FASHION IMAGE CLASSIFICATION

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Mantravadi Nikhita, 221910302031

Vora Sreeja, 221910302056

Jayanth Phani, 221910302044

Under the esteemed guidance of

MRS. G. Karthika

Assistant Professor

CSE Dept.

DEPARTMENT OFCOMPUTER SCIENCE & ENGINEERING

GITAM

(Deemed to be University)

HYDERABAD

November 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY GITAM

(Deemed to be University)

DECLARATION

I/We, hereby declare that the project report entitled "**FASHION IMAGE CLASSIFICATION**" is an original work done in the Department of Computer Science and Engineering, GITAM Institute of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 07 /11/ 2022

Registration No(s).	**Name(s)**	**Signature(s)**
221910302031	**Mantravadi Nikhita**	
221910302056	**Vora Sreeja**	
221910302044	**Jayanth Phan**	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled "**FASHION IMAGE CLASSIFICATION**" is a bonafide record of work carried out by **Mantravadi Nikhita(221910302031), Vora Sreeja(221910302056) and Jayanth Phani(221910302044)** students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide

MRS. G. Karthika

Assitant Professor

CSE Dept

Head of the Department

DR. S. Phani Kumar

Professor

CSE Dep

TABLE OF CONTENTS

1. ABSTRACT

Fashion is the way we present ourselves which mainly focuses on vision, has attracted great interest from computer vision researchers. Fashion knowledge encourages people to properly dress and faces not only physiological necessity of users, but also the requirement of social practices and activities. It usually includes three jointly related aspects of: occasion, person and clothing. Nowadays, social media platforms allow users to interact with each other online to share opinions and information. The main objectives of the paper are to use deep learning (DL) and machine learning (ML) methods to correctly identify and categorize clothing images. In this work, we used ML, DL and the transfer learning using a pretrained model. The main metric used in this study to evaluate the performance of ML and DL algorithms is the accuracy and matrix confusion. The use of social media sites such as Instagram has already spread to almost every fashion brand and been evaluated as business take-off tools. With the heightened use of social media as a means of marketing communication for fashion brands, it has become necessary to empirically analyze and extract fashion knowledge from them. Thus, social brands are investing on them. In this way, they can understand the consumer's preferences. This change is also having a significant impact on social media data analysis. To solve this issue, the Deep learning (DL) and Machine Learning (ML) methods are proven to be effective solutions due to their automatic learning capability. Machine Learning and Deep Learning techniques can be used to identify an item of clothing which could further identify the item on social media which could give us a lot of information for future use.

2. INTRODUCTION

The need for powerful image analysis tools has become a necessity today, especially with images appearing on the internet and being used many times in place of text. Recently, great progress has been made in the field of image recognition and classification, and these developments are attributed to the availability of international databases as well as the development of artificial intelligence in many magazines including the fashion. Our society is becoming more and more intelligent at present, in particular in the field of artificial intelligence and especially the field of image recognition which has undergone a major evolution since the appearance of deep learning.

The fashion is how we present ourselves to the world and it has become one of the biggest industries in the world. Fashion which mainly focuses on vision, has attracted great interest from vision researchers by computer. It is generally used to search fashion products in online shopping malls to know the descriptive information of the product. Fashion companies can use a deep learning model that can help them to categorize clothes more accurately. Moreover, you can create your own online wardrobe according to your artificial lifestyle.

One of the machine learning subfields is deep learning. It has gained more traction in numerous areas and is more comprehensive than ML methods. Convolutional neural network (CNN) is the most widely used algorithm in deep learning. The use of first and second order statistical features, which must be manually extracted from a set of raw image data depending on the progression of convolutional layers, is no longer necessary thanks to deep learning (DL) techniques. There are various DL model algorithms, including AlexNet, GoogleNet, RestNet18, ResNet50, etc.

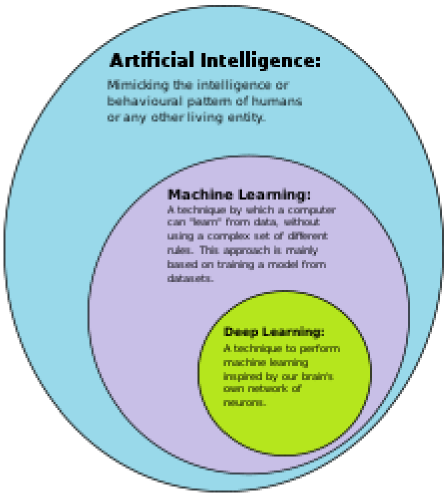

Artificial Intelligence:
Mimicking the intelligence or behavioural pattern of humans or any other living entity.
Machine Learning:
A technique by which a computer can "learn" from data, without using a complex set of different rules. This approach is mainly based on training a model from datasets.
Deep Learning:
A technique to perform machine learning inspired by our brain's own network of neurons.

Fig 1

A convolutional neural network (CNN/ConvNet) is a type of deep neural network that is commonly used to analyze visual imagery in deep learning. When we think of neural networks, we typically think of matrix multiplications, but this is not the case with ConvNet. It employs a technique known as Convolution. Convolution is a mathematical operation on two functions that yields a third function that expresses how the shape of one is modified by the other.

CNN's main advantage over its predecessors is that it automatically identifies relevant features without the need for human intervention. CNNs have been widely used in a variety of fields, including computer vision, speech processing, and face recognition. CNNs, like traditional neural networks, were inspired by neurons in human and animal brains. In particular, the visual cortex in a cat's brain is formed by a complex sequence of cells, which is simulated by the CNN.

CNNs are especially useful for detecting patterns in images in order to recognize objects, faces, and scenes. They are also useful for categorizing non-image data such as audio, time series, and signal data. CNNs are heavily used in applications that require object recognition and computer vision, such as self-driving vehicles and face recognition.

3. LITERATURE REVIEW:

3.1 INTRODUCTION

Recently, in 2022 there was a paper that was published with the Title, ‘Fashion Images Classification using Machine Learning, Deep Learning and Transfer Learning Models’, written by Bougareche Samia from the dept. of Electrical Engineering, Biskra University, Biskra, Algeria, Zehnani Soraya, dept. of Electrical Engineering, Biskra University, Biskra, Algeria and Mimi Malika, dept. of Electrical Engineering, Mostaganem University, Mostaganem, Algeria. According to them, Fashion is the way we present ourselves which mainly focuses on vision, has attracted great interest from computer vision researchers. It is generally used to search fashion products in online shopping malls to know the descriptive information of the product. The main objectives of their paper is to use deep learning (DL) and machine learning (ML) methods to correctly identify and categorize clothing images. In this work, they used ML algorithms (support vector machines (SVM), K-Nearest Neirghbors (KNN), Decision tree (DT), Random Forest (RF)), DL algorithms (Convolutionnal Neurals Network (CNN), AlexNet, GoogleNet, LeNet, LeNet5) and the transfer learning using a pretrained models (VGG16, MobileNet and RestNet50). They trained and tested our models online using google colaboratory with Tensorflow/Keras and Scikit-Learn libraries that support deep learning and machine learning in Python. The main metric used in their study to evaluate the performance of ML and DL algorithms is the accuracy and matrix confusion. The best result for the ML models is obtained with the use of ANN (88.71%) and for the DL models is obtained for the GoogleNet architecture (93.75%). The results obtained showed that the number of epochs and the depth of the network have an effect in obtaining the best results.

They also said that, the need for powerful image analysis tools has become a necessity today, especially with images appearing on the internet and being used many times in place of text. Recently, great progress has been made in the field of image recognition and classification, and these developments are attributed to the availability of international databases as well as the development of artificial intelligence in many magazines including the fashion. Our society is becoming more and more intelligent at present, in particular in the field of artificial intelligence and especially the field of image recognition which has undergone a major evolution since the appearance of deep learning.

They believe that, Garment image analysis is a topic that has gained increasing interest from computer vision companies in recent years. A growing number of articles focus on finding images in everyday life and on websites, a crucial task in the electronic fashion industry. Certainly, the classification of clothing styles is a recent topic in computer vision research and has many interesting applications, including e-commerce, criminal law, and online advertising.

The main objective of their paper was to evaluate the performance of the machines learning (ML) and the deep learning (DL) algorithms on a fashion-MNIST dataset. The types of clothes they consider in the proposed system include shirt, pants, suit, dress, etc. In their proposed approach, the classification was performed.

The results they obtained in their tests were as follows:

3.2 MACHINE LEARNING MODELS:

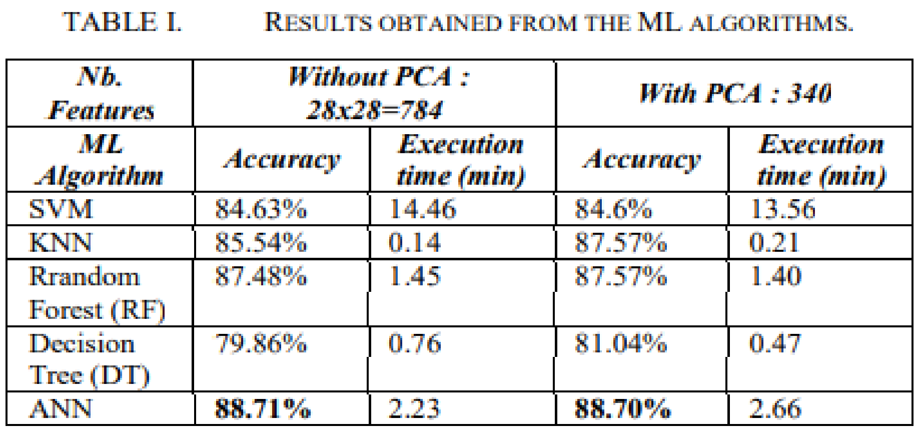


TABLE I. RESULTS OBTAINED FROM THE ML ALGORITHMS.

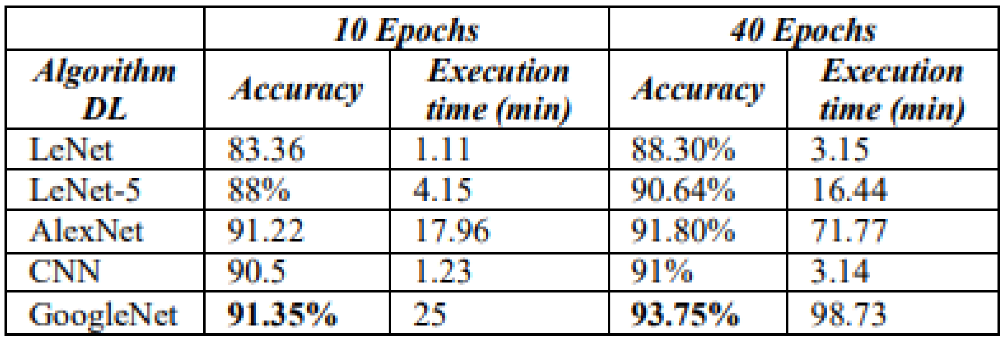
Nb. Features	*Without PCA : 28x28=784*		*With PCA : 340*	
ML Algorithm	***Accuracy***	***Execution time (min)***	***Accuracy***	***Execution time (min)***
SVM	84.63%	14.46	84.6%	13.56
KNN	85.54%	0.14	87.57%	0.21
Rrandom Forest (RF)	87.48%	1.45	87.57%	1.40
Decision Tree (DT)	79.86%	0.76	81.04%	0.47
ANN	**88.71%**	2.23	**88.70%**	2.66

Table 1

According to the Table 1, the best results from the ML algorithms is obtained from the ANN followed by the random forest (RF) then the KNN followed by the SVM and finally the decision tree (DT) algorithms

3.3 DEEP LEARNING MODELS:

TABLE II. RESULTS OBTAINED FROM THE DL MODELS.



	10 Epochs		*40 Epochs*	
Algorithm DL	*Accuracy*	*Execution time (min)*	*Accuracy*	*Execution time (min)*
LeNet	83.36	1.11	88.30%	3.15
LeNet-5	88%	4.15	90.64%	16.44
AlexNet	91.22	17.96	91.80%	71.77
CNN	90.5	1.23	91%	3.14
GoogleNet	**91.35%**	25	**93.75%**	98.73

Table 2

According to the Table 2, the best results from the DL models is obtained from the GoogleNet followed by the AlexNet then the CNN followed by the LeNet-5 and finally the LeNet algorithm. They used a variety of the number of iterations, we notice that the execution time depends on the number of iterations. They presented two cases for 10 and 40 iterations.

3.4 TRANSFER LEARNING:

They used three pre-trained models in their study: the VGG16, MobileNet and ResNet50. They presented two examples (bag and jean) for the application of the transfer learning using the pretrained models (VGG16, ResNet50, and MobileNet) to classify the garments images.

3.5 CONCLUSION:

They concluded their study by saying that their study is based on suggested an automated classifier to efficiently a large number of fashion garments into ten (10) specific categories based on three approaches: the classic ML approach, the DL approach and the transfer learning. They applied different types of ML algorithms (SVM, KNN, Random Forest (RF), Decision Tree (DT)), as well as DL models (CNN, GoogleNet, AlexNet, LeNet5, LeNet), then the Transfer Learning using the pre-trained models (RestNet50, VGG16 and MobileNet). They initially used several ML algorithms, with features reduction using principal component analysis (PCA) and without features reduction. We obtained an accuracy result in both cases between 79.86% and 88.71%. With the use of PCA we noticed an improvement in the results. Then they oriented towards the DL models, they obtained an accuracy result between 83.36% and 93.75%. For the transfer learning application, we get the best results for the ResNet50 model with an accuracy of 99,79%. From these results we notice that the use of DL models gives better results than the use of ML models. But the transfer learning gives better results than DL and ML models.

4. PROBLEM IDENTIFICATION AND OBJECTIVES

4.1 OBJECTIVE

Fashion is the way we present ourselves which mainly focuses on vision, has attracted great interest from computer vision researchers. It is generally used to search fashion products in online shopping malls to know the descriptive information of the product. Fashion knowledge encourages people to properly dress and faces not only physiological necessity of users, but also the requirement of social practices and activities. It usually includes three jointly related aspects of: occasion, person and clothing. Nowadays, social media platforms allow users to interact with each other online to share opinions and information. The dataset we used is the Fashion MNIST. The Fashion-MNIST clothing classification problem is a new standard dataset used in computer vision and deep learning.

Although the dataset is relatively simple, it can be used as the basis for learning and practicing how to develop, evaluate, and use deep convolutional neural networks for image classification from scratch. This includes how to develop a robust test harness for estimating the performance of the model, how to explore improvements to the model, and how to save the model and later load it to make predictions on new data.

4.2 TECHNOLOGIES USED

4.2.1 Python:

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

4.2.2 Google Colab:

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. Google Colab is a must for anyone looking to back their work up to the cloud and to sync their notebooks across multiple devices — but the ease of cloud sharing means reduced data security.

4.3.3 Google:

Google LLC is an American multinational technology company focusing on search engine technology, online advertising, cloud computing, computer software, quantum computing, e-commerce, artificial intelligence, and consumer electronics

4.4.4 Kaggle:

Kaggle, a subsidiary of Google LLC. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges

4.3 FIELD OF STUDY

The field of study for our project may come under: Data Science, Artificial Intelligence, Machine Learning, Deep Learning and Computer Vision

Data science combines math and statistics, specialized programming, advanced analytics, artificial intelligence (AI), and machine learning with specific subject matter expertise to uncover actionable insights hidden in an organization's data. These insights can be used to guide decision making and strategic planning.

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision. AI systems work by ingesting large amounts of labeled training data, analyzing the data for correlations and patterns, and using these patterns to make predictions about future states.

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It’s achieving results that were not possible before.

5. SYSTEM METHODOLOGIES

5.1 Process of Project Selection and Finalization

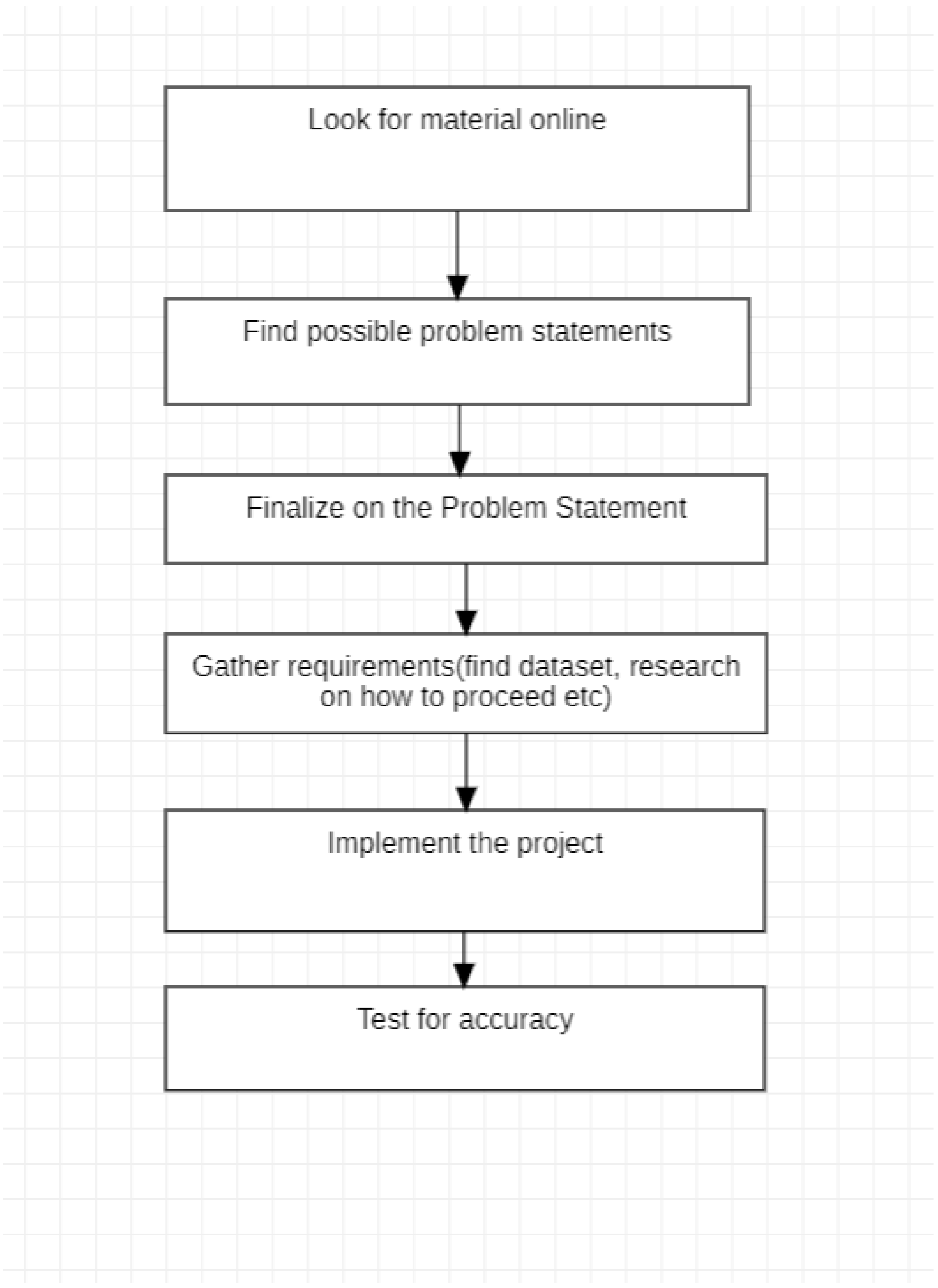

Look for material online
Find possible problem statements
Finalize on the Problem Statement
Gather requirements(find dataset, research on how to proceed etc)
Implement the project
Test for accuracy

Fig 2

5.2 Process of Project Implementation

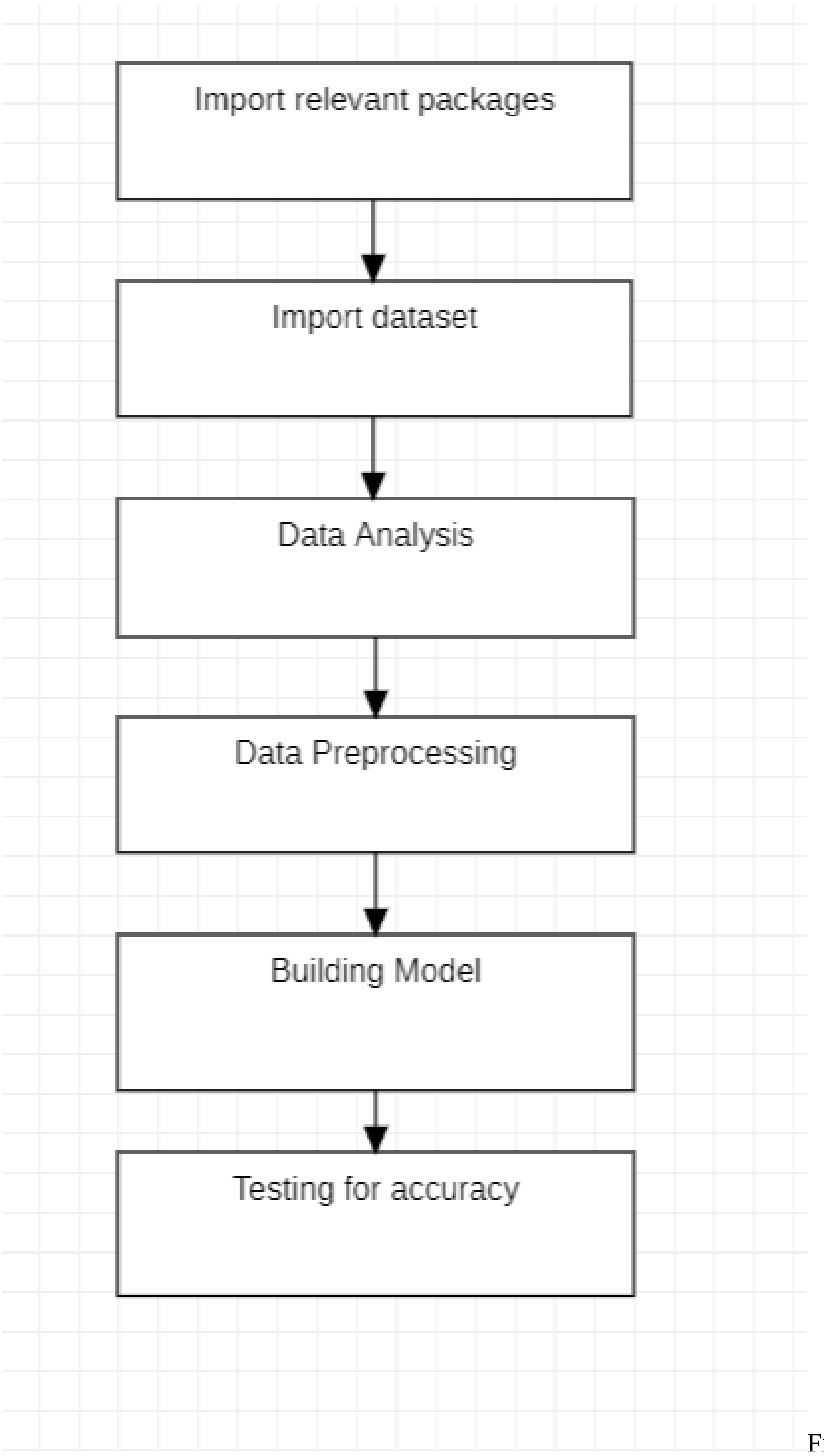


Fig 3

6. OVERVIEW OF TECHNOLOGIES

The technologies used in this project are

6.1 PYTHON

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

6.2 GOOGLE COLAB

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. Google Colab is a must for anyone looking to back their work up to the cloud and to sync their notebooks across multiple devices — but the ease of cloud sharing means reduced data security.

It is one of the most preferred IDEs as most of the libraries that a person may need for artificial intelligence tasks are already available and do not need to be externally installed.

6.3 LIBRARIES

1. Pandas: pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python

2. Numpy: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

3. Matplotlib: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

4. Random: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

5. Tensorflow: TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

6. Keras: Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation

7. Sklearn: Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines

6.4 GOOGLE

Google LLC is an American multinational technology company focusing on search engine technology, online advertising, cloud computing, computer software, quantum computing, e-commerce, artificial intelligence, and consumer electronics.

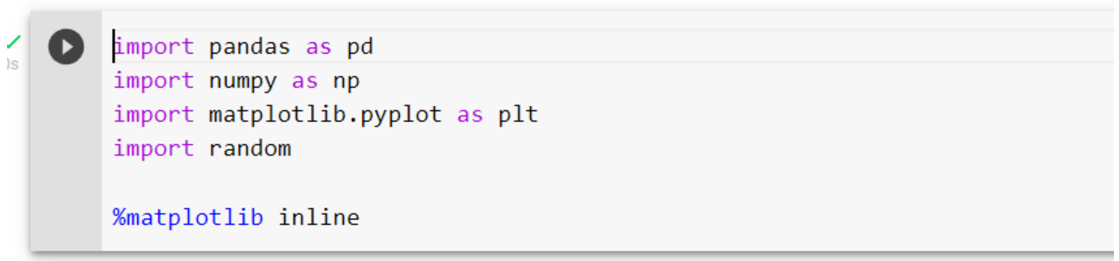
6.5 KAGGLE

Kaggle, a subsidiary of Google LLC. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges

7. IMPLEMENTATION

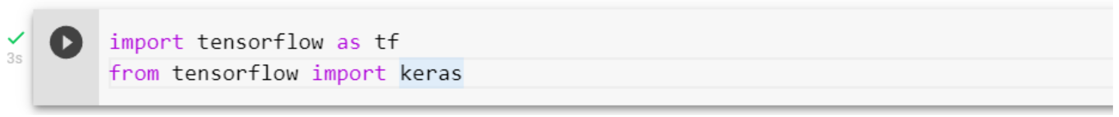
7.1 CODING

7.1.1 Importing Packages:



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import random

%matplotlib inline
```

Fig 4



```
import tensorflow as tf
from tensorflow import keras
```

Fig 5

From Fig 4 and Fig 5, we can see the packages that we have implemented for our project.
Pandas: pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python

Numpy: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

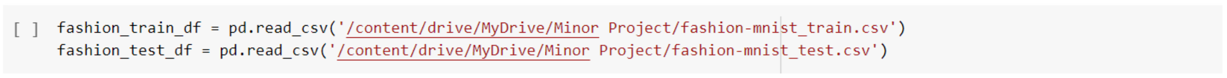
Matplotlib: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

Random: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

Tensorflow: TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

Keras: Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation

7.1.2 Importing the dataset



```
[ ]  fashion_train_df = pd.read_csv('/content/drive/MyDrive/Minor Project/fashion-mnist_train.csv')
     fashion_test_df = pd.read_csv('/content/drive/MyDrive/Minor Project/fashion-mnist_test.csv')
```

Fig 6

The dataset that we used is the ‘Fashion MNIST’ which we found on Kaggle.(Fig 6).

Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Zalando intends Fashion-MNIST to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits.

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

Labels

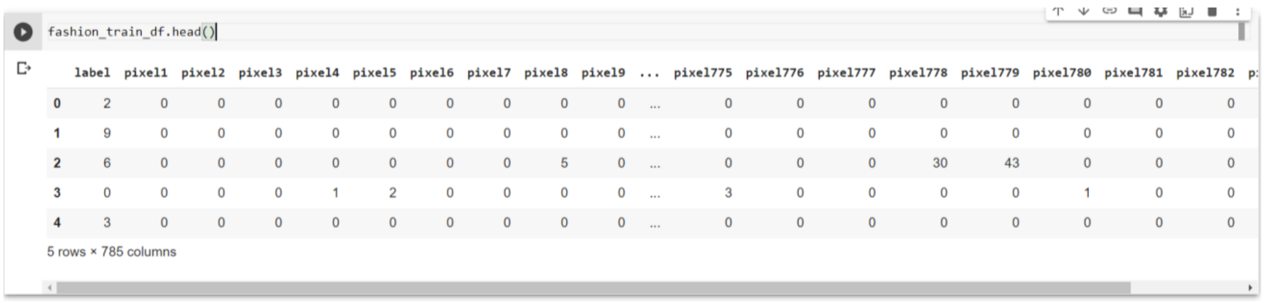
Each training and test example is assigned to one of the following labels:

- 0 T-shirt/top
- 1 Trouser
- 2 Pullover
- 3 Dress

- 4 Coat
- 5 Sandal
- 6 Shirt
- 7 Sneaker
- 8 Bag
- 9 Ankle boot

7.1.3 Data Analysis

Understanding the dataset can be said to be a form of Data Analysis. We performed this step to understand the format or structure of the data before we proceed to manipulate or use the data.



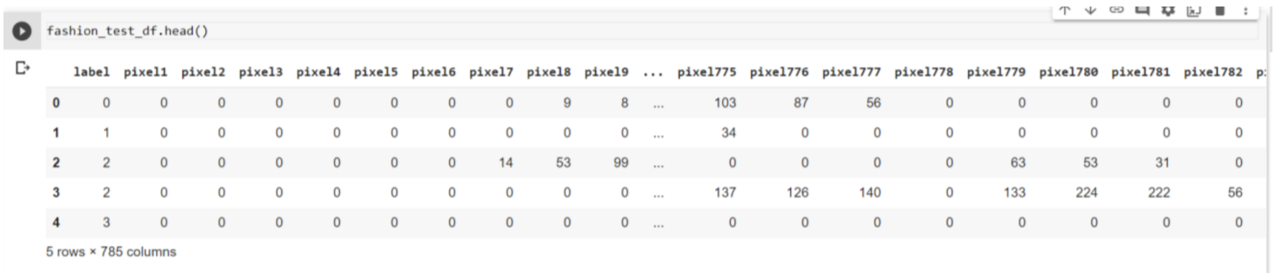
```
fashion_train_df.head()
```

```
   label  pixel1  pixel2  pixel3  pixel4  pixel5  pixel6  pixel7  pixel8  pixel9  ...  pixel775  pixel776  pixel777  pixel778  pixel779  pixel780  pixel781  pixel782  p
0      2       0       0       0       0       0       0       0       0       0  ...         0         0         0         0         0         0         0         0
1      9       0       0       0       0       0       0       0       0       0  ...         0         0         0         0         0         0         0         0
2      6       0       0       0       0       0       0       0       5       0  ...         0         0         0        30        43         0         0         0
3      0       0       0       0       1       2       0       0       0       0  ...         3         0         0         0         0         1         0         0
4      3       0       0       0       0       0       0       0       0       0  ...         0         0         0         0         0         0         0         0
5 rows × 785 columns
```

Fig 7

The head function in Python displays the first five rows of the dataframe by default. It takes in a single parameter: the number of rows. We can use this parameter to display the number of rows of our choice.

In the Fig 7, we displayed the first five rows of the training dataset.

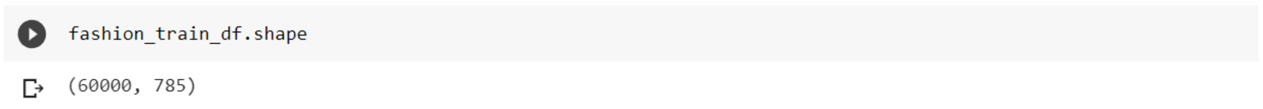


```
fashion_test_df.head()
```

```
   label  pixel1  pixel2  pixel3  pixel4  pixel5  pixel6  pixel7  pixel8  pixel9  ...  pixel775  pixel776  pixel777  pixel778  pixel779  pixel780  pixel781  pixel782  p
0      0       0       0       0       0       0       0       0       9       8  ...       103        87        56         0         0         0         0         0
1      1       0       0       0       0       0       0       0       0       0  ...        34         0         0         0         0         0         0         0
2      2       0       0       0       0       0       0      14      53      99  ...         0         0         0         0        63        53        31         0
3      2       0       0       0       0       0       0       0       0       0  ...       137       126       140         0       133       224       222        56
4      3       0       0       0       0       0       0       0       0       0  ...         0         0         0         0         0         0         0         0
5 rows × 785 columns
```

Fig 8

In the Fig 8, we displayed the first five rows of the testing dataset.



```
fashion_train_df.shape
```

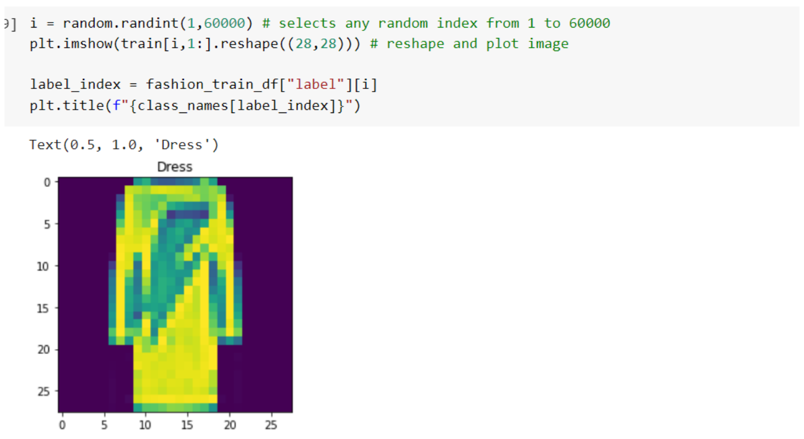
```
(60000, 785)
```

Fig 9

The shape() method is used to fetch the dimensions of Pandas and NumPy type objects in python. Every value represented by the tuple corresponds to the actual dimension in terms of array or row/columns.

Here, 60000 is the number of rows and 785 is the number of columns

The images in the dataset look as follows displayed in Fig 10 and Fig 11



```
i = random.randint(1,60000) # selects any random index from 1 to 60000
plt.imshow(train[i,1:].reshape((28,28))) # reshape and plot image

label_index = fashion_train_df["label"][i]
plt.title(f"{class_names[label_index]}")
```

```
Text(0.5, 1.0, 'Dress')
```

Fig 10

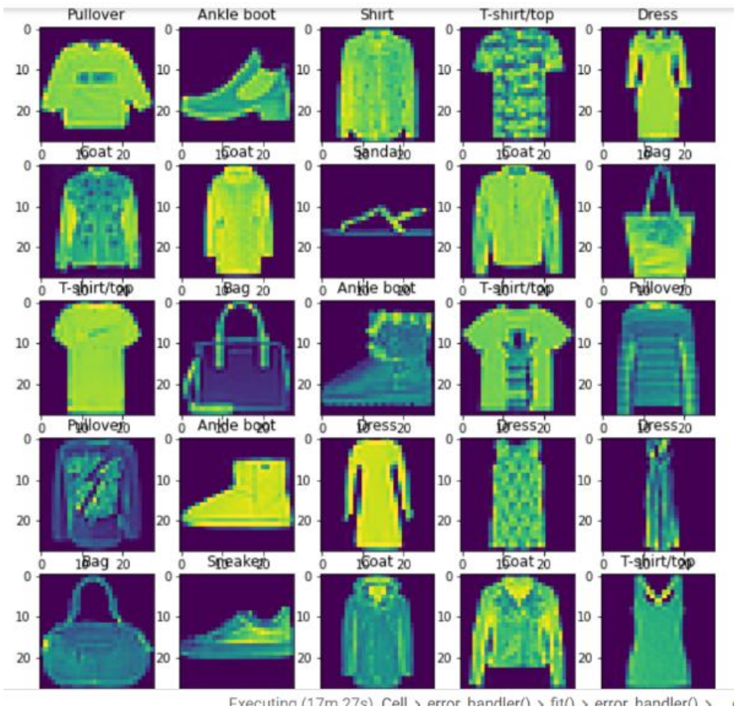
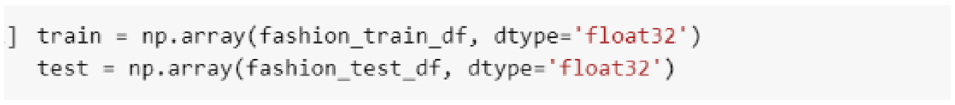


Fig 11

7.1.4 Data Preprocessing

Data preprocessing can refer to manipulation or dropping of data before it is used in order to ensure or enhance performance, and is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects.



```
] train = np.array(fashion_train_df, dtype='float32')
  test = np.array(fashion_test_df, dtype='float32')
```

Fig 12

We started off by converting the training and testing datasets into array(Fig 12)

So now the train dataset is displayed in Fig 13

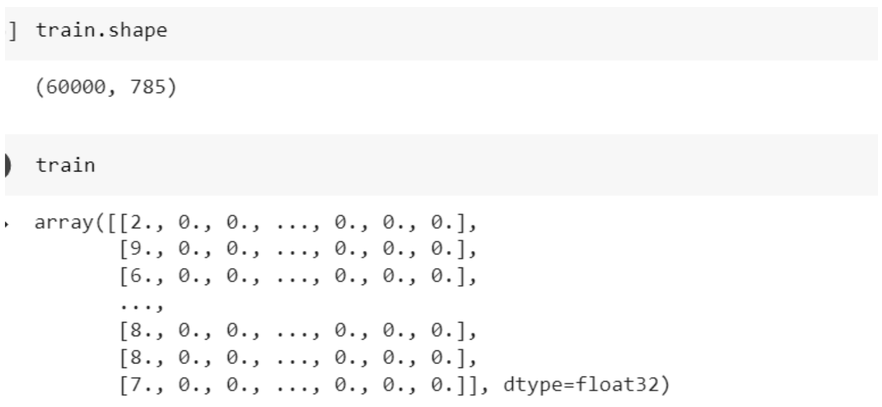

```
] train.shape

(60000, 785)

train

array([[2., 0., 0., ..., 0., 0., 0.],
       [9., 0., 0., ..., 0., 0., 0.],
       [6., 0., 0., ..., 0., 0., 0.],
       ...,
       [8., 0., 0., ..., 0., 0., 0.],
       [8., 0., 0., ..., 0., 0., 0.],
       [7., 0., 0., ..., 0., 0., 0.]], dtype=float32)
```

Fig 13

And the test dataset is displayed in Fig 14

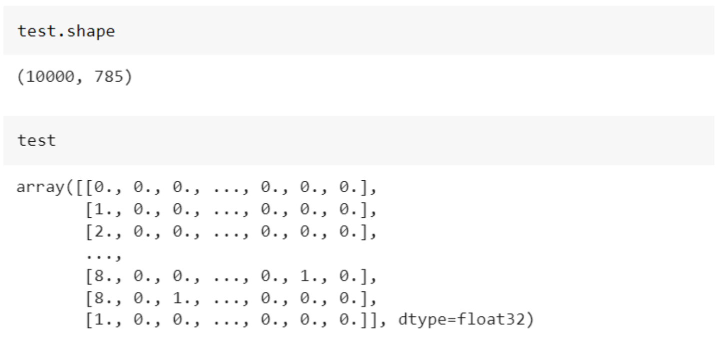

```
test.shape

(10000, 785)

test

array([[0., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [2., 0., 0., ..., 0., 0., 0.],
       ...,
       [8., 0., 0., ..., 0., 1., 0.],
       [8., 0., 1., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.]], dtype=float32)
```

Fig 14

Since, the labels are in the form of 0-9(10 classes) in the dataset, we defined the class names(Fig 15)

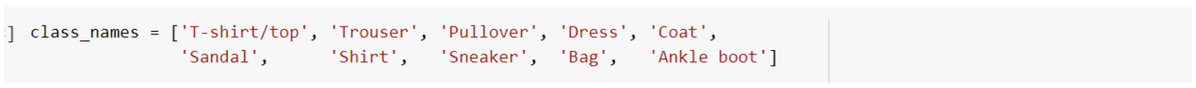
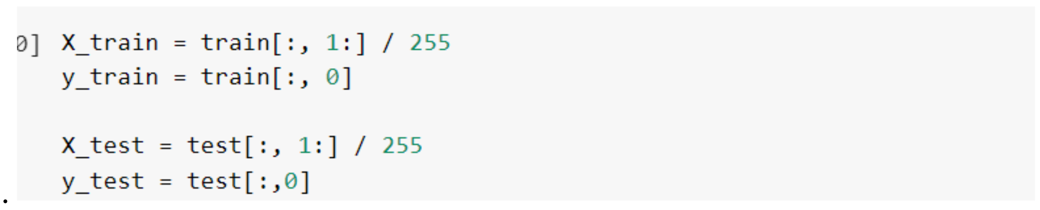

```
] class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
                 'Sandal',      'Shirt',   'Sneaker',  'Bag',   'Ankle boot']
```

Fig 15

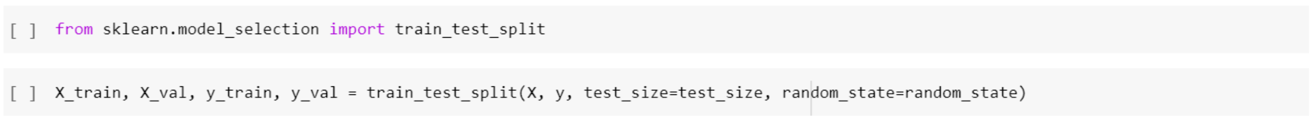
To scale these values to a range of 0 to 1 before feeding them to the neural network model.We divided the values by 255. (



```
0] X_train = train[:, 1:] / 255
   y_train = train[:, 0]

   X_test = test[:, 1:] / 255
   y_test = test[:,0]
```

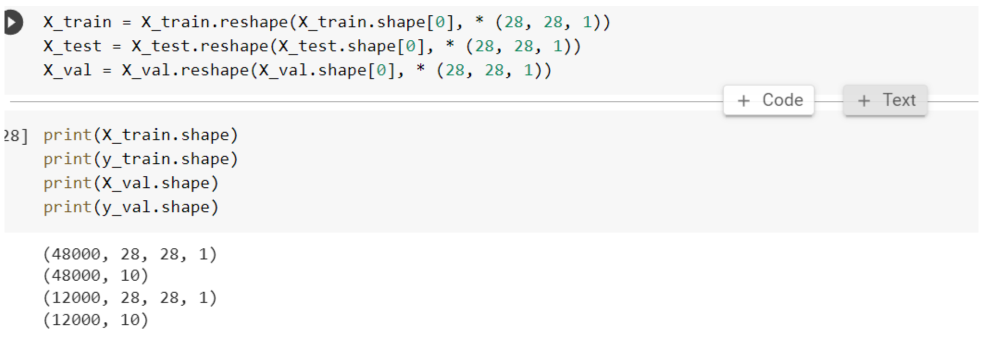
Fig 16

7.1.5 Splitting the dataset



```
[ ]  from sklearn.model_selection import train_test_split

[ ]  X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=test_size, random_state=random_state)
```

Fig 17



```
X_train = X_train.reshape(X_train.shape[0], * (28, 28, 1))
X_test = X_test.reshape(X_test.shape[0], * (28, 28, 1))
X_val = X_val.reshape(X_val.shape[0], * (28, 28, 1))
```

```
28] print(X_train.shape)
    print(y_train.shape)
    print(X_val.shape)
    print(y_val.shape)

    (48000, 28, 28, 1)
    (48000, 10)
    (12000, 28, 28, 1)
    (12000, 10)
```

Fig 18

Train test split is a model validation process that allows you to simulate how your model would perform with new data.

Since, we already loaded our data separately into a training dataset and testing dataset already while importing it, we simply split the training dataset further into a training set and a validation set.

Training Dataset is the sample of data used to fit the model. The actual dataset that we use to train the model (weights and biases in the case of a Neural Network). The model sees and learns from this data.Validation Dataset is the sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.

Test Dataset is he sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

We also reshape the data so that it can fit in our CNN Model.

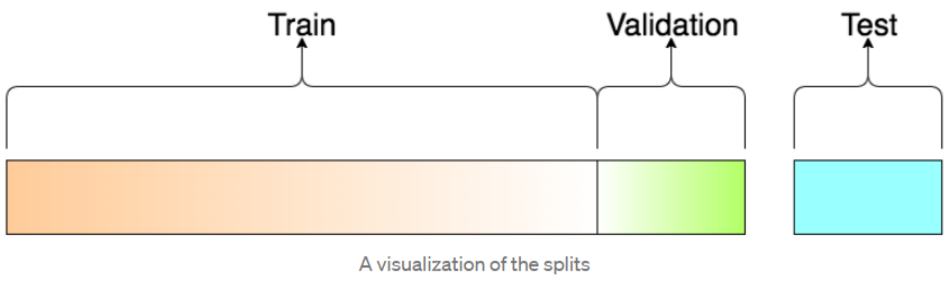
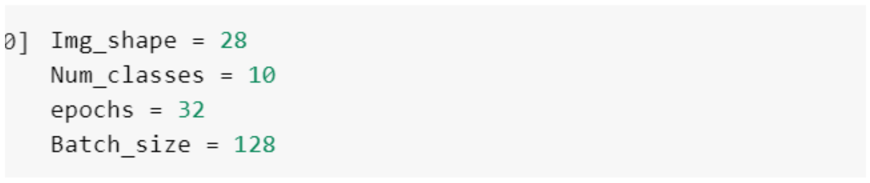


Fig 19

7.1.6 Building the CNN Model



```
0]  Img_shape = 28
    Num_classes = 10
    epochs = 32
    Batch_size = 128
```

Fig 20

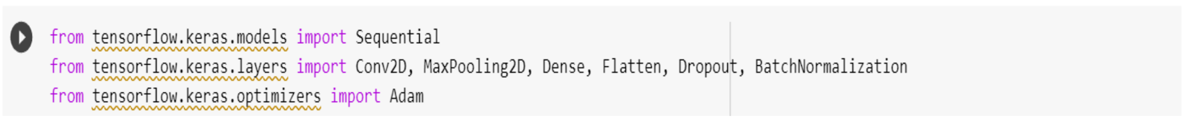
We started off by defining all the classes we may need.

img_shape is the image shape, num_classes is the number of classes

An epoch means training the neural network with all the training data for one cycle. In an epoch, we use all of the data exactly once. A forward pass and a backward pass together are counted as one pass

The batch size is a number of samples processed before the model is updated.

For building the CNN Model, we import the following packages:



```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout, BatchNormalization
from tensorflow.keras.optimizers import Adam
```

Fig 21

A CNN can be instantiated as a Sequential model because each layer has exactly one input and output and is stacked together to form the entire network.

A dense layer is a layer that is deeply connected with its preceding layer which means the neurons of the layer are connected to every neuron of its preceding layer.

Conv2D parameter is the numbers of filters that convolutional layers will learn from.

max pooling reduces the dimensionality of images by reducing the number of pixels in the output from the previous convolutional layer.

Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector.

The Dropout layer is a mask that nullifies the contribution of some neurons towards the next layer and leaves unmodified all others

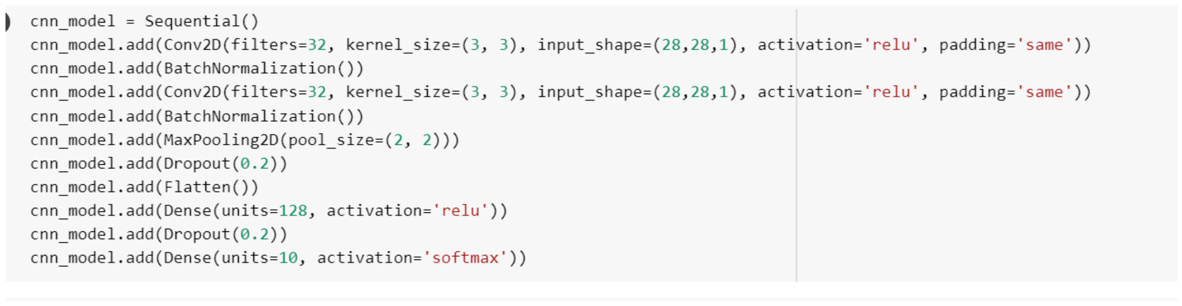
Batch Normalization is a normalization technique done between the layers of a Neural Network instead of in the raw data. It is done along mini-batches instead of the full data set. It serves to speed up training and use higher learning rates, making learning easier. the standard deviation of the neurons' output.

Adam is an adaptive learning rate optimization algorithm that's been designed specifically for training deep neural networks.

7.1.6.1 CNN Model Building

The first layer in this network, keras.layers.Flatten, transforms the format of the images from a two-dimensional array (of 28 by 28 pixels) to a one-dimensional array (of 28 * 28 = 784 pixels). Think of this layer as unstacking rows of pixels in the image and lining them up. This layer has no parameters to learn; it only reformats the data.

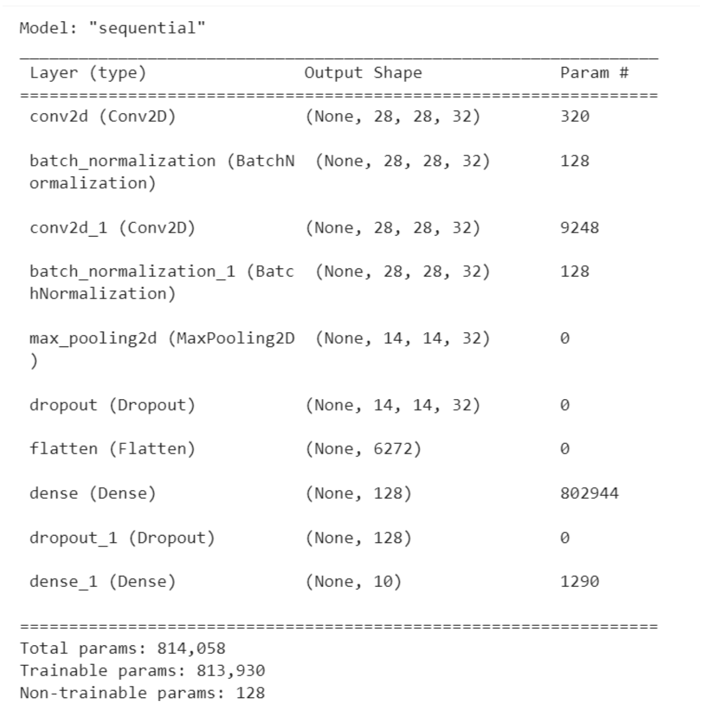
After the pixels are flattened, the network consists of a sequence of two keras.layers.Dense layers. These are densely connected, or fully connected, neural layers. The first Dense layer has 128 nodes (or neurons). The second (and last) layer is a 10-node softmax layer that returns an array of 10 probability scores that sum to 1. Each node contains a score that indicates the probability that the current image belongs to one of the 10 classes.



```
cnn_model = Sequential()
cnn_model.add(Conv2D(filters=32, kernel_size=(3, 3), input_shape=(28,28,1), activation='relu', padding='same'))
cnn_model.add(BatchNormalization())
cnn_model.add(Conv2D(filters=32, kernel_size=(3, 3), input_shape=(28,28,1), activation='relu', padding='same'))
cnn_model.add(BatchNormalization())
cnn_model.add(MaxPooling2D(pool_size=(2, 2)))
cnn_model.add(Dropout(0.2))
cnn_model.add(Flatten())
cnn_model.add(Dense(units=128, activation='relu'))
cnn_model.add(Dropout(0.2))
cnn_model.add(Dense(units=10, activation='softmax'))
```

Fig 22

7.1.6.2 Model Summary

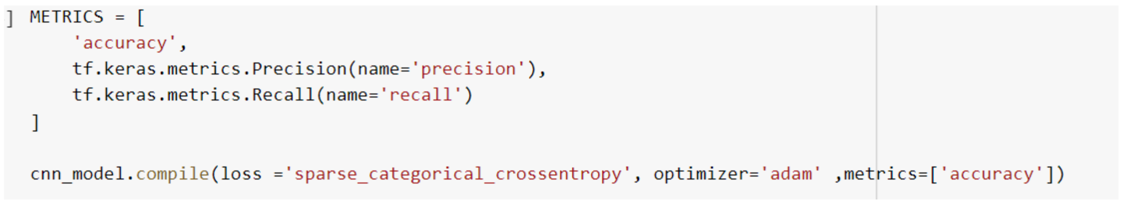


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 32)        320

 batch_normalization (BatchN  (None, 28, 28, 32)       128
 ormalization)

 conv2d_1 (Conv2D)           (None, 28, 28, 32)        9248

 batch_normalization_1 (Batc  (None, 28, 28, 32)       128
 hNormalization)

 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0
 )

 dropout (Dropout)           (None, 14, 14, 32)        0

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 128)               802944

 dropout_1 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 10)                1290

=================================================================
Total params: 814,058
Trainable params: 813,930
Non-trainable params: 128
```

Fig 23

7.1.6.3 Compiling the model

we need to compile our model. Compiling the model takes three parameters: optimizer, loss and metrics.



```
] METRICS = [
      'accuracy',
      tf.keras.metrics.Precision(name='precision'),
      tf.keras.metrics.Recall(name='recall')
  ]

  cnn_model.compile(loss ='sparse_categorical_crossentropy', optimizer='adam' ,metrics=['accuracy'])
```

Fig 24

The optimizer controls the learning rate. We will be using ‘adam’ as our optmizer. Adam is generally a good optimizer to use for many cases. The adam optimizer adjusts the learning rate throughout training.

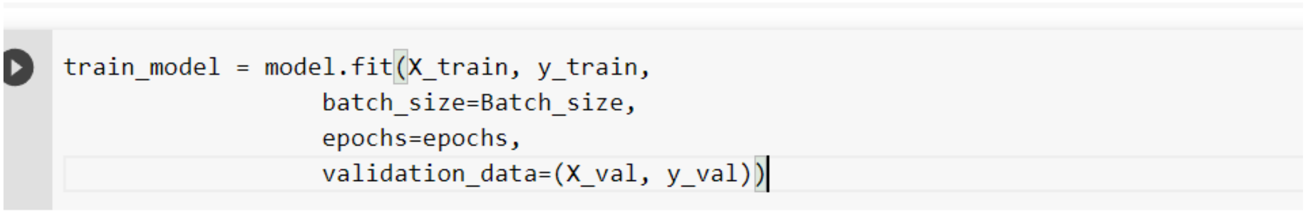
The learning rate determines how fast the optimal weights for the model are calculated. A smaller learning rate may lead to more accurate weights (up to a certain point), but the time it takes to compute the weights will be longer.

Categorical cross-entropy is used when true labels are one-hot encoded, for example, we have the following true values for 3-class classification problem [1,0,0] , [0,1,0] and [0,0,1]. In sparse categorical cross-entropy , truth labels are integer encoded, for example, [1] , [2] and [3] for 3-class problem.

To make things even easier to interpret, we will use the ‘accuracy’ metric to see the accuracy score on the validation set when we train the model.

7.1.6.4 Training the Model

Now we will train our model. To train, we will use the ‘fit()’ function on our model with the following parameters: training data (train_X), target data (train_y), validation data, batch size and the number of epochs.



```
train_model = model.fit(X_train, y_train,
                        batch_size=Batch_size,
                        epochs=epochs,
                        validation_data=(X_val, y_val))
```

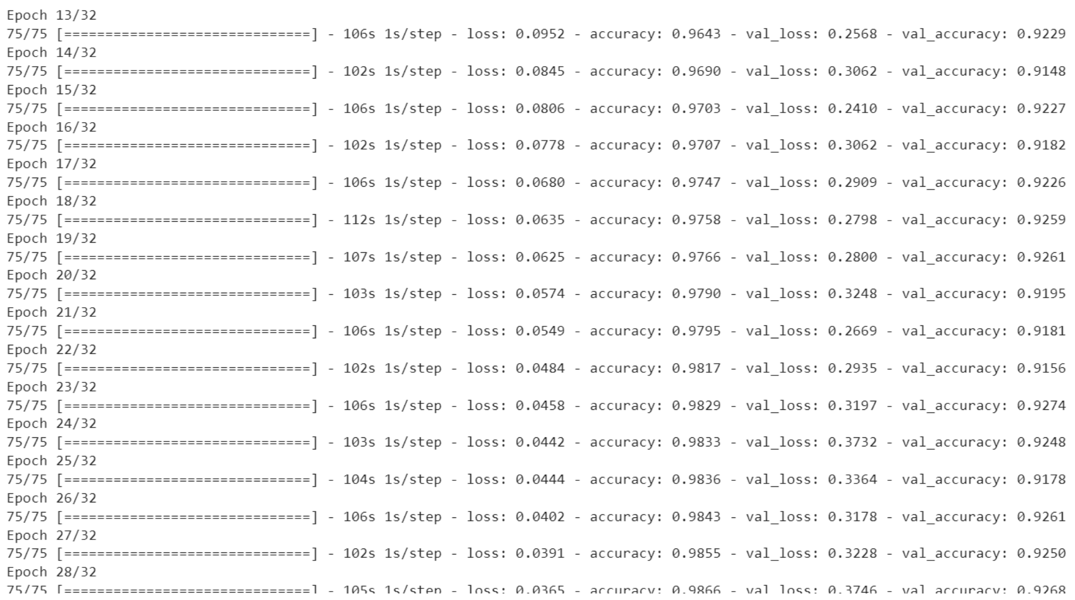
Fig 25

For our validation data, we will use the train set provided to us in our dataset, which we have split into X_train and y_train

The number of epochs is the number of times the model will cycle through the data. The more epochs we run, the more the model will improve, up to a certain point. After that point, the model will stop improving during each epoch.

The batch size is a number of samples processed before the model is updated. The number of epochs is the number of complete passes through the training dataset

The below picture show what the output looks like when the above code in run:



```
Epoch 13/32
75/75 [==============================] - 106s 1s/step - loss: 0.0952 - accuracy: 0.9643 - val_loss: 0.2568 - val_accuracy: 0.9229
Epoch 14/32
75/75 [==============================] - 102s 1s/step - loss: 0.0845 - accuracy: 0.9690 - val_loss: 0.3062 - val_accuracy: 0.9148
Epoch 15/32
75/75 [==============================] - 106s 1s/step - loss: 0.0806 - accuracy: 0.9703 - val_loss: 0.2410 - val_accuracy: 0.9227
Epoch 16/32
75/75 [==============================] - 102s 1s/step - loss: 0.0778 - accuracy: 0.9707 - val_loss: 0.3062 - val_accuracy: 0.9182
Epoch 17/32
75/75 [==============================] - 106s 1s/step - loss: 0.0680 - accuracy: 0.9747 - val_loss: 0.2909 - val_accuracy: 0.9226
Epoch 18/32
75/75 [==============================] - 112s 1s/step - loss: 0.0635 - accuracy: 0.9758 - val_loss: 0.2798 - val_accuracy: 0.9259
Epoch 19/32
75/75 [==============================] - 107s 1s/step - loss: 0.0625 - accuracy: 0.9766 - val_loss: 0.2800 - val_accuracy: 0.9261
Epoch 20/32
75/75 [==============================] - 103s 1s/step - loss: 0.0574 - accuracy: 0.9790 - val_loss: 0.3248 - val_accuracy: 0.9195
Epoch 21/32
75/75 [==============================] - 106s 1s/step - loss: 0.0549 - accuracy: 0.9795 - val_loss: 0.2669 - val_accuracy: 0.9181
Epoch 22/32
75/75 [==============================] - 102s 1s/step - loss: 0.0484 - accuracy: 0.9817 - val_loss: 0.2935 - val_accuracy: 0.9156
Epoch 23/32
75/75 [==============================] - 106s 1s/step - loss: 0.0458 - accuracy: 0.9829 - val_loss: 0.3197 - val_accuracy: 0.9274
Epoch 24/32
75/75 [==============================] - 103s 1s/step - loss: 0.0442 - accuracy: 0.9833 - val_loss: 0.3732 - val_accuracy: 0.9248
Epoch 25/32
75/75 [==============================] - 104s 1s/step - loss: 0.0444 - accuracy: 0.9836 - val_loss: 0.3364 - val_accuracy: 0.9178
Epoch 26/32
75/75 [==============================] - 106s 1s/step - loss: 0.0402 - accuracy: 0.9843 - val_loss: 0.3178 - val_accuracy: 0.9261
Epoch 27/32
75/75 [==============================] - 102s 1s/step - loss: 0.0391 - accuracy: 0.9855 - val_loss: 0.3228 - val_accuracy: 0.9250
Epoch 28/32
75/75 [==============================] - 105s 1s/step - loss: 0.0365 - accuracy: 0.9866 - val_loss: 0.3746 - val_accuracy: 0.9268
```

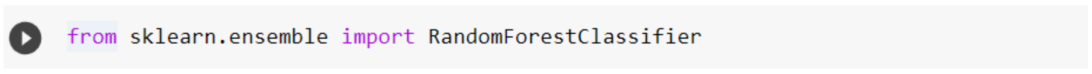
Fig 26

7.1.7 Testing using Random Forest Classifier

An alternate model that we used in our project is the Random Forest Classifier.

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

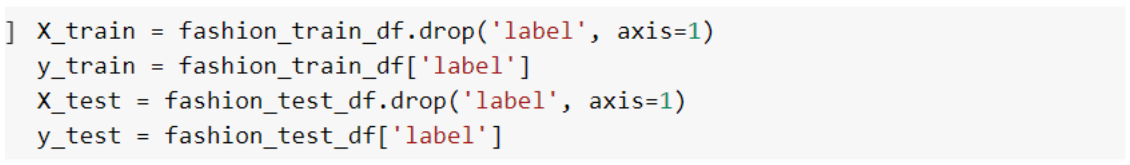
We imported random forest from the sklearn library as shown in Fig 27:



```
from sklearn.ensemble import RandomForestClassifier
```

Fig 27

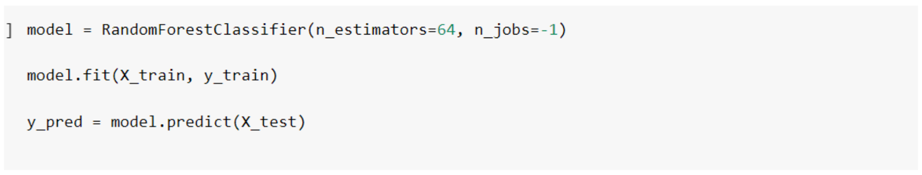
Then we dropped the label columns in both the datasets to help train the model as shown in Fig 28



```
] X_train = fashion_train_df.drop('label', axis=1)
  y_train = fashion_train_df['label']
  X_test = fashion_test_df.drop('label', axis=1)
  y_test = fashion_test_df['label']
```

Fig 28

We then built the model by using X_train and y_train and then predicted using X_test as shown in Fig 29



```
] model = RandomForestClassifier(n_estimators=64, n_jobs=-1)

  model.fit(X_train, y_train)

  y_pred = model.predict(X_test)
```

Fig 29

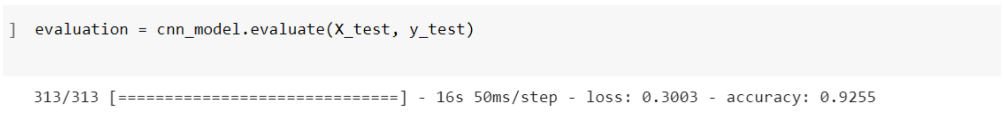
7.2 TESTING

7.2.1 Testing the CNN Model

We started off by checking the accuracy of the model.

The Accuracy score is calculated by dividing the number of correct predictions by the total prediction number. The more formal formula is the following one. As you can see, Accuracy can be easily described using the Confusion matrix terms such as True Positive, True Negative, False Positive, and False Negative.

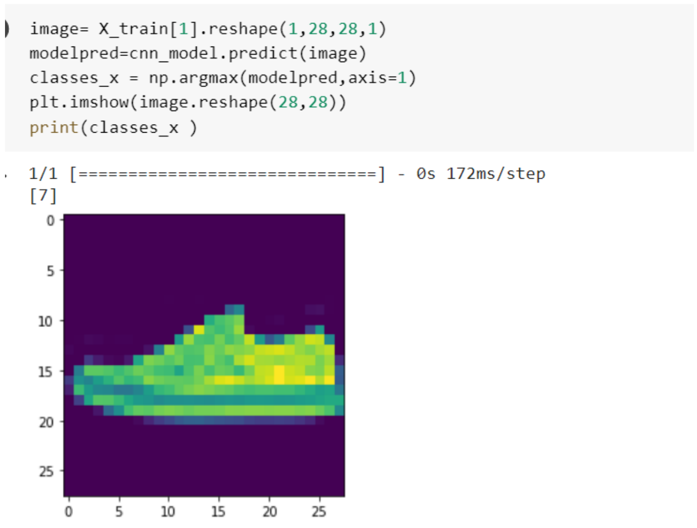
The accuracy obtained for the CNN Model is given in Fig 30.



```
] evaluation = cnn_model.evaluate(X_test, y_test)

  313/313 [==============================] - 16s 50ms/step - loss: 0.3003 - accuracy: 0.9255
```

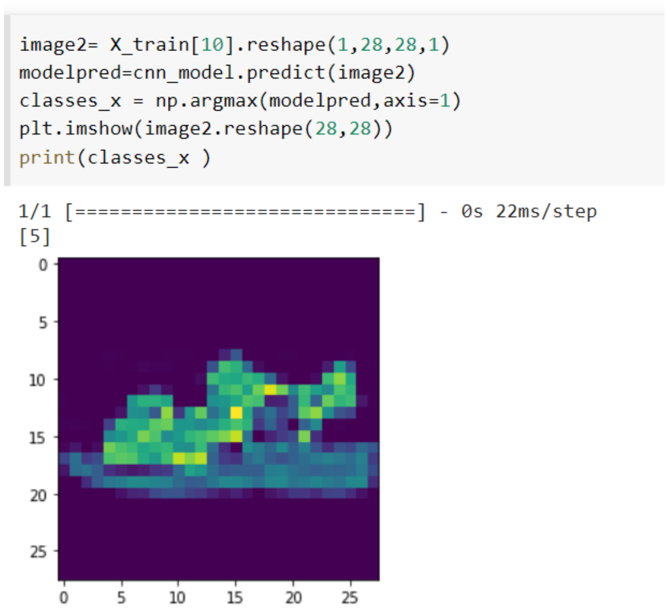
Fig 30

Since the accuracy we received is comparatively high, we decided to make predictions to test it out which are shown in Fig 31 and Fig 32



```
image= X_train[1].reshape(1,28,28,1)
modelpred=cnn_model.predict(image)
classes_x = np.argmax(modelpred,axis=1)
plt.imshow(image.reshape(28,28))
print(classes_x )

1/1 [==============================] - 0s 172ms/step
[7]
```

Fig 31



```
image2= X_train[10].reshape(1,28,28,1)
modelpred=cnn_model.predict(image2)
classes_x = np.argmax(modelpred,axis=1)
plt.imshow(image2.reshape(28,28))
print(classes_x )
```

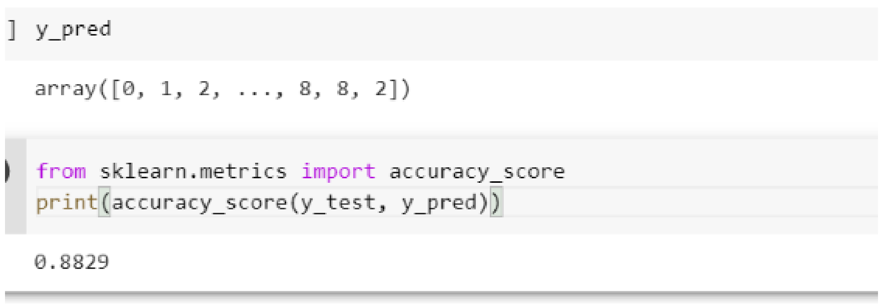
```
1/1 [==============================] - 0s 22ms/step
[5]
```

Fig 32

We notice that in Fig 31, the class that was predicted is 7 which in our case is a Sneaker and in Fig 32, the class that was predicted is 5 which in our case is a Sandal. From these two predictions we can conclude that our model is working well.

7.2.2 Testing the Random Forest Model

The predicted values and the accuracy help us understand how well the model is working which is shown in Fig 33



```
] y_pred
```

```
array([0, 1, 2, ..., 8, 8, 2])
```

```
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test, y_pred))
```

```
0.8829
```

Fig 33

We see that on the predicted values, we received an accuracy of 88.29% which is also pretty high and we can say that our model is working well.

For a classification problem like this, Random Forest is usually deemed most suitable and it did prove itself.

8. RESULTS AND DISCUSSION

After performing our project, we noticed that out of the two models, CNN performs better than Random Forest. We used 32 epochs for CNN and we noticed that with each iteration the accuracy kept increasing. So, we think that if the number of iterations is increased then we may obtain a better result. Since, CNN is a deep learning algorithm and Random Forest is an ML algorithm it is expected that CNN would perform better than Random Forest. Especially as this dataset is based on image classification. In the table below we presented our results.

MODEL	**ACCURACY**
Convolutional Neural Network (CNN)	92.55%
Random Forest Classifier	88.29%

Table 3

We used Random Forest as it is a supervised learning algorithm and is mainly used for classification problems and our project required exactly that. We felt like this was the best ML model that would fit for this project hence we chose to use this model along with CNN and then compare the two.

9. CONCLUSION AND FUTURE USE

Our study is based on suggested an automated classifier to efficiently a large number of fashion garments into ten (10) specific categories based on two approaches: the classic ML approach and the DL approach. We applied Random Forest for the Machine Learning model and CNN for the deep learning model. We can conclude that both of these models may be used for this project as both of them resulted in high accuracy. With the CNN model, we obtained an accuracy or ___ and with Random forest we obtained an accuracy of 88.29%.

Projects like ours help us get a deeper understanding on the process of computer vision. Since, our dataset was already pretty cleaned and labelled, it was easier for our models to learn. But in the real world, this is not the case. Many fashion-oriented websites these days like myntra, ajio and so on show similar results based on the user’s choices. This is the best example to understand the use of computer vision in the world of fashion. Fashion MNIST dataset helps us understand how different clothing items look like, whereas these websites along with identifying the clothing items also have to identify certain patterns or styles. This is essentially a class of extension to our project. This depicts an application of computer vision in the field of fashion.

Our model may also be improved for better accuracy. As perspectives, we propose: to improve these results by improving the architecture of the model built by CNN with changing certain parameters of the model, another suggestion seems important, is to combine CNN with recurrent networks Resnets as feedback may also be important in these cases.

10. REFERENCES

https://ieeexplore.ieee.org/document/9786364

https://www.kaggle.com/datasets/zalando-research/fashionmnist

https://www.analyticsvidhya.com/blog/2021/06/building-a-convolutional-neural-network-using-tensorflow-keras/

https://towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras-329fbbadc5f5