Predicting the Winner of a Tennis Match using Machine Learning Techniques

MSc Research Project
Data Analytics

Akshaya Sekar
Student ID: x18138977

School of Computing
National College of Ireland

Supervisor: Dr Dondio Pierpaolo
<table>
<thead>
<tr>
<th>Student Name:</th>
<th>Akshaya Sekar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID:</td>
<td>x18138977</td>
</tr>
<tr>
<td>Programme:</td>
<td>Data Analytics</td>
</tr>
<tr>
<td>Year:</td>
<td>2018</td>
</tr>
<tr>
<td>Module:</td>
<td>MSc Research Project</td>
</tr>
<tr>
<td>Supervisor:</td>
<td>Dr Dondio Pierpaolo</td>
</tr>
<tr>
<td>Submission Due Date:</td>
<td>20/12/2018</td>
</tr>
<tr>
<td>Project Title:</td>
<td>Predicting the Winner of a Tennis Match using Machine Learning Techniques</td>
</tr>
<tr>
<td>Word Count:</td>
<td>672</td>
</tr>
<tr>
<td>Page Count:</td>
<td>10</td>
</tr>
</tbody>
</table>

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author’s written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: \\
Date: 3rd February 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies). □

Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies). □

You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. □

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only
Signature: \\
Date: \\
Penalty Applied (if applicable):
1 Introduction

The configuration manual represents every step of the implementation process in a detailed manner. The hardware and software specifications are mentioned for the research project on the topic, “Predicting the winner of a tennis match using machine learning techniques.” The goal of this project is to predict the winner of the match with the individual player statistics using various machine learning models such as SVM, Logistic regression, Random forest, Naive Bayes. PCA was used for dimensionality reduction and random search Hyper parameter tuning was performed to increase the efficiency of the models.

2 System Specification

This project was implemented on the cloud platform Google colaboryatory also known as Colab. The colab supports GPU and TPU. Bisong (2019)

2.1 Hardware

- Google Colab: 2vCPU @ 2.2GHz
- The GPU Instance was 250GB
- The RAM was 13 GB
- The Disk Space was 32GB

2.2 Software

Python programming language was used to implement the project. The entire pre-processing tasks such as cleaning, encoding, dimension reduction implementation and evaluation was performed in Python.
3 Importing Libraries

Some libraries required are pre-defined in the cloud platform. The other necessary libraries were imported whenever required. This step involves importing the required libraries.

```python
# libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas_profiling
df = pd.read_csv('binarize_conflicting.csv', engine='python')
```

Figure 1: Importing Libraries

4 Data Extraction

4.1 Importing Files

In this step, the data set is mounted in the google drive and then the file is imported from the google drive.

```python
# from google.colab import files
df = pd.read_csv('binarize_conflicting.csv')
```

Figure 2: Importing Data

4.2 Set Path

In this step, the path of the data set is given and the data is read.

```python
# import io
raw_data = pd.read_csv(io.BytesIO(open('binarize_conflicting.csv').read()))
pdf = pd.DataFrame(raw_data)
```

Figure 3: Working directory path
4.3 Reading the data

```python
# Reading data
raw = pd.read_csv(path='stats (1).csv')
```

Figure 4: Reading Data

5 Exploratory Data Analysis (EDA)

The Exploratory Data Analysis was done with the help of pandas profiling in Python. The pandas profiling is a one line code which gives a better understanding about the insights of the data. It analyses the data and gives a HTML format report of all the missing values, outliers, class balance, correlations and other basic details about the dataset etc.

![Pandas Profiling Report](image)

Figure 5: Exploratory Data Analysis

5.1 Removing null values

In this process, the null values are removed from the raw dataset. This will increase the quality of the dataset and helps to give efficient result.

```python
# Removing null values
raw = raw.dropna()
```

Figure 6: Removing null values

5.2 Checking Class Imbalance

The class should be equally balanced to get efficient results, hence the process of under sampling or over sampling takes place depending on the data. Here in this dataset, the class was equally balanced.
5.3 Dropping the unwanted columns

The columns which are irrelevant and columns with special characters are removed.

```
# dropping unwanted columns
raw = raw.drop(['match_id', 'player_id'], axis = 1)
```

6 Data Pre-processing

6.1 Dependent and Independent variables

In this section, we are splitting the data set into dependent and independent variables. Here, X is denoted as the independent variable and y is denoted as the dependent variable.
6.2 Encoding the data

Since the machine learning models cannot accept characters, we will encode the dependent variables as 0’s and 1’s. This process is known as Label encoding. The Target column is label encoded as 0’s for loser and 1’s for winner.

7 Dimensionality Reduction

In this project, the Principle Component Analysis is used for dimensionality reduction. Since the dataset has continuous values, PCA is used for dimensionality reduction. PCA helps to reduce the number of columns by having a summary of all the important features with high variance. This helps to increase the efficiency of the models and reduce the computation time.

8 Training and testing dataset

In this stage, the data set was split in to training and testing in 80:20 ratio.
9 Machine learning models

There are four machine learning models implemented in this project. SVM, Naive Bayes, Logistic Regression, Random Forest.

9.1 Support Vector Machine

```python
#implementing SVC
svm = SVC()
svm.fit(X_train, y_train)
y_pred = svm.predict(X_test)
acc = accuracy_score(y_pred, y_test)
print("Accuracy ": acc)
false_positive_rate, true_positive_rate, thresholds = roc_curve(y_test, y_pred)
score_auc = auc(false_positive_rate, true_positive_rate)
print("AUC Score ": score_auc)
false_positive_rate, score_auc
score_f1 = f1_score(y_test, y_pred)
print("F1 Score ": score_f1)
score_kappa = cohen_kappa(y_pred, y_test)
print("Cohen's Kappa ": score_kappa)
AUC-ROC Curve:
plt.plot([0, 1], [0, 1], linestyle='--')
plt.plot(false_positive_rate, true_positive_rate, marker='.')
plt.show()
```

9.1.1 Hyper parameter tuning for SVM

The hyper parameter tuning helps in choosing the best parameters which increases the efficiency of the models. Here, the random search Hyper parameter tuning is used.
Figure 15: Selecting the parameters

9.2 Naive Bayes

This Gaussian Naive Bayes is split in to training and testing and then evaluated in terms of Accuracy, Auc, F1 score and Kappa.

Figure 16: Naive Bayes
9.3 Random Forest

Here, the Random Forest data is split into training and testing and evaluated in terms of Accuracy, F1 score, Auc and Cohen's kappa. It is seen that Random Forest has the highest accuracy of 68%.
9.3.1 Random Forest Hyper parameter tuning

The various parameters for hyper parameter tuning are set and the best parameters were selected using the random search hyper parameter tuning.

```python
# Setting range for hyper parameter
random_params = {'n_estimators': range(200, 2000, 10),
                 'max_features': ['auto', 'sqrt'],
                 'max_depth': range(10, 100, 10),
                 'min_samples_split': range(2, 10, 1),
                 'min_samples_leaf': [1, 3, 4]}

# Search for hyper parameter tuning
search = RandomizedSearchCV(estimator = random_forest, param_distributions = random_params, n_iter = 50, cv = 10, verbose=2, random_state=42, n_jobs = -1)
search.fit(X_train, y_train)
```

Figure 20: Random search Hyper Parameter tuning of RF

9.4 Logistic Regression

The Logistic Regression Evaluation is shown below. After splitting the training and testing data set, the accuracy, F1 score, AUC and Kappa are evaluated.

```python
# Implementing logistic regression
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(penalty = 'l2', C = 1.0)
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print("Accuracy : ", acc)
F1 = f1_score(y_test, y_pred)
print("F1 Score : ", F1)
AUC = roc_auc_score(y_test, y_pred)
print("AUC Score : ", AUC)
Kappa = cohen_kappa_score(y_test, y_pred)
print("Kappa Score : ", Kappa)
```

Figure 21: Logistic Regression

Figure 22: Logistic Regression result and AUC curve
9.4.1 Hyper parameter tuning for Logistic Regression

The random search hyper parameter tuning is done with various parameters to select the best parameters.

```python
# setting range for hyper parameter tuning
logi_param={"C":np.logspace(-3,3,7), "penalty": ["l1", "l2"],}

# random search cv for hyper parameter tuning
random_search = RandomizedSearchCV(logi, param_distributions=logi_param, n_iter=50, scoring='accuracy', n_jobs = -1, verbose=3)

random_search.fit(X_train, y_train)
```

Figure 23: Logistic Regression Hyper parameter

Figure 24: Selecting the best parameters

References