Data Science 316 A1 Project

Sebastien Meniere, Jaunré de Swardt

1. Introduction and Problem Statement

Is it possible to construct an accurate and interpretable model to predict the popularity of news articles before they are published?

Online news popularity prediction is a crucial area of research, aiming to anticipate the dissemination of digital media through online social networks and forecast the popularity of news articles before publication. Metrics such as the number of likes, shares, and views are commonly used to measure popularity, with reliable predictions offering significant advantages to news agencies and providers. Anticipating the popularity of news articles allows for data-driven article optimization, competitive advantages in the media landscape, and enhanced user engagement, all of which could contribute to industry success. Furthermore, an acute understanding of what news articles will be popular is valuable to many other sectors, including consumer markets, political affairs, marketing, and entertainment. Trending news influences the public’s opinions, interests, and decision-making. The difficulty in creating a robust predictive model in this context is obtaining data captures and explains the attributes that correlate with high popularity. There are too many unknown, and unmeasured variables in the physical world that will influence which articles become popular; current political affairs, fashion trends, and consumer fads could all impact popularity. These events are difficult to incorporate into training data sets, as they are time/period specific, and the information captured might not generalize to future events. The system is dynamic, rapidly changing, and reacting to real-world events. In this project, we will evaluate the predictive classification models proposed by Fernande et al. and aim to improve their performance. Specifically, we will focus on enhancing the model's classification accuracy, while maintaining interpretability.

2. Data Description

The dataset used in this study was obtained from the UCI Machine Learning Repository and consists of 39,644 articles published by the reputable news organization Mashable over two years. Each article was processed to extract 61 features, categorized into six distinct categories. These features encompass both numerical and ratio variables, where ratio variables typically range between 0 and 1, while numerical variables include both floating-point and integer values. Notably, the dataset contains no missing values, ensuring completeness for analysis.

|  |  |
| --- | --- |
| Category | Feature examples |
| WORD | Number of words in article / title, Average word length, Rate of unique / non-stop words |
| Links | Number of links, Number of Mashable article links |
| Digital Media | Number of images / images |
| Time | Day of the week, Published on weekend |
| Keywords | Number of keywords, Article category |
| NLP | Title subjectivity, Article text subjectivity, Title sentiment polarity |
| Target | Number of shares |

The target variable in the dataset is of integer type, ranging from zero to eight hundred thousand. This variable will be partitioned into bins, enabling the use of classification models for prediction.

It is important to note that certain variables that are deemed non-informative, such as the URL to the article page, have been excluded from the model training process. There are several outliers, where the number of shares was significantly larger than any reasonable amount of standard deviations from the mean.

We would also like to point out some remarks about the data. Firstly, we found that most of the predictor variables are uncorrelated with the target in its continuous form. Partial dependence plots between these predictors and the target revealed little to no trends. Predictors with these properties might not be useful in predicting the response and could be a source of unwanted noise in model construction. The data is sorted chronologically over two years. However, we found no evidence to suggest that the data is dependent on time. We omit the time feature, as the aim of the model is to make predictions about future publications. Finally, we note that the data is highly complex and non-linear.

3. The Current Approach

The study investigates five different classification models, namely Random Forest (RF), Naïve Bayes (NB), Adaptive Boosting (AdaBoost), SVM with a radial basis function (RBF) kernel, and K-Nearest Neighbors (KNN). These models were chosen as they represent the state-of-the-art in classification and offer a diverse range of methodologies for predicting news article popularity.

To prepare the dataset for classification, the continuous numeric target variable, shares, was transformed into a categorical variable using a threshold value of 1400. Values exceeding this threshold are categorized as popular, and those below as unpopular.

Many of these models require a hyperparameter, which is a parameter that must be chosen before the model can be trained. In the case of K-Nearest Neighbors classifier the hyperparameter is the number of neighbors that will be considered for each data point. For AdaBoost and Random Forest, the hyperparameter is the number of trees. The authors fine-tuned / optimized these parameters using a grid search. Which is the process of iterating over a set of predefined hyperparameter values, training the model with each element of the set, and then testing the performance of that model. The result of this is finding the value of the hyperparameter that gives the best performance. Without this step, the authors might have arbitrarily chosen the values of the hyperparameters which could have led to suboptimal, poor preforming models.

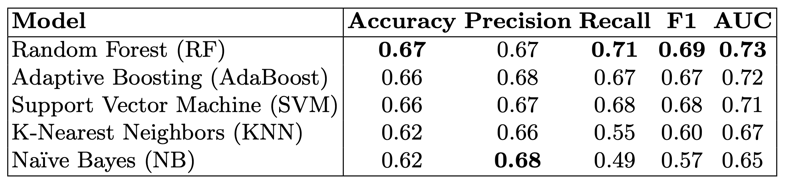
The authors also demonstrate good practices for model training by using disjoint testing and training data sets. They decided to employ a commonly used split of 70% training data and 30% testing data. In which the entire data set is randomly partitioned into two sets, with the above ratios. The validation set can be used to understand the model’s performance on unseen data, which will be the case when then model is deployed into the real world. Additionally, comparing the model’s predictive performance on the training data versus the testing data can reveal whether the model tends to overfit the training data, implying a poor choice of hyperparameters or too much flexibility in the model.

For robust evaluation of the trained models, the authors used a rolling window analysis. This approach uses a training window of W consecutive data points, which is used to fit the model. Then L predictions are made, then the training window is updated by replacing the L oldest samples with L more recent ones. A new model is fit using the new window, and the process is repeated until the end of the data set is reached. A window size of W = 10000 and a prediction size of L = 1000 was chosen. Motivation for these specific values was not discussed.

We note that the authors did not perform any feature selection, and all starting features from the original data set were used in the training of each model.

After the models were trained several performance metrics were calculated for each of models to evaluate their performance. The AUC, F1, precision, accuracy and recall. The results obtained are listed in the below table.

*Model Evaluation*



We note that the metrics: accuracy, precision, recall, and F1 were all calculated with a probability threshold value of 0.5. Meaning that observations were classified as belonging to class 0 if the predicted probability was below 0.5. Whereas the AUC calculation is irrespective of this threshold value, as is the area under the ROC curve which iterates over the interval [0, 1].

The authors award RF as the highest overall performing model.

4. Potential Problems

Overall, we believe the author’s methodology to be very strong. Starting from the beginning by transforming the target into a balanced categorical variable, using testing a validation sets to mitigate overfitting and evaluate model performance, tuning model hyperparameters through grid searching, using rolling windows for robust analysis, and finally calculating meaning metrics to measure each model’s performance.

Problems:

Before we discuss the problems with the current model we would first like to point out a problem with the overall methodology. The authors gathered the data, and then somewhat arbitrarily chose 5 classification models to train and test, and then use the best performing model as the solution. This method of model construction is widely used in research but it is computationally inefficient and time consuming. We believe the researchers should have examined the data and the problem, and then decided on one or two models that fit the needs of the solution, rather than training many different models arbitrarily. For example

The solution model needs to be highly non-linear and relatively interpretable.

The KNN model is not a good fit for this problem for the following reasons:

* Curse of dimensionality (the data set is very large and there are 58 predictor features)
* Highly non-linear and hard to interpret

Naïve Bayes model is also not a good fit for this problem:

* Assumption of independence between observations in each class. Which is a fair assumption however there does exist multicollinearity between predictor, and this assumption is violated without preforming PCA.
* Assumption of linear class boundaries. This assumption simply does not hold in this data as it highly non-linear.
* Assumption of Gaussian class conditional probabilities. There is no evidence to suggest that this assumption holds.

Support vector machines:

* The data is very noisy, with many predictors being uninformative. There are also a lot of outliers in the data, events where news articles went viral for reasons that cannot be quantified from the data set alone. Though SVMs are very robust, these two characteristics of the data set are very undesirable when fitting support vector machines.
* SVM’s are very difficult to interpret, which goes against one of the core aims of this reseach

From these few points alone we see that a Tree based model and Ensemble learning models would be the best to use in this context. They are both capable in handling large, they are robust and relatively unaffected by outliers (as they can be partitioned away into roots), and highly interpretable. This procedure, analyzing the data and the problem to identify what sort of model should be used, means that we can limit the scope of the model construction and focus on the intricate aspects of each model as opposed to blindly and broadly constructing / testing models that do not align with the problem at hand. Therefore we will proceed by focusing exclusively on the training, testing, and evaluation of the Random Forest and AdaBoost models, and later we aim to improve of them.

1. The value selected to be the threshold, did result in a balanced binary classification space. But the fact that there are only two classes limits the model predictive capabilities. Since the range of the target variable is very wide, simply partitioning at this value greatly reduces the information that can be extracted from the prediction. The task might be better suited for regression models. Furthermore, the binary nature of the new target excludes some other powerful models, like Linear discriminant analysis and multiclass logistic regression. Further exploration into different bin sizes could be used.
2. The authors also did not use any dimensionality reduction techniques, nor did they use any feature selection techniques. This is not always a problem but, seeing as the data set has nearly 60 features, there could easily exist some redundant features that could be adding unnecessary noise to the final models. As we mentioned in the data description section, many of the predictors are completely uncorrelated with the target, including all of the predictors adds a lot of noise. Although Random Forest is a very robust model, the training data was not randomly selected, meaning that some trends in these uninformative predictors might exist and be included in the model. This would be a big problem as it could cause the model it include partitions that uses useless predictors, resulting in poor generalization. <site> we found research indicating that the use of PCA prior to training a random forest, even if we are not reducing the dimensionality but simply transforming the data, can result in a higher performing model. If we reduce the number of features, can only select those that are useful in predicting the target, we will increase the chances of the random forest model picking (randomly selecting) those useful predictors to create partitions. Whereas if we keep all of the predictors including the useless ones, we risk creating random subsets of predictors that will be used to create partitions.
3. Rolling windows analysis might have been overkill. It has to do with time series which the data set is, but they took the union of all the predictions, and only trained models on small subset windows. This might have resulted in many poor models. Rather we should explore the results of these models without the rolling windows. However since is time series related and since this is news, and the target is popularity, we know that whats popular now will change, and so this will change the attributes that were popular. So we should experiment with rolling windows with larger windows, and smaller windows. Also there’s no trend with time, so the rolling windows analysis might be useless, k-fold cross validation could be a better solution. As mentioned in the data description section, we can disregard the fact that the data is time dependent. Therefore, using a rolling window analysis could potentially model and incorporate random trends in contiguous data, resulting in a less general model.
4. Grid search values, performing a grid search is a tedious and computationally intensive practice. For this reason, it’s understandable that the authors only selected 6 values in their grid search sets. But we believe that they could have done a better job at fine tuning the hyperparameters. Maybe using grid sets with more values, and then doubling down and zooming in to really find the best values.
5. Eliminating outliers. The authors did not remove any outliers from the data. This could be a pretty significant problem as there exist a substantial number of points that exist far beyond any reasonable number of standard deviations from the mean. The presence of these outliers could be disproportionately skewing the data.
6. Grid search RF and AdaBoost, tuning for max depth.
7. Performing PCA before random forest for potentially better results.
8. Tuning more of the hyperparameters in the SVM.

Really we should boil it down to, better preprocessing of the data, this includes

* Removing outliers
* Potentially using more bins
* Potentially transforming the data with PCA
* For models that don’t do automatic feature selection we should do feature selection
* Fine tune ALL hyperparameters for ALL models, not just some hyperparameters (bigger grid search)
* Get standard error estimates for the predictions
* Use a different method than rolling window analysis
* The accuracy of Random Forest performance can be improved by conducting a feature selection with a balancing strategy