# Predicting News Popularity Competition Report

*Anneke Speijers, Harihara Subramanyam Sreenivasan, Max Elias van Esso*

## Objective

The objective of this competition was to build a model that can predict the popularity of an article on the website mashable.com based on the data available to us.

## *Analysis of the dataset*

There is a substantial class imbalance in the dataset. The percentage of the classes in the training set are as follows:

```
Popularity level 1 -  31.59%
Popularity level 2 -  45.88%
Popularity level 3 -  19.04%
Popularity level 4 -  3.33%
Popularity level 1 -  0.157%
```

This raises a significant challenge as the models will frequently predict classes 1 - 3 and will be unable to accurately predict classes 4 & 5.

We have 59 features, out of which most are continuous variables and there are 2 groups of categorical variables:

1) There are a set of categorical variables which are used to denote the data channel of the article. Each feature belongs to atmost 1 data channel. There are a few articles which belong to none of the data classes. In the training dataset, there are 4,658 articles without a designated data channel and in the test set there are 1476. There are a total of 6 channels which are as follows:

    1. Lifestyle
    2. Entertainment
    3. Business
    4. Social Media
    5. Technology
    6. World

2) The other set of categorical variables are used to denote the day of the week on which the article was released. In additional to these 7 variables corresponding to each day, there is a categorical variable "is_weekend" which has a value of 1 if the article was released on a weekend and 0 if it was released on a weekday.

The continous variables can be grouped the following ways

1) Number of images and videos in the articles.
2) Number of references to the article.
3) Number of shares of the articles referenced by the article being studied.
4) Analyses of the number of words in the article and title of article.

5) Analyses of the keywords present in the article
6) LDA00 to LDA04 : Closeness to the most relevant topics at the time of release.
7) Sentiment analyses of the title and the articles

- LDA topics: Latent Dirichlet Allocation was used to determine the top 5 relevant topics at the time of the release of the article. They are indexed from 0 to 4 and each of the variables has a score ranging from 0 to 1.

In this report, when the term "Original Features" is used, it refers to the above mentioned features which were obtained from the datasets.

# Approach to Model Building

The first step that we took was to randomly sample data from our training dataset without replacement to obtain a 70:30 split. The reason we did this is that all our cross-validation and model building could be done on 70 percent of our data and the testing for accuracy could be done the 30 percent.

This allowed us to develop a reasonable expectation of model performance on the final test data and provide a standard method of comparison for the performance of various models.

The code used for this step is stored in ..... .R

## Feature Transformations:

The following feature transformations have been done to evaluate their predictive capability. The following operations have been applied to x_{i} for all i that are continuous variables.

1) $x^2$
2) $x^3$
3) $x^2 - x$
4) $x^3 - x$
5) $x^3 - x^2$
6) $log(x)$
7) $(x - mean(x))/sd(x)

## Feature Extraction

Feature Extraction has been done in 3 ways:

1) The categorical variables:

Intuitively, it can be concluded that there is an impact of the data channel of the article and the day of the week it was released in predicting the popularity. This was examined with the following steps:

```
In these steps, x_{i} represents the feature(i.e. Data Channel or Weekday) and y_{j} represents the popu

1. We tabulated the x_{i} and y{j} such that each element of the table represented a combination of (i,j
2. We then divided all the elements with (summation over j) for all i.
3. We then weighted these ratios according to the values of j
4. The total for each i was calculated.
```

```
5. These totals were ordered and two new features were generated.
6. One feature was on a scale 1 to i, such that the highest total gets i and the lowest total gets 1.
7. The second feature was on a scale of -i/2 to i/2, the lower half of the totals having negative values

The reason for the positive and negative features is that we wanted to examine if there is:
1) A varying amount of positive effect of being released in a certain data channel/day.
2) A negative effect of being released in some data channels/days and positive effect on being released
```

2) ANOVA interaction variables:

3) xGboost linear regression:

The problem that we are dealing with a multi-class classification. Intuitively, we have a reason to believe that this is not a completely categorical response. Our popularity classes are ordinal. If a new set of features predicts with a high ampunt of probability that an article is in class 3, then there is higher possibility of it belonging to class 4 in comparison to class 2.

Keeping this in mind, we decided to use the output classes as numeric values and generate fitted values by running a linear regression.

We considered several models for this section of feature selection but we decided to use xGboost. It provides us with atleast as many/more parameters for tuning in comparison to other models. In addition, it had a lower RMSE in comparison based on a few initial tests done to determine the model.

The following procedure involves using a grid in order to generate fitted values by performing linear regression on the output. The steps executed are as follows:

1) We determine a set of parameters to use for the XGboost function.

2) The error function being minimized is RMSE and the objective is linear regression.
3) We iteratively generate the predictions for each of the parameter combinations of xGBoost for the parameter combinations.
4) The predicitons generated in step 3 are for both the training and the test data.

Once the whole loop has been executed we will have a total of l columns. Here l is the product of the number of values in each parameter that has been tuned. In our particular code, l was 1080.

# Feature Selection

This was done in 3 ways as well:

## 1) Manual Selection:

This procedure was done in the beginning of the project, various combinations of features were taken and their effects on the accuracy was measured using coefficient values from cv.glmnet and the prediction accuracy from cv.glmnet and multimnom functions.

This was done for the original features, scaled features and log features.

This was a suboptimal procedure mainly because it is extremely time-consuming. The advantage of this method was a very well structured uderstanding of the impacts of the variables and their transformations on accuracy. A couple of clear winner variables were: is_weekend and kw_avg_avg.

## 2) Using accuracy calculating functions and multinom.

Three accuracy functions were built and they worked in the following manner:

### 1) Accuracy Function

1) The training and test datasets are given as input to the function.
2) The function then performs 12 possible transformations to a feature and stores all the resulting accuracies from a multinom function into a vector.
3) If the accuracy of a transformation is better than the accuracy of a model with just the input matrices, it stores the transformation to be applied, else it stores "baseline".
4) It executes this in a loop for all the features and returns the values to be applied to each variable.

The disdvantage with this function was that it doesnt take into account the effects of transformation a feature on the other features. It assumes at each step that only the feature being examined will be transformed. The results were not significant.

### 2) Accuracy Function v2.R

1) The training and test datasets are given as input to the function.
2) The function then performs 12 possible transformations to a feature and stores all the resulting accuracies from a multinom function into a vector.
3) If the accuracy of a transformation is better than the accuracy of a model with just the input matrices, it stores the transformation to be applied, else it stores "baseline".
4) If a trasnformation is effective, then it picks the best possible transformation and then updates the training and test features matrices.
5) The baseline accuracy is replaced with the accuracy of the updated features matrix.
6) This loop is executed for all the features and the trasnformations to be applied are returned.

The disdvantage with this function was that it is a very greedy approach. The order of the features is very significant in terms of whether they will be trasnformed or not. The results were better than the result of the first accuracy function and comparable to the manual selection method.

### 3) Accuracy Function v3.R

1) The training and test datasets are given as inputs to the function.
2) At each feature, the function compares the accuracy with and without the feature in the matrix.
3) If excluding the feature provides an improvement in the accuracy, it removes that feature from the matrix and updates the baseline accuracy to the new value.

4) The function returns the modified features matrices after doing the above steps for all features.

## 3) Using h2o.varimp() and xgb.importance()

The h2o.varimp() function takes h2o.randomForest and h2o.gbm models as an input and returns the features ranked in descending order of importance.

xgb.importance() is used for xgb.train() functions.

xgb.cv() is used for determining the cross-validated performance for a set of parameters.

The procedure for using these functions is as follows:

**h2o.varimp()**

1) Do the parameter tuning that gives you the best cross-validated result for h2o.randomForest()/h2o.gbm
2) Store the value of

Due to computational complexity involved, we will be using h2o.randomForest for extracting the variable importance. We are using a cross-validated randomForest with 5-folds and the maximum depth of trees set to 5, for 1000 trees.

The most relevant features will then be selected, used for prediction and further variable selection.

These steps were done intermittently throughout the duration of project and each set of features used will be described as part of the various functions we have explored for model building. The features sets will be described in detail and the code corresponding to each one of these will be referenced in the respective section.

# The Features used and their derivation:

The main features sets used in the approaches and models described in the next part of this report are as follows:

1) The original features matrix
2) The Manual Selection features matrix
3) The xgboost linear regression result matrix
4) Final 49 features matrix
5) Final 50 features matrix

# Intial Approaches

## The multinom function.

One of the easiest way to approach a multiclass classification problem is applying the multinom function from the nnet pacakge in order to perform multinomial regression.

Doing so on the original features matrix gave us a baseline accuracy of 0.4823

Our goal is to develop a model that can definitively perform better than this one.

Doing various iterations with the function, the set of features that provided the best result were as follows:

1) Scale
2) Take the log of
3) Exclude the features

Doing the above steps resulted in an prediction accuracy of ..... which resulted in a improvement in the accuracy of the function is to

### The glmnet function

The function cv.glmnet is compoaratively much more robust than the multinom function as it performs cross-validation across the dataset and picks the stores the errors for various regularization parameters. It was implemented in the following ways

### PCA

The condition number of the original matrix is ..... This is a clear indication of extremely high collinearity amongst the features, one way to counter this issue is to extract the principal components of the features and use them as regressors of our data.

Doing a PCA on the original features matrix had a resultant accuracy of ..... Doing a PCA on the features that we had selected for the multinom function resulted in an accuracy of .....

Using the prinicipal components of the original features matrix had a resultant accuracy of Using the prinicipal components of the transformed features matrix with cv.glmnet had a resultant accuracy of .....

## h2o

The main reason for the use of the H2o package in R is the fact that it performs parallel computation making it easier to run relatively complex models in a shorter time. This provides a significant edge while doing cross-validation and parameter tuning.

The following functions from h2o have been used:

```
1) h2o.gbm
2) h2o.deeplearning
3) h2o.randomForest
```

### h2o.DeepLearning

We tried implementing a deep learning model using the function h20.DeepLearning from the h2o package. The matrix used was the Final 49 features matrix. The parameters tuned were the following:

1) The number of layers of weights
2) The number of weights in each layer (constant or decreasing with progression layers from input to output)
3) With and without "Dropout"
4) The evaluation metric: "mlogloss" and "misclassification"

The best accuracy we achieved with the tuning of parameters was 0.502 We decided not to use this model because deep learning requires a vast amount of data. The complexity of tuning the number of weights and layers was another disadvantage with this approach.

## xgboost - To fill in Later today

## Our submissions

### Submission 1

### Submission 2

## A brief comparison - randomForest package vs h2o.randomForest

## Other Approaches

### kNN and weighted kNN

kNN and weighted kNN both performed quite poorly on this dataset, we implemented the functions from the () package on 2 sets of features:

1) The original features matrix: The optimal k was ..... and the predicted accuracy was ........
2) The transformed features matrix 1: The optimal k was ..... and the prediction accuracy was ........

We decided not to use kNN and weighted kNN because the other models are much more robust in comparison to them. The advantage of Nearest Neighbour method is that it is very easy to comprehend, implement. The computational complexity is low as well because the most expensive part of the method is the calulating the $i$-dimensional distance between the points (k represents the number of features).

### Separate Logistic Regressions

## Conclusions