YouTube Revenue Prediction Model - Documentation

# 1. Introduction

This document explains how the YouTube Revenue Prediction model works. The model was trained using historical video performance data such as views, likes, comments, watch time, subscribers, and engagement rate. It predicts the expected revenue for a given video.

# 2. Input Data Format

The model requires the following features as input. These features must be provided in the same format as during training:

- views: Number of video views (integer)  
- likes: Number of likes (integer)  
- comments: Number of comments (integer)  
- subscribers: Channel subscribers (integer)  
- engagement\_rate: (likes + comments) / views (float)  
- watch\_time\_minutes: Total watch time in minutes (float)  
- year: Upload year (integer)  
- month: Upload month (integer)  
- dayofweek: Upload day of the week (0=Monday, 6=Sunday)  
- category: Video category (string: Education, Music, Tech, Entertainment, Gaming, Lifestyle)  
- device: Device type (string: TV, Mobile, Desktop, Tablet)  
- country: Country code (string: CA, DE, IN, AU, UK, US)

# 3. Preprocessing with Pipeline

The model was trained using a Scikit-learn Pipeline. This means all preprocessing steps such as encoding and scaling are already included inside the pipeline. At prediction time, you must provide raw input data in the above format, and the pipeline will automatically handle preprocessing.  
  
Preprocessing steps include:  
- Scaling: MinMaxScaler for 'watch\_time\_minutes', RobustScaler for 'engagement\_rate'.  
- Encoding: OneHotEncoding for categorical variables (category, device, country).  
- Passing scaled/encoded features into the XGBoost regressor.

# 4. Prediction Workflow

At runtime (e.g., in Streamlit app):  
1. Collect user input (views, likes, etc.).  
2. Build a DataFrame with the same feature names as training.  
3. Load the pipeline using pickle.  
4. Call pipeline.predict(input\_data).  
5. The pipeline preprocesses the input and predicts the revenue.

# 5. Example Usage

Example Streamlit usage:  
  
model = pickle.load(open('pipe\_xgb.pkl', 'rb'))  
prediction = model.predict(input\_data)[0]  
st.success(f'Predicted Revenue: ${prediction:.2f} USD')