Dcard Machine Learning Homework Report

Mao-Siang Chen

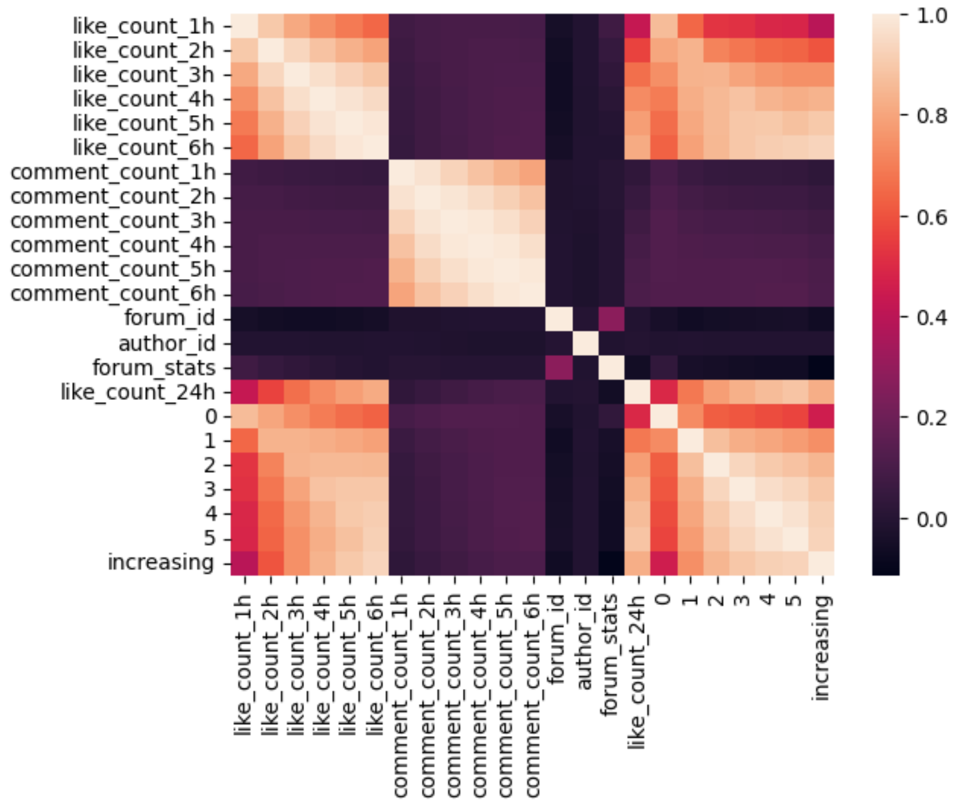
1. Introduction

In this homework assignment, I predict the like count 24 hours after a post is created on Dcard. There are 17 features, including the title, like count for the first 6 hours, comment counts for the first 6 hours, forum ID, forum statistics, author ID, and the time the post was created.

2. Methods

- Easy Data Analysis

  First, I do EDA on the training dataset to find out the correlation between the other features and the target one. This is the correlation matrix using seaborn.



As observed, the like counts of 1 to 6 hours are more correlated with the like count in 24 hours. The count of comments, forum id, author id, forum statistics, etc. do not have a direct correlation with the target feature.

- Feature Engineering
  After analyzing the distribution of the target feature, as shown in the graph below, I noticed that most of the like counts were very low. However, some posts had a relatively large number of likes, resulting in high Skewness and Kurtosis values for the data. To address this issue, I performed a log transform on all of the like count features. Following the log transform, the correlation increased, and the mean absolute percentage error (MAPE) decreased sharply from 24% to 14%.

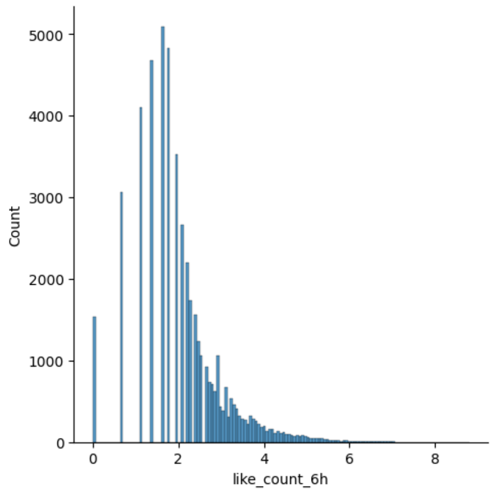


Figure 1: like count 6h after log transform

After that, I remove the outliers by removing those rows with like counts increasing sharply between the one after 6 hours and the one after 24 hours. Removing these outliers reduces the MAPE by around 1%.

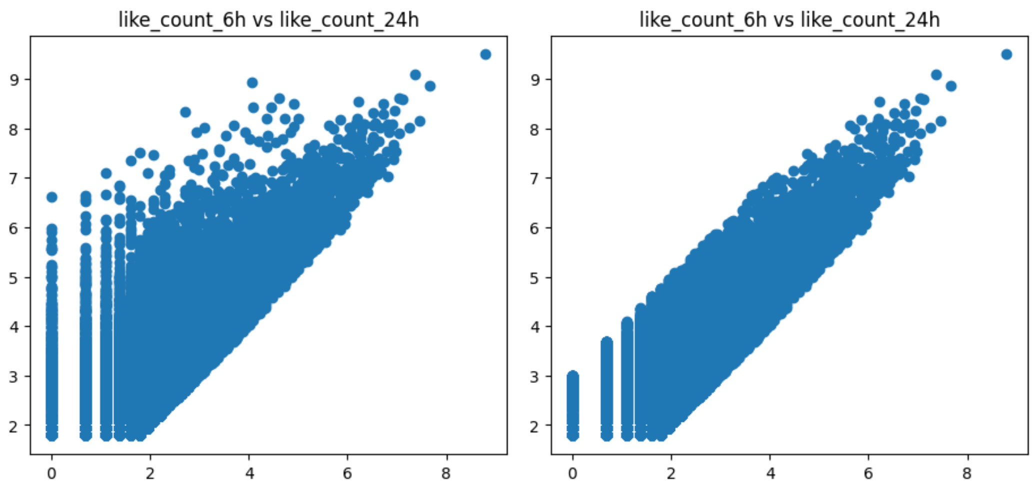


Figure 2: Before and After removing outliers

To further improve the results, I train an RNN model with the first 6 hours as training data and predict the like count after 24 hours. Then, I add the hidden layer output from the RNN model as new features for later regression.

- Model Selection
  Since the target is a regression problem, I have tried some of the well-known regression models, such as XGBoost and Gradient Boosting. They all have better performance and robustness than a simple linear regression. After that, I choose XGBoost since it achieves the best results.

	XGBoost	Gradient Boosting	Linear Regression
MAPE	13.1%	13.7%	14.04%

- Hyperparameter Tuning
  To find out the best hyperparameters for these regression models, I use the GridSearchCV from the scikit-learn package. I have tried tuning the number of estimators, learning rate, max depth, and subsample proportion.

  The best hyperparameters I have found are 300 for estimators and 0.01 for learning rate and 5 for max depth and 0.7 for subsample rate.

3. Results

- The target metric is Mean Absolute Percentage Error, and in this work, I achieve a 13.1% of MAPE.

4. Future Expectations

- Use NLP with Google Trends to improve feature engineering
- Explore further regression-related skills
- Enhance RNN performance

5. Reference

- Comprehensive data exploration with Python - https://www.kaggle.com/code/pmarcelino/comprehensive-data-exploration-with-python/notebook#1.-So...-What-can-we-expect?
- My GitHub Link: https://github.com/Mao-Siang/Dcard_Intern_Homework