**PROJECT REPORT**

On

# “PETROL PRICE FORECASTING USING AUTO KERAS

### A Major Project Report Submitted in Partial Fulfillment of the Requirements Award of Degree of BACHELOR OF TECHNOLOGY

**In**

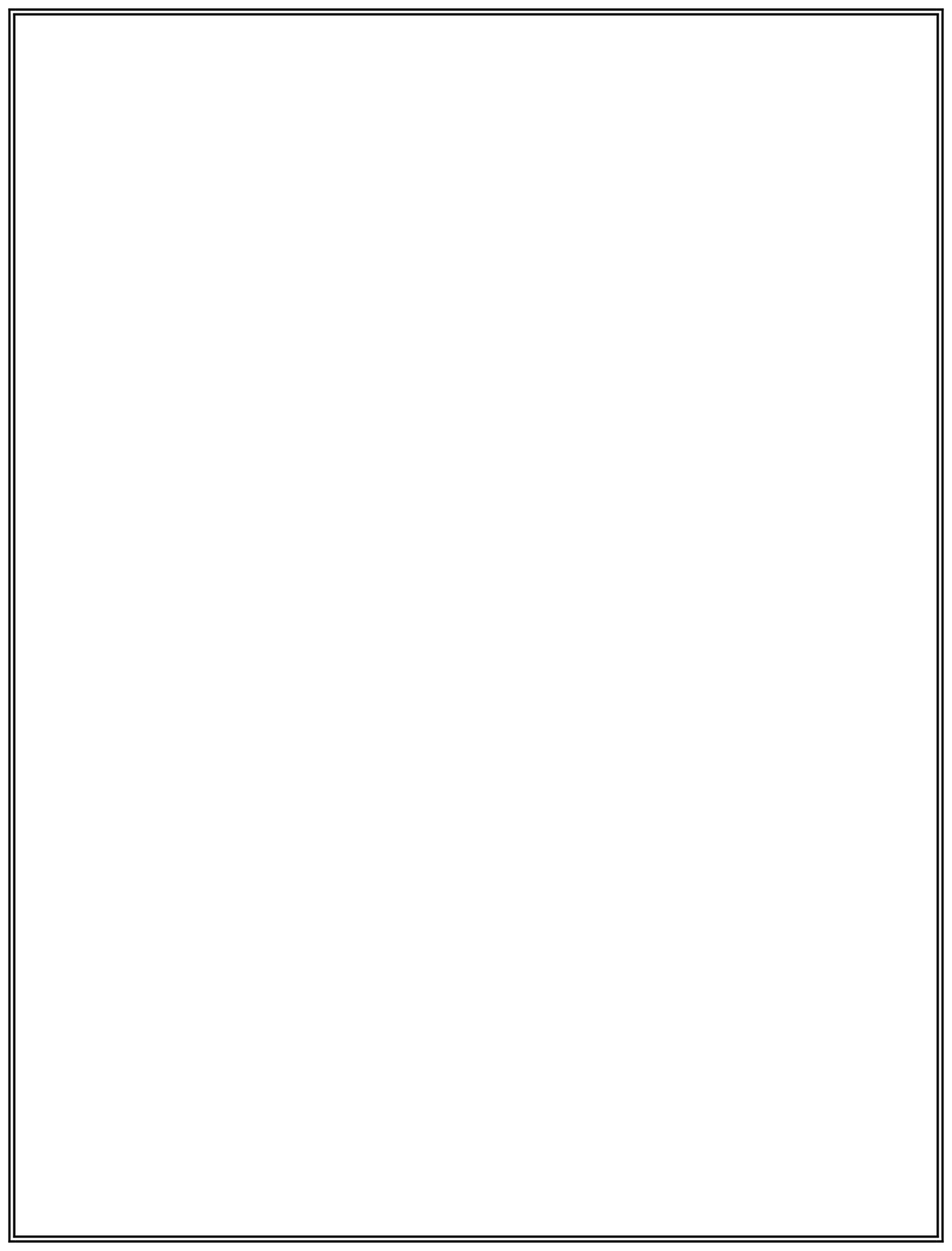
**COMPUTER SCIE NCE & ENGINEERING**

BY

**N.ASHMITHA**

(**21C61A0537**)

**UNDER THE ESTEEMED GUIDANCE OF Mr. M.VENKATESWARAO,** M. Tech



Assistant professor

**DEPARTMENT OF COMPUTERSCIENCE AND ENGINEERING**

**MOTHER TERESA INSTITUTE OF SCIENCE AND TECHNOLOGY**

AUTONOMOUS

Accredited By NAAC with ‘A+’ Grade

Approved by AICTE, Govt. of Telangana & SBTET, Affiliated to JNTUH, Hyderabad.

Recognition under section 2(f) & 12(B) OF THE UGC Act,1956 Sanketika Nagar, Sathupally-507303, Khammam (Dist), Telangana State.

**MOTHER TERESA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**AUTONOMOUS**

Accredited By NAAC with **‘A+’** Grade

Approved by AICTE, Govt. of Telangana & SBTET, Affiliated to JNTUH, Hyderabad.

Recognition under section 2(f) & 12(B) OF THE UGC Act,1956 Sanketika Nagar, Sathupally-507303, Khammam (Dist), Telangana State.

**CERTIFICATE**

This is to certify that the Project entitled **“PETROL PRICE FORECASTING USING AUTO KERAS”**is a bonafied work done by **N.ASHMITHA (21C61A0537)** . In partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING** from Jawaharlal Nehru Technological University Hyderabad During the Academic Year 2024- 2025.

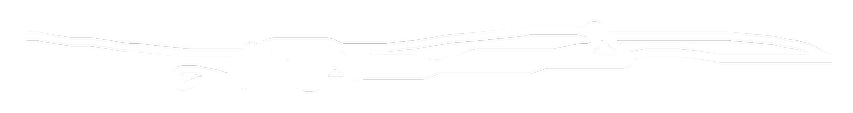
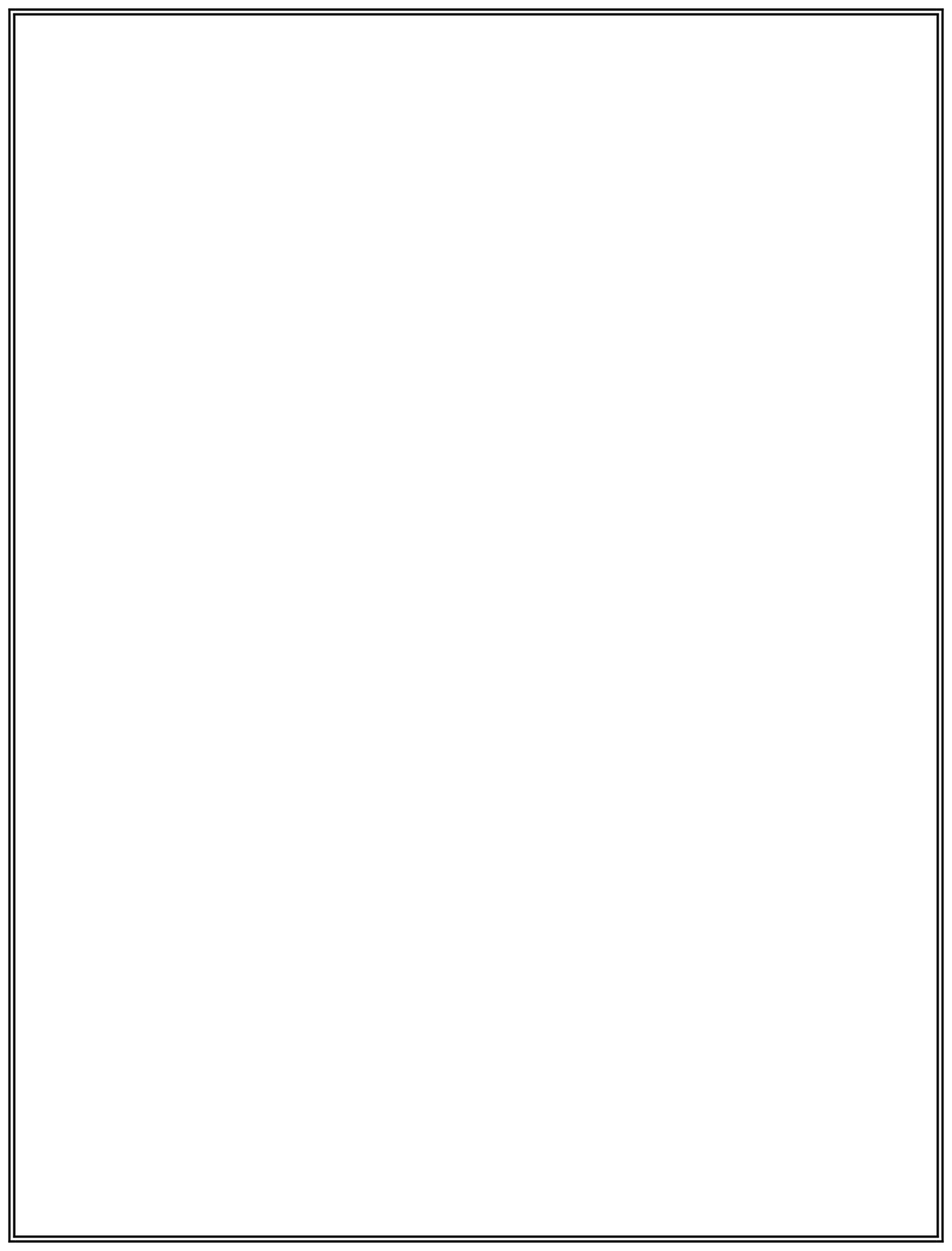
### Guide

**Mr. M.VENKATESHWARAO M. Tech.,**

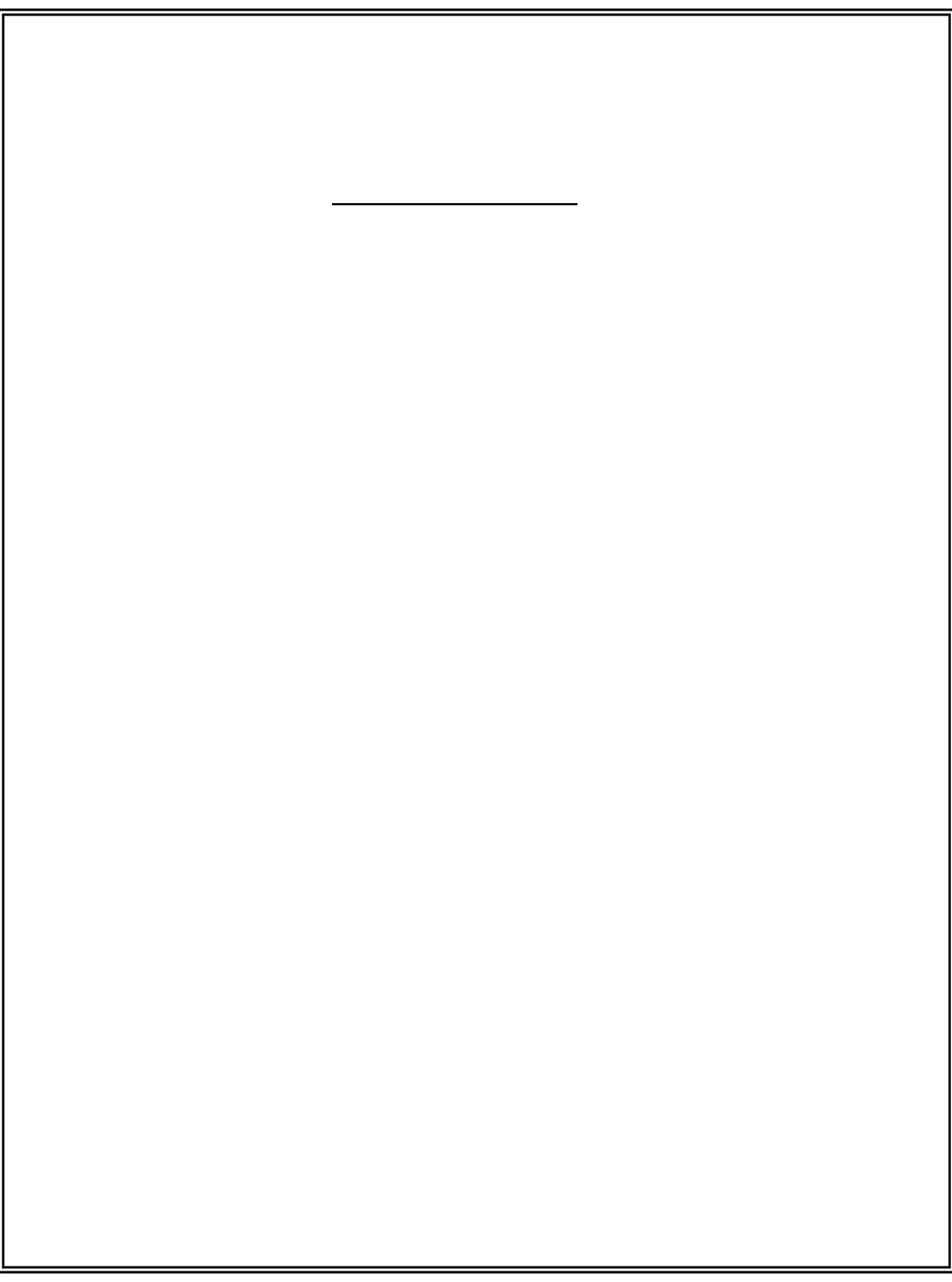
### Head of The Department

**Mr. P. MARESWARA RAO M. Tech., (Ph.D)**

### Internal Examiner Date:



**External Examiner Date:**

****

**ACKNOWLEDGEMENT**

#### I am grateful to numerous individuals who contributed to the preparation of our Project Report.

I wish to express our sincere and heartful gratitude to our Technical seminar guide **Mr. M.VENKATESWARAO M.Tech Assistant Professor**, **COMPUTER SCIENCE & ENGINEERING**, who encouraged us to take up a project in sync with global trends,with a programmatic approach and constant encouragement and cooperation during the project report.

I thank and deep sense of gratitude to **Mr.P.MARESWARA RAO M.Tech,(Ph.D).**

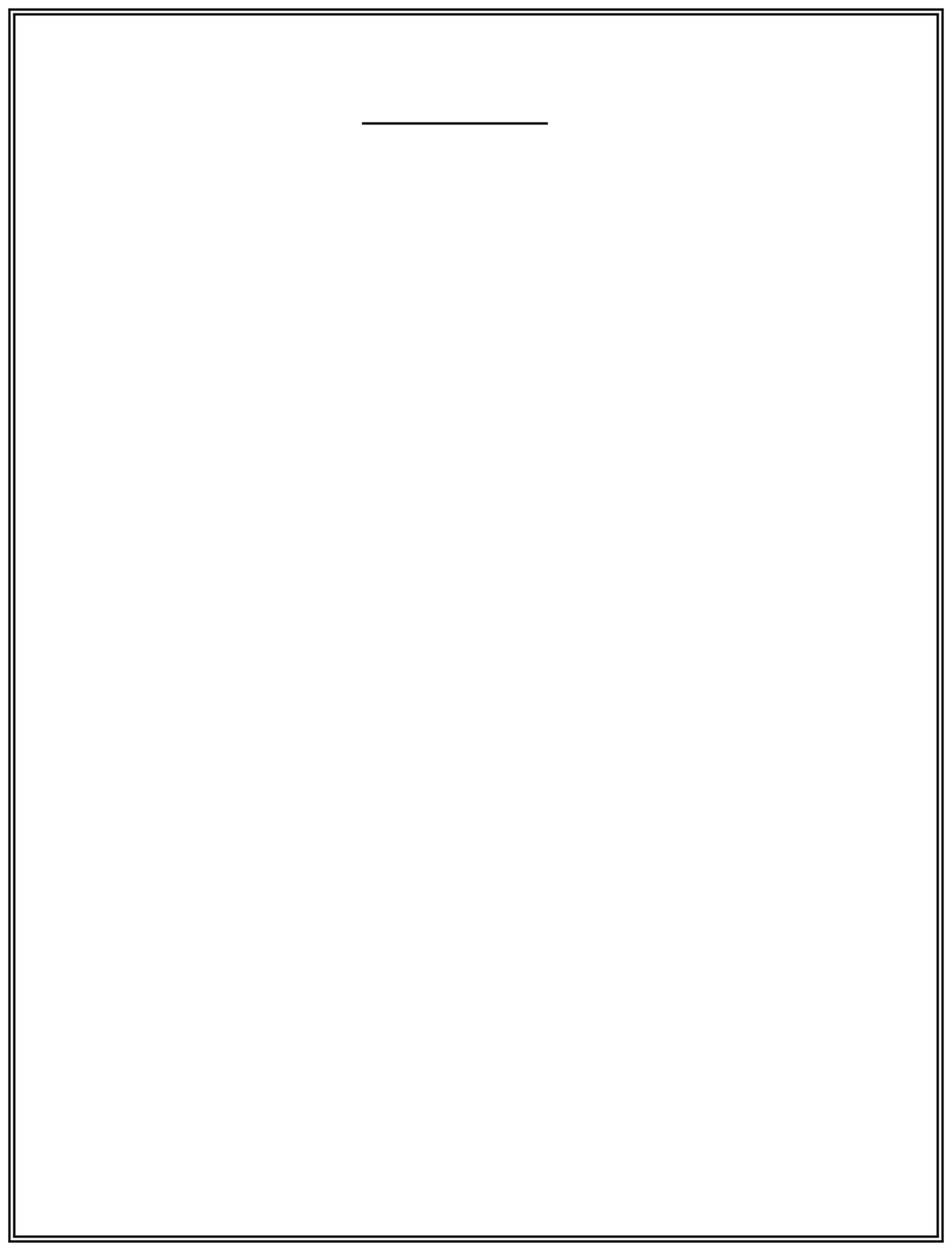
Assistant Professor, Head of Department, **COMPUTER SCIENCE & ENGINEERING**

#### for their constant encouragement and cooperation during the project report.

I would like to thank and express our gratitude to **Dr. SK. JAKEER HUSSAIN, M.Tech, Ph.D,** Dean of Engineering, Mother Teresa Institute of Science & Technology for the support and encouragement during the completion of the project report.

I express my gratitude and utmost regards to **Dr. C. HARI KRISHNA,M.E,Ph.D, MIEEE, MISTE,** Principal, Mother Teresa Institute of Science & Technology for their constant support, encouragement and providing necessary permissions during the project completion of the project.

#### I also thank the entire faculty members and fellow classmates who directly or indirectly helped us.

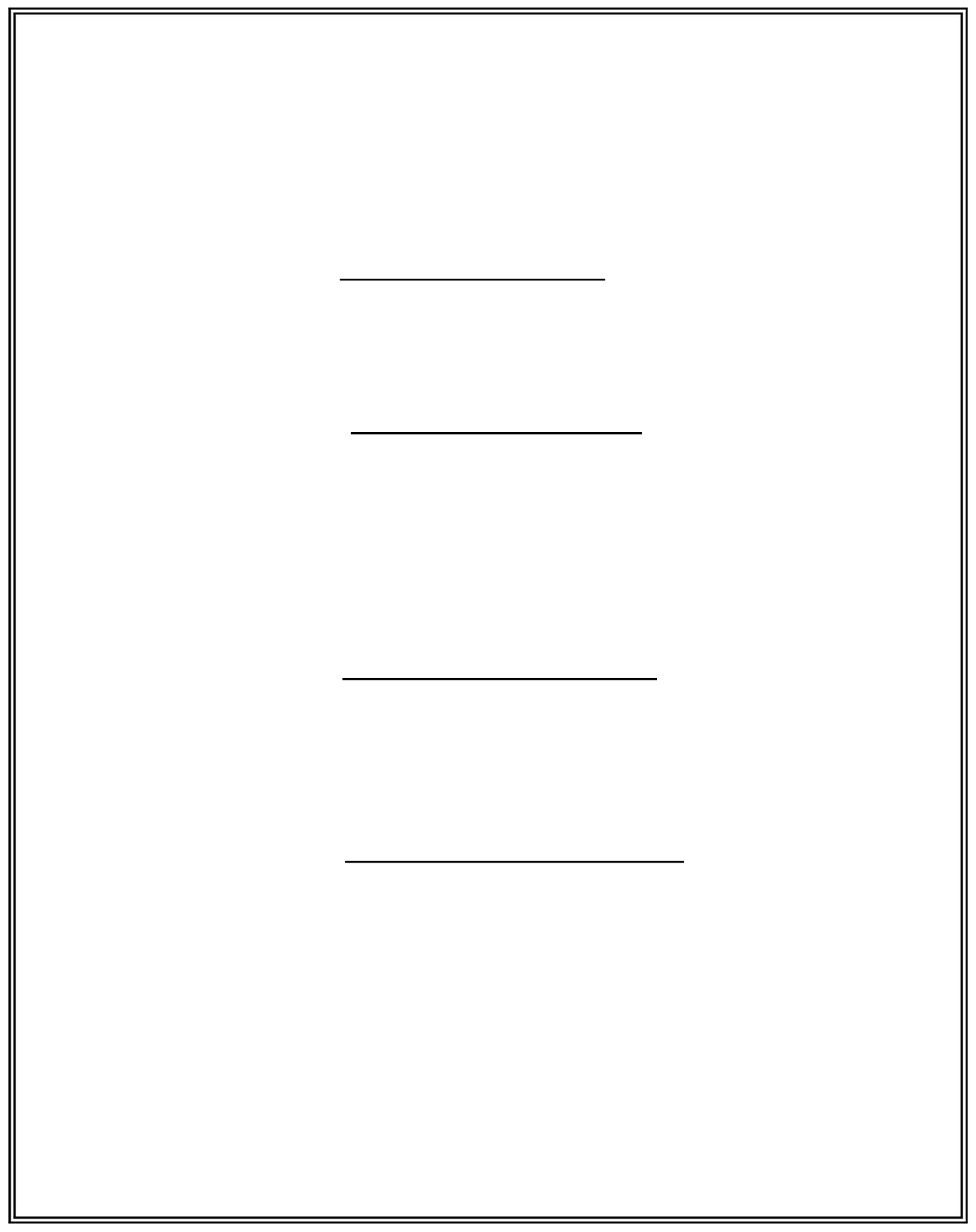
**DECLARATION**

We hereby certify that the Project Report entitled **“PETROL PRICE FORECASTING USING AUTO KERAS”** under the guidance of **Mr. M.VENKATESWARAO**, **M.Tech** is submitted in partial fulfillment of the requirements for the Award of the Degree in Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING**. This is a record of bonafied work carried out by us and the results embodied in this Project Report have not been submitted to any other University or Institute for the Award of any other Degree.

BY

**N.ASHMITHA (21C61A0537)**

**DATE:** Department of Computer Science & Engineering (CSE) Mother Teresa Institute of Science & Technology, Sathupally.



# INSTITUTE VISION

To be a state-of-the-art center for learning with a social commitment transforming the youth into Dynamic Professionals.

# INSTITUTE MISSION

**IM1**: Foster unmatched excellence in professional education.

**IM2:** Provide a quality eco-system to inspire learning aligned to needs.

**IM3:** Inculcate ethical and moral values to groom good citizens.

**IM4:** Involve in activities with team spirit and collaborations towards nation-building.

# DEPARTMENT VISION

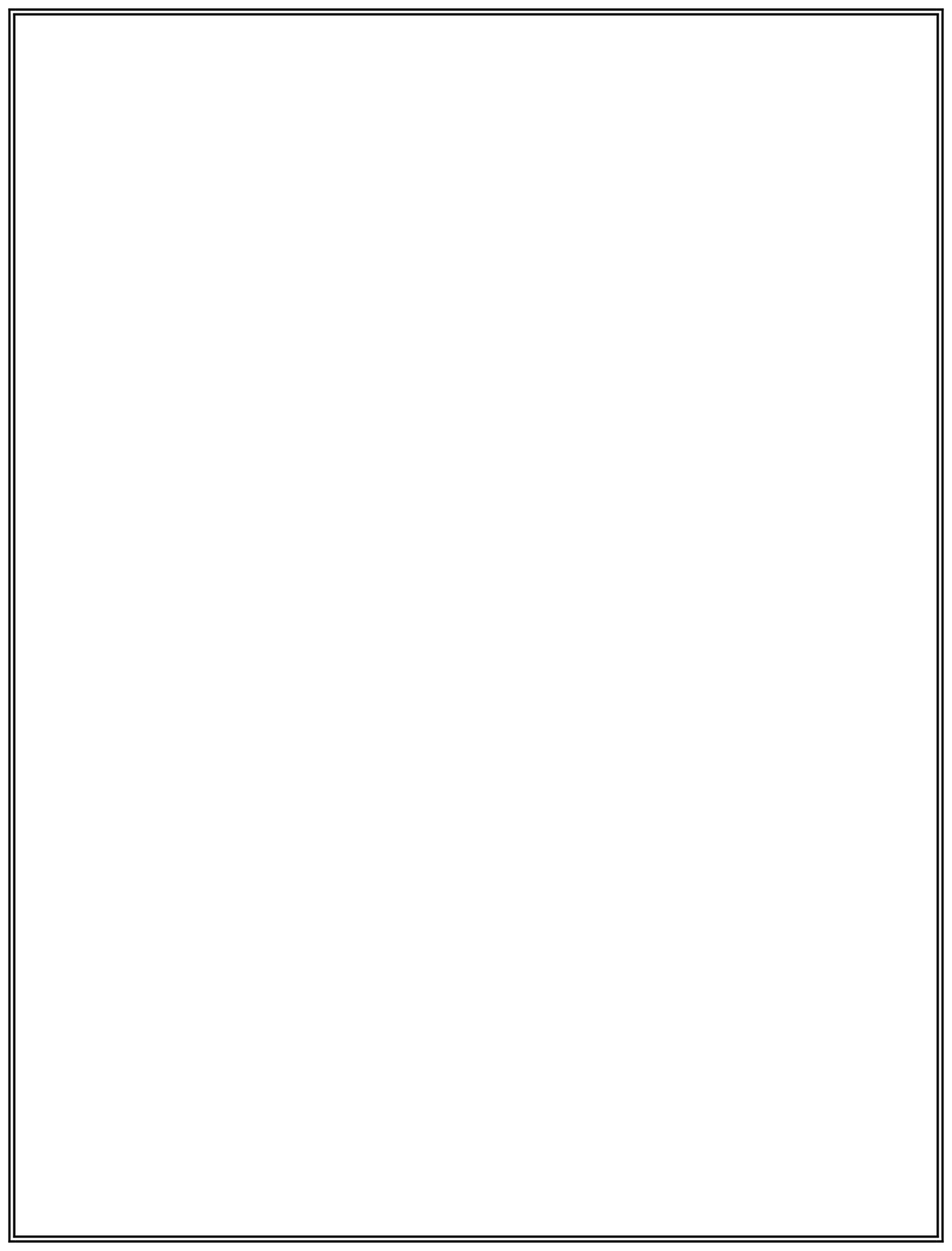
To be recognized as a contributor to **Computer Science & Engineering** proficiency and enable entrepreneurship, innovation, and values.

# DEPARTMENT MISSION

**DM1:** Provide a cutting-edge curriculum that integrates theoretical knowledge with practical skills, preparing students for the evolving tech landscape.

**DM2:** Establish strong partnerships with industry leaders to offer students real-world experience through internships, projects, and mentorship programs.

**DM3:** Promote ethical standards and social responsibility in the development and application of Technology.



### PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

**PSO1**: An ability to specify, analyze & design efficient computer applications that meet requirements and limitations.

**PSO2**: An ability to implement an algorithm for a secure and reliable data communication system.

**PSO3**: Able to excel in competitive exams Zest for higher studies with technical

competency

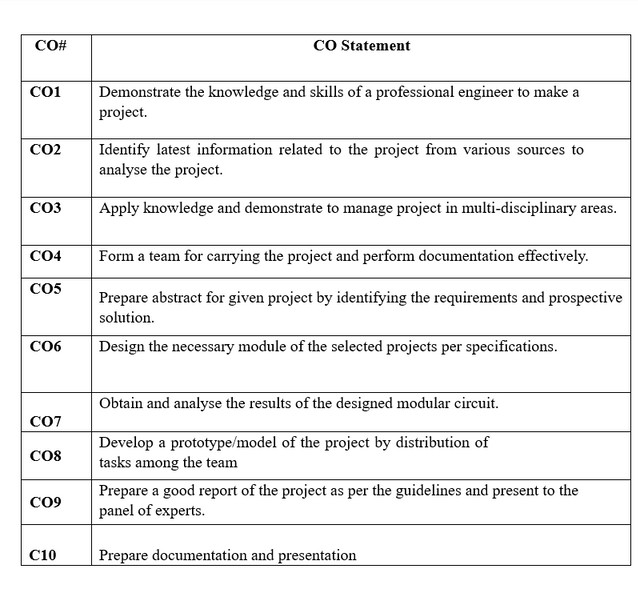
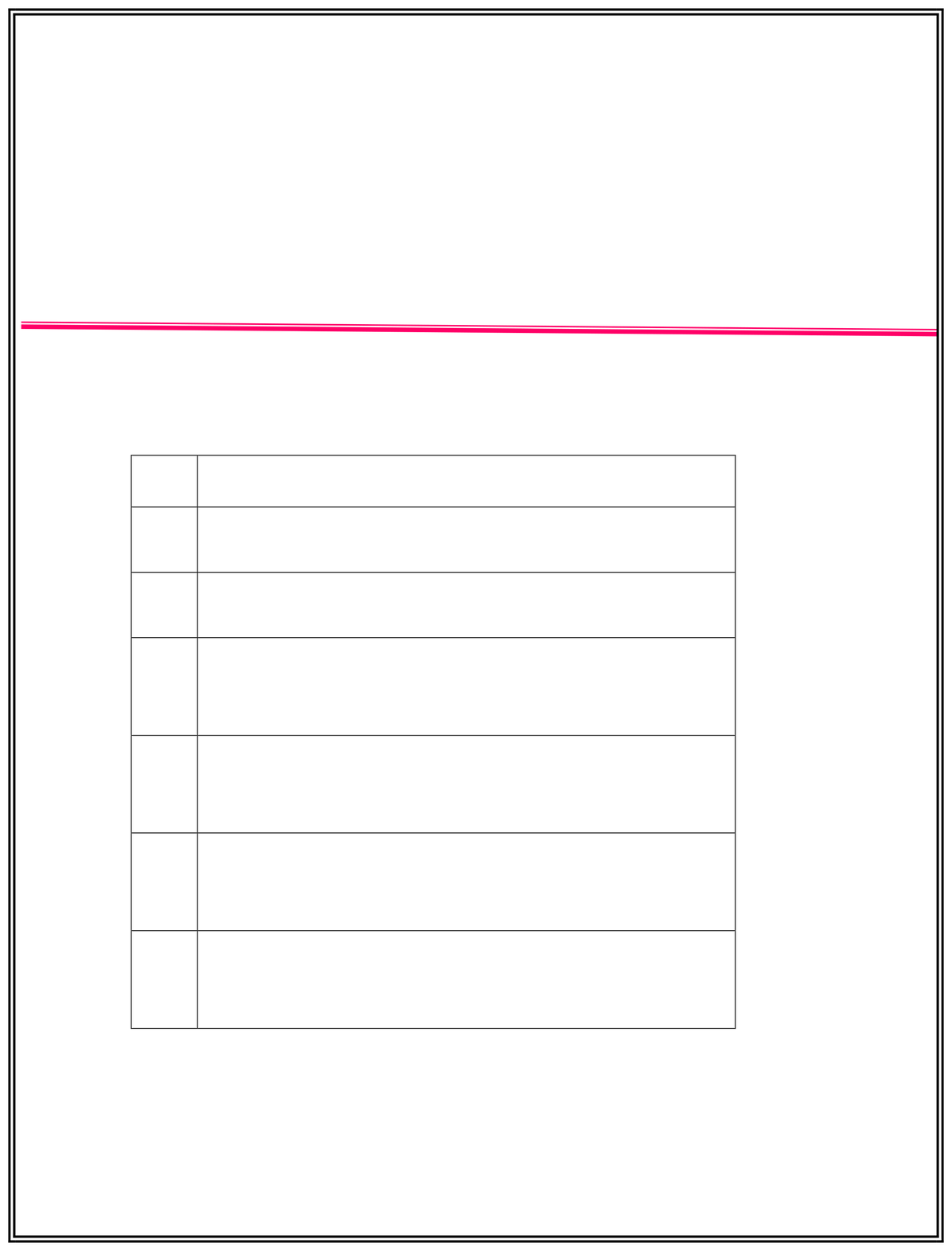
### PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

**PEO1:** Develop innovative computing products and complex solution techniques by utilizing strong technical proficiency and critical thinking.

**PEO2:** Demonstrate professionalism, ethical attitude, teamwork, and leadership

skills with lifelong learning.

**PEO3:** Exhibit effective communication skills and social concerns to meet the challenges of software industries.



Phone: Off: 9121901251 Web : [www.mist.edu.in](http://www.mist.edu.in/) E Mail: [info@mist.edu.in](mailto:info@mist.edu.in)

MOTHER TERESA INSTITUTE OF SCIENCE AND TECHNOLOGY

##### AUTONOMOUS

SANKETIKA NAGAR, KOTHURU (V), SATHUPALLY – 507303, KHAMMAM Dist., TELANGANA

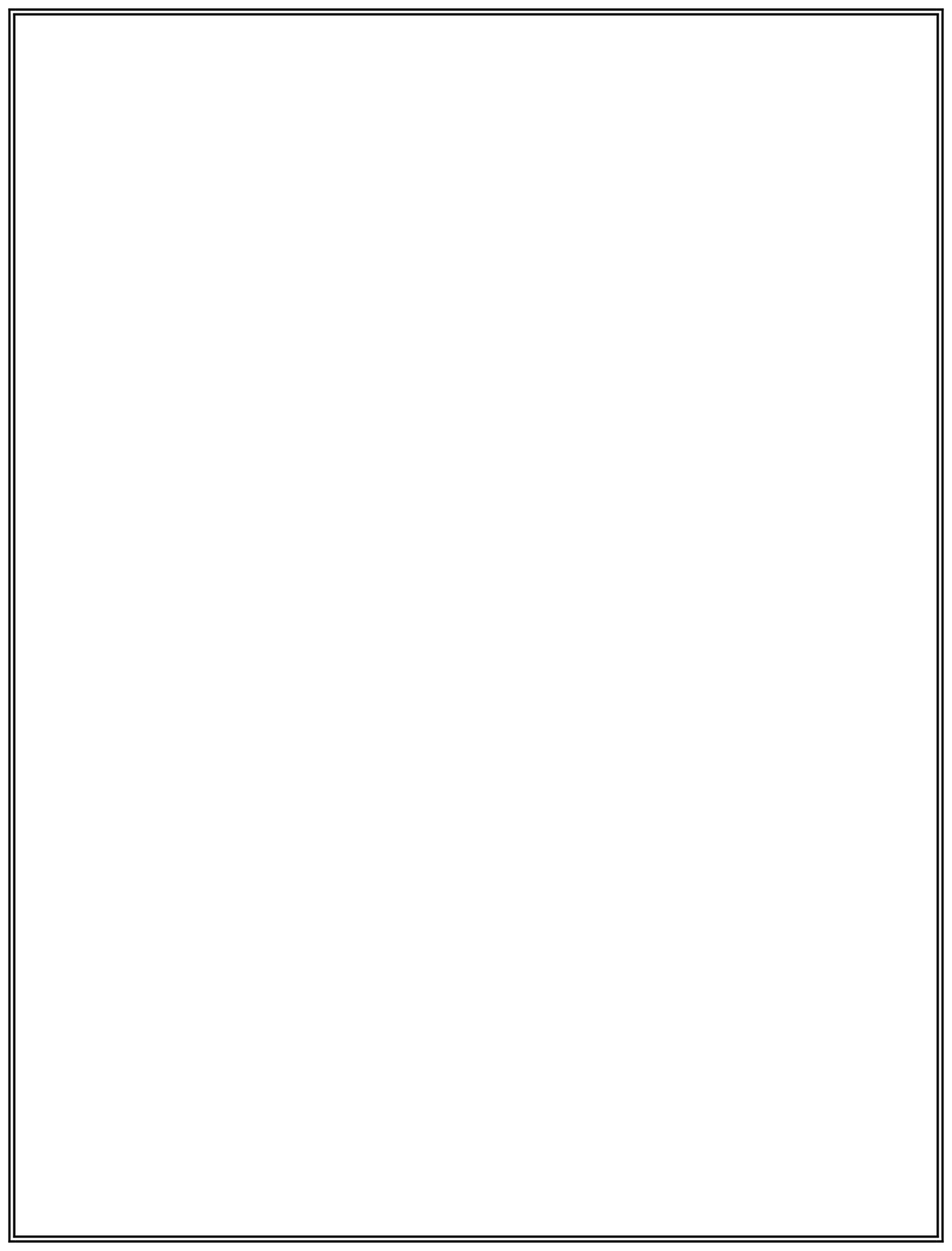
**Course: Project Stage II AY: 2024-2025**

**Class: IV B. Tech II Sem**

**Mr. P. MARESWARA RAO**, M.Tech, (Ph.D), MISTE

Associate Professor

Department of **COMPUTER SCIENCE & ENGINEERING**



# Department of Computer Science and Engineering Program Outcomes (POS)

### PO1: Engineering knowledge:

Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

### PO2: Problem analysis:

Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

### PO3: Design/development of solutions:

Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

### PO4: Conduct investigations of complex problems:

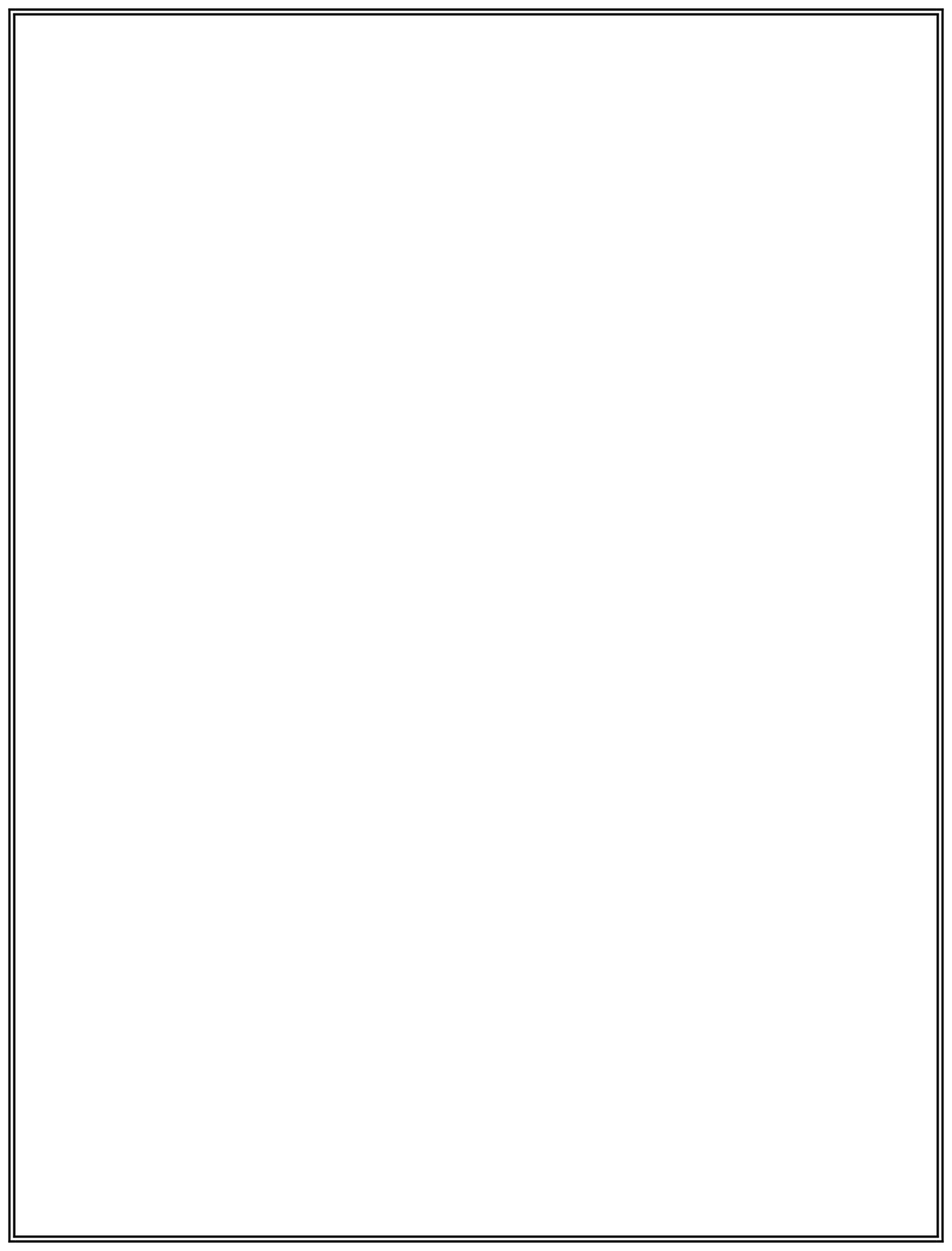
Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

### PO5: Modern tool usage:

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations

### PO6: The engineer and society:

Apply reasoning obtained by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.



# Department of Computer Science and Engineering Program Outcomes (POS)

### PO7: Environment and sustainability:

Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

### PO8: Ethics:

Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

### PO9: Individual and team work:

Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.

### PO10: Communication:

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

### PO11: Project management and finance:

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. **PO12: Life-long learning:**

Recognize the need for, and have the preparation and ability to engage in

independent and life-long learning in the broadest context of technological change.

Phone: Off: 9121901251 Web : [www.mist.edu.in](http://www.mist.edu.in/) E Mail: [info@mist.edu.in](mailto:info@mist.edu.in)

MOTHER TERESA INSTITUTE OF SCIENCE AND TECHNOLOGY

##### AUTONOMOUS

SANKETIKA NAGAR, KOTHURU (V), SATHUPALLY – 507303, KHAMMAM Dist., TELANGANA

**Project – PO Mapping**

**Project Title Guide(s): Student Name:**

**PREDICTION OF FOREST FIRES USING MACHINE LEARNING**

Mr. D. NAVEEN., M. Tech SEELAM SRIVANI

**Student Roll No:** 21C61A0547

**Academic Year:** 2024-25

**Name of Course from**

**which Principles are applied in this project**

Python

Project Work Project Seminar

Project Work

**Description of the application, page number in the report**

The major requirement for implementing this project is Python Programming Language along with its Libraries

PREDICTION OF FOREST FIRES USING MACHINE LEARNING

Able to prepare a thesis and presented to a panel of Experts Train the system to predict the forest fire occurence

**Attained PO**

PO1 PO2

PO7

PO8

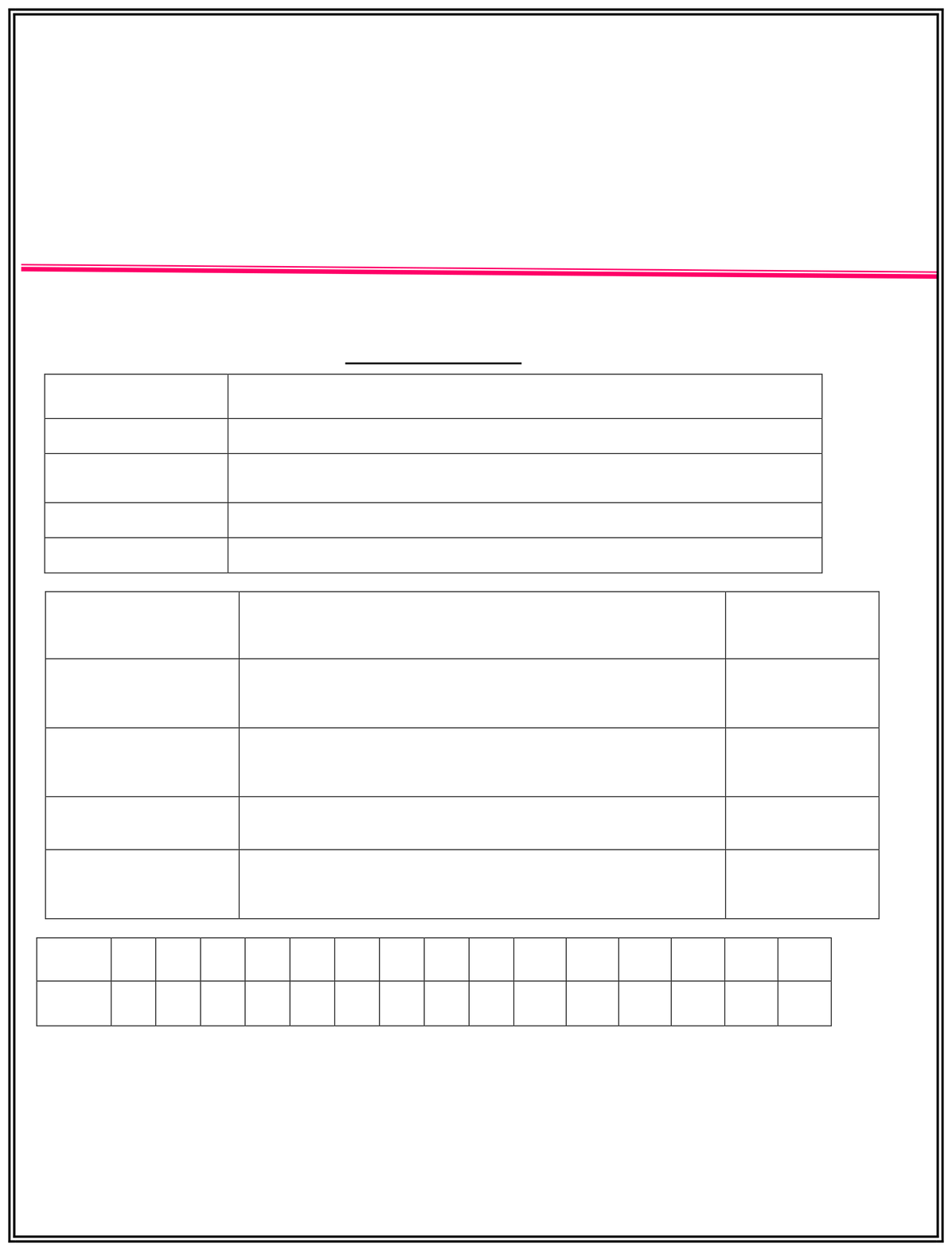
PO3 PO4

**PO** PO1 PO2 PO3 PO4 PO5 PO6 PO7 PO8 PO9 PO10 PO11 PO12 PSO1 PSO2 PSO3

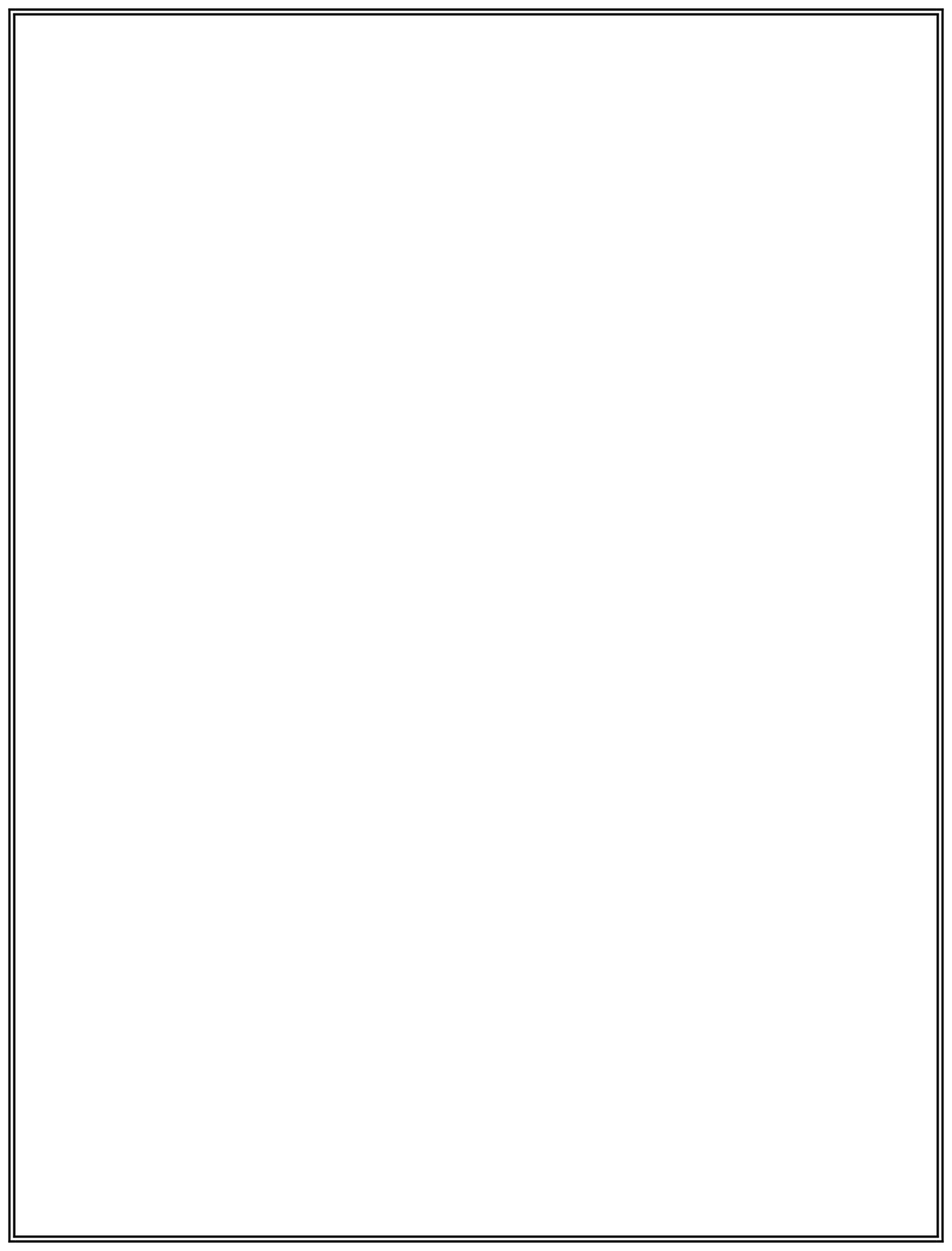
**Mapping** 3 2 3

1 3 3 1 1 3 3 3 3

3 3 2



**Guide/Supervisor Signature**



## PREDICTION OF FOREST FIRES USING MACHINE LEARNING

COURSE: BACHELOR OF TECHNOLOGY

BRANCH: COMPUTER SCIENCE AND ENGINEERING

## PROJECT DONE BY:

BATCH-10

N.ASHMITHA (21C61A0537)

## ABSTRACT

Forest fires are a growing global concern, causing significant environmental and societal damage due to factors like global warming, lightning, and human negligence. Early and accurate detection is crucial for effective mitigation. Current forest fire prediction methods often face challenges with computational time and accuracy, especially when dealing with large datasets.

Many existing machine learning techniques, while useful, exhibit varying levels of performance, with Support Vector Machines (SVM) showing promise for their accuracy in identifying smaller fires that can escalate into larger events. However, traditional SVM implementations can be computationally intensive and memory-demanding for large datasets, producing numerous support vectors that impact efficiency.

This project addresses these limitations by proposing a robust and efficient forest fire prediction system utilizing a Parallel Support Vector Machine (PSVM) algorithm. The primary objective is to develop a machine learning model that accurately predicts forest fires using meteorological data, leverages PSVM for enhanced speed and accuracy, and provides timely alerts to authorities. The methodology involves collecting comprehensive meteorological data from the Indian Meteorological Department (IMD), including temperature, wind, rainfall, and relative humidity. From these basic parameters, crucial forest fire weather indices (FFMC, DMC, DC, ISI, BUI, FWI) are calculated, which are vital for determining wildfire intensity.

The system architecture involves feeding the IMD dataset into a Parallel SVM model. The data is pre-processed, calculating dependent attributes based on independent attributes, and then split into training and testing sets. To overcome the computational challenges of large datasets, the project employs Apache Spark and PySpark for big data processing and parallel computing, significantly reducing computation time and improving accuracy by efficiently managing support vectors. The PSVM approach involves dividing large datasets into smaller subsets, processing them in parallel with multiple SVMs, and iteratively refining support vectors to achieve a global optimum with maximum margin from the hyperplane.

The front-end user interface is developed using the Django web framework, allowing users to upload test data for forest fire prediction and receive alert messages. Upon prediction of a fire (Fire = 1), an alert is sent to relevant departments, such as the Ministry of Environment, Forest, and Climate Change (MOEFCC), facilitating proactive fire management. The results demonstrate that the Parallel SVM model achieves higher accuracy (1.0) and recall (1.0) compared to a linear SVM (accuracy 0.94, recall 0.93), with an execution time of 0.051 seconds, validating its enhanced performance and efficiency in predicting forest fires. This system offers a reliable and efficient tool for forest fire prevention and management, enabling timely interventions and reducing destructive impacts.

## TABLE OF CONTENTS

#### S.No Title Page No.

Certificate Declaration Abstract Acknowledgement Table of contents List of Figures List of Tables

List of Abbreviations

1. INTRODUCTION
   1. Relevance of the Project
   2. Problem Statement
   3. Objectives
   4. Scope of Project
   5. Methodology
2. LITERATURE SURVEY
3. SYSTEM REQUIREMENTS SPECIFICATION
   1. Functional Requirements
   2. Non-Functional Requirements
   3. Hardware Requirements
   4. Software Requirements

SYSTEM ANALYSIS AND DESIGN

4

* 1. System Architecture
  2. Implementation Framework
  3. Dataset
  4. Flowchart

ii iii iv v vi viii ix x

1

3

3

4

4

4

6

16

16

16

17

17

18

18

19

20

25

1. IMPLEMENTATION 26
   1. [Support Vector Machine 26](#_TOC_250007)
   2. [Parallel Support Vector Machine 28](#_TOC_250006)
   3. [Front-end and Back-end implementation details 29](#_TOC_250005)
2. RESULTS AND DISCUSSION 39
3. TESTING 44
   1. [Performance of the model 44](#_TOC_250004)
   2. [Comparison between Linear SVM and Parallel SVM 46](#_TOC_250003)
4. CONCLUSION AND FUTURE SCOPE 47
   1. [Conclusion 47](#_TOC_250002)
   2. [Future Scope 48](#_TOC_250001)
5. REFERENCES 49

[APPENDIX- A 51](#_TOC_250000)



|  |  |  |
| --- | --- | --- |
|  | LIST OF FIGURES |  |
| S.No. | Title | Page No. |
| Fig 1.1 | Data Flow Diagram | 1 |
| Fig 2.1 | Sample Dataset | 6 |
| Fig 4.1 | System Architecture | 18 |
| Fig 4.2 | Django implementation framework | 20 |
| Fig 4.3 | Structure of FWI System | 24 |
| Fig 4.4 | Process Outline | 25 |
| Fig 5.1 | Support Vector Machine | 26 |
| Fig 5.2 | Hyperplanes in 2D and 3D | 27 |
| Fig 5.3 | Parallel Support Vector Machine | 28 |
| Fig 5.4 | Command Prompt | 29 |
| Fig 5.5 | User Interface | 36 |
| Fig 6.1(a) | Linear SVM Results | 39 |
| Fig 6.1(b) | Parallel SVM Results | 40 |
| Fig 6.2 | Predicted Fire Days | 41 |
| Fig 6.3 | Fire Intensity vs. Month | 41 |
| Fig 6.4(a) and (b) | Validation Predictions with Actual Fire Possibilities | 42 |

Fig 6.5 Forest Fire Prediction Report sent as Alert 43

## LIST OF TABLES

S.No.

Title

Page No.

Table 7.1

Table 2.2

Table 7.1

Performances in terms of accuracy, precision and Kappa 9

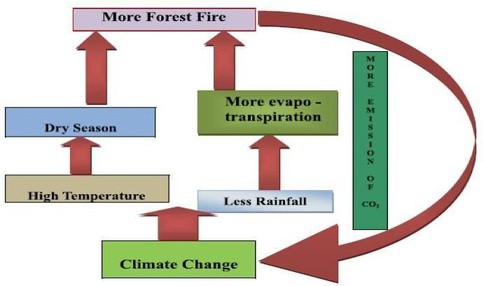
Comparison of different Methodology with pros and cons 13

Comparison between Linear SVM and Parallel SVM 40

## LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| Abbreviations | Full Forms |
| SVM | Support Vector Machine |
| FFMC | Fine Fuel Moisture Code |
| DMC | Duff Moisture Code |
| DC | Drought Code |
| FWI | Forest Fire Weather Index |
| PCA | Principal Computation Analysis |
| KNN | K- Nearest Neighbors |
| MPI | Message Passing Interface |
| PSO | Particle swarm optimization |
| CNN | Cascade correlation network |
| MPNN | Multilayer perceptron neural network |
| PNN | Polynomial neural networks |

CHAPTER 1 INTRODUCTION



Forest fires have become one of the most frequently occurring disasters in recent times which cause destruction to large acres of forests. One of the main reasons behind the occurrence of forest fires is global warming which is one of the major causes of increase in average temperature of the earth. The other reasons are due to lightning, thunderstorms, and human negligence. Forest fires can lead to deforestation which has a lot of negative impact on human society. It is reported that for the last decade, each year, more than 100,000 in all countries. Early detection of forest fires is very important in fighting against fires. Spread features of forest fires show that, in order to put out a fire without causing any permanent damage in the forest, the fire fighter center should be aware of the threat at most 6 min after the start of the fire.

Forest fires prediction combines weather factors, terrain, and dryness of flammable

items, types of flammable items, and ignition sources to analyze and predict the combustion risks of flammable items in the forest. There are many techniques that the authorities use to detect forest fires, satellite detection is widespread among worldwide authorities. Few forest authorities use human observers as detectors and reporters of forest fires.

Fig 1.1: Data Flow Diagram



Fig 1.1 displays a data flow map for forest fires and climate change worldwide. In this article, we compare the various methods used in related research. In each of the methods used, there are different advantages and disadvantages. Data mining and machine-learning techniques may provide an effective mitigation strategy, where forest-related data can be used to predict forest fires. Owing to changes in earth's orbital parameters, solar intensity, and atmospheric composition, the environment and associated weather are not continuous.

Our atmosphere has warmed up in recent years due to changes in human behavior in

the air in radioactively active gases (carbon dioxide, methane, etc.). Such warming is likely to have a profound impact on fire activity in the forest zone.

Based on weather altering factors, we restrict our research to forest fires. From recently

on, it appears that they greatly impact the frequency of wildfires. By analyzing these causes, we are developing a stronger and more efficient way of predicting forest fires. The IMD (Indian Meteorological Department) dataset is split into training data and test data. The training data were used in the supervised learning model. This model will now forecast forest fires on the basis of the input test results. For testing various test data to forecast the occurrence of forest fires a user interface is created.

There are many Machine Learning models used to predict forest fires and alerts the

forest department. Some of the techniques used to detect these fires are logistic regression, random forest, K-Nearest Neighbor, Support Vector Machine, Artificial Neural Networks, Gradient Boosting, Bagging, etc. There are various advantages and disadvantages in each of the methods used. From our literature survey we concluded that SVM is the most suitable technique which can be used. SVM is not only a good classification algorithm but also a good regression algorithm. SVM gives accurate results in case of small fires which eventually lead to large fires. Therefore SVM is the most efficient method for forest fire prediction.



SVM for large datasets produce a large number of support vectors which increase the computation time and storage requirements. Therefore SVM is implemented using parallel computing, in our case parallel SVM. Splitting the dataset into sub datasets and using subSVMs to evaluate each dataset to filter the support vectors from non-support vectors so that the last global sub-SVM will have only one set of support vector which is easy to classify and because we use many sub-SVMs simultaneously, computation speed is increased and accuracy of prediction also is increased.

#### Relevance of the Project

Data mining and machine learning techniques can provide an efficient prevention approach where data associated with forests can be used for predicting the occurrence of forest fires. Fire activity is strongly influenced by four factors – weather/climate, fuels, ignition agents and human activities. Climate and the associated weather are not constant due to changes in the earth’s orbital parameters, solar output and atmospheric composition.

Recently, our climate has been warming as a result of increases of radioactively active gases (carbon dioxide, methane etc) in the atmosphere caused by human activities. Such warming is likely to have a profound impact on fire activity in the forest zone.

We are limiting our study to predict forest fires based on weather altering factors. As

of recent, they seem to be affecting the occurrence of wildfires greatly; we aim to devise a more accurate and faster method to predict forest fires by studying these factors.

#### Problem Statement

To predict the occurrence of forest fire using Parallel Support vector machine algorithm to provide accurate and faster results by calculating forest weather index for basic weather parameters which is an important factor to determine the intensity of the wildfire.



#### Objectives

* + - To create a machine learning model
    - It is used to predict forest fires
    - It uses meteorological data
    - It uses support vector machine (SVM) technique for better accuracy for small forest fires
    - It uses parallel computing (parallel SVM) for high speed computation
    - To create a user interface for predicting forest fires and display an alert message to authorities like the Ministry of Environment, Forest, Climate Change (MOEFCC).

#### Scope of the Project

The goal of this project is to predict forest fires using machine learning algorithms. This project uses SVM to provide better accuracy. It also uses parallel computing for high speed computation.

The dataset provided by IMD (Indian Meteorological Department) will be divided into training data and test data. The training data is fed into the model for supervised learning. This model should now predict forest fires based on the test data given as input. A user interface is developed for testing different test data to predict the occurrence of forest fires.

#### Methodology Parallel Computing

* + - Traditional computation instructions in a queue pass through the processor and

processor does the processing and get the processed data

* + - The problem with this type of approach is we can only make the processor faster to limit i.e., there is a frequency limit, heat.
    - To increase the speed of processing we use two processors which divide the instructions among them.



* + - Two processors running in parallel dividing the instructions among them (workload distributed)

#### Support Vector Machine

* + - It is a supervised learning algorithm for classifying data
    - Uses hyperplane to separate data to segments where each segment has one kind of data
    - Classification based on features of the data
    - SVM is used for regressions as well
    - SVM kernel function for classifying non linear data
    - Plot the data as nodes on n dimensional space and draw hyperplane
    - Closest points to the hyperplane are called support vectors and have max distance from the hyperplane

#### Parallel SVM

* + - SVM is taken as a most efficient classification and regression model.
    - Classical SVM model is difficult to analyze large scale practical problems.
    - Parallel SVM can improve the computation speed greatly.
    - In our project we plan on handling large datasets using Apache Spark and use parallel SVM for more accuracy with better computation speed.



## CHAPTER 2

LITERATURE SURVEY

Forest fires are managed by weather data collection or by the processing of satellite forest photos. Rapid detection can help us control damage effectively. Earlier work has shown that forest fires can be caused for many causes and it is therefore very important to accurately make predictions. Many machine learning techniques were applied and compared in order to construct a prediction model. The SVM model provides reliable forecasts for the use of the environment in small forest fires, which inevitably lead to larger fires. When forecasting forest fires using weather information, the forest weather predictor along with basic weather parameters must be taken into account. We show that through parallel computation with vector support engines the accuracy of this predictive model can be enhanced.

#### Supervised and Ensemble Machine learning algorithm

To analyse the data in datasets and how each feature affects the forest fires and some of the techniques used for solving regression and classification problems used in this study are discussed. The dataset includes: X and Y axes special park coordinates, Fine Fuel Moisture Code (FFMC), Duff Moisture Code (DMC), Drought Code (DC) and Initial Spread Index (ISI).The other characteristics used are temperature, relative humidity, wind speed, rain outside and the burnt forest region. All these features were collected from January to December on all days for a whole year.

Fig 2.1: Sample Dataset



Fig 2.1[1] represents a sample (first 10 rows) of the UCI forest fire dataset. In this paper, there are two graphs plotted, variance of feature with respect to fire and variance of feature with respect to each month. The variance calculated for rain is 0.09 which is close to 0; therefore it does not affect the model. From the dataset, we inferred that, the number of forest fires that happened is 270 and not happened is 247. A bar graph is plotted showing the frequency of occurrence of forest fire with respect to each month. It was observed that forest fires were frequent in the month of August and September and less frequent in May and April and in the other months it is sporadic. The normalization techniques used are Principal computing analysis (PCA), Feature Scaling (Min Max Scaling) and label encoding. The ML techniques used are Logistic regression, Bagging, SVM, KNN, Boosting and Random Forest.

When PCA was applied, Logistic regression had the highest classification rate of 68%.

When PCA was not applied, Gradient Boosting had the highest classification rate of 68%. SVM is best for small fires which lead to larger fires.

#### Soft computing approaches

To compare the various algorithms under Artificial Neural Networks and use Soft Computing Techniques to find the best algorithm for forest fire prediction. The aim of this paper is to examine the quality of five SC techniques to determine the best and most effective forest fire predictor. These methods include cascade correlation network (CCN), multilayer perceptron neural network (MPNN), polynomial neural network (PNN), radial basis function (RBF) and support vector machine (SVM). The data collected from the UCI machine learning database, which took 517 different entries for montesinho natural park (MNP) at different times.

Detection of forest fires requires a variety of sensors distributed over a wide area to be deployed. Pre-processing the data set is the first stage. For this purpose, the principal component analysis (PCA) and particle swarm optimization (PSO) techniques were used to find the critical features and to segment the fire regions (clusters) respectively. Then, five predictors were applied to identify the best and suitable predictor that has the ability to predict forest fires.



The ANN methods were: Principal component analysis (PCA), Particle swarm optimization (PSO), Cascade correlation network (CCN), Multilayer perceptron neural network (MPNN), Polynomial neural networks (PNNs), Support vector machine (SVM).

The results show that SVM predicts fire probability well. SVM has the smallest RMSE

of 54.0, MSE of 2926.4, RAE of 10.5, and MAE of 2.656 and the highest IG of 2.656 in the testing stage. In general, the results indicate that SVM has the best prediction ability for forest fire compared to other selected SC methods.

#### Data mining techniques to predict Forest Fires

The reason for this study was to learn how to predict forest fires in Slovenia using different data mining techniques based on the forest structure GIS (geographical information system) and the weather prediction model - Aladin and MODIS. The data is divided into 3 groups: GIS data (geographic data, part of the land with forest, field, urban part, distance from roads, highways, railways, cities etc.), MODIS data with dependence from the day of the year (average temperature for specific quadrant for specific day, average net primary production for specific day), ALADIN data (temperature, humidity, sum energy, evaporation, speed, direction and course of the wind, transpiration etc.). The data were analyzed with several different data mining algorithms like: logistic regression, random forests, decision trees, bagging and boosting ensemble methods.

Performances of the experiments were measured in terms of precision, recall, accuracy

and kappa statistics for each of the datasets respectively as shown in Table 2.1[3]. From the results we can conclude that Bagging of decision trees shows the best results in terms of predictive accuracy, precision and kappa statistics compared to the other algorithms.

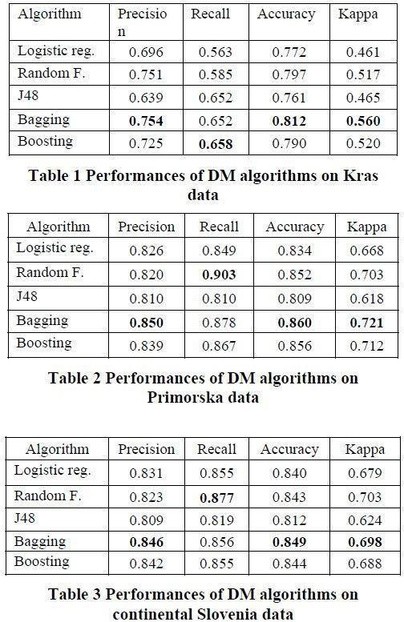


Table 2.1: Performances in Terms of Accuracy, Precision and Kappa

#### Review on various approaches

This paper was a review on different papers to find out the advantages and disadvantages of existing techniques like:

Artificial Neural Network- An Artificial Neural Network (ANN) is inspired by the way biological nervous systems, such as the brain, process information.

Fuzzy logic- Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0).



Image Processing- Image processing is a method to convert an image into digital form and perform some operations on it to extract some useful information from it.

Satellite Constellation- A group of artificial satellites with coordinated ground coverage, operating together under shared control.

Multisensor Data Fusion- Multisensor data fusion is the combining of sensory data from separate sources such that the resulting information is better than individual sources.

Intelligent system- An intelligent system is a computer system that uses infrared images along with ANN with additional information from sensors which can make decisions.

#### Data Mining Approach to predict burned area of Forest Fires

Five different DM techniques, e.g. Support Vector Machines (SVM) and Random Forests, and four distinct feature selection setups (using spatial, temporal, FWI components and weather attributes), were tested on recent real-world data collected from the northeast region of Portugal. The forest Fire Weather Index (FWI) is the Canadian system for rating fire danger and it includes six components: Fine Fuel Moisture Code (FFMC), Duff Moisture Code (DMC), Drought Code (DC), Initial Spread Index (ISI), Buildup Index (BUI) and FWI.

The proposed solution, which is based on SVM and requires only four direct weather

inputs (i.e. temperature, rain, relative humidity and wind speed), is capable of predicting small fires. This procedure indicates that all weather conditions affect the model, with the outside temperature being the most important feature, followed by the accumulated precipitation.

#### Parallel SVM Based on MapReduce

This mode was to use a support vector machine for classification and regression. Large computation and storage requirements increase due to the increase in the number of training vectors, therefore parallel SVM is studied to increase the speed of computing.



The large dataset is first handled using mapreduce. Mapreduce is an efficient distributed computing model to process large scale data mining problems. Mapreduce is developed in software tools like Hadoop and Twisters. Hadoop and Twisters are open source mapreduce software. Mapreduce in Hadoop does not support iterative map reduce tasks. Twisters support both iterative and non-iterative map reduce and combine tasks. Many SVM software models have been developed like, libSVM, lightSVM, ls-SVM and so on. LibSVM is taken as the most efficient SVM model and widely applied in practice because of its excellent property. Using parallelization, training samples are divided into subsections. Each subsection is trained with a libSVM model. The non-support vectors are filtered with subSVMs. The support vectors of each subSVM are taken as the input of the next layer subSVM. The global SVM model will be obtained through iteration. This shows that parallel SVM along with MapReduce reduces the computation time.

#### Artificial neural networks and logistic regression to predict forest fire danger

A model based on the Galician region (north-western Spain) has been proposed and image MODIS was used to track the status and the acquiring land surface temperature (LST). The following have been found: LST 8 days, fire and year history. The LST is an important parameter because higher temperatures correlate with lower humidity, which enables vegetation to ignite In order to assess forest fire hazards from remote sensing and fire history data, artificial neural network (ANN) and logistics regression were used for this work. The land surface temperature and EVI (Enhanced Vegetation Index) were remote sensing inputs used. Different input combinations with logistic regression have been tested. In an artificial neural network, combinations of variables have been implemented and results obtained by the two techniques have been compared. Increased accuracy and recall in artificial neural networks compared to logistic regression. This description was helpful in determining maps of fire hazards that can prevent fires.



#### Forest Fire Danger Index Using Geo-Spatial Techniques

K V Suresh Babu [5] provides a comprehensive explanation of various worldwide indices of forest fire conditions, groups of forests and their areas, classification of different rates of hazards, fire accidents in various classes of land cover, level of fire danger of different types of property, categories of vegetation fire accidents, fire hazards for Uttarakhand vegetation categories. In addition, he has described different class and FWI rating systems, such as the Canadian Forest Fire Danger Rating System, the Canadian Fire Risk Index System and the National Fire Danger Rating System. The different forms of forest in India are also being studied. We plan to use his work to measure forest fire weather indicators and measure fire hazard rates thresholds to improve the performance and accuracy.

#### Mapping regional forest fire probability using artificial neural network

The Multi-Layer perceptron (MLP) techniques based on back propagation algorithms for data on physical, climate, anthropogenic and fire incident facts were explored by Onur Satir [9] et al. in the Mediterranean forest in Turkey. It was concluded that each area should be studied separately for exact fire risk maps in order to accurately track risks posed by forest fires with the most concise features being tree canopy covering, temperature and Digital Elevation Map (DEMs) and according to cause of fire, vegetation dynamic, climatic conditions and physical environment structures.

#### Summary

A comparison on the pros and cons of the different algorithms like Linear regression, Support vector machine, non-linear regression, Random forest, Artificial neural networks and parallel support vector machine that were used in the previous works have been discussed below in Table 2.2.

Table 2.2: Comparison of different Methodology with pros and cons

Algorithm Key Idea Formulas

Pros

Cons

Linear Regression

Method for linking independent variables with dependent variables and finds a linear relationship between

them. It aims to find the value of Y such that the

difference between the predicted value and the

actual value is minimum.

The cost function can be used to find the best values for and which

provides the best fit line for the points.

The equation is of the form,

Y= a +b X,

Where Y is dependent variable and X is independent variable

The cost function is given by,

𝐶

𝑛

1

= ∑(𝑝𝑟𝑒𝑑𝑖

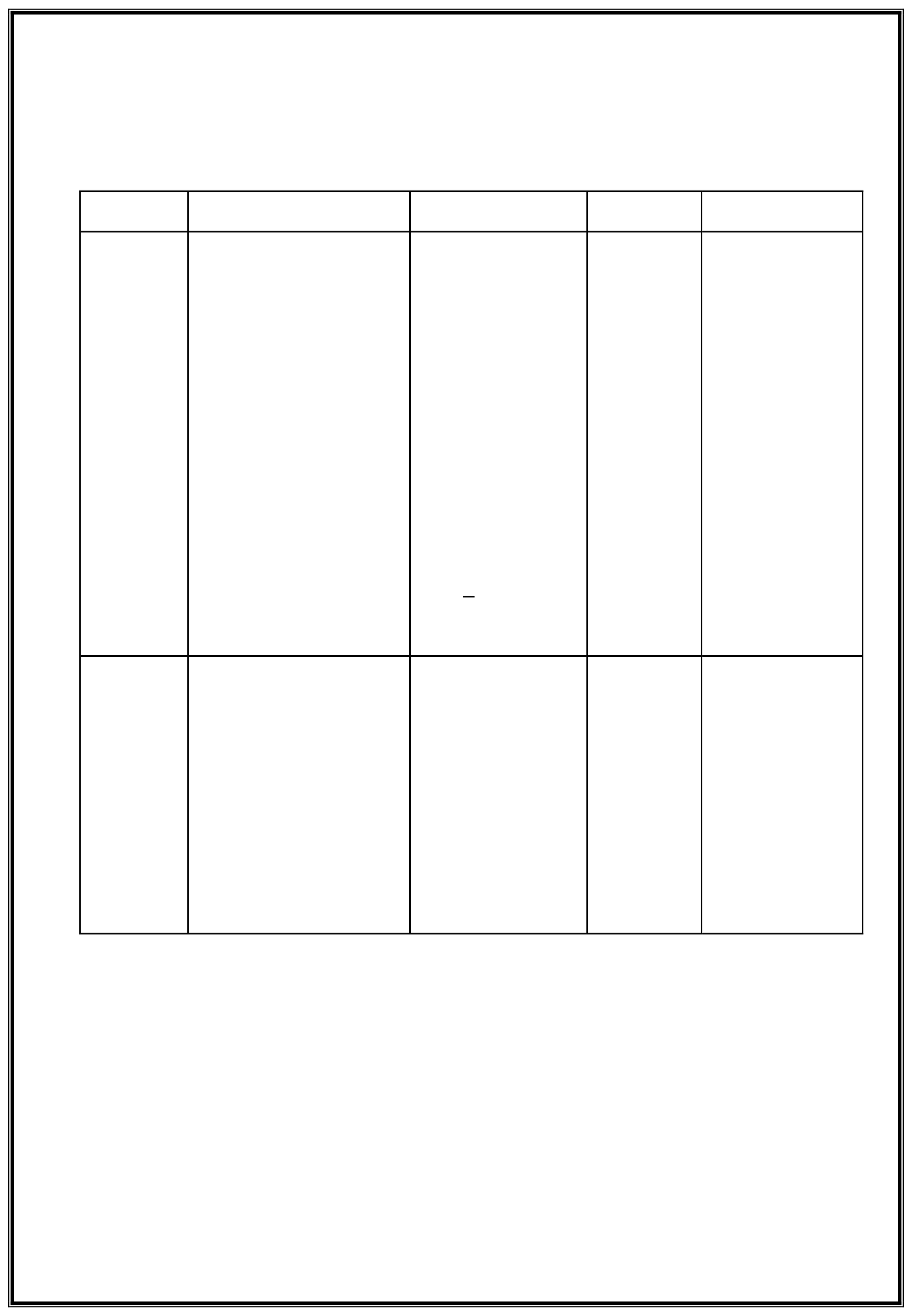
𝑛

Simple and easy to implement

Most problems in

the real world are

not linear and are not realistic



Non- Linear Regression

The dependent variables act as a nonlinear part of model parameters and one or more independent variables. It typically generates a curve where Y is a random variable. It is more complex to develop because it uses multiple iterations.

𝑖=0

– 𝑌)𝑖 2

The equation is of

the form,

Y = f(X,β) + ε Transformed to intrinsically linear

Y = θ1 + θ2\*1/X

Well deals with small datasets and effectively uses information to predict unknown parameters

The most productive model research can be

established because the experimental information does not have

concrete cinematic values.

Random Forest

Many decision trees act as a group in which each individual decision tree gives a class

To make a prediction at a new point x then,

fˆB(x) =1/BΣ (b1, B) (Tb(x)).

Where Tb is the random forest tree and we take the average of B

Works well for large datasets and avoids overfitting

Random forests are difficult to construct

prediction, and this is such trees.

the final prediction for the class with the highest votes. There is low correlation between the trees/models and the predictions are more accurate

Artificial Similar to how the In general if there are n variables

problems by

averaging or combining decision tree results.

It is durable,

and

timely

Requires

neural networks

brain has interconnected

then the equation for ANN will be

versatile and can be used

high data and the

neurons and

f(x)=b+w1⋅x1+w2⋅x2+...+wn⋅xnin complex

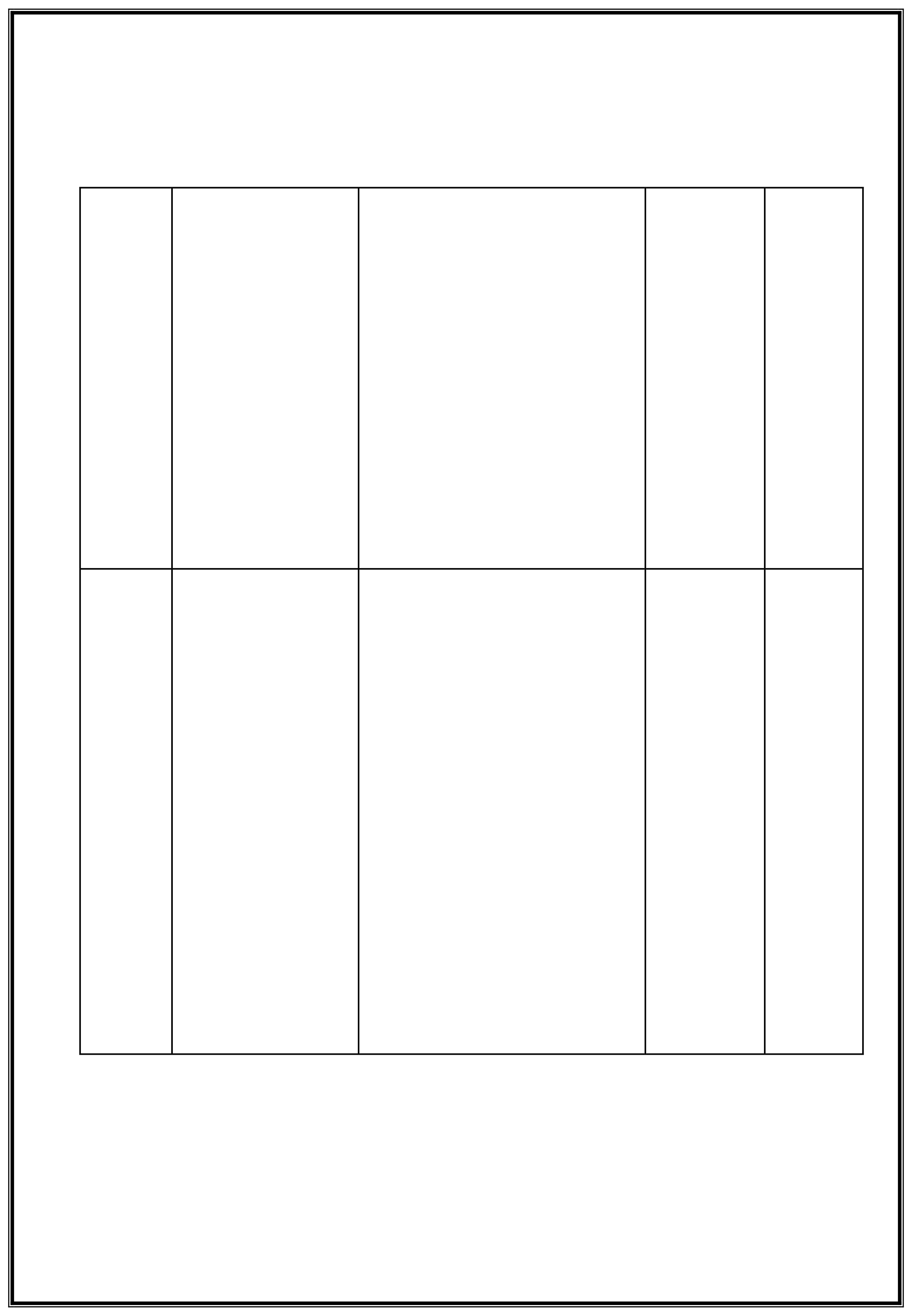
chances

processes

where w is the weights and x is

designs. The

of over-

information, we have the data points. multiple

interconnected elements that work together simultaneously to solve a problem.

They are multi-layer connected nets.

There are three layers, the input layer, hidden layer and output layer.

result is obvious

fitting are high

Support

Every data component is The hyperplane Functions well

If the number

Vector Machine

drawn as one point in an can be found n-dimensional space and using the

in large spaces. It is efficient in

of features is much greater

a hyperplane can be equation: memory because ntuhmanbtehreof samples,

defined to differentiate between the classes by maximizing its margin.

w.x+b = 0, where w is the

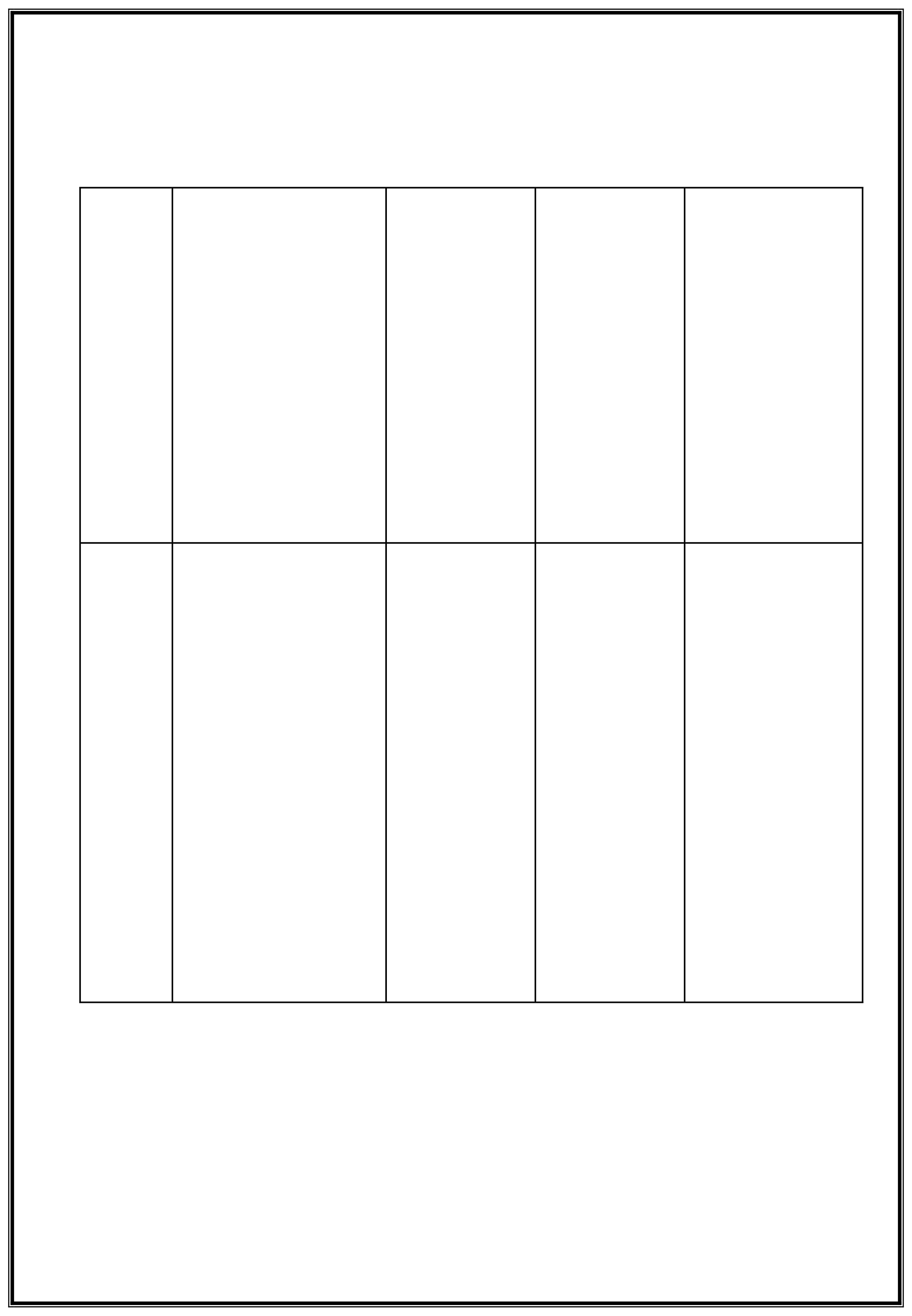
it support vectors which form part of

chances of over- fitting are possible. No memory and

Having maximum

normal vector to garoup of

processing time



Parallel Support Vector Machine

margin between the support vectors and hyperplane is important as this will ensure accurate predictions.

Parallel methods for large scale classification are implemented to

speed up the SVM

process. Here the model has multiple SVMs working in parallel on divided datasets. Each SVM will produce support vectors which will be sent to another set of SVMs. This will continue till we get a refined set of support vectors which will not change further.

the hyperplane and x is the set of points. The width of the margin is (2/|w|)

SVM is implicitly parallelized to

get smaller SV

sets. Parallelizing SVM with split dataset to get refined set of support vectors

training vectors.

Reduces memory from O(n2)

to O(np/m)

and improves computation time to O(np2/m) Where n is the number of instances, p is the reduces matrix and m is the number of machines

Portable.

Reformulations are too memory intensive.



## CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

SRS is a document that completely describes what the proposed software should do without describing how the software will do it. It’s a two-way insurance policy that assures that both the client and the organization understand the other’s requirements from that perspective at a given point in time. Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable attributes.

#### Functional Requirements

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality.

* + - To create a machine learning model to predict forest fires using SVM technique and

LIBSVM model.

* + - To increase the computational speed we use parallel SVM.
    - To create a user interface for predicting forest fires and sending an alert message to authorities like the Ministry of Environment, Forest, Climate Change (MOEFCC).

#### Non-Functional Requirements

Non- Functional Requirement defines the performance of a software system and how scalable and reliable the system is and it includes scalability, recoverability, availability, capacity etc.



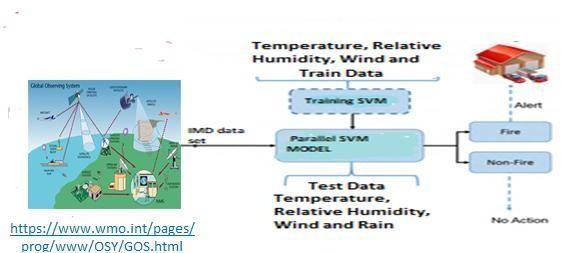
* + - To give a user-friendly experience using the user interface to check and test the dataset.
    - It can be used in real-world using real-time meteorological data.
    - To make the application extremely scalable.
    - To test the application and ensure maintainability on a timely basis.

#### Hardware Requirements

* + - Processors: Intel i3,i5,i7
* Processor Speed: 3.00GHZ
  + RAM: 4GB
  + Storage: 50GB
  + Monitor: 15inches
  + Keyboard: Standard 102 keys
  + Mouse: Standard 3 buttons

#### Software Requirements

* + - Operating System can be either Windows 7,8,9,10, XP
    - Pandas, Numpy, Sklearn, Pyspark, Matplotlib modules in python
    - Jupyter notebook
    - Pyspark for implementing Parallel SVM, Binary and Multi class evaluator modules
    - Visual studio Code
    - Django



## CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

We have used Django web framework to build the UI for this predictor model. The implementation framework is discussed in detail in this section. Data set is collected from IMD to train and test the model. Based on our literature survey we will be using SVM to predict forest fires. SVM is best for small fires and parallel SVM model is implemented to make the predictions accurate and fast. A detailed explanation about the model and how it is implemented is discussed in the below section.

#### System Architecture

Fig 4.1: System Architecture



As shown in Fig 4.1, the IMD dataset is fed as input to the Forest fire detection system implemented using Parallel SVM wherein the dataset contains only the basic attributes like Temperature, Relative humidity, Wind and Rain. Then the data is processed by calculating the Fire Weather Indices (FWI). After the data processing, the model is trained using the train data and for testing purpose, the test data is fed with the same attributes as the train data. The model produces a result based on whether Fire happened (Fire =1) or not. If the output contains Fire = 1, then an alert is sent to the forest department and other concerned departments to take the necessary actions.

#### Implementation Framework

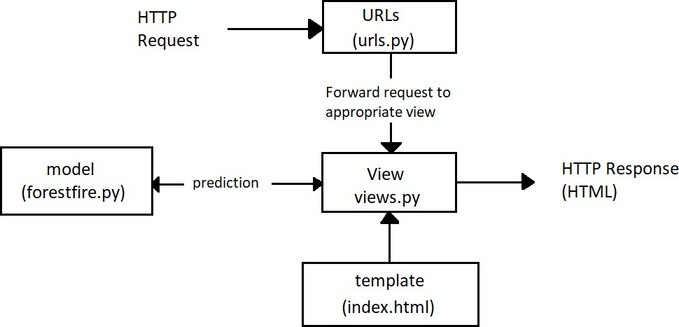
Fig 4.2 is a web application using Django web framework. Like in a traditional data- driven website, a web application waits for HTTP requests from the web browser (or user). When a request is received the application works out what is needed based on the URL and possibly information in POST data or GET data. Depending on what is required it will perform tasks required to satisfy the request, in this case it will take the file uploaded and will run the programs in the background to predict forest fires. The application will then return a response to the web browser, often dynamically creating an HTML page for the browser to display by inserting the retrieved data into placeholders in an HTML template.

Django web applications typically group the code that handles each of these steps into separate files:

* + - URLs: A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
    - View: A view is a request handler function, which receives HTTP requests and returns
    - HTTP responses.

Forest Fire: Model in Python that will run when the dataset is uploaded by the front

end user.



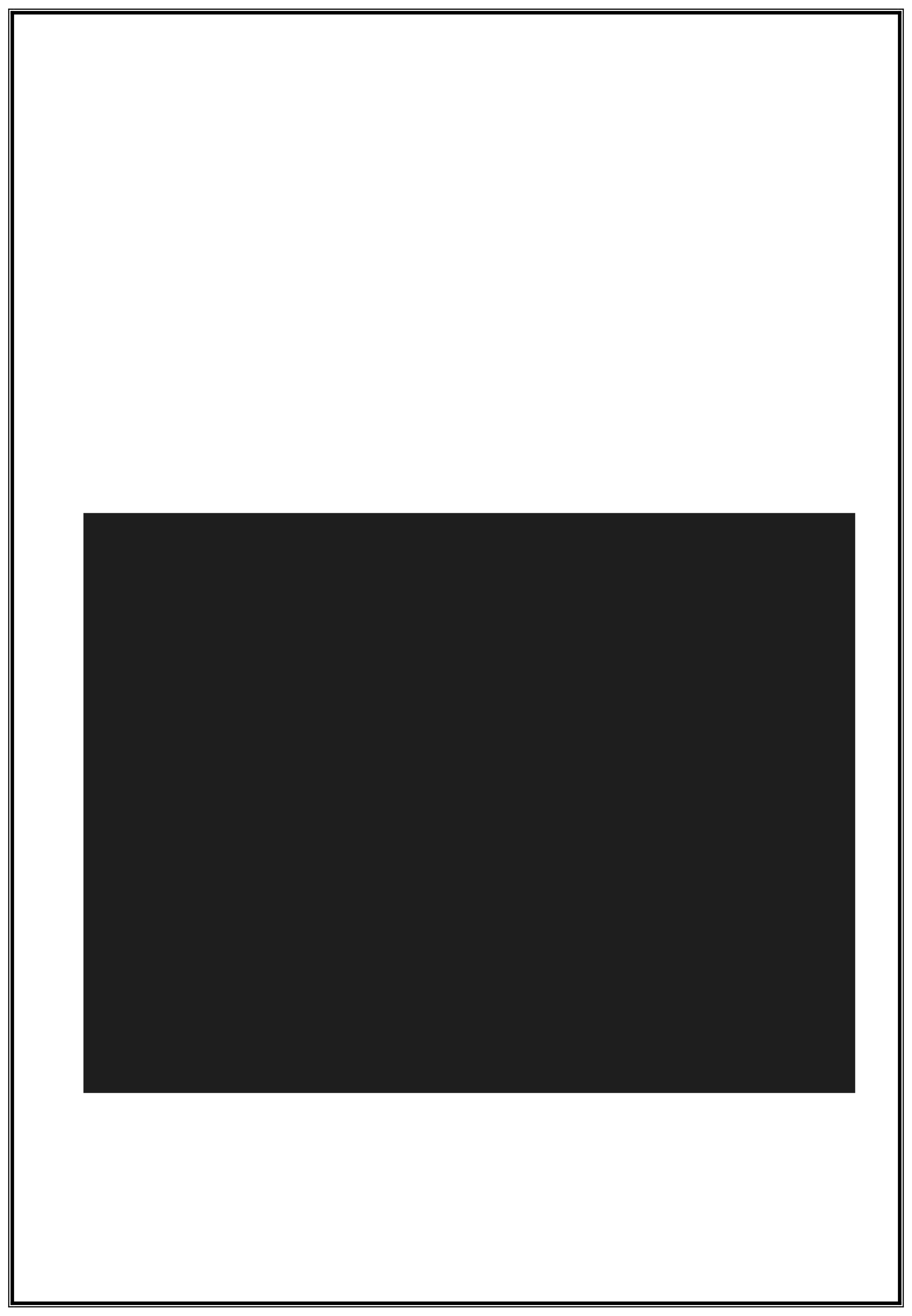
* + - Templates: A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A view

can dynamically create an HTML page using an HTML template, populating it with data from our program. This acts as the client side.

Fig 4.2: Django implementation Framework

#### Dataset

The dataset is collected from Indian Meteorological department (IMD). The four basic attributes considered are: Temperature, Wind, Rainfall and Relative humidity. From these basic parameters, fuel moisture codes like Ffmc (fine fuel moisture code), DMC (duff moisture code), DC (drought code) and fire behaviour indices like ISI (initial spread index), BUI (buildup index) and FWI (fire weather index) are calculated where the derived attributes are a function of



the basic attributes as shown below:

* + - FFMC is a function of Temperature, Relative humidity, Rain and Wind.
    - DMC is a function of Temperature, Relative humidity and Rain
    - DC is a function of Temperature and Rain
    - ISI is a function of ffmc, wind
    - BUI is a function of ffmc, dmc
    - FWI is a function of ISI and BUI Code:

***def main():***

***import sys ffmc0= 85.0***

***dmc0= 6.0 dc0= 15.0 my\_csv\_in = sys.argv[1]***

***with open(my\_csv\_in, 'r') as f\_in: print("opened")***

***next(f\_in)***

***with open('/Users/krsingh/Desktop/datasets/testset3.csv', 'w') as f\_out:***

***h=["Year","Month","Day","FFMC","DMC","DC","ISI","BUI","Temp","RH","Wind","Rain","FWI",***

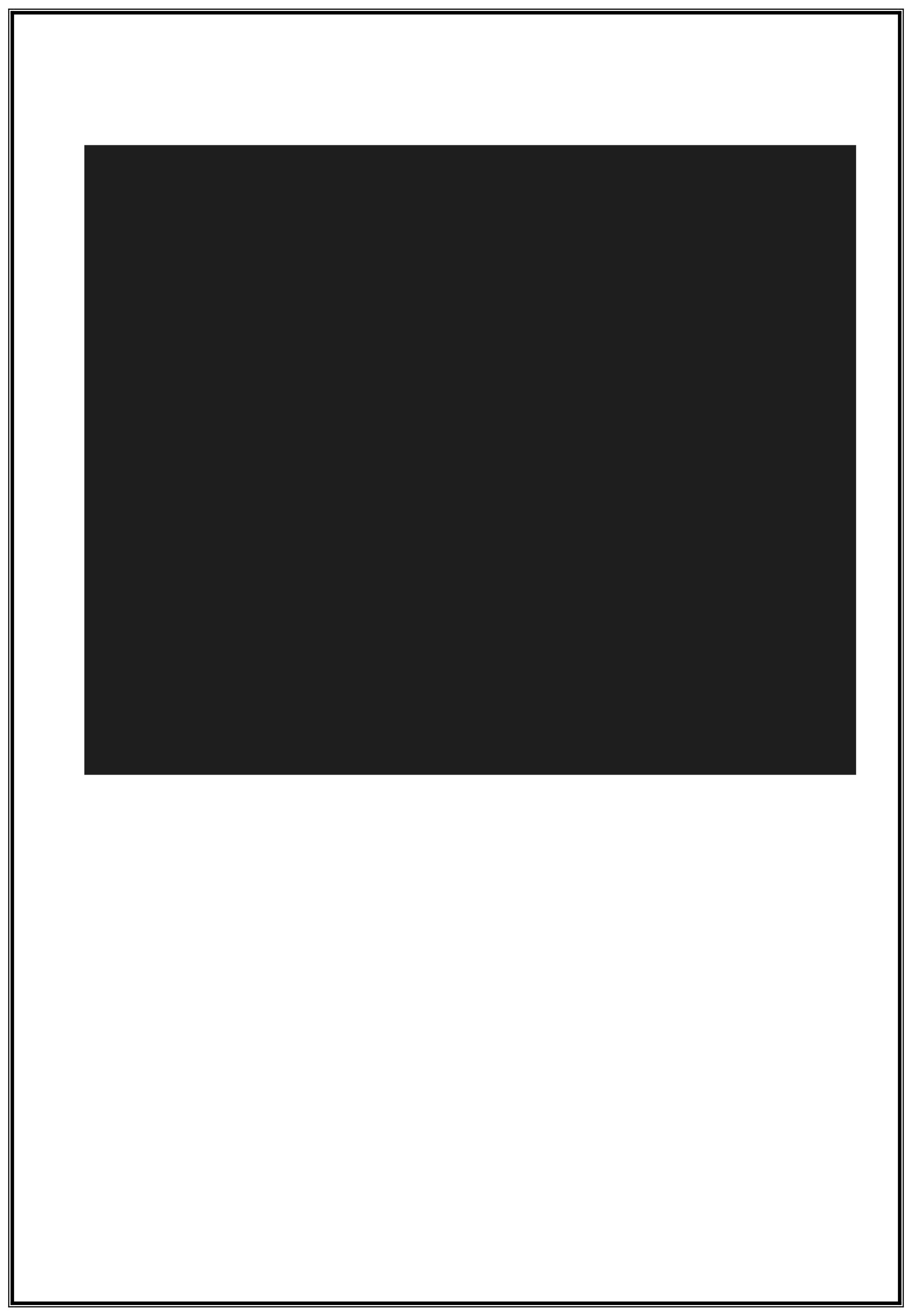
***"Intensity","Fire"]***

***hd=','.j***

***oin(h)***

***f\_out.write(hd) f\_out.write("\r") for line in f\_in:***

***l=line.rstrip().split(',')***

******

***m th=l[1] day=l[2] temp=float(l[3]) rhum=float(l[4]) wind=float(l[5]) prcp=float(l[6]) if rhum> 100.0:***

***rhum***

***=100.0 mth=int(mth)***

***fwisystem= FWICLASS(temp,rhum,wind,prcp) ffmc = fwisystem.FFMCcalc(ffmc0)***

***dmc = fwisystem.DMCcalc(dmc0,mth) dc = fwisystem.DCcalc(dc0,mth) isi = fwisystem.ISIcalc(ffmc)***

***bui = fwisystem.BUIcalc(dmc,dc)***

***l=[str(yr),str(mth),str(day),str(round(ffmc,4)),str(round(dmc,4)),str(round(dc,4)),str***

These factors affect the occurrence of Forest fire in various ways:

* + - Relative Humidity: When the relative humidity is 40 percent, it means that the atmosphere contains 40 percent of the moisture that it could contain at that same

temperature. When the humidity is high, it’s harder for the moisture to evaporate into the air. Consequently, high humidity acts like a damper on a stove. If the humidity is 100 percent or close to it, the fuel will not dry. On the other hand, the lower the relative humidity, the quicker the moisture will evaporate. The lower the relative humidity, the more readily a fire will start and burn; the more vigorously a fire will burn.



* + - Temperature: Air temperature has a direct influence on fire behaviour because of the heat requirements for ignition and continuing the combustion process. Forest fuels receive heat by radiation from the sun. As a result, less heat is required for ignition. The differential heating of the earth’s surface is the driving force behind most of the influences on the atmosphere. The sun emits short-wave energy rays (radiation). When striking a solid object such as trees or grass, it is warmed. Temperature is the most important weather factor affecting fire behaviour. Warm fuels will ignite and burn faster because less heat energy is used to raise the fuels to their ignition temperature. Fuels exposed to sunlight will be warmer than the fuels in shade. They will also be drier. For this reason, fuels not shaded by an overstory will generally be warmer and drier resulting in a more intense fire.
    - Wind: Wind increases the supply of oxygen, which results in the fire burning more

rapidly. It also removes the surface fuel moisture, which increases the drying of the

fuel. Air pressure will push flames, sparks and firebrands into new fuel. By pushing the flames closer to the fuel in front of the fire, the fuel is preheated quicker because

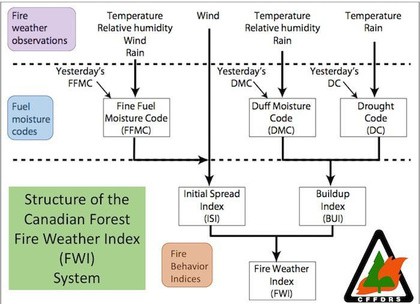
* + - of the increased radiant heat discussed previously. More of the fuel becomes available for combustion since it is dryer and can reach ignition temperature

quicker. Rain: Precipitation (rain or snow) has a direct and immediate effect on fuel moisture and relative humidity. Temperature usually drops as well and the winds become calm. When the atmosphere becomes saturated, precipitation usually occurs if more moisture is added. Precipitation will quickly dampen the surface of fuels to the point that fires cannot ignite and no wildfires will occur. Rain can prevent back-burning, making it harder to build control lines and lead to patchy burnt areas that can flare up again. For rain to extinguish the fires, it will require inches of steady falls over an extended period. As the vegetation greens-up, prescribed burning conditions may deteriorate. If, however, a winter drought occurs and continues into the spring, fires will readily burn

on into the summer because of the larger amount of dead, dry fuel and low fuel

moisture. These fires may be more difficult to control and do more damage due to

burning deeper into the litter and



consuming larger size fuel. During long periods of dry weather, drought, moisture that is toward the centre of larger fuels and deeper in surface litter is able to work its way to the surface and evaporate into the dry atmosphere. As a result, a larger percent of the total fuel becomes available fuel; available to burn.

* + - FFMC: This index classifies the moisture content of litter and other cured fine fuels,

like needles or twigs less than 1 cm in diameter. FFMC is representative of the top litter layer 1-2 cm deep and has a short-term memory, only reflecting weather conditions that have occurred over the past three days.

* + - DMC: This index indicates the moisture content of loosely compacted organic

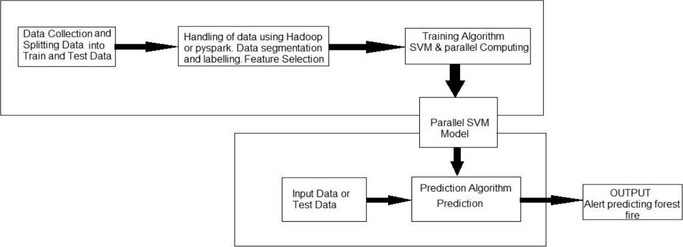
layers with a depth of 5-10cm. DMC fuels have a slower drying rate than FFMC fuels and DMC may be used in predicting the probability of fire ignition by lightning.

* + - DC: reflects the moisture content of compact organic layers, 10-20cm deep. It is an

indicative long-term moisture condition and deep burning fires.

* + - ISI: this index combines FFMC and wind speed being a good indicator for fire spread.
    - BUI: represents the total fuel available for the spreading of fire.

Fig 4.3: Structure of FWI System



#### Flowchart

Fig 4.4: Process Outline

* Data collected will be pre-processed by the model to calculate dependent attributes based on independent attributes and will be added to the dataset. The independent

attributes are namely temperature, rain, rh and wind. The dependent attributes are the forest weather indices like FFMC, DMC, DC, ISI, BUI and FWI. The data is split into test and train data.

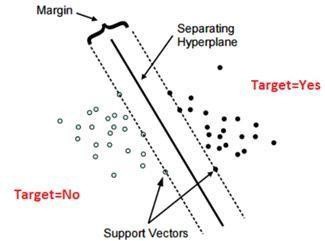
* The training data is hardcoded with the model to train the model while the test data is the data that the user uploads to the predictor to check forest fire alerts for uploaded data. As the data is very large we and the number of support requests created are too huge which will affect the efficiency and accuracy of the system, we will use parallel processing for big data using Spark.

□The python module pyspark is used and rdd data frames are used for data segmentation and labeling. This processed data is then fed to the trained model to predict forest fires

with the uploaded test dataset.

* The output of this model is a table of alerts generated for the uploaded data set. We also have an alerting system that can be used to send alerts to respective departments in

areas of data uploaded.

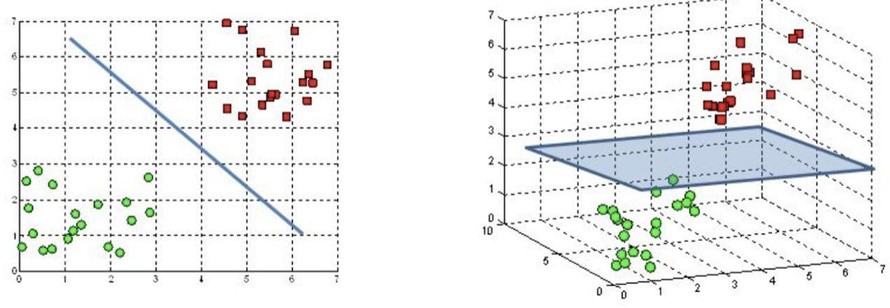


## CHAPTER 5 IMPLEMENTATION

#### Support Vector Machine

Support vector machine is a supervised learning algorithm which can be used for classification and regression. In SVM, the data is represented in an n-dimensional space where it can predict whether a new training example falls into the same category or a different category. The main aim of SVM is to find a hyperplane in the n- dimensional space that can clearly classify the data points. There are several potential hyperplanes which could be chosen to distinguish the two classes of data points. But the ideal hyperplane is the one that maximizes the margin i.e. the maximum distance between data points of both classes as shown in Fig 5.1. Hyperplanes are boundaries for decision making which help to distinguish data points. Data points which fall on either side of the hyperplane can be attributed to various classes. The dimension of the hyperplane depends upon the number of features.

Fig 5.1: Support Vector Machine



As shown in Fig 5.1, Support vectors are the data points that are closer to the hyperplane. They influence the position and orientation of the hyperplane. These support vectors play a major role in classifying the data points to different classes.

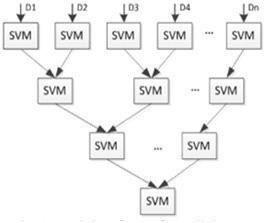
Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the

dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two- dimensional plane like in Fig 5.2. It becomes difficult to imagine when the number of features exceeds 3.

Fig 5.2: Hyperplanes in 2D and 3D

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the

margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.



A major disadvantage of SVM is that it does not perform well when the training set is large in size as the storage and compute requirements increase with increase in the training vectors. Hence, Parallel SVM can be used in order to reduce the computational time and improve the performance.

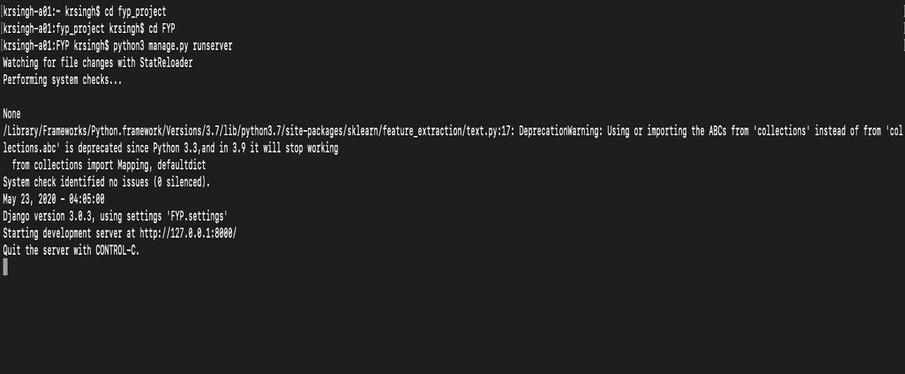
#### Parallel Support Vector Machine

The parallel SVM is based on the Cascade SVM model. The SVM is trained through partial SVMs. Each subSVM is used as a filter which takes us towards the global optimum. Using the Parallel SVM model, large scales of data can be divided into smaller, independent sub data. The former subSVMs support vectors are used as the input to the later subSVMs as shown in Fig 5.3.

The support vectors of two SVMs are combined into one set and sent as input to a new

SVM. This process is continued until a single set of vectors is left which is the global optimum. The size of the training set is reduced in each subsequent stage. Each subSVM is trained with a SVM model like libSVM, lightSVM etc. Most efficient SVM model is libSVM.

Fig 5.3: Parallel Support Vector Machine (Figure taken from paper)



Explicit parallelization approaches parallelize the computation within each iteration as well as parallelizing kernel computations. Parallelize reduces the number of working sets for the model thereby speeding up the testing process and making it more efficient in predictions. Parallelizing the SVM is a method where the huge dataset of 30 years data will be divided into smaller datasets and is fed into multiple SVMs simultaneously that outputs a set of support vectors along with the another set of support vectors are fed into another SVM. This is done until we get one set of support vectors that have maximum margin from the hyperplane which will make this model more reliable and will predict with better accuracy and efficiency. To achieve this we will use spark for big data computing and rdd datasets for parallel computing in our program. Forest fires could be due to multiple reasons; it is very difficult to make a prediction system. There could be multiple reasons for forest fires, what we want to achieve using our model is a reliable system that could predict correctly based on weather conditions if fire occurs and also the intensity of fire.

#### Front-end and Back-end implementation details

The python manage.py run server command is executed in the Django project directory as shown in Fig 5.4 and the page appears in the localhost URL.

Fig 5.4: Command Prompt

#### Sending request to the right Url (url.py)

A URL mapper is typically stored in a file named urls.py. In the example below, the mapper (urlpatterns) defines a list of mappings between routes (specific URL patterns) and corresponding view functions. If an HTTP Request is received that has a URL matching a specified pattern then the associated view function will be called and passed the request.

Code

***urlpatterns = [***

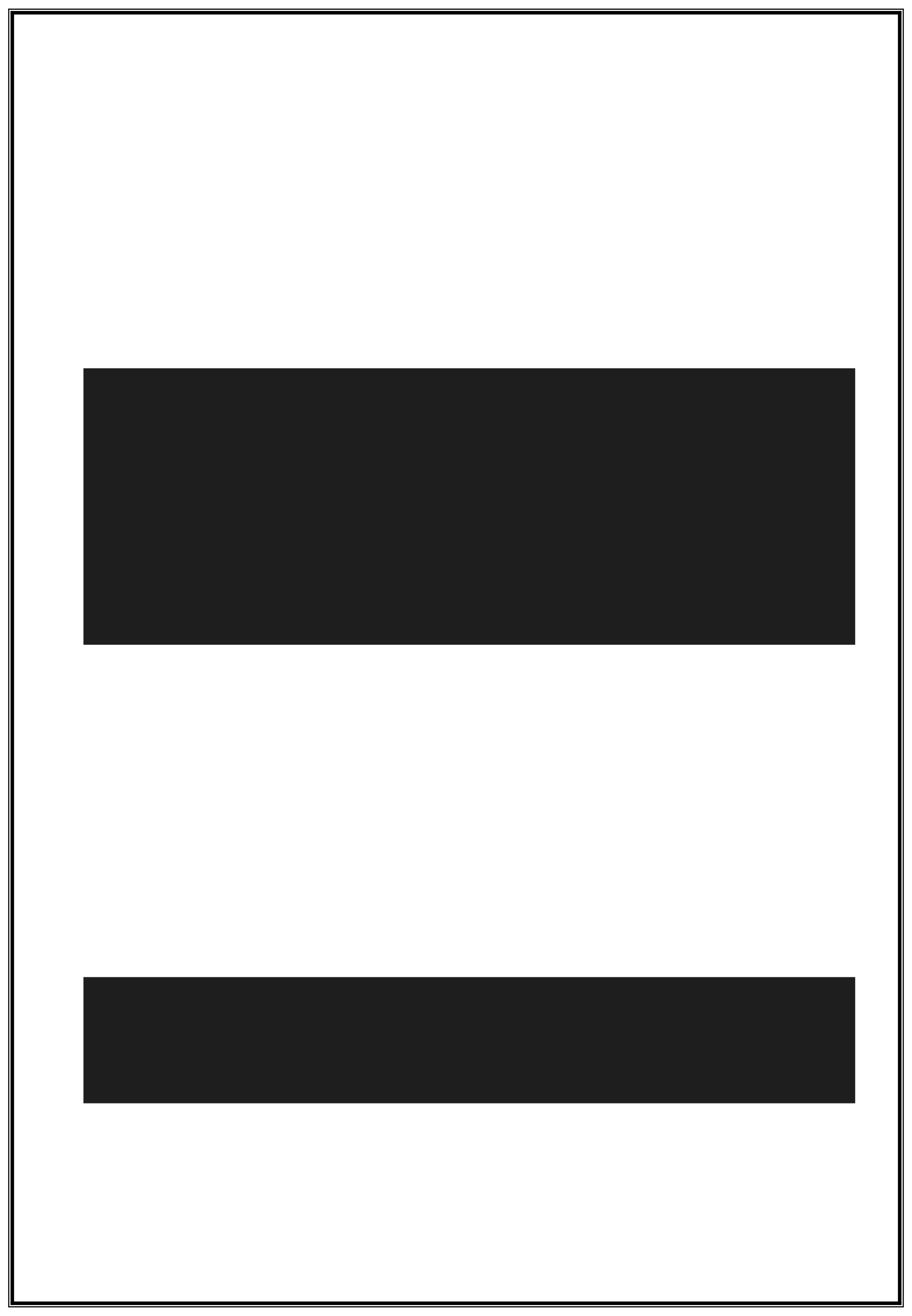
***re\_path(r'^admin/', admin.site.urls), re\_path(r'^$', views.button), re\_path(r'^output',views.output, name='script'), re\_path(r'^external',views.external), re\_path(r'^alert/',views.alert, name="alert"),***

***re\_path(r'^prediction/',views.prediction, name="prediction"), re\_path(r'^validation/',views.validation,***

***name="validation"), name="predictor")***

***]***

***re\_path(r'^predictor/',views.predictor,***

******

The first argument to both methods is a route (pattern) that will be matched. The path() method uses angle brackets to define parts of a URL that will be captured and passed through to the view function as named arguments.

#### Handling the requests (view.py)

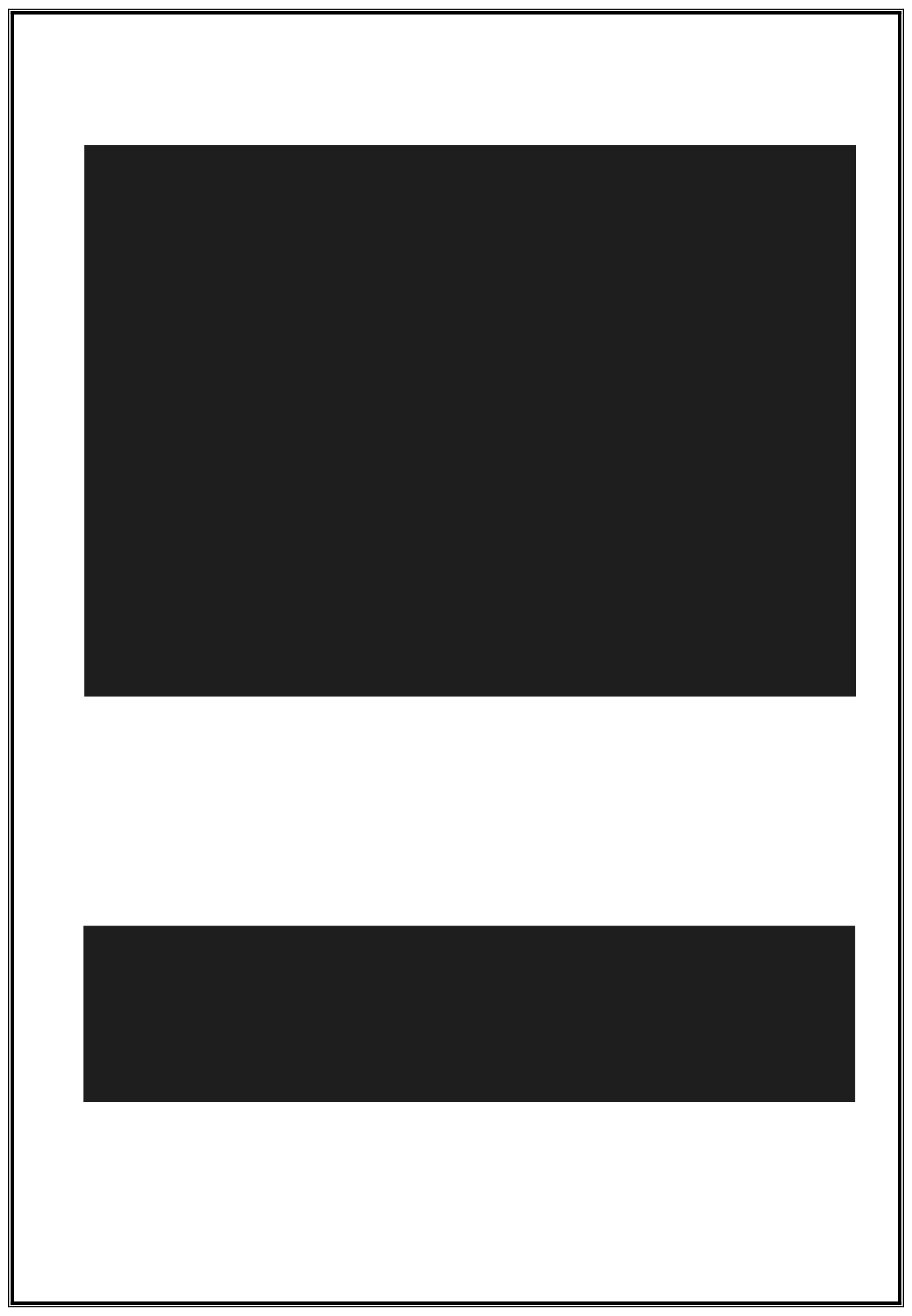
Views are the heart of the web application, receiving HTTP requests from web clients and returning HTTP responses. In between, they marshall the other resources of the framework to access databases, render templates, etc.

Code

***from subprocess import run, PIPE ,Popen, call,check\_output def button(request):***

***return render(request,'index.html') def output(request):***

***return render(request,'index.html')***

******

***def external(request):***

***file=request.FILES['myfile'] f="/Users/krsingh/Desktop/datasets/"+str(file)***

***output=Popen(['python3','/Users/krsingh/fyp\_project/FYP/FYP/forestfire.py',f],stdout=P IPE,universal\_newlines=True)***

***l=[]***

***for line in output.stdout.readlines(): l.append(line)***

***print(len(l))***

***return render(request, 'index.html',***

***{'Accuracy\_LSVM':l[1],'Precision\_LSVM':l[2],'Recall\_LSVM':l[3],'Time\_LSVM':l[4],'Accur***

***acy\_PSVM':l[5],'Precision\_PSVM':l[6],'Recall\_PSVM':l[7],'Time\_PSVM':l[8]})***

***def alert(request):***

***return render(request,'pg2.html') def prediction(request):***

***return render(request,"prediction.html") def validation(request):***

***return render(request,"validation.html") def predictor(request):***

***return render(request,"index.html")***

#### Defining model(foresfire.py)

The definition of the model is independent of the underlying database and in our model it runs the predictor algorithm when the test dataset gets uploaded on the predictor page of UI. The view external() gets called when the file gets uploaded and the model predicts the results for the dataset uploaded.

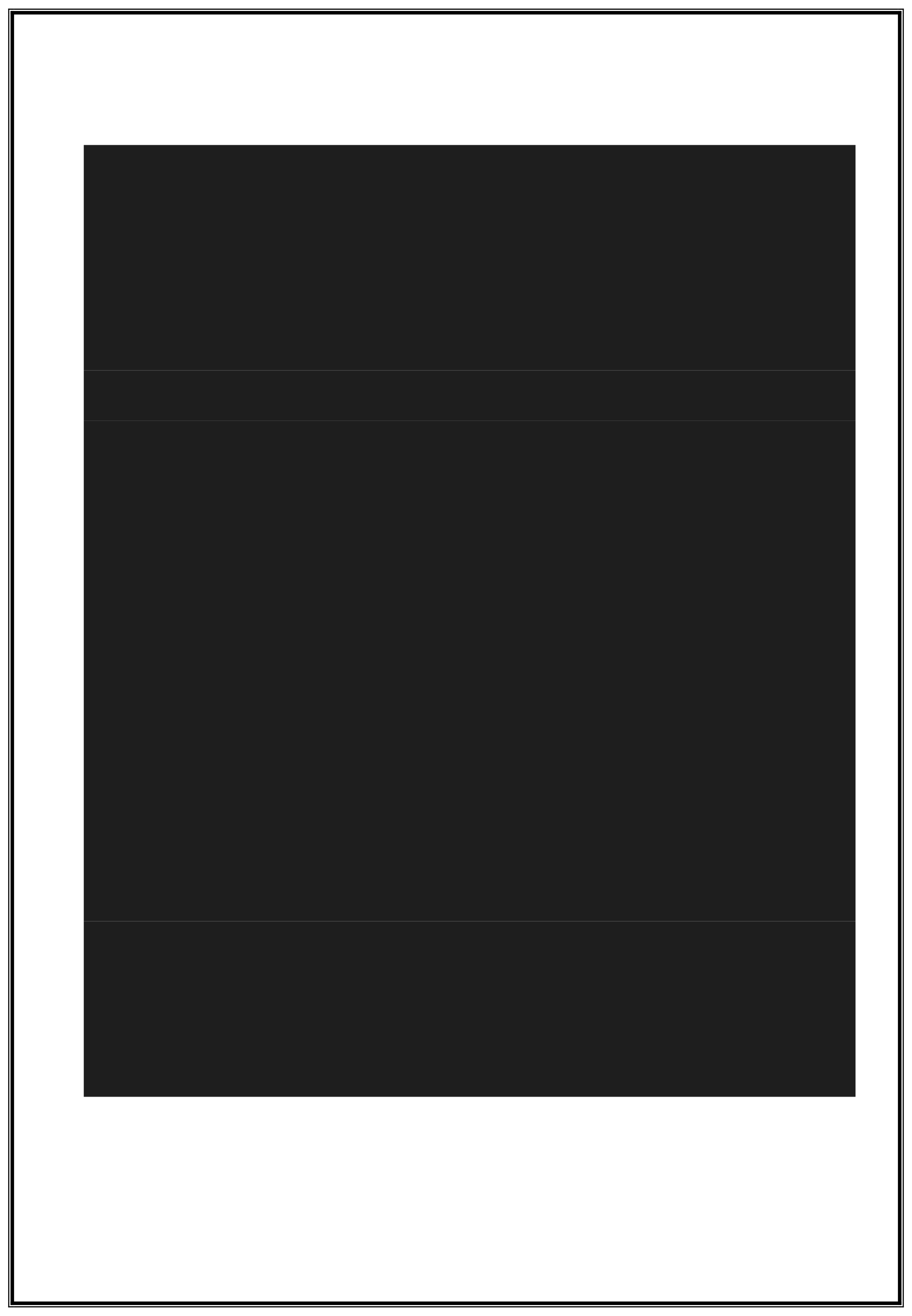
Code

import pyspark

from pyspark.sql import SparkSession from pyspark.ml import Pipeline

from pyspark.ml.feature import StringIndexer from pyspark.ml.feature import VectorAssembler

from pyspark.ml.evaluation import



#import Spark and MLlib packages

from pyspark import SparkContext, SparkConf

from pyspark.mllib.regression import LabeledPoint

from pyspark.mllib.classification import SVMWithSGD, SVMModel

from pyspark.mllib.classification import LogisticRegressionWithLBFGS #import data analysis packages

import numpy as np import pandas as pd import sklearn

from pandas import Series, DataFrame from sklearn import svm

from sklearn.svm import SVC

from sklearn.cross\_validation import train\_test\_split from sklearn import metrics

from numpy import array

from timeit import default\_timer as timer #import data visualization packages import matplotlib.pyplot as plt

import seaborn as sns sns.set\_style('whitegri d') import random

dataframets= pandas.read\_csv(r"/Users/krsingh/Desktop/datasets/testset3.csv") dataframetr =

pandas.read\_csv(r"/Users/krsingh/Desktop/datasets/newtraintdata7.csv") featuredatatr=dataframetr.iloc[:, :14] targetvaluestr=dataframetr.iloc[:,14:] featuredatats=dataframets.iloc[:, :14] targetvaluests=dataframets.iloc[:,14:]

x\_train=featuredatat r x\_test=featuredatats y\_train=targetvalues tr y\_test=targetvaluest s

# SVM regularizaion parameter C = 1.0

svc = svm.SVC(kernel = 'linear', C=C).fit(x\_train, y\_train) start = timer()

predicted = svc.predict(x\_test) end

=timer()



predicted = svc.predict(x\_test) expected = y\_test

from pyspark.ml import Pipeline from pyspark.ml.feature import

StringIndexer from pyspark.ml.feature import VectorAssembler

from pyspark.ml.evaluation import MulticlassClassificationEvaluator from pyspark.ml.feature import QuantileDiscretizer

from pyspark.ml.classification import LinearSVC

from pyspark.ml.evaluation import BinaryClassificationEvaluator

spark = SparkSession \

.builder \

.appName("Spark ML example on data ") \

.getOrCreate()

datatrain= "/Users/krsingh/Desktop/datasets/newtraintdata7.csv" dftr = spark.read.csv(datatrain,header = 'True',inferSchema='True') datatest= "/Users/krsingh/Desktop/datasets/testset3.csv"

dfts = spark.read.csv(datatest,header = 'True',inferSchema='True') featuretr = VectorAssembler(

inputCols=[x for x in dftr.columns], outputCol='features')

feature\_vector\_tr= featuretr.transform(dftr) featurets = VectorAssembler(

inputCols=[x for x in dfts.columns], outputCol='features')

feature\_vector\_ts= featurets.transform(dfts) trainingData=feature\_vector\_tr testData=feature\_vector\_ts

from pyspark.ml.classification import LinearSVC

from pyspark.ml.evaluation import BinaryClassificationEvaluator svm = LinearSVC(labelCol="Fire", featuresCol="features") svm\_model = svm.fit(trainingData)

#### Rendering data using templates (index.html)

Templates are often used to create HTML and code snippets show what the HTML template called by the render() function in the previous section might look like.



Code

{% load static %} <!DOCTYPE html> <html> <head>

<title>Forest\_fire\_predictor</title> </head> <link rel="stylesheet" href="{% static 'FYP.css' %}">

<body style="background-image: url({% static 'wall.jpg' %}); background-size:100%; background-repeat: y-repeat;">

<header>

<div class="wrapper" id="nav">

<div><h1 style="color :white; background-color: black; padding-top: 10px; padding- bottom: 10px; padding-left:15px;text-align:left">Forest Fire Prediction System</h1>

<ul class="nav-area">

<li><a href="{% url 'predictor'

%}">PREDICTOR</a></li>&nbsp;

<li><a href="{% url 'alert' %}">ALERTS</a></li>&nbsp;

<li><a href="{% url 'prediction'

%}">INTENSITY</a></li>&nbsp;

<li><a href="{% url 'validation'

%}">VALIDATION</a></li>&nbsp;

</ul><div>

<div>

</header>

<div class="filechoose" style="background: black; opacity: 0.8; box-shadow: 0 4px 8px

0 rgba(0,0,0,0.2); border-radius: 30px; margin-left: 500px; width: 450px; height: 150px;">

<form action='/external/' enctype="multipart/form-data" method="post" enctype="multipart/form-data" style="color: white; margin-top: 50px; margin-left: 50px; padding-top: 30px;">

<label for="myfile" style="font-size: 20px; ">Select a file: &nbsp; </label>

{% csrf\_token %}{{ form.as\_p }}

<input type="file" id="myfile" name="myfile" ><br><br>

{{Accuracy}}

<input type="submit" style="border: 1px solid black; border-radius: 10px; padding: 10px 20px 10px 20px; margin-left: 140px; margin-top: 10px; font-weight:



</div>

<div style="margin-top: 50px; padding-top: 10px; border: 2px 2px 2px 2px; border- style: solid;border-color:#6B8E23; background-color: black;">

<h2 style="font-size: 40px;margin-left: 650px;color: white; ">ALGORITHMS</h2>

</div>

<div class="row">

<div class="column" onclick="openTab('b2');" style="background: black; opacity: 0.8; box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2); border-radius: 30px; height: 315px;">

<img src="{% static 'svm.gif' %}" style="border-radius: 25px; width: 60%; height: 75%; margin-top: -5px;">

<h2 style="color: white; font-size: 25px; font-style: italic; opacity: 1;"><b>LINEAR SVM</b></h2> <br>

</div>

<div class="column" onclick="openTab('b3');" style="background: black; opacity: 0.8; box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2); border-radius: 30px;height: 317px;">

<img src="{% static 'ksvm.gif' %}" style="border-radius: 25px; width: 50%; height: 70%; margin-top: 5px;">

<h2 style="color: white; font-size: 25px; font-style: italic; opacity: 1;"><b>PARALLEL SVM</b></h2> <br>

</div>

</div>

<div id="b2" class="containerTab" style="display:none;background:black; opacity: 0.8;">

<span onclick="this.parentElement.style.display='none'" class="closebtn" style="cursor: pointer;">&times;</span>

<div style="margin-top: 10px;"><center style="color: white; font-size: 15px; font-style: italic; ">Accuracy of Linear SVM: {{Accuracy\_LSVM}} <br>

Precision of Linear SVM:

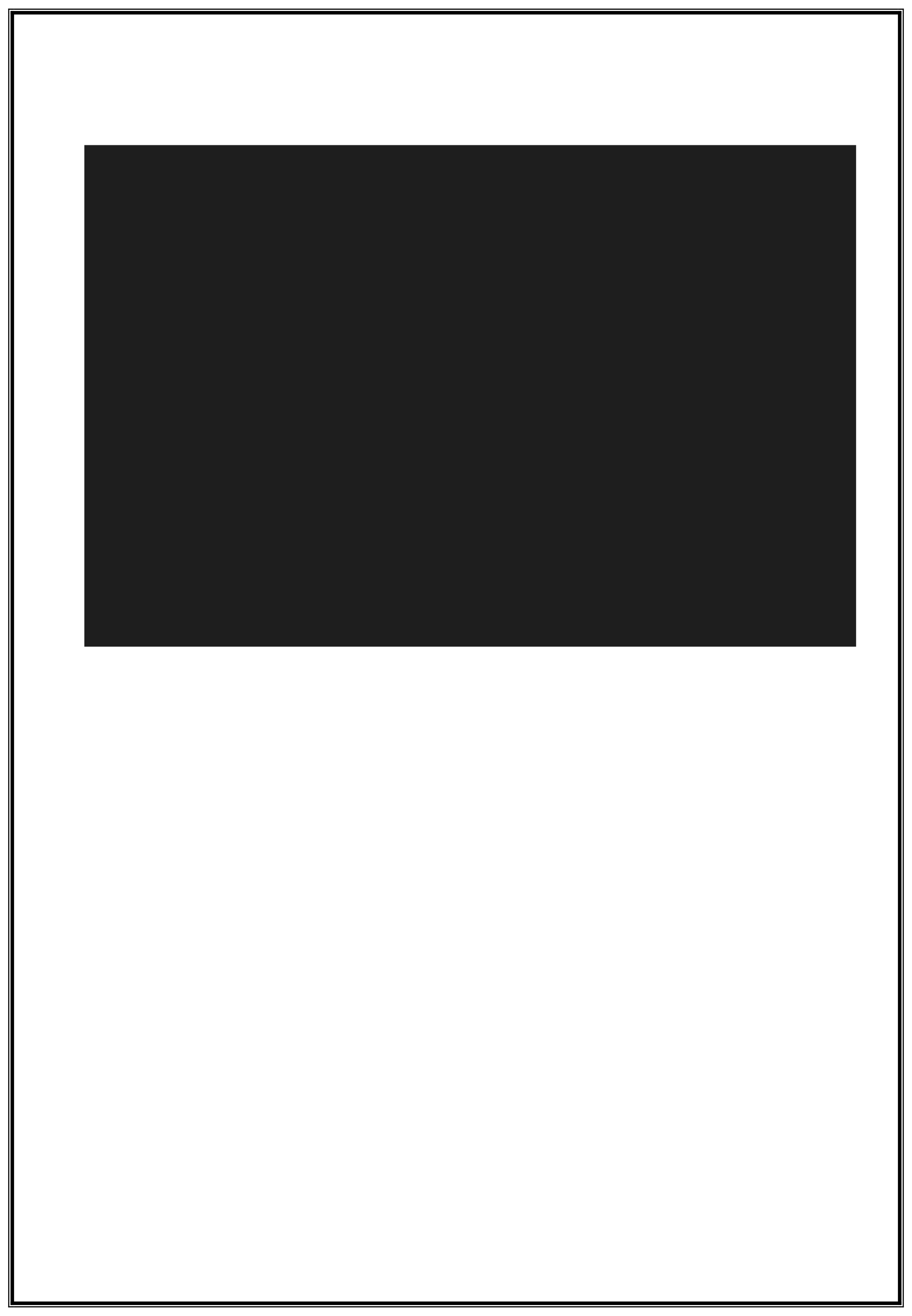
{{Precision\_LSVM}} <br> Recall of Linear SVM:

{{Recall\_LSVM}} <br>

Execution time taken by Linear SVM: {{Time\_LSVM}} <br>

</center>

</div>



<div id="b3" class="containerTab" style="display:none;background:black; opacity: 0.8;">

<span onclick="this.parentElement.style.display='none'" class="closebtn" style="cursor: pointer;">&times;</span>

<div style="margin-top: 10px;"><center style="color: white; font-size: 15px; font-style: italic; "> Accuracy of Parallel SVM: {{Accuracy\_PSVM}}<br>

Precision of Parallel SVM:

{{Precision\_PSVM}} <br> Recall of Parallel SVM:

{{Recall\_PSVM}} <br>

Execution time taken by Parallel SVM: {{Time\_PSVM}} <br>

</center></div>

</div>

<script> function openTab(tabName) {

var i, x;

x =

document.getElementsByClassName("containerTab"); for (i = 0; i < x.length; i++) {

x[i].style.display = "none";

}

The home page of our forest fire prediction system is shown in Fig 5.3 which contains four tabs: Predictor, Alerts, Intensity and Validation. The dataset can be uploaded and the results of two algorithms can be seen.

Fig 5.5: User Interface

#### Triggered Alerts

The model has an automated alert system that gets triggered whenever the predictions are made. The alerts are in tabular form and give the month day details for when the fire is predicted and also the intensity of the fire based on the thresholds applied on the weather indices.

Code

***from email.mime.text import MIMEText***

***from email.mime.multipart import MIMEMultipart import smtplib***

***fromaddr = "ForestfireAlerts.gmail.com" toaddr = [“whom so ever specified”***

***for i in range(len(toaddr)): html =***

***open("/Users/krsingh/fyp\_project/FYP/FYP/template/pg2.html") msg =***

***MIMEMultipart()***

***msg['From'] = fromaddr msg['To'] = toaddr[i]***

***msg['Subject'] = "Fire Alerts Report" part2 = MIMEText(html.read(), 'html') msg.attach(part2)***

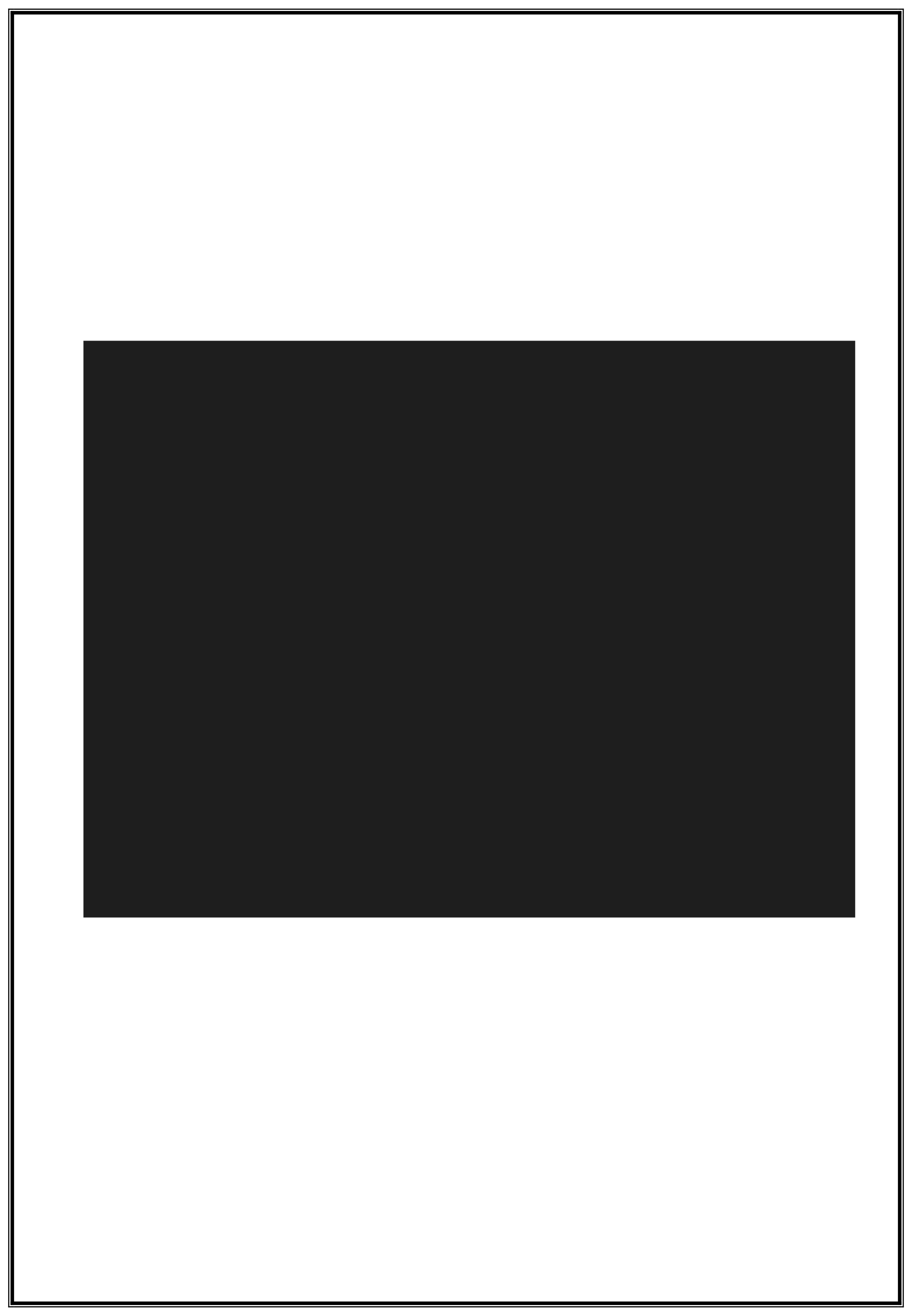
***debug***

***= False if debug:***

***()) else:***

***print(msg.as\_string***

***server = smtplib.SMTP('smtp.gmail.com',587)***

***server.starttls() server.login("***[***forestfire.alerts@gmail.com***](mailto:forestfire.alerts@gmail.com)***", "\*\*\*\*\*\*\*\*\*\*")***



## CHAPTER 6

RESULTS AND DISCUSSION

The results are divided into 4 tabs: Predictor, Alerts, Intensity and Validation.

#### Predictor Tab:

A comparison of Linear SVM and Parallel SVM is shown.

Both algorithms are trained to predict forest fires. The accuracy of prediction, precision, and recall values are displayed in the results.

The estimated time required for the algorithms to run is also displayed.

The results show a slight variation in the displayed values, thus proving the higher efficiency of parallel SVM compared to linear SVM.

The accuracy, precision, and recall values for parallel SVM is higher than SVM.

Fig 6.1 (a) shows the results of Linear SVM and Fig 6.1 (b) shows the results of Parallel SVM.

Fig 6.1 (a): Linear SVM Results

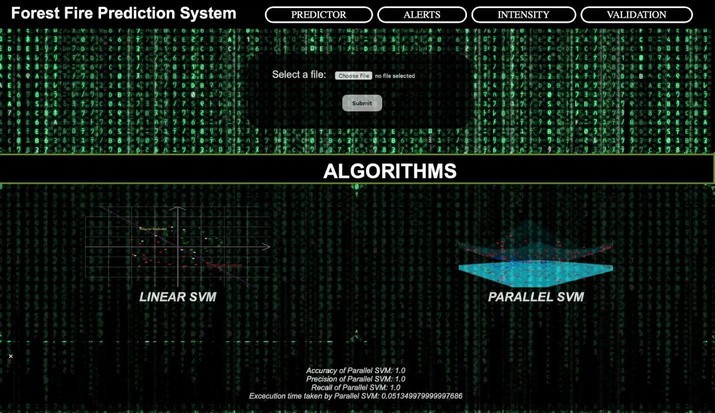


Fig 6.1 (b): Parallel SVM Results

#### Alerts tab:

* + In the second tab, a table of the data values is displayed.
  + These data values correspond to the entries where the occurrence of fire was predicted as 1.
  + A column displaying the intensity of the fire is also displayed.
  + Fig 6.2 shows the Alert tab showing the predicted fire days.

#### Intensity Tab:

* + A graph showing month wise occurrence of fire is plotted as shown in Fig 6.3.

#### Validation Tab:

* + A graph showing the predicted fire graph is plotted as shown in Fig 6.4 (a).
  + A graph showing the validation of our model with predicted and actual is plotted as shown in Fig 6.4 (b).

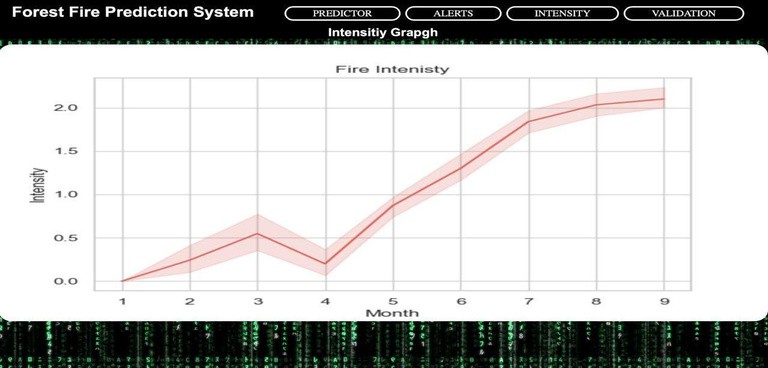
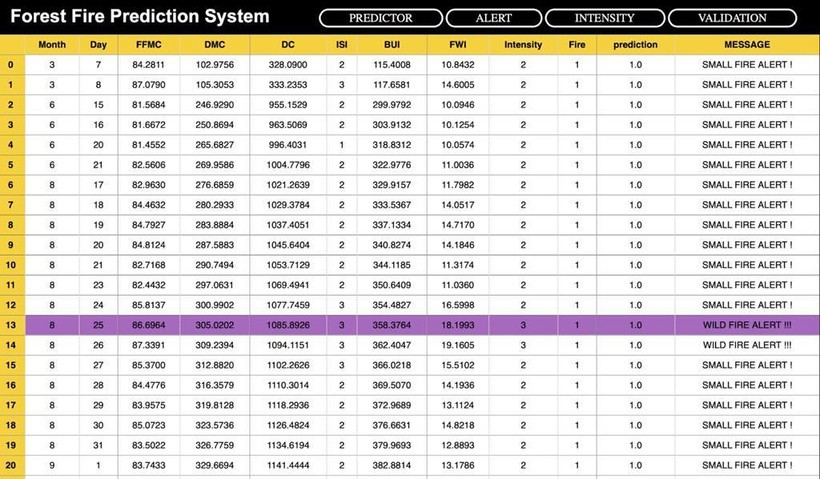


Fig 6.2: Predicted Fire Days

Fig 6.3: Fire Intensity vs. Month



Fig 6.4 (a) and (b): Validation Predictions with Actual Fire Possibilities

* Finally, an automated alert system sends the generated forest fire report via Email as shown in Fig 6.5.

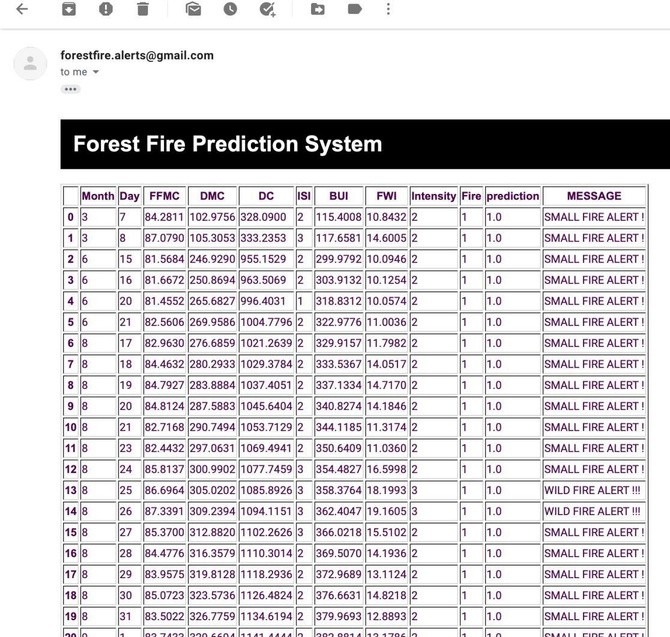


Fig 6.5: Forest Fire Prediction Report sent as Alert



## CHAPTER 7

TESTING

#### Performance of the model

The performance of the two algorithms, Linear SVM and Parallel SVM are tested on the basis of four parameters which are Accuracy, Precision, Recall and Execution time.

#### Accuracy:

Accuracy is the number of correctly predicted data points out of all the data points. More formally, it gives us an idea about how well our model predicted the occurrence of forest fires.

Accuracy = (True Positive + True Negative)/ (True Positive + False Positive + True Negative + False Negative)

#### Precision:

Precision means the percentage of your results which are relevant i.e. ratio of all correct positive classifications to the total no. of positive classifications.

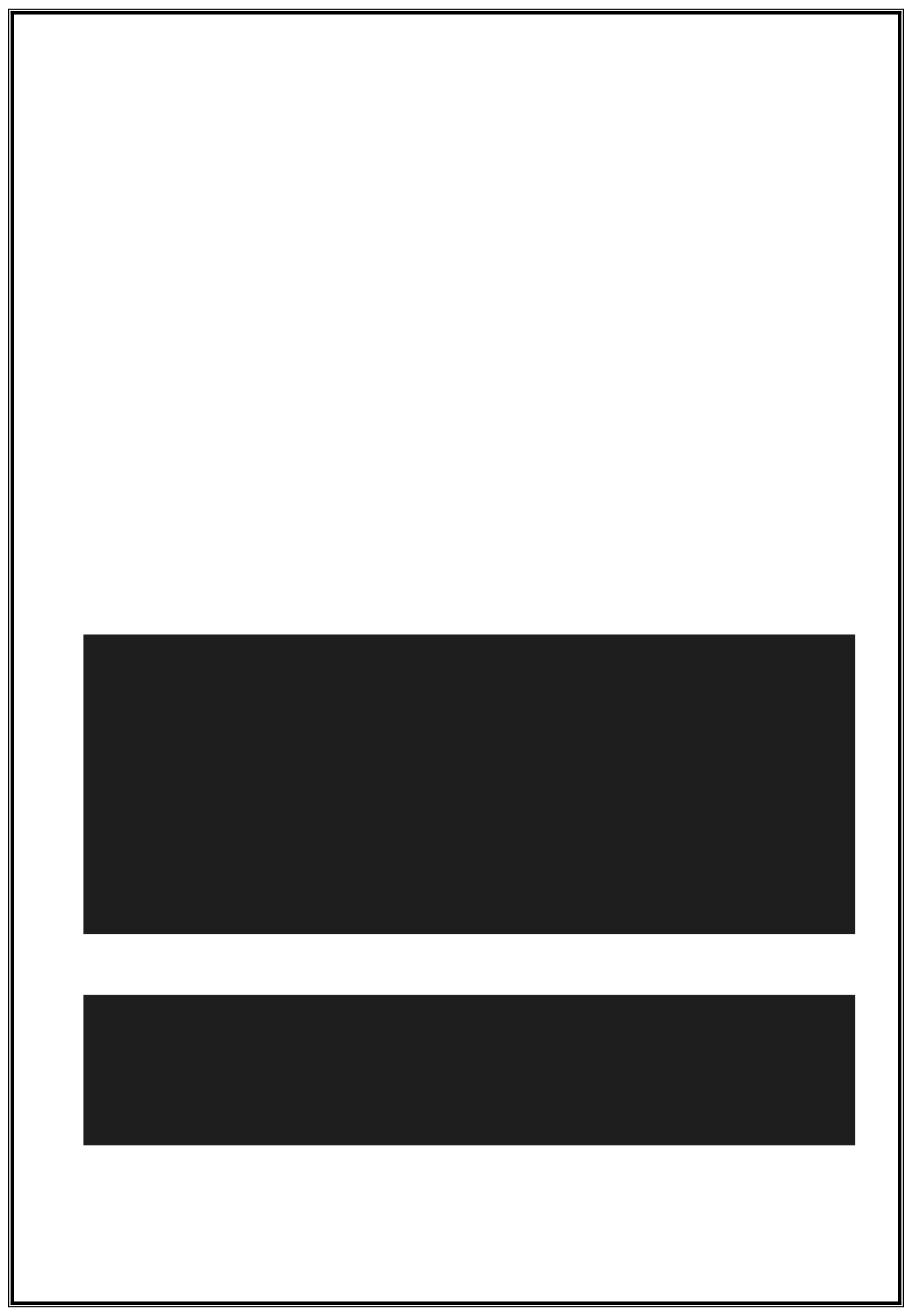
Precision = True Positive/ (True Positive + False Positive)

For our model, it tells us when the model predicted TRUE i.e., the forest fire will happen and how often it was right.

#### Recall:

Recall refers to the percentage of total relevant results correctly classified i.e., this is the case when in reality, fire has actually happened and how often did our model get it right.

Recall = True Positive/ (True Positive + False Negative)



#### Execution time:

The time taken by the model to make predictions is considered as the execution time for the model. We use the timer () function to calculate the time taken to predict.

#### Observation:

The performance of the system was tested for different datasets using LSVM and PSVM. It was observed that PSVM was more efficient and had better accuracy compared to LSVM. The results were stable for PSVM compared to LSVM. Therefore we conclude that the predictions made by PSVM are more reliable to predict forest fires. Linear SVM has lower accuracy when the dataset is too huge and becomes less reliable while parallel SVM filters the most optimal support vectors and gives a more reliable model to predict forest fires. The alerts generated are based on the Parallel SVM model.

#### Code for LSVM Performance

***from sklearn.metrics import classification\_report, confusion\_matrix***

***lsvmaccuracy = metrics.accuracy\_score(expected,predicted)***

***lsvmprecision=metrics.precision\_score(expected, predicted) lsvmrecall=metrics.recall\_score(expected, predicted)***

***print(lsvmaccuracy) print(lsvmprecision)***

***print(lsvmrecall) print(end-start)***

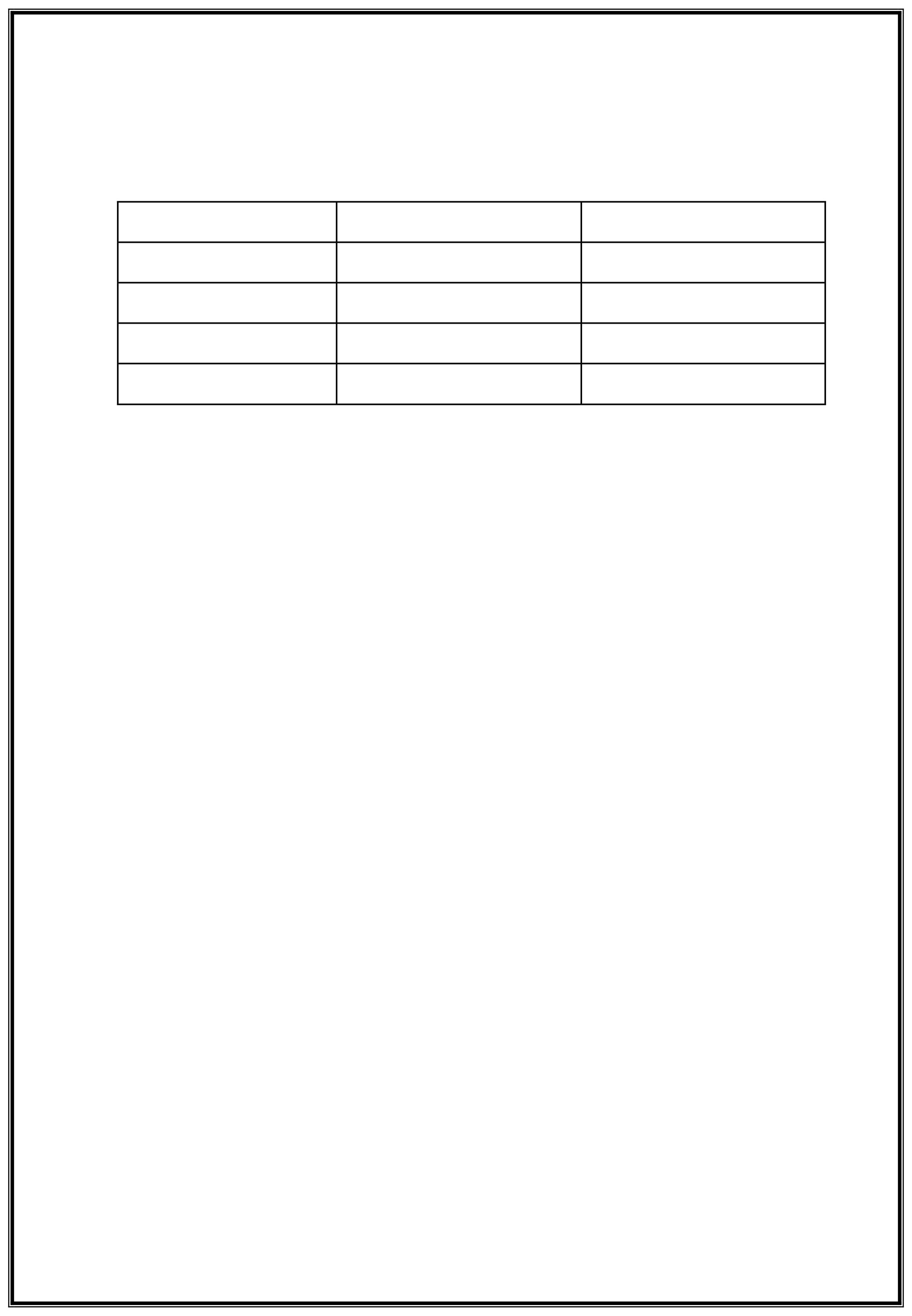
#### Code for PSVM Performance

***from sklearn.metrics import classification\_report, confusion\_matrix***

***print(metrics.precision\_score(l, y\_pred))***

***print(metrics.recall\_score(l, y\_pred))***

***print(end1-start1)***

******

#### Comparison between Linear SVM and Parallel SVM

|  |  |  |
| --- | --- | --- |
| Parameters | Linear SVM | Parallel SVM |
| Accuracy | 0.94 | 1.0 |
| Recall | 0.93 | 1.0 |
| Precision | 1.0 | 1.0 |
| Execution time(in s) | 0.0006 | 0.051 |

Table 7.1: Comparison between Linear SVM and Parallel SVM



## CHAPTER 8

CONCLUSION AND FUTURE SCOPE

#### Conclusion

Forest fires may happen because of many causes, accurate predictions are our target. Using weather data and analysing the forest weather indices for training our model is one of the best methods. Our work shows that the easiest way to predict forest fires is by using Support Vector Machines. For small fires, which ultimately trigger bigger fires, there are more reliable tests. But support for large data sets on Support vector machines would have several support vectors to minimize accuracy. We need accurate and reliable data. It is critical. Via parallel computation, we can boost the support vectors. We conclude that by modifying the algorithm and using parallel calculation, performance and calculation time are improved. There is however, a higher memory requirement and an increase in computational time. For this, we make use of the Apache Spark framework.

For implementing our model we have used the django framework for our user interface

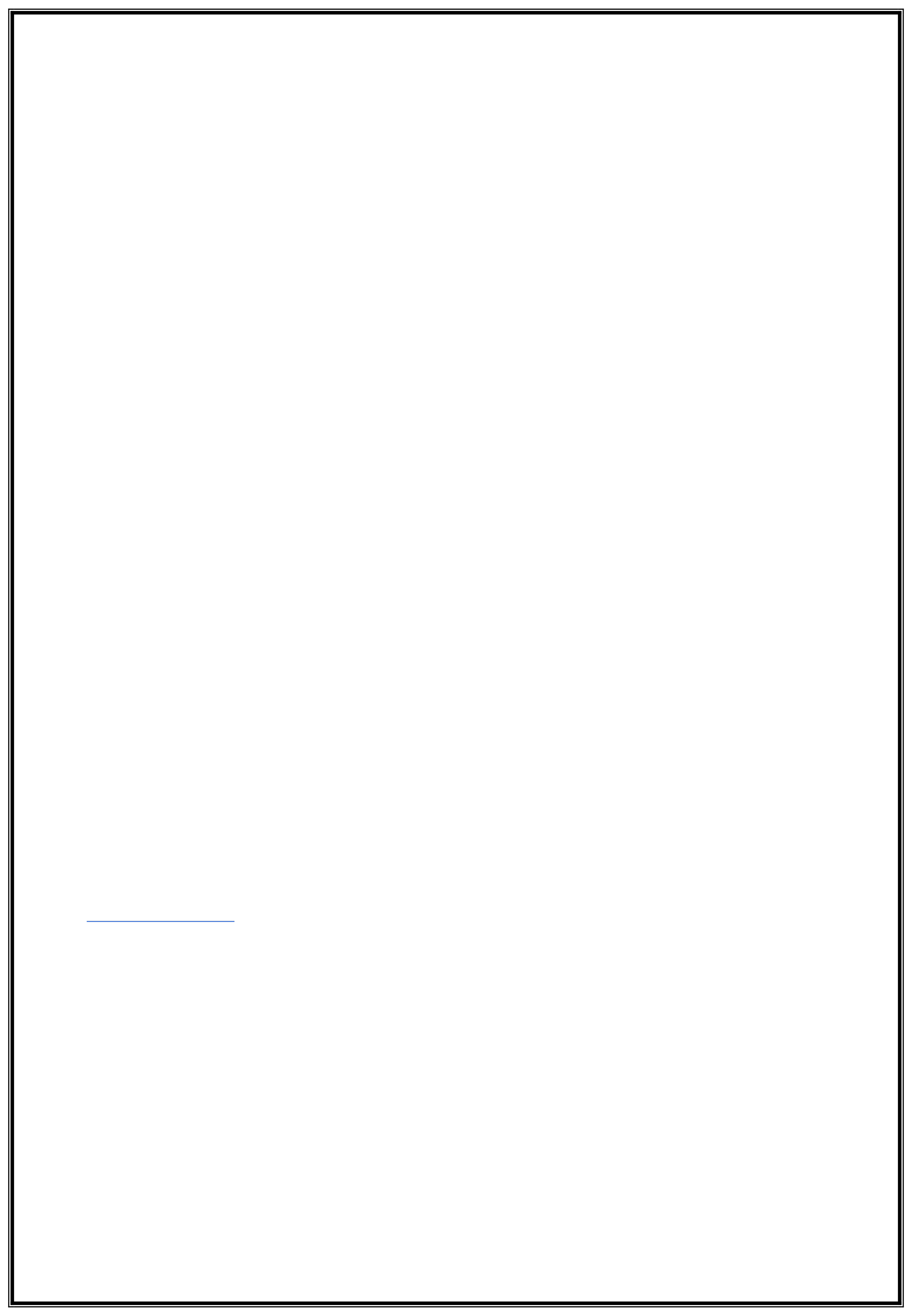
and integrated the prediction algorithm as the django model. This prediction model makes use of pyspark which is the Python API written in python to support Apache Spark. Apache Spark is a distributed framework that can handle Big Data analysis. Spark is basically a computational engine that works with huge sets of data by processing them in parallel and batch systems.

Using these modules we have implemented parallel SVM that uses the weather data and uses parallel computing to predict forest fires. This way we want to make our model more efficient and reliable. We have an additional alerting system that can be used to alert a specific department by just uploading the weather data of a particular station. This way we can help the managing forest fires before it destroys the whole forest. This makes prevention easier by predicting forest fires easily.



#### Future scope

Forest fires cause deforestation and land burning. Forest fires are considered socially and economically unwelcome as they burn large amounts of land and may require some time to recover it. Thus it is important to prevent forest fires. Our model is used to predict forest fires. It has an accuracy of 0.99. Our model can be improved by making it live data processing, on site predictions, making it sensor based etc. In the future we hope to improve the speed and accuracy of the working model.



## REFERENCES

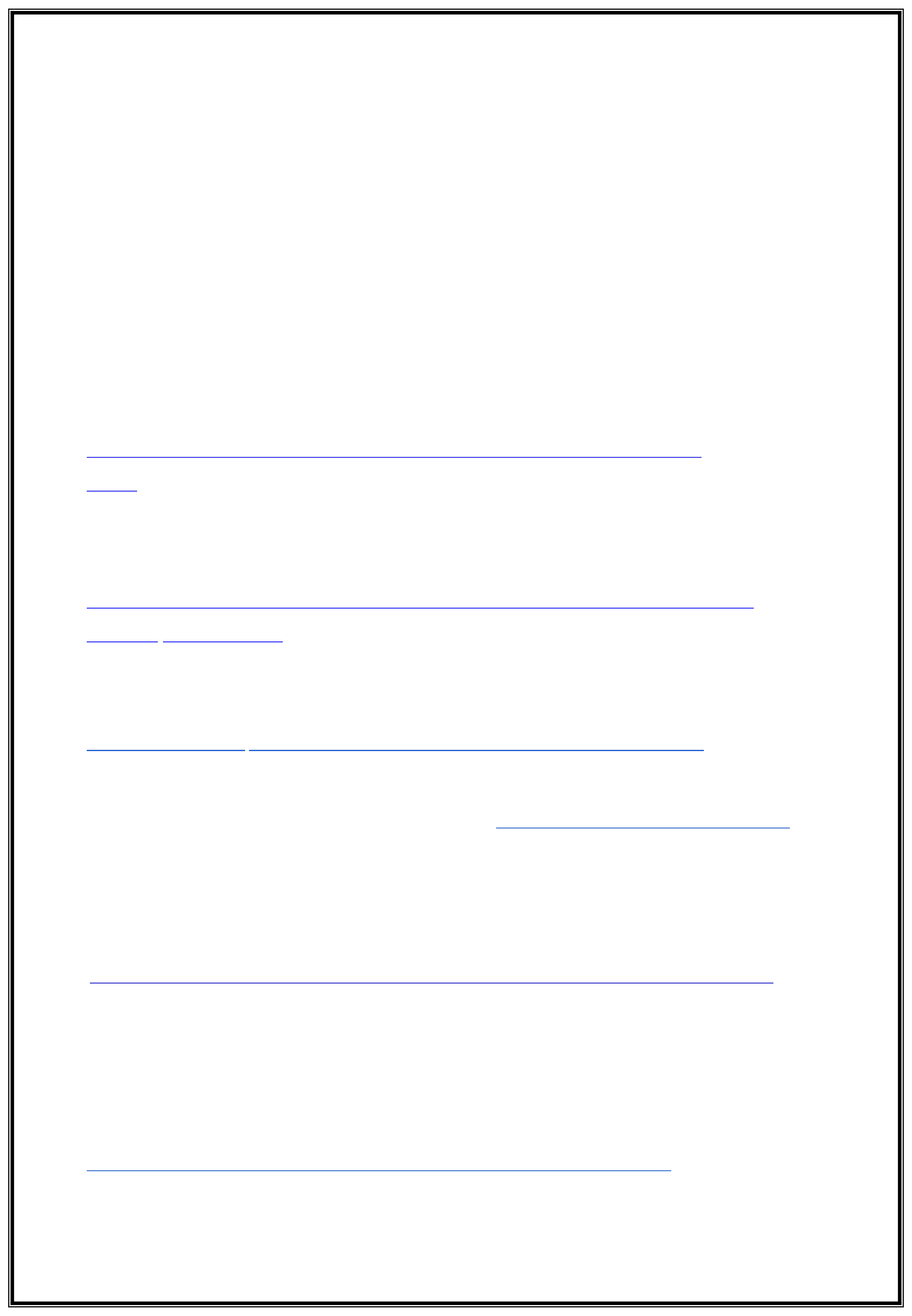
1. Predicting forest fires using supervised and ensemble machine learning algorithms by R. Rishickesh, A. Shahina, A. Nayeemulla Khan
2. Assessing the suitability of soft computing approaches for forest fire prediction by Samaher Al\_Janabi a, Ibrahim Al\_Shourbaji b, Mahdi A. Salman a
3. Learning to predict forest fires with different data mining techniques by Daniela Stojanova, Panče Panov, Andrej Kobler, Sašo Džeroski, Katerina Taškova
4. Various approaches in Forest Fire Detection by Parul Mohindru, Vikshant Khanna, Rajdeep Singh
5. A Data Mining Approach to Predict Forest Fires using Meteorological Data by Paulo Cortez and An´ıbal Morais
6. Study on parallel SVM based on Mapreduce by Zhanquan Sun1 , Geoffrey Fox
7. Bisquert M., Caselles E., Sánchez J.M., Caselles V. “Application of artificial neural networks and logistic regression to the prediction of forest fire danger in Galicia using MODIS data”,July 2012 International Journal of Wildland Fire 21(8):1025-1029 DOI: 10.1071/WF11105
8. Satir, Onur, Suha Berberoglu, and Cenk Donmez. "Mapping regional forest fire probability using artificial neural network models in a Mediterranean forest ecosystem." Geomatics, Natural Hazards and Risk 7.5 (2016): 1645-1658.
9. K V Suresh Babu, Thesis on ”Developing Forest Fire Danger Index Using Geo-Spatial Techniques”, IIT Hyderabad Date: 2019-03-09. Report no: IIIT/TH/2019/21.
10. Zhu, Kaihua, Hao Wang, Hongjie Bai, Jian Li, Zhihuan Qiu, Hang Cui, and Edward Y. Chang. "Parallelizing support vector machines on distributed computers." In Advances in Neural Information Processing Systems, pp. 257-264. 2008.
11. Guo, W., Alham, N. K., Liu, Y., Li, M., & Qi, M. (2016). A resource aware MapReduce based parallel SVM for large scale image classifications. Neural Processing Letters, 44(1), 161- 184.
12. “Pandas Dataframe: Plot Examples with Matplotlib and Pyplot” , https://queirozf.com/entries/pandas-dataframe-plot-examples-with-matplotlib- pyplot
13. “Apply CSS class to Pandas DataFrame using to\_html”, https://stackoverflow.com/questions/50807744/apply-css-class-to-pandas-dataframe- using-to- html/50939211
14. “Machine Learning with Text in PySpark “,

https://datascience- enthusiast.com/Python/PySpark\_ML\_with\_Text\_part1.html

1. “django spark - event sourcing and handling” , https://pypi.org/project/django-spark/

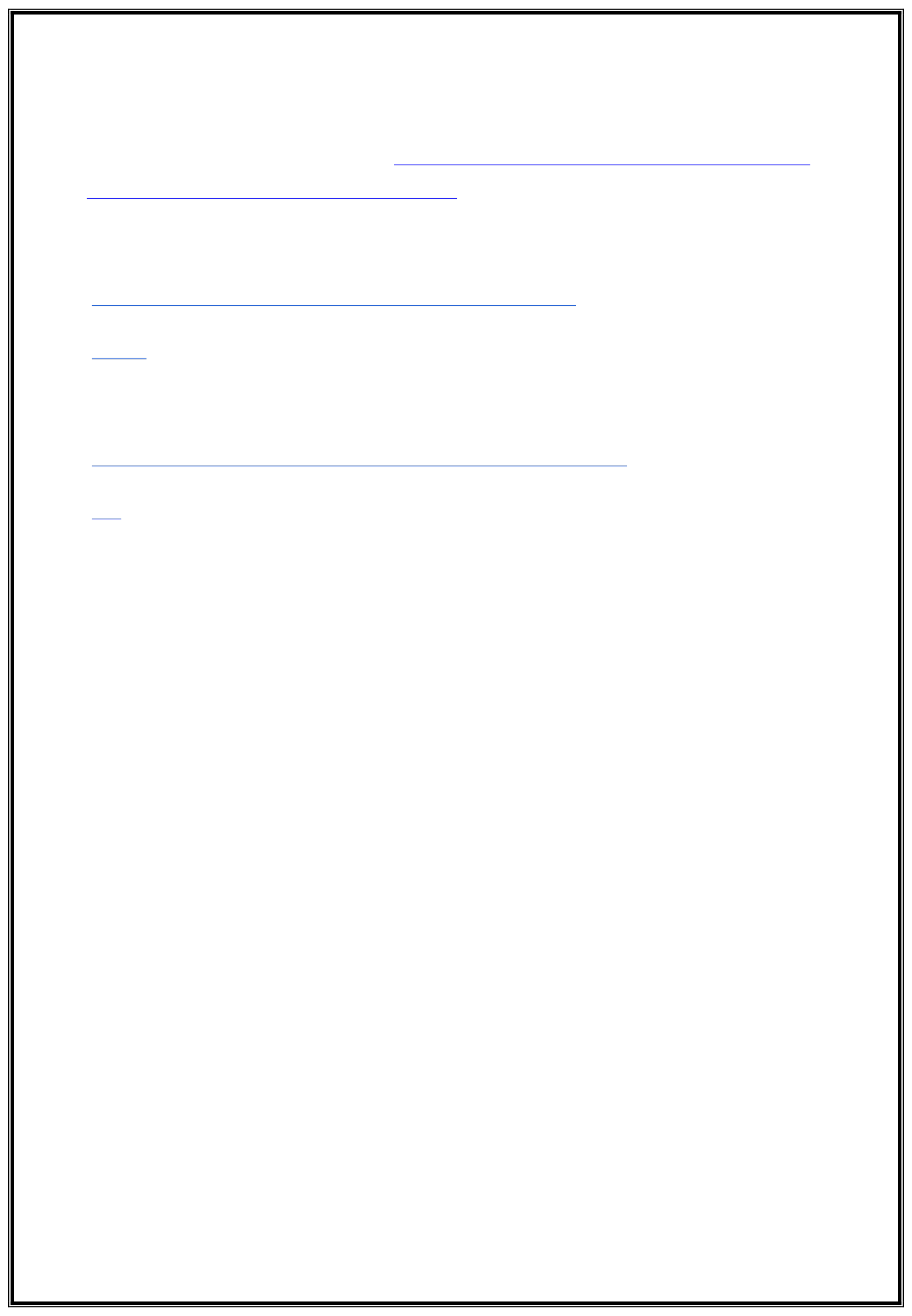
“Interpreting (Fwi) System“

The Canadian Forest Fire Weather Index

,<http://www.dnr.state.mi.us/WWW/FMD/WEATHER/Reference/FWI_Background.pdf>

1. “Identifying the Threshold of Dominant Controls on Fire Spread in a Boreal Forest Landscape of Northeast China”,

https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0055618



1. “Classification and regression” , https://spark.apache.org/docs/2.2.0/ml-classification- regression.html#linear-support-vector-machine
2. “How to extract weather data from Google in Python” ,

ht t ps://www.t hep yt ho nco de. co m/art ic le/e xt r act - weat her - dat a- python

1. “Case studies in fire safety” , https://[www.sciencedirect.com/science/article/pii/S2214398X16300](http://www.sciencedirect.com/science/article/pii/S2214398X16300) 127



## APPENDIX- A

Our Project was selected for KARNATAKA STATE COUNCIL FOR SCIENCE AND TECHNOLOGY.

