Classifier Analysis Application

Functional Specification

School of Computing Cian Manley 15563637

CONTENTS

FUNCTIONAL SPECIFICATION

1. INTRODUCTION

1.1 Overview

Machine learning and Data Analysis is continuously growing field at the moment. I intend the application to be a helpful tool for users involved in predictive data analysis. It will involve the implementation of 4 different machine learning algorithms. The algorithms which I will be aiming to implement are Naïve Bayes, Decision Tree, K-Nearest Neighbours and Support Vector Machine. Each of the algorithms will have their own configurations needed to be inputted in to the application. These will be implemented alongside a simple GUI (Graphical User Interface) in order to allow the user a clean interface to both use and interpret. For each of the algorithms chosen, it will produce a number of different graphical description of the outcomes while also showing actual result. Once each of the algorithms have run there will also be an area where you can see the outcomes side by side in order to have a clean view to contrast. The comparison will be done through a statistical comparison of each of the different algorithms run. [1]This means producing different results for each of the algorithms and averaging them in order to give a description of the accuracy score for the run.

1.2 Business Context

Data analysis and machine learning is a continuously growing area in many fields. Machine learning in particular is a very broad subject and takes a great deal of time to wrap your head around the basic concepts. Then to be able to analyse your data to the lengths of being able to choose a machine learning algorithm suited to your data takes an even longer amount of time. This application will provide the ability to assess different machine learning algorithms used for predictive calculations. A Data Analyst is often out of question for a lot of smaller companies. This gives those companies a function which often they cannot afford.

Since it shows the outputs of well known classification algorithms, it may also be used for helping students learn machine learning. As it gives a result based on the different algorithms run, students will be able to see what algorithms suit the different type of Data sets.

1.3 Glossary

React	- A JavaScript library used for building user interfaces.
Anaconda	- Anaconda is a free and open-source distribution of the Python and R programming languages for data science and machine learning applications.
REST API	- Representational State Transfer. Rest is a software architectural style which defines a set of restraints to aid in the development of a web application.
Flask	- It is a micro web framework written in Python.
Webpack	- This is an open-source module bundler used to take in modules with dependencies and transforms them into static assets representing those modules.
Data Analysis	- The application of different statistical techniques in order to evaluate, describe and illustrate data.
KNN Algorithm	- A machine learning algorithm used for both classification and regression. It works based on minimum distance from the query to the training samples to determine the K-nearest neighbours
SVM Algorithm	- Support vector machines algorithm is a machine learning algorithm. A classification method that generates non-overlapping partitions and usually employs all attributes.
Decision Trees	- The decision trees algorithm builds models in the form of a tree structure. It works by continuously splitting the data into subsets in order to produce a sine curve with a set of rules.
Naïve Bayes	- Naïve Bayes is a classification algorithm based on Bayes' Theorem with an assumption of independence among predictors.

2. GENERAL DESCRIPTION

2.1 Application Functions

The primary functions for this project is to create a web application which a user can upload a data set in the form of a csv file and choose the different metrics they want to use on a various number of classification algorithms.

UPLOADING A CSV FILE

When uploading the dataset, the user will have an option to upload the dataset completely or split into both training and testing data. If the user is uploading a complete dataset they will have the option of choosing how they want to split the data set. They will also have the option to use cross validation.

CHOOSING DIFFERENT ALGORITHMS

The algorithms will be separated into different python modules. By inputting these as individual APIs it will allow for the addition of another algorithm If needed. Once, the user chooses which classification algorithm they want to use, it will get the different modules instead of having to call them all at once and in turn increase the speed of the overall run time.

SETTING THE DIFFERENT INPUTS FOR THE ALGORITHMS

Each of the Algorithms have different inputs needed to run. The user will be required to enter in the certain different types of inputs needed for their chosen algorithms. For example, in Support Vector Machines you may be asked whether your data set is binomial or a multinomial data set.

GRAPHICAL VISUALISATION OF OUTCOMES

Once the user chooses their configurations for each of the different algorithms used, the graphical user interface will a display a side by side view of each of the outcomes. It will show the results in an easy to read manner to allow the user to contrast the different algorithms outcomes.

2.2 User Characteristics and Requirements

This application will provide the user with a simplistic graphical user interface. However, it will still require a certain amount of knowledge surrounding the subject of Data Analysis. This is due to the complication of pre-processing and assessing the outcomes of each of the algorithms.

Pre-processing involves a number of different techniques like Data cleaning, Data Integration and normalisation. The user will have to be able to read their own data to a certain degree as the only one I will be looking to implement is certain pre-processing techniques. Two techniques that could be used are standardization and normalization.

I will be providing a help function in order to illustrate the different results. There will be only so much I can explain so the user will have to be able to asses which of the algorithms will best used on their dataset.

2.3 Operational Scenarios

UPLOAD DATA

This will be the initial stage of the application. The user will be asked to upload their own data. They will have the choice to add a pre-split dataset or both the training set and the evaluation set. If a non-split dataset has been uploaded, users will be prompted how they want their dataset split.

SELECTING ALGORITHM

Once the data has been uploaded and split, users will then be displayed the different algorithms available to them. This is to allow a presentation of the same algorithms under different configurations. They may also have an idea of which algorithms they are looking to compare and are only looking at certain algorithms.

RUN ALGORITHM

Once they have their algorithms selected they will then run the program and be presented with the different accuracies. They will be provided with both a numerical version and a graphical version. This is where the statistical comparison will be done in order to give the best possible accuracy value for the user.

HELP FUNCTION

For user which are less knowledgeable on the subject area, I will be providing as much explanation in a help function as I can. However, the user will still need to have a certain level of understanding surrounding the outcomes of the program.

RESET ALGORITHM

Once they have run the program, the user may have forgotten to include a certain algorithm or wants to upload another dataset. There will be and added reset function to reset the program back to its original state to allow the user to upload to the web application and run through it another time.

2.4 Constraints

TIME CONSTRAINTS

Time will be a big factor in this project as there is a good bit of workload involved. This will be from researching the separate algorithms to understand how they work and the different configurations that may have to be included. To design a graphical user interface will take a good deal of time to make it as simple and easy to read as possible. This must be done all while meeting different deadlines to keep on track of the workload. I believe I will be able to produce a platform to high standard, but I will have a backup plan in place to make sure that the specifications are met.

LEARNING CURVE

Behind this project there will have to be a huge research element. I will have to understand the different algorithms used in order to implement and display them in a way that is understandable to most users. There is also the design of the project. The design the of the graphical user interface will also take a bit of learning as I will be using the react library for JavaScript in an attempt to create a single page web application. Once I have my MVP, I will be looking at whether it is possible to add another algorithm or not. This will only become clear once I have my MVP.

3. FUNCTIONAL REQUIREMENTS

3.1 Web App Interface

Description: This is going to be the graphical user interface in order to give the end-user a visual on the program. It will also display the different graphical visuals used for the different results of algorithms. This is also how the user will input their dataset and select the different classification algorithms they want to use on the data.

Criticality: This is a major factor in the application to give users a simplistic approach to the program.

Technical Issue: It must be designed in a simplistic manner to give a clear result from each of the different algorithms used.

Dependencies: It has dependencies with each of the different algorithms in order to display a graphic for the accuracy, certain results for each algorithm ran.

3.2 Naïve Bayes Algorithm

Description: Based on Baye's Theorem. It is applied with the assumption of strong independence between different feature columns. This classification algorithm will be implemented as a module into the python application. It will then be applied to the given data set and return the accuracy and run time to the GUI interface.

Criticality: It is a major component as it is one of the main algorithms used in the program. It will be essential to implement it as soon as possible.

Technical Issue: It will be coded using external python libraries. This will allow me to provide the user with different statistical measures for each of the programs run.

Dependencies: The algorithm is dependent on the GUI interface. It needs a data set imported into it to run and return the results based on it.

3.3 Support Vector Machines Algorithm

Description: SVM is used by finding a hyperplane that best divides datasets into classes. The Support Vector Machines classification algorithm will be implemented as a module into the Python API. It will then be applied to the given data set and return the accuracy and run time to the GUI interface.

Criticality: It is a major component as it is one of the main algorithms used in the program. It will be essential to implement it as soon as possible.

Technical Issue: It will be coded using external python libraries. This will allow me to provide the user with different statistical measures for each of the programs run.

Dependencies: It will be dependent on react to send HTTPS requests in order to receive the dataset and send out REST API containing a JSON file with results of the algorithm.

3.4 Decision Trees Algorithm

Description: A decision support which uses a tree-like model which details the decisions and consequences. The Decision Trees Algorithm will be implemented as a separate module in the Python API. It will then be applied to the data set received from the GUI interface in order to return the results of the application on the dataset.

Criticality: It is a major component as it is one of the main algorithms used in the program. It will be essential to implement it as soon as possible.

Technical Issue: It will be coded using external python libraries. This will allow me to provide the user with different statistical measures for each of the programs run.

Dependencies: It will be dependent on react to send HTTPS requests in order to receive the dataset and send out REST API containing a JSON file with results of the algorithm.

3.5 K-nearest neighbours Algorithm

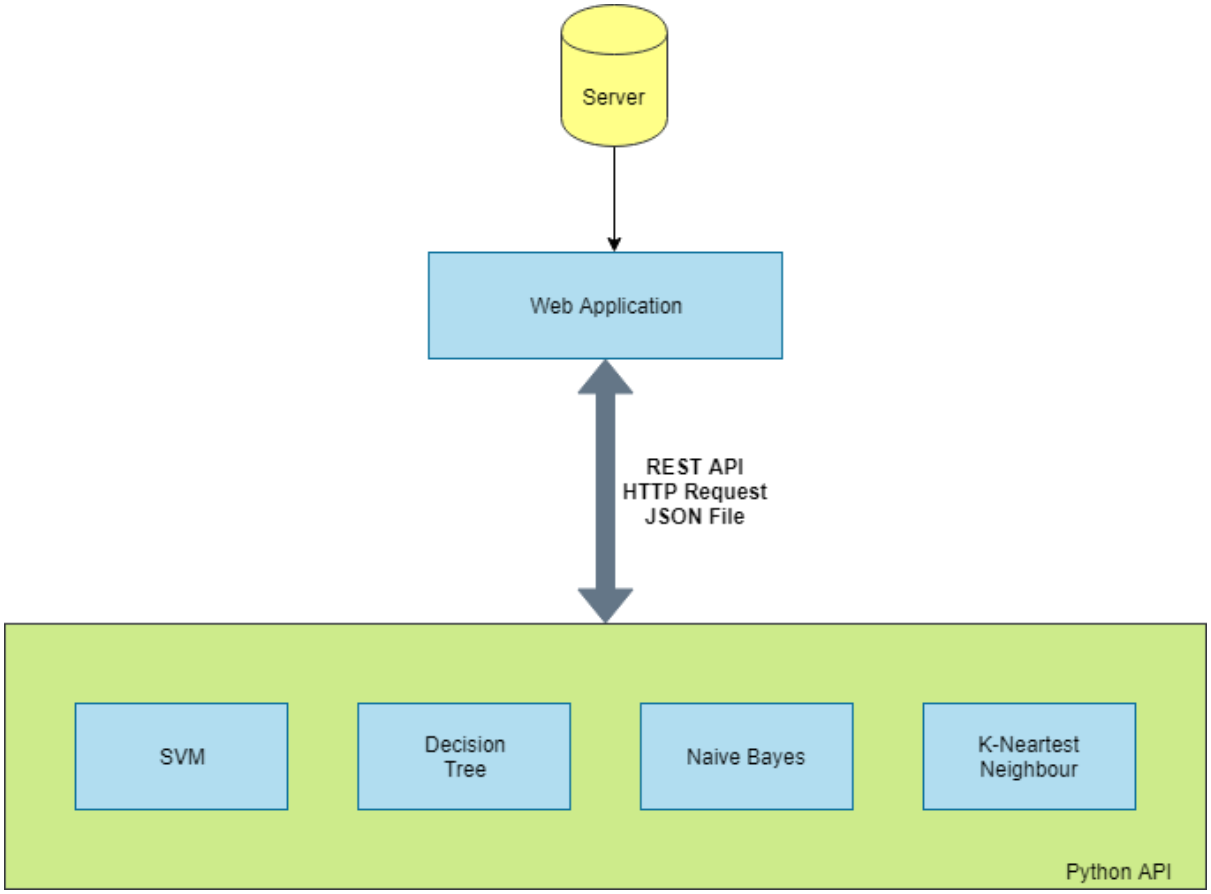
Description: One of the simpler machine learning algorithms. It classifies new data in the dataset based on a distance function. The K-nearest neighbours algorithm will be implemented as a separate module in the python API. It will then be applied to the data set received from the GUI interface to return the results of the application on the dataset.

Criticality: It is a major component as it is one of the main algorithms used in the program. It will be essential to implement it as soon as possible.

Technical Issue: It will be coded using external python libraries. This will allow me to provide the user with different statistical measures for each of the programs run.

Dependencies: It will be dependent on react to send HTTPS requests in order to receive the dataset and send out REST API containing a JSON file with results of the algorithm.

4. SYSTEM ARCHITECTURE

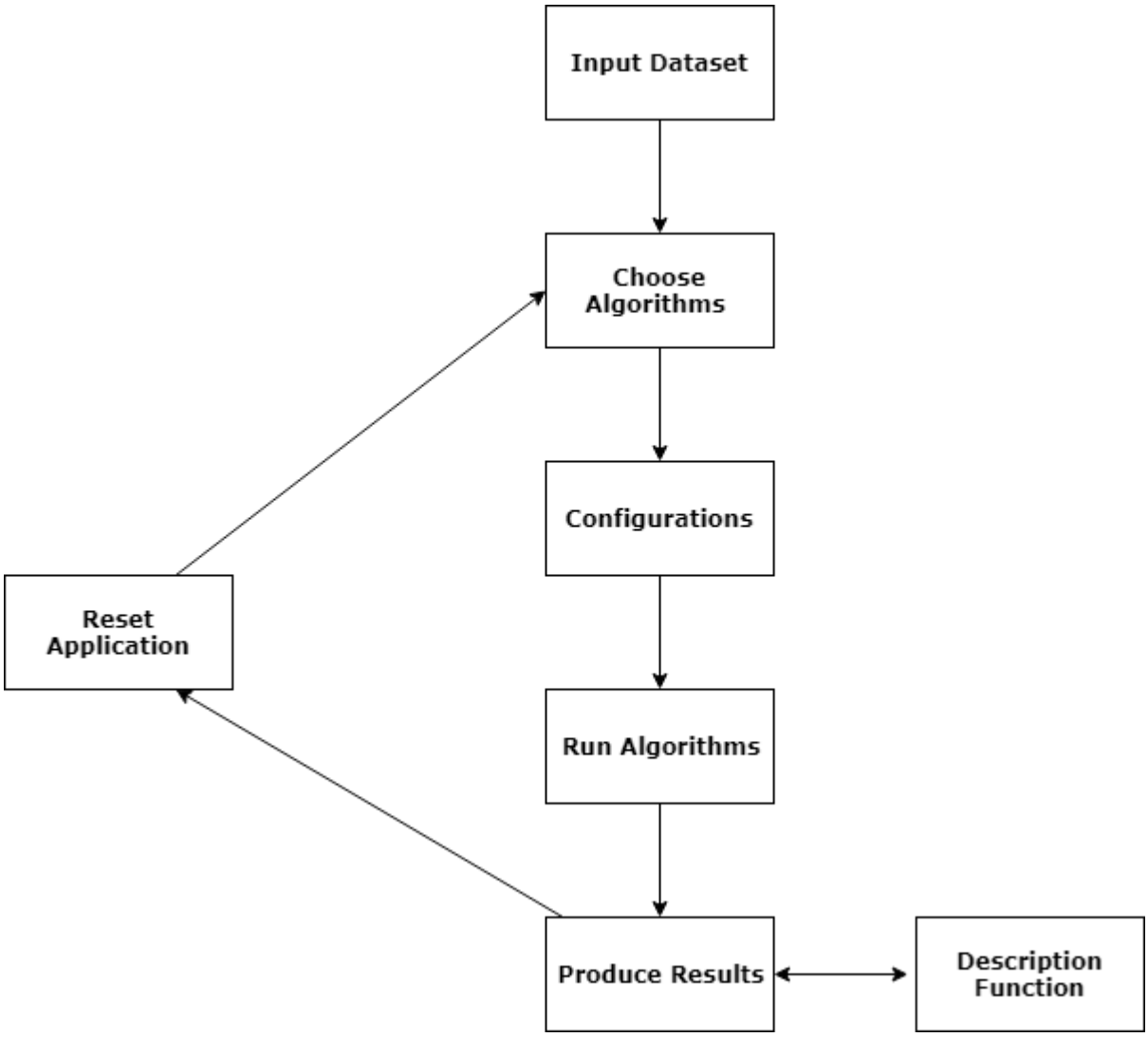


The above shows the architecture of the proposed application. It includes three main features. This includes the different modules used within the python API, the web application and the server system to where the user will be able to drag and drop their data set.

This python-based API will hold 4 different modules for each of the algorithms used. The algorithms have been separated into different modules to allow them to be used independently. This will allow testing to each of the classifiers singularly instead of a collective. This will allow for the simple addition of more machine learning algorithms to increase the scope if time allows. Each of the modules which receive information through REST API in the form of a JSON file and will return the results in the same manner from whatever modules are called.

5. HIGH-LEVEL DESIGN

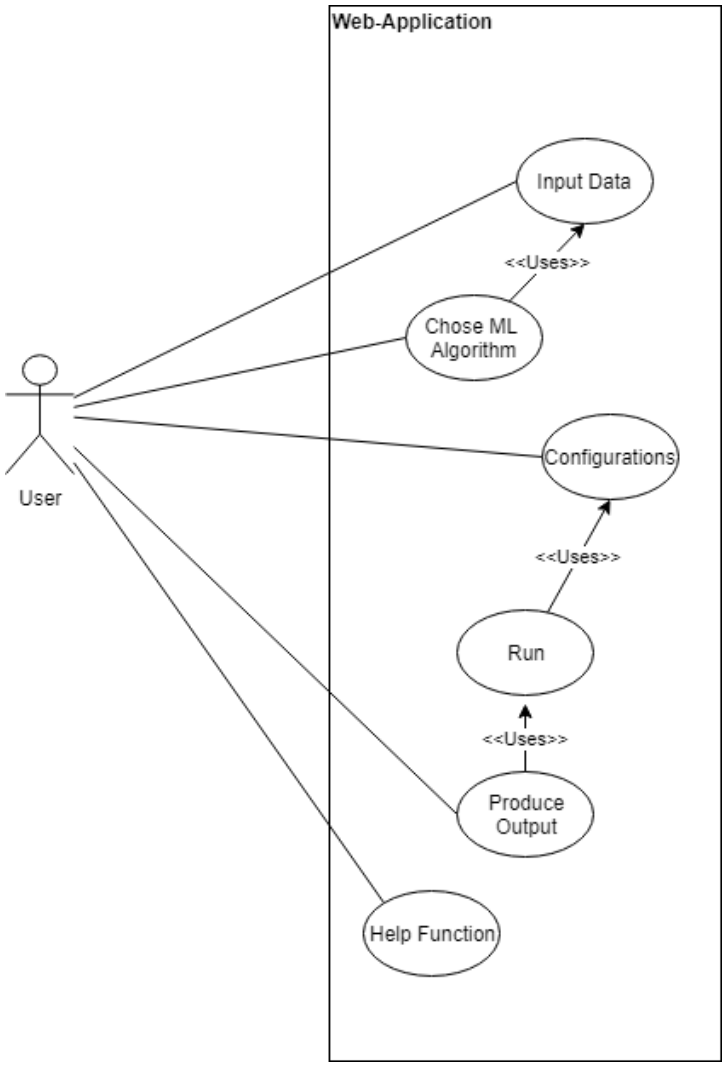
5.1 Higher-Order Diagram



Above is a higher-order diagram of the system. Each of the processes provide a function to the application. First the user starts by uploading their Dataset, once they have done that they have the option of choosing and removing the algorithms before they move on to the different configurations. With each of the algorithms, a description will be provided in order to reduce confusion about what the algorithm may be doing. Once the user has their selection(s), they will be required to enter in specific configurations e.g. feature columns, label column, K-value. It will then run the algorithms based on the inputted configurations

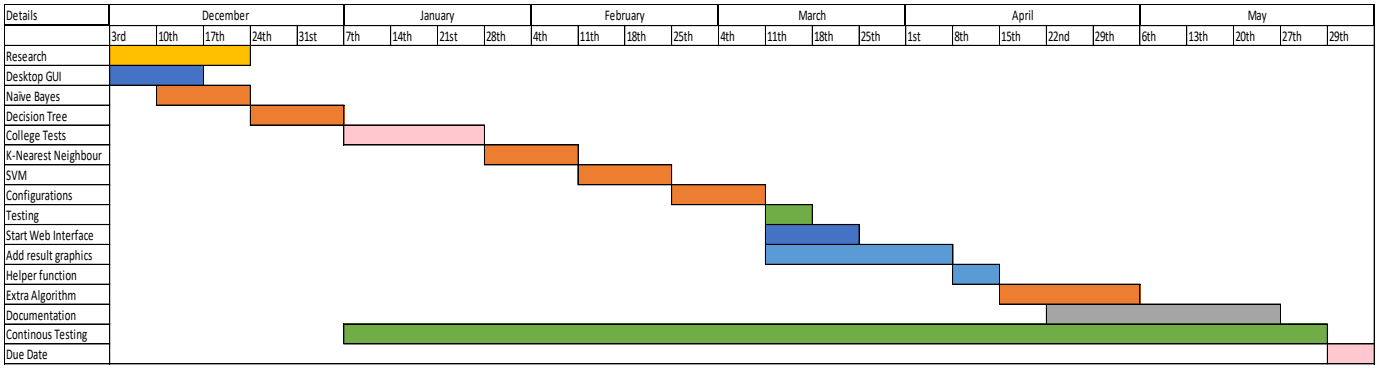
and produce both a graphical and numerical result. The user will also have an option to open a description function in to help understand the values.

5.2 Use Case Diagram



The use case diagram above is to show when the user will have interactions with the application. I will be attempting to make user interaction as minimal as possible. All the user will have to do is input in the data and their configurations. Once they have done this the python API will do the rest. Then once all the results are produced the user will interpret from their own knowledge to which they think is the most suitable ML algorithm for them.

6. PRELIMINARY SCHEDULE



Above is the current schedule I will be sticking to. Most of the research on the project has nearly been completed. I will be aiming to have all of the algorithms coded and have a desktop interface by February. This might be a tight schedule but should be completed in time. Once that is done by February, I will be producing the interface to show the results. Throughout the coding element, I will be continuously testing the code using varying types of datasets from different sources. If time allows at the end there will be time to add another algorithm to show another set of results.

7. APPENDICES

7.1 Resources

- gitlab.com
- stackoverflow.com
- stackexchange.com
- Kaggle.com
- **Machine Learning Repo** - http://archive.ics.uci.edu/ml/index.php

TOOLS

- React
- Anaconda
- Jupyter Notebook
- Visual studio code
- Python
- JavaScript

7.2 References

[1] Kotsiantis, S.B., Zaharakis, I. and Pintelas, P., 2007. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, *160*, pp.3-24.