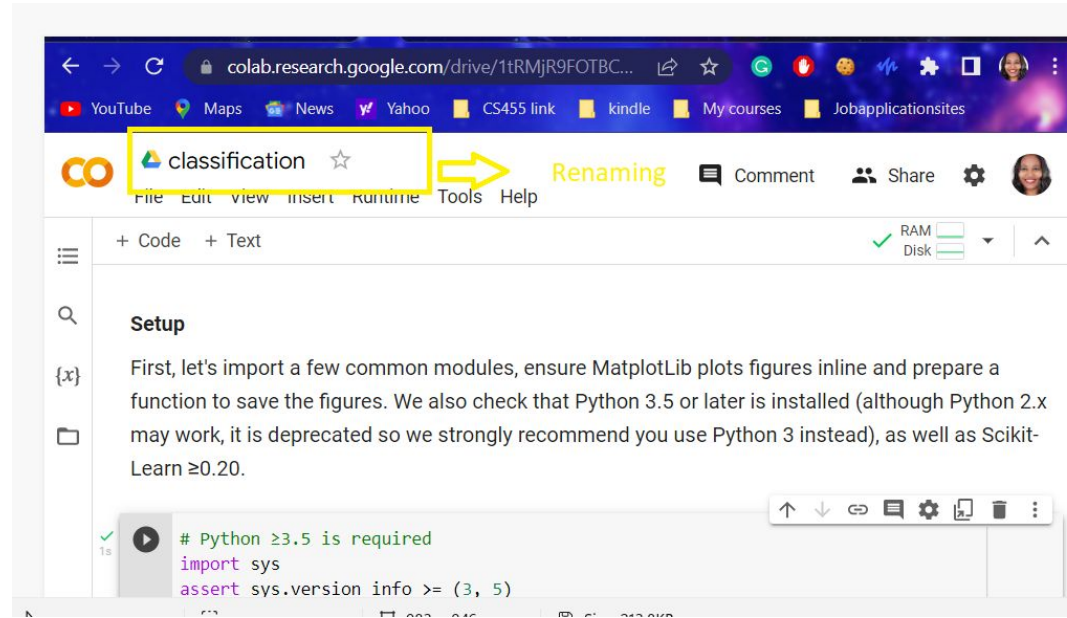
Libraries



```
import sys
import sklearn
import numpy as np
import os
import matplotlib as mpl
import matplotlib.pyplot as plt
```

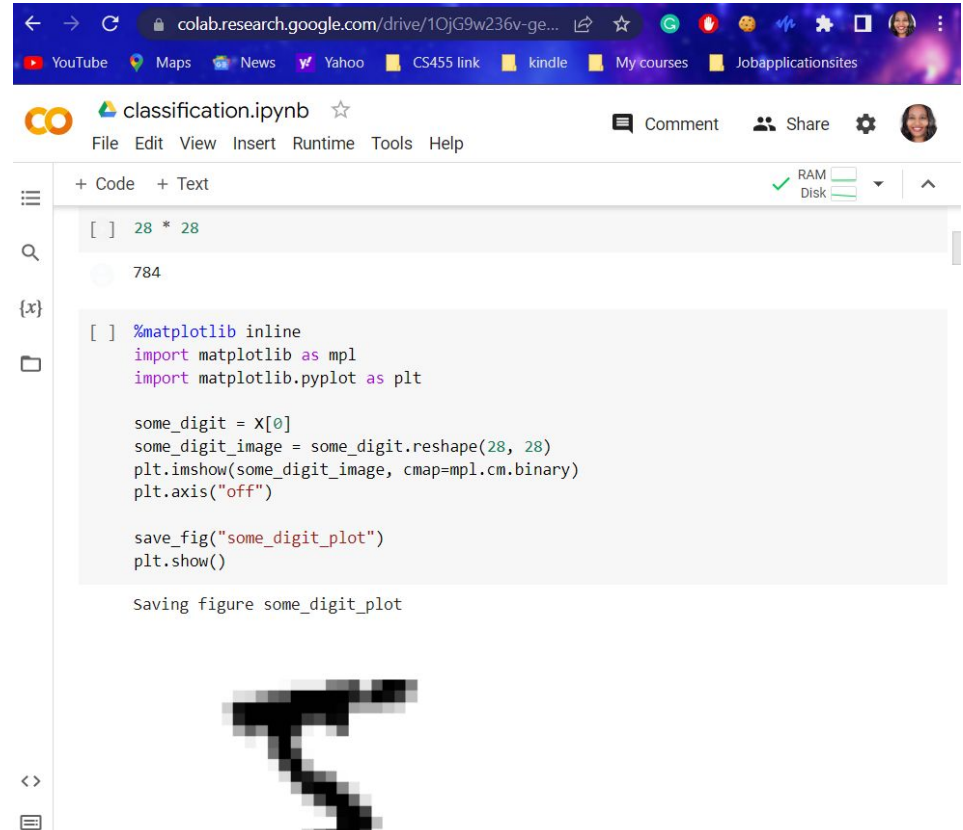
Implementation (Using Colab)

As a sample implementation we have the screenshot of the code running in colab



Implementation (Using Colab)

Running the code



The screenshot shows a Google Colab notebook titled "classification.ipynb". The interface includes a top navigation bar with various icons and a sidebar on the left with icons for file management and search. The main area displays a code cell with the following Python code:

```
[ ] 28 * 28

784

[ ] %matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt

some_digit = X[0]
some_digit_image = some_digit.reshape(28, 28)
plt.imshow(some_digit_image, cmap=mpl.cm.binary)
plt.axis("off")

save_fig("some_digit_plot")
plt.show()

Saving figure some_digit_plot
```

Below the code, a small, pixelated image of a handwritten digit '7' is visible, which is the output of the plot command in the code.

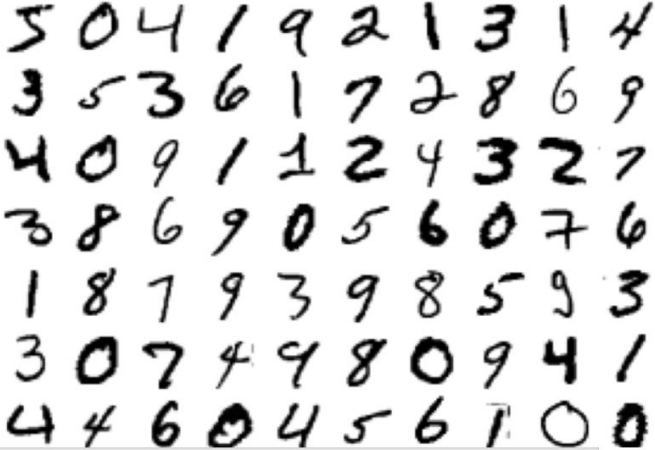
Implementation (Using Colab)

colab.research.google.com/drive/1OjG9w236v-ge...

classification.ipynb

```
plt.figure(figsize=(9,9))
example_images = X[:100]
plot_digits(example_images, images_per_row=10)
save_fig("more_digits_plot")
plt.show()
```

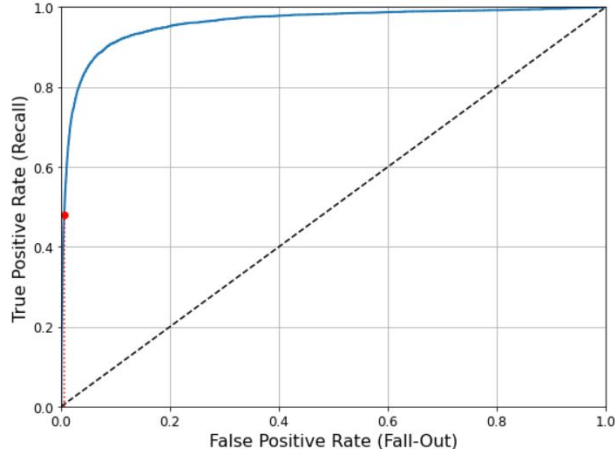
Saving figure more_digits_plot



classification.ipynb

```
plt.plot([fpr_90], [recall_90_precision], "ro")
save_fig("roc_curve_plot")
plt.show()
```

Saving figure roc_curve_plot





Conclusion

In conclusion, Colab is a powerful tool for performing classification on large datasets like MNIST. The scikit-learn library provides easy-to-use tools for loading and splitting the data, as well as implementing and evaluating machine learning models like KNN. With these tools, it's easy to get started with machine learning on Colab and start exploring the power of this technology.



References

- Brownlee, J. (2020, August 19). 4 types of classification tasks in machine learning. MachineLearningMastery.com. Retrieved February 14, 2023, from <https://machinelearningmastery.com/types-of-classification-in-machine-learning/>
- Google. (n.d.). Google colaboratory. Google Colab. Retrieved February 14, 2023, from https://colab.research.google.com/github/ageron/handson-ml2/blob/master/03_classification.ipynb