**Covid-19 Data Analysis And Prediction**

Deep Learning And Machine Learning

Bachelor Of Computer Applications

Chaudhary Charan Singh University (CCS), Meerut

**Submitted By: Submitted To:**

Parv Goyal Mr. Prateek Gupta

Sanyam (Trainer)

Rahul Sharma

Vanshaj Pal

**Institute Of Technology And Science**

**GHAZIABAD, 201007**

**Batch (2022-2025)**

**Acknowledgement**

I am profoundly thankful to everyone who contributed to the successful completion of my summer training project.

Firstly, I would like to extend my deepest gratitude to Mr. Prateek Gupta, whose invaluable guidance and insights were essential throughout the training. His expertise and patience were crucial in helping me navigate the complexities of advanced data science and machine learning.

I also wish to sincerely thank ITS for collaborating with Shape My Skills Pvt. Ltd. to provide this exceptional learning opportunity. Special appreciation goes to our course coordinator, Mr. Neeraj Jain, for his dedicated efforts and support during the training. Furthermore, I am deeply grateful to, Dr. Vidushi Singh Chairperson, for his encouragement and for providing the necessary resources and environment to facilitate this learning experience.

Finally, I acknowledge the unwavering support of my family and friends, who have been my pillars of strength and encouragement throughout this journey.

Thank you all for making this experience valuable and memorable.

Sincerely,

PARV

**Certificate**

This is to certify that Group Members has successfully completed the project titled " COVID-19 DATA ANALYSIS” as part of the Machine Learning and Data Science summer training program organized by ITS in collaboration with ShapeMySkills Pvt. Ltd.

This project was conducted under the esteemed guidance of Mr. Prateek Gupta, whose expertise and mentorship were instrumental in its successful completion. The project exemplifies a thorough understanding of machine learning and data science techniques, highlighting the skills acquired during the training program.

We commend PARV, Sanyam, Rahul Sharma and Vanshaj Pal for his dedication, hard work, and enthusiasm throughout the project duration.

**Coordinator:** Mr. Neeraj Jain

**Chairperson:** Dr. Vidushi Singh

**Date:**

**Signature:**

**List of Figures**

|  |  |
| --- | --- |
| **S.no** | **Name** |
| **1.** | Count Plot Graph |
| **2.** | Chart For “Number of positive cases” |
| **3.** | Fever presence in positive and negative case |
| **4.** | Dry Cough presence in positive and negative case |
| **5.** | Sore throat presence in positive and negative case |
| **6.** | Abroad travel occurrence in positive and negative case |
| **7.** | Fever Vs Covid-19 (Stacked Bar Plot) |
| **8.** | Probability of covid-19 given fever |
| **9.** | Heatmap |

**List of Tables**

|  |  |
| --- | --- |
| **S.no** | **Name** |
| **1.** | Table 1 shows the Measure of central tendency and five-point summary |

**List of Abbreviations**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Name** | **Page** |
| **1.** | OOP: Object Orient Programming | **1** |
| **2.** | I/O: Input/Output | **1** |
| **3.** | ML: Machine Learning | **2** |
| **4.** | AI: Artificial intelligence | **2** |
| **5.** | NumPy: Numerical Python | **4** |
| **6.** | Pandas: Panel Data | **4** |
| **7.** | CSV: Command Separated Value | **5** |
| **8.** | SQL: structured Query Language | **5** |
| **9.** | JSON: JavaScript Object Notation | **5** |
| **10.** | 3D: 3 Dimensional | **5** |
| **11.** | SciPy: Scientific Python | **6** |
| **12.** | **AUC-ROC: Area Under the Receiver Operating Characteristics Curve** | **8** |

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Title** | **Page** |
| **1.** | Chapter-1 Introduction to Python and Machine Learning | **1-3** |
| **2.** | Chapter-2 Introduction to Python libraries | **4-6** |
| **3.** | Chapter-3 Introduction to machine learning algo. | **7-9** |
| **4.** | Chapter-4 Python and machine learning code | **10-25** |
| **5.** | Chapter-5 Conclusion and Result | **26** |
| **6.** | Chapter-6 Future Scope | **26** |

**CHAPTER 1**

**Introduction to Python and Machine Learning**

**Introduction to python:**

Python is a general-purpose, dynamically typed, high-level, compiled and interpreted, garbage-collected, and purely object-oriented programming language that supports procedural, object-oriented, and functional programming.

It was Created by Guido van Rossum and first released in 1991, Python emphasizes code readability and allows programmers to express concepts in fewer lines of code compared to languages like C++ or Java.

**Why learn Python?**

* **Easy to use and Learn:** Python has a simple and easy-to-understand syntax, unlike traditional languages like C, C++, Java, etc., making it easy for beginners to learn.
* **Interpreted Language:** Python does not require compilation, allowing rapid development and testing. It uses Interpreter instead of Compiler.
* **Object-Oriented Language:** It supports object-oriented programming i.e. (inheritance, encapsulation, polymorphism, abstraction) making writing reusable and modular code easy.
* **Extensive Libraries:** Python has a rich ecosystem of libraries and frameworks, such as NumPy, Pandas, and Matplotlib, which simplify tasks like data manipulation and visualization.

**Python Popular Frameworks and Libraries:**

* **Mathematics:** - NumPy, Pandas, etc.
* **REST framework:** a toolkit for building RESTful APIs.
* **Machine Learning:** NumPy, Seaborn, Matplotlib etc.

**Where is Python used?**

* **Data Science:** Python is important in this field because it is easy to use and has powerful tools for data analysis and visualization like NumPy, Pandas, and Matplotlib.
* **Machine Learning:** Python is widely used for machine learning due to its simplicity, ease of use, and availability of powerful machine learning libraries

**Introduction to Machine Learning**

Machine learning (ML) is a subfield of artificial intelligence (AI) that involves the development of algorithms and statistical models enabling computers to perform tasks without explicit instructions. Instead, these systems learn patterns and make decisions based on data. Machine learning is transforming various industries by automating complex processes, providing insights from large datasets, and creating new opportunities for innovation.

**Definition and Scope**

Machine learning leverages computational methods to improve performance on a given task over time with experience. This process involves:

1. **Data Collection:** Gathering large and diverse datasets.
2. **Data Preprocessing:** Cleaning and formatting data to be suitable for analysis.
3. **Model Selection:** Choosing an appropriate algorithm or model based on the task.
4. **Training:** Feeding the data into the model to learn patterns.
5. **Evaluation:** Assessing the model's performance using metrics and validation techniques.
6. **Deployment:** Implementing the model in real-world applications.
7. **Maintenance:** Continuously updating and refining the model as new data becomes available.

**Types of Machine Learning-**

Machine learning techniques can be broadly categorized into three types:

1. **Supervised Learning:** The model is trained on a labelled dataset, meaning that each training example is paired with an output label. Common algorithms include:

* Linear Regression
* Decision Trees
* Support Vector Machines (SVM)
* Neural Networks

1. **Unsupervised Learning:** The model is provided with unlabelled data and must find inherent patterns or groupings. Common algorithms include:
   * Clustering (e.g., K-Means, Hierarchical Clustering)
   * Association Rules (e.g., Apriori, Eclat)
   * Principal Component Analysis (PCA)
2. **Reinforcement Learning:** The model learns by interacting with an environment, receiving rewards or penalties based on its actions, and aims to maximize cumulative rewards. Key concepts include:

* Markov Decision Processes (MDP)
* Q-Learning
* Deep Q-Networks (DQN)

**Chapter 2**

**Libraries of Python**

**NumPy**

**Introduction**

NumPy, short for Numerical Python, is a fundamental package for scientific computing with Python. It provides support for arrays, matrices, and many mathematical functions to operate on these data structures.

Some of the most popular libraries:

1. **NumPy:**

NumPy, originally stands for Numerical Python, is a core package for numerical computing in Python. It supports massive, multidimensional arrays and matrices, as well as a set of mathematical functions for effectively manipulating these arrays.

**Features of NumPy:**

* NumPy includes an extremely efficient multi-dimensional array object called numpy. ndarray, which can store and manage big datasets rapidly.
* NumPy includes a large number of numerical computing tools and methods that can operate on these arrays.
* NumPy has routines for manipulating arrays such as reshaping (reshape ()), stacking (stack (), hstack (), vstack ()), splitting (split ()), indexing (indexing and slicing), and sorting.

**Advantages of NumPy**

1. **Performance:**

* Speed: Faster than Python lists due to optimized C code.
* Vectorization: Allows element-wise operations without loops.

1. **Memory Efficiency:**

* Contiguous Allocation: Enhances cache efficiency.
* Homogeneous Types: Consistent memory usage.

1. **Pandas:**

Pandas is a flexible and advanced Python toolkit for data manipulation and analysis. It includes high-level data structures like Data Frame and Series, which make it easier to work with organized data.

**Key Features of Pandas:**

* Data Frame: A two-dimensional labelled data structure containing columns of various categories. It looks like a spreadsheet or a SQL table and is ideal for working with tabular data.
* Series: A one-dimensional labelled array that can carry data of any type (integer, float, text, etc.). Series function similarly to columns in a Data Frame or named arrays.
* Pandas can read and write data from a variety of file formats, including CSV, Excel, SQL databases, and JSON.
* Pandas includes sophisticated indexing techniques (loc and iloc) for picking subsets of data. This includes selecting rows and columns based on labels (loc) or integer positions (iloc), giving users easy access to data.

**Advantages of Pandas:**

1. **Data Manipulation and Analysis:**

* **Data Frames:** Efficiently handle tabular data with labelled axes (rows and columns).
* **Series:** Simplify manipulation of one-dimensional labelled arrays.

1. **Data Cleaning:**

* **Handling Missing Data:** Functions for detecting, filling, and removing missing values.
* **Data Transformation:** Easy methods for merging, reshaping, and transforming datasets.

1. **Data Selection:**

* **Indexing and Slicing:** Powerful, flexible, and intuitive data selection capabilities.
* **Label-based and Position-based Indexing:** Access data using labels or positions.

**3. Matplotlib**

Matplotlib is a robust Python package that allows you to create static, animated, and interactive visualizations. It is commonly used for data visualization jobs and offers a versatile framework for creating plots and figures in a variety of formats.

**Key Features of Matplotlib:**

* Matplotlib provides a wide range of graphs, including line plots, scatter plots, bar plots, histogram plots, pie charts, 3D plots, and more.
* Matplotlib works seamlessly with Pandas and NumPy, enabling direct plotting from Data Frame and Series objects.
* Matplotlib works well with Seaborn, a statistical data visualization toolkit in Python. Seaborn expands Matplotlib's capabilities by providing higher-level functions for statistical plots such as violin plots, box plots, and regression plots.

**Advantages of Matplotlib**

1. **Versatility:**

* **Wide Range of Plots:** Supports various types of plots such as line, bar, scatter, histogram, pie, and more.
* **Customizable:** Highly customizable plots, allowing for detailed adjustments to suit specific needs

**2. Integration:**

* **Seamless with NumPy and Pandas:** Easily integrates with NumPy and Pandas for plotting data from arrays and Data Frames.
* **Compatible with Other Libraries:** Works well with other libraries like SciPy and scikit-learn for enhanced functionality.

**3. Publication Quality:**

* **High-Quality Output:** Produces high-quality figures suitable for publication.
* **Multiple Formats:** Exports plots in various formats including PNG, PDF, SVG, and EPS

**4. Seaborn**

Seaborn is a Python data visualization package built on Matplotlib. It provides a high-level interface for constructing visually appealing and useful statistical graph.

**Key Features of Seaborn:**

* Seaborn provides a straightforward and intuitive API for constructing complicated statistical graphs, using less lines of code than Matplotlib.
* Seaborn includes built-in colour palettes to improve visualizations. It provides both qualitative (categorical data), sequential (numeric data), and divergent (data with a crucial midpoint) colour schemes.
* Seaborn works perfectly with Pandas Data Frames, allowing for direct visualization of data contained in Data Frame objects

**Advantages of Seaborn:**

**1. High-Level Interface:**

* **Ease of Use:** Simplifies complex visualizations with fewer lines of code.
* **Intuitive API:** Provides a user-friendly syntax for creating informative and attractive statistical graphics.

**2. Statistical Visualization:**

* **Built-in Support:** Directly integrates with Pandas Data Frames and handles statistical aggregations and visualizations effortlessly.
* **Advanced Plotting Functions:** Includes specialized plots like categorical plots, distribution plots, and regression plots.

**Chapter 3**

**Introduction to Machine Learning Algorithm**

**Introduction to Categorial**

A categorical variable, also known as a discrete variable, is a type of variable that can take on one of a limited, fixed number of possible values, representing distinct categories or classes.

Examples include:

Binary categories: Yes/No, True/False, 0/1

Multiclass categories: Red/Green/Blue, Dog/Cat/Horse, Low/Medium/High

**Goal:**

The primary objective of categorical machine learning algorithms is to predict the category or class of new instances based on learned patterns from a labelled dataset. This process is known as classification.

**Key Characteristics of Categorical Data**

* Categorical data has discrete values.
* Non-ordinal vs. ordinal: Some categorical data is non-ordinal (e.g., fruit types), whilst others are ordinal (e.g., rating scales such as "low," "medium," and "high").
* Encoding Required: Many machine learning techniques require categorical data to be translated into numerical format before they can be used.

**Types of Algorithms**

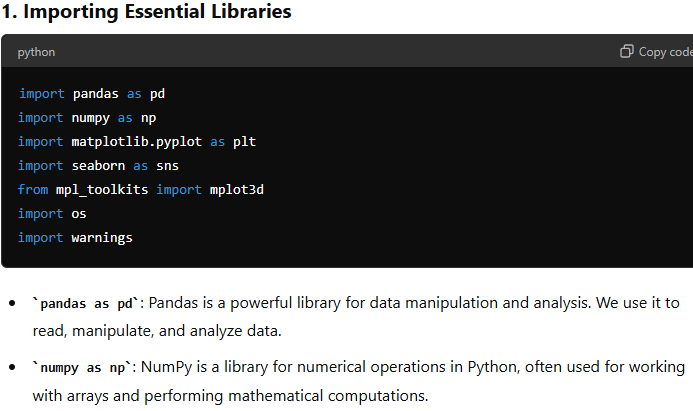
Categorical algorithms are built specifically for processing and analysing categorical data, which is made up of variables that indicate discrete categories or groups. Several types of categorical algorithms have been created to handle the specific issues caused by categorical data, ensuring that machine learning models execute accurately and efficiently.

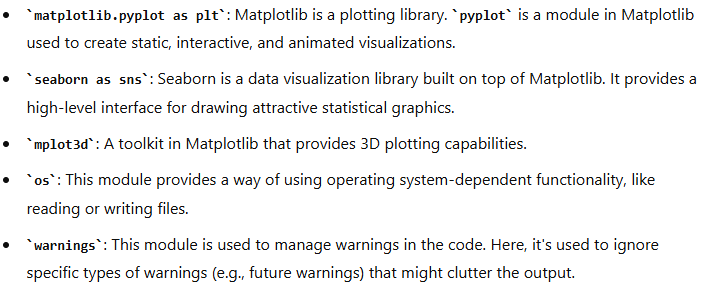
* **Logistic Regression**
* Logistic regression is often utilized in binary classification situations. It calculates the likelihood that a given input belongs to a particular category.
* It represents the relationship between a binary dependent variable and one or more independent variables
* Uses a logistic function (sigmoid function) to convert the relationship to a probability value between 0 and 1.
* Assumes that the independent variables have a linear connection with the dependent variable's log chances.
* Accuracy, precision, recall, F1-score, and area under the ROC curve (AUCROC) are common metrics used for evaluation
* **Decision Tree**
* Decision trees are hierarchical, tree-like structures that make judgments depending on input features.
* Recursively divides the data into subsets based on the values of the input features.
* Each node represents a feature, whereas each branch denotes a decision rule or conclusion.
* Terminal nodes (leaves) indicate the ultimate conclusion or classification. Suitable for both classification and regression workloads.
* The visual, tree-like form makes it easy to learn and interpret.
* Overfitting is common, especially with complicated trees, but it can be minimized using strategies such as pruning.
* **Random Forest**
* Random forest is an ensemble learning method that uses several decision trees to increase forecast accuracy and robustness.
* During training, a large number of decision trees are constructed and the findings are combined to provide a more accurate and reliable forecast.
* During tree creation, a random subset of characteristics is selected for splitting at each node, increasing tree variety.
* Averaging the outcomes of numerous trees reduces the risk of overfitting, which is common with individual decision trees.
* Generally, achieves good accuracy and robust performance on a variety of datasets.
* **K- Nearest Neighbour:**
* KNN is an instance-based learning algorithm that makes predictions based on the similarity of fresh data points to the training set.
* Unlike many other algorithms, KNN does not require a formal training phase. It saves the complete training dataset and uses it in the prediction phase.
* The "K" in KNN denotes the number of nearest neighbours to consider when making a forecast. The choice of K influences the algorithm's performance
* KNN, or classification, determines the class label by a majority vote of the K nearest neighbours.
* The value of K is critical; too little K can lead to overfitting, while too much K can lead to underfitting. Cross-validation is commonly used to select an optimal K.
* **Gaussian Naive Bayes:**
* Gaussian Naive Bayes is a probabilistic classifier based on Bayes' Theorem that assumes feature independence.
* Calculates the likelihood of each class based on the feature distribution, and then assigns the data point to the class with the highest probability.
* For each new data point, it calculates the likelihood of belonging to each class using the Gaussian distribution parameters and chooses the class with the highest posterior probability.
* Gaussian Naive Bayes is prized for its simplicity and efficiency, especially when the feature independence criterion is roughly met and the features have a Gaussian distribution

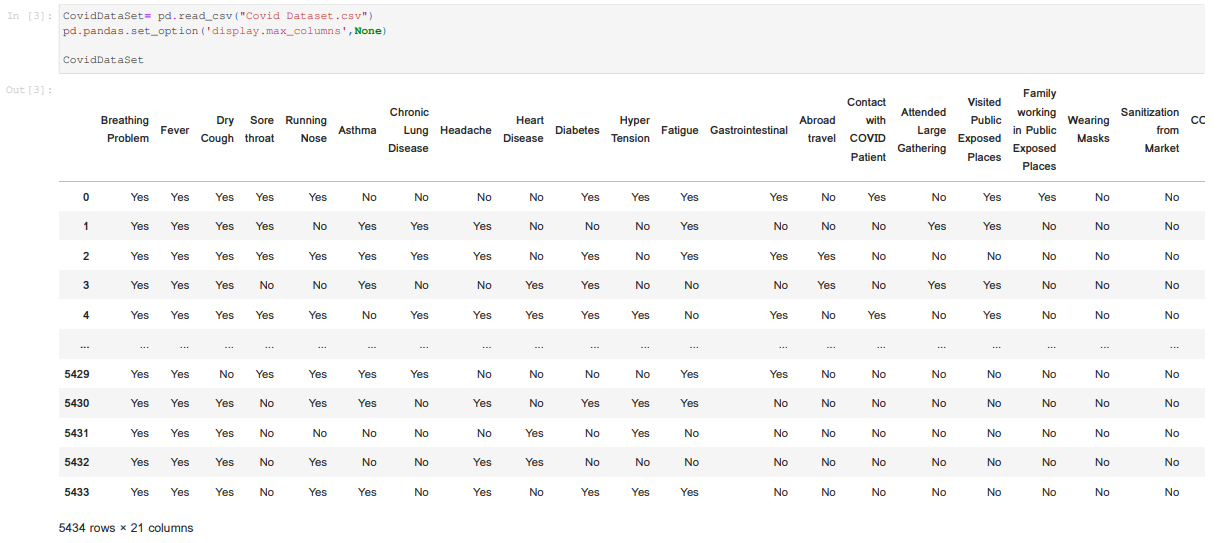
**Chapter 4**

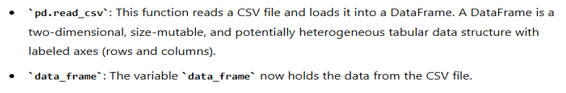
**Python and Machine Learning Code**

* **Libraries**

****

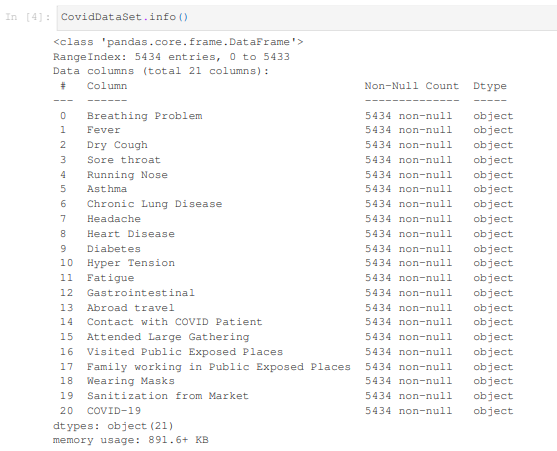
****

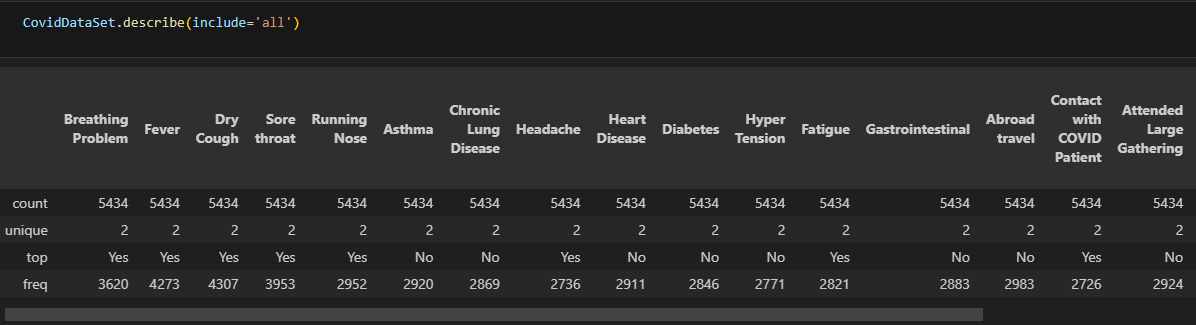
* ** DATASET READ:**

****

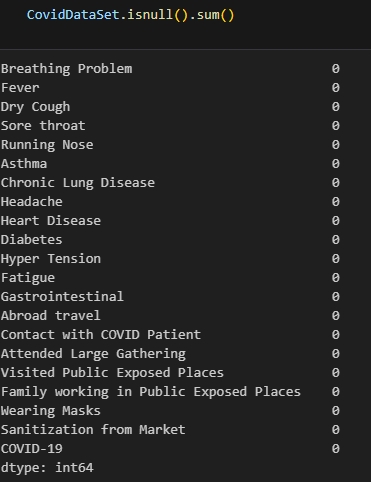
* **Gathering Some Basic Information:**

**df.info ():** is a function that returns the count of rows, null status and the datatype of each column.

****

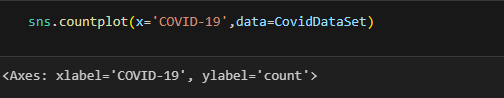
**Table 1: Table 1 shows the Measure of central tendency and five-point summary**

CovidDataSet.isnull().sum() is a chain function formed using isnull() and sum() functions it shows the sum of null values in a column.

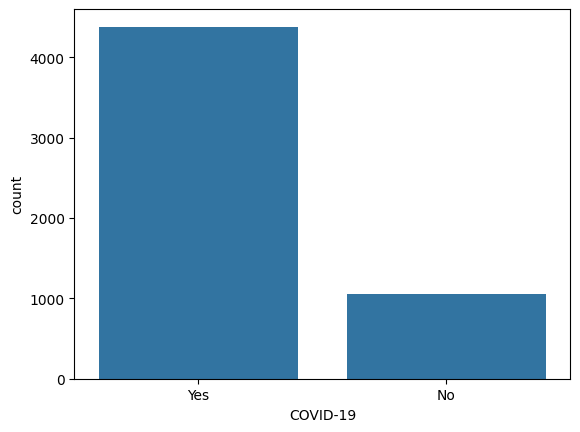
****

**Plotting graphs and Visualization:**

1. **Count Plot Graph:**

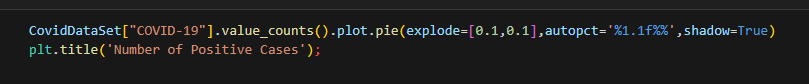
****

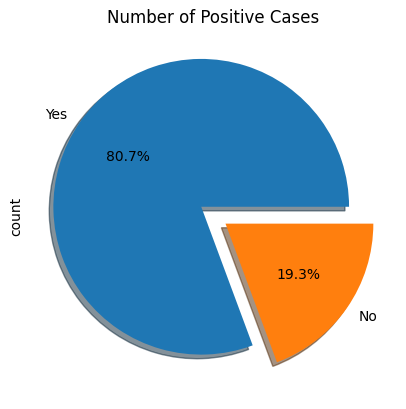
<Axes: xlabel='COVID-19', ylabel='count'>

****

The bar plot depicts the distribution of COVID-19 positive and negative cases in the dataset. The x-axis represents the COVID-19 outcome, categorized as "Yes" (positive) and "No" (negative), while the y-axis shows the count of occurrences for each category.

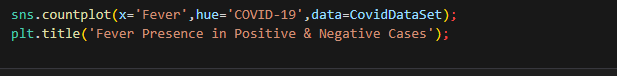
**2. Chart For “Number Of Positive Cases”:**

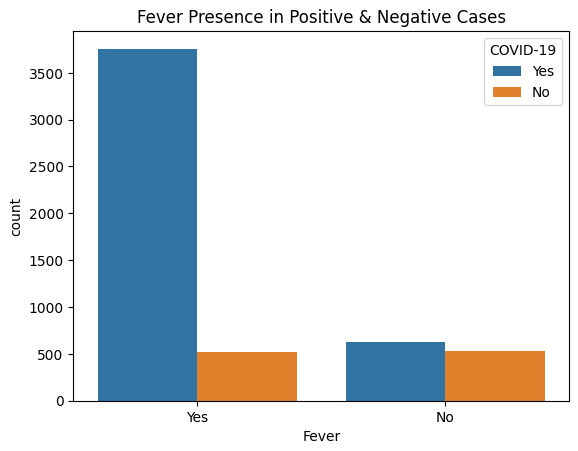
****

****

The pie chart shows the distribution of positive cases. The majority of cases (80.7%) are positive, while 19.3% are negative. The chart is titled "Number of Positive Cases" and has labels for "Yes" and "No" to indicate the categories.

**3.** **Fever Presence In Positive And Negative Cases:**

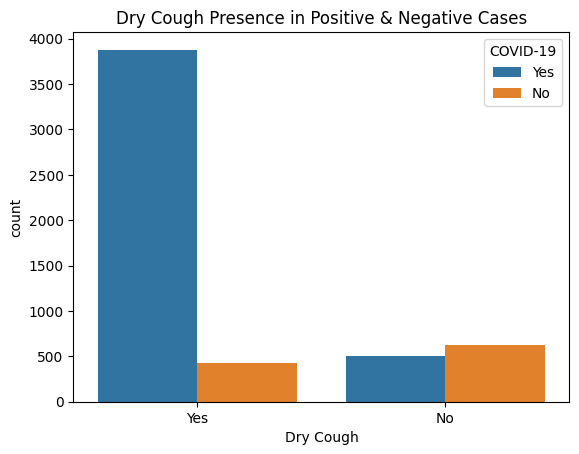
****



The bar chart shows the presence of fever in positive and negative COVID-19 cases. The x-axis represents the presence of fever ("Yes" or "No"), and the y-axis represents the count of cases.

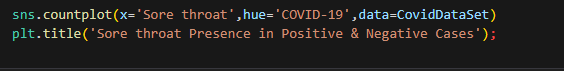
**4. Dry cough present in positive and negative cases:**

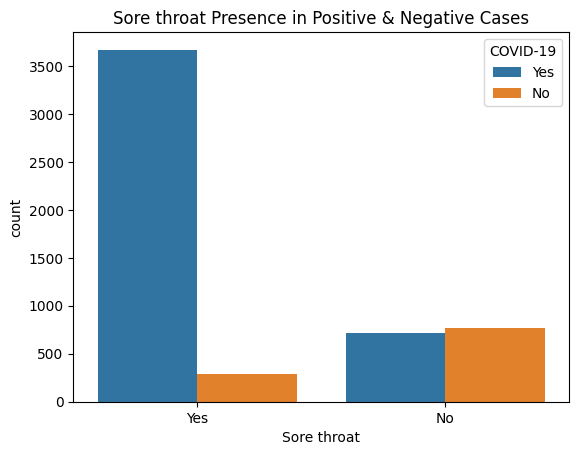
****

****

The bar chart shows the presence of a dry cough in positive and negative COVID-19 cases. The x-axis represents the presence of a dry cough ("Yes" or "No"), and the y-axis represents the count of cases.

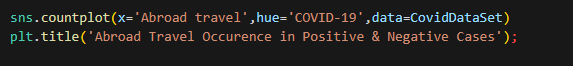
**5. Sore throat present in positive and negative cases:**

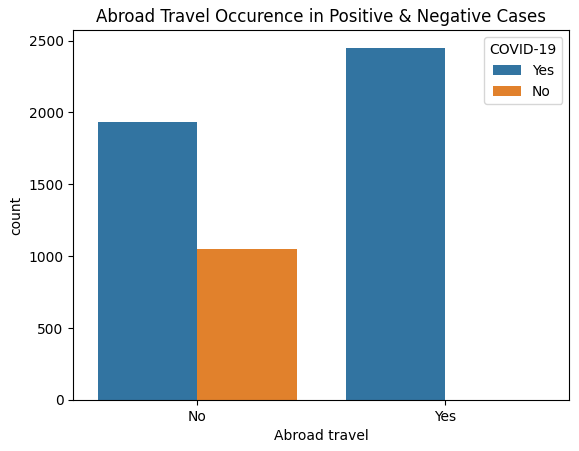
****

****

The bar chart shows the presence of a sore throat in positive and negative COVID-19 cases. The x-axis represents the presence of a sore throat ("Yes" or "No"), and the y-axis represents the count of cases.

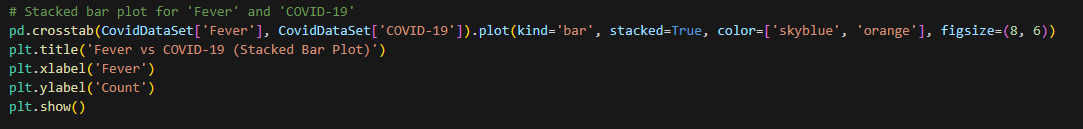
**6. Abroad travel occurrence in positive and negative cases:**

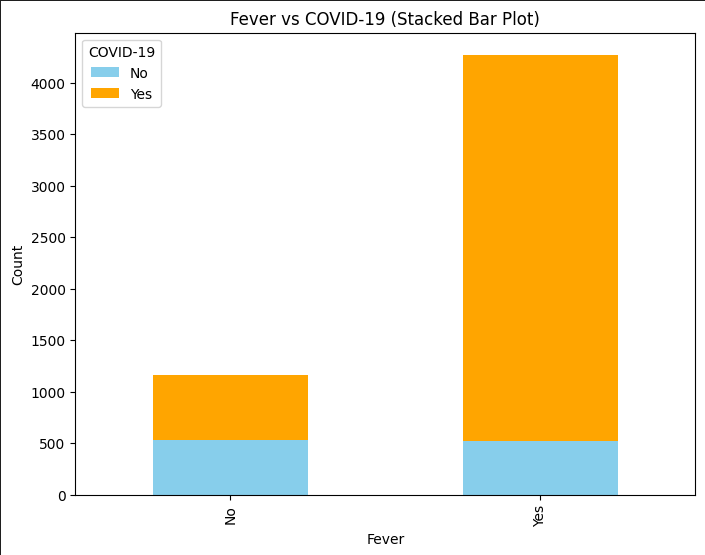
****

****

The bar chart shows the occurrence of abroad travel in positive and negative COVID-19 cases. The x-axis represents whether the individual has travelled abroad ("No" or "Yes"), and the y-axis represents the count of cases.

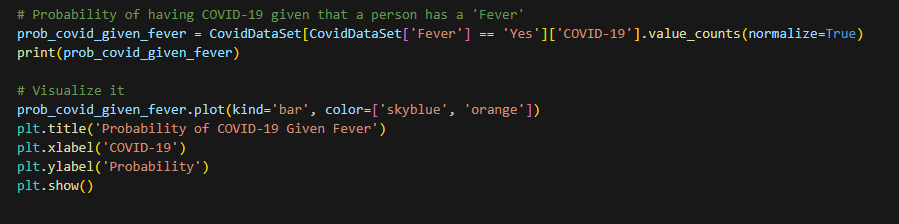
**7. Fever Vs covid-19 (Stacked bar plot):**

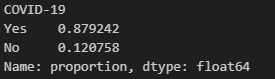
****

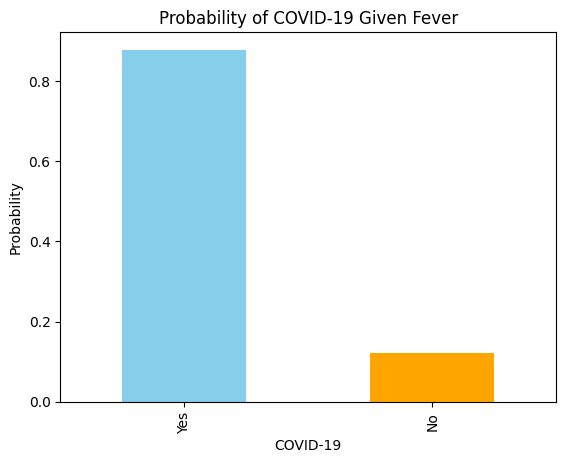
****

The stacked bar chart shows the relationship between fever and COVID-19 cases. The x-axis represents the presence of fever ("No" or "Yes"), and the y-axis represents the count of cases.

**8. Probability of covid-19 given fever:**

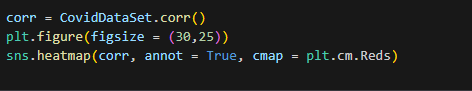
****

****

****

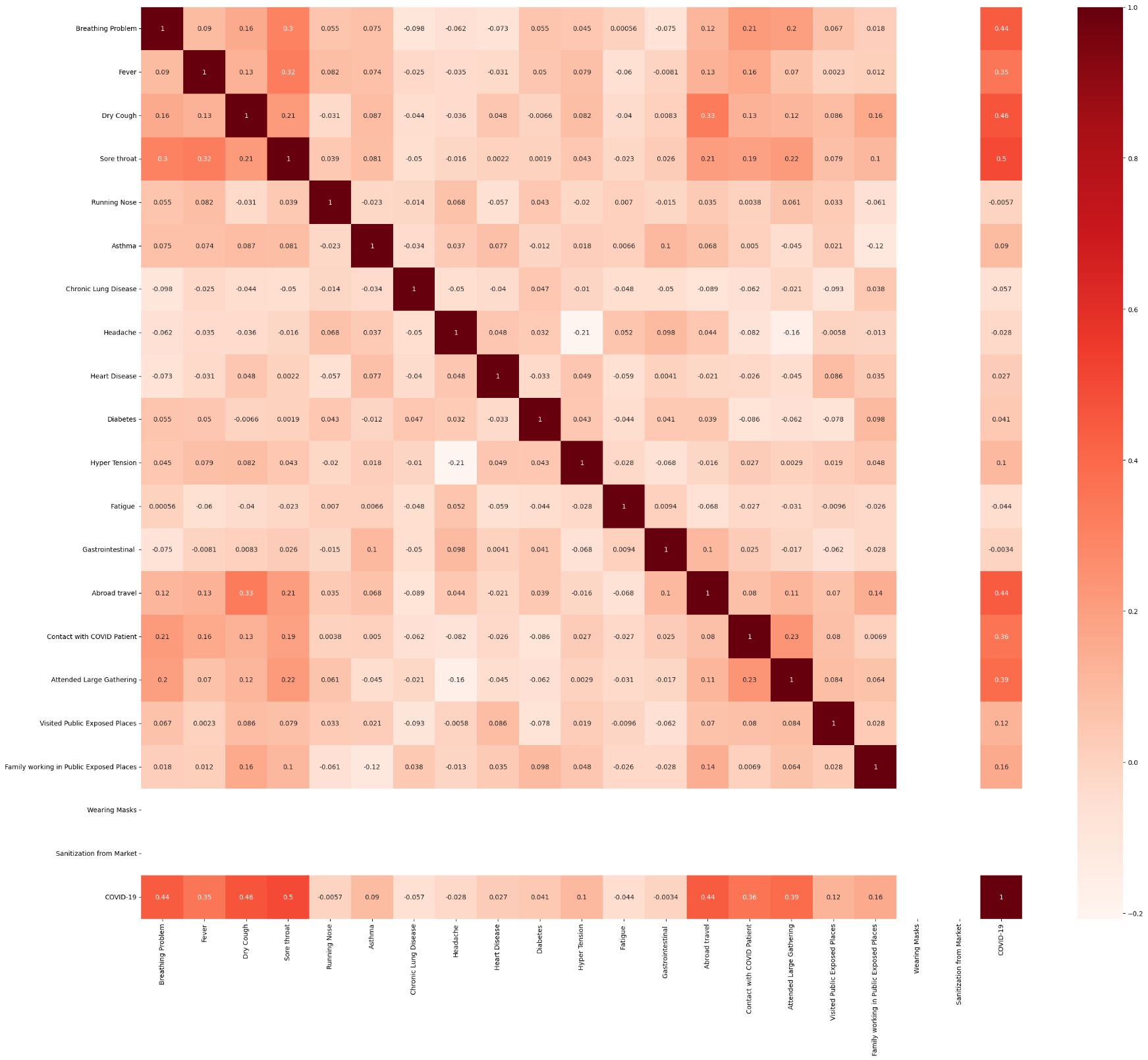
The bar chart shows the probability of having COVID-19 given that a person has fever. The x-axis represents the presence of COVID-19 ("Yes" or "No"), and the y-axis represents the probability.

**9. Heatmap:**

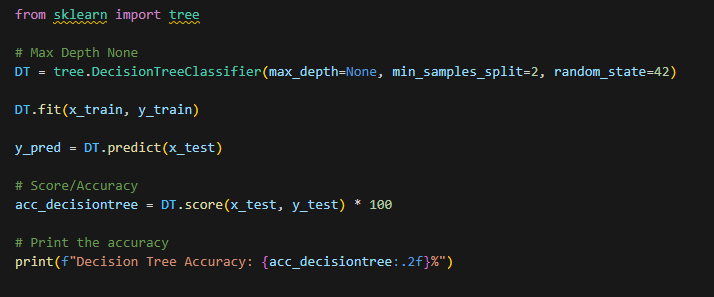
****

****

The heatmap shows the correlations between different variables related to COVID-19. Darker colours indicate stronger correlations, either positive or negative. The specific variables and their correlations can be interpreted by examining the labels and values on the heatmap.

****

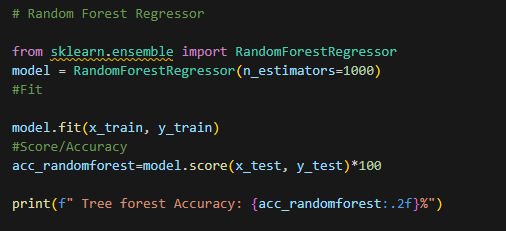
**Decision Tree Accuracy:**

****

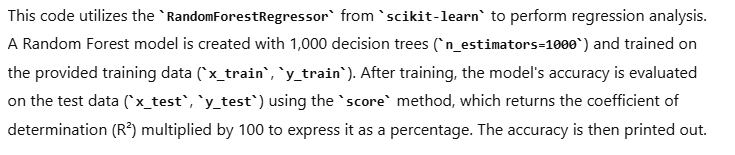
****

This code implements a Decision Tree classifier using ‘scikit-learn’. The model is created with no depth limit (‘max\_depth=None’) and is trained on the training data (‘x\_train’, ‘y\_train’). After training, it predicts labels for the test data (‘x\_test’). The model's accuracy is calculated by comparing these predictions with the actual test labels (‘y\_test’) and is printed as a percentage. The random seed is set to ensure reproducibility.

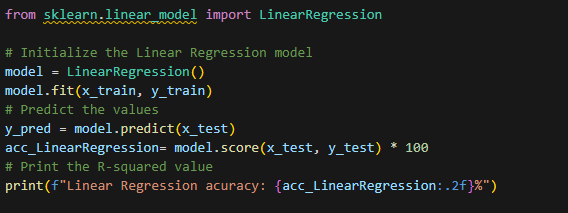
**Random Forest Regressor:**



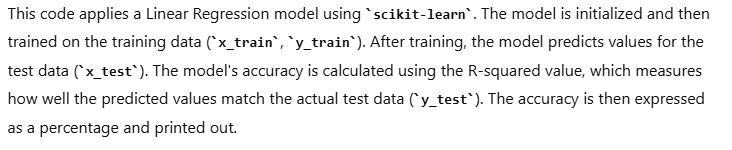




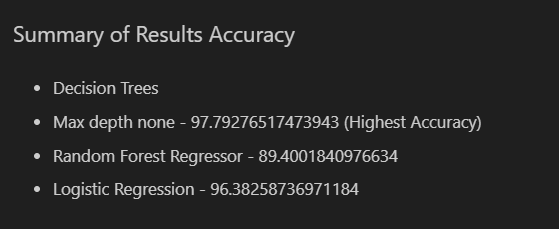
**Linear Regression:**







**SUMMARY OF ACCURACY:**

****

Decision Tree

**Chapter 5: Conclusion and Results**

In conclusion, the evaluation of various models on the COVID-19 DATA ANALYSIS dataset revealed that the Random Forest algorithm is the most effective, followed closely by Decision Tree and KNN. Logistic Regression, while less accurate than the tree-based models, still provides a solid performance. On the other hand.

For the COVID-19 DATA prediction task, the overall accuracy across all algorithms was 90% suggesting a need for further model improvement and optimization. These findings can guide future efforts in COVID-19 DATA ANALYSIS task, with an emphasis on enhancing model performance and accuracy.

**Chapter 6: Future Scope of Covid-19 Data Analysis Tasks**

The future scope of COVID-19 DATA prediction using various machine learning algorithms is both vast and promising. Enhanced data collection methods, such as integrating advanced Vaccine System and utilizing real-time data sources, will provide comprehensive datasets that capture detailed aspects of patients, WHO region wise charts etc. The implementation of advanced machine learning algorithms, including deep learning, ensemble learning, and reinforcement learning, along with improved feature engineering, will significantly enhance model accuracy. Developing features that better capture the nuance of COVID-19 data, such as death ratio, recovered ratio, active case, external economic factors, will further refine predictive capabilities.