Predicting Outcomes of Chess Endgames Using Machine Learning Algorithms

K. Balachandra Reddy¹, P. Yaswanth², G. Hari Chandana³, J. Peddi Raju⁴
¹Adhoc Lecturer, Dept of CSE, JNTUACEP, Pulivendula, AP, India
², ³, ⁴Student, Dept of CSE, JNTUACEP, Pulivendula, AP, India

Abstract: The main aim is to predict the outcome of the chess endgames i.e. King-Rook vs. King and King-Rook vs. King-pawn endgames. We will find the algorithm that will predict the outcome of these endgames accurately to the great extent. So our goal is the predict the outcome of the game whether the white king will win or not. We will also predict in how many moves white king will be able to win the game. Her we will use classification models. Accuracy is used as the evaluation metric and the best model is chosen based on the accuracy.

Keywords: Prediction, Classification models, Accuracy

I. INTRODUCTION
Rook and pawn endgames are the most common type of endgames there is in the game of chess. These endgames take place in about in 80% of all the games. The main idea is checkmate of black king with white king and white rook. This should be done in less than 16 steps. Chess endgames are complex domains which are enumerable. The game theoretic values stored denote whether or not positions are won for either side, or include also depth of wins (number of moves). There will be mainly two types of outcomes in this game. The outcomes are whether the white king can win the game or cannot win the game. If the white king wins the game we can also predict that in how many moves the white king will win the game.

II. MOTIVATION
We will use this project in the final stage of the chess game i.e. king-rook vs. king and king-rook vs. king-pawn. We have developed this project by using classification algorithms that are involved in machine learning. Once we give the board positions of the white king, white rook, black king, black rook we will be able to predict whether the white king will win the game or not.

III. PROPOSED WORK
By using data mining techniques predicting the outcome of the chess endgames is a time consuming task. So in the proposed system we will use different supervised classification models to find the accuracy given by the each model and finally selects the best model which gives the highest accuracy. Some of the classification models which used are Logistic Regression, Random Forest, Support Vector Machine and Decision Tree. Task flow involved during developing of this application is

IV. METRICS
We will use accuracy score as evaluation metric to predict the outcome of chess end games. It is defined as the number of correct predictions made as a ratio of all predictions made. Accuracy is most common evaluation metric for classification problems.

\[
\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}
\]
V. D A T A S E T

In our project we consider two types of chess end games one is King-Rook vs. King and other is King-Rook vs. King-Pawn. For King-Rook vs. King: The dataset is comprise of multivariate data and it has both categorical and integer values. There will be total of 28056 instances and the total number of attributes is 6 they are White King File(Column), White King Rank(Row), White Rook File, White Rook Rank, Black King File, Black King Rank, Class(Target attribute).

For King-Rook vs. King-Pawn: The dataset is comprise of multivariate data and it has both categorical and integer values. There will be total of 3196 instances and the total number of attributes is 36 these are from feature value list that appear in every King-Rook vs. King-Pawn end games. Some of the feature values are Bkblk: Black King Block, Stlmt: Stalemate, Wkpos: White King Position …etc. Target attribute is Classes(2): White-Can-Win("won"), White-Cannot-Win ("nowin").

VI. D A T A P R E P R O C E S S I N G

Some of the datasets contain irrelevant information and noisy data. These datasets should be handled properly to get a better result. Data preprocessing includes data cleaning, transformation and dimensionality reduction which convert the raw data into a form that is suitable for further processing. The first step in data preprocessing is read the dataset, it is done by using read_csv(). And the shape of the dataset will helps to understand briefly about the size of the data. For King-Rook vs. King:

```
# know the shape of the data
df.shape
(28056, 7)
```

For King-Rook vs. King-Pawn:

```
(3196, 37)
```

First five instances of the dataset For King-Rook vs. King:

```
# using head() print the first five columns of the data
df.head()

| wht | wkr | wrf | wrn | bklf | bkr | bkn | bkmc | bkmp | blxwn | blxwp | blxwp | bxq2 | scrop | stlmt | trsbl | wklci | wklad | wklcnk | wklck2 | wklnd | wklon | wklpg | wklq | wpq2 | wpq2 | wpr2 | wpr2 | wpr2 | wpr2 | wpr2 | class |
|-----|-----|-----|-----|------|-----|-----|------|------|-------|-------|-------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0   | a   | 1   | b   | 3    | c   | 2   | draw |     |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |}
For King-Rook vs. King-Pawn:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3195 entries, 0 to 3194
Data columns (total 37 columns):
bkblk    3195 non-null object
bkmy     3195 non-null object
bkon8    3195 non-null object
bkona    3195 non-null object
bkspr    3195 non-null object
bkbq     3195 non-null object
bkxcr    3195 non-null object
bkxwp    3195 non-null object
blxwp    3195 non-null object
bxqsq    3195 non-null object
cntxt    3195 non-null object
dospp    3195 non-null object
dhwpd    3195 non-null object
hdcch    3195 non-null object
katri    3195 non-null object
mulch    3195 non-null object
qxnscq   3195 non-null object
r2anb    3195 non-null object
reskd    3195 non-null object
reskr    3195 non-null object
rimxx    3195 non-null object
rkxwp    3195 non-null object
rxmlsq   3195 non-null object
simpl    3195 non-null object
skach    3195 non-null object
skwtr    3195 non-null object
```

There is no missing or null values present both the datasets.
VII. DATA VISUALIZATION

Data Visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers. By using visualization we will easily find the correlation between different attributes in the dataset.

The above heat map shows the correlation between different attributes in the dataset.

VIII. MODELLING

In order to model the data we will first divide the entire dataset into training and testing datasets. 70 percent of the data comes under training data and the rest of the data goes into testing data set.

IX. ALGORITHMS AND TECHNIQUES

For the King-Rook vs. King dataset we apply three supervised classification algorithms LogisticRegression (Benchmark model), Random Forest, Support Vector Machine.

For the King-Rook vs. King-Pawn dataset the supervised classification algorithms applied are: Logistic Regression (Benchmark model), Random Forest, Decision Tree.

1) Logistic Regression: Logistic regression is one of the supervised classification algorithm used for binary classification. It gives a discrete binary outcome between 0 and 1. This algorithm works by measuring the relationship between the dependent variable and one or more independent variables. Advantage: Outputs have a nice probabilistic interpretation and the algorithm can be regularized to avoid overfitting. Disadvantage: Logistic regression tends to underperform when there are multiple or non-linear decision boundaries.
2) **Random Forest**: Random Forest is a predictive modeling algorithm which is used for both classification and regression tasks. It works well with default hyper parameters. It can be used to rank the importance of variables in a regression or classification problem. **Advantage**: Reduction in overfitting by averaging several trees, there is significantly lower risk of overfitting. **Disadvantage**: It takes more time to train the samples.

```python
# Import RandomForestClassifier from sklearn.ensemble
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=1)
# Fit the training data
rf.fit(X_train, y_train)

# find the accuracy score of trained data
print("Random Forest - Train accuracy: ", round(rf.score(X_train, y_train), 3))

# find the accuracy score of testing data
print("Random Forest - Test accuracy: ", round(rf.score(X_test, y_test), 3))
```

3) **Support Vector Machine**: Support vector Machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space which can be used for classification, regression or other tasks like outlier detection. **Advantage**: It works very well with clear margin of separation and it is effective in high dimensional spaces. **Disadvantage**: It doesn’t perform well when we have large dataset because the required training time is higher.

```python
# Import svm from sklearn
from sklearn import svm
# Support vector machine
rbfSVM = svm.SVC(kernel="rbf", C=1, gamma=0.1)
# Fit the training data
rbfSVM.fit(X_train, y_train)
# find the accuracy score of trained data
pred = rbfSVM.predict(X_train)
print("SVM - Train accuracy: ", round(svm.score(pred, y_train), 3))

# find the accuracy score of testing data
pred = rbfSVM.predict(X_test)
print("SVM - Test accuracy: ", round(svm.score(pred, y_test), 3))
```

4) **Decision Tree Algorithm**: It is one the most popular machine learning algorithm used for both classification and regression tasks and it is often mimic the human level thinking so it is very simple to understand the data and make some good interpretations. **Advantage**: Decision Trees are able to generate understand the rules and performing classification without requiring much computation. **Disadvantage**: Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.

```python
# Decision tree classifier
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(criterion="gini", max_depth=5, min_samples_split=2, min_samples_leaf=1)
t = tree.fit(X_train, y_train)

tree.score(X_train, y_train)
```

©IJRASET: All Rights are Reserved
X. RESULTS

The accuracy scores given by the classification algorithms are:

For King-Rook vs. King end game

| Model name       | Accuracy |
|------------------|----------|
| Logistic Regression | 0.358    |
| SVM              | 0.654    |
| Random Forest    | 0.737    |

For King-Rook vs. King-Pawn end game

| Model name       | Accuracy |
|------------------|----------|
| Logistic Regression | 0.951    |
| Decision Tree    | 0.932    |
| Random Forest    | 0.977    |

From the above results we conclude that Random Forest algorithm gives the best accuracy score in both the end games and it is the best algorithm when compared with the others.

XI. REFINEMENT

We will refine the models by using GridSearchCV. And we will tune the parameters are ‘n_estimators’=[450,500] and ‘criterion’=['gini','entropy']. After tuning the final accuracies are:

For King-Rook vs. King: 0.7379
For King-Rook vs. King-Pawn: 0.9896

XII. CONCLUSION

By using machine learning algorithms we can easily predict the outcomes of chess endgames. These models are trained with the dataset and gives the accuracy score as results. From all the algorithms Random Forest algorithm gives the best accuracy score in King-Rook vs. King and King-Rook vs. King-Pawn end games so it is selected as best algorithm from the others.

This application can be further developed for other endgames like Tic-Tac-Toe endgame and many more.

REFERENCES

[1] https://archive.ics.uci.edu/ml/datasets/Chess+%28King-Rook+vs.+King%29
[2] https://archive.ics.uci.edu/ml/datasets/Chess+%28King-Rook+vs.+King-Pawn%29
[3] https://towardsdatascience.com/the-random-forest-algorithm-d457d499fcd
[4] https://www.analyticsvidhya.com/blog/2017/09/understanding-support-vector-machine-example-code/
[5] https://www.ritchieng.com/machine-learning-decision-trees/