Predicting Lumpy Skin Disease

Introduction

Lumpy Skin Disease (LSD) is a highly contagious viral disease that affects cattle and poses

a significant threat to livestock industries worldwide. Early detection and accurate

prediction of LSD outbreaks are crucial for effective disease control and prevention. In this

project, we aim to develop a machine learning model to predict the occurrence of Lumpy

Skin Disease using a dataset containing various geographical and environmental factors.

Dataset Description

The dataset used for this project includes the following columns:

- Longitude (X-axis spatial coordinates)
- Continent of the outbreak
- Latitude (Y-axis spatial coordinates)
- Monthly Cloud Cover in percent
- Diurnal Temperature Range in degrees Celsius
- Country of outbreak
- Frost Day Frequency in a month
- Potential Evapotranspiration in millimetres per day
- Precipitation in millimetres per month
- Daily Mean Temperature in degrees Celsius
- Temperature in degrees Celsius
- Monthly Average Maximum and Minimum Temperature in degrees Celsius
- Vapor Pressure in hectopascals
- Wet Day Frequency in days
- Altitude of geographic location in meters
- Dominant Land Cover
- Lumpy (target variable)

Project Flow:

1. Data Collection and Preparation

- Collect the dataset from reliable sources.
- Perform data cleaning and preprocessing.

2. Exploratory Data Analysis (EDA)

- Analyse the dataset using descriptive statistics and visualizations.
- Explore the distribution of variables and identify any patterns or trends.

3. Feature Engineering

- Extract relevant features from the dataset.
- Handle missing values and outliers, if any.
- Transform categorical variables into numerical representations, if required.

4. Model Building

- Split the dataset into training and testing sets.
- Train various machine learning models on the training set.
- Evaluate the performance of each model using appropriate evaluation metrics.
- Select the best-performing model for further analysis.

5. Model Evaluation

- Evaluate the optimized model on the testing set.
- Assess its predictive accuracy and reliability.

6. Model Deployment

- Deploy the final model to make predictions on new, unseen data.
- Develop a user-friendly interface or API for easy access to the model's predictions.

7. Documentation and Reporting

- Prepare a comprehensive project report documenting the entire process.
- Present the findings, insights, and conclusions derived from the project.
- Provide recommendations for further improvements or future research.

By accurately predicting the occurrence of Lumpy Skin Disease, this machine learning

project can significantly contribute to early detection and effective management of the

disease, ultimately leading to improved livestock health and the prevention of economic

losses in the livestock industry.

Milestone 1: Define Problem / Problem Understanding

Activity 1: Specify the Business Problem

The business problem for the accurate prediction of Lumpy Skin Disease is to develop a machine learning model that can effectively predict the occurrence of Lumpy Skin Disease in cattle. Lumpy Skin Disease is a highly contagious viral disease that affects cattle, causing significant economic losses in the livestock industry. By accurately predicting the disease

occurrence, proactive measures can be taken for disease control and prevention, reducing the spread and impact of Lumpy Skin Disease.

Activity 2: Business Requirements

To ensure that the Lumpy Skin Disease prediction model meets business requirements

and can be deployed effectively, the following rules and requirements need to be

considered:

1. Accuracy: The model should demonstrate a high level of accuracy in predicting the occurrence of Lumpy Skin Disease. It should provide reliable and precise predictions to support decision-making processes related to disease control and prevention.
2. Early Detection: The model should be able to detect the presence of Lumpy Skin Disease at an early stage to facilitate timely intervention and minimize the risk of disease spread within cattle populations.
3. Scalability: The model should be scalable to handle large volumes of data and accommodate future growth in the livestock industry. It should be capable of processing data from multiple sources and adapting to evolving disease patterns.
4. Interpretability: The model should be interpretable, meaning that its predictions can be explained and understood by stakeholders. Interpretability is essential for building trust in the model and enabling informed decision-making based on its outputs.

5. Privacy and Security: The model should adhere to privacy and security regulations to protect sensitive data. Measures should be implemented to ensure secure storage, handling, and access to data used for training and prediction purposes.

Activity 3: Literature Survey

A literature survey for the accurate prediction of Lumpy Skin Disease would involve researching and reviewing existing studies, articles, and publications related to Lumpy Skin Disease in cattle. The survey aims to gather insights on the following aspects:

1. Disease Characteristics: Understanding the aetiology, epidemiology, and clinical manifestations of Lumpy Skin Disease in cattle. Exploring factors that contribute to disease transmission and spread.

2. Risk Factors: Identifying risk factors associated with Lumpy Skin Disease, such as breed susceptibility, age, geographical location, and environmental conditions.

3. Diagnostic Methods: Reviewing existing diagnostic methods for Lumpy Skin Disease, including clinical observations, laboratory tests, and imaging techniques. Exploring their limitations and potential for improvement.

4. Machine Learning Approaches: Investigating previous studies that have utilized machine learning techniques for disease prediction in cattle. Assessing the performance of different algorithms and feature selection methods.

5. Data Availability: Identifying potential sources of data for training and validating the prediction model. Assessing the quality, completeness, and reliability of available datasets.

The literature survey will help in gaining a comprehensive understanding of Lumpy Skin Disease, its predictive modelling approaches, and the gaps in knowledge that can be addressed through this project.

Milestone 2: Data Collection & Preparation

Activity 1: Collect the Dataset

To develop an accurate prediction model for Lumpy Skin Disease, a comprehensive dataset related to the disease and cattle characteristics needs to be collected. The dataset should include relevant features that can contribute to the prediction of Lumpy Skin Disease occurrence. The following steps should be followed to collect the dataset:

Activity 1.1: Importing the libraries

Utilize the necessary software frameworks and dependencies as illustrated in the accompanying visual representation, in order to facilitate the successful implementation of this machine learning endeavour.

Activity 1.2: Dataset Reading

The dataset provided may be in various formats such as .csv, Excel files, .txt, .json, among others. To effectively process the dataset, we will employ the pandas library.

Considering that the dataset is in a CSV file format, we will utilize the pandas function read_csv() to ingest the dataset. This function requires the directory path to the CSV file as a parameter.

To preview the initial 5 rows of the dataset, we will employ the df.head() function, which displays the desired subset of the data.

Activity 2: Data Preparation

Data preparation, or data preprocessing, refers to the essential steps of refining, transforming, and organizing raw data prior to its utilization in data analysis or machine learning models.

The outlined activity encompasses the following steps:

- Identification and removal of missing values
- Restoring the missing values.
- Encoding categorical variables.
- Normalizing the data.

Please note that these steps serve as a general guideline for pre-processing data before its application in machine learning training. The specific pre-processing requirements may vary based on the characteristics of the dataset.

2.1 Identification and removal of missing values.

Upon thorough examination, it has come to our attention that there exists a discernible pattern among the missing values observed in three specific variables. However, we have been unable to identify the precise reporting date within our dataset. Consequently, we have made the decision to remove the column pertaining to the reporting date. Nonetheless, after careful consideration, we have determined that the continent and countries columns bear significant importance as they play a pivotal role in exploratory data analysis, visualization, and overall model construction. Therefore, we have opted to retain these columns within our dataset, recognizing their value and relevance to our objectives.

2.2 Restoring the missing values.

Remarkably, approximately 80% of the data contained within the continent and country columns has been identified as missing. Fortunately, we possess comprehensive information in the form of longitude and latitude coordinates. Leveraging the capabilities of the Python modules "pycountry" and "geocoder," we can utilize geospatial coordinates to derive and compute the corresponding country and continent for each data point. This approach enables us to bridge the gap in the dataset and successfully determine the missing values for the continent and country variables.

Executing the aforementioned code snippet to implement the proposed solution.

In order to enhance comprehension and facilitate better understanding, we will assign country names based on the existing country codes available in the dataset. By utilizing the

country codes as references, we can replace the country codes with corresponding country names, enabling clearer interpretation of the data.

Upon restoring a significant portion of the missing values in the two columns, a subsequent examination reveals that approximately 14% of the country names remain unresolved.

Further scrutiny has confirmed that all of these countries, except for one European country, belong to the African continent. To address this, we shall replace the remaining null values with suitable values, taking into consideration the geographic context and assigning the appropriate country names accordingly.

2.3 Encoding categorical variables.

We have identified two categorical columns within our dataset. Considering the extensive number of countries, which exceeds a hundred, we have made the decision not to encode the country column. Instead, we will focus on encoding the continent column. To achieve this, we will leverage the column transformer functionality offered by the sklearn module. It is important to note that the column transformer converts the provided data into an array format following the transformation process. However, for our subsequent analysis, we require the dataset to be in a dataframe format. As a result, we will apply the column transformer at the initial stages of our model building process, subsequent to the completion of exploratory data analysis (EDA) and visualization tasks.

Milestone 3: Exploratory Data Analysis

Activity 1: Descriptive Statistical Analysis

In this activity, the collected dataset for Lumpy Skin Disease is subjected to descriptive statistical analysis to gain insights into the data. Various statistical measures such as mean, median, mode, standard deviation, and quartiles are calculated for numerical variables

related to the disease, such as lesion size, duration of symptoms, and severity of infection. Frequency distributions and histograms are generated to visualize the distribution of categorical variables, including geographic regions, affected livestock breeds, and vaccination status. These descriptive statistics help in understanding the central tendencies, variabilities, and distributions of the dataset, providing initial insights into the prevalence and characteristics of Lumpy Skin Disease.

Activity 2: Visual analysis

Activity 2.1: Univariate analysis

The code snippet presented below facilitates the generation of histograms to visualize the distribution of numerical columns, namely "wet day" and "temperatures." By employing Python's Matplotlib library, these histograms provide a graphical representation of the frequency distribution for each respective column. This aids in gaining a deeper understanding of the data's characteristics and patterns related to "wet day" and "temperatures" variables.

Activity 2.2: Bivariate analysis

Utilizing the code provided in the accompanying visual representation, we can ascertain the top ten countries that experienced the highest impact from the disease. The code employs a specific methodology to analyze the dataset and extract the relevant information, enabling the identification of the countries that suffered the most significant effects of the disease outbreak.

Similarly, employing the code depicted in the aforementioned visual representation, we can also determine the ten least affected countries. This code utilizes a specific approach to analyze the dataset and extract the pertinent information, enabling the identification of countries that experienced relatively lower impact from the disease outbreak. By

examining the data, we can ascertain the countries that were least affected by the disease.

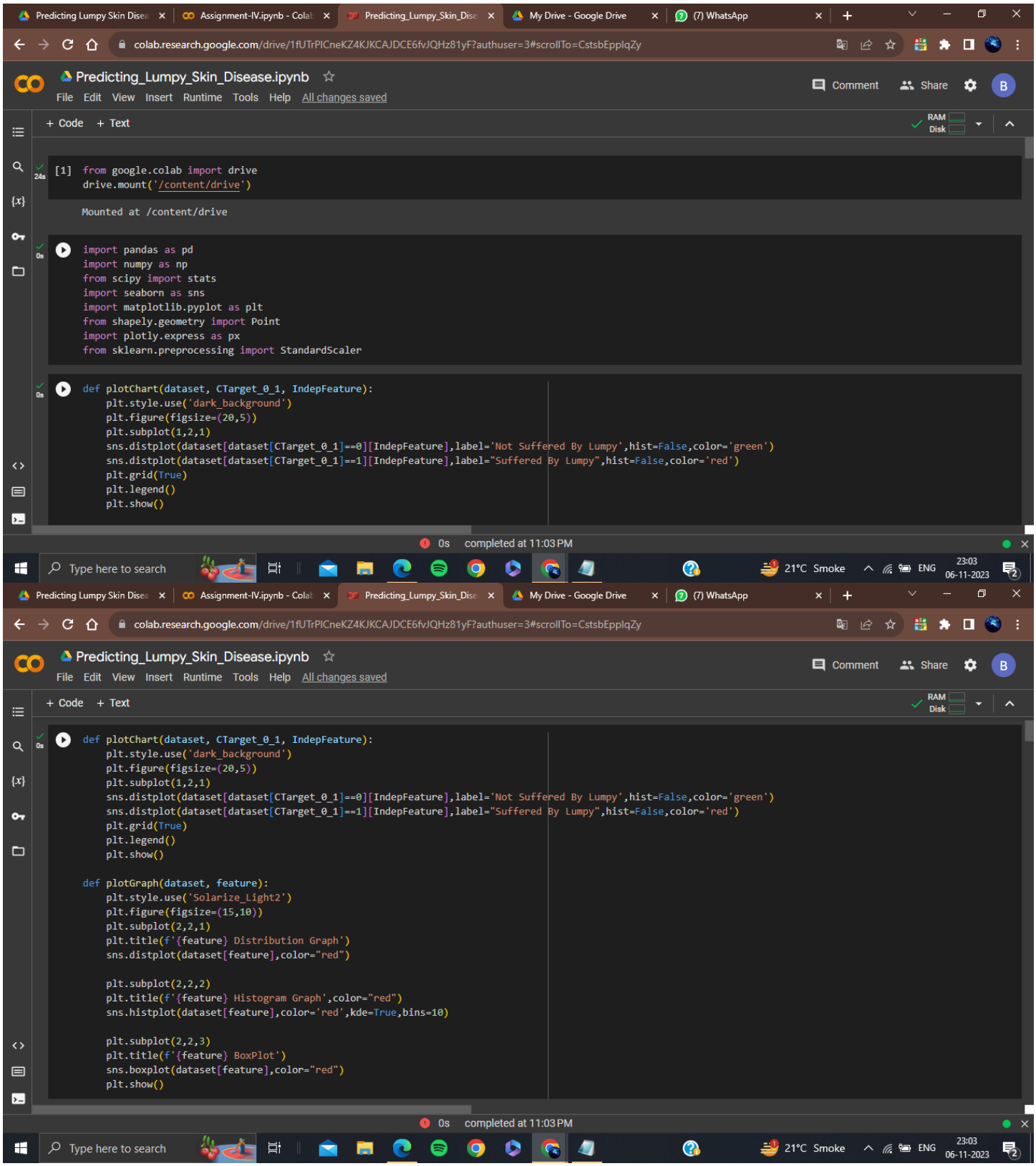
To determine the quantity of datapoints available in this dataset, we can leverage the Plotly module and its Scatter Geo function. By plotting the "longitude" and "latitude" columns on a map using this function, we can visualize the geographical distribution of the data points. This enables us to gain insights into the density and spread of the datapoints across different locations on the map, providing an estimate of the dataset's extent and coverage.

Activity 2.3: Multivariate analysis

Continuing our utilization of the Plotly module, we employ the Scatter mapbox functionality for multivariate analysis. By leveraging the Scatter mapbox function, we can visualize the distinction between locations that were affected by the disease and those that were not. This analysis allows us to observe and discern any discernible patterns, spatial relationships, or differences between diseased and non-diseased locations on a geographical map. The interactive nature of Plotly enables us to explore and gain deeper insights into the spatial dynamics of the disease's impact.

Once again, we harness the power of the Plotly module to explore the relationship between two numerical variables and a categorical column. By employing Plotly's visualization capabilities, we can create interactive charts or graphs that provide insights into the connections, dependencies, or patterns that may exist between these variables. This analysis enables us to better comprehend how the categorical column interacts with and influences the numerical variables, allowing for a more comprehensive understanding of the dataset's underlying dynamics.

To ascertain the interplay between elevation and geographical coordinates, we employ the following code snippet as part of our analytical methodology.



```
[1] from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive
```

```
import pandas as pd
import numpy as np
from scipy import stats
import seaborn as sns
import matplotlib.pyplot as plt
from shapely.geometry import Point
import plotly.express as px
from sklearn.preprocessing import StandardScaler
```

```
def plotChart(dataset, CTarget_0_1, IndepFeature):
    plt.style.use('dark_background')
    plt.figure(figsize=(20,5))
    plt.subplot(1,2,1)
    sns.distplot(dataset[dataset[CTarget_0_1]==0][IndepFeature],label='Not Suffered By Lumpy',hist=False,color='green')
    sns.distplot(dataset[dataset[CTarget_0_1]==1][IndepFeature],label="Suffered By Lumpy",hist=False,color='red')
    plt.grid(True)
    plt.legend()
    plt.show()
```

```
def plotChart(dataset, CTarget_0_1, IndepFeature):
    plt.style.use('dark_background')
    plt.figure(figsize=(20,5))
    plt.subplot(1,2,1)
    sns.distplot(dataset[dataset[CTarget_0_1]==0][IndepFeature],label='Not Suffered By Lumpy',hist=False,color='green')
    sns.distplot(dataset[dataset[CTarget_0_1]==1][IndepFeature],label="Suffered By Lumpy",hist=False,color='red')
    plt.grid(True)
    plt.legend()
    plt.show()

def plotGraph(dataset, feature):
    plt.style.use('Solarize_Light2')
    plt.figure(figsize=(15,10))
    plt.subplot(2,2,1)
    plt.title(f'{feature} Distribution Graph')
    sns.distplot(dataset[feature],color="red")

    plt.subplot(2,2,2)
    plt.title(f'{feature} Histogram Graph',color="red")
    sns.histplot(dataset[feature],color='red',kde=True,bins=10)

    plt.subplot(2,2,3)
    plt.title(f'{feature} BoxPlot')
    sns.boxplot(dataset[feature],color="red")
    plt.show()
```

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
[3]  sns.boxplot(dataset[feature],color="red")
     plt.show()
```

```
path="/content/drive/MyDrive/Lumpy skin disease data.csv"
df_lumpy=pd.read_csv(path)
df_lumpy.head()
```

	x	y	region	country	reportingDate	cld	dtr	frs	pet	pre	tmn	tmp	tmx	vap	wet	elevation	dominant_land_cover	X5_Ct_2010_Da	X5_Bf_2(
0	90.380931	22.437184	Asia	Bangladesh	10/9/2020	41.6	12.8	0.00	2.3	1.7	12.7	19.1	25.5	15.7	0.00	147	2	27970.983100	3691.
1	87.854975	22.986757	Asia	India	20/12/2019	40.5	13.3	0.00	2.4	0.0	13.2	19.8	26.5	16.3	0.00	145	2	25063.646690	671
2	85.279935	23.610181	Asia	India	20/12/2019	27.3	13.6	0.08	2.3	0.6	9.4	16.2	23.0	13.0	0.98	158	2	6038.477155	1426
3	81.564510	43.882221	Asia	China	25/10/2019	45.3	12.8	31.00	0.4	8.8	-22.5	-16.1	-9.7	0.9	4.64	178	2	760.703340	0.
4	81.161057	43.834976	Asia	China	25/10/2019	38.8	13.2	31.00	0.4	10.5	-20.4	-13.8	-7.2	1.2	1.69	185	3	270.367426	0.

```
[6]  df_lumpy.describe()
```

	x	y	cld	dtr	frs	pet	pre	tmn	tmp	tmx	vap	wet
count	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000
mean	79.221374	46.370056	59.452159	9.107777	23.978048	0.803487	26.271137	-15.794755	-11.227807	-6.681212	3.728230	8.542482

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:03 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help Saving...

Comment Share B

+ Code + Text

RAM Disk

count	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000	24803.000000
mean	79.221374	46.370056	59.452159	9.107777	23.978048	0.803487	26.271137	-15.794755	-11.227807	-6.681212	3.728230	8.542482
std	43.338530	19.220555	19.423029	2.988448	11.518315	1.172915	33.630747	17.587685	17.989715	18.540915	4.952353	6.205199
min	-179.750000	-28.750000	0.000000	2.000000	0.000000	0.000000	0.000000	-52.100000	-48.100000	-44.200000	0.000000	0.000000
25%	45.083150	34.750000	43.800000	6.800000	23.210000	0.000000	5.900000	-30.100000	-25.500000	-20.900000	0.400000	3.000000
50%	80.750000	48.250000	62.300000	8.300000	31.000000	0.200000	14.700000	-19.100000	-14.200000	-9.700000	1.500000	8.020000
75%	109.750000	61.750000	75.300000	11.100000	31.000000	1.100000	33.400000	-2.200000	1.400000	4.900000	4.800000	12.710000
max	179.750000	81.750000	98.700000	20.600000	31.000000	7.500000	341.900000	23.900000	28.500000	36.400000	28.600000	30.920000

```
[7]  df_lumpy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24803 entries, 0 to 24802
Data columns (total 20 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   x                   24803 non-null  float64
 1   y                   24803 non-null  float64
 2   region              3039 non-null   object
 3   country             3039 non-null   object
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:03 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
[8] df_lumpy.duplicated().sum()
```

```
608
```

```
[9] df_lumpy = df_lumpy.drop_duplicates()
```

```
df_lumpy.shape
```

```
(24195, 20)
```

```
[11] df_lumpy.isnull().sum()
```

```
x                    0
y                    0
region           21764
country          21764
reportingDate    21764
cld                  0
dtr                  0
frs                  0
pet                  0
pre                  0
tmn                  0
tmp                  0
tmx                  0
vap                  0
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:03 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
[12] df_lumpy.rename(columns={"x":"Longitude" , "y":"Latitude", "cld":"Monthly Cloud Cover", "dtr":"Diurnal Temperature Range", "frs":"Frost Day Frequency", "pet":"Pc
```

```
df_lumpy.head()
```

	Longitude	Latitude	region	country	reportingDate	Monthly Cloud Cover	Diurnal Temperature Range	Frost Day Frequency	Potential EvapoTranspiration	Precipitation	Minimum Temperature	Mean Temperature	Maximum Temperature
0	90.380931	22.437184	Asia	Bangladesh	10/9/2020	41.6	12.8	0.00	2.3	1.7	12.7	19.1	25.5
1	87.854975	22.986757	Asia	India	20/12/2019	40.5	13.3	0.00	2.4	0.0	13.2	19.8	26.5
2	85.279935	23.610181	Asia	India	20/12/2019	27.3	13.6	0.08	2.3	0.6	9.4	16.2	23.0
3	81.564510	43.882221	Asia	China	25/10/2019	45.3	12.8	31.00	0.4	8.8	-22.5	-16.1	-9.7
4	81.161057	43.834976	Asia	China	25/10/2019	38.8	13.2	31.00	0.4	10.5	-20.4	-13.8	-7.2

```
[14] z = np.abs(stats.zscore(df_lumpy['Monthly Cloud Cover']))
     out=np.where(z>3)[0]
     print('Monthly Cloud Cover'+ " " + str(out))
     df_lumpy['Monthly Cloud Cover'] = df_lumpy['Monthly Cloud Cover'].drop(out)
```

```
Monthly Cloud Cover [18124]
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEppIqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[15]
```
Diurnal Temperature Range [18309 18310 18311 18312 18325 18326 18327 18328 18341 18342 18343 18344
 18358 19872 19906 19907 19908 19910 19911 19912 19913 19914 19941 19942
 19943 19944 19945 19946 19947 19948 19949 19950 19979 19981 20039 20078
 20079 20133 20134 20188 20241 20296 21139 21151 21152 21159 21160 21166
 22524 22525 22526 22569 22570 22571 22572 22613 22614 22615 22616 22617
 22656 22657 22658 22659 22660 22661 22702 22703 22704 22705 22706 22707
 22748 22749 22750 22751 22752 22753 22793 22794 22795 22796 22797 22798
 22799 22840 22841 22842 22843 22844 22889 22890 22891 22892 22939 22940]
```

```
z = np.abs(stats.zscore(df_lumpy['Frost Day Frequency']))
out = np.where(z>3)[0]
print('Frost Day Frequency'+ " " + str(out))
df_lumpy['Frost Day Frequency'] = df_lumpy['Frost Day Frequency'].drop(out)
```

```
Frost Day Frequency []
```

[17]
```
z = np.abs(stats.zscore(df_lumpy['Mean Temperature']))
out = np.where(z>3)[0]
print('Mean Temperature'+ " " + str(out))
df_lumpy['Mean Temperature'] = df_lumpy['Mean Temperature'].drop(out)
```

```
Mean Temperature []
```

[18]
```
z = np.abs(stats.zscore(df_lumpy['Minimum Temperature']))
out = np.where(z>3)[0]
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEppIqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[17]
```
print('Mean Temperature'+ " " + str(out))
df_lumpy['Mean Temperature'] = df_lumpy['Mean Temperature'].drop(out)
```

```
Mean Temperature []
```

[18]
```
z = np.abs(stats.zscore(df_lumpy['Minimum Temperature']))
out = np.where(z>3)[0]
print('Minimum Temperature'+ " " + str(out))
df_lumpy['Minimum Temperature'] = df_lumpy['Minimum Temperature'].drop(out)
```

```
Minimum Temperature []
```

[20]
```
z = np.abs(stats.zscore(df_lumpy['Maximum Temperature']))
out = np.where(z>3)[0]
print('Maximum Temperature'+ " " + str(out))
df_lumpy['Maximum Temperature'] = df_lumpy['Maximum Temperature'].drop(out)
```

```
Maximum Temperature []
```

[21]
```
z = np.abs(stats.zscore(df_lumpy['Wet Day Frequency']))
out = np.where(z>3)[0]
print('Wet Day Frequency'+ " " + str(out))
df_lumpy['Wet Day Frequency'] = df_lumpy['Wet Day Frequency'].drop(out)
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
[23] df_lumpy = df_lumpy.fillna(method='bfill')
     df_lumpy.isnull().sum()
```

```
Longitude                      0
Latitude                       0
region                     21764
country                    21764
reportingDate              21764
Monthly Cloud Cover            0
Diurnal Temperature Range      0
Frost Day Frequency            0
Potential EvapoTranspiration   0
Precipitation                  0
Minimum Temperature            0
Mean Temperature               0
Maximum Temperature            0
Vapour Pressure                0
Wet Day Frequency              0
elevation                      0
dominant_land_cover            0
X5_Ct_2010_Da                  0
X5_Bf_2010_Da                  0
lumpy                          0
dtype: int64
```

```
[24] geolocation = df_lumpy.loc[df_lumpy['lumpy'] != 0][['Longitude', 'Latitude', 'country']]
     geolocation.head()
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
[23] dominant_land_cover            0
     X5_Ct_2010_Da                  0
     X5_Bf_2010_Da                  0
     lumpy                          0
     dtype: int64
```

```
geolocation = df_lumpy.loc[df_lumpy['lumpy'] != 0][['Longitude', 'Latitude', 'country']]
geolocation.head()
```

	Longitude	Latitude	country
0	90.380931	22.437184	Bangladesh
1	87.854975	22.986757	India
2	85.279935	23.610181	India
3	81.564510	43.882221	China
4	81.161057	43.834976	China

```
[25] fig = px.scatter_geo(geolocation,lat='Latitude',lon='Longitude')
     fig.update_layout(title = 'Lumpy Skin Disease Area', title_x=0.5)
     fig.show()
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=CstsbEpplqZy

Predicting_Lumpy_Skin_Disease.ipynb

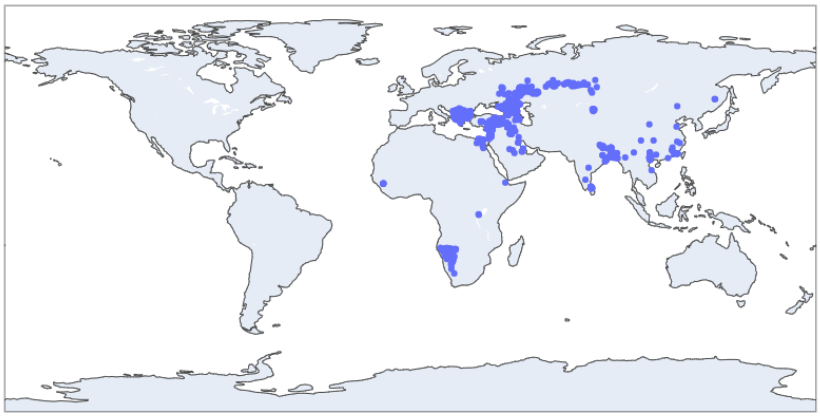
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

Lumpy Skin Disease Area



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=n7yTrVJsg8c2

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

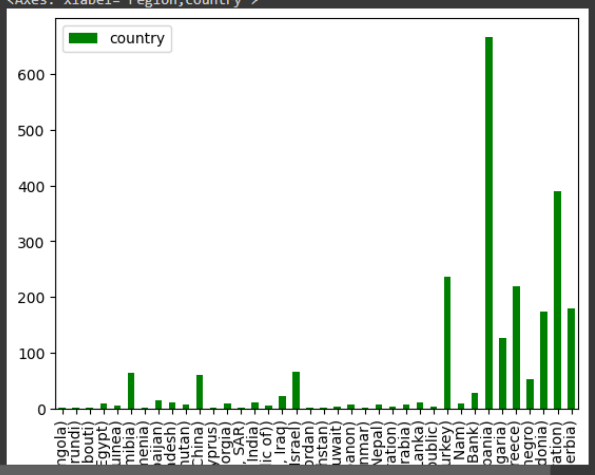
Comment Share B

+ Code + Text

RAM Disk

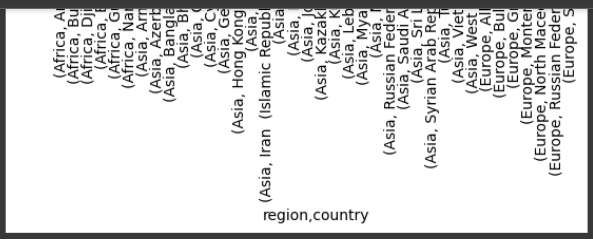
```
pd.DataFrame(df_lumpy.groupby(['region','country'])['country'].count()).plot(kind="bar",color="green")
```

```
<Axes: xlabel='region,country'>
```



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023



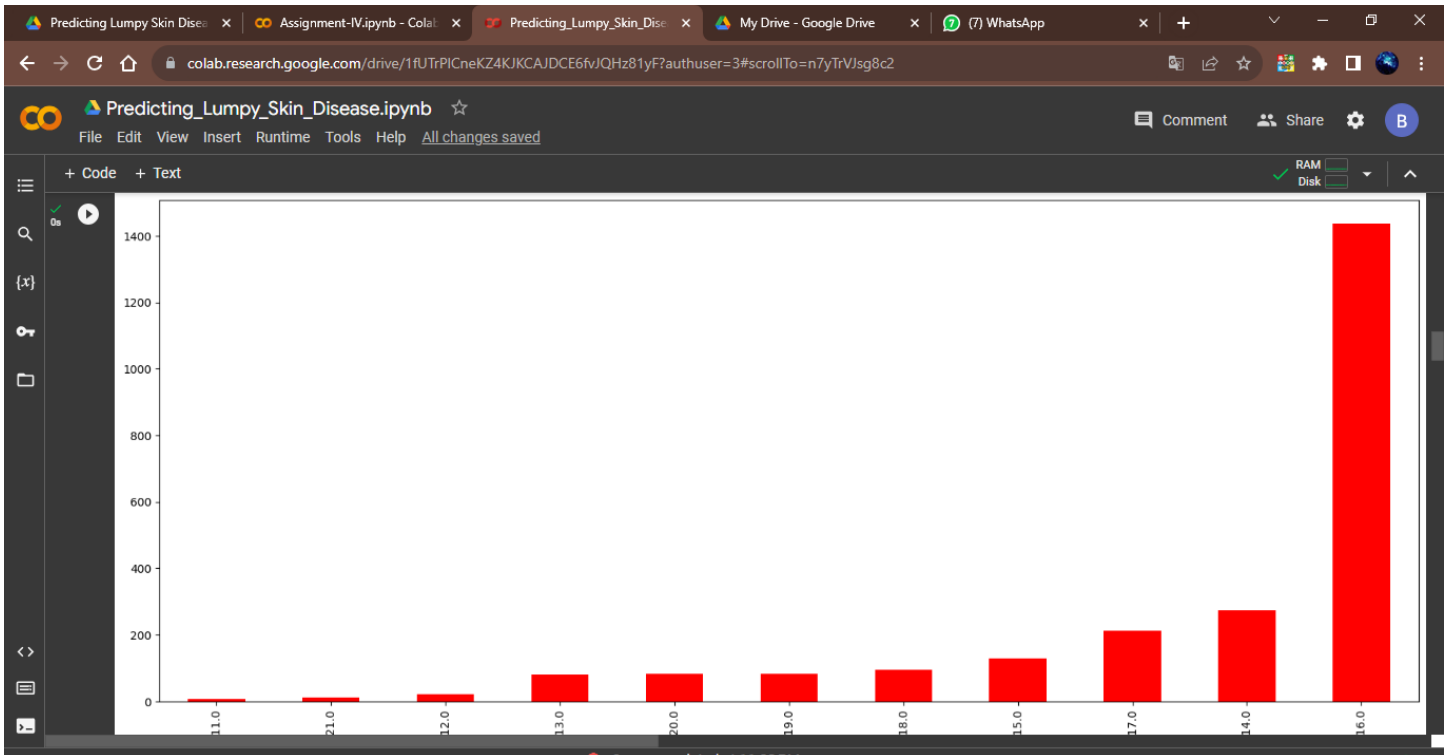

```
[31] df_lumpy['reportingDate'] =pd.to_datetime(df_lumpy['reportingDate'])
     df_lumpy['Year'] = df_lumpy['reportingDate'].dt.year
     pd.DataFrame(df_lumpy['Year'].value_counts().sort_values(ascending=True)).rename(columns={"Year" : "Case Report Count"})
```

```
<ipython-input-31-479aa8050f70>:1: UserWarning:

Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consi
```

	Case Report Count
2011.0	6
2021.0	11
2012.0	21
2013.0	81
2020.0	83
2019.0	83
2018.0	94
2015.0	129
2017.0	212
2014.0	275
2016.0	1436

```
[32] plt.figure(figsize=(20,8))
     df_lumpy[df_lumpy['Year'] == df_lumpy['Year']]['Year'].value_counts().sort_values(ascending=True).plot(kind="bar",color="red")
```

```
<Axes: >
```



Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

0s completed at 11:03 PM

```
[33] df_lumpy['Month'] = df_lumpy['reportingDate'].dt.month
     df_lumpy['MonthTitle']= df_lumpy['reportingDate'].dt.month_name()
```

```
pd.DataFrame(df_lumpy.groupby(['Year','Month'])['Month'].count()).rename(columns={'Month':'CaseCounts'})
```

Year	Month	CaseCounts
2011.0	7.0	1
	10.0	3
	11.0	2
2012.0	5.0	2
	7.0	2
...	...	...
2020.0	9.0	8
	10.0	13
	11.0	7
	12.0	10

0s completed at 11:03 PM

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
df_lumpy.pivot_table(index=['Month','MonthTitle'],columns=['Year','region'],values=['lumpy'],aggfunc='count')
```

	lumpy										...										
Year	2011.0	2012.0	2013.0		2014.0		2015.0		2016.0		...	2017.0		2018.0		2019.0		2020.0		2021.0	
region	Africa	Asia	Africa	Asia	Africa	Asia	Asia	Europe	Africa	Asia	...	Asia	Europe	Africa	Asia	Europe	Asia	Europe	Africa	Asia	Asia
MonthTitle																					
January	NaN	NaN	NaN	1.0	NaN	11.0	1.0	NaN	2.0	1.0	...	NaN	31.0	NaN	NaN	24.0	NaN	NaN	NaN	NaN	11.0
February	NaN	NaN	NaN	3.0	NaN	4.0	NaN	6.0	NaN	2.0	...	NaN	12.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
March	NaN	NaN	NaN	10.0	NaN	10.0	NaN	2.0	NaN	NaN	...	NaN	6.0	NaN	3.0	12.0	1.0	1.0	NaN	1.0	NaN
April	NaN	NaN	NaN	32.0	NaN	50.0	2.0	8.0	NaN	NaN	...	NaN	5.0	NaN	NaN	NaN	5.0	3.0	NaN	1.0	NaN
May	NaN	2.0	NaN	5.0	NaN	33.0	1.0	NaN	NaN	NaN	...	NaN	17.0	NaN	NaN	2.0	NaN	NaN	NaN	1.0	NaN
June	NaN	NaN	NaN	NaN	NaN	31.0	NaN	NaN	NaN	NaN	...	NaN	31.0	NaN	NaN	NaN	2.0	1.0	NaN	2.0	NaN
July	1.0	2.0	NaN	4.0	NaN	93.0	NaN	NaN	2.0	6.0	...	1.0	20.0	10.0	NaN	13.0	8.0	3.0	NaN	13.0	NaN
August	NaN	4.0	1.0	NaN	NaN	13.0	1.0	15.0	NaN	2.0	...	NaN	14.0	NaN	1.0	8.0	4.0	8.0	NaN	27.0	NaN
September	NaN	1.0	NaN	4.0	6.0	NaN	2.0	63.0	2.0	NaN	...	NaN	4.0	NaN	NaN	11.0	8.0	6.0	NaN	8.0	NaN
October	3.0	3.0	NaN	NaN	NaN	19.0	NaN	28.0	NaN	NaN	...	NaN	10.0	NaN	1.0	4.0	21.0	1.0	NaN	13.0	NaN

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:04 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

Predicting_Lumpy_Skin_Disease.ipynb

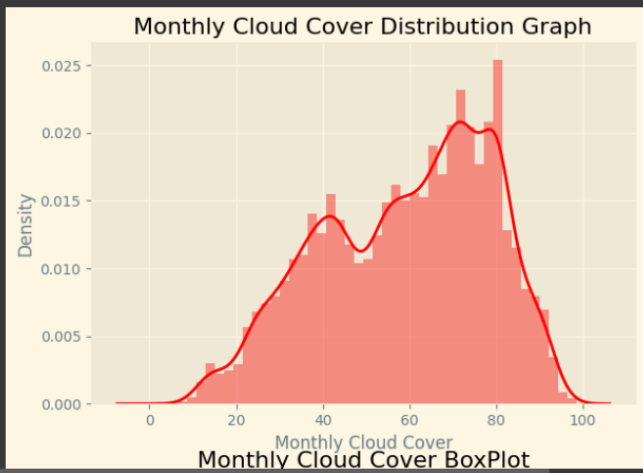
File Edit View Insert Runtime Tools Help All changes saved

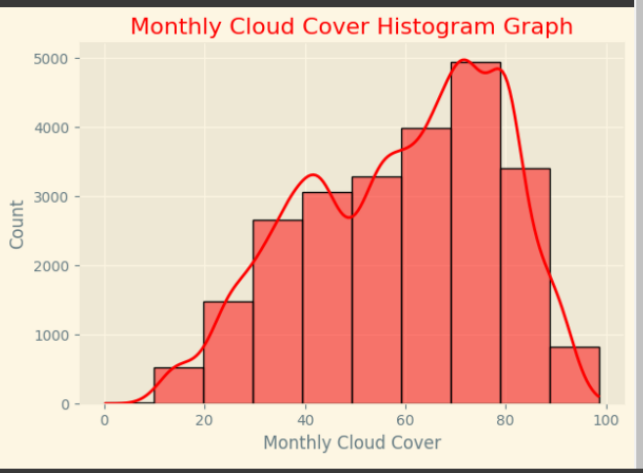
Comment Share B

+ Code + Text

RAM Disk

```
For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```





Monthly Cloud Cover BoxPlot

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:05 06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

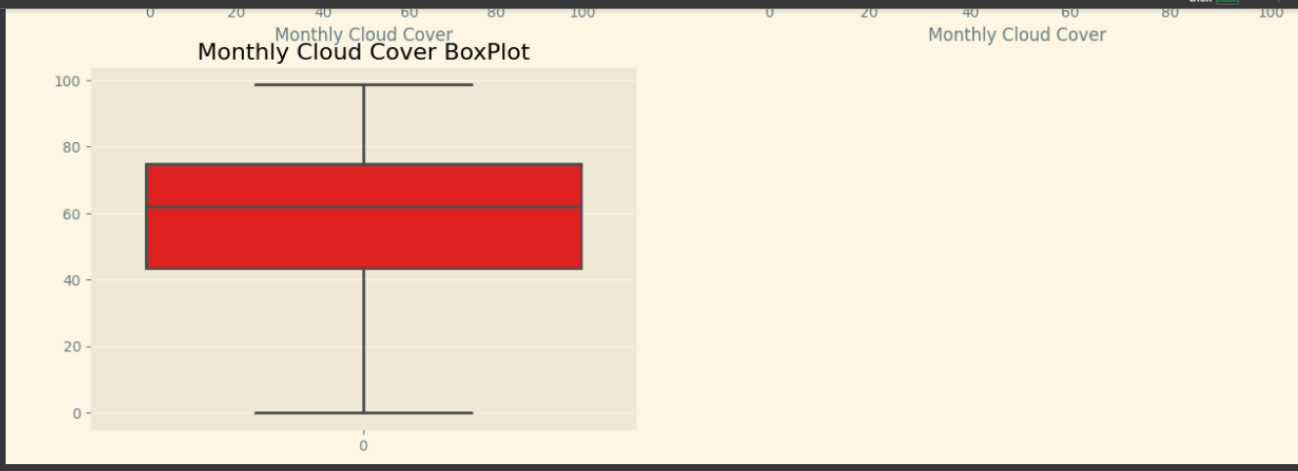
Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk




```
[37] plotGraph(df_lumpy,'Diurnal Temperature Range')
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:05 06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

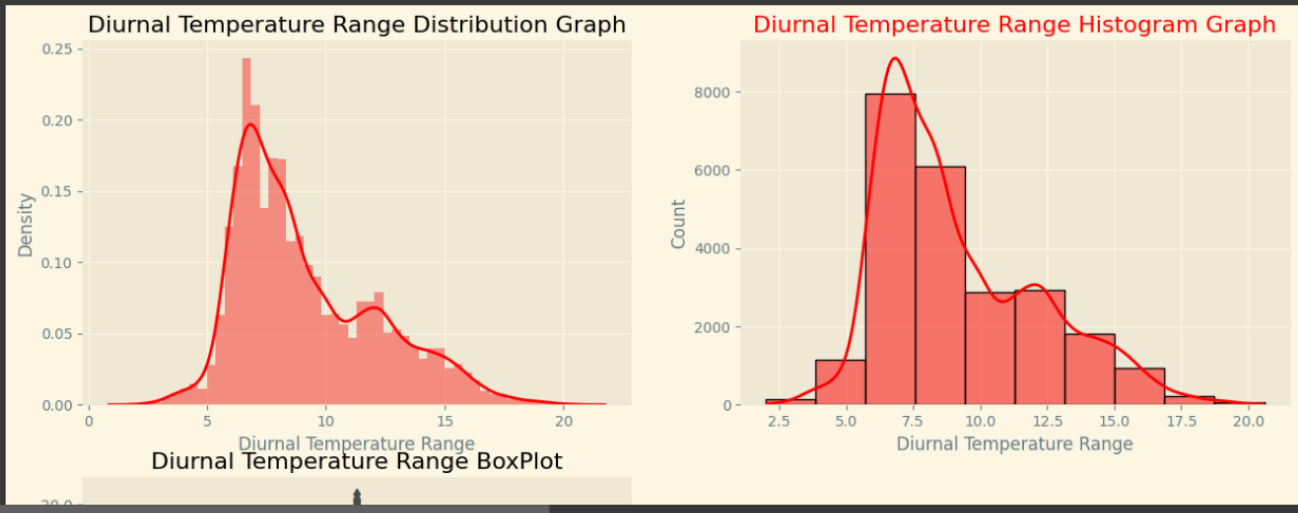
Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

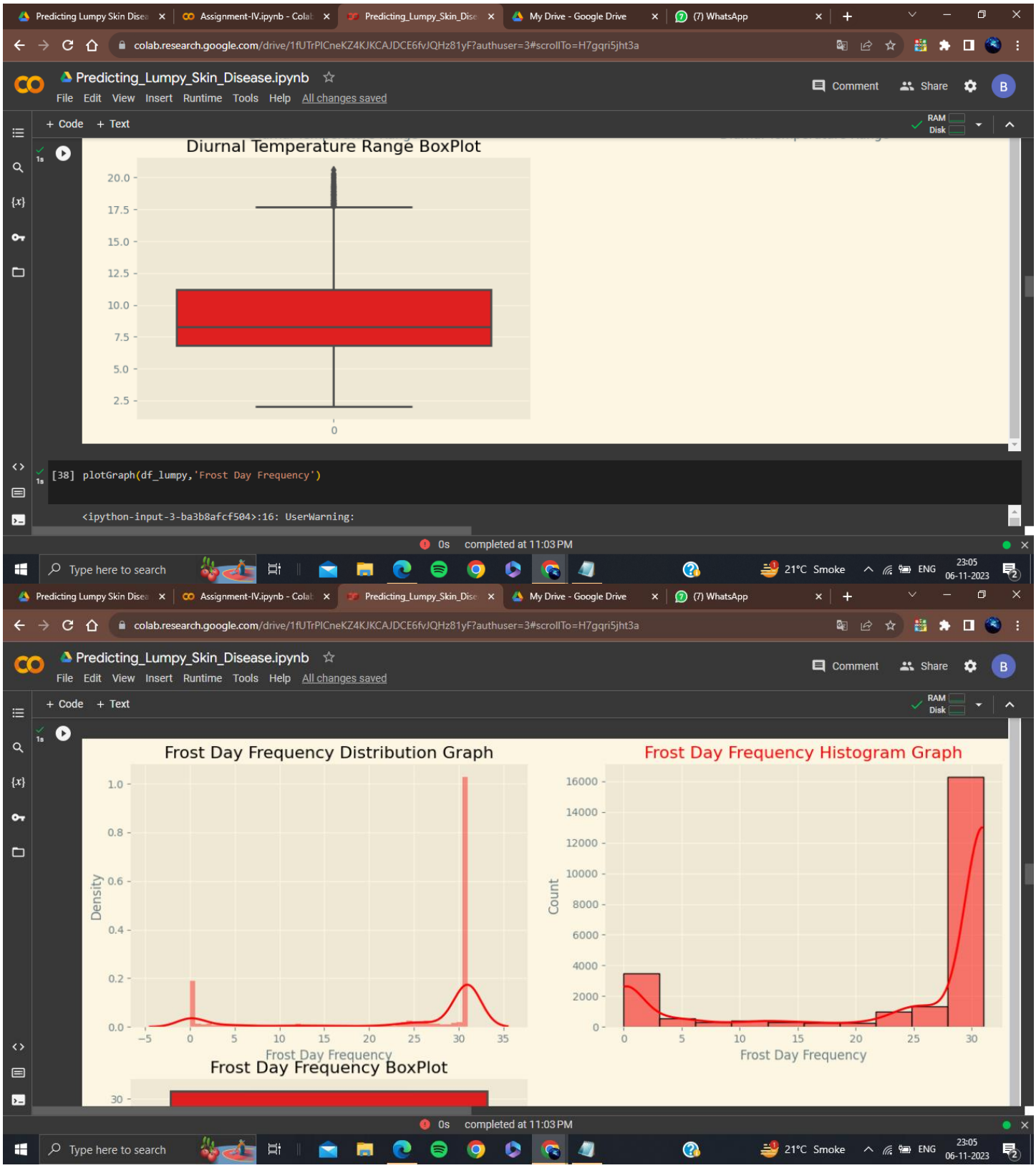
+ Code + Text

RAM Disk



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:05 06-11-2023


Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Diurnal Temperature Range BoxPlot
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0
[38] plotGraph(df_lumpy,'Frost Day Frequency')
<ipython-input-3-ba3b8afcf504>:16: UserWarning:
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:05
06-11-2023

Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Frost Day Frequency Distribution Graph
Density
Frost Day Frequency
Frost Day Frequency Histogram Graph
Count
Frost Day Frequency
Frost Day Frequency BoxPlot
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:05
06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

Predicting_Lumpy_Skin_Disease.ipynb

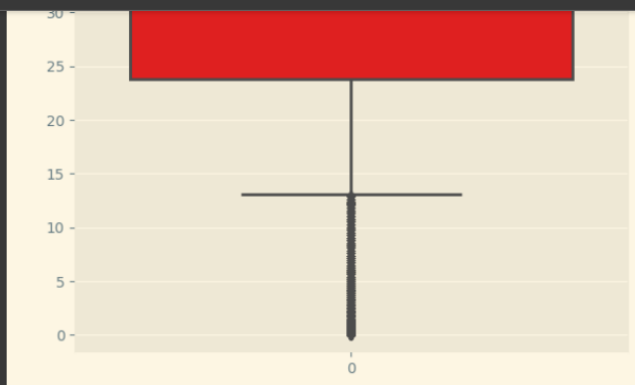
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[38]




```
plotGraph(df_lumpy,'Potential EvapoTranspiration')
```

```
<ipython-input-3-ba3b8afcf504>:16: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:05 06-11-2023

Predicting Lumpy Skin Disea | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=H7gqri5jht3a

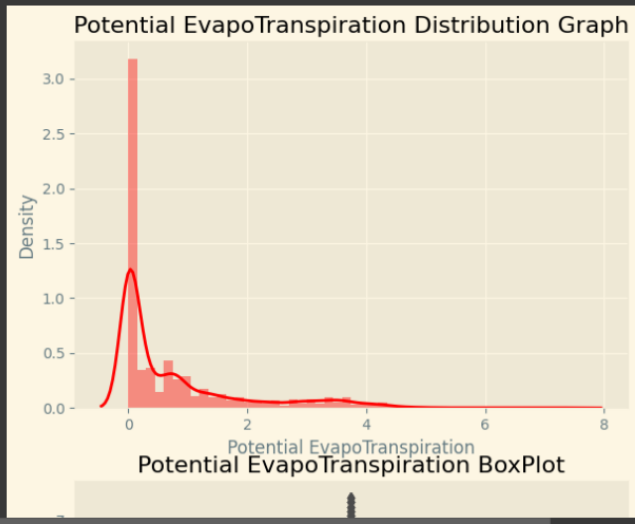
Predicting_Lumpy_Skin_Disease.ipynb

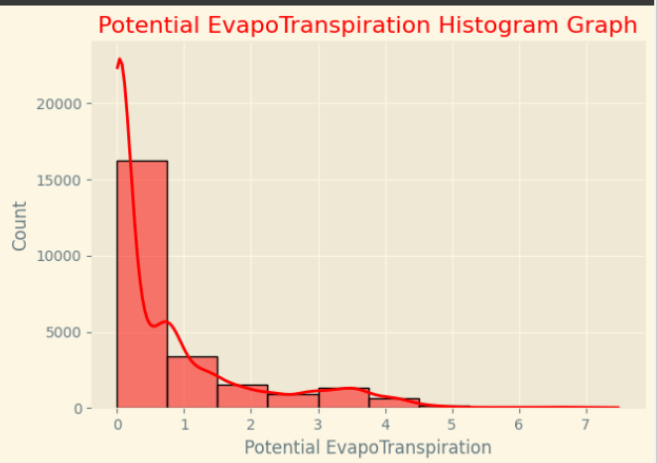
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

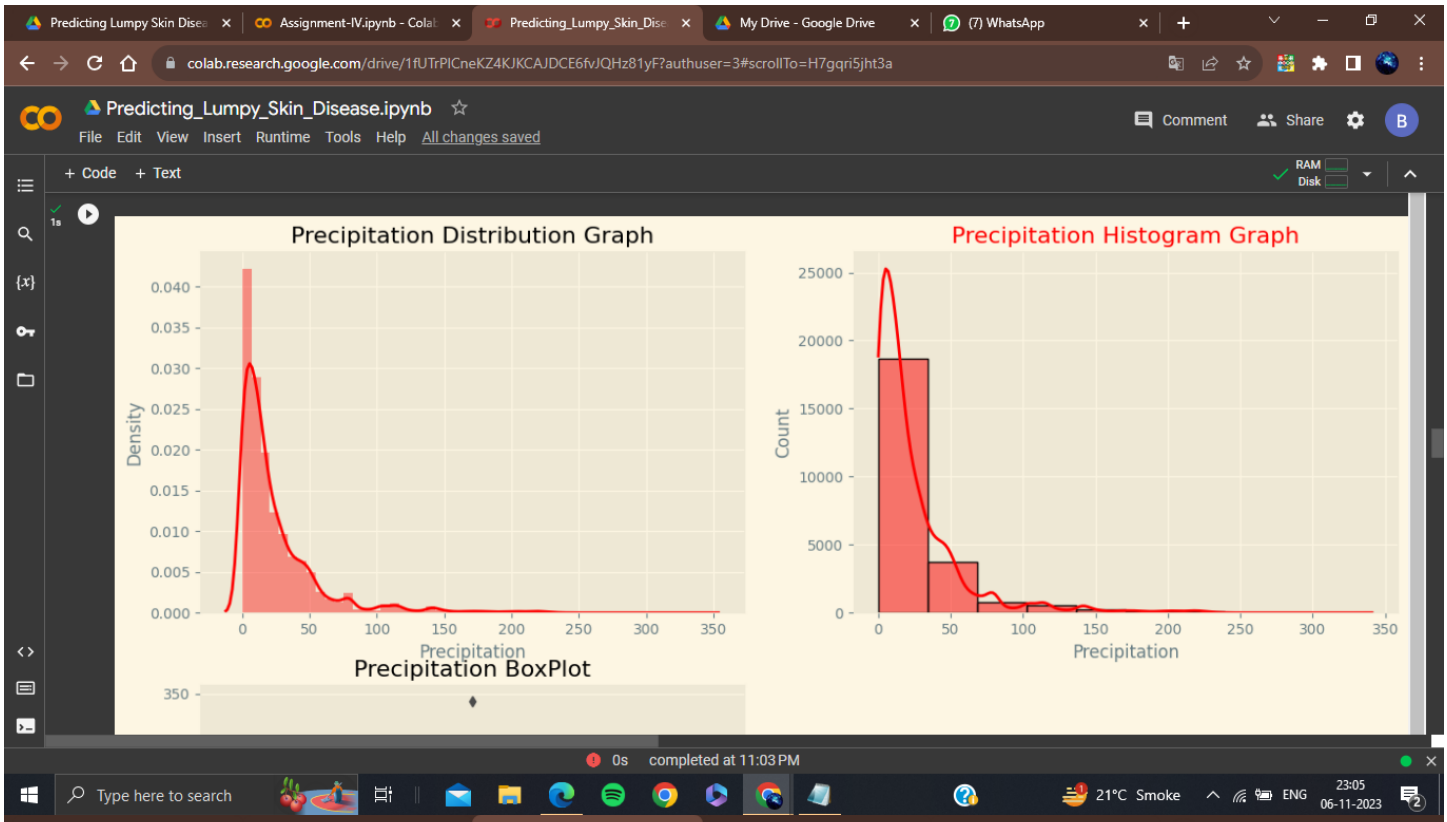
RAM Disk

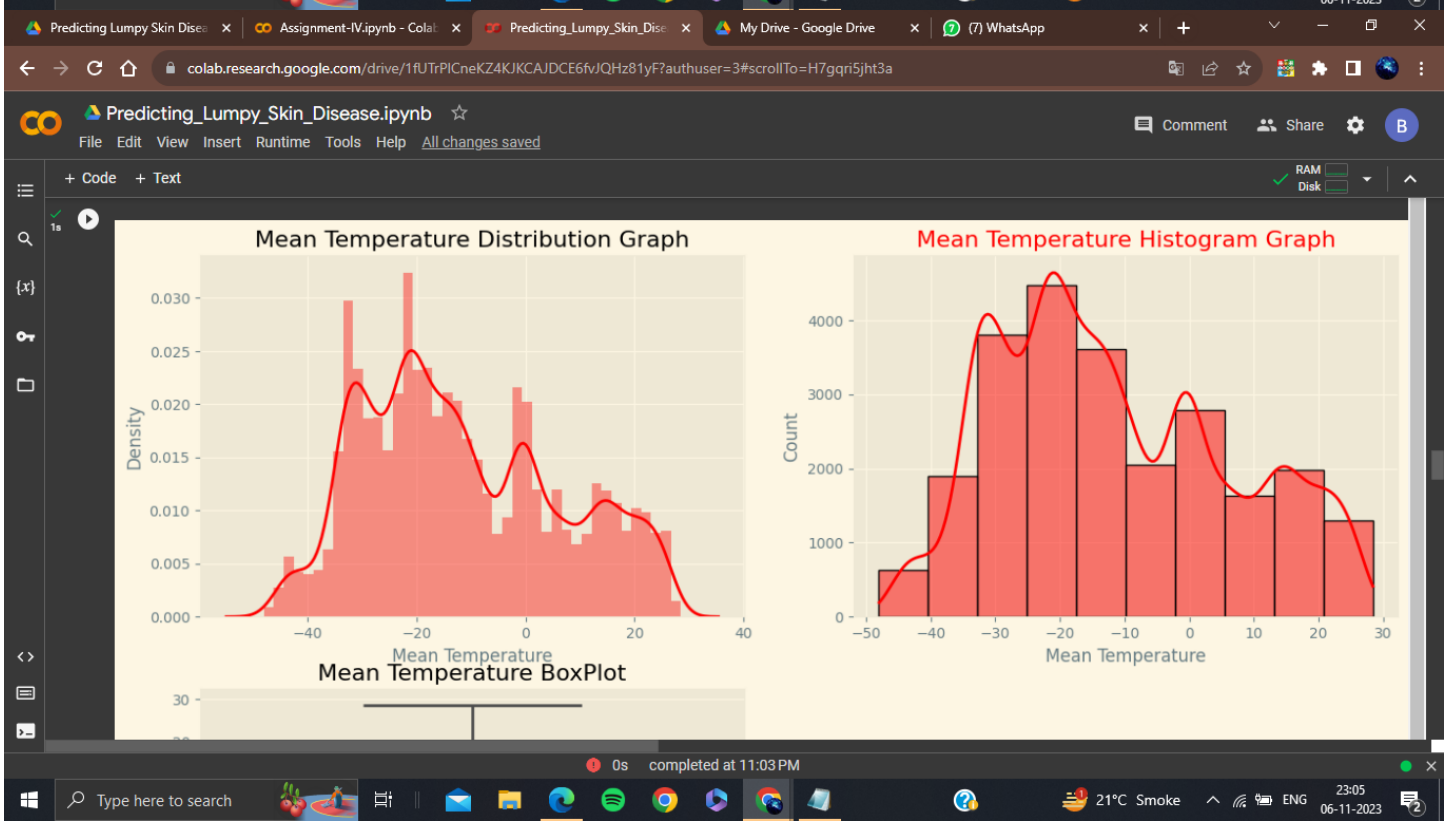


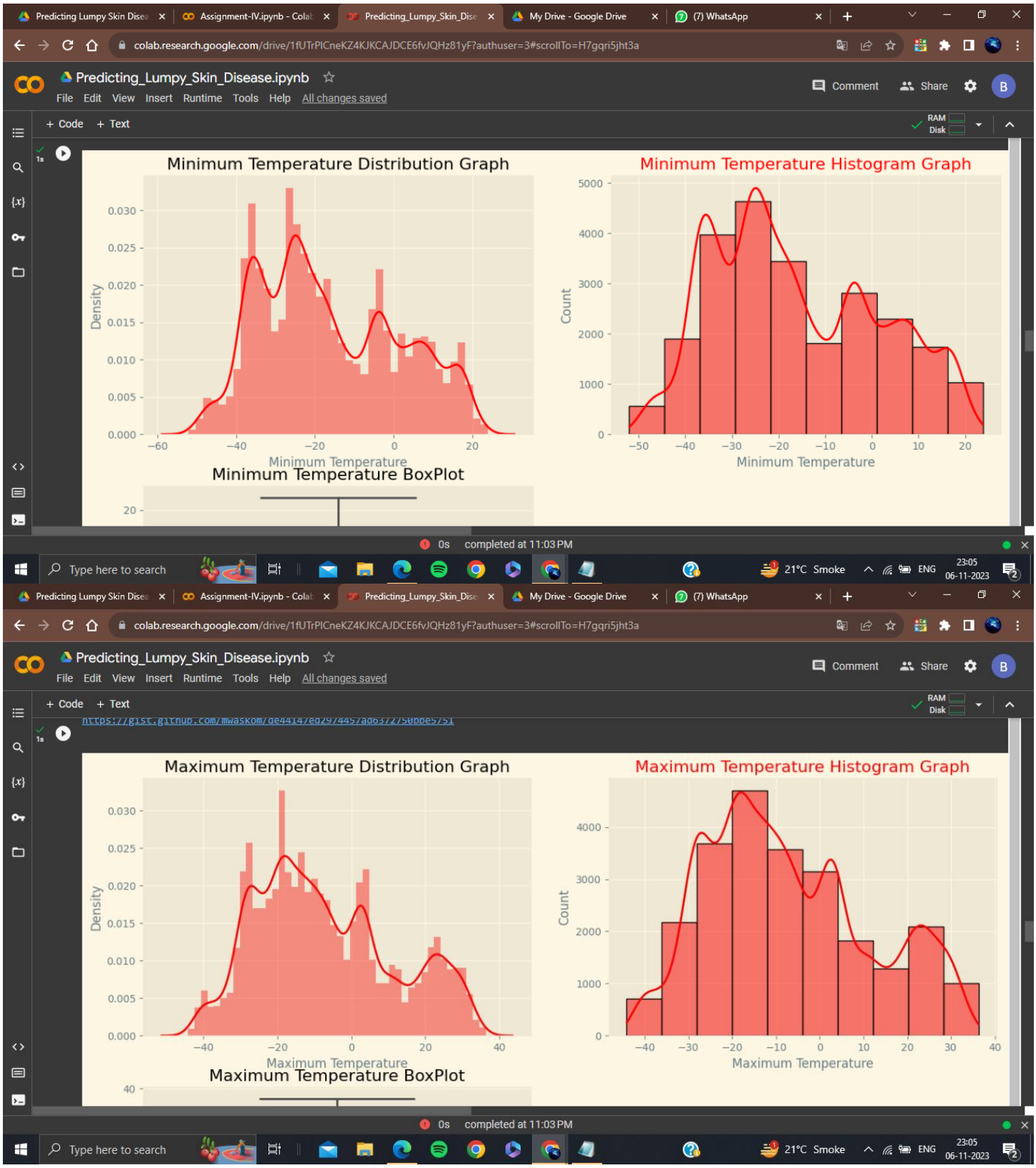


0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:05 06-11-2023


Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Precipitation Distribution Graph
Density
Precipitation
Precipitation Histogram Graph
Count
Precipitation
Precipitation BoxPlot
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:05
06-11-2023


Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Mean Temperature Distribution Graph
Density
Mean Temperature
Mean Temperature Histogram Graph
Count
Mean Temperature
Mean Temperature BoxPlot
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:05
06-11-2023


Predicting_Lumpy_Skin_Disease.ipynb
Minimum Temperature Distribution Graph
Density
Minimum Temperature
Minimum Temperature Histogram Graph
Count
Minimum Temperature
Minimum Temperature BoxPlot
completed at 11:03 PM
Maximum Temperature Distribution Graph
Density
Maximum Temperature
Maximum Temperature Histogram Graph
Count
Maximum Temperature
Maximum Temperature BoxPlot
completed at 11:03 PM

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

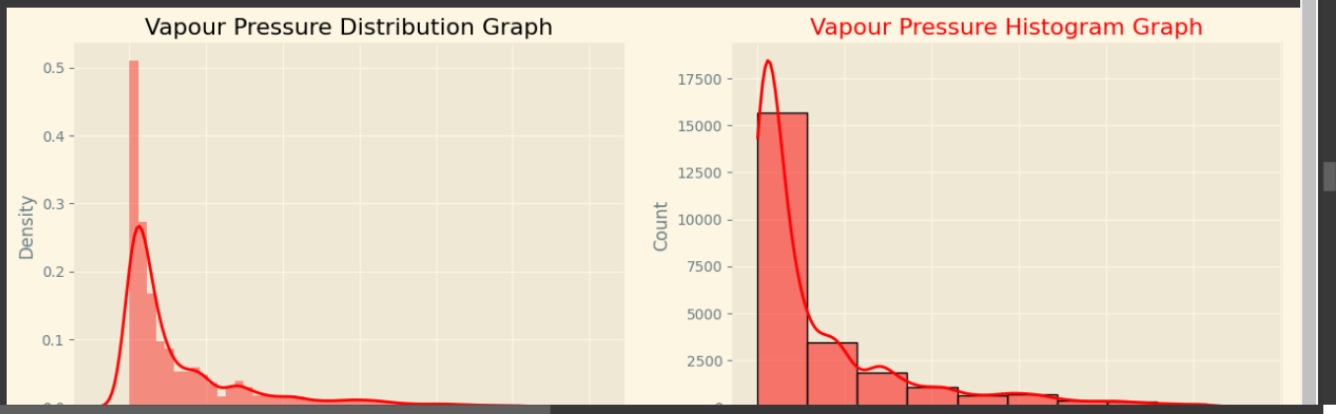
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

```
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:06 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

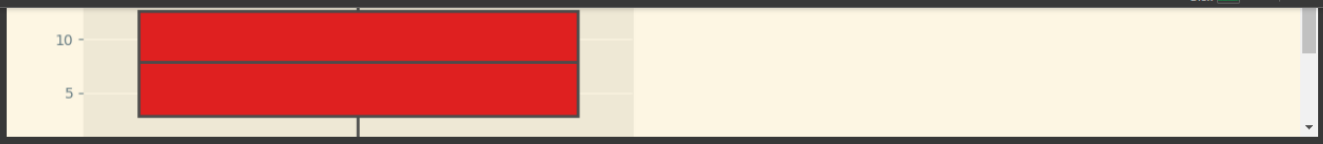
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

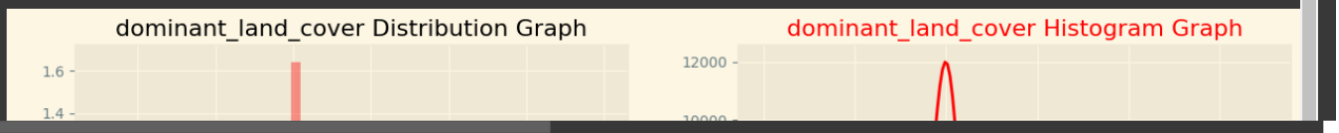
RAM Disk

[45]



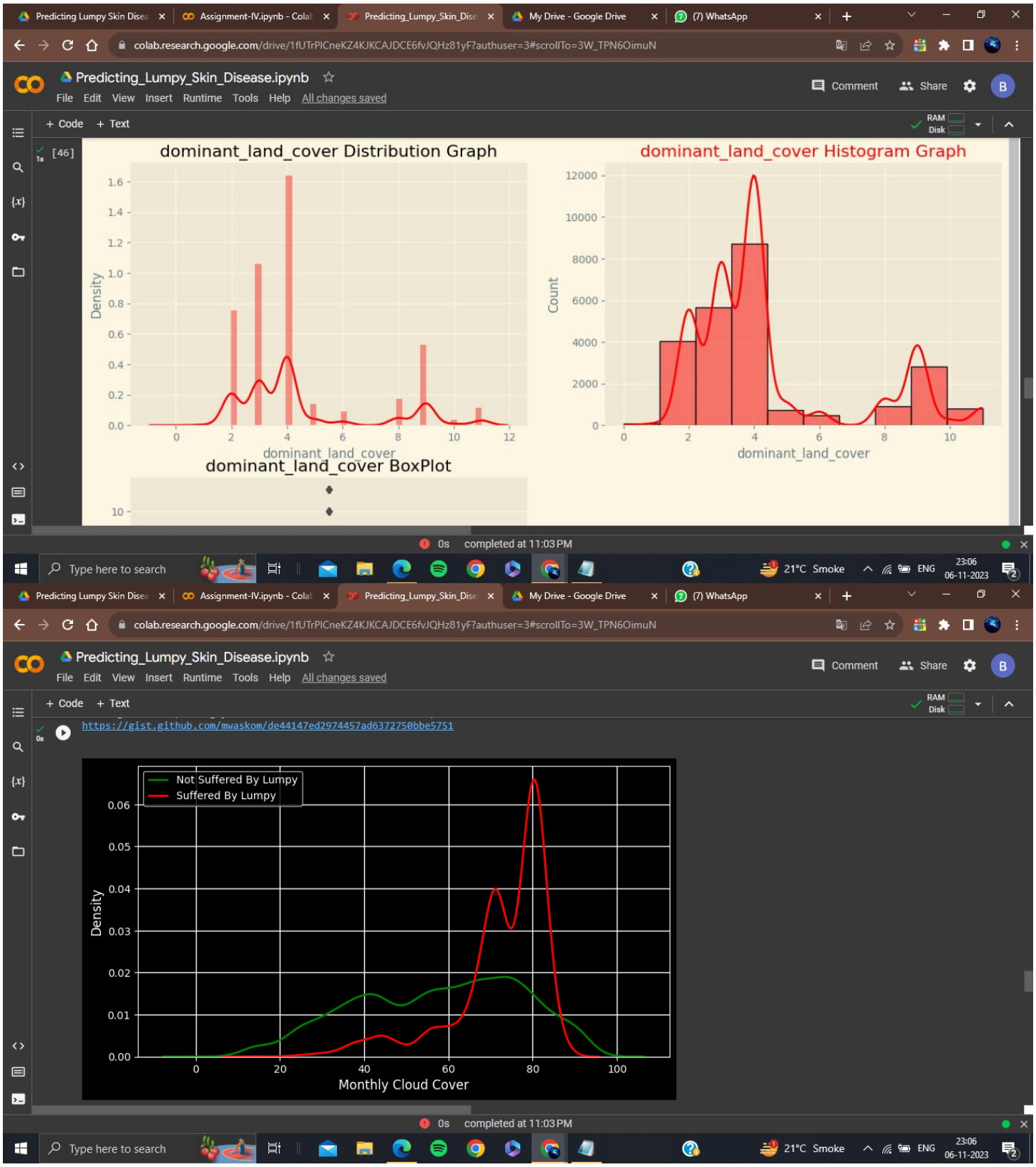

```
[46] plotGraph(df_lumpy,'dominant_land_cover')
```

```
<ipython-input-3-ba3b8afcf504>:16: UserWarning:


`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:06 06-11-2023


Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
[46]
dominant_land_cover Distribution Graph
Density
dominant_land_cover
dominant_land_cover Histogram Graph
Count
dominant_land_cover
dominant_land_cover BoxPlot
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:06
06-11-2023

Predicting_Lumpy_Skin_Disease.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
Not Suffered By Lumpy
Suffered By Lumpy
Density
Monthly Cloud Cover
0s completed at 11:03 PM
Type here to search
21°C Smoke
ENG
23:06
06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

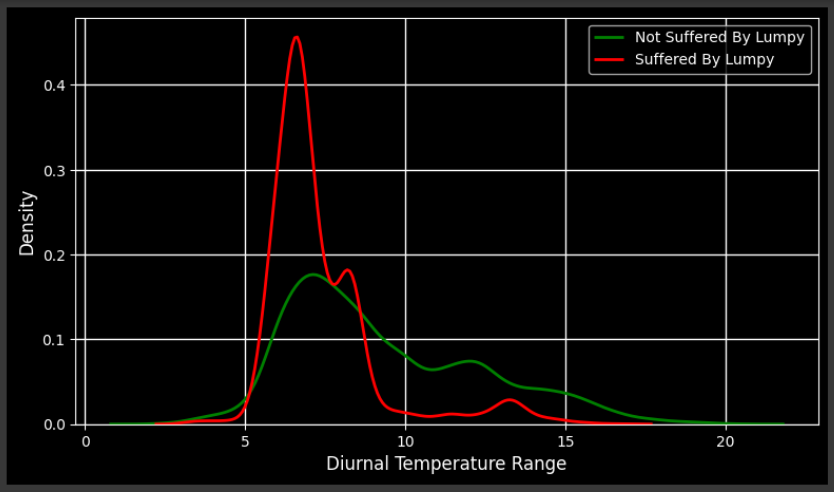
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[48]




```
[49] plotChart(df_lumpy,'lumpy','Potential EvapoTranspiration')
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:06 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPlCneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

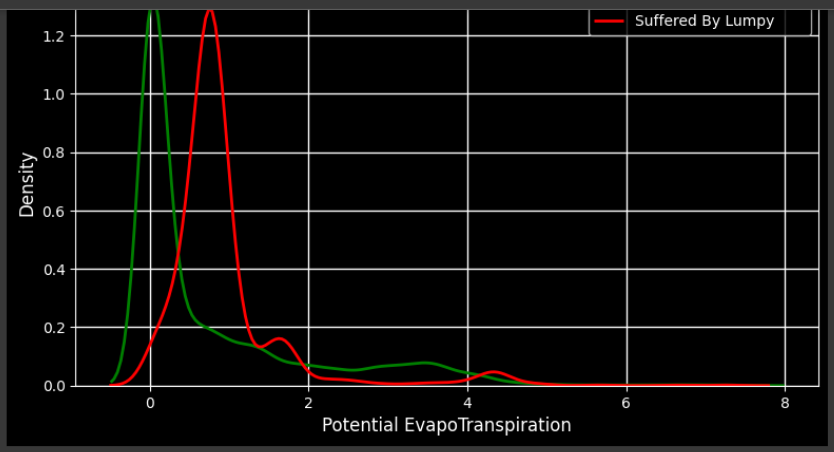
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[49]




```
[50] plotChart(df_lumpy,'lumpy','Mean Temperature')
```

```
<ipython-input-3-ba3b8afcf504>:5: UserWarning:
```

0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:06 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

File Edit View Insert Runtime Tools Help All changes saved

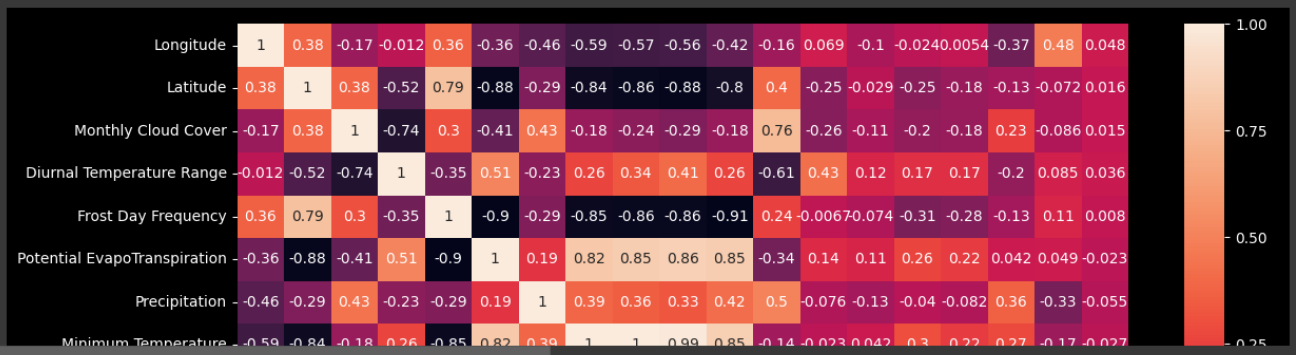
Comment Share B

+ Code + Text

RAM Disk

```
[56] plt.figure(figsize=(30,10))
     plt.subplot(121)
     sns.heatmap(df_lumpy.corr(), annot=True)
     plt.show()
```

```
<ipython-input-56-d555d33d1e77>:3: FutureWarning:

The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the valu
```



0s completed at 11:03 PM

Type here to search 21°C Smoke ENG 23:06 06-11-2023

Predicting Lumpy Skin Dise | Assignment-IV.ipynb - Colab | Predicting_Lumpy_Skin_Dise | My Drive - Google Drive | (7) WhatsApp

colab.research.google.com/drive/1fUTrPICneKZ4KJKCAJDCE6fvJQHz81yF?authuser=3#scrollTo=3W_TPN6OimuN

Predicting_Lumpy_Skin_Disease.ipynb

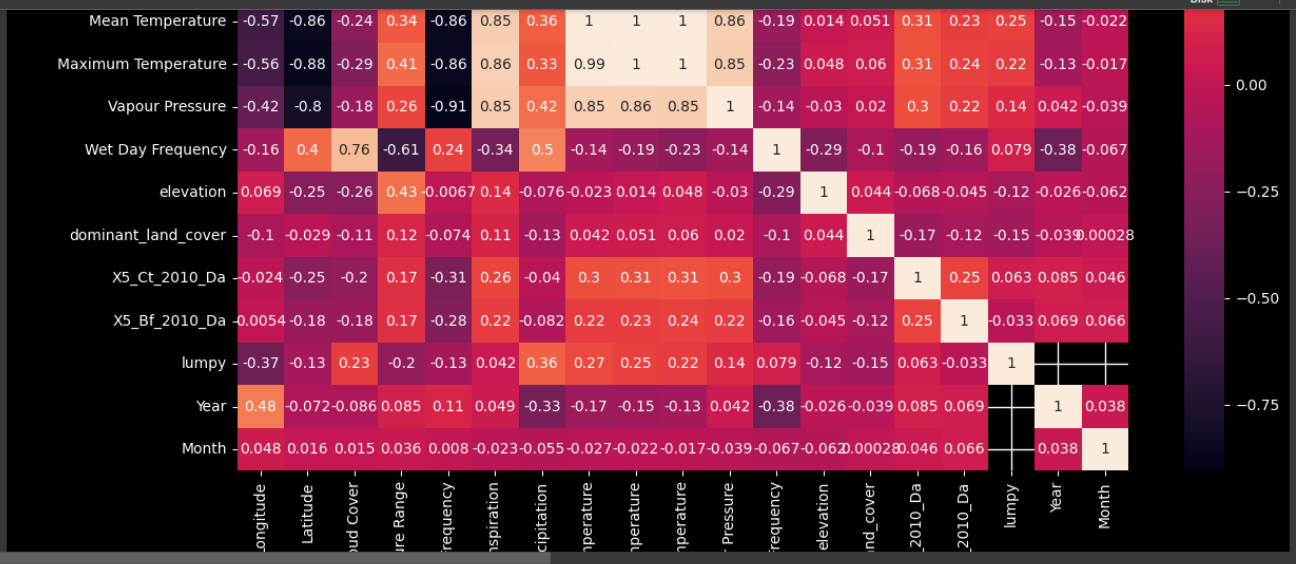
File Edit View Insert Runtime Tools Help All changes saved

Comment Share B

+ Code + Text

RAM Disk

[56]



0s completed at 11:03 PM

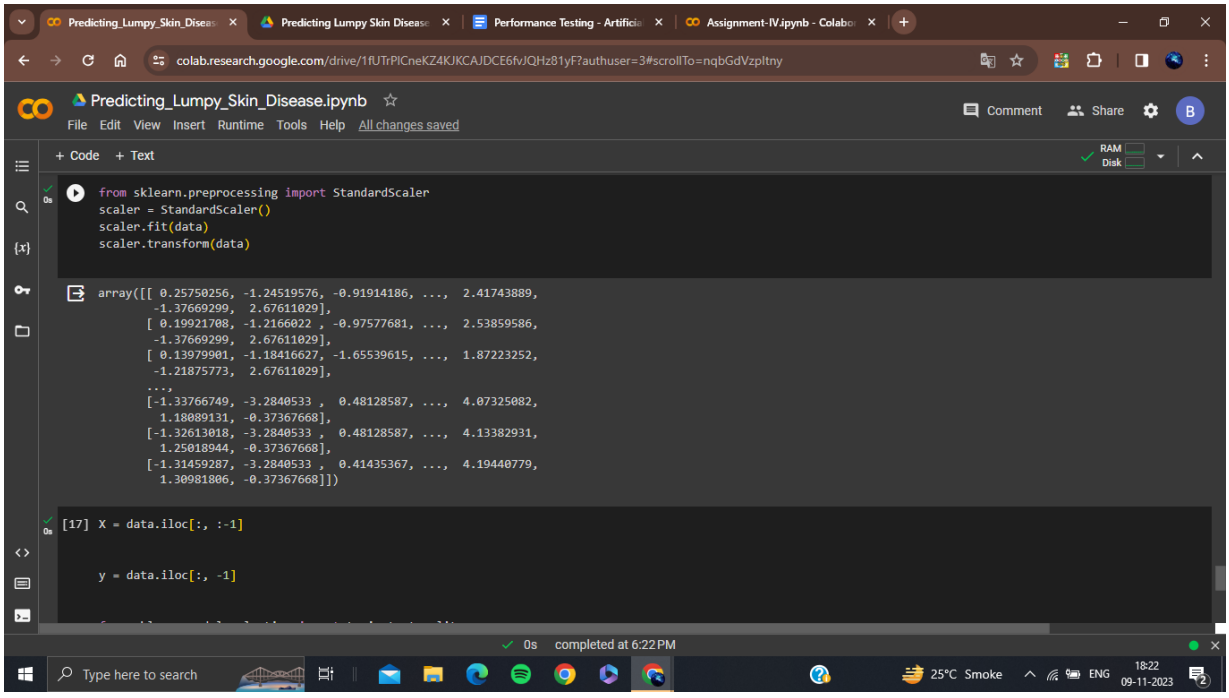
Type here to search 21°C Smoke ENG 23:06 06-11-2023

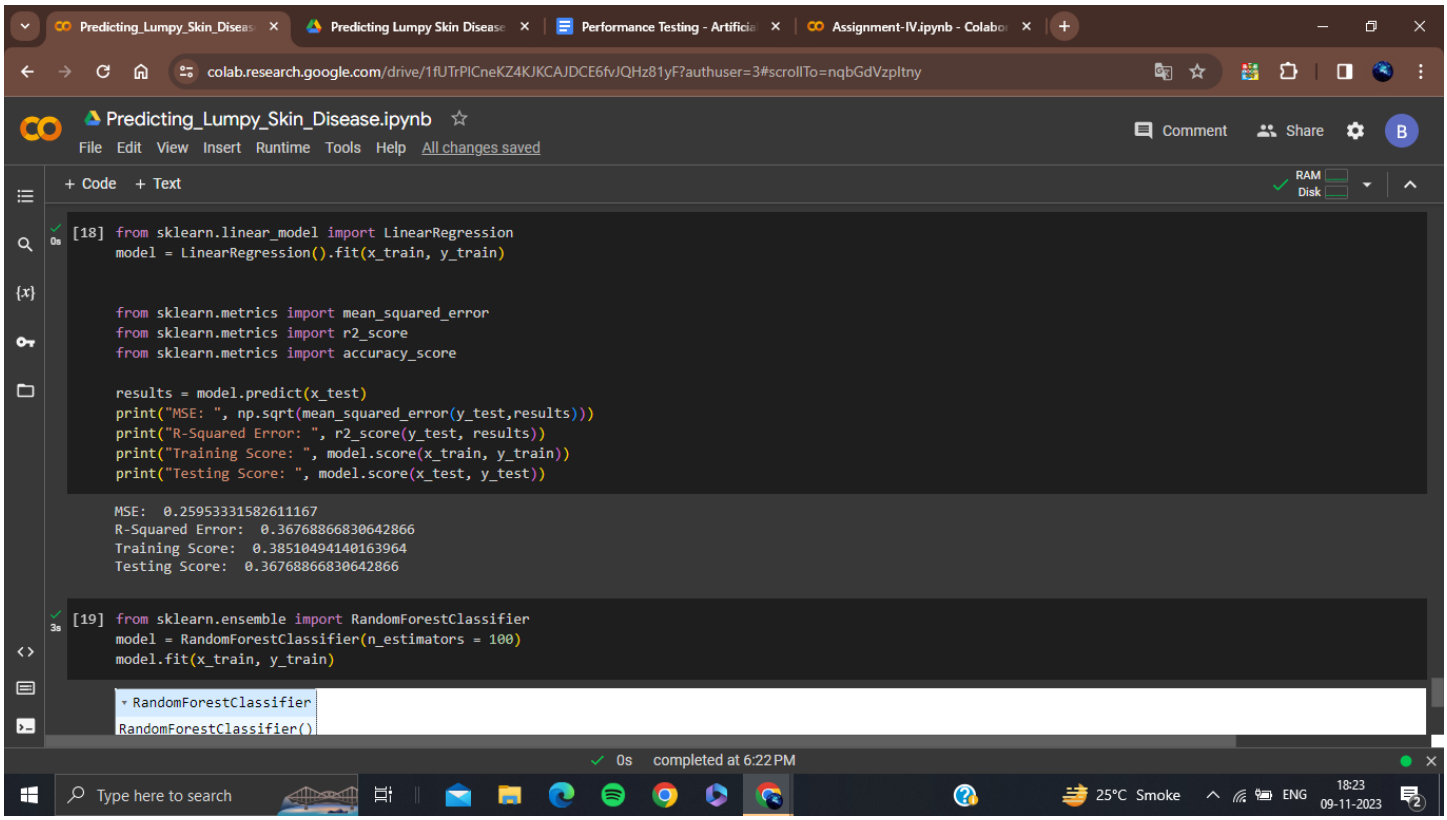
Activity 2: Training the Model with Multiple Algorithms

With the dataset now cleaned and prepared, we proceed to construct our model. To ensure a comprehensive evaluation, we train our data using multiple algorithms. In this particular project, we have selected four classification algorithms to apply. By employing this ensemble of algorithms, we can leverage their unique strengths and characteristics, enabling us to obtain a more robust and accurate model.

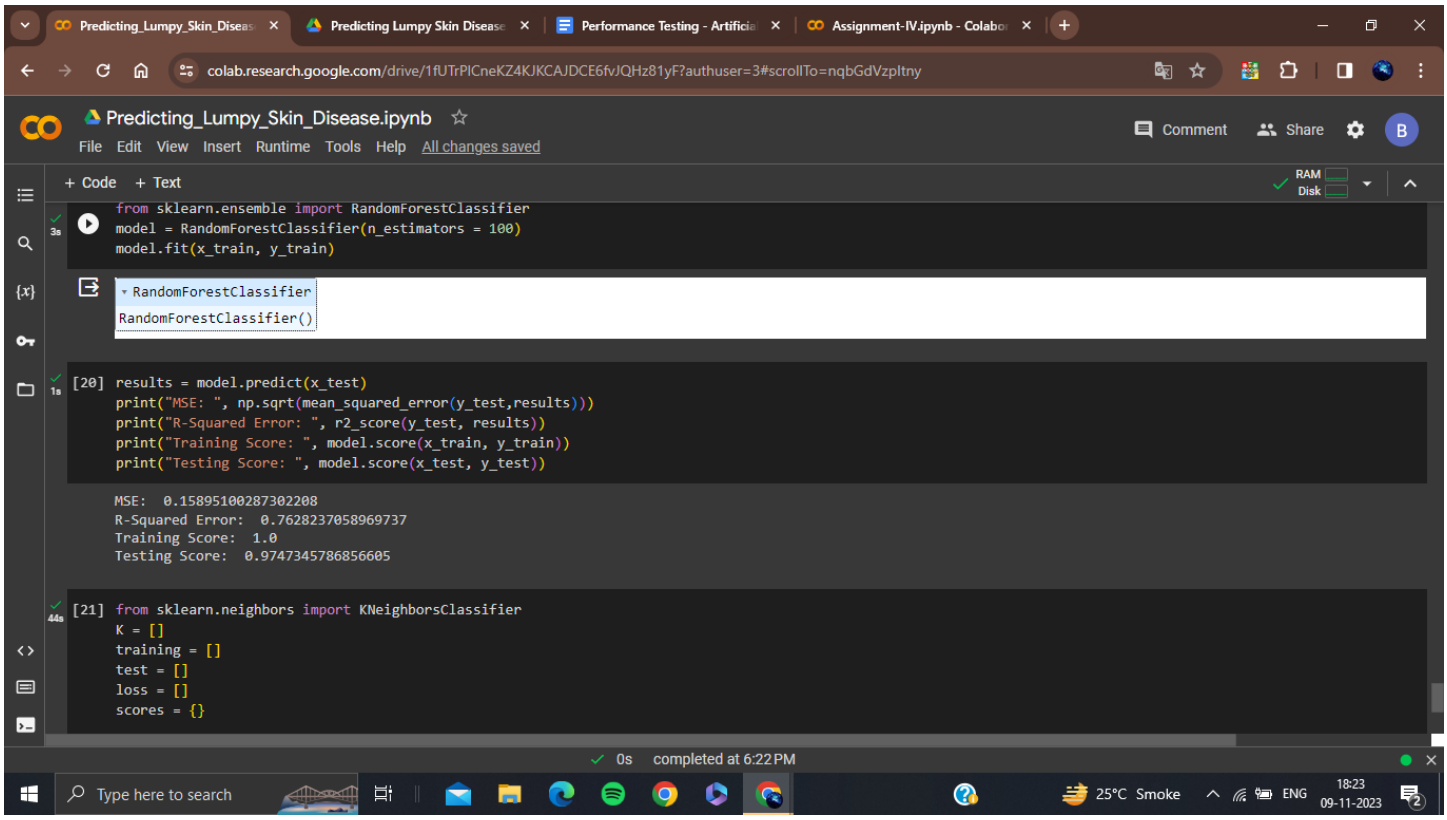
During the training process, we carefully monitor the performance of each algorithm. Based on their respective performance metrics, we identify the best-performing model. This superior model is then saved, ensuring that we retain the optimal solution for subsequent use and further analysis.

Acticity 2.1:Linear Regression Model

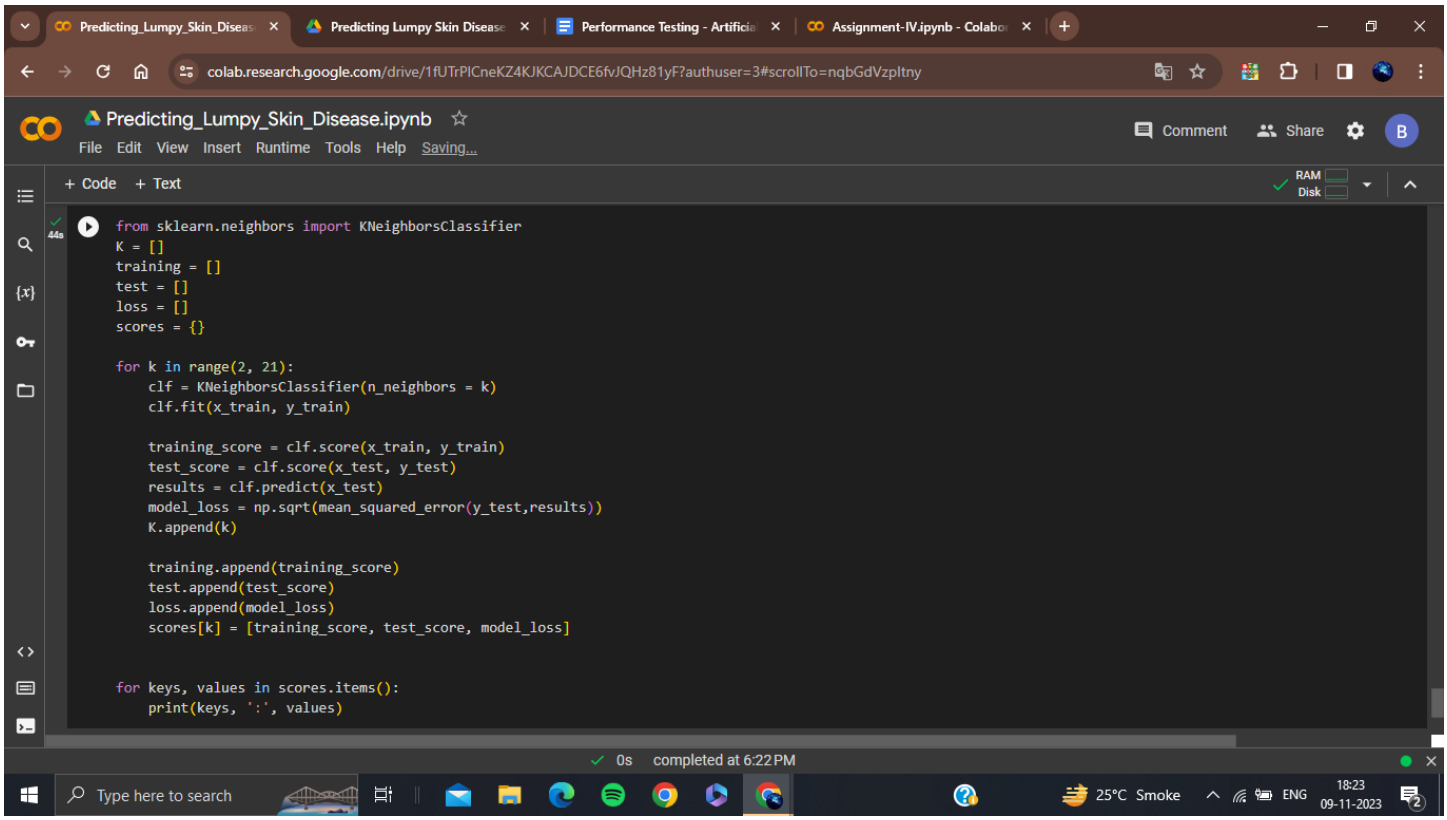


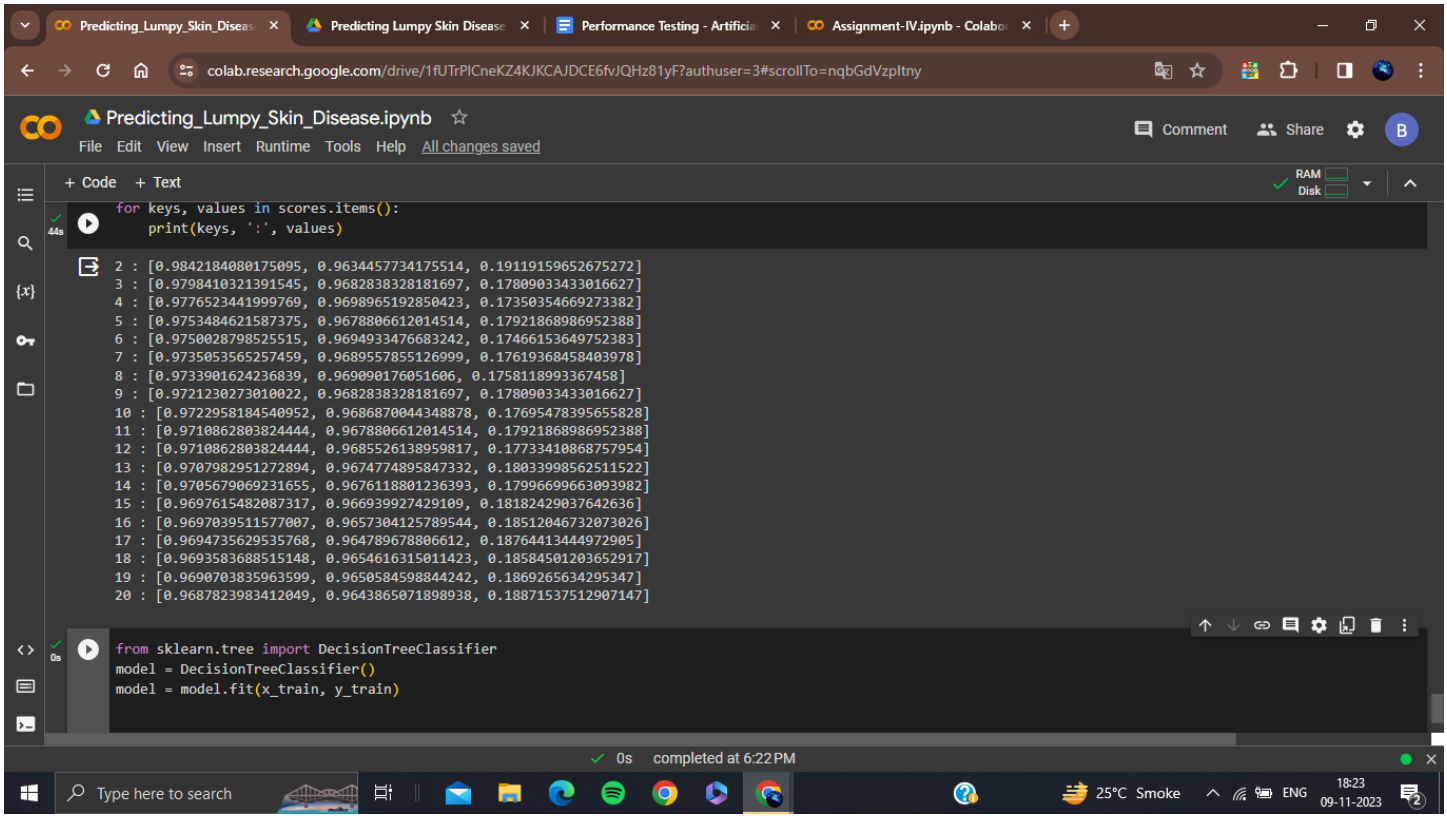


Activity2.2 Random Forest Classifier

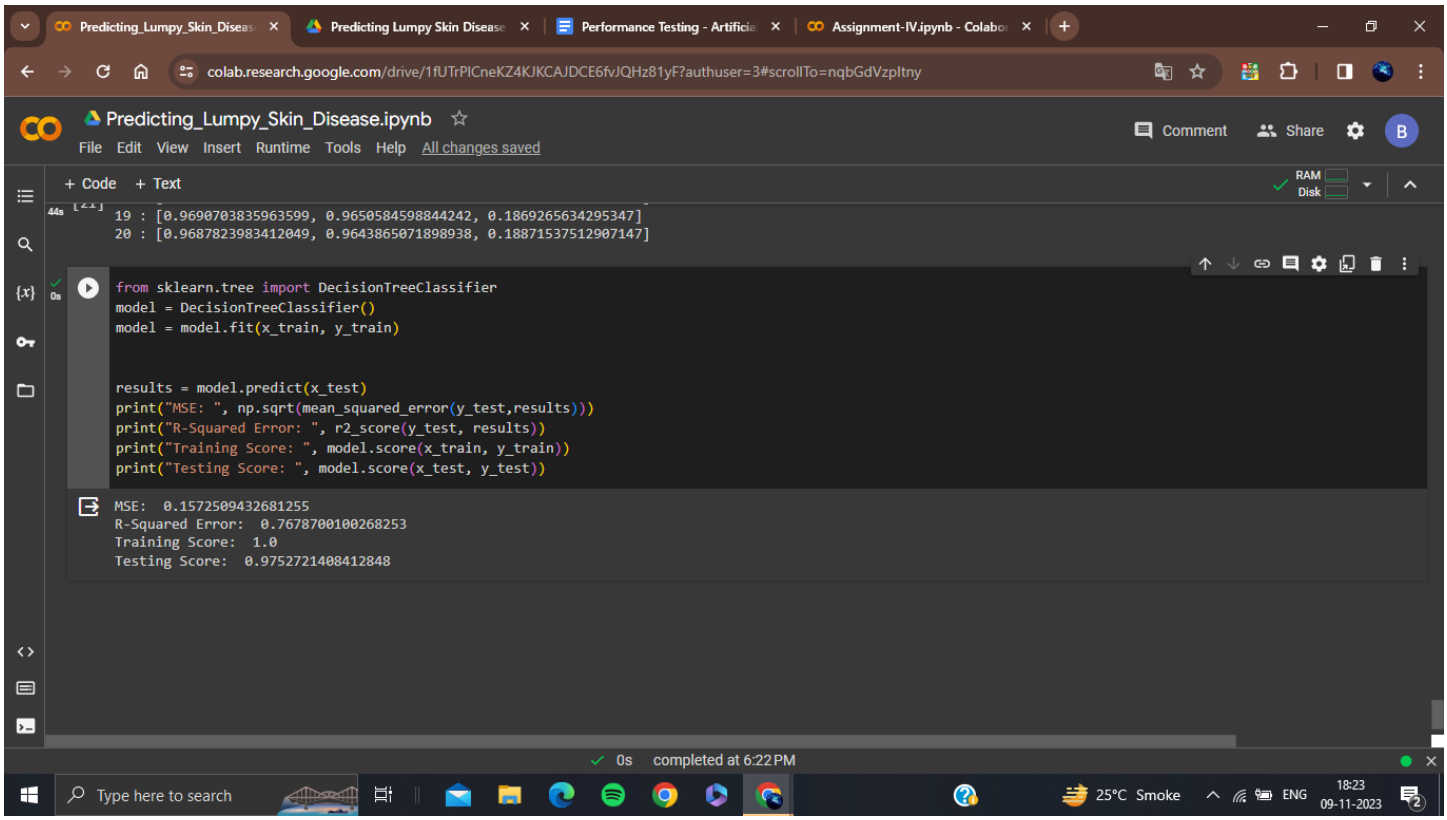


Activity 2.3 K Neighbor Classifier



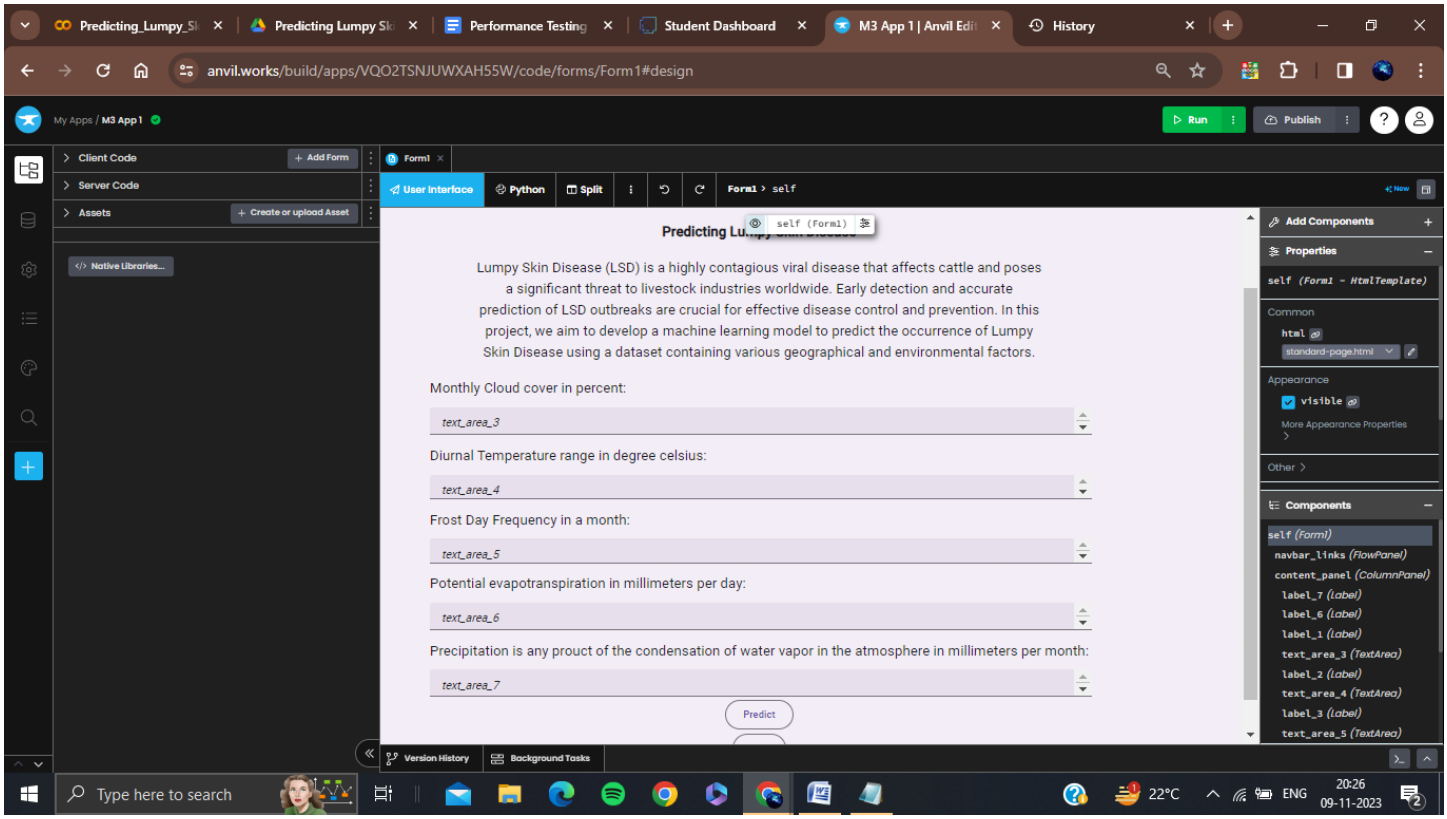


Activity 2.4 Decision Tree Classifier



Milestone 6: Model Deployment

Using Anvil and Google colab



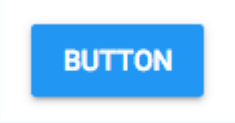
Step 1:Design the page

To classify the species of iris a flower comes from, we need to collect several measurements, so let's design the user interface for entering that data.

We construct the UI by dragging-and-dropping components from the Toolbox. Let's start by dropping a Card  into our form – this will be a neat container for the other components. Then let's add a Label  and a TextBox  into the card component:

Next we will set up the **`Label`** and **`TextBox`** components to collect enter the sepal length.

Select the **`Label`** we just added and, in the properties panel on the right, change the text to 'Sepal length: ' and align the text to the right.

Next, let's add a Button 

to run the classifier. Name it **`predict_button`** and change the text to 'Categorise'. Clicking this button will trigger a Python function to send the iris measurements to our Colab notebook. (We'll set that up in a moment.)

Step 2:Set up the predict button

We want our **`predict_button`** to do something when it's clicked, so let's add a click event.

With the button selected, go to the bottom of the properties panel. Then click the blue button with two arrows in it next to the click event box. This will open our code view and create a function called **`predict_button_click()`**. From now on, every time the button is clicked by a user, this function will be called.

Step 3:Enable the uplink

From the Anvil editor, let's enable the Uplink. This gives us everything we need to connect our web app to our Colab notebook. Select the blue '+' button in the Sidebar Menu to open the list of available services. Then add the `Uplink` and click 'Enable Server Uplink':

This will then give us an Uplink key we can use in our Google Colab notebook, to connect to this app.

Now let's install the Uplink in our Colab environment, and connect our script using the key we just created.

Step 4: Install the Uplink Library in our Colab Environment

In the next few steps, we will be connecting a Colab notebook to the web app we have built. For simplicity, I've created a notebook that already handles the iris classification for us. Make a copy of the following notebook to follow along:

The first thing we need to do is install the anvil-uplink **library in our Colab environment. Let's add** !pip install anvil-uplink **to the top of our notebook.**

Step 5:Connecting our Script

Now that the Uplink library will be installed when we start our notebook, we can connect our notebook in the same way as any other Uplink script.

Start by importing the `anvil.server` module:

Then connect to the Uplink:

Replace "your-uplink-key" with the Uplink key from your app.

That's it! When we run our notebook, it will now connect to our web app via the Uplink. Next, let's create a function we can call from our Anvil app.

Step 6 - Creating a callable function

With a classification model built and trained, we can create a function that takes our iris data and returns the name of the iris species. Let's create a `predict_` function and add `@anvil.server.callable` so it is available to call from our app. Finally at the end of our notebook we will call the `wait_forever()` function. This keeps our notebook running and allows our app to call functions indefinitely.

Step 9 - Deploying your model

Downloading your model

We'll start by going back into our Colab notebook. At the end of the cell that builds and trains the iris classification model, we'll import the `joblib` library and the files module from google.colab.

Uploading the model to our app

Now, back in the Anvil editor, let's add the Data Files service. Select the blue '+' button in the Sidebar Menu and add Data Files.

Next, we can upload our model as a file by clicking the 'Upload' button and selecting the model we downloaded earlier.

Configuring your server environment

With our model uploaded, we need to configure our app's server environment to include all the packages we need to use the model.

We'll start by selecting settings icon  from the Sidebar Menu and opening 'Python versions'.

Then, in the Python version dropdown, select 'Python 3.10'. Under 'Base packages', choose the 'Machine Learning' base image. This includes all of the packages we'll need to run the model.

Using your hosted model

Create a Server Module by selecting the App Browser  in the Sidebar Menu and clicking '+ Add Server Module'.

Inside the predict_ function, we'll reconstruct our model using `joblib.load()`. We will get the path to the model file on disk using `data_files['knn.skmodel']`. Lastly, we'll get the classification using the same code we used in our Colab notebook.