Spring 5-22-2020

USING MACHINE LEARNING TO OPTIMIZE PREDICTIVE MODELS USED FOR BIG DATA ANALYTICS IN VARIOUS SPORTS EVENTS

Akhil Kumar Gour
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the [Artificial Intelligence and Robotics Commons](#)

Recommended Citation

Gour, Akhil Kumar, "USING MACHINE LEARNING TO OPTIMIZE PREDICTIVE MODELS USED FOR BIG DATA ANALYTICS IN VARIOUS SPORTS EVENTS" (2020). *Master's Projects*. 940.
DOI: <https://doi.org/10.31979/etd.wctq-k8bz>
https://scholarworks.sjsu.edu/etd_projects/940

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

USING MACHINE LEARNING TO OPTIMIZE PREDICTIVE MODELS USED FOR BIG
DATA ANALYTICS IN VARIOUS SPORTS EVENTS

A Project Report

Presented to

Dr. Ching-Seh (Mike) Wu

Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By

Akhil Kumar Gour

May 2020

The Designated Project Committee Approves the Project Titled

USING MACHINE LEARNING TO OPTIMIZE PREDICTIVE MODELS USED FOR BIG
DATA ANALYTICS IN VARIOUS SPORTS EVENTS

By

Akhil Kumar Gour

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE
SAN JOSE STATE UNIVERSITY

Spring 2020

Dr. Ching-Seh (Mike) Wu, Department of Computer Science

Dr. Nada Attar, Department of Computer Science

Mr. Ajay Wisawe, Senior Engineer, HPE

©2020

Akhil Kumar Gour

ALL RIGHTS RESERVED

ABSTRACT

In today's world, data is growing in huge volume and type day by day. Historical data can hence be leveraged to predict the likelihood of the events which are to occur in the future. This process of using statistical or any other form of data to predict future outcomes is commonly termed as predictive modelling. Predictive modelling is becoming more and more important and is trending because of several reasons. But mainly, it enables businesses or individual users to gain accurate insights and allows to decide suitable actions for a profitable outcome.

Machine learning techniques are generally used in order to build these predictive models. Examples of machine learning models ranges from time-series-based regression models which can be used for predicting volume of airline related traffic and linear regression-based models which can be used for predicting fuel efficiency. There are many domains which can gain competitive advantage by using predictive modelling with machine learning. Few of these domains include, but are not limited to, banking and financial services, retail, insurance, fraud detection, stock market analysis, sentimental analysis etc.

In this research project, predictive analysis is used for the sports domain. It's an upcoming domain where machine learning can help make better predictions. There are numerous sports events happening around the globe every day and the data gathered from these events can very well be used for predicting as well as improving the future events. In this project, machine learning with statistics would be used to perform quantitative and predictive analysis of dataset related to soccer. Comparisons of these models to see how effectively the models are is also presented. Also, few big data tools and techniques are used in order to optimize these predictive models and increase their accuracy to over 90%.

Keywords- Machine learning, predictive modelling, predictive analysis, sports analytics.

ACKNOWLEDGEMENTS

I stretch out my appreciation to my advisor Prof. Mike Wu for helping me whenever I needed guidance. His experience on various subjects assisted me in learning numerous industry practices while actualizing my decisions. Without his guidance and encouragement, it would not have been a conceivable task to come up with this idea for further implementation.

I also want to thank Dr. Nada Attar and Mr. Ajay Wisawe for consenting to be on my committee and giving their valuable inputs to my project, without which the project would not have been successful.

Table of Contents

ABSTRACT	4
ACKNOWLEDGEMENTS	5
INTRODUCTION	9
1.1 Overview	9
1.2 Motivation.....	11
1.3 Research Objective	13
RELATED APPLICATIONS AND PREDICTION MODELS	15
2.1 Banking and Finance	15
2.2 Human Resources	20
2.3 Air Quality and Environment	24
2.4 Industry Automation and Manufacturing	27
2.5 Computer Related Fields.....	31
2.6 Healthcare.....	34
IMPLEMENTATION PLATFORM AND LIBRARIES USED	37
DATASET	39
EXPLORATORY ANALYSIS	42
APPROACHES AND IMPLEMENTATION	49
6.1 Logistic Regression	50
6.2 Decision Tree.....	53
6.3 KNN	56
6.4 SVM	59
6.5 Artificial Neural Networks.....	62
EXPERIMENTS & RESULTS	66
7.1 Feature Engineering.....	69
7.2 Dimensionality Reduction	71
7.3 Principal Component Analysis.....	72
7.4 Hyperparameter Tuning.....	73
CONCLUSION & FUTURE WORKS	80
REFERENCES	81

List of Tables

Table 1: Results from Banking & Finance Domain	19
Table 2: Results from Human Resources Domain.....	22
Table 3: Results from Air Quality and Environment Domain	25
Table 4: Early-Stage Timing Prediction	30
Table 5: Predictive Maintenance of Industrial Machines	30
Table 6: Other Results in Industry Automation and Manufacturing	30
Table 7: Virtual Machine Failure Prediction	33
Table 8: Multi-Objective Model Predictive Control	33
Table 9: ML Pipeline to Optimally Utilize Limited Samples	33
Table 10: Machine Learning Framework for Cyber Security Operations.....	33
Table 11: Predictive Models for Air Quality Monitoring.....	35
Table 12: Novel Maximal Oxygen Uptake Prediction	35
Table 13: CO2 Emission Prediction Using Machine Learning.....	35
Table 14: Machine Learning-based Risk of Hospital Readmissions	35
Table 15: Dataset API Resources.....	40
Table 16: Initial Comparison of ML Models	68
Table 17: Final Results of ML Models	76

List of Figures

Figure 1: Flowchart explaining the sequence of events for this research	14
Figure 2: Raw data gathered from football-data.org.....	40
Figure 3: Final/structured data	41
Figure 4: Countries with most number of players	42
Figure 5: 'Overall' distribution among players.....	43
Figure 6: Further explorations	44
Figure 7: Best players according to the different skillsets	45
Figure 8: Player potential vs Age.....	46
Figure 9: Decreasing sprint speed when players age.....	47
Figure 10: Logistic Regression [33]	50
Figure 11: Logistic Regression Prediction	52
Figure 12: Example of a simple Decision Tree [34]	53
Figure 13: Decision Tree Initialization.....	54
Figure 14: Decision Tree Predictions	55
Figure 15: KNN Example.....	56
Figure 16: Distance Functions [33].....	57
Figure 17: Hamming Distance [33].....	57
Figure 18: Initializing KNN	58
Figure 19: KNN Predictions	58
Figure 20: A simple SVM [35].....	59
Figure 21: Support Vector Regressor [36].....	60
Figure 22: SVM Predictions	61
Figure 23: Artificial Neural Network	62
Figure 24: Neural Network Libraries.....	62
Figure 25: Neural Network Example	64
Figure 26: Accuracy solely based on odds	66
Figure 27: Data Used For Initial Experiments	67
Figure 28: Comparison of Results from Initial Experiments	68
Figure 29: Analyzing amount of data in each column	70
Figure 30: Dimensionality Reduction	72
Figure 31: Grid Search CV	73
Figure 32: Sample Input with New Attributes	74
Figure 33: Predicted Value.....	75
Figure 34: Mean Squared Error for ANN.....	78
Figure 35: Comparison of ML models implemented	79

CHAPTER 1

INTRODUCTION

1.1 Overview

Predictive modelling has found its applications in numerous domains over the years. These domains or industries vary from healthcare, finance, banking, process control etc. The implementations of the predictive modelling vary from use case to use case. However, their basic principle remains the same: typically looking to classify, categorize, and make predictions over the dataset. The applications and the domains in which these models can be used have incremented exponentially over the years. This provides for an exciting opportunity to study the implementation and utility of these models.

Sports is one such domain in which the use of predictive modelling is increasing day by day. Old data gathered throughout a games history is being used extensively in order to make smarter decisions in the future. Few examples of its applications include broadcasting, betting, drafting new players etc. A soccer dataset is hence gathered and used. Machine learning models are then implemented in order to demonstrate how better decisions can be made. And also to predict future events with the help of predictive analysis. A few optimizations are suggested that can be used to make the machine learning models a bit more efficient.

It is important to study existing models and their applications as it helps in gaining more knowledge. It also helps in optimizing existing models in order to extend their applications to new domains. This research project provides a pragmatic overview of the most commonly used prediction models and their utilities in different industries. Also, this project documents the findings that were observed after running experiments with the soccer dataset. Logistic

Regression, Decision Trees, KNN, SVM, Multinomial Naïve Bayes and Artificial Neural Networks are used in these experiments and present prediction accuracies ranging from 82% to 98%. In turn, this project serves as a brief literature survey and also provides examples of how machine learning can be useful for predictive analysis in the sports domain.

1.2 Motivation

Sports is a category in which competitions always keep scores and stats. These stats can help draw comparisons between the opponents involved. So the data generated after every year in the sports industry is huge. And so is the potential to use that data in order to improve future decisions. The current state of the industry does not rely too much on mathematical or statistical techniques for decision making, let alone the use of machine learning. Data has been gathered on a regular basis in almost all the sports. However, this data has mostly just been used for broadcasting purposes until recently.

It's quite evident now that teams are making decisions related to drafting, transferring, selection etc. by analyzing data from previous years. Take the case of Billy Beane, for example. He is the former General Manager for the Oakland Athletics, who used data analytics in order to draft a team that won 20 consecutive MLB games. Eventually, Billy was made an offer of \$12.5 million by the Boston Red Sox in 2002, making him the most expensive manager of the time. Billy used data related to players extensively. He made insightful observations related to not just the player's batting average but also other factors, like their on-base percentage. In other words, measuring not just the value but also the abilities of the players.

This helped him come up with a theory to challenge the conventional system in which the team with the most money picks the best players and wins the league. He managed to draft a team with almost one-tenth of the budget of other teams and do wonders. Observations, as astute as these, have helped transform so many sports in so many different ways. Just like Billy Beane and Oakland Athletics, there have been numerous examples where teams have used data available to them as practically as they can and achieved more of what was expected of them.

Also, the previous win/loss data can also be used in order to making bets over a team or a play. In places and sports where betting is legal, predictive analysis can be of great importance. These are just a few applications of using predictive analysis in this domain. It is still a growing field where data analytics has just started to build its foundations. Therefore, sports is an appropriate candidate domain to be studied and performed experiments over. In this research, the example of a widely spectated and followed sport like soccer is taken.

1.3 Research Objective

The possibilities of the applications of predictive models in various domains are numerous. Hence, there are a number of objectives that to be accomplished from this research. First, the previous work done in few of the domains which leverage from using data analytics and especially predictive models is being studied. The specific objectives of this research can listed as below:

- Gathering the dataset from a reliable and up-to-date source.
- Perform exploratory data analysis over the dataset.
- Understand data and observe the correlations that exists within the data, if any.
- Perform basic data pre-processing and data cleaning necessary.
- Explore the challenges of using predictive modelling with the soccer dataset.
- Use this data for predictive analysis and experimenting with the studied predictive models as initial experiments.
- Apply valid optimizations to the selected ML models.
- Test the models' performance again and compare them to each other.
- Plot the accuracy and other traits of the models in order to visualize them.

The figure below explains the flow of this research in the way it is being documented in this report:

Figure 1: Flowchart explaining the sequence of events for this research

CHAPTER 2

RELATED APPLICATIONS AND PREDICTION MODELS

2.1 Banking and Finance

One of the major key industries, banks and related financial institutions, are producing data at an exponential rate. Customers interact with these institutions for various services and through various mediums. This helps these institutions gather data in huge proportions. In turn, these financial service providers use this gathered data in different ways in order to make their business more profitable [1].

This data may be generated by several channels such as online banking, ATM machines, online shopping etc. and may be present in different format. Yet, it's more than beneficial for these companies to leverage the gathered data than discarding it. Predictive modelling help financial institutions to minimize their costs and reduce risks in the market in several ways [2]. One of the way it helps in reducing risks is by helping banks and other financial service providers corroborate or verify the creditworthiness of a management or company. This is generally referred to as bankruptcy prediction.

Authors of [1], [2] and [3] document studies related to bankruptcy prediction. Bankruptcy has a lot of obligations involved, like repayment, security obligation, elimination of property etc. This is one of the critical issues faced by financial decision makers. The failures affect not only the customers but also the management and stockholders of these institutions [2]. The prediction model basically has to categorize whether or not a business will run into bankruptcy. This is an example of a binary classification problem. Due to the high stakes involved, most of these models or techniques need to be improved from time to time [1].

In [1], both statistical and machine learning based approaches are used to create predictive models, or in other words, to make accurate predictions. The overhead to handle large datasets is also something that presents itself as a challenge in a lot of scenarios. Even in this case, the dataset used by the researchers was huge. This needed statistical methods for better analysis and in turn, implementation of statistical models. Linear Discriminant Analysis (LDA), Multivariate Discriminant Analysis (MDA) and Logistic Regression (LR) are used as the models that fall under this statistical models category.

Machine learning, the vastly trending branch of Artificial Intelligence, has also found its usage in bankruptcy prediction. It basically integrates the idea that machines can learn from the data and with minimum human input. And it can make decisions after identifying patterns from the data. In [1], [2] and [3], machine learning is used to build analytical models that help in the binary classification between bankruptcy and non-bankruptcy. Artificial Neural Network (ANN), Support Vector Machine (SVM) and Decision Trees are the machine learning algorithms used in this research. These models generally give better accuracy with datasets of smaller size.

Along with these machine learning techniques, Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are used. These optimization techniques are used in order to decrease error and improve the model's accuracy. These are especially useful in case of large datasets. [4] presents a similar case to build models that help us in solving the challenges in the banking and finance domain. Specifically, the researchers have used a ML model to perform predictive model on a banking dataset.

The dataset was gathered by the authors from the University of California, Irvine Machine Learning repository. It contains data related to customers; what the customer like/prefers. This helps the banks or any other service provider to see a correlation between the product/services they are offering. In turn, this helps them to conclude whether the customer will like the products or services, or not. This vastly helps to improve the institute's services and also help customize customer's experience. For their experimentation, the researchers used Amazon S3 data service where data from the data source was stored. This dataset had undergone data cleaning & other data pre-processing steps. The researchers also replaced the binary attributes Yes and No with 1 and 0 respectively.

There are three basic models that are supported by Amazon primarily with S3: regression model, binary classification and multiclass classification models. The approach used by the authors of [4] involved 3 phases. The first phase involves choosing the data source i.e. choosing the data to input from the S3 bucket that was uploaded to AWS after all the pre-processing. The second phase consists of creating a Recipe. There are several decisions to be taken before actually using the data to train the model and to use it to predict something. Like the percentage of the data to be split for training and testing, whether the splitting of the data would be random or sequential in nature, etc.

This phase to set evaluation parameters is basically the Recipe creation phase. It's a general term for parameterization involved in creating an ML model. Parameterization involves deciding max model size, max passes over transaction data, shuffle type, regularization type & amount etc. The third step is to basically evaluate the machine learning model. There are several

metrics that can be used for model evaluation. The authors of [4] have chosen to use Area Under the curve as their metric. AUC gives a value between 0 and 1 to signify the accuracy, precision and recall.

KamalaKanta Mishra et al. further extend the benefits of using predictive models in the banking domain. One of the well-known problem they realized was a lack of systems that predict device failures. The authors have used a similar concept in [5] to come with a predictive model to predict the failure likelihood of various banking devices and components like ATMs. Regularized Random Forest is the machine learning model used for forecasting device failures by them. The findings and evaluations in [4] are significantly well documented and serve as a standard guideline for tackling similar issues.

The results obtained by various researchers can be summarized as follows:

<u>ML and statistical techniques in Bankruptcy Prediction</u>			
<u>Sr.</u>	<u>Model Used</u>	<u>Accuracy (%)</u>	<u>Precision (%)</u>
1.	Linear Discriminant Analysis (LDA)	75.47	68.75
2.	Logistic Regression (LR)	79.25	71.87
3.	Decision Trees	86.79	78.12
4.	Learning vector quantization (LVQ)	87.5	85
5.	Artificial neural networks (ANN)	90	94.11
6.	SVM	90	94.11
7.	SVM-GA (genetic algorithm)	92.5	94.44
8.	SVM-PSO (particle swarm optimization)	95	94.73

<u>Predictive Analytics for Banking User Data using AWS</u>			
<u>Sr.</u>	<u>Model Used</u>	<u>Threshold value of 0.5</u>	<u>Threshold value of 0.77</u>
1.	Linear Regression	False Positive rate – 0.0289 Precision – 0.6486 Recall 0.4307 Accuracy 0.9115	False Positive rate – 0.0076 Precision – 0.7594 Recall 0.1922 Accuracy 0.9042
<u>Failure Prediction Model for Predictive Maintenance</u>			
<u>Sr.</u>	<u>Model Used</u>	<u>Probability of Failure</u>	
1.	Regularised Random Forest (RRF)	0.259	

Table 1: Results from Banking & Finance Domain

As evident from Table 1, in case of bankruptcy prediction, Support Vector Machines provides the best results. The researchers have used SVM with Genetic Algorithm and Particle Swarm Optimization techniques which make them so efficient. By looking at these three experiments, it can be concluded that machine learning algorithms provide better results when used in bankruptcy predictions, than statistical models.

2.2 Human Resources

Predictive analytics can also be applied to domains where human presence and interaction is highly involved. In case of human resources, it is important that suitable arrangements be made in order to keep the manpower of the company sufficiently appropriate. So that the manpower is in necessary quantity to the demand of the workplace [6]. But as far as most of the workplaces are concerned, it is not only the quantity that makes a difference. The quality of manpower is also a distinguishing factor in almost all of the industries.

Skilled labor is an important asset for any employer or any institute, be it a commercial profit making business, or a not-for-profit organization. Basically, the availability of high skilled labor is essential for the survival of any kind of business [7]. And the importance of high-skilled workers is not only realized by individual organizations alone. Developing nations see these skillful workers as the foundation over which the nation is to be built upon further. These skills are acquired by individuals by gaining valuable experience through the years and/or by continuously immersing themselves in a learning process.

Whenever skilled workers switch jobs, it creates a major skill gap in the organization. This is a big problem for not just the organization but if the migration is on a global scale, it brings up a lot of challenges for the country itself. Researchers in [6], [7], [8] figured out that this is where machine learning algorithms can play a vital role, to create predictive models which can help in making useful predictions regarding human resources. The researchers have experimented with models in various ways.

One of the experiments they conducted was to find out or predict if a skill gap, created by a skilled worker leaving the organization, can be filled by some other person [6]. For this, authors of [6] and [7] have analyzed the historic data of the workers of the organization. For their implementation in [6], they have taken a dataset from the FIFA2017 game. This dataset contains numerous attributes related to the players. Typically, during a football season, players can transfer to other teams and other teams can lure players by offering better wages and transfer bonuses. The teams which loses its player needs to find a suitable replacement in this case.

This is a more specialized and specific version of a general problem attaining to the replacement of a skilled worker by an equally or a better skilled worker. This is the problem researchers have tried to portray by using the FIFA2017 dataset and by trying to find an identical replacement of a player when he leaves a team. The dataset contains several different attributes related to the player. There are basically 53 characteristics of 17,588 players. These characteristics include labelled data like name, nationality, club, playing position in the team (like forward, backward, midfield, goalkeeper etc.).

Other than these characteristics, there are few more characteristics which are quantitative in nature. Quantitative values like height, vision, rating, aggression, ball control, weight, marking, ball sliding, etc. correspond to the skillset of the player and help us evaluate him quantitatively. The model developed by the researchers primarily takes into account the “rating” of the players which can be seen as the overall score assigned to the player based on his various skills. There were few pre-processing steps that were undertaken before the actual experiments.

Pearson correlation co-efficient was used to measure the statistical association, or relationship between these continuous variables. Other pre-processing measures include removing null data, removing irrelevant or invalid data, categorization of players etc. Machine learning models used by the researchers aim to take into consideration the quantitative characteristics and predict whether the new player considered is appropriate, or suitable to fill the skill gap of the transferred player. Authors of [8] have used a similar approach for student placement purposes to evaluate and advice students to adhere to the most appropriate job/program based on his/her aptitude.

Researchers in [6], [7] and [8] have used a number of machine learning algorithms to create their predictive model. Binary classification models and multi-class classification models which include AdaBoost, Decision Tree, Gradient Descent, k Nearest Neighbor, Linear Discriminant Analysis, Logistic Regression, Multi-Layer Perceptron, Naïve Bayes, Random Forest, Stochastic Gradient Descent Classifier and Support Vector Machine are used. The accuracy and other evaluating factors are plotted and the effect of number of classes and number of features were also taken into account. Table 2 gives a summary of the findings:

<u>Human Resource Predictive Models</u>		
<u>Sr.</u>	<u>Model</u>	<u>Accuracy</u>
1.	Adaboost	0.01
2.	Decision Tree	0.83
3.	Gradient Boost	0.82
4.	KNN	0.06
5.	LDA	0.86
6.	LR	0.01
7.	Multi-layer perception	0.03
8.	Naive Bayes	0.81
9.	Random Forest	0.19
10.	Stochastic gradient descent classifier	0.03

Table 2: Results from Human Resources Domain

From Table 2, it is quite evident that Linear Discriminant Analysis performs well for various experiments performed for the Human Resources domain. Also, Naïve Bayes outperforms a number of other predictive models. From the results of this research it can be observed that the efficiency of statistical models in these experiments is way better than machine learning models.

2.3 Air Quality and Environment

Many a times, road traffic and air traffic is heavily impacted by conditions pertaining to low visibility [9]. For meteorologists, this is still a big problem. Especially for the spatial coverage, low visibility presents some big challenges. Adverse low visibility also causes numerous problems in the aviation or aircraft industry. At major/busy airports, poor weather conditions coupled by poor visibility restrict the flow of air traffic. This causes major aircraft delays in a lot of cases. Many a times, due to such adverse conditions, the aircrafts have to be diverted to other airports which costs considerably in terms of fuel and other expenses for the airlines.

Hence, there is a need for improvement when it comes to forecasting visibility conditions. Historically, physical Numerical Weather Prediction (NWP) models were implemented by meteorologists to overcome these problems. There is a high level of complexity involved due to the interactions between the physical processes, especially at the airports at the time of low visibility conditions. This makes successfully gaining precise predictions over visibility forecasts still a major issue.

Data mining is seen as another approach to solve this problem. In [9], the study has applied ML regression techniques to outputs of the kilometric NWP. The suitability of KDD (Knowledge from Database Discovery) can be derived from this proposed method for spatial visibility forecast. The output of the operational 3D NWP model AROME is extracted and is used as the primary dataset for this paper. Researchers have tackled similar visibility forecast problems by using a few of the data mining tools and methods like deep learning and boosting tree-based regression in order to train the data[9], [10]. Also, Extreme Gradient Boosting (XGBoost) can be used in order to develop the boosted trees as used by the authors of [10].

One of the other creative way in which researchers have found the use of predictive models is to predict the amount of rainfall that'll be present at a given time [11]. For this, the researchers have used an acoustic disdrometer which is a tool that measures the amount of rainfall through the sound produced as the raindrops hit the piezoelectric sensors. Piezoelectric effect is the ability depicted by a handful of materials to generate electricity from mechanical forces or mechanical stress and this is the same concept that is used by piezoelectric sensors (*source: nanomotion.com*).

The main aim of the paper [11] was to build predictive models used to categorize the intensity of the amount of rainfall by obtaining audio data from the ambient noise. The amount of rainfall defines the intensity category in which it falls. There were 5 categories which were defined by researchers: light, moderate, heavy, intense, torrential. Various machine learning models viz. Support Vector Machine, k Nearest Neighbor and Naïve Bayes have been used in order to create a predictive model that achieves this aim. Further, cross-validation and out-of-sample validation is used to evaluate the model and to test its assumptions.

Table 3 gives an overview of the major predictive models used under this domain:

<u>Air Quality and Environment</u>		
<u>Sr. No.</u>	<u>Model</u>	<u>Accuracy</u>
1.	SVM	99.14
2.	KNN	99.14
3.	Naïve Bayes	89.27

Table 3: Results from Air Quality and Environment Domain

Support Vector Machine and k Nearest Neighbor models provide the best results for the experiments studied in this domain. Naïve Bayes is another model with which experiments were performed but gives comparatively low accuracy. However, the range of accuracy seems fair and the results are accurate.

2.4 Industry Automation and Manufacturing

Predictive analytics can help manufacturers identify problems in advance and that proves to be the biggest advantage of including predictive models in the manufacturing process. This ability to foresee failures also gives manufacturers a good idea about the timing closure. [12] talks about the importance of the importance of timing closure in the physical design of a System on a Chip (SoC). A SoC is nothing but a microchip or an IC with the necessary components of an electronic system, like a computer or a smartwatch, integrated on it.

As the CMOS (Complementary metal–oxide–semiconductor) technology keeps on scaling, the complexity of SoC also keeps on increasing. Timing is very critical for the products to be put on the market shelves. The actual design flow of an SoC or any IC basically involves a number of important steps including blueprint creation, placement along with floorplan, routing, clock-tree synthesis and finally verification. Verification and simulation, the last phase is the one that takes up most of the time and every time a floorplan is set, all the following step are repeated in their respective order in order to meet the timing requirements.

This is where predictive analytics can help, as per authors of [12]. To correct early mistakes, if the final timing of the product can be predicted in the floorplan phase itself, it may be possible to avoid the rest multiple iterative and repetitive interactions. For this, the researchers have considered the middle processes, the ones after the floorplan creation, as a black box and have done a time analysis after the floorplan is created. Back Propagation Neural Network, Support Vector Machine, AdaBoost with Back Propagation, AdaBoost with Support Vector Machine and Hybrid Model are used by the authors to create the predictive model for this task.

Industrial IoT or IIoT is a term that seems to be very trending these days. Manufacturing process completed with the help of IoT technologies is IIoT. Authors of [13] have described how different data analytics can be applied over the machine generated data & harness it as per our usage. They have used a slitting machine as their data source. Basically, they have gathered data which they received upon using the machine which is used for rewinding and cutting package films, the slitting machine. There are various sensors that can be integrated with this machine. These sensors then produce a time-series data.

Generally, when it comes to time-series data, researchers tend to go with something like ARIMA, which stands for Auto Regressive Integrated Moving Average, as demonstrated in [13] and [14]. ARIMA is a class of model that are generally fitted to time-series data and can capture the essence or complexity of the time-series data to help understand it better. It also makes it easier to predict future points in the series (*source: Wikipedia.com*). This helps in increasing the overall quality of manufacturing by predicting quality defects and possible failures and in turn, improving quality control.

The authors of [13] have used the Daisy chaining concept for setting up their system. Daisy chaining can be considered as a wiring scheme which help in reducing the complexity of the system by decreasing the quantity of wiring between the Programmable Logic Controller and other machines. The data from the slitting machine was gathered for a one month period and sampled per second. Port RS485 present on the machine sensors transmit the data to an adaptor which feeds it into a IPC after converting it into TCP form. The IPC or Industrial Personal Computer is used to store the data locally. MQTT protocol is used by IPC to transfer the data to the cloud in the form of packets.

The sensors transmit the data whenever there is a change in the state of the machine [14]. So, as expected, a lot of null values representing the seconds in which there was no state change are present. Hence, the first step was to pre-process or clean the data which involved replacing the null values with sliding mean values. Also, outlier detection methods were used to study the outliers existing in the data using clustering methods. The file stored in the cloud is in the form of a CSV and has columns like timestamp, tension, pressure, width, & diameter; all pertaining to the slitting machine characteristics.

To come up with a more accurate prediction over the future state of the machine, the researchers can follow two stages as talked about in [13] [14] [15] [16]. The first stage is to gain useful insights/business insights into the data by using data analytics, supervised learning methods, clustering algorithms based on the need and efficiency. During the second stage, ARIMA is used after following the first stage to add predictive analytics capabilities to the final predictive model. Since the time-series data shows non-stationarity, in order to forecasting the machine parameters so as to map the future states of the machine, ARIMA plays a vital role.

As in the case of any other machine learning model, historic data, generated by the slitting machine and gathered for a months' duration, is used to train the supervised models. ARIMA is also trained using the same dataset. The dataset split was 70% for test, 20% for training and 10% for cross-validation. In the end, all these approaches are stacked together as one system. The machine learning models used include Naïve Bayes (NB), Support Vector Machine (SVM), Classification And Regression Trees (CART) and Deep Neural Network (DNN). The observations are summarized below:

<u>Early-Stage Timing Prediction</u>	
<u>Model Used</u>	<u>Mean Squared Error</u>
BP	3.7 (Mean square error)
SVM	3.4
Adaboost with BP	3.5
Adaboost with SVM	3.2
Hybrid Model	3.1

Table 4: Early-Stage Timing Prediction

<u>Predictive Maintenance of Industrial Machines using IoT Sensor Data</u>	
<u>Model Used</u>	<u>Accuracy</u>
Naive Bayes	96.61
Support Vector Machine	95.52
CART	94.46
Deep Neural Network	98.69

Table 5: Predictive Maintenance of Industrial Machines

<u>Other results in Industry Automation and Manufacturing</u>	
Least-squares support-vector machine (LSSVM)	0.146% (mean relative error)
Model predictive control (MPC)	$U_0 = 0, \tau_1 = 20, \tau_2 = 1 \times 10^3, q = 1 \times 10^3, r = 0.1.$

Table 6: Other Results in Industry Automation and Manufacturing

Table 4 gives an overview of experiments that were performed to predict the stage timing of a manufacturing process as early as possible. As observed, a hybrid approach which can be an ensemble of ML models can be considered the best as it gives the lowest mean squared error. Table 5 and 6 shows that deep neural network can be considered and favored for predictive maintenance of IIoT machines. Though other models are efficient too, but comparatively, deep neural networks perform the best.

2.5 Computer Related Fields

Virtualization has seen a big boost in its usage as the demand and utilization of cloud-based environments increase [17]. Virtualization is a technology that enables user to run a virtual instance of a system including storage, applications, servers, networks etc. These instances are hidden or abstracted from the underlying actual hardware layer (*source: opensource.com*). However, as the cloud platforms are trying to integrate more and more features and more and more environments, the complexity of such platforms have also increased. This in turn, results in increased failures in general operations of virtual machines.

Authors of [17] have proposed that predictive modelling can be used to guarantee the reliability of the virtual machines. There are several predictive modelling methods that can be utilized in order to achieve this. In [17], a Hidden Markov Model (HMM) based predictive model is used along with Ada-Boost for foreseeing the failures in VMs. HMM is a probabilistic model used to model basically the sequential or continuous data. HMM helps identify or predict a sequence of hidden or unknown states (viz. unknown variables) after being provided a set of observed states or observed variables.

AdaBoost is a boosting algorithm that is used with a number of machine learning algorithms in order to increase their performance. It is especially helpful in converting weak learner or weak classifiers to strong ones. Various researchers use AdaBoost in conjunction with HMMs for better results [6] [12] [17]. In [17] AdaBoost is used to realize or identify the observation states and HMM was used to establish the security state of the VM. Table 5 gives an overview of the observations from these experiments.

In most of the cases, predictive modelling can help predict future outcomes as much precise as it can get in order to gain substantial insight and help make better decisions [18]. However, it requires a large amount of data to be gathered in order to get anything useful out of the system. Several times, a research might just end up tediously time consuming and costly without any fruitful end [19]. Many statistical models are unfit or unable to function properly without a huge amount of data. Same is the case with several machine learning models as well.

There have been few studies for optimally utilizing the limited data samples as discussed in [19]. These studies try to increase the profit to be gained from the small amount of datasets in order to efficiently leverage it. The aim of authors of [19] was similar; to gain valuable knowledge in order to know the value or weigh of the input variable. For this, they have implemented a machine learning pipeline to further improve the predictive performance of the system.

The researchers have used two popular publicly available datasets: a UCI ML repository dataset titled the 'Wine Dataset' and a Seaborn dataset titled the 'Titanic Dataset'. After pre-processing and cleaning the data, the researchers have used machine learning algorithms to design their pipeline. Support Vector Machine has been used to simultaneously produce and observe the classification accuracy. This is thought to be an optimal or an ideal way to distinguish the most prescient variable measurement and blend. This also helps in better classifying individual data samples by providing a generalizable parametric detailing.

Their results, summarized below, give an idea of the importance and traits of the proposed ML pipeline:

<u>Virtual Machine Failure Prediction</u>	
Hidden Markov Model	0.443 (Error), 0.205 (Failure)

Table 7: Virtual Machine Failure Prediction

<u>Multi-Objective Model Predictive Control</u>		
<u>Models</u>	<u>Control Time</u>	<u>Energy</u>
SVM	J1:T=30	J2:E=1000
SVR	J1:T=15	J2:E=3000
C-SVR	J1:T=10	J2:E=800

Table 8: Multi-Objective Model Predictive Control

<u>ML Pipeline to Optimally Utilize Limited Samples</u>	
SVM	0.877 (Mean AUC)

Table 9: ML Pipeline to Optimally Utilize Limited Samples

<u>Machine Learning Framework For Cyber Security Operations</u>	
MNN	5.52 (Average top % predictions)
RF	4.74
SVM	4.56
LR	5.23

Table 10: Machine Learning Framework for Cyber Security Operations

The above tables provide details of the results which the researchers got after performing various experiments related to virtual machines and cyber-security. It can be fair to conclude from the above tables that Support Vector Machines and Support Vector Regressor provide the best results in these experiments. Notably, Hidden Markov Models are efficient too and can provide almost the same amount of accuracy as SVMs or SVRs.

2.6 Healthcare

The United States Environment Protection Agency (US EPA) uses Air Quality Index (AQI) for reporting daily air quality. AQI helps to standardize air quality measures. It is generally used to communicate or forecast to the public the amount of pollution. In other words, it provides details of how polluted the air is [23]. It is a good index to have as it helps estimate the public health risk that increases with increasing poor air quality. The levels of health concerns according to AQI are: good, moderate, unhealthy for sensitive, unhealthy, very unhealthy and hazardous. However, to come up with a very good estimation, AQI requires complex calculations from precise readings from the sensors.

This large-scale calculation is not always feasible for portable air quality monitoring devices [24]. The aim of [23], [24] is to find alternate ways of monitoring & characterizing air quality. One of the ways they do this is by using gas sensors and building ML prediction models. [23] proposes a methodology of characterizing air quality by building predictive models to relate sensor values to AQI. The process involved in this approach include several phases: hardware development followed by data gathering. And the final phase involves creating predictive models. ML algorithms used for creating a prediction model include k Nearest Neighbor (KNN), Support Vector Machine (SVM), Naïve Bayes (NB), Random Forest (RF), Neural Network (NN).

Throughout the years, various studies have shown that CO₂ is one of the main reasons of changing temperatures and climate, the biggest one actually [25]. Statistical techniques like t-test, ANOVA test, regression, derivation etc have traditionally being used for prediction of CO₂ emissions. [25] provides various ML based techniques in order to predict CO₂ emission. A regression model is used in this study to predict CO₂ emissions and Root Mean Square Error

(RMSE) is used to validate the results. The dataset is taken from World Bank data from 1960 to 2014 in which the CO₂ is measured in Kiloton(kt). Data is normalized and then split into training (70%) and testing(30%) after which a linear regression model is used to make predictions. Cross-validation technique is used to evaluate the model and RMSE is used to check the precision of the results.

<u>Predictive Models for Air Quality Monitoring</u>	
GRNN	8.30 (SEE) 0.35 (R)
SVM	8.14 (SEE) 0.38 (R)
DTF	8.68 (SEE) 0.29 (R)
RBF	9.33 (SEE) 0.18 (R)

Table 11: Predictive Models for Air Quality Monitoring

<u>Novel maximal oxygen uptake prediction</u>	
Regression Model	0.2557 (Root Mean squared error RMSE)

Table 12: Novel Maximal Oxygen Uptake Prediction

<u>CO₂ emission prediction using machine learning</u>	
K-Nearest Neighbors	0.9867 (Confusion Accuracy)
Polynomial SVM	0.9778 (confusion Accuracy)
Random Forest	0.9867 (Confusion Accuracy)
Naive Bayesian Classifier	0.9422 (Confusion Accuracy)
Neural Network	0.9956 (Confusion Accuracy)

Table 13: CO₂ Emission Prediction Using Machine Learning

<u>Machine Learning-based Risk of Hospital Readmissions</u>	
LACE	0.702 (Area Under the Receiver Operating Characteristics)
PARR	0.669 (Area Under the Receiver Operating Characteristics)

Table 14: Machine Learning-based Risk of Hospital Readmissions

From the above tables, it can be observed that all these experiments used different measures for measuring the efficiency. The tables give an overview of predictive models for air quality monitoring, out of which, Support Vector Machines have the least error rate. This makes it more preferable than the other models. The experiment performed for novel oxygen uptake prediction used only one model, Logistic Regression, which had a mean squared error rate of 0.26. The experiments involving CO₂ emission predictions had neural network with the accuracy of 99.56%. This was the best result out of all the experiments performed.

CHAPTER 3

IMPLEMENTATION PLATFORM AND LIBRARIES USED

Predictive analytics can be used in Sports domain as well, similar to as discussed for the above domains. The approach may vary as far as the implementation is concerned. Data gathered historically over time from previous games, or players, can be used to make various predictions. As in the case of baseball, basketball, hockey, soccer or any other sport, the use of previous data to determine future events is becoming a trend. Experts from the big data and data analytics domain are in big demand because of the growing need for analyzing historic data.

In this research, soccer is considered as the sport for which predictions are to be made based on the data which is available. The details regarding the dataset are included in chapter 4 of this report. For programming purposes, python has been selected as the primary language. Although few JavaScript snippets were used in the beginning in order to optimize the kernel provided by Google Colab in order to perform some operations. But soon they were later replaced by few lines of python code.

As suggested above, primarily, Google Colab was used in order to utilize the advantages of a cloud platform and also the extra GPU power that it provides. But there were a lot of complexities hindering the smooth usage of jupyter notebooks hosted over Colab. Especially scrapping/importing dataset and re-using them was not so easy. The dataset had to be saved on Google drive and then imported or could be imported directly from the PC. But this needed to be done every time notebook was refreshed or kernel restarted. The same was the issue with Kaggle kernels which provided less computation power too.

Several open-source libraries like scikit-learn, numPy, Pandas, matplotlib, Seaborn, eli5, and sciPy were used extensively. Along with these libraries, I had to use few other libraries like sys, os, http, csv, codecs etc. to scrape, encode and write the data. Finally, Anaconda was chosen to be the ideal platform to complete the experiments of this research. Anaconda, to be simply put, is a data science platform which is open source and supports python as well as R. The system used had 16 GB RAM with Intel i7 (8th Gen) 2.11 GHz processor and 64-bit Windows 10 operating system.

CHAPTER 4

DATASET

The dataset used for this project was scrapped from football-data.org. It is a website dedicated to providing up-to-date statistics and data related to football (*a.k.a. soccer in other parts of the world except US, and to be used as a placeholder for soccer hereafter in this paper*). The website is updated every week with the latest matches and player statistics. Also, player transfer data, squad and lineups, and also match results from football leagues around the world are included.

The data from the website basically needs to be scrapped using RESTful APIs. The endpoints (URI) vary and can be looked up in the documentation provided on the website itself. Few of the resources that one can access, along with the URI and available filters are as follows:

Resource	Action	URI	Filters
Competition	List all available competitions.	/v2/competitions /	areas={AREAS} plan={PLAN}
Competition	List one particular competition.	/v2/competitions /2000	
Team	List all teams for a particular competition.	/v2/competitions /{id}/teams	season={YEAR} stage={STAGE}
Standings	Show Standings for a particular competition.	/v2/competitions /{id}/standings	standingType={standingType}
Match	List all matches for a particular competition.	/v2/competitions /{id}/matches	dateFrom={DATE} dateTo={DATE} stage={STAGE} status={STATUS} matchday={MATCHDAY} group={GROUP} season={YEAR}
Scorers	List goal scorers for a particular competition.	/v2/competitions /{id}/scorers	limit={LIMIT}
Match	List matches across (a set of) competitions.	/v2/matches	competitions={competitionIds} dateFrom={DATE} dateTo={DATE} status={STATUS}

Match	Show one particular match.	/v2/matches/{id}	
Match	Show all matches for a particular team.	/v2/teams/{id}/matches/	dateFrom={DATE} dateTo={DATE} status={STATUS} venue={VENUE} limit={LIMIT}
Team	Show one particular team.	/v2/teams/{id}	-
Areas	List all available areas.	/v2/areas/	-
Areas	List one particular area.	/v2/areas/{id}	-
Player	List one particular player.	/v2/players/{id}	-
Player	Show all matches for a particular player.	/v2/players/{id}/matches	dateFrom={DATE} dateTo={DATE} status={STATUS} competitions={competitionIds} limit={LIMIT}

Table 15: Dataset API Resources

Table 15 provides a list of URIs that can be used to send a GET request to and gather the data one by one. Along with the URIs, various filters are also mentioned in the above image which can be used to filter out specific data. The website was intended to be free of cost which is why data from 12 leagues and competitions are provided free of cost. Though website requires users to sign up and pass the API key received in the email as a header for the HTTP request. There are only 10 API calls allowed per minute under the free tier. This is sufficient to gather the data that is required for a small scale project like this one. This is the example of a part of data obtained originally:

```
print(response)

{'count': 149, 'filters': {}, 'competitions': [{'id': 2006, 'area': {'id': 2001, 'name': 'Africa', 'countryCode': 'AFR', 'ensignUrl': None}, 'name': 'WC Qualification', 'code': None, 'emblemUrl': None, 'plan': 'TIER_FOUR', 'currentSeason': {'id': 555, 'startDate': '2019-09-04', 'endDate': '2021-11-16', 'currentMatchday': None, 'winner': None}, 'numberOfAvailableSeasons': 2, 'lastUpdated': '2018-06-04T23:54:04Z'}, {'id': 2023, 'area': {'id': 2011, 'name': 'Argentina', 'countryCode': 'ARG', 'ensignUrl': None}, 'name': 'Primera B Nacional', 'code': None, 'emblemUrl': None, 'plan': 'TIER_FOUR', 'currentSeason': {'id': 547, 'startDate': '2019-08-16', 'endDate': '2020-06-14', 'currentMatchday': 20, 'winner': None}, 'numberOfAvailableSeasons': 3, 'lastUpdated': '2020-03-06T00:00:08Z'}, {'id': 2024, 'area': {'id': 2011, 'name': 'Argentina', 'countryCode': 'ARG', 'ensignUrl': None}, 'name': 'Superliga Argentina', 'code': 'ASL', 'emblemUrl': None, 'plan': 'TIER_TWO', 'currentSeason': {'id': 539, 'startDate': '2019-07-27', 'endDate': '2020-03-01', 'currentMatchday': 22, 'winner': None}, 'numberOfAvailableSeasons': 3, 'lastUpdated': '2020-03-06T00:00:07Z'}, {'id': 2025, 'area': {'id': 2011, 'name': 'Argentina', 'countryCode': 'ARG', 'ensignUrl': None}, 'name': 'Supercopa Argentina', 'code': None, 'emblemUrl': None, 'plan': 'TIER_FOUR', 'currentSeason': {'id': 430, 'startDate': '2019-04-04', 'endDate': '2019-04-04', 'currentMatchday': None, 'winner': None}, 'numberOfAvailableSeasons': 2, 'lastUpdated': '2019-05-03T05:08:18Z'}, {'id': 2147, 'area': {'id': 2014, 'name': 'Asia', 'countryCode': 'ASI', 'ensignUrl': None}, 'name': 'WC Qualification', 'code': None, 'emblemUrl': None, 'plan': 'TIER_FOUR', 'currentSeason': {'id': 465, 'startDate': '2019-06-06', 'endDate': '2021-11-16', 'currentMatchday': 7, 'winner': None}, 'numberOfAvailableSeasons': 2, 'lastUpdated': '2020-02-14T17:22:50Z'}, {'id': 2008, 'area': {'id': 2015, 'name': 'Australia', 'countryCode': 'AUS', 'ensignUrl': None}, 'name': 'A League', 'code': 'AAL', 'emblemUrl': None, 'plan': 'TIER_TWO', 'currentSeason': {'id': 548, 'startDate': '2019-10-11', 'endDate': '2020-05-17', 'currentMatchday': 22, 'winner': None}, 'numberOfAvailableSeasons': 2, 'lastUpdated': '2020-05-17T00:00:00Z'}]
```

Figure 2: Raw data gathered from football-data.org

The `http.client` was used in order to create a connection to the site and the token received via email was passed as a header. As evident from Figure 2, the data returned was in json format and had numerous nested json. It is pretty difficult to work with a dataset like this. A few lines of code and few encoding libraries were required to make this data more useful and readable. The scope of this project does not require the complete data from the API.

Player data and the data from previous matches the player participated in was to be used primarily, to make predictions related to the matches and the players. Hence, not all the (sub)resources were queried and only a handful of them were queried in order to gather the required data. In order to achieve the objectives of using regression based ML models, it is better if the data is organized. Therefore, the scraped data was organized into rows and columns using a couple of code snippets and libraries like csv, codecs, utf etc. The data looked something like this in the end:

In [8]: `df.head()`

Out [8]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Composure	Marking	StandingTackle	...
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	96.0	33.0	28.0	...
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	95.0	28.0	31.0	...
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	94.0	27.0	24.0	...
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	68.0	15.0	21.0	...
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	88.0	68.0	58.0	...

5 rows x 89 columns

Figure 3: Final/structured data

Figure 3 shows a part of the dataset which gives details about various players. It also displays the various attributes that are associated with the players. Attributes such as Age, Overall, Potential etc. are very useful for the experiments.

CHAPTER 5

EXPLORATORY ANALYSIS

Upon successful gathering of data, the next step important task is to look into the data in order to observe it. It is also important so as to decide the approach to follow, for making predictions using this data. A number of explorations are done on the dataset along with including respective visualizations to get a clearer picture. This basically helps in converting raw facts into some useful information which can be further utilized for various purposes.

In order to gain a better insight into the data, it was important to look into it and dig better co-relations within it. This was also helpful to extract useful attributes. First the common observations that could've been gathered were plotted. Observations, such as the number of players from a country were found and this observation was visualized as shown below:

Figure 4: Countries with most number of players

As evident from Figure 4, maximum number of players listed in this dataset are from England, followed by Germany and Spain. In fact, the English Premier League and the German Bundesliga are the most spectated and followed league in the world. This fact is supported by the high number of soccer players these countries produce as evident from the figure. Although other than providing an idea for the quantity of players and the fan base, these statistics do not have much of an impact on the win/loss ratio of the clubs. Therefore, other statistics were looked into and related correlations were derived.

Figure 5: 'Overall' distribution among players

Figure 5 shows how the 'Overall' attribute of the players was distributed. Almost all the players listed under this dataset had an Overall of greater than 55. It can be said that this dataset consists of well-known players, given the fact that many of the players belong to popular countries and

also observing that many of the players have Overall more than 60. Although, for every player the Overall keeps on changing with every season, yet it can be seen that most of them have an Overall between 60 and 75.

Similarly, other correlations such as the number of players in each country, the number of players in each club, the player with maximum potential, & the player with maximum overall performance were found:

```
In [213]: print('Total number of clubs : {0}'.format(df['Club'].nunique()))
          print(df['Club'].value_counts().head(5))

Total number of clubs : 651
Real Madrid           33
Manchester City       33
Wolverhampton Wanderers 33
Fortuna Düsseldorf   33
AS Monaco             33
Name: Club, dtype: int64

In [214]: print('Maximum Potential : '+str(df.loc[df['Potential'].idxmax()][1]))
          print('Maximum Overall Performnce : '+str(df.loc[df['Overall'].idxmax()][1]))

Maximum Potential : K. Mbappé
Maximum Overall Performnce : L. Messi
```

Figure 6: Further explorations

Figure 6 provides further exploration of the dataset. It is important to know the players and their attributes, as these big players make a significant difference in deciding the fate of a match. The more number of these well-known and highly skilled players a team has, the better the chance of its winning.

Figure 7 gives an overview of a similar observation in which the best players according to the different skillset was found. For example, L. Messi is the best when it comes to finishing so the best player in this category is him. Similarly, the best player from each category was found out:

Best Crossing : K. De Bruyne
 Best Finishing : L. Messi
 Best HeadingAccuracy : Naldo
 Best ShortPassing : L. Modrić
 Best Volleys : E. Cavani
 Best Dribbling : L. Messi
 Best Curve : Quaresma
 Best FKAccuracy : L. Messi
 Best LongPassing : T. Kroos
 Best BallControl : L. Messi
 Best Acceleration : Douglas Costa
 Best SprintSpeed : K. Mbappé
 Best Agility : Neymar Jr
 Best Reactions : Cristiano Ronaldo
 Best Balance : Bernard
 Best ShotPower : Cristiano Ronaldo
 Best Jumping : Cristiano Ronaldo
 Best Stamina : N. Kanté
 Best Strength : A. Akinfenwa
 Best LongShots : L. Messi
 Best Aggression : B. Pearson
 Best Interceptions : N. Kanté
 Best Positioning : Cristiano Ronaldo
 Best Vision : L. Messi
 Best Penalties : M. Balotelli
 Best Composure : L. Messi
 Best Marking : A. Barzagli
 Best StandingTackle : G. Chiellini
 Best SlidingTackle : Sergio Ramos
 Best GK Diving : De Gea
 Best GK Handling : J. Oblak
 Best GK Kicking : M. Neuer
 Best GK Positioning : G. Buffon
 Best GK Reflexes : De Gea

Figure 7: Best players according to the different skillsets

Generally, as a player gets older, his skills start to decrement or he is not able to perform as good as he used to be during his 20s. The player eventually loses his value and also his place in the team. The team's performance degrades too if there are too many under-performing players in the squad. If a team has too many players which belong to the higher age group, eventually the team starts requiring too many replacements, too many injury substitutes etc.

Eventually the team chemistry starts to suffer as well. Therefore, it was necessary to explore this correlation from the dataset as well:

Figure 8: Player potential vs Age

From Figure 8, it can be observed that most of the higher potential is cumulated around age 18-28. This is the age range during which most of the players are at the peak of their potential. The dataset portrays a similar story. Similarly, there is another correlation that can be considered as an example that the skillset of a player decreases as he gets older. The trait that

gets hampered the most due to aging: the sprint speed can be visualized against the age of players. This correlation is depicted in Figure 9.

Figure 9: Decreasing sprint speed when players age

As expected, the sprint speed of players go down as they age. The decrease in various skill levels for players of a team also impacts the overall performance of the team. Therefore, a squad full of number of aging players or a squad having a lot of older players tend to win less matches. Therefore, providing us with one of the important observation, which may seem obvious, but can

be used extensively while making predictions in the sports domain. It can be concluded after the exploratory analysis, that the dataset present interesting correlations and observations. These observations are easy to understand and reflect the real world observations making it a reliable dataset.

CHAPTER 6

APPROACHES AND IMPLEMENTATION

To achieve the objective of making sports predictions based on previous data, various statistical and machine learning models are constructed. These models are chosen based on their similarity to the applications they are generally used in, with the problems they are used to solve. Based on the literature survey performed, it is necessary to include not just machine learning but also statistical models. Statistical models, in all their simplicity, can prove helpful in making better predictions.

The objective to predict the win percentage of a team makes it a regression problem. Basically, the aim of the experiments is to make predictions over the soccer dataset for how many matches a team will win out of 100. Therefore, the models with better regression capabilities are chosen for the experiments. Logistic Regression, Decision Tree Regressor, k-Nearest Neighbor Regressor, Support Vector Regressor and Multinomial Naïve Bayes are chosen for experiments along with an Artificial Neural Network model at the end. There were two phases to the experiments performed: first by without using various optimization techniques and then extensively using various optimization techniques.

The following sub-sections give a brief overview of all the ML and statistical models that are being used for the experiments in this research.

6.1 Logistic Regression

The term, logistic regression, was primarily popularized by Joseph Berkson, which, in the simplest of words, is a statistical model consisting of a logistic function, which basically is responsible to model an event or a class of data. This event or class of data may vary in a number of ways but most commonly they are binary in nature. Meaning, they can only attain one of two possible values at a time. This can be seen and termed as a simple binary classification. The dependent variable or variant has two categories in a binary logistic regression model. Multinomial logistic regression is used in case of outputs with greater than two values. In case if the multiple groups are ordered then ordinal logistic regression is to be designed.

Figure 10: Logistic Regression [33]

Generally speaking, linear regression can be seen as a simpler version of logistic regression. There are a number of reasons why linear regression is not ideal in most of the scenarios where there is a need to predict the value of a variable (binary variable in most cases). The biggest problem of all is that the acceptable range for predictions of probability is between

0 and 1 and linear regression, many a times, predicts values outside this acceptable range. This can be easily inferred from the figure above. In case of logistic regression, as the curve is created using log value (natural log) of the variable's probability, the value remains between 0 and 1.

These types of models, do not perform analytical categorization since they are not classifiers. They simply model the likelihood of a particular output provided a (or in regards to) an input. Although these types of models can be used to build models that can perform classification, or in other words, may serve as building blocks to build a classifier. One of the ways this can be done is by defining a threshold and then identifying inputs with odds greater than the threshold as one category, and below the threshold as the other. This would be in a case of a simple binary classifier.

For the initial phase of the experiments, logistic regression is used to make predictions over the odds provided in the dataset. The raw odds and the differences among the odds for winning, losing and drawing a match are given as input to the logistic model. Further experiments have utilized logistic regression in different ways. There have been optimizations which included feature engineering, parameter tuning and dimensionality reduction. These optimizations result in different subset of data with more or fewer attributes included at a time. Other selected features like 'Overall', 'Age', 'Finishing', 'Dribbling', etc. which are the predictors, represented by 'X' throughout the equations and experiments.

This was later split into X_train and X_test sets. The experiments therefore result in different error rate and different accuracy. The algorithm then tries to create a curve to fit the data which is contained within the values 0 and 1 (number of wins). The 'Y' is the target variable or the final outcome which gives a value between 0 and 100 to denote the win % or the number of matches won out of 100. The efficiency of the model is then derived from a custom method that returns the error rate and accuracy of the model. The final predicted values of Y, look as shown in figure 11.

```
print(LRpredictions)
[68 64 66 ... 55 65 68]
```

Figure 11: Logistic Regression Prediction

6.2 Decision Tree

In terms of general data analysis, the term decision tree was derived as a placeholder for a decision support tool which, similar to a tree in real life, uses tree like substructures to visually as well as explicitly represent decision and their possible outcomes. In its simplest of form, a decision tree can be thought of as a flowchart type of structure. Decision trees have become vital in the machine learning domain because of their simplicity, as they do not require parameter setting or knowledge about the domain, as well as effectiveness in conditional control scenarios.

Figure 12: Example of a simple Decision Tree [34]

A decision tree generally employs a top-down greedy approach and is drawn upside down from the root node (the first node). As mentioned before, decision trees have number of

substructures. Along with the root node, there are internal nodes (aka decision nodes) which define a decision/condition or in other words, a test on the attribute. Then there are branches which can be seen as the result of this test or outcome of the decision. There should be two or more branch emerging from internal nodes, except for the leaf nodes, which represent the terminal node and do not split further. These leaf nodes provide a class label or basically, the final decision on the target variable.

Decision trees are able to deal with continuous as well as categorical data which is what makes them more efficient. J.R. Quinlan was the one who formulated the core algorithm for creating a decision tree with no backtracking. The algorithm is known as ID3 and is still used extensively in order to create decision trees by employing a greedy search based top-down approach. This algorithm can be used to create decision trees with standard deviation reduction, replacing information gain.

The experiments carried out in this research use decision trees in two phases. In the first phase, only the odds given by the bookies are taken as the input. The model then makes decision over further matches based on these odds. For the following experiments, a number of attributes related to the players in the team are taken into consideration. Attributes such as 'Overall', 'Potential', 'Age', 'Finishing', 'Dribbling', etc. represented by 'X' are taken. And this input set was then split into training and testing set. This results in helping the model make better decisions internally. Figure 13 portrays how the model was initialized.

```
from sklearn.tree import DecisionTreeRegressor
DTree = DecisionTreeRegressor()
DTree.fit(X_train, y_train)
DTreepredictions = DTree.predict(X_test)
```

Figure 13: Decision Tree Initialization

There are branches which can be seen as the result of this test or outcome of the decision. There are two or more branch emerging from internal nodes, except for the leaf nodes, which represent the terminal node and do not split further. These leaf nodes provide a class label or basically, the final decision on the target variable. This target variable is represented by Y, which gives the output as shown in figure 14. After further optimizations, the results given by the decision tree improve.

```
print(DTreepredictions)
[72. 55. 63. ... 51. 68. 65.]
```

Figure 14: Decision Tree Predictions

6.3 KNN

K-nearest neighbor or KNN, is a supervised ML algorithm which is fairly popular due to its easy-to-use simplicity. It's a non-parametric algorithm which can be used for regression as well classification. The k denotes the 'k' number of close training points in the sample space. The output of KNN depends upon the purpose for which it is being used: a class membership or categorization into one of the categories if it's being used for classifications, and a value (numerical in most cases) in case it's used for regression.

Figure 15: KNN Example

A basic KNN implementation works by assuming that related entities are in proximity closer to each other. It works by finding the distance between the target point in question and all the other points in the data. In regression, it averages the distance between this target point and the 'k' nearest points to this selected point. There are a number of distance functions that might be used to calculate this distance.

Distance functions

Euclidean	$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$
Manhattan	$\sum_{i=1}^k x_i - y_i $
Minkowski	$\left(\sum_{i=1}^k (x_i - y_i)^q \right)^{1/q}$

Figure 16: Distance Functions [33]

In case the data consists of continuous variables, the above three distances can be used, but in case of categorical data, Hamming distance should be used which is given as follows:

Hamming Distance

$$D_H = \sum_{i=1}^k |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

Figure 17: Hamming Distance [33]

When it comes to KNN classification, the class for the target variable is decided by the majority of the neighboring class. That is, the variable is assigned the class which is the most frequent within the k nearest neighbors. As expected, k is an integer, positive. Typically, the value of k is small but for a large diverse dataset, a smaller k may not work too well. For e.g. a value of

1 for k just assigns the class of the point next to the target variable to it. Generally, a larger value works fine for k as it reduces noise but depending upon the data, there might always be exceptions.

For the experiments in this research, KNeighborsRegressor from sklearn was used. As KNN works based on feature similarity, the models basically tried to find the distance between the target valuable and the given data point. In the initial experiments, these points were the odds that help predict the win % for a team. There were multiple attributes included in the experiment that followed. The input includes the odds for the initial experiments and other selected features like 'Overall', 'Age', 'Finishing', 'Dribbling', etc. which are the predictors, represented by 'X'. Figure 18 shows how KNN was initialized.

```
from sklearn.neighbors import KNeighborsRegressor
KNN = KNeighborsRegressor(n_neighbors=20)
KNN.fit(X_train, y_train)
KNNpredictions = KNN.predict(X_test)
```

Figure 18: Initializing KNN

The value of k is the most determining factor in KNN and finding an optimal k is an important step. For example, in figure 15, if k=3, the unknown object will be classified as red triangle. But if k=5, the object will be classified as blue square. Grid Search CV is very useful for multiple iterations and after multiple attempts for trying out different values of k, k is determined to 20 for better results. The target variable Y is then compared with the nearest 20 neighbors and based on the maximum likelihood, its predictions are made. The final target values predicted is shown in figure 19. With k = 20, and after multiple optimizations, and hyperparameter tuning, accuracy of 97% was reached.

```
print(KNNpredictions)
[70.85 59.8  65.   ... 51.95 66.65 66.2 ]
```

Figure 19: KNN Predictions

6.4 SVM

Support-vector machines (or support-vector networks) are supervised learning algorithms which can be used in combination with, or as building blocks to, other learning algorithms for classification and regression purposes. At first, an N-dimensional plane is created, where n is the number of attributes in our data and plot each data value as a point in this plane. In a more general sense, a SVM creates a set of hyperplanes or just one hyperplane in this high-dimensional space. This can be used for classic use-cases of regression or classification, or to perform some other operations like outlier-detection. In the case of SVM classifier, the classes need to be as distant as possible from the hyperplane.

Figure 20: A simple SVM [35]

As can be inferred, a better separation is achieved by a hyperplane which is most distant to the nearest training data variable from any class. And classification, is then performed by choosing the hyperplane which is the best in categorizing the two classes. The SVM regressor or Support Vector Regression (SVR) also use similar principles but has a few differences. First of which is that it uses a margin of error/tolerance or epsilon as the output has to be a real number and which makes it very difficult to predict. Basically, in regression, a model that can predict the future quantity of the target variable is required. Therefore, the observation or target variable as near to the hyperplane as possible is needed.

Figure 21: Support Vector Regressor [36]

The experiments for the Support Vector Regressor model were carried out in similar fashion as the other experiments. The initial phase had the model trained over the odds given by the bookies. For further experiments, the input includes the other selected features like 'Overall', 'Age', 'Finishing', 'Dribbling', etc. which are the predictors, represented by 'X'. For SVRs, the important thing is to set an appropriate margin of tolerance. This is referred to as the epsilon value. Figure 21 portrays the different curves that can be plotted with different values of epsilon.

The main motive of the experiments was always to get the error rate as low as possible, and to select the hyperplane which maximizes the margin. From the multiple experiments attempted, an epsilon value of 0.2 was selected which gave the highest accuracy of the sorted dataset. And after a few optimizations, the error reduced to 6.86 (RMSE). After experiments epsilon was decided to be 0.2 and C was 1.0. The target variable, 'Y' gives the win percentage as portrayed in figure 22.

```
print(SVMpredictions)
[66.23152299 66.23152298 66.23152299 ... 66.23152171 66.23153356
 66.23152299]
```

Figure 22: SVM Predictions

6.5 Artificial Neural Networks

Artificial Neural Networks (ANN) simulate the biological network of neurons which makes up the human brain. They are envisioned to replicate the human learning process. There are specific rules or tasks that the system follows or performs in order to learn from the data supplied. ANN consist of one or multiple input layers, along with multiple hidden and output layers. Deep learning neural networks and back-propagation has made ANN more popular in the last few decades.

Figure 23: Artificial Neural Network

There are many types of neural networks: feedforward neural network, recurrent neural network, convolutional neural network, Hopfield network etc. For the purpose of this project, a feedforward neural network is used. In these types of neural networks, the information flows from input layer to output layer, in only one direction. In order to implement the ANN model, the following libraries were imported:

```
from keras.layers import Dense, Activation
from keras.models import Sequential
```

Figure 24: Neural Network Libraries

The odds of a team winning or losing or drawing a match are used for the initial experiments which are contained within the input variable X. These values are then scaled using MinMaxScalar and StandardScalar methods. This was done in order to scale the dimensions of the data uniform across the different sets of data. Once the values were scaled and had uniform dimensions, an ANN model is created using Sequential() constructor.

An input layer is added using:

```
model.add(Dense(32, input_dim=54, kernel_initializer='normal',  
activation='softmax'))
```

The hyperparameters for this add method were iteratively tested and then finalized. For example, the activation function was tried among 'tan', 'relu', 'swish', and 'sigmoid'. It was observed that these functions did not fit the data very well. They had no effect on the data and there was no transformation over it. When the 'softmax' activation function was used, the data was being manipulated and transformed successfully. Therefore, this activation function was used for implementation.

With respect to other hyperparameters, 'Dense(32, input_dim=54' basically indicates creating a dense layer with 32 neurons and 54 input dimensions. For example, consider the case of using

```
>> model.add(Dense(12, input_dim=8, init='uniform',  
activation='relu'))
```

Here, it means that the layer will have 8 input parameters, with 12 neurons in the FIRST hidden layer as shown in figure 25.

Figure 25: Neural Network Example

For further layers, again `model.add()` method was used along with softmax activation function. This activation method basically scale downs the output between 0 and 1. It also divides each output in such a way that their sum becomes equal to 1. So, from the previous layers, and in the input layer, it basically converts the input to a categorical probability distribution which is what is required in case of regression.

And just as in the case of input layer, 'Dense(32)' has been used again which basically creates another layer with 32 neurons. For the second last layer, 'linear' activation function has been used as the probability distribution was required as the input to the last layer. For the Output layer, `model.add(Dense(units = 1))` has been used. This creates just one neuron and upon compilation, given the output as transformed by the previous layers. The model is then compiled and the following parameters are used: loss, optimizer, and metrics.

An optimizer is one of the two arguments required for compiling a Keras model. 'adam' is used as the optimizer which is a stochastic gradient descent method. If it was a classification problem, 'accuracy' would have been used as a metric in order to get the accuracy of ANN straight away. But because this was a regression problem, the only options for the metric argument were mean squared error, mean absolute error, root mean squared error etc. In order to calculate accuracy, a custom method was implemented which is discussed in further chapters.

CHAPTER 7

EXPERIMENTS & RESULTS

For the initial experiments with the dataset that was gathered, the odds given by bookies for the game was taken under consideration for game predictions. Basically, odds are the amount a person is supposed to win if they bet \$1 correctly on that result. Suppose the odds for an away win is 5, in that case, the person will win \$5 for each \$1 that was bet on an away win. Therefore, for any result that are predictable or more likely to occur, the odds are set lower by the bookies. Looking at the dataset, the result of a match can be predicted by looking at the odds. The odds were the only thing that were considered for the initial experiments.

In order to achieve this, there were a number of custom methods implemented which would:

- return the result predicted by the bookies based on odds for each result which are higher.
- return the actual result of the match.
- check the predictions of the game accurately based solely on the odds.

```
In [8]: check_pred(game_info)

C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:5: FutureWarning: `item` has
been deprecated and will be removed in a future version
"""
C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:6: FutureWarning: `item` has
been deprecated and will be removed in a future version

C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:7: FutureWarning: `item` has
been deprecated and will be removed in a future version
import sys
C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:8: FutureWarning: `item` has
been deprecated and will be removed in a future version

C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-packages\ipykernel_launcher.py:9: FutureWarning: `item` has
been deprecated and will be removed in a future version
if __name__ == '__main__':

Out[8]: '0.53% correct'
```

Figure 26: Accuracy solely based on odds

Figure 26 portrays the 53% accuracy obtained in the first experiment. Basically, the figure above shows that if the results of the game were based on just predictions from odds, probably the predictions would be accurate about half of the time. Going ahead, the odds as well as the differences in odds are used, to accurately make predictions over the match results.

```
x_var["diff_h_d"]=abs(x_var["odd_h"]-x_var["odd_d"])
x_var["diff_d_a"]=abs(x_var["odd_d"]-x_var["odd_a"])
x_var["diff_h_a"]=abs(x_var["odd_h"]-x_var["odd_a"])
x_var=x_var.drop(["fthg","ftag"],axis=1)
x_var.tail()
```

C:\Users\gour\AppData\Local\Continuum\anaconda3\lib\site-pack
been deprecated and will be removed in a future version
after removing the cwd from sys.path.

	odd_h	odd_d	odd_a	diff_h_d	diff_d_a	diff_h_a
10107	1.97	4.35	8.0	2.38	3.65	6.03
10108	1.19	8.50	20.0	7.31	11.50	18.81
10109	1.32	6.24	11.5	4.92	5.26	10.18
10110	1.74	4.07	5.9	2.33	1.83	4.16
10111	1.79	3.96	5.4	2.17	1.44	3.61

Figure 27: Data Used For Initial Experiments

This dataset was represented by 'X' and was taken as input. This data was then split in in training, testing and validation in 70:20:10 proportion and KFold with n_split to be 10 was used. Five ML models: Logistic Regression, Decision Tree Regressor, k-Nearest Neighbor Regressor, Support Vector Regressor and Multinomial Naïve Bayes are then trained on this data. The accuracy of these models was then calculated using a custom method. Table 16 shows the results after experiments with these 5 models.

<u>Sr. No.</u>	<u>Model</u>	<u>Accuracy</u>
1.	Logistic Regression	52.76%
2.	Decision Tree Regression	97.49%
3.	KNN Regression	56.11%
4.	SVM	53.17%
5.	Multinomial Naïve Bayes	49.09%

Table 16: Initial Comparison of ML Models

Figure 28: Comparison of Results from Initial Experiments

Upon observation, it is evident that the decision tree model gives good results with 97.49% accuracy. However, other models do not perform as efficiently. This left a big margin for

corrections and various techniques are then applied to improve the efficiency of these models. To improve the performance of these models, a number of methodologies are used which also require re-processing and manipulating the dataset. Multiple optimization techniques are then implemented which include feature engineering, hyperparameter tuning, dimensionality reduction and principal component analysis.

7.1 Feature Engineering

To improve the performance of these models, a number of methodologies are used which also require re-processing and manipulating the dataset. Data gathered from the real world can be extremely chaotic and unstructured. Therefore, feature engineering is needed to transform the data into something more meaningful. And this also helps generate new features from raw data. Features can be weakly relevant or strongly relevant. Weakly features are those which are carrying information other features might fully or partially might carry.

Strong features are those which convey information which other features do not. It's considered a good practice to have extra features than to miss out on a few important ones. Therefore, feature selection techniques can be used in order to reduce or avoid overfitting. The process generally includes not only adding features but also removing few unknown or unnecessary ones. For example, upon initial examination, there were a number of unimportant columns observed in the data, like link to the Club image, Photo, Flag etc. which wouldn't have been useful for my experiments. Therefore, these columns had to be removed from the dataset. Then the total amount of data which is actually present in each column was visualized.

Figure 29: Analyzing amount of data in each column

As evident from the Figure 29, few of the columns didn't held a lot of data. Hence, these columns were removed completely. This was mainly due to missing data in these columns. Some of the attributes do not have data which is gathered and kept historically. Or it might be possible that the attribute is not something which applies or stays true for every player or every match. For example, in the figure above, the lowest count is of the attribute "loaned from", which is expected as only a handful of players would have been loaned from some other club.

Further, in order to simplify the dataset a little bit, the number of playing positions were reduced from a more specific position title to a more generic one. For example, different midfield

positions like LM, CM, CAM, RM etc. were converted to just be labeled as just one position, MID. Similarly, various attacking and defensive positions were also given a single label. Also, the values in the player's salary column were observed to be uneven. Few of the values had 'M' denoting millions in them, and few of the rows did not have an 'M'. These were made uniform in order to make this attribute more useful. As a player's wage signifies a player's quality and presence of more expensive players can be a good tipping point for teams to win matches.

7.2 Dimensionality Reduction

There might be too many variables or features on the basis of which the final regression or classification would be carried on. The higher the number of features, the more difficult to visualize and train the ML models with the dataset. Many of the features might be correlated and hence, their influence would be redundant. This is where the dimensionality reduction algorithms come into the picture. Dimensionality reduction is basically the process of reducing the number of indiscriminate features under consideration and obtaining just a smaller set of principle features.

As per further observations, a number of features had a higher degree of correlation. For example, players with better dribbling skills had better overall scores. And players with better finishing abilities had a better overall scores. Also, teams with more players with higher overall had better chances of winning, which eventually meant better wages for higher overall. The technique called Principal Component Analysis (PCA) was therefore used. It is a form of feature extraction technique for dimensionality reduction.

7.3 Principal Component Analysis

This method was originally developed by Karl Pearson. The goal behind this technique is to pick or extract a set of features/variables from a set of random or known variables. It basically helps in bringing out or portraying the stronger patterns in the data by emphasizing variation. It creates a mapping (linear in nature) of the data to a lower dimension space on the provision that the variance of the data while in lower dimension is maximum. To explain this diagrammatically, by using the Seaborn library, the following graph can be visualized for the used dataset:

Figure 30: Dimensionality Reduction

Figure 30 plots the various number of attributes or components which are present in the dataset against the cumulative variance of the data. The variance is given between 0 and 1. Basically, in PCA, the best fitting line in the n-dimensional space is the one which minimizes the mean square distance of the line to a point. And the next better fitting line can be similarly chosen from the first line's perpendicular directions. This process is repeated till the point a set of orthogonal elements which portray data from dimensions, which are not co-related, is achieved. Upon observing the variance distribution from the above figure, 6 principal components can be identifies. Therefore, for further experiments, `n_components = 6` is used as PCA optimization.

7.4 Hyperparameter Tuning

Hyperparameter refer to the parameter which is not directly learnt within the estimator but is used to rather control the learning process. And hyperparameter tuning is the process of choosing the most optimal set of for these hyperparameters [12]. The hyperparameters are passed as an argument to the constructor of the estimator [13]. GridSearchCV and RandomizedSearchCV are two methods that can be used for sampling search options. Grid Search is used as shown in figure 31.

```
parameters = [{'kernel': ['rbf'],
                    #'epsilon': [0.1, 0.2, 0.5],
                    #'gamma': [1e-4, 1e-3, 0.01, 0.1, 0.2, 0.5],
                    'C': [1, 10, 100, 1000]
                },
               #{'kernel': ['linear'], 'C': [1, 10, 100, 1000]}
               ]

regressor_pca_cv = GridSearchCV(log_reg, param_grid = parameters, return_train_score=True)
regressor_pca_cv.fit(x_train_pca, y_train)

#LR
log_reg=LogisticRegression()
log_reg.fit(x_train_pca,y_train)

LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                    intercept_scaling=1, l1_ratio=None, max_iter=100,
                    multi_class='warn', n_jobs=None, penalty='l2',
                    random_state=None, solver='warn', tol=0.0001, verbose=0,
                    warm_start=False)
```

Figure 31: Grid Search CV

The available or optional set of parameter are passed as a tuple to the estimator and Grid Search or Random Search method finds the set of hyperparameters which helps in reducing the error of the model. For example, in the case of SVM, the value of epsilon is estimated by passing a set of probable optimal epsilon (0.1, 0.2, 0.5 etc.). And in the case of KNN, the value of K are estimated by specifying the candidate values for an optimal K. Upon experiments, optimal value of K was determined to be 20 and optimal value of epsilon was determined to be 0.2.

After implementing and testing with all the optimizations valid for the dataset and models, further experiments were performed. The dataset, after feature engineering and PCA, contain new attributes. The models have better and optimal hyperparameters. The input set, X_train and X_test were split based on data as shown in figure 32 and the predicted target values are discussed in previous chapters.

Age	Overall	Potential	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing
31	94	94	5.0	4.0	4.0	84.0	95.0
33	94	94	5.0	4.0	5.0	84.0	94.0
26	92	93	5.0	5.0	5.0	79.0	87.0
27	91	93	4.0	3.0	1.0	17.0	13.0
27	91	92	4.0	5.0	4.0	93.0	82.0

Figure 32: Sample Input with New Attributes

After the models are trained and predictions made over the testing data, the error rates of the models were found and based on the error rate, the accuracies were calculated. This was done with the help of a custom method which had errors formulated based on official definitions of

these errors. Consider the case where the predicted values from a model are obtained as in figure 33.

```
print(pred)
[[0.4373208 ]
 [0.32177258]
 [0.15125634]
```

Figure 33: Predicted Value

The custom method contained the following important calculations:

```
>> errors = abs(pred - y_test)          .. (1)
>> round(np.mean(errors), 2)            .. (2)
>> mape = 100 * (errors/y_test)         .. (3)
>> acc = 100 - np.mean(mape)           .. (4)
```

(1) gives the **Mean Absolute Error**. It is the absolute difference between the actual and the predicted values of the target variable. The average of error value from every sample is taken from the experimented data and then the output is provided by using (2). Then this error is converted this to Mean Absolute Percentage error using (3). And finally, (4) is used to calculate the accuracy.

Mean Squared Error is imported from sklearn.metrics and is used as:

```
>> mse = mean_squared_error(y_test, pred)
```

Mean Squared Error is very popular and is generally used to calculate the performance of regression models. The average of the square of the difference between the predicted and actual values is taken in order to obtain mse. And np.sqrt(mse) gives the **Root Mean Squared Error**. It basically reflects the standard deviations of the errors made while predictions are made over the dataset. Both Mean Absolute Error and Root Mean Squared Error are used to calculate the efficiency of the models.

This was done for all the models and following results were obtained in the end:

Sr. No.	Model	Accuracy	RMSE	MAE
1.	Logistic Regression	95.48%	3.83	2.94
2.	Decision Tree	98.07%	1.92	1.22
3.	KNN	97.2%	2.42	1.86
4.	SVM	91.63%	6.88	5.45
5.	Multinomial Naïve Bayes	82.77%	14.43	11.26
6.	ANN	95.51%	0.017	0.01

Table 17: Final Results of ML Models

Table 17 gives a comparative result of the error and accuracy observed for the models after implementation. The accuracy of each model has improved significantly from the previous experiments. Logistic Regression gives an accuracy of 95.48% and has an optimal error rate of 3.83 RMSE and 2.94 MAE. The performance from the initial experiment has improved but it is still not the best comparatively. The Decision Tree model still gives the highest accuracy of 98.07% for the dataset used. With an RMSE of 1.92 and MAE of 1.22, Decision Tree does stand out as the most efficient model from the other models implemented.

Also, it can be observed that KNN shows significant improvement as well. With an accuracy of 97.2% and 2.42 RMSE and 1.86 MAE, KNN is the second best model after Decision Trees. The optimal value of k which was found out to be 20 gives better results than the other values experimented with. As per previous studies, SVR has shown to perform better for

continuous data. However, for the experiments in this research, SVR does not perform as well as KNN. The epsilon value of 0.2 was decided after a few experiments and showed lowest error rate out of other values chosen.

As can be expected from a statistical model, Multinomial Naïve Bayes does not perform as well as the other ML model chosen. This was a common observation in few of the literature studies done prior during the research. Multinomial Naïve Bayes relies on probabilities and makes assumptions that the variables are mutually independent of each other. This assumption is a major drawback of using Naïve Bayes in such scenarios. On the other hand, Artificial Neural Network model is efficient and produces lowest error rate among all the models implemented.

The RMSE for ANN is 0.017 and MAE was 0.01 with an accuracy of 95.51%. Although multiple executions of the ANN model produces significantly different results with accuracy varying by 1.5-2% and error rates changing by ± 2 . The number of epochs were experimented with from initially being 150 to finally choosing 500 epochs. With every epoch, the error rate dropped significantly for the first hundred epoch. The decrease was then observed to be less and the error rate remained same for the last 400-500 epochs.

The following plot shows the root mean squared error for the ANN model implemented:

Figure 34: Mean Squared Error for ANN

In Figure 34, the blue line signifies the training error measured by mean squared error and the orange line signifies the validation error. The x-axis gives a measure of number of epochs and the y-axis provides the error during that epoch for the training and validation data. Though not perfect, the artificial neural network still gives a better result with a lower error rates. The following figure gives a comparison between the accuracy of all the models that have been implemented:

Figure 35: Comparison of ML models implemented

Figure 35 makes it evident that it is difficult to point to a single algorithm that exceptionally outperforms all the other models. From all the results above, it can be concluded that machine learning models do perform well in comparison to statistical models. Especially, when it comes to regression based analysis, there are a number of machine learning algorithms which are extensively used. Also, these models are simple and easy-to-implement too. There are a number of optimization techniques that can be used in conjunction of each other. As observed from the differences in results of the experiments, it is quite evident that these optimization techniques are important to improve the predictions of predictive models.

CHAPTER 8

CONCLUSION & FUTURE WORKS

Predictive models constructed using machine learning are very powerful in predicting future events. There are different industries with the potential to leverage predictive modelling which are summarized in this research. A comparative summary of how the predictive models perform in different scenarios is also discussed. In this research, machine learning and statistics based predictive models are implemented to predict future event in the sports domain.

Steps to gather the dataset and prepare it for regression based analytics are documented in detail. Further, exploratory analysis is done to provide better insights into the data. Initial experiments with predictive models showed high error rates and poor accuracy. Therefore, various optimizations are discussed that can be done over the models. The re-processing of the dataset along with feature engineering helped include important attributes to the dataset. This is well documented as it was a necessary step to improve the performance of the models.

To summarize, this research provides an application of using predictive models for sports analytics. And also provides key techniques to optimize these models which can be used for other domains as well in order to improve their performance. As a next step, this research can be extended by using an ensemble of the models discussed. The efficiency of the models can be further optimized as a result. Also, other applications in the domain of sports analytics can be explored going ahead. This research included betting and win predictions and as for next steps, many more applications can be explored. The implementation and optimization details can then be referred from this research.

REFERENCES

- [1] S. Sarojini Devi and Y. Radhika, "A Survey on Machine Learning and Statistical Techniques in Bankruptcy Prediction," *International Journal of Machine Learning and Computing*, Vol. 8, No. 2, 2018.
- [2] G. Kumar and S. Roy, "Development of hybrid boosting technique for bankruptcy prediction," in *Proc. International Conference on Information Technology*, 2016, pp. 248-253.
- [3] L. Becchetti, and J. Sierra, "Bankruptcy risk and productive efficiency in manufacturing firms," *Journal of Banking & Finance*, vol. 27, no. 11, pp. 2099-2120, November 2003.
- [4] Ranjith Ramesh, "Predictive Analytics for Banking User Data using AWS Machine Learning Cloud Service," *Second International Conference On Computing and Communications Technologies (ICCCT'17)*, 2017.
- [5] KamalaKanta Mishra, Sachin Kumar Manjhi, "Failure Prediction Model for Predictive Maintenance," *IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, 2018.
- [6] Vimala Mathew, Dr. Anu Mary Chacko, Dr. A. Udhayakumar, "Prediction of suitable human resource for replacement in skilled job positions using Supervised Machine Learning," *Eighth International Symposium on Embedded Computing and System Design (ISED)*, 2018.
- [7] T.T., Weeramanthrie; C. N., Thilakumara; K.N.A.C., Wijesiri; N. I., Fernando; S., Thelijjagoda; A., Gamage., "A Web-Based Employee Turnover Analysis Tool for Effective

- Human Resource Management in Large-Scale Organizations," in National Information Technology Conference, Colombo, 2017.
- [8] Mohammad Asif Nawaz, Mirsad Hadzikadic, "Changing the Dynamics of Training by Predictive Modeling," 15th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT (HONET-ICT), 2018.
- [9] Driss BARI, "Visibility Prediction based on kilometric NWP Model Outputs using Machine-learning Regression," IEEE 14th International Conference on e-Science, 2018.
- [10] Bergot, T., Carrer, D., Noilhan, J., and Bougeault, P. "Improved site-specific numerical prediction of fog and low clouds: A feasibility study". *Weather and Forecasting*, 20(4), 627-646. 2005.
- [11] Ma. Madecheen S. Pangaliman, Febus Reidj G. Cruz, Timothy M. Amado, "Machine Learning Predictive Models for Improved Acoustic Disdrometer," IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2018.
- [12] Lida Bai, Lan Chen, "Machine-Learning-Based Early-Stage Timing Prediction in SoC Physical Design," 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), 2018.
- [13] Ameeth Kanawaday, Aditya Sane, "Machine Learning for Predictive Maintenance of Industrial Machines using IoT Sensor Data," 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), 2017.

- [14] E. H. M. Pena, M. V. O. de Assis and M. L. Proença, "Anomaly Detection Using Forecasting Methods ARIMA and HWDS," 2013 32nd International Conference of the Chilean Computer Science Society (SCCC), Temuco, 2013, pp. 63-66.
- [15] Yi WAN, Chengwen WU, "Fitting and Prediction for Crack Propagation Rate Based on Machine Learning Optimal Algorithm," International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government, 2019.
- [16] Li Jia, Chao Han, Min-sen Chiu, "An integrated Model Predictive Control Strategy for Batch Processes," 28th Chinese Control and Decision Conference (CCDC) 2016.
- [17] Zhixin Li, Lei Liu, Degang Kong, "Virtual Machine Failure Prediction Method Based on AdaBoost-Hidden Markov Model," International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2019.
- [18] Manar D. Samad, Megan A. Witherow, "A Machine Learning Pipeline to Optimally Utilize Limited Samples in Predictive Modeling," 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018.
- [19] Yeboon Yun, Hirotaka Nakayama, Min Yoon, "Multi-Objective Model Predictive Control," Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), 2018.
- [20] Charles Feng, Shuning Wu, Ningwei Liu, "A user-centric machine learning framework for cyber security operations center," IEEE International Conference on Intelligence and Security Informatics (ISI), 2017.

- [21] Dong-Jun Yu, Jun Hu, Qian-Mu Li, Zhen-Min Tang, Jing-Yu Yang, Hong-Bin Shen, "Constructing Query-Driven Dynamic Machine Learning Model With Application to Protein-Ligand Binding Sites Prediction," IEEE Transactions on NanoBioscience, 2015.
- [22] Bogdan Czejdo, Sambit Bhattacharya, Catherine Spooner, "Improvement of Protein Model Scoring Using Grouping and Interpreter for Machine Learning, IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 2019.
- [23] Timothy M. Amado, Jennifer C. Dela Cruz, "Development of Machine Learning-based Predictive Models for Air Quality Monitoring and Characterization," Proceedings of TENCON 2018 - 2018 IEEE Region 10 Conference, 2018.
- [24] M. Fatih Akay, Ebru Çetin, İmdat Yarım, Özge Bozkurt, M. Mikail Özçiloğlu, "Development of novel maximal oxygen uptake prediction models for Turkish college students using machine learning and exercise data," 9th International Conference on Computational Intelligence and Communication Networks (CICN), 2017.
- [25] Pooja Kadam, Dr. Suhasini Vijayumar, "Prediction Model: CO2 emission using machine learning," 3rd International Conference for Convergence in Technology (I2CT), 2018.
- [26] Mirza Mansoor Baig ; Ning Hua ; Edmond Zhang ; Reece Robinson ; Delwyn Armstrong ; Robyn Whittak; Tom Robinson; Farhaan Mirza; and Ehsan Ullah, "Machine Learning-based Risk of Hospital Readmissions: Predicting Acute Readmissions within 30 Days of Discharge," 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2019.
- [27] Karun P Warrior, Shrenik M, Nimish Soni, "Short-Term Electrical Load Forecasting using Predictive Machine Learning Models,"IEEE Annual India Conference (INDICON), 2016.

- [28] Yuan Ren, Guang-Yi Cao, Xin-Jian Zhu, "Predictive control of proton exchange membrane fuel cell (PEMFC) based on support vector regression machine," International Conference on Machine Learning and Cybernetics, 2005.
- [29] Ping Sun, Guimu Guo, Rongjie Yu, "Traffic Crash Prediction Based on Incremental Learning Algorithm," IEEE 2nd International Conference on Big Data Analysis, 2017.
- [30] B. Yu, D. Z. Pan, T. Matsunawa, and X. Zeng, "Machine learning and pattern matching in physical design," Proc. of the IEEE/ACM Asian and South Pacific Design Automation Conference (ASPDAC), pp. 19-22, 2015.
- [31] Y. Sun, S. Todorovic, and S. Goodison "Local-learning-based feature selection for high-dimensional data analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 9, pp. 1610-1626, September. 2010.
- [32] D. S. Sisodia, S. Vishwakarma and A. Pujahari, "Evaluation of Machine Learning Models for Employee Churn Prediction," in Proceedings of the International Conference on Inventive Computing and Informatics, 2017.
- [33] <https://saedsayad.com/>, Dr. Saed Sayad, MSDS, Rutgers University, 2020 February 26.
- [34] <https://www.medium.com/>, A Medium Corporation, 2020 February 26.
- [35] <https://www.analyticsvidhya.com/>, Analytics Vidhya, 2020 March 12.
- [36] <https://www.wikipedia.org/>, A Wikimedia Foundation, Inc, 2020 April 6.