

RICE CROP MONITORING TIME SERIES ANALYSIS

Project Hand-out, Faculty Development Program – NaanMudhalvan

SmartInternz

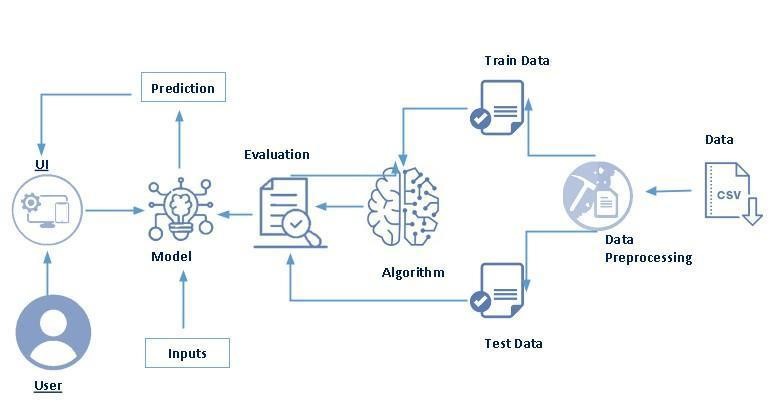
www.smartinternz.com

**Rice Crop Monitoring TIme Series Analysis**

Rice crop monitoring and analysis involves the use of modern technologies and methods to track and analyze the growth, health, and yield of rice crops. This process includes the use of remote sensing tools, such as drones and satellites, to collect data on crop health, soil moisture, and other key metrics. Additionally, data analysis techniques are used to identify patterns, trends, and potential issues that may impact crop performance. This information can be used by farmers and other stakeholders to make informed decisions about irrigation, fertilization, and other crop management strategies. Ultimately, the goal of rice crop monitoring and analysis is to improve crop yields, reduce waste and environmental impact, and enhance food security for communities around the world.

Rice is a staple food crop for different countries. The production of staple crop will reflect the situations prevailing in the country like famine or flood. This dataset captures production of rice ,along with unit of measure ( Tonnes) and collected data source ( official or estimated).

**Technical Architecture:**



**Project Flow:**

* User interacts with the UI to enter the input.
* Entered input is analysed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

* Define Problem / Problem Understanding
  1. Specify the business problem

○ Business requirements

○ Literature Survey

○ Social or Business Impact.

● Data Collection & Preparation

○ Collect the dataset

○ Data Preparation

* Exploratory Data Analysis
  1. Descriptive statistical

○ Visual Analysis

* Model Building
  1. Training the model in multiple algorithms

○ Testing the model

* Performance Testing & Hyperparameter Tuning
  1. Testing model with multiple evaluation metrics

○ Comparing model accuracy before & after applying hyperparameter tuning

* Model Deployment
  1. Save the best model

○ Integrate with Web Framework

* Project Demonstration & Documentation
  1. Record explanation Video for project end to end solution

○ Project Documentation-Step by step project development procedure

**Prior Knowledge:**

To complete this project, you must require following software’s , concepts and packages

* Anaconda navigator:
  + Refer to the link below to download anaconda navigator
  + Link : <https://www.youtube.com/watch?v=5mDYijMfSzs>
* Python packages:
  + open anaconda prompt as administrator
  + Type “pip install prophet” (make sure you are working on python 64 bit)
  + Type “pip install flask”.
* Deep Learning Concepts
  + CNN: <https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add>
  + ARIMA: <https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp#:~:text=An%20autoregressive%20integrated%20moving%20average%2C%20or%20ARIMA%2C%20is%20a%20statistical,values%20based%20on%20past%20values>.
  + SARIMA: <https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/>
  + Facebook Prophet: <https://www.geeksforgeeks.org/time-series-analysis-using-facebook-prophet/>
  + Flask Basics : <https://www.youtube.com/watch?v=lj4I_CvBnt0>

**Project Objectives:**

By the end of this project you will:

* know fundamental concepts and techniques of Convolutional Neural Network.
* gain a broad understanding of image data.
* Knowhow to pre-process/clean the data using different data preprocessing techniques.
* know how to build a web application using Flask framework.

**Project Structure:**

Create the Project folder which contains files as shown below

Text

Description automatically generated

* We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
* Static folder and assets folder contain the CSS and JavaScript files along with images.
* **fbprophet.pkl** and **sarima.pkl** are our saved models.
* Further we will use these models for flask integration.

# Milestone 1: Define Problem / Problem Understanding

**Activity 1: Specify the business problem**

Refer Project Description

## Activity 2: Business requirements

Here are some potential business requirements for an ecommerce product delivery estimation predictor using machine learning:

## Accurate forecasting: The predictor must be able to accurately forecast rice crop production for a given time period in the future. The accuracy of the forecasting is crucial for farmers, agribusinesses, and other stakeholders to make informed decisions on the production and marketing of rice.

## Real-time data acquisition: The predictor must be able to acquire real-time data on weather patterns, soil moisture, and other relevant factors that affect rice crop production. The data acquisition must be seamless and efficient to ensure that the predictor is always up-to-date with the latest information.

## User-friendly interface: The predictor must have a user-friendly interface that is easy to navigate and understand. The interface should present the results of the predictor in a clear and concise manner to enable farmers and other stakeholders to make informed decisions.

## Scalability: The predictor must be able to scale up or down depending on the size of the rice production area. The predictor should be able to handle large volumes of data without compromising on its accuracy or efficiency. This is important as rice production areas can range from small farms to large commercial operations.

## Activity 3: Literature Survey (Student Will Write)

A literature survey would involve researching and reviewing existing studies, articles, and other publications on the topic of project. The survey would aim to gather information on current systems, their strengths and weaknesses, and any gaps in knowledge that the project could address. The literature survey would also look at the methods and techniques used in previous projects, and any relevant data or findings that could inform the design and implementation of the current project.

**Activity 4: Social or Business Impact.**

1. Improved rice crop yields: The rice crop production predictor can help farmers optimize their rice crop yields by providing accurate forecasts and real-time data. This can lead to increased crop yields and better crop quality, which can positively impact the income of farmers and the availability of rice for consumers.
2. Reduction in production costs: The predictor can also help farmers reduce production costs by providing insights into optimal planting and harvesting times, as well as data on weather patterns and soil moisture. This can lead to better resource allocation, reduced waste, and increased profitability for farmers.
3. Sustainable rice production: The rice crop production predictor can contribute to sustainable rice production by enabling farmers to make informed decisions about their farming practices. For example, by using the predictor to identify optimal planting and harvesting times, farmers can reduce the use of water, fertilizer, and pesticides, which can have positive impacts on the environment and public health. This can also contribute to the long-term viability of the rice industry.

# Milestone 2: Data Collection & Preparation

DL depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

## Activity 1: Collect the dataset.

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

Link: <https://www.kaggle.com/datasets/bharathposa/rice-production-by-country-from-1961-to-2021>

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

**Note:** There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

**Activity 1.1: Importing the libraries**

Import the necessary libraries as shown in the image.

Graphical user interface, text

Description automatically generated

## Activity 1.2: Read the Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of the csv file.

Graphical user interface, text, application

Description automatically generated

## Activity 2: Data Preparation

As we have understood how the data is, let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

* Checking for missing values
* Resampling data

Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

## Activity 2.1: Checking for missing values

* Let’s find the shape of our dataset first. To find the data type, shape, null values df.info() function is used.

Text, table

Description automatically generated

* For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.

Text

Description automatically generated with medium confidence

## Activity 2.2: Converting year to timestamp

First lets check the type of data of the column “Year” using the type() function.

Graphical user interface, application

Description automatically generated

We will convert the data of “Year” column using the pandas “to\_datetime” function. To do this first we have to convert the values of “Year” column to string first. We can check the type of data as shown below.

Graphical user interface, text, application, email

Description automatically generated

## Activity 2.3: Resampling the Data

We are going to resample the data by taking average of “Value” column based on the “Year” column i.e. we will be getting a dataset with average production of rice for the years 1961 – 2021. To do this we’ll first create a new dataset with “Year” column as “ds” and “Value” column as “y”. We create the data with the column names “ds” and “y” to train the Facebook prophet model as it accepts the data with columns formatted as above.

Table

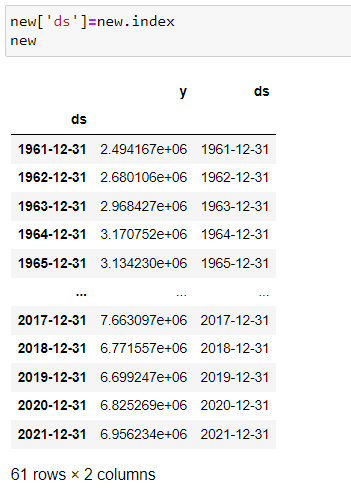
Description automatically generated

We’ll now resample the data to annual frequency by using the resample() function of pandas dataframe. To do this first we have to set our column containing years to index. Using the index as years the resample function will then process the data to give us resampled data as shown below.

Graphical user interface, text, application

Description automatically generated

We’ll now add the index back using the following piece of code.



# Milestone 3: Exploratory Data Analysis

## Activity 1: Descriptive statistical

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

A picture containing table

Description automatically generated

## Activity 2: Visual analysis

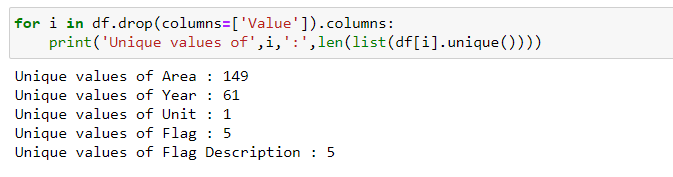
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

**Activity 2.1: Univariate analysis**

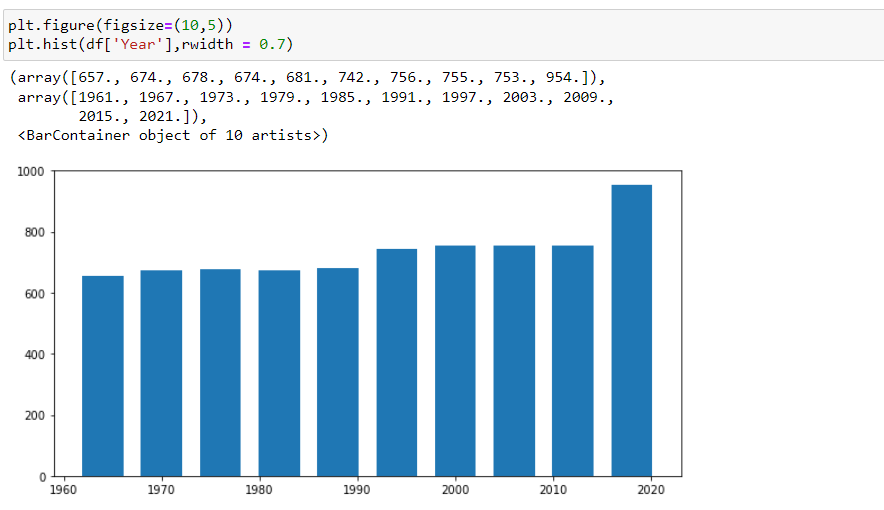
In simple words, univariate analysis is understanding the data with single feature. Here we have displayed two different graphs such as histplot and countplot.

Seaborn package provides a wonderful function histplot. With the help of histplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.

From the plot we came to know,



From the above piece of code we can observe that there are 149 Areas from which production of rice has been collected for 61 years with upto 5 flags to classify the data collected as official or un-official and other types. The production of rice was recorded in “Value” column with only one unit i.e. tonnes.



After plotting a histogram to check the number of rows available for every year we can see that the data is balanced with highest number of data available for the year 2020.

## Activity 2.2: Bivariate analysis

To find the relation between two features we use bivariate analysis.

Chart, scatter chart

Description automatically generated

In the bar chart visualised below following observations can be done:

* Country with highest rice production is China clearly stating the popularity of rice as it is their staple food.
* The countries producing the highest quantity of rice are China, India, Thailand.
* There are also other countries which consume rice on a daily basis as staple food such as Indonesia, Bangladesh and others but the production of rice in these areas is less which can be due to many reasons such as inadequate land.

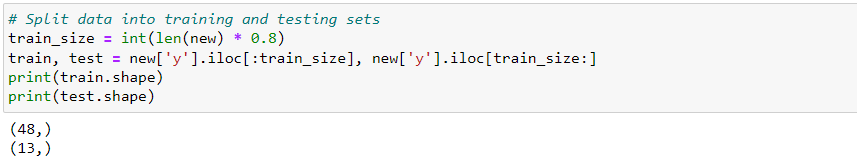
Chart

Description automatically generated

**Splitting data into train and test**

Now let’s split the Dataset into train and test sets. First split the dataset into x and y and then split the data set.

Here x and y variables are created. On x variable, data is passed with dropping the target variable. And on y target variable is passed.



# Milestone 4: Model Building

## Activity 1: Training the model in multiple algorithms

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying seven classification algorithms. The best model is saved based on its performance.

## Activity 1.1: ARIMA model

First we are going to initialise the ARIMA model with an order of (1,1,1) to start with (later we will optimise the parameters) and training data is passed to the model with.fit() function. Test data is forecasted/predicted with forecast() function and saved in a new variable. For evaluating the model, r2\_score and MAE scores are used.

Chart, line chart

Description automatically generated

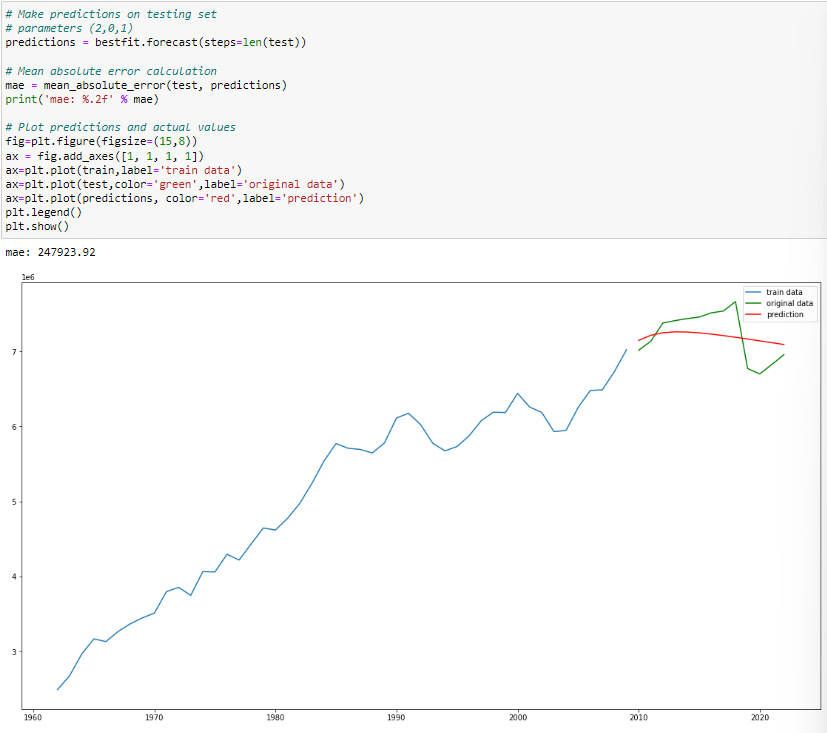
## Activity 1.2: Optimising ARIMA model

Clearly, we can see that the model is not performing well on the data. We’re getting a very high error and a very low r2\_score. So, we’ll optimise the hyperparameters of models using the following code where will train the model iteratively to find the optimal set of parameters.

Text

Description automatically generated

From the output above we choose the best performing set of parameters i.e. (0,0,0). Since, the best performing model is saved in “bestfit” variable we don’t need to re-train the model.

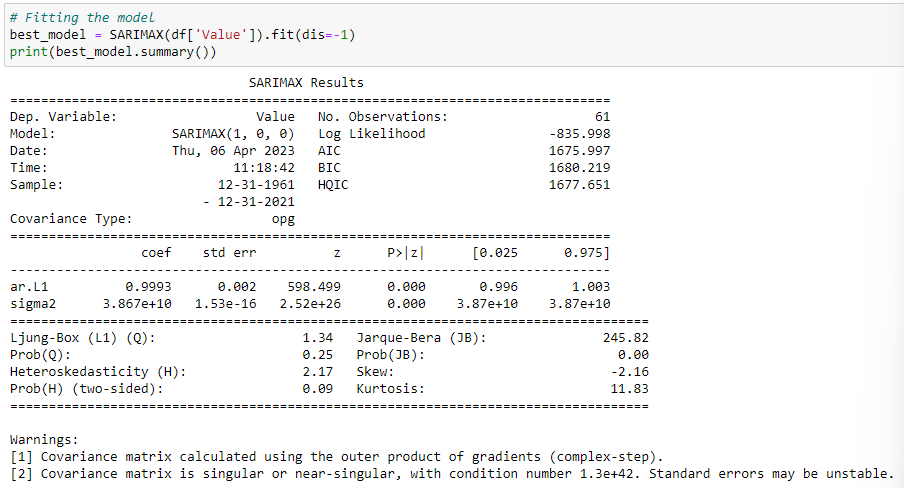


## Activity 2: SARIMA model

Our data has seasonality meaning that there are predictable patterns in the time series. So we can use seasonal ARIMA algorithm to get more accurate results.

**Activity 2.1: Train the model**

We are going to train the SARIMA model using SARIMAX().fit() function with parameters as shown below. The model summary can be observed as below.



**Activity 2.2: Evaluating the model**

Chart

Description automatically generatedThe model diagnosis can be done using the plots as shown below which clearly show that the fit of the data is good with the help of QQ plots, Scatter and line plot, Histogram and correlogram.

Chart

Description automatically generated

In the above line plot we can observe that the model performs well on the data and the prediction can be observed after 2021.

Graphical user interface, text, application, email

Description automatically generated

As we can see the MAE and r2\_score are far better than ARIMA model.

You can save the model as pickle files using the following piece of code.

Graphical user interface, text, application, email

Description automatically generated

## Activity 3: Facebook Prophet model

Our data has seasonality meaning that there are predictable patterns in the time series. Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

**Activity 2.1: Train the model**

First we are going to install the facebook prophet model using the following command and import the required libraries.

Graphical user interface, text, application, email

Description automatically generated

Now we are going to initialise our prophet model with the seasonality mode as ‘multiplicative’. This type of data tends to show an exponential trend. And if you've got a growing business your sales data tends to look more like the latter. And if you want to forecast such data, you want to try 'Multiplicative' for the seasonality effect. We train the model by calling the fit() method.

Text

Description automatically generated

Predictions are then made on a dataframe with a column **ds** containing the dates for which a prediction is to be made. You can get a suitable dataframe that extends into the future a specified number of days using the helper method .**make\_future\_dataframe**. By default it will also include the dates from the history, so we will see the model fit as well.

Graphical user interface, text, application

Description automatically generated

The **predict( )** method will assign each row in future a predicted value which it names **yhat**. If you pass in historical dates, it will provide an in-sample fit. The forecast object here is a new dataframe that includes a column **yhat** with the forecast, as well as columns for components and uncertainty intervals.

Table

Description automatically generated

**Activity 2.2: Evaluating the model**

You can plot the forecast by calling the **plot( )** method and passing in your forecast dataframe.

Chart, line chart

Description automatically generated If you want to see the forecast components, you can use the **plot\_components** method. By default you’ll see the trend, yearly seasonality, and weekly seasonality of the time series. If you include holidays, you’ll see those here, too.

Chart, line chart

Description automatically generated

We can calculate the r2\_score and mean absolute error as shown below.

Graphical user interface, text, application, email

Description automatically generated

As we can see we got a higher Mean absolute error than SARIMA model and a slightly less r2\_score showing that the SARIMA model is a better approach for this use case.

You can save the model as pickle files using the following piece of code.

Graphical user interface, text, application

Description automatically generated

# Milestone 6: Model Deployment

## Activity 1: Save the best model

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance and saving its weights and configuration. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

Graphical user interface, text, application, email

Description automatically generated

## Activity 2: Integrate with Web Framework

In this section, we will be building a web application that is integrated to the model we built. An UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

* Building HTML Pages
* Building server-side script
* Run the web application

**Activity 2.1: Building Html Pages:**

For this project create two HTML files namely

* index.html
* inner-page.html
* portfolio-details.html

and save them in the templates folder.

It is not necessary to follow the exact format as above so feel free to use whatever templates or format you like. Be creative!

**Activity 2.2: Build Python code:**

Import the libraries

****

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.



Render HTML page:

Text

Description automatically generated

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, ‘/’ URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered.

Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:

Text

Description automatically generated

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

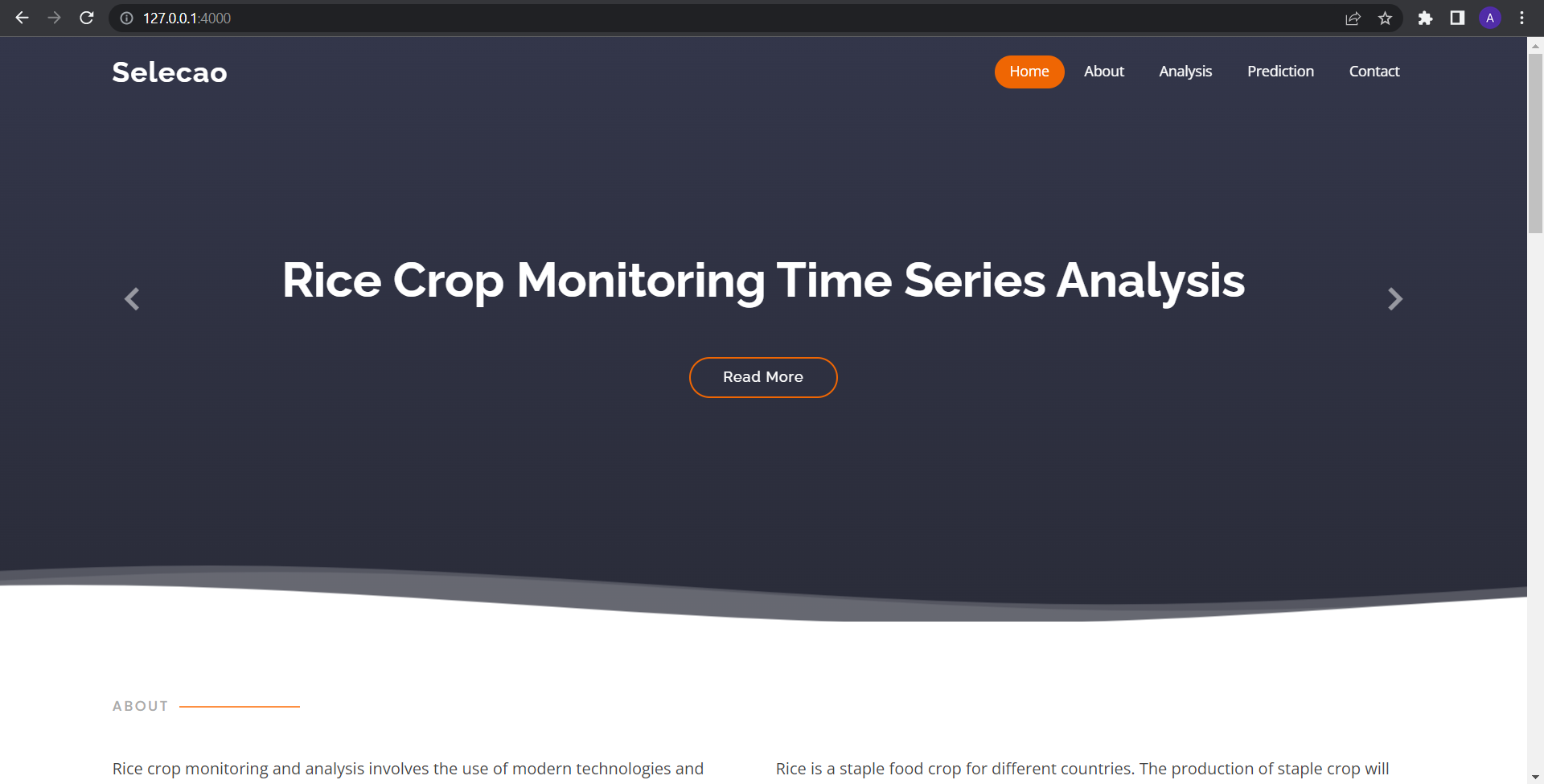
## Activity 2.3: Run the web application

* Open anaconda prompt from the start menu
* Navigate to the folder where your python script is.
* Now type “python app.py” command
* Navigate to the localhost where you can view your web page.
* Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

Text

Description automatically generated

Now, Go to the web browser and write the localhost URL ( <http://127.0.0.1:4000> ) to get the below result



Graphical user interface, text

Description automatically generated

Chart, line chart

Description automatically generated

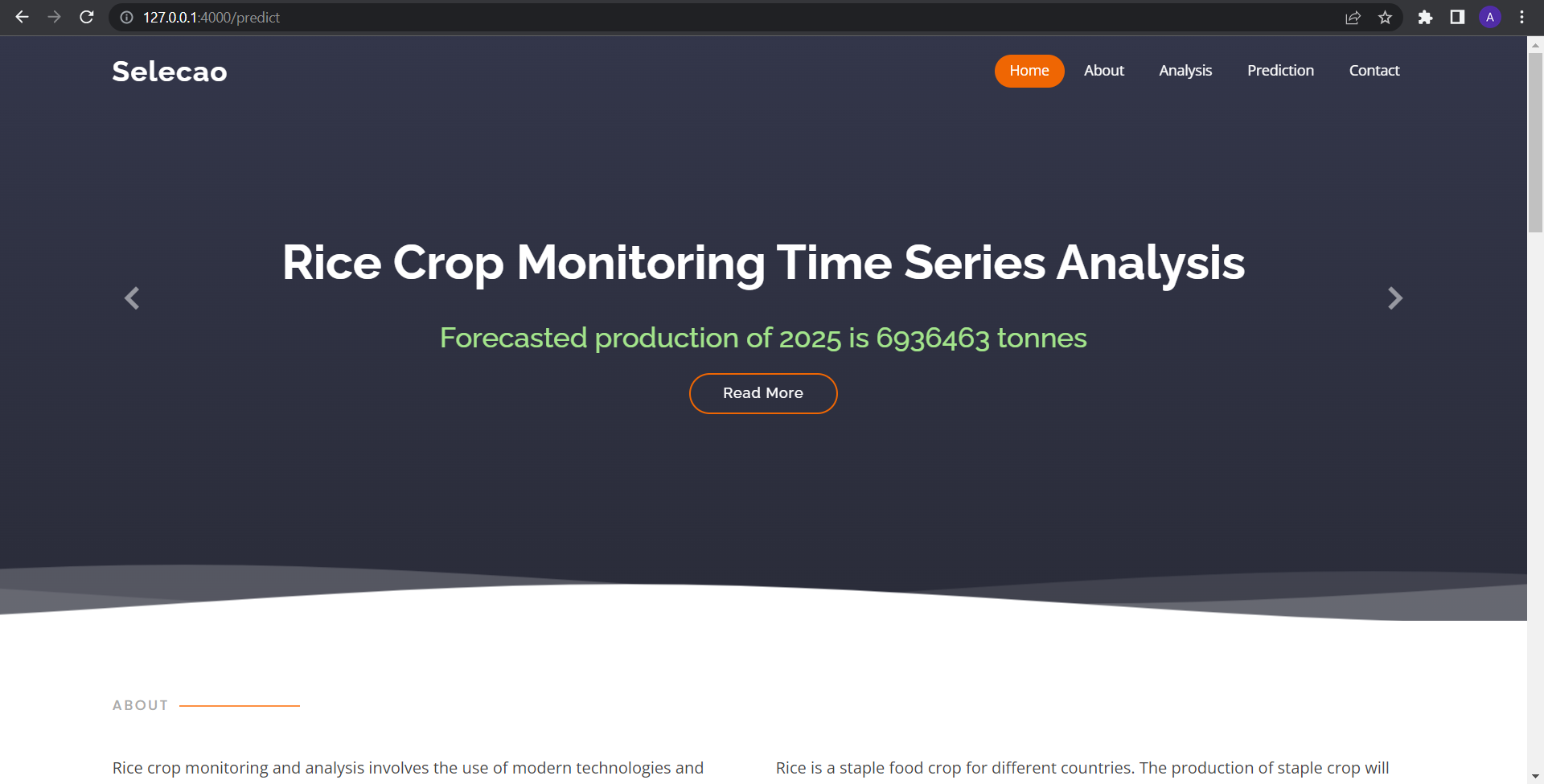
Graphical user interface, text, application, website

Description automatically generated

Graphical user interface, application

Description automatically generated

Upon entering the data in input fields and clicking on submit and predict we’ll get the prediction as shown below.



**Milestone 7: Project Demonstration & Documentation**

Below mentioned deliverables to be submitted along with other deliverables.

**Activity 1: - Record explanation Video for project end to end solution.**

## Activity 2: - Project Documentation-Step by step project development procedure.

Create document as per the template provided.