Medium Stories - Analysis and Predictions

CMPE 255 Data Mining

Devashish Nyati (013732342), Sweety Sojrani (013731783),
Sweta Kumari (013708201)

Introduction

Medium is one of the best places for worlds most insightful writers, masterminds, and storytellers to present the smartest takes on topics that matter. One can always find fresh thinking and unique perspectives on Medium. There are a lot of amazing articles in the field of data science, machine learning and artificial intelligence. The scope of our project is to give insights about a post before the user even publishes it. For this project, we focused on Medium articles and stories.

Our project is going to help user with the following goals:

1. Assisting users to write effective articles
2. Help users to generate tags
3. Predictive keyboard for writing articles

The objective is to implement and experiment on medium data to decide what sort of stories become prominent, predict the tags that can be associated with a post, and make predictive keyboard for the articles. We have experimented with various models, including Regression (Linear, Random Forest, Support Vector, Gradient Boosting), Recurrent Neural Network, Sequential, Multi Label K Nearest Neighbors, Naive Bayes. We have experimented different ways of extracting features from the Medium dataset. The objective metrics used include R^2 coefficient of determination, root mean squared error and mean absolute error, accuracy score, and F1 score. Using these objective metrics to evaluate the different prediction systems should translate to providing meaningful predictions to users. We have compared the results of multiple models to see what algorithm performs the best. We also did a analysis on article text to predict next possible word while writing articles.

System Design & Implementation

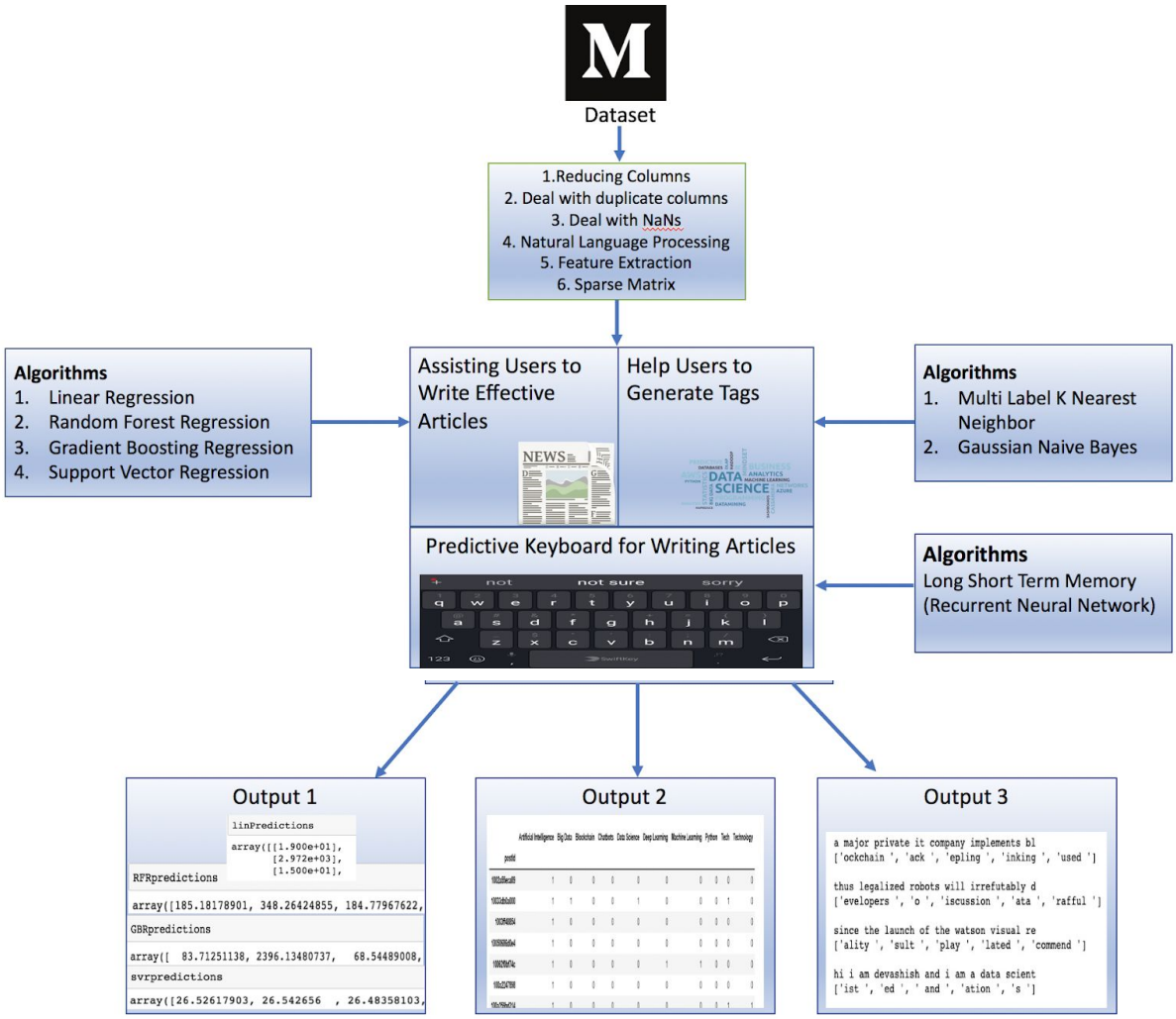
The design choice was to use algorithms that were somewhat robust to large amounts of data, since based on the exploration of the data, we noticed that many users had large amounts of article data. Another design choice was to choose algorithms that were fast and efficient for real time processing.

For assisting users to write effective articles, we have considered regression models to predict the effectiveness of a particular article. We have considered claps count as a dependent variable and title, text, author, reading time as independent variables. We experimented with multiple machine learning algorithms such as linear regression, random forest regression, gradient boosting regression, support vector regression for the prediction of articles' clap.
We use the scikit-learn library for the implementations of these machine learning algorithms. The best model for each respective articles' clap based on the appropriate evaluation metric (R^2 score and RMSE) will be assigned to the articles.

To help users to generate tags, we have considered classification model to predict tags and recommend most popular relevant tags based on the text and title. This can help the readers associate most relevant tags for their article and hence achieve optimum readership. This is performed by training a classification model based on tag_name, text, title, post_id of past articles for top 10 tag_name. We have used MLKNN (k=7) and Naive Bayes for multi label classification of post_id to tag_name(labels). We are calculating accuracy and F1 score to evaluate and compare the model for different parameters.

For predicting keyboard for writing articles, we have considered Long Short Term Memory model to predict or complete the word. From the dataset we have considered text field for this approach. We have used LSTM to examine current input and the way it was provided one step earlier. For hypertunning of the model we have taken epochs as 5, validation split as 0.05, and shuffle as true. We are calculating accuracy after each epochs to predict the best values. The details of all the experiments will be mentioned in the next section.

Workflow of our model -



Data Preprocessing & Exploration

For the dataset, we used articles from Medium which was provided on Kaggle. It consisted of CSV files that had information about the articles like text, title, reading time, number of claps, tags associated to the article, etc.

The original dataset consists of all posts tagged AI, Machine Learning, Data Science or Artificial Intelligence on Medium. There are around 280000 articles in this data set. Each article has around 50 different columns associated with it.

To preprocess the data, we followed the following steps:

1. Reducing Columns: Based on the goals, we removed the columns that are not relevant. For number of claps prediction, we have used title, text, author, reading time and total clap counts as columns. For tags we have used text, title, tag_name … For predictive keyboard, we are using text column only.

2. Dealing with Duplicate Rows:
   In our dataset, the tags are in different rows, so there are a duplicate posts. We are extracting those tags and adding them to a single row and deleting the duplicate rows.

3. Dealing with NaNs:
   Some of the fields like title and subtitle have some values as NaN, i.e. these fields are left empty. Since this is categorical data, we are removing the rows which has NaN values.

4. NLP: The text and title fields were performed filtering for removing characters and numbers, stopwords, lemmatize words for tags prediction. This cleaned text and title was then used by the model for prediction.

5. Feature Extraction: We have used Tf-idf vectorizer for text feature extraction and then applied standard scaler to scale the extracted feature.

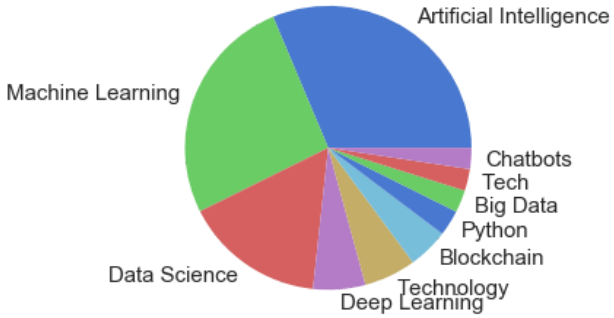
6. Splitting Strings to Columns : The raw data for tag prediction has tag_name for each post id and multiple rows for each post id for different tag_name in each row. We considered only the top 10 tag_names and added one tag name as column.

7. Get Dummies: The resultant df of tag_name for post_id from above step 6 was converted into sparse matrix for faster computation and reduce memory usage.

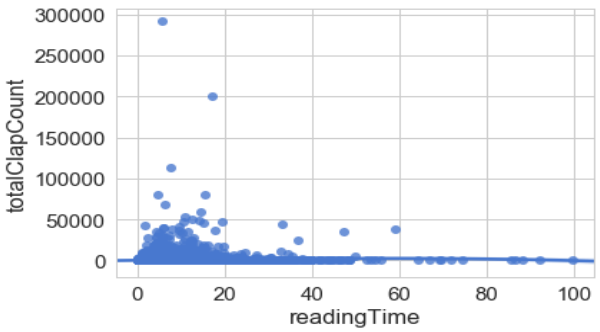
8. Creating Article Corpus: For predictive keyboard, we created an article corpus which had text from all the articles. We used this article corpus to train our model.

We did some data exploration to get a better idea of the dataset. After analyzing the dataset, we found top ten tags present in the the article dataset. Also, after going deeper into the data we found how the reading time varies with the number of claps we found that people tend to give more claps when the reading time is less.The plots of the distributions are shown below.

Top 10 Tags

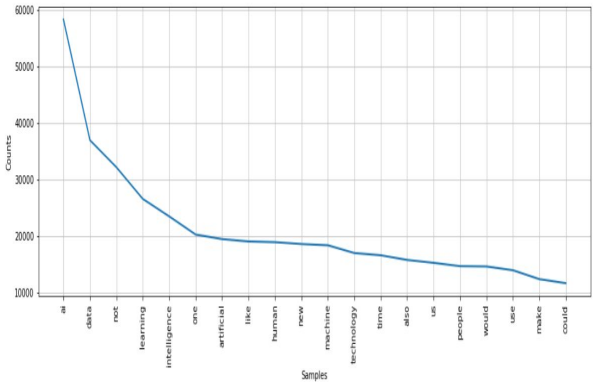


Reading Time vs Claps



Word Cloud of text

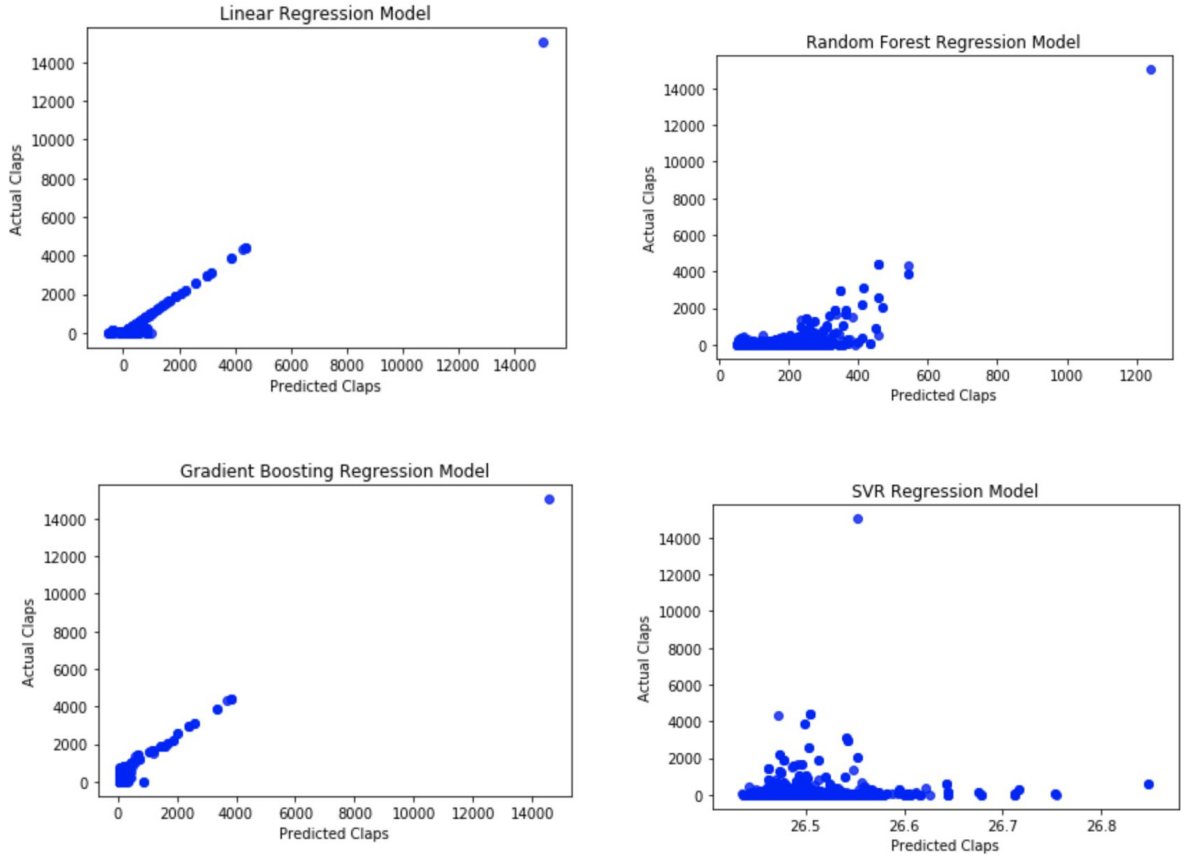
Word frequency of text



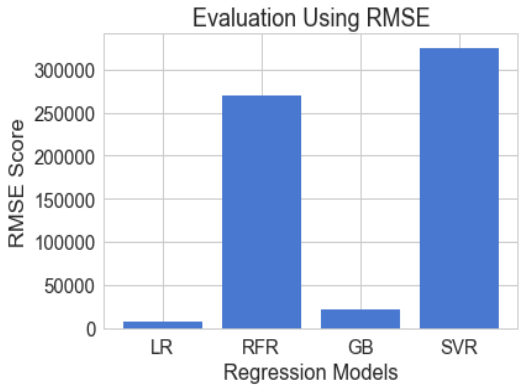
Experiments & Analysis

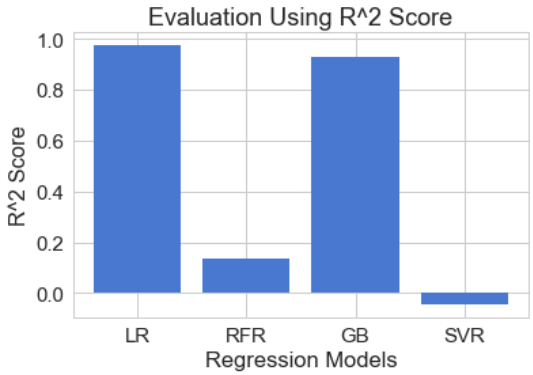
We utilized 8 columns for feature extraction of Medium articles. The columns are title attributes, text categories, author, and reading time, total clap count, language, tag name, and postId. The total clap count column is numeric, so we standard scaled it to prevent it from dominating the feature space.

As mentioned before, for regression, we kept the total clap as is. The workflow for this was to use the training set for training the models, and the hold-out validation set for selecting the best model for each user. Essentially, whichever model performed the best on the hold-out validation set was selected as the final model for the user articles. Four models per regression (Linear, Random forest, Gradient Boosting, Support Vector) were chosen. The results are shown to get an overall view each respective algorithms performance.



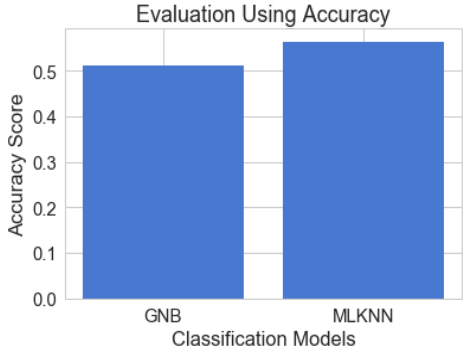
The final test evaluation result for all the regression models for predicting claps for the articles are shown below.

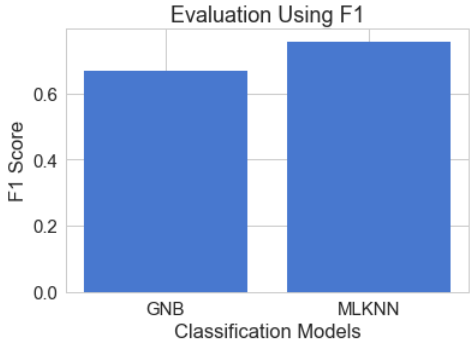




In general, it seems that gradient boosting algorithm performed the best out on the test set when compared to other regression techniques.

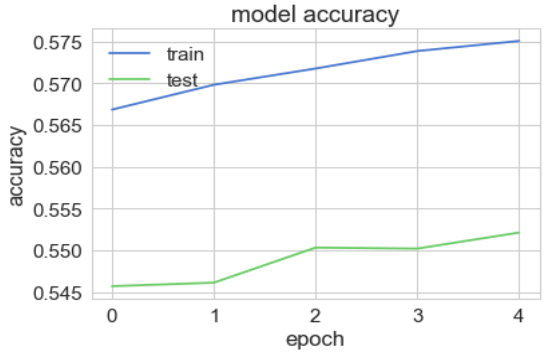
The final test evaluation result for all the classification models for predicting tags associated with articles are shown below.





It seems MLKNN(Multi-label K-nearest neighbor) performed the best out on the test set when compared to Gaussian Naive Bayes technique.

The final test evaluation result for the LSTM model for predicting keyboard for writing articles is shown below.




```
a major private it company implements bl
['ockchain ', 'ack ', 'epling ', 'inking ', 'used ']

thus legalized robots will irrefutably d
['evelopers ', 'o ', 'iscussion ', 'ata ', 'rafful ']

since the launch of the watson visual re
['ality ', 'sult ', 'play ', 'lated ', 'commend ']

hi i am devashish and i am a data scient
['ist ', 'ed ', ' and ', 'ation ', 's ']
```

It seems model predicting the best accuracy on the test set after 5 epochs.

Discussion & Conclusion

Decision Made

1. We decided to drop some of the columns from the dataset that provided no apparent value to our objective.
2. Initially, we were going to implement a hybrid system, but decided to separate each individual system to do a better analysis of the performance of the individual systems.
3. We decided to go for the regression approaches for assisting users to write effective articles, classification approaches to help users to generate tags, and neural network approach to predict keyboard for writing articles.

Difficulties Faced

1. Due large dataset, we had dependency on HPC. However, HPC was not working properly and kept disconnecting frequently.
2. The dataset had large number of class imbalanced.
3. There were multiple tags on same ID. So, we dealt with tags by creating lists and applying one hot encoding on it.

Things that worked

1. The word cloud and histogram were giving a nice zest of what is the article all about.
2. Linear regression worked smoothly on dataset although it took a long time, and comparing it with other regression algorithm.
3. LSTM model predicted best accuracy after increasing the epochs.
4. MLKNN worked better on dataset for predicting tags, and comparing it with Gaussian Naive Bayes.

Things that didn’t work well

1. We tried implementing CNN for clap prediction, however we got similar clap count for each test data. So, we decided not to implement classification algorithm for it.

Conclusion:

In conclusion, we experimented with multiple models on the Medium article dataset. We were able to help user write effective articles. Give them the insights like number of claps without even publishing the article. We were able to help them generate effective tags. Also, we were able to build a predictive keyboard to help user type articles.

PROJECT PLAN

The project had different approaches towards the goal of predicting claps, tags, and keyboard. The tasks were equally distributed amongst the team and the distribution is as follows:

Task	Devashish	Sweety	Sweta
Data Exploration / Visualization	x	x	x
Data Preprocessing	x	x	x
Algorithm - Support Vector Regression, LSTM	x		
Algorithm - Linear, Random Forest, Gradient Boosting Regression			x
Algorithm - Multilabel KNN, Gaussian Naive Bayes		x	
Report	x	x	x
Presentation	x	x	x

References

[1] Jekaterina Kokatjuhha (Nov 13, 2018) “How to build a data science project from scratch”. Retrieved from https://medium.freecodecamp.org/how-to-build-a-data-science-project-from-scratch-dc4f096a62a1.

[2] Hidekazu HoshinoItsuo TakeuchiTokio Sakauchi , 2019, “Identification Medium, Article Equipped With Identification Medium, Identifying Method And Device”. Retrieved from https://patents.google.com/patent/US20080138543A1/en