

Final Report

IDS 575: Machine Learning Statistics

Online News Popularity

Group 13

Sravya Chouta | Roshan Dhanasiri
Sierra Laoratanapong | Sathwik Maddi

Abstract

In our project, news article statistics of a company mashable.com are used to evaluate the popularity and noteworthy information of the target variable using various Machine Learning techniques. First, we prepared and cleaned our data. Then we altered our target variable “shares” to be predicted. We created a binary class where the output based on the number of shares is either 0 or 1. We did hyperparameter tuning to increase the accuracy of the models. We also visualized our results, compared our accuracies and decided on the best models for our prediction.

Introduction

Nowadays, social media is one of the most important communication channels for every type of information including news. There are a tremendous amount of online news posts per day and they interact with people in different ways or manner. The difference of attributes of each post may affect the popularity represented by the number of shares which is the target of this study. The popularity of posts can depict if the news is widely spread in social media. Furthermore, the data in social media is important to help predict the favor of the audience to posts. Therefore, we are strongly interested in analytics of social media data to investigate and forecast the popularity represented by shares, for instance, what types of the data, variables, styles have impacts to the websites' or articles' popularity. There are attributes in the dataset, such as the number of words, the number of links, the number of images/videos, date posted, and so on, that can be used for the study.

We used the online news popularity dataset from [UCI repository](#) to create prediction models using various classification machine learning techniques, for example, decision trees, random forest, logistic regression, SVM, KNN, Naïve Bayes. Then, compare each method and select the best one. The study will be performed by following the steps:

1. Data exploration: review the type and distribution of data, the relationship between variables and how they relate to the predicting target. [1]
2. Data cleansing: Identify and rectify inconsistencies, anomalies, missing data, outliers. [1]

3. Visualization of the data: shows the data's statistics and basic analysis to see the nature of data. [1]
4. Data structuring: consider the machine learning algorithms that will be used in the study. [1]
5. Feature engineering and selection: create new variables to improve model's output and choose relevant and eliminate non relevant variables. [1]
6. Running machine learning models
7. Comparing the models, selecting the best model, visualizing the results.

Related Work

In recent years, popularity prediction has been the subject of extensive research. It is creating new challenges and new research interest. Many researchers devote themselves into finding the law of information diffusion. Predicting popularity is an essential part with significant practical benefit.

Ren et al. [2] applied many machine learning algorithms using Online News Popularity Dataset from UCI machine learning repository. They used Mutual information and Fisher criterion for feature ranking and selected 20 features which got the best rank. Many machine learning algorithms were used in Ren et al. prediction -namely: Linear Regression, Logistic Regression, Support Vector Machine, Random Forests, K-Nearest Neighbors, SVR (Linear Kernel), REPTree, Kernel Partial Least Square, Kernel Perceptron, and C4.5 Algorithm. They applied with 5-fold cross validation. As a classification model, logistic regression achieved a decent accuracy, better than most of the models. Random Forest had the best result for this classification problem. They were able to utilize different number of decision trees and different number of features for each decision point. Therefore, they got an accuracy of about 66% in Linear Regression and an accuracy of about 55% in SVM with polynomial kernel with $d = 9$. Finally, they changed the number of trees continuously from 5 to 500. The accuracy reached a limit of 69%, which is the best among all Algorithms.

Frenandes et al. [3] used an Online News Popularity dataset in implementing various machine-learning algorithms namely: Random Forests, AdaBoost, Support Vector Machine, K-Nearest Neighbor and Naïve Bayes. AdaBoost and SVM obtained an accuracy of about 66%. However, KNN, Naïve Bayes obtained an accuracy of about 62%, and the best result was achieved by a Random Forest (RF) i.e. with an overall area under the Receiver Operating Characteristic (ROC) curve of 73%, which corresponded to an acceptable discrimination and an accuracy of about 67%.

Methods

1. Logistic regression:

Logistic regression is a classification model that estimates the probability of a binary outcome, such as, yes-no, depending on the given dataset. Because of the outcome as a probability, the value is bounded between 0 and 1.

The equation of the logistic regression are below [4]

$$odds = \frac{P}{1-P} \rightarrow \text{logit}(P) = \ln\left(\frac{P}{1-P}\right)$$

$$\text{logit}(P) = mx + b \rightarrow mx + b = \ln\left(\frac{P}{1-P}\right)$$

$$\left(\frac{P}{1-P}\right) = e^{(mx+b)} \rightarrow P = \frac{e^{(mx+b)}}{1+e^{(mx+b)}} \rightarrow P(x) = \frac{1}{1+e^{-(mx+b)}}$$

Pseudocode [5]:

- For i from 1 to k
- For each training data instance d_i
- Set the target value to

$$z_i \leftarrow \frac{y_j - P(1 | d_j)}{[P(1 | d_j) \cdot (1 - P(1 | d_j))]}$$

- Initialize the weight of instance d_j to $P(1 | d_j) \cdot (1 - p(1 | d_j))$
- Finalize a $f(j)$ to the data with class value (z_j) and weight (w_j)
- If $P(1 | d_j) > 0.5$ >> assign class label 1, otherwise class label 0



2. XGboost

XGboost is the extreme gradient boosting that is an ensemble machine learning algorithm. It can be used for classification or regression predictive modeling problems. It is an open-source implementation of the gradient boosting algorithm created by Tianqi Chen and was described in the paper “XGBoost: A Scalable Tree Boosting System.” by Chen and Carlos Guestrin in 2016. It is highly effective, high performance and fast to execute, that makes it one of the most popular implementations. [6]

Its calculation is based on function approximation by optimizing specific loss functions.

The objective function (loss function and regularization) at iteration t that we need to minimize is the following [7]

$$\mathcal{L}^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t)$$

Real value (label) known from the training data-set


Can be seen as $f(x + \Delta x)$ where $x = \hat{y}_i^{(t-1)}$

Pseudocode [8]:

- Define an initial model F0 to predict the target variable y. This model will be associated with a residual (y – F0)
- A new model h1 is fit to the residuals from the previous step
- F0 and h1 are combined to get F1, an enhanced version of F0. The mean square error of F1 will be lower than that of F0:

$$F_1(x) \leftarrow F_0(x) + h_1(x)$$

- We could model after the residuals of F1 and create a new model F2 to improve the performance of F1:

$$F_2(x) \leftarrow F_1(x) + h_2(x)$$

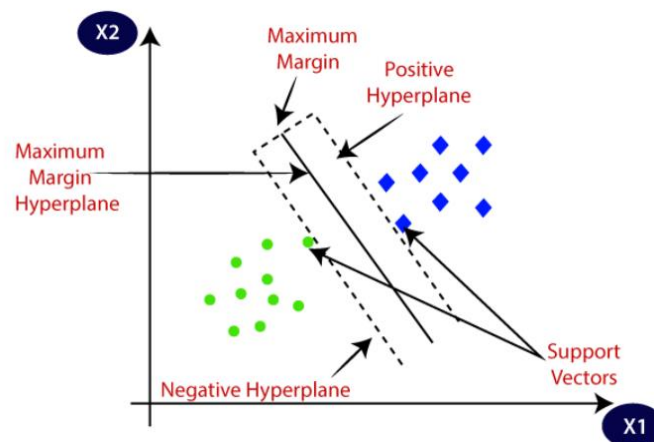
- This can be done for 'm' iterations, until residuals have been minimized as much as possible:
- This can be done for 'm' iterations, until residuals have been minimized as much as possible:

$$F_m(x) \leftarrow F_{m-1}(x) + h_m(x)$$

- The additive learners do not disturb the functions created in the previous steps. Instead, they impart information of their own to bring down the errors.

3. Support Vector Machine (SVM)

SVM is a powerful supervised algorithm that suits small complex datasets. It can be used for both regression and classification methods, but normally it works better in classification type. For SVM, it finds a hyperplane that separates the two classes the best by finding the maximum margin which is the maximum distance between two classes among many hyperplanes. [9]



The equation of a hyperplane is $w \cdot x + b = 0$, w is a vector normal to hyperplane and b is an offset [9]

To classify a point as negative or positive we need to define a decision rule. We can define decision rule as: [9]

$$\vec{X} \cdot \vec{w} - c \geq 0$$

putting $-c$ as b , we get

$$\vec{X} \cdot \vec{w} + b \geq 0$$

hence

$$y = \begin{cases} +1 & \text{if } \vec{X} \cdot \vec{w} + b \geq 0 \\ -1 & \text{if } \vec{X} \cdot \vec{w} + b < 0 \end{cases}$$

If the value of $w \cdot x + b > 0$ >> it is a positive point, otherwise it is a negative point. [9]

Pseudocode:

- Dataset with p^* variables and binary outcome
- Ranked list of variables according to their relevance
- Find the optimal values for the tuning parameters of the model
- Train the model

$p = p^*$

while $p \geq 2$

$SVM_p = SVM$ with the optimized tuning parameters for the p

variables and observations in data

$w_p =$ calculate weight vector of the SVM_p

rank criteria

min rank criteria = variable with the lowest value

$Rank_k =$ min rank criteria

$p = p-1$

End

$Rank_1 =$ value in data

Return ($Rank_1, \dots, Rank_{p^*}$)

4. Naïve Bayes

Naïve Bayes is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Given a class variable y and dependent feature vector x_1 through x_n , the Bayes' hypothesis expresses the beneath referenced relationship.

$$p(x|y) = p(x_1, \dots, x_n|y) = p(x_1|y)p(x_2|y)p(x_3|y) \cdots p(x_n|y) = \prod_{j=1}^n p(x_j|y)$$

Pseudocode:

- Input : Training dataset T
- $F = (f_1, f_2, f_3, \dots, f_n)$ // value of the predictor variable in testing dataset.
- Output : A class of testing dataset

Steps:

- Read the training dataset
- Calculate the mean and standard deviation of the predictor variables in each class.
- Repeat this, and Calculate the probability of f_i using the gauss density equation in each class until the probability of each predictor variable ($f_1, f_2, f_3, \dots, f_n$) has been calculated.
- Calculate the likelihood of each class.
- Get the greatest likelihood.

5. K Nearest Neighbor (KNN)

K Nearest Neighbor algorithm falls under the Supervised Learning category and is a data classification method for estimating the likelihood that a data point will become a member of one group or another based on what group the data points nearest to it belong to. It is used for classification (most commonly) and regression. It is a versatile algorithm also used for imputing missing values and resampling datasets.

$$d = \sqrt{\sum_{i=1}^N (X_i - Y_i)^2}$$

Pseudocode:

- Load the training data.
- Prepare data by scaling, missing value treatment, and dimensionality reduction as required.
- Find the optimal value for K:
- Predict a class value for new data:
- Calculate euclidean distance(X, X_i) from $i=1,2,3,\dots,n$.
- where X = new data point, X_i = training data, distance as per the chosen distance metric.
- Sort these distances in increasing order with corresponding train data.
- From this sorted list, select the top 'K' rows.
- Find the most frequent class from these chosen 'K' rows. This will be the predicted class.

6. Decision Tree Classifier

Decision tree algorithms are versatile supervised learning algorithms that can be used for both regression and classification problems. These algorithms use past data to predict or choose a class under which a new record may fall under. As suggested by the name, a decision tree can be visualized as a 'rooted' tree with multiple nodes, where an attribute value of a data point is evaluated and classification is performed based on this value. This classification is done in such a way that the 'information entropy' change remains minimal.

The algorithm decides which attribute to split on at a particular node based on what is called 'Information Gain'. Based on this concept, the attribute to be measured to split classes at a node is done in such a way that known objects are split less randomly. The information gain, is given by the following formula -

$$Gr(T, A_k) = \frac{Gain(T, A_k)}{SplitInfo(T, A_k)},$$

In the above formula, the information gain is given by the ratio of information gain to the 'SplitInfo', also known as 'intrinsic information'. This intrinsic information is basically the entropy of the training data set.

Pseudo code:

1. Original data set is taken as the root node.
2. On every iteration of the algorithm, it iterates through various attributes of the original data set and calculates the information gain and entropy.
3. The attribute with the smallest entropy and largest information gain is selected to be split on.
4. This attribute is then used to split the original data set to produce smaller subsets into which new data can be classified.
5. The algorithm repeats this process on each newly generated subset, only splitting on attributes that haven't been used for earlier splits.

Experimental Results

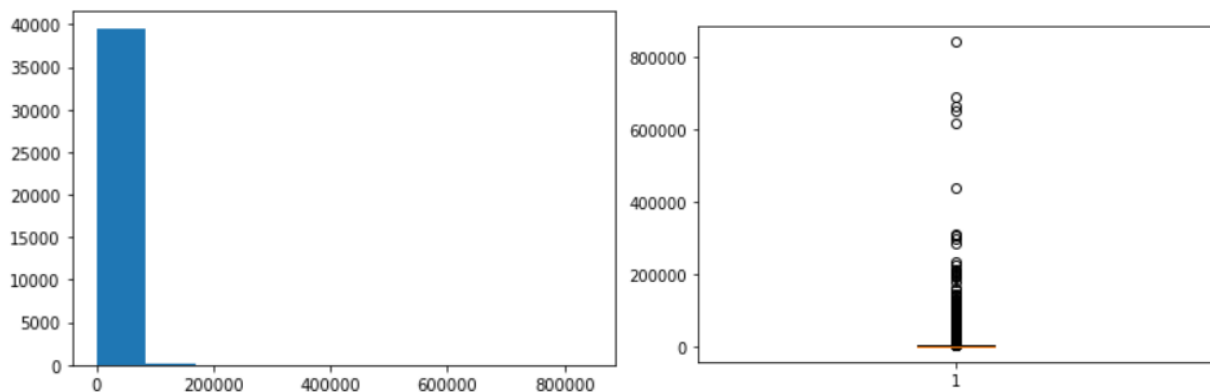
Dataset statistics and a basic analysis

The dataset named 'Online News Popularity Data Set' downloaded from [UCI repository](#)

The dataset had 39644 samples and 61 attributes, no null information.

The dataset was cleaned by deleting unrelated attributes, 'URL' and 'timedelta'.

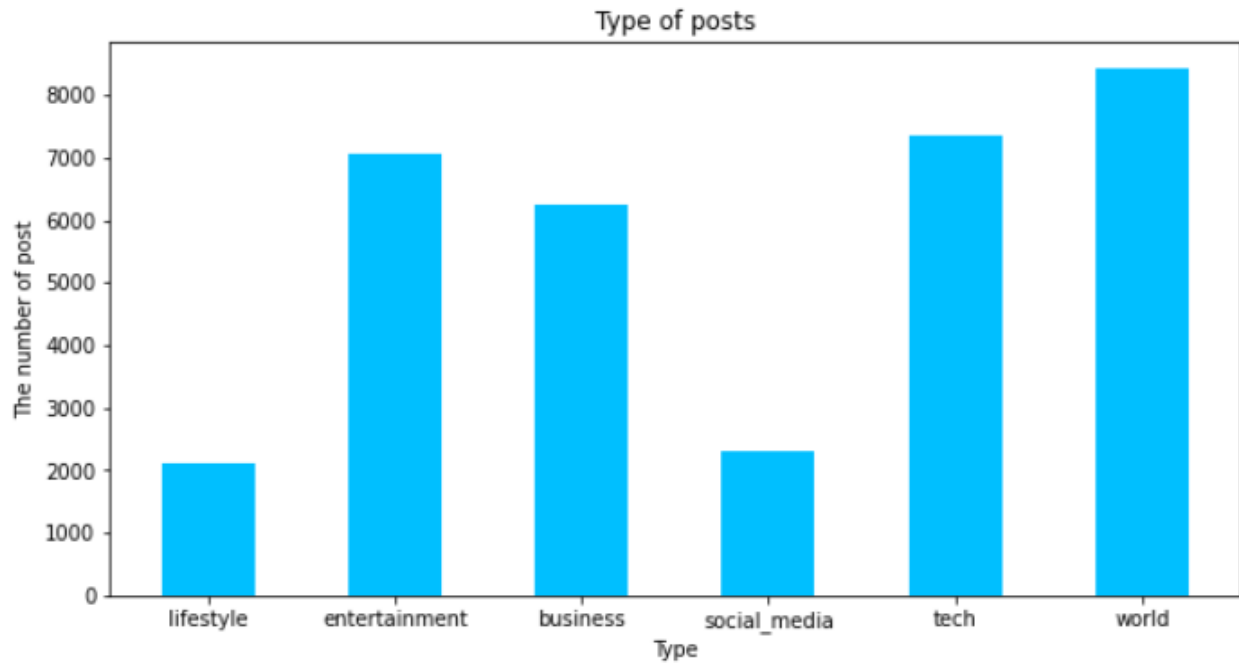
The distribution of 'shares'



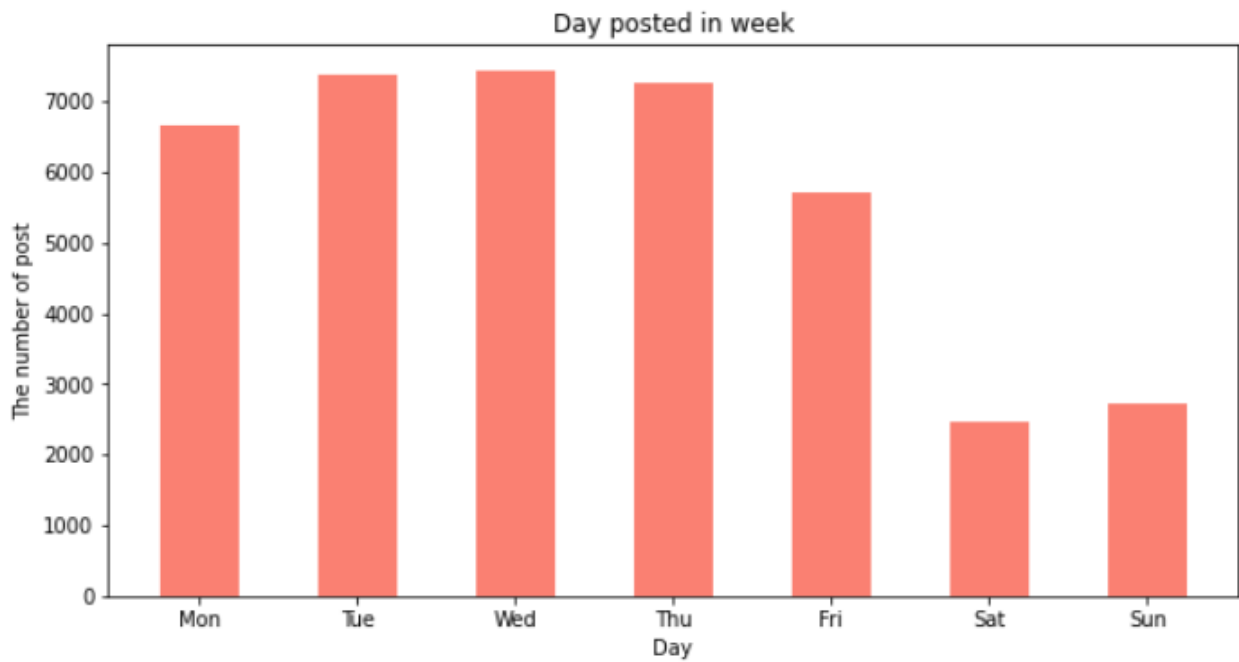
Target variable was shares which were continuous values. Because we tested with classification techniques, we changed it to categorical value by assigning a 'y' value to all samples due to the number of shares. We divided samples by median which was 1400.

After cleaning the dataset, there were 39644 samples and 59 attributes left. The dataset was randomly split 70:30 into training dataset and testing dataset.

There are 6 different types of posts as below



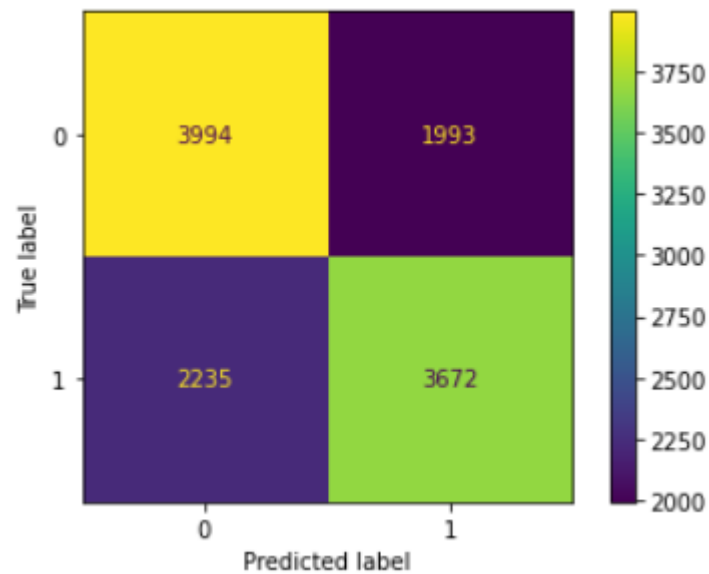
The number of posts in each day of week has been shown below



Results from methods

1. Baseline model: Logistic regression

Confusion matrix:



Classification report:

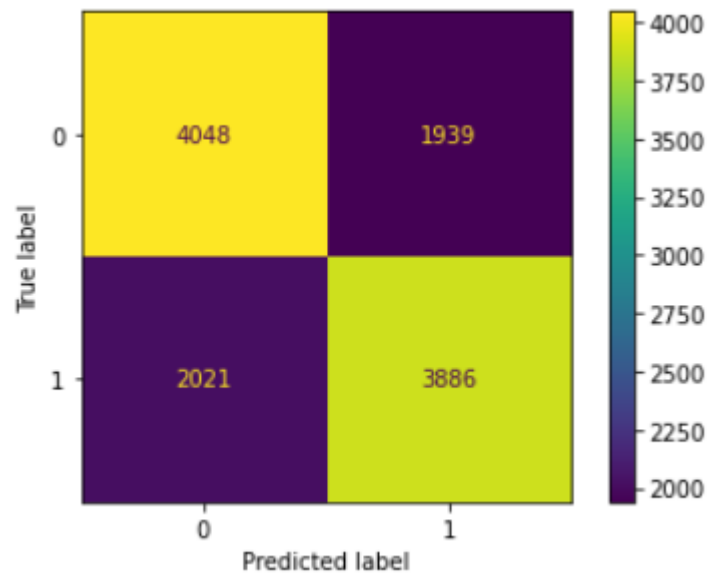
	precision	recall	f1-score	support
0	0.64	0.67	0.65	5987
1	0.65	0.62	0.63	5907
accuracy			0.64	11894
macro avg	0.64	0.64	0.64	11894
weighted avg	0.64	0.64	0.64	11894

The accuracy was increased from 0.60 to 0.64 after tuning.

The precision is 0.65 that means among predicted positive samples, there are 3672 samples that predict correctly and 1993 samples that predict wrong. The recall is 0.62 that means among actual positive samples, there are 3672 samples that predict correctly and 2235 samples that predict wrong.

2. XGboost

Confusion matrix:



Classification report:

	precision	recall	f1-score	support
0	0.67	0.68	0.67	5987
1	0.67	0.66	0.66	5907
accuracy			0.67	11894
macro avg	0.67	0.67	0.67	11894
weighted avg	0.67	0.67	0.67	11894

Important features:

	columns	importance
0	data_channel_is_entertainment	0.107246
1	data_channel_is_tech	0.059235
2	data_channel_is_socmed	0.053718
3	kw_avg_avg	0.052652
4	is_weekend	0.051650

The accuracy did not change or decreased after tuning, and the tuning process takes a long time to run. So the primary model has been chosen.

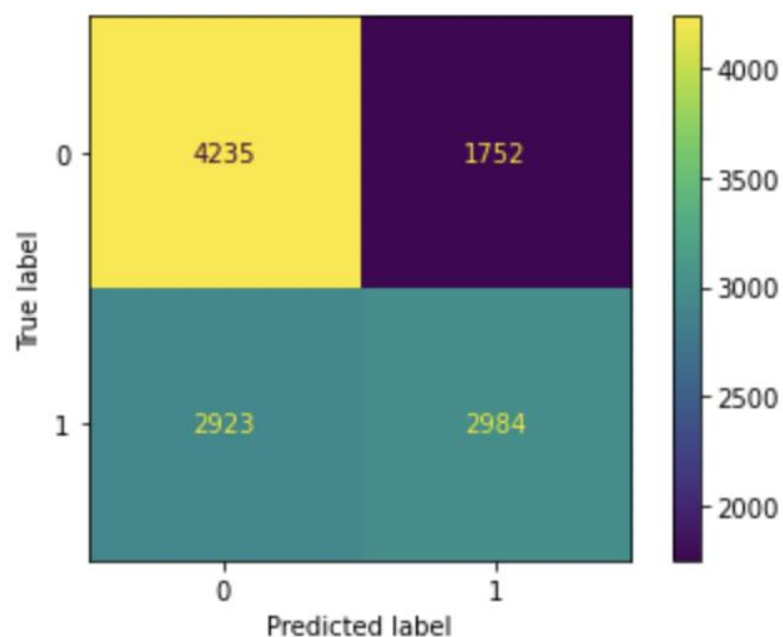
The precision is 0.67 that means among predicted positive samples, there are 3886 samples that predict correctly and 1939 samples that predict wrong. The recall is

0.66 that means among actual positive samples, there are 3886 samples that predict correctly and 2021 samples that predict wrong.

The most important feature for this model is data_channel_is_entertainment. In other words, the entertainment posts may have more chances to be popular or unpopular than other types.

3. Support Vector Machine (SVM)

Confusion matrix:



Classification report:

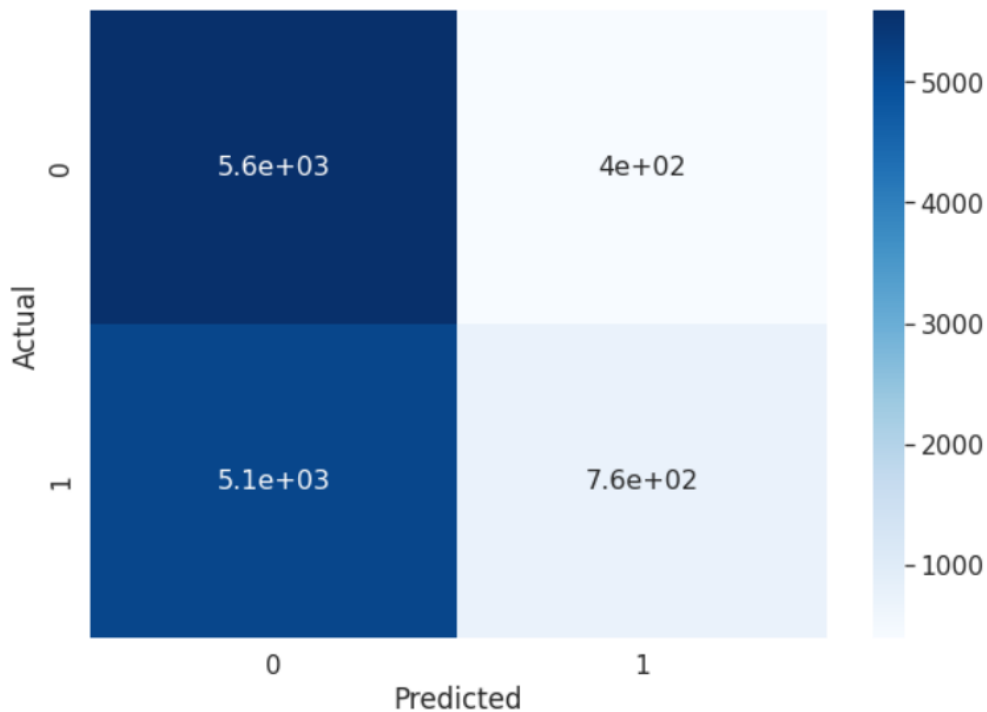
Classification	Report				
	precision	recall	f1-score	support	
0	0.59	0.71	0.64	5987	
1	0.63	0.51	0.56	5907	
accuracy			0.61	11894	
macro avg	0.61	0.61	0.60	11894	
weighted avg	0.61	0.61	0.60	11894	

The problem is the model takes extremely long time to run, so tuning is impossible. The primary plain model was chosen. The SVM model is suitable for small to medium dataset, but this dataset is too large to fit with SVM and the result is not better than other models which take shorter running time.

The precision is 0.63 that means among predicted positive samples, there are 2984 samples that predict correctly and 1752 samples that predict wrong. The recall is 0.51 that means among actual positive samples, there are 1984 samples that predict correctly and 2923 samples that predict wrong.

4. Naive Bayes

Confusion matrix:



Classification report:

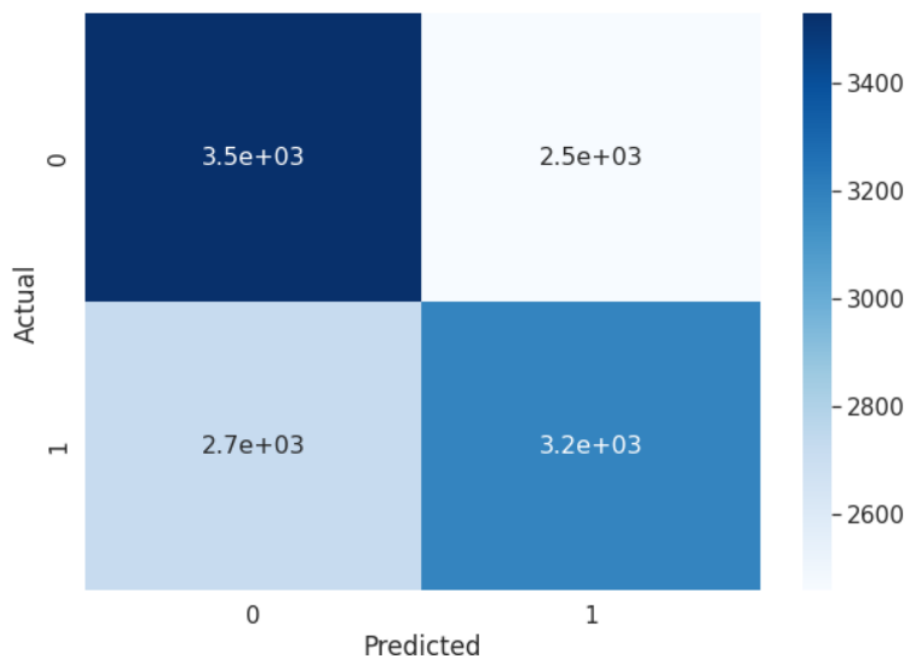
	precision	recall	f1-score	support
0	0.52	0.93	0.67	5987
1	0.66	0.13	0.21	5907
accuracy			0.53	11894
macro avg	0.59	0.53	0.44	11894
weighted avg	0.59	0.53	0.44	11894

The precision is 0.66 that means among predicted positive samples, there are 760 samples that predict correctly and 400 samples that predict wrong. The recall is 0.13 that means among actual positive samples, there are 760 samples that predict correctly and 5100 samples that predict wrong.

The problem of this model is it predicts a very low amount of positive (only 1160 samples), while predicting mainly negative (10,700 samples). That makes it has high precision and relatively low recall.

5. KNN

Confusion matrix:



Classification report:

	precision	recall	f1-score	support
0	0.56	0.59	0.58	5987
1	0.56	0.54	0.55	5907
accuracy			0.56	11894
macro avg	0.56	0.56	0.56	11894
weighted avg	0.56	0.56	0.56	11894

The precision is 0.56 that means among predicted positive samples, there are 3200 samples that predict correctly and 2500 samples that predict wrong. The recall is 0.54 that means among actual positive samples, there are 3200 samples that predict correctly and 2700 samples that predict wrong.

The accuracy, precision, and recall are a little bit above random model (0.5), so this is not a good enough model and it needs more tuning or more data wrangling.

6. Decision Tree Classifier:

Classification Report:

	precision	recall	f1-score	support
0	0.54	0.52	0.53	1468
1	0.58	0.59	0.59	1627
accuracy			0.56	3095
macro avg	0.56	0.56	0.56	3095
weighted avg	0.56	0.56	0.56	3095

Classification Report after Parameter Tuning:

	precision	recall	f1-score	support
0	0.60	0.49	0.54	1468
1	0.60	0.70	0.65	1627
accuracy			0.60	3095
macro avg	0.60	0.60	0.59	3095
weighted avg	0.60	0.60	0.60	3095

After initially running the algorithm, grid-search cross validation was used to extract the best parameters for our decision tree classifier model. The best parameters were found to be:

```
{'criterion': 'entropy',  
  'max_depth': 10,  
  'max_features': 'sqrt',  
  'splitter': 'random'}
```

Upon setting these parameters, there was found to be a slight improvement in f1 scores and accuracy, as seen above.

Conclusion

	LOGISTIC REGRESSION	XGBOOST	SVM	KNN	NAÏVE BAYES	DECISION TREE
ACCURACY	0.64	0.67	0.61	0.56	0.53	0.60
PRECISION	0.65	0.67	0.63	0.56	0.66	0.60
RECALL	0.62	0.66	0.51	0.54	0.26	0.70
F1-SCORE	0.63	0.66	0.51	0.55	0.21	0.65

We have experimented and analyzed different classification and regression models to choose the best models for the project where the logistic regression model is considered as the base model and XG model as the main model. Different models showed up different results and have varied amounts of accuracy as shown in the classification report. So, here is the detailed picture of the classification report results of the different models executed, which lets one decide the best model based on the rate of accuracy. The accuracy rate for each model seems to be more or less equal to each other but the accuracy rate for the main model which is XGBoost model stands on the top at 0.67 when in comparison to the other models.

All in all, although the accuracy of all models is not so high, we chose XG Boost as the best model where the accuracy stands at 0.67, precision at 0.67, recall at 0.66 and F-1 score at 0.66. But, in the future, the condition would be better if there is more tuning to the models. There are mainly two kinds of modeling we mainly perform one is regression modeling and another one is classification modeling. Since there is always a requirement for highly accurate models, we need to know about the techniques that can increase the accuracy of the model. To increase the accuracy of the models, try retrieving the estimates of the model's performance using score metrics, find and diagnose best parameter values using Validation curves, hold out the cross validation such as K fold cross validation, use the grid search to optimize the hyperparameter combination, fine tune the parameters of machine learning models constantly and continuously.

Reference

1. *Data Preparation in Machine Learning: 6 Key Steps*. (n.d.). SearchBusinessAnalytics.
<https://www.techtarget.com/searchbusinessanalytics/feature/Data-preparation-in-machine-learning-6-key-steps>
2. Ren, H. and Yang, Q., 2012. *Predicting and Evaluating the Popularity of Online News*.
3. Fernandes, K., Vinagre, P. and Cortez, P., 2015. A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. *Progress in Artificial Intelligence*, pp.535-546.
4. Haghighat, A., & Belyadi, H. (2021). In *Machine Learning Guide for oil and gas using python: A step-by-step breakdown with data, algorithms, codes, and applications* (pp. 169–295). essay, Gulf Professional Publishing.
5. Kumar, V. (2020). Evaluation of computationally intelligent techniques for breast cancer diagnosis. *Neural Computing and Applications*, 33(8), 3195–3208.
<https://doi.org/10.1007/s00521-020-05204-y>
6. Brownlee, J. (2021, March 6). *XGBoost for regression*. Machine Learning Mastery. Retrieved November 6, 2022, from <https://machinelearningmastery.com/xgboost-for-regression/>
7. Leventis, D. (2022, January 2). XGBoost mathematics explained. Medium. Retrieved November 27, 2022, from <https://dimleve.medium.com/xgboost-mathematics-explained-58262530904a>
8. guest_blog. (2020, December 23). *XGBoost algorithm: XGBoost in machine learning*. Analytics Vidhya. Retrieved November 27, 2022, from <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>
9. Saini, A. (2021, November 12). *Support Vector Machine(SVM): A Complete Guide for Beginners*. Analytics Vidhya. Retrieved November 27, 2022, from <https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/#:~:text=So%20now%20we%20can%20say,margin%20error%2C%20and%20vice%20versa.>

Team member responsibilities

- Abstract: Roshan
- Introduction: Sierra
- Related work: Roshan
- Model:
 - Logistic regression: Sierra
 - XGBOOST: Sierra
 - SVM: Sierra
 - Naive Bayes: Roshan
 - KNN: Roshan
 - Decision tree: Sathwik
- Experimental Results:
 - Dataset statistics and a basic analysis: Sierra
 - Results from methods
 - Logistic regression: Sierra
 - XGBOOST: Sierra
 - SVM: Sierra
 - Naive Bayes: Roshan, Sierra (summary)
 - KNN: Roshan, Sierra (summary)
 - Decision tree: Sathwik
- Conclusion: Sravya Chouta