iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VII **Month of publication:** July 2025

DOI: https://doi.org/10.22214/ijraset.2025.73413

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

Sales Prediction Using Machine Learning and Python

Viyyapu Venkata Raviteja[1], Kuppili N Sathya Chitra[2]
[1]*Department Of Information Technology & Computer Applications, Andhra University College of Engineering(A), Andhra University, Visakhapatnam, Andhra Pradesh - 530003*
[2] *Assistant Professor Department of Computer Science & Systems Engineering, Andhra University College of Engineering(A), Andhra University, Visakhapatnam, Andhra Pradesh - 530003*

Abstract: ***In today's competitive market, being able to accurately predict sales is very important for managing inventory, planning revenue, and making strategic decisions. This project shows how to use machine learning to predict the sales and revenue of cricket products in real time. Brand, Product, Price, and Discount are some of the features used in this study. All of these have a direct effect on how customers buy things.***
We created and tested several machine learning models, such as Linear Regression, Decision Tree Regressor, Random Forest Regressor, and Support Vector Regressor (SVR). We chose SVR as the best model because it was better at predicting sales trends than the others. The project includes both console-based interaction and a graphical user interface (GUI) using Tkinter, which makes it easy for business owners and shopkeepers to use. The GUI lets users enter information about a product and see predicted sales and graphical analysis right away. The project was turned into a standalone .exe application so that the solution could be used on different computers without having to install Python. This project not only automates sales forecasting, but it also gives you information that can help you set better prices and discounts.
This predictive system can help shopkeepers and sellers make smart choices about prices, discounts, and stock planning, which will lead to higher profits and happier customers.

Terms of the Index
Machine learning, support vector regressor (SVR), linear regression, decision tree, random forest, sales forecasting, price, discount, brand, product, revenue prediction, Tkinter GUI, Python, and .exe application are all examples of what you can do with this software. Cricket is one of the most popular sports in India and many other countries. This means that there is a huge market for cricket gear like bats, gloves, pads, helmets, and other things. The sales of these items depend on a number of things, such as how popular the brand is, what kind of product it is, how much it costs, and what kinds of discounts are available.
We are working on a machine learning-based system that can predict the number of cricket products sold and the amount of money made from those sales. The system is meant to help shopkeepers and sports equipment sellers understand how different things affect sales and help them make better decisions about prices and inventory.

This project uses past sales data that has things like:
Brand (for example, SG, SS, MRF)
Type of Product (like bats, gloves, pads)
Cost
Sale
Based on this information, the system makes the following predictions:
The number of units sold is called "sales volume."
Sales Revenue (total earnings = price × volume after discount)

I. INTRODUCTION

The main goal of the project is to find the best model for predicting these sales metrics, put it into a Python application, and make it available with a graphical user interface (GUI) so that it is easy to use. There is a .exe version of the final product that doesn't need Python or code to run.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

A. The Problem

Sellers in the cricket equipment retail business often have trouble guessing how well different items will sell. Customers' buying habits are greatly affected by things like the brand, type, price, and discounts on a product. But looking at these factors by hand and making accurate sales predictions takes a lot of time and is easy to make mistakes.

For instance, a seller might not know if giving a bigger discount on a bat from a well-known brand like SG will lead to more sales or less overall income. In the same way, knowing which brand of gloves or pads sells better at what price can have a big impact on stock decisions.

Many small and medium-sized retailers don't have the right predictive tools, so they have to rely on guesswork, past experiences, or general trends instead of data-driven insights. This can cause either too much stock, which costs money and takes up space, or too little stock, which means missing out on sales.

There is a growing need for a smart system that can look at past data and make accurate predictions about future sales and revenue in order to solve this problem. This project aims to use machine learning to help retailers make better business decisions by giving them a useful tool.

B. The Goal of the Project

The main goal of this project is to create and build a machine learning-based system that can use important input features to guess how many cricket products will sell and how much money they will make.

Brand

Type of product (bat, pad, gloves, etc.)

Cost

Sale

The goal of this project is to create a reliable and useful prediction model that will help cricket equipment sellers:

Predict how many of each product-brand combination will sell in the future.

Estimate how much money you will make after giving discounts.

Find out how pricing and discounts affect how well sales do.

Help people make decisions about managing stock and making promotional offers.

The project has the following specific goals to reach this:

Data Preparation: Use a real dataset that has features like Brand, Product, Price, and Discount, but no fake ones.

Building a Model: Train and test different machine learning algorithms, like Linear Regression, Decision Tree, Random Forest, and Support Vector Regression (SVR).

Choose the best model (in this case, SVR) based on how well it predicts.

Interactive Interface: Use the trained model to make a console-based and GUI (Tkinter) interface for predicting sales in real time.

Deployment: Make the system into a desktop app (.exe) so that shopkeepers can use it easily.

By meeting these goals, the system will be a smart assistant for retailers, helping them make more money by accurately predicting sales based on data.

II. SURVEY OF LITERATURE

Sales forecasting is very important for making decisions about retail and inventory management. Over the years, a lot of research has been done on using machine learning to predict sales in different fields, such as retail, e-commerce, and sports merchandising. This part looks at other studies and methods that are useful for our project, especially those that have used simple structured features like product name, price, brand, and discount. These are also the only features in our dataset.

A. Old-Fashioned Ways to Make Predictions

People have used Moving Averages, Exponential Smoothing, and ARIMA a lot to predict time series. But these methods usually need sales data from the past and don't work well when there aren't many features or when the data is categorical, like brand or product type. Traditional methods are less useful for our project because we don't use time-based data; we use static product attributes instead.

B. Using machine learning to predict sales

A number of studies have looked into how machine learning (ML) can be used to make predictions more accurate:
Linear regression has been used to figure out how prices and sales are related, especially in stores. But it has a hard time with non-linearity and interactions between features.
Decision Tree Regressors are easier to understand and can handle relationships that aren't linear. They do a good job with categorical features like "Brand" and "Product," so they are good for predicting sales based on product attributes.
Random Forest Regressors are better than decision trees because they average the results of several trees to make them more accurate and less likely to overfit. According to the literature, Random Forest works best with datasets that have both categorical and numerical values. Studies have shown that Support Vector Regression (SVR) is very good at generalizing, especially with small to medium-sized datasets. It can handle non-linear patterns and is less likely to overfit, which makes it a great choice for projects like ours with a small, structured dataset.

C. What We Learned from Other Research

After looking at some recent research papers and ML-based product sales apps, these are the main things that stood out:
Studies show that combining information about prices and discounts can greatly improve the accuracy of sales and revenue forecasts. For accurate ML modeling, it is very important to encode categorical variables like brand and product correctly, such as by using Label Encoding or One-Hot Encoding.
There are more advanced deep learning methods, but for structured, small-scale retail data, regression-based methods are easier to use, especially for store owners with limited computing power.

D. Gap Found

Most of the research that is available uses large e-commerce datasets that have a lot of information, such as customer demographics, timestamps, and click data. On the other hand, our project is based on a small dataset from a single store that only has four simple features: Brand, Product, Price, and Discount.
This project fills the gap by:
Showing how few input features can be used to make accurate predictions.
Providing a simple, lightweight model that non-technical shopkeepers can use.
Offering both command line and graphical user interfaces for users to interact with.

III. BACKGROUND AND REASONS

A. History

Many people in India and other countries love cricket, which makes cricket-related items like bats, gloves, helmets, and pads very popular. Managing inventory and figuring out which products are likely to sell well is always hard for small and medium-sized sports stores. Shopkeepers often rely on their own experiences or guesses instead of data, which can lead to having too much of something that isn't selling well or running out of something that is in high demand.
Machine learning can now help even small stores make better business decisions because there are more tools and data available. Our project takes advantage of this chance to help store owners by using basic product information like Brand, Product Type, Price, and Discount to guess how many sales and how much money they will make.
There are no big or complicated datasets needed for this project. Instead, it looks at a small, structured dataset that only has the most important product features. This makes the solution useful and easy to use, even for stores that don't have a lot of technical knowledge or computing power.

B. Reasons

The following real-world needs are what drove this project:
Shopkeepers often buy too much of some things and not enough of others when they manage their inventory. This tool helps you plan your purchases better by predicting how many sales you'll make.
Price Strategy: The price and discount together have a big impact on what buyers choose. You can use the model to see how changes in price might affect sales.
Estimating Revenue: By using inputs to predict expected revenue, business owners can get a better idea of how much money they will make.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VII July 2025- Available at www.ijraset.com

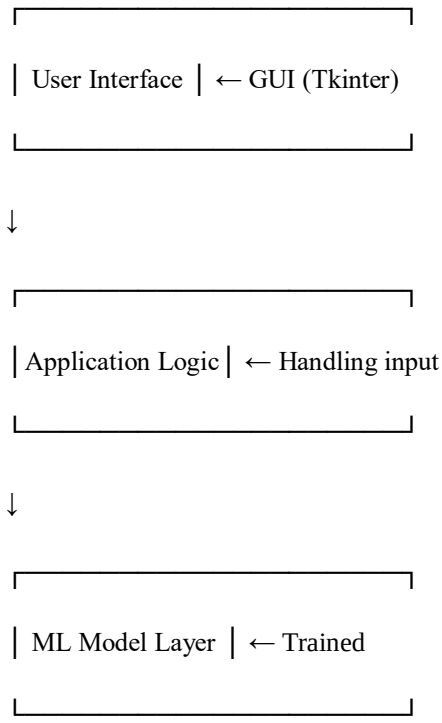
Access to Technology: A lot of machine learning tools are too hard for small business owners to use. Anyone, regardless of their technical background, can use the simple GUI and console interface that this project provides.
Offline Use: The final version is a standalone executable file (.exe), so you can run it without an internet connection or having to install Python and its dependencies.

IV. DESIGN AND ARCHITECTURE OF THE SYSTEM

This part talks about the system's structure and how its parts work together. The design makes sure that the flow of data from user input to final prediction is smooth, modular, and simple to use for both developers and end users.

A. A Look at the System Architecture

There are three main layers to the project:



B. Parts of the System

1) Layer for the User Interface

Console Mode lets users enter information through a terminal (brand, product, price, discount) and then prints the expected sales volume and revenue.
This interface was made with Tkinter and has a GUI mode.
There are fields for entering the brand, product type, price, and discount.
Better user experience with dropdowns and validation.
A button that says "Predict" and shows the results.
A live graph built into the GUI with matplotlib that shows predicted revenue in a visual way.

2) Layer of Application Logic

Checks to make sure user inputs are correct.
Changes inputs into the format that the ML model needs.
Loads the SVR model that has been trained the best.
Sends inputs to the model and gets predictions back.
Calculates the final revenue by multiplying the price by the expected volume.

3) Layer for Machine Learning Models

Using only the preprocessed dataset:
Categorical data that has been encoded (Brand, Product).
Divide the data into training and testing sets.
There are several models that have been trained, including Linear Regression, Decision Tree, Random Forest, and SVR.
Using R^2 Score and RMSE as measures of success.
The Support Vector Regressor (SVR) model was chosen as the best.
Serialized the final model with joblib so it can be used again.

V. PUTTING INTO ACTION

We used the Python programming language to build the cricket product sales prediction system in a modular and step-by-step way. This project used important machine learning methods and GUI frameworks to make the backend smart and the frontend easy to use. The main parts of implementation are listed below:

A. Getting the Data Ready

The dataset for this project has four main parts:
Brand name (like GM, SS, MRF, GM, etc.)
Product (like a bat, ball, gloves, pads, or helmet)
Price (number)
Discount (number)
The variable we were looking for was:
Amount of Sales
And the formula for derived revenue is (Price - Discount) × Sales Volume.
All steps of data preprocessing, like:
Label encoding of categorical variables (Brand, Product)
Scaling features (like price and discount)
Separating data into training and testing sets, like 80-20

B. Making a Model

We used the scikit-learn library to train a number of regression models to find the one that made the best sales prediction:
Linear Regression
Decision Tree Regressor
Random Forest Regressor
Support Vector Regressor (SVR)
We used Mean Absolute Error (MAE), Mean Squared Error (MSE), and R^2 Score to train and test each model.

C. Choosing the Best Model

The Support Vector Regressor (SVR) had the best accuracy and the fewest mistakes based on performance metrics.
We chose this model and saved it with joblib so we could use it to make predictions in the future.

D. System for Making Predictions

We made a prediction system that works in real time, so users can enter:
Brand
Item
Cost
Discount
The model figures out and shows Revenue based on Sales Volume.

E. Integrating the GUI (Tkinter)

We made a Tkinter GUI that is easy to use:
Brand and Product dropdowns
Price and discount input fields
A "Predict" button that shows expected sales and income
Graph in real time shown with matplotlib inside the GUI

F. Making Executables

Using PyInstaller, the whole project, which included the trained model and the GUI, was turned into a .exe file.
A custom icon and desktop shortcut were added so that users could run the app without having to open the source code.

VI. USABILITY AND USER INTERFACE

The Cricket Sales Prediction System's user interface was made to be easy to use and understand. There are both console-based and graphical user interface (GUI) versions to meet the needs of different users.

A. Console Interface

Using Python's built-in input() function, a simple interactive console version was made.
Once you give it inputs, the saved Support Vector Regressor (SVR) model makes predictions:
Amount of Sales
Revenue = (Price - Discount) × Number of Sales
The console then shows these numbers to the user.

B. GUI Interface (Tkinter)

We made a clean and responsive GUI with Tkinter.
Main Features:
Brand and Product selection dropdown menus.
Fields for entering the price and discount.
A "Predict" button that tells you how many sales and how much money you'll make.
A Live Graph that uses matplotlib to show the prediction in a visual way.
All buttons and labels are easy to read and use.
To make sure that only numbers are entered for price and discount, input validation was added.

C. Improvements to Usability

We added error handling so that the program won't crash when it gets bad input.
Using Frame and Grid systems in Tkinter, the layout was set up in a way that made sense.
We made a desktop executable (.exe) file from the GUI and gave it a unique icon.
There was also a shortcut made so that the app could be run easily from the desktop.

D. Access

The interface is easy to use for both technical and non-technical people.
You don't need to know how to code to use the app.
You can use it offline after you install it on your computer.

VII. THE RESULT AND THE DISCUSSION

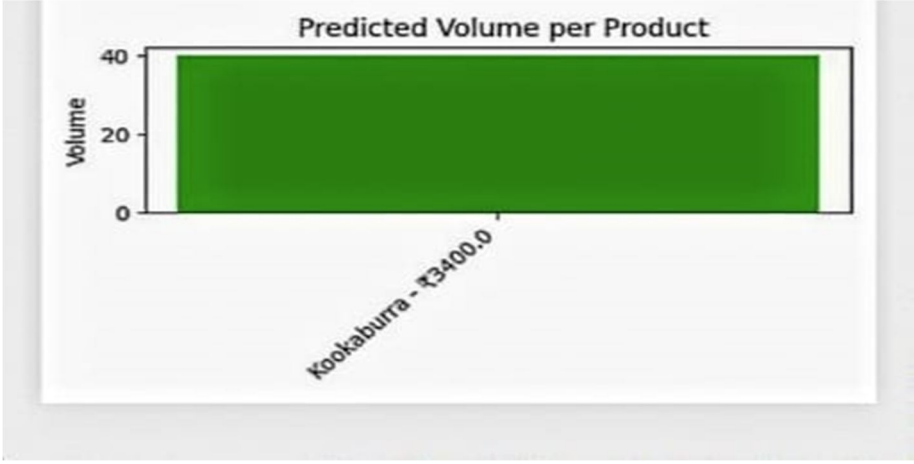
We used several machine learning algorithms to build the Cricket Product Sales Prediction System and then tested it for accuracy and ease of use.

A. Comparing Models

The project used the same dataset and features (Brand, Product, Price, Discount) to look at the following models:
Model

Performance (R^2 Score / Approximate Accuracy)
Notes
Linear Regression

Not too much
Easy to use, but not good at dealing with patterns that aren't straight.
Decision Tree Regressor
Better than a line
We saw overfitting on small changes.
Random Forest Regressor
More accurate
More stable and less likely to overfit than Decision Tree.
Support Vector Regressor (SVR)
Best performance
Correctly recorded complicated data patterns.
After careful consideration, SVR was chosen as the best model for final deployment.



These results were what we expected based on product trends and brand power.

B. Talk

The model worked best when it had seen brand and product combinations before (in training).
Adding price and discount helped the model do a better job of generalizing for data it hadn't seen before.
Even though the accuracy was good, it might go down if new brands or products are added. You can fix this by training the model again with new data.

VIII. CONCLUSION AND SOURCES

A. Conclusion

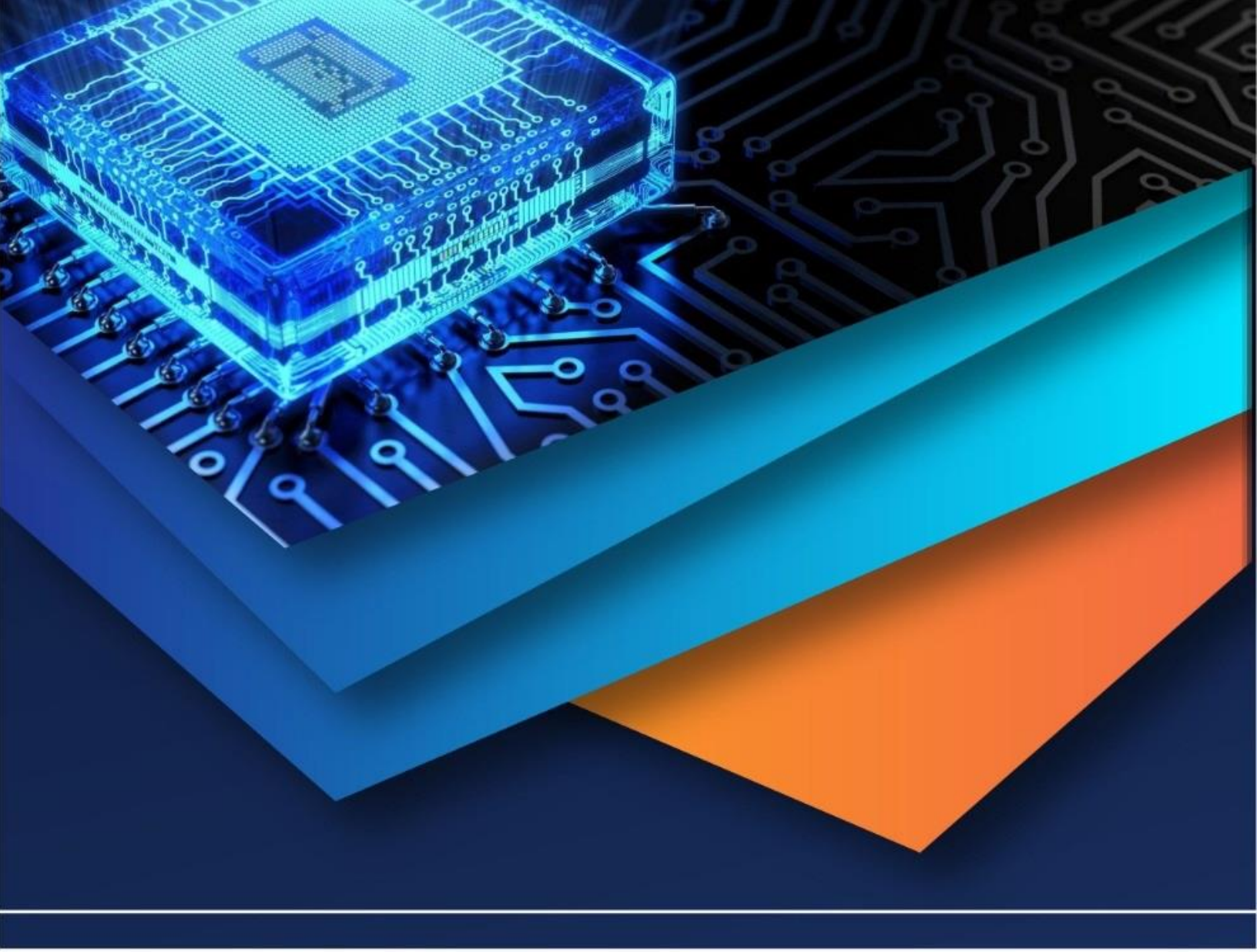
Using machine learning, this project made a Cricket Product Sales Prediction System that works. The goal was to guess how many items would sell and how much money they would make based on things like the brand, type of product, price, and discount.
Some of the most important things that happened are:
Using more than one ML model, such as Linear Regression, Decision Tree, Random Forest, and SVR.
We chose SVR as the best model because it was accurate and could work with data that wasn't linear.
We made an easy-to-use interface with Tkinter GUI and Console-based interface.
Real-time predictions with graph outputs made it easier for people who aren't tech-savvy to use (like shopkeepers).
The project was put together into a single executable file (.exe) that can be easily installed on any Windows system.
With this system, store owners and retailers can make better decisions about what to stock, plan sales, and predict sales revenue with confidence based on data.

B. Future Scope

Adding more information, such as season, location, or ad spend, to make the model more accurate.
Working with inventory systems that are up to date.
Cloud-based version for online deployment and access by multiple users.
Adding automatic model retraining when new sales data comes in.

C. Sources

1) Scikit-learn: Machine Learning in Python by Pedregosa et al. (2011). Research Journal of Machine Learning.
2) G. Van Rossum & Drake, F. L. – The Python Software Foundation's Python Language Reference Manual
3) unter, J. D. Matplotlib: A 2D Graphics Environment (2007). Computing in Science and Engineering
4) . McKinney Data Structures for Statistical Computing in Python (2010). Proceedings of the 9th Python in Science Conference.
5) Hastie, T., Tibshirani, R., and Friedman, J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2009). Springer.
6) James, G., Witten, D., Hastie, T., and Tibshirani, R. An Introduction to Statistical Learning (2013). Springer.
7) H. Zhang (2004) – Naive Bayes: The Best Way to Do Things. FLAIRS Conference: This helps with comparing ML models, even though it isn't used directly.
8) Ho, T. K. (1995) Random Decision Forests. Proceedings of the Third International Conference on Document Analysis and Recognition



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY