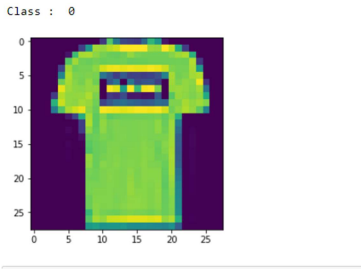
Analysis Report:

The image processing problem I selected was to classification of images in 10 classes such as T-shirt/top, Pullover, Bag etc. I selected this problem it is little more complicated to achieve perfect classification than Mnist handwritten digits dataset. Implemented using CNN.

- **Dataset and processing**

The dataset I selected to demonstrate Convolution Neural Network (CNN) is Fashion-MNIST. This contains 28x28 grayscale image, associated with a label from 10 classes. Dataset is divided into training and testing. Training set contains 60,000 examples of images and testing set contains 10,000 examples.



This homework is implemented using Keras library. Keras includes fashion_mnist dataset in its inbuilt library and we have to just import that in our code.

I referred some online tutorials for building CNN with details of step by step explanation. Mainly I referred CNN tutorial from Machine Learning Mastery website. This provided processing on MNIST handwritten dataset whereas I implemented Fashion MNIST.

I started implementation with loading of data and dividing into training and testing datasets. Then performed some of preprocessing like reshaping of input data to make it suitable for training in Keras, Normalizing of pixel value to value between 0 and 1. Then I used one hot encoding of output values. Here I experimented different ways of doing this processings and visualizing of shapes and values in Ipython notebook. I displayed some of training data images to get idea of how data looks like. Then after performing processing observed changes in data.

- **CNN Model**

Then I started to build CNN using sequential model. I added Conv2D, MaxPooling2D and Dropout layers for 2 times. And then added flatten and fully connected layers. So initial model looks like below-

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 12, 12, 32)        4128
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 32)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               147584
_________________________________________________________________
dropout_3 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 153,322
Trainable params: 153,322
Non-trainable params: 0
```

Then I compiled model using “adam” optimizer.
Then I trained this model using batch size of 128 for 10 epochs. After evaluation I got accuracy of 90.71%.

To gain best optimized results I performed hyper parameter tuning and performed multiple combinations and evaluations to get best performed result. The best model I came up with as follows-

In this model I increased feature maps from 32 to 64 in the first convolution layer. Also reduced its size from (3 x 3) to (2 x 2) to capture more details. Also increased dropout after second fully connected network from 0.25 to 0.30.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 27, 27, 64)        320
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 13, 13, 64)        0
_________________________________________________________________
dropout_4 (Dropout)          (None, 13, 13, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 12, 12, 32)        8224
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
dropout_5 (Dropout)          (None, 6, 6, 32)          0
_________________________________________________________________
flatten_2 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               147584
_________________________________________________________________
dropout_6 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 10)                1290
=================================================================
Total params: 157,418
Trainable params: 157,418
Non-trainable params: 0
```

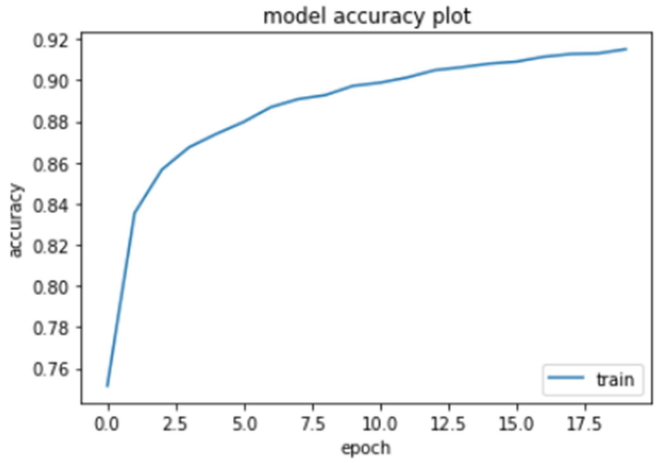
I also trained this model with 20 epochs and batch_size of 128. This model took considerable amount of more time to train than earlier model.

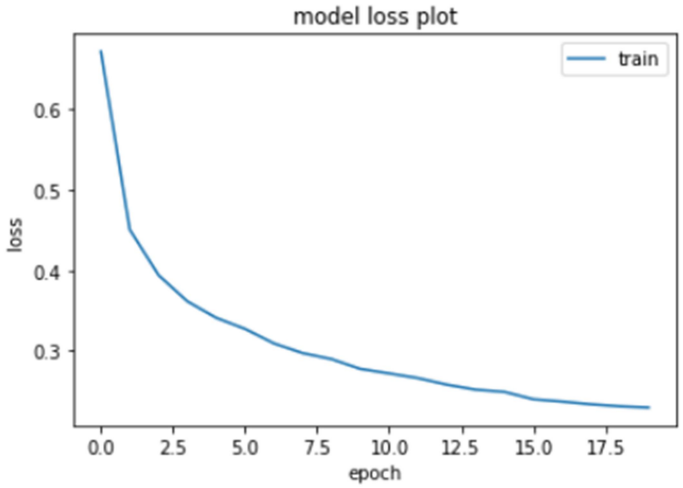
After evaluation I got accuracy of 91.37% and following results-

```
10000/10000 [==============================] - 2s 247us/step

acc: 91.37%

loss: 23.40%
```





So from this result it can be observed that accuracy is increased over increase in the number of epochs. Also increase in feature maps and capturing more details also trained model to more correctly classify images. Dropout also plays an important role in reduce of overfitting of data.