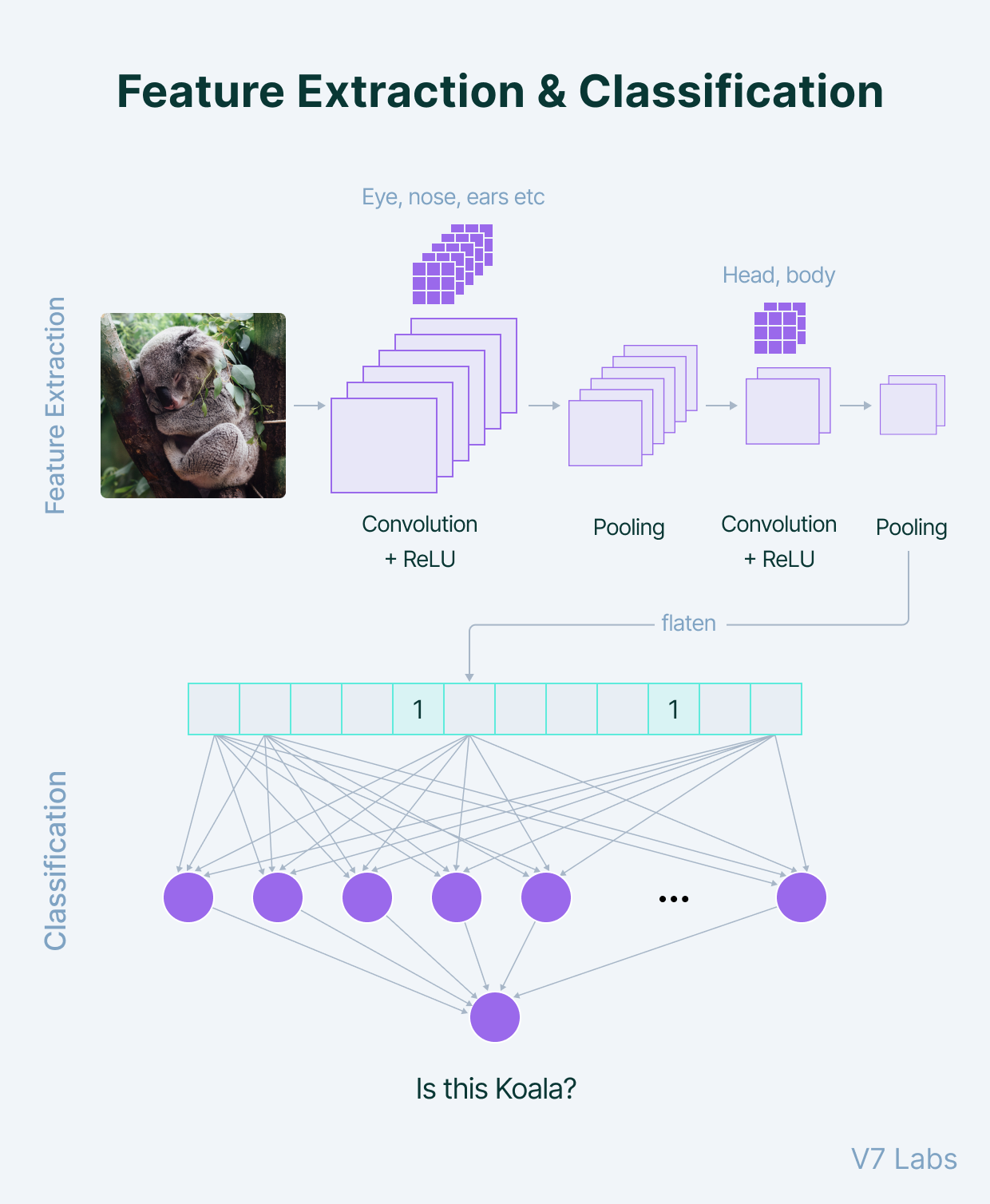
Multi Class Image Classification using CNN

* Define and explain the neural network architecture which you want to implement.

In this project I will be performing the multi class image classification using Convolutional Neural Networks.

Convolutional neural networks are a subclass of feed-forward neural networks used for complicated picture classification challenges as well as image analysis and natural language processing.

In CNN, features refer to minute details in the image data like edges, borders, shapes, textures, objects, circles, etc. Convolutional layers use filters to discover these patterns at a higher level in the visual data. The first several convolutional layers handle the most intricate information. The pattern searching becomes more complex as the network depth increases.



A convolutional layer's number of filters must be specified when it is added to a network. We may think of a filter as a relatively tiny matrix for which we choose the number of rows and columns. Random integers are used to initialise the value of this feature matrix. The filter will convolve across each patch of the input matrix as soon as this convolutional layer gets pixel values from the input data.

In order to add nonlinearity to the model, the output of the convolutional layer is often sent via the ReLU activation function. It uses the feature map and substitutes zero for each negative value. ConvNets need pooling, which accelerates computation and increases the model's resilience to distortions and fluctuations. A flattened feature matrix would be used to generate predictions by a fully connected dense neural network.

* Define and explain the activation functions (middle layer, last layer), loss function, learning rate, epoch, optimization algorithm etc. which will be used.

**Sequential Model**

A Sequential model is appropriate for **a plain stack of layers** where each layer has **exactly one input tensor and one output tensor**. Layer ordering or sequencing inside a model is the focus of the keras sequential algorithm. It essentially makes neural network layers compatible with the Keras API or Keras library for seamless functionality

**Relu activation function**

The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.

Applies the rectified linear unit activation function.

With default values, this returns the standard ReLU activation: max(x, 0), the element-wise maximum of 0 and the input tensor.

Modifying default parameters allows you to use non-zero thresholds, change the max value of the activation, and to use a non-zero multiple of the input for values below the threshold.

**Softmax activation function**

**Softmax** is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector.

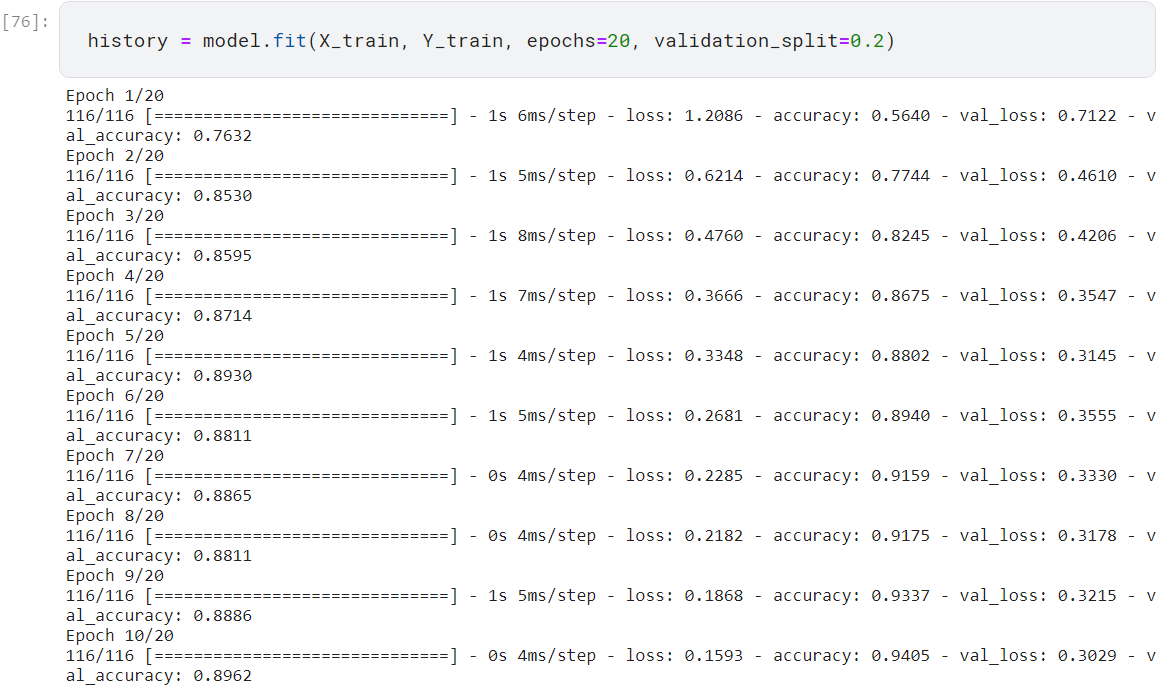
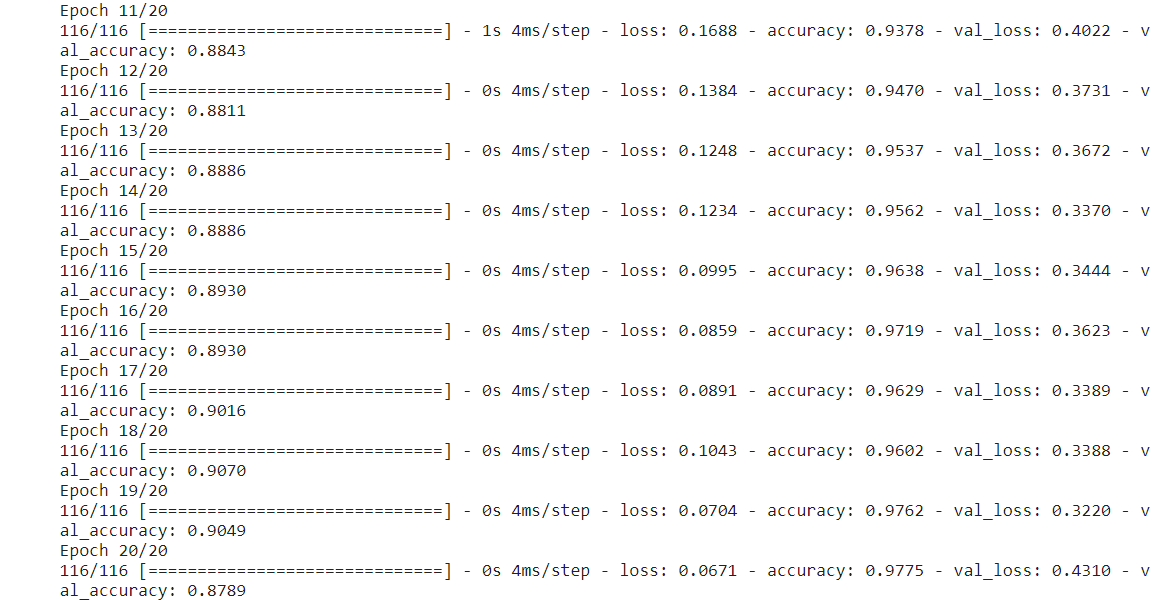
**Adam Optimizer**

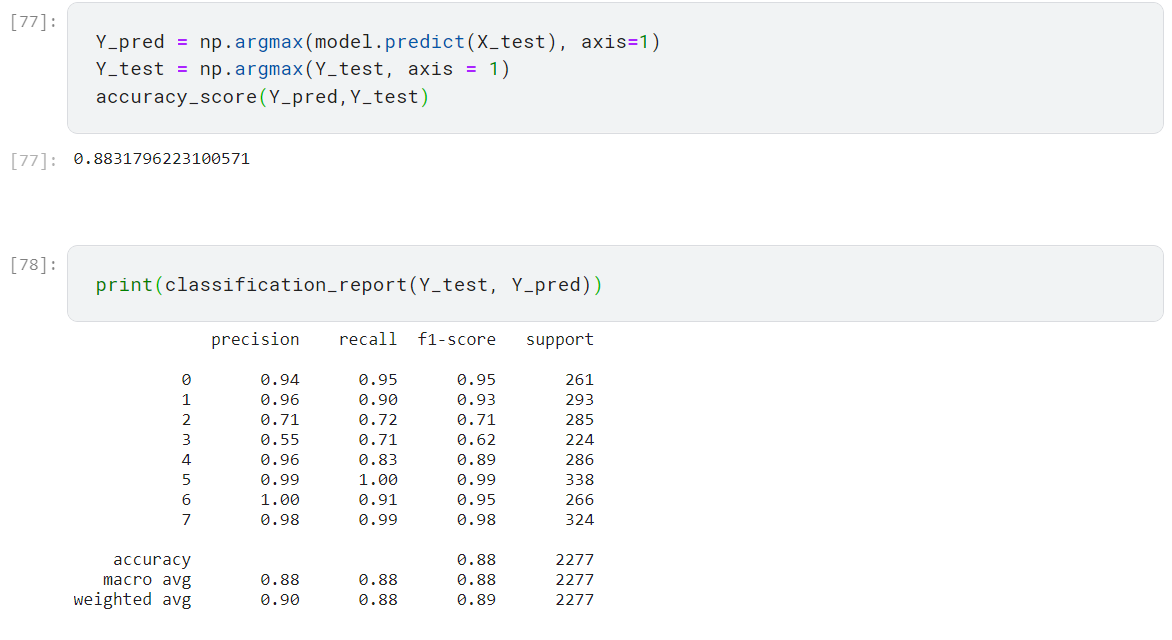
Adam optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments.

When compared to the other optimizers, Adam outperforms them by a significant margin for a better-optimized gradient

**Categorical Cross entropy loss function**

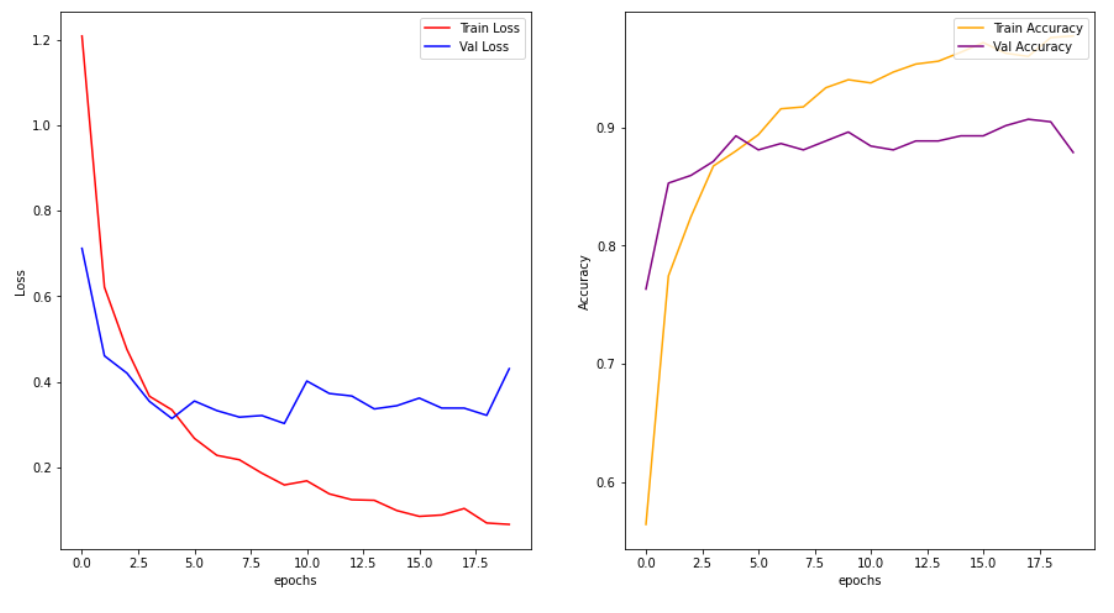
It is used as a loss function for multi-class classification model where there are two or more output labels. The output label is assigned one-hot category encoding value in form of 0s and 1. The output label, if present in integer form, is converted into categorical encoding using keras.utils to\_categorical method.

* Classify the image dataset in different class.
* Demonstrate the results with various learning rates &amp; epoch.
* Demonstrate the results with all possible evaluation criteria.



* Demonstrate the learning, results in all possible visualizing ways.





**Implementation Steps:**

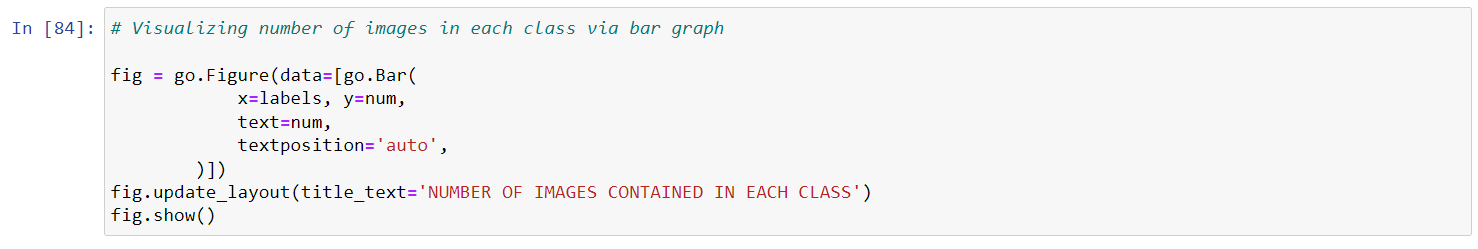
Importing Libraries:



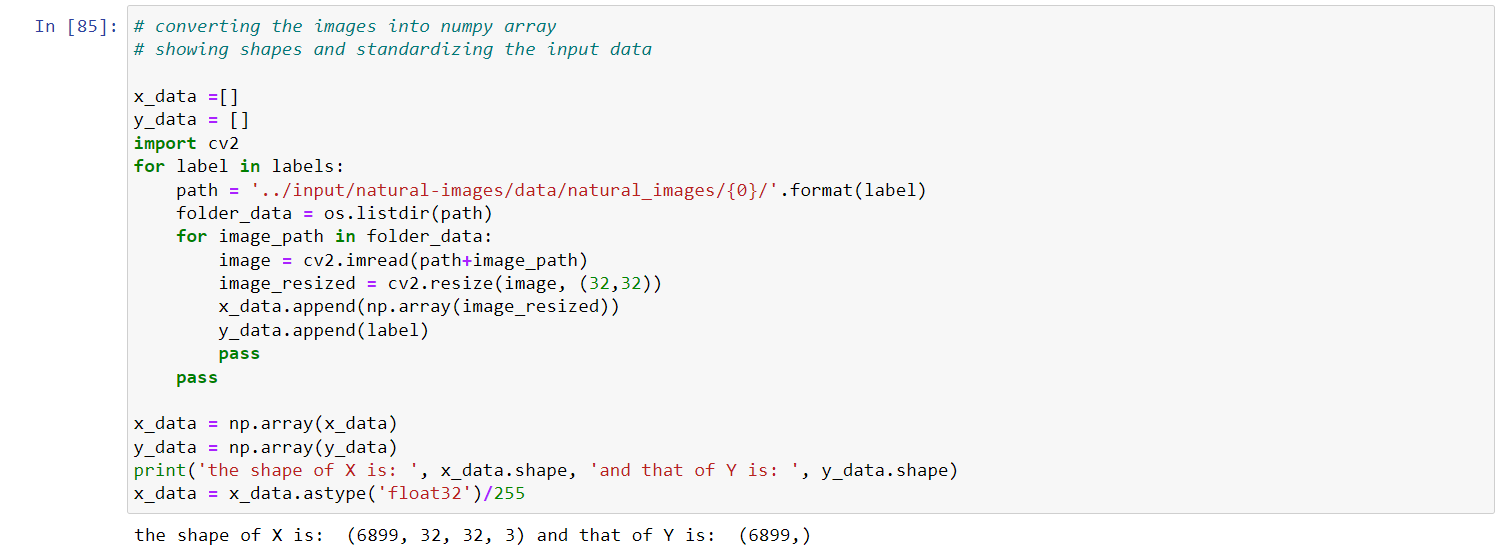
Setting up path directory:



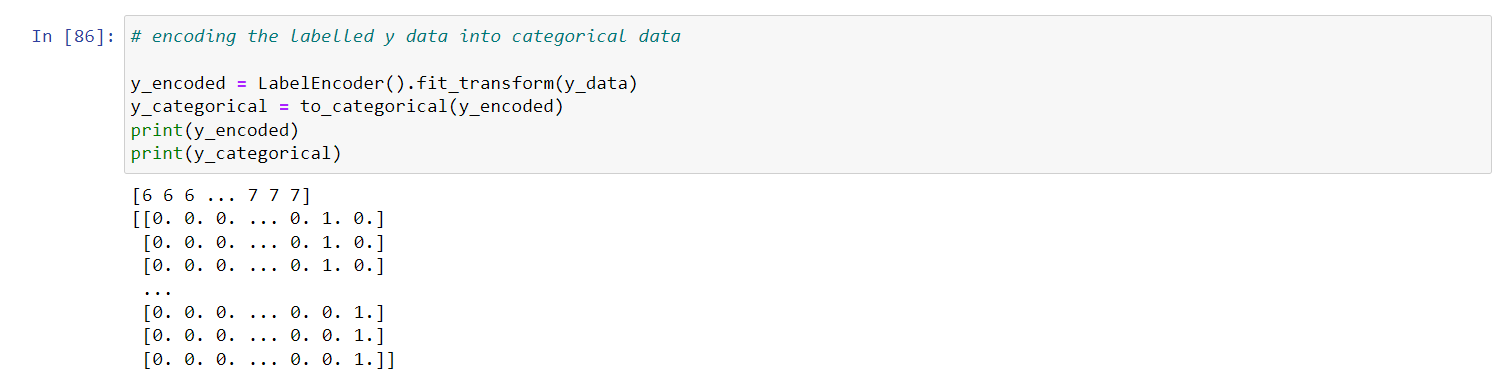
Visualizing input data:



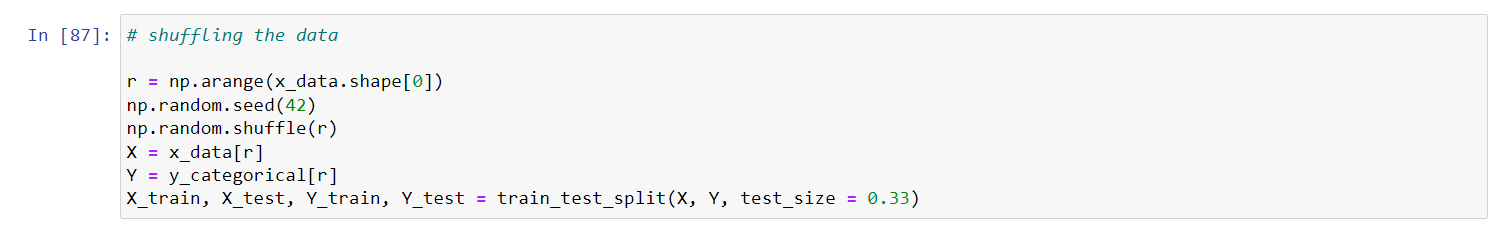
Normalization, Standardization and Conversion to Numpy Array:



Encoding into Categorical Data:



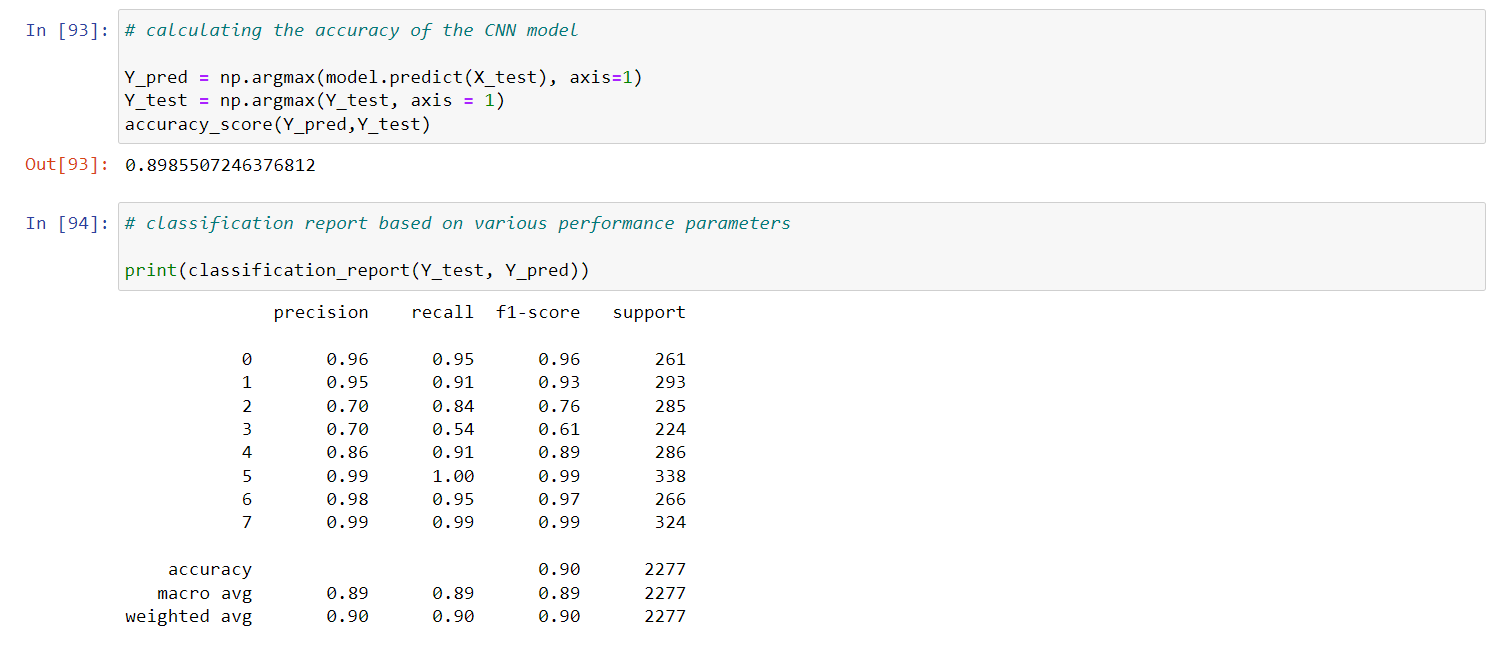
Shuffling the encoded data:



Creating, Compiling and fitting the model:



Evaluating the performance and accuracy:



Visualizing the Results:



**LINK OF CODE:**

[**https://drive.google.com/file/d/1VCfERARGQXC4lhKvgWGzDOg5KyDFNhuf/view?usp=share\_link**](https://drive.google.com/file/d/1VCfERARGQXC4lhKvgWGzDOg5KyDFNhuf/view?usp=share_link)