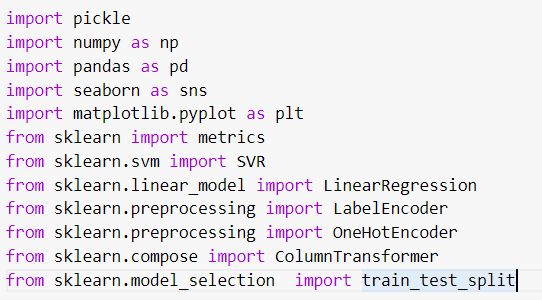
Before importing any module, we should make sure these modules are present in our work environment. If these modules are not found then do: pip install <library\_name>.

For eg: if numpy module is not present in your system then do: pip install numpy

Likewise, pip install all the required modules.



We have imported libraries like pickle to save the trained model to our local system. Again we use pickle to load the model in our testing environment.

Next, we imported numpy library to do mathematical functionalities, especially linear algebra functions.

Then we imported pandas library to load the dataset into our development or working environment. Next we imported the seaborn library to do data visualization on our dataset. Seaborn helps in plotting graphs like pairplot, . Barplot, hist plot, count plot etc.,.

We have also used matplotlib library for data visualization tasks.

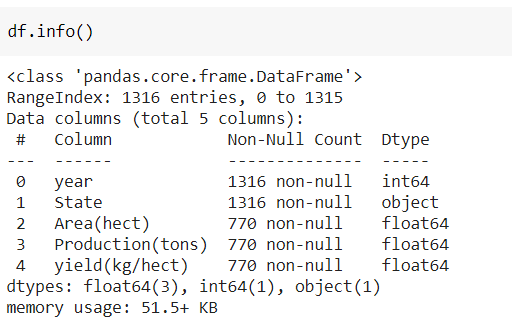
Then we imported metrics from sklearn library to check the metrics like accuracy which tells how our model is working on train data & test data.

Then we imported SVR function from sklearn library, this function helps in building the support vector regressor model for the training dataset.

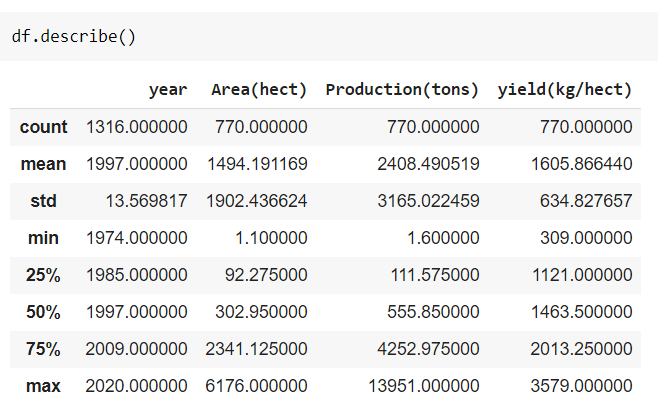
Then we imported LinearRegression function from sklearn library to build the linear regression model on train dataset.

Then we imported labelencoder from sklearn library. This function helps in converting the categorical column into numerical values( integers ).

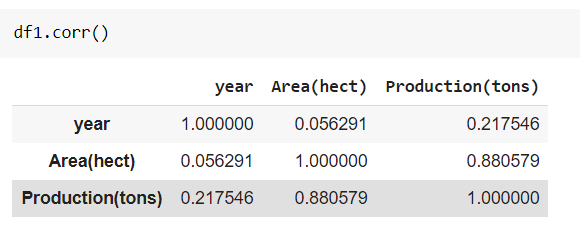
Because we dont pass text columns directly to machine learning models. We do some data pre-processing to such columns first. After label encoding, we do one hot encoding for label encoded columns to make sure that we don't give unequal weightage to that column values. We have also imported train test split function from sklearn library as we need to divide the dataset into training & testing data.



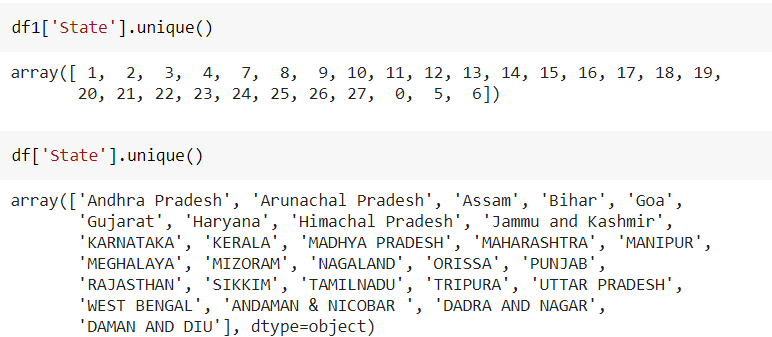
Once we load the paddy crops dataset using pandas we need to see how the data is ?. So we used info function on our dataset. This function tells us how many different columns we have. What are the names of the columns. How many missing values do we have in each column. Whats the data type of each column. How much memory is used. This helps us mainly in estimating the null values in each column. We can also tell the shape of the dataset from info().



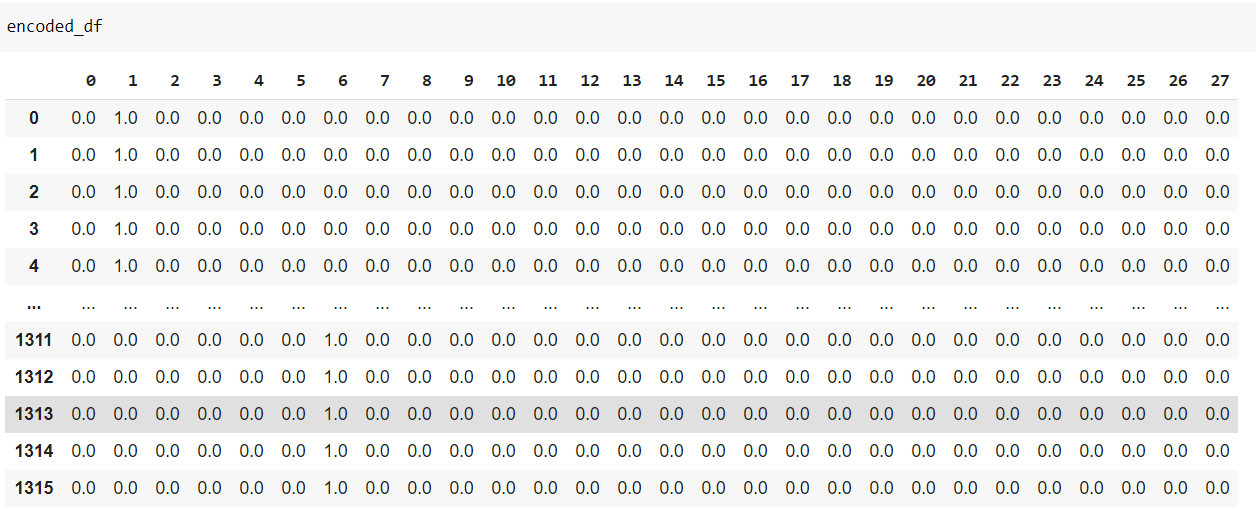
Now we have applied descrobe function to our dataframe df. This gives us count mean standard deviation of numerical columns. It also gives us the minimum & maximum values in a column. Apart from this information it also gives us Q1 Q3 of every column. From this we can also calculate inter quartile range of every column.



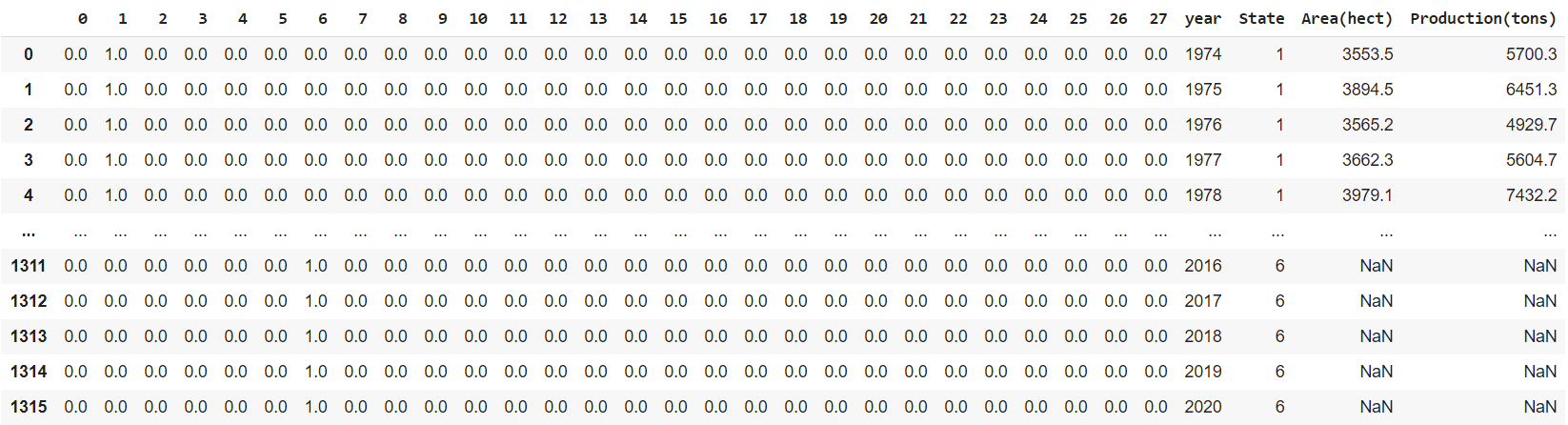
Now we calculated correlation of each numerical column with another numerical column. We can cleary see a positive correlation between area & production columns. And we can see there is no correlation between area & year, year & production columns. So the output production is highly correlated with input column Area.



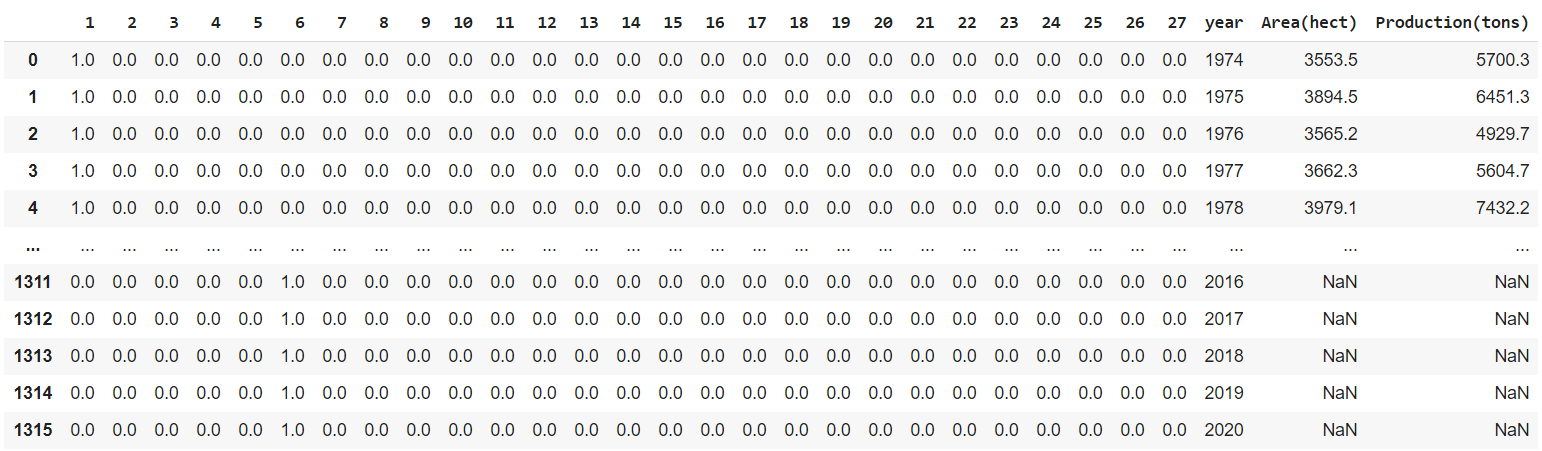
After label encoding the state column. We can see how the state names have changed to integers. df1 is the label encoded data frame for the state column. df is the original dataframe. So we can find the unique values for the state column before label encoding and after labelencoding after applying unique function to state column.



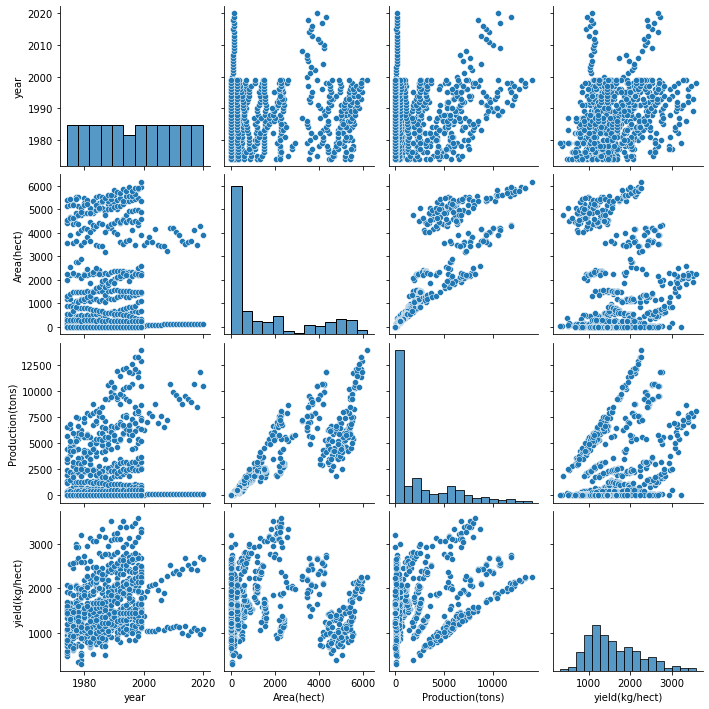
After label encoding we have done the one hot encoding. This converts the label encoded column which has 28 unique values to 28 different columns.



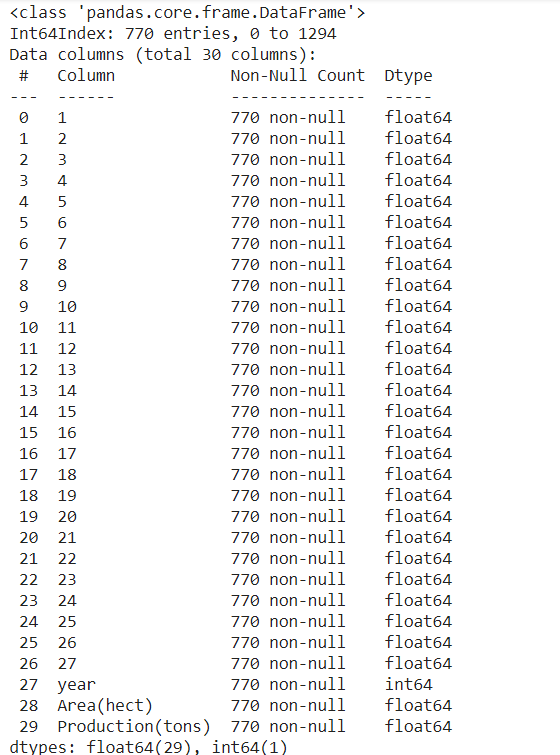
Now we have added the one hot encoded column with the original dataframe.



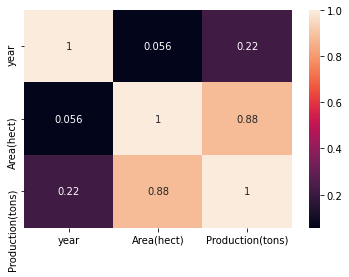
We have labelencoded the state column which converted the state names into integers. Then we did one hot encoding to the same column which converted all the different states into an individual column. Before passing the dataset to the machine learning algorithm, We remove any one column from those 28 columns. For eg: if we remove column 0 then we will have 1 to 27 columns in the dataset. If all these 27 values are 0 then it means column 0 value will be 1 which is equal to Andaman & Nicobar state. So, instead of keeping all 28 columns in the dataset, we remove any one of the column from these 28 columns.



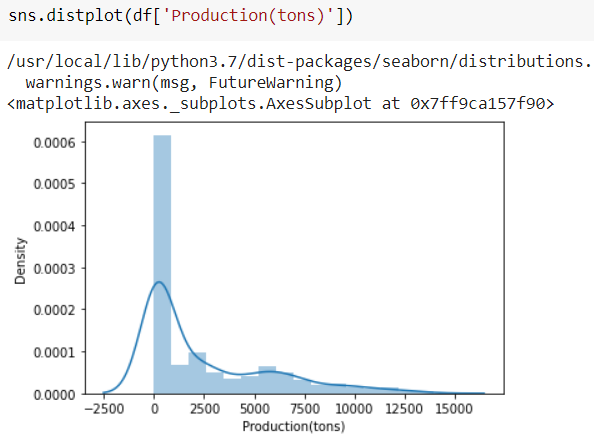
Here, we have plotted a pairplot using seaborn library which displays pair wise relation through a scatter plot. We can see a pattern between production and area columns, as area increases, production value is also increasing. This indicates the correlation between these columns. Output column production is influenced more by input feature area



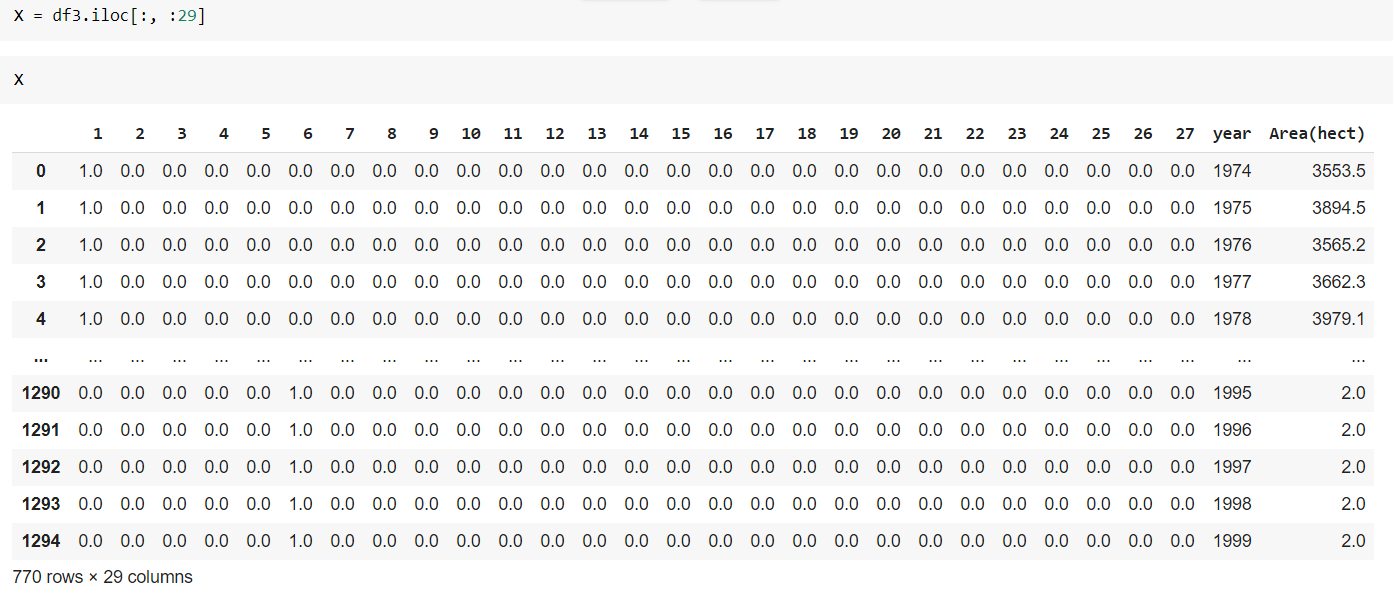
We have used dropna to remove the rows which have null values in the dataset. After removing such rows, this is the info of the day which shows 770 rows compared to 1316 rows previously.



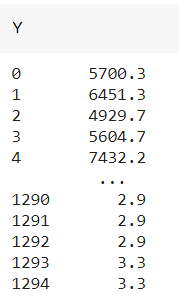
Previously, we have used .corr() function to see the correlation between numerical columns, for the same correlation data we are now visualizing through a heat map, we can see the legend to the right of the heatmap which indicates the range from 0 to 1. According to the correlation value between numerical columns, it displays the color as shown in the legend.



Here, we can visualize how the data is distributed in the production column using distplot in seaborn. We can clearly see that the data lies more in between 0 & 2500 tons.



Now, we have selected the input columns to train the machine learning model. We have Year, Area, 1 to 27 (individual states) columns as the input data. We stored this input data in variable X. You can see the shape of the input data which is (770, 29). 770 is the number of rows. 29 is the number of columns.



Y is the production(Tons) column which is the output column for the given input data. We stored this column in variable Y.

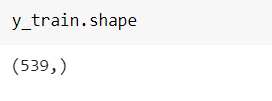


Now we have divided the data into train dataset & test dataset using train test split function. We have used test\_size as 0.3 which means 30% of the dataset will be used for testing the data & remaining 70% of the dataset will be used for training the machine learning algorithm.

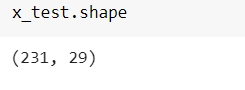
We also mentioned random state as 20 which means the data division for training & testing will be constant for all the time as we mentioned random state. So, the data partitioning will be constant how many times you execute this command. If we don’t mention the random state then also it’s fine but when we run it different times, the data that will be partitioned will be different each time you execute this command.



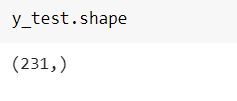
Shape of the input data used for training the algorithm is (539, 29) which means we have 539 rows & 29 columns.



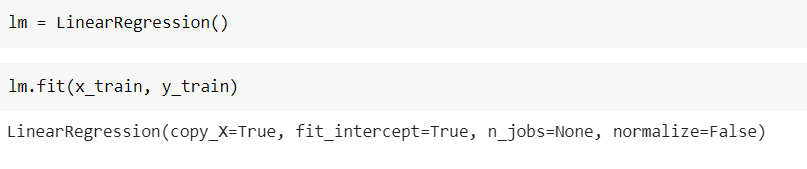
Shape of the output data for training the algorithm is (539,) which is a 1 dimensional array.



Shape of the input data used for testing the algorithm once the algorithm gets trained is (231, 29) which means we have 231 rows & 29 columns.



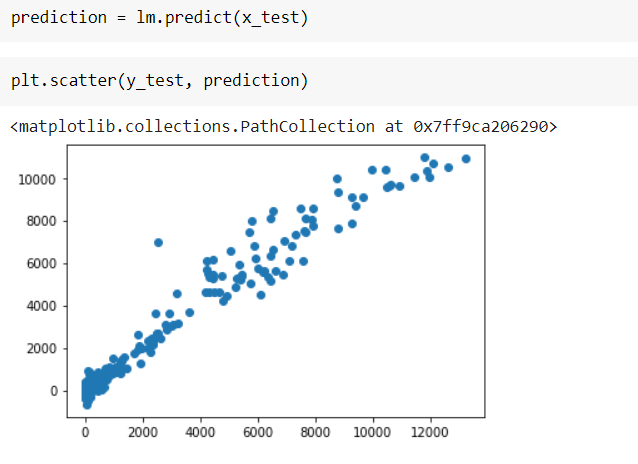
Shape of the output data for testing the algorithm once the algorithm gets trained is (231,) which is a 1 dimensional array.



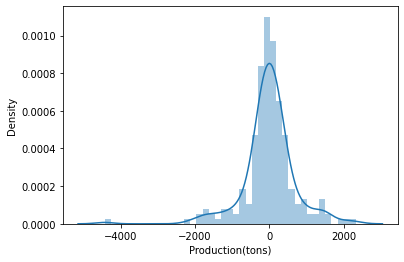
We have initiated a linear regression algorithm and fitted the training data to linear regression algorithm.

“Multiple linear regression” is a supervised machine learning algorithm which can be used for regression tasks. However, it is used when we have more than one input feature. So, the multiple linear regression has one continuous dependent variable(y) and two or more independent variables(x1, x2, …..xn).

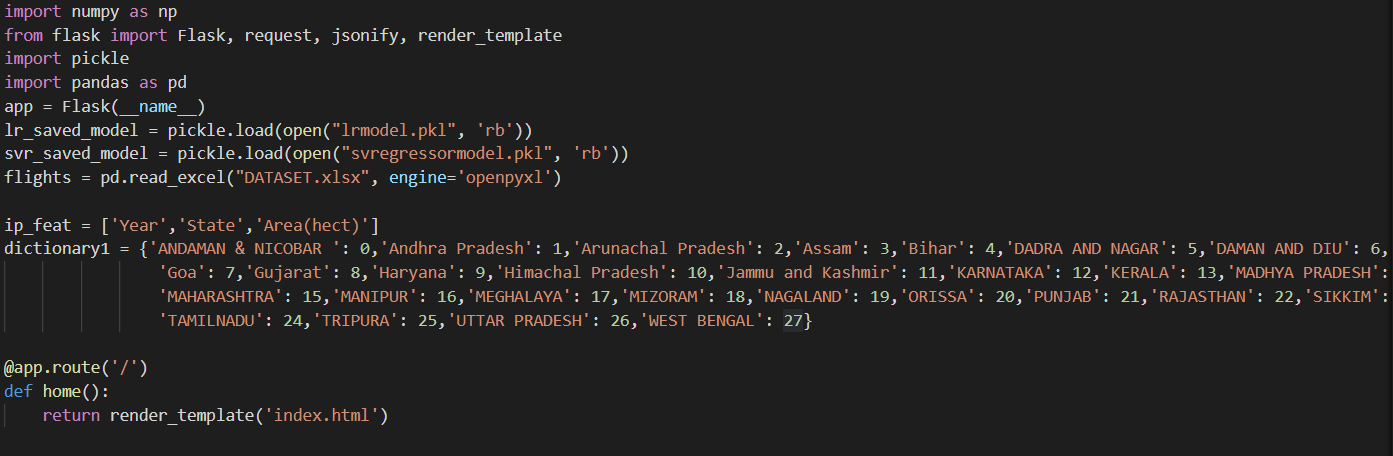




Here we have taken the trained multi linear regression model and used this model to predict the production value for unseen data (i.e., test data). We have plotted the scatter plot for actual production values for test data & predicted production values for test data. If you see the pattern of these data points, it says that the actual values are very close to predicted values.



We have also plotted the distribution plot for the difference of actual & predicted values. If you see most of the data lies at & around zero which means among most of the predicted values the difference between actual & predicted values is zero. So, this concludes that predicted values are very close to actual values.



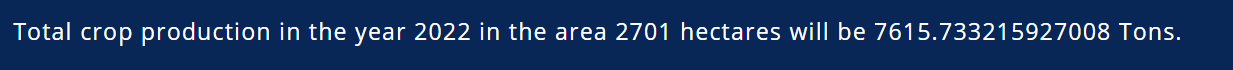
We have also built a flask application to run the application in localhost server and pass input through UserInterface(UI) & display the predicted output on UI.



In the UI, We have State Number, Year and Area as the input parameters, once we pass these input values and click on predict button we’ll the get production in Tons. In the backend, we’re actually using the trained ml model which is giving this output.



We have passed input values as 15 for state number which corresponds to Maharastra, year we passed as 2022, area we passed as 2701 hectares. Now, we click on predict button to predict the production.



We got the output as 7615.7 Tons which means if we do paddy crop in the area of 2701 hectares in the year 2022 in Maharastra state, we’ll get 7615.7 Tons of production.