DENIZ KARACA – Project Portfolio

1. Correlation – Simulation:

This program analyzes the historical transaction data provided and creates a list of items with each of them having a list of other items that are ordered together. Frequency of each item being ordered together is calculated and then a new location is assigned to each item depending on the distance to the main picking station. A set of orders is created depending on cumulative frequencies for the simulation stage. Then, the old and new plans are simulated; output is the total distance traveled for a set of order with two different location plans.

Details: Data.java parses data from the input historical transaction data Excel file. The correlation list is created after that by going over the parsed data set and constructing a list of items that are ordered together for each inventory item. Order.java has the Order class with all necessary variables to store the item list and location coordinates, and a constructor for initialization. OrderList.java calculates the frequency of each item being ordered and computes the cumulative frequency for each item. Using this, it creates the order list with specified cumulative frequency to be covered for the simulation stage. Simulation.java has all the old and new assigned location data and it goes over the entire list to calculate the total distance traveled. It also uses helper functions to compute the rectilinear distance between two given points.

Technologies used: MS Excel, Apache POI, Java

Link: https://github.com/dkaraca/past-projects/tree/master/Correlation-Simulation

2. k-Means:

Given a data set, this program creates k clusters with user specified stopping conditions and iterations as an input. After the creation of clusters, the F score (depending on recall and precision values) are calculated in order to determine the accuracy of each clustering (total distance of data points to their cluster's centroid). Later, a 3D plot is created to visualize and compare sets of clusters with different parameters.

Details: DataParser.java parses data from the input file; initializes a list of points and adds elements as data is parsed. Point.java has the Point class with variables for each coordinate and a constructor for initialization. Cluster.java has the Cluster class with variables to store centroid and point list data for each cluster that is formed, and a constructor that takes a centroid point as an input. Helper functions are also included to set scores and print point lists. KMeans.java has the KMeans class that implements the clustering algorithm with specified stopping conditions; initial centroids are randomly selected from the input data set. Helper functions are included to update k clusters and their lists of points after each iteration, compute distance metrics and total accuracy scores of k clusters. F1.java has the F1 class with necessary variables of accuracy, and helper functions to calculate weighted (determined by String labels) and unweighted accuracy scores. State.java has the State class with list of points in the given data set and a HashMap that stores different k, stopping condition and weighted/unweighted accuracy score data. Driver.java has the main function that calls other class functions to perform tasks.

Technologies used: Java, Python, plotly

Link: https://github.com/dkaraca/past-projects/tree/master/k-Means

3. Perceptron for Digit Recognition:

Given a training data set which consists of handwritten digits, this multiclass perceptron learns and classifies each digit. Later, the accuracy of these perceptrons are tested for accuracy over the test set of handwritten digits provided.

Technologies used: Java

Link: https://github.com/dkaraca/past-projects/tree/master/Perceptron

4. Demand Forecasting:

Implementing the demand forecasting algorithm for an airline revenue management system using Big Data processing technologies. This algorithm accurately predicts the untruncated demand of each price class with the help of Machine Learning techniques and statistical applications. Input is the booking data that is collected from the reservation system of the airline. This is an ongoing project.

Technologies used: MS Excel, Scala, Spark

5. Revised Simplex Method

Implemented the revised simplex method from scratch using MATLAB; input parameters are constraint matrix and objective function coefficients. Consists of initialization and simplex step functions and another method that calls these two functions in order to solve the linear program.

Technologies used: MATLAB

Link: https://github.com/dkaraca/past-projects/tree/master/RevisedSimplex

6. CS225 – Data Structures:

- **Image Manipulation I** – use pointers and C++ libraries for image manipulation
- **Image Manipulation II** – implement constructors and the Big Three, use inheritance and existing C++ functions and classes to manipulate images
- **Images and Lists** – use of templates to manipulate linked memory by writing functions to modify linked lists
- **Stacks and Queues** – implement stack and queue data structure and DFS/BFS traversals using them
- **Quadtrees** – implement constructors and manipulators for quadtrees
- **PhotoMosaic** – implement a nearest neighbor search algorithm on k-d trees, use of templates, form an application
- **Mazes** – implementing disjoint sets, graphs and graph traversals to solve a maze; represent the maze and its solution as a PNG

Technologies used: C++, Subversion, GDB, Valgrind

7. CS125 – Introduction to Computer Science:

- **Debug Me** – debugging exercise for spotting runtime and compile time errors
- **Hollywood** – input text file processing
- **Top Secret** – string processing through nested loops and ciphers
- **Photoscoop** – image processing (pixels)
- **Data Structures** – exercises with stacks, queues, arrays and use of public/static methods
- **Recursion See** – use of linked lists, family trees and recursive functions for a Gene Sequencing Analysis problem
- **Recursive Knight** – implementing recursive methods for determining shortest paths; selection sort algorithm and review of ASCII pictures

Technologies used: Java

8. Financial Engineering – Binomial Model

Implemented the binomial model on a C++ platform for pricing American and European vanilla options. Input parameters included type of option, number of iterations and strike prices. In addition to option prices, computational time was analyzed given different parameters. Outputs of different parameters were later analyzed in Excel with all other option price data and compared with the prices obtained through the Black-Scholes model.

Technologies used: C++, MS Excel

Link: https://github.com/dkaraca/past-projects/tree/master/Binomial%20Model