A SHORT-TERM INTERNSHIP REPORT ON

ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

BY

Sri Lekhana Jutthu

Mounika Jyosthna

Shivani

Sneha Pidugu

III BCA

Under the esteemed Guidence of

Mr. G.V.S.S PRASANTH SIR

(Tutor of Artificial Intelligence & Machine Learning)

(Affiliated to **ANDHRA UNIVERSITY**)

Gajuwaka-530026, Visakhapatnam, Andhra Pradesh 2022-2025

ADITYA DEGREE COLLEGE

DECLARATION BY THE STUDENT

I hereby declare that the work described in this short-term Internship, entitled "Artificial Intelligence & Machine Learning" which is being submitted by me in partial fulfilment of the requirements for the award of degree of Bachelor of Computer Applications from the Department of Bachelor of Computer Applications to Aditya Degree College Women's, Gajuwaka under the guidance of Mr. G.V.S.S PRASANTH Sir tutor of Artificial Intelligence & Machine Learning in Aditya Degree College Women's, Gajuwaka.

Place: Gajuwaka

Date:

(Sri lekhana, Sneha,
Mounika, Shivani)

ADITYA DEGREE COLLEGE

CERTIFICATE FROM THE SUPERVISOR

This is to certify that the Short-Term Internship entitled, "ARTIFICIAL INTELLIGENCE & MACHINE LEARNING", that is being submitted by Sri lekhana, Shivani, Sneha, Mounika of III BCA which is being submitted to us in partial fulfilment of the requirements for the award of degree of Bachelor of Computer Applications from the department of Bachelor of Computer Applications to Aditya Degree College Women's, bonafide work carried out by him under my guidance and supervision.

(Mr. G.V.S.S PRASANTH SIR)

ACKNOWLEDGEMENT

We are all started this internship as a complete beginners. We would like to take this opportunity to extend our sincere gratitude to all those who have contributed to the successful completion of this Short-Term Internship Project Report.

We express our deep sense of gratitude to Mr. PRADEEP KUMAR Sir, Principal, for his efforts and for giving us permission for carrying out this Short-Term Internship.

We feel deeply honoured in expressing my sincere thanks to Mr. G.V.S.S Prasanth Sir tutor of ULearn for making the resources available at right time and providing valuable insights leading to the successful completion of my Short-Term Internship Project Report.

Finally, We all thank the Aditya Degree College and all the faculty members of our department who contributed their valuable suggestions in completion of Short-Term Internship. Now we are all ready with academic and practical learnings to shape our career for the future.

Thank you.

(Sri lekhana, Sneha,

Shivani, Mounika)

CONTENTS

- Bayes theorem
- LSTM- long short-term memory
- Restricted Boltzmann Machine
- RNN- Recurrent Neural Network
- Auto encoders and types
- VGG Net and architecture
- Google Net and architecture
- Data types in Python
- Arithmetic operations in python
- Declaration of comments and variables
- Reserved words in python
- Control statements in python
- Programs

- **Problem statement & Explanation**
- **Source and Outputs**
- **Conclusion**

INTRODUCTION

Air pollution refers to the introduction of harmful substances into the atmosphere, leading to adverse effects on health, the environment, and the climate. These substances, known as pollutants, can be in the form of gases, liquids, or particles. Air pollution can come from natural sources, such as wildfires and volcanic eruptions, as well as human activities, including industrial processes, vehicle emissions, and agricultural practices.

In machine learning (ML) and artificial intelligence (AI), the concept of air pollution often involves using advanced techniques to monitor, analyse, predict, and mitigate the impacts of pollutants in the atmosphere. Here's a detailed breakdown of how ML and AI are applied to the issue of air pollution. Machine learning and AI are revolutionizing the way we monitor, predict, and manage air pollution. These technologies offer powerful tools for improving our understanding of pollution dynamics, enhancing public health, and supporting environmental policy and decision-making

LEARNING OUTCOME OF SHORT-TERM INTERNSHIP

Introduction to AI & ML:

Artificial Intelligence (AI) is a branch of computer science. It stands for artificial intelligence and dedicated to creating systems that can perform tasks requiring human-like intelligence. This field encompasses various techniques and approaches aimed at enabling machines to replicate or simulate aspects of human cognition, such as learning, reasoning, problem-solving, and decision-making.There are two subsets in AI.

They are:

1. Machine Learning

2. Deep Learning

Machine Learning:

It is a subset of AI which focus on the use of data and algorithms to imitate the way that human learn and gradually increasing its accuracy. It learns from data & solve the problems.

Deep Learning:

A subset of AI that focuses on developing algorithms that enable computers to learn from and make predictions based on data without being explicitly programmed.
A specialized form of machine learning that uses neural networks with many layers (deep networks) to model complex patterns and representations in large datasets.

Machine Learning:

It is a subset of AI which focus on the use of data and algorithms to imitate the way that human learn and gradually increasing its accuracy. It learns from data & solve the problems.

Types of Machine Learning:

There are three types of machine learning:

1. supervised learning – it is a labelled data or structured data
2. unsupervised learning – it is un-labelled data or unstructured data
3. reinforcement learning -it uses both structured data and unstructured data

Supervised Learning is a type of machine learning where an algorithm is trained on a labelled dataset. In supervised learning, the data used for training includes input-output pairs, where the output (or label) is known. The goal of supervised learning is to learn a mapping from inputs to outputs that can be applied to new, unseen data. In supervised learning, each training example is paired with an output label or value. This means the dataset includes both the features (input variables) and the target (output variable).

Supervised learning is classified into two categories of algorithms:

- **Regression:** The goal is to predict a continuous output value based on input features. Regression is a type of supervised learning that is used to predict continuous values, such as house prices, stock prices, or customer churn.

  Some common regression algorithms include:
  - Linear Regression
  - Polynomial Regression
  - Support Vector Machine Regression
  - Decision Tree Regression
  - Random Forest Regression

- **Classification** is a type of supervised learning that is used to predict categorical values, such as whether a customer will churn or not, whether an email is spam or not, or whether a medical image shows a tumour or not. Classification algorithms learn a function that maps from the input features to a probability distribution over the output classes.

  Some common classification algorithms include:
  - Logistic Regression

- Support Vector Machines
- Decision Trees
- Random Forests
- Naive Baye

Unsupervised Learning: - Unsupervised Learning is a type of machine learning where the algorithm is trained on data without labelled responses or predefined output values. In unsupervised learning, the model attempts to identify patterns, structures, or relationships in the data on its own, without any guidance on what the outcomes should be. This type of learning is useful for exploring data and discovering hidden patterns or features.

Unsupervised learning is classified into two categories of algorithms:

❖ **Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behaviour. Clustering is a type of unsupervised learning that is used to group similar data points together. Clustering algorithms work by iteratively moving data points closer to their cluster centres and further away from data points in other clusters.

Some types of clustering are:

- Hierarchical clustering
- K-means clustering
- Principal Component Analysis
- Singular Value Decomposition
- Independent Component Analysis

❖ **Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y. Association rule learning is a type of unsupervised learning that is used to identify patterns in a data. Association rule learning algorithms work by finding relationships between different items in a dataset.

Some common association rule learning algorithms include:

- Apriori Algorithm
- Eclat Algorithm
- FP-Growth Algorithm

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by interacting with an environment. The goal is to find an optimal strategy, or policy, that maximizes cumulative rewards over time. Unlike supervised learning, where the model is trained on labelled data, reinforcement learning involves learning through trial and error, receiving feedback in the form of rewards or penalties based on actions taken.

Some algorithms that learn from outcomes and decide which action to take next.

- Policy
- Reward function
- Value function
- Model of the environment

Applications of Machine Learning:

Artificial Intelligence (AI) has a broad and transformative impact across various industries and domains. Its applications are diverse, leveraging its capabilities to enhance processes, create new solutions, and improve outcomes.

Here's a comprehensive overview of some key **applications of Machine learning:**

1. Healthcare:

- Medical Diagnosis: AI algorithms can analyse medical images (e.g., X-rays, MRIs) and data to assist in diagnosing diseases such as cancer, diabetic retinopathy, and cardiovascular conditions.

- Personalized Medicine: AI helps in developing tailored treatment plans based on individual patient data and genetic information

2. Finance:
- Fraud Detection: AI systems analyse transaction patterns to detect and prevent fraudulent activities.

- Algorithmic Trading: AI algorithms execute high-frequency trading strategies and analyse market trends to make investment decisions.

3. Manufacturing:
- Predictive Maintenance: AI predicts equipment failures and schedules maintenance to prevent downtime.
- Quality Control: AI systems inspect products for defects and ensure consistent quality.

Deep Learning:

Deep learning differs from traditional machine learning techniques in that they can automatically learn representations from data such as images, video or text, without introducing hand-coded rules or human domain knowledge. Their highly flexible architectures can learn directly from raw data and can increase their predictive accuracy when provided with more data.
Computer vision apps use deep learning to gain knowledge from digital images and videos. Conversational AI apps help computers understand and communicate through natural language. Recommendation systems use images, language, and a user's interests to offer meaningful and relevant search results and services.
Deep learning has led to many recent breakthroughs in AI such as Google DeepMind's AlphaGo, self-driving cars, intelligent voice assistants and many more. With NVIDIA GPU-accelerated deep learning frameworks, researchers and data scientists can significantly speed up deep learning training, that could otherwise take days and weeks to just hours and days. When models are ready for deployment, developers can rely on GPU-accelerated inference platforms for the cloud, embedded device or self-driving cars, to deliver high-performance, low-latency inference for the most computationally-intensive deep neural networks.

ANN, NLP, CC:

ANN (Artificial neural networks):

Commonly, Artificial Neural Network has an input layer, an output layer as well as hidden layers. The input layer receives data from the outside world which the neural network needs to analyse or learn about. Then this data passes through one or multiple hidden layers that transform the input into data that is valuable for the output layer. Finally, the output layer provides an output in the form of a response of the Artificial Neural Networks to input data provided.

NLP (Natural language processing):

Natural Language Processing (NLP) is a subfield of artificial intelligence (AI) and machine learning (ML) that focuses on the interaction between computers and human languages. In Deep learning applications, second application is NLP. NLP, the Deep learning model can enable machines to understand and generate human language.

Some of the main applications of deep learning in NLP include:

1. Machine Translation
2. Text Classification
3. Language Modelling

CC (Congestion Control):

Congestion Control is a mechanism that controls the entry of data packets into the network, enabling a better use of a shared network infrastructure and avoiding congestive collapse. Congestive-Avoidance Algorithms (CAA) are implemented at the TCP layer as the mechanism to avoid congestive collapse in a network. There are two congestion control algorithm which are as follows:

- Leaky Bucket Algorithm: -
  The leaky bucket algorithm discovers its use in the context of network traffic shaping or rate-limiting. This algorithm is used to control the rate at which traffic is sent to the network and shape the burst traffic to a steady traffic stream. The large area of network resources such as bandwidth is not being used effectively.
- Token bucket Algorithm: -

In some applications, when large bursts arrive, the output is allowed to speed up. This calls for a more flexible algorithm, preferably one that never loses information. Therefore, a token bucket algorithm finds its uses in network traffic shaping or rate-limiting. It is a control algorithm that indicates when traffic should be sent. This order comes based on the display of tokens in the bucket.

AI Tools In Our Daily Life:

Artificial Intelligence (AI) tools have increasingly become a part of our daily lives, often in ways that might not be immediately noticeable. Here are some common AI tools and technologies integrated into everyday activities:

1) Virtual Assistants

- Siri (Apple), Google Assistant, Amazon Alexa, and Microsoft Cortana help with tasks like setting reminders, answering questions, playing music, and controlling smart home devices.

2) Search Engines

- Google Search, Bing, and Yahoo use AI algorithms to provide relevant search results, personalized recommendations, and predictive search suggestions based on user queries.

3) Email and Messaging

- Spam Filters: AI algorithms identify and filter out unwanted spam emails.
- Smart Replies: AI suggests quick responses in email and messaging apps like Gmail and WhatsApp.

4) Social media

•Content Recommendations: Platforms like Facebook, Instagram and Twitter use AI to recommend posts, friends, and ads based on your interactions and preferences. Image Recognition: AI helps in tagging friends in photos and identifying objects in images.

5) Navigation and Maps

• Google Maps and Apple Maps use AI to provide real-time traffic updates, suggest optimal routes, and estimate travel times

Back Propagation:

Backpropagation is a key algorithm used in training artificial neural networks, allowing them to learn from errors and improve their performance by adjusting the weights of the network. Backpropagation, short for "backward propagation of errors," is a method for optimizing the weights of a neural network by propagating the error gradient backward through the network. It aims to minimize the loss function, which measures how far the network's predictions are from the actual values.

Calculation of weights:

Following are the terms to keep in mind while calculating weights: -

- Weights should be in decimal value.
- Sum of neurons or predictive values is called bias.
- From the adjacent diagram, the terms are: -
- i1 and i2 are input values
- h1 and h2 are hidden layers
- o1 and o2 are output values
- b1 and b2 are bias values
- w1, w2, ……. are weights

Now for NETh1 = w1*i1 + w2*i2 + b1*1

= 0.15*0.05 + 0.20*0.10 + 0.35*1 =0.3775

Now for NETh2 = w3*i1 + w4*i2 + b2*1

= 0.25*0.05 + 0.30*0.10 + 0.60*1= 0.645

Difference between Neural and Deep Neural Networks:

The differences between the neural networks and deep learning neural networks are tabulate as follows:

s.no	Differences in	Neural Networks	Deep Learning Neural Networks
1.	Definition	A neural network is a model of neurons inspired by the human brain. It is made up of many neurons that at inter-connected with each other.	Deep learning neural networks are distinguished from neural networks on the basis of their depth or number of hidden layers.
2.	Architecture	Feed Forward Neural Networks Recurrent Neural Networks Symmetrically Connected Neural Networks	Recursive Neural Networks Unsupervised Pre-trained Networks Convolutional Neural Networks
3.	Structure	Neurons Connection and weights Propagation function Learning rate	Motherboards PSU RAM Processors
4.	Performance	It gives low performance compared to Deep Learning Networks.	It gives high performance compared to neural networks.

5.	Task Interpretation	Your task is poorly interpreted by a neural network.	The deep learning network more effectively.

Differences between ChatGPT & Google

The differences between CHATGPT & GOOGLE are tabulated as follow: -

s.no	CHATGPT	GOOGLE
1.	CHATGPT is an AI powered tool.	GOOGLE is a search engine.
2.	It works like a chat box between the user and server.	It gives information by showing different websites.
3.	It gives the information based on the question entered.	It shows the relevant information in different sites.
4.	Data may not be accurate	Gives most accurate data.
5.	It gives the answer based on the information it trained on.	Gives the answer based on the searches and reviews.
6.	It focused on generating humanlike texts.	It can be used for variety of tasks like voices and image recognition.
7.	It provides an answer based on the personal views and subjective views found in the data.	It provides information based on the Articles opinions of experts and activists.
8.	It is an Artificial intelligence model.	It is a worldwide search engine.
9.	Gives the information from its data source.	Gives the information that is already on the internet
10.	It is developed by OpenAI.	It is developed by Google Inc.

POS Tagging:

Part-of-Speech (POS) tagging is a fundamental task in Natural Language Processing (NLP) and involves assigning parts of speech (like nouns, verbs, adjectives, etc.) to each word in a text. This task is crucial for understanding the grammatical structure of sentences and is used in various NLP applications.

Rule-Based Methods:

- **Handcrafted Rules**: These methods rely on a set of grammatical rules to assign tags. Rules might be based on the word itself or its surrounding context. For example, a rule might state that if a word follows a determiner (like "the"), it is likely a noun.

- **Pattern Matching**: This involves matching words against predefined patterns or template.

OBJECT DETECTION:

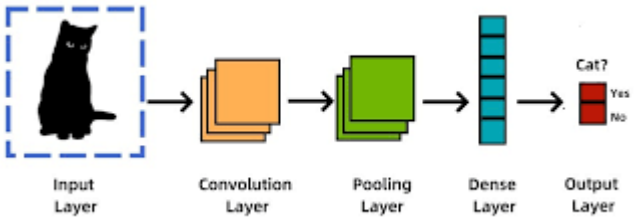
Object detection is a computer vision technique used to identify and locate objects within images or video streams. It combines classification (identifying what an object is) with localization (determining where it is) by drawing bounding boxes around detected objects.

Object detection is the computer vision task of detecting instances (such as humans, buildings, or cars) in an image. Object detection models receive an image as input and output coordinates of the bounding boxes and associated labels of the detected objects. An image can contain multiple objects, each with its own bounding box and a label (e.g. it can have a car and a building), and each object can be present in different parts of an image (e.g. the image can have several cars). This task is commonly used in autonomous driving for detecting things like pedestrians, road signs, and traffic lights. Other applications include counting objects in images, image search, and more.

objects in images using a single deep neural network. It is a supervised learning algorithm that takes images as input and identifies all instances of objects within the image scene. The object is categorized into one of the classes in a specified collection with a confidence score that it belongs to the class. Its location and scale in the image are indicated by a rectangular bounding box. Object detection allows us to at once classify the types of things found while also locating instances of them within the image.

CNN Algorithm:

A convolutional neural network (CNN) is a regularized type of feed forward neural network that learns features by itself via filter (or kernel) optimization. Vanishing gradients and exploding gradients, seen during backpropagation in earlier neural networks, are prevented by using regularized weights over fewer connections. CNNs are also known as **shift invariant** or **space invariant artificial neural networks** (**SIANN**), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation-equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are not invariant to translation, due to the down sampling operation they apply to the input.



Deep Fake, Deep Dream:

Deep Fake, a portmanteau of "deep learning" and "fake," refers to media that has been digitally altered to replace a person's face or body with that of another. Deepfake videos have been doing the rounds on social media of late and have been used to create fake news, hoaxes, and fake content.

How Does a Deep Fake Works?

Deepfakes use advanced deep learning techniques to first encode features, then reconstruct images from the encoded features. Autoencoders, a type of neural network, are the most commonly used deep learning architecture for creating deepfakes.

Deep Dream is a computer vision program developed by Google that uses convolutional neural networks (CNNs) to enhance and modify images in a way that produces surreal and often psychedelic results. It was introduced by Google engineer Alexander Mordvintsev in 2015 and has since gained popularity for its artistic and creative applications

How Does a Deep Dream Works?

The DeepDream algorithm tries to modify the input image and in the process, it boosts some of the neurons more than others. We can specify the type of layer and neuron we want to strengthen precisely. The process will continue until all the elements of the input image have been disclosed appropriately.

GAN Model & Architecture:

Generative adversarial networks (GANs) are an exciting recent innovation in machine learning. GANs are generative models: they create new data instances that resemble your training data. For example, GANs can create images that look like

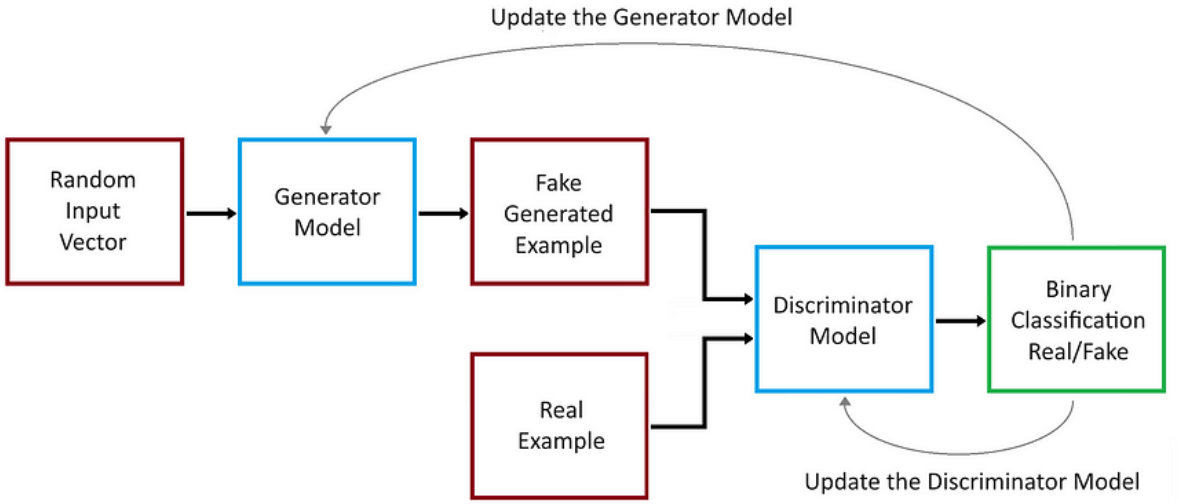
photographs of human faces, even though the faces don't belong to any real person. These images were created by a GAN:

GAN Architecture:

A GAN Architecture consists of two neural networks, namely the Generator and the Discriminator, which are trained simultaneously through adversarial training.

- Generator: This network takes random noise as input and produces data (like images). Its goal is to generate data that's as close as possible to real data.

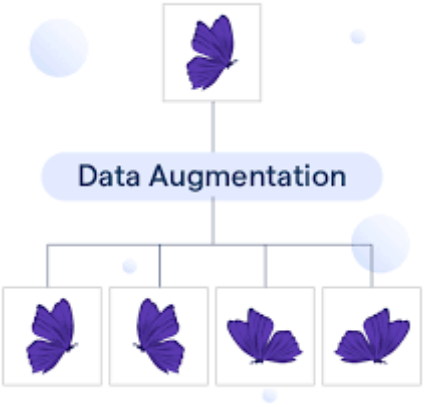
- Discriminator: This network takes real data and the data generated by the Generator as input and attempts to distinguish between the two. It outputs the probability that the given data is real.



Data augmentation:

Data augmentation refers to perturbing an image through transformations, including flipping, cropping, rotating, scaling etc. The purpose of data augmentation is to produce additional samples of the class under the underlying category unchanged. Data augmentation can be used in training, testing, or both. In a sense, the

performance of deep learning is improved via using a large amount of data. Similarly, the detection performance of small objects can also be boosted by increasing the types and numbers of small objects samples in the dataset. Kisantal et al. investigated the problem of small object detection task and carefully analyzed the state-of-the-art model called Mask-RCNN on MS-COCO dataset. They demonstrate that one of the factors behind the poor detection performance for small objects is lack of representation of small objects in a training set. That is to say, only a few images contain small objects, and small objects do not appear enough even within each image containing small objects.



Parameter Sharing and Typing:

Parameter sharing forces sets of parameters to be similar as we interpret various models or model components as sharing a unique set of parameters. We only need to store only a subset of memory.

Parameter tying is a regularization technique. We divide the parameters or weights of a machine learning model into groups by leveraging prior knowledge, and all parameters in each group are constrained to take the same value. In simple terms, we want to express that specific parameter should be close to each other.

Ensemble Methods:

Ensemble methods are learning algorithms that construct a set of classifiers and then classify new data points by taking a (weighted) vote of their predictions. The original ensemble method is Bayesian averaging, but more recent algorithms include error-correcting output coding, Bagging, and boosting. This paper reviews these methods and explains why ensembles can often perform better than any single classifier. Some previous studies comparing ensemble methods are reviewed, and

some new experiments are presented to uncover the reasons that Adaboost does not overfit rapidly.

Bagging:

- Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy data set.
- Bagging is a popular ensemble learning technique that focuses on reducing variance and improving the stability of machine learning models.
- One of the primary goals of bagging is to reduce overfitting by exposing each base learner to slightly different variations of the training data.
- The most well-known algorithm for bagging is the Random Forest.
- One key advantage of bagging is its ability to handle noisy data and outliers effectively.

Boosting:

- Boosting, like bagging, is an ensemble learning technique, but it aims to improve the performance of weak learners by combining them in a sequential manner.
- The core idea behind boosting is to give more weight to misclassified instances during the training process, enabling subsequent learners to focus on the mistakes made by their predecessors.
- Unlike bagging, boosting does not rely on bootstrapped subsets of the data.
- The most well-known boosting algorithm is AdaBoost (Adaptive Boosting).

- ❖ AdaBoost assigns a weight to each weak learner based on its performance, and the final prediction is made by combining the weighted predictions of all weak learners.

Bayes Theorem:

Bayes theorem is one of the most popular machine learning concepts that helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred. Bayes theorem gives a mathematical rule for inverting conditional probabilities, allowing us to find the probability of a cause given its effect. The theorem can be mathematically expressed as:

$P(A/B) = P(B/A) * P(A)/P(B)$

Where:

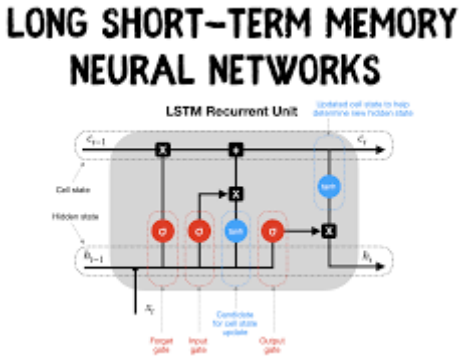
P(A/B) is the posterior probability of event A given event B.

(B/A) is the likelihood of event B given event A.

P(B) is the total probability of event B.

LSTM (Long Short – Term Memory):

Long Short-Term Memory (LSTM) is a type of recurrent neural network with a strong ability to learn and predict sequential data. The research shows that RNN is limited in maintaining long-term memory. Therefore, the LSTM was invented to overcome this limitation by adding memory structure, which can maintain its state over time, with gates to decide what to remember, what to forget and what to output. The LSTM shows effective results in many applications that are inherently sequential such as speech recognition, speech synthesis, language modeling and translation and handwriting recognition.



Architecture of LSTM

The LSTM architecture consists of a cell (the memory part of LSTM), an input gate, an output gate and a forget gate. Each of these components has a specific role in the functioning of the LSTM.

- **Cell:** The cell stores the state of a sequence, so it has the ability to either keep or forget certain information.
- **Input Gate:** It decides the extent of information to be stored in the cell.
- **Output Gate:** It determines what the next hidden state should be.
- **Forget Gate:** It decides what information should be thrown away or kept.

RBM- Restricted Boltzmann Machine:

Restricted Boltzmann Machines are stochastic two layered neural networks which belong to a category of energy based models that can detect inherent patterns automatically in the data by reconstructing input. They have two layers visible and hidden. Visible layer has input nodes (nodes which receive input data) and the hidden layer is formed by nodes which extract feature information from the data and the output at the hidden layer is a weighted sum of input layers. It might seem strange but they don't have any output nodes and they don't have typical binary output through which patterns are learnt. The learning process happens without that capability which makes them different. We only take care of input nodes and don't worry about hidden nodes. Once the input is provided , RBM's automatically capture all the patterns , parameters and correlation among the data. That's the beauty of the Restricted Boltzmann Machine.

Some important features of Boltzmann Machine:

- ❖ They use recurrent and symmetric structure.
- ❖ RBMs in their learning process try to associate high probability with low energy states and vice-versa.
- ❖ There are no intra layer connections.
- ❖ It is an unsupervised learning algorithm i.e., it makes inferences from input data without labeled responses.

RNN- Recurrent Neural Networks:

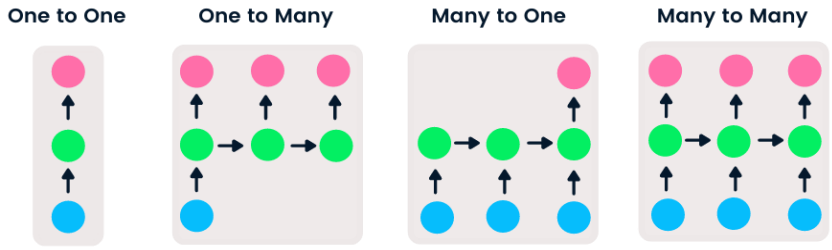
Recurrent Neural Networks imitate the function of the human brain in the fields of Data science, Artificial intelligence, machine learning, and deep learning, allowing computer programs to recognize patterns and solve common issues.

RNNs are a type of neural network that can be used to model sequence data. RNNs, which are formed from feedforward networks, are similar to human brains in their behaviour.

There are four types of RNN based on different lengths of inputs and outputs.

- ❖ **One-to-one** is a simple neural network. It is commonly used for machine learning problems that have a single input and output.
- ❖ **One-to-many** has a single input and multiple outputs. This is used for generating image captions.
- ❖ **Many-to-one** takes a sequence of multiple inputs and predicts a single output. It is popular in sentiment classification, where the input is text and the output is a category.

- **Many-to-many** takes multiple inputs and outputs. The most common application is machine translation.



Auto Encoders & Types:

Autoencoders are a special type of unsupervised feedforward neural network (no labels needed!). The main application of Autoencoders is to accurately capture the key aspects of the provided data to provide a compressed version of the input data, generate realistic synthetic data, or flag anomalies.

Autoencoders are composed of 2 key fully connected feedforward neural networks:

- **Encoder**: compresses the input data to remove any form of noise and generates a latent space/bottleneck. Therefore, the output neural network dimensions are smaller than the input and can be adjusted as a hyperparameter in order to decide how much lossy our compression should be.

- **Decoder**: making use of only the compressed data representation from the latent space, tries to reconstruct with as much fidelity as possible the original input data (the architecture of this neural network is, therefore, generally a mirror image of the encoder). The "goodness" of the prediction can then be measured by calculating the reconstruction error between the input and output data using a loss function.

Types of Autoencoders:

- Undercomplete Autoencoder

- Sparse Autoencoder

- Contractive Autoencoder

- Denoising Autoencoder
- Convolutional Autoencoder
- Variational Autoencoder

VGG NET:

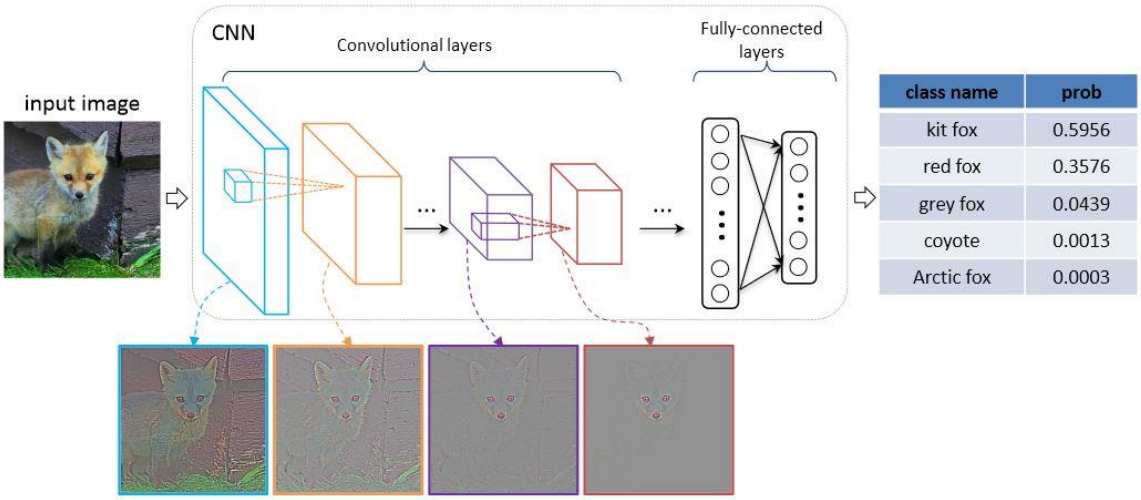
VGG (Visual Geometry Group) it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. The "deep" refers to the number of layers with VGG-16 or VGG-19 consisting of 16 and 19 convolutional layers. The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGG Net also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures.

VGG Architecture:

VGG Nets are based on the most essential features of convolutional neural networks (CNN). The following graphic shows the basic concept of how a CNN works:

Google Net & Architecture:

It is used in deep learning model which is developed by researchers of google and it consist of 22-layers and trained on the image net dataset. it can classify objects into 1000 different categories.



Data Types in Python:

Data types that you can use out of the box because they're built into the language. From all the built-in types available, you'll find that a few of them represent *basic* objects, such as numbers, strings and characters, bytes, and Boolean values.

- INT: -It consists the integer values i.e. numbers. 2, 3, ….
- FLOAT: - It consists of floating values i.e. decimals 0.78, 0,66, …
- COMPLEX: - It consists of imaginary and real numbers. 2+3i, ….
- CHAR: - It consists of characters i.e. alphabets. A, C, e ….
- STR: -It consists of string values i.e. group of characters. 'Python'
- BYTE: - Consists of 0 and 1. • BOOL: -Consists of Boolean type i.e. TRUE and FALSE.
- SET: -Consists the data items in a curly-braces. ,1, 5-
- TUPPLE: - Consists the data items in parentheses. (78, 98)
- DICT: -Consists the data items in curly-braces. ,1, 6, 9, 8-

- ❖ FROZEN SET: - Consists of set Operations. union, intersection
- ❖ RANGE: - Consists of range values. for i in range
- ❖ LIST: -Consists the data items in square brackets. *3, 6, 8+

Arithmetic Operations in Python:

The Python operators are fundamental for performing mathematical calculations in programming languages like Python, Java, C++, and many others. Arithmetic operators are symbols used to perform mathematical operations on numerical values. In most programming languages, arithmetic operators include addition (+), subtraction (-), multiplication (*), division (/), and modulus (%).

Arithmetic Operators in Python:

There are 7 arithmetic operators in Python. The lists are given below:

Operator	Description	Syntax	
Addition Operator	+	Addition: adds two operands	x + y
Subtraction Operator	–	Subtraction: subtracts two operands	x – y
Multiplication Operator	*	Multiplication: multiplies two operands	x * y
Division Operator	/	Division (float): divides the first operand by the second	x / y

Operator	Description	Syntax	
Floor Division Operator	//	Division (floor): divides the first operand by the second	x // y
Modulus Operator	%	Modulus: returns the remainder when the first operand is divided by the second	x % y
Exponentiation Operator	**	Power (Exponent): Returns first raised to power second	x ** y

Declaration of Comments & Variables:

Declaration of variables means while declaring variables, we have to keep some rules in mind. They are: -

- It should not start with digit or symbols. We can use underscore at first then use digits or symbols.

- Some of the examples are as follows: - _python123, n=50 etc. Comments in python: - Comments are used for the description of the code.

- There are two types of comment lines.

  They are: -
  1. Single line comment
  2. Multiple line comment.

- ❖ Single line comments are declared with the symbol (#) in-front of them.
- ❖ Multiple line comments are declared with triple quotations (‘ ’ ’).
- ❖ Examples are as follows: - # Python is a high-level language.

Reserved Words in Python:

There are 35 reserved words in python. They are: -

Boolean constraints:

- True
- False
- none

Logical operators:

- and
- or
- not
- is

Conditional statements:

- if
- elif
- else

Looping statements:

- while
- for
- break
- continue
- return
- in
- yield

Exception handling:

- try
- expect

- finally
- raise
- assert

Module and import:

- 31
- import
- from
- as

Function and class definition:

- class
- def
- pass
- global
- nonlocal
- lambda
- del

Context managers:

- with
- async
- await

Control Statements in Python:

Control statements are designed to serve the purpose of modifying a loop's execution from its default behaviour. Based on a condition, control statements are applied to alter how the loop executes. In this tutorial, we are covering every type of control statement that exists in Python.

The three types of control statements are: -

1. Conditional statements
2. Jumping statements
3. Looping statements

Conditional statements:

- **If Statement**: If the simple code of block is to be performed if the condition holds then the if statement is used. Here the condition mentioned holds then the code of the block runs otherwise not.

Syntax of If Statement:

```
if condition:
        statement
```

- **If-Else:** In a conditional if Statement the additional block of code is merged as an else statement which is performed when if condition is false.

Syntax of If-Else:

```
if condition:
        statement
else:
        statement
```

- **If-Elif-Else:** The if statements are executed from the top down. As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the ladder is bypassed. If none of the conditions is true, then the final "else" statement will be executed.

Syntax of If-Elif-Else:

```
if
    condition1:
        statement
elif
    condition2:
        statement
else:
        statement
```

- **Nested If:** Nested if-else means an if-else statement inside another if statement. Or in simple words first, there is an outer if statement, and inside it another if-else statement is present and such type of

statement is known as nested if statement. We can use one if or else if statement inside another if or else if statements.

Syntax of Nested If:

```
if
   condition1:
if
   condition2:
         statement
```

Jumping statements:

- **Break statement:** The break statement terminates the loop it is currently in, regardless of whether the loop condition is true or false.

EXAMPLE:

```
for i in range (10):
    If i==5:
Break
    Print(i)
```

- **Continue statement:** The continue statement skips the rest of the code inside the loop for the current iteration and proceeds to the next iteration of the loop.

EXAMPLE:

```
for i in range (10):
    If i==5:
Continue
    Print(i)
```

- Pass statement: The pass statement is a null operation; nothing happens when it is executed.

EXAMPLE:

```
for i in range (5):
    If i==3:
Pass
    else:
```

```
print(i)
```

Looping statements:

- **While loop:** A while loop is used to execute a block of statements repeatedly until a given condition is satisfied.

Syntax of While Loop:

```
while expression:
    statement
```

- **For loop:** It can be used to iterate over a range and iterators.

Syntax of for Loop:

```
for iterator_var in range:
    statement
```

- **Nested loop:** Python programming language allows to use one loop inside another loop which is called nested loop.

Syntax of Nested for Loop

```
for iterator_var in sequence:
    for iterator_var in sequence:
statements(s)
statements(s)
```

Programs:

Program to check if a number is prime or not

```
num = 29

# To take input from the user
```

```
#num = int(input("Enter a number: "))

# define a flag variable
flag = False

if num == 1:
   print(num, "is not a prime number")
elif num > 1:
   # check for factors
   for i in range(2, num):
      if (num % i) == 0:
         # if factor is found, set flag to True
         flag = True
         # break out of loop
         break

   # check if flag is True
   if flag:
      print(num, "is not a prime number")
   else:
      print(num, "is a prime number")
```

Python Program to find the factors of a number

```
# This function computes the factor of the argument passed
def print_factors(x):
  print("The factors of",x,"are:")
```

```
    for i in range(1, x + 1):
        if x % i == 0:
            print(i)

num = 320

print_factors(num)
```

#Python Program to reverse a number

```
        num = 1234
reversed_num = 0

    while num != 0:
        digit = num % 10
        reversed_num = reversed_num * 10 + digit
        num //= 10

print("Reversed Number: " + str(reversed_num))
```

#Python Program to remove duplicate number

```
            list_1 = [1, 2, 1, 4, 6]
            list_2 = [7, 8, 2, 1]

        print(list(set(list_1) ^ set(list_2)))
```

Problem statement & Explanation

Air Pollution

Recently, much has been discussed about air pollution and its consequences on the environment. These discussions always gain prominence when some of their consequences haunt the world and leave us wondering what will be of future generations.

Maybe greenhouse gases and the consequent global warming are the most well-known names for most of the population. However, other air pollutants cause many harmful effects to the population and end up not being well regulated in many regions. These are the cases of particulate matter and nitrogen oxides, for example. In large cities, these pollutants are usually emitted on a large scale by diesel vehicles that lack proper maintenance.

For a more assertive assessment of the state of pollutant concentrations in a given region, an in-depth study of the data is necessary. For that, a couple of years ago I started using the OpenAir package of R, which by the way is excellent. However, over the last few months, I've started using Python and I've come to like it. But what I miss the most about R is OpenAir.

With that in mind, I decided to develop two functions in Python, for visualizing pollutants datasets, inspired by OpenAir of R.

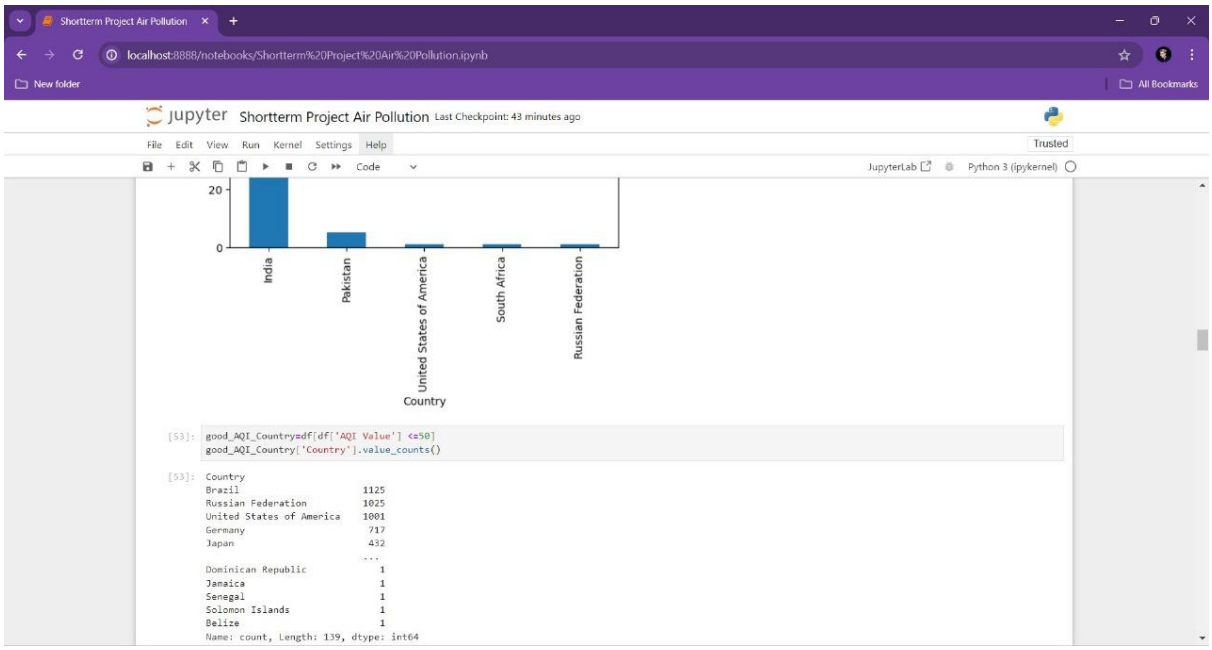
So, in this article, I'll present these functions and I'll detail how they were developed.

It's important to make it clear that I'm not going to deal with the analysis of local pollutants here, but rather use the data as an example to illustrate our analysis tools. By the way, I used a dataset from: https://qualar.cetesb.sp.gov.br/qualar/home.do, which is the agency responsible for regulating environmental issues in the state of Sao Paulo.

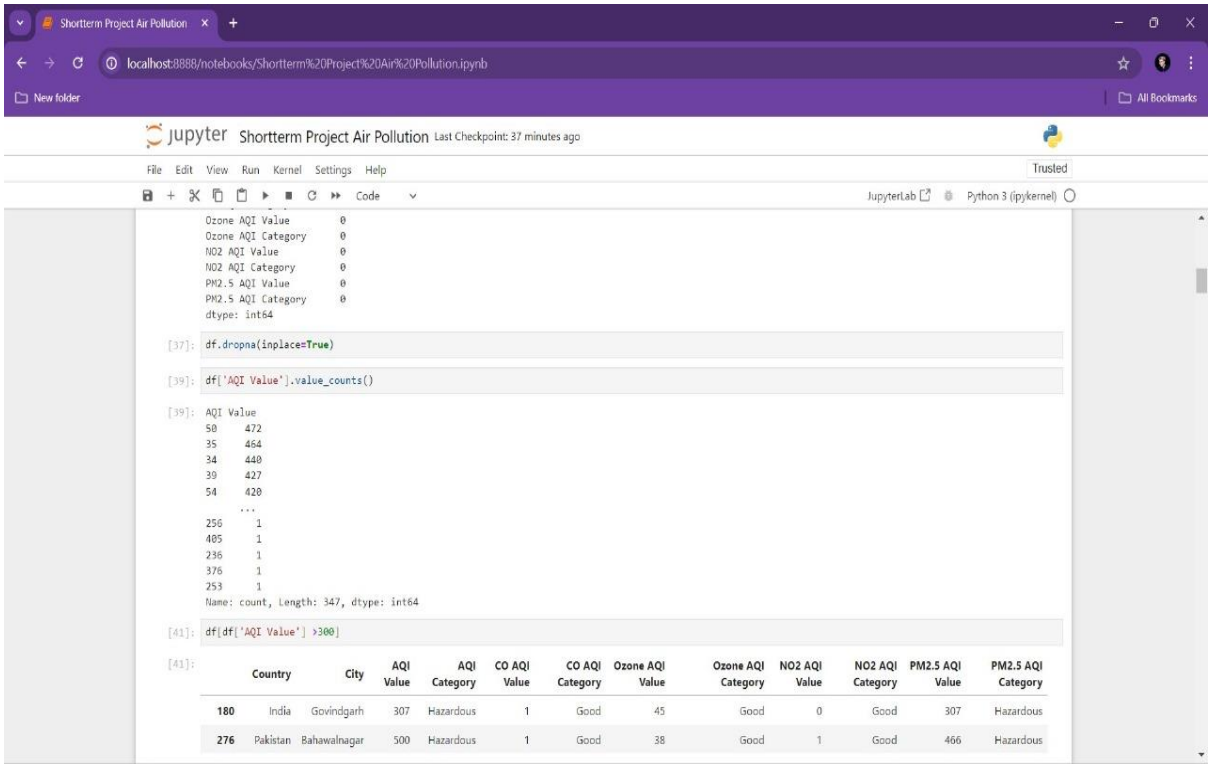
Air pollution is contamination of the indoor or outdoor environment by any chemical, physical or biological agent that modifies the natural characteristics of the atmosphere.

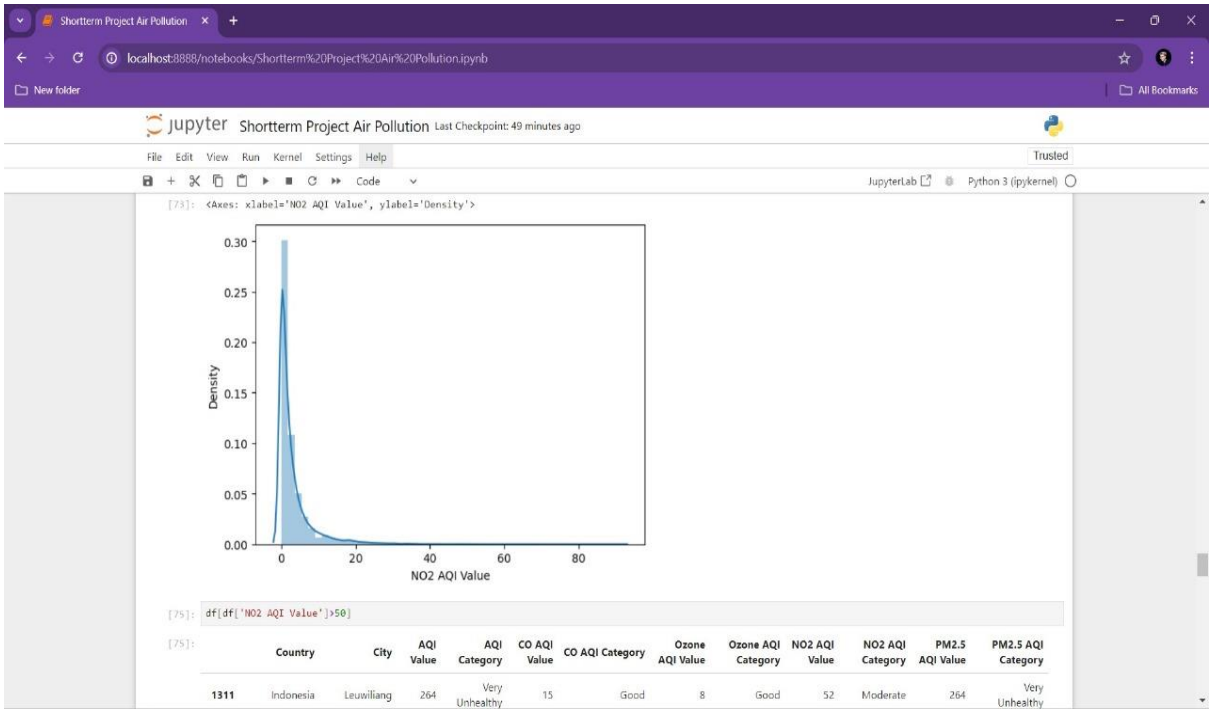
Household combustion devices, motor vehicles, industrial facilities and forest fires are common sources of air pollution. Pollutants of major public health concern include particulate matter, carbon monoxide, ozone, nitrogen dioxide and sulphur dioxide. Outdoor and indoor air pollution cause respiratory and other diseases and are important sources of morbidity and mortality.

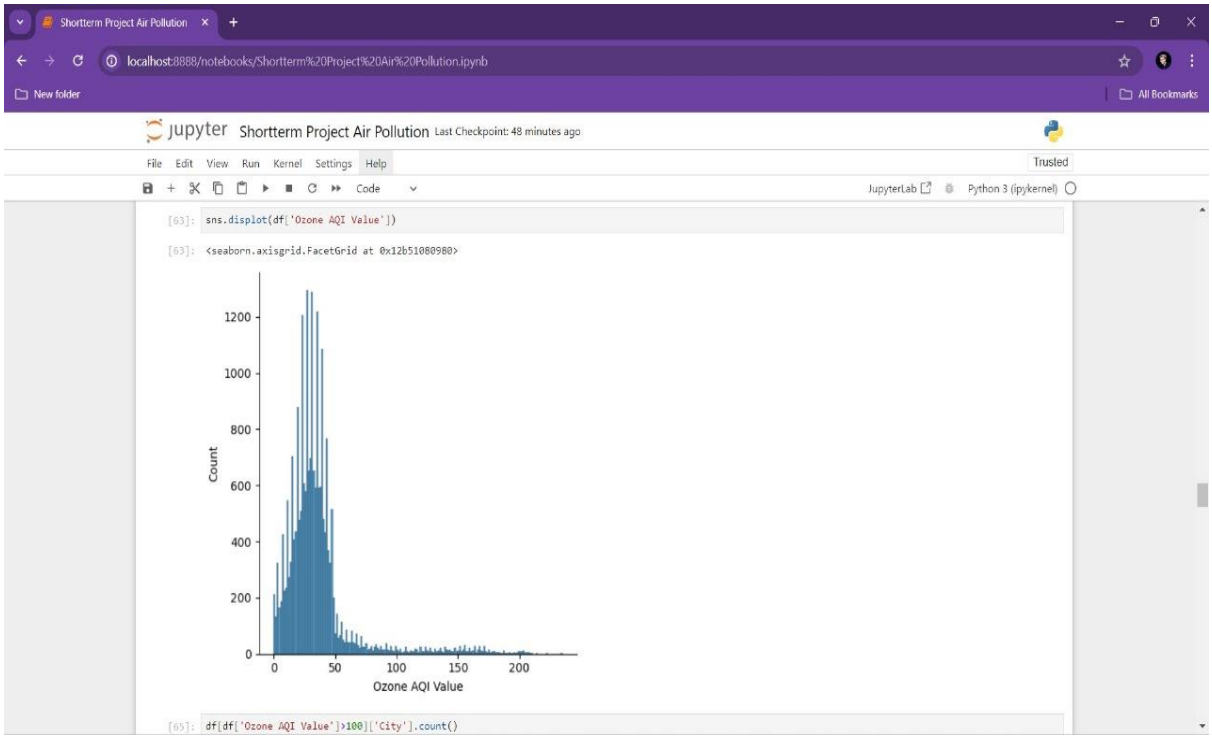
SOURCE CODE

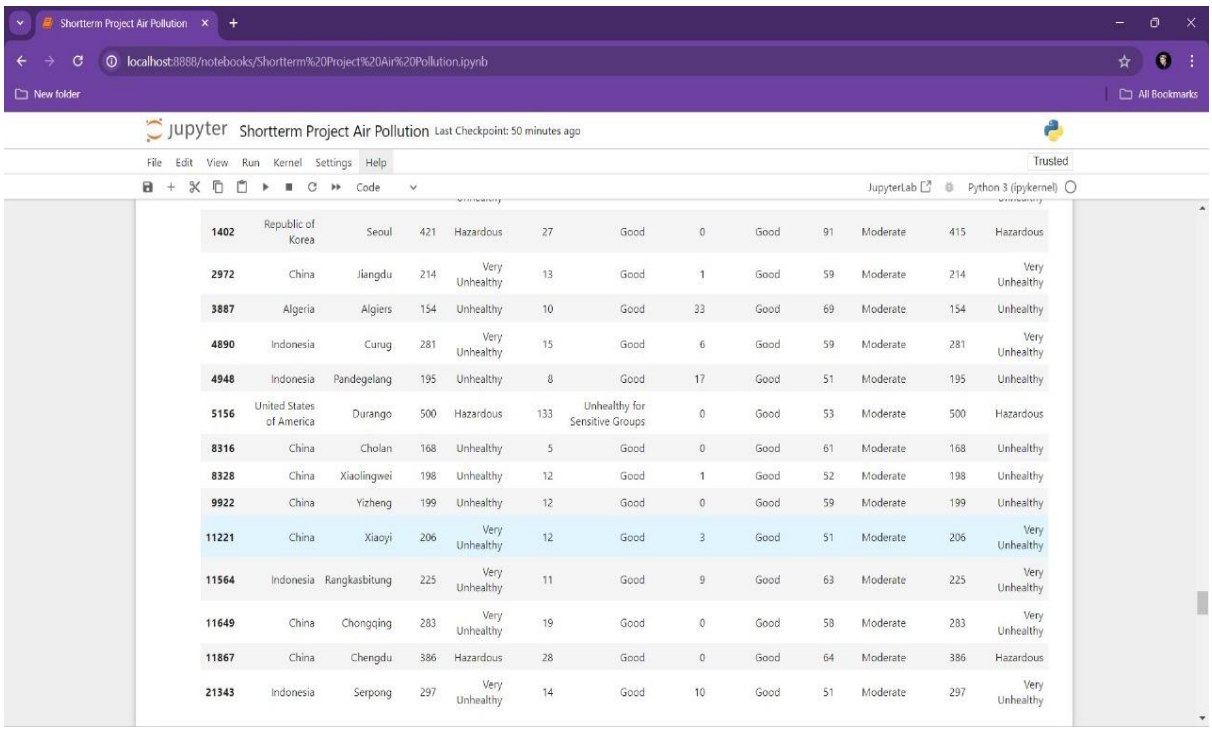

Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
jupyter Shortterm Project Air Pollution Last Checkpoint: 43 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
20
0
India
Pakistan
United States of America
South Africa
Russian Federation
Country
[53]: good_AQI_Country=df[df['AQI Value'] <=50]
good_AQI_Country['Country'].value_counts()
[53]: Country
Brazil 1125
Russian Federation 1025
United States of America 1001
Germany 717
Japan 432
Dominican Republic 1
Jamaica 1
Senegal 1
Solomon Islands 1
Belize 1
Name: count, Length: 139, dtype: int64


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
jupyter Shortterm Project Air Pollution Last Checkpoint: 52 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
NO2 AQI Value vs AQI Value
PM2.5 AQI Value vs AQI Value
AQI Value
NO2 AQI Value
PM2.5 AQI Value
[85]: df.corr(numeric_only=True)
[85]:
AQI Value CO AQI Value Ozone AQI Value NO2 AQI Value PM2.5 AQI Value
AQI Value 1.000000 0.429643 0.405086 0.230845 0.984518
CO AQI Value 0.429643 1.000000 0.144838 0.487627 0.437751


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 37 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Ozone AQI Value 0
Ozone AQI Category 0
NO2 AQI Value 0
NO2 AQI Category 0
PM2.5 AQI Value 0
PM2.5 AQI Category 0
dtype: int64
[37]: df.dropna(inplace=True)
[39]: df['AQI Value'].value_counts()
[39]: AQI Value
50 472
35 464
34 440
39 427
54 420
...
256 1
405 1
236 1
376 1
253 1
Name: count, Length: 347, dtype: int64
[41]: df[df['AQI Value'] >300]
[41]:
Country City AQI Value AQI Category CO AQI Value CO AQI Category Ozone AQI Value Ozone AQI Category NO2 AQI Value NO2 AQI Category PM2.5 AQI Value PM2.5 AQI Category
180 India Govindgarh 307 Hazardous 1 Good 45 Good 0 Good 307 Hazardous
276 Pakistan Bahawalnagar 500 Hazardous 1 Good 38 Good 1 Good 466 Hazardous


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 49 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[73]: <Axes: xlabel='NO2 AQI Value', ylabel='Density'>
Density
NO2 AQI Value
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
[75]: df[df['NO2 AQI Value']>50]
[75]:
Country City AQI Value AQI Category CO AQI Value CO AQI Category Ozone AQI Value Ozone AQI Category NO2 AQI Value NO2 AQI Category PM2.5 AQI Value PM2.5 AQI Category
1311 Indonesia Leuwiliang 264 Very Unhealthy 15 Good 8 Good 52 Moderate 264 Very Unhealthy





1402	Republic of Korea	Seoul	421	Hazardous	27	Good	0	Good	91	Moderate	415	Hazardous
2972	China	Jiangdu	214	Very Unhealthy	13	Good	1	Good	59	Moderate	214	Very Unhealthy
3887	Algeria	Algiers	154	Unhealthy	10	Good	33	Good	69	Moderate	154	Unhealthy
4890	Indonesia	Curug	281	Very Unhealthy	15	Good	6	Good	59	Moderate	281	Very Unhealthy
4948	Indonesia	Pandegelang	195	Unhealthy	8	Good	17	Good	51	Moderate	195	Unhealthy
5156	United States of America	Durango	500	Hazardous	133	Unhealthy for Sensitive Groups	0	Good	53	Moderate	500	Hazardous
8316	China	Cholan	168	Unhealthy	5	Good	0	Good	61	Moderate	168	Unhealthy
8328	China	Xiaolingwei	198	Unhealthy	12	Good	1	Good	52	Moderate	198	Unhealthy
9922	China	Yizheng	199	Unhealthy	12	Good	0	Good	59	Moderate	199	Unhealthy
11221	China	Xiaoyi	206	Very Unhealthy	12	Good	3	Good	51	Moderate	206	Very Unhealthy
11564	Indonesia	Rangkasbitung	225	Very Unhealthy	11	Good	9	Good	63	Moderate	225	Very Unhealthy
11649	China	Chongqing	283	Very Unhealthy	19	Good	0	Good	58	Moderate	283	Very Unhealthy
11867	China	Chengdu	386	Hazardous	28	Good	0	Good	64	Moderate	386	Hazardous
21343	Indonesia	Serpong	297	Very Unhealthy	14	Good	10	Good	51	Moderate	297	Very Unhealthy

Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

New folder | All Bookmarks

jupyter Shortterm Project Air Pollution Last Checkpoint: 48 minutes ago

File Edit View Run Kernel Settings Help | Trusted

Code | JupyterLab | Python 3 (ipykernel)

```
[65]: df[df['Ozone AQI Value']>100]['City'].count()
```

```
[65]: 930
```

```
[67]: df[df['Ozone AQI Value']>200].head(3)
```

[67]:

	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
77	China	Hangzhou	203	Very Unhealthy	5	Good	203	Very Unhealthy	5	Good	151	Unhealthy
613	China	Haian	207	Very Unhealthy	4	Good	207	Very Unhealthy	3	Good	150	Unhealthy
722	China	Kangshan	201	Very Unhealthy	3	Good	201	Very Unhealthy	4	Good	142	Unhealthy for Sensitive Groups

```
[69]: df[df['Ozone AQI Value']>200]['City'].count()
```

```
[69]: 52
```

```
[71]: df['NO2 AQI Value'].value_counts()
```

```
[71]: NO2 AQI Value
0     8033
1     4595
2     2668
3     1866
4     1248
5      871
6      655
7      475
```

Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

New folder | All Bookmarks

jupyter Shortterm Project Air Pollution Last Checkpoint: 43 minutes ago

File Edit View Run Kernel Settings Help | Trusted

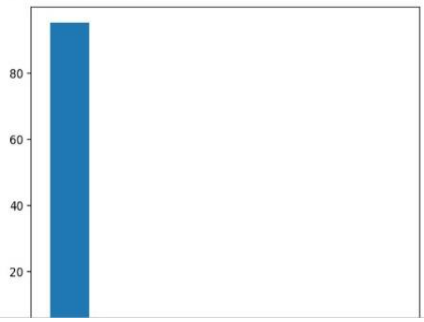
Code | JupyterLab | Python 3 (ipykernel)

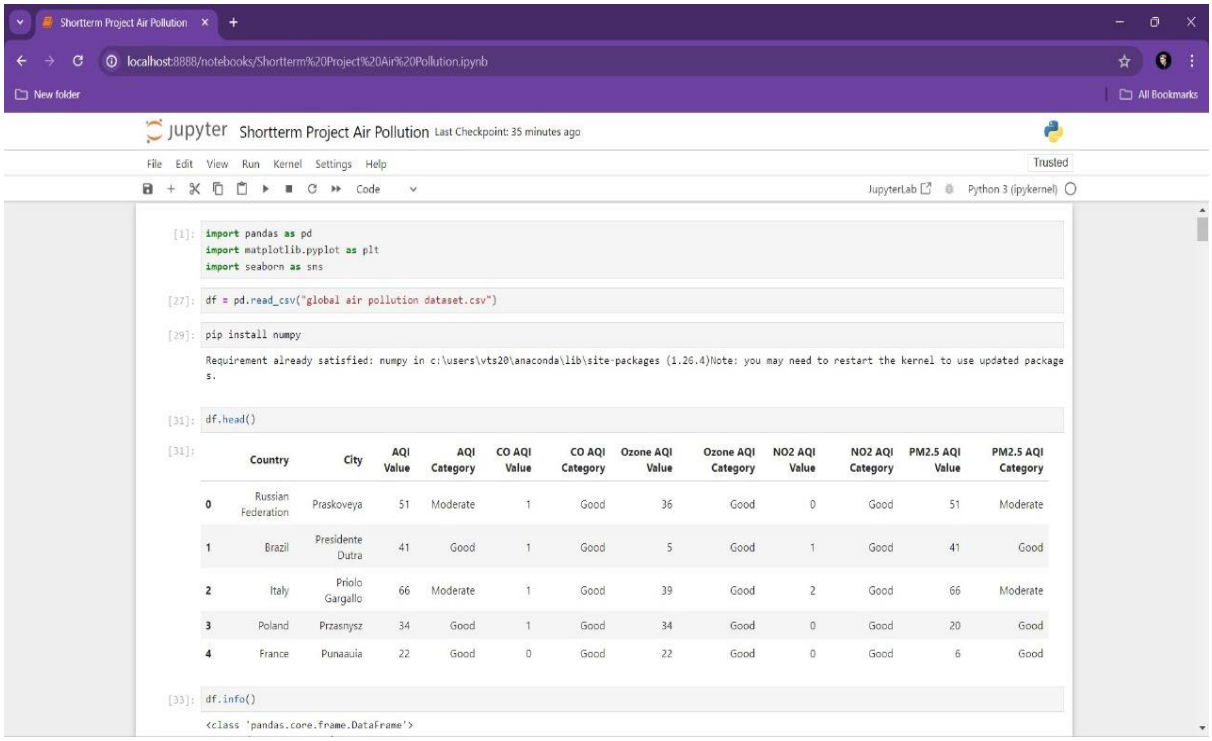
```
[49]: maxm['Country'].value_counts()
```

```
[49]: Country
India                       95
Pakistan                     5
United States of America     1
South Africa                 1
Russian Federation           1
Name: count, dtype: int64
```

```
[51]: maxm['Country'].value_counts().plot.bar()
```

```
[51]: <Axes: xlabel='Country'>
```





Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

jupyter Shortterm Project Air Pollution Last Checkpoint: 35 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

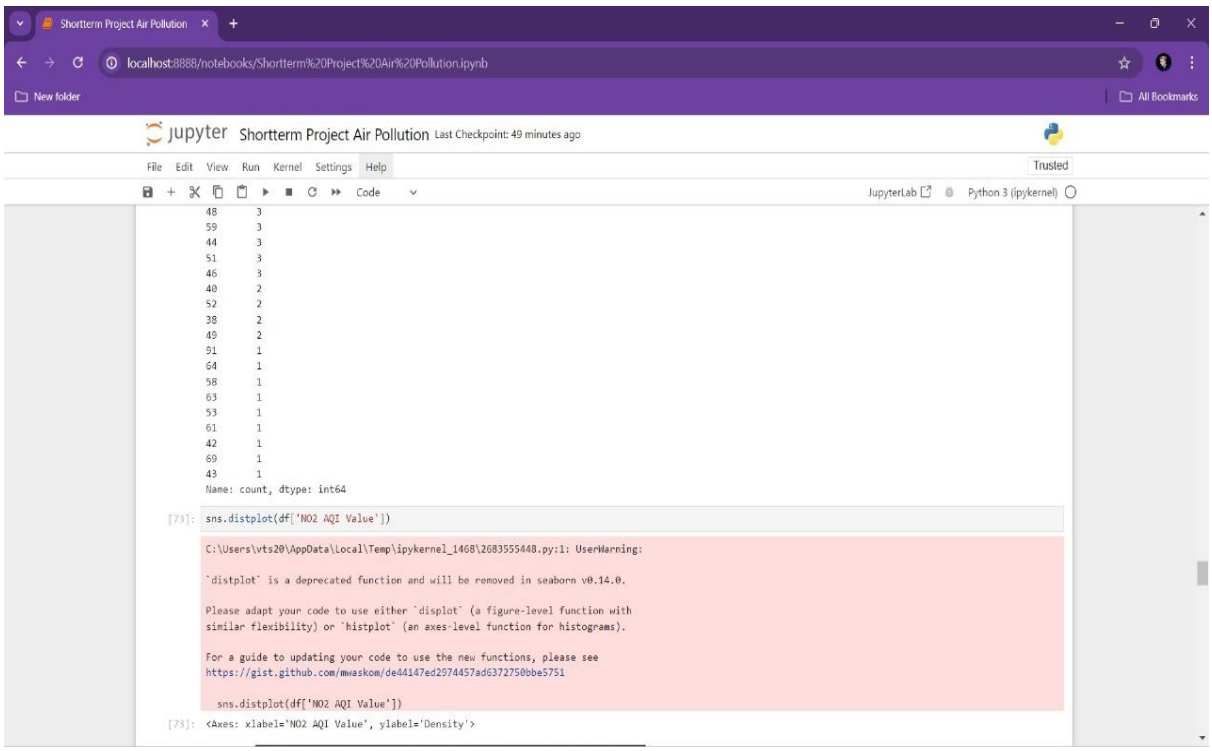
JupyterLab Python 3 (ipykernel)

```
[1]: import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns

[27]: df = pd.read_csv("global air pollution dataset.csv")

[29]: pip install numpy

Requirement already satisfied: numpy in c:\users\vts20\anaconda\lib\site-packages (1.26.4)Note: you may need to restart the kernel to use updated package
s.

[31]: df.head()
```

[31]:

	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
0	Russian Federation	Praskoveya	51	Moderate	1	Good	36	Good	0	Good	51	Moderate
1	Brazil	Presidente Dutra	41	Good	1	Good	5	Good	1	Good	41	Good
2	Italy	Priolo Gargallo	66	Moderate	1	Good	39	Good	2	Good	66	Moderate
3	Poland	Przasnysz	34	Good	1	Good	34	Good	0	Good	20	Good
4	France	Punaauia	22	Good	0	Good	22	Good	0	Good	6	Good

```
[33]: df.info()

<class 'pandas.core.frame.DataFrame'>
```



Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

jupyter Shortterm Project Air Pollution Last Checkpoint: 49 minutes ago

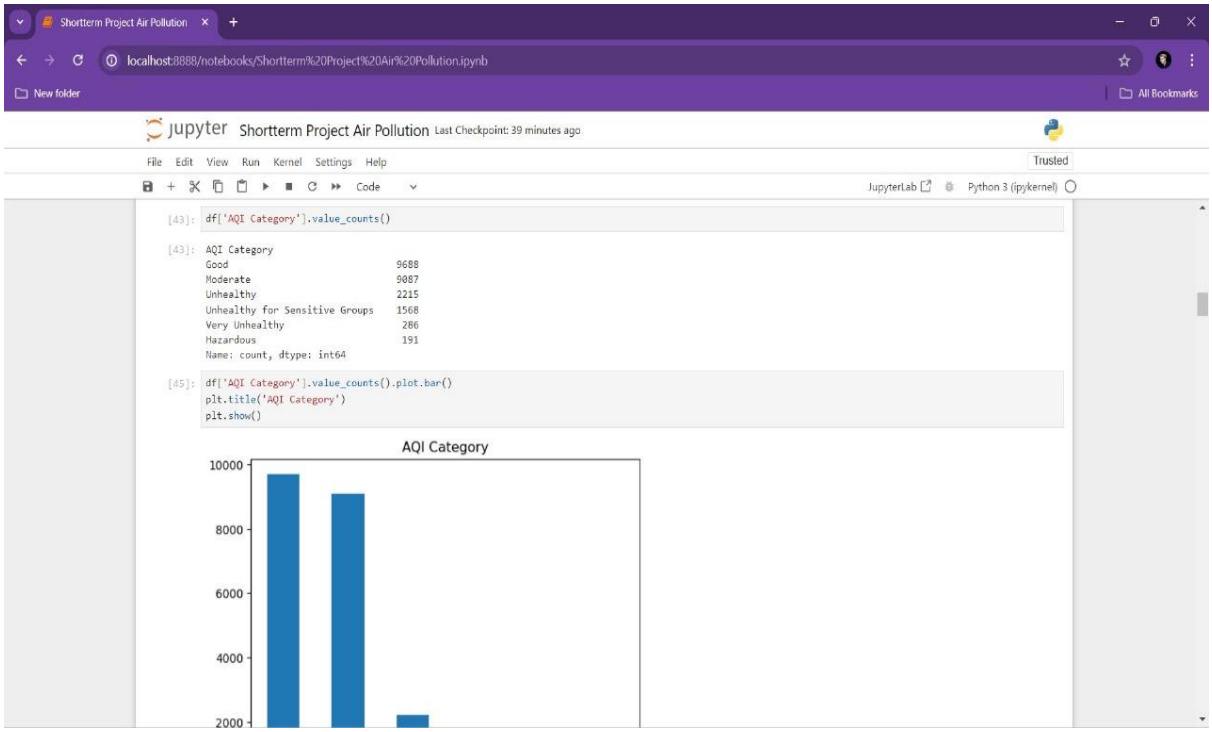
File Edit View Run Kernel Settings Help

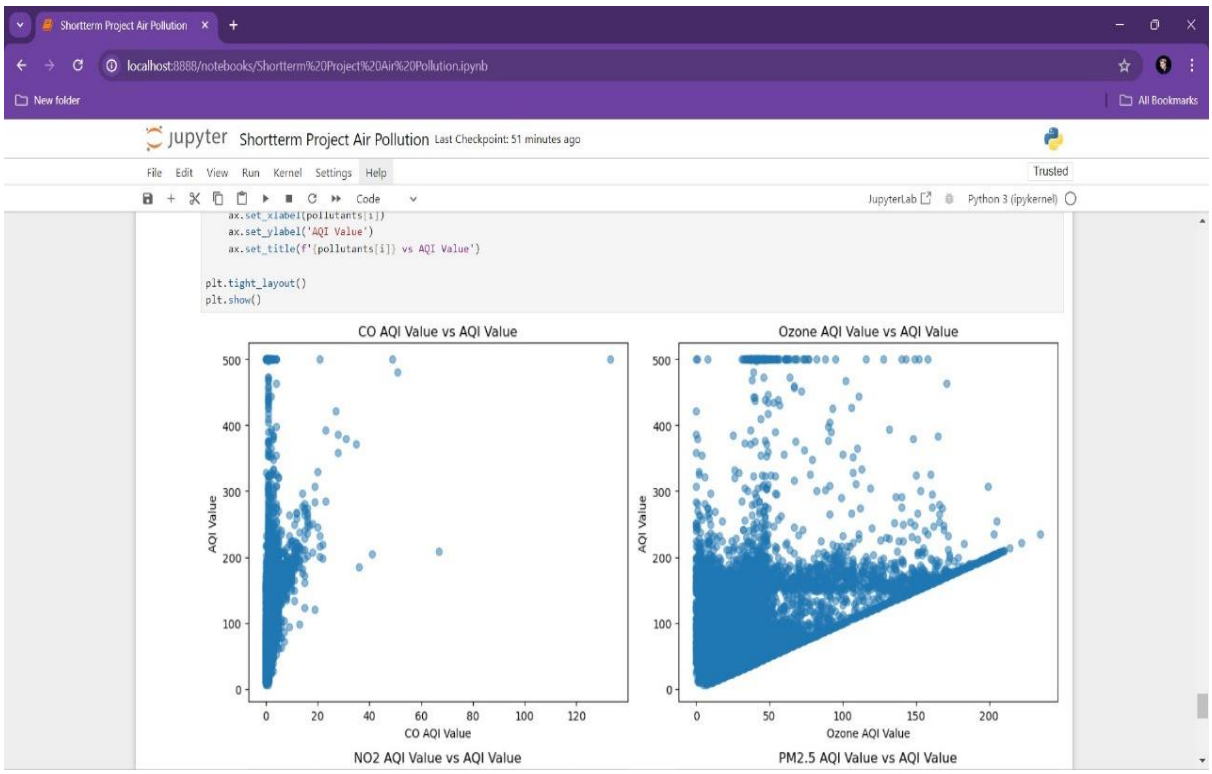
Trusted

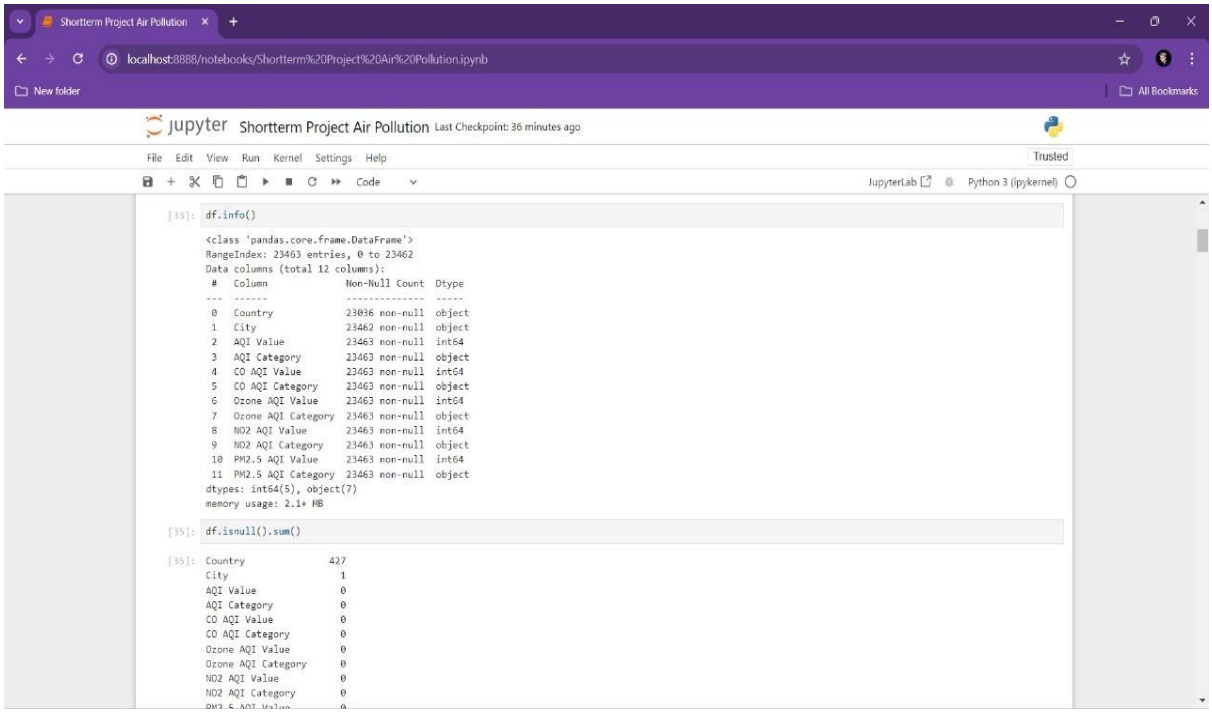
Code

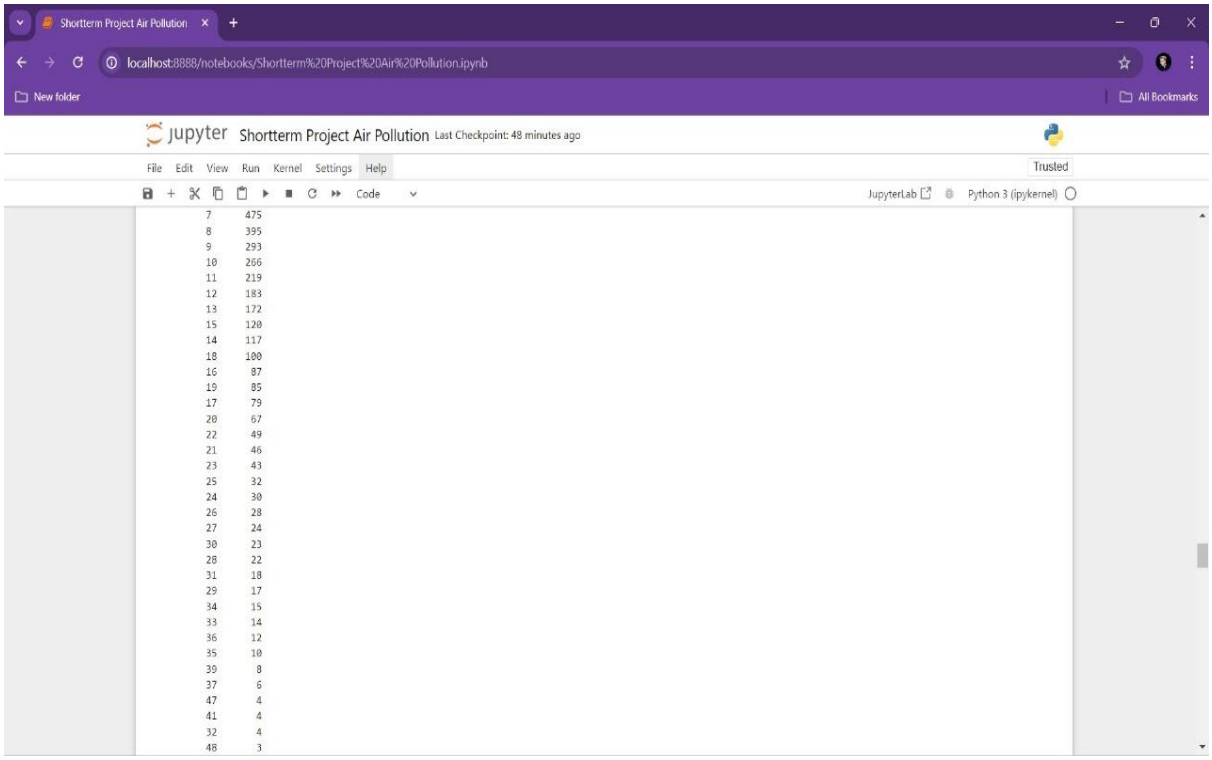
JupyterLab Python 3 (ipykernel)

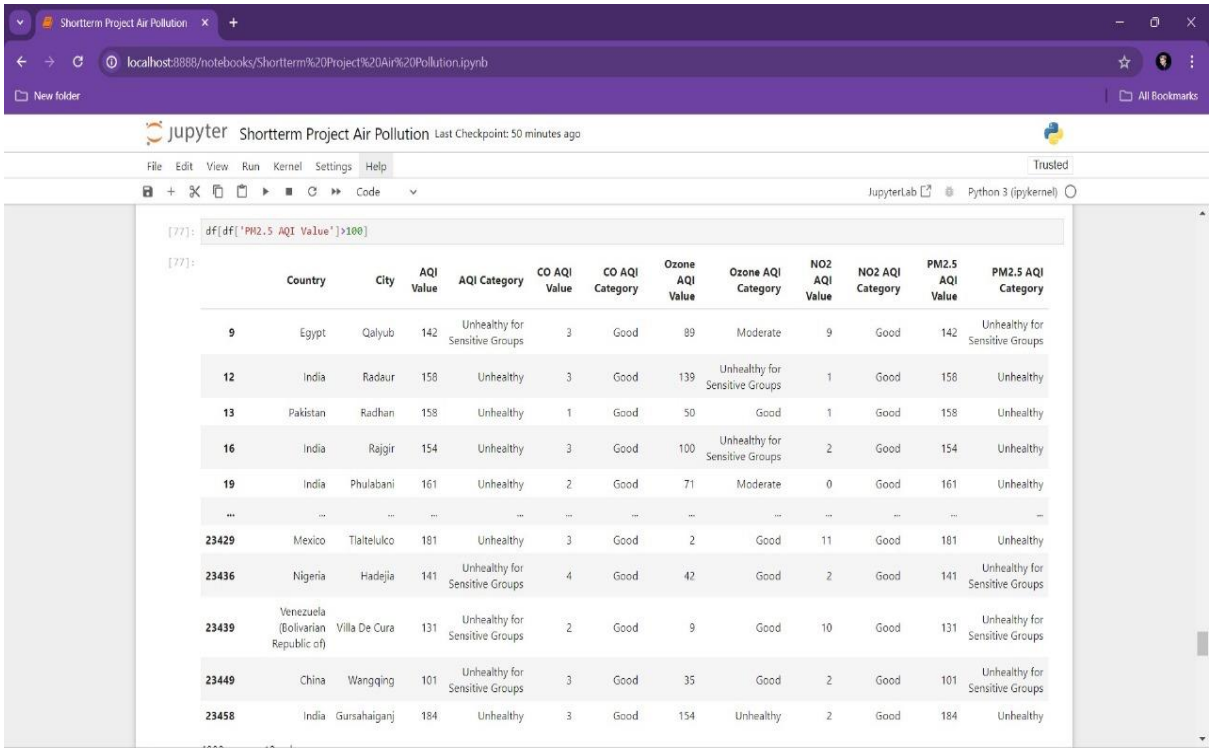
```
48    3
59    3
44    3
51    3
46    3
40    2
52    2
38    2
49    2
91    1
64    1
58    1
63    1
53    1
61    1
42    1
69    1
43    1
Name: count, dtype: int64

[73]: sns.distplot(df['NO2 AQI Value'])

C:\Users\vts20\AppData\Local\Temp\ipykernel_1468\2683555448.py:1: 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

  sns.distplot(df['NO2 AQI Value'])

[73]: <Axes: xlabel='NO2 AQI Value', ylabel='Density'>
```


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 39 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[43]: df['AQI Category'].value_counts()
[43]: AQI Category
Good 9688
Moderate 9087
Unhealthy 2215
Unhealthy for Sensitive Groups 1568
Very Unhealthy 286
Hazardous 191
Name: count, dtype: int64
[45]: df['AQI Category'].value_counts().plot.bar()
plt.title('AQI Category')
plt.show()
AQI Category
10000
8000
6000
4000
2000

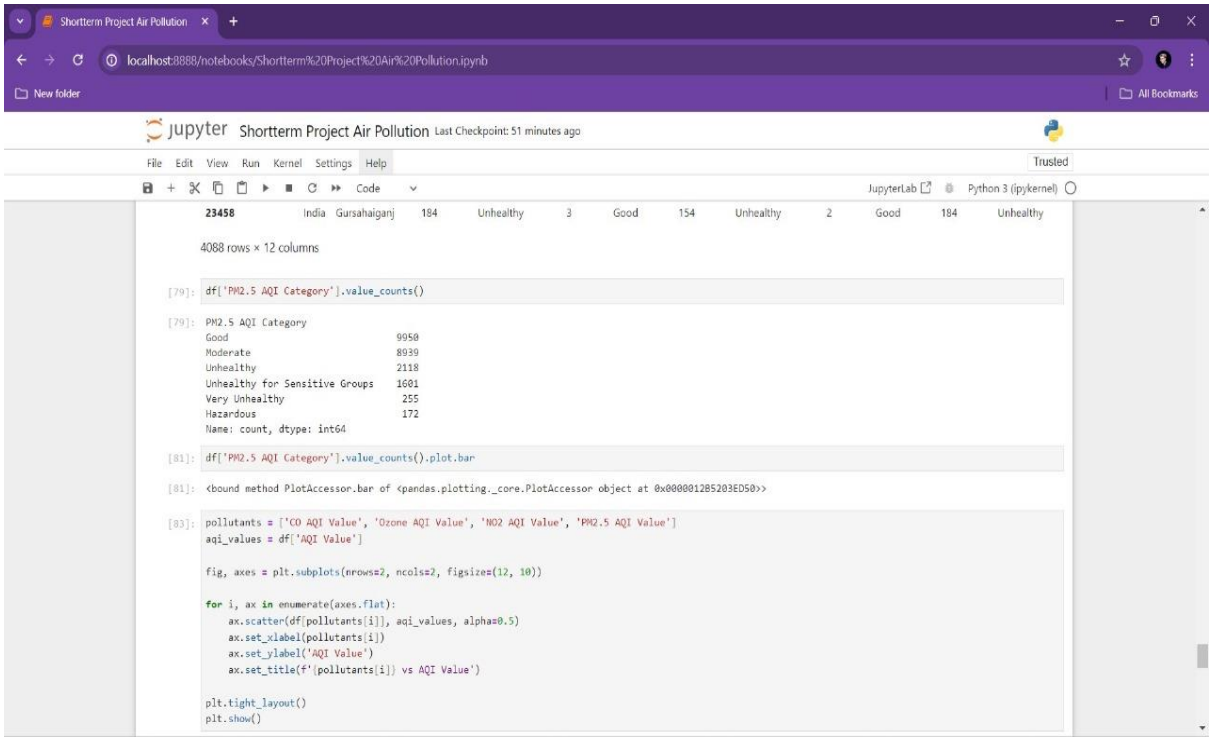

Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 51 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
ax.set_xlabel(pollutants[i])
ax.set_ylabel('AQI Value')
ax.set_title(f'{pollutants[i]} vs AQI Value')
plt.tight_layout()
plt.show()
CO AQI Value vs AQI Value
Ozone AQI Value vs AQI Value
AQI Value
CO AQI Value
Ozone AQI Value
NO2 AQI Value vs AQI Value
PM2.5 AQI Value vs AQI Value


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 36 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[33]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23463 entries, 0 to 23462
Data columns (total 12 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 Country 23036 non-null object
1 City 23462 non-null object
2 AQI Value 23463 non-null int64
3 AQI Category 23463 non-null object
4 CO AQI Value 23463 non-null int64
5 CO AQI Category 23463 non-null object
6 Ozone AQI Value 23463 non-null int64
7 Ozone AQI Category 23463 non-null object
8 NO2 AQI Value 23463 non-null int64
9 NO2 AQI Category 23463 non-null object
10 PM2.5 AQI Value 23463 non-null int64
11 PM2.5 AQI Category 23463 non-null object
dtypes: int64(5), object(7)
memory usage: 2.1+ MB
[35]: df.isnull().sum()
[35]: Country 427
City 1
AQI Value 0
AQI Category 0
CO AQI Value 0
CO AQI Category 0
Ozone AQI Value 0
Ozone AQI Category 0
NO2 AQI Value 0
NO2 AQI Category 0


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 48 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
7 475
8 395
9 293
10 266
11 219
12 183
13 172
15 120
14 117
18 100
16 87
19 85
17 79
20 67
22 49
21 46
23 43
25 32
24 30
26 28
27 24
30 23
28 22
31 18
29 17
34 15
33 14
36 12
35 10
39 8
37 6
47 4
41 4
32 4
48 3



	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
9	Egypt	Qalyub	142	Unhealthy for Sensitive Groups	3	Good	89	Moderate	9	Good	142	Unhealthy for Sensitive Groups
12	India	Radaur	158	Unhealthy	3	Good	139	Unhealthy for Sensitive Groups	1	Good	158	Unhealthy
13	Pakistan	Radhan	158	Unhealthy	1	Good	50	Good	1	Good	158	Unhealthy
16	India	Rajgir	154	Unhealthy	3	Good	100	Unhealthy for Sensitive Groups	2	Good	154	Unhealthy
19	India	Phulabani	161	Unhealthy	2	Good	71	Moderate	0	Good	161	Unhealthy
...	...	...	...	...	...	...	...	...	...	...	...	...
23429	Mexico	Tlaltelulco	181	Unhealthy	3	Good	2	Good	11	Good	181	Unhealthy
23436	Nigeria	Hadejia	141	Unhealthy for Sensitive Groups	4	Good	42	Good	2	Good	141	Unhealthy for Sensitive Groups
23439	Venezuela (Bolivarian Republic of)	Villa De Cura	131	Unhealthy for Sensitive Groups	2	Good	9	Good	10	Good	131	Unhealthy for Sensitive Groups
23449	China	Wangqing	101	Unhealthy for Sensitive Groups	3	Good	35	Good	2	Good	101	Unhealthy for Sensitive Groups
23458	India	Gursahaiganj	184	Unhealthy	3	Good	154	Unhealthy	2	Good	184	Unhealthy



23458	India	Gursahaiganj	184	Unhealthy	3	Good	154	Unhealthy	2	Good	184	Unhealthy

4088 rows × 12 columns

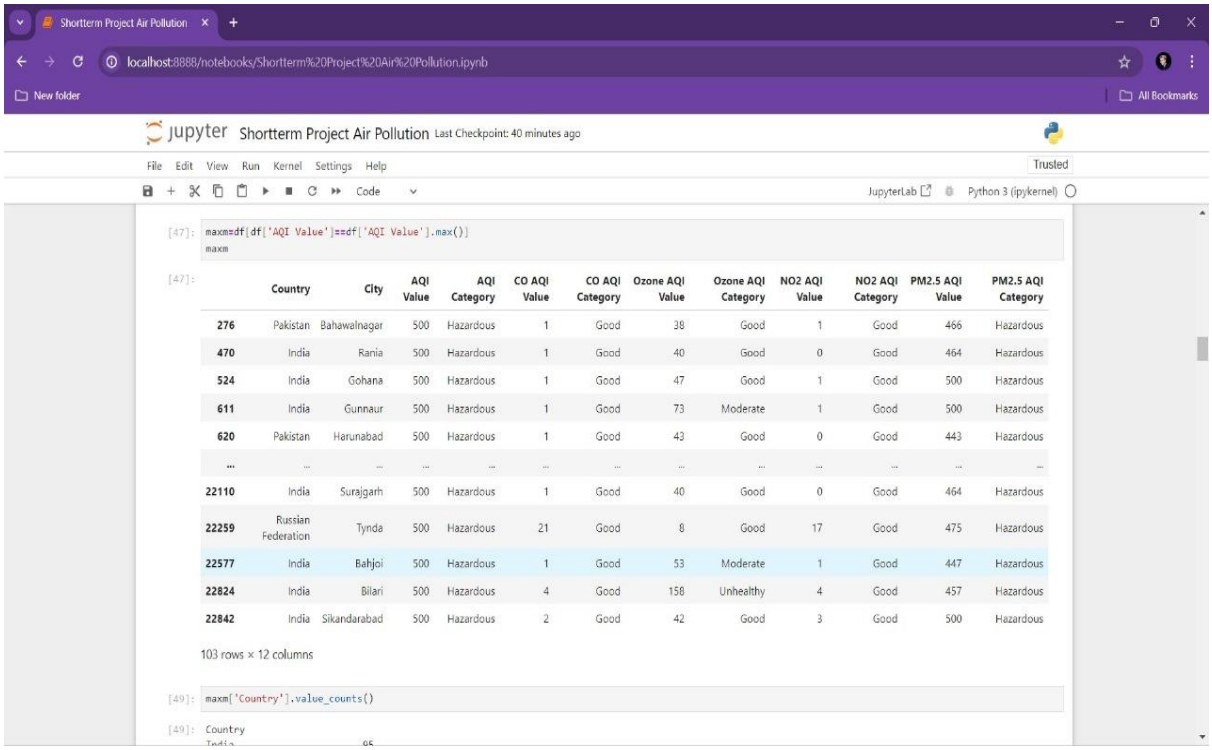
```
[79]: df['PM2.5 AQI Category'].value_counts()
```

```
[79]: PM2.5 AQI Category
Good                            9950
Moderate                        8939
Unhealthy                       2118
Unhealthy for Sensitive Groups  1601
Very Unhealthy                   255
Hazardous                        172
Name: count, dtype: int64
```

```
[81]: df['PM2.5 AQI Category'].value_counts().plot.bar
```

```
[81]: <bound method PlotAccessor.bar of <pandas.plotting._core.PlotAccessor object at 0x0000012B5203ED50>>
```

```
[83]: pollutants = ['CO AQI Value', 'Ozone AQI Value', 'NO2 AQI Value', 'PM2.5 AQI Value']
aqi_values = df['AQI Value']

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 10))

for i, ax in enumerate(axes.flat):
    ax.scatter(df[pollutants[i]], aqi_values, alpha=0.5)
    ax.set_xlabel(pollutants[i])
    ax.set_ylabel('AQI Value')
    ax.set_title(f'{pollutants[i]} vs AQI Value')

plt.tight_layout()
plt.show()
```




```
[47]: maxm=df[df['AQI Value']==df['AQI Value'].max()]
      maxm
```

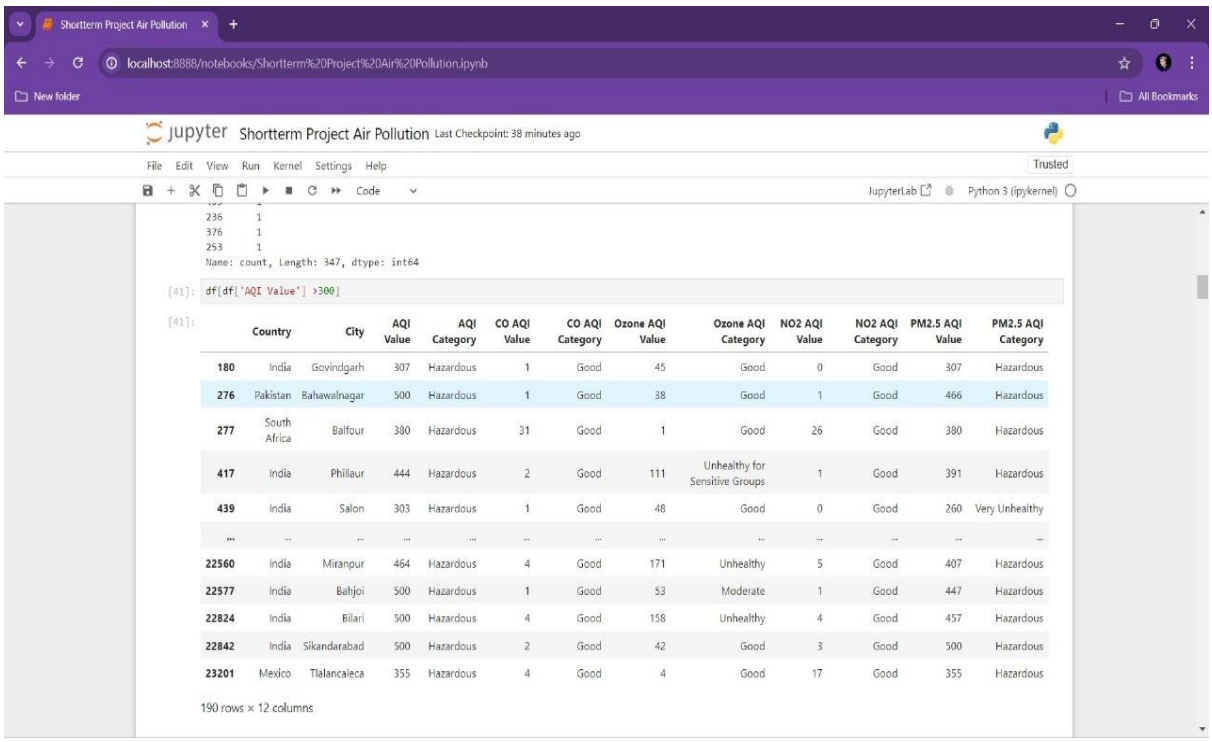
[47]:

	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
276	Pakistan	Bahawalnagar	500	Hazardous	1	Good	38	Good	1	Good	466	Hazardous
470	India	Rania	500	Hazardous	1	Good	40	Good	0	Good	464	Hazardous
524	India	Gohana	500	Hazardous	1	Good	47	Good	1	Good	500	Hazardous
611	India	Gunnaur	500	Hazardous	1	Good	73	Moderate	1	Good	500	Hazardous
620	Pakistan	Harunabad	500	Hazardous	1	Good	43	Good	0	Good	443	Hazardous
...	...	...	...	...	...	...	...	...	...	...	...	...
22110	India	Surajgarh	500	Hazardous	1	Good	40	Good	0	Good	464	Hazardous
22259	Russian Federation	Tynda	500	Hazardous	21	Good	8	Good	17	Good	475	Hazardous
22577	India	Bahjoi	500	Hazardous	1	Good	53	Moderate	1	Good	447	Hazardous
22824	India	Bilari	500	Hazardous	4	Good	158	Unhealthy	4	Good	457	Hazardous
22842	India	Sikandarabad	500	Hazardous	2	Good	42	Good	3	Good	500	Hazardous

103 rows × 12 columns

```
[49]: maxm['Country'].value_counts()
```

```
[49]: Country
      India                    95
```



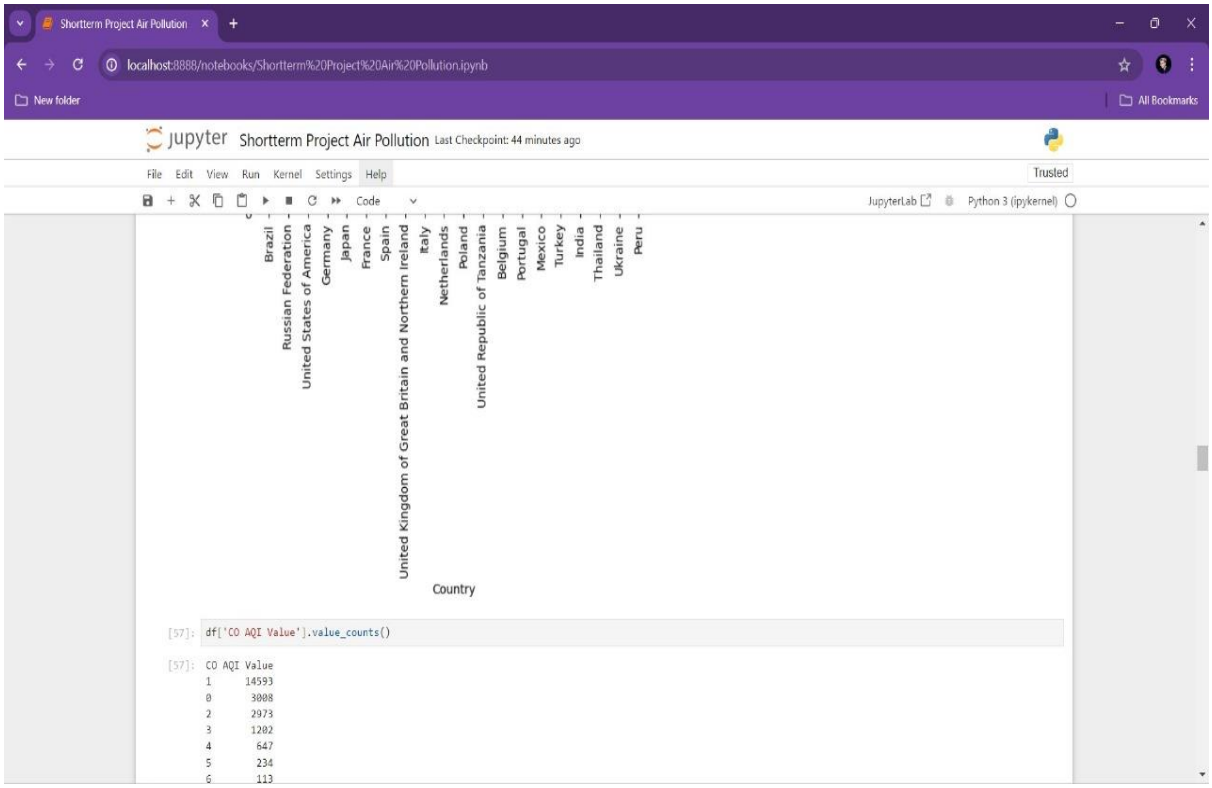

```
236    1
376    1
253    1
Name: count, Length: 347, dtype: int64
```

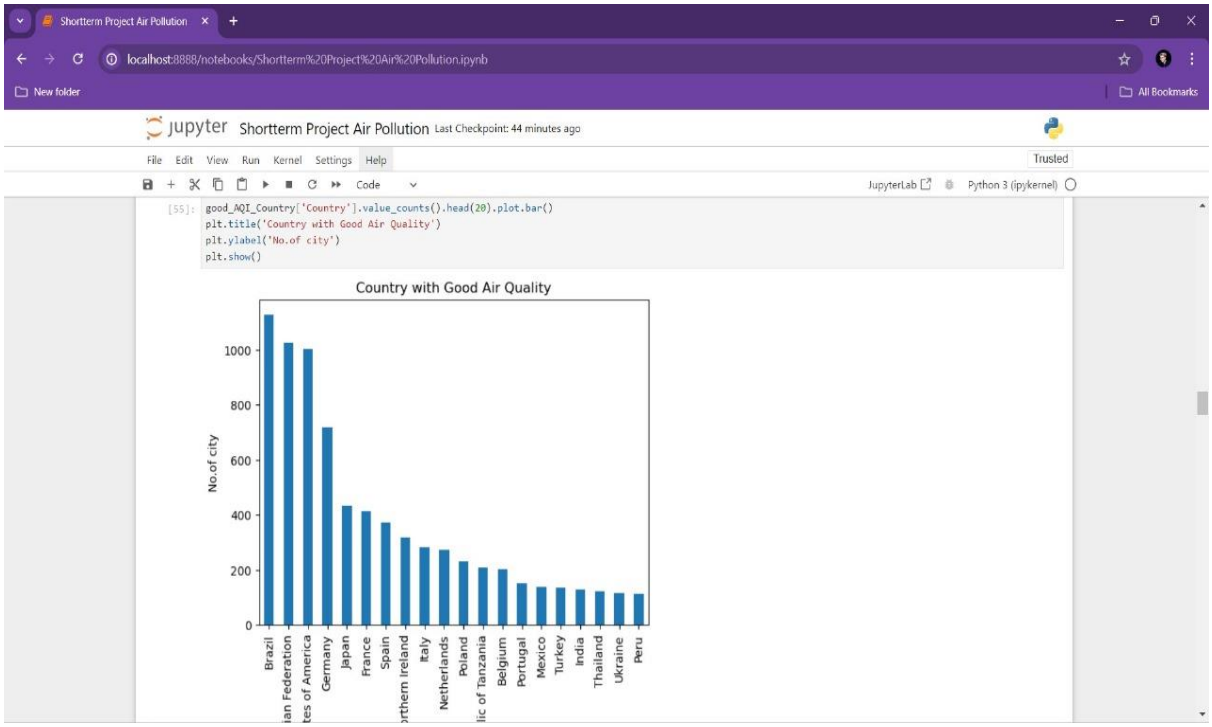
```
[41]: df[df['AQI Value'] >300]
```

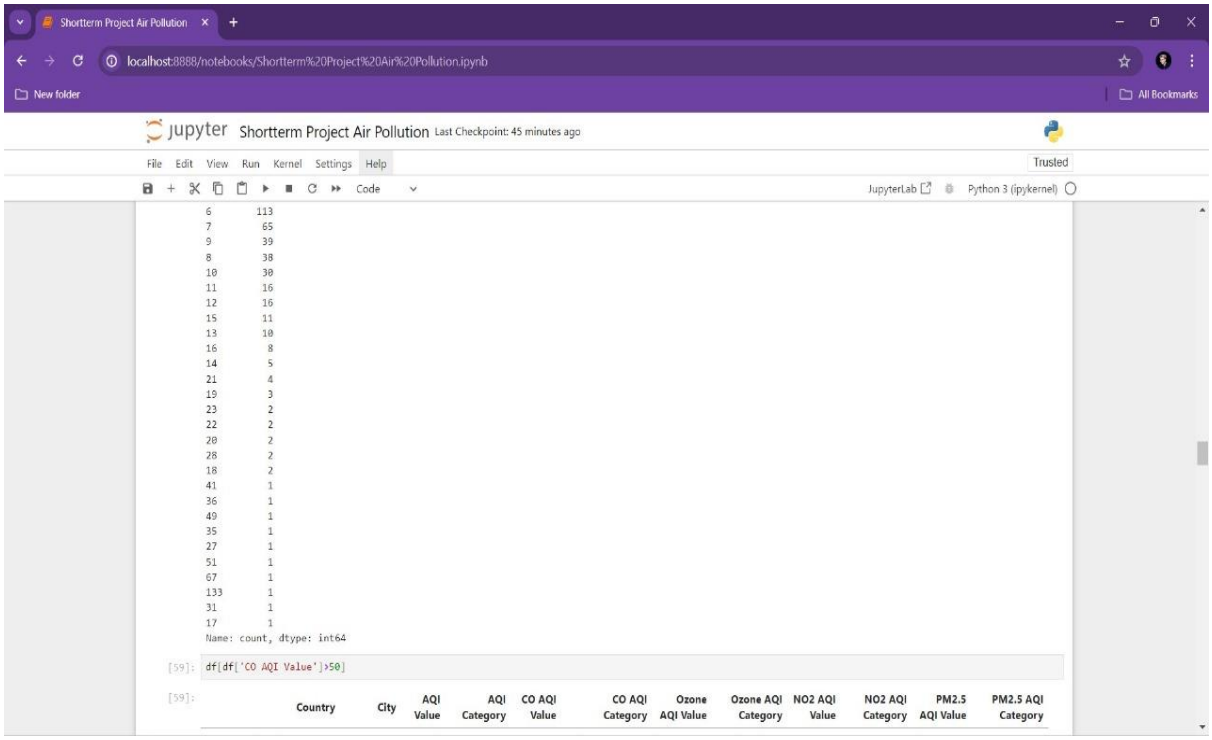
[41]:

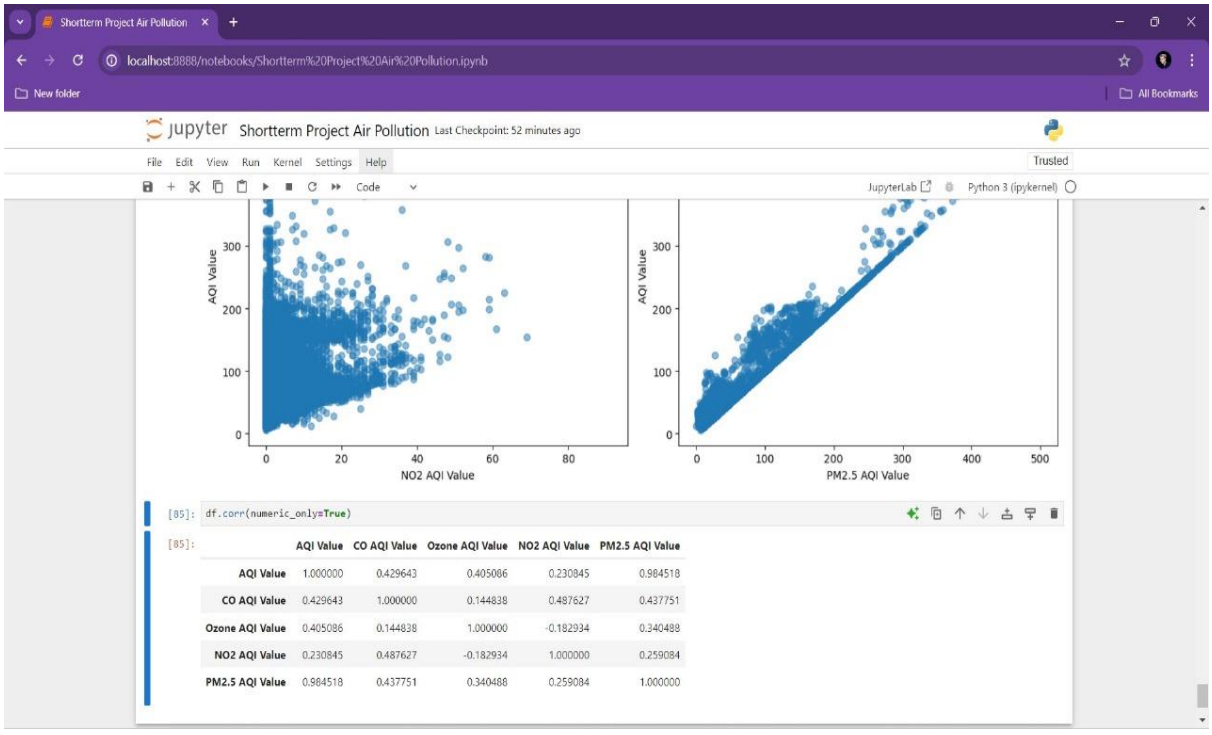
	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
180	India	Govindgarh	307	Hazardous	1	Good	45	Good	0	Good	307	Hazardous
276	Pakistan	Bahawalnagar	500	Hazardous	1	Good	38	Good	1	Good	466	Hazardous
277	South Africa	Balfour	380	Hazardous	31	Good	1	Good	26	Good	380	Hazardous
417	India	Phillaur	444	Hazardous	2	Good	111	Unhealthy for Sensitive Groups	1	Good	391	Hazardous
439	India	Salon	303	Hazardous	1	Good	48	Good	0	Good	260	Very Unhealthy
...	...	...	...	...	...	...	...	...	...	...	...	...
22560	India	Miranpur	464	Hazardous	4	Good	171	Unhealthy	5	Good	407	Hazardous
22577	India	Bahjoi	500	Hazardous	1	Good	53	Moderate	1	Good	447	Hazardous
22824	India	Bilari	500	Hazardous	4	Good	158	Unhealthy	4	Good	457	Hazardous
22842	India	Sikandarabad	500	Hazardous	2	Good	42	Good	3	Good	500	Hazardous
23201	Mexico	Tlalancaleca	355	Hazardous	4	Good	4	Good	17	Good	355	Hazardous

190 rows × 12 columns


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 44 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Brazil
Russian Federation
United States of America
Germany
Japan
France
Spain
United Kingdom of Great Britain and Northern Ireland
Italy
Netherlands
Poland
United Republic of Tanzania
Belgium
Portugal
Mexico
Turkey
India
Thailand
Ukraine
Peru
Country
[57]: df['CO AQI Value'].value_counts()
[57]: CO AQI Value
1 14593
0 3008
2 2973
3 1202
4 647
5 234
6 113


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
New folder
All Bookmarks
jupyter Shortterm Project Air Pollution Last Checkpoint: 44 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[55]: good_AQI_Country['Country'].value_counts().head(20).plot.bar()
plt.title('Country with Good Air Quality')
plt.ylabel('No.of city')
plt.show()
Country with Good Air Quality
No.of city
0
200
400
600
800
1000
Brazil
Germany
Japan
France
Spain
Italy
Netherlands
Poland
Belgium
Portugal
Mexico
Turkey
India
Thailand
Ukraine
Peru


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
jupyter Shortterm Project Air Pollution Last Checkpoint: 45 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
6 113
7 65
9 39
8 38
10 30
11 16
12 16
15 11
13 10
16 8
14 5
21 4
19 3
23 2
22 2
20 2
28 2
18 2
41 1
36 1
49 1
35 1
27 1
51 1
67 1
133 1
31 1
17 1
Name: count, dtype: int64
[59]: df[df['CO AQI Value']>50]
[59]: Country City AQI Value AQI Category CO AQI Value CO AQI Category Ozone AQI Value Ozone AQI Category NO2 AQI Value NO2 AQI Category PM2.5 AQI Value PM2.5 AQI Category


Shortterm Project Air Pollution
localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb
jupyter Shortterm Project Air Pollution Last Checkpoint: 52 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
AQI Value
NO2 AQI Value
PM2.5 AQI Value
[85]: df.corr(numeric_only=True)
[85]: AQI Value CO AQI Value Ozone AQI Value NO2 AQI Value PM2.5 AQI Value
AQI Value 1.000000 0.429643 0.405086 0.230845 0.984518
CO AQI Value 0.429643 1.000000 0.144838 0.487627 0.437751
Ozone AQI Value 0.405086 0.144838 1.000000 -0.182934 0.340488
NO2 AQI Value 0.230845 0.487627 -0.182934 1.000000 0.259084
PM2.5 AQI Value 0.984518 0.437751 0.340488 0.259084 1.000000

Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

jupyter Shortterm Project Air Pollution Last Checkpoint: 45 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[59]:
```
df[df['CO AQI Value']>50]
```

[59]:

	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
5156	United States of America	Durango	500	Hazardous	133	Unhealthy for Sensitive Groups	0	Good	53	Moderate	500	Hazardous
12800	Malaysia	Miri	209	Very Unhealthy	67	Moderate	209	Very Unhealthy	2	Good	157	Unhealthy
13398	Democratic Republic of the Congo	Kasongo Lunda	481	Hazardous	51	Moderate	39	Good	7	Good	446	Hazardous

[61]:
```
df['Ozone AQI Value'].value_counts()
```

[61]:
```
Ozone AQI Value
30     698
32     677
33     653
28     653
29     651
      ...
199      1
189      1
222      1
235      1
214      1
Name: count, Length: 213, dtype: int64
```

[63]:
```
sns.displot(df['Ozone AQI Value'])
```

Shortterm Project Air Pollution

localhost:8888/notebooks/Shortterm%20Project%20Air%20Pollution.ipynb

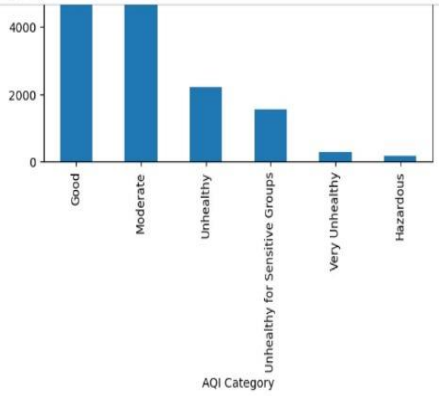
jupyter Shortterm Project Air Pollution Last Checkpoint: 39 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)



[47]:
```
maxm=df[df['AQI Value']==df['AQI Value'].max()]
maxm
```

[47]:

	Country	City	AQI Value	AQI Category	CO AQI Value	CO AQI Category	Ozone AQI Value	Ozone AQI Category	NO2 AQI Value	NO2 AQI Category	PM2.5 AQI Value	PM2.5 AQI Category
276	Pakistan	Bahawalnagar	500	Hazardous	1	Good	38	Good	1	Good	466	Hazardous
470	India	Rania	500	Hazardous	1	Good	40	Good	0	Good	464	Hazardous

CONCLUSION

In conclusion, ambient air pollution is a health hazard. It is a global challenge, as evidence shows that adverse effects still exist even at relatively low air pollutant concentrations, and so no threshold values for classical air pollutants can be established based on the available data. It is more important in Asian developing countries due to the severe pollution levels and high population densities associated with them. Improving air quality has substantial, measurable and important public health benefits. Efforts should be made and goals set in order to control air pollution in every country.