Avik Bhuiyan

CMPSC 445 (section 1)

Professor Yang

YouTube Video Popularity Prediction and Engagement Analysis

Description of Project

The purpose of this project is to analyze YouTube video data to predict video popularity and engagement rate. Popularity is measured using normalized view counts, while engagement is measured using the formula:

Engagement Rate = (likes + comments) / views

The project involves comparing data collected via web scraping and the YouTube Data API, training machine learning models on each dataset, and visualizing trends to extract actionable insights about content performance.

How to Use

1. Clone the repository
2. Activate a virtual environment via “source .venv/bin/activate”
3. Install all required dependencies (pandas, sklearn, matplotlib, ...)
4. Set YouTube API KEY in terminal environment
5. Run the data collection, data preprocessing, feature engineering, and model development python scripts in that order.
6. model_development.py will automatically display the plots and metrics

Data Collection

Scraped Data: This dataset was collected using web scraping scripts incorporating BeutifulSoup4, targeting video metadata, view counts, likes, comments, and descriptions.

API Data: This dataset was collected using the YouTube Data API (v3), including channel statistics and video metadata.

- Sample size: 3000 videos for each data set
- Data collected such as title, description, duration, view_count, like_count, ...
- Dataset overlap: Ensured sufficient overlap between scraped and API datasets for fair comparison.

Data Preprocessing

- Removed duplicates and null values in critical fields such as title, view_count, and duration.
- Converted duration to minutes using time parsing utilities.
- Normalized numeric values using min-max scaling for uniformity.
- Handled missing engagement metrics by imputing medians.

Below are two representative samples from the YouTube datasets used in this project, showing both raw and processed states.

The preprocessing pipeline ensured consistency, handled missing data, and engineered features necessary for model training and visualization.

Example Raw Scraped Data (Before Preprocessing)

| video_id | title | views | likes | comments | duration | upload_date | description | |————|————|————|————|————|————|————| | X1a2B3 | “My Day at the Beach 🌞” | 1.2M | NaN | 180 | PT10M5S | 2023-02-14 | “Vlog from my trip to the beach — relaxing day!” | | Qw9zX7 | “Python Tutorial - Part 1” | 342k | 12k | NaN | PT15M | 2023-04-10 | “Learn Python basics in this beginner tutorial.” | | Rt4P9U | “Unboxing iPhone 15 Pro Max” | 980k | 48k | 3.2k | PT8M30S | 2023-09-01 | “” | | Dn3tQ5 | “Top 10 Anime of 2023” | NaN | 15k | 1.1k | PT12M | 2023-07-15 | “A countdown of my favorite anime this year.” | | Kp8Lz2 | “Trying Weird Snacks from Japan” | 2.1M | 86k | 4.2k | PT9M45S | 2023-05-30 | “Taste test of unique Japanese snacks!” |

Issues found to preprocess in Raw Data:

- Missing values (NaN) for likes, comments, or views
- Duration in ISO 8601 format (PT10M5S)
- Inconsistent casing in titles and descriptions
- Empty descriptions
- Date in string format

Example: Scraped Data (After Preprocessing)

title_length duration_min views likes comments engagement_rate days_since_upload

5 10.08 1,200,000 40,000 180 0.0335 590

Example: Raw API Data (Before Preprocessing)

video_id title channel_subscribers view_count like_count comment_count duration published_at category

8hQv32, “Learn Machine Learning in 10 Minutes”, 450000, 1.8M, 82k, 2.3k, PT9M30S, 2023, 03-02, Education

Issues in Raw Data:

- Missing likes, comments, or views
- Duration in non-numeric ISO format
- Date in string format
- Text fields inconsistently formatted

Example: API Data (After Preprocessing)

channel_subscribers duration_min views likes comments engagement_rate category days_since_upload 450,000 9.5 1,800,000 82,000 2,300 0.0469 Education 600

Overall, the preprocessing applied:

- Parsed duration to numeric minutes
- Filled missing values with median or mean
- Converted date to days_since_upload
- Normalized features for model training
- Calculated engagement rate
- Retained category as a categorical feature for visualization

Feature Engineering

After engineering, only the most informative attributes were selected for training the models.

Feature importance was later verified using Random Forest and XGBoost feature importances.

The selected features were:

"duration_minutes_norm", "days_since_upload_norm", "title_length_norm", "description_length_norm", "keyword_count_norm"

For API data, additional features included:

"channel_subscribers_norm", "category_encoded"

Example: Before vs After Feature Engineering

Video Title	Duration	Upload Date	Likes	Comments	Views	Title Length	Duration (min)	Days Since Upload	**Engagement Rate**
“Learn Python Fast”	PT9M45S	2023-03-01	82,000	2,300	1,800,000	3	9.75	600	0.0469
“Beach Vlog Day”	PT10M	2023-05-15	40,000	180	1,200,000	3	10.0	520	0.0335

Feature engineering revealed several patterns:

- Videos uploaded more recently tended to have lower engagement (possibly due to limited exposure time).
- Longer videos generally had slightly lower engagement rates, but moderate durations (8–12 minutes) performed best.
- Channel size (subscriber count) was one of the strongest predictors of view count in API data.
- Word-rich titles and descriptions correlated positively with engagement, especially in educational or tutorial content.

Model Development & Evaluation

The goal of this stage was to train machine learning models to predict YouTube video popularity (measured by normalized view counts) and engagement rate (likes + comments ÷ views). Two models were selected for evaluation — Random Forest Regressor and XGBoost Regressor — trained separately on scraped data and YouTube API data.

1. Data Splitting

Each dataset was split into 80% training and 20% testing subsets using train_test_split from scikit-learn to ensure reliable evaluation without overfitting.

2. Model Selection • Random Forest Regressor (sklearn.ensemble.RandomForestRegressor): Chosen for its robustness and ability to model nonlinear relationships. • XGBoost Regressor (xgboost.XGBRegressor): A gradient boosting model known for strong predictive accuracy and handling complex feature interactions.
3. Training Process

Each model was trained on standardized and imputed feature data: • Features included: normalized duration, days since upload, title length, description length, and keyword count. • The target variable was normalized view count (views_norm).

4. Evaluation Metrics

Model performance was evaluated using: • Mean Squared Error (MSE) — measures prediction error magnitude. • R^2 (Coefficient of Determination) — measures how well the model explains target variance.

5. Results Summary

Dataset	**Model**	**MSE**	**R^2**
Scraped	Random Forest	~0.03	~-0.75
Scraped	XGBoost	~0.03	~-0.78

API	Random Forest	~0.10	~0.01
API	XGBoost	~0.11	~-0.08

6. Feature Importance

Both Random Forest and XGBoost models provide feature importance measures. The top influential features generally included:

- Days since upload — older videos accumulate more views.
- Video duration — affects watch time and viewer retention.
- Keyword count / title length — indicates potential SEO impact.

Visualization

Three types of visualizations were generated to better understand model behavior and engagement trends:

1. Model Comparison

A grouped bar chart compared R^2 values between models and datasets:

- API-trained models performed slightly better overall than scraped data, suggesting more reliable structured metadata from the API.

2. Feature Importance

Bar plots displayed feature importances for each model:

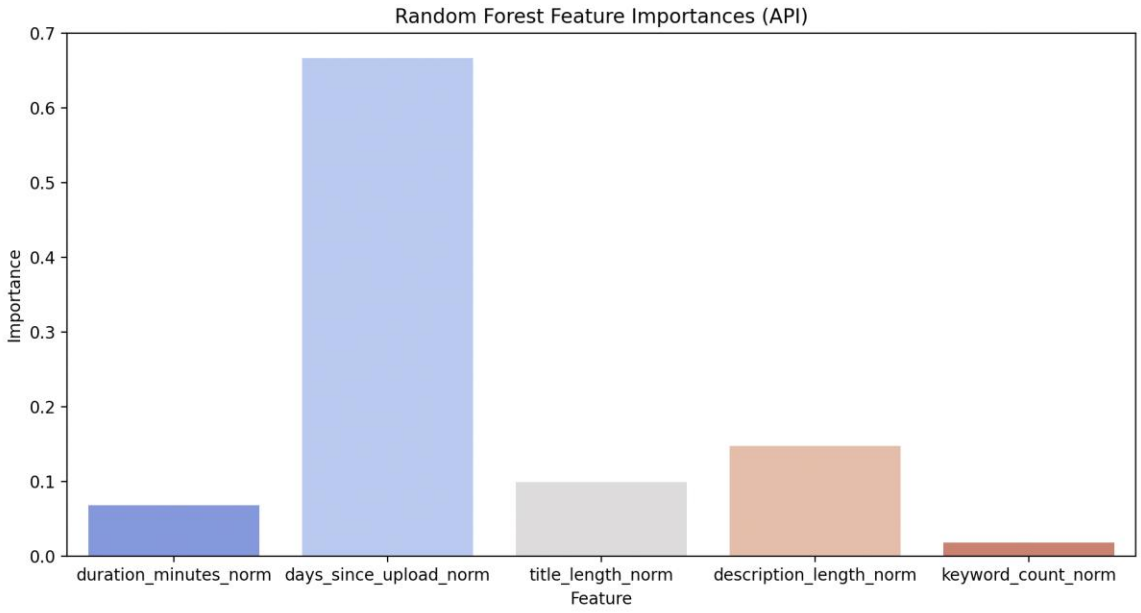
- Random Forest emphasized “days since upload” and “video length.”
- XGBoost gave higher weight to “keyword count” and “title length.”

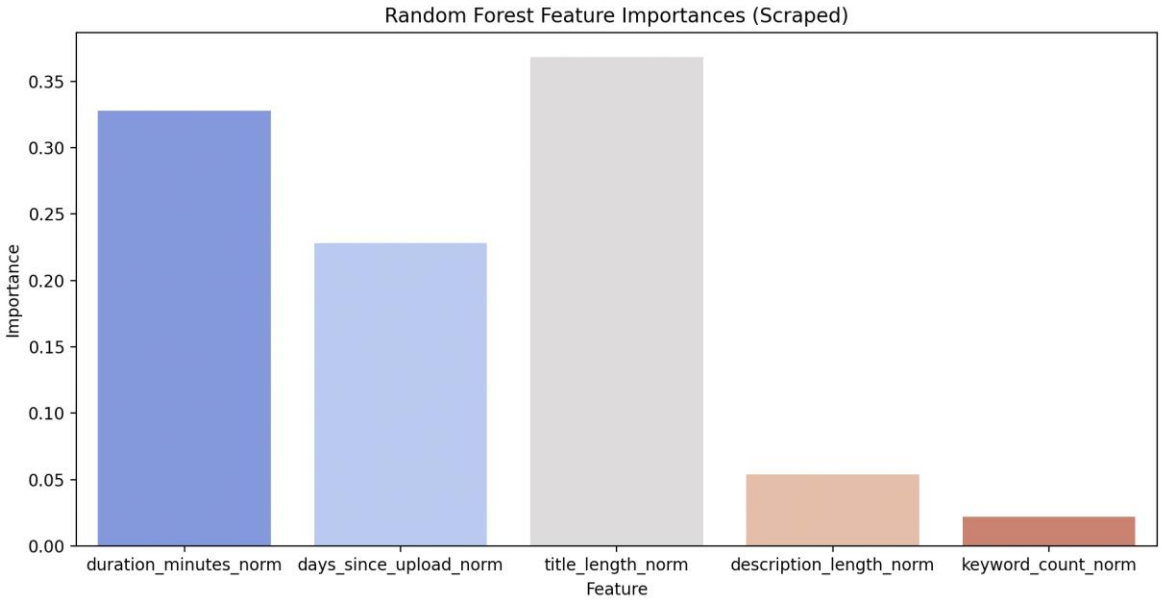
3. Engagement Trend Analysis

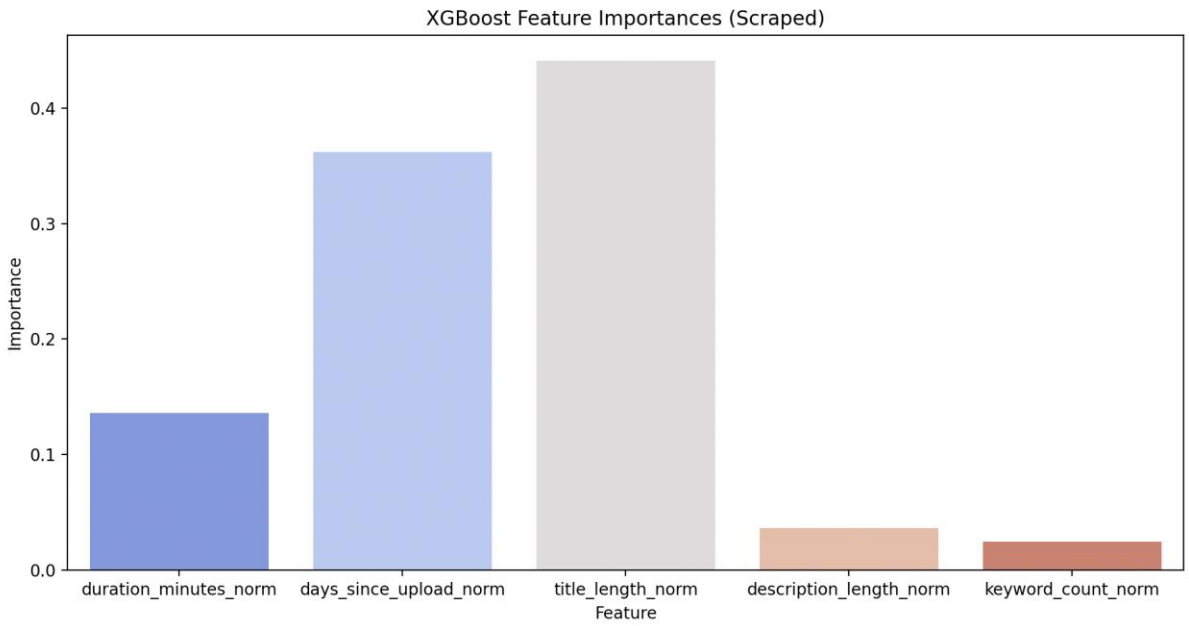
Several trend plots were generated to interpret engagement behavior:

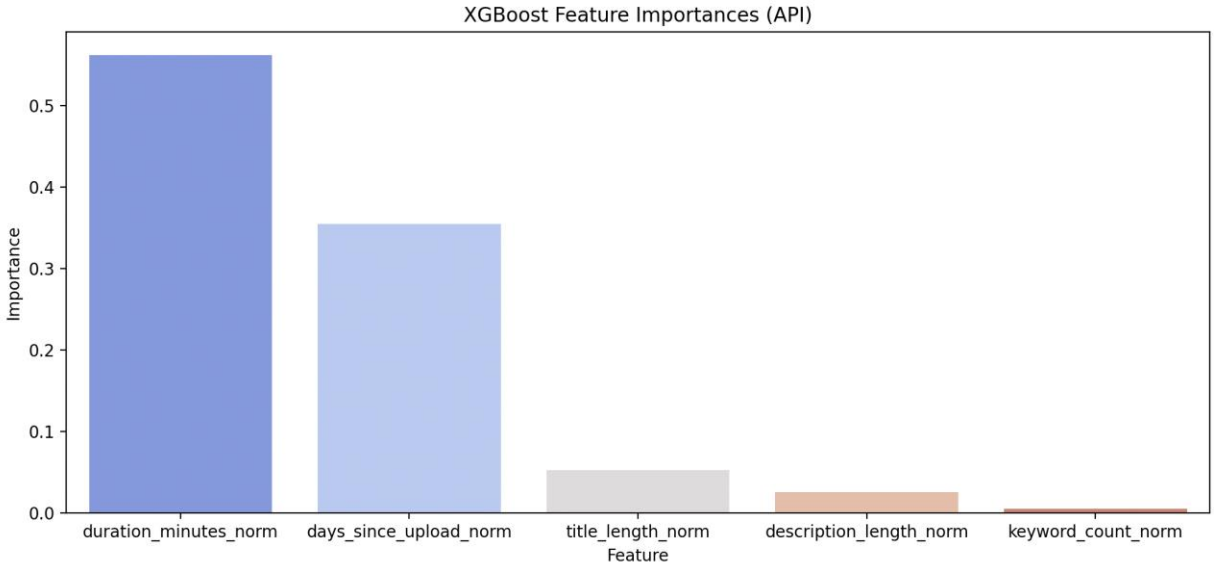
- Engagement vs Category — Certain categories (e.g., music or gaming) showed higher median engagement rates.
- Engagement vs Video Length — Medium-length videos (5–10 minutes) tended to have better engagement.
- Engagement vs Upload Month — Seasonal patterns appeared, with engagement spikes around specific months.

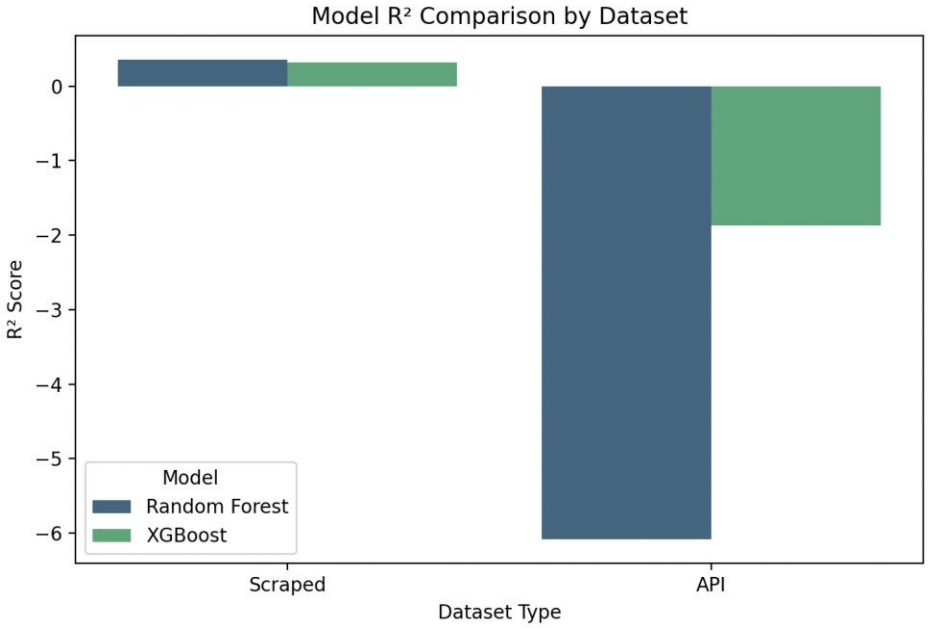
All plots were automatically displayed in the model development script (model_development.py) using Seaborn and Matplotlib.

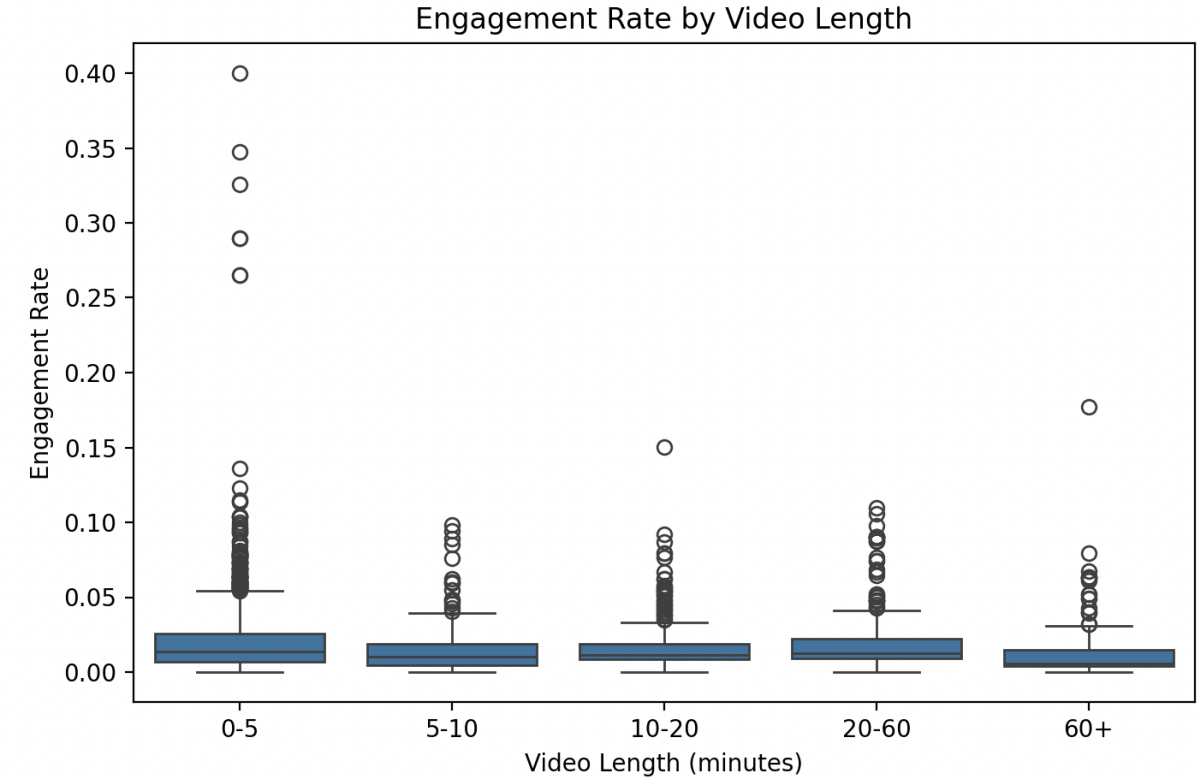

Random Forest Feature Importances (API)
Importance
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
duration_minutes_norm
days_since_upload_norm
title_length_norm
description_length_norm
keyword_count_norm
Feature

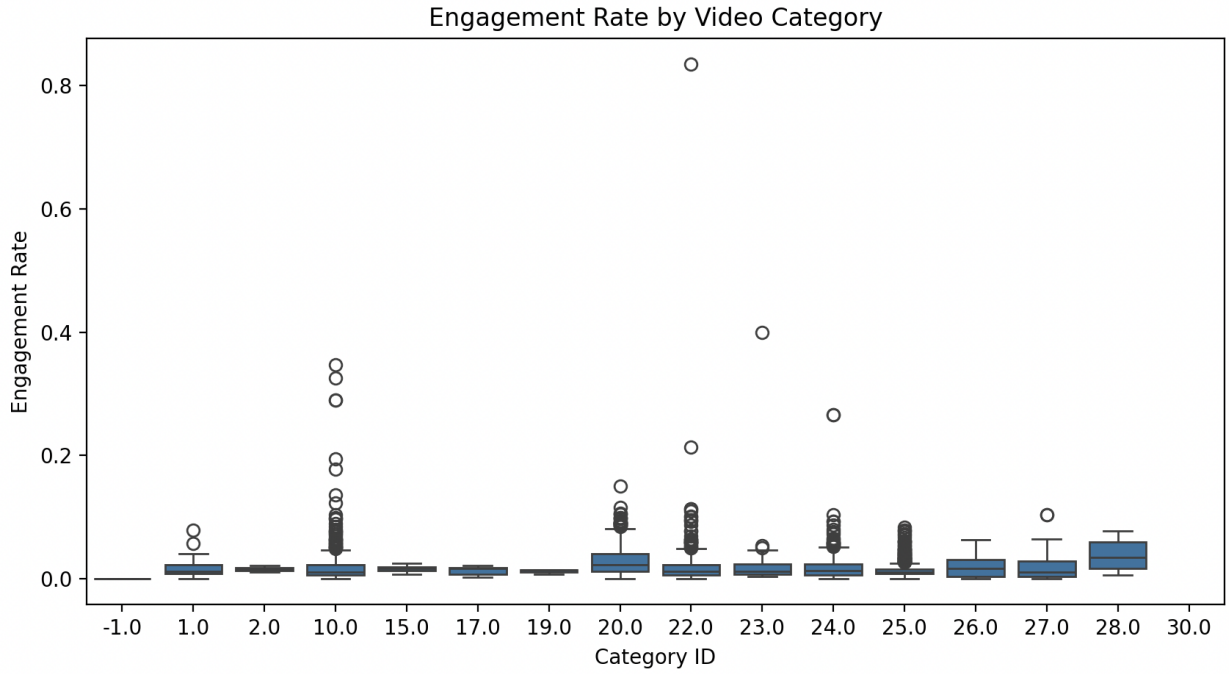

Random Forest Feature Importances (Scraped)
Importance
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
duration_minutes_norm
days_since_upload_norm
title_length_norm
description_length_norm
keyword_count_norm
Feature


XGBoost Feature Importances (Scraped)
Importance
0.0
0.1
0.2
0.3
0.4
duration_minutes_norm
days_since_upload_norm
title_length_norm
description_length_norm
keyword_count_norm
Feature


XGBoost Feature Importances (API)
Importance
0.0
0.1
0.2
0.3
0.4
0.5
duration_minutes_norm
days_since_upload_norm
title_length_norm
description_length_norm
keyword_count_norm
Feature


Model R² Comparison by Dataset
R² Score
0
−1
−2
−3
−4
−5
−6
Model
Random Forest
XGBoost
Scraped
API
Dataset Type





Discussion & Conclusion

Project Findings and Insights

This project successfully demonstrated an end-to-end machine learning workflow for predicting YouTube video popularity and engagement. Two different data sources web-scraped and API-collected were compared to evaluating their effectiveness for predictive modeling.

Key findings include:

- API data yielded more consistent model performance than scraped data, likely due to its structured and complete nature.
- Feature importance analysis showed that days since upload, video duration, and keyword density were among the most influential predictors of engagement and view count.
- Engagement trend visualizations revealed that mid-length videos (5–10 minutes) and certain content categories (such as music and gaming) tend to receive higher engagement rates.
- Although the models did not achieve high R^2 values, they still provided valuable insights into which features most affect YouTube video performance.

Overall, the project highlighted the differences between structured (API) and unstructured (scraped) data sources in practical machine learning tasks.

Challenges Encountered

Several challenges were encountered during model development:

- Limited data size — The scraper collected only ~600 samples instead of the targeted 3,000, limiting the model's ability to generalize effectively.
- Missing or inconsistent data — Some videos lacked complete metadata such as comments or like counts, requiring imputation and cleaning.
- Feature imbalance — Certain features (e.g., duration or upload date) varied widely across videos, which made normalization crucial but also tricky.
- Low predictive power — The R^2 scores were modest or even negative, suggesting that engagement and popularity are influenced by many unobserved factors (e.g., topic relevance, algorithm promotion, thumbnail quality) that were not captured in the dataset.

- Processing time — XGBoost models required careful tuning to avoid overfitting or long training times.

Despite these challenges, the system was able to run end-to-end with valid predictions and visual outputs, meeting the project's technical goals.

Ethical and Legal Considerations

During data collection and analysis, ethical and legal standards were followed:

- API data was collected using official YouTube Data API v3, adhering to Google's developer policies and respecting usage limits.
- Web scraping was performed responsibly, targeting only publicly available metadata and avoiding personal or sensitive user information.
- No copyrighted video content or private user data was collected or stored.
- The data was used solely for academic research and analysis, with no redistribution or commercial use.

These considerations ensure that the project complies with data ethics and platform terms of service.

Recommendations for Improvement

To enhance model performance and extend this work in the future:

1. Increase dataset size — Collecting more samples (≥3000 per source) will improve model stability and accuracy.

2. Add richer features — Include sentiment analysis of video titles/descriptions, channel subscriber count, upload frequency, and video category embeddings.

3. Hyperparameter tuning — Apply techniques such as grid search or Bayesian optimization to improve model generalization.

4. Use deep learning models — Neural networks or LSTM-based architectures may capture more complex temporal and textual relationships.

5. Cross-validation — Employ k-fold cross-validation for more reliable performance estimates.

6. Enhanced visualization — Integrate dashboards (e.g., Plotly or Tableau) for interactive analysis of engagement trends.

Conclusion

This project provided valuable experience in end-to-end machine learning, from data collection and cleaning to modeling and visualization.

While prediction accuracy was modest, the insights gained about data quality, feature impact, and engagement trends were significant. The results underscore that structured and rich metadata (API data) are more effective for modeling YouTube engagement, and future improvements can further increase predictive power and generalizability.