Open Access and Peer Reviewed Journal ISSN 2394-2231 https://www.ijctjournal.org/

RESEARCH ARTICLE OPEN ACCESS

PREDICTING YOUTUBE VIDEO VIEWS USING SUPERVISED REGRESSION MACHINE LEARNING

Sakshi Bankar, Shruti Holkar, Shraddha Chavan

School of Information Technology, Shree Chanakya Education Society's Indira University, Pune, Maharashtra, India

sakshi.bankar@indirauniversity.edu.in | shruti.holkar@indirauniversity.edu.in | shraddha.chavan@indirauniversity.edu.in

Under the Guidance of Mr. Nitin Joshi and Dr. Manisha Patil

Predicting the popularity of online video content is an important machine learning problem because video views are influenced by multiple measurable characteristics such as engagement, publication timing, and platform metadata. This research presents a supervised learning approach for predicting the number of views received by trending YouTube videos in India. The study uses the INvideos.csv dataset from Kaggle, which contains 37,352 records and 16 original columns. The target variable is views, a continuous numerical quantity, making the task a regression problem. During preprocessing, identifiers and free-text fields were removed, while publication timestamps were transformed into year, month, weekday, and hour features. Missing records were removed, and the data was divided into training and testing sets using an 80:20 split. Four regression algorithms—Linear Regression, Decision Tree Regression, Random Forest Regression, and K-Nearest Neighbors Regression—were trained and evaluated using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2. Random Forest produced the best performance, achieving an R^2 of 0.968, compared with 0.927 for Decision Tree, 0.922 for K-Nearest Neighbors, and 0.763 for Linear Regression. The results demonstrate that ensemble-based regression can effectively estimate YouTube video views from structured metadata and engagement indicators. However, because likes, dislikes, and comments may be recorded after publication, the model should be interpreted as an estimation of popularity using available metadata rather than a strictly pre-publication forecasting system.

Keywords

YouTube analytics, machine learning, supervised learning, regression, video views, Random Forest, Kaggle, predictive modeling, engagement metrics

1. Introduction

The growth of online video platforms has created large volumes of behavioral and engagement data that can be used for data-driven analysis. YouTube videos vary substantially in the number of views they receive, and understanding this variation can support content analysis, creator strategy, and platform-level research. Machine learning provides a systematic way to learn relationships between observed video attributes and view counts.

This study focuses on predicting YouTube video views using the Trending YouTube Video Statistics dataset for India. The supplied dataset contains information such as category, publication time, views, likes, dislikes, comment count, and Boolean indicators describing comment and rating availability. The objective is to build a supervised regression pipeline that learns from these structured features and predicts the continuous target variable, views.

The research follows a complete machine learning workflow: dataset acquisition, exploratory data analysis, data preprocessing, feature engineering, train-test splitting, model training, evaluation, comparison, and selection of the best-performing model. Four regression algorithms are compared, with Random Forest selected as the strongest model according to the R^2 criterion.

2. Problem Statement

Given structured metadata and engagement indicators associated with trending YouTube videos in India, develop a supervised machine learning model capable of estimating the number of views a video receives. The system should preprocess heterogeneous data, engineer useful temporal features,

the model that provides the strongest predictive performance on unseen test data.

3. Objectives

• To study the structure and characteristics of the Kaggle INvideos.csv dataset.

• To perform exploratory data analysis on video views and numeric features.

• To preprocess the dataset by removing identifiers and free-text fields and extracting useful time features.

• To build supervised regression models for predicting YouTube video views.

• To compare Linear Regression, Decision Tree, Random Forest, and K-Nearest Neighbors regression.

• To evaluate the models using MAE, RMSE, and R^2.

• To identify the best-performing model for the supplied dataset.

• To examine feature importance using the selected Random Forest model.

4. Dataset Description

The study uses the Trending YouTube Video Statistics dataset available on Kaggle. The supplied case study specifies the India file, INvideos.csv. The dataset contains 37,352 rows and 16 original columns, with each record representing a trending YouTube video in India.

Feature	Description	Treatment
video_id	Unique identifier for a video	Removed
trending_date	Date on which the video was trending	Removed
title	Video title	Removed from numeric baseline
channel_title	Publishing channel name	Removed
category_id	Numeric YouTube category identifier	Retained
publish_time	Publication timestamp	Converted to temporal features
tags	Video tags	Removed
views	Total number of views	Target variable
likes	Number of likes	Retained
dislikes	Number of dislikes	Retained
comment_count	Number of comments	Retained
thumbnail_link	Thumbnail URL	Removed
comments_disabled	Whether comments were disabled	Retained
ratings_disabled	Whether ratings were disabled	Retained
video_error_or_removed	Video error/removal indicator	Retained
description	Free-text video description	Removed

5. Proposed Methodology

5.1 Data Loading

The dataset is loaded using Pandas from the file path data/INvideos.csv. Initial inspection includes the dataset shape, first records, data types, descriptive statistics, and missing-value counts.

5.2 Exploratory Data Analysis

Exploratory analysis is used to understand the distribution of the target variable and relationships among numeric features. A histogram is used to visualize the distribution of views, while a correlation heatmap is generated for numeric variables. These analyses provide an initial understanding of the scale and relationships present in the dataset.

5.3 Data Preprocessing and Feature Engineering

The publication timestamp is converted to a datetime representation and transformed into four numerical features: publish_year, publish_month, publish_dayofweek, and publish_hour. Identifier fields, URLs, and free-text fields are removed from the numeric baseline. Remaining incomplete rows are dropped. The target variable views is separated from the input features.

5.4 Train-Test Split and Scaling

The prepared dataset is divided into training and testing subsets using an 80:20 train-test split with random_state=42. StandardScaler is fitted on the training data and then applied to both training and test features. This produces scaled feature matrices for model training and evaluation.

5.5 Regression Models

Model	Configuration used
Linear Regression	Default LinearRegression()
Decision Tree Regressor	random_state=42, max_depth=20
Random Forest Regressor	n_estimators=100, random_state=42, n_jobs=-1, max_depth=20
K-Nearest Neighbors Regressor	n_neighbors=7

5.6 Evaluation Metrics

Three regression metrics are used. Mean Absolute Error (MAE) measures the average absolute prediction error. Root Mean Squared Error (RMSE) penalizes larger errors more strongly because the errors are squared before taking the square root. R^2 measures the proportion of variance in the target explained by the model; higher values indicate stronger fit on the held-out test set.

6. System Architecture

The proposed prediction workflow can be represented as a sequential pipeline:

Kaggle INvideos.csv → Data Loading → Exploratory Data Analysis → Date-Time Feature Engineering → Removal of Identifiers/Text → Missing-Value Handling → Feature/Target Separation → Train-Test Split → Feature Scaling → Regression Models → Evaluation → Best Model Selection → View Prediction

Stage	Description
Input	INvideos.csv containing trending YouTube video statistics for India
Preprocessing	Datetime conversion, feature extraction, column removal, missing-value handling
Feature preparation	Separate views as target and scale predictors
Modeling	Train four supervised regression algorithms
Evaluation	Compare MAE, RMSE, and R^2 on held-out test data
Output	Predicted views and selected best-performing model

7. Experimental Results

The supplied case study reports model performance on the held-out test set. Random Forest achieved the strongest R^2 score and the lowest MAE and RMSE among the four tested algorithms.

Model	MAE	RMSE	R^2
Linear Regression	645,698	1,637,702	0.763
Decision Tree	250,586	909,578	0.927
Random Forest	226,778	603,833	0.968
K-Nearest Neighbors	367,586	939,442	0.922

7.1 Best Model

Random Forest is selected as the best-performing model because it obtains the highest R^2 value of 0.968 and the lowest MAE and RMSE among the compared models. This indicates that, for the supplied train-test experiment, Random Forest captures the relationships between the structured input variables and video views more effectively than the other tested approaches.

7.2 Sample Predictions

The case study also reports a sample comparison between actual and predicted views using the selected Random Forest model.

Actual Views	Predicted Views
1,202,255	619,942
1,776,191	899,536
317,985	307,998
2,090,215	2,087,341
80,337	177,797
60,502	57,220
1,018,981	623,643
64,834	61,658
325,802	388,462
27,044	25,795

8. Discussion

The experimental results show a clear performance advantage for tree-based models over Linear Regression. Linear Regression produced an R^2 of 0.763, suggesting that a simple linear relationship

does not capture all of the variation in video views represented in the dataset. Decision Tree and K-Nearest Neighbors improved the R^2 to 0.927 and 0.922 respectively.

Random Forest produced the strongest results, with an R^2 of 0.968, MAE of 226,778 views, and RMSE of 603,833 views. The improvement over a single Decision Tree is consistent with the ability of an ensemble of trees to capture more complex relationships while reducing dependence on a single tree structure.

The reported sample predictions also show that the model performs very differently across individual videos. Some predictions are close to the actual view counts, while others show substantial deviations. This is expected in a highly variable popularity dataset and indicates that aggregate metrics should be considered alongside individual prediction behavior.

9. Feature Importance and Model Persistence

The practical case study includes an optional extension for calculating feature importance from the Random Forest model. Feature importance values are obtained from the model's feature_importances_ attribute and visualized as a horizontal bar chart. This provides a way to inspect which structured predictors contribute most strongly to the fitted Random Forest model.

The trained Random Forest model and the fitted StandardScaler are also saved using Joblib as outputs/best_model.pkl and outputs/scaler.pkl. Persisting both objects supports reuse of the trained pipeline for future prediction tasks, provided that new input data follows the same preprocessing and feature structure.

10. Limitations

• The baseline removes title, channel name, tags, and description, so potentially informative textual signals are not modeled.

• The model uses engagement indicators such as likes, dislikes, and comments that may only be available after publication; therefore, the experiment is not a pure pre-publication forecasting setup.

• The study uses a single train-test split rather than cross-validation, so the reported metrics describe that specific split.

• Dropping incomplete rows may reduce the amount of usable data and can potentially introduce selection effects.

• The dataset represents trending videos in India and therefore may not generalize directly to all YouTube videos, regions, or time periods.

• The reported experiment does not establish causal relationships between features and views; it measures predictive association.

11. Future Scope

• Include natural-language features from video titles, tags, and descriptions using NLP and text embeddings.

• Add channel-level historical statistics, upload frequency, subscriber-related variables, and other creator metadata where available.

• Use cross-validation and systematic hyperparameter tuning to improve robustness.

investigate models designed for highly skewed count targets, including log-transformed targets and other suitable regression approaches.

• Develop a web dashboard that accepts video metadata and returns an estimated view count.

• Evaluate the model across different countries and time periods to study generalization.

• Use explainable AI methods to provide human-readable reasons for individual predictions.

12. Conclusion

This research demonstrates a complete supervised regression workflow for predicting YouTube video views using the Kaggle Trending YouTube Video Statistics dataset for India. The 37,352-record INvideos.csv dataset was prepared by removing identifiers and free-text fields, extracting temporal features from publish_time, handling incomplete rows, separating the target variable, and scaling the predictors. Four regression algorithms were trained and evaluated using MAE, RMSE, and R^2.

Among the tested models, Random Forest achieved the best performance with an R^2 of 0.968, MAE of 226,778, and RMSE of 603,833 on the held-out test set. These findings show that ensemble-based machine learning can provide strong estimates of video popularity from structured metadata and engagement indicators. At the same time, the timing of engagement variables and the use of a single train-test split should be considered when interpreting the results. Overall, the study provides a practical foundation for further work in YouTube analytics, popularity prediction, and data-driven content analysis.

13. References

[1] Kaggle, “Trending YouTube Video Statistics,” dataset by datasnaek, including the India file INvideos.csv.

[2] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[3] W. McKinney, Python for Data Analysis, O’Reilly Media.

[4] C. R. Harris et al., “Array programming with NumPy,” Nature, vol. 585, pp. 357–362, 2020.

[5] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Springer.