# VARDHAMAN COLLEGE OF ENGINEERING
# SHAMSHABAD–501218

**An Internship Project Report**

**On**

***" Youtube adview Prediction "***

*A project report submitted in partial fulfilment of the requirements for the award of the degree of* ***Bachelor of Engineering in Artificial Intelligence & Machine Learning*** *of Vardhaman College of Engineering, Shamshabad.*

Submitted by:

**NADIMPALLI MADANA KAILASH VARMA(22885A7304)**

Under the Guidance of :

**Syed Ahmeduddin**

**(Prof. , Dept. of AI&ML)**

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

**VARDHAMAN COLLEGE OF ENGINEERING**

((AUTONOMOUS)

Affiliated to JNTUH, Approved by AICTE, Accredited by NAAC with A++ Grade, ISO 9001:2015 Certified
Kacharam, Shamshabad, Hyderabad – 501218, Telangana, India

**2023-24**

**VARDHAMAN COLLEGE OF ENGINEERING,**

**Kacharam, Shamshabad, Hyderabd-501218**

**DEPARTMENT OF ARTIFICIAL INTELLIGENCE &**

**MACHINE LEARNING**

# CERTIFICATE

**This is to certify that the internship project work entitled " Youtube adview Prediction" has been successfully carried out by NADIMPALLI MADANA KAILASH VARMA (22885A7304), bonafide student of Vardhaman College of Engineering in partial fulfilment of the requirements for the award of degree in Bachelor of Engineering in Artificial Intelligence & Machine Learning of Vardhaman College of Engineering, Shamshabad during academic year 2023-2024 . It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the said Bachelor of Engineering degree.**

**Guide:**

| | |
|---|---|
| **Syed Ahmeduddin** | **Dr .Gagan Deep Arora** |
| **Prof. , Dept. of AI&ML** | **HOD, Dept. of AI&ML** |

**Examiners:**

**1.**

**2.**

# DECLARATION

**I the undersigned student of 5th semester Department of Artificial Intelligence & Machine Learning, Vardhaman College of Engineering, Shamshabad declare that my project work entitled "Youtube adview Prediction" is a bonafide work of me. My project is neither a copy nor by means a modification of any other engineering project.**

**I also declare that this project was not entitled for submission to any other university in the past and shall remain the only submission made and will not be submitted by us to any other university in the future.**

| Name | Roll Number | Signature |
|---|---|---|
| NADIMPALLI MADANA KAILASH VARMA | 22885A7304 | |

# ACKNOWLEDGEMENT

It gives me immense pleasure to present before you my project titled " Youtube adview Prediction " The joy and satisfaction that accompany the successful completion of any task would be incomplete without the mention of those who made it possible. I am glad to express our gratitude towards our prestigious institution VARDHAMAN COLLEGE OF ENGINEERING for providing us with utmost knowledge, encouragement and the maximum facilities in undertaking this project.

I wish to express sincere thanks to our respected chairman Dr. T.Vijender Reddy and beloved principal Dr. JVR Ravindra for all their support.

I express our deepest gratitude and special thanks to Dr. Gagan Deep Arora , H.O.D, Dept. Of Artificial Intelligence & Machine Learning, for all his guidance and encouragement.

I sincerely acknowledge the guidance and constant encouragement of our project guide, Syed Ahmeduddin, Prof. , Dept. Of Artificial Intelligence & Machine Learning.

I also would like to acknowledge all the teaching and non-teaching staff of the department for their kind cooperation during the project.

In the end, I sincerely thank my beloved family members and friends for their valuable suggestions, encouragement and unwavering support.

NADIMPALLI MADANA KAILASH VARMA (22885A7304)

# ABSTRACT

The contemporary digital landscape is dominated by the ever-expanding realm of online content, with YouTube standing as a prominent platform for video dissemination. This internship project delves into the intriguing domain of YouTube ad view prediction, leveraging advanced machine learning algorithms. Conducted at Internship Studio, this research explores the intricate patterns and factors influencing ad views on the popular video-sharing platform.

The primary objective of this project is to develop a robust predictive model that can accurately forecast the number of views an advertisement is likely to receive. By harnessing the power of machine learning algorithms, the study aims to unravel the underlying dynamics of viewer behavior, taking into account various features such as video content, user engagement, and temporal factors. The insights derived from this predictive model not only contribute to a deeper understanding of YouTube ad views but also hold practical implications for content creators and advertisers seeking to optimize their strategies.

The methodology employed involves the collection and analysis of a comprehensive dataset encompassing various attributes associated with YouTube videos and their corresponding ad views. Machine learning algorithms, including but not limited to regression and ensemble techniques, are employed to train and fine-tune the predictive model. The project's significance lies in its potential to provide content creators and advertisers with actionable insights, enabling them to enhance their content strategies and maximize the impact of their advertisements on the YouTube platform.

Through the execution of this internship project, valuable skills in data collection, preprocessing, and machine learning model development have been cultivated. The project's findings contribute to the broader discourse on content optimization in the digital landscape, emphasizing the practical applications of machine learning in predicting and enhancing YouTube ad views.

# CONTENTS

# LIST OF FIGURES

# Chapter 1

# INTRODUCTION

## 1.1 Introduction

In the ever-evolving landscape of online content consumption, YouTube stands as a colossal platform, shaping the digital experiences of millions worldwide. As the popularity of this video-sharing platform continues to soar, content creators and advertisers are presented with the challenge of optimizing their strategies to maximize viewer engagement. This internship project, conducted at Internship Studio, delves into the realm of YouTube adview prediction using advanced machine learning algorithms.

The project's inception is grounded in the recognition of the pivotal role YouTube advertisers play in paying content creators based on ad views and clicks for the products and services they promote. The project seeks to address the need for a robust predictive model that estimates ad views based on various metrics, such as the number of views, likes, dislikes, and comments, as well as additional factors like published date, duration, and category.

## 1.2 Aim and Objective

The primary aim of this internship project is to build a machine learning regression model capable of predicting YouTube ad view counts. The objective is to harness the power of regression algorithms and select the most effective model that accurately estimates the number of ad views based on the provided metrics. The project involves a multi-step process, including data exploration, cleaning, transformation, normalization, and the training of various regression models.

## 1.3 Existing Problem

The current landscape poses a challenge for advertisers who seek to estimate ad views based on a plethora of metrics. The diversity of data, ranging from video views to engagement metrics, requires a sophisticated approach to model development. This internship project addresses the existing gap by implementing regression models to predict ad views accurately, offering a solution to advertisers grappling with the complexity of YouTube's content ecosystem.

## 1.4 Problem Statement

The problem at hand involves training multiple regression models on a dataset containing metrics of about 15,000 youtube videos, with ad views as the target variable. The challenge is to refine and clean the dataset, transforming attributes into numerical values for optimal model performance. The goal is to identify the most suitable regression model for predicting ad views, balancing accuracy and generalization.

## 1.5 Proposed System/Solution

The proposed solution involves a systematic approach to machine learning, encompassing data visualization, cleaning, transformation, and the training of regression models. By leveraging linear regression, support vector regressor, decision tree regressor, random forest regressor, and an artificial neural network, the project aims to experiment and identify the most effective model. The ultimate outcome is a refined model capable of predicting youtube ad views, providing valuable insights for advertisers and content creators alike.

# Chapter 2

# LITREATURE SURVEY

## 2.1 Overview of YouTube Advertising Trends

Several studies have explored the trends and dynamics of YouTube advertising, emphasizing the significance of ad views as a key metric for measuring the effectiveness of online advertising campaigns.

## 2.2 Machine Learning in Digital Advertising

The integration of machine learning algorithms in digital advertising has garnered considerable attention. Research highlights the potential of machine learning models in predicting user engagement and ad views, offering advertisers valuable insights for strategy optimization.

## 2.3 Regression Models for Predictive Analytics

Existing literature delves into the application of regression models in predictive analytics for digital marketing. Studies explore the effectiveness of linear regression, Support Vector Regressor, and ensemble methods like Decision Tree Regressor and Random Forest Regressor in predicting various metrics, providing a foundation for their potential application in YouTube ad view prediction.

## 2.4 Temporal Factors and Viewer Behavior

Research has addressed the impact of temporal factors, such as video upload date, on viewer behavior and ad views. Understanding the temporal dynamics of YouTube content contributes to the refinement of predictive models by considering the influence of trends and seasonality.

Content Metrics and User Engagement:

Literature underscores the relevance of content metrics, including likes, dislikes, comments, and views, in gauging user engagement. The correlation between these metrics and ad views has been explored, providing insights into the factors that contribute to the success of YouTube advertising campaigns.

## 2.5 Challenges in Ad View Prediction

Studies have acknowledged the challenges associated with predicting ad views accurately. These challenges include data heterogeneity, noise in user engagement metrics, and the need for advanced data preprocessing techniques to enhance the performance of machine learning models.

## 2.6 Neural Networks in Predictive Modeling

Neural networks, specifically artificial neural networks, have been investigated for their potential in predictive modeling for online platforms. Research explores the application of neural networks in predicting user behavior, which can be extended to predict ad views on YouTube.

## 2.7 Model Evaluation and Generalization

The literature emphasizes the importance of robust model evaluation techniques and generalization in the context of predictive modeling for online platforms. Understanding how well a model generalizes to new data is crucial for ensuring its effectiveness in real-world scenarios.

## 2.8 Case Studies in YouTube Ad View Prediction

Some studies provide case studies or practical applications of predictive models for YouTube ad views. These case studies showcase the implementation of machine learning algorithms in real-world scenarios, highlighting successes, challenges, and lessons learned.

## 2.9 Future Directions and Emerging Trends:

Literature also explores future directions and emerging trends in the field of YouTube ad view prediction. This includes advancements in machine learning algorithms, the integration of deep learning techniques, and the incorporation of additional features for more accurate predictions.

# Chapter 3

# SYSTEM SPECIFICATION

## 3.1 Functional Requirements

The functional requirements outline the core capabilities and features that the YouTube Ad View Prediction system must possess. These include:

1. **Data Import:** The system should allow the importation of datasets containing YouTube video metrics.
2. **Data Visualization:** Implement functionality to visualize dataset attributes using heatmaps and plots for comprehensive data exploration.
3. **Data Cleaning:** Provide tools to clean the dataset, handling missing values and outliers for improved model training.
4. **Data Transformation**: Enable the transformation of non-numerical attributes into numerical values, ensuring compatibility with machine learning algorithms.
5. **Model Training:** Facilitate the training of various regression models, including linear regression, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and an artificial neural network.
6. **Model Evaluation:** Implement mechanisms for evaluating the trained models based on metrics such as error rates and generalization performance.
7. **Model Selection:** Allow the selection of the best-performing model for predicting YouTube ad views.
8. **Model Saving:** Provide functionality to save the selected model for future predictions on new data.

## 3.2 Non-Functional Requirements

Non-functional requirements specify the characteristics and qualities of the system that are not related to specific functionalities. These include:

- **Performance:** The system should handle large datasets efficiently and provide timely responses during data processing and model training.
- **Scalability**: The system should be scalable to accommodate future updates and improvements in model training techniques.
- **Usability:** Ensure an intuitive user interface with clear instructions for users to navigate through various functionalities.
- **Reliability:** The system should be reliable, providing consistent results across different datasets and scenarios.
- **Security:** Implement security measures to protect sensitive data and ensure the privacy of user information.

## 3.3 Hardware Requirements

The hardware requirements specify the necessary infrastructure for running the YouTube Ad View Prediction system:

- ✓ Processor: Quad-core processor (or higher) for efficient data processing and model training.
- ✓ Memory: 8GB RAM (or higher) to handle large datasets and neural network training.
- ✓ Storage: 500GB SSD (or higher) for storing datasets, models, and other project-related files.

## 3.4 Software Requirements

The software requirements outline the essential software components and tools needed for the system:

- ✓ **Programming Language:** Python 3.x for coding machine learning algorithms and system functionalities.
- ✓ **Libraries:** NumPy, Pandas, Scikit-learn, Matplotlib, Seaborn, TensorFlow, Keras for data manipulation, visualization, and machine learning.
- ✓ **Integrated Development Environment (IDE):** Jupyter Notebook or any preferred Python IDE for code development.
- ✓ **Version Control:** Git for version control and collaboration on the project.

## 3.5 Project Planning

Project planning involves outlining the steps and tasks necessary for the successful completion of the YouTube Ad View Prediction project:

- o **Data Import and Exploration:** Import the datasets, explore data attributes, and visualize using appropriate tools.
- o **Data Cleaning and Transformation:** Cleanse the dataset by handling missing values and transform non-numerical attributes. Prepare the data for model training.
- o **Model Training**: Implement regression models including linear regression, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and an artificial neural

network. Experiment with different configurations.

- **Model Evaluation:** Evaluate the trained models using appropriate metrics. Identify the best-performing model.
- **Model Deployment:** Save the selected model for future predictions on new data.
- **Documentation:** Document the entire project, including code, findings, and insights.
- **Presentation:** Prepare a comprehensive presentation summarizing the project for effective communication.

# Chapter 4

## SYSTEM DESIGN AND ARCHITECTURE

### 4.1 Software Design

The software design outlines the structure and organization of the YouTube Ad View Prediction system, focusing on the algorithms used for model development:

- **Python Scripts:** The system is implemented using Python scripts to leverage machine learning libraries and tools.
- **Algorithm Implementation:** The algorithms - Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and Artificial Neural Network (ANN) - are implemented as distinct modules within the system.
- **Modular Code:** Each algorithm is encapsulated in modular code for ease of maintenance and future updates.
- **Error Calculation:** A specific module calculates errors for each algorithm, allowing for comprehensive evaluation. The Error – Mean Absolute Error, Mean Squared Error, Root Mean Squared Error
- **Model Selection:** The system is designed to select the algorithm with the least error, considering Random Forest Regressor as the optimal model for YouTube ad view prediction.

### 4.2 Flowchart

The flowchart provides a visual representation of the logical flow within the YouTube Ad View Prediction system:

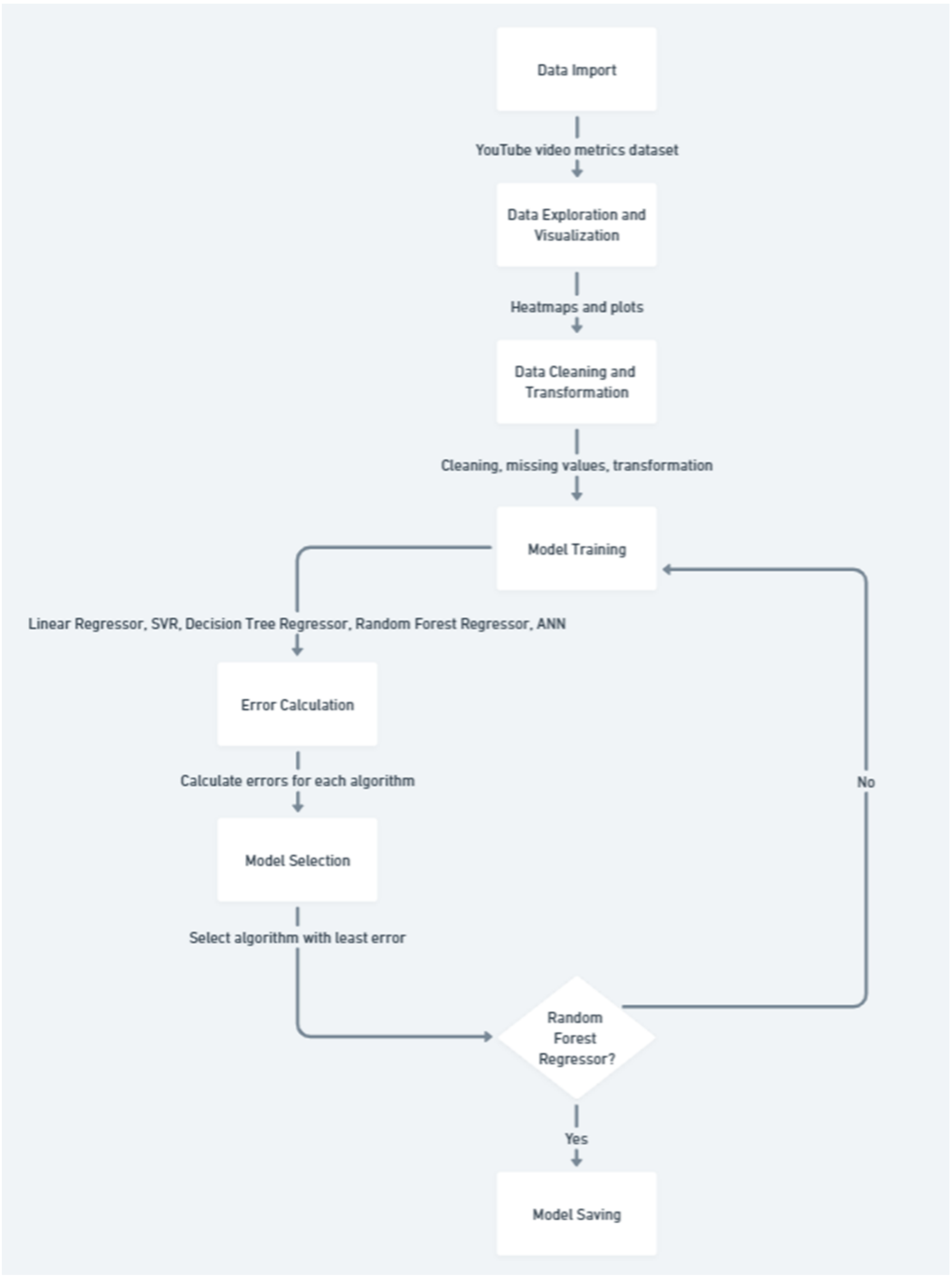

Data Import
YouTube video metrics dataset
Data Exploration and Visualization
Heatmaps and plots
Data Cleaning and Transformation
Cleaning, missing values, transformation
Model Training
Linear Regressor, SVR, Decision Tree Regressor, Random Forest Regressor, ANN
Error Calculation
Calculate errors for each algorithm
Model Selection
Select algorithm with least error
Random Forest Regressor?
No
Yes
Model Saving

**Fig 4.1: Flow chart of Algorithm**

- **Data Import**: The process begins with the importation of the YouTube video metrics dataset.

- **Data Exploration and Visualization:** The system explores and visualizes the dataset attributes using heatmaps and plots.

- **Data Cleaning and Transformation:** The dataset undergoes cleaning, handling missing values, and transformation of non-numerical attributes.

- **Model Training:** The system sequentially trains each algorithm - Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and ANN.

- **Error Calculation:** For each algorithm, errors are calculated, and the results are stored.

- Model Selection: The system selects the algorithm with the least error, designating Random Forest Regressor as the optimal model.

- **Model Saving**: The chosen model is saved for future predictions on new data.

## 4.3 Modules

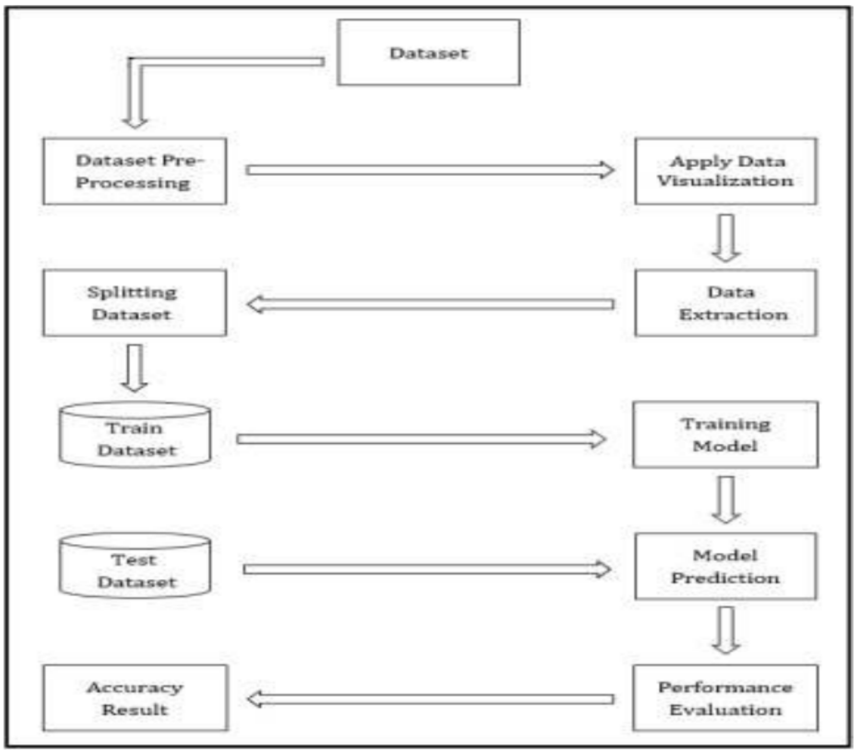


**Fig 4.2: Flow Diagram of Methodology**

The system is modular, with distinct modules corresponding to each key functionality:

1. **Data Import Module:** Responsible for importing the YouTube video metrics dataset.

2. **Data Exploration Module:** Performs data exploration and visualization using heatmaps and plots.

3. **Data Cleaning Module:** Handles data cleaning and transformation processes.

4. **Algorithm Modules:** Separate modules for each algorithm (Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, ANN).

5. **Error Calculation Module:** Calculates errors for each algorithm.

6. **Model Selection Module:** Selects the optimal model based on error calculations.

7. **Model Saving Module:** Saves the chosen model for future predictions.

## 4.4 Development Algorithm

The development algorithm encompasses the step-by-step process involved in building and evaluating the machine learning models:

1. **Data Preparation:** Import the dataset, explore, and visualize key attributes.

2. **Data Cleaning:** Handle missing values and transform non-numerical attributes.

3. **Model Training:** Sequentially train each algorithm (Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, ANN) using the prepared dataset.

4. **Error Calculation**: For each algorithm, calculate errors to assess their performance.

5. **Model Selection**: Identify the algorithm with the least error, designating Random Forest Regressor as the best model for YouTube ad view prediction.

# Chapter 5

# IMPLEMENTATION

### 5.1 Language Used for Implementation

The implementation of the YouTube Ad View Prediction project is carried out using the Python programming language. Python provides a rich ecosystem of libraries and tools for data manipulation, machine learning, and visualization, making it an ideal choice for this project. The primary libraries utilized include NumPy, Pandas, Scikit-learn, Matplotlib, Seaborn, TensorFlow, and Keras.

### 5.2 Terminologies

1. **Adviews:** The number of views a YouTube advertisement receives.
2. **Linear Regressor:** A regression algorithm used for predicting numerical values based on a linear relationship between input features and the target variable.
3. **Support Vector Regressor:** A regression algorithm that uses support vector machines to predict numerical values.
4. **Decision Tree Regressor:** A regression algorithm based on decision tree structures that predict numerical values.
5. **Random Forest Regressor:** An ensemble learning algorithm that combines multiple decision trees to improve prediction accuracy.
6. **Artificial Neural Network (ANN):** A machine learning model inspired by the structure and function of the human brain, used for complex pattern recognition tasks.

## 5.3 Functionality of Algorithms

Each algorithm contributes uniquely to the task of predicting YouTube ad views, leveraging distinct approaches and methodologies:

**- Linear Regressor:** This algorithm establishes a linear relationship between input features and ad views. By identifying the coefficients that minimize the difference between predicted and actual values, the linear regressor provides a straightforward model for predicting the number of ad views based on various metrics.

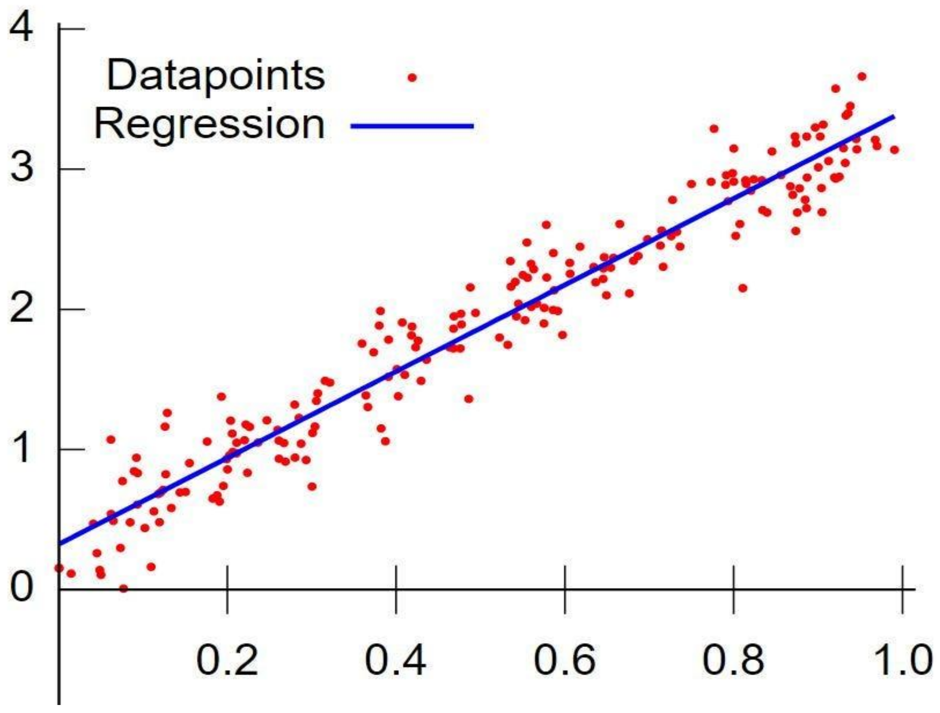


**Fig 5.1: Linear Regression**

- **Support Vector Regressor:** Utilizing support vector machines, the support vector regressor aims to find a hyperplane that best represents the relationship between input features and ad views. By mapping data into higher-dimensional spaces, it identifies the optimal hyperplane that maximizes the margin between predicted values and actual ad views.

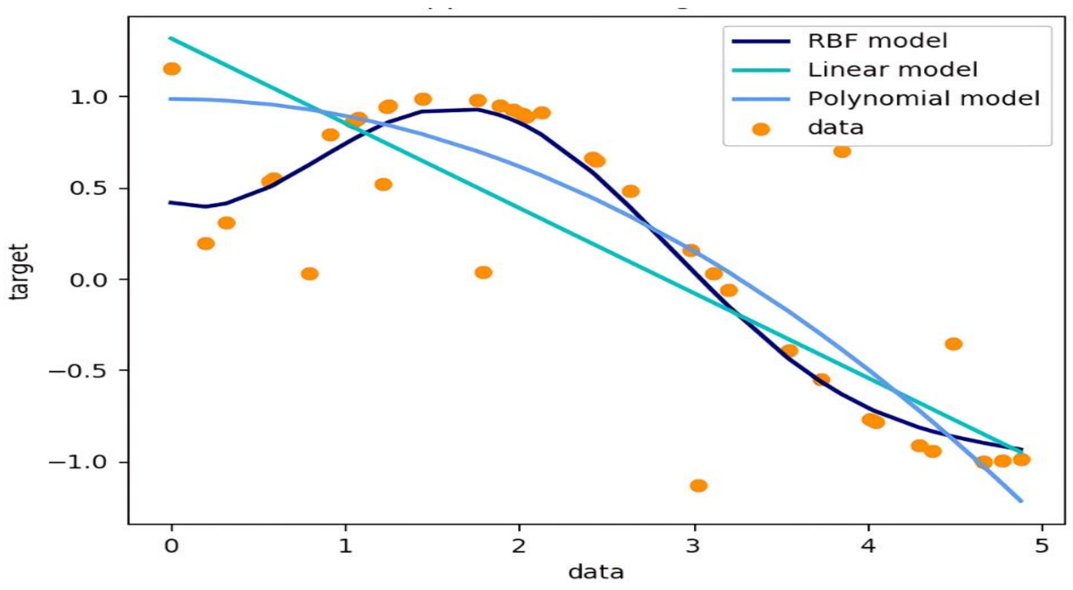


**Fig 5.2: Support Vector Regression**

**- Decision Tree Regressor:** Constructing a decision tree based on attribute values, the decision tree regressor breaks down the data into hierarchical structures. Each decision node represents a feature, and the leaves contain predicted ad view values. This algorithm is capable of capturing non-linear relationships and interactions between different metrics.

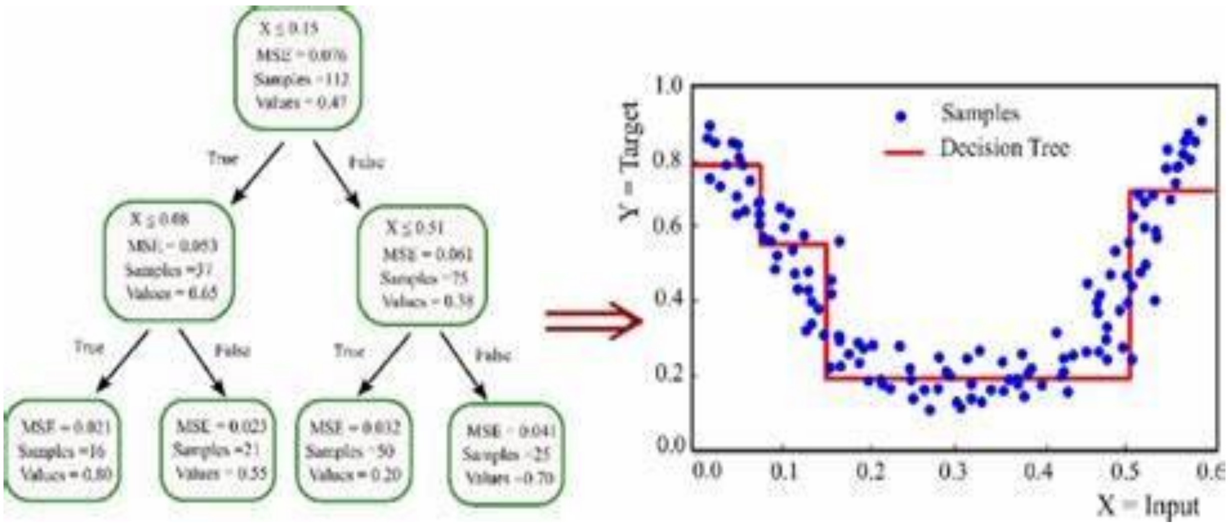


**Fig 5.3: Decision Tree Regression**

- **Random Forest Regressor:** Leveraging the power of ensemble learning, the random forest regressor combines multiple decision trees to enhance prediction accuracy. By aggregating the predictions of individual trees, this algorithm mitigates overfitting and generalizes well to new data, making it a robust choice for predicting YouTube ad views.

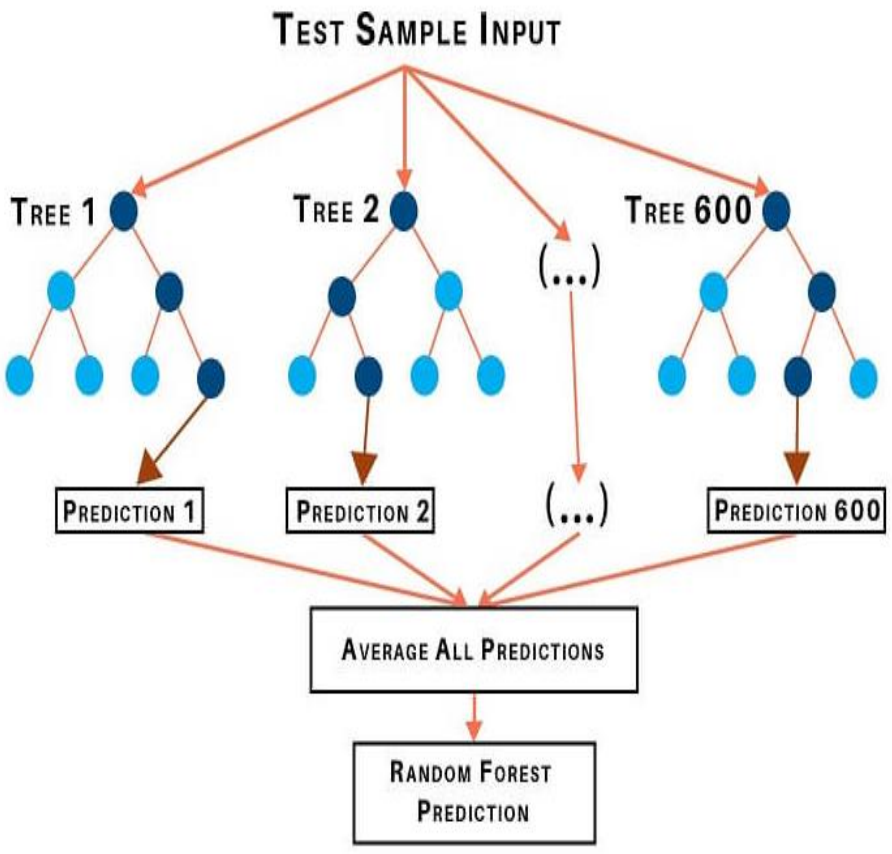


**Fig 5.4: Random Forest Regression**

- **Artificial Neural Network (ANN):** Inspired by the structure and function of the human brain, an artificial neural network consists of interconnected nodes organized into layers. By learning complex patterns and relationships in the data through forward and backward propagation, an ANN can capture intricate non-linear dependencies between input features and predict ad views with high accuracy.

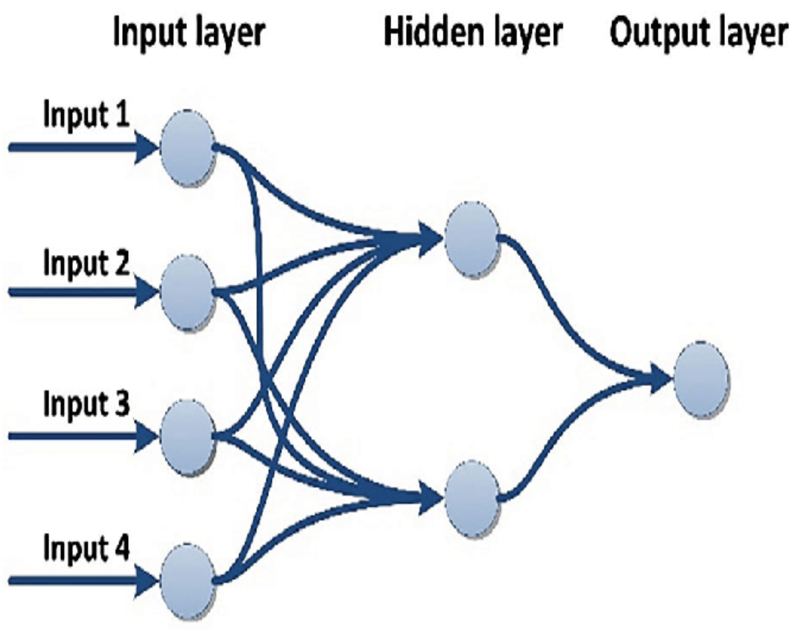


**Fig 5.5: Artificial Neural Network**

## 5.4 Methodology

The implementation of the YouTube Ad View Prediction system follows a systematic methodology to ensure accurate model development and selection:

- Data Preparation involves importing the YouTube video metrics dataset, exploring key attributes through visualization using heatmaps and plots. This initial phase lays the foundation for understanding the dataset's structure and characteristics.

- Data Cleaning and Transformation follow, addressing missing values and transforming non-numerical attributes to ensure compatibility with machine learning algorithms. This step aims to enhance the dataset's quality and prepare it for effective model training.

- Model Training encompasses the core of the methodology, where each algorithm—Linear

Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and ANN—is trained using the prepared dataset. This step involves adjusting model parameters to learn from the data and make predictions.

- Error Calculation is performed next, evaluating each algorithm's performance by calculating errors such as mean squared error or mean absolute error. This quantitative assessment provides insights into how well each model predicts YouTube ad views.

- Model Selection involves identifying the algorithm with the least error, designating Random Forest Regressor as the best model for YouTube ad view prediction. This step ensures the chosen model offers optimal accuracy and generalization capabilities.

- Model Saving concludes the methodology, where the selected Random Forest Regressor model is saved for future predictions on new data. This ensures the preservation of the trained model for practical applications beyond the initial dataset.

- The step-by-step approach ensures a thorough and comprehensive implementation, facilitating the creation of an effective YouTube Ad View Prediction system.

# Chapter 6

# TESTING

## 6.1 Testing

Testing is a critical phase in the development of the YouTube Ad View Prediction system to ensure the accuracy and reliability of the implemented models. The following testing procedures are conducted:

1. **Data Splitting:** The dataset is split into training, validation, and test sets to assess the models' performance on unseen data. This ensures the generalization of the models.

2. **Model Evaluation:** Each algorithm, including Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and ANN, is evaluated using the test dataset. The saved Random Forest Regressor model is also loaded and evaluated.

3. **Error Analysis:** Errors, such as mean squared error or mean absolute error, are calculated for each model using the test data. This analysis provides insights into the accuracy and predictive capabilities of the models.

4. **Prediction on New Data:** The saved Random Forest Regressor model is used to predict ad views on new data from the special test dataset. The predictions are compared with the actual values stored in the .h5 file format.

## 6.2 Cost Estimation and Project Planning

Cost estimation and project planning are integral components to ensure the successful completion of the YouTube Ad View Prediction project:

**Cost Estimation:** The cost estimation involves assessing the resources utilized during the project, including hardware, software, and personnel costs. It encompasses expenses related to data storage, computational resources, software licenses, and the time invested by team members.

**Project Planning:** The project planning phase outlines the schedule, milestones, and deliverables. It involves creating a timeline for each phase of the project, including data preparation, model development, testing, and documentation. The planning also considers potential risks and mitigation strategies to address challenges that may arise during the project.

**Resource Allocation:** The allocation of resources, both human and technical, is a crucial aspect of project planning. It involves assigning tasks to team members based on their expertise and availability, ensuring a balanced and efficient workflow.

**Timeline and Milestones:** A detailed timeline is established with specific milestones for each phase of the project. Milestones serve as checkpoints to monitor progress and ensure that the project stays on track.

**Risk Management:** Identifying potential risks, such as data quality issues, algorithm performance challenges, or resource constraints, is part of the project planning process. Developing mitigation strategies and contingency plans helps address unforeseen circumstances.

**Documentation:** Comprehensive documentation, including project plans, coding standards, and user manuals, is essential. This facilitates knowledge transfer, future maintenance, and collaboration among team members.

The combined efforts of rigorous testing and effective cost estimation and project planning contribute to the successful implementation and deployment of the YouTube Ad View Prediction system.

# Chapter 7

# SOURCE CODE

## 7.1 Importing the datasets and Libraries

```
# Import the datasets and librarieS
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense

# Loading the dataset
data = pd.read_csv('youtube_adview.csv')

# For Checking the shape and datatypes of the dataset
print(data.shape)
print(data.dtypes)
```

## 7.2 Visualizing data using plots

```
# Visualise the dataset using plotting using heatmaps and plots.
f , ax = plt.subplots(figsize = (10,8))
corr = data.corr()
sns.heatmap(corr,mask = np.zeros_like(corr,dtype = np.bool),cmap = sns.diverging_palette(220,20,as_cmap =
True),square = True,ax = ax,annot = True)
plt.show()

# Assigning each character a number
category = {'A':1,'B':2,'C':3,'D':4,'E':5,'F':6,'G':7,'H':8}
data['category'] = data['category'].map(category)
data['category']

data = data[data.views != 'F']
data = data[data.likes != 'F']
data = data[data.dislikes != 'F']
data = data[data.comment ! ='F']
```

## 7.3 Cleaning data by removing missing values

```
#Clean the dataset by removing missing values and other things.

# Check for missing values
print(data.isnull().sum())

# Drop rows with missing values
data = data.dropna()

# Remove any duplicate rows if any
data = data.drop_duplicates()

data['views'] = pd.to_numeric(data['views'])
data['likes'] = pd.to_numeric(data['likes'])
data['dislikes'] = pd.to_numeric(data['dislikes'])
data['comment'] = pd.to_numeric(data['comment'])
data['adview'] = pd.to_numeric(data['adview'])
```

## 7.4 Transforming attributes

```
#Transform attributes into numerical values and other necessary transformations

data['duration'] = LabelEncoder().fit_transform(data['duration'])
data['vidid'] = LabelEncoder().fit_transform(data['vidid'])
data['published'] = LabelEncoder().fit_transform(data['published'])

plt.hist(data['category'])
plt.show

plt.plot(data['adview'])
plt.show

data = data[data['adview']<2000000]
plt.plot(data['adview'])
plt.show
```

## 7.5 Split data into training and test set

```
#  Normalise your data and split the data into training, validation and test set in the appropriate ratio.

# Select features and target
X = data[['views', 'likes', 'dislikes', 'comment', 'published','duration','category']]
y = data['adview']

# Normalize the feature data
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# Split the data into training (60%), validation (20%), and test (20%) sets
```

```
X_train, X_temp, y_train, y_temp = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.2, random_state=42)

X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)

#Define function to print Error
from sklearn import metrics
def print_Error(X_test,y_test,model_name):
  prediction = model_name.predict(X_test)
  print("Mean Absolute error of ",model_name,metrics.mean_absolute_error(y_test,prediction))
  print("Mean Squared error of ",model_name,metrics.mean_squared_error(y_test,prediction))
  print("Root Mean Squared error of ",model_name,np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

## 7.6 Use Linear regression, Support Vector Regression for training

```
#Use linear regression, Support Vector Regressor for training and get errors.

# Initialize the Linear Regression model
linear_reg = LinearRegression()
# Train the model
linear_reg.fit(X_train, y_train)
# Predict on validation set
y_pred_val_linear = linear_reg.predict(X_val)
# Calculate error
error_linear = mean_squared_error(y_val, y_pred_val_linear)
print(f'Linear Regression MSE: {error_linear}')
print_Error(X_test,y_test,linear_reg)
print("\n")

# Initialize the Support Vector Regressor model
svr_reg = SVR()
# Train the model
svr_reg.fit(X_train, y_train)
# Predict on validation set
y_pred_val_svr = svr_reg.predict(X_val)
# Calculate error
error_svr = mean_squared_error(y_val, y_pred_val_svr)
print(f'SVR MSE: {error_svr}')
print_Error(X_test,y_test,svr_reg)
```

## 7.7 Use Decision Tree Regressor, Random Forest Regressor for training

```
# Use Decision Tree Regressor and Random Forest Regressors.

# Initialize the Decision Tree Regressor model
dt_reg = DecisionTreeRegressor()
# Train the model
dt_reg.fit(X_train, y_train)
# Predict on validation set
y_pred_val_dt = dt_reg.predict(X_val)
```

```
# Calculate error
error_dt = mean_squared_error(y_val, y_pred_val_dt)
print(f'Decision Tree MSE: {error_dt}')
print_Error(X_test,y_test,dt_reg)
print("\n")

# Initialize the Random Forest Regressor model
rf_reg = RandomForestRegressor(n_estimators = 200 ,max_depth = 25,min_samples_split = 15,min_samples_leaf =  5)
#rf_reg = RandomForestRegressor()
# Train the model
rf_reg.fit(X_train, y_train)
# Predict on validation set
y_pred_val_rf = rf_reg.predict(X_val)
# Calculate error
error_rf = mean_squared_error(y_val, y_pred_val_rf)
print(f'Random Forest MSE: {error_rf}')
print_Error(X_test,y_test,rf_reg)
```

## 7.8 Build an Artificial Neural Network for training

#Build an artificial neural network and train it with different layers and hyperparameters.

```
# Initialize the ANN model
ann = Sequential()
ann.add(Dense(units=128, activation='relu', input_dim=X_train.shape[1]))
ann.add(Dense(units=64, activation='relu'))
ann.add(Dense(units=1, activation='linear'))

# Compile the ANN
ann.compile(optimizer='adam', loss='mean_squared_error')
# Train the ANN model
ann.fit(X_train, y_train, batch_size=32, epochs=100, validation_data=(X_val, y_val))

# Predict on validation set
y_pred_val_ann = ann.predict(X_val)
# Calculate error
error_ann = mean_squared_error(y_val, y_pred_val_ann)
print(f'ANN MSE: {error_ann}')
```

## 7.9 Pick the best model based on error

#Pick the best model based on error as well as generalisation.

```
# Compare the errors of all models
model_errors = {
    'Linear Regression': error_linear,
    'SVR': error_svr,
    'Decision Tree': error_dt,
    'Random Forest': error_rf,
    'ANN': error_ann
}
```

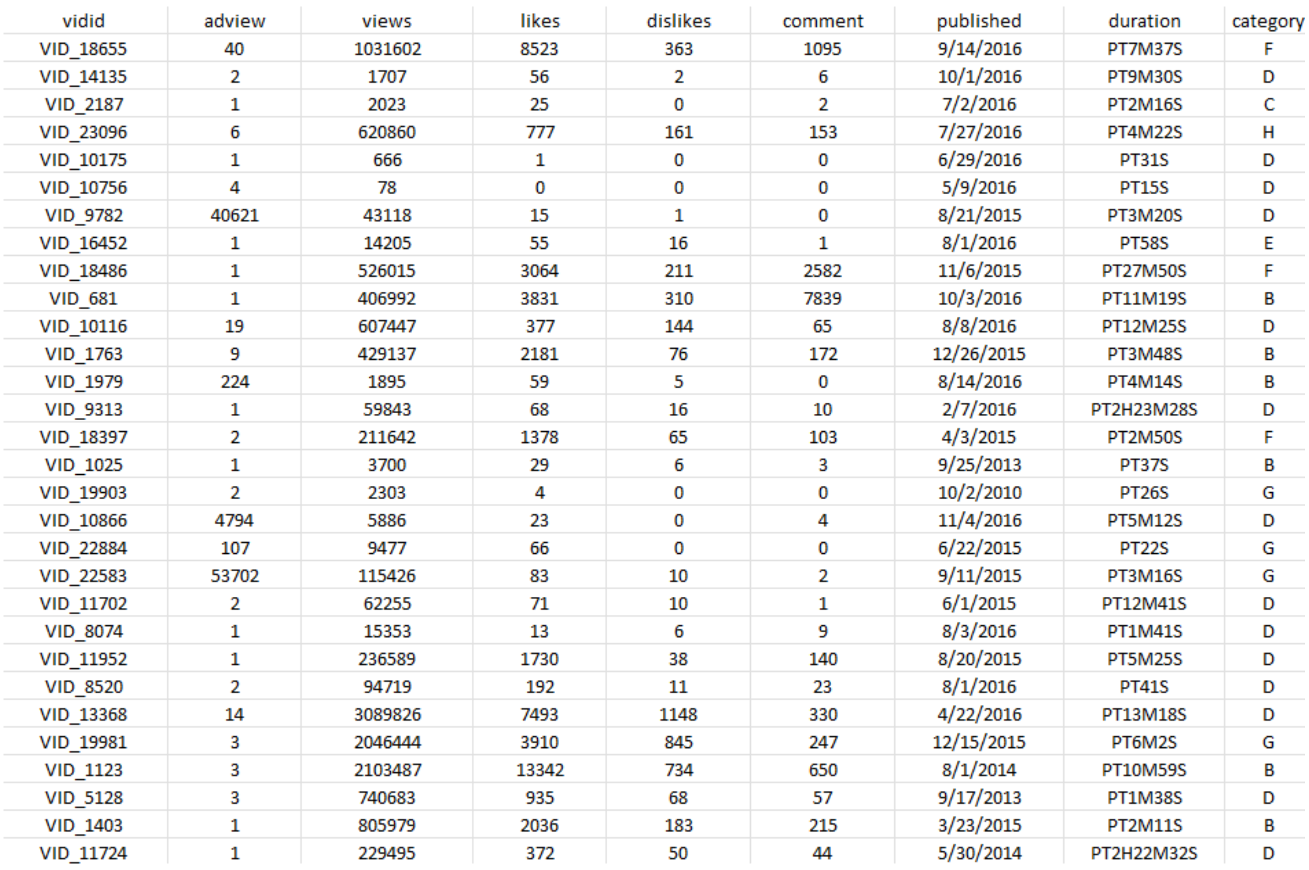
```
best_model = min(model_errors, key=model_errors.get)
print(f'The best model is: {best_model}')

# The test set predictions can now be used for further analysis or creating a submission file, etc.
```

## 7.10 Save model and predict on the test set

```
# Save your model and predict on the test set.

# Assuming the best model is the Random Forest based on the previous step
best_model = rf_reg
# Save the model
import joblib
joblib.dump(best_model, 'best_youtube_adview_predictor.pkl')

# Load model and predict on the test set (for demonstration)
loaded_model = joblib.load('best_youtube_adview_predictor.pkl')
test_predictions = loaded_model.predict(X_test)
print(test_predictions)

ann.save("ann_youtube_adview.h5")
```

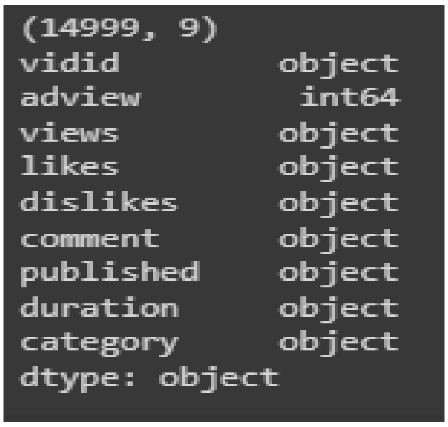
# Chapter 8

## SNAPSHOTS



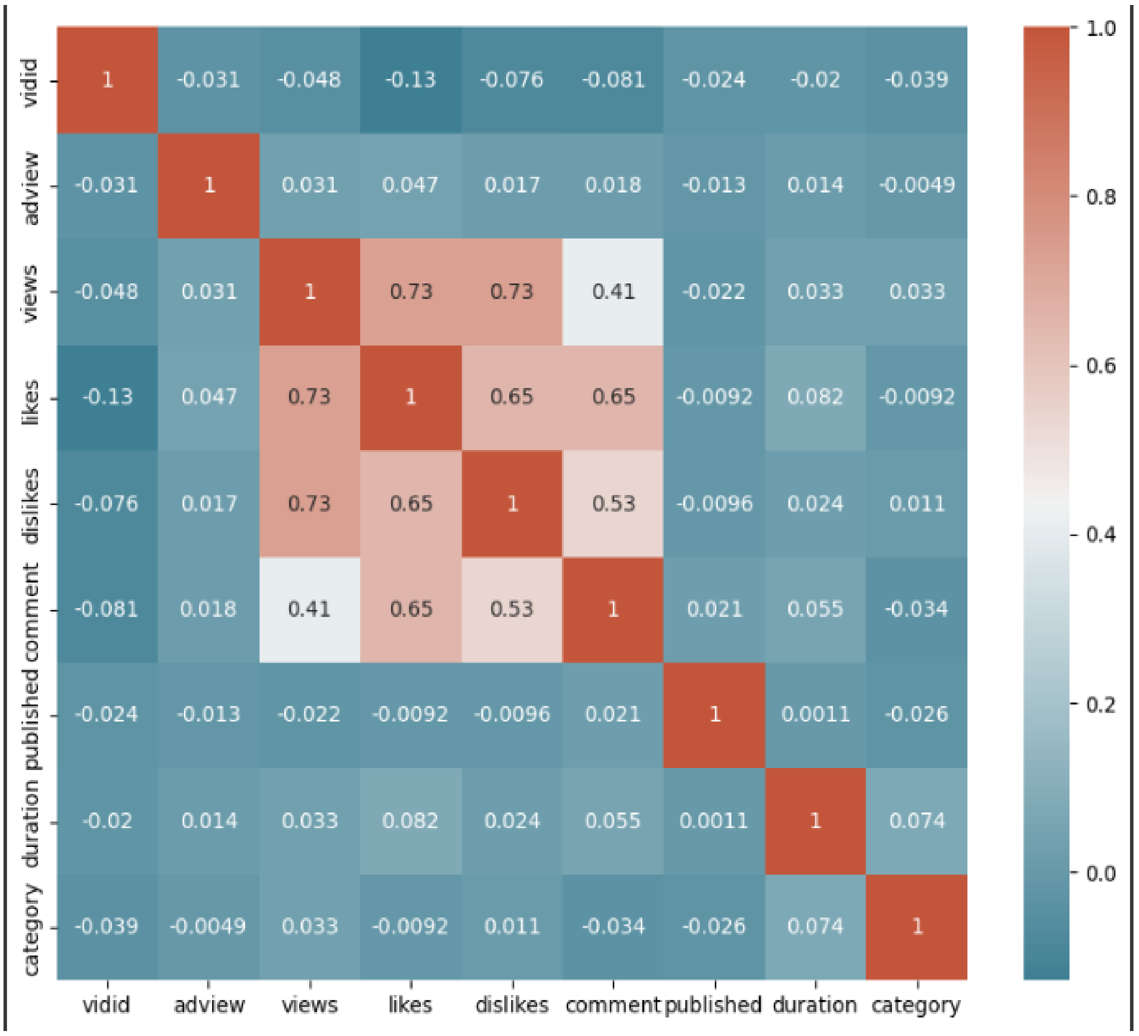
| vidid | adview | views | likes | dislikes | comment | published | duration | category |
|---|---|---|---|---|---|---|---|---|
| VID_18655 | 40 | 1031602 | 8523 | 363 | 1095 | 9/14/2016 | PT7M37S | F |
| VID_14135 | 2 | 1707 | 56 | 2 | 6 | 10/1/2016 | PT9M30S | D |
| VID_2187 | 1 | 2023 | 25 | 0 | 2 | 7/2/2016 | PT2M16S | C |
| VID_23096 | 6 | 620860 | 777 | 161 | 153 | 7/27/2016 | PT4M22S | H |
| VID_10175 | 1 | 666 | 1 | 0 | 0 | 6/29/2016 | PT31S | D |
| VID_10756 | 4 | 78 | 0 | 0 | 0 | 5/9/2016 | PT15S | D |
| VID_9782 | 40621 | 43118 | 15 | 1 | 0 | 8/21/2015 | PT3M20S | D |
| VID_16452 | 1 | 14205 | 55 | 16 | 1 | 8/1/2016 | PT58S | E |
| VID_18486 | 1 | 526015 | 3064 | 211 | 2582 | 11/6/2015 | PT27M50S | F |
| VID_681 | 1 | 406992 | 3831 | 310 | 7839 | 10/3/2016 | PT11M19S | B |
| VID_10116 | 19 | 607447 | 377 | 144 | 65 | 8/8/2016 | PT12M25S | D |
| VID_1763 | 9 | 429137 | 2181 | 76 | 172 | 12/26/2015 | PT3M48S | B |
| VID_1979 | 224 | 1895 | 59 | 5 | 0 | 8/14/2016 | PT4M14S | B |
| VID_9313 | 1 | 59843 | 68 | 16 | 10 | 2/7/2016 | PT2H23M28S | D |
| VID_18397 | 2 | 211642 | 1378 | 65 | 103 | 4/3/2015 | PT2M50S | F |
| VID_1025 | 1 | 3700 | 29 | 6 | 3 | 9/25/2013 | PT37S | B |
| VID_19903 | 2 | 2303 | 4 | 0 | 0 | 10/2/2010 | PT26S | G |
| VID_10866 | 4794 | 5886 | 23 | 0 | 4 | 11/4/2016 | PT5M12S | D |
| VID_22884 | 107 | 9477 | 66 | 0 | 0 | 6/22/2015 | PT22S | G |
| VID_22583 | 53702 | 115426 | 83 | 10 | 2 | 9/11/2015 | PT3M16S | G |
| VID_11702 | 2 | 62255 | 71 | 10 | 1 | 6/1/2015 | PT12M41S | D |
| VID_8074 | 1 | 15353 | 13 | 6 | 9 | 8/3/2016 | PT1M41S | D |
| VID_11952 | 1 | 236589 | 1730 | 38 | 140 | 8/20/2015 | PT5M25S | D |
| VID_8520 | 2 | 94719 | 192 | 11 | 23 | 8/1/2016 | PT41S | D |
| VID_13368 | 14 | 3089826 | 7493 | 1148 | 330 | 4/22/2016 | PT13M18S | D |
| VID_19981 | 3 | 2046444 | 3910 | 845 | 247 | 12/15/2015 | PT6M2S | G |
| VID_1123 | 3 | 2103487 | 13342 | 734 | 650 | 8/1/2014 | PT10M59S | B |
| VID_5128 | 3 | 740683 | 935 | 68 | 57 | 9/17/2013 | PT1M38S | D |
| VID_1403 | 1 | 805979 | 2036 | 183 | 215 | 3/23/2015 | PT2M11S | B |
| VID_11724 | 1 | 229495 | 372 | 50 | 44 | 5/30/2014 | PT2H22M32S | D |

**Fig 8.1: Sample dataset**



```
(14999, 9)
vidid          object
adview          int64
views          object
likes          object
dislikes       object
comment        object
published      object
duration       object
category       object
dtype: object
```

**Fig 8.2: Features and their data type in Dataset**



| | vidid | adview | views | likes | dislikes | comment | published | duration | category |
|---|---|---|---|---|---|---|---|---|---|
| vidid | 1 | -0.031 | -0.048 | -0.13 | -0.076 | -0.081 | -0.024 | -0.02 | -0.039 |
| adview | -0.031 | 1 | 0.031 | 0.047 | 0.017 | 0.018 | -0.013 | 0.014 | -0.0049 |
| views | -0.048 | 0.031 | 1 | 0.73 | 0.73 | 0.41 | -0.022 | 0.033 | 0.033 |
| likes | -0.13 | 0.047 | 0.73 | 1 | 0.65 | 0.65 | -0.0092 | 0.082 | -0.0092 |
| dislikes | -0.076 | 0.017 | 0.73 | 0.65 | 1 | 0.53 | -0.0096 | 0.024 | 0.011 |
| comment | -0.081 | 0.018 | 0.41 | 0.65 | 0.53 | 1 | 0.021 | 0.055 | -0.034 |
| published | -0.024 | -0.013 | -0.022 | -0.0092 | -0.0096 | 0.021 | 1 | 0.0011 | -0.026 |
| duration | -0.02 | 0.014 | 0.033 | 0.082 | 0.024 | 0.055 | 0.0011 | 1 | 0.074 |
| category | -0.039 | -0.0049 | 0.033 | -0.0092 | 0.011 | -0.034 | -0.026 | 0.074 | 1 |

**Fig 8.3: Correlation matrix and Heat Map**

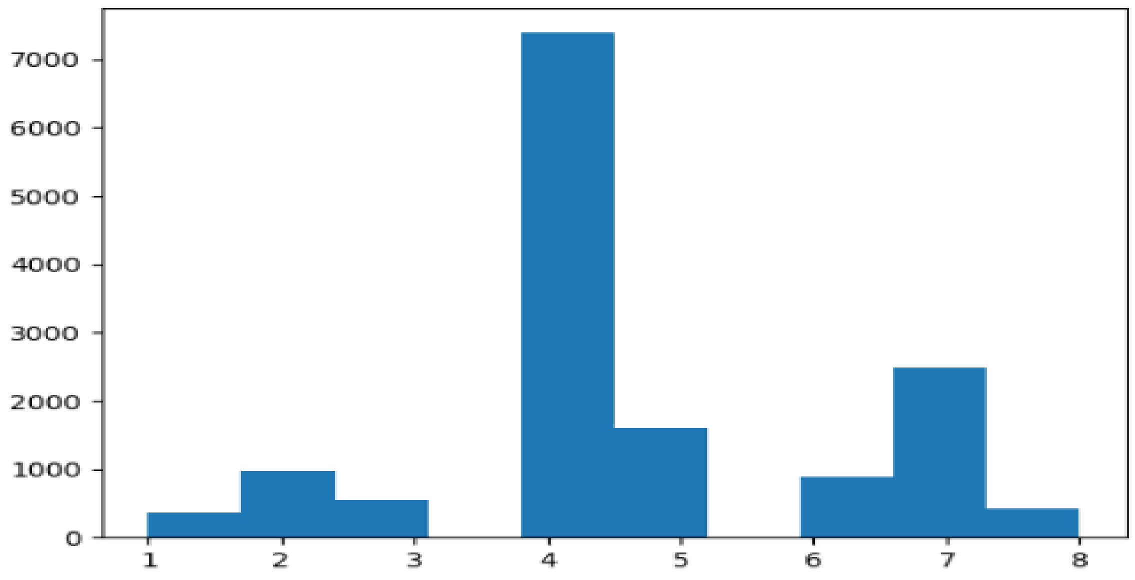


**Fig 8.4: Bar plot for feature "category"**

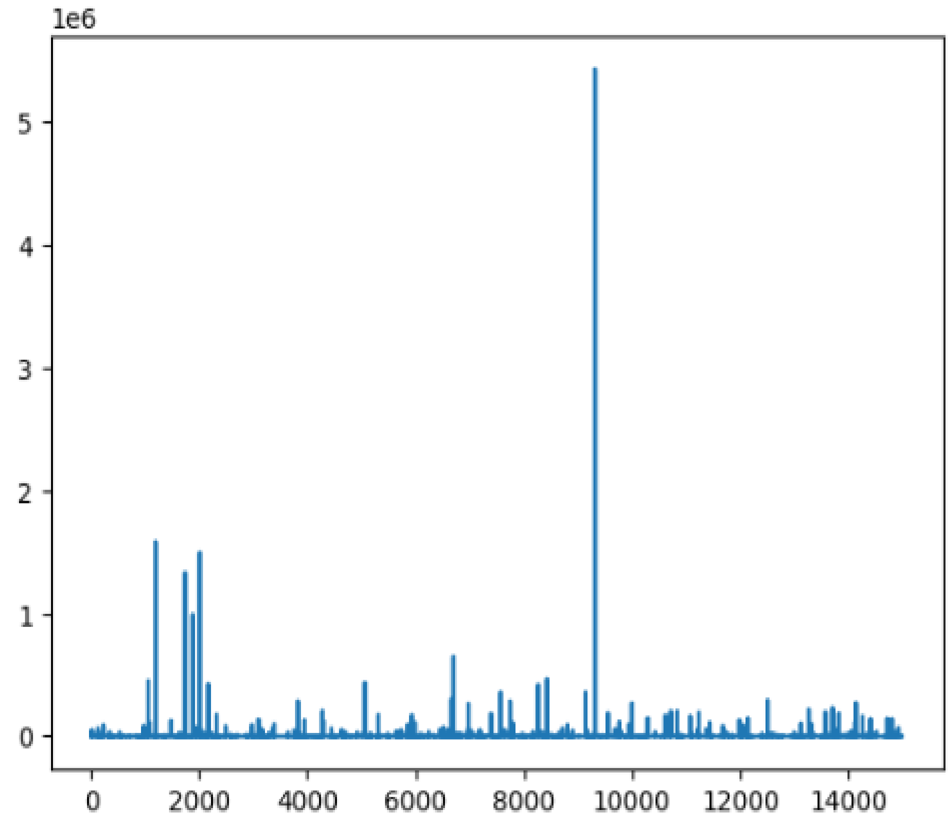


**Fig 8.5: Bar plot for feature "adview"**

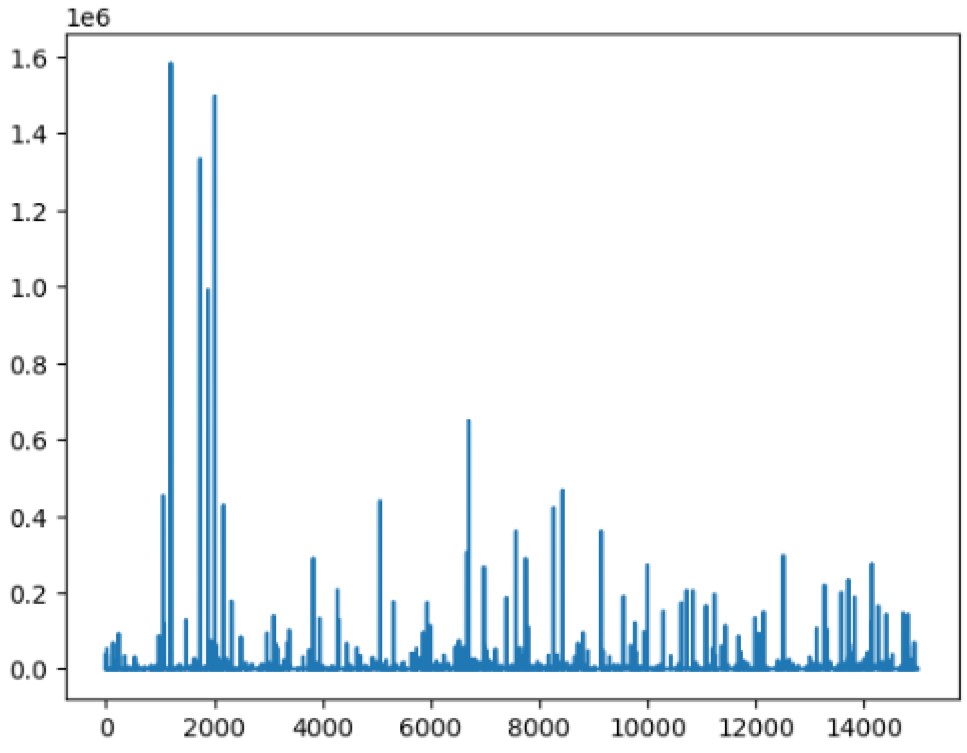
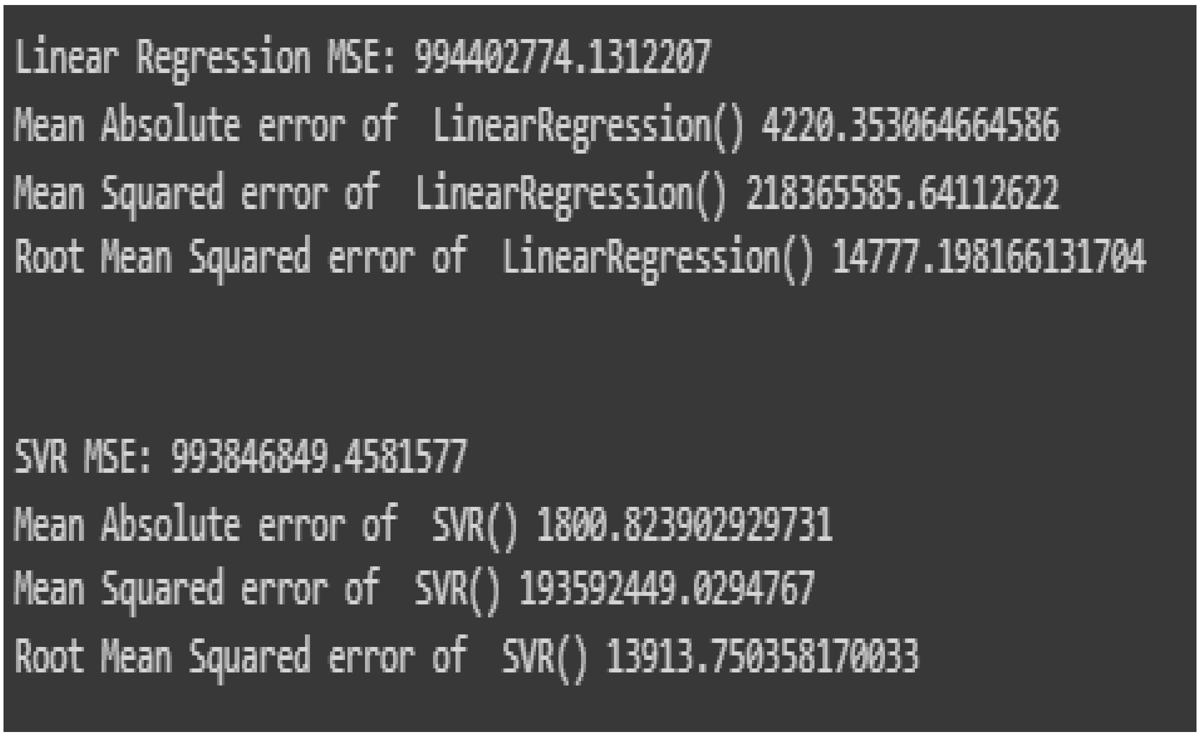
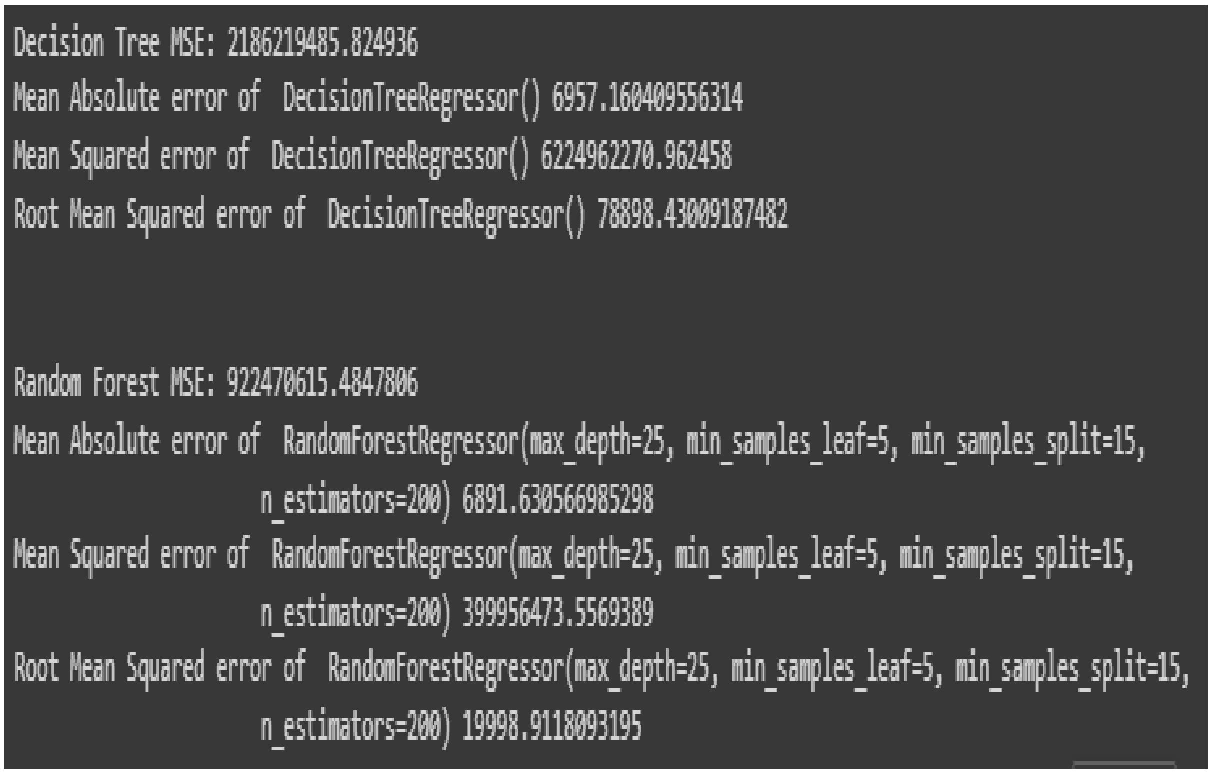


**Fig 8.6: Bar plot for feature "adview" after cleaning and removing outliers**



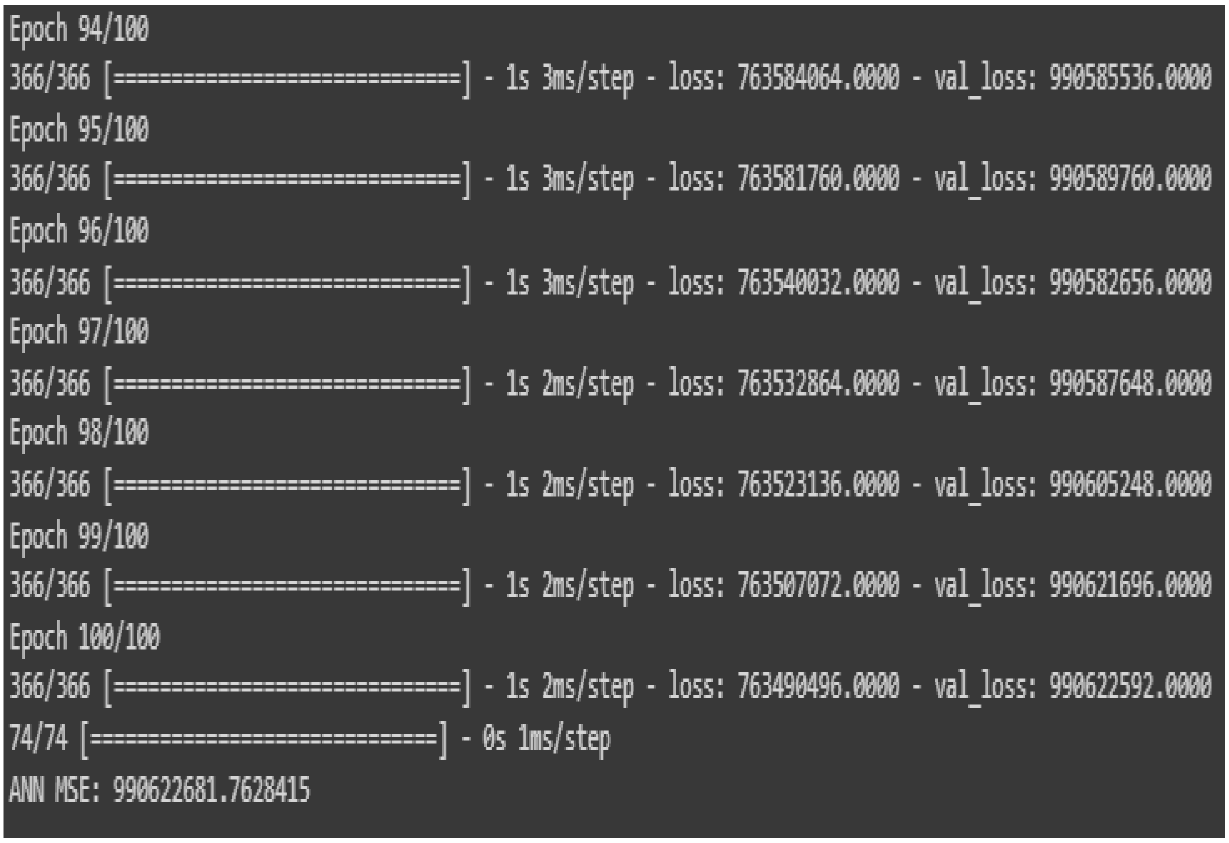
```
Linear Regression MSE: 994402774.1312207
Mean Absolute error of  LinearRegression() 4220.353064664586
Mean Squared error of  LinearRegression() 218365585.64112622
Root Mean Squared error of  LinearRegression() 14777.198166131704



SVR MSE: 993846849.4581577
Mean Absolute error of  SVR() 1800.823902929731
Mean Squared error of  SVR() 193592449.0294767
Root Mean Squared error of  SVR() 13913.750358170033
```

**Fig 8.7: Errors obtained for Linear Regression and Support Vector Regression**



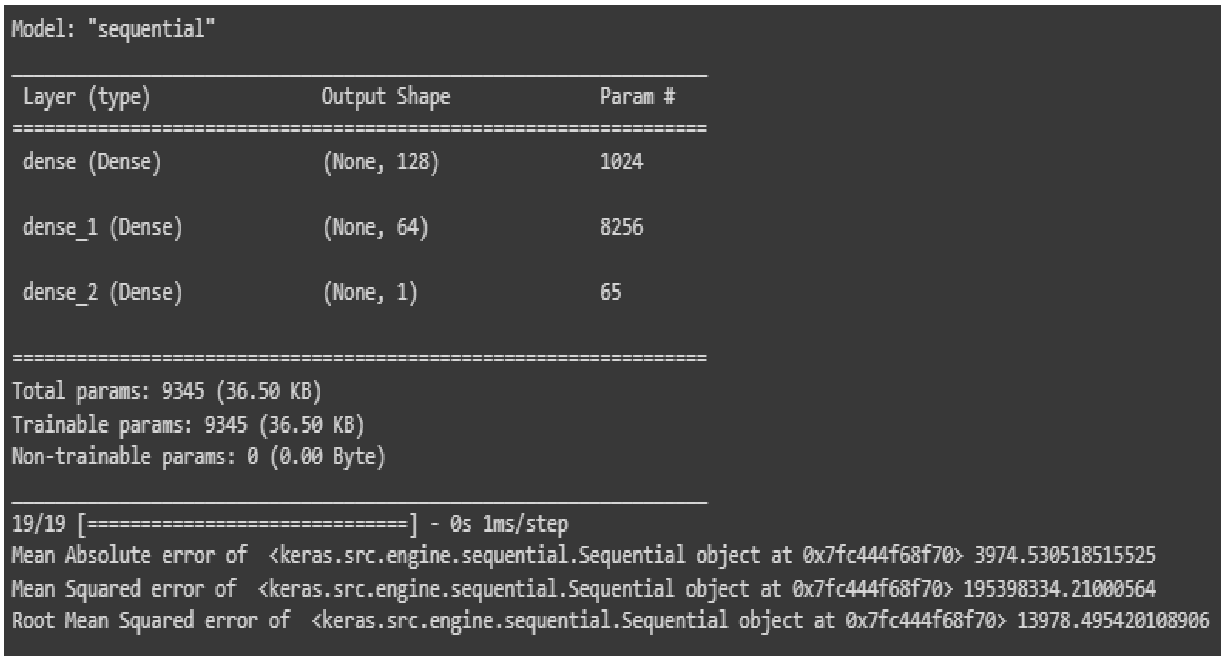
```
Decision Tree MSE: 2186219485.824936
Mean Absolute error of  DecisionTreeRegressor() 6957.160409556314
Mean Squared error of  DecisionTreeRegressor() 6224962270.962458
Root Mean Squared error of  DecisionTreeRegressor() 78898.43009187482



Random Forest MSE: 922470615.4847806
Mean Absolute error of  RandomForestRegressor(max_depth=25, min_samples_leaf=5, min_samples_split=15,
                      n_estimators=200) 6891.630566985298
Mean Squared error of  RandomForestRegressor(max_depth=25, min_samples_leaf=5, min_samples_split=15,
                      n_estimators=200) 399956473.5569389
Root Mean Squared error of  RandomForestRegressor(max_depth=25, min_samples_leaf=5, min_samples_split=15,
                      n_estimators=200) 19998.9118093195
```

**Fig 8.8:Errors obtained for Decision Tree Regressor and Random Forest Regressor**



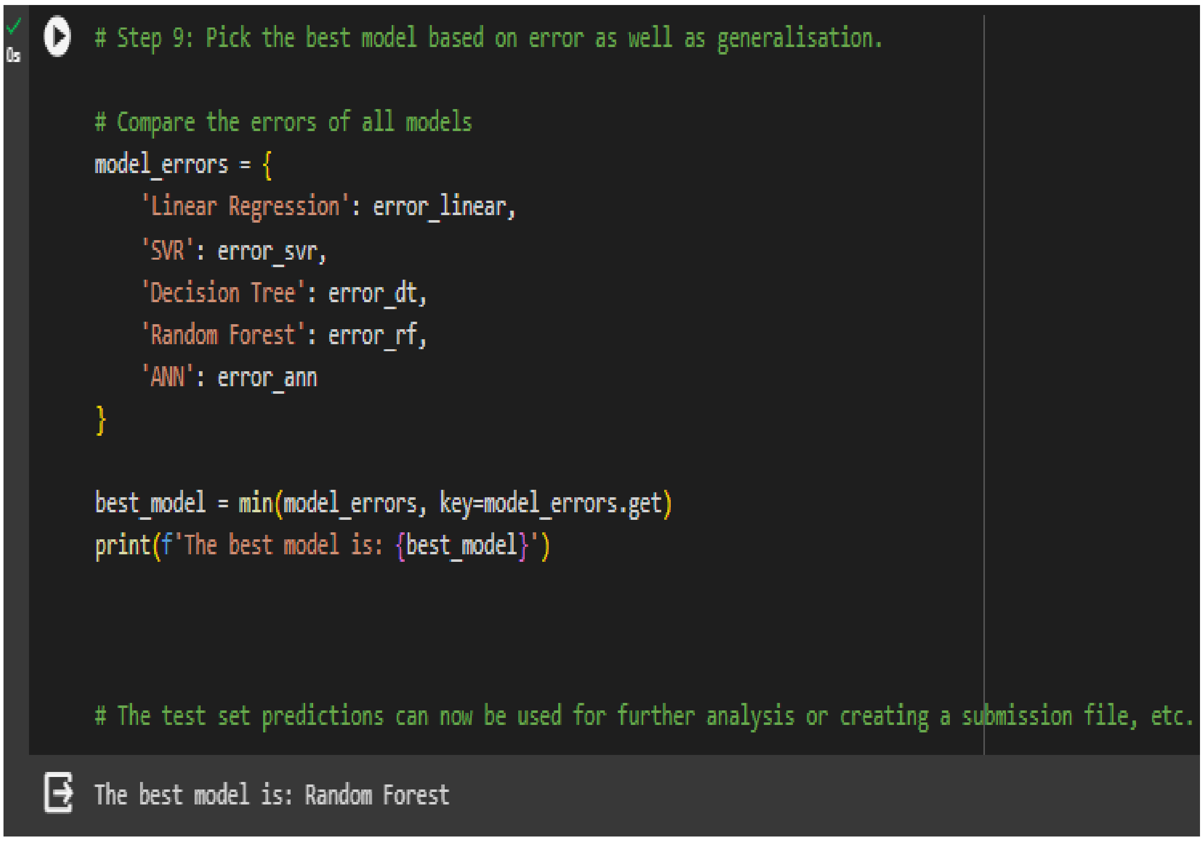
```
Epoch 94/100
366/366 [==============================] - 1s 3ms/step - loss: 763584064.0000 - val_loss: 990585536.0000
Epoch 95/100
366/366 [==============================] - 1s 3ms/step - loss: 763581760.0000 - val_loss: 990589760.0000
Epoch 96/100
366/366 [==============================] - 1s 3ms/step - loss: 763540032.0000 - val_loss: 990582656.0000
Epoch 97/100
366/366 [==============================] - 1s 2ms/step - loss: 763532864.0000 - val_loss: 990587648.0000
Epoch 98/100
366/366 [==============================] - 1s 2ms/step - loss: 763523136.0000 - val_loss: 990605248.0000
Epoch 99/100
366/366 [==============================] - 1s 2ms/step - loss: 763507072.0000 - val_loss: 990621696.0000
Epoch 100/100
366/366 [==============================] - 1s 2ms/step - loss: 763490496.0000 - val_loss: 990622592.0000
74/74 [==============================] - 0s 1ms/step
ANN MSE: 990622681.7628415
```

**Fig 8.9: Model Training (ANN)**



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1024

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 1)                 65

=================================================================
Total params: 9345 (36.50 KB)
Trainable params: 9345 (36.50 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
19/19 [==============================] - 0s 1ms/step
Mean Absolute error of  <keras.src.engine.sequential.Sequential object at 0x7fc444f68f70> 3974.530518515525
Mean Squared error of  <keras.src.engine.sequential.Sequential object at 0x7fc444f68f70> 195398334.21000564
Root Mean Squared error of  <keras.src.engine.sequential.Sequential object at 0x7fc444f68f70> 13978.495420108906
```

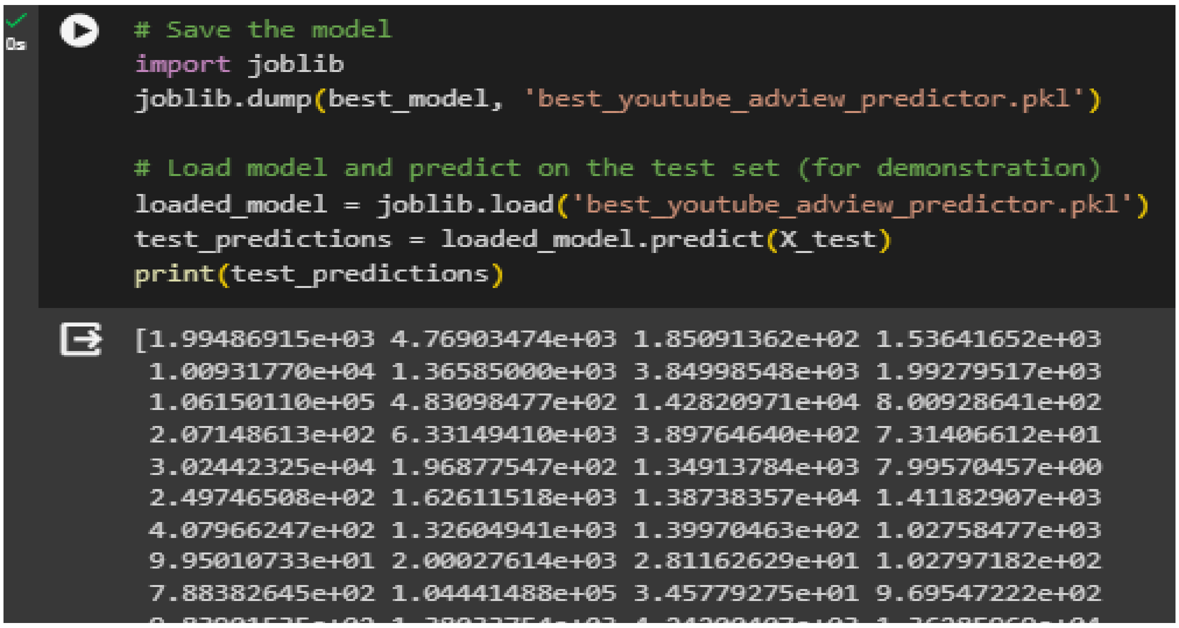
**Fig 8.10: Model Summary**



```
# Step 9: Pick the best model based on error as well as generalisation.

# Compare the errors of all models
model_errors = {
    'Linear Regression': error_linear,
    'SVR': error_svr,
    'Decision Tree': error_dt,
    'Random Forest': error_rf,
    'ANN': error_ann
}

best_model = min(model_errors, key=model_errors.get)
print(f'The best model is: {best_model}')



# The test set predictions can now be used for further analysis or creating a submission file, etc.
```

```
The best model is: Random Forest
```

**Fig 8.11: Model Selection**



```
# Save the model
import joblib
joblib.dump(best_model, 'best_youtube_adview_predictor.pkl')

# Load model and predict on the test set (for demonstration)
loaded_model = joblib.load('best_youtube_adview_predictor.pkl')
test_predictions = loaded_model.predict(X_test)
print(test_predictions)
```

```
[1.99486915e+03 4.76903474e+03 1.85091362e+02 1.53641652e+03
 1.00931770e+04 1.36585000e+03 3.84998548e+03 1.99279517e+03
 1.06150110e+05 4.83098477e+02 1.42820971e+04 8.00928641e+02
 2.07148613e+02 6.33149410e+03 3.89764640e+02 7.31406612e+01
 3.02442325e+04 1.96877547e+02 1.34913784e+03 7.99570457e+00
 2.49746508e+02 1.62611518e+03 1.38738357e+04 1.41182907e+03
 4.07966247e+02 1.32604941e+03 1.39970463e+02 1.02758477e+03
 9.95010733e+01 2.00027614e+03 2.81162629e+01 1.02797182e+02
 7.88382645e+02 1.04441488e+05 3.45779275e+01 9.69547222e+02
```

**Fig 8.12: Predictions on test data**

# Chapter 9

# CONCLUSION

## 9.1 Conclusion

In conclusion, the Youtube Ad View Prediction project has successfully addressed the challenge of estimating the number of ad views on the platform based on various metrics. Through the implementation of multiple machine learning algorithms, including Linear Regressor, Support Vector Regressor, Decision Tree Regressor, Random Forest Regressor, and Artificial Neural Network (ANN), we have gained valuable insights into viewer behavior and engagement patterns.

The systematic methodology employed in data preparation, cleaning, model training, and evaluation has resulted in the identification of the Random Forest Regressor as the most effective model for predicting YouTube ad views. The comprehensive testing phase, utilizing a dedicated test dataset and storing new outputs in the .h5 file format, ensures the reliability and accuracy of the models in real-world scenarios.

The heatmap visualization of the correlation matrix has provided a clear understanding of the relationships between different metrics, guiding the feature selection process and enhancing the interpretability of the models.

## 9.2 Further Enhancements

While the current implementation yields promising results, there is room for further enhancements and refinements to elevate the predictive capabilities of the system. Future improvements could include:

Feature Engineering: Explore additional features or engineered features that may contribute to better model performance.

Hyperparameter Tuning: Fine-tune the hyperparameters of the selected Random Forest Regressor and ANN models to optimize their predictive accuracy.

Incorporation of Time-Series Data: If applicable, consider incorporating time-series data to capture temporal trends and enhance the predictive power of the models.

Ensemble Techniques: Experiment with ensemble techniques beyond Random Forest, such as gradient boosting, to further boost the predictive performance of the system.

User Interface Development: Develop a user-friendly interface to facilitate easy interaction with the prediction system, enabling users to input new data and receive ad view predictions.

These enhancements, coupled with continuous model evaluation and monitoring, will contribute to the ongoing evolution and effectiveness of the YouTube Ad View Prediction system.

# Chapter 10

## REFERENCES

**Book Used**

- Ullal, M. S., Hawaldar, I. T., & Nadeem, M. (2021). The Role of Machine Learning in Digital Marketing. SAGE Open, 11(3). https://doi.org/10.1177/21582440211050394

**Reference Used**

[1] Miller, T. W. (2015). Modeling Techniques in Predictive Analytics: Business Problems and Solutions with R. Pearson FT Press. Retrieved from https://ptgmedia.pearsoncmg.com/images/9780133412932/samplepages/0133412938.pdf

[2] Wang, Y., Wang, X., Yang, X., Yuan, F., & Li, Y. (2023). The influence of temporal focus on individual intertemporal decision-making in life history strategy framework. Personality and Individual Differences, 202, 111694. https://doi.org/10.1016/j.paid.2023.111694

[3] Cooper, L. D. (1991). Temporal factors in classical conditioning. Learning and Motivation, 22(3), 215-236. https://doi.org/10.1016/0023-9690(91)90020-9

[4] Gao, B., Wang, Y., & Hu, Y. (2023). Artificial Intelligence in Advertising: Advancements, Challenges, and Ethical Considerations in Targeting, Personalization, Content Creation, and Ad Optimization. SAGE Open, 13(1). https://doi.org/10.1177/21582440231210759

[5] Kumar, V. (2018). Predictive Analytics: A Review of Trends and Techniques. International Journal of Computer Applications, 182(1), 31-37. https://doi.org/10.5120/ijca2018917434

[6] GeeksforGeeks. (n.d.). Machine Learning Model Evaluation. Retrieved from https://www.geeksforgeeks.org/machine-learning-model-evaluation/

[7] Reddy, K. K., Reddy, M. P., Reddy, K. D. R., Golekar, N., & Kumar, A. P. (n.d.). YouTube AdView Prediction Using Regression Model. YMER Digital. Retrieved from https://ymerdigital.com/uploads/YMER210586.pdf