CS361 ARTIFICIAL INTELLIGENCE FALL 2022

COURSE PROJECT INSTRUCTIONS

Instructions to Students:

- This is a group work project. Each group consists of **Six to Seven students** "General Program" / **Four to Five** "Med. Info. Program" *(group members must be approved by the Teaching Assistant through registration)*. Each group must develop the idea assigned to them using Python.

    - **Project Objectives:** The objectives of this project can be summarized as applying the main ideas, fundamental concepts, and key algorithms in the fields of artificial intelligence, and machine learning.

- **Submission:** Submission is done according to the following schedule:
    - **Week 7:** During the Weekly Labs.
        - *Registration: Every team leader should register his/her team according to the regulations/steps announced by the TAs during the labs.*
    - **Week 9 / 10:** Submission and Discussion of a (1) Documentation. *The report should include the following: (1) Project idea in details, (2) Main functionalities, (3) Similar applications in the market, (4) An initial literature review of Academic publications (papers) relevant to the idea (at least 2 papers), (5) the Dataset employed (preferably a publicly available dataset), (6) Details of the algorithm(s)/approach(es) that will be used.*
    - **Week 13:** Submission and Discussion of the (1) Project, and (2) Documentation. *The report should include the following: (1) Project idea in details, (2) Main functionalities, (3) Similar applications in the market, (4) A literature review of Academic publications (papers) relevant to the idea (at least 5 papers), (5) the Dataset employed (preferably a publicly available dataset), (6) Details of the algorithm(s)/approach(es) used and the results of the experiments, and (7) Development platform.*

- **Assessment:** Assessment will be on the reports and code submitted, in addition to discussions with team members. All the team members must contribute to all the phases, and the role of each member must be clearly stated in each report.
    - The Project will be assessed based on the following criteria:
        - The complexity of the problem, & the correctness of the algorithms employed.
        - The quality/comprehensiveness of your experiments & documentation.
        - The correctness of your analysis and design diagrams.
        - Implementation correctness.

- **Feedback:** Further details & Feedback could be provided for each group – if requested – through discussions with the teaching assistant(s) during the weekly-labs./office-hours.

- You can only submit your own work. Any student suspected of plagiarism will be subject to the procedures set out in by the Faculty/University (including failing the course entirely).

- **Academic Integrity:** The University's policies on academic integrity will be enforced on students who violate University standards of academic integrity. Examples of behaviour that is not allowed are:
  - Copying all or part of someone else's work and submitting it as your own;
  - Giving another student in the class a copy of your work; and
  - Copying parts from the internet, textbooks, etc.
  - If you have any questions concerning what is allowed, please don't hesitate to discuss with me.

CS361 Artificial Intelligence Fall 2022

Course Project Descriptions (28 Ideas)

Projects' Description:

1) N-Queens Problem Solver *(for different sizes – n should be selected by the user)* using Differential Evolution AND the Backtracking Algorithm.

2) A Sudoku Solver using Differential Evolution AND the Backtracking Algorithm.

3) An Intelligent Connect-Four Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*.

4) An Intelligent Checkers Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*.

5) An Intelligent Chess Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*. + code

6) An Intelligent Go Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*.

7) An Intelligent N-Puzzle Solver (for sizes: 8, 15, and 24) using a Best-First Search algorithm *(designing & implementing at least 4 heuristic functions)*.

8) Solving the Nurse Scheduling Problem using Genetic Algorithms.

9) Solving the Nurse Scheduling Problem using Differential Evolution.

10) Solving a Faculty's Timetable Scheduling Problem using Genetic Algorithms.

11) Solving a Faculty's Timetable Scheduling Problem using Differential Evolution.

12) Solving the VRP "Vehicle Routing Problem" using both Genetic Algorithms & Differential Evolution.

13) Solving the Knapsack Problem using both Genetic Algorithms & Differential Evolution (Solve both the 0-1 Knapsack Problem and the Unbounded Knapsack Problem).

14) An Automated Optical Character Recognition of Handwritten English Letters using Artificial Neural Networks.

15) An Automated Optical Character Recognition of Handwritten English Letters using Decision Trees & Random Forests.

16) An Automated Optical Character Recognition of Handwritten Arabic Numerals/Digits using Artificial Neural Networks.

17) An Automated Optical Character Recognition of Handwritten Arabic Numerals/Digits using Decision Trees & Random Forests.

18) Biometrics: Automated Face Recognition using Artificial Neural Networks.

19) Biometrics: Automated Face Recognition using Decision Trees & Random Forests.

20) Automated Facial Expression Recognition using Artificial Neural Networks.

21) Automated Object Detection using Artificial Neural Networks.

22) Automated Object Detection using Decision Trees & Random Forests.

23) Biometrics: Automated Iris Recognition using Artificial Neural Networks.

24) Biometrics: Automated Fingerprint Recognition using Artificial Neural Networks.

25) Automatic Document Classification / Categorization by Subject.

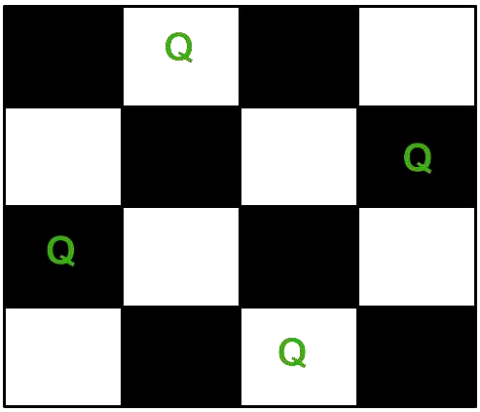
26) Automatic Language Identification using K-means Clustering.

27) Biometrics: A Voiceprint (Speaker Verification or Identification) System.

28) A Recommender System using both Collaborative Filtering & Content-based Filtering.

1) N-Queens Problem Solver *(for different sizes – n should be selected by the user)* using Differential Evolution AND the Backtracking Algorithm.

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. The user should select the number of queens (N). The standard form is the Eight Queens Puzzle, in which one must place eight queens on a standard chessboard such that no queen is attacking any other (i.e. no two queens occupy the same row, column or diagonal). For example, following is a solution for 4 Queen problem:



2) A Sudoku Solver using Differential Evolution AND the Backtracking Algorithm.

Sudoku is a logic-based, combinatorial number-placement puzzle. The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 sub-grids that compose the grid (also called "boxes", "blocks", or "regions") contain all the digits from 1 to 9. The puzzle setter provides a partially completed grid, which for a well-posed puzzle has a single solution. For example, the (left) figure demonstrates a typical Sudoku puzzle, and its solution (right).

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

3) An Intelligent Connect-Four Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*

Build an artificial Connect-Four player, that can play games against a human opponent. Connect Four *(also known as Four Up, Plot Four, Find Four, Captain's Mistress, Four in a Row, Drop Four, and Gravitrips in the Soviet Union)* is a two-player connection board game, in which the players choose a color and then take turns dropping colored discs into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the lowest available space within the column. The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own discs. Connect Four is a solved game. The first player can always win by playing the right moves.

4) An Intelligent Checkers Player using an Alpha-Beta Depth-First algorithm (designing & implementing at least 2 heuristic functions)

Build an artificial Checkers player, that can play games against a human opponent. Checkers (or Draughts in British English) is a group of strategy board games for two players which involve diagonal moves of uniform game pieces and mandatory captures by jumping over opponent pieces. Checkers/Draughts developed from alquerque (a.k.a. Qirkat from Arabic: القرقات - a strategy board game that have originated in the Middle East). The most popular form of the game is the American checkers (also called English draughts), played on an 8×8 checkerboard.

The General Rules of the Game: Checkers is played by two opponents, on opposite sides of the gameboard. One player has the dark pieces; the other has the light pieces. Players alternate turns. A player may not move an opponent's piece. A move consists of moving a piece diagonally to an adjacent unoccupied square. If the adjacent square contains an opponent's piece, and the square immediately beyond it is vacant, the piece may be captured (and removed from the game) by jumping over it. Only the dark squares of the checkered board are used. A piece may move only diagonally into an unoccupied square. When presented, capturing is mandatory in most official rules and where the player does not capture the opposing player can remove the opponent piece as penalty (or muffin) and where there are two or more such positions the player forfeits that/those he cannot move. Although some rule variations make capturing optional. In almost all variants, the player without pieces remaining, or who cannot move due to being blocked, loses the game.

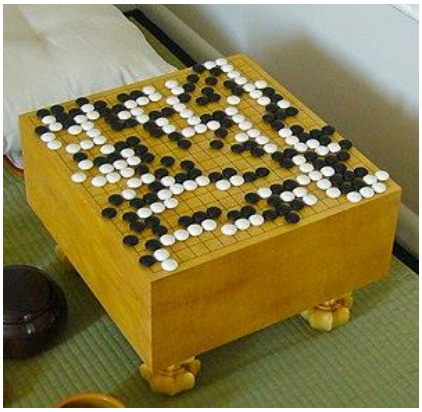
5) An Intelligent Chess-Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*

Build an artificial Chess player, that can play games against a human opponent. Chess is a two-player strategy board game played on a chessboard, a checkered game-board with 64 squares arranged in an 8×8 grid. The game is played by millions of people worldwide. Chess is believed to be derived from the Indian game chaturanga sometime before the 7th century. Chess reached Europe by the 9th century, due to the Umayyad conquest of Hispania. The pieces assumed their current powers in Spain in the late 15th century with the introduction of "Mad Queen Chess"; the modern rules were standardized in the 19th century. Play does not involve hidden information. Each player begins with 16 pieces: one king, one queen, two rooks, two knights, two bishops, and eight pawns. Each of the six piece types moves differently, with the most powerful being the

queen and the least powerful the pawn. The objective is to checkmate the opponent's king by placing it under an inescapable threat of capture. To this end, a player's pieces are used to attack and capture the opponent's pieces, while supporting each other. During the game, play typically involves making exchanges of one piece for an opponent's similar piece, but also finding and engineering opportunities to trade advantageously, or to get a better position. In addition to checkmate, a player wins the game if the opponent resigns, or (in a timed game) runs out of time. There are also several ways that a game can end in a draw.

6) An Intelligent Go-Player using an Alpha-Beta Depth-First algorithm *(designing & implementing at least 2 heuristic functions)*

Build an artificial Go player, that can play games against a human opponent. Go is an abstract strategy board game for two players, in which the aim is to surround more territory than the opponent. The game was invented in China more than 2,500 years ago and is believed to be the oldest board game continuously played to the present day. The playing pieces are called stones. One player uses the white stones and the other, black. The players take turns placing the stones on the vacant intersections ("points") of a board. Once placed on the board, stones may not be moved, but stones are removed from the board if "captured". Capture happens when a stone or group of stones is surrounded by opposing stones on all orthogonally-adjacent points. The game proceeds until neither player wishes to make another move. When a game concludes, the winner is determined by counting each player's surrounded territory along with captured stones and komi *(points added to the score of the player with the white stones as compensation for playing second)*. The standard Go board has a 19×19 grid of lines, containing 361 points. Beginners often play on smaller 9×9 and 13×13 boards.



7) An Intelligent N-Puzzle Solver (for sizes: 8, 15, and 24) using a Best-First Search algorithm (designing & implementing at least 4 heuristic functions).

The 15-puzzle (also called Game of Fifteen, Mystic Square and many others) is a sliding puzzle that consists of a frame of numbered square tiles in random order with one tile missing. The puzzle also exists in other sizes, particularly the smaller 8-puzzle. If the size is 3×3 tiles, the puzzle is called the 8-puzzle or 9-puzzle, and if 4×4 tiles, the puzzle is called the 15-puzzle or 16-puzzle named, respectively, for the number of tiles and the number of spaces. The object of the puzzle is to place the tiles in order by making sliding moves that use the empty space.

The n-puzzle is a classical problem for modelling algorithms involving heuristics. Commonly used heuristics for this problem include counting the number of misplaced tiles and finding the sum of the taxicab-distances between each block and its position in the goal configuration.

8 *and* 9) Solving the Nurse Scheduling Problem using [idea 8] Genetic Algorithms, or [idea 9] Differential Evolution.

The nurse scheduling problem (NSP), also called the nurse rostering problem (NRP), is the operations research problem of finding an optimal way to assign nurses to shifts, typically with a set of hard constraints which all valid solutions must follow, and a set of soft constraints which define the relative quality of valid solutions. Solutions to the nurse scheduling problem can be applied to constrained scheduling problems in other fields. The nurse scheduling problem involves the assignment of shifts and holidays to nurses. Each nurse has their own wishes and restrictions, as does the hospital. The problem is described as finding a schedule that both respects the constraints of the nurses and fulfills the objectives of the hospital. Conventionally, a nurse can work 3 shifts because nursing is shift work:

• day shift • night shift • late night shift

In this problem we must search for a solution satisfying as many wishes as possible while not compromising the needs of the hospital. There are two types of constraints:

1. Hard constraints: if this constraint fails then the entire schedule is invalid.

2. Soft constraints: it is desirable that these constraints are met but not meeting them does not make the schedule invalid.

Some examples of constraints are:

• A nurse does not work the day shift, night shift and late-night shift on the same day (for obvious reasons).

• A nurse may go on a holiday and will not work shifts during this time.

• A nurse does not do a late-night shift followed by a day shift the next day.

Hard constraints typically include a specification of shifts (e.g. morning, afternoon, and night), that each nurse should work no more than one shift per day, and that all patients should have nursing coverage. Differences in qualifications between nurses also create hard constraints. Soft constraints may include minimum and maximum numbers of shifts assigned to a given nurse in a given week, of hours worked per week, of days worked consecutively, of days off consecutively, and so on. The shift preferences of individual nurses may be treated as a soft constraint, or as a hard constraint.

10 *and* 11) Solving a Faculty's Timetable Scheduling Problem using [idea 10] Genetic Algorithms, or [idea 11] Differential Evolution.

A very famous scenario where Genetic Algorithms can be used is the process of making timetables or timetable scheduling. Consider you are trying to come up with a weekly timetable for classes in a college for a batch/class. We must arrange classes and come up with a timetable so that there are no clashes between classes. Here, the task is to search for the optimum timetable schedule. A possible definition for the problem is: Given a set of lecturers, a set of courses on individual topics and a Course Requirements matrix with integer elements representing the number of hours a lecturer teaches a course during each week, the problem is to allocate times to these hours so that a student may take as many suitable combinations of courses as possible. Or, simply to create a practical timetable for a whole faculty in which courses offered by different departments may be combined in various ways to suit individual students.

12) Solving the VRP "Vehicle Routing Problem" using both Genetic Algorithms & Differential Evolution.

The vehicle routing problem (VRP) is a combinatorial optimization and integer programming problem which asks: "What is the optimal set of routes for a fleet of vehicles to traverse in order to deliver to a given set of customers?". It generalises the well-known travelling salesman problem (TSP). The VRP concerns the service of a delivery company. How things are delivered from one or more depots which has a given set of home vehicles and operated by a set of drivers who can move on a given road network to a set of customers. It asks for a determination of a set of routes, S, (one route for each vehicle that must start and finish at its own depot) such that all customers' requirements and operational constraints are satisfied, and the global transportation cost is minimized. This cost may be monetary, distance or otherwise.

13) Solving the Knapsack Problem using Swarm Intelligence using both Genetic Algorithms & Differential Evolution *(Solve both the 0-1 Knapsack Problem and the Unbounded Knapsack Problem)*.

The knapsack problem or rucksack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. It derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items. The most common problem being solved is the 0-1 knapsack problem, which restricts the number of

copies of each kind of item to zero or one. The unbounded knapsack problem (UKP) places no upper bound on the number of copies of each kind of item.

14 *and* 15) An Automated Optical Character Recognition of Handwritten English Letters using [idea 14] Artificial Neural Networks, or [idea 15] Decision Trees & Random Forests.

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). Build an OCR using Artificial Neural Networks to identify Handwritten English Letters.

You MAY use the following data-set: https://www.kaggle.com/crawford/emnist

16 *and* 17) An Automated Optical Character Recognition of Handwritten Arabic Numerals/Digits using [idea 16] Artificial Neural Networks, or [idea 17] Decision Trees & Random Forests.

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). Build an OCR using Artificial Neural Networks to identify Handwritten Arabic Numerals *(Arabic Numbers: 0, 1, 2, .. 9)*.

You MAY use the following data-sets: https://www.kaggle.com/mloey1/ahcd1 & https://www.kaggle.com/mloey1/ahdd1

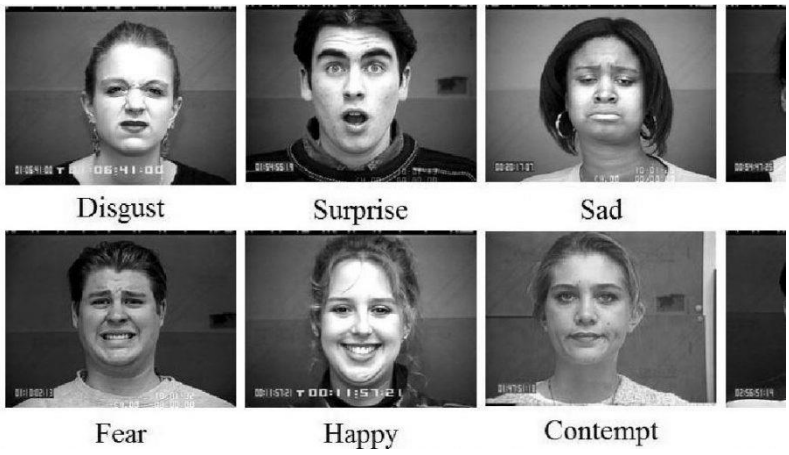
18 *and* 19) Biometrics: Automated Face Recognition using [idea 18] Artificial Neural Networks, or [idea 19] Decision Trees & Random Forests.

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analysing patterns based on the person's facial textures and shape. Build a face recognition system for identifying a person from grayscale digital images using Artificial Neural Networks. You MAY use the following data-set: https://www.kaggle.com/serkanpeldek/face-recognition-on-olivetti-dataset/notebook

20) Automated Facial Expression Recognition using Artificial Neural Networks.

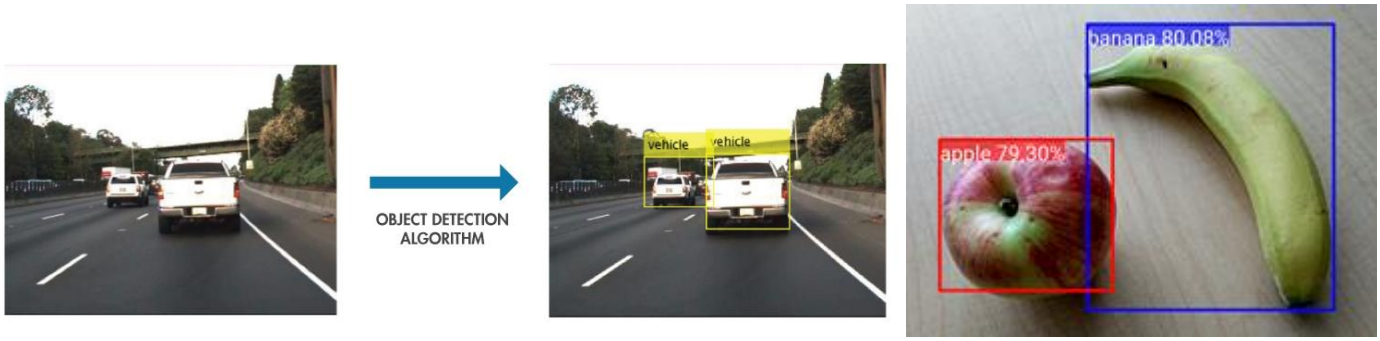
Facial expression recognition is the task of classifying the expressions on face images into various categories such as anger, fear, surprise, sadness, happiness and so on. Emotional facial expressions can inform researchers about an individual's emotional state. You MAY use the following data-sets: FER-2013: Learn facial expressions from an image;

https://www.kaggle.com/msambare/fer2013 OR Facial Expression Recognition on AffectNet: https://paperswithcode.com/sota/facial-expression-recognition-on-affectnet



21 *and* 22) Automated Object Detection using [idea 21] Artificial Neural Networks, or [idea 22] Decision Trees & Random Forests.

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Object detection is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer. Select 5 to 10 different objects and train a Random Forest to detect them.



23) Biometrics: Automated Iris Recognition using Artificial Neural Networks.

Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on video images of one or both irises of an individual's eyes, whose complex patterns are unique, stable, and can be seen from some distance. It is a method of identifying people based on unique patterns within the ring-shaped region surrounding the pupil of the eye. The iris usually has a brown, blue, gray, or greenish color, with complex patterns that are visible upon close inspection. Because it makes use of a biological characteristic, iris recognition is considered a form of biometric verification. Several hundred million persons in several countries around the world have been enrolled in iris recognition systems for convenience purposes

such as passport-free automated border-crossings and some national ID programs. A key advantage of iris recognition, besides its speed of matching and its extreme resistance to false matches, is the stability of the iris as an internal and protected, yet externally visible organ of the eye.

24) Biometrics: Automated Fingerprint Recognition using Artificial Neural Networks.

Fingerprint recognition refers to the automated method of identifying or confirming the identity of an individual based on the comparison of two fingerprints. **Fingerprint recognition** is one of the most well-known biometrics, and it is by far the most used biometric solution for authentication on computerized systems. The reasons for fingerprint recognition being so popular are the ease of acquisition, established use and acceptance when compared to other biometrics, and the fact that there are numerous (ten) sources of this biometric on everyone.

25) Automatic Document Classification / Categorization by Subject.

Text documents are one of the richest sources of data for businesses: whether in the shape of customer support tickets, emails, technical documents, user reviews or news articles, they all contain valuable information. Document classification or document categorization is a problem in library science, information science and computer science. The task is to assign a document to one or more classes or categories. This may be done "manually" (or "intellectually") or algorithmically. The intellectual classification of documents has mostly been the province of library science, while the algorithmic classification of documents is mainly in information science and computer science. The problems are overlapping, however, and there is therefore interdisciplinary research on document classification. The documents to be classified may be texts, images, music, etc. Each kind of document possesses its special classification problems. When not otherwise specified, text classification is implied. Documents may be classified according to their subjects or according to other attributes (such as document type, author, printing year etc.). Automatic document classification tasks can be divided into three sorts: supervised document classification where some external mechanism (such as human feedback) provides information on the correct classification for documents, unsupervised document classification (also known as document clustering), where the classification must be done entirely without reference to external information, and semi-supervised document classification, where parts of the documents are labeled by the external mechanism.

26) Automatic Language Identification using K-means Clustering.

In natural language processing, language identification or language guessing is the problem of determining which natural language given content is in. It refers to the problem of identifying the language of a given spoken utterance from short-term acoustic features, or of a given text (In the case of identifying the language from a written text it is viewed as a special case of text categorization). Apply the K-means algorithm (a clustering technique) to automatically identify a natural language, either from a spoken utterance or a written text.

27) Biometrics: A Voiceprint (Speaker Verification or Identification) System.

There are two major applications of speaker recognition technologies and methodologies. If the speaker claims to be of a certain identity and the voice is used to verify this claim, this is called verification or authentication. On the other hand, identification is the task of determining an unknown speaker's identity. In a sense, speaker verification is a 1:1 match where one speaker's voice is matched to a particular template whereas speaker

identification is a 1:N match where the voice is compared against a certain amount of templates. From a security perspective, identification is different from verification. Speaker verification is usually employed as a "gatekeeper" in order to provide access to a secure system. These systems operate with the users' knowledge and typically require their cooperation. Speaker identification systems can also be implemented covertly without the user's knowledge to identify talkers in a discussion, alert automated systems of speaker changes, check if a user is already enrolled in a system, etc. In forensic applications, it is common to first perform a speaker identification process to create a list of "best matches" and then perform a series of verification processes to determine a conclusive match. Build a voice-print system using Artificial Neural Networks.

28) A Recommender System using both Collaborative Filtering & Content-based Filtering.

A recommender system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. They are primarily used in commercial applications. Recommender systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommenders for services such as Amazon, or content recommenders for social media platforms such as Facebook and Twitter, and open web content recommenders like Outbrain. These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommender systems for specific topics like restaurants and online dating. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services.