An Automated Optical Character Recognition of Handwritten English Letters using.

Decision Trees & Random Forests

١- Introduction and Overview

- Project idea in detail:

Optical character recognition or optical character reader (OCR) is the automated conversion of images of typed, handwritten, or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (e.g., the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (e.g., from a television broadcast). The goal of a character recognition system is to transform handwritten text documents on paper into a digital format that can be manipulated by word processor software. The system is required to identify a given input character form by mapping it to a single character in each character set.

- Similar applications in the market:

١- Evernote:

his app can quickly capture all your handwritten ideas and notes with Evernote's built in camera on Android and Apple devices. It was already possible to digitize texts in Evernote with the help of the Penultimate app, but an update also allows it to be done from your own application thanks to the Evernote Scannable function. Among its various options, it allows you to digitize the text you find, expanding the possibilities and being a perfect ally for, for example, passing notes between colleagues.

٢- Pen to print:

his app can quickly capture all your handwritten ideas and notes with Evernote's built in camera on Android and Apple devices. It was already possible to digitize texts in Evernote with the help of the Penultimate app, but an update also allows it to be done from your own application thanks to the Evernote Scannable function. Among its various options, it allows you to digitize the text you find, expanding the possibilities and being a perfect ally for, for example, passing notes between colleagues.

٣- Write for iPad:

If you prefer to write longhand but need to see your text in digital format, consider Write Pad for iPad. You can configure a host of options to recognize input forms and predefined commands, or you can input lettering with your finger or a stylus.

-A Literature Review of Academic publications(papers/books/articles) relevant to the problem

١)

Optical Character Recognition (OCR) is a piece of software that converts printed text and images into digitized form such that it can be manipulated by machine. Unlike the human brain which has the capability to recognize the text/ characters very easily from an image, machines are not intelligent enough to perceive the information available in image. Therefore, many research efforts have been put forward that attempts to transform a document image to format understandable for machine.

OCR is a complex problem because of the variety of languages, fonts and styles in which text can be written, and the complex rules of languages etc. Hence, techniques from different disciplines of computer science (i.e. image processing, pattern classification and natural language processing etc. are employed to address different challenges. This paper introduces the reader

to the problem. It enlightens the reader with the historical perspectives, applications, challenges, and techniques of OCR.

TYPES OF OPTICAL CHARACTER RECOGNITION SYSTEMS

There has been multitude of directions in which research on OCR has been carried out during past years. This section discusses different types of OCR systems that have emerged because of this research. We can categorize these systems based on image acquisition mode, character connectivity, font-restrictions etc. Fig. ١ categorizes the character recognition system.

Based on the type of input, the OCR systems can be categorized as handwriting recognition and machine printed character recognition. The former is a relatively simpler problem because characters are usually of uniform dimensions, and the positions of characters on the page can be predicted [٣].

Handwriting character recognition is a very tough job due to the different writing style of the user as well as different pen movements by the user for the same character. These systems can be divided into two sub-categories i.e. on-line and off-line systems. The former is performed in real-time while the users are writing the character. They are less complex as they can capture the temporal or time-based information i.e. speed, velocity, number of strokes made, direction of writing of strokes etc. In addition, there no need for thinning techniques as the trace of the pen is few pixels wide. The offline recognition systems operate on static data i.e. the input is a bitmap. Hence, it is very difficult to perform recognition.

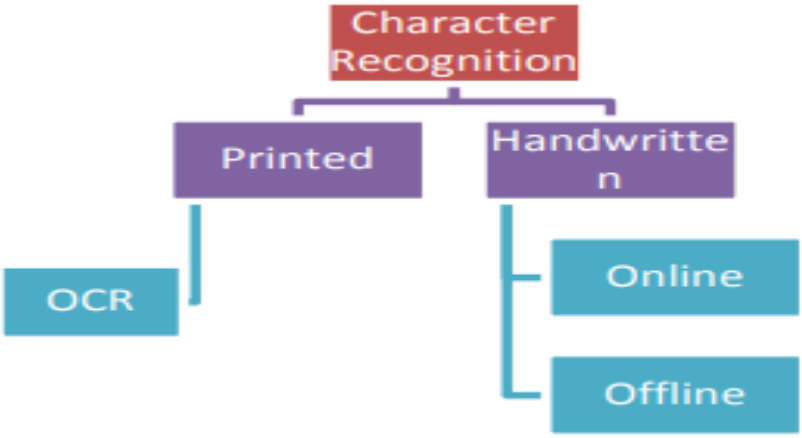


Figure.1: Types of character recognition system

---

٢)

Three classifications of optical reader machines are normally used:

- Mark Readers
- Bar Code Readers
- Character Readers

In some usage the term OCR is applied to systems of all three types. However, OCR is more correctly used for character readers only, though in some instances, systems possess all three capabilities.

How OCR Works:

١. **Image Acquisition:**

- OCR starts with the acquisition of an image, which can be a scanned document, a photo of a document, or a PDF file.

٢. **Preprocessing:**

- The image may undergo preprocessing steps, such as noise reduction, binarization (converting to black and white).

٣. **Text Detection:**

- OCR algorithms locate areas in the image where text is present. This involves identifying lines, paragraphs, and individual characters.

٤. **Character Recognition:**

- OCR recognizes individual characters within the detected text regions. This process involves pattern recognition and machine learning algorithms.

٥. **Text Post-processing:**

- The recognized characters are often post-processed to correct errors and improve accuracy. This may involve dictionary-based corrections, language modeling, or contextual analysis.

٦. **Output:**

- The final output is usually in a text format (e.g., TXT, DOC, PDF with selectable text), allowing users to edit, search, and manipulate the content.

Resources

١. **First s:**

- Journal of Information & Communication Technology-JICT Vol. ١٠ Issue. ٢, December ٢٠١٦ (A Survey on Optical Character Recognition System) Noman Islam, Zeeshan Islam, Nazia Noor. https://arxiv.org/abs/١٧١٠,٠٥٧٠٣

٢. **Second s:**

- An Overview of Optical Character Recognition (OCR) Theology and Techniques. https://apps.dtic.mil/sti/citations/ADA١٣١٣٤١

٣. **Third s:**

- Google scholar.

  https://scholar.google.com/scholar?hl=ar&as_sdt=٠٪٢C٥&q=Over+view+of+OCR&btnG=

٤. **Fourth s:**

- Using Random Forests for Handwritten Digit Recognition. https://www.researchgate.net/publication/٤٢٨٨٢٢١_Using_Random_Forests_for_Handwritten_Digit_Recognition

٥. **Fifth s:**

- Preprocessing The image resource: Neural_network_EMNIST | Kaggle

٢-Proposed solution & dataset:

- Main Functionalities

١- load image from dataset.

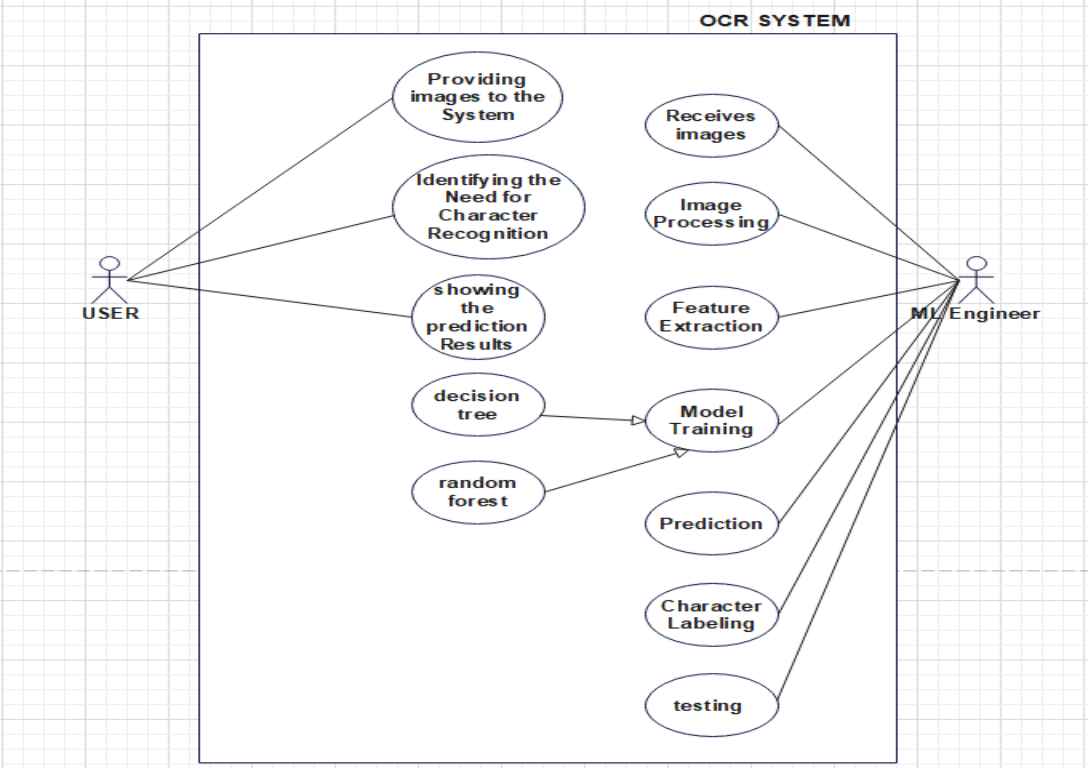
٢- Image Processing.

٣- Training Model.

٤- testing Model.

٥- Show the results and the accuracy of testing model.

٦-predict an image from dataset.



- Dataset:

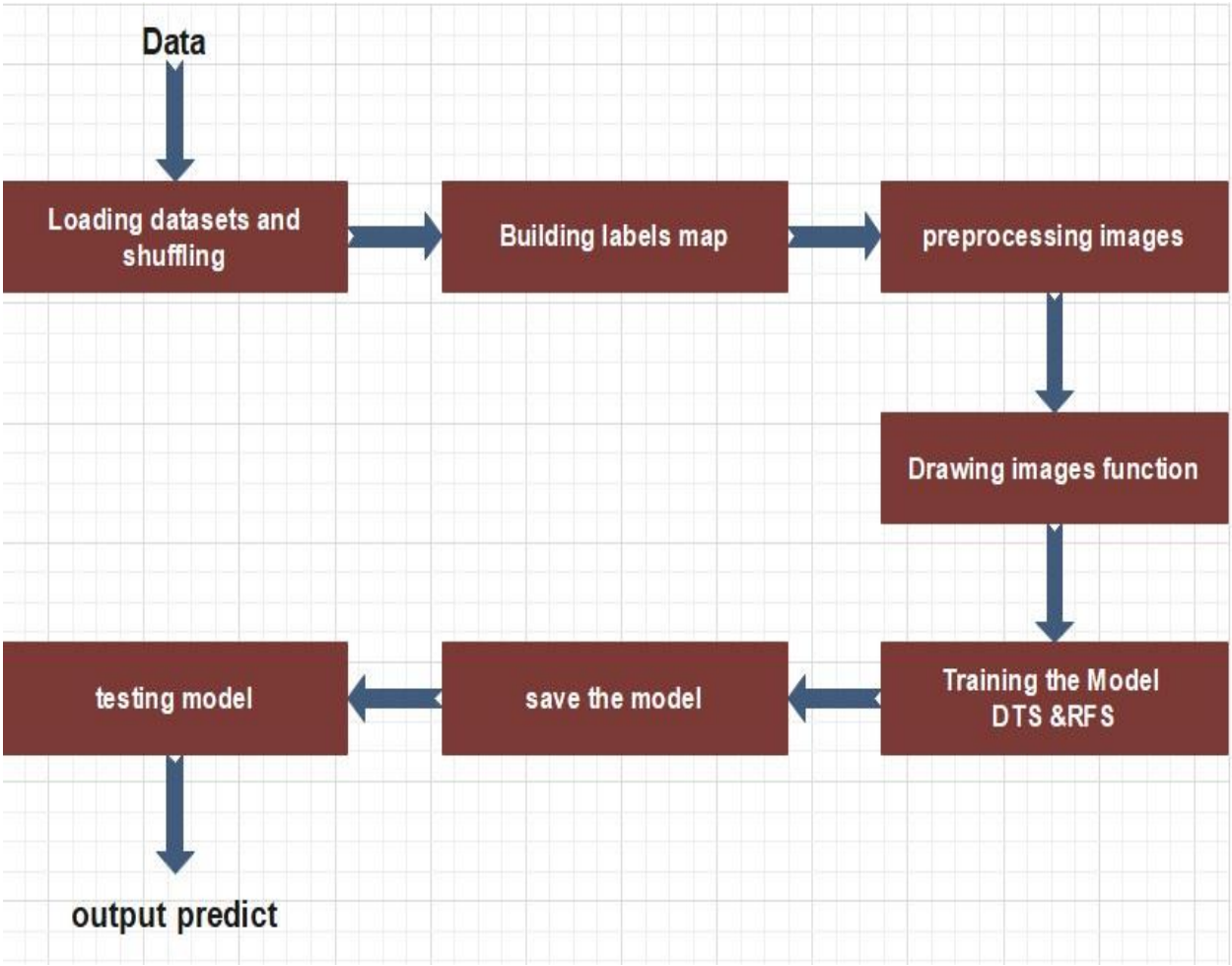
In Kaggle: https://www.kaggle.com/crawford/emnist

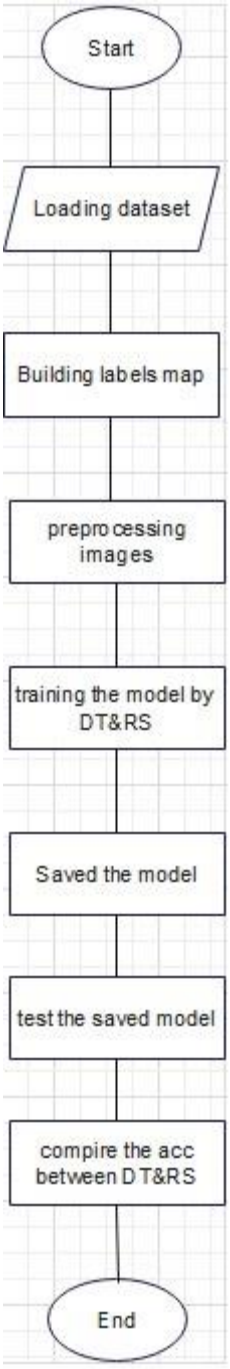
- Github Repo:

SeifMohamed٥٥/AI_ML_DecisionTree_RF: Decison Tree and Random Forest models done on MINST English letters Dataset (github.com)

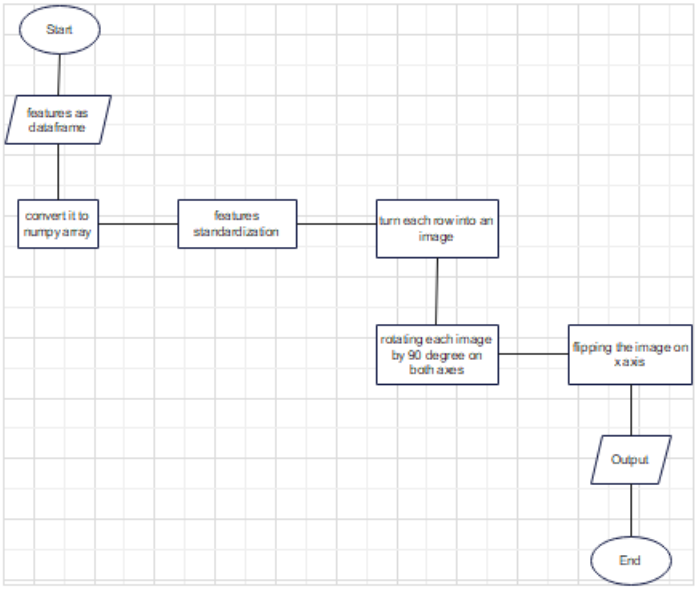
٣- Applied Algorithms:

-Block diagram





Flowchart Of Our Project



٤- Experiments & Results:

Experiments: **We tried the decision tress algorithm and random forests algorithm. The decision tress algorithm was ٧٠,٧١% accuracy in the model, and the random forests algorithm was ٨٢,٢٪accuracy.**

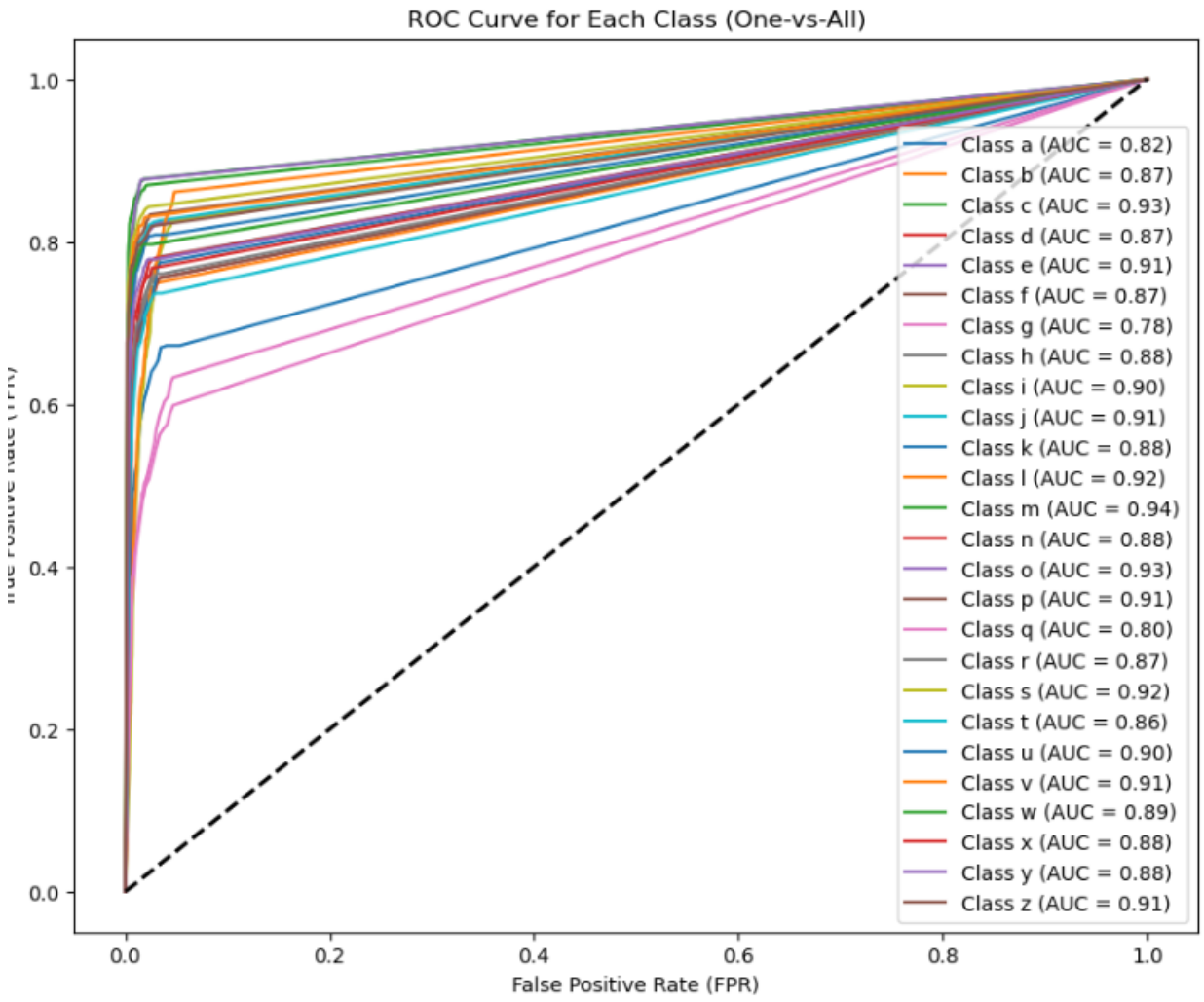
Time: **decision tree model took about ٤ hours of training because of the grid_search done on the parameters then we took those parameters and used them at Random forest. Random Forest with n_estimators = ١٠٠ took about an hour and a half with cross validation folds n_splits = ٣ and with n_estimators = ٢٠٠ took about two hours and so little difference in cross validated accuracy :**

RF_١٠٠ accuracy = ٨١,٨ % , RF_٢٠٠ accuracy = ٨٢,٢ %

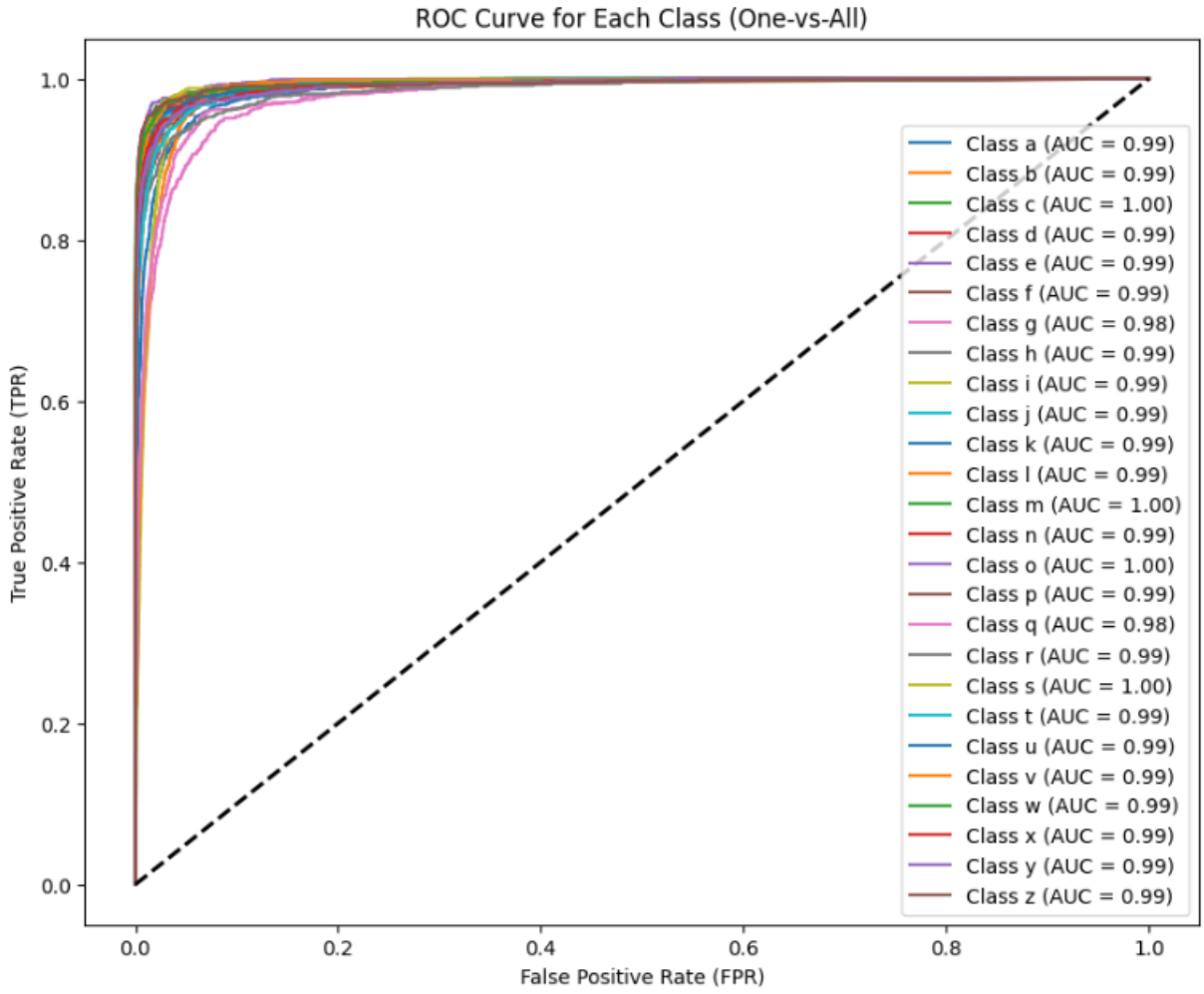
Params used to search for the best DecisionTree hyperparametrs

```
# param_grid = {
#     'criterion': ['gini', 'entropy'],
#     'splitter': ['best', 'random'],
#     'max_depth': [None, ١٠, ٢٠, ٣٠],
#     'min_samples_split': [ ٢, ٥, ١٠],
#     'min_samples_leaf': [ ١, ٢, ٤],
#     'max_features': [None, 'sqrt', 'log ٢']
# }
# kf = KFold(n_splits= ٢, shuffle=True, random_state= ٤ ٢)
# classifier = DecisionTreeClassifier(random_state= ٤ ٢)
# grid_search = GridSearchCV(classifier, param_grid, cv=kf, scoring='accuracy')
```

Decision Tree Merged Dataset ROC Curve:



Random Forest Merged Dataset ROC Curve:



٥- Analysis, Discussion and Future work:

-By analyzing the results of both algorithms, we got insights about the efficiency of both algorithms then we found that performance of results of Random Forest algorithm is better than Decision tree.

- Random Forest gives overall better performance but needs more computational power. It consists of a collection of decision trees to improve its performance

- Future work: Adding functionality to generalize on any handwritten English charachters

٦- Development Platform:

Tools: Jupyter Notebook, VScode

Programming Languages: Python.

Python Libraries:

- Pandas **=>** Dealing with csv files.
- NumPy => Dealing with arrays.
- sklearn => Used Models.
- joblib => saving models.

- Github Repo:

SeifMohamed٥٥/AI_ML_DecisionTree_RF: Decison Tree and Random Forest models done on MINST English letters Dataset (github.com)