Content Analysis For Online MultiMedia

Afshin Moin (afshinm) Peng Seng Kuok (pkuok) Abhishek Bharani (abharani)

1 Introduction

Content sharing has been the key ingredient of the biggest success stories in the last decade. Social networks like Facebook and LinkedIn, video sharing services like YouTube, Ecommerce websites like Amazon and online travel agencies like TripAdviser and Expedia are all examples of online platforms that provide a convenient way for their users to exchange content, goods or services. It is obvious that the success of such online platforms depends on how effectively they link the end users to relevant content. This latter cannot be achieved without proper content classification and user feedback analysis. This motivated us to apply machine learning techniques to automate the process of user feedback analysis and content classification. We work with a real sample dataset taken from YouTube [11]. Nevertheless, same techniques can be applied to any other database with similar structure like Yelp and Netflix. In Section 3, we discuss our approach to data labeling and feature extraction. In Section 4, we review our approach to sentiment and category classification of YouTube comments. Results of the experiments for different classification techniques are compared with each other in Section 5. Also, calibration curves are presented and effect of dimensionality reduction on accuracy is examined. Section 6 concludes the report.

2 Related Work

In this section we review a number of works related to feature extraction and content classification. A rich literature is available on classification techniques based on probabilistic and discriminative models [1]. Learning regression and ranking functions are discussed in [7].

Hu and Liu [2] provide a methodology to summarize a list of positive and negative words based on customer reviews in sentiment analysis. The positive list contains 2006 words and the negative list has 4783 words. Both lists also include some misspelled words that are frequently present in social media content. Sentiment categorization is essentially a classification problem where features containing opinions or sentiment information should be identified before the classification. Yet another work on sentiment classification and opinion mining is [4] dealing with assigning positive, negative or neutral sentiment to a text using text orientation and linguistic features.

Pang and Lee [3] suggest a feature selection method removing objective sentences by extracting subjective ones. They proposed a text-categorization technique able to identify subjective content based on identifying graph minimum cuts. Gann [9] applies a feature selection scheme where $6,799$ tokens are first selected based on Twitter data. Then, each token is assigned a sentiment score, called TSI (Total Sentiment Index), featuring itself as a positive token or a negative one. TSI is computed as a function of the number of times a token appears in positives tweets and the number of time it appears in negative tweets.

In [8] recommendation of tags is done by training a classifier that maps audiovisual features from millions of you-tube video to supplied tags by an uploader of videos. The system learns a vocabulary of tags and suggests tags relevant to the video. A detailed analysis of usefulness of comments is presented in [6]. This work investigates the influence of comment sentiment on the ratings of the comment. The authors also predict the community acceptance of an unrated comment using Support Vector Machine classification. They rely on term-based representation of the comments to categorize them as *accepted* or *not accepted*.

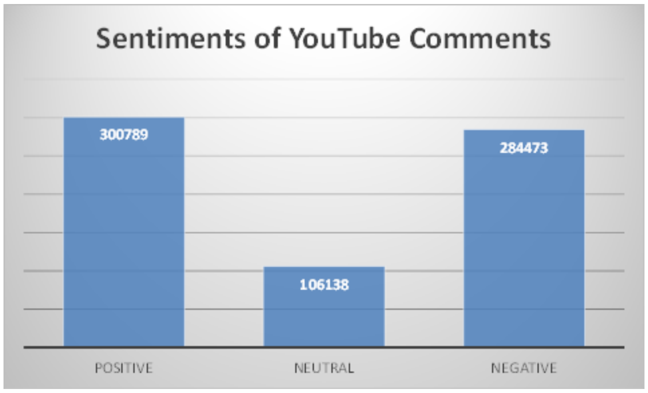


Figure 1: YouTube Dataset Statistics

In our project we apply these techniques in the context of automatic classification of comments. In addition, we consider the problem of category prediction based on comments and tags.

3 Dataset And Feature Extraction

In this section we discuss the properties of the dataset we use for our experiments as well as feature extraction techniques adopted.

3.1 YouTube Dataset

We apply classification techniques on a sample dataset taken from YouTube [11]. IThis dataset includes 691407 comments for 200 most trending YouTube videos in US in a two weeks period. The same source also offers a similar dataset for the most trending videos in GB. We will use a fraction of this dataset as our test set for sentiment analysis. The data includes comment data and video data. The comment data maps video IDs to comments and the number of likes and dislikes for the corresponding comments. The video data contains video title, channel title, category, tags, number of views, likes and dislikes and user reviews. The US videos belong to 15 categories.

3.2 Label Generation

Our goal is to classify YouTube comments into positive, negative or neutral sentiment classes. However, there are no labels showing the sentiment of the comments. Considering the large number of comments, it was not possible for us to manually label a reasonable fraction of this dataset. Consequently, we used the TextBlob [10] library for Python to generate the labels. TextBlob applies NLP (Natural Language Processing) techniques to estimate the **polarity** of each comment independently from the rest of the comments, and only based on its content. For us, it replaces the burdensome task of manual labeling. TextBlob generates polarity scores in the range of $(-1, 1)$. We convert them into categorical data using the following equation.

$$\text{sentiment} = \begin{cases} -1 & \text{polarity} < 0 \\ 0 & \text{polarity} = 0 \\ 1 & \text{polarity} > 0 \end{cases}.$$

Statistics on the size of each class is shown in Figure 1. Words that happen more frequently in positive and negative comments are shown as word clouds in Figures 2a and 2b. In Section 5.3, we do category classification based on tags and comments. The category labels are already available in video data.

3.3 Feature Extraction

For sentiment classification based on comments, we used TF-IDF [5] to convert comments into feature vectors. In this technique, TF (Term Frequency) counts the number of times a word has occurred in each document. However, the number of occurrences of words is in general higher for longer documents. Then, it is helpful to divide the number of word occurrences by the number of words in the document. The output of TF is a sparse matrix mapping documents to a normalized vector representing how many times each word has occurred in them. IDF (Inverse Document Frequency) accounts for the number of occurrences of a word in the entire document corpus. Namely, some words are more likely than others to happen in a given corpus of documents. Then, we scale TF terms by a decreasing function of the number of occurrences of each word in the whole corpus which is indeed the IDF term.

For category classification, we took two approaches. First, we used the same TF-IDF comment features. Second, we used the tags from the video data. Since video data has been gathered daily over two weeks, it may contain each video multiple times along with different tags. To remove duplicate entries, we removed punctuations and English stop words from the tags. Stop words are first removed. Stop words are commonly used words such as *the* which are filtered out before the analysis. Then, the videos were grouped based on their ID and tags were aggregated taking their union. We denote the tags of a video by a binary feature vector where each entry is zero in case the video was tagged by the corresponding tag and zero otherwise.

4 Content Classification Approach

We used 4 supervised learning methods for the problems of sentiment and category classification. In both cases, we deal with multi-class classification problems. In sentiment analysis, there are 3 classes corresponding to negative, neutral and positive sentiments while in category classification there are 15 classes corresponding to different categories. Theses techniques include Logistic Regression, Ridge classifier, Bernoulli Naive Bayes classifier and linear Support Vector Machine classifier.

Bernoulli Nave Bayes model is considered a common baseline for text classification due to its speed. This model assumes conditional independence between the features given their class. This assumption leads to relatively straightforward analysis and efficient running time. Nevertheless, the performance is less desirable than other methods because the independence condition is usually not satisfied in real-world problems.

Bernoulli Nave Bayes model estimates the classes probabilities as follows. Given $x^{(i)} = (x_1^{(i)}, x_2^{(i)} .....x_j^{(i)})$ being the i^{th} training example and K classes, we first maximize the joint likelihood of the data:

$$l(\phi_y, \phi_{j|y=1}, ..., \phi_{j|y=k}) = \prod_{i=1}^{m} \prod_{j=1}^{n} p(x_j^{(i)}|y^{(i)})p(y^{(i)}).$$

Maximizing the joint likelihood of the training data yields:

$$p(x_j^{(i)}|y^{(i)} = k) = \phi_{j|y=k} = \frac{\sum_{i=1}^{m} 1[x_j^{(i)} = 1 \, \Lambda \, y^{(i)} = k]}{\sum_{i=1}^{m} 1[y^{(i)} = k]},$$
$$p(y^{(i)} = k) = \phi_{y=k} = \frac{\sum_{i=1}^{m} [y^{(i)} = k]}{m},$$

where m is the number of training examples. Then, class label is chosen as the one with the highest posterior probability:

$$y = \text{argmax}_k \frac{\phi_{y=k} \prod_{j=1}^{n} \phi_{j|y}}{\sum_{k=1}^{K} \phi_{y=k} \prod_{j=1}^{n} \phi_{j|y}}$$

Multinomial Logistic Regression, also known as softmax regression is a generalization of Logistic Regression to more than 2 classes. Matrix of parameters θ is found maximizing the log-likelihood function defined as:

$$l(\theta) = \sum_{i=1}^{m} log \prod_{j=1}^{n} \frac{exp(\theta^T x^{(i)})}{\sum_k exp(\theta^T x^{(i)})}$$

The probability of each data point is then computed as:

$$p(y = k|x; \theta) = \frac{exp(\theta_k^T x^{(i)})}{\sum_{j=1}^{n} exp(\theta_j^T x^{(i)})}$$

Ridge classification is a generalization of regularized linear regression to classification problems. Class probabilities are estimated by first minimizing a least squares cost function with L2 norm:

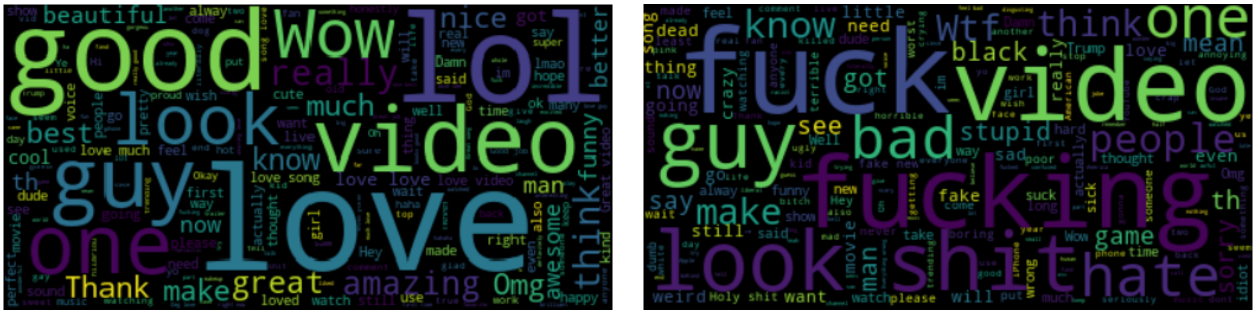
$$\sum_{i=1}^{m} \left\| W x^{(i)} - y^{(i)} \right\|_2^2 + \lambda \left\| W \right\|_2^2,$$

where $\lambda >= 0$ is the regularization constant. Once parameters are computed, continuous values are estimated for each data point through a linear prediction function $\hat{Y} = X\theta$. These values are then converted into discrete class values through proper thresholds. The second term is used to penalize the matrix W being too large. This will prevent the model from being unnecessarily complicated by overfitting the train data.

Linear Support Vector classifier (LinearSVC) applies a Support Vector Machine (SVM) with a linear Kernel. SVM is a maximum-margin classifier seeking to find a hyperplane $W^T X + b$ in a high-dimensional space solving the following *binary* constrained optimization problem.

$$\begin{aligned} &\min_{w_\tau, b_k, \tau_k} (\frac{1}{2} W_k^T W_k + c \sum_{i=1}^{m} \tau_k^{(i)}), \\ &\text{s.t.} \quad y^{(i)}(W_k^T x^{(i)} + b_k) \geq 1 - \tau_k^{(i)}, \\ &\text{s.t.} \quad 0 \leq \tau_k^{(i)} \leq 1, \end{aligned}$$

where W_k, τ_k and b_k are the parameters for the k-th SVM classifier corresponding to the k-th class. Support Vector Machines are inherently binary classifiers. To generalize them to the case of a multi-class problem with K classes, there are two possible methods: one-vs-one and one-vs-all. The one-vs-one method builds a classifier for every two classes. The class of one data point is then the one chosen by the most classifiers. The one-vs-all approach builds a classifier for each class against the data points of all the remaining classes. The class of a data point is the one whose classifier achieves the greatest margin. In our experiment, we apply the one-vs-all method.



(a) Word cloud for positive comments

(b) Word cloud for negative comments

Figure 2: Word cloud of common words in YouTube comments

Model	Train Accuracy	Test Accuracy
Logistic Regression	0.9578	0.9461
Bernoulli NB	0.7285	0.7212
Ridge Classifier	0.9419	0.9064
Linear SVC	0.9712	0.9527

Table 1: Sentiment classification accuracy of YouTube comments

5 Experiments and Evaluation

In this section, we present our experimental methodology, evaluation metrics and the results of the experiments. In all of the experiments, we discuss how the data was split into train, dev and test sets, but the results are reported only on train and test sets.

5.1 Sentiment Analysis

In order to measure the performance of different classification techniques on the YouTube data, we split the data from US comments into train and test subsets. To generate the train set, we randomly choose 80% of the comments from each of the three categories. The remaining comments form the dev set. We use 20% of the GB comments as the test set. To compare the performance of different methods we use accuracy, precision, recall and $F1$ score. Accuracy is defined as:

$$\text{Accuracy} = \frac{|\text{True predictions}|}{|\text{All predictions}|}$$

Precision indicates what fraction of the predictions made by a classifier are true.

$$P = \frac{|\text{True positives}|}{|\text{Predicted positives}|}$$

Recall indicates what fraction of the population of class are classified correctly by a classifier.

$$R = \frac{|\text{True positives}|}{|\text{Real positives}|}$$

Precision-recall is often used to determine the success rate of a model when the classes are have imbalanced population . Once the precision and recall metrics are obtained, we compute the harmonic mean also known as F1 score. F1 score is defined as

$$F1 = \frac{2PR}{P+R},$$

where P is the precision and R is the recall. The maximum value of $F1$ score is 1 which means the model reached perfect precision and recall.

In case of multi-class classification problems, we first need to binarize the output and produce a curve for each class label. We take a one-vs-all approach to deal with this issue. Precision and recall can then be generalized to the case of multi-class classification using a technique called micro-averaging. In this technique, every quantity in the above equations is replaced with the sum of corresponding quantities in different classes. For example, $|\text{True positives}|$ is converted to $|\text{True positives}_{\text{class1}}| + |\text{True positives}_{\text{class2}}|$.

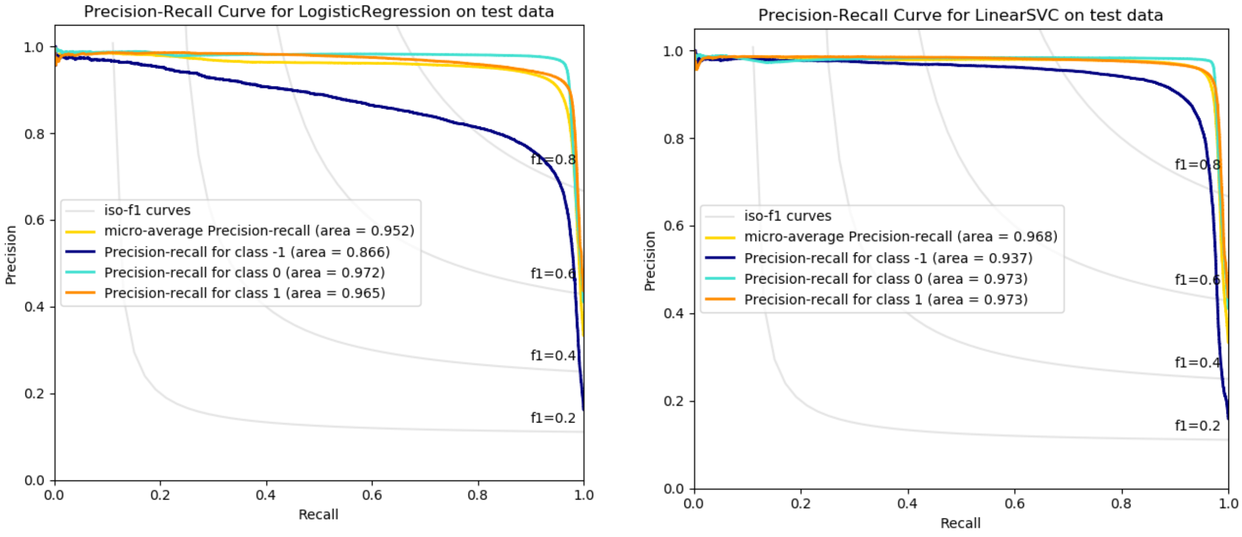
Table 1 shows the accuracy measured on train and test sets. Naive Bayes classifier has the worst performance among all methods. This result is predictable as the conditional independence condition required by the Naive Bayes classifier is not satisfied in this dataset. In particular, the probability of two words happening in a comment given the category of the comment are not independent from one another. On this dataset, SVM classifier has the best performance. It is observed that the other three models have reasonable accuracy while LinearSVC outperforms the rest of the models.

Figure 3a shows the Precision-Recall curve of Logistic Regression while Figure 3b shows that of LinearSVC. It is seen that precision and recall have an inverse relationship with each other. As the area under the curve increases, the precision and recall increase, where the increase of precision represents the increase of true positive rate and increase of recall represents the decrease of false negative rate. It happens because the more positive cases an algorithm reports, the more it is likely to report all positives points of the dataset (high recall), but it can also report more false positives (lower precision) and vice versa. We optimized the regularization term of all models using cross validation. For sentiment analysis, the default values of Scikit performed well. These values can be found in the first column of Table 2. Note that regularization is proportional to α for Ridge regression and *inverse* of C for Softmax regression and Linear SVC.

Figure 3 shows the calibration curve of the aforementioned techniques. It is observed that Naive Bayes classifier is not well-calibrated over the range $[0, 1]$. From among the remaining three methods, Logistic Regression is well-calibrated over the range $[0, 1]$ and is in general better calibrated than the other two methods.

5.2 Effect of Dimensionality Reduction

In this set of experiments, we examined how dimensionality reduction can affect the accuracy of predictions. Dimension reduction is usually applied to reduce the complexity of computations. We apply TruncatedSVD to project TF-IDF features to fewer dimensions than those of the original TF-IDF feature space. We use TruncatedSVD rather than Principal Component Analysis (PCA) because the TF-IDF features are stored in a sparse matrix by the SKLearn library which PCA cannot process. TruncatedSVD is very similar to PCA. However, the data is not centered around its mean in the for-



(a) Precision vs. Recall for Logistic Regression

(b) Precision vs. Recall for Linear SVC

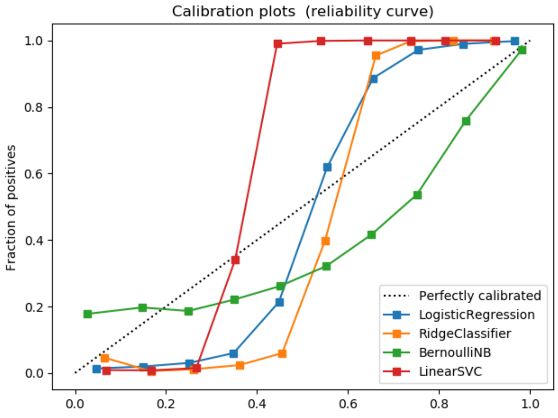


Figure 3: Calibration curves

	sentiment	category
Softmax Regression	$C = 1.0$	$C = 1.0$
Linear SVC	$C = 1.0$	$C = 0.05$
Ridge Classifier	$\alpha = 1$	$\alpha = 10$

Table 2: Optimal regularization constant

mer. Figure 4 shows the accuracy of sentiment analysis with TruncatedSVD applied to TF-IDF features versus number of dimensions of the projection space. It is observed that accuracy increases with the number of dimensions. Nevertheless, with 175 dimensions, it is still far from the case where no dimensionality reduction is applied. It seems that in this dataset, 175 dimensions is not enough to project the data without loosing much information. We could not experiment with higher dimensions due to memory limitations of our desktops.

5.3 Category Classification

First, we report the results of experiments for category classification using TF-IDF comment features. In this method we try to detect the category of the video corresponding to a comment using the

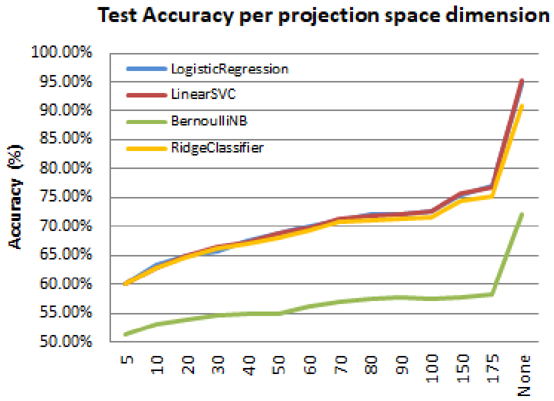


Figure 4: Accuracy vs. project space dimensions

TF-IDF features of the comments. To split the data into train, dev and test, we randomly chose 80% of the videos and their associative tags as the train dataset. The remaining 20% of videos were split equally into dev and test sets. We did not use GB videos in this experiment as their categories were not the same as US videos. the same datasets used for sentiment analysis. Table 5a shows the accuracy of different techniques for category classification using comment features. It is seen that the results are not as good as sentiment classification. Of course, considering there are 15 categories versus 3 sentiment classes, this problem is deemed to be harder. Yet, the probability of a successful random category guess is only $1/15$ (6.7%). Consequently, category prediction based on TF-IDF features of comments is still a big improvement over random guessing. We then conducted category classification based on video tags. To generate train, dev and test datatsets, we first Table 5b summarizes the results. It is observed that tags can predict the category with much higher accuracy. Note that tags are considered a property of videos while TF-IDF features are a property of comments. In addition, tags are explicitly chosen to classify the videos. As a result, it is expected that they be more directly relevant to the content of the video and less noisy leading to higher category prediction accuracy. Although category prediction by tags

Model	Train Accuracy	Test Accuracy
Logistic Regression	0.6070	0.5322
Bernoulli NB	0.4554	0.4161
Ridge Classifier	0.6578	0.905
Linear SVC	0.6895	0.5780

(a) category classification accuracy using comments

Model	Train Accuracy	Test Accuracy
Logistic Regression	00.9987	0.7139
Bernoulli NB	0.4909	0.3835
Ridge Classifier	0.9955	0.7286
Linear SVC	0.9954	0.7286

(b) category classification accuracy using tags

Figure 5: Category prediction of YouTube videos

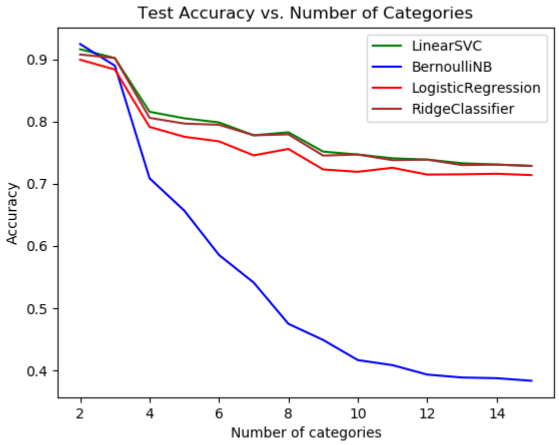


Figure 6: Accuracy versus projection space dimensions

has higher accuracy, each of these two methods can be used in different scenarios depending the data at hand. If we have clean data about videos at disposal with tags and the learning task is to classify the videos based on their tags, then tags are the better option. However, if all we have is comments and the association between comments and videos is not available (ex. the database containing videos is lost) and the prediction task is find what category of movies a comment is about, then we have to content with category classification using comment features. The optimal regularization constant for category classification based on tags is denoted in the second column of Table 2.

In another experiment, we examined the effect of the number of categories on accuracy. Intuitively, the larger the number of categories, the more difficult the classification problem is expected to be. To experiment this intuition, we created a list of categories with ascending order of each category population. We considered the first 2 categories, that is, the two categories with greatest number of videos, ignored the remaining videos in train and test set and measured the accuracy. The same experiment was repeated for $3, 4, \ldots, 15$ categories. Figure 6 shows how test accuracy decreases by increasing the number of categories.

6 Conclusion

We applied classification techniques to the problems of sentiment and category classification on a sample dataset of popular YouTube videos. We observed that, using TF-IDF features, the sentiment of YouTube comments can be estimated with high precision. Naive Bayes classifier showed the lowest accuracy among the 4 models we tried. This result conforms with expectation as Naive Bayes independence assumption is not satisfied by TF-IDF features. Multinomial Logistic Regression (Softmax regression), Ridge classifier and Linear SVC all had acceptable prediction accuracy with Linear SVC outperforming the other models. It was observed that Naive Bayes is not well-calibrated over the range $[0, 1]$ while Logistic Regression is. It is also better calibrated than the other models in general. Precision-Recall curves were plot and we observed that Precision and Recall have an inverse relation. We examined the effect of dimensionality reduction on TF-IDF features input to the aforementioned classification techniques. Increasing the number of dimensions of the projection space enhances accuracy. However, for the YouTube dataset, the accuracy achieved using dimensionality reduction is still far from the accuracy achieved with no dimensionality reduction. This shows that with the highest dimension of the projection space we could try, that is 175, there is still much information in the original data space which cannot be captured in the projection space. We showed that the accuracy of classifiers decreases with the number of categories. This is because choosing the right class becomes more difficult for higher number of classes.

We then considered the problem of category prediction using comment TF-IDF features and video tags. We observed that video categories can be estimated with higher accuracy using tags than comment features. This is because tags are clean explicit features designed to describe the content of videos while comment features are noisy and may not always be indicative of video categories. Despite, both methods can lend themselves well to different scenarios depending the data at hand. Comment features can be used if the association between the video and the comments is lost due to reasons like loss of database while tags can be used in ordinary situation when enough metadata is available about videos. Similar to entiment analysis, Linear SVC performed better in category classification.

References

[1] S. Chakrabarti. *Mining the Web: Discovering Knowledge from HyperText Data*. Science & Technology Books, 2002.

[2] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 168–177, 2004.

[3] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, 2004.

[4] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. *CoRR*, cs.CL/0205070, 2002.

[5] A. Rajaraman and J. D. Ullman. *Mining of Massive Datasets*. Cambridge University Press, 2011.

[6] S. Siersdorfer, S. Chelaru, W. Nejdl, and J. San Pedro. How useful are your comments?: Analyzing and predicting youtube comments and comment ratings. In *Proceedings of the 19th International Conference on World Wide Web*, pages 891–900, 2010.

[7] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, pages 199–222, 2004.

[8] G. Toderici, H. Aradhye, M. Pasca, L. Sbaiz, and J. Yagnik. Finding meaning on youtube: Tag recommendation and category discovery. In *Computer Vision and Pattern Recognition*, 2010.

[9] G. WJK. Twitter analytics for insider trading fraud detection system. In *Proceedings of the second ASE international conference on Big Data.ASE*, 2014.

[10] YouTube. TextBlob Library For Python. https://textblob.readthedocs.io/en/dev.

[11] YouTube. YouTube Database For Kaggle Competitions. https://www.kaggle.com/datasnaek/youtube.

7 Contributions

All of us contributed in the discussions, analysis, and choice of techniques. All of us helped with writing and reviewing the report. Peng

- implemented precision-recall code and conducted the corresponding experiments,
- generated statistics about data and conducted the experiments for dimensionality reduction.

Abhishek

- implemented sentiment polarity analysis using TextBlob and generated word clouds,
- searched the related work and reported on them.

Afshin

- implemented feature extraction, dimensionality reduction and classification techniques,
- conducted the remaining experiments.