Data mining project - Kaggle "Stumble Upon Evergreen Classification Challenge"

Students: Dor Bank 301380416, XXXXXXXXXXXXXXXXXXXXX

# Executive summary

The data consists 26 features and 2 classes (labeled as 0,1). This report goes through the CRSIP-DM steps, where each section is written as the final product after 2-3 iterations of the process. At each "notebook" related part, there will be a [cell xxx] pointing to the relevant cell in the notebook. All the data exploration and feature engineering mostly visual based. The modeling consists of 2 parts: The first is fitted on the tf-idf of the "boilerplate" and added as a feature. For this we used logistic regression and SVM. The second is fitted on the features, and includes Random forest and SVM.

# Data Exploration

In this part we will elaborate on the logic behind some of the preprocessing procedures in next section.

We first plot all numeric features histograms and summary. From that we remove outliars and engineer some of them. All features have logical "summary data" and distribution (such as normal/exponential for example), except 3: embed ratio, compression ratio, image ratio (histograms in appendix).

embed ratio - all the values are either exactly -1, or very small positive. Therefore, it seems best to only separate between the 2 cases.

compression ratio - this feature has a "normal like" distribution in the range (-1.5,1.5), and there is an extra value of 21 which seems irrelevant. An appropriate way of dealing with it will include differentiating between the '21's and the rest, and then to somehow treat this feature as a normally distributed one.

image ratio - this feature distributes mostly with small values, and has some extreme outliers.

In addition, the "boilerplate" feature is a text file which probably contain a lot of data. it can be split to 4: title, body, related, url, . Unfortunately, those fields possess many missing values, which would be to inefficient to complete.

An important dependency we came across is between the features "is news" and "is front page news". Notice that if a sample is a front page news, than it is probably news, and if it isn't news, it is probably not front page news.

# Pre processing

This part (and the next) was done at the whole data. It has a total of xxxxxx samples and 26 features, was split to train and test right before the data modeling.

1. Data types: all numerical features were modified to floats (even the binary). This was done because the integer doesn't have an advantage over the float, and it helped us with our missing values treatment strategy.
2. Missing values: 4 features had missing values -
   1. Alchemy category: total of xxxx missing values. was filled as an "unknown" category. we should mention that this category already existed in this feature by 6 samples.
   2. Alchemy score: total of xxx missing values. was filled by the median to be "more immune" to outliers.
   3. Is news: total of xxxx missing values. filled as 1 if "is front page news" is 1, and average otherwise (median would be pointless for binary features).
   4. Is front page news: total of xxxx missing values. filled as 0 if "is news" is 0, and average otherwise (median would be pointless for binary features).
3. Outlier removal: done visually. Main influence can be seen at "image ratio".
4. Feature Engineering (explanations in EDA section):
   1. embed ratio - (-1) was mapped to 0, and all others mapped to 1.
   2. compression ratio - created an extra feaure "is high comression" which is 1 if the "comression ratio" is 21, and 0 otherwise. then, all 21 values were reduced to the second highest value.
   3. boilerplate - first done tf-idf on all the data. we got 238000 features. adding 25 features didn't seem relevant. Instead, we modeled by it (as explained at the modeling part) and added the prediction result as an extra feature.
5. Normalization: We wanted all values to be between 0 and 1. Therefore, we used the formula of . This way, if we turn the categorical feature to dummy variables with one hot encoding, the impact of different categories would be maximal.

# Feature Selection

our first model will be random forest, as elaborated in next section. Therefore, a natural feature selection method is the trees feature importance.

this resulted with:

# Data Modeling

Our model consist of 2 "sub models". The outer one is the model as we all know. The inner one, is the model fit for predicting the boilerplate tf-idf, and getting the "boilerplate prediction" feature.

For our first attempt we wanted to use the simplest model we know - logistic regression. This will be used for the inner model. The outer one will be, as stated before, random forest. This method is used to ease the feature selection part (which relies on tree model), to conveniently use the categorical feature of "alchemy category", and to "break" the inner model linearity.

hyper parameters:.....

results:

Train AUC:

Test AUC:

For the next one

# Summary