Week 8: Third Modeling Approach

Yujia Cao, Wendi Yuan, Yuhan Zhao

This week, as part of our exploration of additional modeling possibilities, we implemented Latent Dirichlet Allocation (LDA), a statistical model commonly used for topic modeling in natural language processing (NLP). LDA is particularly useful for uncovering hidden topics within a collection of documents by assuming that each document consists of a mix of topics, with each topic defined by a distribution of words. Given our dataset's text components, including video *descriptions*, *titles*, and *tags*, we aimed to leverage LDA to analyze the underlying thematic structure of YouTube content. In our case, video *titles* and *descriptions* are the main focus, as we hope to extract popular words and topics from trending videos. LDA allows us to identify prevalent topics within this content, helping us understand what themes consistently capture viewers' attention. This insight not only provides a deeper understanding of our dataset but also offers valuable information that can guide content strategy, aligning future video topics with trending themes on the platform.

The hyperparameter for the LDA is the Number of topics (n_components). This hyperparameter determines how many distinct topics LDA should find in the data. Evaluating different numbers of topics helps understand the granularity of topics extracted.To choose appropriate number of topics, it depends on several factors: the data size, the experimentation and the model interpretability. The number of topics could be a trial to find the most appropriate number of topics for the dataset and develop better results.

Here, we try three variations in the LDA model. We run three times of a different number of topic choices: 10,20 and 30. Even though the LDA is the ultimate model we plan to choose, we still want to check model performances of these three variations. (Appx.1) Based on this analysis, it would be reasonable to select the model with 10 topics. This model shows improved coherence, indicating better-defined topics, and exhibits the best performance on both training and testing data. The results from the final model suggest that with a larger number of topics, the model is better able to capture the complexities and nuances of the dataset, making it a more suitable choice for analysis. There are two main model performance metrics we used here: Coherence Score and perplexity. Coherence score measures the semantic similarity of words in a topic. Higher coherence scores indicate that the topics are more meaningful. Perplexity measures how well the model predicts a sample. Lower perplexity indicates a better fit. Another reason to select 10 topics in the LDA model is that we want to reduce the processing difficulty of the model. In our XGBoost model in the previous two weeks, we had problems with too much accuracy or

inability to produce results, so to prevent similar problems from happening again, we chose fewer topics to mosaic into the model.

When we set the topic numbers as 10, we could analyze different metrics from the result. The Training Coherence value is -725899.262579, indicating how semantically related the words within each topic are. A lower (more negative) coherence score suggests that the topics may not be very meaningful or distinct. And Training Perplexity(8761.885394) indicating how well the model predicts a sample. Lower perplexity values indicate a better fit of the model to the training data. Testing Perplexity 40805.434212 shows how well the model performs on unseen data. Higher perplexity on testing data relative to training data indicates that the model may not generalize well. When we obtain useful results from the LDA, we try to find the dominant topics in the raw dataset and see how the actual dominant titles are in both train and test datasets. According to our finding(Appx. 2), the best model holds the details of the optimal LDA model (with the best number of topics), and the Using this index from LDA , dominant topics were selected and assigned them to the train and test DataFrames, respectively. We notice that the dominant topics are different in train and test datasets. Also the Appx 2 and Appx 3 indicate that each video title is now associated with its most prominent topic as identified by the best LDA model. We select the top 10 words from topics, and it means the 10 most important words for each topic will be shown.

It is the last step before we deploy into the XGBoost model, we eagering to improve the performance of the model and enhance the comprehensive capabilities of this model, so we add labels to the dominant topics. The output shows the count of each topic's occurrences in the training dataset.

For example, *dominant_topic 2* appears 3545 times, while *dominant_topic 7* appears 3447 times. (Appx.4) We quantify the textual category, since the XGBoost and LDA models have different categories, in order to unify the results of these two models, we chose to optimize the processing and understanding of the number of topics. Since we did the XGBoost model before, we just re-discuss the complexity of the XGBoost briefly to address the model's importance. XGBoost is a powerful and flexible gradient boosting algorithm known for its ability to handle a wide range of prediction tasks, including regression, and by combining LDA-derived features with traditional numerical and categorical features (e.g., *views*, *likes, comment_count*), XGBoost can leverage both textual and numerical information.

There are three settings for hyperparameters (Appx.5). These hyperparameters were chosen to evaluate their effect on model complexity and performance. Learning Rate controls the contribution of each tree in updating the model and lower values generally lead to a more gradual learning process but may require more trees (n_estimators) for better accuracy. The n_estimators represent the number of trees

in the model. More trees generally help capture more complex patterns. We also change the number of max_depth in order to set the depth of each tree distinctively. Since we embed the LDA setting in the XGBoost model, we fit 10 topics in both the train and test dataset. To measure the accuracy of the model's performance, two metrics are appropriate: RMSE and R-squared. RMSE measures the model's error on the training data and is suitable for this regression task because it provides a direct measure of prediction error. Lower RMSE indicates better predictive accuracy.R^2 iIndicates the proportion of variance in the training data that is explained by the model.R^2 is useful for understanding how well the model explains the variation in the target variable.

Then we could observe how each variation performs by comparing metrics. Variation 1 shows moderate training performance with a balance between train and test RMSE. It suggests that the model generalizes well without overfitting. Variation 2 shows a lower RMSE for both training and test compared to Variation 1, and it also has higher R-squared value. Based on Appx. 6, the variation 3 archives the similar RMSE value and train R^2, but the test R-squared value is lower than Variation 2. This suggests that the model may be overfitting the training data, as the performance gap between training and validation has increased. From these three variations, the best model is the variation 2 with 0.05 learning rate, n_estimator is 200 and the max_depth is 6. Variation 2 provides balance between training accuracy and test performance. It has the lowest test in RMSE value , suggesting it generalizes better to new data compared to the other variations. The relatively high R^2 also means it effectively explains the variance in the target variable. The low Train RMSE (75.719178) and high Train R^2 (0.855187) indicate that the model fits the training data well, capturing the relationships between the features (including LDA topics) and the target variable.The test RMSE of 95.550469 and test R^2 of 0.764912 suggest that the model generalizes well to unseen data, maintaining high predictive accuracy on the test dataset.

This week, we had the T5 model for content generation, preparing it for future use. T5's ability to convert NLP tasks into text-generation tasks makes it ideal for our dataset. To guide relevant predictions, we used trending words by creating a new TF-IDF feature from the tags column to identify and visualize the most frequent terms (Appx. 7 & 8). The top 10 words were formatted as a comma-separated string to support tag generation.All team members encountered an error during T5 deployment (Appx. 9), causing kernel crashes. We'll consult our instructor for guidance. While this step is ahead of schedule, we're optimistic about fixing the issue to achieve our goal of generating a Q&A by project's end.Afterall, the winning model for this week is XGBoost by deploying the results from LDA model, with #2 variations combination. And we will try to find the best fit for our ultimate goal for this project.

Appendix

```
Number of Topics: 10.0
Training Coherence: -725899.2626
Training Perplexity: 8761.8854
Testing Perplexity: 40805.4342
----------------------------------------
Number of Topics: 20.0
Training Coherence: -725644.2709
Training Perplexity: 8733.9887
Testing Perplexity: 54593.4854
----------------------------------------
Number of Topics: 30.0
Training Coherence: -725736.0239
Training Perplexity: 8744.0164
Testing Perplexity: 64437.9216
----------------------------------------
```

Appx.1

```
print("\nSample of Test Dataset with Dominant Topics:")
print(test[['title', 'dominant_topic']].head())
```

```
Sample of Test Dataset with Dominant Topics:
                                                   title  dominant_topic
22112                Lucas the Spider - Musical Spider                1
2231          LaVar Ball: What did Trump do to help me?                1
44543  I Dressed According To My Zodiac Sign For A Week                9
10399                                         LOGAN PAUL                9
19141                    Bazzi - Mine (Official Video)                 6
```

Appx.2

```
print("\nSample of Train Dataset with Dominant Topics:")
print(train[['title', 'dominant_topic']].head())
```

```
Sample of Train Dataset with Dominant Topics:
                                                   title  dominant_topic
23604                  Marshmello & Anne-Marie: Friends                5
25630  Kirby Star Allies' Surprising HD Rumble Secret...                8
68698  Stephen A.: Kevin Hart 'got his feelings hurt'...                9
39559                             How to be an Aquarius                3
62877  Charlie Puth - Done For Me (feat. Kehlani) [Of...                2
```

Appx.3

```
Dominant Topic Label Counts in Train Dataset:
dominant_topic
2     3545
7     3447
18    3386
3     3339
17    3315
12    3282
19    3277
14    3264
15    3258
1     3218
9     3186
4     3171
8     3112
13    3102
6     3095
5     3091
16    3054
11    3028
0     2897
10    2825
Name: count, dtype: int64
```

Appx.4

```python
# Define hyperparameter variations
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}
]

# Assume LDA topic assignment is available as a Series
# Let's say `lda_topics_train` and `lda_topics_test` contain the do
# Example:
lda_topics_train = pd.Series([1,2,3,4,5,6,7,8,9,10])
lda_topics_test = pd.Series([1,2,3,4,5,6,7,8,9,10])
```

Appx.5

```
Comparison of XGBoost Model Variations:
     Variation  Train RMSE   Test RMSE  Train R^2  Test R^2
0  Variation 1  132.183923  137.748778   0.558682  0.511415
1  Variation 2   75.719178   95.550469   0.855187  0.764912
2  Variation 3   73.443774   99.250643   0.863760  0.746352

Best Model Variation: 2
Hyperparameters: {'learning_rate': 0.05, 'n_estimators': 200, 'max_depth': 6}
Variation     Variation 2
Train RMSE      75.719178
Test RMSE       95.550469
Train R^2        0.855187
Test R^2         0.764912
Name: 1, dtype: object
```

Appx.6

Top Trending Words:

	Trending Word	Frequency
0	youtube	36934
1	film	28598
2	fashion	26784
3	house	25494
4	food	16178
5	hop	12900
6	music	6738
7	video	6718
8	new	5575
9	funny	4617

Appx. 7

Appx.8

Appx. 9