AIR QUALITY DETECTION USING PYSPARK

P. Syam Prasad

Artificial Intelligence and Engineering Amrita School of Engineering Bangalore, India [BL.EN.U4AIE21103@bl.students.amri](mailto:BL.EN.U4AIE21103@bl.students.amrita.edu) [ta.edu](mailto:BL.EN.U4AIE21103@bl.students.amrita.edu)

Vinay.K

Artificial Intelligence and Engineering Amrita School of Engineering Bangalore, India [BL.EN.U4AIE21142@bl.students.amri](mailto:BL.EN.U4AIE21142@bl.students.amrita.edu) [ta.edu](mailto:BL.EN.U4AIE21142@bl.students.amrita.edu)

P. Mokshith

Artificial Intelligence and Engineering Amrita School of Engineering Bangalore, India [BL.EN.U4AIE21094@bl.students.amri](mailto:BL.EN.U4AIE21094@bl.students.amrita.edu) [ta.edu](mailto:BL.EN.U4AIE21094@bl.students.amrita.edu)

***Abstract -* Dealing with air pollution presents a major environmental challenge in smart city environments. This project addresses the imperative need for effective air quality monitoring and prediction using machine learning techniques within the Apache Spark framework for scalable big data analysis. The dataset under consideration encompasses a diverse range of air quality parameters, meticulously normalized for flexibility and consistency using the StandardScaler from the Scikit-Learn library. Key features include various particulate matter, nitrogen oxides, ammonia, carbon monoxide, sulfur dioxide, ozone, benzene, and toluene, culminating in the prediction of the Air Quality Index (AQI) and classification into distinct AQI\_Bucket categories, from 'good' to 'severe.' Leveraging the distributed computing capabilities of Apache Spark, our project navigates through crucial stages of the machine learning pipeline. Throughout the project, the integration of machine learning models with Spark MLlib empowers us to harness the parallel processing capabilities of Apache Spark. Uses machine learning models including Naive Bayes, Support Vector Machines (SVM), Decission tree, Random Forest, and Logistic regression, and Navie Bays in the context of predictive modeling and classification tasks. And SMOTE algorithm for balancing the imbalanced dataset. enhancing the robustness of the overall predictive modeling framework. This comprehensive approach ensures a holistic and scalable solution for addressing air quality challenges in smart city environments.**

**Keywords—Apache Spark, Machine Learning, Feature Selection, air quality index (AQI), SMOTE**

1. INTRODUCTION

Air quality plays a pivotal role[1] in the well-being of communities, influencing both public health and environmental sustainability. In the pursuit of healthier living conditions, the monitoring and prediction of air quality have become essential components of modern urban planning and management. Mohd Anul [1] Has introduced five ML models, featuring SMOTEDNN, achieving exceptional 99.90% accuracy in air pollution classification. Rigorous data processing and hyperparameter tuning were pivotal. Future emphasis lies on IoT-based real-time datasets for improved air quality assessment and forecasting.The aim of this project is to develop a robust air quality detection system utilizing machine learning techniques within the Apache Spark framework for scalable big data analysis.

Our project centers around the comprehensive analysis of air quality parameters, including but not limited to Particulate Matter (PM2.5, PM10), Nitrogen Oxides (NO, NO2, NOx), Ammonia (NH3), Carbon Monoxide (CO), Sulfur Dioxide (SO2), Ozone (O3), as well as specific volatile organic compounds such as Benzene and Toluene. The dataset, normalized for flexibility and consistency using the StandardScaler from the Scikit-Learn library, provides a rich source of information for predicting the Air Quality Index (AQI) and classifying it into distinct categories represented by AQI\_Buckets, ranging from 'good' to 'severe.'

To ensure the efficiency and scalability required for handling large volumes of air quality data, we have chosen Apache Spark as our analytical engine. Dong-Her Shih [3] utilized the Spark Big Data Framework for an advanced air pollution forecasting system, focusing on real-time PM2.5 prediction from LASS community data. Their study aims for short-term predictions, providing decision-makers with a valuable tool for managing air pollution's impact on health and the environment. Marjan Asgari [4] developed a distributed solution employing Apache Hadoop and Spark for efficient urban air pollution prediction. Utilizing machine learning within the Spark framework on a Hadoop cluster, the study enhances speed and accuracy, generates predictive maps, evaluates system performance, addresses imbalanced datasets, and contributes valuable insights for public awareness and decision-making. Spark's distributed computing capabilities empower us to process and analyze vast datasets in parallel, making it well-suited for real-time and batch processing scenarios. The integration of machine learning models within Spark MLlib enables us to build predictive models, providing insights into air quality dynamics and supporting informed decision-making.

Throughout this project, we will delve into crucial stages of the machine learning pipeline, including data preprocessing, feature scaling, feature selection, and model training. The selected machine learning models are tailored to address the specific challenges posed by air quality prediction and classification, with a focus on accuracy, interpretability, and scalability. By the project's conclusion, we anticipate delivering a deployable solution that contributes to a deeper understanding of air quality patterns and facilitates proactive measures for healthier living environments.

1. Related work

Sweta Ketu [1] created a strong predictive model for Air Quality Index (AQI) and Nitrogen Oxide (NOx) levels. She used a unique approach that combines Linear Regression with Recursive Feature Elimination and Random Forest Regression (RFERF). The RFERF model was tested for accuracy in predicting air quality, and performance metrics like MAPE, MAE, MSE, RMSE, and R2 score were used for evaluation. The results show that the RFERF model performs better than other machine learning models, demonstrating higher accuracy and better prediction rates for forecasting AQI and NOx levels.

Wenjuan Wei and team [2] explored Indoor Air Quality (IAQ) prediction using a literature review, focusing on statistical models and machine learning. Key IAQ parameters, such as PM2.5, PM10, carbon dioxide, and radon, were highlighted. The study revealed widespread use of statistical models like artificial neural networks, multiple linear regression, partial least squares, and decision trees for IAQ prediction. The diverse range of approaches showcased the application of advanced modeling techniques to understand and predict IAQ across different indoor environments.

Chi-Yeh Lin, Yue-Shan Chang, Hsin-Ta Chiao [3] and Satheesh Abimannan addressed the pressing issue of PM2.5 air pollution, driven by economic development and posing threats to health and the environment. Recognizing the limitations of traditional monitoring stations in providing real-time updates for short-term PM2.5 peaks, the researchers proposed a Spark big data framework. This framework efficiently handles large and intricate datasets, offering a solution for forecasting short-term PM2.5 air pollution. Their approach incorporates three machine learning algorithms—Linear Regression, Random Forest, and Gradient Boosting Decision Tree—employing ensemble learning to predict PM2.5 concentrations for the next 30 to 180 minutes.

Yu-Ren Zeng, Yue Shan Chang, and You Hao Fang [4] the focus lies on the crucial relationship between air quality and health. Emphasizing the significance of visualization in communicating forecasted air quality data, the authors propose an architecture employing Extract-Transform-Load (ETL) within a big data platform. Computational nodes are designated for data collection and air quality forecasting, while storage nodes handle data retrieval, analysis, and preprocessing. The paper utilizes RESTful Web Service as an API and incorporates the Google Map API and D3 JavaScript library for visualization, enhancing the accessibility and comprehensibility of forecasted and monitored air quality results. The study's findings underscore the effective contribution of visualization on a big data framework to air quality analysis

Huixiang Liu, Qing Li, Dongbing Yu, and Yu Gu [5] explored the prediction of air quality index (AQI) and pollutant concentration using machine learning algorithms. Their analysis involved constructing regression models, with examples drawn from AQI predictions for Beijing and pollutant concentration predictions for an Italian city. Experimental findings revealed that both the Support Vector Regression (SVR)-based and Random Forest Regression (RFR)-based models demonstrated good results, with the RFR model exhibiting superior performance. The study established two prediction models for distinct scenarios, enhancing the accuracy of air quality indicators and offering valuable guidance for modeling and analyzing urban air quality.

Saba Ameer, Munam Ali Shah, Abid Khan, Houbing Song, Carsten Maple, Saif Ul Islam, and Muhammad Nabeel Asgar [6] conducted an in-depth examination of four established approaches for addressing air pollution prediction challenges. The analyzed techniques included Decision Tree regression, Random Forest regression, Multi-Layer Perceptron regression, and Gradient Boosting regression. The comparison was based on error rates and processing times. The results highlighted that Random Forest regression emerged as the most effective technique, demonstrating consistent performance across datasets of varying size, locations, and characteristics. Despite Decision Trees exhibiting the shortest processing time, its error rate remained higher compared to most techniques. Gradient Boosting regression, on the other hand, performed least favorably among the evaluated algorithms.

K. Kumar and B. P. Pande [7] conducted a comprehensive examination of air pollution prediction in 23 Indian cities over a six-year period. The dataset underwent meticulous cleaning and preprocessing, involving the handling of NAN values, normalization of data, and addressing data imbalance through SMOTE analysis. The data was then split into a 75:35 ratio for further analysis. Utilizing Support Vector Machine (SVM) and XGBoost machine learning models, the study evaluated metrics such as precision, recall, and F1- score. Notably, the XGBoost model demonstrated the highest accuracy for both the train and test sets, while the SVM model exhibited the lowest accuracy among the models examined.

In the 2021 Journal of Engineering Research publication by Joshi Kumar Viswanadhapalli and Albert Alexander S [8] , the study delves into the prediction and forecasting of the Air Quality Index (AQI) in Chennai. The research employs machine learning regression (MLR) and ARIMA time series models for this purpose. The MLR model exhibits satisfactory performance in predicting AQI by evaluating correlations among air pollutants and training on collected data. Meanwhile, the ARIMA model effectively forecasts AQI values for the next 15 days with a 95% confidence level, indicating its suitability for real-time implementation. The study recommends the proposed methodology for AQI prediction, emphasizing the potential of ARIMA for real- time forecasting and advocating for its practical application in future implementations.

III Dataset description

The dataset utilized for this study is accessible at the following link: https://www.kaggle.com/rohanrao/air-quality-data-in-india. It comprises both hourly and daily air quality metrics as well as the Air Quality Index (AQI) data from various monitoring stations across multiple cities in India. The dataset spans the years 2015 to 2020 and originally consisted of 29,532 rows and 16 columns.

The attributes are Date YYYY-MM-DD, City, PM2.5, PM10, NO, NO2, NOx, NH3, CO, SO2, O3, Benzene, Toluene, AQI, and AQI\_Bucket. AQI\_Bucket has six values such as good, satisfactory, moderate, poor, very poor, and severe the link to the dataset

https://www.kaggle.com/rohanrao/air-quality-data-inindia.

IV Methodology

In this paper the proposed method uses 5 machine learning algorithms that were present in pyspark mlib library. In this project, Apache Spark and PySpark are employed to leverage the benefits of distributed computing for handling large-scale datasets and training machine learning models efficiently. PySpark is utilized for splitting the dataset into training and testing sets in a distributed manner. Decision Tree, Random Forest, and Naive Bayes classifiers are trained on the Spark cluster, taking advantage of parallel processing capabilities.

Synthetic Minority Over-sampling Technique (SMOTE) plays a crucial role in addressing the challenge of imbalanced classes within the air quality dataset. Imbalanced dataset, where certain classes have significantly fewer instances than others, can lead machine learning models to be biased towards the majority class.

V Implementation

*1.Data collection and Loading*

Data collected from measuring different paramenters like PM2.5, PM10, NO, NO2, NOx, NH3, CO, SO2, O3, Benzene, Toluene, AQI is taken in an excel sheet and loaded the file into pandas dataframe

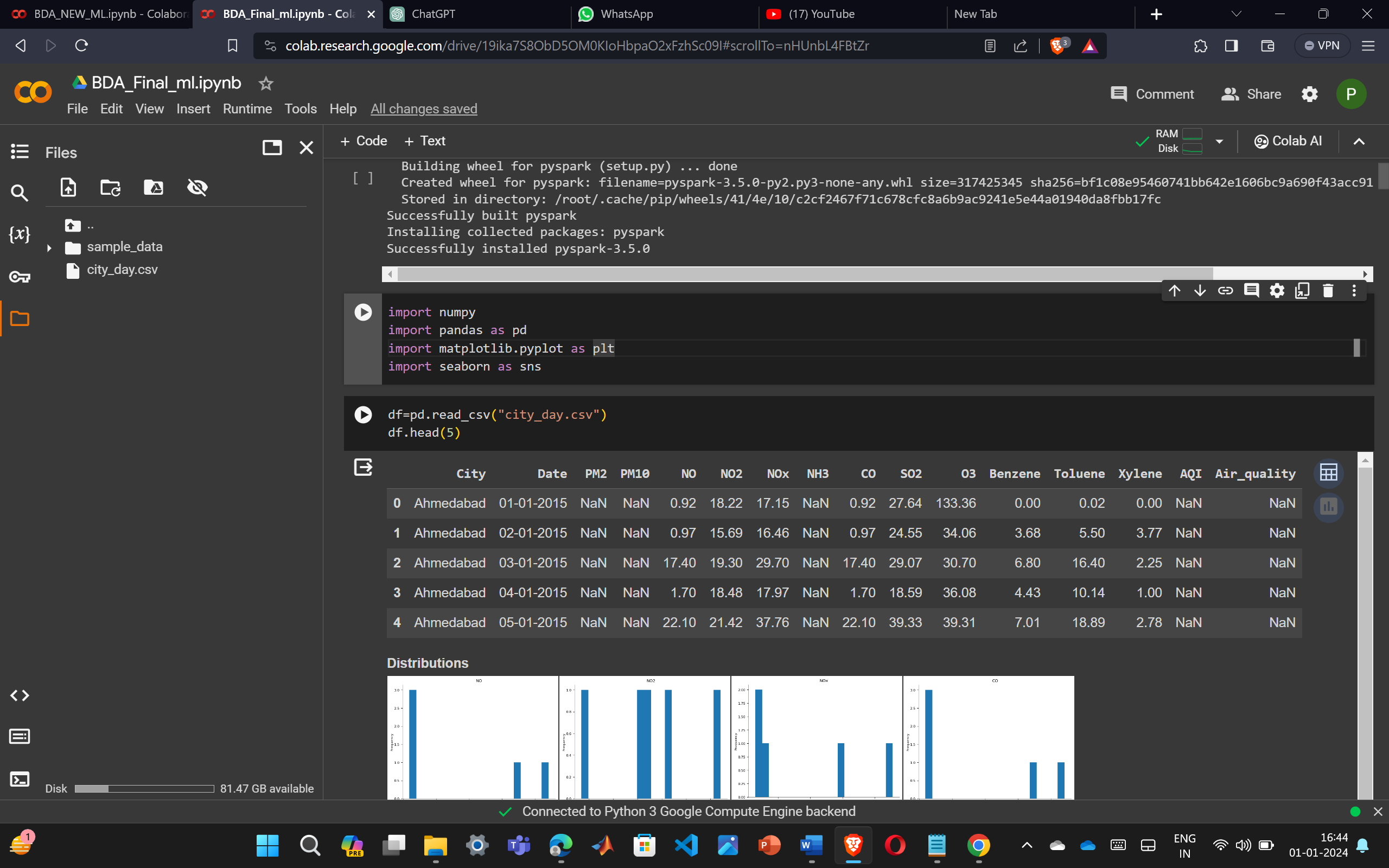


Fig 1 – Data collection printing first 5 rows

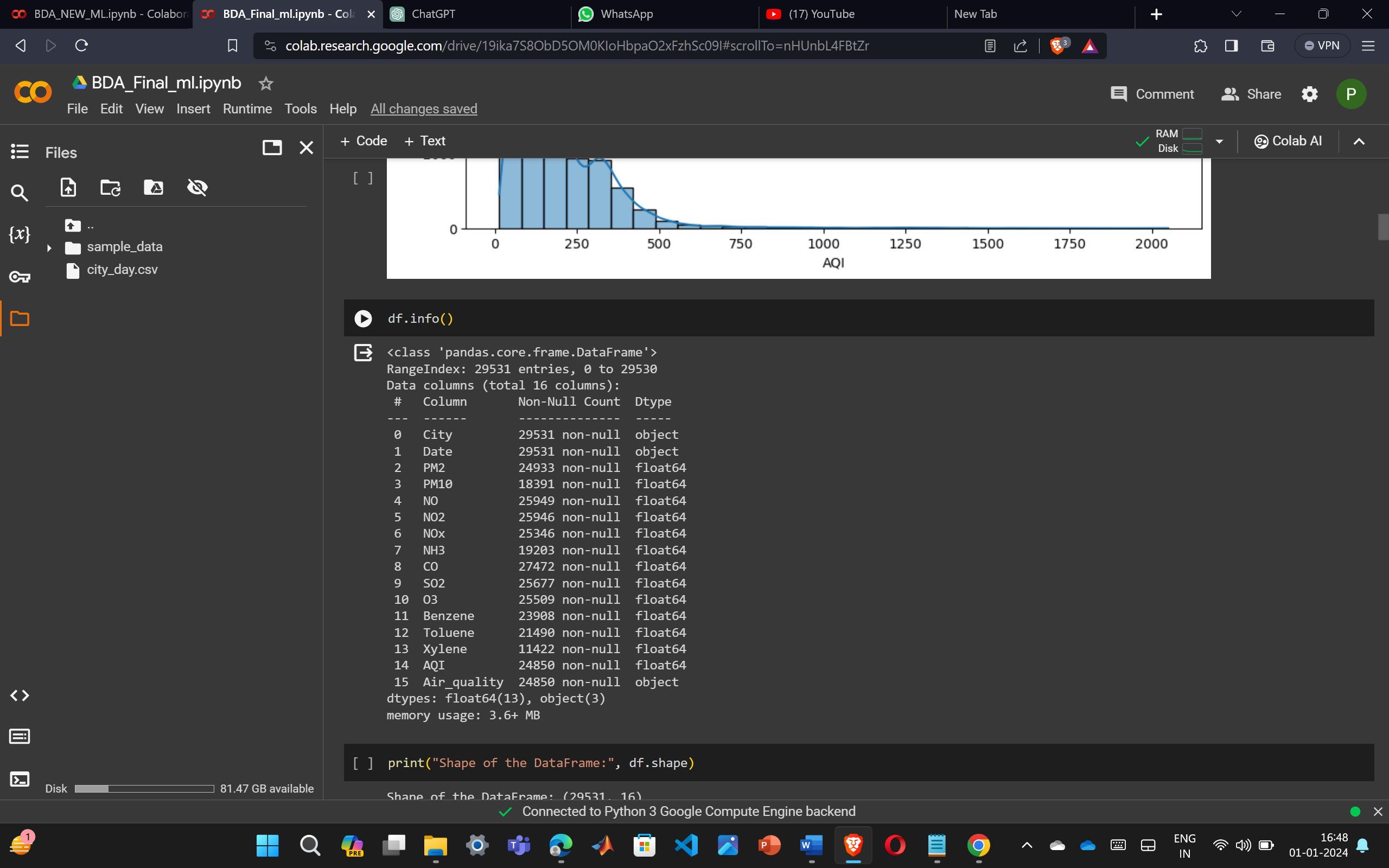


Fig 2 – data information before preprocessing

*2.Data preprocessing*

Data preprocessing is a crucial step in the pipeline of this air quality prediction project, aimed at enhancing the quality and suitability of the raw dataset for subsequent analysis and modeling

Handling Missing Values: The dataset is assessed for missing values in various features, and appropriate strategies are employed to fill or impute these gaps. For numerical features such as PM2.5, PM10, and others, missing values may be replaced with statistical measures like the median or mean. Categorical features, such as "Air\_quality," are imputed with the a default value.

The 'City' and ‘Date’ column is removed because, in machine learning models, the city and Date might not be a relevant predictor for air quality. Removing this categorical variable helps streamline the dataset and avoids potential biases associated with specific geographical locations.

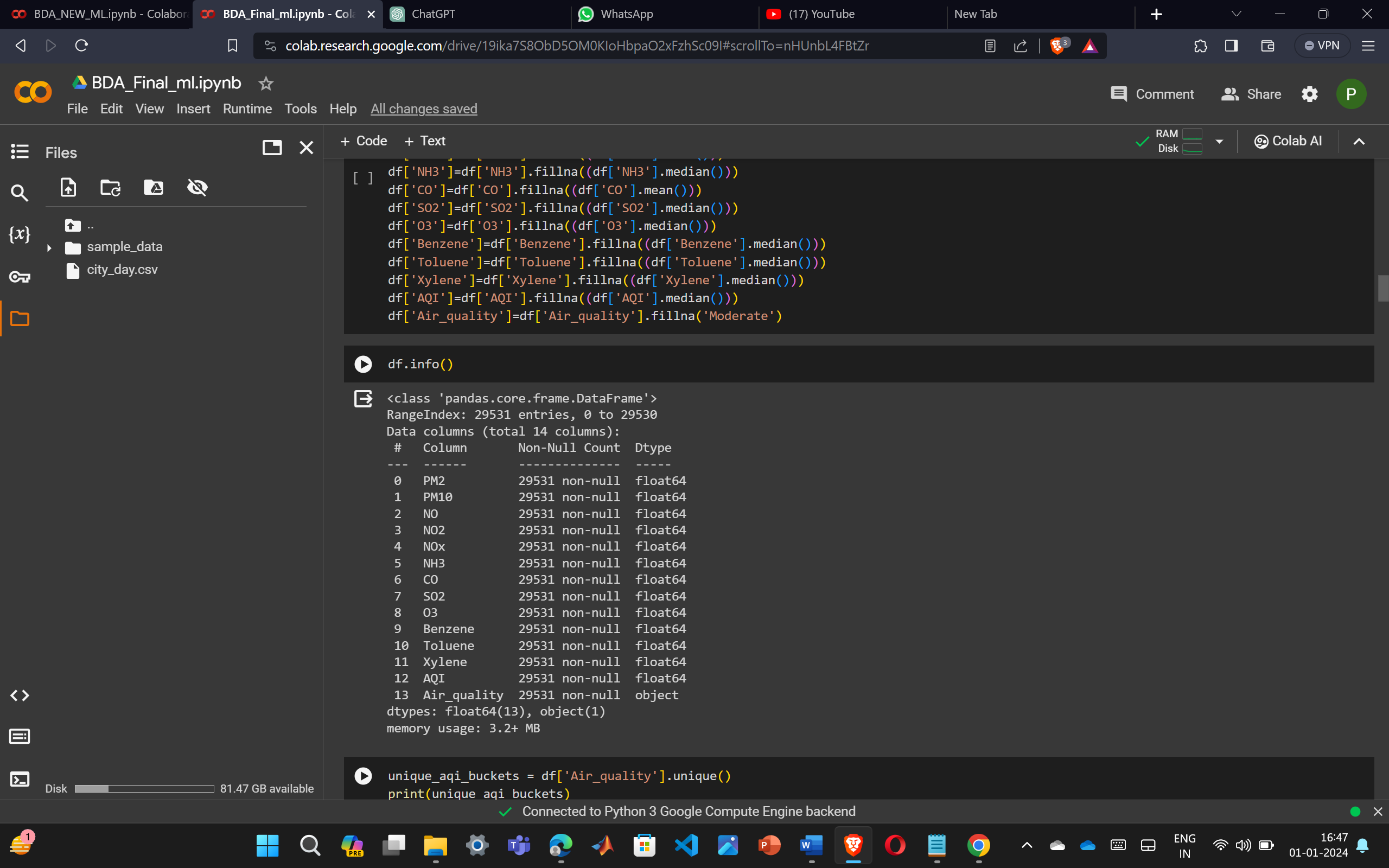


Fig 3 – Data information after preprocessing

*3 SMOTE - Synthetic Minority Over-sampling Technique*

SMOTE is a resampling technique commonly employed to address class imbalance in machine learning datasets. In this project , SMOTE is applied to mitigate the imbalance in the distribution of air quality categories ('Air\_quality') within the dataset.

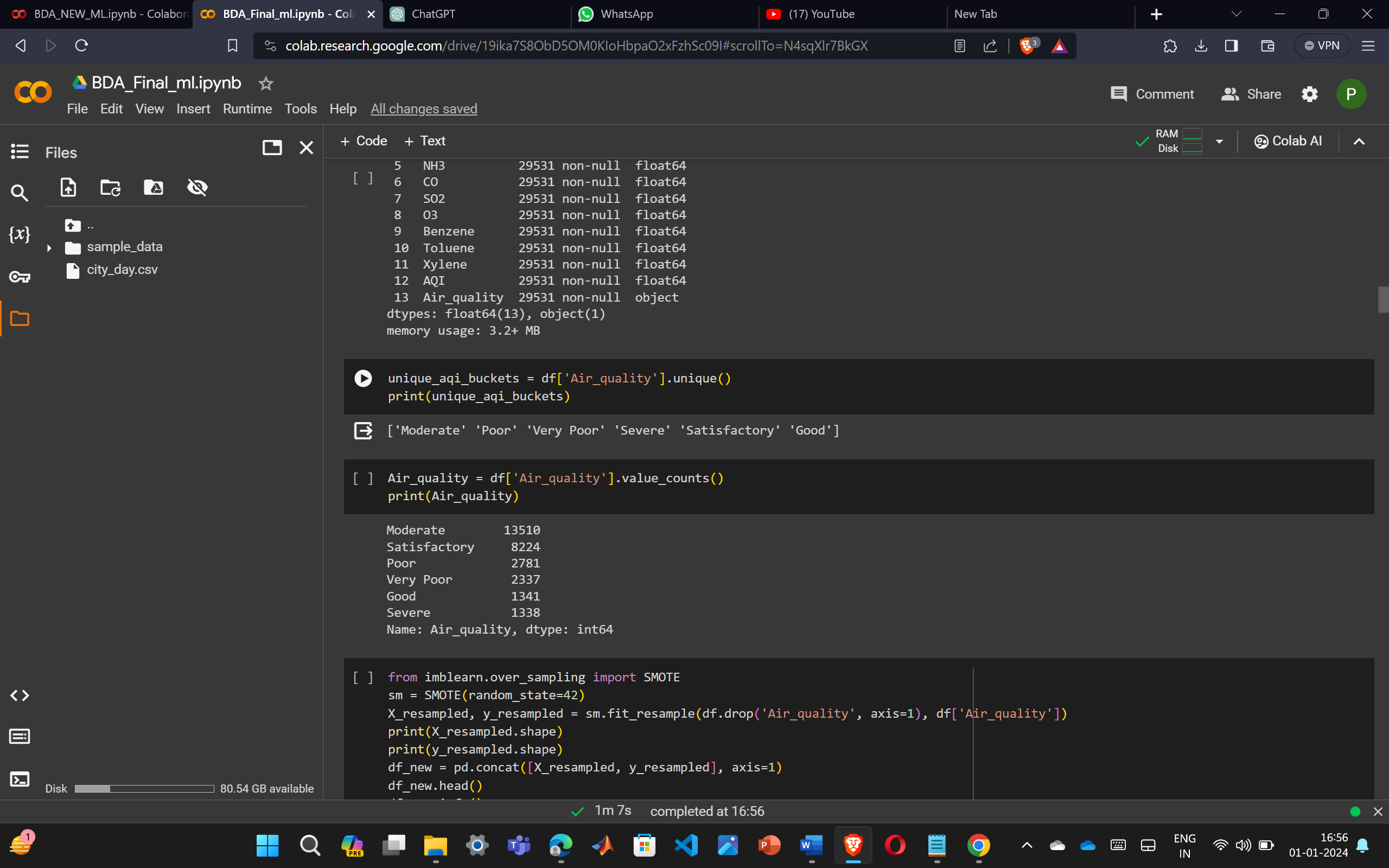


Fig 4- Before Smote analysis

The initial distribution of air quality categories exhibits significant imbalances, with 'Moderate,' 'Satisfactory,' 'Poor,' 'Very Poor,' 'Good,' and 'Severe' categories having varying counts. After applying SMOTE, the resulting dataset now exhibits a balanced distribution across all air quality categories, with each category having approximately the same number of occurrences 13510 occurances

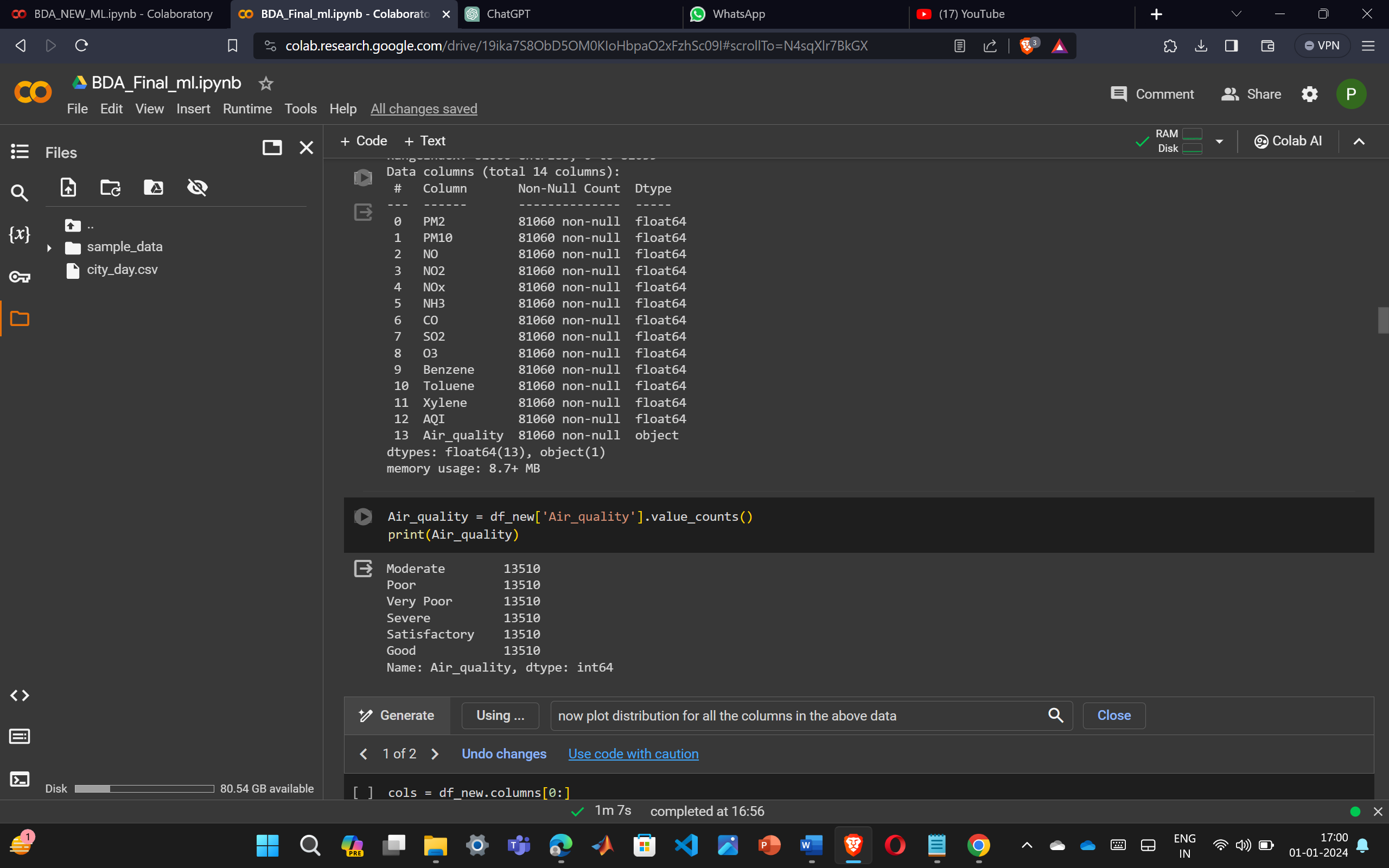


Fig – 5 After Smote analysis

*4. Feature Selection*

feature selection is performed using the chi-squared (chi2) statistical test through the scikit-learn library's SelectKBest method. This technique is employed to identify and retain the most relevant features from the dataset based on their statistical significance. The SelectKBestclass is instantiated with the chi-squared test as the scoring function, and the parameter k is set to 10, indicating the intention to select the top 10 features

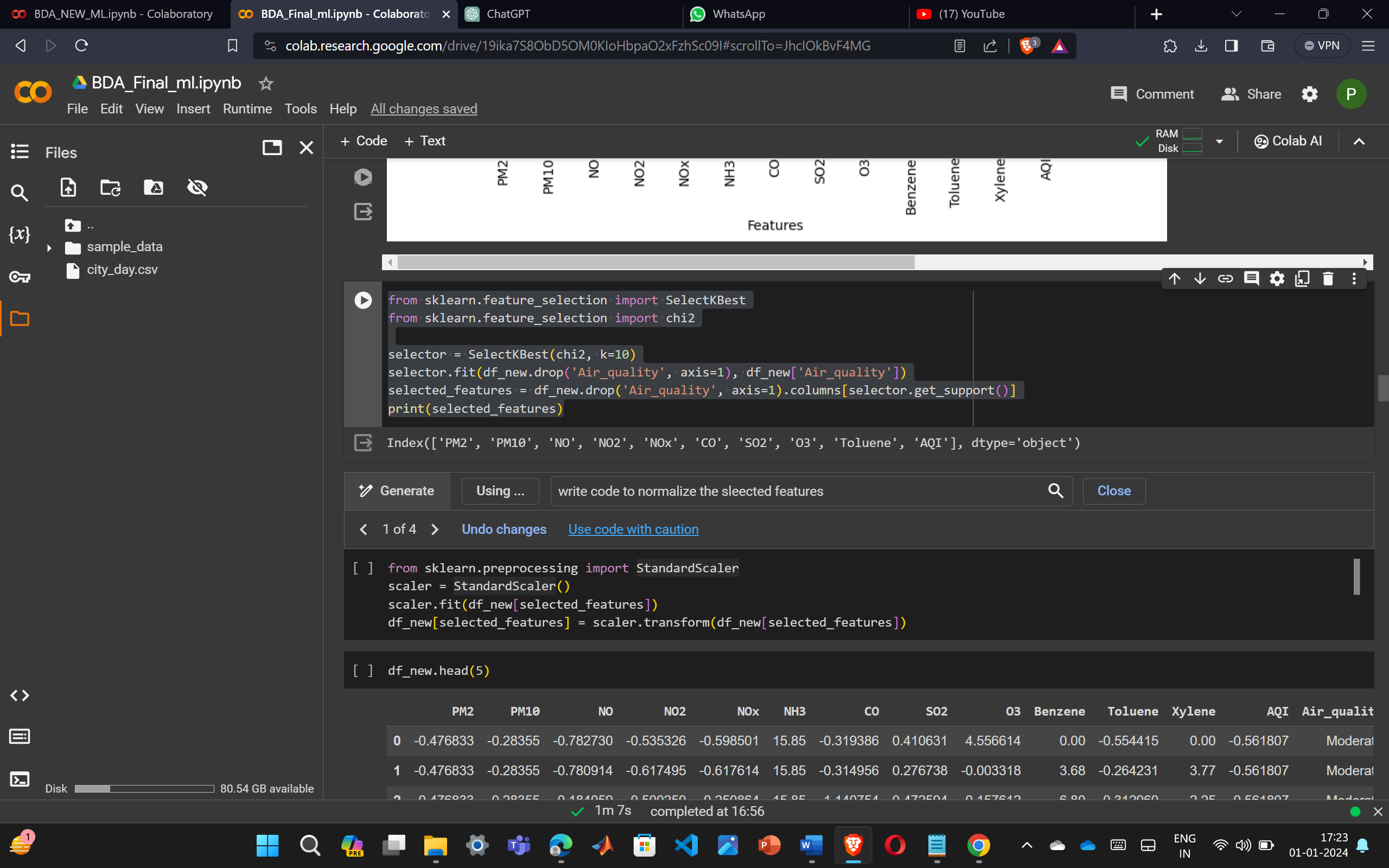


Fig 6- Feature selection

*6. Model preparation*

A Spark session is initiated, creating a platform for leveraging distributed computing capabilities. The Pandas DataFrame is then transformed into a Spark DataFrame, allowing for the efficient processing of large-scale data.

The data for classification, the target column,"Air\_quality," undergoes string indexing. This transformation is essential for converting categorical labels into a numerical format. The indexed labels are stored in a new column called "label," this is a fundamental step in PySpark's machine learning workflow.

The VectorAssembler is a feature transformer in PySpark's MLlib that combines a given list of columns into a single vector column

*7. Correlation Matrix*

A correlation matrix is a table that shows the correlation coefficients between many variables. Each cell in the table represents the correlation between two variables. Correlation coefficients are statistical measures used to describe the strength and direction of a linear relationship between two variables.

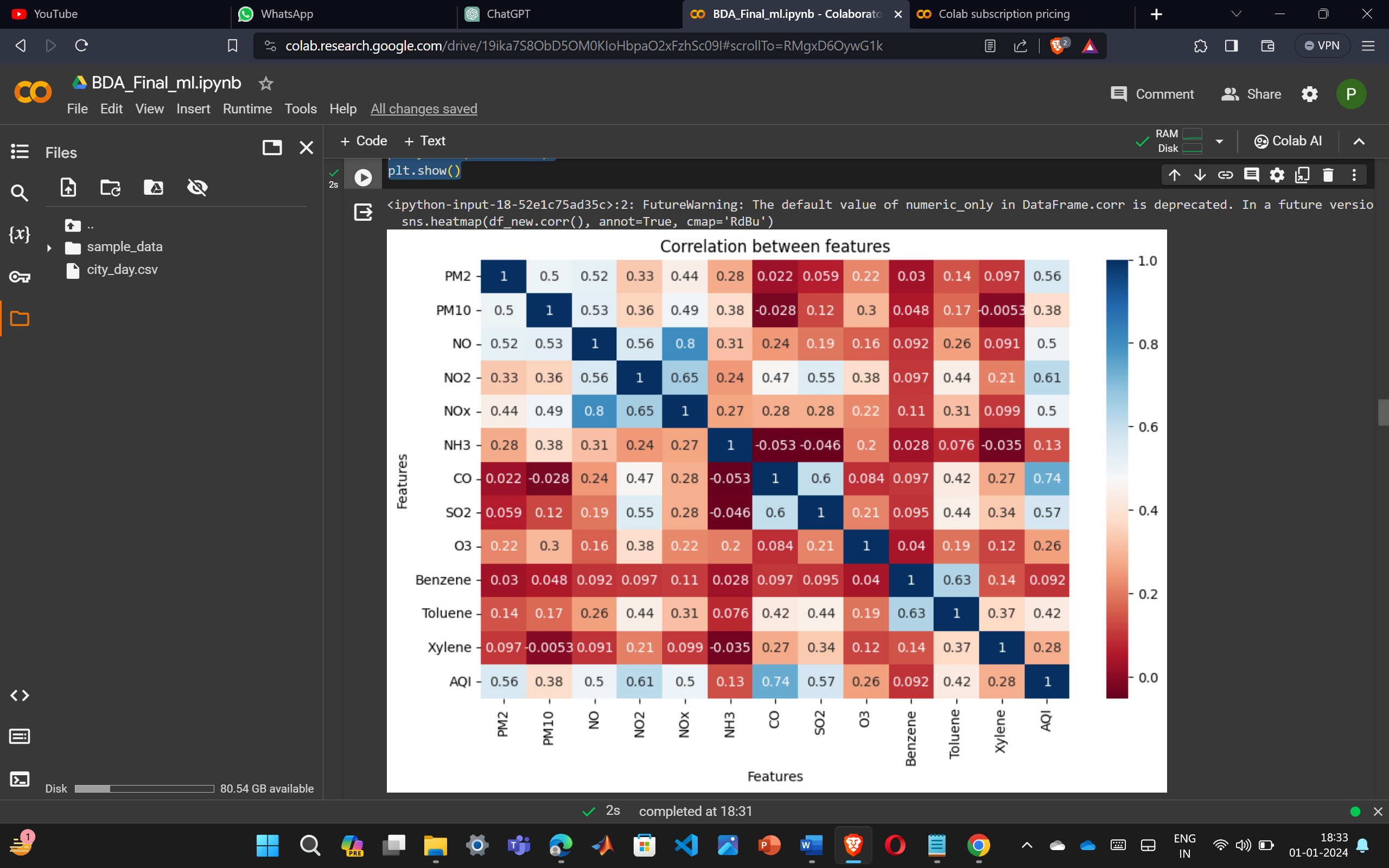


Fig 7 - correlation matrix

*7. Testing and training*

The testing and training step involves splitting the data into training and testing sets, denoted as train\_data and test\_data. This partitioning is essential for evaluating the model's performance. The random splitting is conducted with a 70-30 ratio, where 70% of the data is used for training and 30% for testing.

8*. Applying Machine learning techniques present in mlib* pyspark

SVM: In PySpark's MLlib, Support Vector Machines (SVM) are implemented for multiclass classification using the LinearSVC class. The process typically involves preparing the data, assembling features into a single column using VectorAssembler, creating the SVM model, and training it using the provided training dataset. For multiclass classification, the OneVsRest strategy is used ,where a binary classifier is trained for each class against the rest. This approach enables the extension of binary classifiers to handle multiple classes

Avg-F-measure: 0.6070760440864348

Accuracy: 0.625329163923634

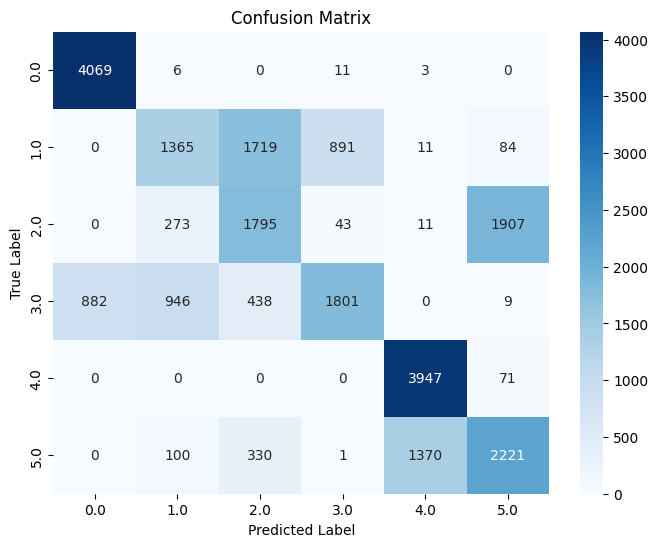


Fig 8 - Confusion Matrix for SVM

Decission tree- In PySpark's MLlib, Decision Trees serve as powerful tools for both classification and regression tasks. Leveraging the DecisionTreeClassifier for classification models recursively partition the input data based on feature values, creating a hierarchical structure that aids in decision-making. Model evaluation is performed using appropriate metrics, such as the Multiclass Classification Evaluator for classification tasks.

Avg-F1 score: 0.9662180869258652

Accuracy: 0.9660961158657011

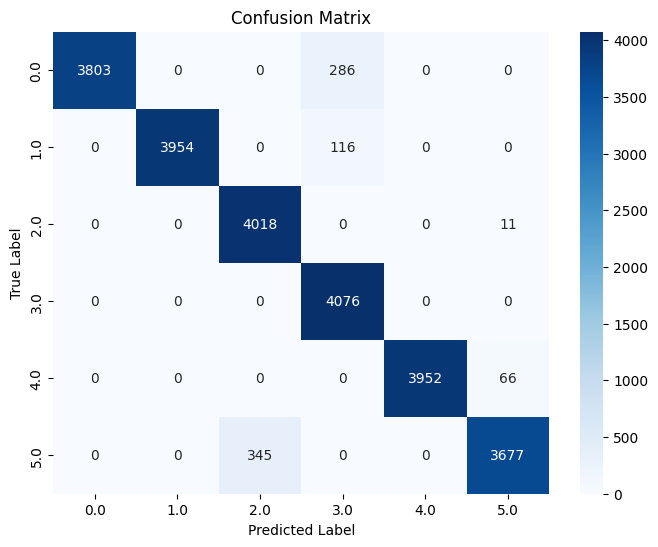


Fig 9 – Confusion matrix for decision tree

Random Forest - In PySpark's MLlib, Random Forests standas a powerful ensemble learning technique used for classification classes encapsulate the implementation of Random Forests, which consist of an ensemble of decision trees. Each tree is constructed independently, contributing to the overall prediction by means of a majority vote in classification or an average in regression

Avg-F1 score for Random Forest: 0.9485653178152871

Accuracy for Random Forest: 0.9482801184990125

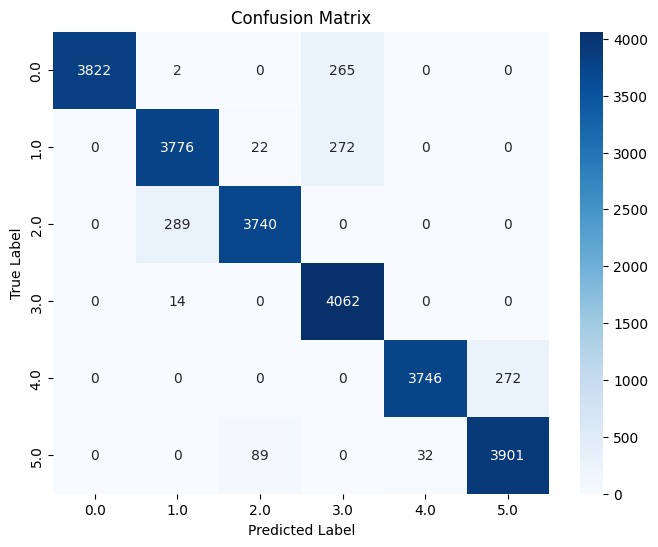


Fig-10 Confusion matrix for random forest

Navie Bays - In PySpark's MLlib, the Naive Bayes classifier is implemented for probabilistic classification. Specifically, the NaiveBayes class is available for both binary and multiclass classification tasks. Naive Bayes is a probabilistic model based on Bayes' theorem and assumes that the features are conditionally independent given the class label.

Avg-F1 score for Naive Bayes: 0.47118206235059534

Accuracy for Naive Bayes: 0.4735023041474654

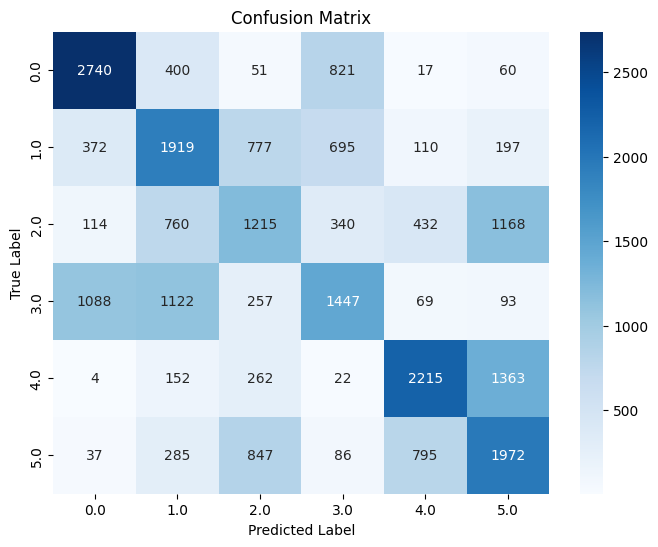


Fig – 12 Confusion matrix for navie bays

Logistic regression- In PySpark's MLlib, Logistic Regression serves as a powerful classification algorithm for both binary and multiclass classification tasks. It leverages the logistic function to model the probability that a given instance belongs to a particular class

Avg-F1 score for Logistic Regression: 0.9720295889363959

Accuracy for Logistic Regression: 0.9721445029624753

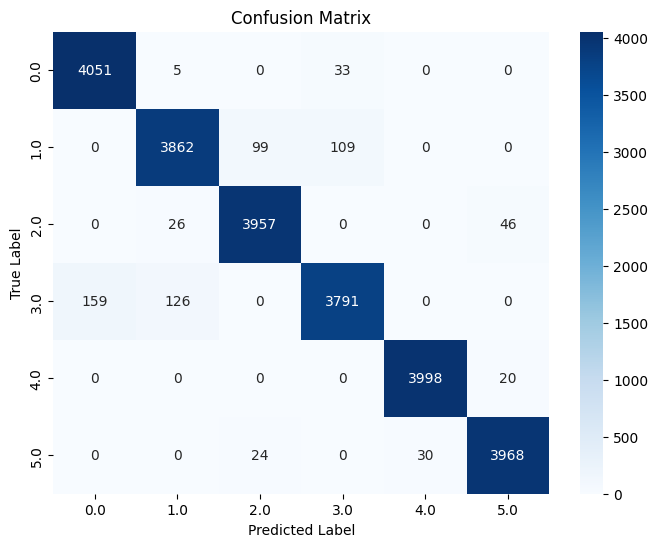


Fig – 11 Confusion matrix for logistic regression

VI Result

In our experiment, we have achieved the highest model precision for the Logistic regression compared to the other models and the logistic regression, making it a reliable model for the air quality prediction .

A graph showing a line

Description automatically generated

Fig 13 - accuracy Graph for all 5 models

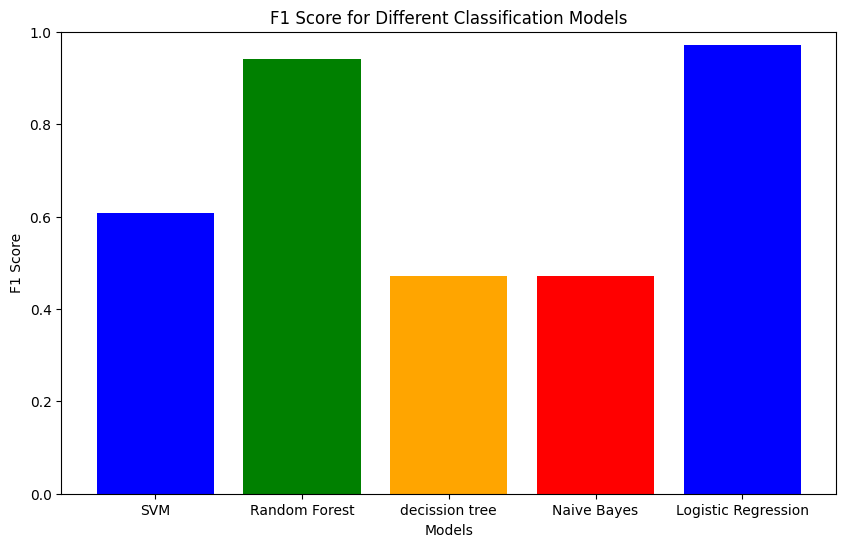


Fig 14 - F1 Score for all 5 models

VII Conclusion and future scope

Out of all 5 models navie bays performed less with 48% . Next to navie bays SVM performed worse with 60 % accuracy. Out of all 5 models Logistic regression performed the best with 97.5 % which is suitable for predicting the air quality

incorporating real-time data streams to make the model adaptable to dynamic changes in air quality. This could involve setting up data pipelines that continuously update the model with the latest information. And deploying the model in a cloud environment for scalability and ease of access. Platforms like AWS, Azure, or Google Cloud offer services that facilitate the deployment and management of machine learning models

VIII References

[1] Shwet Ketu “Spatial Air Quality Index and Air Pollutant Concentration prediction using Linear Regression based Recursive Feature Elimination with RFERF: a case study in India -2022.”

[2] Wenjuan Wei, Olivier Ramalho, Laeticia Malingre, Sutharsini Sivanantham, John C. Little, Corinne Mandin A Machine learning and statistical models for predicting indoor air quality -2019.

[3] Chi-Yeh , Lin,Yue-Shan Chang,Hsin-Ta, Chiao,Satheesh Abimannan “Design of a Spark Big Data Framework forPM2.5 Air Pollution Forecasting -2021”

[4] Yu-Ren Zeng; Yue Shan Chang; You Hao Fang “Data Visualization for Air Quality Analysis on Bigdata Platform (July 2019)”

[5] Huixiang Liu,Qing Li,Dongbing Yu andYu Gu “Air quality index and air pollutant concentration prediction based on machine learning algorithms(Applied Sciences, vol. 9, p. 4069, 2019)”

[6] Saba Ameer , Munam Ali Shah , Abid Khan , Houbing Song , Carsten Maple , Saif Ul Islam And Muhammad Nabeel Asgar “Comparative Analysis of Machine Learning Techniques for Predicting Air Quality in Smart Cities” – 2019

[7] K. Kumar & B. P. Pande “Air pollution prediction with machine learning: a case study of Indian cities” International Journal of Environmental Science and Technology volume 20 - 2023

[8] Joshi Kumar Viswanadhapalli Albert Alexander S “Prediction and Forecasting of Air Quality Index in Chennai using Regression and ARIMA time series models(Journal of Engineering Research, vol. 9, 2021)