Predicting votes of short stories

Submitted By :
Alejandro Garcia
(alega695)
732A92

1 Abstract

Writing has always been an artistic expression of an author, trying to put their feelings and thoughts into a story. With the purpose of entertaining the reader, the writer tries to create an atmosphere where the reader can get involved and absorbed by the story. However, what if we could predict how entertaining or well received a story would be, just by the words the writer used, or maybe the style of writing. Is there a more popular topic than others? In this study, we will test different algorithms such as Logistic Regression, Recurrent Neural Networks and Latent Dirichlet Allocation, to try to detect which are the characteristics of a popular story.

Contents

2 Introduction

Stories have existed for centuries, we grow up listening and reading them. They can teach us an important lesson, or even transport our mind to another world. However, the purpose of the writer is to at least entertain the reader. Nowadays, the amount of storytelling available has increased geometrically with the ease of new technologies. But, can we predict the popularity of a story? That is what we will see in this study.

The data set used for this project has been extracted from the WritingPrompts subreddit page [16]. This page lets people post a title or subject as a description, which other people can use to write a short story. This data set is interesting for our goal as we have a large amount of short stories and the number of people that liked them, also used to determine the quality of them. As the summary of the story is the information given by the title, we will use that to predict the number of "upvotes" and the whole analysis.

The supervised algorithms used to classify can be separated in two types. Algorithms that only depend on a bag of words and its frequency used in the stories, and algorithms that also consider the flow of the text and how the sentences are written. The first algorithms analyzed which words have more weight for the attractiveness or entertainment of the reader, we used Logistic Regression for this type of classification. For the latter we used Recurrent Neural Networks (RNN).

Also, unsupervised topic modeling was implemented to detect the topics with better performance using Latent Dirichlet Allocation (LDA), using coherence as the metric to choose the initial number of topics to model.

3 Related work

Nowadays the use of social media has been geometrically increasing, the amount of information that these pages extract is massive, giving us a proper tool to do a behavioral analysis with machine learning techniques.

The amount of research projects done in the text of social media is quite large, for example in [11], tries to predict the different metrics on a post such as likes with the information of a post using Suppor Vector Regression SVR, also a following study is [17] that uses other algorithms to improve the first study. However, there is no academic study trying a story given the style of the text or the words used.

There is some academic work done on topic modelling, [15] tries to model a multi-party spoken discourse with unsupervised learning, however our study differs as they use a different technique than LDA.

On the other hand, there are some non academic projects that we can relate with this project, for example [3] tries to predict the amount of likes and retweets using similar techniques as we do, lacking LSTM and LDA which we include in our project.

4 Theory

In this section we present theoretical background used for this project, and the machine learning models used. The algorithms used can be separated in two types, supervised learning and unsupervised learning.

4.1 Supervised learning

The supervised learning algorithms used to classify can also be separated as we commented in section 2, algorithms depending on bag of words or algorithms depending on a sequence. However, the number of algorithms using a bag of words tested has been quite large, NaiveBayes, Random Forest Classifier, Support Vector Machine (SVM), Logistic Regression and SGDClassifier. But only the one with the best performance will be reported and explained, in this case Logistic Regression. The other algorithm used in supervised learning has been RNN, also explained below.

4.1.1 Logistic Regression

Logistic regression, this model is named after the core statistical function that it is based on, the logistic function. The logistic regression estimates the parameters of this function (coefficients), and as result it predicts the probability of presence of the characteristic of interest. This model was chosen, because provides probabilities for outcomes and a convenient probability score for observations.

4.1.2 Recurrent Neural Networks

As RNN were not part of the course, we will explain in depth how they work, and especially Long Short-Term Memory (LSTM) which are the ones we used in this project.

RNN are neural networks with loops inside that allow the information to persist. We can see how in Figure 1 this recurrent connection is adapted into an RNN allowing the network to have memory. It can be interpreted as copies of the same network, where in each time step information is passed through. Figure 1 also shows this enrollment of an RNN, the gray box indicates a layer with a sigmoid activation function. The sigmoid activation function is given in Equation 2. In the figure, the input consists of a sequence of length T. At each time step t, x_t is entered to the model and an outcome h_t is computed, this output is given back to the model as an

input for the next time step. The formula corresponding to the RNN model is given in Equation 2. In this equation, W and U are weights to regulate the input and b is the bias.

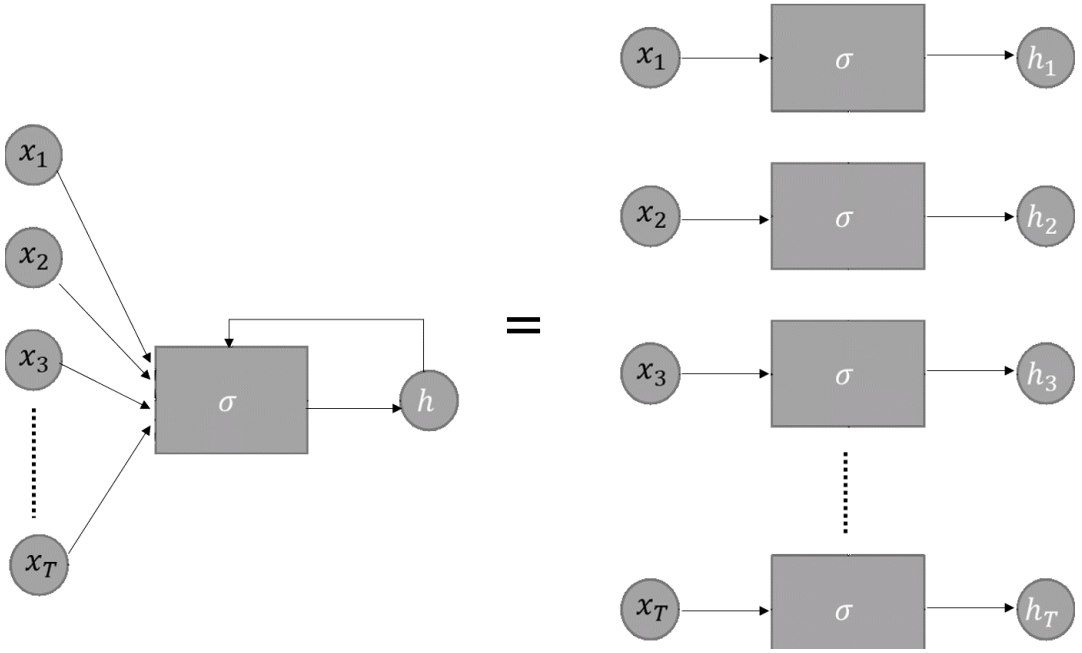

Figure 1: RNN enrolled

$$\sigma(x) = \frac{1}{1 + \exp^{-x}} \tag{1}$$

$$h_t = \sigma(W \times x_t + U \times h_{t-1} + b) \tag{2}$$

A drawback from RNN is that it is only able to use previous information, as described in [10]. But for example in the case of trying to predict the next word in the sentence:

"I live in Spain. ... I speak fluent (...).".

We need a long-term dependency to show that obviously the next word would be "Spanish". Short term memory shows that the next word is probably the name of a language, but to know which language, we have to look back earlier in the text. Long Short-Term Memory networks addres this issue of dealing with long-term dependencies.

LSTM have both short-term and long-term memory and thus are capable of handling long-term dependencies. In [19] it is described how LSTMs are used for processing

sequenced data by using gate vectors at each position to control the passing of information along the sequence. At each time step t there is a set of vectors, including an input gate i_t, a forget fate f_t, an output gate o_t, and a memory cell C_t. All these together are used to compute the output of the hidden layer h_t as follows:

$$ft = \sigma(W_f \times x_t + U_f \times h_{t-1} + b_f), \tag{3}$$

$$i_t = \sigma(W_i \times x_t + U_i \times h_{t-1} + b_i), \tag{4}$$

$$\hat{C}_t = tanh(W_C \times x_t + U_C \times h_{t-1} + b_c), \tag{5}$$

$$C_t = i_t \times \hat{C}_t + f_t \times C_{t-1}, \tag{6}$$

$$o_t = \sigma(W_o \times x_t + U_o \times h_{t-1} + b_o), \tag{7}$$

$$h_t = o_t \times tanh(C_t). \tag{8}$$

In this model, σ is the sigmoid activation function, $tanh$ the hyperbolic tangent activation function, xt the input at time t, W_i, W_C , W_f , W_o, U_i, U_C, U_f, U_o are weight matrices to regulate the input and b_i, b_C, b_f, b_o are bias vectors.

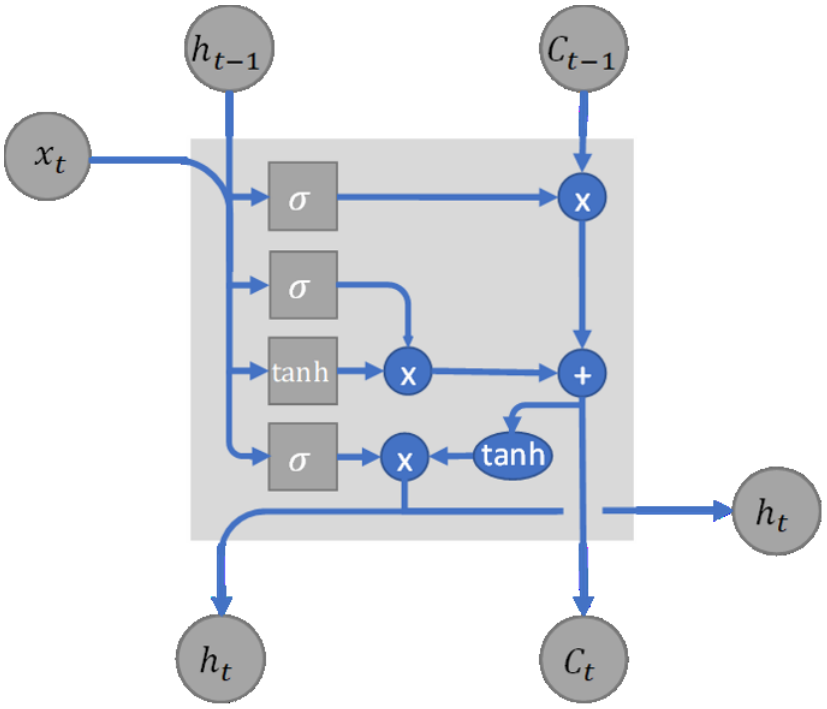

Figure 2: LSTM Network

In [14] a step by step description of the LSTM network is given, instead of having just one single network layer like the RNN, the LSTM has four interacting network layers. The structure of an LSTM network is described in Figure 2. In this figure, the gray boxes represent a neural network layer and the blue circles are pointwise operations like vector addition. The horizontal line running through the right of the diagram is the cell state. This cell state resembles a conveyor belt, it runs down the entire chain with only minor linear interactions. The cell state can also be interpreted as the memory of the network and has the ability to remove or add information by structures called gates. Figure 3 shows the four steps of the LSTM network, step A, B, C and B. Each step of the LSTM network will be discussed.

- **Step 1** First, the model needs to determine what to throw away from the cell state, this part of the network is shown in Figure 3(A) and Equation 3. This is referred to as the forget gate values f_t. The input in this step is the output of the previous step h_{t-1} and the input x_t. A sigmoid activation function is used to give output values between 0 and 1, where 0 corresponds to "let nothing through" and 1 to "remembering everything".
- **Step 2** The next step is to determine what information is going to be added to cell state, shown in Figure 3(B) and Equations 4 and 5. In this step again the inputs are $h_{t-1}$and x_t. The input layer gate it first applies a sigmoid layer over the input to determine which parts of the cell state will be updated. Then a *tanh* layer is used to create new candidate values $\hat{C}_t$. In the next step, these two will be combined to update the cell state C_t.
- **Step 3** Now the old cell state is multiplied by f_t, to forget the things that are not needed anymore and the new information is added to the cell state memory. This part of the network is shown in Figure 3(C) and Equation 6.
- **Step 4** In the final step, it is determined what the output h_t is, shown in Figure 3(D) and Equations 7 and 8. This output is based on the cell state but in a filtered version. First, a sigmoid layer is applied to the previous output h_{t-1} and input x_t, to determine the output gate values o_t. This is a value between 0 and 1 indicating which parts of the cell state are going to be output. Then the cell state C_t is transformed by a *tanh* function to get values between -1 and 1. These transformed cell state values are then multiplied by the output gate values o_t to end up with the output h_t. This output will be printed and pushed through to the next step of the network.

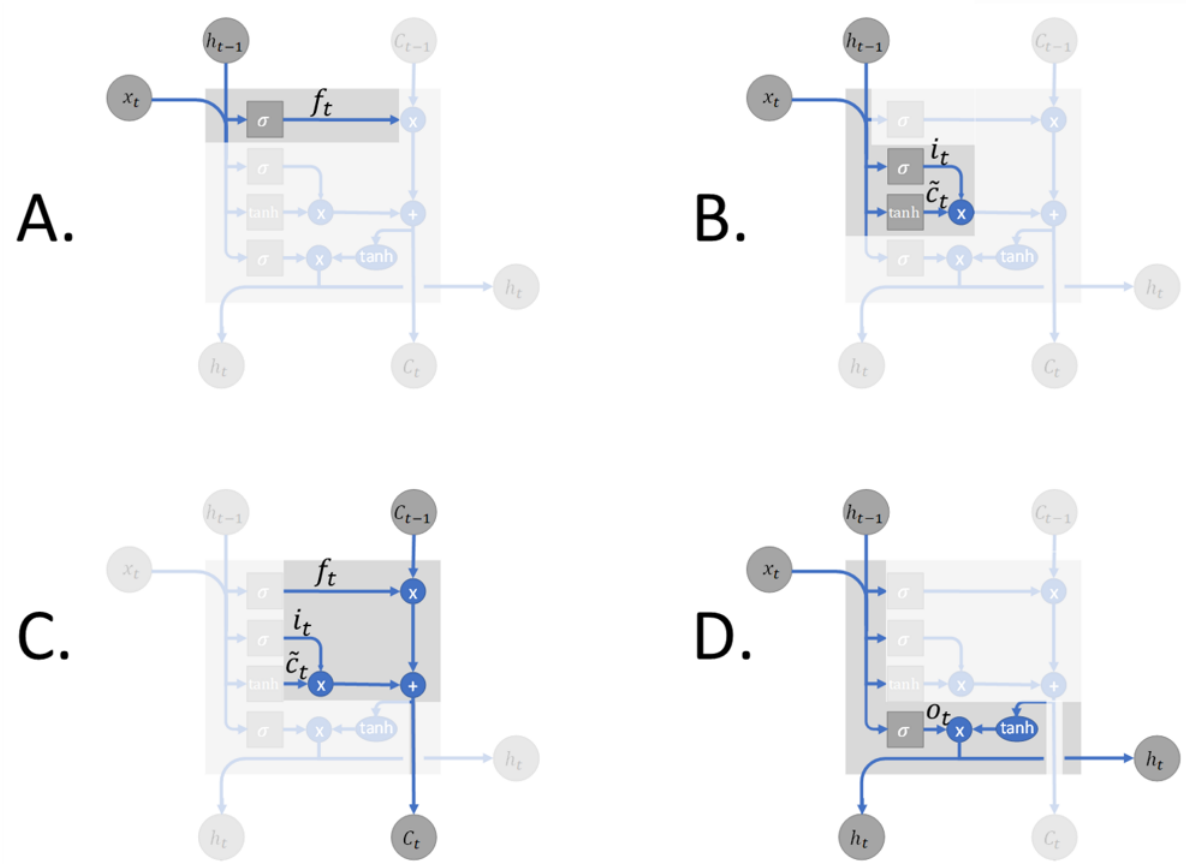

Figure 3: LSTM Network Steps

4.2 Unsupervised learning

In this part the unsupervised learning algorithm is explained. There is only one algorithm used to perform topic modeling, Latent Dirichlet Allocation (LDA), and as it was explained in the course and done in the lab we will not explain it in depth. However, it is worth mentioning that as we are using the package gensim [1] for the implementation of the LDA, the version of the algorithm in the package is the Online Variational Bayes for LDA [9].

4.2.1 Latent Dirichlet Allocation, LDA

Latent Dirichlet Allocation [7] is a Bayesian probabilistic model of text documents. It assumes a collection of K "topics." Each topic defines a multinomial distribution

over the vocabulary and is assumed to have been drawn from a Dirichlet, $\beta_k \sim$ Dirichlet(η). Given the topics, LDA assumes the following generative process for each document d. First, draw a distribution over topics $\theta_d \sim$ Dirichlet(α). Then, for each word i in the document, draw a topic index $z_{di} \in \{1, \ldots, \mathrm{K}\}$ from the topic weights $z_{di} \sim \theta_d$ and draw the observed word w_{di} from the selected topic, $w_{di} \sim \beta_{z_{di}}$. For simplicity, we assume symmetric priors on θ and β, but this assumption is easy to relax. [18].

5 Data

In this section we present the dataset used for the project, and also explain the process done to clean and prepare for further use.

The data was extracted from the subreddit WritingPrompts [16], as commented in section 2, this dataset is interesting as we have a large amount of short stories, and the number of "upvotes", which were also used as the attractiveness of the text.

5.1 Data Preprocessing

5.1.1 Data Extraction

We gathered the data via two reddit APIs, PRAW [8] and pushshift [4]. These APIs facilitates the extraction of the data, as it is possible to query and get the reddit posts as a JSON file, which can be then easily transformed into a data frame with the pandas library [2].

5.1.2 Data Cleaning

The first part of the data cleaning is extracting the posts which are more relevant for the project, as the page has a lot of irrelevant posts that do not have any comment or upvote giving us no information, for that reason those were considered as noise. When extracting the data, as it is possible to sort the posts by some feature, we sorted the posts by the score and extracted the 21000 most relevant posts during a period of time of 210 days. Also, to make it more restricted we only extracted the posts with more than one comment inside.

When getting the data, we only selected the features that we need for our classification and analysis, thus the ones that we extracted from the API are shown below as a sample in Table 1 and explained in Table 2.

Table 1: Sample of data

Title	Score	NCom	TimeUTC
[WP]Every intelligent, skilled or rich human has been evacuated from the Earth to avoid the alien invasion. This leaves the dregs, criminals and the poor behind on Earth to fend for themselves. Centuries later, the other return to "liberate" Earth only to find that they have won and prospered.	14504	404	1538208868
[WP] The zombie apocalypse has come and gone. Humanity has survived and prospered, but with the virus still inside every single human. Centuries in the future, we are at war with an alien race, and they are horrified to learn that we don't stay dead easily.	12198	310	1538240730
[WP] You die and find yourself in hell, where apparently everyone spends time to negate their sins before they go to heaven. The guy in front of you, who cheated on his wife, gets 145 years. Feeling like you led a fairly average and peaceful life, you're not worried. You get 186,292 years.	10808	761	1537956661

Table 2: Description of the data fields

Field	Description
Title	The summary of the topic to write a story about
Score	The number of "upvotes" that the title received
NCom	Number of comments inside the post, it can be considered as a kind of score
TimeUTC	The time in UTC time zone, when the post was created

Once the data set was extracted, as some of the data points shared the same titles, we removed the duplicates so we had unique data points. Also, there were some of the data points that had no score shown which would interfere in our classification, in that case we deleted the data points that contained some NA in any feature. Finally the number of data points that we used were 20000.

When analyzing the data set we detected that the histogram of the score was exponentially distributed, so it had a lot of data points with score between 0-10, but then it had a large tail. For that reason, we transformed it by computing the log of the score, so the data was more scaled and had less variance. As seen in Figure 4.

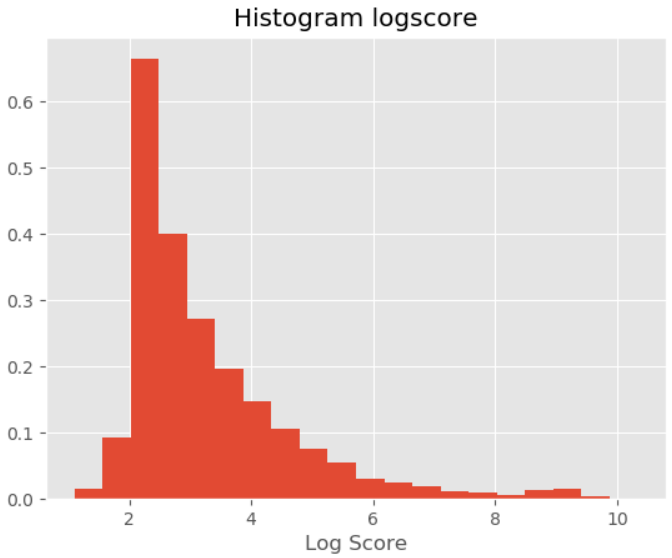

Figure 4: Histogram of log-score

5.1.3 Assignation of classes

In this study, the purpose is to detect if a story or text is entertaining or not, thus to make that classification easier, we decided to transform the data into two classes. The range was defined as following.

- 0: log of the score below the mean
- 1: log of the score above the mean

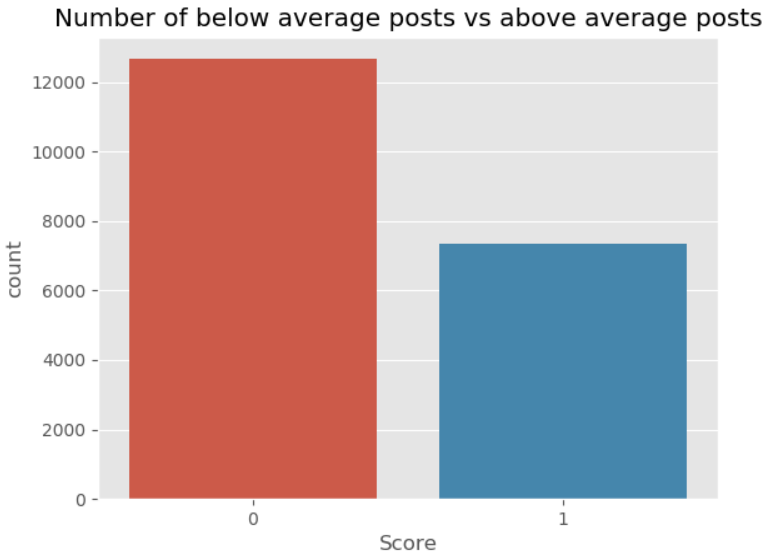

Figure 5: Count of classes

5.1.4 Bag of words

This section explains the data processing for the Logistic Regression and LDA, as it has some differences compared to the one done in RNN.

To be possible to analyze the title on each data point using those algorithms, it is needed to map each word into a number. This is necessary because normally machine learning models don't process raw text, but numerical values. To reach this we used a bag of words model (BoW) [12]. In this model it is taken into consideration the presence and often the frequency of the word, but the order or position is ignored. For the calculation of the BoW, a measure called Team Frequency, Inverse Document Frequency (TF-IDF) [6] was used. The goal is to limit the impact of the tokens (words) that occur very frequently.

Before generating the BoW, we stem the words in the titles, which corresponds to the process of reducing inflected (or sometimes derived) words to their word stem, base or root. That way the variety of words will decrease, making the training data more dense and with less input dimensions.

During the Logistic Regression classification, anther process tested was removing the stop words in the titles. After further testing, the results were worse after removing them, so we kept them. However, in the LDA we do remove the stop words as they interfere with the topic modeling, also we delete the words that do not exist in the English language as they could interfere with the classification.

5.1.5 Text Sequences

The preprocessing of the data for the LSTM is a bit different, as the LSTM has memory, it is required to maintain the order or position of the word. To be possible to analyze the titles using LSTM, something similar to a bag of words is needed, but with the order of the words also saved.

First, we use a corpus or dictionary to map the words into a number, similar to a BoW, but in this case instead of calculating a TF-IDF, we transform the text into a sequence of numbers. The result is a sequence of mapped words matrix, that is used a the input of the LSTM.

Moreover, no stemming or remove of stop words have been used in this section, as it could remove important information for the LSTM.

5.2 Data Exploration

As we can see in Figure 5 the number of texts below average is around 60% of the total data points.

Also, we will analyze the distribution of the other features and its correlation with the log-score. Figure 6 represents the number of created posts related to the score. We can see that there's a higher number of posts during 15:00-00:00. However, there is no particular time where there is a high amount of high scores, with respect to the number of posts.

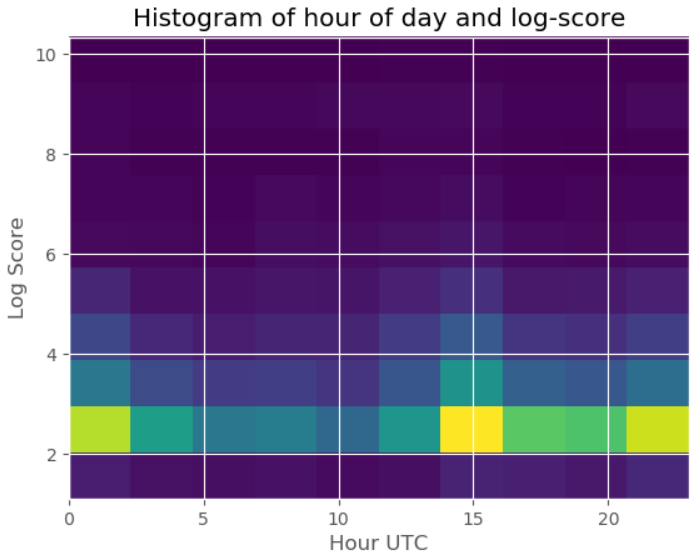

Figure 6: Histogram hour and log-score

Figure 7 shows the relation between the length of the title and its score. Even though, there's a higher score in titles with length between 20 and 75 words, it is not correlated enough to be considered as an important feature.

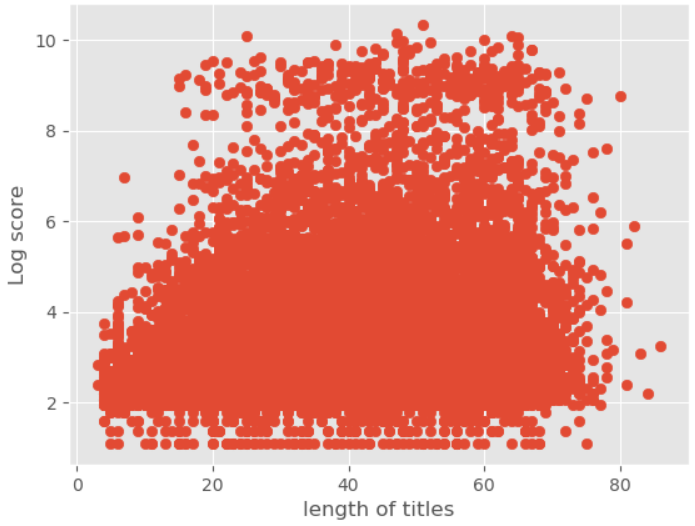

Figure 7: Relation between lenght titles and log score

The only one that shows a correlation is the number of comments in the post, however, using that feature would be cheating as it is a similar feature to the score.

Finally, we can see that there is not a relevant feature to consider when classifying, so for this project only the text was considered.

6 Method

In this section the project implementation and the details on how it was carried out is explained.

6.1 Train and test splits

The first step once the data is completely clean is to split the data set into train and test sets. Having 20000 data points, we select 80% (16000) as the training set and 20% (4000) as the test set.

6.2 Training Models

For the evaluation of the classification model and perform the prediction, we tested different models: MultinomialNB, SVM, Random Forest Classifier, Logistic Regression and LSTM. However, the ones reported will be Logistic Regression and LSTM, as they are the most relevant in the project.

The classification process followed the following steps.

- 1. Load the data (titles, score, number of comments and time created).
- 2. Filter the data points removing NA and duplicates.
- 3. Classify the features in ranges 5.1.3.
- 4. Divide data set into training and test.
- 5. Create BoW using TF-IDF for Logistic Regression 5.1.4, or sequence of words matrix for LSTM 5.1.5.
- 6. Train the model and calculate accuracy.

The topic modeling followed a similar process, but instead of separating the data into train and test, we used the whole data set to train the LDA for k topics. We checked which was the best topic number by coherence, a measure of how well the number of topics fits into the data set, with higher coherence being better (more "coherent"). After training the model, we then check which topics corresponds to each title, by using each title as input in the trained model, the output given was the number of topics and the probability of each topic in the title, we then selected the title with the highest probability to represent each title.

6.3 Model and Performance Metrics

We separated the training set into learning and validation, as it is important to use a validation set to prevent over-fitting the model. We used Cross-validation, to avoid discarding relevant data points. Cross-validation splits the training dataset into k folds, being k-1 folders used to train the model and the last one to test it. This strategy will repeat multiple times and the overall performance is the average of the computed values. To estimate the model's accuracy, we used 5-fold cross validation for Logistic Regression and 1-fold for the LSTM.

To choose the best number of topics, a metric called coherence has been used. Coherence is measured considering that each such generated topic consists of words, and the topic coherence is applied to the top N words from the topic. It is defined as the average / median of the pairwise word-similarity scores of the words in the topic [13].

The parameters used for the implementation of the Logistic Regression and LDA are shown in Table 3, these parameters have been tuned adequately.

Table 3: Tuned Parameters

Model	Parameters
Logistic Regression	penalty="l2", dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='warn', max_iter=100, multi_class="warn", verbose=0, warm_start=False, n_jobs=None
LDA	corpus=bow_corpus, num_topics=10, id2word=dictionary, passes=2, workers=2

For the LSTM architecture we got the inspiration from a Kaggle implementation to classify Spam or not Spam [5]. The main layers are seen in Table 4. It consists of an Input layer, an Embedding layer that transforms the words mapped into vectors, an LSTM layer explained in 4.1.2, another layer with an Activation function being a relu, which is basically $max\{0, x\}$, we then add a Dropout rate to prevent overfitting, and finally another layer with a sigmoid Activation function.

Table 4: LSTM Layers and Parameters

Layer	Parameters
Input	shape = [86], name='inputs'
Embedding	input_dim=12000,output_dim=8,input_length=86
LSTM	units=128
Dense	units=256,name='FC1'
Activation	activation='relu'
Dropout	rate=0.5
Dense	units= 1,name='out_layer'
Activation	activation='sigmoid'

6.4 Challenges

During the implementation of the code, the biggest challenge was to get the best accuracy. To reach an acceptable value, it was necessary to reiterate over the same code several times and try to understand what were the factors that increase or decrease this metric.

When doing the class split, as we had to transform the data into two categories, it was hard to select which would be the threshold for the best split. Finally, we decided to choose the mean of the log-score, as it was a reasonable measure when splitting the data points.

Also, when implementing the LDA, it was a challenge to select the number of topics to cluster the data, as explained before we chose the coherence metric. However, another metric called perplexity was also contemplated, but after further research we decided that coherence was a better choice.

7 Results

The table 5 describe the accuracy values we reached with the proposed models.

Table 5: Accuracy of score(%)

Model Name	Accuracy
Logistic Regression	61.77
LSTM	63.375

We can also see the confusion matrix of the models, as it can be useful to know the distribution of the classifications.

Table 6: Confusion Matrix Logistic Regression

	0	1
0	2259	306
1	1189	246

Table 7: Confusion Matrix LSTM

	0	1
0	2249	303
1	1162	286

Given the results shown, there is no better model to predict the score in a title, the accuracy values are quite similar, even though LSTM has almost a 2% higher accuracy, we can see in the confusion matrix that they have similar results. The amount of titles wrongly classified as below average is too high, as seen in Table 6 and Table 7 showing that the model will rather classify a title as below average than above.

The topics most frequent words are shown in the tables below, with the number of topics, chosen by the higher coherence metric being k =10.

Table 8: Topic 1

Topic 1	
Term	Frequency
human	2.3
alien	1.5
earth	1.4
find	1.1
door	0.8
come	0.8
first	0.7
day	0.7
race	0.7
war	0.7

Table 9: Topic 2

Topic 2	
Term	Frequency
find	2.2
know	1.8
get	1.0
day	0.9
never	0.8
hous	0.7
way	0.7
littl	0.7
come	0.7
world	0.7

Table 10: Topic 3

Topic 3	
Term	Frequency
people	1.6
world	1.3
live	1.2
see	1.1
everyon	1.0
day	1.0
earth	1.0
life	0.9
back	0.8
end	0.8

Table 11: Topic 4

Topic 4	
Term	Frequency
time	3.9
travel	1.5
die	1.3
life	1.2
day	1.2
back	1.0
peopl	1.0
last	0.9
death	0.9
like	0.9

Table 12: Topic 5

Topic 5	
Term	Frequency
peopl	1.7
human	1.2
world	1.1
wish	1.0
good	1.0
day	1.0
turn	1.0
technolog	0.9
everi	0.8
go	0.9

Table 13: Topic 6

Topic 6	
Term	Frequency
world	3.0
magic	1.9
power	1.5
day	1.3
super	0.8
get	0.7
old	0.7
tri	0.7
become	0.7
use	0.6

Table 14: Topic 7

Topic 7	
Term	Frequency
stori	2.7
write	2.5
book	1.1
earth	0.9
day	0.8
come	0.8
dragon	0.8
first	0.8
planet	0.7
univers	0.6

Table 15: Topic 8

Topic 8	
Term	Frequency
day	2.4
time	1.6
everi	1.4
see	1.0
dog	0.9
decid	0.9
world	0.9
someon	0.8
come	0.8
walk	0.8

Table 16: Topic 9

Topic 9	
Term	Frequency
life	2.2
get	2.1
game	1.2
make	1.1
hell	0.9
world	0.9
human	0.8
earth	0.7
dark	0.7
time	0.7

Table 17: Topic 10

Topic 10	
Term	Frequency
person	2.0
world	1.9
day	1.6
time	1.6
everi	0.8
abil	0.8
discov	0.8
live	0.8
peopl	0.8
two	0.8

We can see that there are some interesting topics.

- Topic 1 may refer to stories related with aliens and a war of races.
- Topic 4 may refer to stories related with time travel.
- Topic 6 is related with magic and super powers.
- Topic 7 can be related to fantastic stories with creatures like dragons.

Unfortunately, we have not been able to interpret properly the other Topics. A further analysis would be necessary in order to understand them properly.

In addition, as we can see in Figure 8, the histogram of the topics and the log-score tell us that the number of stories written in Topic 1 and Topic 4 are higher, but there's not a relevant difference in the distribution of the score.

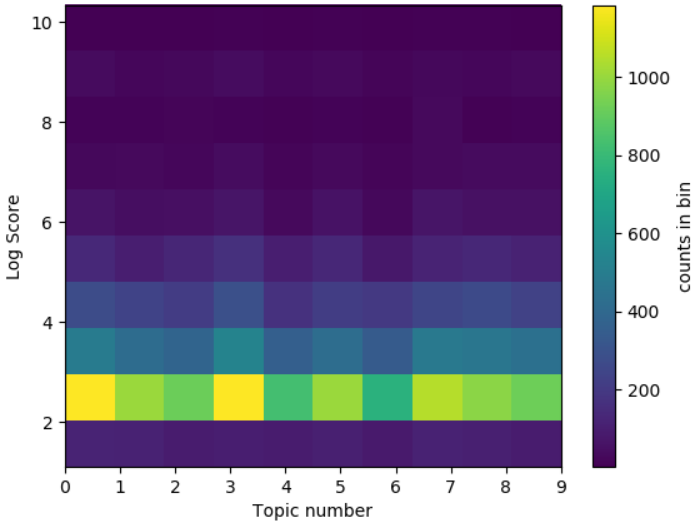

Figure 8: Histogram of Topics and Log-score

8 Discussion

The main idea of this project was to analyze the characteristics of an entertaining text or story. When we first started the project, the hypothesis was that the way the text was written would affect on the score the text. However, as we can see in the results section 7, there's no difference when classifying with a bag of words, rather than with the sequence of the text. So, apparently the style of the text does not make it more appealing for the writer.

Moreover, the predictions made are quite disappointing, we could not predict with good accuracy if a title would be popular or not, meaning that there's something more apart from the words and the style, that we are not taking into consideration. It could be that there are some authors getting a higher score, the popularity of the topic given a time point...

Nevertheless, we can see that there are some topics that are more popular than others. For example Topic 1 about alien wars and Topic 3 about time travels, receive a higher number of posts. However, that means that they are more popular, not that they are more entertaining.

9 Conclusions

Given the results and discussion presented during this project, we can conclude that the style of writing or the words used in a text, are not enough features to predict how entertaining a story will be. And that there must be other features not considered in this project important for that same goal. Also, there are some topics that gather more attention (number of posts) than others, but that does not imply a better repercussion on the score.

9.1 Further Improvements

In this section we propose some additional improvements to the project.

- Adding more information to the classification problem, such as authors, the people reading it...
- Implement a better separation for the score of the titles, as it may be one of the reasons why the classification has a poor accuracy.
- Use a grid to optimize the parameters in the algorithms used, Logistic Regression, SVM... Even though we already did some.
- Implementing a deeper RNN with more layers.
- Implementing a Convolutional Neural Network which may improve the accuracy when adding it to the LSTM.
- Further analysis in the Topic Modeling study, trying another algorithms such as Latent semantic analysis (LSA) or Non-negative Matrix Factorization (NMF).
- Trying another methods like sentiment analysis and see if there's a correlation with the score.

References

[1] Gensim: Topic modelling for humans. https://radimrehurek.com/gensim/.

[2] Pandas: powerful python data analysis toolkit. https://github.com/pandas-dev/pandas.

[3] Predicting article retweets and likes based on the title using machine learning. https://github.com/flaviohenriquecbc/machine-learning-capstone-project/blob/master/final-report.pdf.

[4] pushshift.io. https://pushshift.io/.

[5] Simple lstm for text classification. https://www.kaggle.com/kredy10/simple-lstm-for-text-classification.

[6] Ricardo Baeza-Yates, Berthier de Araújo Neto Ribeiro, et al. *Modern information retrieval*. New York: ACM Press; Harlow, England: Addison-Wesley,, 2011.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[8] Bryce Boe. Praw the python reddit api wrapper. https://praw.readthedocs.io/en/latest/index.html.

[9] Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[10] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[11] Sérgio Moro, Paulo Rita, and Bernardo Vala. Predicting social media performance metrics and evaluation of the impact on brand building: A data mining approach. *Journal of Business Research*, 69(9):3341–3351, 2016.

[12] Un Yong Nahm and Raymond J Mooney. Text mining with information extraction. In *Proceedings of the AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 60–67. Stanford CA, 2002.

[13] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

[14] Christopher Olah. Understanding lstm networks. 2015.

[15] Matthew Purver, Thomas L Griffiths, Konrad P Körding, and Joshua B Tenenbaum. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 17–24. Association for Computational Linguistics, 2006.

[16] Reddit. Writingprompts, 2019. https://www.reddit.com/r/WritingPrompts/.

[17] Emmanuel Sam, Sergey Yarushev, Sebastián Basterrech, and Alexey Averkin. Prediction of facebook post metrics using machine learning. *arXiv preprint arXiv:1805.05579*, 2018.

[18] Hanna M Wallach, David M Mimno, and Andrew McCallum. Rethinking lda: Why priors matter. In *Advances in neural information processing systems*, pages 1973–1981, 2009.

[19] Shuohang Wang and Jing Jiang. Learning natural language inference with lstm. *arXiv preprint arXiv:1512.08849*, 2015.