By: Xu Wu

Online New Popularity

1.Problem definition

The article [1] is expected to predict a online news' popularity based on its 60(58 predictive and 2 non-predictive) binary or quantified attributes/features, such as number of words in the title, number of images, whether channel "business" or not, etc. Each observation comes from the site(www.mashable.com) within sequentially increased time period. The goal/predicted attribute is binary classified as popular vs unpopular using a decision threshold of 1400 social interactions(shares). We will split the dataset to 60% training set and 40% testing set and then build 4 different models based on different algorithms. Next, we use cross-validation to reduce the overfitting and try to get a better performance. Then we change size of training/testing dataset and observe models' performance changes.

2.Description of background

Machine learning is programming computers to optimize a performance criterion using example data or past experience. In machine learning, classification is used to predict categorical class labels, and it constructed a model based on the training dataset and uses it in classifying new data.

There are a lot of algorithms for classification in machine learning: Decision Tree, Random Forest , Adaboost, SVM, KNN, Naïve Bayes etc.

Because every model will be used for classifying future or unknown objects, it is necessary to evaluate each model's performance, including:

- Accuracy: compared by the data's labeled class with the class predicted by the model.
- Speed: time to construct and use the model
- Robustness: handle noise and missing values
- Scalability: efficiency as data size grows

In this article, the result model will be used for predicting a online news' popularity based on its features. With respect to such practical useage, this project will construct 4 models based on 4 different algorithms: Decision Tree, KNN, Random Forest and Naïve Bayes.

3.Description of dataset

The dataset is obtained from UC Irvine's repository [2]. All data is retrieved from Mashable [3]. The dataset contains 61 features (60 attributes and 1 goal/output), and 39797 instances. Here is the detail attribute information:

Non-predictive features:

0. url: URL of the article (non-predictive)
1. timedelta: Days between the article publication and the dataset acquisition (non-predictive)

Predictive features:

2. n_tokens_title: Number of words in the title
3. n_tokens_content: Number of words in the content
4. n_unique_tokens: Rate of unique words in the content
5. n_non_stop_words: Rate of non-stop words in the content

…

...

56. title_subjectivity: Title subjectivity
57. title_sentiment_polarity: Title polarity
58. abs_title_subjectivity: Absolute subjectivity level
59. abs_title_sentiment_polarity: Absolute polarity level

Goal predicted attribute:

60. shares: Number of shares (target)

4.Description of method used:

Algorithms:

a.**Decision Tree** [4]: decision tree is commonly used in classification in machine learning, it is a tree-like graph or model to make decisions and get consequent results. Decision tree implicitly perform variable screening or feature selection and the best feature of using it is easy to interpret and explain!

b.**KNN** [5]: K nearest neighbors is a non-parametric method used for classification and regression. The input normally consists of k closest training instances in the space. A object is classified to a class because it is most common among its k nearest neighbors. It is chosen because knn don't need make any assumption on the underlying data distribution, it is also a lazy algorithm and don't use training data to do any generalization.

c.**Random Forest** [6]: in the classification, random forest constructs multitude of decision trees at training time and then output the class that is the mode of classes or mean prediction of individual trees. It can overcome the shortage of single decision tree's overfitting to training set.

d.**Naïve Bayes** [7]: naive Bayes classifier is a probabilistic classifier based on Bayes' theorem. It depends on the assumption that some features are conditionally independent. It is super simple

and can converge quicker than some other models, so it have a chance to achieve fast and easy process.

External python library for data analysis:

a.**Pandas** [8] : Pandas is an open source, BSD-licensed library. It provides high performance and simple tools for data analysis.

For here, it is mainly used to load data from the raw Excel .csv file.

b.**Scikit-learn** [9] : Scikit-learn is an open source, BSD licensed, machine learning python library. It built on NumPy, SciPy and matplotlib.It also provides simple and efficient tools for data mining and data analysis.

For here, we use for many purposes: split dataset into training and testing sets; construct 4 different estimators(model) and then fit the model; get the score(accuracy) with input of testing data set; or do cross-validation directly and get the score(which is considered a improvement process in this project).

c. **XlsxWriter** [10] : XlsxWriter is a Python module that can be used to write text, numbers, formulas and hyperlinks to multiple worksheets in an Excel 2007+ XLSX file.

For here, it is used for write results (accuracy and time,per train/test set increase, based on 4 models) into a excel output file(.xlsx). Then in the excel file, we use internal Excel chart graphing functionality to plot comparison charts.

Process:

a.Use Pandas, to read data from Excel .csv file and then process the dataset:

Replace the goal attribute values with binary value according to the following decision threshold:

- If the goal attribute(“shares”) < 1400---unpopular--->value: 0
- Else----------popular------->value: 1

b.Use method from Scikit-learn, cross_validation.train_test_split to split the dataset to two parts: training set(60%) and testing set(40%)

c.Using the classifier in Scikit-learn, construct estimator object based on different algorithms(Decision Tree, KNN,Random Forest, Naïve Bayes):

- dt = DecisionTreeClassifier(min_samples_split=20,random_state=99)
- knn = KNeighborsClassifier()
- rf = RandomForestClassifier(n_estimators=100,n_jobs=-1)
- nb = BernoulliNB()

d.Use fit method to fit the mode, with training data set as the input, and then use score method to get the score(accuracy) with testing data set as the input:

clf_xx.fit(X_train,y_train);

Accuracy=clf_xx.score(X_test,y_test);

> ***Alternative process of b,c,d above is to use k-fold cross-validation*** [11] ***: The training data set is splitted into k subsets. Then the model is train on k-1 sets and validated on the rest of sets. Keep running the loop for k times and the final result model is got from average of the values computed in loops. The approach can be very expensive, but does not waste too much data.***

e.Use timestamp t=time() to record every model's construction start and end time, then end_time minus start_time to get every model's time elapsed(construction and use model). Collect every model's result(time elapsed and accuracy) and plot it on tables below.

f.Keep training/testing set unchanged and increases another set(testing/training) 5% of original size every time, and then get the each result (accuracy and time elapsed) for every model, record it and use XlsxWriter to write it to excel file(.xlsx). In the excel file, use internal chart graphing functionality to plot charts for comparison and analysis (See charts below for details).

5.Experiment: performance analysis and results

Accuracy:

Algorithm	Decision Tree	KNN	Random Forest	Naïve Bayes
Accuracy(Original)	57.9%	57%	66.4%	61.2%
Accuracy(Cross-Validation)	55.5%	54.9%	64.4%	60.8%

Speed:

Algorithm	Decision Tree	KNN	Random Forest	Naïve Bayes
Time elapsed (Original)	1.2s	3.23s	6.33s	0.07s
Time elapsed (Cross-Validation)	8.44s	10.83s	37.71s	0.48s

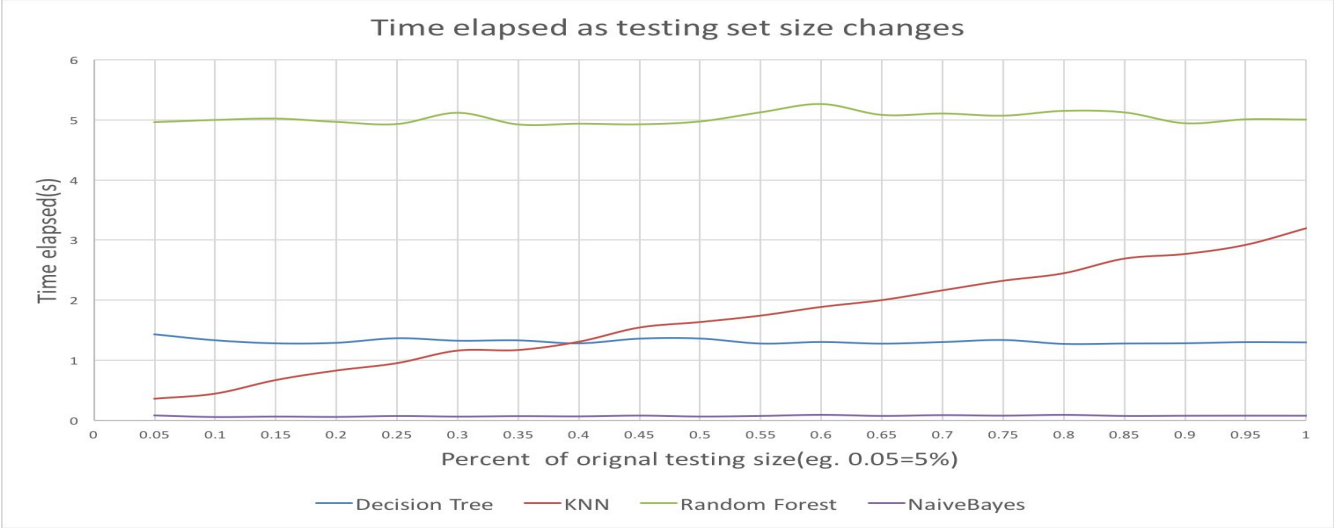
From two tables above, we can see Random Forest model achieves the highest accuracy (66.4%), but it consuming the most time (6.63s) among 4 models. Naive Bayes model achieves

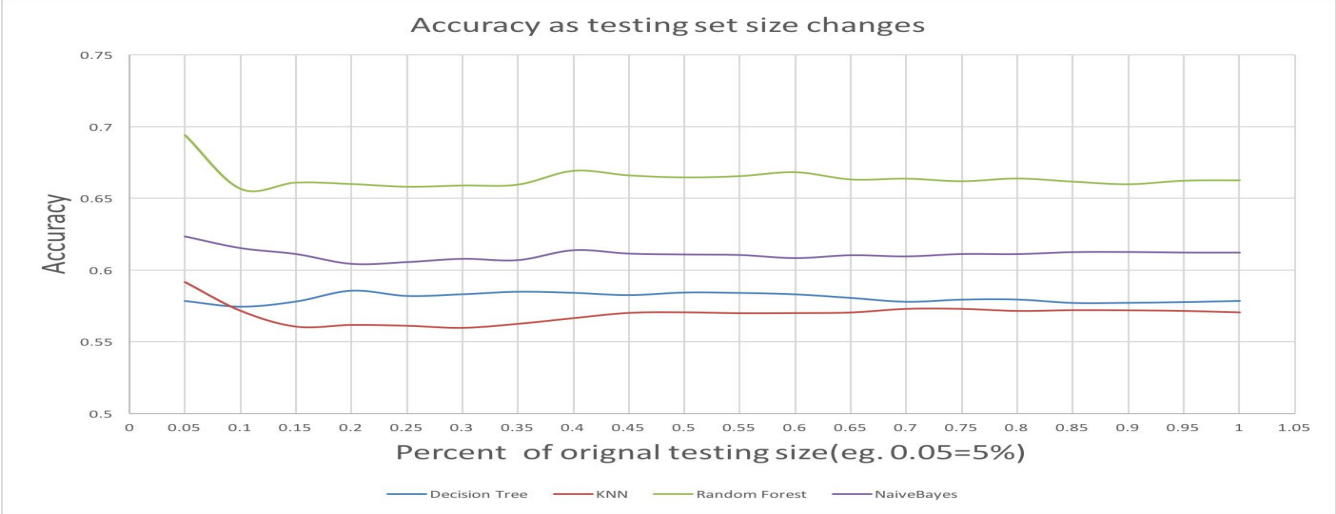
the least time elapsed(0.07s) and a properly medium accuracy. The other two models have bad accuracy.The k-fold cross-validation has proved that the accuracy should be less than that gained from the original method. But considering its big time consuming, the difference can be ignored relatively under the purpose of comparing these 4 models.

Scalability:

- **Performance by changing testing set**

For every model, training set keeps unchanged(60% of total dataset), testing set increases 5% of original testing set(40% of total dataset) each time. So testing set increases 2%(40% x 5%) of total size each time. Here is how result (Time elapsed/Accuracy) looks:

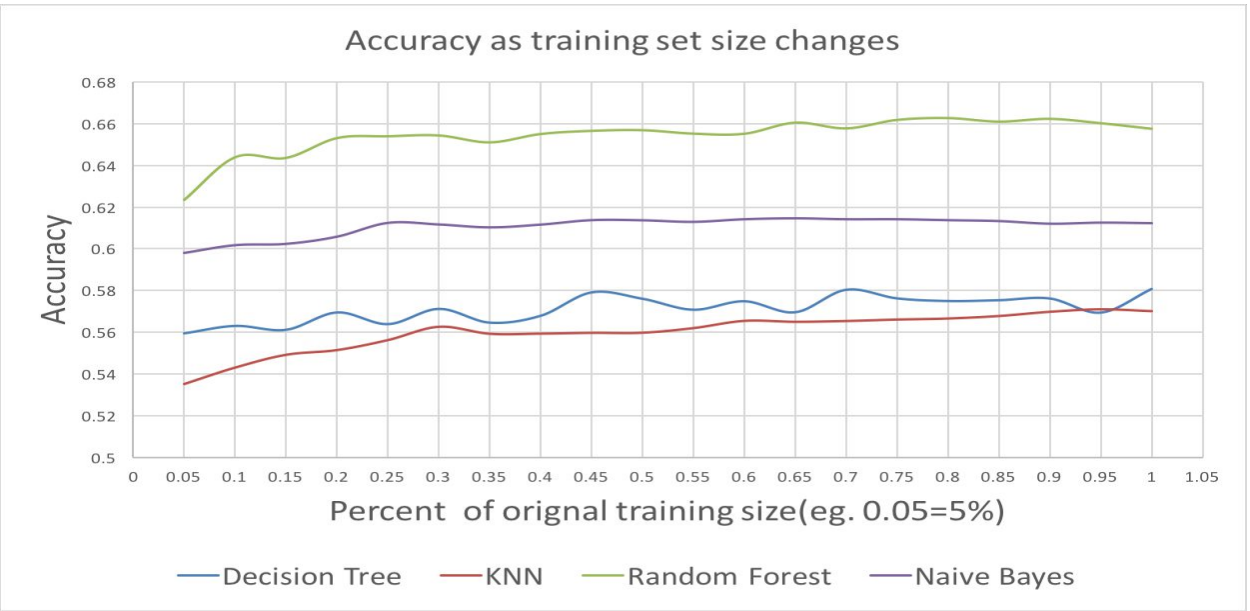


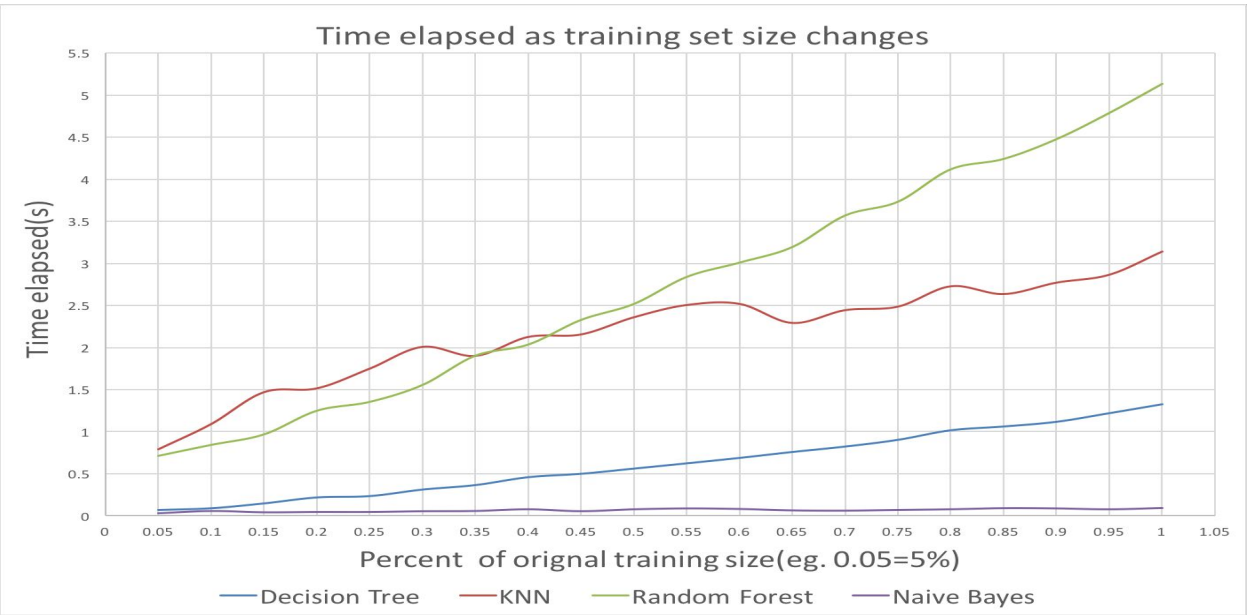


From the two charts above, as size of testing set increases, every model's accuracy keeps stable. For time elapsed, KNN increases linearly, and other three models keeps stable.

- **Performance by changing training set**

For every model, testing set keeps unchanged(40% of total dataset), training set increases 5% of original testing set(60% of total dataset) each time. So testing set increases 3%(60% x 5%) of total size each time. Here is how result (Time elapsed/Accuracy) looks:





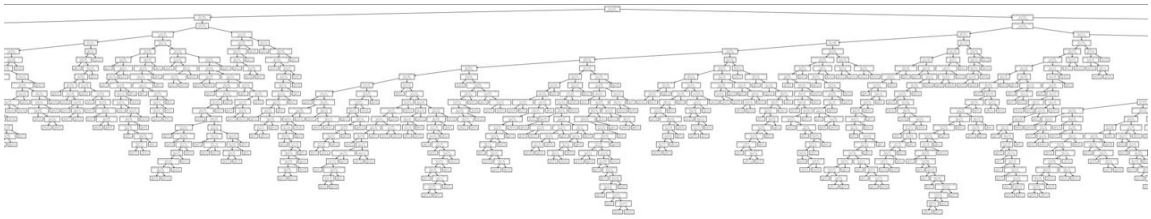
From the two charts above, as size of training set increases, every model's accuracy increases linearly at the beginning and then slightly increases and tend to keep stable, and KNN's change rate is higher than any other three models. For the time elapsed, Navie Bayes model still keeps stable and other three models change linearly: Random Forest changed faster than other two models.

6.Observation and conclusion

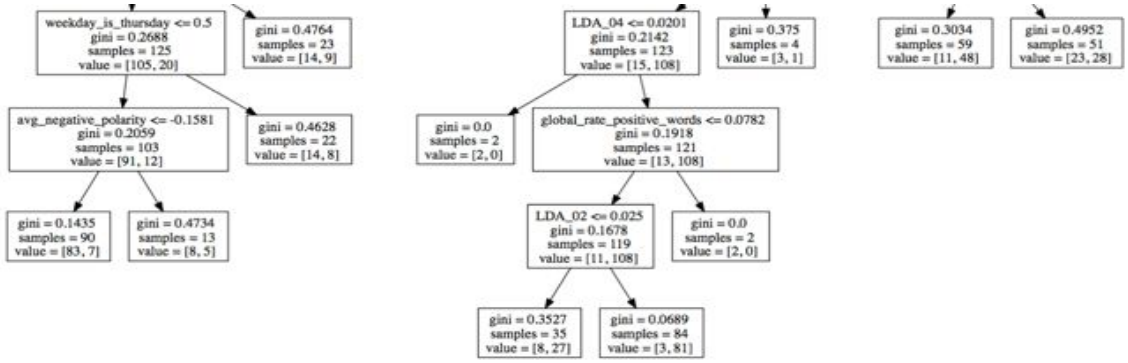
According to the analyzing each model's performance:

Decision Tree has a bad performance (55.5% accuracy based on cross validation), it is because the dataset's volume is not enough to take so many attributes(58 predictive attributes in total), and then it occurs overfitting due to relatively lack of samples. We can see from decision tree result graph below, most of nodes at bottom are not fully distributed to result.

(The graph is created by the use of scikit-learn export_graphviz)



Overview of the decision tree



Part of the decision tree result

KNN also has a bad performance(54.9% accuracy based on cross validation), it is because KNN is very sensitive to local structure of the data, every node's value is mostly determined by the contribution of near neighborhoods than further nodes.

Random Forest has the best accuracy, which is 64.4% based on cross validation and 66.4% based on original dataset splitting(60% training and 40% testing). It builds a forest of decision trees based on differing attributes in the nodes. Different trees have access to a different random sub-collection of the dataset, and it constructs a collection of decision trees with

controlled variance. So it overcome the shortage of decision tree's easy overfitting and can have a better performance.

Naive Bayes model has a medium accuracy, which is 61.2%. But it has a faster(always less than 1s) constructing and model testing time than any other models. And as the training/testing sets increases, it always keeps stable. It is basically because it depends on the assumption that some features are conditionally independent. It is super simple and can converge quickly.

Scalability:

Based on experiment result, as testing dataset increases, models are not significantly influenced except time elapsed on the KNN model. As training dataset increases, every model have increased accuracy at the beginning and then go up to a "limit boundary" and keeps stable.

Conclusion for this article:

Overall, if anyone pursues a faster process, Navie Bayes is the best choice for this problem without sacrificing too much accuracy. Otherwise, if accuracy is considered first, obviously Random Forest is a better one.

7.References

[1]K. Fernandes, P. Vinagre and P. Cortez. A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News.
http://link.springer.com/chapter/10.1007%2F978-3-319-23485-4_53

[2]Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science
http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

[3]Mashable www.mashable.com

[4]Decision Tree https://en.wikipedia.org/wiki/Decision_tree

[5]KNN https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[6]Random Forest https://en.wikipedia.org/wiki/Random_forest

[7]Naïve Bayes Classifier https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[8]Pandas http://pandas.pydata.org/

[9]Scikit-learn http://scikit-learn.org/stable/index.html

[10]XlsxWriter http://xlsxwriter.readthedocs.org/

[11]Cross Validation https://en.wikipedia.org/wiki/Cross-validation_(statistics)