Project Description – Machine Learning with Python

DIGIT RECOGNITION WITH SCIKIT-LEARN AND MNIST

## Ana Carolina Souza Dias (AnaCarolinaDias)

Alexandre Guye-Bergeret (AlexG)

Armin Handanagic (scriptxbaby)

Janek Mattheus (Muggy)

Michael Sigg (Michi98)

**Table of content**

[**1.** **Introduction** 1](#_Toc107066111)

[**2.** **Data preparation** 2](#_Toc107066112)

[i. Data set: 2](#_Toc107066113)

[ii. Data exploration: 2](#_Toc107066114)

[**3.** **Prediction model** 5](#_Toc107066115)

[i. Logistic regression model: 5](#_Toc107066116)

[ii. Random Forest model: 6](#_Toc107066117)

[iii. Neural network classifier: 6](#_Toc107066118)

[**4.** **Model evaluation** 7](#_Toc107066119)

[i. The package import: 7](#_Toc107066120)

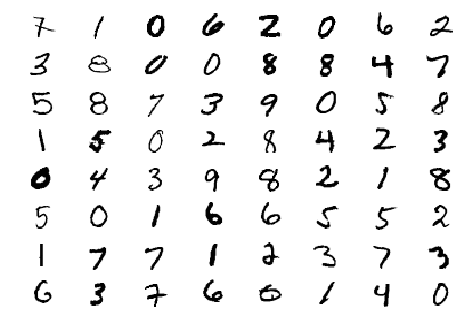
[ii. K-Fold Cross Validation model evaluation: 7](#_Toc107066121)

[**5.** **Model Performance** 8](#_Toc107066122)

[**6.** **Conclusion** 10](#_Toc107066123)

1. **Introduction**

In this Project, we’re going to create a machine learning model to predict the content of images on the MNIST handwritten digits dataset. The goal for all the model is to take an input image (28x28 pixels) of a handwritten single digit (0–9) and classify the image as the appropriate digit.

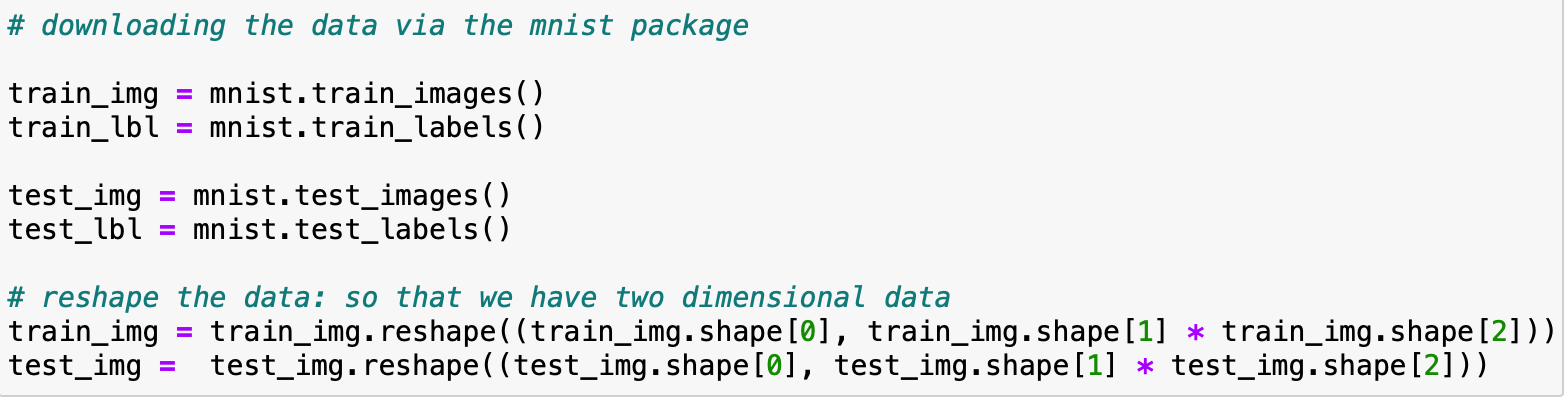


1. **Data preparation**

As a first step the appropriate libraries have been loaded, to handle the data we used pandas and to do mathematics operations numpy.

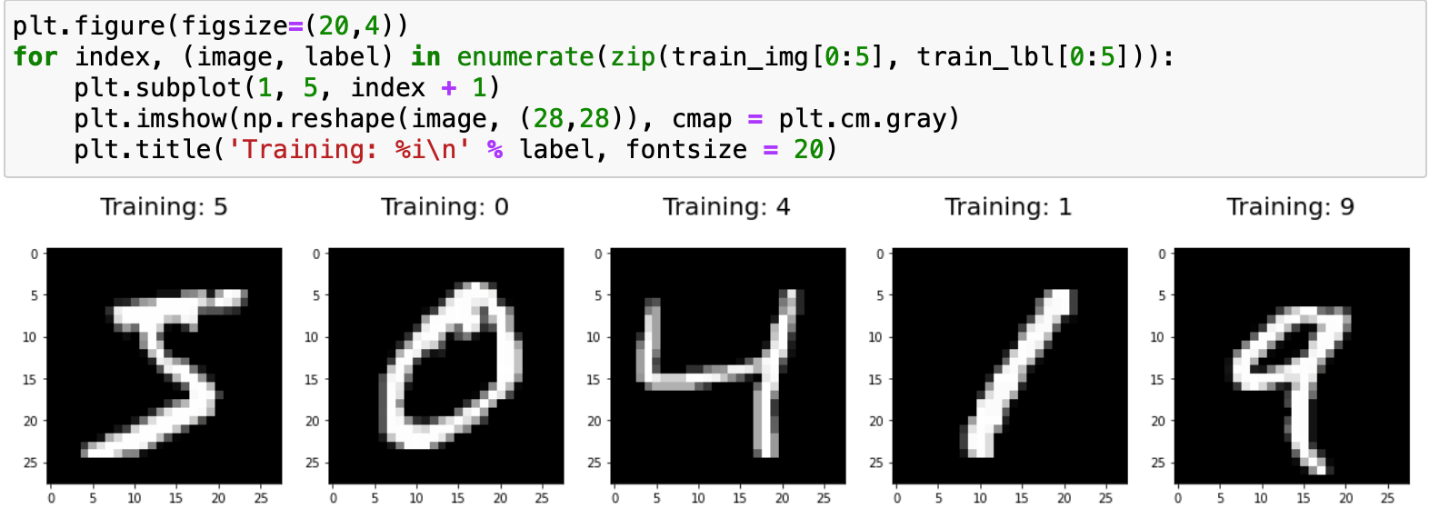
1. Data set:

The MNIST data set was downloaded from the mnist library:

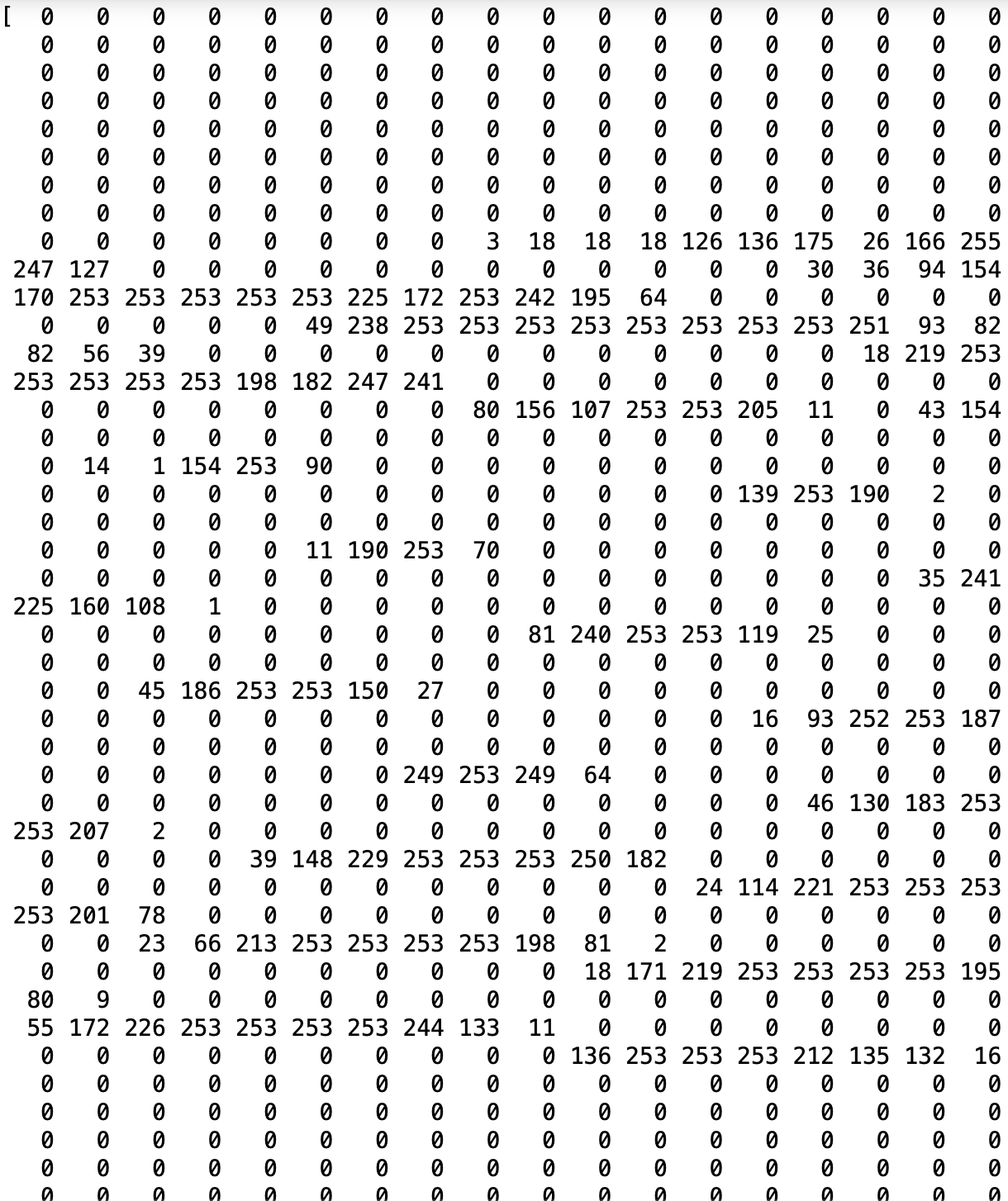


1. Data exploration:

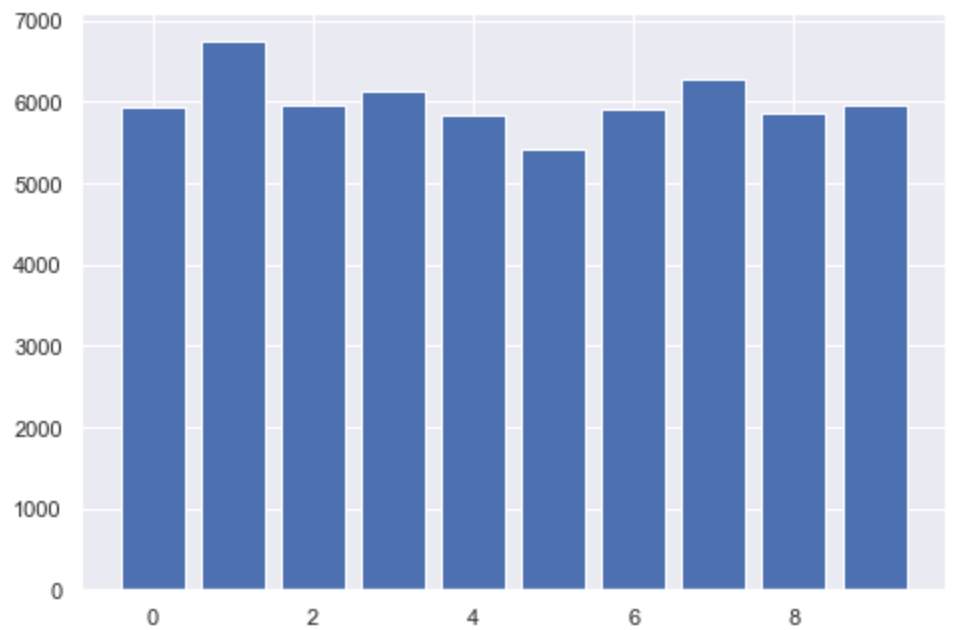
* The data representation as an image:



* The tabular representation of the number 5 in the data set:



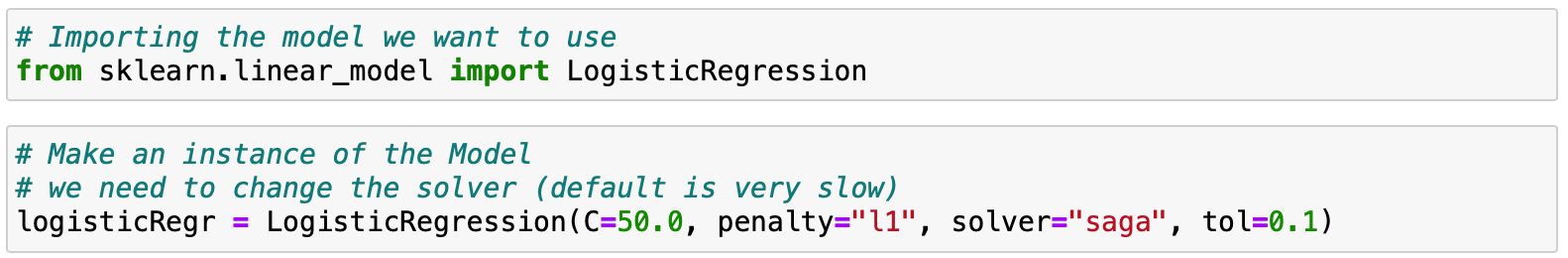
* data labels balance check:



The above chart shows that all the labels from 0 to 9 are balanced, which indicates that the prediction model will have solid ground and equal odds for the prediction of all the labels.

1. **Prediction model**
2. Logistic regression model:

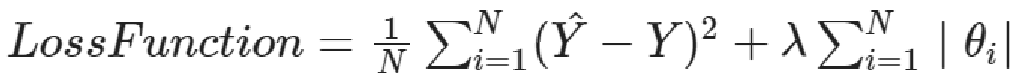
* Logistic regression estimates the probability of an event occurring, such as voted or didn’t vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds, or the natural logarithm of odds, and this logistic function is represented by the following formulas:
* the package used in this project for logistic regression was imported from sklearn, as shown in the code, which holds the instance to feed a logistic model from given data and create a custom appropriate model as shown in the code:



* The hyper parameters used for this model are:

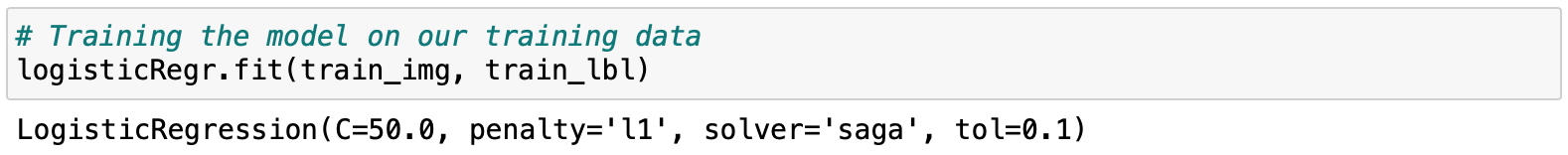
1. L1 regularization, also known as L1 norm or Lasso (in regression problems), combats overfitting by shrinking the parameters towards 0. This makes some features obsolete.

Essentially, when we use L1 regularization, we are penalizing the absolute value of the weights.



1. The SAGA solver is a variant of SAG that also supports the non-smooth penalty L1 option (i.e., L1 Regularization). This is therefore the solver of choice for sparse multinomial logistic regression and it’s also suitable for very large dataset.
2. Tolerance for stopping criteria = 0.1.

* Model training with the training set:



* The logistic model score after testing the test set is 92.4%

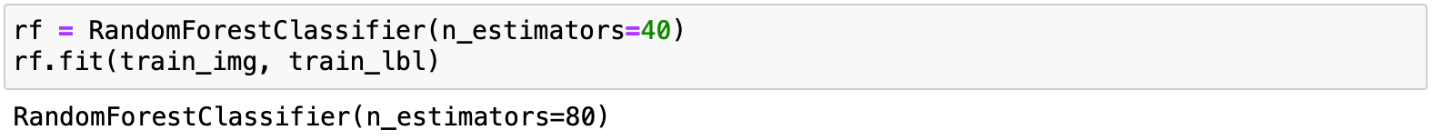


1. Random Forest model:

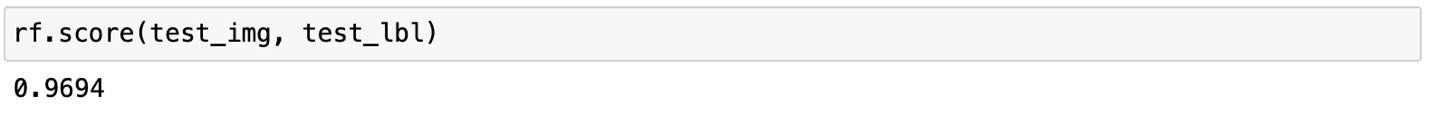
* The package required for this model was imported from sklearn ensemble library:



* The number of the trees in the forest was set to 40:



* The Random Forest model scored 96.94% after testing the test set of the data:

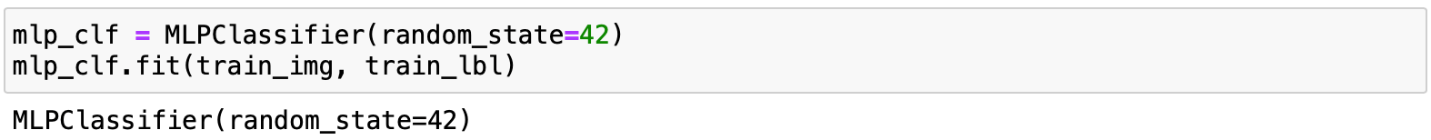


1. Neural network classifier:

* The package used is from the sklearn.neural\_network library:



* The hyper parameter remained as default while creating the model:



* The model scored 95.93% after testing the test data:



1. **Model evaluation**

K-fold cross-validation is one of the most commonly used model evaluation methods. Even though this is not as popular as the validation set approach, it can give us a better insight into our data and model.

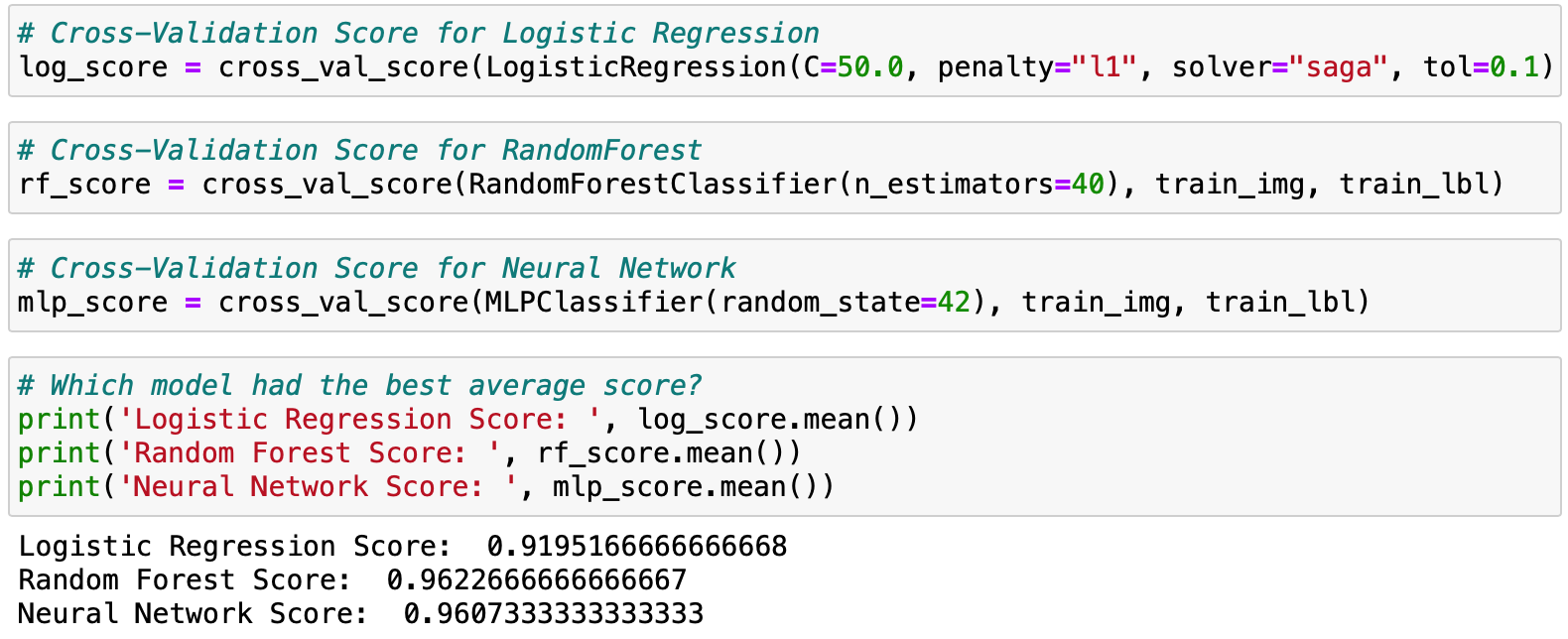
It works by splitting the data into subsets, where over multiple epochs each subset will once be used as a training set and the others as the testing sets. It is an easy way to compare how our models perform on the data.

1. The package import:



The package was imported from sklearn library which creates an instance that holds the model and performs the calculations for the accuracy score.

1. K-Fold Cross Validation model evaluation:

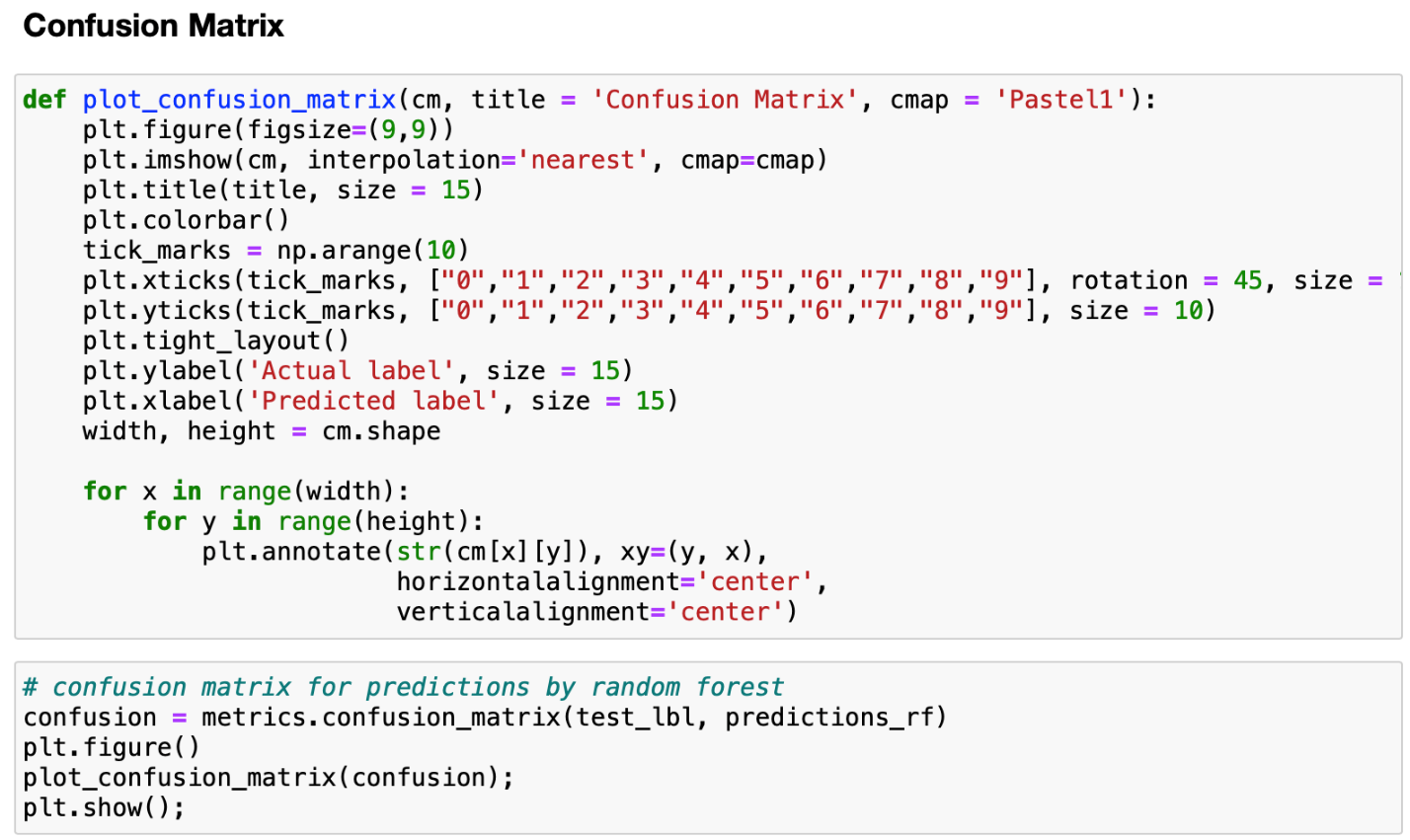


As can be seen above, using a 5-Fold Cross Validation, the Random Forest Classifier achieves a slightly higher accuracy than our neural network classifier and is significantly higher than a Logistic Regression. Therefore, the Random Forest model is best suited to predict the digits in our data and the rest of our project will focus on it.

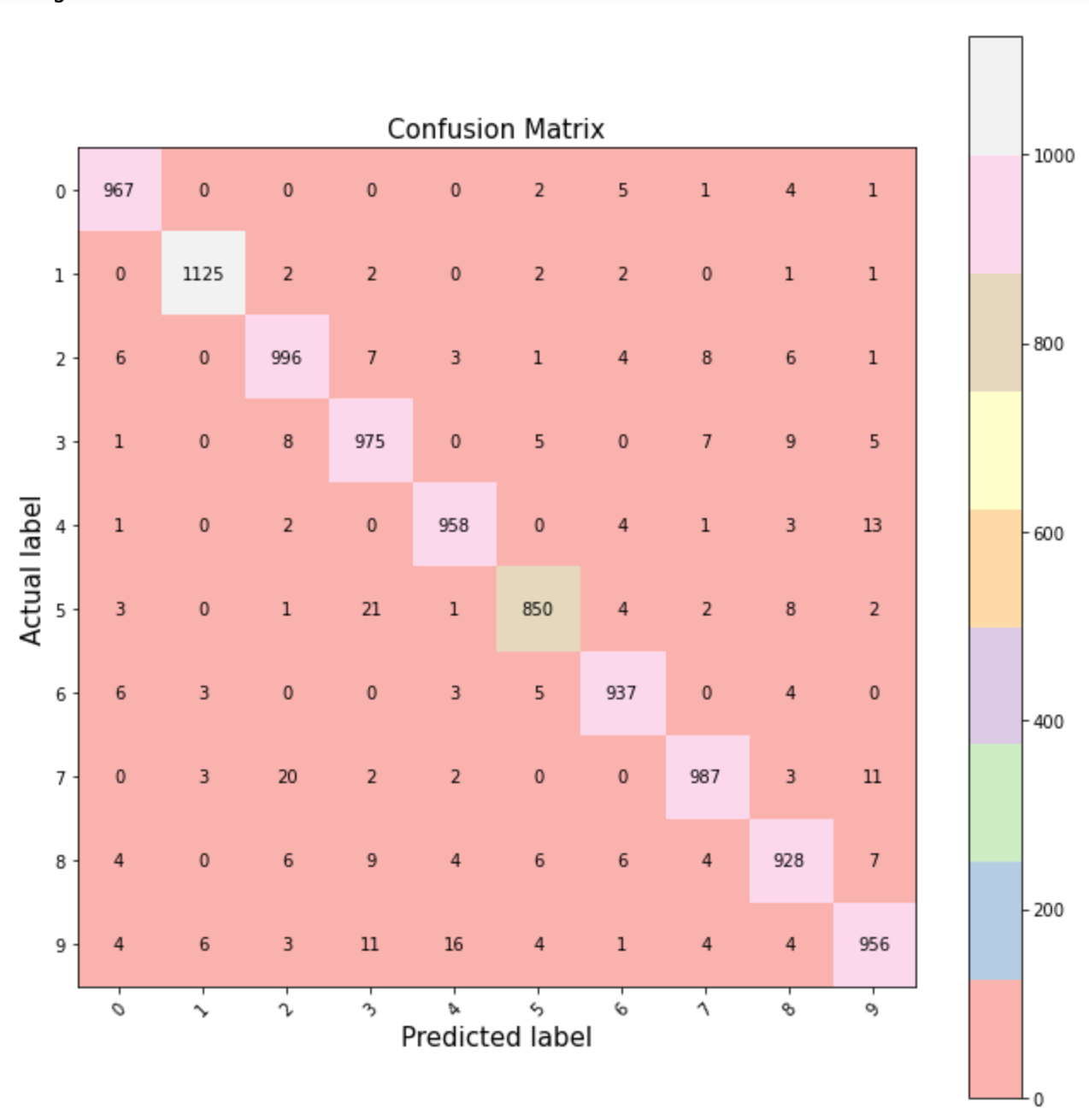
1. **Model Performance**

The model selected for performance measuring is the Random Forest Model, since it has the highest accuracy score against the other models.

The performance will be represented by a confusion matrix of the test label compared with the predicted labels:



* Results:



1. **Conclusion**

The Random Forest Classifier represents the appropriate model for the prediction of the MNIST data set with an accuracy of 96.22%. During the evaluation of the model performance some misclassified images are displayed (as seen below), which often hold some writing mistakes that lead them to be wrongly classified.

* Display some misclassified images with predicted labels:

