PROJECT REPORT

1 INTRODUCTION

1.1 PROJECT REVIEW

Handwriting recognition is one of the compelling research works going on because every individual in this world has their own style of writing. It is the capability of the computer to identify and understand handwritten digits or characters automatically. Because of the progress in the field of science and technology, everything is being digitalized to reduce human effort.

Hence, there comes a need for handwritten digit recognition in many real-time applications. MNIST data set is widely used for this recognition process and it has 70000 handwritten digits. We use Artificial neural networks to train these images and build a deep learning model. Web application is created where the user can upload an image of a handwritten digit. this image is analyzed by the model and the detected result is returned on to UI

Handwritten Digit Recognition is the capability of a computer to fete the mortal handwritten integers from different sources like images, papers, touch defenses, etc, and classify. them into 10 predefined classes (0-9). This has been a Content of bottomless-exploration in the field of deep literacy. Number recognition has numerous operations like number plate recognition, postal correspondence sorting, bank check processing, etc . (2). In Handwritten number recognition, we face numerous challenges . because of different styles of jotting of different peoples as it . is not an Optic character recognition. This exploration provides a comprehensive comparison between different machine literacy and deep literacy algorithms for the purpose of handwritten number recognition.

1.2 PURPOSE

Handwritten digit recognition is a technology that is very much needed in this world as of today. It is used in the detection of

Vehicle number
Banks for reading cheques
Post office for arranging letters.

It can solve more complex problems and makes humans' job easier. An example is handwritten digit recognition. This is a system widely used in the world to recognize zip code or postal code for mail sorting. There are different techniques that can be used to recognize handwritten characters.

2 LITERATURE SURVEY

2.1 Existing Problem

Recently handwritten digit recognition becomes vital scope and it is appealing many researchers because of its using in variety of machine learning and computer vision applications. However, there are deficient works accomplished on Arabic pattern digits because digits are more challenging than english alphabets. Hence, the lacking research of using digits endeavours us to dig deeper by creating our challenge handwritten digits recognition system

2.2 References

1. Hayder M. Albeahdili, Haider A. Alwzwazy, Naz E. Islam.” Robust Convolutional Neural Networks for Image Recognition”. (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 6, No. 11, 2015

2. Fabien Lauer, Ching Y. Suen, and G´erard Bloch “A trainable feature extractor for handwritten digit recognition‖”, Journal Pattern Recognition,Elsevier, 40 (6), pp.1816-1824, 2007.

3. Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, Zhuowen Tu, “ Deeply-Supervised Nets “ NIPS 2014.

4. M. Fischler and R. Elschlager, “The representation and matching of pictorial structures”, IEEE Transactions on Computer, vol. 22, no. 1, 1973.

5. Kevin Jarrett, Koray Kavukcuoglu, Marc’Aurelio Ranzato and Yann LeCun “What is the Best Multi-Stage Architecture for Object Recognition” CCV’09, IEEE, 2009.

6. Kaiming, He and Xiangyu, Zhang and Shaoqing, Ren and Jian Sun “ Spatial pyramid pooling in deep convolutional networks for visual recognition‖ European”, Conference on Computer Vision, arXiv:1406.4729v4 [cs.CV] 23 Apr 2015.

7. X. Wang, M. Yang, S. Zhu, and Y. Lin. “ Regionlets for generic object detection”. In ICCV, 2013.

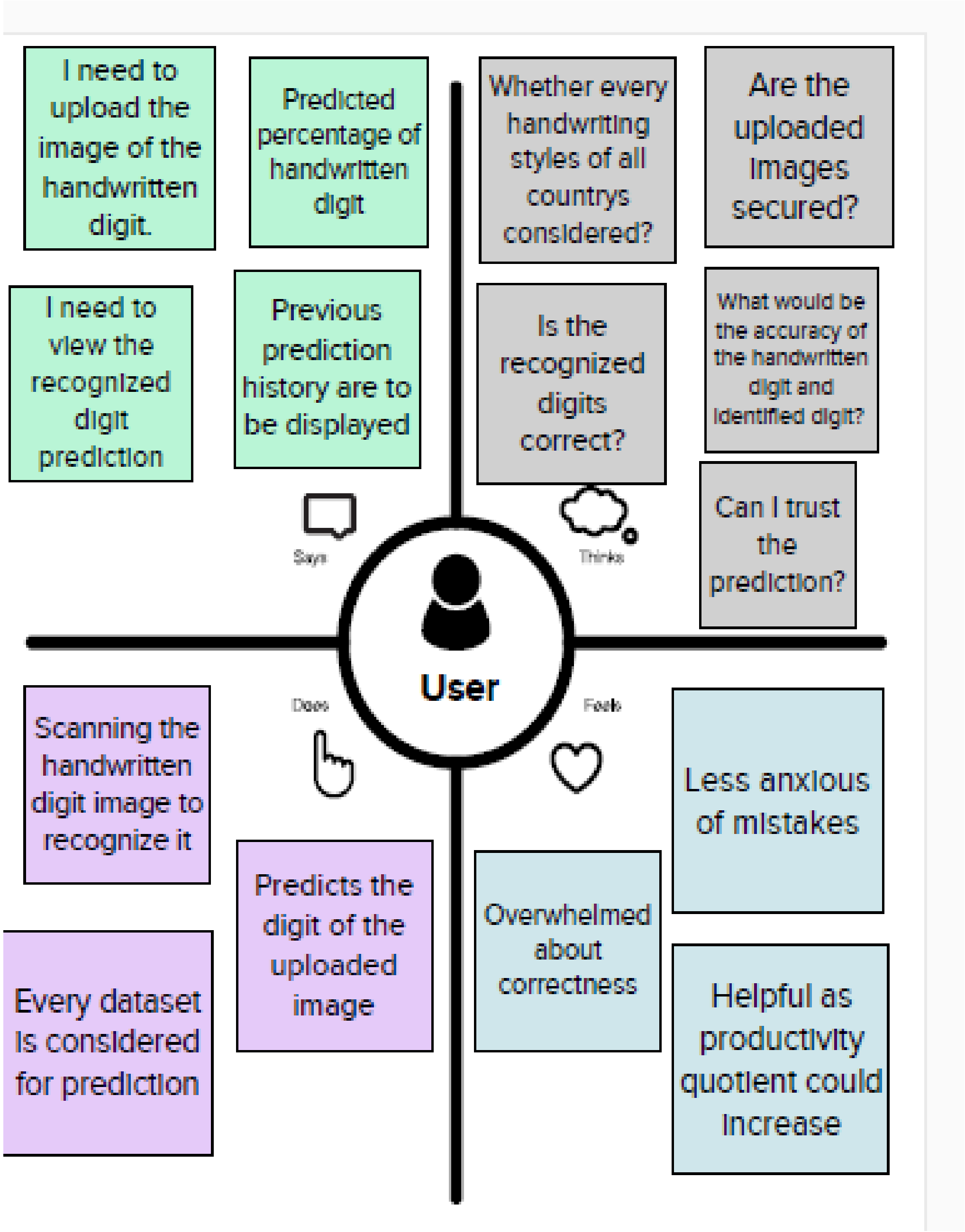
8. Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey. “ImageNet classification with deep convolutional neural networks”. In Advances in Neural Information Processing Systems 25 (NIPS’2012). 2012.

2.3 Problem Statement Definition

The handwritten digit recognition is the capability of computer applications to recognize the human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different shapes and sizes. The handwritten digit recognition system is a way to tackle this problem which uses the image of a digit and recognizes the digit present in the image

3 IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP



I need to upload the image of the handwritten digit.

Predicted percentage of handwritten digit

I need to view the recognized digit prediction

Previous prediction history are to be displayed

Whether every handwriting styles of all countrys considered?

Are the uploaded images secured?

Is the recognized digits correct?

What would be the accuracy of the handwritten digit and identified digit?

Can I trust the prediction?

Scanning the handwritten digit image to recognize it

Predicts the digit of the uploaded image

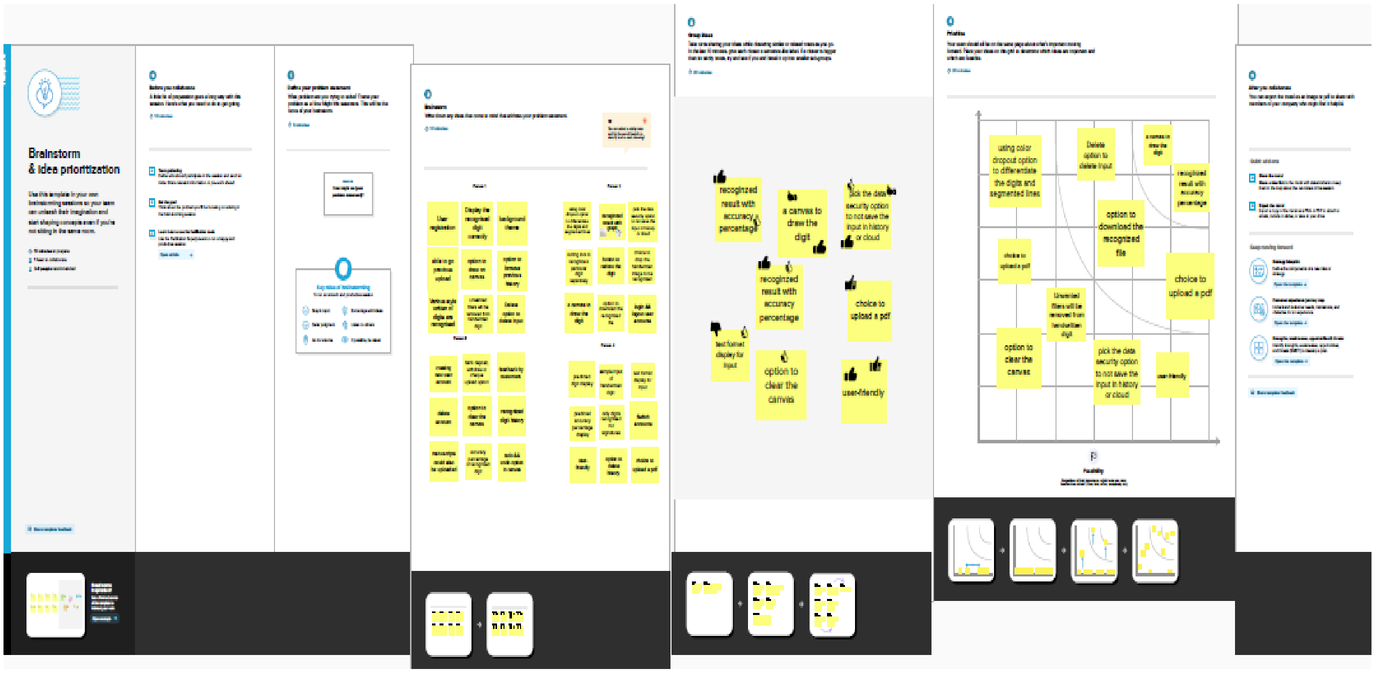
Every dataset is considered for prediction

Overwhelmed about correctness

Less anxious of mistakes

Helpful as productivity quotient could increase

3.2 IDEATION AND BRAINSTROMING

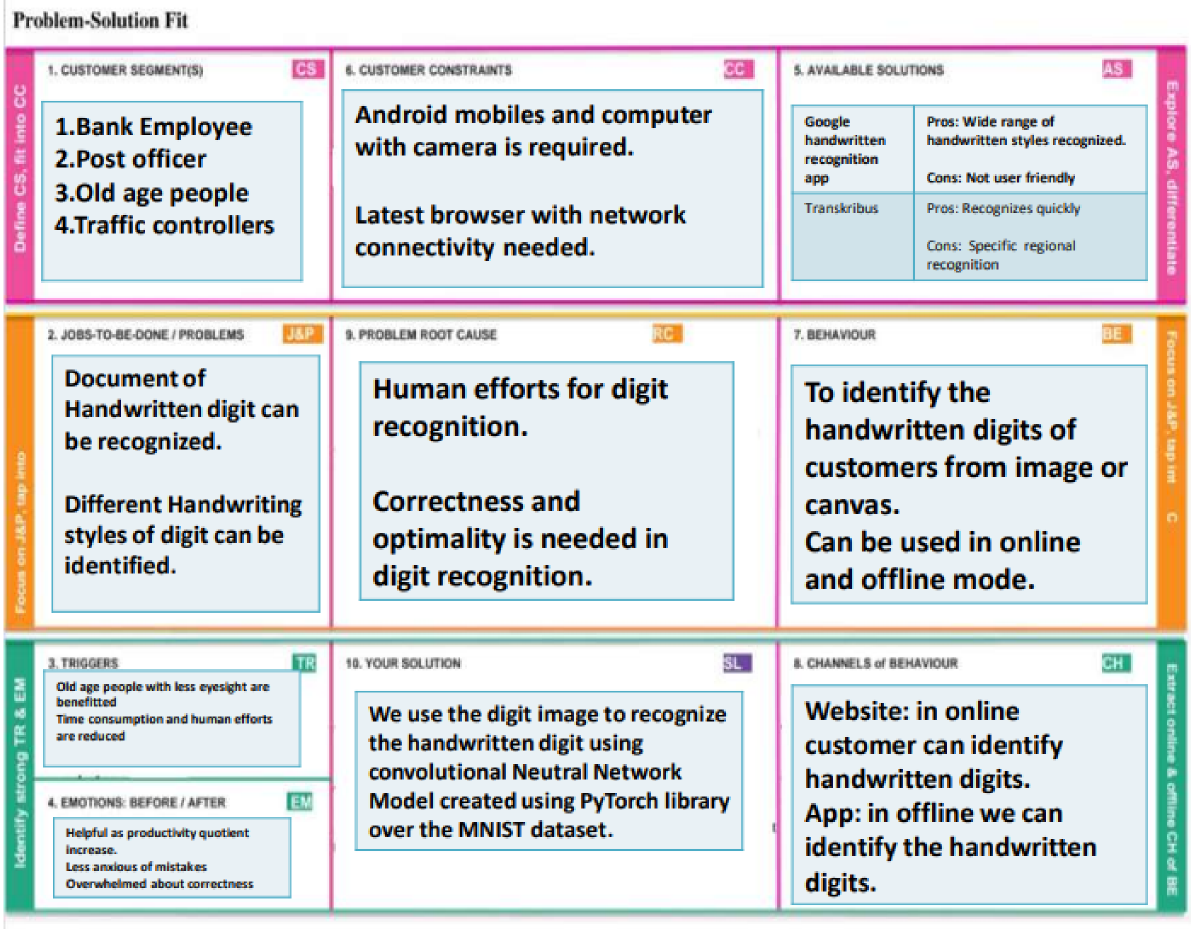

Brainstorm
& idea prioritization
recognized result with accuracy percentage
a canvas to draw the digit
pick the data security option to not save the input in history or cloud
recognized result with accuracy percentage
choice to upload a pdf
text format display for input
option to clear the canvas
user-friendly
using color dropout option to differentiate the digits and segmented lines
Delete option to delete input
recognized result with accuracy percentage
option to download the recognized file
choice to upload a pdf
choice to upload a pdf
Unwanted filters will be removed from handwritten digit
option to clear the canvas
pick the data security option to not save the input in history or cloud
user-friendly
Feasibility

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	An optimized way to recognize and predict the handwritten digit with increased accuracy (expecting to achieve results more than 97% accuracy), low run time and low memory requirements.
2	Idea / Solution description	The idea is to calculate features that make it possible to distinguish between different numbers. Some example features for this dataset might include the number of colour pixels, or maybe the width and the height of the digits and use the SVMS algorithm to optimize the prediction.

		However, generally just the algorithm is not enough to get optimal classification rates. Another important aspect to improve scores is feature extraction. Although the other algorithms might not be what you are looking for, it is possible that adding more features can improve the performance of the algorithm you are currently using as well.
3	Novelty / Uniqueness	Precisely recognise and predict the uploaded document and canvas drawn digits using the SVMS algorithm and the feature extraction aspect
4	Sociusiness Model (Revenue Model)al Impact / Customer Satisfaction	It helps the postal department, banking sector and traffic department. It also helps the old age humans with less eye sigh
5	Business Model (Revenue Model)	We can generate revenue by advertisement part of our website and application play store revenue. We are aiming to collaborate with banking sectors, postal sectors and traffic control department to use our project in recognising cheque digits, zip code and number plate digit. We share the profit by group driveways.
6	Scalability of the Solution	The scalability of our solution is to get accuracy of around 97% or more and also to grow exponentially in revenue.

3.4 Problem Solution Fit


Define CS, fit into CC
Focus on J&P, tap into
Identify strong TR & EM
Explore AS, differentiate
Focus on J&P, tap int C
Extract online & offline CH of BE

Problem-Solution Fit

1. CUSTOMER SEGMENT(S) CS

1.Bank Employee
2.Post officer
3.Old age people
4.Traffic controllers

6. CUSTOMER CONSTRAINTS CC

Android mobiles and computer with camera is required.

Latest browser with network connectivity needed.

5. AVAILABLE SOLUTIONS AS

Google handwritten recognition app	Pros: Wide range of handwritten styles recognized. Cons: Not user friendly
Transkribus	Pros: Recognizes quickly Cons: Specific regional recognition

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Document of Handwritten digit can be recognized.

Different Handwriting styles of digit can be identified.

9. PROBLEM ROOT CAUSE RC

Human efforts for digit recognition.

Correctness and optimality is needed in digit recognition.

7. BEHAVIOUR BE

To identify the handwritten digits of customers from image or canvas.
Can be used in online and offline mode.

3. TRIGGERS TR

Old age people with less eyesight are benefitted
Time consumption and human efforts are reduced

4. EMOTIONS: BEFORE / AFTER EM

Helpful as productivity quotient increase.
Less anxious of mistakes
Overwhelmed about correctness

10. YOUR SOLUTION SL

We use the digit image to recognize the handwritten digit using convolutional Neutral Network Model created using PyTorch library over the MNIST dataset.

8. CHANNELS of BEHAVIOUR CH

Website: in online customer can identify handwritten digits.
App: in offline we can identify the handwritten digits.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Email Registration through Phone number
FR-2	User Confirmation	Confirmation via Email
FR-3	User Details Storage	The details of the users are stored
FR-4	Scan the image	The handwritten digit is scanned or uploaded
FR-5	Analyse the image in the Dataset	The scanned or uploaded image is compared with the dataset
FR-6	Recognize the digit	Based on the dataset the handwritten digit is recognized

4.2 Non-Functional requirements

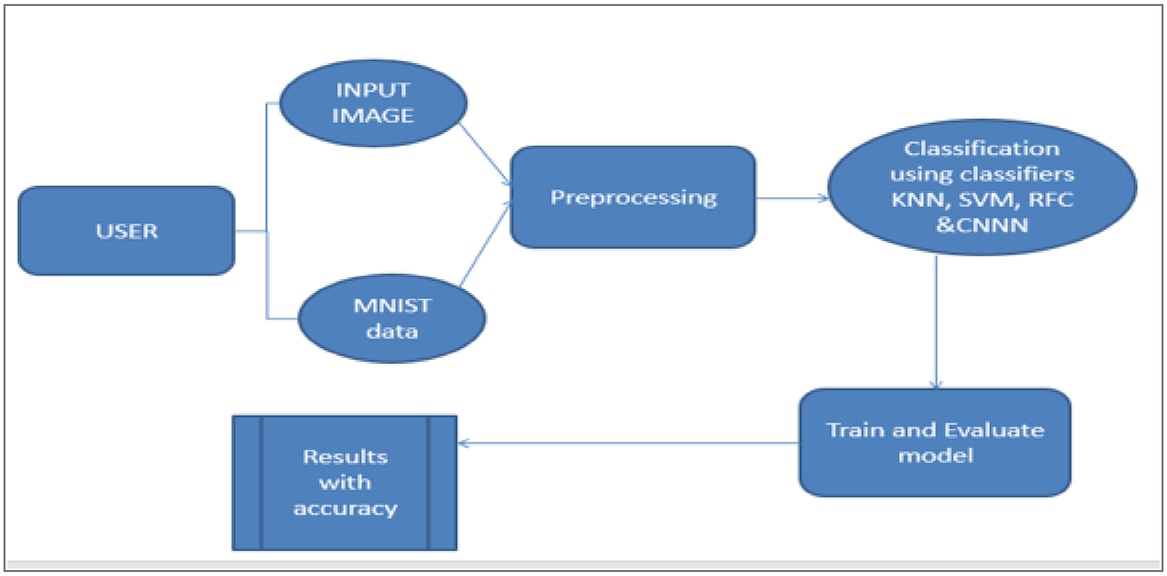
Following are the functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The handwritten digit recognition is used in postal office and bank check processing
NFR-2	**Security**	The user's bank details and postal addresses are secured
NFR-3	**Reliability**	Almost the digit recognized is reliable
NFR-4	**Performance**	The handwritten image is recognized at the maximum speed
NFR-5	**Availability**	The software will be available at any time
NFR-6	**Scalability**	Algorithm of our model can recognize and predict the handwritten digit up to 95% and above.

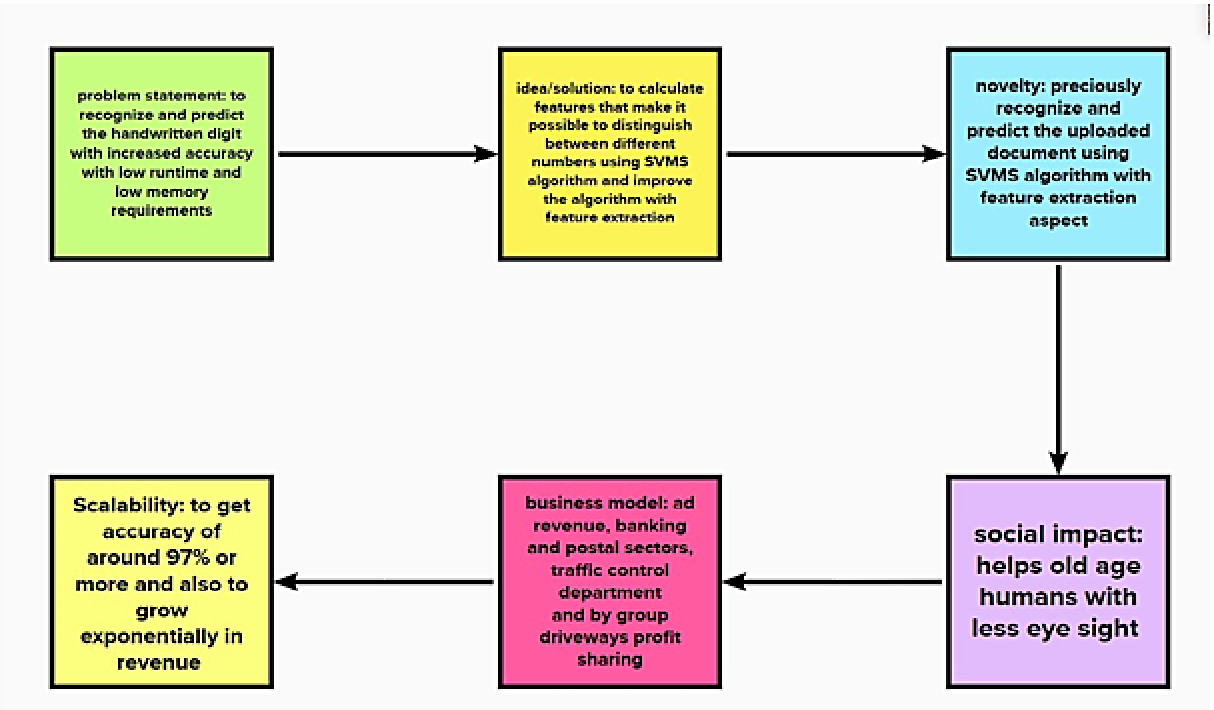
5. PROJECT DESIGN

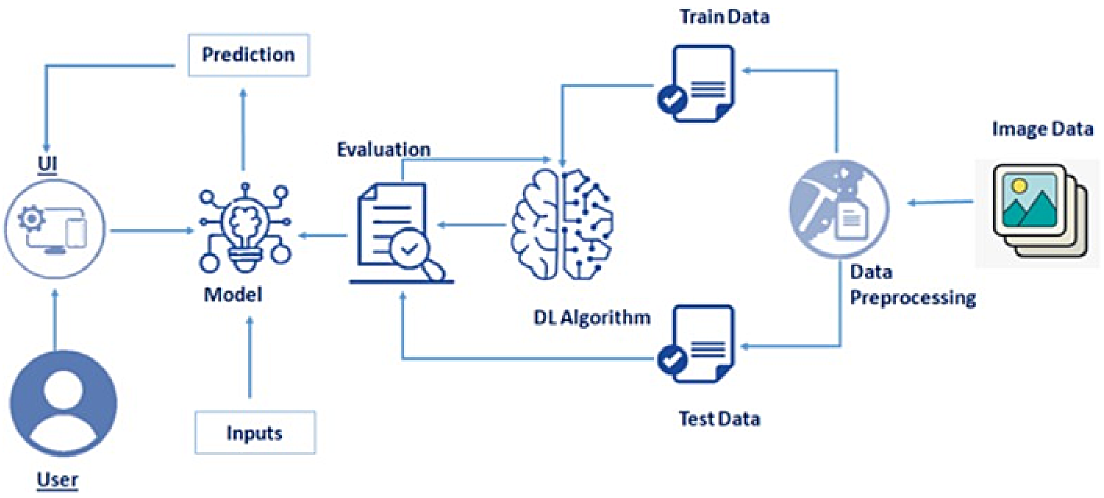
5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical Architecture


problem statement: to recognize and predict the handwritten digit with increased accuracy with low runtime and low memory requirements
idea/solution: to calculate features that make it possible to distinguish between different numbers using SVMS algorithm and improve the algorithm with feature extraction
novelty: preciously recognize and predict the uploaded document using SVMS algorithm with feature extraction aspect
social impact: helps old age humans with less eye sight
business model: ad revenue, banking and postal sectors, traffic control department and by group driveways profit sharing
Scalability: to get accuracy of around 97% or more and also to grow exponentially in revenue



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can login into the application through email & password	I can login into the application through email &	Medium	Sprint-1

				password		
	Dashboard	USN-4	As a user, I can go to dashboard	I can access the dashboard	High	Sprint-1
Customer (Web user)	Login	USN-1	As a user, I can login the application using the registered email	I can access my account	High	Sprint-1
	Upload	USN-2	As a user I can upload the handwritten digit	I can upload my image	High	Sprint-1
	Result	USN-3	As a user I can get the recognized digit	I can view the output image	High	Sprint-1
Customer Care Executive	Dashboard	USN-1	As a user I can access the dashboard and send queries	I can send the queries and receive	High	Sprint-1
Administrator	Security	USN-1	As a administer I can access the user data	I can access data	Medium	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

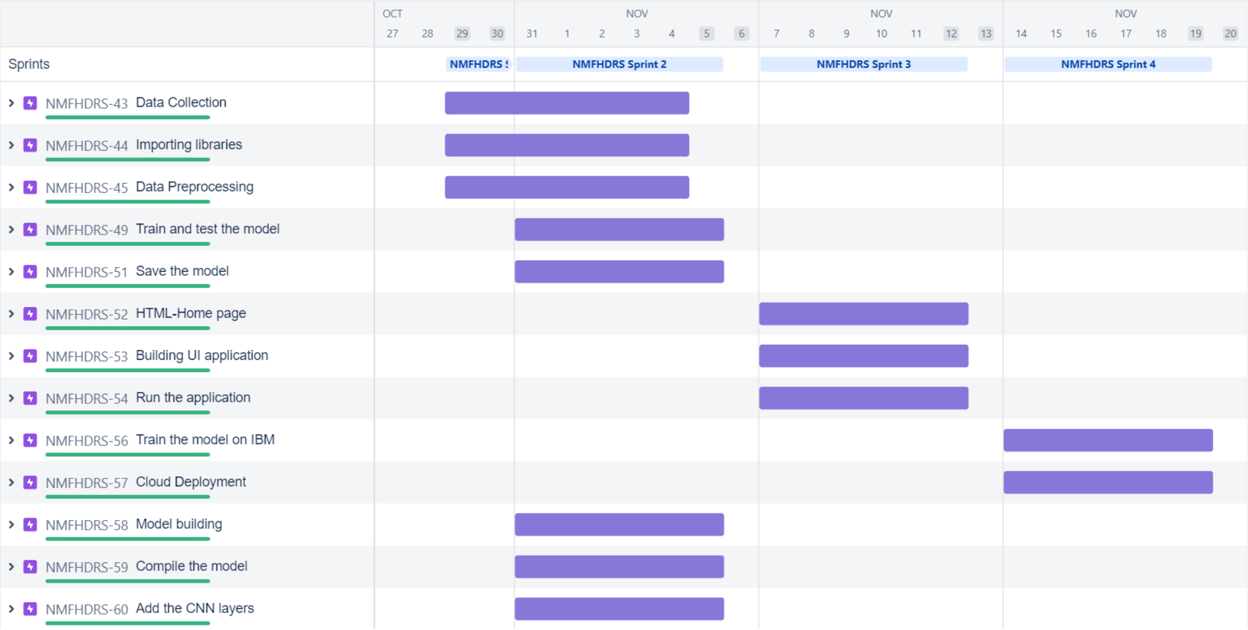
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data collection	USN-1	We need to collect the data with different handwriting to train the model.	6	High	Preetha.S Thasneem Banu E
Sprint-1	Importing the libraries	USN-2	We have to implement necessary libraries in python package.	4	Low	Sindhiya.R Madhumitha S M
Sprint-1	Data pre processing	USN-3	We will load the dataset, handle the missing values, scale and split the data.	10	Medium	Thasneem Banu E Sindhiya.R
Sprint-2	Model building	USN-4	We will get an application with ML model which provide high accuracy of recognized handwritten digit.	5	High	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M
Sprint-2	Add the CNN layers	USN-5	We add input convolutional layer, max pooling layer, flatten, hidden and output layer to the model	5	High	Preetha.S Madhumitha S M
Sprint-2	Compile the model	USN-6	We compile the model for trained dataset.	2	Medium	Sindhiya.R Thasneem Banu E
Sprint-2	Train and test the model	USN-7	We train and test the model for the dataset collected and data are validated.	4	High	Thasneem Banu E Madhumitha S M
Sprint-2	Save the model	USN-8	The compiled data are saved and integrated with an android application or web application.	2	Low	Preetha.S Sindhiya.R
Sprint-3	HTML-Home page	USN-9	We upload the input image that contains handwritten digits.	10	Medium	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M
Sprint-3	Building UI application	USN-10	We provide the fundamental details about the usage of application to customer.	5	Low	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M
Sprint-3	Run the application	USN-11	We can see the predicted or recognized digits in the application.	5	Medium	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M

Sprint - 4	Train the model on IBM	USN-12	We train the model in IBM cloud and integrate the results.	10	High	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M
Sprint - 4	Cloud Deployment	USN-13	We can access the web application	10	High	Preetha.S Sindhiya.R Thasneem Banu E Madhumitha S M

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	04 Nov 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA



Sprints

NMFHDRS-43 Data Collection

NMFHDRS-44 Importing libraries

NMFHDRS-45 Data Preprocessing

NMFHDRS-49 Train and test the model

NMFHDRS-51 Save the model

NMFHDRS-52 HTML-Home page

NMFHDRS-53 Building UI application

NMFHDRS-54 Run the application

NMFHDRS-56 Train the model on IBM

NMFHDRS-57 Cloud Deployment

NMFHDRS-58 Model building

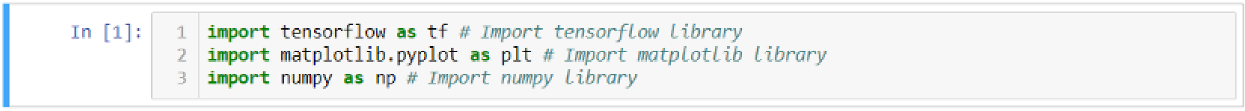
NMFHDRS-59 Compile the model

NMFHDRS-60 Add the CNN layers

7. CODING & SOLUTIONING

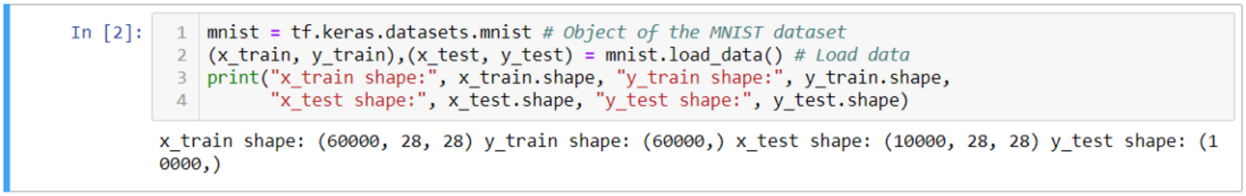
7.1 Feature 1

Required libraries and packages are imported during the implementation of project.



```
In [1]:
import tensorflow as tf # Import tensorflow library
import matplotlib.pyplot as plt # Import matplotlib library
import numpy as np # Import numpy library
```

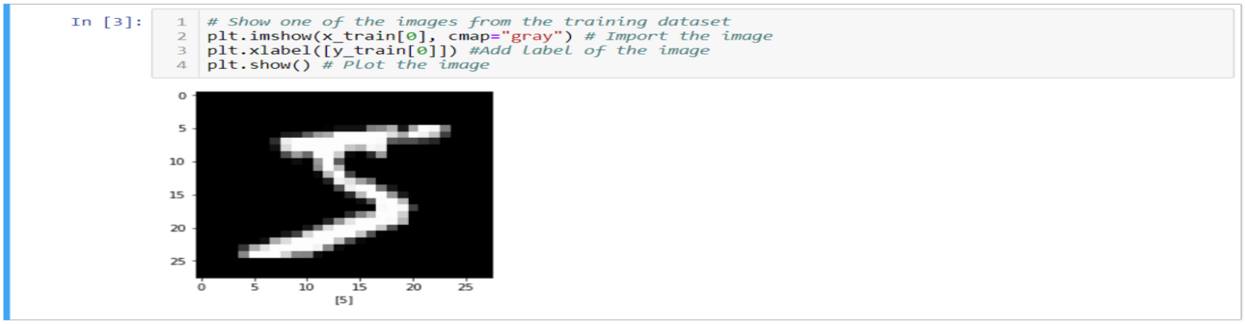
Then, object of the MNIST dataset is created and later loaded through the help of Tensorflow backend using Keras. After that, total number of train along with their dimension and test data set is printed out in order to view total number of elements present in the dataset.



```
In [2]:
mnist = tf.keras.datasets.mnist # Object of the MNIST dataset
(x_train, y_train),(x_test, y_test) = mnist.load_data() # Load data
print("x_train shape:", x_train.shape, "y_train shape:", y_train.shape,
      "x_test shape:", x_test.shape, "y_test shape:", y_test.shape)
```

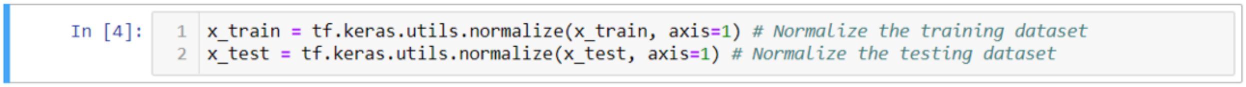
```
x_train shape: (60000, 28, 28) y_train shape: (60000,) x_test shape: (10000, 28, 28) y_test shape: (1
0000,)
```

After the dataset is loaded, one of the images from the training dataset is loaded and displayed in gray scale format by using matplotlib library.



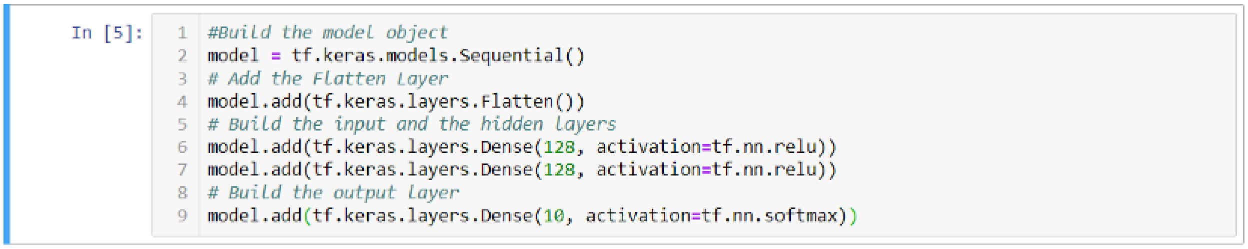
```
In [3]:
# Show one of the images from the training dataset
plt.imshow(x_train[0], cmap="gray") # Import the image
plt.xlabel([y_train[0]]) #Add label of the image
plt.show() # Plot the image
```

Then the training and testing data sets are normalized where image data values are converted in terms of 0 and 1.



```
In [4]:
x_train = tf.keras.utils.normalize(x_train, axis=1) # Normalize the training dataset
x_test = tf.keras.utils.normalize(x_test, axis=1) # Normalize the testing dataset
```

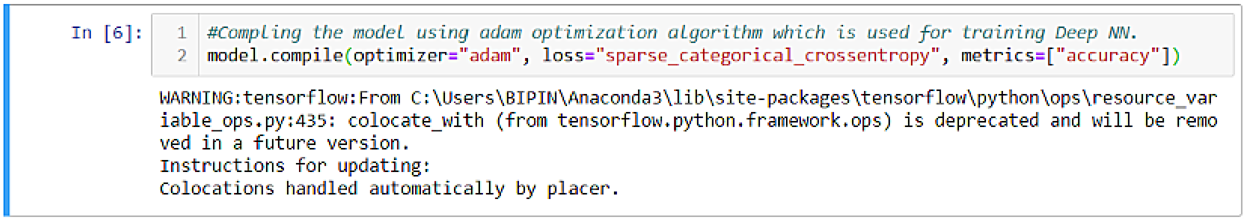
After normalizing the data, a CNN model is created using keras library. Then the Flatten layer is added into the model. Then after, input and hidden layer followed by output layers are built using CNN algorithm.



```
In [5]:
#Build the model object
model = tf.keras.models.Sequential()
# Add the Flatten Layer
model.add(tf.keras.layers.Flatten())
# Build the input and the hidden layers
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
# Build the output layer
model.add(tf.keras.layers.Dense(10, activation=tf.nn.softmax))
```

7.2 Feature 2

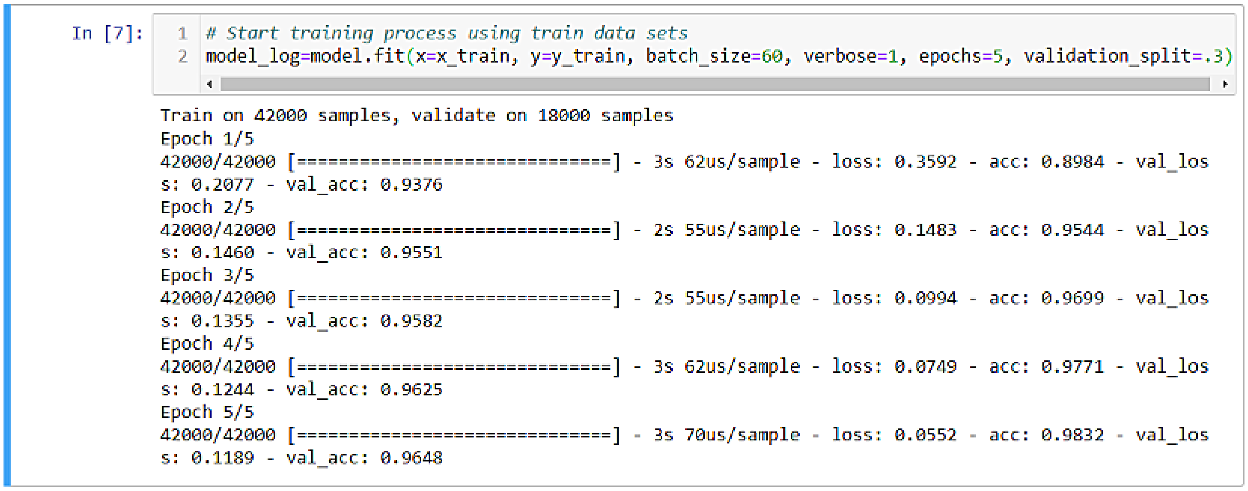
After building the model successfully, model is compiled using Adam optimization algorithm where this algorithm is used for training Deep Neural Networks (DNN).



```
In [6]:
#Compling the model using adam optimization algorithm which is used for training Deep NN.
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"])
```

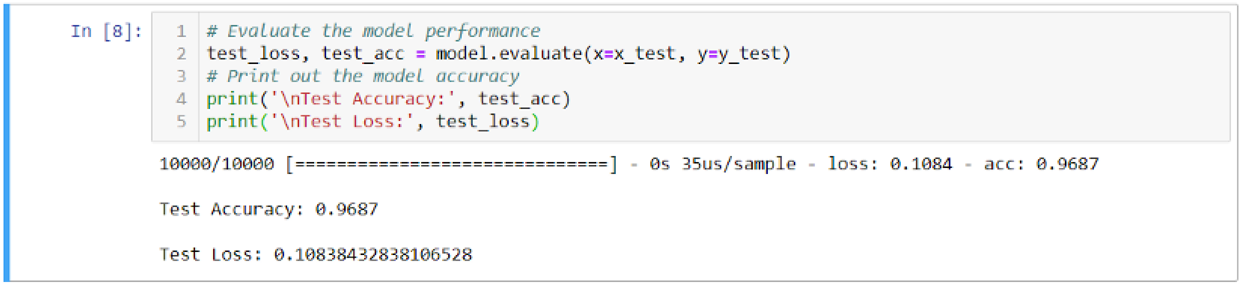
```
WARNING:tensorflow:From C:\Users\BIPIN\Anaconda3\lib\site-packages\tensorflow\python\ops\resource_var
iable_ops.py:435: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be remo
ved in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

After compiling the model, model is fit to train data train and training is started using train data set, due to which cross entropy loss and accuracy of the model can be achieved while training the data from dataset.



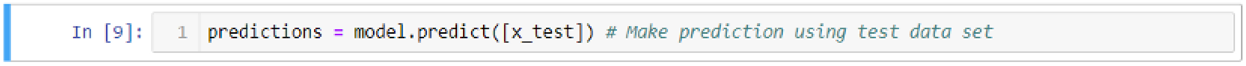
```
In [7]:  # Start training process using train data sets
         model_log=model.fit(x=x_train, y=y_train, batch_size=60, verbose=1, epochs=5, validation_split=.3)

Train on 42000 samples, validate on 18000 samples
Epoch 1/5
42000/42000 [==============================] - 3s 62us/sample - loss: 0.3592 - acc: 0.8984 - val_los
s: 0.2077 - val_acc: 0.9376
Epoch 2/5
42000/42000 [==============================] - 2s 55us/sample - loss: 0.1483 - acc: 0.9544 - val_los
s: 0.1460 - val_acc: 0.9551
Epoch 3/5
42000/42000 [==============================] - 2s 55us/sample - loss: 0.0994 - acc: 0.9699 - val_los
s: 0.1355 - val_acc: 0.9582
Epoch 4/5
42000/42000 [==============================] - 3s 62us/sample - loss: 0.0749 - acc: 0.9771 - val_los
s: 0.1244 - val_acc: 0.9625
Epoch 5/5
42000/42000 [==============================] - 3s 70us/sample - loss: 0.0552 - acc: 0.9832 - val_los
s: 0.1189 - val_acc: 0.9648
```

After the completion of training of the data set, performance of the model is evaluated using test data set, as well as accuracy and loss of data set is achieved.



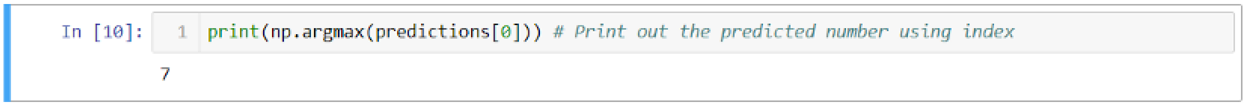
```
In [8]:  # Evaluate the model performance
         test_loss, test_acc = model.evaluate(x=x_test, y=y_test)
         # Print out the model accuracy
         print('\nTest Accuracy:', test_acc)
         print('\nTest Loss:', test_loss)

10000/10000 [==============================] - 0s 35us/sample - loss: 0.1084 - acc: 0.9687

Test Accuracy: 0.9687

Test Loss: 0.10838432838106528
```

After evaluation of model, prediction of the model is made using test data set.



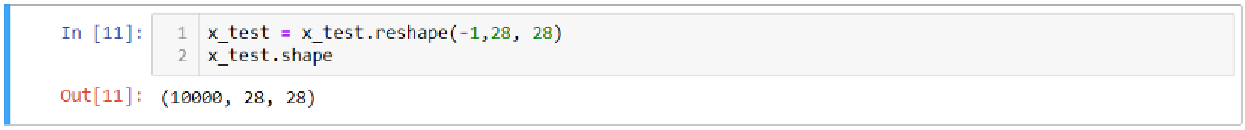
```
In [9]:  predictions = model.predict([x_test]) # Make prediction using test data set
```

Then, predicted data is printed and displayed using the index of the array.



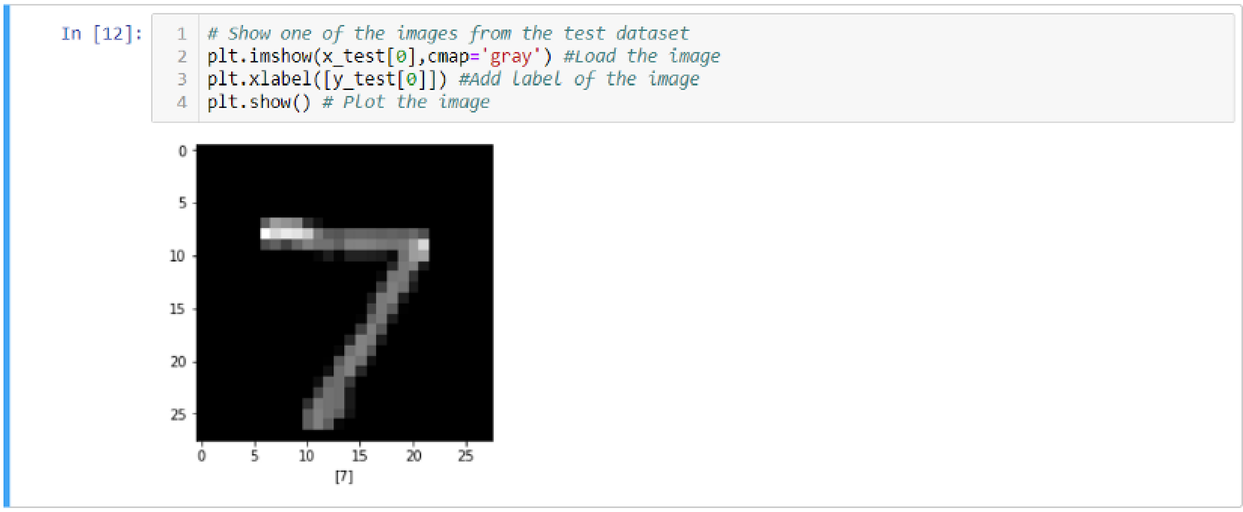
```
In [10]:  print(np.argmax(predictions[0])) # Print out the predicted number using index

7
```

After the test data of the model is predicted, all available test data set (Image) in the trained model is reshaped to width and height of 28 respectively.



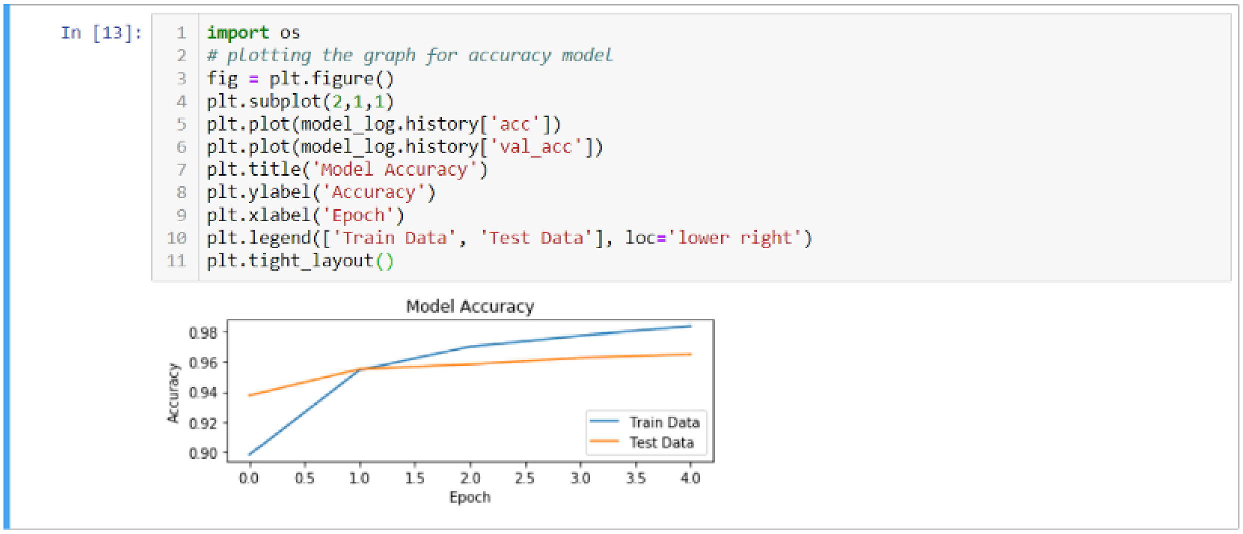
```
In [11]:  x_test = x_test.reshape(-1,28, 28)
          x_test.shape
Out[11]:  (10000, 28, 28)
```

After the image is re shaped, one of the images followed with its label from the test data set is displayed using matplotlib library.



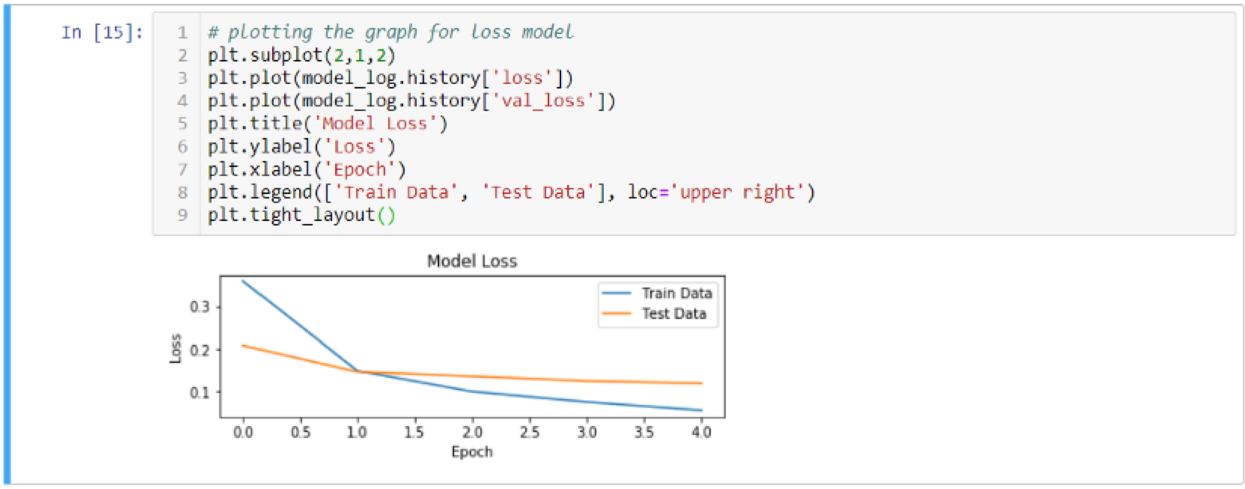
```
In [12]:  # Show one of the images from the test dataset
          plt.imshow(x_test[0],cmap='gray') #Load the image
          plt.xlabel([y_test[0]]) #Add label of the image
          plt.show() # Plot the image
```

Then, graph is plotted to measure accuracy of the built model using train and test dataset.



```
In [13]:  import os
          # plotting the graph for accuracy model
          fig = plt.figure()
          plt.subplot(2,1,1)
          plt.plot(model_log.history['acc'])
          plt.plot(model_log.history['val_acc'])
          plt.title('Model Accuracy')
          plt.ylabel('Accuracy')
          plt.xlabel('Epoch')
          plt.legend(['Train Data', 'Test Data'], loc='lower right')
          plt.tight_layout()
```

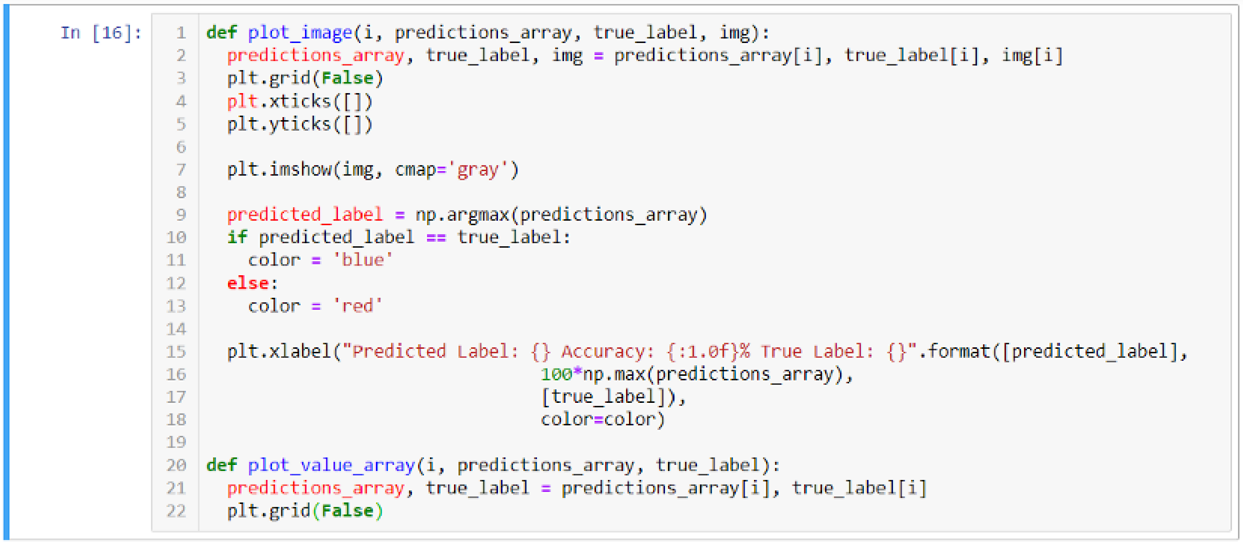
Then, graph is also plotted to measure loss of the built model using train and test dataset.



In [15]:

```
# plotting the graph for loss model
plt.subplot(2,1,2)
plt.plot(model_log.history['loss'])
plt.plot(model_log.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train Data', 'Test Data'], loc='upper right')
plt.tight_layout()
```

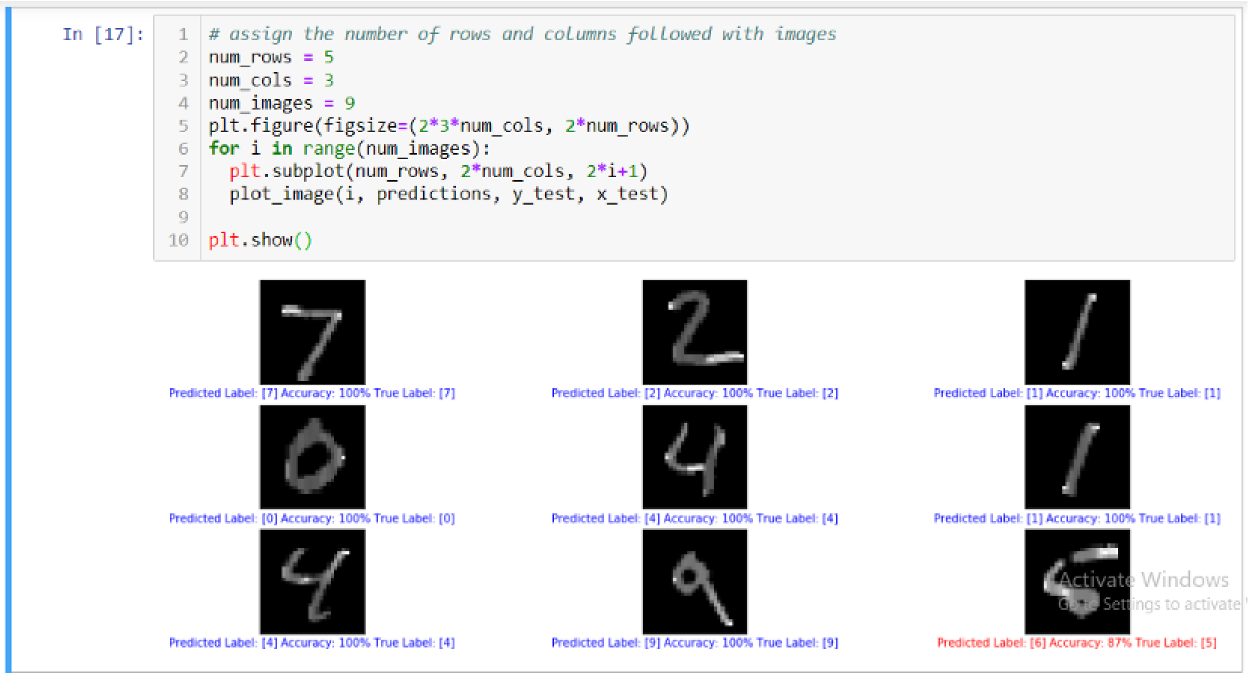
After that, image is plotted in gray scale view followed with their predicted label, true label and accuracy. Color is also defined in the class to segregate the expected result outcome.



In [16]:

```
def plot_image(i, predictions_array, true_label, img):
  predictions_array, true_label, img = predictions_array[i], true_label[i], img[i]
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])

  plt.imshow(img, cmap='gray')

  predicted_label = np.argmax(predictions_array)
  if predicted_label == true_label:
    color = 'blue'
  else:
    color = 'red'

  plt.xlabel("Predicted Label: {} Accuracy: {:1.0f}% True Label: {}".format([predicted_label],
                                100*np.max(predictions_array),
                                [true_label]),
                                color=color)

def plot_value_array(i, predictions_array, true_label):
  predictions_array, true_label = predictions_array[i], true_label[i]
  plt.grid(False)
```

Since, the data set is too huge, it will be difficult to display the whole data set images. So that, number of rows and columns is assigned to display the limited images from the data sets.



```
# assign the number of rows and columns followed with images
num_rows = 5
num_cols = 3
num_images = 9
plt.figure(figsize=(2*3*num_cols, 2*num_rows))
for i in range(num_images):
  plt.subplot(num_rows, 2*num_cols, 2*i+1)
  plot_image(i, predictions, y_test, x_test)

plt.show()
```

```
Predicted Label: [7] Accuracy: 100% True Label: [7]    Predicted Label: [2] Accuracy: 100% True Label: [2]    Predicted Label: [1] Accuracy: 100% True Label: [1]
Predicted Label: [0] Accuracy: 100% True Label: [0]    Predicted Label: [4] Accuracy: 100% True Label: [4]    Predicted Label: [1] Accuracy: 100% True Label: [1]
Predicted Label: [4] Accuracy: 100% True Label: [4]    Predicted Label: [9] Accuracy: 100% True Label: [9]    Predicted Label: [6] Accuracy: 87% True Label: [5]
```

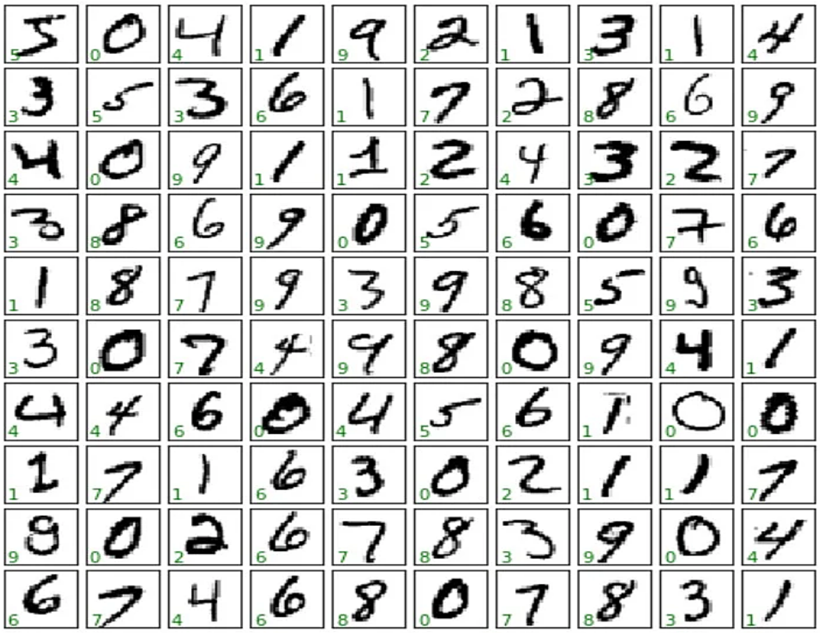
8. TESTING

8.1 Test Cases

Handwritten digit recognition using MNIST dataset is a major project made with the help of Neural Network. It basically detects the scanned images of handwritten digits.We have taken this a step further where our handwritten digit recognition system not only detects scannedd images of handwritten digits but also allows writing digits on the screen with the help of an integerated GUI for recognition.

Handwritten digit recognition using MNIST dataset is a major project made with the help of Neural Network. It basically detects the scanned images of handwritten digits.

We have taken this a step further where our handwritten digit recognition system not only detects scanned images of handwritten digits but also allows writing digits on the screen with the help of an integrated GUI for recognition._



8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

Even though various models with Dropouts applied to few different CNN layers produced similar results, actual performance on the test sheet scoring differed from one another in a complementary way. Due to this complementary performance, I decided to use an ensemble of the three top performing models and pick the maximum score across the predictions to arrive at an ensemble prediction. The ensemble model had slightly higher performance in the real-world than any of the individual models.

Below is a sample output from the automation using the ensemble model. Digits in green are shown to display the resulting numbers after grabbing the individual digits, running them through the CNN model, and combining them into a multi-digit number. Note that in some cases the final number ends up with additional digits because additional contours might have been identified while pre-processing the answer cell image. Even though the model provides high precision during training, it under performs in the real-world scenario.

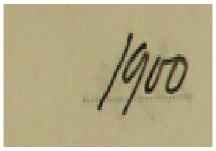
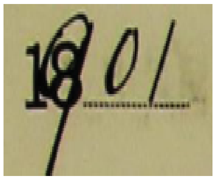
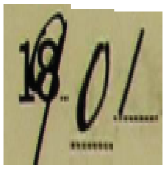
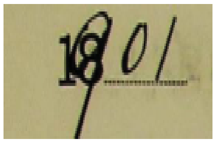
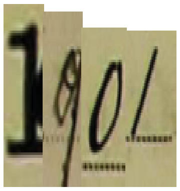
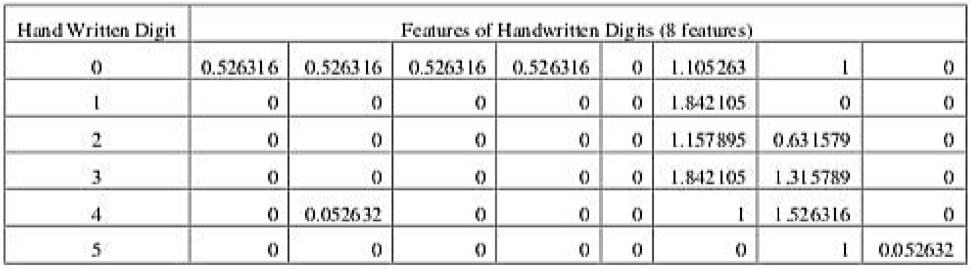
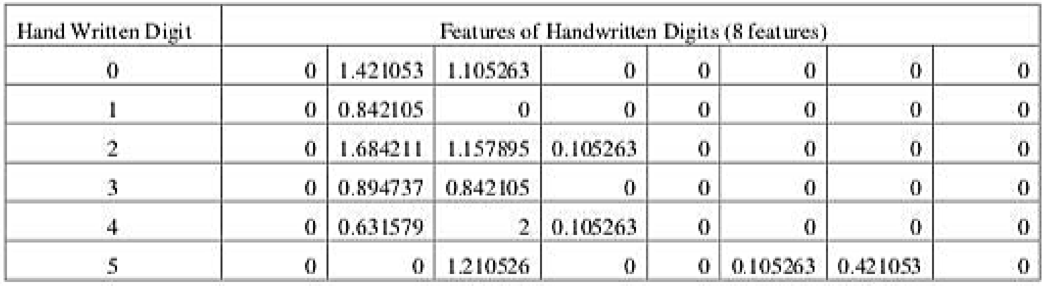
Correct segmentation and recognition		1900		1900
Incorrect segmentation		1901		1901
Incorrect recognition		1901		1701

Table 1: Features of Handwritten Digits 0 to 5



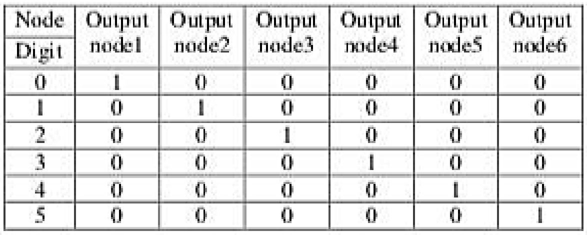
Hand Written Digit	Features of Handwritten Digits (8 features)							
0	0.526316	0.526316	0.526316	0.526316	0	1.105263	1	0
1	0	0	0	0	0	1.842105	0	0
2	0	0	0	0	0	1.157895	0.631579	0
3	0	0	0	0	0	1.842105	1.315789	0
4	0	0.052632	0	0	0	1	1.526316	0
5	0	0	0	0	0	0	1	0.052632

Table 2: Features of Handwritten Digits 0 to 5



Hand Written Digit	Features of Handwritten Digits (8 features)							
0	0	1.421053	1.105263	0	0	0	0	0
1	0	0.842105	0	0	0	0	0	0
2	0	1.684211	1.157895	0.105263	0	0	0	0
3	0	0.894737	0.842105	0	0	0	0	0
4	0	0.631579	2	0.105263	0	0	0	0
5	0	0	1.210526	0	0	0.105263	0.421053	0

Table 3: Target vector for Hand written digits 0 to 5



Node Digit	Output node1	Output node2	Output node3	Output node4	Output node5	Output node6
0	1	0	0	0	0	0
1	0	1	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	0	0
4	0	0	0	0	1	0
5	0	0	0	0	0	1

10. ADVANTAGES

- The system not only produces a classification of the digit but also a rich description of the instantiation parameters which can yield information such as the writing style The generative models can perform recognition driven segmentation
- A paper based form are often became an electronic form which is straightforward to store or send by mail.
- It is cheaper than paying someone amount to manually enter great deal of text data. Moreover it takes less time to convert within the electronic form.
- It can be used by countless organization, schools, banks and even for family activities.

- Handwritten digit focus will be beneficial for government bodies or any different organization to identify citizenship identification range which helps in automation.
- Likewise, license card quantity of any individual can be diagnosed thru this system.
- Similarly,postal addresses, bank cheque digit consciousness can be made less complicated thru automation the usage of this system.

DISADVANTAGES

- Performance of Recognizing system depends on many factors including high accuracy, low runtime and low memory requirements.
- Larger recognizers in turn require larger training sets.
- There is the need of lot of space required by the image produced.
- The quality of the image can be lose during this process.
- Quality of the ultimate image depends on quality of the first image.
- All the documents got to be checked over carefully then manually corrected.
- Not 100% accurate, there are likely to be some mistakes made during the method.
- Not worth doing for little amounts of text.

11. CONCLUSION

Digit recognition is vital. It is capable of solving increasingly difficult problems and making humans' jobs easier. This is a worldwide system for recognizing zip codes or postal codes for mail sorting. Handwritten digit recognition can be accomplished using a variety of approaches. The machine has a difficult duty because handwritten digits are not flawless and can be generated with a variety of flavors. The solution to this issue is handwritten digit recognition, which uses an image of a digit and identifies the digit represented in the image.Numerical digits being written throughout world have different writing styles which can be recognized with handwritten recoginization systems using proper algorithm and strategies. We have learning for recognition of digits. It has been found that recognition of handwritten digits becomes difficult due to presence of similarity in shapes for multiple digits. Scanned image is pre-processed to get a cleaned image and the digits are isolated into individual characters. Preprocessing work is done in which normalization, filtration is performed using preprocessing steps which produce noise free and clean output. Managing our evolution algorithm with proper training, evaluation other step wise process will lead to successful output of system with better efficiency. Use of some statistical features and geometric features through neural network will provided better recognition result of numerical digits.

12. FUTURE SCOPE

This work further extended to the digits recognition for different handwritting styles throughot the world. It can be used to convert the fax and news papers into text format. In order to recognize words, sentences or paragraphs we can use multiple ANN for classification. It can be used in post office for reading postal address.A revenue model w startup can make money. The major revenue sources can consist of sales, government funds, and public donations. The introduction of novel ideas increases revenue streams, such as introducing gesture or touch features, voice read out of recognised digits, etc.. which in turn reduces the work of human and precise action by computers are made for future world.

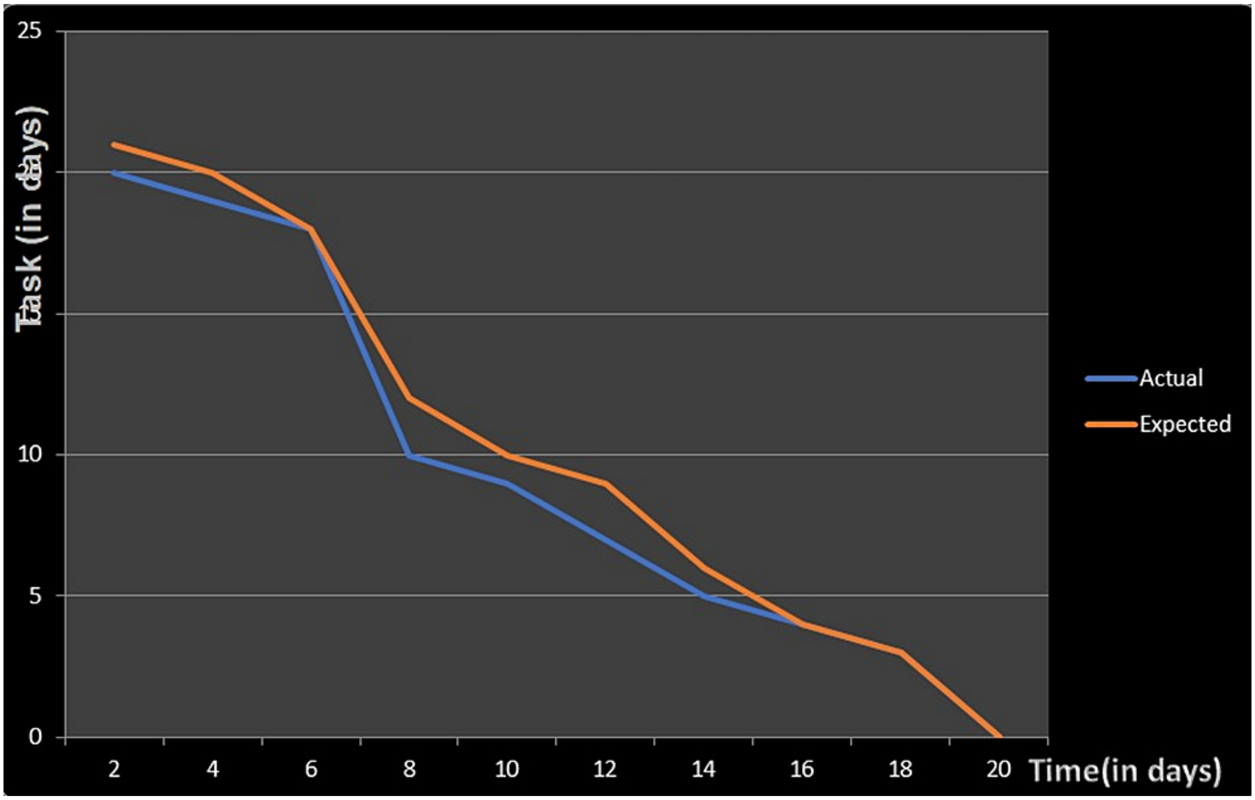
13. APPENDIX

13.1.Testing Table

Functional Requirement (Epic)	**User Story / Task**	**Acceptance criteria**	**Remark**
Registration	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	Pass
Confirmation	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	Pass
Login	As a user, I can login into the application through email & password	I can login into the application through email & password	Pass

Upload	As a user I can upload the handwritten digit	I can upload my image	Pass
Result	As a user I can get the recognized digit	I can view the output image	Pass
Dashboard	As a user I can access the dashboard and send queries	I can send the queries and receive	Pass

13.2 Burndown Chart


Task (in days)
Time(in days)
Actual
Expected