

**DATA WAREHOUSE MINING (CT-463)**

**Department of Computer Science and Information Technology**

**PROJECT: DATA WAREHOUSE DESIGN IMPLEMENTATION**

**GROUP MEMBERS:**

1. Muhammad Abdul Rafay Shaikh (CT-21035).
2. Rohan Ahmed Siddiqui (CT-21050).
3. Muhammad Umair Shaikh (CT-21048).
4. Syed Sumam Zaidi (CT-21038).

**Section:** A.

Contents

[DATA FROM CSV 3](#_Toc182149355)

[Introduction 3](#_Toc182149356)

[Data Extraction: 3](#_Toc182149357)

[Data Transformation: 3](#_Toc182149358)

[Data Load: 4](#_Toc182149359)

[Machine Learning Model 4](#_Toc182149360)

[DATA FROM WEB SCRAPING 8](#_Toc182149361)

[Introduction 8](#_Toc182149362)

[Data Extraction 8](#_Toc182149363)

[Data Transformation 8](#_Toc182149364)

[Data Load 9](#_Toc182149365)

[DATA FROM API 12](#_Toc182149366)

[Introduction 12](#_Toc182149367)

[Data Extraction 12](#_Toc182149368)

[Data Transformation 13](#_Toc182149369)

[Data Load 14](#_Toc182149370)

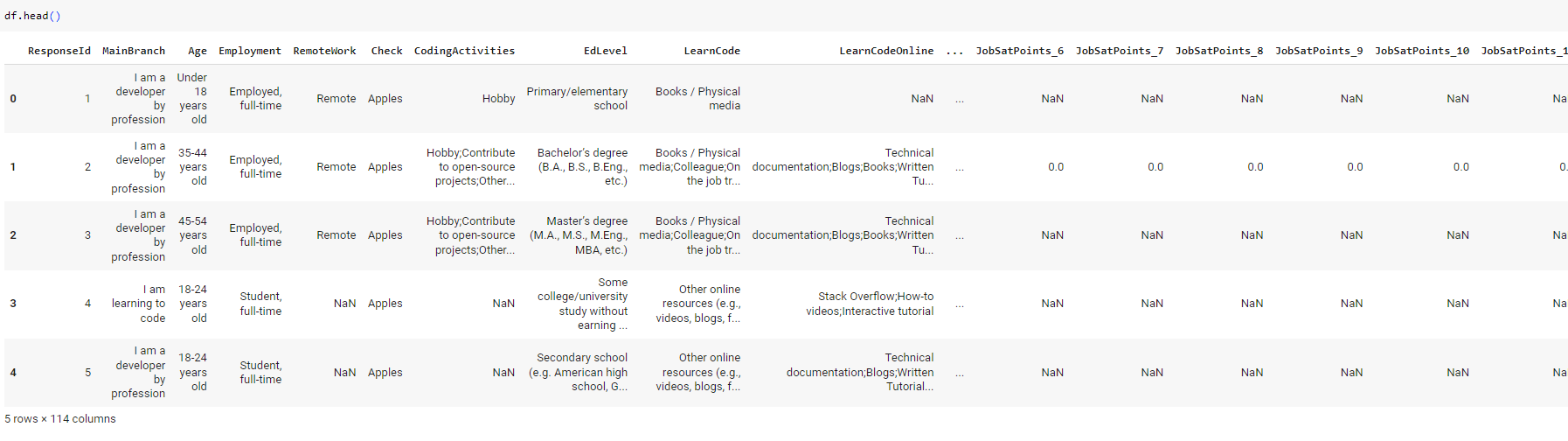
# 

# DATA FROM CSV

## Introduction

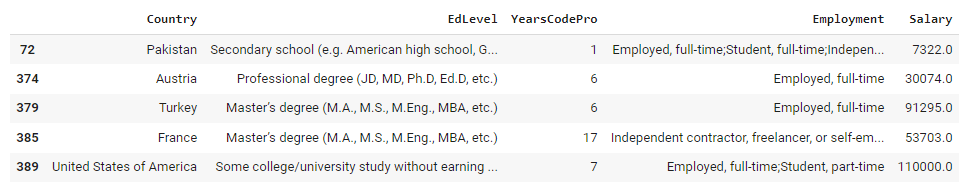
The objective of this dataset is to analyze data collected from a survey (likely on employment, educational level, coding experience, and salary) and develop a machine learning model to predict annual salary. The dataset, structured with variables like Country, Education Level (EdLevel), Years of Professional Coding Experience (YearsCodePro), Employment, and Annual Compensation (ConvertedCompYearly), is explored and refined to extract insights and prepare it for predictive modeling.

Data Extraction:  
The initial data extraction step involves reading the survey data into a pandas DataFrame. The data is loaded from a CSV file containing various columns, with only the relevant ones—Country, EdLevel, YearsCodePro, Employment, and ConvertedCompYearly—being retained for analysis.

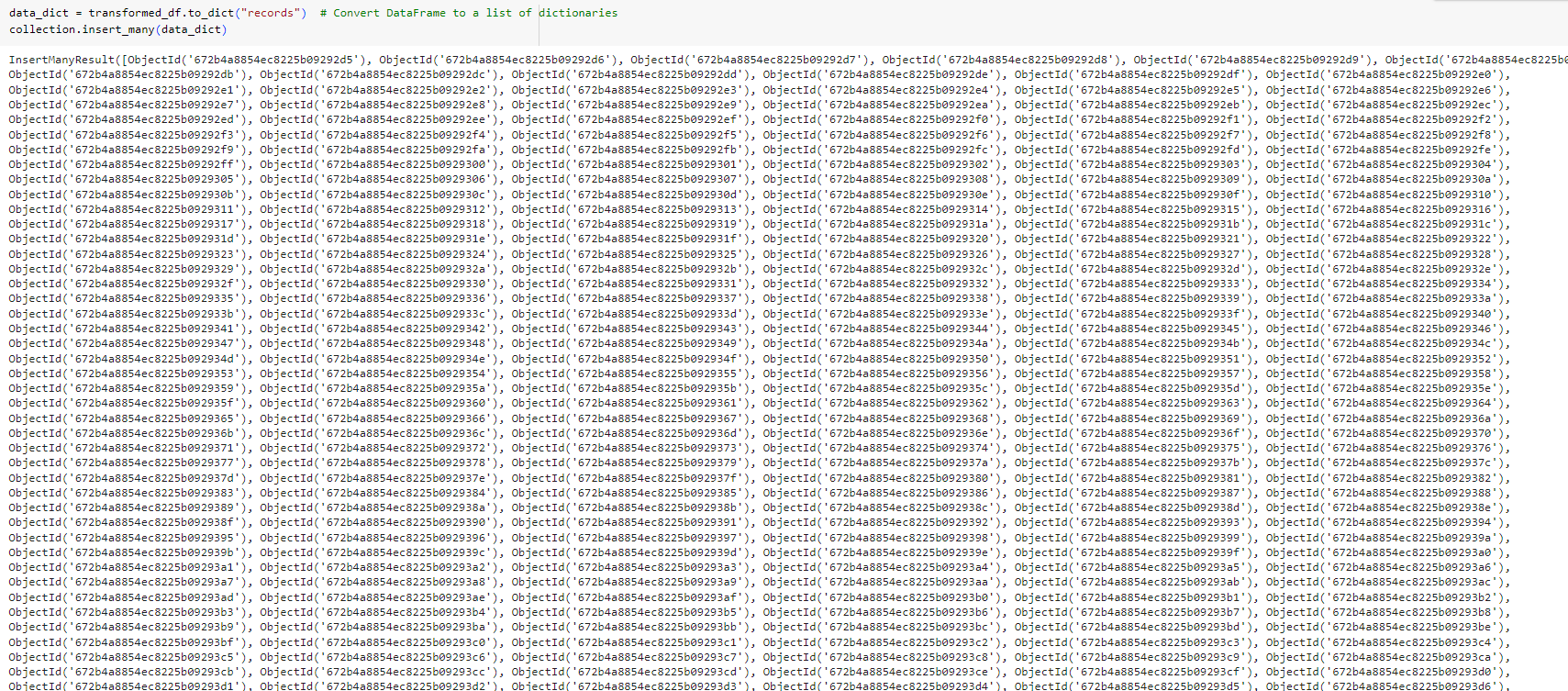


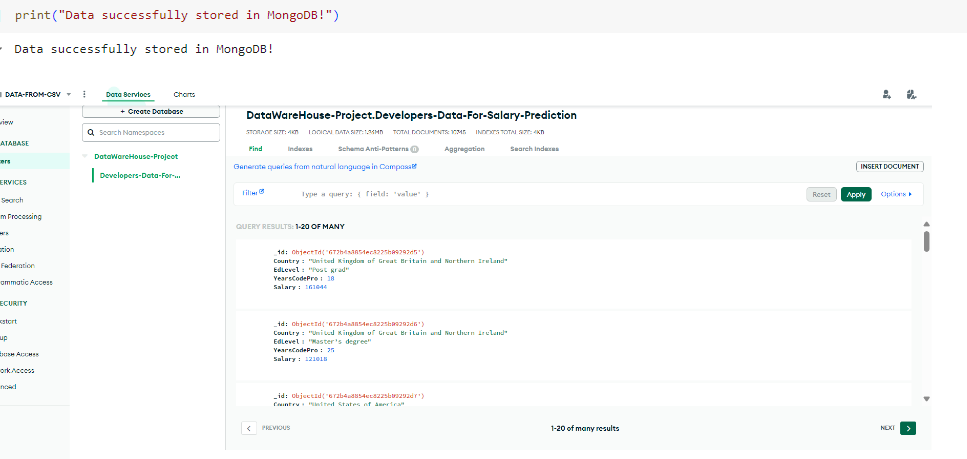
Data Transformation:  
The dataset undergoes a series of transformations to ensure it is suitable for modeling. This process includes:

* **Data Cleaning**: Handling missing values, if any, in critical columns, and filtering out or imputing invalid or outlier entries.
* **Feature Engineering**: Transforming YearsCodePro (years of professional experience) and EdLevel (educational level) into numerical forms if necessary. These variables may also be normalized or categorized to improve the model’s accuracy.
* **Encoding Categorical Variables**: Variables like Country and Employment are converted into numerical values through encoding methods (e.g., one-hot encoding or label encoding).
* **Data Splitting**: The data is split into training and test sets to allow for model training and evaluation.



Data Load:  
Once transformed, the prepared data is loaded into the machine learning pipeline, where it undergoes further processing, including scaling and any necessary adjustments to optimize the data for model training.





Machine Learning Model  
A machine learning model (possibly a regression model, given the focus on salary prediction) is implemented to predict the ConvertedCompYearly variable. The process includes:

1. **Model Selection**: Selection of an appropriate algorithm, potentially involving a comparison of several algorithms to determine the best fit.
2. **Training**: Training the model on the prepared dataset.
3. **Evaluation**: Evaluating model performance using standard metrics such as Mean Absolute Error (MAE) or Mean Squared Error (MSE) to gauge accuracy in salary prediction.
4. **Hyper parameter Tuning**: Fine-tuning model parameters to improve performance.



**Models Tested**

1. **Decision Tree Regressor**:
   * This model partitions the data recursively based on feature values, creating a tree structure. It’s known for its interpretability and ability to capture non-linear relationships.
   * **RMSE**: $31,646.22
2. **Random Forest Regressor**:
   * An ensemble model that creates multiple decision trees and averages their predictions, which generally improves accuracy and reduces overfitting.
   * **RMSE**: $31,776.30
3. **Decision Tree Regressor (Hyperparameter Tuning)**:
   * Using **GridSearchCV**, we tuned the max\_depth parameter of the Decision Tree Regressor to find the best-performing tree depth.
   * **RMSE**: $33,683.23

**Best Model Selection**

Based on the RMSE values, the **Decision Tree Regressor** without hyperparameter tuning had the lowest error, with an RMSE of $31,646.22, making it the most accurate model for our dataset. Although the Random Forest Regressor usually performs well, in this case, the simpler Decision Tree Regressor performed slightly better.

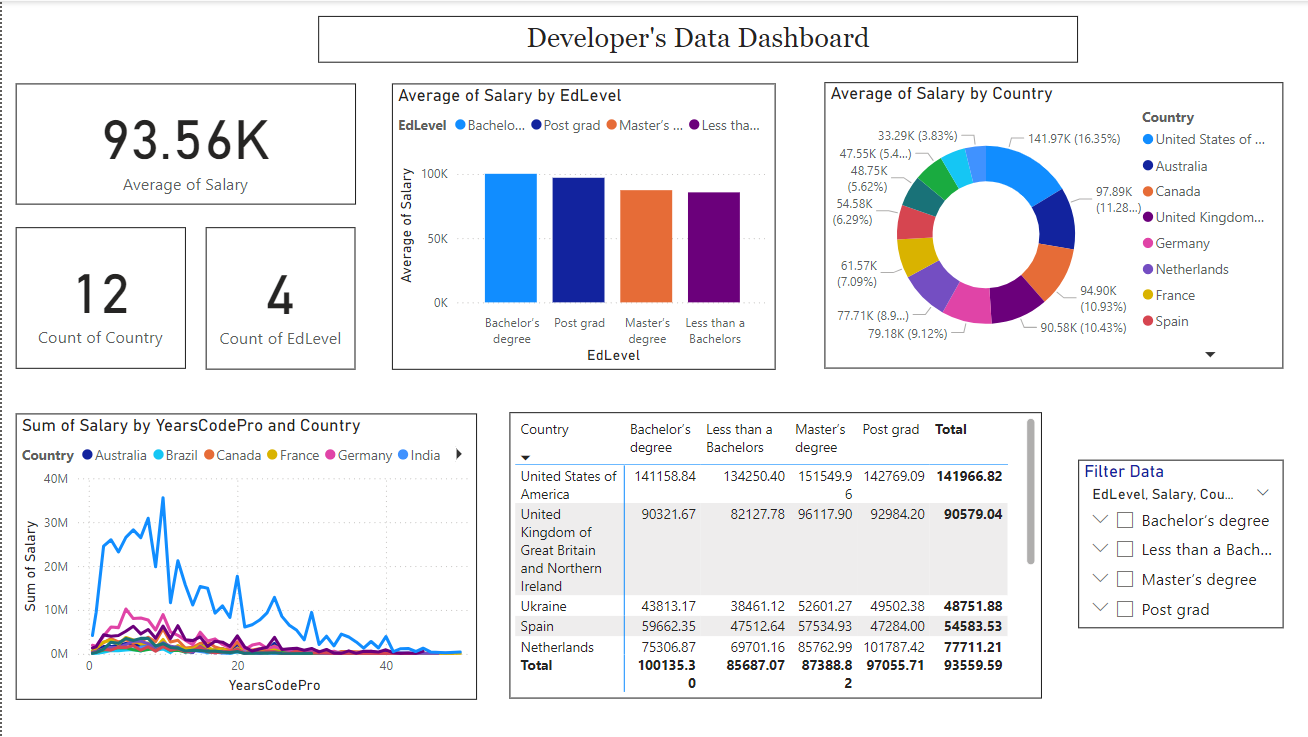
#### **Predictions**

We used the **Decision Tree Regressor** model to make predictions on the test dataset. Below are the prections

|  |  |
| --- | --- |
|  |  |

# 

With the help of PowerBI, Dashboard has been created to analyze insights.



# DATA FROM WEB SCRAPING

Introduction  
The purpose of this project is to predict car prices based on data scraped from the PakWheels website. PakWheels is a popular platform for buying and selling vehicles, and it provides valuable insights into car market trends in Pakistan. This project aims to gather data from PakWheels, process it, store it in a database, and then use machine learning models to predict car prices based on features such as model, year, and other car specifications.

Data Extraction  
To collect data from PakWheels, we used web scraping techniques to gather information from car listings. This process involved:

* **HTML Tags**: We navigated through the HTML structure of the PakWheels website to locate relevant data elements. Each car listing had specific tags for information like model, year, price, and contact details. By examining these tags, we identified where each piece of data was stored within the HTML document.
* **strip() Function**: The strip() function was used to clean up the text data extracted from these tags, removing unnecessary whitespace and formatting issues. This helped in making the data consistent and ready for further processing.

This extraction process resulted in a dataset with essential car details, either showing a price directly or indicating that the user should call for price information.

Data Transformation  
After extracting the raw data from PakWheels, several data transformation steps were performed to clean and prepare the dataset for analysis and modeling. This process included **handling missing values**, **splitting the data into subsets**, and **applying label encoding** to categorical variables.

**Handling Missing Values**

The dataset contained some missing values, especially in fields like price (when listings prompted users to call for the price) and certain car specifications. To manage this:

1. **Price Column**: Listings without explicit price values were categorized separately into the data\_call subset (listings with “call" instead of a price). Listings with price values were placed into the data\_update subset, which was used for predictive modeling.
2. **Other Missing Values**: For missing values in other columns, such as car specifications, we either imputed these values with a suitable placeholder (such as "Unknown") or dropped rows with extensive missing data, depending on the importance of the feature for our analysis.

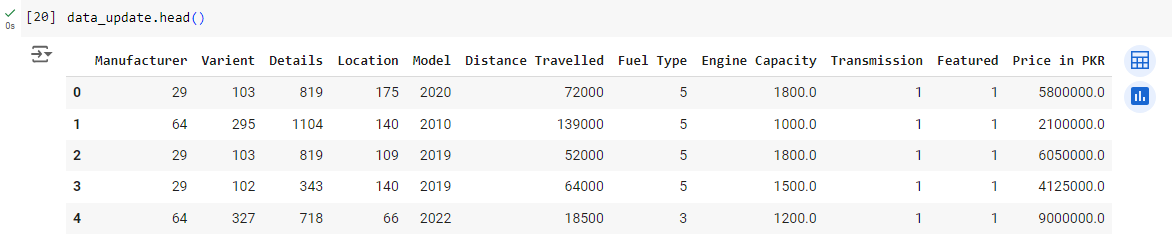
This process ensured that the data was clean and ready for machine learning, with minimal missing information that could interfere with model training.

**Label Encoding**

To enable machine learning algorithms to process categorical data, we applied **label encoding** to relevant categorical features. Label encoding is a technique that converts categorical variables (text) into numerical values, allowing models to interpret them effectively.

1. **Car Model and Brand**: Car models and brands were label-encoded so that each unique model and brand was represented by a unique numerical identifier.
2. **Fuel Type, Transmission, and Condition**: Other categorical fields, such as fuel type (e.g., Petrol, Diesel), transmission (e.g., Manual, Automatic), and car condition (e.g., New, Used), were also label-encoded.

Label encoding helped to transform the data into a numerical format, making it compatible with the machine learning algorithms while preserving the distinct categories.



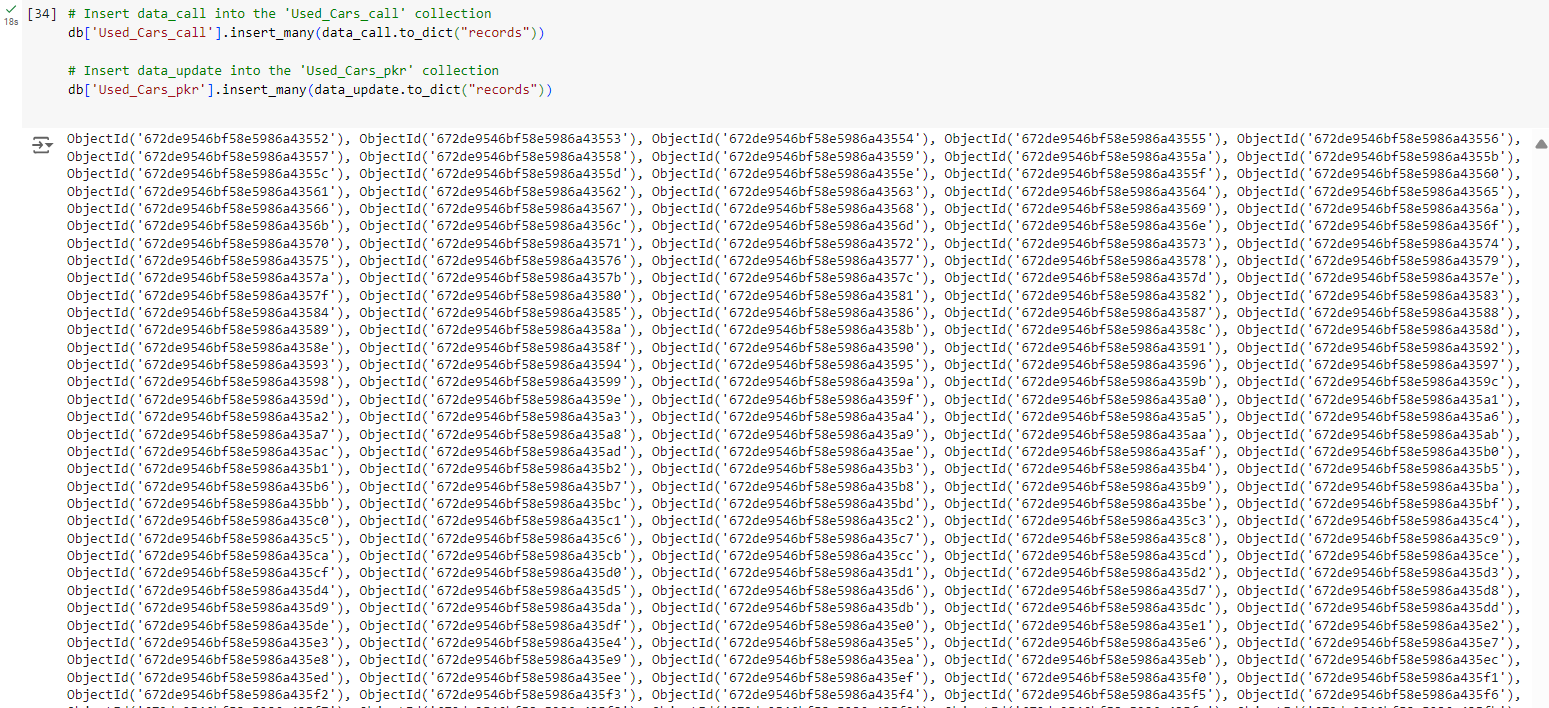
**Data Splitting**

Following data cleaning and encoding, we split the dataset into two main subsets:

1. **data\_call**: This subset contained listings where users were prompted to "call" for price information. Since it did not contain actual price values, this subset was not used for model training but could be retained for future analysis.
2. **data\_update**: This subset contained listings with explicit price values. It served as the primary dataset for machine learning, as it provided the target variable (price) necessary for model training.

# Data Load

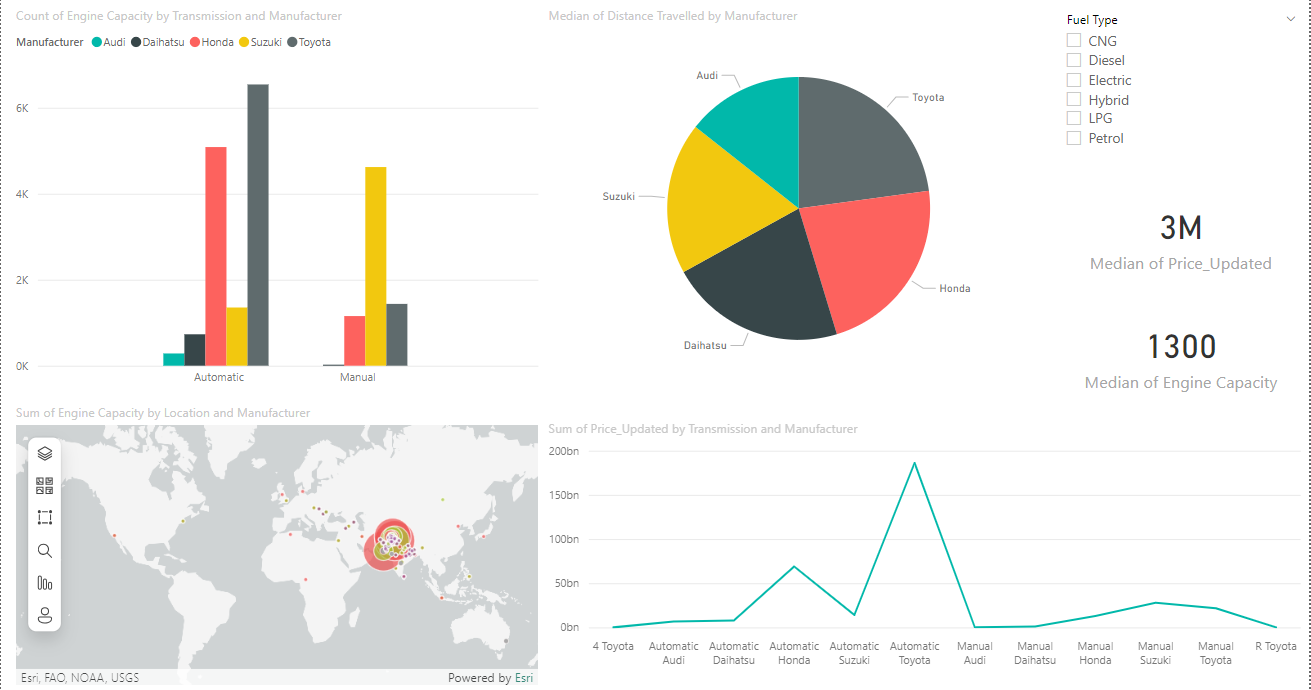
The transformed data was stored in MongoDB for easy access and scalability. We created two separate **collections** in MongoDB to store each subset:



* **data\_call collection**: This collection contains the listings where users are prompted to call for price details.
* **data\_update collection**: This collection stores listings with an explicit price, making it the primary dataset for our machine learning model.

|  |  |
| --- | --- |
|  |  |

Dashboard



# DATA FROM API

## Introduction

In today’s data-driven landscape, centralized data repositories are crucial for deriving meaningful insights. This project involves creating a data warehouse using the ETL (Extract, Transform, Load) process to consolidate COVID-19 statistics from an external API. By extracting, transforming, and loading this data into MongoDB, we facilitate easy access and analysis of pandemic-related information.

## Data Extraction

Data extraction is the first step in the ETL pipeline, where raw data is sourced from the COVID-19 API. This API provides comprehensive statistics on COVID-19, such as infection rates, recoveries, and testing figures by country.

Libraries Used: The requests library is used to make API calls, and json is utilized for parsing the API’s JSON response, allowing smooth data retrieval.



API URL: https://covid-193.p.rapidapi.com/statistics

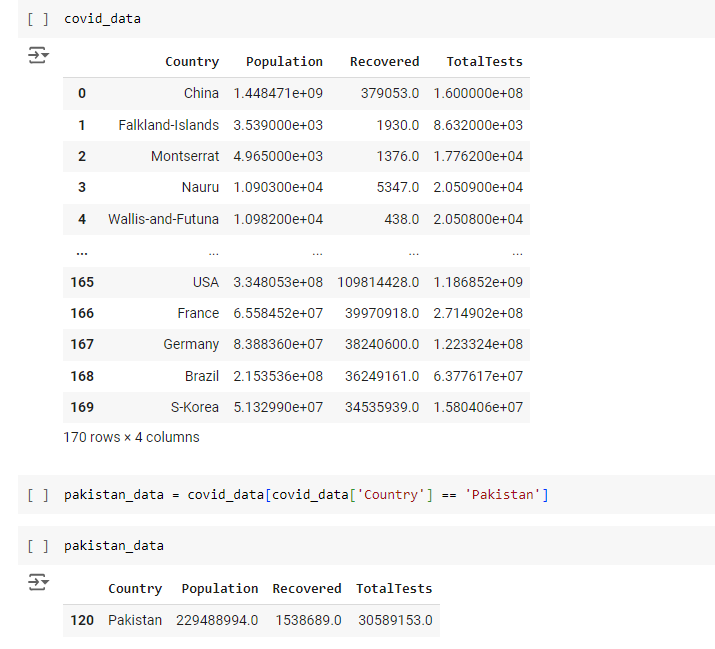
## Data Transformation

The transformation stage ensures that data is standardized, cleaned, and optimized for storage and querying. Key transformations include:

Data Cleaning: Using Pandas, missing values are handled, data types are corrected, and country names are standardized.

Schema Structuring: Data is organized into a structured format that aligns with MongoDB’s document-based structure.

Libraries Used: Pandas for data manipulation and NumPy for numerical operations to ensure efficient data processing.



## Data Load

In the final ETL phase, the transformed data is loaded into a MongoDB database. MongoDB’s document-oriented model is well-suited for storing JSON-like data, making it ideal for handling the COVID-19 statistics.

Database: Data is stored in a MongoDB collection, providing a flexible and scalable storage solution.

Batch Loading: Data is inserted into MongoDB in batches, ensuring a smooth loading process and efficient access.

Libraries Used: pymongo for connecting to MongoDB and inserting data into collections

