



BIA 660 C
Web Mining

Fall 2022

Categorizing Kaggle Competitions by Degree of Difficulty

Group 5

*Siddharth Parag Nilakhe (CWID 20011134),
Mehul Sudhakar Ingale (CWID 20011461),
Bhanoday Sakamuri (CWID 20011279),
Preeti Reddy Koppolu (CWID 20011032).*

Supervised By
Prof. Rong Liu

Abstract

Kaggle competitions are made to offer challenges to participants at all levels of Data Science careers. These competitions offer a wide variety of difficult problems to motivated learners. Along with providing motivation, the competitive element and fixed headlines enhance coding abilities. When a new user visits the Kaggle competition page, they find it difficult to choose a competition based on their skills since they are unclear of the competitions' various levels of difficulty.

As a result, the user gets stuck and is unable to move forward, which won't help them in improving their abilities to develop creative ideas and test out different alternative theories as fast as possible. In this project, we developed a model that determines a competition's difficulty level based on several factors, including the timeline, prize, teams, tiers, and points, and predicts the most important feature that will determine the competition's success based on the number of entries.

Keywords

**eg. Topic Modelling; Text Classification; Decision Tree Classifier;
AdaBoost Classifier**

Contents

1	Introduction	4
2	Motivation and Research Question	4
3	Background and Related Work	5
4	Methodology	5
4.1	Web Scraping	5
4.2	Algorithms	7
4.3	Popularity Feature.....	10
4.4	Topic Modelling.....	11
5	Conclusion and Future work	12
6	References	12
7	Table and Signature	13

1 Introduction

Kaggle is a well-renowned platform for data science and artificial intelligence. Today, every aspiring data scientist has a Kaggle account, and every one of them takes an active part in the competitions that are held on Kaggle. There are various rewards that teams/users get when they win competitions. Some of them have cash prizes as a reward while some of them may have swags. There are various types of competitions on Kaggle - for users with different skill levels. Therefore, we will be trying to ease your task of selecting a competition based on your skill level by categorizing it into levels of difficulty. Finding out the adversity of any competition will be a very useful feature for any Kaggle user.

2 Motivation and Research Question

Our motivation to build this project:

- Offer advice to novice Kaggle users on how to select a competition depending on their level of competence.
- Provide competition creators with a description of what makes their content successful on this platform and offer them a sense of assessing how well their new competition will do among Kaggle users.

With this project, we hope to use information about competitions that was scraped from <https://www.kaggle.com> to find answers to two key questions.

Research question 1: Do the Kaggle competitions have varying levels of difficulty? How many and what elements go into these hardness levels? How many different levels of difficulty may the contests be divided into. Examine whether this classification is assisting brand-new Kaggle users in acclimating to the system.

Research question 2: Is the popularity of the organization that posts a competition the only factor in the competition's success? Or is it the product of several factors? Determine all such characteristics that make a competition successful on Kaggle.

3 Background and Related Work

In the AutoCompete paper[2], the author proposed a highly automated machine learning framework for tackling machine learning competitions. The proposed system helps in identifying data types, choosing a machine learning model and tuning hyper-parameters. They also observe that the proposed system produces better (or comparable) results with less runtime as compared to other approaches. The underlying implementation is based purely on Python and scikit-learn with some modules written in Cython

In the other paper [3], the author states that feature engineering is one of the most important and time consuming tasks in predictive analytics projects. A framework is presented for automation of feature engineering from relational databases. OneBM outperformed the state of the art DSM system in a Kaggle competition. The results show that OneBM can be useful for both data scientists and non-experts. The given framework can aid data scientists during exploration of the data and enable them to save considerable amount of time during feature engineering.

4 Methodology

4.1 Web Scraping

The dataset for this project has been scraped from <https://www.kaggle.com> competitions page. We have used selenium chrome web driver to navigate and log into Kaggle's website. Then the algorithm opens each competition page and scrapes multiple fields namely,

- name,
- description,
- price,
- launch and close timeline,
- number of teams, competitions and entries,
- tags related to the competition,
- number and size of data files

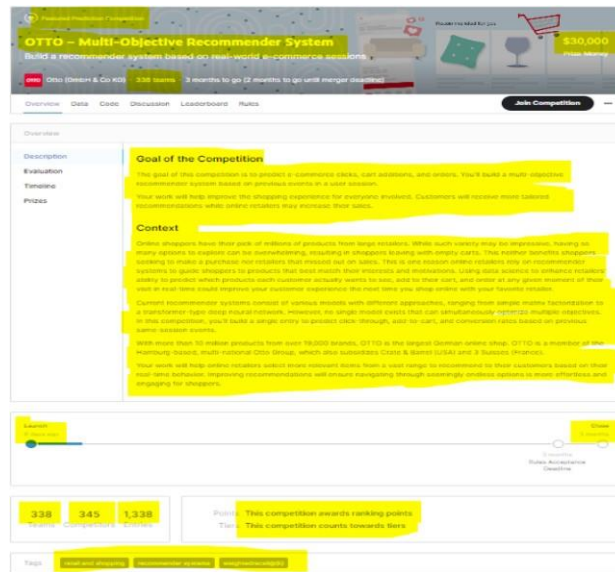


Figure 1: Scraping

Featured Predictive Competition

OTTO - Multi-Objective Recommender System

Build a recommender system based on real-world e-commerce sessions

OTTO (Otto GmbH & Co KG) · 338 teams · 3 months to go (2 months to go until merger deadline)

Overview Data Code Discussion Leaderboard Rules

Join Competition

Dataset Description

The goal of this competition is to predict e-commerce clicks, cart additions, and orders. You'll build a multi-objective recommender system based on previous events in a user session.

The training data contains full e-commerce session information. For each session in the test data, your task is to predict the `aid` values for each session type that occurs after the last timestamp `ts` in the test session. In other words, the test data contains sessions truncated by timestamp, and you are to predict what occurs after the point of truncation.

For additional background, please see the published OTTO Recommender Systems Dataset GitHub.

Files

- train.jsonl** - the training data, which contains full session data
 - `session` - the unique session id
 - `events` - the time ordered sequence of events in the session
 - `aid` - the article id (product code) of the associated event
 - `ts` - the Unix timestamp of the event
 - `type` - the event type, i.e., whether a product was clicked, added to the user's cart, or ordered during the session
- test.jsonl** - the test data, which contains truncated session data

Files

3 files

Size

11.89 GB

Type

jsonl, csv

Figure 2:

4.2 Algorithms

We have implemented four algorithms for text classification:

1. Logistic Regression
2. K Nearest Neighbors Classifier
3. Decision Tree Classifier
4. Random Forest Classifier

First, we divided the dataset into training and test data, with "difficulty" as the target variable. Then we initiated the above algorithms and fitted them with training and testing data, and we got some accuracy. Using hyperparameter tuning, we increased the accuracy and obtained some of the best parameters for each model. We then used these best parameters as input for each algorithm and obtained better accuracy than before. Finally, we evaluate the performance of each model using the classification report.

4.2.1 Logistic Regression

After using the GridSearchCV() for the logistic regression model we got the accuracy of 82.4%.

The classification report of the logistic regression model:

Classification report for Logistic Regression					
	precision	recall	f1-score	support	
1	0.95	0.75	0.84	77	
2	0.70	0.94	0.80	48	
accuracy			0.82	125	
macro avg	0.83	0.85	0.82	125	
weighted avg	0.86	0.82	0.83	125	

Figure 3: Classification Report of Logistic Regression

4.2.2 K Nearest Neighbors Classifier

First, we implemented the GridSearchCV() method, which gave us 68% accuracy. Then, after training the model with 'K' values ranging from (1,40), we observed that 25 was the best K value.

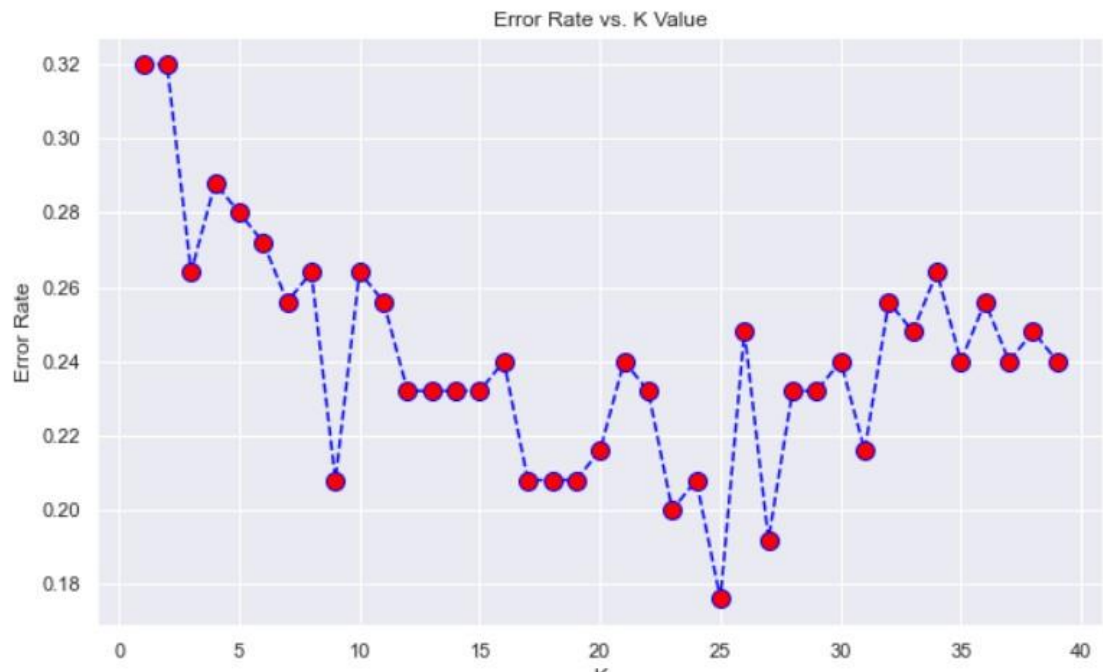


Figure 4: K value

We trained the model with K = 25 and got the accuracy of 82.4%. The classification report of the K Nearest Neighbors model:

Classification report for KNN

	precision	recall	f1-score	support
1	0.92	0.77	0.84	73
2	0.73	0.90	0.81	52
accuracy			0.82	125
macro avg	0.83	0.84	0.82	125
weighted avg	0.84	0.82	0.83	125

Figure 5: Classification report of K Nearest Neighbor

4.2.3 Decision Tree Classifier

We obtained an accuracy of 84.8% after training the model with a decision tree classifier. After that, we performed "post pruning" and "pre pruning," achieving accuracy rates of 86.4% and 83.2%, respectively. The model gives a better result after performing the post-pruning method with 86.4%.

The classification report of the Decision Tree Classifier model:

Classification report for Decision Tree				
	precision	recall	f1-score	support
1	0.90	0.83	0.87	66
2	0.83	0.90	0.86	59
accuracy			0.86	125
macro avg	0.86	0.87	0.86	125
weighted avg	0.87	0.86	0.86	125

Figure 6: Classification report of Decision Tree Classifier

4.2.4 Random Forest Classifier Classifier

After using the GridSearchCV() for the Random Forest model we got the accuracy of 88%. The classification report of the Random Forest model:

Classification report for Random Forest				
	precision	recall	f1-score	support
1	0.95	0.83	0.89	70
2	0.81	0.95	0.87	55
accuracy			0.88	125
macro avg	0.88	0.89	0.88	125
weighted avg	0.89	0.88	0.88	125

Figure 7: Classification report of Random Forest Classifier

4.3 Popularity Feature

We used the number of entries as the target feature that predicts the popularity of the successful Kaggle competition. Three algorithms—RandomForestRegressor, AdaBoostRegressor[4], and XGBRegressor—were implemented to predict popularity. We enhanced the test score and attained some of the best parameters by using the hyperparameter tuning method.

RandomForest regressor

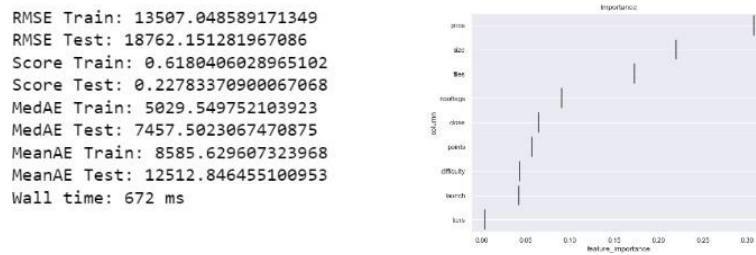


Figure 8: RandomForest Regressor

AdaBoost Regressor



Figure 9: AdaBoost Regressor

XGB Regressor

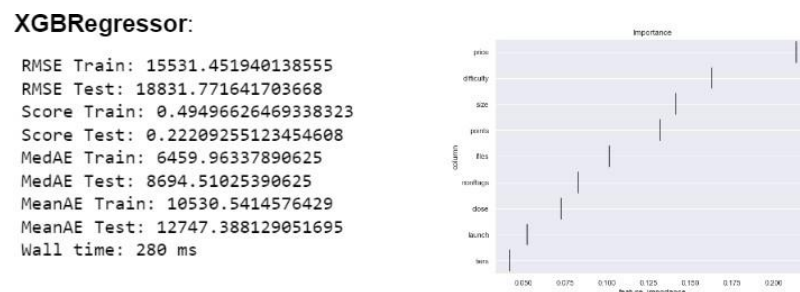


Figure 10: XGB Regressor

4.4 Topic Modelling

A machine learning technique called topic modeling examines text data automatically to identify word clusters for a collection of texts. When a text or corpus of data is being analyzed; topic modelling is used to identify the terms from the topics.

In topic modeling [5], we preprocessed the description of Kaggle competitions by removing all the English stop words. We also removed some unnecessary stopwords that appear most of the time in the description, such as “goal”, “of”, “the”, “competition”, “description”, “task”, “kaggle”, “problem”, “statement”, and “introduction”. After that we tokenized and lemmatized the description of the kaggle competition and further we used the tfvectorizer that simply counts the number of times the word appears. Each description of the competition consists of various words, and each topic can be associated with some of these words. We used the Latent Dirichlet Allocation method to find topics that the description belongs to, based on the words it contains. It is assumed that descriptions with similar topics will use a similar group of words.



Figure 11: Topic Modelling

5 Conclusion and Future work









According to the above mentioned findings, when categorizing the degree of difficulty, the `DecisionTreeClassifier()` and `RandomForestClassifier()` algorithms performed best. After applying the post pruning procedure to the decision tree, we obtained the best accuracy of 86.4%. Using the grid search technique, we were able to achieve an accuracy of 88% for the random forest classifier. We come to the conclusion that random forest classifiers is the most effective model for categorizing a competition's difficulty level. In the case of popularity prediction, the majority of the algorithms predicted that the most important factor in determining the success of a Kaggle competition is the 'price' feature. `AdaBoostRegressor` provided us with the highest test result among all the other algorithms we used after applying the random search approach.

The root mean squared error we obtained while predicting the popularity of a competition is high. We have interpreted that we should expand our dataset to contain more features that can best describe the target variable, i.e. number of entries for a competition. This is what we would want to implement in the future. We will better analyze the competitions page on <https://www.kaggle.com> and scrape more features that can explain the popularity of the competition.

6 References

1. <https://www.kaggle.com/competitions>
2. Thakur, Abhishek, and Artus Krohn-Grimberghe. "Autocompete: A framework machine learning competition." arXiv preprint arXiv:1507.02188 (2015)
3. Lam, Hoang Thanh, Johann-Michael Thiebaud, Mathieu Sinn, Bei Chen, Tiep Mai, and Ozgur Alkan. "One button machine for automating feature engineering in relational databases." arXiv preprint arXiv:1706.00327 (2017).
4. Solomatine, Dimitri P., and Durga L. Shrestha. "AdaBoost. RT: a boosting algorithm for regression problems." In 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541), vol. 2, pp. 1163-1168. IEEE, 2004..
5. Tong, Zhou, and Haiyi Zhang. "A text mining research based on LDA topic modelling." In International conference on computer science, engineering and information technology, pp. 201-210. 2016.

7 Tasks and Signature

Tasks	Assignee
1) Data Scraping	<div>   </div> Preeti, Siddharth
2) Data collection & EDA	<div>   </div> Mehul, Bhanoday
3) Account Classification Methods	<div>   </div> Preeti, Mehul
4) Hypothesis testing and Interpretation	<div>   </div> Siddharth, Bhanoday
5) Research Report Writing	All

