A MNIST-like fashion product database

In this, we classify the images into respective classes given in the dataset. We use a Neural Net ar and check the accuracy scores.

Load tensorflow

```
import tensorflow as tf
tf.set_random_seed(42)
```

```
tf.__version__
```

```
'1.15.0'
```

Collect Data

```
import keras
```

```
(trainX, trainY), (testX, testY) = keras.datasets.fashion_mnist.load_data()
```

```
print(testY[0:5])
```

Your session crashed. Automatically restarting.

Convert both training and testing labels into one-hot vectors.

Hint: check **tf.keras.utils.to_categorical()**

```
from keras.utils import to_categorical

# one hot encode
trainX_encoded =  tf.keras.utils.to_categorical(trainX)
print(trainX_encoded)
```

```
# one hot encode
testX_encoded = to_categorical(testX)
print(testX_encoded)
```

```
print(trainY.shape)
print('First 5 examples now are: ', trainY[0:5])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-e3383ffab5c1> in <module>()
----> 1 print(trainY.shape)
      2 print('First 5 examples now are: ', trainY[0:5])

NameError: name 'trainY' is not defined
```

SEARCH STACK OVERFLOW

Visualize the data

Plot first 10 images in the triaining set and their labels.

```
plt.figure(figsize=(10,10))
for i in range(10):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(trainX_encoded[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[testX_encoded[i]])
plt.show()
```

Your session crashed. Automatically restarting.

```
9 0 0 3 0 2 7 2 5 5
```

Build a neural Network with a cross entropy loss function and sgd optimizer in neurons as we have 10 classes.

```
#Initialize Sequential model
model = tf.keras.models.Sequential()

#Add Dense Layer which provides 3 Outputs after applying softmax
model.add(tf.keras.layers.Dense(10, input_shape=(28,28), activation='softmax'))

#Comile the model
model.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])
```

Execute the model using model.fit()

```
model.fit(trainX_encoded, trainY_encoded, epochs=10)
```

In the above Neural Network model add Batch Normalization layer after the inp

```
#Normalize the data
model.add(tf.keras.layers.BatchNormalization())

#Add Dense Layer which provides 3 Outputs after applying softmax
model.add(tf.keras.layers.Dense(10, input_shape=(28,28), activation='softmax'))

#Comile the model
model.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])

model.fit(trainX_encoded, trainY_encoded, epochs=10)
```

Execute the model

```
pred = model.predict(testX)
pred
```

Customize the learning rate to 0.001 in sgd optimizer and run the model

```
from keras.optimizers import SGD

opt = SGD(lr=0.001, momentum=0.9)
```

Your session crashed. Automatically restarting.

```
model.compile(optimizer= opt, loss= 'categorical_crossentropy', metrics=['accuracy'])
```

Build the Neural Network model with 3 Dense layers with 100,100,10 neurons r entropy loss function and singmoid as activation in the hidden layers and softr output layer. Use sgd optimizer with learning rate 0.03.

```
model = Sequential()
model.add(Dense(5, input_dim=2))
model.add(Activation('relu'))
model.add(Dense(1))
model.add(Activation('sigmoid'))

#Initialize Sequential model
model = tf.keras.models.Sequential()

#Add Dense Layer which provides 3 Outputs after applying softmax
model.add(tf.keras.layers.Dense(100, input_shape=(100,), activation='softmax'))

model.add(Activation('sigmoid'))
```

```
model.add(Activation( sigmoid ))

model.add(tf.keras.layers.Dense(100, input_shape=(100,), activation='softmax'))

model.add(Activation('sigmoid'))

model.add(tf.keras.layers.Dense(10, input_shape=(10,), activation='softmax'))



from keras.optimizers import SGD

opt = SGD(lr=0.03, momentum=0.9)

#Comile the model
model.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])
```

Review model

```
model.fit(trainX_encoded, trainY_encoded, epochs=10)
```

Run the model

Your session crashed. Automatically restarting.

```
pred
```