1. **INTRODUCTION**

**AGRICULTURE:**

Agriculture has always been one of the vital occupations that serve mankind, both in terms of livelihood and employment. Due to the substantial increase in the population, the nutritional status of the poor is growing bad, which must be improved. The major effect of population increase has been prominently shown on the environment, the damage of which is increasing rapidly, which ultimately hinders agricultural production. Studies show that the modern techniques used in agriculture have not been environment-friendly, though they are technologically advanced than the primitive techniques. The past achievements in the field of agriculture clearly depict the power and ability of man being able to meet the agricultural demand in spite of the population growth. Hence, a balanced relationship between the nature’s major creations i.e. human beings and their environment has to be maintained in order to lead a sustainable life. It clearly presents the contribution of agriculture to the national income and its share in export for a period of 50 years. Now agriculture contributes only about one-third to the national income as against 54% in 1950-51. Similarly, the share of agricultural goods in export has declined from 52.5% in 1950-51 to only 16.5% in 1990-91. To increase the agricultural contribution to the national income; the production of crops should be increased. Yield prediction is one of the most critical issues faced in the agricultural sector. Farmer’s lack of knowledge about harvest glut, uncertainties in the weather conditions and seasonal rainfall policies, depletion of nutrition level of soils, fertilizer availability and cost, pest control, post–harvest loss and other factors leads to decrease in the production of the crops.

During the monsoon season of the year 2011, Indian agriculture achieved an all-time record of producing 85.9 million tons of wheat, an increase of 6.4% from the previous year. During the same period, rice production showed an increase of 7%, hitting a new record of producing 95.3 million tons. India exported $39 billion worth of agricultural products in 2013, making it the seventh largest agricultural exporter worldwide and the sixth largest net exporter. According the survey conducted in 2013, India stands second in farm output. Also, Agriculture, forestry and fisheries made up 13.7% of the total GDP (Gross Domestic product). Because of the broad based and fast economic growth in India, agriculture’s contribution to the total GDP is declining. But, agricultural sector plays a significant role in the socio-economic growth of India and is still the broadest economic sector. India exported $39 billion worth of agricultural products in 2013, making it the seventh largest agricultural exporter worldwide and the sixth largest net exporter. India is now one among the world’s largest suppliers of rice, wheat, cotton and sugar. It has exported over 2 million metric tons of wheat and 2.1 million metric tons of rice to most of the Asian and African countries. In the production of dry fruits, agro-based textile raw materials, roots and tuber crops, pulses, farmed fish, egg, coconut, sugarcane and numerous vegetables, India stands second or third in the world. In 2010, India stood one among the top 5 world’s largest producers of over 80% agricultural products like coffee and cotton.

Agriculture is the prime source of income in India. About 54%of the income in India raises from agricultural sector and about 50% – 60% of the population depends on agriculture. The production of crop decides its price. Most of the agricultural products are even more exported to foreign countries. So the productivity of the crop is essential. Andhra Pradesh is one such state in India where about 60% – 70% of the population depends on irrigation sector. The state is known for food production especially rice. The yield of crop is based on the climatic conditions and the extent of rainfall in the area. Hence, forecasting of crop yield depending on the seasonal forecasts helps in minimizing the loss due to effect of climate variability and extremes under present-day climatic conditions [1].

The climatic conditions differ from place to place and state to state in India resulting in variation of productivity. This indicates the importance of grain yield monitors. Especially, in the state of Andhra Pradesh also called as “Annapurna” means the goddess of food grains has some regions where the Rice is produced for 3 times a year, in some regions for 2 times a year and few regions in which for only 1 time a year. As the departed state of Andhra Pradesh consists of 13 states in which about 9 states are in the coastal belt to the Bay of Bengal which is prone to 3 cyclones during October, November and December months every year apart from the seasonal rainfalls occurring during June to September every year. Hence, the need for dynamic yield prediction based on rainfall is essential in Andhra Pradesh where the sometimes there might be severe rainfalls due to which the crop might get washed away and sometimes there shall be severe droughts where the crop might get dried away due to lack of water wherein the Rice crop needs huge water during initials days of cultivation. Thus, the need for yield prediction is very much essential to plan for next crop season. Exploring the literature, the research work has been done widely based on various computational intelligence [2] techniques to predict the yield.

The major essential parameters for yield prediction are Area Under Cultivation (AUC), Annual Rainfall (AR). Therefore in this, the parameter AUC, AR are used to compute the productivity i.e. yield predication.

Regression Analysis can be defined as a structured approach which stresses on the analysis of data for the research purpose on decision making and problem solving. There are problems/situations that require simultaneous analysis of multiple variables or objects for efficient decision making. We consider various factors like Area under Cultivation (AUC), Annual Rainfall (AR) and Food Price Index (FPI) that contributes to the Indian Journal of Science and Technology Prediction of Crop Yield using Regression Analysis yield of crop.

* 1. **Requirements specification**

**1.2.1 Hardware Specification**

Hard-disk : 1TB

RAM : 4 GB

Processor : I3 Or Above Processor

**1.2.2 Software Specification**

Operating System : Windows 10

Technology : Python 3

**2. LITERATURE SURVEY**

**2.1 Related Work**

Exploring the literature, the research work has been done widely based on various computational intelligence techniques to predict the yield [3-8]. Literature also shows that the early work has used even Self organizing maps(SOM) [9] to predict the yield which are suitable for multivariate statistic problems [10], K-nearest neighbor [11], Machine Learning Techniques and Deep Learning Techniques [12-13] suggested that the Deep Learning Techniques are more advantageous for the reason that we can overcome overfitting problem.

Also, the parameters considered for the yield prediction are Soil moisture, Soil fertility, Other soil parameters, Images acquired on extent of region under cultivation, Normalized Water index (NWI)[14], Nitrogen extent in soil [15-18]. The major essential parameters for yield prediction are Area Under Cultivation (AUC), Annual Rainfall (AR), and Food Price India (FPI).

Machine learning techniques are most reliable for solving the problems where relationship between the input variables and the output variables is hard to obtain or is not known [19].

Hence in this project, we would like to propose Multiple Linear Regression Model to predict the crop yield.

**Existing System**

* The agriculture office people go to fields and take some random samples and calculate the yield of the samples
* They take the average of all those samples and estimate the approximate yield.
* They carry out the same process in different areas in a place (district) and calculate the approximate average yield on the whole for that place.
* Yield prediction is being done manually based on the running average at present.
* Human intervention leads to errors when large amount of data is involved. So we are proposing an automated system.

**Proposed System**

* The proposed system considers the yield of the previous years in the specific areas.
* Based on the past data considered we predict the yield that we can expect in the coming year
* Based on this prediction we can in prior take some measures in order to reduce the loss.
* The different fields that we are going to consider in this prediction process are water sources, area under irrigation.
* The prediction details will be sent to the marketing department.

1. **SYSTEM DESIGN**

**3.1 System Diagram**

The data flow diagram is one of the most important tools used for the system analysis. ADEMACRO (1978) and SARSON (1979) populated the use of data flow diagrams as modelling tool through their structured analysis methodologies. They suggested that a data flow diagram should be the first tool used by the analyst to model the system components. There are four types of system components. They are Data flow diagram is a graphical tool used to describe analyze the movement of data through a system manual or automated including the processes, stores of data, and delays in the system.

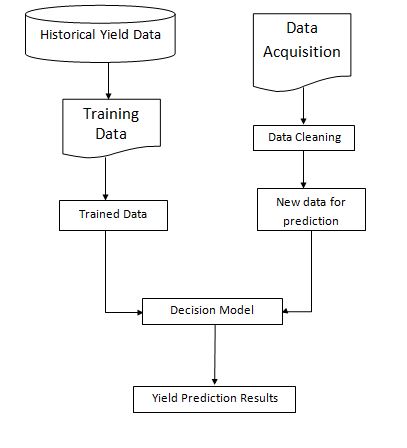


Fig-1: Proposed System Diagram

**Yield Data and Training**

* The historical data is taken for analysis.
* The data is considered to be the training data
* The trained data is given to the decision model as an input.

**Data Acquisition and cleaning**

* Data acquisition involves gathering the current data .
* The data we gathered is not complete.
* In Order to get a meaningful data we need to perform data cleaning.
* The data cleaning process removes the gaps in the data.

**Decision Model**

* The trained data contains different patterns corresponding to different conditions.
* The current data is mapped to the related trained data and the prediction results are produced

## **3.2 TOOLS USED**

**Python:** Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

**Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

**Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

**Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

**Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python has a big list of good features, few are listed below −

It supports functional and structured programming methods as well as OOP.

It can be used as a scripting language or can be compiled to byte-code for building large applications.

It provides very high-level dynamic data types and supports dynamic type checking.

It supports automatic garbage collection.

**Python Installation:**

Python distribution is available for a wide variety of platforms. You need to download only the binary code applicable for your platform and install Python.

If the binary code for your platform is not available, you need a C compiler to compile the source code manually. Compiling the source code offers more flexibility in terms of choice of features that you require in your installation.

### Windows Installation

Here are the steps to install Python on Windows machine.

Open a Web browser and go to <https://www.python.org/downloads/>.

Follow the link for the Windows installer *python-XYZ.msi* file where XYZ is the version you need to install.

To use this installer *python-XYZ.msi*, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.

Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

## **Setting up PATH**

Programs and other executable files can be in many directories, so operating systems provide a search path that lists the directories that the OS searches for executables.

The path is stored in an environment variable, which is a named string maintained by the operating system. This variable contains information available to the command shell and other programs.

The **path** variable is named as PATH in Unix or Path in Windows (Unix is case sensitive; Windows is not).

## **Setting path at Windows**

To add the Python directory to the path for a particular session in Windows −

**At the command prompt** − type path %path%;C:\Python and press Enter.

## **Running Python**

There are three different ways to start Python −

### Interactive Interpreter

You can start Python from Unix, DOS, or any other system that provides you a command-line interpreter or shell window.

Enter **python** the command line.

Start coding right away in the interactive interpreter.

$python # Unix/Linux

or

python% # Unix/Linux

or

C:> python # Windows/DOS

Here is the list of all the available command line options −

|  |  |
| --- | --- |
| **Sr.No.** | **Option & Description** |
| 1 | **-d**  It provides debug output. |
| 2 | **-O**  It generates optimized bytecode (resulting in .pyo files). |
| 3 | **-S**  Do not run import site to look for Python paths on startup. |
| 4 | **-v**  verbose output (detailed trace on import statements). |
| 5 | **-X**  disable class-based built-in exceptions (just use strings); obsolete starting with version 1.6. |
| 6 | **-c cmd**  run Python script sent in as cmd string |
| 7 | **File**  run Python script from given file |

### Script from the Command-line

A Python script can be executed at command line by invoking the interpreter on your application, as in the following −

$python script.py # Unix/Linux

or

python% script.py # Unix/Linux

or

C: >python script.py # Windows/DOS

**Note** − Be sure the file permission mode allows execution.

### Integrated Development Environment

You can run Python from a Graphical User Interface (GUI) environment as well, if you have a GUI application on your system that supports Python.

* **Unix** − IDLE is the very first Unix IDE for Python.
* **Windows** − PythonWin is the first Windows interface for Python and is an IDE with a GUI.
* **Macintosh** − The Macintosh version of Python along with the IDLE IDE is available from the main website, downloadable as either MacBinary or BinHex'd files.

If you are not able to set up the environment properly, then you can take help from your system admin. Make sure the Python environment is properly set up and working perfectly fine.

**Pandaas:**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

**Installation Procedure:**

Standard Python distribution doesn't come bundled with Pandas module. A lightweight alternative is to install NumPy using popular Python package installer, **pip.**

pip install pandas

**Numpy:**

NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

**Numeric**, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open source project.

**Installation Procedure:**

Standard Python distribution doesn't come bundled with NumPy module. A lightweight alternative is to install NumPy using popular Python package installer, **pip**.

pip install numpy

**Scikit-Learn:**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

**NumPy**: Base n-dimensional array package

**SciPy**: Fundamental library for scientific computing

**Matplotlib**: Comprehensive 2D/3D plotting

**IPython**: Enhanced interactive console

**Sympy**: Symbolic mathematics

**Pandas**: Data structures and analysis

Extensions or modules for SciPy care conventionally named [SciKits](http://scikits.appspot.com/scikits). As such, the module provides learning algorithms and is named scikit-learn.

The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.

Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations, [LAPACK](http://www.netlib.org/lapack/), [LibSVM](http://www.csie.ntu.edu.tw/~cjlin/libsvm/) and the careful use of cython.

**Installation Procedure:**

Scikit-learn requires:

* Python (>= 2.7 or >= 3.4),
* NumPy (>= 1.8.2),
* SciPy (>= 0.13.3).

If you already have a working installation of numpy and scipy, the easiest way to install scikit-learn is using pip

pip install -U scikit-learn

**SkLearn preprocessing OneHotEncoder**

One-Hot Encoding transforms each categorical feature with n possible values into n binary features, with only one active.

Most of the ML algorithms either learn a single weight for each feature or it computes distance between the samples. Algorithms like linear models (such as logistic regression) belongs to the first category.

Lets take a look at an example from loan\_prediction data set. Feature Dependents have 4 possible values 0,1,2 and 3+ which are then encoded without loss of generality to 0,1,2 and 3.

We, then have a weight “W” assigned for this feature in a linear classifier,which will make a decision based on the constraints W\*Dependents + K > 0 or eqivalently  W\*Dependents < K.

Let f(w)= W\*Dependents

Possible values that can be attained by the equation are 0, W, 2W and 3W. A problem with this equation is that the weight “W” cannot make decision based on four choices. It can reach to a decision in following ways:

* All leads to the same decision (all of them <K or vice versa)
* 3:1 division of the levels (Decision boundary at f(w)>2W)
* 2:2 division of the levels (Decision boundary at f(w)>W)

Here we can see that we are loosing many different possible decisions such as the case where “0” and “2W” should be given same label and “3W” and “W” are odd one out.

This problem can be solved by One-Hot-Encoding as it effectively changes the dimensionality of the feature “Dependents” from one to four, thus every value in the feature “Dependents” will have their own weights. Updated equation for the decison would be f'(w) < K.

where,  f'(w) = W1\*D\_0 + W2\*D\_1 + W3\*D\_2 + W4\*D\_3

All four new variable has boolean values (0 or 1).

The same thing happens with distance based methods such as kNN. Without encoding, distance between “0” and “1” values of Dependents is 1 whereas distance between “0” and “3+” will be 3, which is not desirable as both the distances should be similar. After encoding, the values will be new features (sequence of columns is 0,1,2,3+) : [1,0,0,0] and [0,0,0,1] (initially we were finding distance between “0” and “3+”), now the distance would be √2.

**SkLearn Label Encoding**:

sklearn provides a very efficient tool for encoding the levels of a categorical features into numeric values. LabelEncoder encode labels with value between 0 and n\_classes-1.

*# Importing LabelEncoder and initializing it*

>> from sklearn.preprocessing import LabelEncoder

>> le=LabelEncoder()

*# Iterating over all the common columns in train and test*

>> for col in X\_test.columns.values:

*# Encoding only categorical variables*

if X\_test[col].dtypes=='object':

*# Using whole data to form an exhaustive list of levels*

data=X\_train[col].append(X\_test[col])

le.fit(data.values)

X\_train[col]=le.transform(X\_train[col])

X\_test[col]=le.transform(X\_test[col])

All our categorical features are encoded. You can look at your updated data set using X\_train.head().

# **Graphical User Interfaces with Tkinter**

Tk/Tcl has long been an integral part of Python. It provides a robust and platform independent windowing toolkit, that is available to Python programmers using the [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) package, and its extension, the [tkinter.tix](https://docs.python.org/3/library/tkinter.tix.html#module-tkinter.tix) and the [tkinter.ttk](https://docs.python.org/3/library/tkinter.ttk.html#module-tkinter.ttk) modules.

The [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) package is a thin object-oriented layer on top of Tcl/Tk. To use [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter), you don’t need to write Tcl code, but you will need to consult the Tk documentation, and occasionally the Tcl documentation. [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) is a set of wrappers that implement the Tk widgets as Python classes. In addition, the internal module \_tkinterprovides a threadsafe mechanism which allows Python and Tcl to interact.

Tkinter’s chief virtues are that it is fast, and that it usually comes bundled with Python. Although its standard documentation is weak, good material is available, which includes: references, tutorials, a book and others. [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) is also famous for having an outdated look and feel, which has been vastly improved in Tk 8.5

Tkinter is Python’s default GUI library. It is based on the Tk toolkit, originally designed for the Tool Command Language (Tcl). Due to Tk’s popularity, it has been ported to a variety of other scripting languages, including Perl (Perl/Tk), Ruby (Ruby/Tk), and Python (Tkinter). The combination of Tk’s GUI development portability and flexibility along with the simplicity of a scripting language integrated with the power of systems language gives you the tools to rapidly design and implement a wide variety of commercial-quality GUI applications. Python, along with Tkinter, provides a fast and exciting way to build useful applications that would have taken much longer if you had to program directly in C/C++ with the native windowing system’s libraries. Once you have designed the application and the look and feel that goes along with your program, you will use basic building blocks known as widgets to piece together the desired. Once you get Tkinter up on your system, it will take less than 15 minutes to get your first GUI application running.

Tkinter is not necessarily turned on by default on your system. You can determine whether Tkinter is available for your Python interpreter by attempting to import the Tkinter module (in Python 1 and 2; renamed to tkinter in Python 3). If Tkinter is available, then no errors occur, as demonstrated in the following: >>> import tkinter >>> If your Python interpreter was not compiled with Tkinter enabled, the module import fails. You might need to recompile your Python interpreter to gain access to Tkinter. This usually involves editing the Modules/Setup file and then enabling all the correct settings to compile your Python interpreter with hooks to Tkinter, or choosing to have Tk installed on your system.

Getting Tkinter Installed and Working on the RPi

Type the following line into a terminal window:

sudo apt-get install python-tk

Open a Python Shell:

idle3

Import the Tkinter module: >>> import tkinter

First, it is not necessary to have an application already. You can create a pure GUI if you want, but it probably isn’t too useful without some underlying software that does something interesting. There are basically five main steps that are required to get your GUI up and running:

1. Import the Tkinter module (or from Tkinter import \*).

2. Create a top-level windowing object that contains your entire GUI application.

3. Build all your GUI components (and functionality) on top (or within) of your top-level windowing object.

4. Connect these GUI components to the underlying application code.

5. Enter the main event loop.

TKINTER WIDGETS 25 Tk Widgets Widget Description Button Similar to a Label but provides additional functionality for mouse-overs, presses, and releases, as well as keyboard activity/events Canvas Provides ability to draw shapes (lines, ovals, polygons, rectangles); can contain images or bitmaps Checkbutton Set of boxes, of which any number can be “checked” Entry Single-line text field with which to collect keyboard input Frame Pure container for other widgets Label Used to contain text or images LabelFrame Combo of a label and a frame but with extra label attributes Listbox Presents the user with a list of choices from which to choose Menu Actual list of choices “hanging” from a Menubutton from which the user can choose Menubutton Provides infrastructure to contain menus (pulldown, cascading, etc.) Message Similar to a Label, but displays multiline text PanedWindow A container widget with which you can control other widgets placed within it Radiobutton Set of buttons, of which only one can be “pressed” Scale Linear “slider” widget providing an exact value at current setting; with defined starting and ending values Scrollbar Provides scrolling functionality to supporting widgets, for example, Text, Canvas, Listbox, and Entry Spinbox Combination of an entry with a button letting you adjust its value Text Multiline text field with which to collect (or display) text from user Toplevel Similar to a Frame, but provides a separate window container.

1. **Code Module**

**Regression Analysis:**

Regression analysis is a reliable method of identifying which variables have impact on a topic of interest. The process of performing a regression allows you to confidently determine which factors matter most, which factors can be ignored, and how these factors influence each other.

In order to understand regression analysis fully, it’s essential to comprehend the following terms:

**Dependent Variable:** This is the main factor that you’re trying to understand or predict.

**Independent Variables:** These are the factors that you hypothesize have an impact on your dependent variable.

In order to conduct a regression analysis, you’ll need to define a dependent variable that you hypothesize is being influenced by one or several independent variables

Plotting your data is the first step in figuring out if there is a relationship between your independent and dependent variables

Our dependent variable should be plotted on the y-axis, while our independent variable should be plotted on the x-axis.

Regression Analysis is used to establish the relationship among these 3 factors and to identify their influence on crop yield. Regression Analysis is a commonly used technique in the research where relationship among the three considered variables (AUC, AR) has to be established and to identify their effects on crop yield. Crop yield is considered as a dependent variable and AUC, AR, are considered as independent variables.

Regression Analysis is used to find the relative strength between a dependent variable and an independent variable i.e. impact of AUC on Yield, AR on yield and FPI on yield. The crop considered for analysis is rice because it is the most common crop cultivated in many areas of India.

**Multiple Linear Regression:**

Multiple linear regression assumes

* Normality
* Homogeneity of Variance
* Fixed X (X represents explanatory variables)
* Independence

**Train/Test**

The model is trained to predict the known outputs and later tested using test data and applied to generalize other non-trained data. Test data is used to test the prediction ability (accuracy) of the model. Training data (X\_train,y\_train) is used to fit the regression model(make a linear model).This model is used to predict data from independent variables.

Train/Test split is not enough to guarantee the randomness of the samples. If samples fail to be random, this might result in overfitting. Overfitting means the model is “too well trained”, although it cannot be applied to other data. Overfitting happens when the model uses too many predictors; while it works too well on the training set, it fails on new untrained data. This means we cannot make inferences from our model.

**Computation:**

A linear regression model that contains more than one predictor variable is called a *multiple linear regression model*. The following model is a multiple linear regression model with two predictor variables, http://reliawiki.org/images/math/7/2/e/72e6d89b6fb959d74d6d5c5be0a47c5a.png and http://reliawiki.org/images/math/e/a/a/eaad754033d6b926959a47c7c5c999b3.png.

http://reliawiki.org/images/math/1/3/3/1330581a877c4dc631b00c24577f5293.png (1)

This regression model is a first order multiple linear regression model. This is because the maximum power of the variables in the model is 1. (The regression plane corresponding to this model is shown in the figure below.) Also shown is an observed data point and the corresponding random error, http://reliawiki.org/images/math/2/2/3/223a19789dba17194e2f96e9c37b87d1.png. The true regression model is usually never known (and therefore the values of the random error terms corresponding to observed data points remain unknown). However, the regression model can be estimated by calculating the parameters of the model for an observed data set. This is explained in [Estimating Regression Models Using Least Squares](http://reliawiki.org/index.php/Multiple_Linear_Regression_Analysis#Estimating_Regression_Models_Using_Least_Squares).

One of the following figures shows the contour plot for the regression model the above equation. The contour plot shows lines of constant mean response values as a function of x2 and x1. The contour lines for the given regression model are straight lines as seen on the plot. Straight contour lines result for first order regression models with no interaction terms.

A linear regression model may also take the following form:

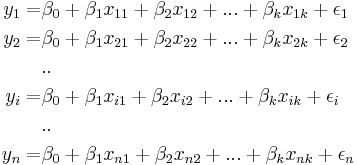
http://reliawiki.org/images/math/4/0/a/40a806d1d12f81d6f2ce8878536742bc.png (2)

**Estimating Regression Models Using Least Squares**

Consider a multiple linear regression model with http://reliawiki.org/images/math/a/2/d/a2df1f34f2b5c43bdc70e2462762b190.png predictor variables:

http://reliawiki.org/images/math/0/4/7/047b6637704da980c76306ebf9400781.png (3)

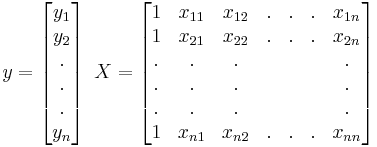
Let each of the http://reliawiki.org/images/math/a/2/d/a2df1f34f2b5c43bdc70e2462762b190.png predictor variables, http://reliawiki.org/images/math/7/2/e/72e6d89b6fb959d74d6d5c5be0a47c5a.png, http://reliawiki.org/images/math/e/a/a/eaad754033d6b926959a47c7c5c999b3.png... http://reliawiki.org/images/math/f/e/4/fe4d9f705149061aeca82fa1b65b29fc.png, have http://reliawiki.org/images/math/b/a/a/baa52b85c066dbd5eeff3c078a69205b.png levels. Then http://reliawiki.org/images/math/7/c/5/7c59c310114c31ca2ced639bdd2a78a9.png represents the http://reliawiki.org/images/math/a/7/9/a796b40d92e81ae190a1e4f4e2a2c3ed.png th level of the http://reliawiki.org/images/math/b/4/0/b40669fa7371edf874815830e563b2ce.png th predictor variable http://reliawiki.org/images/math/0/d/6/0d62f8e16275477fc37a781f0b3e55f3.png. For example, http://reliawiki.org/images/math/5/0/4/50431ff09f61842833c459e4c8cc3b04.png represents the fifth level of the first predictor variable http://reliawiki.org/images/math/7/2/e/72e6d89b6fb959d74d6d5c5be0a47c5a.png, while http://reliawiki.org/images/math/f/e/1/fe1d6a96a5e530af07c74ccd388ce880.pngrepresents the first level of the ninth predictor variable, http://reliawiki.org/images/math/f/e/8/fe8547765f3148babbc1df1af2627fcc.png. Observations, http://reliawiki.org/images/math/8/c/6/8c650234ff4810d6c5bd0352117be27e.png, http://reliawiki.org/images/math/f/7/d/f7d1ec6f032e746d91d6890f1c668986.png... http://reliawiki.org/images/math/6/3/0/630e1f8c3961a948ad3a06ffb62c9862.png, recorded for each of these http://reliawiki.org/images/math/b/a/a/baa52b85c066dbd5eeff3c078a69205b.png levels can be expressed in the following way:

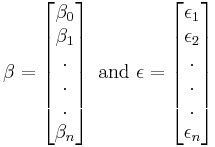
 (4)

The system of http://reliawiki.org/images/math/b/a/a/baa52b85c066dbd5eeff3c078a69205b.png equations shown previously can be represented in matrix notation as follows:

http://reliawiki.org/images/math/9/b/f/9bf89d25ce96ee8295f5ac5dec072eb5.png (5)

where

 (6)

 (7)

The matrix http://reliawiki.org/images/math/c/0/d/c0d0bafa786ddace7cb3a8a209484454.png is referred to as the *design matrix*. It contains information about the levels of the predictor variables at which the observations are obtained. The vector http://reliawiki.org/images/math/5/b/3/5b320b6d3d3254d936c752ae308dbfd8.png contains all the regression coefficients. To obtain the regression model, http://reliawiki.org/images/math/5/b/3/5b320b6d3d3254d936c752ae308dbfd8.png should be known. http://reliawiki.org/images/math/5/b/3/5b320b6d3d3254d936c752ae308dbfd8.png is estimated using least square estimates. The following equation is used:

http://reliawiki.org/images/math/f/f/b/ffbf405ec56cf0430ee1c45c04983368.png (8)

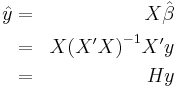
where http://reliawiki.org/images/math/2/6/e/26ed1354f274036ac3ecd17ebd042fd5.png represents the transpose of the matrix while http://reliawiki.org/images/math/1/d/e/1de6b04ea78562291eceaeab2c3b51b8.png represents the matrix inverse. Knowing the estimates, http://reliawiki.org/images/math/f/0/7/f075f0fde79709f067f3f71349f4745b.png, the multiple linear regression model can now be estimated as:

http://reliawiki.org/images/math/0/7/1/0716ada40cec56bfd0abca2df1e9adac.png (9)

The estimated regression model is also referred to as the *fitted model*. The observations, http://reliawiki.org/images/math/c/4/5/c45e804fbf9c218111a1ae1d249a64b3.png, may be different from the fitted values http://reliawiki.org/images/math/1/c/3/1c32a91e3ccfe166de7e554aedeca78b.png obtained from this model. The difference between these two values is the residual, http://reliawiki.org/images/math/4/7/4/474902daa046ce38f108c5106bf77670.png. The vector of residuals, http://reliawiki.org/images/math/4/6/0/460a1940ceddf45878d2e095af31128a.png, is obtained as:

http://reliawiki.org/images/math/4/4/f/44f0c789a94f11fd616a51be00990531.png (10)

The fitted model can also be written as follows, using http://reliawiki.org/images/math/f/f/b/ffbf405ec56cf0430ee1c45c04983368.png:

 (11)

where http://reliawiki.org/images/math/2/a/b/2abc58fcc6b19b7673d2a5caffa87b69.png. The matrix, http://reliawiki.org/images/math/2/c/6/2c629235daf48b8ccce8241024fd3195.png, is referred to as the hat matrix. It transforms the vector of the observed response values, http://reliawiki.org/images/math/b/f/b/bfb6488d6c250ac5aeed1bbf139baaa5.png, to the vector of fitted values, http://reliawiki.org/images/math/8/a/4/8a4840345980b086bba6c6949711b504.png.

**Properties of the Least Square Estimators for Beta**

The least square estimates, http://reliawiki.org/images/math/8/d/9/8d96d2248ddf1b7d28ae037ae2511ee6.png, http://reliawiki.org/images/math/5/0/1/5012b550502942d968fbfc1022b15f32.png, http://reliawiki.org/images/math/2/4/9/24924dfb246c4ca4aec5f52cd089b01d.png... http://reliawiki.org/images/math/e/8/5/e858d105aa6b86b7b34783dd95c139e9.png, are unbiased estimators of http://reliawiki.org/images/math/9/6/8/968b272cf0f668e88d34ffd67022b89d.png, http://reliawiki.org/images/math/3/f/9/3f912a9f94bb8705de28c11fb1dcb786.png, http://reliawiki.org/images/math/3/e/5/3e54332768980a30a9aade00e30c19de.png... http://reliawiki.org/images/math/e/9/c/e9c91ca45d1e34eaf08df7091c443b00.png, provided that the random error terms, http://reliawiki.org/images/math/b/d/0/bd0ec05f4874b4d8ab555f7a157720f8.png, are normally and independently distributed. The variances of the http://reliawiki.org/images/math/f/0/7/f075f0fde79709f067f3f71349f4745b.png s are obtained using the http://reliawiki.org/images/math/b/e/c/bec42ce6fbc154eaf9925819fd0da9c7.png matrix. The variance-covariance matrix of the estimated regression coefficients is obtained as follows:

http://reliawiki.org/images/math/8/9/e/89e6eb770cebb4ae689de45dd5085290.png (12)

http://reliawiki.org/images/math/3/6/a/36a0396f882f0f9260ed9c6b3a3a07a9.png is a symmetric matrix whose diagonal elements, http://reliawiki.org/images/math/3/6/1/361a42068073bfd5521cc11965ab6cff.png, represent the variance of the estimated http://reliawiki.org/images/math/b/4/0/b40669fa7371edf874815830e563b2ce.png th regression coefficient, http://reliawiki.org/images/math/8/8/2/8824ad81a94bf03dd9b4398e7251eccb.png. The off-diagonal elements, http://reliawiki.org/images/math/c/d/0/cd0ffdec0d0ccf492adeb899a8e6b2ee.png, represent the covariance between the http://reliawiki.org/images/math/a/7/9/a796b40d92e81ae190a1e4f4e2a2c3ed.png th and http://reliawiki.org/images/math/b/4/0/b40669fa7371edf874815830e563b2ce.png th estimated regression coefficients, http://reliawiki.org/images/math/8/c/a/8ca8f461183464cce2a37446fcb9335c.png and http://reliawiki.org/images/math/8/8/2/8824ad81a94bf03dd9b4398e7251eccb.png. The value of http://reliawiki.org/images/math/7/7/0/7702229ff5898a61e539bf8d47ddb015.png is obtained using the error mean square, http://reliawiki.org/images/math/f/3/0/f30afa85183798c09d35d4aa3cbbd076.png.

**Source Code:**

import pandas as pd

import numpy as np

import tkinter as tk

import statsmodels.api as sm

#Creating a tkinter window for taking input

root1=tk.Tk()

canvas2 = tk.Canvas(root1,width=270,height=150)

canvas2.pack()

# Dropdownbox for State and crop

OPTIONS1 = [

'Andhra Pradesh',

'Assam',

'Bihar',

'Chattisgarh',

'Gujarat',

'Haryana',

'Himachal Pradesh',

'Jammu and Kashmir',

'Jharkhand',

'Karnataka',

'Kerala',

'Madhya Pradesh',

'Maharashtra',

'Orissa',

'Punjab',

'Rajasthan',

'Tamil Nadu',

'Uttar Pradesh',

'UttaraKhand',

'West Bengal'

] #etc

OPTIONS2 = [

'Rice',

'Wheat',

'Maize',

'Small Millet',

'Gram'

]

variable1 = tk.StringVar(root1)

variable1.set(OPTIONS1[0]) # default value

variable2 = tk.StringVar(root1)

variable2.set(OPTIONS2[0])

label3 = tk.OptionMenu(root1, variable1, \*OPTIONS1)

label4 = tk.OptionMenu(root1, variable2, \*OPTIONS2)

label3.pack()

label4.pack()

New\_State=''

New\_Crop=''

#Taking input of state and crop

def ok():

#our 1st input

global New\_State

New\_State = variable1.get()

#our 2nd input

global New\_Crop

New\_Crop = variable2.get()

button2 = tk.Button (root1, text='OK',command=ok, bg='orange') # button to call the 'values' command above

canvas2.create\_window(270, 150, window=button2)

#importing dataset

if New\_Crop == 'Rice' :

dataset=pd.read\_csv('Rice.csv')

elif New\_Crop == 'Wheat' :

dataset=pd.read\_csv('Wheat.csv')

elif New\_Crop == 'Maize' :

dataset=pd.read\_csv('Maize.csv')

elif New\_Crop == 'Small Millet' :

dataset=pd.read\_csv('Millet.csv')

else :

dataset=pd.read\_csv('Gram.csv')

df = pd.DataFrame(dataset,columns=['State','Rainfall','Area','Rice\_Yield'])

X = df[['State','Rainfall','Area']]

Y = df['Rice\_Yield']

#print(df['State'])

#categorical data

cat\_list = df['State']

encoded\_data, mapping\_index = pd.Series(cat\_list).factorize()

#print(encoded\_data)

#print(mapping\_index)

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

le\_x=LabelEncoder()

df['State']=le\_x.fit\_transform(df['State'])

onehotencoder=OneHotEncoder(categories='auto')

X=onehotencoder.fit\_transform(X).toarray()

#avoidng dummy variable

X=X[:,1:]

#print(np.squeeze(X).shape)

#print(X)

#df['State']=df['State'].astype(np.float64)

#print(New\_State)

#print(df['State'])

#Applying regression model

from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X,Y)

print('Intercept: \n', round(regressor.intercept\_,2))

#print('Coefficients: \n', regressor.coef\_)

# with statsmodels

X = sm.add\_constant(X) #adding a constant

model = sm.OLS(Y,X).fit()

predictions = model.predict(X)

#tkinter GUI

root=tk.Tk()

canvas1 = tk.Canvas(root,width=1200,height=450)

canvas1.pack()

#with sklearn

Intercept\_result = ('Intercept: ', round(regressor.intercept\_,2))

label\_Intercept = tk.Label(root, text=Intercept\_result, justify = 'center')

canvas1.create\_window(260, 220, window=label\_Intercept)

#with sklearn

'''Coefficients\_result = ('Coefficients: ', regressor.coef\_)

label\_Coefficients = tk.Label(root, text=Coefficients\_result, justify = 'center')

canvas1.create\_window(260, 240, window=label\_Coefficients)'''

# with statsmodels

'''print\_model = model.summary()

label\_model = tk.Label(root, text=print\_model, justify = 'center', relief = 'solid', bg='LightSkyBlue1')

canvas1.create\_window(800, 220, window=label\_model)'''

# New\_Area label and input box

label1 = tk.Label(root, text='Type Area: ')

canvas1.create\_window(100, 100, window=label1)

entry1 = tk.Entry (root) # create 1st entry box

canvas1.create\_window(270, 100, window=entry1)

# New\_Rainfall label and input box

label2 = tk.Label(root, text=' Type Rainfall: ')

canvas1.create\_window(120, 120, window=label2)

entry2 = tk.Entry (root) # create 2nd entry box

canvas1.create\_window(270, 120, window=entry2)

def values():

global New\_Area #our 3rd input variable

New\_Area = float(entry1.get())

#print(New\_Area)

global New\_Rainfall #our 4th input variable

New\_Rainfall = float(entry2.get())

#print(New\_Rainfall)

#print(New\_State)

#print(variable1.get())

state=mapping\_index.get\_loc(variable1.get())

#print(state)

x\_new=np.array([[state , New\_Rainfall , New\_Area]])

#print(x\_new.shape)

z = np.zeros((1, 289), dtype=x\_new.dtype)

x\_new=np.c\_[x\_new, z]

#x\_new=np.pad(x\_new,(0,289),'constant')

#print(x\_new.shape)

Prediction\_result = ('Predicted Yield: ', np.around(regressor.predict(x\_new),2))

label\_Prediction = tk.Label(root, text= Prediction\_result, bg='orange')

canvas1.create\_window(260, 280, window=label\_Prediction)

button1 = tk.Button (root, text='Predict Yield',command=values, bg='orange') # button to call the 'values' command above

canvas1.create\_window(270, 150, window=button1)

root.mainloop()

1. **TESTING AND DEPLOYMENT**

## **5.1 TESTING**

**SYSTEM TESTING**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**WHITE BOX TESTING**

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

**BLACK BOX TESTING**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Test Cases:**

1) The user input New\_State is of type String. While training the data we encoded the state values to numerals using label encoder and one hot encoder. So inorder to predict the values we need to encode the user input to its corresponding numeral and then we need give it as one of the parameters to the predict method.

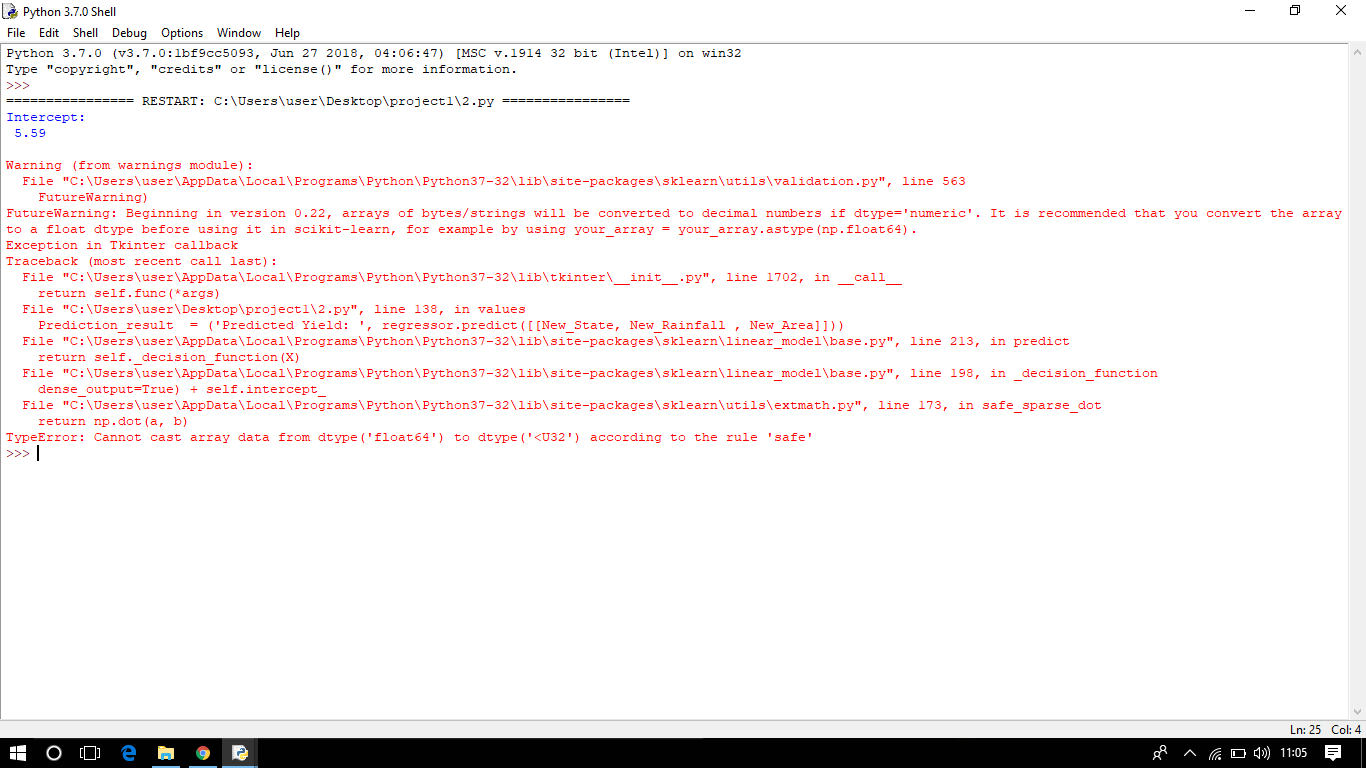


Fig-2: Error Screen depicting the data type error

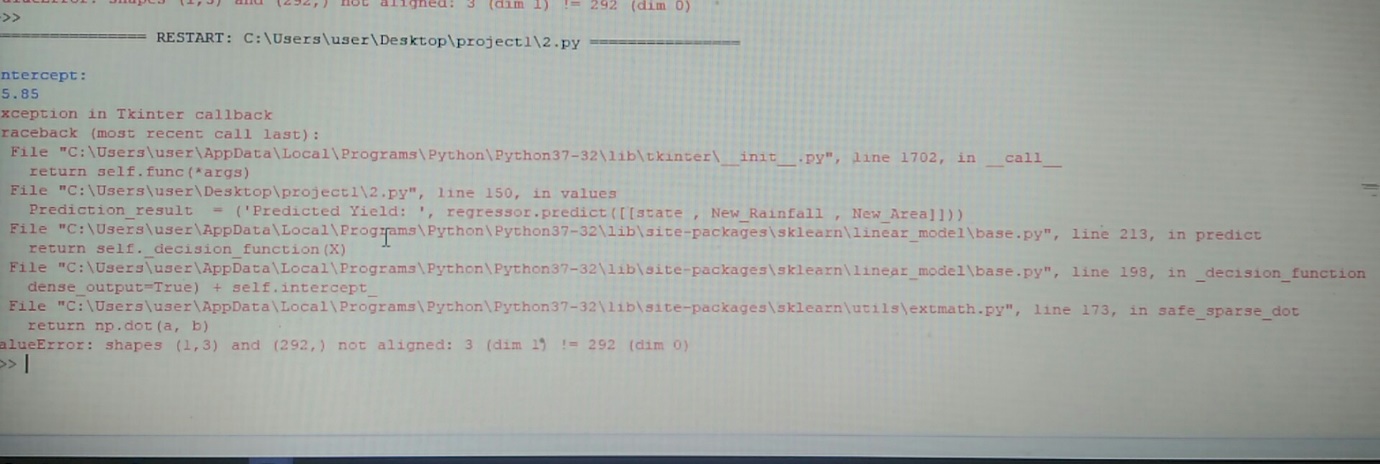


Fig-3: Error Screen depicting the dimensionality error

While performing the prediction the number of columns in training data set and test data set should be the same. As we performed one hot encoding, the numpy array shape increased in terms of the number of columns. To overcome this scenario, we padded the array with zeros.

**Outputs:**

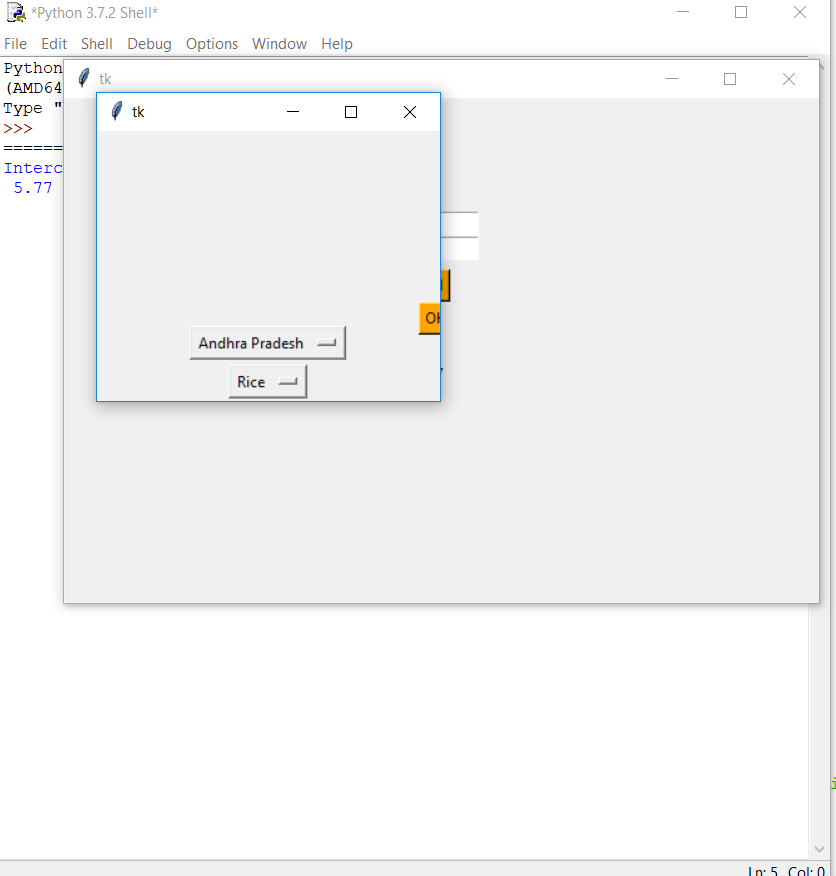


Fig-4: Output Screen for choosing the State and Crop as Option

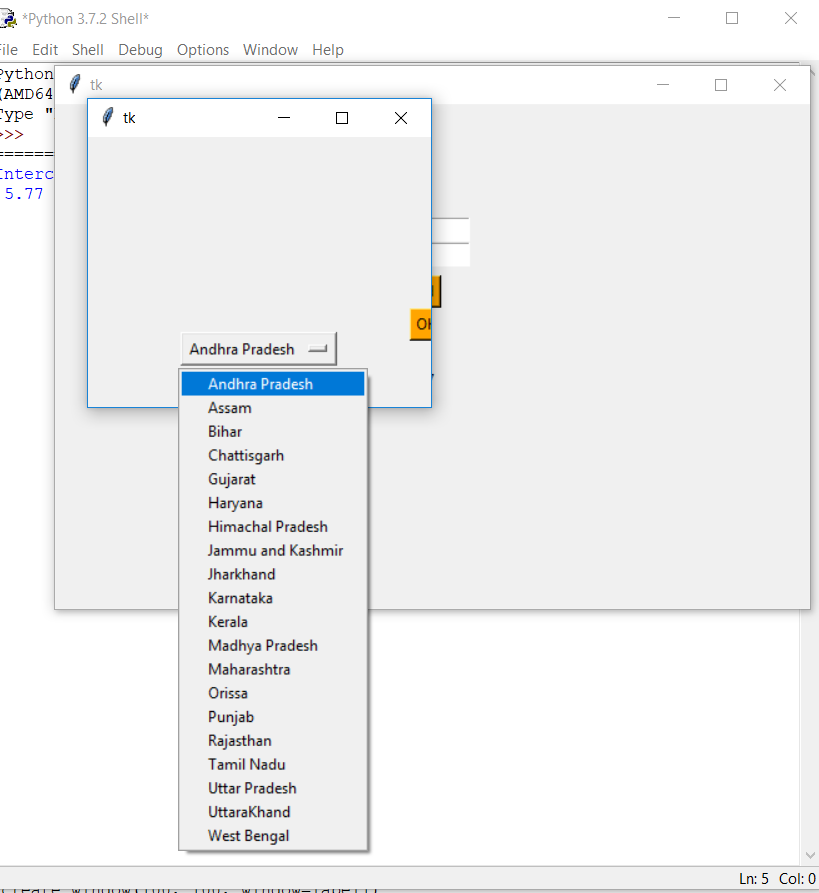


Fig-5: Output Screen for choosing the State as Option

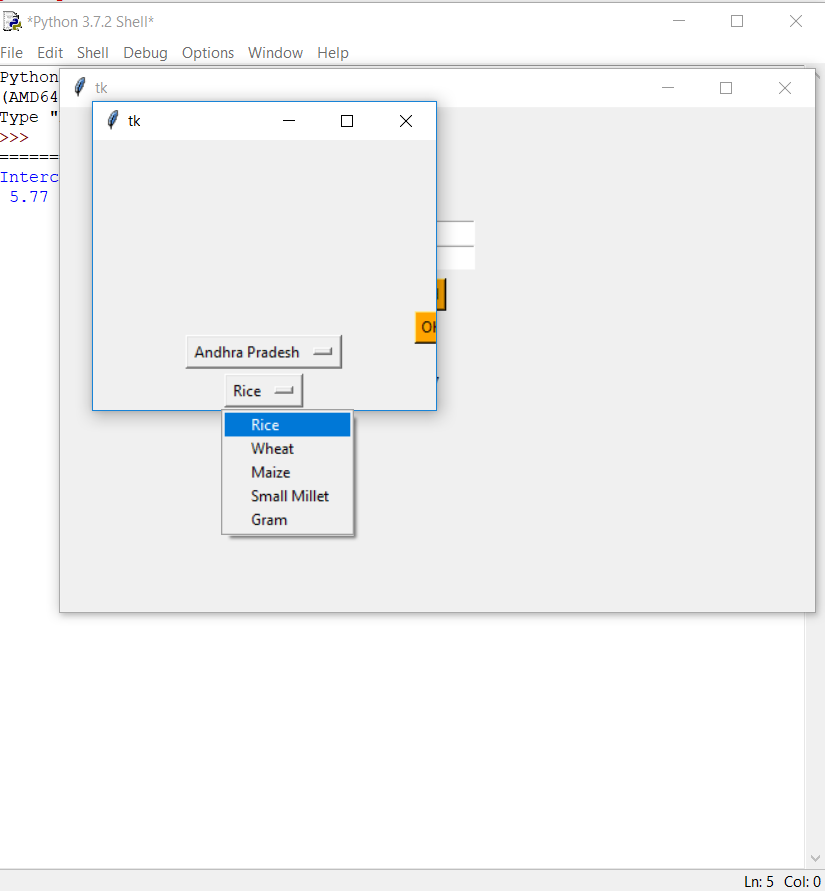


Fig-6: Output Screen for choosing the Crop as Option

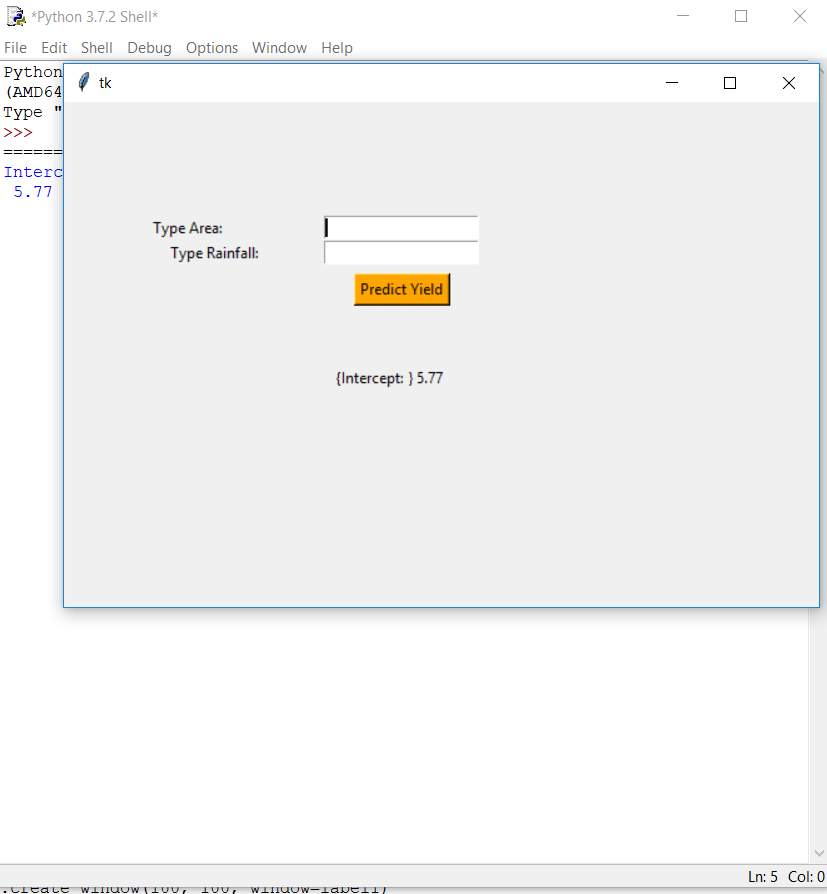


Fig-7: Screen to Input Extent and Rainfall data

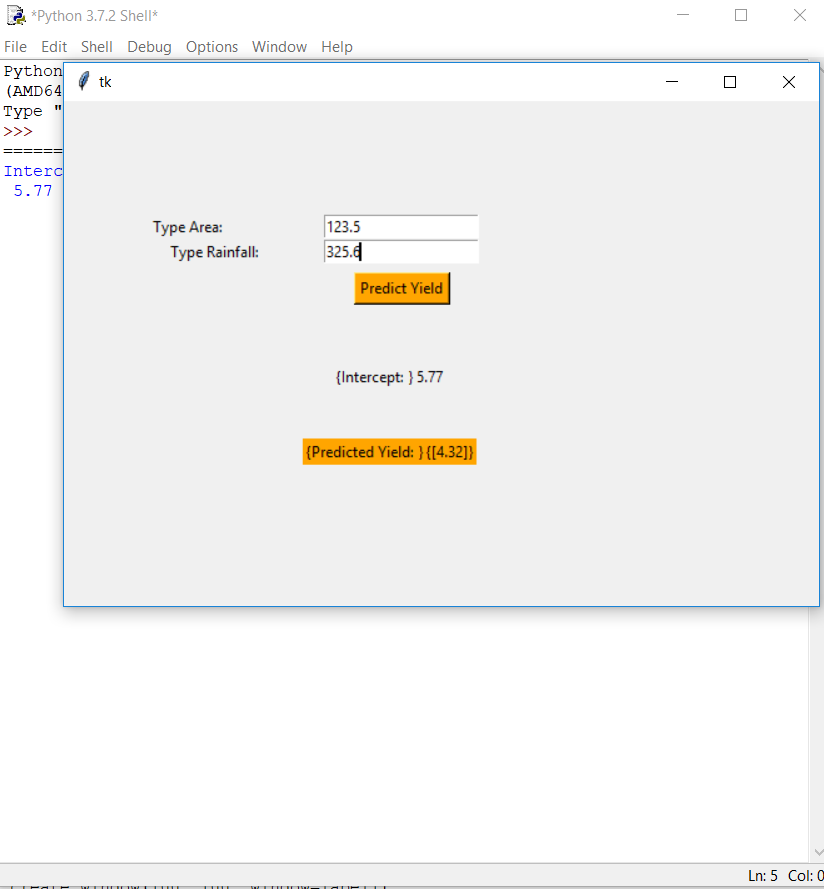


Fig-8: Screen After Giving Input of Extent and Rainfall data

**REFERENCES**

1. Toshichika et al, “Global Crop Yield forecasting using seasonal climate information from a multi-model ensemble”, Climate Services, 11 (2018), 13-23.
2. X.E.Pantazi et al, “Wheat yield prediction using machine learning and advanced sensing techniques” Computers and Electronics in Agriculture, Volume 121, February 2016, Pages 57-65.
3. V. Sellam and E. Poovammal, “Prediction of Crop Yield using Regression Analysis”, Indian Journal of Science and Technology, Vol 9(38), 10.17485/ijst/2016/v9i38/91714, October 2016.
4. P K Sahoo and Indra Mani, “Prospects of Precision Agriculture in India”, Advances in Agricultural Engineering, Annual Technical Volume-II, The Institution of Engineers India, Vol-2, 2018.
5. Singh K K. et al., “Crop Yield forecasting under FASAL”, FASAL Technical Report-2017, IMD-Delhi..
6. Miao, Y., Mulla, D.J., Robert, P.C., “Identifying important factors influencing corn yield and grain quality variability using artificial neural networks”, Precis. Agric. 7 (117), 135, 2006.
7. Effendi, Z., Ramli, R., Ghani, J.A., “A back propagation neural networks for grading Jatropha curcas fruits maturity”, Am. J. Appl. Sci. 7 (3), 390–394, ., 2010.
8. Fortin, J.G., Anctil, F., Parent, L.É., Bolinder, M.A., “A neural network experiment on the site-specific simulation of potato tuber growth in Eastern Canada”, Comput. Electr. Agric. 73 (2), 126–132, 2010.
9. Irmak, A., et al., “Artificial neural network model as a data analysis tool in precision farming”, Trans. ASABE 49 (6), 2027–2037, 2006.
10. Drummond, S.T., et al., “Statistical and neural methods for site-specific yield prediction”, Trans. ASAE 46 (1), 5–14, 2003.
11. Liu, J., Goering, C.E., Tian, L., “A neural network for setting target corn yields”, Trans. ASAE 44 (3), 705–713, 2001.
12. Ayoubi, S., Sahrawat, K.L., “Comparing multivariate regression and artificial neural network to predict barley production from soil characteristics in northern Iran”, Arch. Agron. Soil Sci. 57 (5), 549–565, 2011.
13. Norouzi, M., et al., “Predicting rainfed wheat quality and quantity by artificial neural network using terrain and soil characteristics”, Acta Agric. Scand. Sect. B – Soil Plant Sci. 60 (4), 341–352, 2010..
14. Zolfaghari, Z., Mosaddeghi, M.R., Ayoubi, S., “ANN-based pedotransfer and soil spatial prediction functions for predicting Atterberg consistency limits and indices from easily available properties at the watershed scale in western Iran”, Soil Use Manage. 31 (1), 142–154, 2015.
15. Shearer, S.A, et al., “Yield Prediction using a Neural Network Classifier Trained using Soil Landscape Features and Soil Fertility Data”, ASAE Paper No. 993042. St. Joseph, Michigan, USA, 1999.
16. Kohonen, T., “Self-Organization and Associative Memory”, Springer Verlag, Berlin, 1988.
17. Marini, F., “Artificial neural networks in food analysis: trends and perspectives”, Anal. Chim. Acta 635, 121–131, 2009.
18. Anna Chlingaryan, Salah Sukkarieh, Brett Whelan, “ Machine Learning approaches for Crop Yield prediction and nitrogen status estimation in precision agriculture: A Review”, Computer & Electronics in Agriculture, 151, 61-69, 2018.
19. Alberto Gonzalez et al, “Predictive ability of machine learning methods for massive crop yield prediction”, Spanish Journal of Agricultural Research, 12(2):313-328, 2014.
20. <https://data.gov.in/catalog/all-india-area-weighted-monthly-seasonal-and-annual-rainfall-mm>
21. <http://mospi.nic.in/statistical-year-book-india/2011/177>
22. <http://mospi.nic.in/statistical-year-book-india/2012/177>
23. <http://mospi.nic.in/statistical-year-book-india/2013/177>
24. <http://mospi.nic.in/statistical-year-book-india/2014/177>
25. <http://mospi.nic.in/statistical-year-book-india/2015/177>
26. <http://mospi.nic.in/statistical-year-book-india/2016/177><http://mospi.nic.in/statistical-year-book-india/2017/177>
27. <https://en.wikipedia.org/wiki/Agriculture_in_India>
28. <http://reliawiki.org/index.php/Multiple_Linear_Regression_Analysis>