****

**DUX**

**Data Utility with eXcellence**

*Your personal Data Scientist*

Software Requirement Specification

Session: 2019-2020

CSE 339

*Team name Creative Engine*

Nikhil JSK 11601789

Prasanth G. 11601526

Anil Raj T. 11601794

Raja Rahul M. 11601793

**Table of Contents**

1. Introduction

1.1 Purpose 4

1.2 Description 4

1.3 Scope 4

1.4 Overview 5

1.5 Definitions, Acronyms and Abbreviations 6

1.6 Technologies to be used 6

2. Overall Description

2.1 Use-Case Model 7

2.2 Web Architecture diagram 8

2.3 Challenges 8

2.4 ER Diagram 9

2.5 Workflow description 10

2.6 Assumptions and Dependencies 12

3. Module Description

3.1 Component Description 12

3.2 Flow Chart 14

4. Specific Requirements

4.1 Supplementary Requirements 15

4.2 Non-Functional Requirements 15

5. Test Cases 16

6. System Evolution 19

7. References 20

**Table of Figures**

1. Use case model 7

2. Web Architecture 8

3. ER Diagram 9

4. Upload Segment Workflow 10

5. Process Segment Workflow 10

6. Visualize Segment Workflow 11

7. Model Segment Workflow 11

8. Flow Chart 14

9. Test Case - III 17

10. Test Case - IV 18

***1. Introduction***

* 1. *Purpose*

To extremely stream-line the process of Data Analysis in the field of Data science. Helps every data-enthusiast to pre-process the data, visualize, gain insights of the data and also train a machine learning model (depending on the type of data provided) just while communicating with *DUX* in high-level language (English). This tool is highly influential to prove that Data Science is highly powerful, simultaneously demonstrating how easy it is to use Data to produce significant and critical results for any domain in a global perspective. This tool will prove that analysis of any given data can be done ***without*** a single line of code, also no local dependencies

* 1. *Description*

A REST-API service integrated with a user-friendly and compelling UI where the user can upload the data, communicate with the assistant in a high-level language (English), use different graphs to visualize the data points, gain insights about the data in-hand, and train a basic model. And, all of that is done only through typing the text in the tool and not a single line of code, where real-time processing is done in the backend. These are the basic four steps in a conventional data science field, all streamlined to achieve the best productivity along with ease of analyzing the data.

* 1. *Scope*
* Maintain a global perspective
* Make data science compelling
* Operating System independent.
* Useful for amateur data scientists
* Easy access to tools in data science
* Text-based preprocessing, visualization and model building
* Streamline the process of data analysis to the greatest extent
* Train a holistic model without any local-installed libraries
* A viable solution which serves as a foundation for solving global problems**­­­**
* Targeting multiple levels of user types, newbies, beginners, amateurs, professionals in Data science field.

*1.4 Overview*

1. The user is asked to upload a dataset.
2. The user is then prompted to choose between Pre-processing, Visualization or Model building.
3. Choosing pre-process, the user can communicate in the form of text with DUX data assistant which is basically a REST API service.
4. Preprocessing is one of the essential features to visualize or even create or build a Machine Learning model.
5. Pre-processing enables user to write text to remove null values, nan values, drop columns in a dataframe, data type conversion, and also encode the string or categorical features to make processing easy for machine learning models.
6. He can do some basic math calculations like finding the minimum, maximum, mean, standard deviation, variance, skewness and kurtosis of any give column.
7. Visualization is the radical element for understanding the distribution of data points in the dataset.
8. Choosing visualization, the user is directly taken to the visualizations segment where the he can communicate with the assistant to utilize features of different graph and features in Seaborn and Matplotlib library. Simple text is converted to visualizations of the data.
9. The user can also upload an image and select the text in the image and then make a query based on the data.
10. This gives the user, the flexibility of choosing between text-based query or image-based query.
11. Gaining insights about the data is an integral part of data analytics, this part is included in the visualization segment of DUX. The user is given the inference from the graph.
12. Insights and inferences from are graph are derived from the data and presented to user in the form of text
13. Handling of graphs might be straight forward, if it is about handling direct data columns. But the user can also ask for the correlation between two different columns along with the variance of the third.
14. Choosing model building, the user is given recommendation on the type of machine learning model to be applied on the dataset.
15. Once the user selects between Regression, Classification, Clustering, Natural Language processing, the user is then given the recommendation on the sub-type or the algorithm to be used on the dataset to get the best accuracy or efficiency.
16. The efficiency of each algorithm and sub-category of algorithms, are applied on the subset of the dataset.
17. Once the training is done, metrics related to the training are displayed. The user is given an option to download the trained model to save loads of time. He can then use the saved model later to get predictions of related data.

*1.5 Definitions, Acronyms and Abbreviations*

1. Data Science – A multi-disciplinary field which uses algorithms and math to extract insights from data.
2. Data Analytics – The process of inspecting, cleansing, transforming of data and modeling.
3. Regression – A supervised machine learning for prediction continuous values
4. Classification – A supervised machine learning model for classifying multi-label or multi-class data.
5. Clustering – An unsupervised machine learning model for grouping similar data points together
6. NLP – Natural Language Processing – Subfield of AI to understand the interactions between human and machines
7. Data Visualization – The process of gaining insights from the data by looking at the distribution of data points in the dataset, and also drawing some basic inferences of the data.
8. Image Processing - Set of computational techniques for analyzing, enhancing, compressing, and reconstructing images.
9. Data Pre-processing – Clearing-up the noise in the data, structuring the data in a way that is understandable.
10. Train and test model – To train or build a model, which first trains on a subset of data and then the other data is used for testing the model on the basis of performance, accuracy and precision of predictions.
    1. *Technologies and libraries to be used*

* Python
* OpenCV
* TensorFlow
* Flask
* Flask Restful
* NLTK
* Sklearn
* SciPy
* Seaborn
* Matplotlib
* Keras
* Feature Extraction tool
* React js
* Node js
* Firebase
* API calls (Requests and Fetch)
* Custom library for text highlight.

*2.1 Use-Case Model*

* The use-case model shows the features and options for user, from frontend perspective.
* Five different segments
  + Upload Dataset/Image
  + Load a trained model
  + Pre-processing
  + Visualization
  + Model Training
* User has the entire control over the program. There is no admin.
  1. *Web Architecture diagram*

DATABASE

Process Invoke

REST CALL

Firebase

Python Flask API

Node

Server

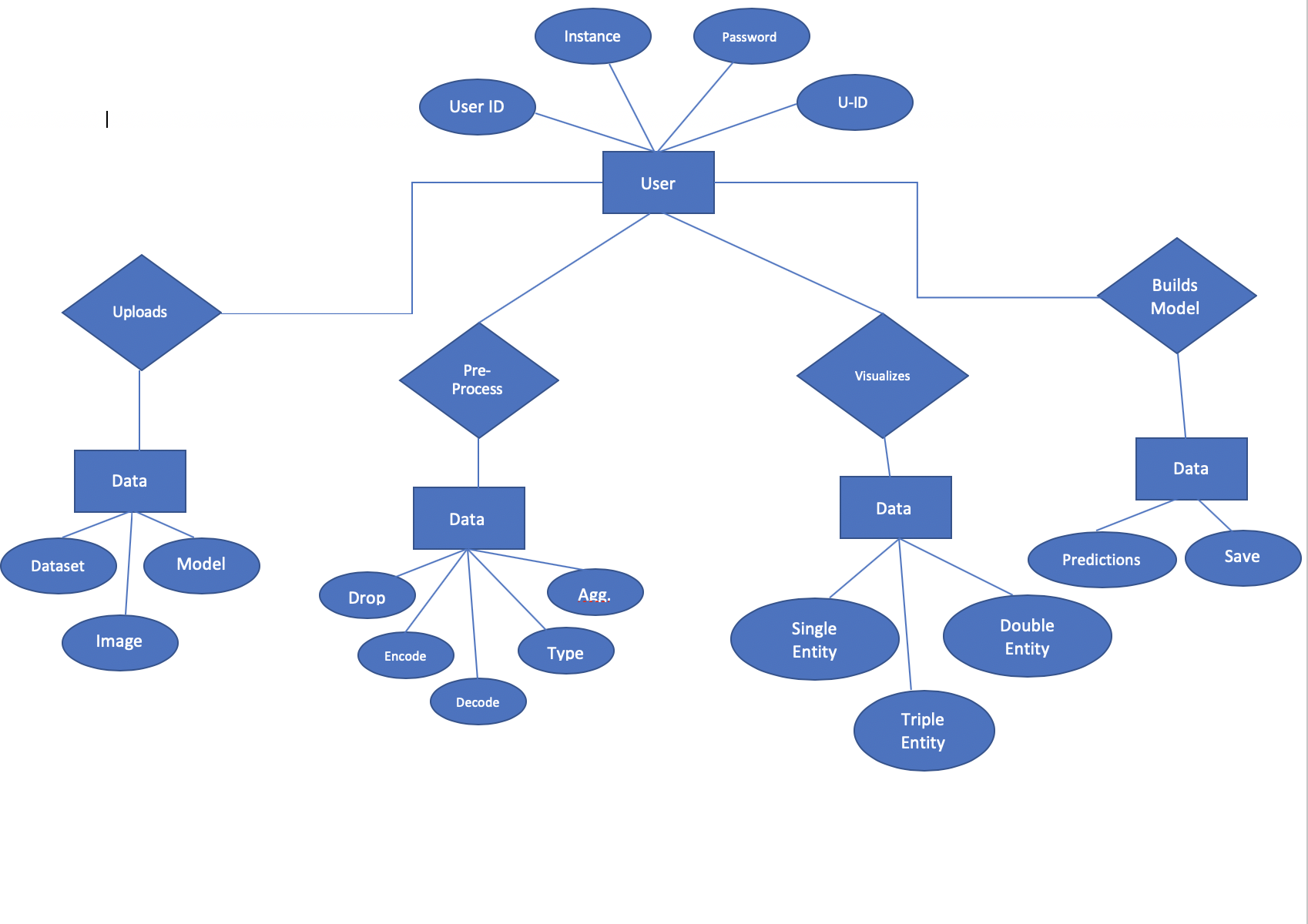
DATABASE

DATABASE

* This is a simple yet elegant web architecture of DUX.
* REST API calls are implemented at every level of communication between server, database and UI.
* This architecture is highly reliable and used universally.

2.3 *Challenges*

* Using Natural language processing to classify the higher level language into machine-understandable language
* Finding the key points in the datasets, to help the user visualize and draw inferences
* Organizing the data into a format to make it comprehensible
* Processing image input, to find the right text and match it with a movie in the database.
* Organizing all of the data in the UI using React.
* Connecting the data with backend server.
* Real-time processing of the data in the backend
* Dealing with huge amounts of data, using the right type of graphs and plots
* Understanding syntactic meanings, semantics of the data entered by the user
  1. *ER Diagram*

**

* The above diagram explains the relationship between all the entities in the systems.
* The actions that can be performed by the user are also expalined
  1. *Workflow Description*

***Step - 1***

Step-3

Store Changes in Db

Step-2

Apply Functions

Pre-Processing

Drop

Aggregation

Encoding

Decoding

Type Conversion

Merge

Work Flow

Upload Datasets

START

Step-2

Store in DB

Operations

Dataset

Image

Dataset

Model

Model

Dataset

Image

* Use-case – The user uploads the data in either of the three different forms. The uploaded data is stored in the firebase
* Work flow -- Demonstrates the backend-perspective of the data that is to be processed

***Step – 2***

Work Flow

* Use-case – The user chooses to modify or pre-process the data. Different options (included in use-case) can be performed on data.
* Work flow – Demonstrates the data storing process in backend.

***Step – 3***

\

Work Flow

Save Model in DB

Stop

Build Model

Step-4

Analyze Text

Plot Graphs

Step-3

Prediction

Step-4

Predictions

Save Model

Work Flow

Graph Output

Inference

Visualization

* Use-case – The user visualizes the data with upto three different variables, also the inferences are automatically shown.
* Work flow -- Demonstrates the backend-perspective of the data to be processed
* Output is either in the form of graph or a text based inference in this segment of the system.

***Step - 4***

Model Building

* Use-case – Recommended model for the dataset is given to the user. The user chooses to build a model of his choice. Predictions are then returned, the trained model can be saved to DB or downloaded.
* Work flow -- Demonstrates the backend-perspective of the model training and results process.

*2.6 Assumptions and Dependencies*

* User needs to have a basic understanding of data science.
* User inputs are expected to be according to the documentation.
* Assumed that the data is not more than ~200MB
* File formats supported are csv, tsv, jpg, jpeg, png
* Only python machine learning models are supported
* Training time varies according to the data uploaded by the user

***3.0 Module Description***

*3.1 Component-wise Description*

1. *Data Pre-Processing*
   1. The dataset given, can contain a lot of un-processed information. Using Python library **“***Pandas***”** and “*Numpy*” we can convert the datasets to form or draw inferences.
   2. The data pre-processing part can include text to, remove null values, nan values, drop columns in a dataframe, data type conversion, and also encode the string or categorical features to make processing easy for machine learning models.
   3. User can do some basic math calculations like finding the minimum, maximum, mean, standard deviation, variance, skewness and kurtosis of any give column
2. *Image Input format*
   1. The query from the user is in the form of image. The image is then converted to four different formats like RGB and grayscale etc. Using this combination, we detected the text in the image differentiating it with the background using edge-detection. *Open-cv* is used for the above.
   2. Now, using *Pyteserract* from the converted image, text is extracted and then an array of all required constraints and keyword parameters are created and sent for processing/visualization.
   3. The user can select the any of the highlighted text in the UI to make a query on that particular text
   4. The option is with the user, if he/she wishes to make a query or just output everything related to that word in the dataset.
3. *Text Input format*
   1. The query from the user is in the form of text. The text is converted into vector assigning different values using Tensorflow (Tokenizer) simulating Natural Language Processing (NLP).
   2. The converted vectors are then matched with derived keywords and constraints from the dataset to extract the values as parameters for sending it to processing/visualization
4. *Data Visualization*
   1. The converted input which is generated using Image and Text processing is used for fetching the data from the datasets.
   2. In some cases, Output should be just a simple sentence. These cases have been taken into consideration.
   3. In some other cases, Output should be in the form of graphs.
   4. Complex queries are also handled, where multiple queries can also be asked by the user.
5. *Model Building*
   1. Once the dataset is preprocessed and visualized, the dataset can be used to train or build a model.
   2. The user is given a recommendation for the type of machine learning model to be applied on the dataset, amongst Regression, Classification, Clustering or Natural Language processing based classification
   3. The user can select any of those, or build a custom model based on different parameters.
   4. Once the user chooses a type of algorithm, the user is then given the recommendation based on the type of algorithm to be used to get best accuracy or precision of the dataset.
6. *Saving the model*
   1. The trained model can then be saved to use it for the next time.
   2. Predictions can be made on the basis of this without training the model over and again.

*3.2 Flow Chart*

*4.1 Supplementary Requirements*

* To maintain a global perspective the server should be well maintained and should support multiple threads for better handling of the requests
* Large scale handling of the data requests is required for the server.
* Documentation must be well written and followed throughout the process of development for better user-experience
* Making the project open-source can yield better results, as there is a lot of scope and expected enhancements for the project

*4.2 Non-functional Requirements*

*Service Availability*

The service in the server is expected to run at all times except when there is a maintenance scheduled by the admin team. The maintenance is scheduled & reported to the user before-hand.

*System Security*

The system is a secure service. The RESTAPI calls used takes in the system-generated security tokens.

*Throughput*

The system is designed to handle asynchronous requests. There can be multiple processes processed and handled at the same time, increasing throughput and memory utilization.

*Reliability*

The system is highly reliable as it handles multiple asynchronous requests. If there is any service issue, it is handled by the backend with intimating the UI about the service changes.

***5.0******Test - Cases***

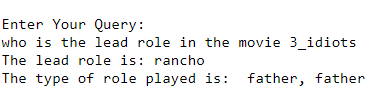
*Dataset Assumptions*

* Assuming the user entered a dataset which is related to the movies released between 1970 and 2007
* The dataset contains different columns like the lead role, movie name, songs in the movie, singers in the movie
* Assuming it also has some noise data in the columns of emotions in the movie,. Year, plot of the move
* The genre of the movie can be inferred from the column of emotion in the movie.
* The length of the movie, appearances of a character in the movie are also provided.
* All of this data can be used to visualize the graph after pre-processing
* Real-life data usually has a lot of noise including nan values, null values, odd characters, non-ascii characters. All of this can be handled by the user by typing-in some text.

***Test case – 1***

***Query :*** *Who is the lead role in the move 3 idiots*

***Solution:***

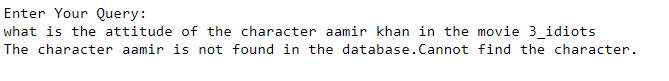
******

* The user enters a text, which is about the column and row corresponding element.

***Test case – 2***

***Query:*** *What is the attitude of the character aamir khan in the movie 3 idiots?*

***Solution:***

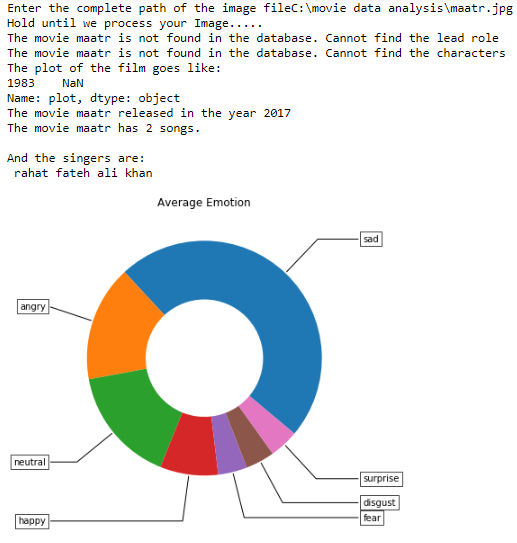
******

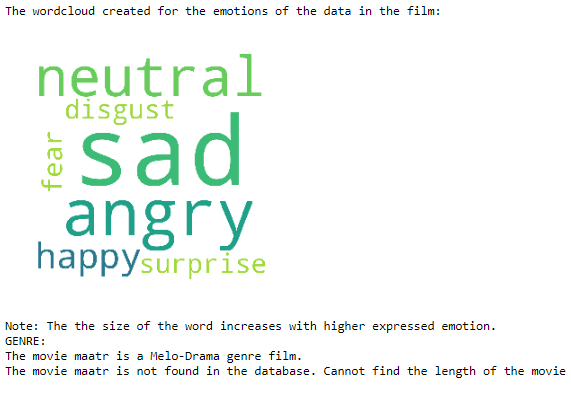
* The user would get a solution if he would have entered “Rancho” instead of aamir khan.
* The user enters a data element which is not present in the dataset.
* Theses test-cases are also handled to show the end-result to the user.

***Test case – 3***

***Path of image:*** */Users/user/home/maatr.jpg*

***Solution:***

**

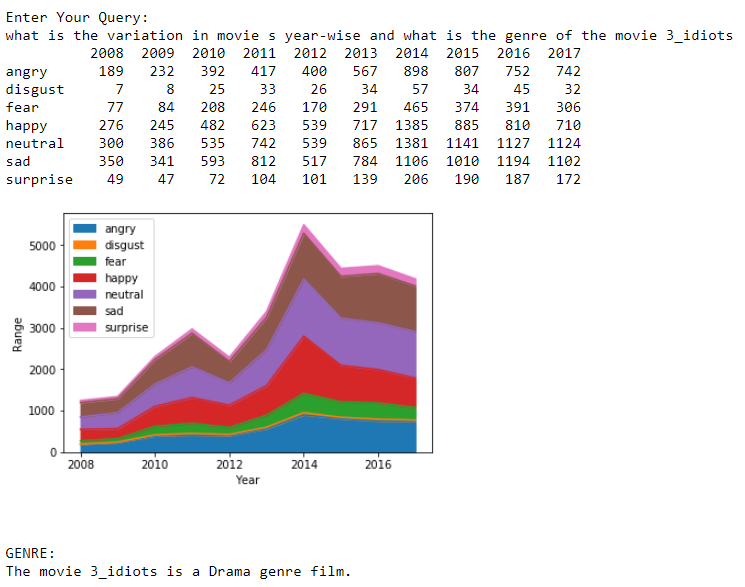
**

* Assuming the user has entered an image path and then selects the text “maatr” which is amongst the highlighed text.
* The user is given all the possible solutions for the keyword “maatr”, which is a movie, from the dataset.
* The same can be visualized above.

***Test case – 4***

***Query :*** *what is the variation in movie year-wise and what is the genre of the movie 3 idiots?*

***Solution:***

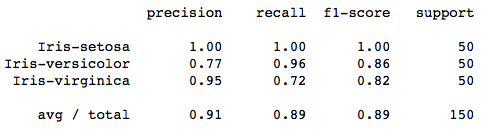
**

* The query is self-explanatory

***Test case – 5***

* Assuming the user has uploaded a dataset, selecets the recommended model type and the recommended algorithm for the given model-type
* The user would get the performance metrics of the model
* The user can also choose to save the model to disk for further use.
* The testcase is demonstrated in the next page.

*Query : <*Build a model with model-type classification with Random Forest classifier as the algorithm type on the dataset movies. Let the features be 1, 2, 3, 4, 5 columns and the label be 6 columns> All of this will be choosed by the user in the UI.



***6.0 System Evolution***

This system is designed to address the problem “*Data noise removal, data pre-processing, visualization and basic model training are time consuming*”. The evolution of this kind of system is endless. It not only addresses a global problem, but provides a viable and feasible solution to solve the problem by streamlining the process to the greatest extent.

* Adding another step of Data analysis – *Fine tuning* the model. This step might be tricky but can be done using backward elimination of features. If this can be done it would be highly useful and serves a critical process element
* Analysis huge corpus of data using Natural Language Processing to support NLP features
* Scaling the project to a higher level to support Big Data Analytics
* Increasing the catalogue of available Machine learning models to support higher custom data analytics
* Support to create ensemble models by passing output of one model to another to yield better results.
* Utilizing the REST API service, many applications can be developed. Just create a UI and then use the existing service to add the backend-magic. Chatbots, Desktop applications, Enterprise-level applications, IOS and Android applications can all be supported with the help of the service created.
* Establish a version control system which handles the present work items created, like different model version with different parameters and many more.
* Save the entire process of steps made to create a pipeline, which can later be used to do the same process all over again just in one go.

***Evolution is Endless***

*5.0 References*

1. <https://github.com/nikhiljsk/movie_data_analysis>
2. <https://www.quora.com/What-are-the-functional-and-non-functional-requirements-in-software-engineering>
3. <https://www.studytonight.com/dbms/er-diagram.php>
4. https://www.ibm.com/developerworks/community/files/form/anonymous/api/library/3cf803c7-f973-4051-99a8-2949fd4ceab1/document/e6bd87f7-a158-4997-b5b6-2a8e2f3e11ed/media/SRS\_Sample.doc

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*