Week 7: Second Modeling Approach

Yujia Cao, Wendi Yuan, Yuhan Zhao

After developing Random Forest Regression, XGBoost Regression, and SVM models last week, we are exploring additional approaches for analyzing YouTube video trending patterns. One adjustment we made was modifying the feature set in the XGBoost model: instead of using *days_since_published*, we incorporated TF-IDF features created earlier. However, the performance was suboptimal, partly due to duplicate feature names that we are currently addressing(Appx. 5). We are still using the same variation to keep the model consistent(Appx. 6). Given that XGBoost previously delivered strong performance with features such as engagement metrics, *trending_day_of_week*, and other key variables, we plan to retain the model from last week.

For predicting the number of days it takes for a YouTube video to trend, we selected a Neural Network model. Neural networks are well-suited for this task due to their ability to capture complex relationships between input features and the target variable. In this case, features such as views, comment_count, and other variables that exhibit nonlinear interactions are effectively modeled by the neural network.

Neural networks work by mimicking the structure of the human brain, utilizing interconnected layers of neurons that adjust their weights through training. This flexibility allows them to learn intricate patterns within the data that might not be easily captured by more traditional models like linear regression or decision trees. Given the complex nature of video trends—where various engagement metrics interact in non-obvious ways—a neural network is a suitable choice for capturing these relationships and making accurate predictions.

Last week we tried three different types of classical models which are Random Forest Regressor, the XGBoost and the SVM, and from the Metrics Result, the XGBoost and RandomForest Regressor are better to analyze the youtube statistic trend. This week besides these three models, we transfer our focus from the linear relationship throughout variables to the nonlinear relationship. Compared to Random Forest or SVM, Neural networks are well-suited for capturing complex, non-linear relationships in data.A neural network might provide an improvement on the intricate patterns between selected features and target variables. Otherwise, CNN provides the complex patterns through multiple layers.It may be useful to us to get know the potential trend when the relationship between the selected features and the target variables are not that straightforward.

The model complexity is reflected in multiple ways. With multiple layers and neurons, neural networks can model complex, non-linear relationships between features like *likes, views*, and *comment count.* This makes them suitable for capturing the nuanced factors that influence YouTube trends.Training involves forward and backpropagation, which can be computationally intensive. Adjusting hyperparameters like learning rate and epochs adds to this complexity but is necessary for optimal performance.

Hyperparameter setting based on the CNN and there are still three model variations for different hyperparameters.

Variation 1: A simple model with 1 hidden layer and 64 neurons, learning rate 0.001, batch size 32.

Variation 2: A deeper model with 2 hidden layers and 128 neurons each, learning rate 0.001, batch size 64.

Variation 3: An even deeper model with 3 hidden layers and 256 neurons each, learning rate 0.0001, batch size 128.

Training the Models: Each model is trained for 50 epochs, and the mean squared error (MSE) and R^2 score are calculated for both the training and validation (test) datasets.(Appx.1)

After running the whole process of three models, we choose two metrics to measure model performance: The MSE and R-square. Two metrics that seizes the performance of the neural network: MSE quantifies the average squared difference between the predicted and actual values. It provides a measure of how well the model's predictions align with the actual outcomes, and R^2 represents the proportion of variance in the target variable explained by the model's predictors. It ranges from 0 to 1, where higher values indicate that the model explains a significant portion of the variance.A high R^2 indicates that the model is effectively capturing the trends and relationships within the data, which is essential for understanding how various features contribute to the time it takes for a video to trend. These results were shown in Appx.2 in both training and test datasets.

From the comparison of these three variations, as we change the hyperparameters(including batch size, neuron amount and learning rate), we could observe different results. After comparing them, the best model is the neural network of Variation 2. For metrics comparison, these are their specific meanings:

Variation 1 (32 Neurons):

Training MSE(38195.26):This value indicates that the average squared difference between the predicted and actual training values is quite high. This suggests that while the model may be learning from the training data, it isn't doing exceptionally well. Test MSE(37291.30):This value shows that the model is not generalizing well to unseen data, as indicated by the slightly lower validation MSE compared

to training. Train R^2(0.0397):This means that only about 3.97% of the variance in the training target (days to trend) is explained by the model, indicating weak explanatory power. Test R^2(0.0302):with only about 3.02% of the variance in the validation target explained.

Variation 2 (64 Neurons):

Training MSE(37651.89):This value is slightly lower than Variation 1's training MSE, indicating improved performance on the training data. Test MSE(36520.55)This is the lowest validation MSE among all variations, suggesting that this model generalizes better to unseen data. Train R^2(0.0473):This shows that about 4.73% of the variance in the training target is explained by this model, which is an improvement over Variation 1. Test R^2(0.0358): About 3.58% of the variance in the validation target is explained.

Variation 3 (128 Neurons):

Training MSE(37213.45): This is the lowest training MSE among all variations, indicating that the model fits the training data exceptionally well. Test MSE(37321.47): Despite performing well on the training set, the validation MSE is higher than Variation 2, suggesting potential overfitting. Train R^2(0.0551): About 5.51% of the variance in the training target is explained, which is the highest among the Test R^2(0.0319): This shows that about 3.19% of the variance in the validation target is explained, which is lower than Variation 2, indicating a decline in generalization capability.

From these three models, if we had to choose one as the best model, the variation 2 performs most effectively(Appx. 4).Variation 2 has a validation MSE of 36520.55, which is the lowest among the three variations. This indicates it is the most accurate model in predicting the number of days for a video to trend.Improved R^2 Scores: With a validation R^2 of 0.0358, this model explains more variance in the validation target than the others, making it a better choice for predictive performance.

Even though we made a choice from these three model variations and picked the best model, we probably don’t want to use CNN as model inference. First of all, performance metrics are not good enough for us to develop our analytics goal. The R-square value is rather low around 0.03-0.05, which means only a small fraction of variance in the target variable is explained by the model, and the model cannot capture the pattern very well. The MSE values suggest that the average squared difference between the predicted and actual values is significant, and it remains high through all the variations.

Secondly, the Neural networks are often criticized for being "black boxes," making it challenging to interpret their predictions. This lack of transparency can be a significant drawback when attempting to

understand why certain videos trend or do not trend, which may limit the usefulness of the model in a practical context. It was because the speciality of the model that requires a lot of data points and the model's ability to predict accurately will be hindered if data points are not that complete. For instance, additional features such as video content type or previous trending history might provide valuable insights that are currently missing.

We also integrated BERT to generate text embeddings from video titles, descriptions, and tags in the YouTube dataset. Initially, we used the DeBERTa model but encountered compatibility issues with the SentencePiece library. To resolve this, we switched to the *bert-base-uncased* model, which avoided these dependencies. While BERT was successfully initialized, we faced an exception during tokenization and embedding extraction. After partially resolving the issue, we processed the text data in batches to manage memory efficiently, generating embeddings from the [CLS] token for each text element. These embeddings, combined with structured features such as views, likes, and comment counts, are now stored in the training and test datasets for use in future model training to predict video popularity.

In conclusion, based on the model performance outlined above, the best-performing model this week was CNN (variation 2), though it still falls short of meeting the project's objectives. Despite adjustments to variables and features incorporated into XGBoost, its performance did not meet our expectations either. Therefore, we will continue using the XGBoost model from last week, which delivered more reliable results. Additionally, the DeBERTa model presented compatibility challenges, limiting further text analysis, but we will consider addressing these issues for future research.

Appendix

Appx. 1

```python
# Define the hyperparameter variations
variations = [
    {"hidden_layers": 1, "neurons": 32, "learning_rate": 0.001, "batch_size": 32},
    {"hidden_layers": 2, "neurons": 64, "learning_rate": 0.001, "batch_size": 64},
    {"hidden_layers": 3, "neurons": 128, "learning_rate": 0.0001, "batch_size": 128},
]
```

Appx. 2

```python
results = {
    "Variation": ["Variation 1 (32 Neurons)", "Variation 2 (64 Neurons)", "Variation 3 (128 Neurons)"],
    "Train MSE": [38195.26, 37651.89, 37213.45],
    "Validation MSE": [37291.30, 36520.55, 37321.47],
    "Train R²": [0.0397, 0.0473, 0.0551],
    "Validation R²": [0.0302, 0.0358, 0.0319],
}
```

Appx. 3

	Variation	Train MSE	Validation MSE	Train R²	Validation R²
0	Variation 1 (32 Neurons)	38195.26	37291.30	0.0397	0.0302
1	Variation 2 (64 Neurons)	37651.89	36520.55	0.0473	0.0358
2	Variation 3 (128 Neurons)	37213.45	37321.47	0.0551	0.0319

Appx. 4

```
Best Model:
Variation         Variation 2 (64 Neurons)
Train MSE                        37651.89
Validation MSE                   36520.55
Train R²                           0.0473
Validation R²                      0.0358
Name: 1, dtype: object
```

Appx. 5

```
ValueError: feature_names must be unique. Duplicates found: ['10', '2017', '2018', 'album', 'amzn', 'apple', 'ava
ilable', 'baby', 'beauty', 'best', 'bit', 'black', 'bowl', 'cardi', 'cat', 'celebrity', 'challenge', 'channel',
'charlie', 'chris', 'christmas', 'com', 'comedy', 'dance', 'day', 'disney', 'diy', 'dog', 'don', 'ellen', 'entert
ainment', 'episode', 'exclusive', 'facebook', 'family', 'fashion', 'film', 'follow', 'food', 'free', 'funny', 'ga
me', 'games', 'gl', 'goo', 'google', 'highlights', 'hip', 'hollywood', 'hop', 'house', 'http', 'https', 'instagra
m', 'interview', 'iphone', 'iqid', 'itunes', 'james', 'jedi', 'jimmy', 'john', 'just', 'kids', 'know', 'late', 'l
atest', 'life', 'like', 'lil', 'links', 'list', 'live', 'll', 'lnk', 'love', 'ly', 'make', 'makeup', 'man', 'marv
el', 'movie', 'movies', 'music', 'nand', 'nba', 'nbc', 'new', 'news', 'nfacebook', 'nfollow', 'nget', 'nhttp', 'n
https', 'ni', 'night', 'ninstagram', 'nlike', 'nmusic', 'nmy', 'nsnapchat', 'nsubscribe', 'nthe', 'ntwitter', 'nw
atch', 'official', 'org', 'panther', 'patreon', 'paul', 'people', 'play', 'playlist', 'po', 'pop', 'rap', 'reac
t', 'recipe', 'records', 'red', 'review', 'school', 'science', 'season', 'series', 'smarturl', 'smith', 'social',
'song', 'soundcloud', 'sp', 'sports', 'spotify', 'st', 'star', 'subscribe', 'super', 'talk', 'team', 'televisio
n', 'test', 'time', 'today', 'trailer', 'trailers', 'trump', 'tumblr', 'tutorial', 'tv', 'twitter', 'uk', 'use',
'user', 've', 'video', 'videos', 'vlog', 'voice', 'vs', 'want', 'war', 'wars', 'watch', 'way', 'website', 'worl
d', 'wwe', 'www', 'youtu', 'youtube', 'yt']
```

Appx. 6

```python
variations = [
    {"learning_rate": 0.1, "n_estimators": 100, "max_depth": 4},
    {"learning_rate": 0.05, "n_estimators": 200, "max_depth": 6},
    {"learning_rate": 0.01, "n_estimators": 300, "max_depth": 8}
]
```