

***Faculty of Science and Technology***

**Assignment Coversheet**

|  |  |
| --- | --- |
| **Student ID number &**  **Student Name** | Natalie Dixon-Campbell u3242089 |
| **Unit name** | Software Technology 1 |
| **Unit number** | 4483 |
| **Unit Tutor** | Girija Chetty |
| **Assignment name** | ST1 Capstone Project – Semester 1 2023 |
| **Due date** | 12/05/23 |
| **Date submitted** | 12/05/23 |

**You must keep a photocopy or electronic copy of your assignment.**

**Student declaration**

I certify that the attached assignment is my own work. Material drawn from other sources has been appropriately and fully acknowledged as to author/creator, source and other bibliographic details.

**Signature of student: Natalie Dixon-Campbell Date: 12/05/23**

# Introduction

This report describes the Python Capstone Project for the Software Technology 1 unit. I have decided to work on the project used a chess games dataset available at Kaggle [1].

Chess is a game that has been played for centuries, and throughout that time strategies have constantly evolved to gain an edge. This project aims to analyse a large quantity of online chess games in order to detect patterns, and to create an algorithm in order to predict the result of a game based on the skill of the competitors and the opening used.

# Methodology

The methodology used for developing the software platform involves 3 stages as outlined below:

1. Design and development of decision support algorithms based on exploratory data analysis and predictive analytics, for identifying the best performing algorithm for solving a real world problem.
2. Implementation of best performing algorithm as a desktop Tkinter software tool.
3. Deployment of the tool as a web or cloud enabled platform tool.

# Dataset Description

The dataset used was sourced from Kaggle, and included over 20000 unique games. There are over 9000 unique players represented in the dataset. The dataset has 16 meaningful columns, which held details on the games rated status, the number of turns, the winner and the method of victory, the time increment of the game, the ratings and names of both players, the moves used, and the details of the openings used. The target variable is the winner column, where a 0 is a black win, a 1 is a draw, and a 2 is a white win. Therefore the task is to develop a software tool to predict the result of a chess game, using the other variables.

# Exploratory Data Analysis

The first phase of the software development activity involved understanding the data, basic exploratory data analysis and visualisation. Google Colab was chosen as the experimental environment as it incorporates virtual hardware and resources which does not require additional physical hardware requirement and can be ran directly of a web browser. The python language was used to create the scripts which ran directly on online Jupyter notebook using Google Colab with the help of free google account created, and by saving all the notebook files virtually on google drive without additional configurations. Before the exploratory data analysis can begin, some of the python libraries for EDA need to be imported and dataset acquired, by using the following Python script

The first phase of software development is to understand the data using basic exploratory data analysis and visualisation. Google colab was the environment used as it uses virtual hardware and resources. Before EDA can begin, the python libraries for EDA and the dataset itself need to be imported with the following script:

A picture containing text, font, screenshot, line

Description automatically generated

A picture containing text, electronics, screenshot, display

Description automatically generated

### #Shape of the data (First and Last 5 rows

df

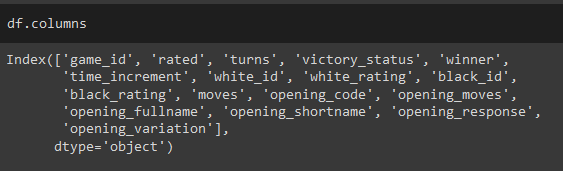


### #Shape of the data (Attributes and total samples)

A screen shot of a computer

Description automatically generated with low confidence

### #List of attributes



### #Unique values present for each attribute

A screenshot of a computer

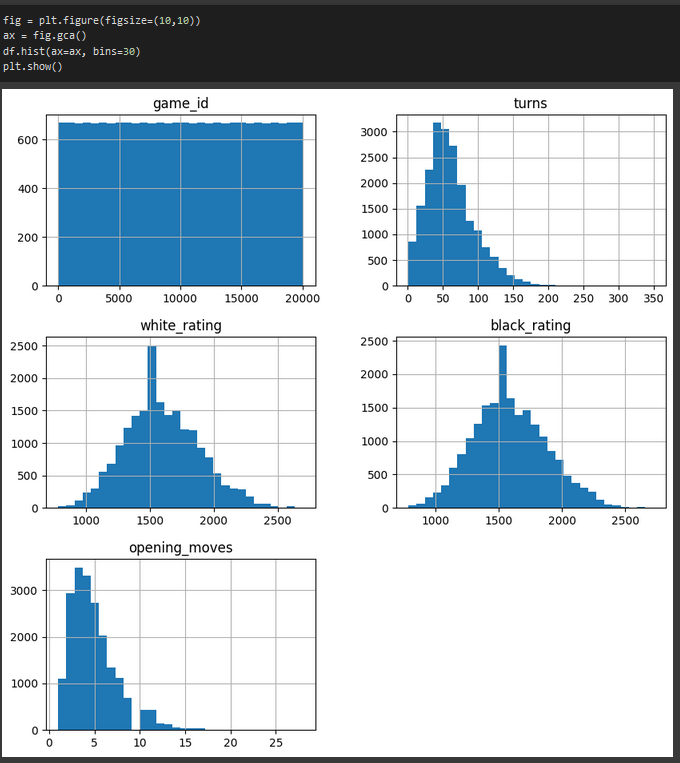
Description automatically generated

### #Complete info on data

A screenshot of a computer

Description automatically generated

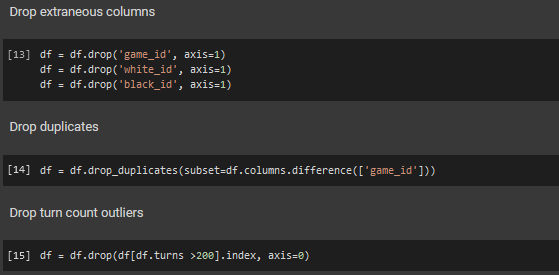
### #Distribution of integer based data in detail:



A screenshot of a graph

Description automatically generated with medium confidence

### #Remove Unnecessary Data

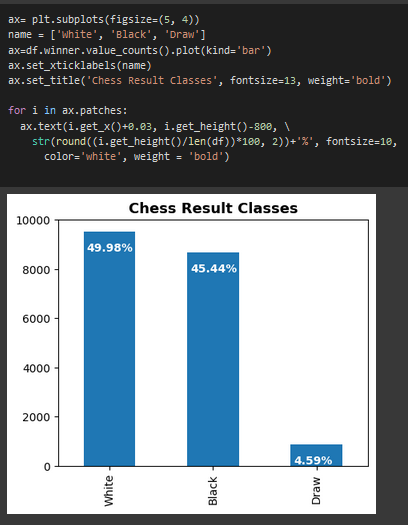


### #Check data shape after data removal

A screenshot of a computer

Description automatically generated with medium confidence

#Checking distribution of target value (winner)



A screenshot of a computer screen

Description automatically generated with medium confidence

### #Complete remaining EDA with Profiler Report

A screen shot of a computer code

Description automatically generated with low confidence

### #Correlation Map

A picture containing text, screenshot, diagram, line

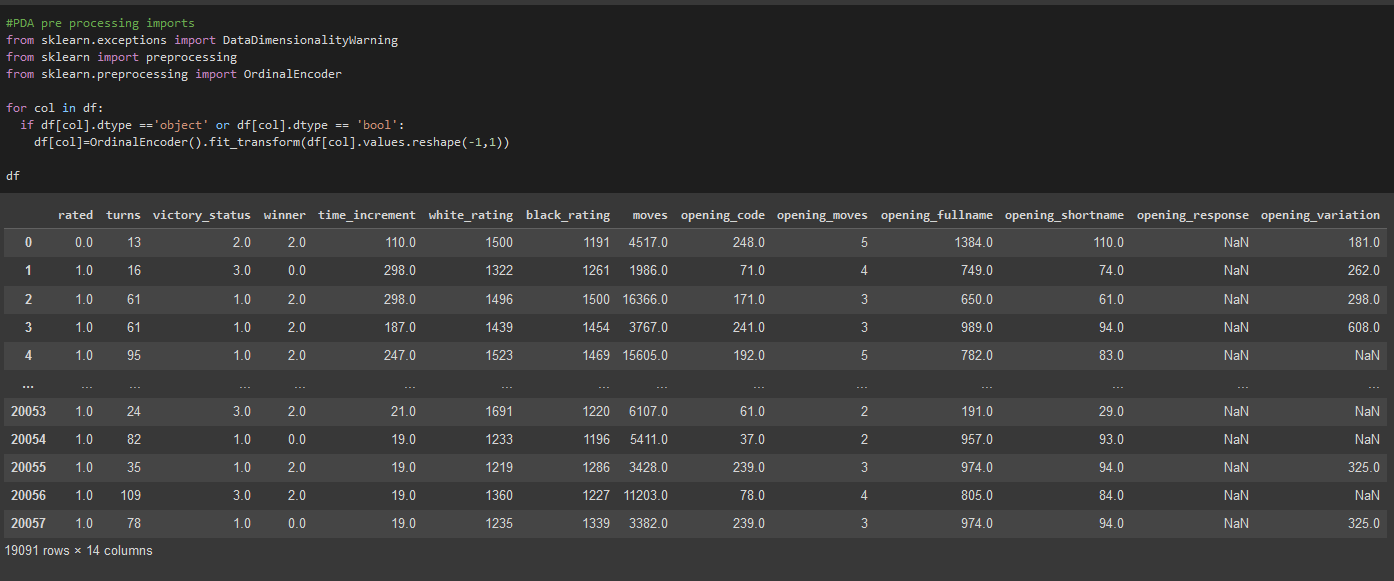
Description automatically generated

# Predictive Data Analytics

For predictive data analytics, several prerequisite processing steps are required. These are pre processing, classifier comparison and performance evaluation with different objective metrics. These have been obtained using the Python sci-kit learn package.

Pre-processing: Since the dataset consists of a combination of continuous and categorical attributes/variables, there is a need to pre-process the data with attribute transformation, standardization and normalisation. We used scikit-learn’s OrdinalEncoder() function to perform attribute transformation.

Pre-processing: Since the dataset consists of both integers as well as Booleans and objects, the data needs to be processed with attribute transformation, standardisation and normalisation. This was done with sci-kit learn’s OrdinalEncoder().



A screen shot of a computer

Description automatically generated with low confidence

A picture containing text, screenshot, font, software

Description automatically generated

### Model Preparation and Development

#### Steps used for machine learning model preparation are described below:

* Convert the dataframe to training and validation/test subsets by taking a random sample of 80% of the data and defining it as train subset. This leaves 20% of the data for validation/testing
* Create the validation/test set by dropping all of the rows that comprise the training set from the dataframe.
* Create y\_train by using using the last column of train (target class).
* Create x\_train by using all of the columns in train except the last one.
* The validation set of y\_val and x\_val or (y\_test and x\_test), can be created using the same methodology that used to create y\_train and x\_train

### #Import necessary python libraries for PDA

A screen shot of a computer

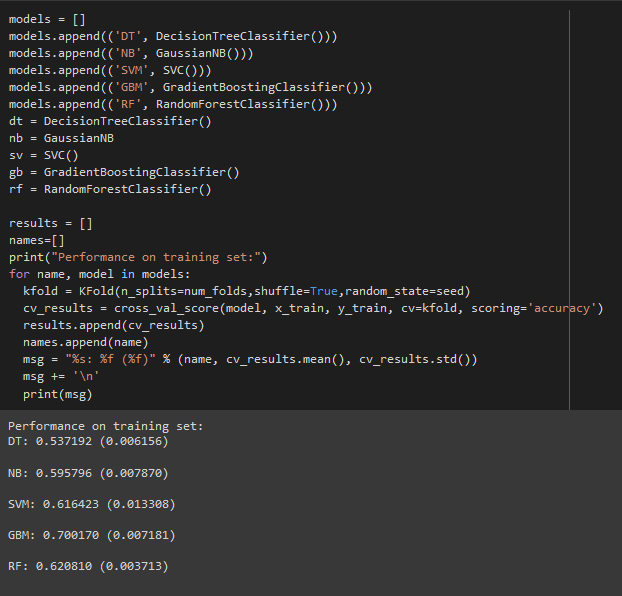
Description automatically generated with medium confidence

### #Splitting the data into training and testing subsets, and showing the shape.

A screen shot of a computer

Description automatically generated with medium confidence

### #Model development by comparing different sci-kit learn classification algorithms



### #Comparing performance of different algorithms

A screen shot of a graph

Description automatically generated with medium confidence

### #Making predictions on the test dataset

A screenshot of a computer code

Description automatically generated with low confidence

### #Model Performance Evaluation Metric 1 - Classification Report

A screenshot of a computer

Description automatically generated with medium confidence

### #Model Performance Evaluation Metric 2 – Confusion Matrix

A screenshot of a computer screen

Description automatically generated with low confidence

## 

# Stage 2: Algorithm Implementation Stage

1. Implementation of best performing algorithm as a desktop Tkinter software tool.
2. Deployment of the tool as a web or cloud enabled platform tool.

Once the best algorithm and model for chess game prediction has been identified, the next step is the implementation of the algorithm as a desktop software tool using Python’s Tkinter package.

The pycharm project for this is available in the github repository.

## https://github.com/nat-dc/ST1Capstone/tree/main/chess\_games\_gui

# Stage 3: Software Deployment Stage

The deployment of the software as a desktop tool limits applicability. Therefore, there is a need to deploy it as a web based application using Flask API.

The pycharm project for this is available in the following github repository.

https://github.com/nat-dc/ST1Capstone/tree/main/chess\_games\_flask

# References

1. https://www.kaggle.com/datasets/ulrikthygepedersen/online-chess-games