Predicting Viewership of Educational Videos

Shivali Chainani

GSTEM

NYU COURANT

Table of contents

GSTEM

NYU COURANT

Introduction: The rise of Educational Video Content

Question

What truly makes videos engaging, and can we predict how successful an educational video will be?

THE GAP

- several studies have explored individual factors influencing video viewership
- there is a lack of comprehensive models that analyze multiple traits simultaneously to predict viewership and engagement.

Objectives

This research aims to predict the viewership and engagement of educational videos across platforms that are reputable for publishing content, by means of an AI model.

Methodology

Data Acquisition

1. Video Transcripts
2. Word Rate Calculations
3. Bit Calculations

	channelName	viewCount	likeCount	audio_duration	duration	Nwords	bits	bit_avg	wpm	bps	bpw
0	Saturday_Night_Live	11625656.0	125606.0	202.696731	161.39500	387.0	3457.681066	10.022264	143.870628	21.423719	8.934576
1	Saturday_Night_Live	1689234.0	16884.0	491.115320	460.07500	1132.0	10359.399950	9.999421	147.628104	22.516763	9.151413
2	Key_Peele	13237452.0	253336.0	260.487628	238.65000	822.0	7110.973340	9.443524	206.662476	29.796662	8.650819
3	Aunty_Donna	30084.0	1135.0	1784.672244	1739.53740	4235.0	36276.144580	9.756897	146.073318	20.853903	8.565796
4	Key_Peele	4223115.0	107902.0	230.373141	203.81500	521.0	4512.611894	9.580917	153.374384	22.140725	8.661443
5	Saturday_Night_Live	1646080.0	17878.0	710.570705	678.95500	1596.0	14038.070000	9.920898	141.040275	20.675995	8.795783
6	Saturday_Night_Live	2436520.0	28337.0	359.431410	328.00000	1060.0	9553.047849	9.828239	193.902439	29.125146	9.012309
7	Saturday_Night_Live	8039348.0	85251.0	304.426923	282.10000	603.0	5485.349915	9.606567	128.252393	19.444700	9.096766
8	Key_Peele	23779763.0	218844.0	171.456923	158.39500	456.0	4180.613960	9.883248	172.732725	26.393598	9.168013
9	Key_Peele	1788529.0	46659.0	347.106987	322.74500	445.0	3903.283822	9.405503	82.727850	12.094018	8.771424
10	Saturday_Night_Live	17232603.0	263390.0	341.560962	318.00500	650.0	6398.628128	10.753997	122.639581	20.121156	9.844043
11	Saturday_Night_Live	694939.0	13125.0	156.399679	136.03000	383.0	3822.532045	10.472691	168.933323	28.100655	9.980501
12	Key_Peele	21144105.0	176359.0	254.378974	224.15500	194.0	1744.914178	9.802889	51.928353	7.784409	8.994403
13	Saturday_Night_Live	4624479.0	48452.0	403.933333	359.29500	811.0	7105.790502	9.733960	135.431887	19.777037	8.761764
14	Saturday_Night_Live	31824992.0	119756.0	232.462885	181.91000	353.0	2874.990152	9.583301	116.431202	15.804465	8.144448
15	Key_Peele	14867270.0	283598.0	179.735769	159.67500	279.0	2320.506612	9.510273	104.837952	14.532686	8.317228

Figure 1: Data CSV used in programming environment

Pre-processing and data cleaning

```python
import pandas as pd
from IPython.display import display
import pandas as pd
import matplotlib.pyplot as plt

# identifying the file path
file_path = 'data.csv'

# Load the CSV file into a pandas DataFrame
data = pd.read_csv(file_path)

# Delete unwanted or unuseful columns
data = data.drop(columns=['id', 'title', 'url', 'category', 'downloaded', 'transcript', 'level', 'playlist_name', 'playlist_id', 'audio'])

display(data.head(20))
```

Figure 2: Snippet of Pre-processing code

Exploring Data

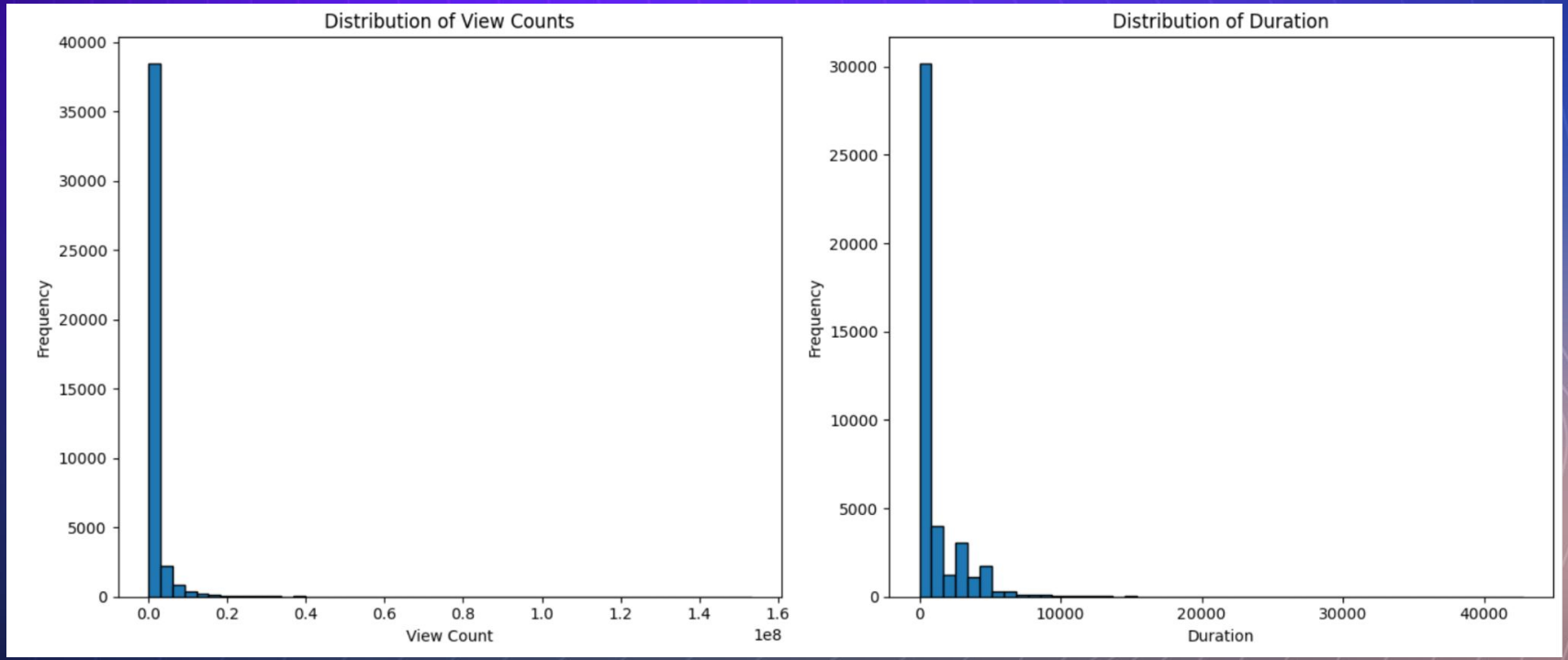


Figure 3: Duration distribution

Exploring Data (cont.)

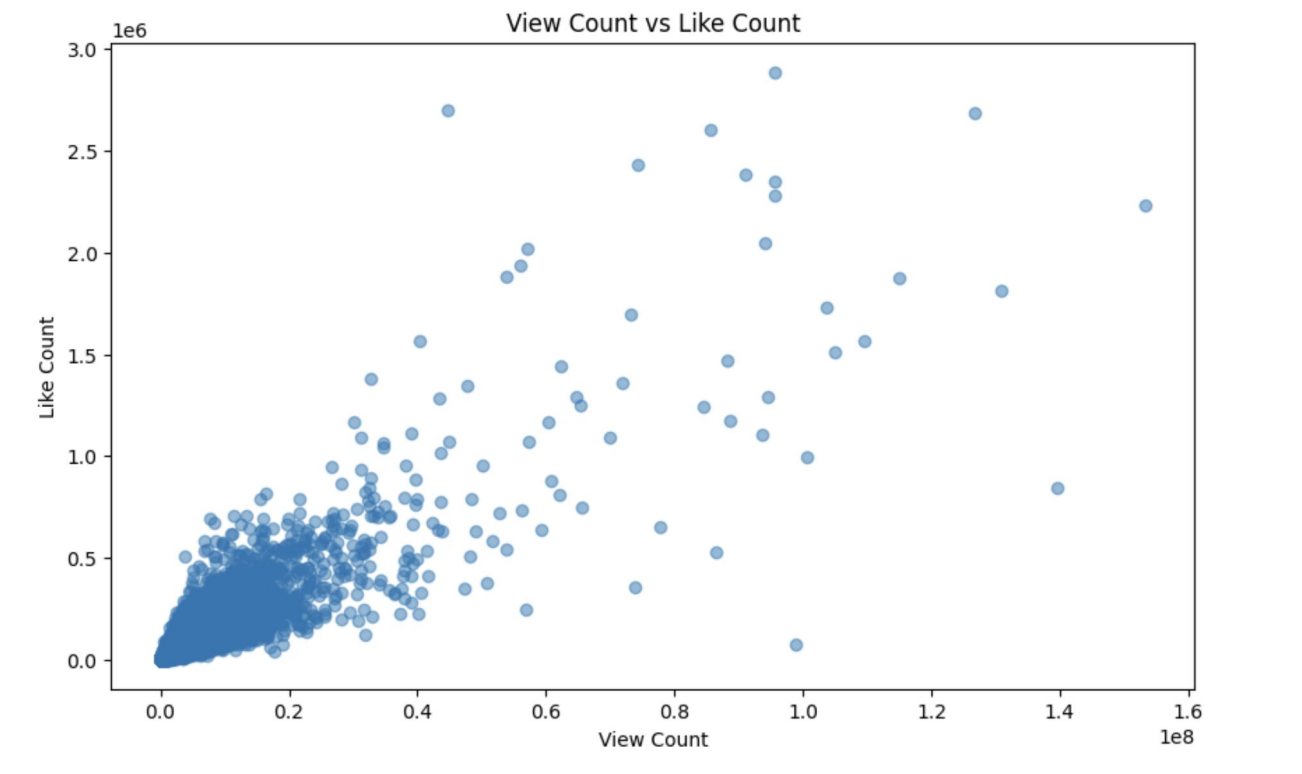
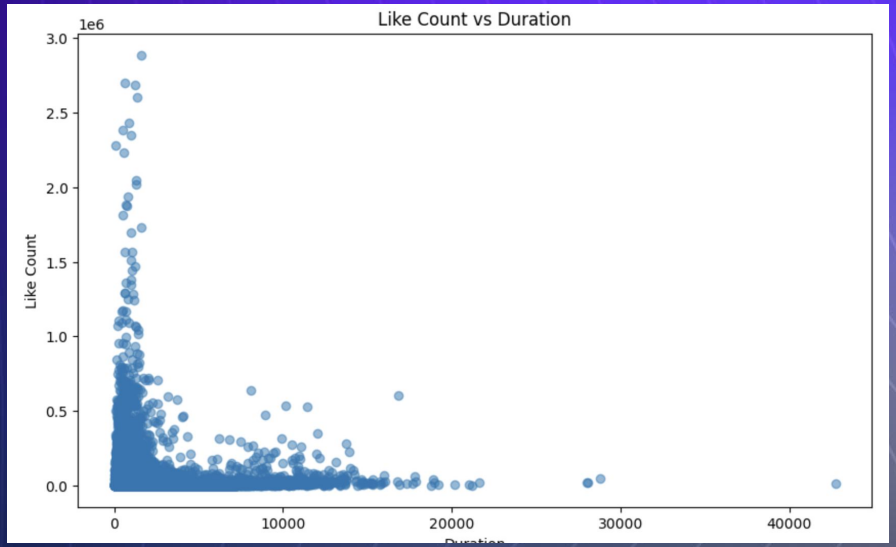


Figure 4: Graphic Analysis of Linear correlation

Exploring Data (cont.)



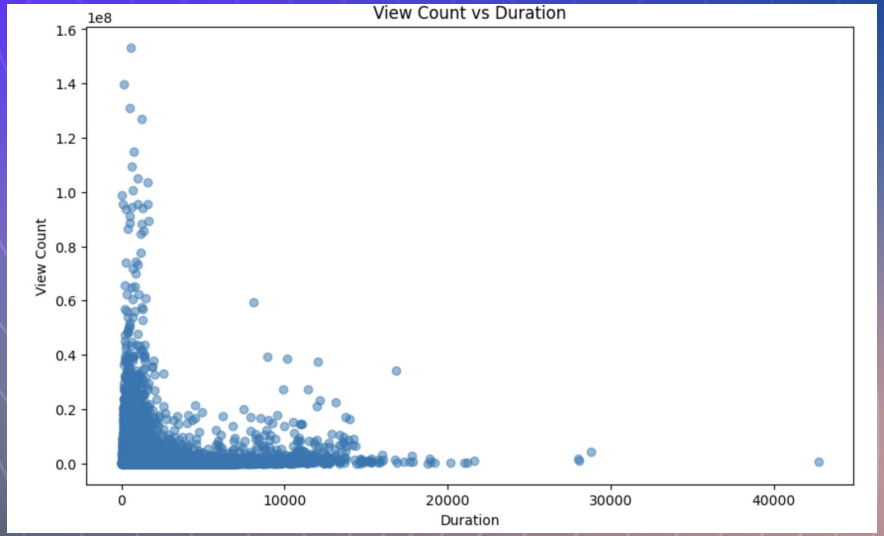
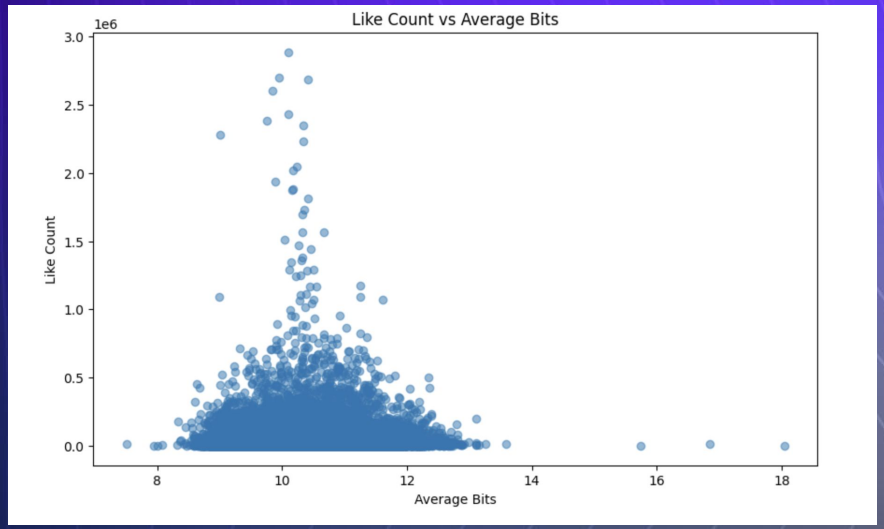


Figure 5: Duration vs. Dependent Variables

Exploring Data (cont.)



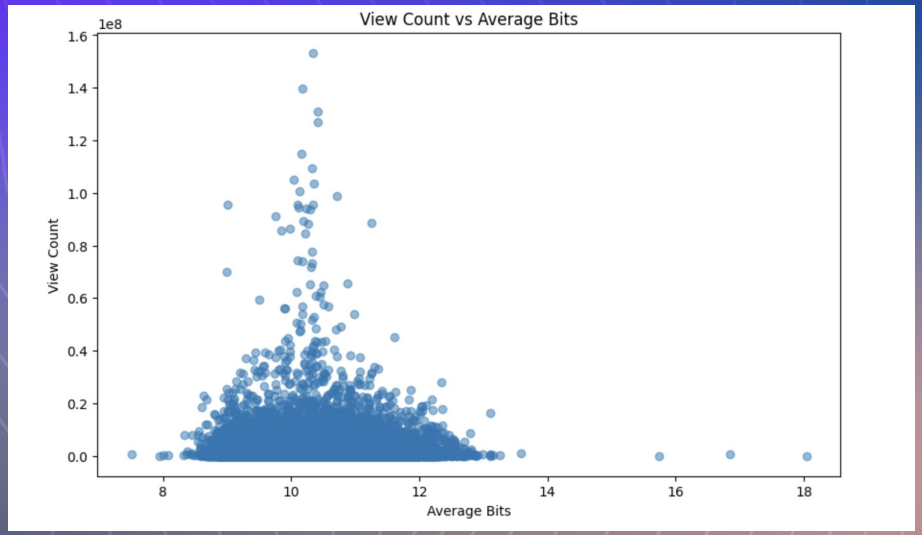


Figure 6: Average Bits vs. Dependent Variables

Selecting a model

Neural Network vs. Multiple Linear Regression

Neural Network

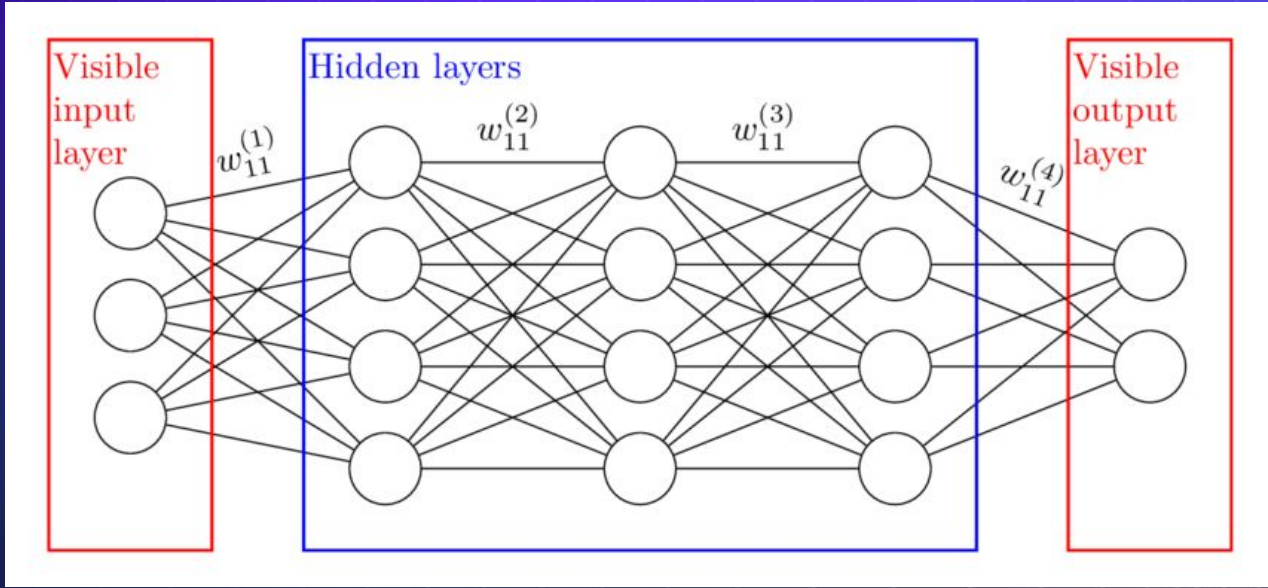


Figure 7: Logic of a neural network

- Takes a LONG time to train
- Is tolerant to noisy or incomplete data
- Works well with non-linear relationships

Multiple Linear Regression

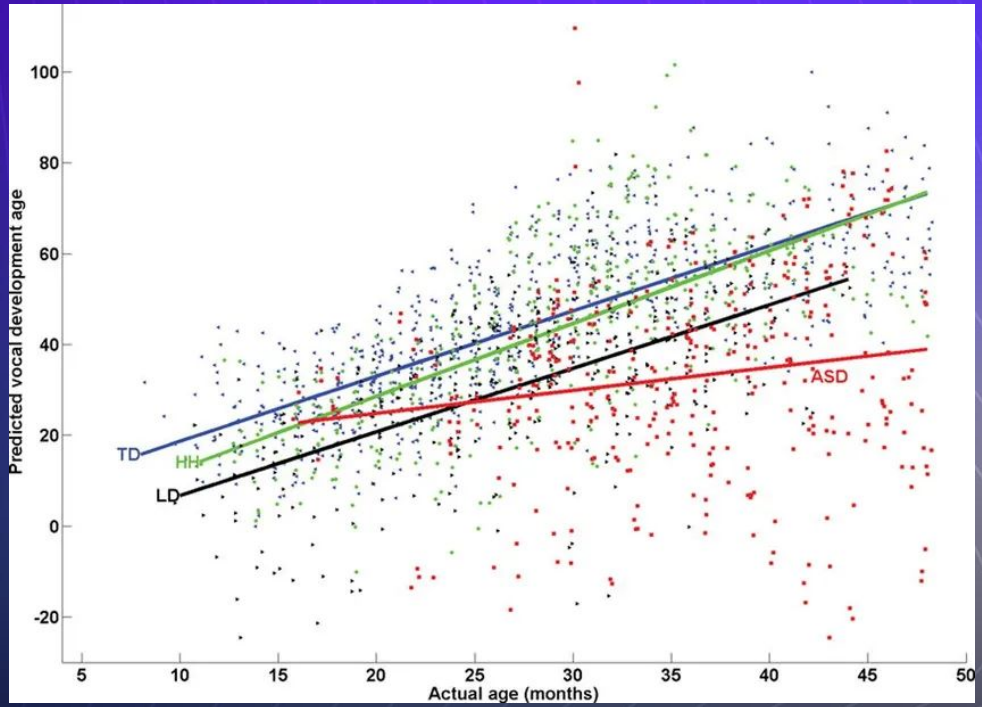


Figure 8: Linear Regression Logic

- Handles Several Independent Variables

- One of the key advantages of multiple linear regression models is their interpretability.

- used to predict the value of a dependent variable based on the values of multiple independent variables.

How the Regression Model Functions

The multiple linear regression model can be represented by the following equation:

$$Y = \beta 0 + \beta 1X1 + \beta 2X2 + \beta 3X3 + \beta 4X4 + \beta 5X5 + \beta 6X6 + \beta 7X7 + \epsilon$$

Example Calculation:

- $\beta_1(\text{Video Length}) = 0.3$
- $\beta_2(\text{Thumbnail Quality}) = 0.4$
- $\beta_3(\text{Content Type}) = 0.2$
- $\beta_4(\text{Upload Frequency}) = 0.1$
- $\beta_5(\text{Words Per Minute}) = 0.15$
- $\beta_6(\text{Bits Per Word}) = 0.25$

$$0.3 + 0.4 + 0.2 + 0.1 + 0.15 + 0.25 = 1.4$$

- **Video Length:**

$$\left(\frac{0.3}{1.4}\right) \times 100 \approx 21.43\%$$

- **Thumbnail Quality:**

$$\left(\frac{0.4}{1.4}\right) \times 100 \approx 28.57\%$$

- **Content Type:**

$$\left(\frac{0.2}{1.4}\right) \times 100 \approx 14.29\%$$

- **Upload Frequency:**

$$\left(\frac{0.1}{1.4}\right) \times 100 \approx 7.14\%$$

- **Words Per Minute:**

$$\left(\frac{0.15}{1.4}\right) \times 100 \approx 10.71\%$$

- **Bits Per Word:**

$$\left(\frac{0.25}{1.4}\right) \times 100 \approx 17.86\%$$

Figure 9: Internal Logic of Multiple Linear Regression Model

Demonstration

```
Input audio duration:
Input duration:
Input bits:
Input bit average:
Input total words:
Input audio duration:
Input words per minute:
Input bits per word:
Input bits per second:
```

```python
# This is a demonstration using hard coded values
audio_duration = 120.0
duration = 150.0
Nwords = 500
bits = 4000
bit_avg = 8.0
wpm = 150.0
bps = 32.0
bpw = 0.8
```

```
Predicted View Count: 8229166.188352078
Predicted Like Count: 62634.778979425035
```

Figure 10: Active Demonstration

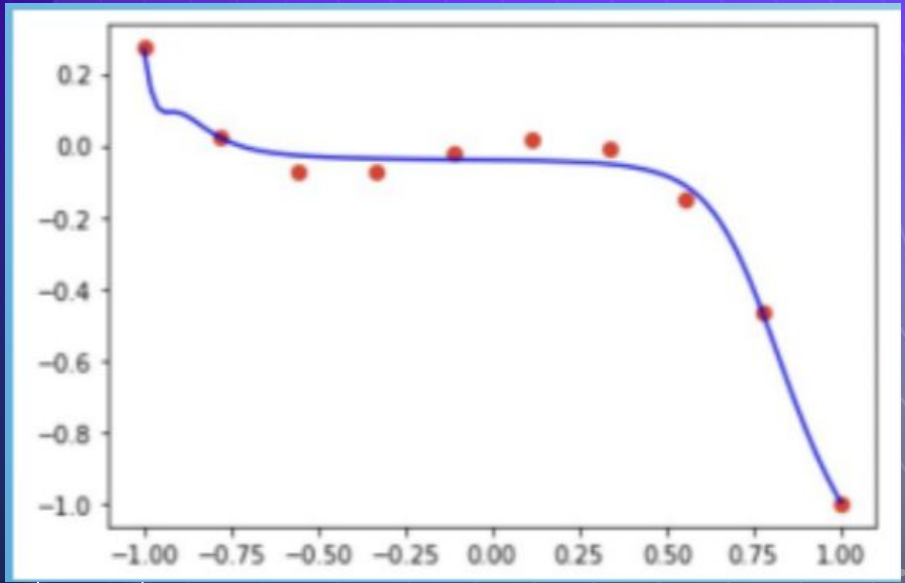
Demonstration (cont.)

```python
# Function to predict likeCount and viewCount
def predict_likes_views(audio_duration, duration, Nwords, bits, bit_avg, wpm, bps, bpw):
    # Create a numpy array with the input values
    input_data = np.array([[audio_duration, duration, Nwords, bits, bit_avg, wpm, bps, bpw]])

    # Standardize the input data using the same scaler used for training
    input_data_scaled = scaler.transform(input_data)

    # Predict viewCount and likeCount using the trained models
    predicted_viewCount = model_view.predict(input_data_scaled)
    predicted_likeCount = model_like.predict(input_data_scaled)

    return predicted_viewCount[0], predicted_likeCount[0]

# This is a demonstration using hard coded values
audio_duration = 120.0
duration = 150.0
Nwords = 500
bits = 4000
bit_avg = 8.0
wpm = 150.0
bps = 32.0
bpw = 0.8

predicted_viewCount, predicted_likeCount = predict_likes_views(audio_duration, duration, Nwords, bits
print(f'Predicted View Count: {predicted_viewCount}')
print(f'Predicted Like Count: {predicted_likeCount}')
```

```
<ipython-input-39-9eb6ee00cc09>:10: DtypeWarning: Columns (7,8) have mixed types. Specify dtype optio
e.
  data = pd.read_csv(file_path, dtype={
/lib/python3.11/site-packages/sklearn/base.py:465: UserWarning: X does not have valid feature names,
feature names
  warnings.warn(
Predicted View Count: 8229166.188352078
Predicted Like Count: 62634.778979425035
```

Figure 10: Code snippet of full demonstration

Back End Workings



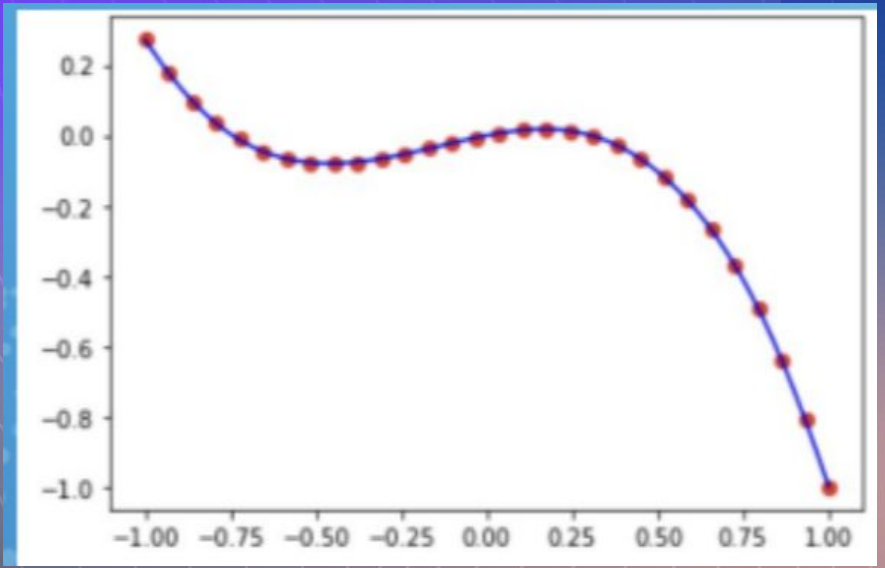


Figure 11: Fitting Visualization

THANK YOU

Special Thanks to:

Professor Jens Madsen of CCNY

Catherine Tissot & Matthew Leingang

Course Assistants

GSTEM Friends