DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2021

Sentiment Analysis on Youtube Comments to Predict Youtube Video Like Proportions

ISAC LORENTZ

GURJIWAN SINGH

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Sentiment Analysis on Youtube Comments to Predict Youtube Video Like Proportions

ISAC LORENTZ & GURJIWAN SINGH

Bachelor in Computer Science
Date: June 9, 2021
Supervisor: Alexander Kozlov
Examiner: Pawel Herman
School of Electrical Engineering and Computer Science
Swedish title: Sentimentanalys på Youtube-kommentarer för att förutsäga like-andel på Youtube-videor

Abstract

Social media websites are some of the world's most popular websites and allow all users to have a voice and express opinions and emotions. Using sentiment analysis, these users' opinions and emotions can be extracted and quantified. This study examines sentiment analysis on Youtube comments and how well the number of comments classified as positive, neutral and negative can be useful in predicting the like proportion of a Youtube video. Four different prediction formulas were tested that utilize neutral comments in different ways. Five different classifiers were examined with pretraining on Youtube comments, tweets, and a combination of tweets and comments. Some positive correlation between the predicted and actual like proportion was found. The best performing configuration was a logistic regression classifier trained only on Youtube comments with a prediction that attributes all comments classified as positive and neutral to likes. However, the errors of this type of prediction are so large that it likely has little real-world application. Possible method improvements include filtering out spam comments and include emoji sentiment.

Sammanfattning

Sociala medier är några av världens mest populära webbplatser och låter alla användare ha en röst och uttrycka sina åsikter och känslor. Med sentimentanalys kan dessa åsikter och känslor extraheras och kvantifieras. Denna studie undersöker sentimentanalys på Youtube-kommentarer och hur användbara antalet positivt, neutralt och negativt klassificerade kommentarer kan vara i förutsägelsen av like-proportionen på en Youtube-video. Fyra olika formler för förutsägelsen som använder neutrala kommentarer på olika sätt undersöktes. Fem olika klassificerare undersöktes med förhandsträning på Youtubekommentarer, tweets och en kombination av dessa. Ett positivt samband mellan faktisk och estimerad like-proportion hittades. Den bäst presterande konfigurationen var en logistisk regression klassificerare där alla neutrala och positiva kommentarer tillskrivs till likes och träning endast sker på Youtubekommentarer. Felen för dessa förutsägelser är så stora att förutsägelserna troligen har begränsad nytta i verkligheten. Möjliga förbättringar av metoden inkluderar att filtrera ut spamkommentarer och ta med emoji-sentiment.

Contents

Chapter 1

Introduction

In today's information society, information and opinions flow freely online. Through social media, everyone has a voice. This enables information and opinions to spread rapidly. The most popular social media platforms such as Facebook, Youtube, and Instagram each have more than a billion active users who generate content [1]. Besides creating content, most social media platforms enable users to react to content created by others and interact with other users through features such as liking, disliking, commenting, and sharing.

One particularly popular social media platform is Youtube. Youtube is a video-sharing platform where users can submit videos and other users can among other things like or dislike the videos and/or comment on the videos. In the comment section, users can express opinions and emotions related to (or even unrelated to) the video. This is also a way for video viewers to interact with the video creator. The number of likes and the like percentages of videos are important for creators on the platform since videos with exceptionally high dislike proportions give bad publicity. Obtaining methods of predicting the relationship between comment sentiments and like percentages, e.g. using sentiments expressed in comments to predict like proportion is of value to companies and content creators on Youtube as this platform is used for financial gain, to generate fame and the promotion of goods and services.

With a large amount of user data available through social media, it is possible to gain insights into the users' opinions and emotions.The field of sentiment analysis is the computational study of people's emotions, opinions, and attitudes expressed in written text [2]. With the use of statistical models such as machine learning and natural language processing, sentiment analysis can be performed to classify and quantify user emotions and opinions expressed on social media. With quantified user sentiments of Youtube comments, pre-

dictions can be made about the like percentage of the same video. This study will implement the same prediction formula presented by Hyberg & Isaacs [3] but with adaptations to handle neutral comments.

On March 30, 2021, Youtube announced the testing of a feature that allows content creators to hide the dislike count. The stated reason was "creator feedback around well-being and targeted dislike campaigns" [4]. The future of this feature seems uncertain at the moment, but there is a possibility that this feature will be made available to all content creators on Youtube and there is also a possibility that the dislike counts will always be hidden as a default. The possibility to hide dislike counts could be abused by content creators that their videos are highly disliked, perceived of low quality, contain misinformation, or are scams. Hiding dislikes deprives viewers of useful information about the videos and users risk viewing videos that they otherwise would have avoided, thus wasting time. If dislikes were to be hidden, predictions would be required to get insight into the dislikes of videos and other creators on Youtube. Knowing the like count and with a prediction of the like proportion, the dislike count of a video could be predicted.

1.1 Objective and Research Questions

The objective of the study is to determine if a simple prediction based on the sentiment of the comments can be useful for predicting like proportions on Youtube videos. We will also consider how neutral comments can be used in the prediction and compare performance when training on Youtube comments and tweets.

This paper searches to answer the following questions:

1. To what extent can a simple prediction based on sentiment analysis with standard machine learning algorithms be useful in predicting the like proportion on trending Youtube videos?
2. Is regarding neutral comments as positive or negative beneficial for prediction accuracy?
3. Does training the classifiers on a mix of tweets and Youtube comments outperform training on Youtube comments only?

1.2 Scope

Only out-of-the-box machine learning classifiers will be used in this study, with all classifiers applied with default settings from a Python library (Scikit-learn). The study will only use a limited testing dataset consisting of 3100 trending Youtube videos (from the Youtube trending toplists) from the UK and the US during a period in 2017. Training datasets are limited to 3500 Youtube comments and 14500 Tweets.

Chapter 2

Background

2.1 Youtube

Youtube is a platform for distribution and consumption of videos and is one of the most popular websites [5] and the social media platform with the second-highest number of active users [1]. Content creators consist of individuals, organizations, music artists among others. When a user/channel has published a video, other Youtube users can interact with the video. Users can give the video a like or a dislike, leave comments or reply to other comments. The maximum length of Youtube comments is 9999 characters, including spaces.

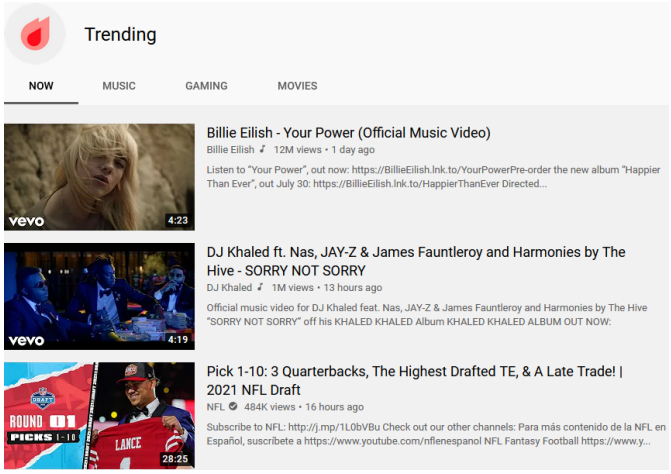


Figure 2.1: Top 3 Youtube Trending Videos in USA at 2021-04-24 13.58 EDT.

Youtube trending is a collection of nationwide top lists updated every 15

minutes, It aims to show among other things videos that are novel and "appealing to a wide range of viewers". Ranking indicators include view count, speed of views generation, age of the video, and the relative performance of a video compared to other videos from the same channel [6]

2.2 Twitter

Twitter is a social media and microblog platform and is one of the world's 20 most used social media platforms [1]. Twitter enables users to write short text posts with a length of up to 280 characters, thus making all tweets short. Twitter users can like and retweet (share) the tweets of other users.

2.3 Machine Learning, Supervised Machine Learning Classification

Machine learning is the study of computer algorithms that improve with experience and learn from data without having to be explicitly programmed [7, p.4], [8, p.xv]. This study is in the subfield of supervised machine learning, which refers to algorithms with learning functions that maps input to outputs based on labeled input-output pairs [9]. A common supervised learning task, which is also used in this study, is classification. Classification is the task of identifying which set of categories an observation belongs to. As an example, an email spam filter is a classifier model/algorithm that classifies incoming emails as spam or non-spam.

2.4 Supervised Machine Learning Algorithms

The classifiers that are used in this paper will be explained below.

2.4.1 Logistic Regression

Logistic regression is a way of applying principles of linear regression to classification problems [10]. The regression is modeling the probability that the dependent variable belongs to a certain category. It is a way to model the probabilities of classes via linear functions that sum to one and where all probabilities lie between 0 and 1. The outcome will be the categorical variable with

the highest probability. Logistic regression can be used to model two or more categorical variables as dependent variables [11, pp.130-131].

The linear regression used in the model to represent the probabilities has the form [11, p.131], [10]:

$$p(X) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_n x_n$$

where x represents the values and β the weights.

In multinomial logistic regression, the dependent variable can assume three or more categories. A reference category K is selected and each outcome is calculated based on K [12].

A logistic regression model of K different classes has the form:

$$\log \frac{P(Y = 1|X = x)}{P(Y = K|X = x)} = \beta_{0_1} + \beta_1^T x$$
$$\log \frac{P(Y = 2|X = x)}{P(Y = K|X = x)} = \beta_{0_2} + \beta_2^T x$$
$$\log \frac{P(Y = K - 1|X = x)}{P(Y = K|X = x)} = \beta_{0_{K-1}} + \beta_{K-1}^T x$$

where Y is the dependent variable, X is the independent variables, and β is the weights [13, p.119].

The probability for the dependent variable assuming a certain category can be expressed as [13, p.119]:

$$P(Y = k|X = x) = \frac{exp(\beta_{k_0} + \beta_k^T x)}{1 + \sum_{i=1}^{K-1} exp(\beta_{0_i} + \beta_i^T x)}, k = 1, 2, ..., K - 1$$

2.4.2 Support Vector Machine

Support vector machine (SVM) is a popular classification approach that performs well in many different settings. The support vector machine is an extension of the support vector classifier which in turn is an extension of the maximal margin classifier [11, p.350-354].

The support vector machine aims to create a hyperplane to separate the two data classes with the biggest possible margin, that is the distance between the hyperplane and the closest data point. By maximizing the margin, this enables support vector machines to have good generalization performance.

In the case where the two data classes are linearly separable, a linear support vector classifier can be used, so the hyperplane is just a straight line [14], [15]. When the classes are not suitable for linear separation, the hyperplane can be constructed using polynomials [13, p.418].

A multiclass SVM problem can be reduced to binary classification problems in several ways. This project utilizes the one versus all multiclass strategy where n binary classification models will be trained for n subproblems and the classifier with the largest confidence will decide the output [15].

2.4.3 Stochastic Gradient Descent Classifier

Linear classifiers such as linear support vector machines or logistic regression classifiers can be optimized using the stochastic gradient descent method and this is called a stochastic gradient descent classifier (SGDC). Stochastic gradient descent is one of the three types of gradient descent.

Gradient descent is a way to minimize a multi-variable function by following minus the gradient for setting the next function value. Traveling in the direction of the negative gradient will make the function decrease fastest. This step is repeated until the absolute value of the gradient falls below a predefined limit.

Calculating the gradient at every point can be time-consuming. Stochastic gradient descent combats this problem by approximating the gradient. This optimization will in theory guarantee convergence [16]. The approximated gradient will have a random error [17].

In stochastic gradient descent, instead of looking at the whole gradient based on all the data points, an approximation of the gradient is made from a single random data point that is a part of the gradient. Each iteration a new random data point is selected [17].

2.4.4 Naive Bayes

Naive Bayes classification is based on Bayes' theorem with the assumption that all attributes are conditionally independent given the class. That means that all features are assumed to contribute to the classification independent of each other. This assumption does not hold in many real-world scenarios but this type of classifier is still useful.

The multinomial naive Bayes is popular for use on texts. It assumes that the data follows a multinomial distribution. Each word in a document is assumed to have a position generated independently of the other words. Multinomial

naive Bayes consider the word counts, which not all naive Bayes classifiers do [18].

Complement naive Bayes was presented by Rennie et al to combat the faulty assumptions of multinomial naive Bayes for text data. It uses the complement of each class in estimating the model's weight [18].

2.5 Sentiment Analysis

Sentiment analysis is the computational study of people's opinions, emotions, attitudes, sentiments, and appraisals expressed in written text. The growth of the field has coincided with the growth of social media, which gives access to large volumes of opinion data [2].

Several levels of sentiment analysis exist, mainly document level, sentence level, and aspect level. Text can be classified on a polarity scale, running from for example negative-neutral-positive or very negative - negative - neutral - positive - very positive [2].

Raw text data needs to be transformed to be able to extract numerical features from text content. One way to do this is called the bag-of-words model, which is the feature extraction approach used in this study. This is popular feature extraction approach for sentences and documents [19, p.69], [20]. In this model, the grammar and word order is ignored. Only a list of the words and the associated frequencies are kept [21], [22, p.24].

2.6 Scikit-learn

Scikit-learn is a Python library for machine learning. It enables access to state-of-the-art classifiers with little need for configuration [23]. All classifiers used in this project are implemented with Scikit-learn.

2.7 Statistics

2.7.1 Pearson Correlation

Pearson's product-moment correlation coefficient for a sample gives a "standardized measure of the linear relationship between two variables" [24, p.84]. It provides both the direction and strength of the linear relationship. Pearson's correlation coefficient assumes normal distribution and linear correlation between the two variables. The coefficient varies from -1 to +1 where a coeffi-

cient close to -1 signals a negative linear relationship and a coefficient close to +1 signals a positive linear relationship. If the coefficient is 0, there is no linear relationship. The coefficient is calculated by dividing the correlation between two variables by the product of the standard deviation of the two variables [24, p.84]. The formula is given below:

$$r_{xy} = \frac{Cov(x,y)}{S_X S_Y} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

It is generally agreed among researchers that a coefficient smaller than 0.1 indicates a negligible relationship and a coefficient above 0.9 indicates a very strong relationship [25] but the values in between are not as easily classified. There exist no definite rule of thumb regarding categorical classification of the Pearson correlation into negligible, weak, moderate, strong, and very strong correlation or similar.

2.7.2 P-value

Given a test statistic and a null hypothesis the p-value is the probability of obtaining a test statistic at least as extreme as the actual value, given that the null hypothesis is true [24, p.355]. For the case of a Pearson's correlation coefficient, the p-value is the probability of two uncorrelated variables resulting in a correlation coefficient at least as extreme as the Pearson's correlation coefficient calculated from these two variables. A low p-value gives higher statistical significance of the Pearson's correlation coefficient [26]. P-values decrease as sample sizes increases [27].

2.7.3 Mean Absolute Error

Mean absolute error (MAE) is a measurement for model evaluation within regression analysis. The mean absolute error of a model on a test set is the mean of absolutes of the individual prediction errors in the test set [28]. It is formulated as:

$$MAE = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n} = \frac{\sum_{i=1}^{n}|e_i|}{n}$$

where y_i is the true value, x_i is the predicted value and n is the number of paired values.

2.7.4 Standard Deviation of Errors

Given two pairwise random samples $X_1, ..., X_n$ and $Y_1, ..., Y_n$ from two populations, the differences between the samples, $d_i = X_i - Y_i$ can be calculated.

The standard deviation for the differences are calculated as [29]:

$$s_d = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N} d_i}$$

2.8 Related Work

2.8.1 Sentiment Classification using Supervised Learning

Pang et al [18] was the first paper to classify movie reviews into positive or negative reviews using supervised learning and the bag-of-words model. They showed that standard machine learning models could outperform human-produced baselines on this task. In their results, SVMs outperformed naive Bayes slightly. In their discussion, they cover the important aspect that bag-of-words-based classifiers will perform poorly when a few words indicative of the true overall sentiment contrasts many words expressing another sentiment. In the case of movie reviews, a reviewer can write five good points about the movie and then end the review with "However, it can't hold up" [18]. They point out that a review like this is difficult to classify for bag-of-words-based classifiers. Gammon [30] used support vector machines to classify customer feedback data, which is short and noisy. This paper shows that supervised machine learning classifiers can classify texts that humans would find difficult, by using large feature vectors and feature reduction.

2.8.2 Sentiment Analysis on Social Media

Cunha et al [31] developed a classifier using Deep Neural Networks for classifying Youtube comments as positive, negative, or neutral. Their classifier had accuracy in the range of 60-84% when compared to a human annotator. Pak and Paroubek [22] developed a classifier with supervised learning where multinomial naive Bayes and SVMs were trained on tweets that they then used to classify tweets as positive, negative, or neutral. Bifet and Frank [32] recommended using stochastic gradient descent in training a linear classifier for

tweets. They also suggested that tweets could be labeled based on the emojis that the tweets contain.

2.8.3 Sentiment Analysis on Youtube Comments

Hyberg & Isaacs [3] have written a bachelor thesis on the topic of predicting Youtube video like proportions from comment sentiment. They also used a simple formula based on the numbers of comments classified as positive or negative by supervised learning classifiers for prediction but their training was only done on tweets and their classifiers only classified comments as positive or negative. They implemented SVM, logistic regression, and multinomial naive Bayes classifiers. Their results showed Pearson correlation coefficients between the actual like proportion and the predicted proportion based on their formula for their different classifiers and Youtube channels that ranged from -0.1 to 0.62. Their predicted like proportions were almost all in the 0.3 - 0.8 range, while the actual like proportions had a larger spread from about 0.05 to 0.95. This study aims to expand upon their work by training on Youtube comments, include more classifiers, include neutral comments in the classification, and test several formulas for prediction. Kumar Verma & Sahu [33] have written a course paper partly on the topic of predicting Youtube video like proportions from the sentiment of the comments. They used a neural network and added the comment count and view count beside the comments' sentiment in their prediction. Kumar Verma & Sahu [33] found that based on their study the correlation between comment sentiment and video like ratio was weak.

Chapter 3

Material and Method

The material and method section will explain; the gathering of data as videos and comments, the selective process regarding machine learning algorithms and feature extraction method, the identification and removal of unreadable data as well as the choice of prediction formulas and analysis of results. The code implementation in Python is inspired by the source code provided by Hyberg & Isaacs [3]. See Appendix A for the source code that we used.

3.1 Data Gathering

Labeled data (positive/negative/neutral) was needed to train our classifiers using supervised learning. Two labeled datasets of Youtube comments were found and used. The first dataset consists of about 2900 comments from the video "The Boring Company | Tunnels" by the channel The Boring company [34]. The comments were manually labeled positive, neutral, or negative [35]. The second YouTube dataset was found on Github [36] and features about 1100 Youtube comments. The comments appear to belong to one or several videos from the channel Dude Perfect and some videos featuring news footage. Automatic labeling, the lowest accuracy scenario, was assumed as no information was given about manual or automatic labeling. A few examples from the Youtube training dataset are shown in Table 3.1.

Table 3.1 **A subset of labeled comments from Youtube dataset**

Comment	Label
These guys are going to be the second channel that reach 100M subs	Positive
An interesting concept! Can't wait to see where this idea goes.	Positive
Make this comment have 1 like	Neutral
lol America, your resistance to public transport has led to some really weird ideas. Why not just build high-speed trains underground?	Neutral
New video idea, scare your wives and each other	Negative
this video was super boring.	Negative

Twitter training data was taken from the dataset Twitter US Airline Sentiment. The dataset contains about 14500 tweets from February 2015 where six American airlines were tagged. Tweets are manually labeled positive, negative or neutral [37]. Tweets were included to compare results against Youtube comments, they were also included since only a limited amount of labeled Youtube comments were found and since Twitter is also a social media platform, we believed this could be a useful complement. There is a possibility that training on a combination of Youtube comments and tweets outperform Youtube training only, and that is another reason we added tweets. A few examples from the Twitter training dataset are shown in Table 3.2. Descriptive statistics of the training datasets are given in Table 3.3. As seen in Table 3.3 the Twitter dataset is far larger than the Youtube dataset. Neutral comments are most common in the Youtube dataset while negative comments are most common in the Twitter dataset.

Table 3.3 shows the prediction accuracy of the testing dataset using an 80/20 test/train split for the five different classifiers. As seen in table 3.3 the Youtube dataset prediction accuracy lags the accuracy on the other two datasets. All classifiers have the highest prediction accuracy on the Twitter dataset (Table 3.3).

Table 3.2 **A subset of labeled comments from Twitter dataset**

Comment	Label
@VirginAmerica it was amazing, and arrived an hour early. You're too good to me.	Positive
@VirginAmerica plus you've added commercials to the experience... tacky.	Positive
@VirginAmerica when can I book my flight to Hawaii??	Neutral
@VirginAmerica my drivers license is expired by a little over a month. Can I fly Friday morning using my expired license?	Neutral
@VirginAmerica Your website is down and I'm trying to check in!	Negative
@united plus what about food? And taxis?	Negative

Table 3.3 **Descriptive statistics of training datasets**

Dataset	Number of comments	Number of comments labeled as positive	Number of comments labeled as neutral	Number of comments labeled as negative
Youtube	3489	543	2100	846
Twitter	14494	2341	3053	9100
Combined	18483	2884	5153	9946

Testing data was taken from the Kaggle dataset Trending YouTube Video Statistics and Comments [38], consisting of Youtube videos that were trending in the United States of America and the United Kingdom during a period in 2017. Data includes among other information video likes, video dislikes, and comment texts. The data is separated per country. All videos from the UK dataset and only the videos in the US dataset that were not also featured in the UK dataset were used. For videos that were trending several times and thus featuring several times in the dataset, all comments were added along with the last appearing like count and dislike count. Descriptive statistics for the testing dataset are given in Table 3.4. The average like proportion on the testing dataset is high (Table 3.4).

Table 3.4 **Descriptive statistics of Youtube trending dataset (testing dataset)**

Number of videos	Number of comments	Average video like proportion	SD of like proportion
3098	1111965	0.92945	0.12369

3.2 Data Cleaning

Some preprocessing of the data was necessary. Our chosen method could not handle all comments from the datasets without failing. Since the data files were read line by line, newlines () within the comments had to be removed. Certain emojis couldn't be properly encoded in our chosen file format (UTF-8) so those emoji characters had to be deleted. This did not affect the results whatsoever since the word preprocessing and tokenization we implemented through Scikit-learn (CountVectorizer) only considers alphanumeric characters for words with the parameters we used [39]. Regex and character replacing were used to make all datasets adhere to the same format. As for creating the Youtube and the combined datasets, the different subdatasets had to be combined since not all used classifiers supported training multiple times.

3.3 Training

All classifiers were trained on the training datasets with a test train split of 80/20 percent. This enabled us to see the accuracy of the classifiers on the training datasets. The same random state was used between the classifiers to make sure that the training is reproducible between the classifiers. Text feature extraction was done using the bag-of-words model using the CountVectorizer in Scikit-learn. As mentioned in section *2.5 Sentiment Analysis*, this a popular approach to feature extraction. The accuracy of the classifiers on the training datasets is shown in Table 3.5.

Table 3.5: **Training dataset prediction accuracy per classifier**

Classifier	Youtube training dataset	Twitter training dataset	Combined training dataset
SVM	0.66189	0.77268	0.74701
SGDC	0.66476	0.78889	0.76481
Comp. NB	0.58309	0.77234	0.72366
Multi. NB	0.61748	0.75957	0.71393
Logistic Reg.	0.68052	0.79890	0.77398

3.4 Classifiers

All used classifiers were used with the standard parameters in Scikit-learn except for logistic regression where the max_iter parameter was increased from the default value of 100 to 1000. This was done since the logistic regression classifier reached the maximum allowed iterations before the optimal solution to the classifying problem was found. Classifiers were selected based on what is suitable for text and social media sentiment analysis and what has been used in previous work in the field. Naive Bayes classifiers such as multinomial and complement naive Bayes are common for use in text classification due to being fast and simple to implement [18]. Stochastic gradient descent classifier was recommended for use on tweets by Bifet and Frank [32]. Since Youtube comments are also part of social media and tend to be of short length, like tweets, we believe this to be appropriate for this study. Support vector machines are used since they are effective at a variety of traditional text categorization tasks and generally outperform naive Bayes classifiers [18], [40]. Logistic regression is another classifier commonly used in sentiment analysis [41]. The International Workshop on Semantic Evaluation (SemEval) had between 2013 - 2018 a task about sentiment analysis on Twitter. Several years this task included variations of classifying the tweet on a scale from positive or negative. SVM- and logistic regression-based classifiers were used by several teams attempting the task of classifying tweets on a scale from positive to negative [42]. Since Youtube comments are also a part of social media and tend to be of short length like tweets, we believe that these types of classifiers could also perform well on Youtube comments. A few comments classified by the logistic regression classifier with Youtube dataset training are shown in Table 3.6. It can be noted that the second, fourth and sixth comments would

likely be classified differently by a human (Table 3.6).

Table 3.6 **A subset of comments classified by the logistic regression classifier with Youtube dataset training**

Comment	Classified sentiment
lol,great interview, Cardi B is funny...	Positive
Would be pretty great if it were not for Stormzy	Positive
The filter you used at the end was crazy	Neutral
What's on your nails???? Love it	Neutral
todays music sucks and is fake as fuck	Negative
Bojack Horseman is a show to watch late night	Negative

3.5 Prediction

Four formulas for making the prediction were tested. This will be explained below.

Prediction 1 / the base prediction assumes that only the number of comments classified as positive and negative contributes to the like proportion. The formula for the base prediction is given below:

$$predicted\ like\ proportion = \frac{N_{positive}}{N_{positive} + N_{negative}}$$

where $N_{positive}$ & $N_{negative}$ are the number of comments classified as positive and negative respectively.

A consequence of this formula for the base prediction is that the videos whose comments are only labeled as neutral had to be excluded since the denominator would be 0. This causes the size of the testing dataset to vary by small amounts between the classifiers for the base prediction.

The following three predictions consider neutral comments to some extent. Any factor for the neutral comments could be used in the numerator of the predicted like proportion but we have only considered those cases we believe make reasonable assumptions.

Prediction 2 assumes that all comments labeled as neutral contribute to dislikes. The predicted like proportion for prediction 2 is given below:

$$predicted\,like\,proportion = \frac{N_{positive}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive},\ N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Prediction 3 assumes that half of the neutral comments contribute to likes and that half of the neutral comments contribute to dislikes. The predicted like proportion for prediction 3 is given below:

$$predicted\,like\,proportion = \frac{N_{positive} + 0.5 \cdot N_{neutral}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive},\ N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Prediction 4 assumes that all neutral comments contribute to likes. The formula is given below:

$$predicted\,like\,proportion = \frac{N_{positive} + N_{neutral}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive},\ N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Given the correct like proportion as;

$$like\,proportion = \frac{number\,of\,likes}{number\,of\,likes\ +\ number\,of\,dislikes}$$

and the predicted like proportion for each video, the prediction error could be calculated. Results were calculated for each of the four predictions.

3.6 Evaluation

The accuracy of all classifiers on the training dataset was calculated. Knowing the actual and predicted like proportions on the Youtube trending dataset, the Pearson correlation, the p-value for the Pearson correlation, mean absolute error, and standard deviation of differences were calculated. This way the performance of the four different predictions and using all configurations of classifiers and training datasets could be compared.

Chapter 4

Results

This study aims to investigate if a simple prediction is useful in predicting like proportion on trending Youtube videos based on comments sentiments, how neutral comments should be incorporated in the prediction and if training the classifiers on Youtube comments and/or tweets produce the best predictions.

Pearson correlation, p-value for the Pearson correlation, mean absolute error, standard deviation of differences are presented in a table and a graph covering all classifiers with all predictions and training datasets. It can be noted that all classifiers showed significant p-values for all three training datasets and all four predictions so this will not be commented further. All classifiers, training datasets, and predictions show a positive linear correlation between the predicted like proportion and actual like proportion (Table 4.1-12). When a classifier performs the best or worst on all three of Pearson correlation, MAE, and standard deviation (SD), this will be commented on. Histograms showing the distribution of absolute errors between the different classifiers for each prediction and training dataset are also presented.

The results section is divided into five parts. The first part gives descriptive statistics for the training and testing dataset and training set prediction accuracy for the different classifiers. The following four parts present the results for the four different predictions on the testing dataset.

4.1 Base Prediction

This section shows the obtained results on the testing dataset using the prediction:

$$predicted\,like\,proportion = \frac{N_{positive}}{N_{positive} + N_{negative}}$$

where $N_{positive}$ & $N_{negative}$ are the number of comments classified as positive and negative respectively.

Classifier statistics for this prediction with training done on the different training datasets are presented in Table 4.1-3 and Figure 4.1. Histograms of mean absolute errors for the different classifiers and training datasets are presented in Figure 4.2.

This prediction performs the second-worst in regards to mean absolute errors after prediction 2. The Youtube training dataset gives this prediction the lowest MAE, followed by the combined dataset and the Twitter dataset giving the highest MAE. Overall the mean absolute errors for this prediction are quite high with support vector machine and stochastic gradient descent classifier performing the best and stochastic gradient descent classifier and multinomial naive Bayes performing the worst (Table 4.1-3, Figure 4.1). The lowest standard deviations are found for the Twitter dataset (Table 4.2). The highest Pearson correlations are found for the Youtube and the combined dataset (Table 4.1, 4.3). The two naive Bayes classifiers have the lowest Pearson correlations (Table 4.1-3, Figure 4.1).

The lowest standard deviations across all base prediction tables are 0.14068, 0.16045 and 0.16515, the rest range between 0.17236-0.22742. These are high standard deviations that give a rather insignificant MAE. A clear best-performing classifier on any of the training datasets is indeterminable (Table 4.1-4.3).

Multinomial naive Bayes performs the worst on Pearson, MAE, and standard deviation with Youtube training but no clear worst-performing classifier is found for the other two training datasets (Table 4.1-3).

In the histograms (Figure 4.2) a low distribution towards high absolute errors is preferable. We can see that all classifiers perform the worst with training on the Twitter dataset. Multinomial NB has a high distribution around 0.8-0.9 on the Twitter and combined dataset which is not preferable.

Table 4.1: **Classifier statistics for the base prediction with training on Youtube training dataset**

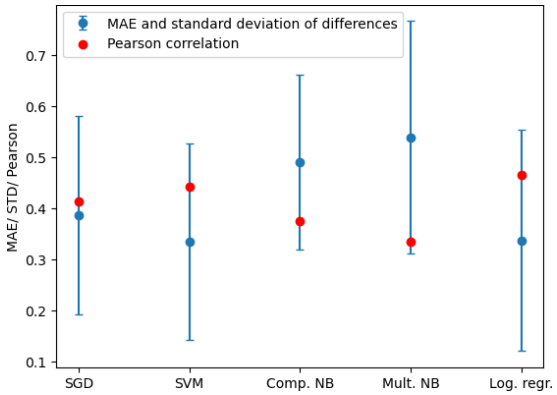
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.41362	p < 0.0001	0.38693	0.19404
SVM	0.44283	p < 0.0001	0.33489	0.19263
Comp. NB	0.37559	p < 0.0001	0.49056	0.17236
Multi. NB	0.33373	p < 0.0001	0.53966	0.22742
Logistic Reg.	0.46563	p < 0.0001	0.33737	0.21761

Table 4.2: **Classifier statistics for the base prediction with training on Twitter training dataset**

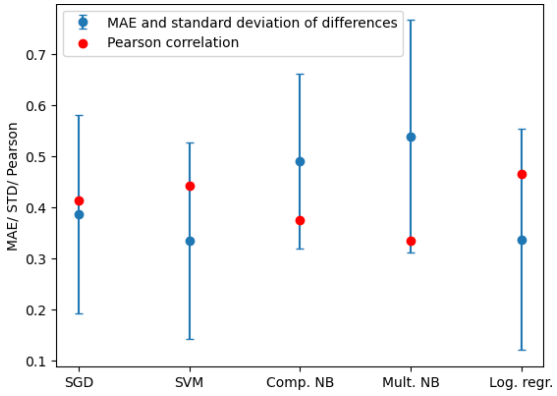
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.38950	p < 0.0001	0.53551	0.20438
SVM	0.38656	p < 0.0001	0.55413	0.18653
Comp. NB	0.35836	p < 0.0001	0.58391	0.16515
Multi. NB	0.31127	p < 0.0001	0.75389	0.14068
Logistic Reg.	0.37502	p < 0.0001	0.59465	0.18946

Table 4.3: **Classifier statistics for the base prediction with training on combined training dataset**

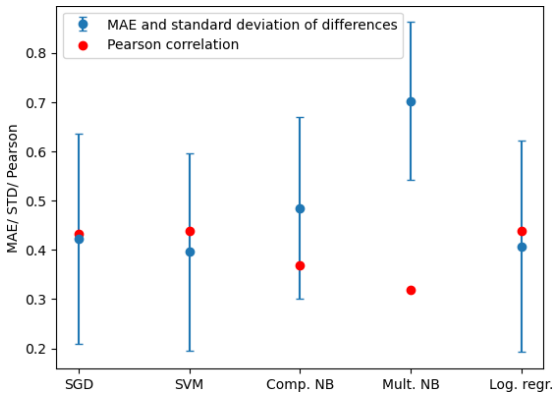
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.43269	p < 0.0001	0.42216	0.21374
SVM	0.43768	p < 0.0001	0.39615	0.20078
Comp. NB	0.36780	p < 0.0001	0.48524	0.18357
Multi. NB	0.31887	p < 0.0001	0.70162	0.16045
Logistic Reg.	0.43869	p < 0.0001	0.40739	0.21498



(a) Youtube Training

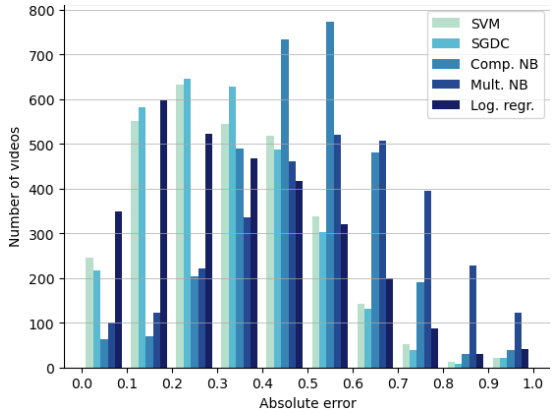


(b) Twitter Training

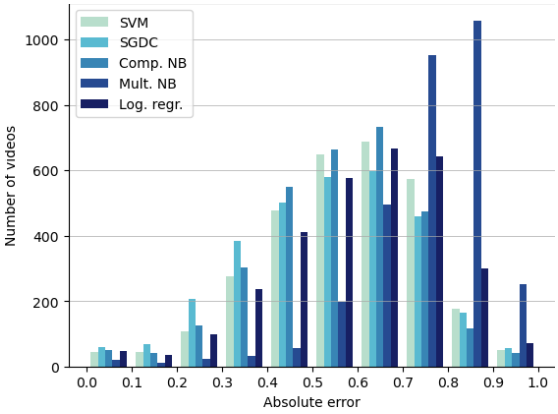


(c) Combined Training

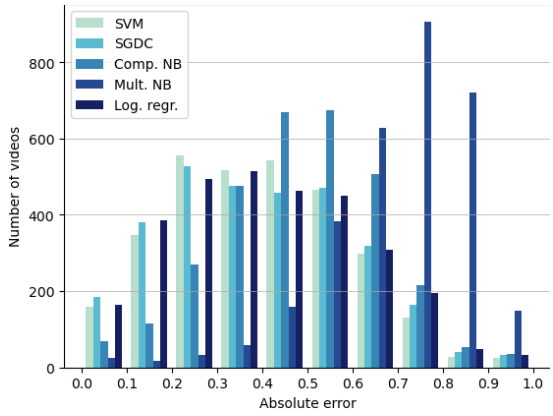
Figure 4.1: Classifier statistics for the base prediction



(a) Youtube Training



(b) Twitter Training



(c) Combined Training

Figure 4.2: Absolute error for all classifiers using the base prediction

4.2 Prediction 2, all neutral comments attributed to dislikes

This section shows the obtained results on the testing dataset using the following prediction that attributes all comments classified as neutral to dislikes:

$$predicted\ like\ proportion = \frac{N_{positive}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive}$, $N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Classifier statistics for this prediction with training done on the different training datasets are presented in Table 4.4-6 and Figure 4.3. Histograms of mean absolute errors for the different classifiers and training datasets are presented in Figure 4.4.

This prediction performs the worst of all four in regards to mean absolute errors. The Youtube training dataset gives this prediction the lowest MAE. Overall the mean absolute errors for this prediction are quite high with Support Vector Machines and Complement naive Bayes performing the best and multinomial naive Bayes performing the worst. Standard deviations for this prediction are lower than for the base prediction. Multinomial naive Bayes has the lowest standard deviation of errors for all training datasets and the lowest Pearson correlation for two out of three training datasets (Table 4.4-6, Figure 4.3). No clear best-performing classifier is found. Multinomial naive Bayes has is the worst performing classifier on the Twitter dataset but no clear worst-performing classifier is found on the other two training datasets. The histograms show the largest concentration for all classifiers and training datasets to the higher end, indicating high absolute errors (Figure 4.4).

Table 4.4: **Classifier statistics for prediction 2 with training on Youtube training dataset**

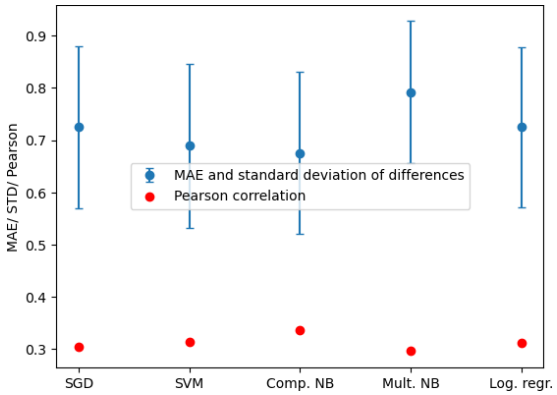
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.30421	p < 0.0001	0.72477	0.15443
SVM	0.31367	p < 0.0001	0.68922	0.15643
Comp. NB	0.33710	p < 0.0001	0.67518	0.15538
Multi. NB	0.29628	p < 0.0001	0.79191	0.13549
Logistic Reg.	0.31192	p < 0.0001	0.72478	0.15347

Table 4.5: **Classifier statistics for prediction 2 with training on Twitter training dataset**

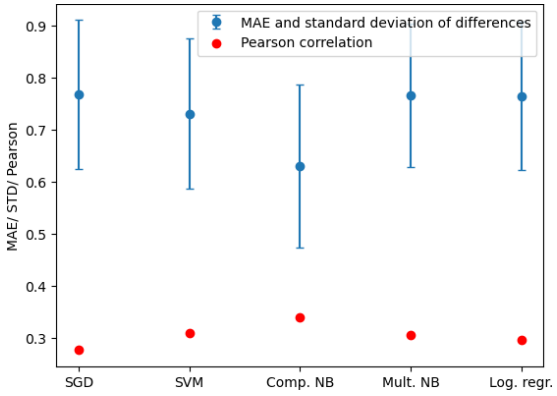
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.27676	p < 0.0001	0.76785	0.14294
SVM	0.30962	p < 0.0001	0.73104	0.14447
Comp. NB	0.34059	p < 0.0001	0.63126	0.15653
Multi. NB	0.30577	p < 0.0001	0.76693	0.13771
Logistic Reg.	0.29630	p < 0.0001	0.76490	0.14212

Table 4.6: **Classifier statistics for prediction 2 with training on combined training dataset**

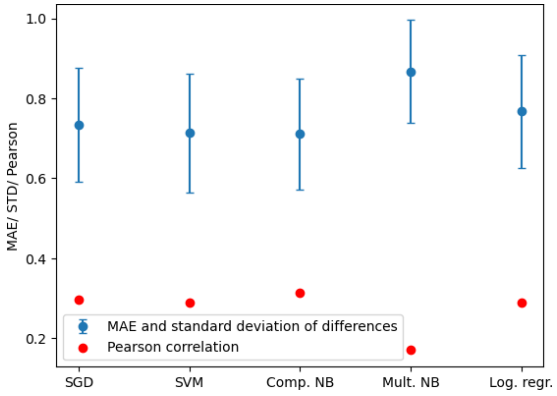
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.29606	p < 0.0001	0.73387	0.14242
SVM	0.28989	p < 0.0001	0.71287	0.14882
Comp. NB	0.31465	p < 0.0001	0.71071	0.13833
Multi. NB	0.16974	p < 0.0001	0.86731	0.12824
Logistic Reg.	0.28956	p < 0.0001	0.76758	0.14189



(a) Youtube Training

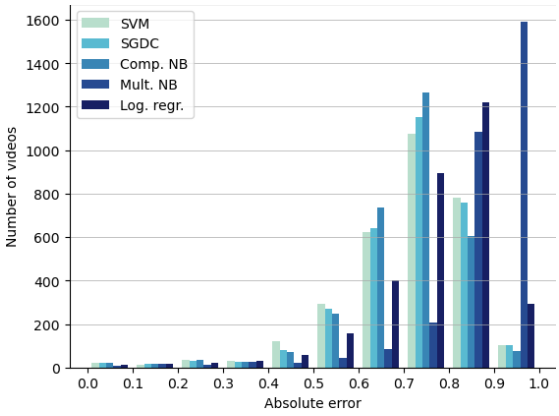


(b) Twitter Training

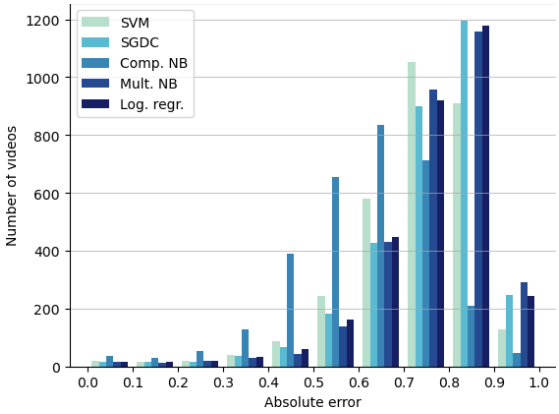


(c) Combined Training

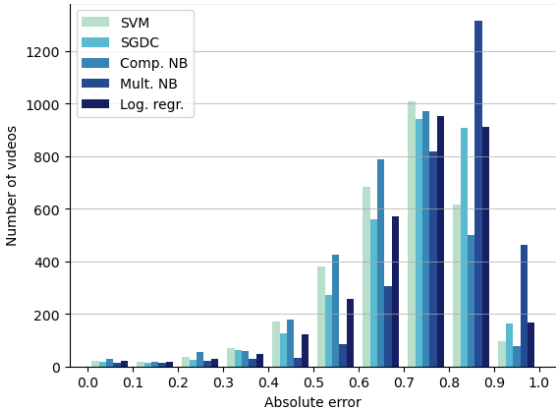
Figure 4.3: Classifier statistics for prediction 2



(a) Youtube Training



(b) Twitter Training



(c) Combined Training

Figure 4.4: Absolute error for all classifiers using prediction 2

4.3 Prediction 3, half of neutral comments attributed to likes

This section shows the obtained results on the testing dataset using the following prediction that attributes half of all comments classified as neutral to likes:

$$predicted\,like\,proportion = \frac{N_{positive} + 0.5 \cdot N_{neutral}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive},\ N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Classifier statistics for this prediction with training done on the different training datasets are presented in Table 4.7-9 and Figure 4.5. Histograms of mean absolute errors for the different classifiers and training datasets are presented in Figure 4.6.

Overall the mean absolute errors for this prediction are better than the first two predictions, with support vector machine and stochastic gradient descent classifier performing the best and complement and multinomial naive Bayes performing the worst (Table 4.7-9, Figure 4.6). No clear best-performing classifier is found. Multinomial naive Bayes performs the worst on the Youtube dataset. It can be noted in the histograms that with Youtube training all classifiers have the bulk of absolute errors between 0,3 and 0,6 while for the other two classifiers, the errors are more spread out and especially for the Twitter training dataset (Figure 4.6).

Table 4.7: **Classifier statistics for prediction 3 with training on the Youtube training dataset**

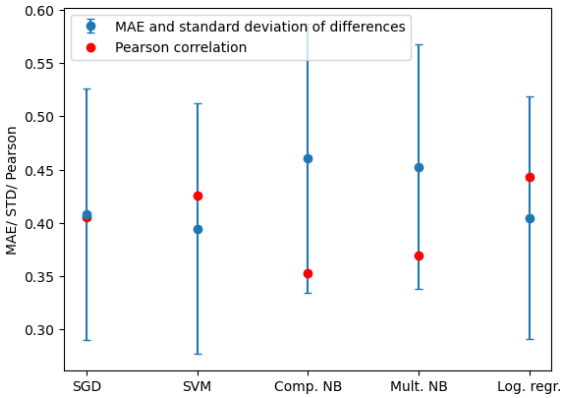
Classifier	Pearson correlation	Pearson correlation p-value	Mean Absolute Error	SD of Errors
SGDC	0.40494	p < 0.0001	0.40838	0.11826
SVM	0.42559	p < 0.0001	0.39461	0.11793
Comp. NB	0.35281	p < 0.0001	0.46053	0.12611
Multi. NB	0.36906	p < 0.0001	0.45277	0.11509
Logistic Reg.	0.44316	p < 0.0001	0.40487	0.11372

Table 4.8: **Classifier statistics for prediction 3 with training on the Twitter training dataset**

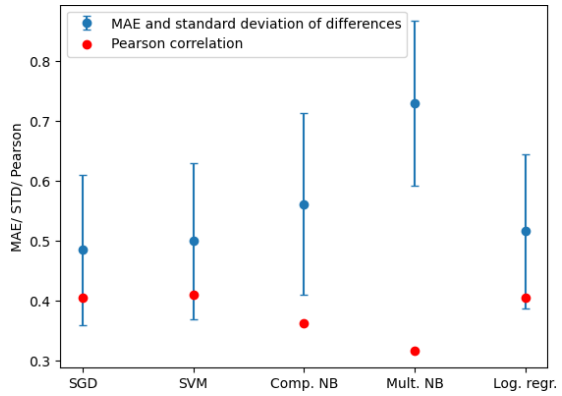
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.40579	p < 0.0001	0.48528	0.12568
SVM	0.41113	p < 0.0001	0.50000	0.13034
Comp. NB	0.36215	p < 0.0001	0.56193	0.15133
Multi. NB	0.31621	p < 0.0001	0.72968	0.13719
Logistic Reg.	0.40601	p < 0.0001	0.51614	0.12910

Table 4.9: **Classifier statistics for prediction 3 with training on the combined training dataset**

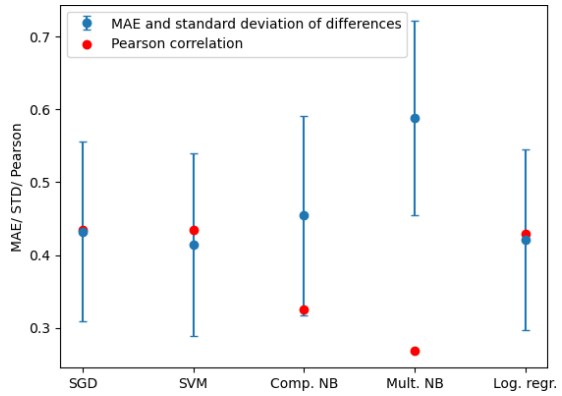
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.43478	p < 0.0001	0.43209	0.12365
SVM	0.43426	p < 0.0001	0.41451	0.12549
Comp. NB	0.32502	p < 0.0001	0.45416	0.13697
Multi. NB	0.26771	p < 0.0001	0.58771	0.13367
Logistic Reg.	0.42916	p < 0.0001	0.42107	0.12404



(a) Youtube Training

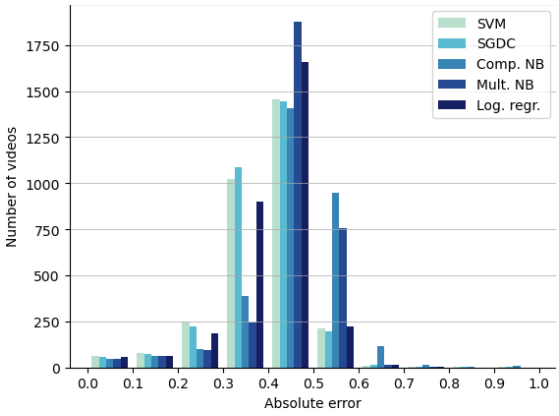


(b) Twitter Training

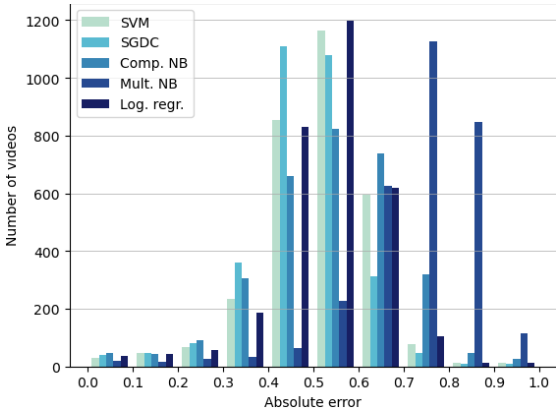


(c) Combined Training

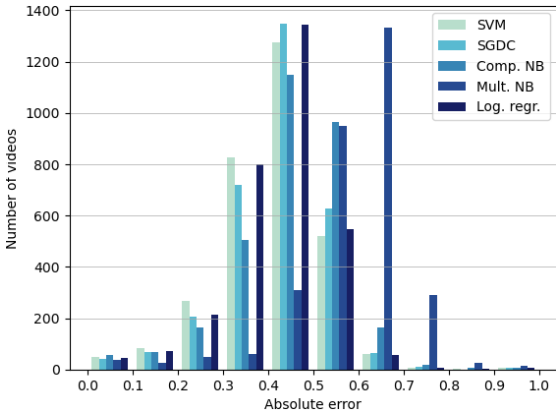
Figure 4.5: Classifier statistics for prediction 3



(a) Youtube Training



(b) Twitter Training



(c) Combined Training

Figure 4.6: Absolute error for all classifiers using prediction 3

4.4 Prediction 4, all neutral comments attributed to likes

This section shows the obtained results on the testing dataset using the following prediction that attributes all comments classified as neutral to likes:

$$predicted\,like\,proportion = \frac{N_{positive} + N_{neutral}}{N_{positive} + N_{neutral} + N_{negative}}$$

where $N_{positive}$, $N_{neutral}$ & $N_{negative}$ are the number of comments classified as positive, neutral and negative respectively.

Classifier statistics for this prediction with training done on the different training datasets are presented in Table 4.10-12 and Figure 4.7. Histograms of mean absolute errors for the different classifiers and training datasets are presented in Figure 4.8.

Overall the mean absolute errors for this prediction are clearly the best performing out of the four predictions with logistic regression and stochastic gradient descent classifier performing the best and complement and multinomial naive Bayes performing the worst (Table 4.10-12, Figure 4.8). This is the only prediction with mean absolute errors below 0.1 (Table 4.10). Logistic regression performs best on all three metrics with the Youtube training dataset. Logistic regression with the Youtube training dataset has the lowest standard deviation of all results (0.11233), the lowest MAE of all results (0.07632), and the second-highest Pearson correlation (0.46331 vs 0.46563).

The histograms show the largest concentration for all classifiers and training datasets towards lower absolute errors for the Youtube and combined training datasets. The distribution of absolute errors was more spread out for the Twitter training dataset. SVM, SGDC, and logistic regression were concentrated towards the lower end, complement naive Bayes around the middle, and multinomial naive Bayes towards the higher end. This shows that multinomial naive Bayes performing the worst, whereas SGDC performs among the best. A definite best-performing classifier is indeterminable (Figure 4.8).

Table 4.10: **Classifier statistics for prediction 4 with training on the Youtube training dataset**

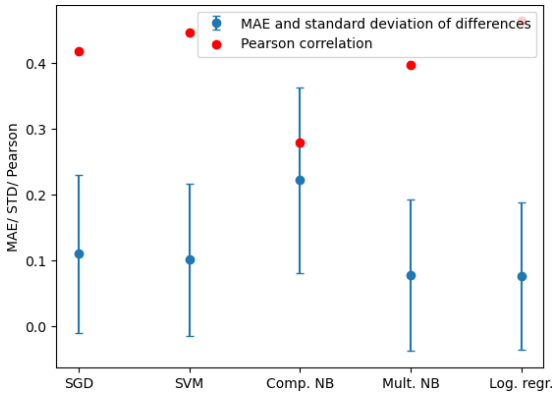
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.41826	p < 0.0001	0.10989	0.12010
SVM	0.44656	p < 0.0001	0.10124	0.11580
Comp. NB	0.27911	p < 0.0001	0.22192	0.14106
Multi. NB	0.39665	p < 0.0001	0.07732	0.11530
Logistic Reg.	0.46331	p < 0.0001	0.07632	0.11233

Table 4.11: **Classifier statistics for prediction 4 with training on the Twitter training dataset**

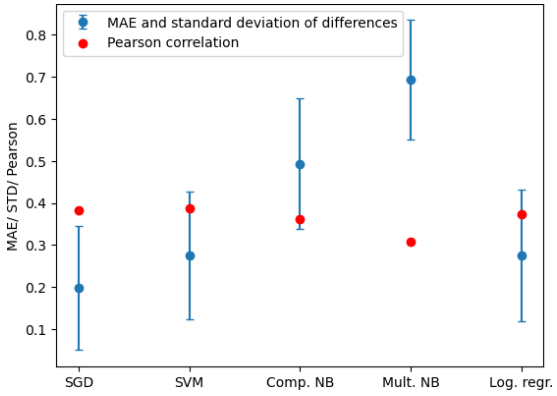
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.38269	p < 0.0001	0.19766	0.14798
SVM	0.38718	p < 0.0001	0.27526	0.15252
Comp. NB	0.36098	p < 0.0001	0.49312	0.15486
Multi. NB	0.30744	p < 0.0001	0.69260	0.14235
Logistic Reg.	0.37323	p < 0.0001	0.27444	0.15689

Table 4.12: **Classifier statistics for prediction 4 with training on the combined training dataset**

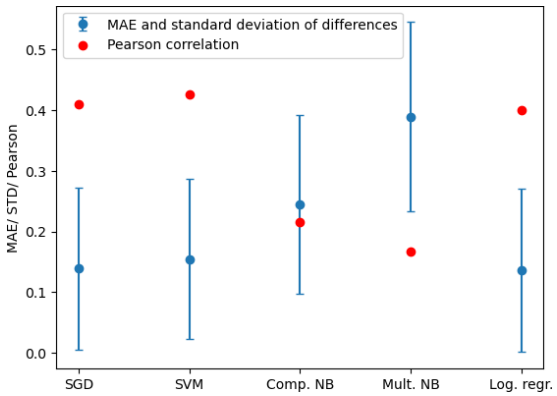
Classifier	Pearson correlation	Pearson corr. p-value	Mean Absolute Error	SD of Errors
SGDC	0.40950	p < 0.0001	0.13916	0.13359
SVM	0.42582	p < 0.0001	0.15423	0.13198
Comp. NB	0.21557	p < 0.0001	0.24483	0.14675
Multi. NB	0.16741	p < 0.0001	0.38965	0.15556
Logistic Reg.	0.39966	p < 0.0001	0.13592	0.13494



(a) Youtube Training

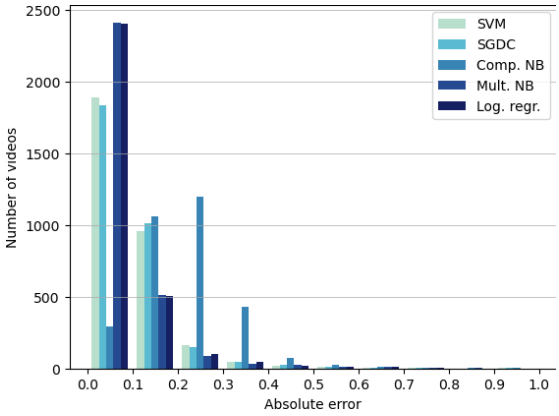


(b) Twitter Training

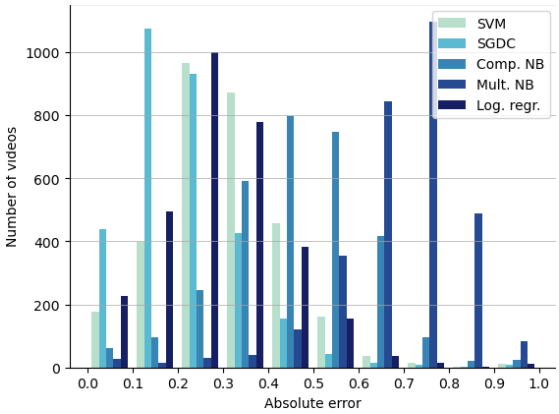


(c) Combined Training

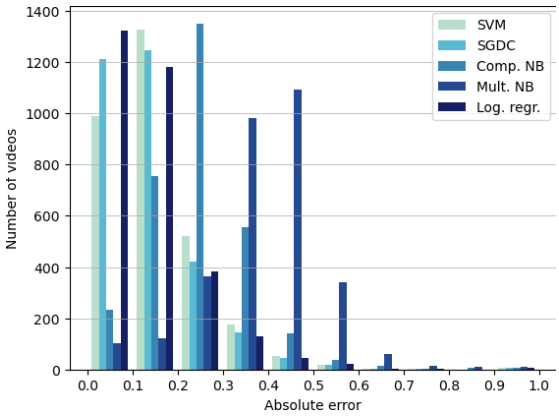
Figure 4.7: Classifier statistics for prediction 4



(a) Youtube Training



(b) Twitter Training



(c) Combined Training

Figure 4.8: Absolute error for all classifiers using prediction 4

Chapter 5

Discussion

5.1 Results

5.1.1 Results Analysis

Our methods have some merit in predicting the like proportions on Youtube videos, since all methods show positive Pearson correlations between the predicted and actual like proportion ranging from 0.17 to 0.47 . However, most classifiers gave on average predictions that were not very accurate and we, therefore, believe that our results will have little real-world use in regards to predicting like proportions for videos with hidden dislike numbers. Our methods need fine-tuning before they can be useful in a wider context. (see *5.3 Future Work*) Training the classifiers on the Youtube training set only consistently made them outperform the Twitter and the combined training datasets The only prediction with a small error is to attribute all comments classified as neutral to likes (Prediction 4) and training on the Youtube training dataset only. The best performing classifiers in this configuration were logistic regression and stochastic gradient descent classifier, this is the lowest MAE achieved in our study, with decent Pearson correlations but high standard deviations.

By looking at classified comments, we saw that even the best performing classifiers did not classify all comments correctly, or the same way as a human would do (Table 3.2). Misclassified comments have likely affected the result. This can somewhat be mitigated by having a larger testing dataset, but using bag-of-words for feature extraction will still see classifiers struggle when a few words reflecting the true sentiments are contrasted with many more words reflecting another sentiment (see *2.8.1 Sentiment Classification using Supervised Learning*).

The testing data used has on average a very high like proportion, 93% (see *3.1 Data Gathering*). This leads to any prediction regardless of method to give a strong correlation if the predicted like proportions are high. Instances where comments classified as neutral are attributed to likes will therefore give results with higher correlation. This would explain our results for classifiers where all neutral comments being attributed to likes having higher correlations. Therefore the attribution of neutral comments to likes may not be revealing of a relation between comments classified as neutral and like proportions on Youtube but may rather be a bias caused by the inclination for higher prediction proportions being favored. Such a bias would also indicate that neutral comments attributed to dislikes would give weaker Pearson correlations as these results would give a lower like proportion. This is also reflected in the tables of section 4.2 Prediction 2, the Pearson correlations of these tables are on average the lowest of all tables.

5.1.2 Interpretation of Metrics

The base prediction uses only comments labeled as positive and negative to calculate the like proportion, this was deemed to be the base prediction, since it resembles the metrics provided by Youtube the most. This means that there are no neutral counts provided on the website, only likes and dislikes. Hence, inserting comments classified as neutral into the calculation of like proportions is an additional act to explore the effects of assuming half, none, or all of comments labeled as neutral contributing to likes.

The results display the Pearson correlation, mean absolute error, and standard deviation of errors for a specified classifier under a specific testing dataset. The strength of a Pearson correlation varies depending on the context. Referring to previous studies in Sentiment Analysis for an interpretation of correlations may give us a clue about the strength, however, no studies were found for this specific context. We can use mean absolute errors and the standard deviation of errors to further speculate the interpretation of our Pearson correlations. A small standard deviation would indicate that the mean absolute error is an accurate representation of most errors for a given classifier and training dataset. By analysing the Pearson correlations in relation to multiple of these accurately representative errors we can get an indication of the strength of a Pearson correlation, e.g. if a +0.4 or -0.4 is a weak or moderate correlation.

Such speculation would require identifying values for mean absolute errors and standard deviations of errors as low, high, or moderate. We will consider a standard deviation for the differences between actual and predicted like propor-

tion of 0.05 as low as this suggests that under a normal distribution, 95.4% of errors fall within 2 standard deviations, e.g +/-0.1 of the corresponding mean absolute error. A 0.2 value range covers 20% of the 0 to 1 range of values for errors under a normal distribution. A standard deviation of 0.1 would suggest that under a normal distribution 95.4% of errors to fall within +/-0.2 of the corresponding MAE, giving a 0.4 range of possible values. 1 standard deviation, eg, 68.2% of these values would fall under a 0.2 value range under a normal distribution. Such a range for deviation is rather large since it ranges on 20% of all possible values for only slightly more than 2/3 of its content and further deviation for the rest of the content. This indicates that a 0.1 standard deviation could be considered as high while 0.05 could be considered low.

While almost 66% of errors getting distributed over 20% of the value range under a normal distribution and the residual errors over an even larger range, it is rather uncomplicated to identify as a value on the higher end. It is still difficult to conclude strict values for where low, moderate, and high begins. However, these values do give us some insight into what MAEs have a too sizeable standard deviation for drawing accurate conclusions in regards to our method of interpreting Pearson correlations.

To apply this speculative interpretation to our results, we would need low standard deviations, all of our standard deviations are higher than 0.1 and thus we cannot use our speculative method for interpretation of Pearson correlations. Under the Youtube training dataset in section 4.4 Prediction 4 the classifiers multinomial naive Bayes and logistic regression have similar MAEs and standard deviations of errors, but with very different Pearson correlations. In general, the results show that a higher Pearson correlation is usually accompanied by a lower MAE in relation to other values in the same table. Further, due to the p-value being low in all cases, the Pearson correlation is not of random nature. The Pearson correlation is also well above 0 in all cases. These factors lead us to conclude that there is some positive correlation between a Youtube video's like proportion and the predicted like proportion which has a relation to the sentiment of the video's comments classified by the given classifiers.

5.1.3 Considerations

Deeming a 1 in Pearson correlation as the best outcome might not be realistic in relation to our prediction formulas. That would require the relation between sentiments to be fully indicative of the like proportion, which might differ in reality, positive and negative sentiments might lead to likes or dislikes to differing extents.

Secondly, conclusions of how well certain classifiers worked might not be accurate, due to an improper prediction model and not necessarily a poor sentiment analysis. In reality, such a model might be much more complex and would take into account the precise relation between sentiments of comments and like proportions.

5.2 Datasets

Before analysing results it is important to examine biases in our data for a better understanding of the results.

5.2.1 Training Dataset

The Youtube training dataset is rather small, we only managed to find 2 labeled sentiment analysis Youtube datasets. A larger training dataset could result in better performance on the testing dataset.

5.2.2 Testing Dataset

The testing dataset includes only trending videos, these might have different like proportions than the average Youtube video. The relation between sentiments of comments and like proportions may be more homogenous between trending videos and might not reflect the average relation. Videos outside of the trending sections may differ in such a relation. It may vary depending on community, content creator, type of channel, and many more factors.

5.3 Future Work

Accuracy of the predictions could possibly be improved by using more advanced classifers such as neural networks or using other feature extraction approaches than bag-of-words. As videos of Youtube trending are not necessarily representative of Youtube as a whole, it would be interesting to see this method applied to a large testing dataset that is more representative of Youtube as a whole. All comments on the Youtube videos were used in our prediction but some comments are spam and do not represent views on the video. Spam videos could be filtered out so that only non-spam comments are used in predicting the like proportion. This would require manual work or an accurate way to detect spam comments. One could also examine if the proportion of

spam comments is predictive of the like proportion. Emojis are common in Youtube comments. It would be interesting to include both the text and emojis in the sentiment analysis since emojis can be indicative of sentiment in the comment.

If Youtube video like proportions could be accurately predicted based on the sentiment one interesting follow-up question can be asked. If it was possible to accurately predict the end-of-life-cycle video like proportion from the sentiment from the comments at an earlier point in a video's life-cycle, this would have some implications for content creators and advertisers. If a content creator predicts that their video will receive a high end-of-life-cycle dislike proportion, they could choose to delete it to try to avoid negative reputation and/or publicity. Advertisers could use predicted end-of-life-cycle like proportions as a metric when deciding which content creators to do collaborations with or change existing collaborations with content creators that are predicted to get high dislike proportions on their videos. This way, advertisers could try to disassociate themselves from Youtube content that receives bad press.

Chapter 6

Conclusion

Our study investigated the prediction of like proportions of trending Youtube videos using a simple prediction based on the number of comments classified as positive, neutral and negative. Some correlation was found between the like proportions of Youtube videos and the predicted like proportions based on the sentiment of their comments classified by the classifiers used in this study (SGDC, SVM, multinomial NB, complement NB, and logistic regression). But due to the high prediction errors and standard deviations obtained this type of prediction we used is not of great use in predicting like proportions for trending Youtube videos in real-world scenarios, such as if dislike counts were to be hidden. Best predictions were obtained when comments labeled as neutral were attributed to likes in the likes proportion, though this might be due to biases (see *5.1.1 Result Analysis*), and logistic regression was the best performing classifier in that case. The classifiers performed better with only Youtube comments as training compared to a larger dataset of Youtube comments and tweets.

Chapter 7

Acknowledgements

The authors thanks our supervisor Alexander Kozlov and also Jacob Brännström, Max Persson, Susanna Zeitler Lyne, and Charlotta Lorentz who provided helpful feedback.

Bibliography

[1] Statista. *Most popular social networks worldwide as of January 2021*. https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users. Accessed 2021-04-01. 2021.

[2] Lei Zhang and Bing Liu. "Sentiment Analysis and Opinion Mining". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 1152–1161. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_907.

[3] Martin Hyberg and Teodor Isaacs. "Predicting like-ratio on YouTube videos using sentiment analysis on comments". MA thesis. KTH Royal Institute of Technology, 2018.

[4] Youtube. *Youtube tweet regarding testing of hidden dislike count feature*. https://twitter.com/YouTube/status/1376942486594150405. Accessed: 2021-05-29. 2021.

[5] Alexa Internet. *The top 500 sites on the web*. https://www.alexa.com/topsites. Accessed 2021-03-25. Apr. 2021.

[6] Youtube Help Center. *Trending on Youtube*. hhttps://support.google.com/youtube/answer/7239739?hl=en. Accessed 2021-03-25. 2021.

[7] Daniel Kirsch Judith Hurwitz. *Machine Learning for Dummise , IBM Limited Edition*. Hoboken, New Jersey: John Wiley Sons, 2018.

[8] T.M. Mitchell. *Machine Learning*. McGraw-Hill, 1997. ISBN: 9780071154673.

[9] "Supervised Learning". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 1213–1214. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_803. URL: https://doi.org/10.1007/978-1-4899-7687-1_803.

[10] "Logistic Regression". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 780–781. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_951.

[11] James G. "Logistic Regression". In: *An introduction to Statistical Learning; with Applications in R*. 7th ed. Springer, 2013, pp. 130–131.

[12] Mikael Klockare. *Logit oddskvot och sannolikhet - En analys av multinomial logistisk regression*. 2019.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. New York, NY: Springer, 2017.

[14] Xinhua Zhang. "Support Vector Machines". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 1214–1220. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_810.

[15] Scikit-learn developers. *1.4 Support Vector Machines*. Scikit-learn. 2021. URL: https://scikit-learn.org/stable/modules/svm.html.

[16] Jay Dawani. "Gradient descent". In: *Hands-On Mathematics for Deep Learning*. Birmingham: Packt, 2020.

[17] Forsyth D. "Learning to Classify". In: *Probability and Statistics for Computer Science*. Cham: Springer, 2018, pp. 253–279. DOI: https://doi.org/10.1007/978-3-319-64410-3.

[18] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs Up? Sentiment Classification Using Machine Learning Techniques". In: *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. Association for Computational Linguistics, 2002, pp. 79–86. DOI: 10.3115/1118693.1118704.

[19] Yoav Goldberg and Graeme Hirst. *Neural Network Methods in Natural Language Processing*. Morgan Claypool Publishers, 2017. ISBN: 1627052984.

[20] Navin Sabharwal and Amit Agrawal. "Introduction to Natural Language Processing". In: *Hands-on Question Answering Systems with BERT: Applications in Neural Networks and Natural Language Processing*. Berkeley, CA: Apress, 2021, pp. 1–14. ISBN: 978-1-4842-6664-9. DOI: 10.1007/978-1-4842-6664-9_1. URL: https://doi.org/10.1007/978-1-4842-6664-9_1.

[21] Scikit-learn developers. *6.2 Feature extraction*. Scikit-learn. Apr. 2021. URL: https://scikit-learn.org/stable/modules/feature_extraction.html.

[22] Bing Liu. *Sentiment Analysis and Opinion Mining*. Morgan Claypool, 2012.

[23] Fabian Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (Jan. 2012).

[24] Paul Newbold et al. *Statistics for Business and Economics*. Global. Pearson, 2013.

[25] Patrick Schober, Christa Boer, and Lothar Schwarte. "Correlation Coefficients: Appropriate Use and Interpretation". In: *Anesthesia Analgesia* 126 (Feb. 2018), p. 1. DOI: 10.1213/ANE.0000000000002864.

[26] Haldun Akoglu. "User's guide to correlation coefficients". In: *Turkish Journal of Emergency Medicine* 18.3 (2018), pp. 91–93. ISSN: 2452-2473. DOI: https://doi.org/10.1016/j.tjem.2018.08.001.

[27] Matthew Thiese, Brenden Ronna, and Ulrike Ott. "P value interpretations and considerations". In: *Journal of Thoracic Disease* 8 (Sept. 2016), E928–E931. DOI: 10.21037/jtd.2016.08.16.

[28] "Mean Absolute Error". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 806–806. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_953.

[29] Jack Prins et al. "7.3.1.1. Analysis of paired observations". In: *NIST/SEMATECH e-Handbook of Statistical Methods*. NIST/SEMATECH, 2012. DOI: https://doi.org/10.18434/M32189.

[30] Michael Gamon. "Sentiment classification on customer feedback data: Noisy data, large feature vectors, and the role of linguistic analysis". In: *Proceedings of the 20th International Conference on Computational Linguistics*. COLING '04. Association for Computational Linguistics, Jan. 2004. DOI: 10.3115/1220355.1220476.

[31] Alexandre Cunha, Melissa Costa, and Marco Pacheco. "Sentiment Analysis of YouTube Video Comments Using Deep Neural Networks". In: *Artificial Intelligence and Soft Computing. ICAISC 2019*. Cham: Springer International Publishing, May 2019, pp. 561–570. ISBN: 978-3-030-20911-7. DOI: 10.1007/978-3-030-20912-4_51.

[32] Albert Bifet and Eibe Frank. "Sentiment Knowledge Discovery in Twitter Streaming Data". In: *DS'10: Proceedings of the 13th International Conference on Discovery Science*. DS'10. Canberra, Australia: Springer-Verlag, Sept. 2010, pp. 1–15. ISBN: 978-3-642-16183-4. DOI: 10.1007/978-3-642-16184-1_1.

[33] Pranay Kumar Verma and Prateek Sahu. *YouTube video comment sentiment analysis*. https://github.com/pranaykmr/YouTubeSentimentAnalysisandPredictingTrends/blob/master/Reporteport.pdf. Accessed 2021-04-01. 2020.

[34] The Boring Company. *The Boring Company | Tunnels*. https://www.youtube.com/watch?v=u5V_VzRrSBI. Accessed 2021-04-07. Apr. 2017.

[35] Kristo Eklekta. *Relevance and Sentiment of YouTube Comments of a Vision Video*. Version version 1. Zenodo, Feb. 2021. DOI: 10.5281/zenodo.4533302.

[36] Kumar Verma et al. *Labeled comments*. https://github.com/pranaykmr/YouTubeSentimentAnalysisandPredictingTrends/tree/master/data. Accessed 2021-04-01. 2020.

[37] Crowdflower. *Twitter US Airline Sentiment*. https://www.kaggle.com/crowdflower/twitter-airline-sentiment. Oct. 2019.

[38] Jolly M. *Trending YouTube Video Statistics and Comments*. https://www.kaggle.com/datasnaek/youtube. Accessed 2021-04-03. Oct. 2017.

[39] Scikit-learn developers. *sklearn.feature$_e$xtraction.text.CountVectorizer*. Scikit-learn. May 2021. URL: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html.

[40] Thorsten Joachims. "Text categorization with Support Vector Machines: Learning with many relevant features". In: *Machine Learning: ECML-98*. Ed. by Claire Nédellec and Céline Rouveirol. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 137–142. ISBN: 978-3-540-69781-7.

[41] Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. "Identifying Sarcasm in Twitter: A Closer Look". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*. HLT '11. Portland, Oregon: Association for Computational Linguistics, 2011, pp. 581–586. ISBN: 9781932432886.

[42] Saif Mohammad et al. "SemEval-2018 Task 1: Affect in Tweets". In: *Proceedings of The 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1–17. DOI: 10.18653/v1/S18-1001. URL: https://www.aclweb.org/anthology/S18-1001.

Appendix A

Source Code

```
from sklearn import svm
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import SGDClassifier
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import ComplementNB
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import csv
import numpy
from scipy import stats

YT = "Youtube dataset filepath"
twitter = "Twitter dataset filepath"
YTtwitter = "combined dataset filepath"

YoutubeTrendingDatasetCommentsFilepath = 'Youtube trending dataset comments filepath'
YoutubeTrendingDatasetVideosFilepath = 'Youtube trending dataset videos filepath'

def main():

    #base prediction youtube training
    sv = predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YT,'Youtube','base','Base prediction')
    #last parameter can be used for printouts and graph titles
    sg = predictionFunction(SGDClassifier(),'SGDC',YT,'Youtube', 'base','Base prediction')
    co = predictionFunction(ComplementNB(),'Comp. NB',YT,'Youtube' ,'base','Base prediction')
    mu = predictionFunction(MultinomialNB(),'Mult. NB',YT,'Youtube' ,'base','Base prediction')
    lo = predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YT,'Youtube','base','Base prediction')

    #can be used to create histograms
    values = [sv, sg, co, mu, lo]
    colors = [(0.72,0.87,0.8),(0.35,0.73,0.82),(0.22,0.52,0.71),
              (0.15,0.29,0.57),(0.09,0.12,0.39)]
    labels = ['SVM', 'SGDC', 'Comp. NB', 'Mult. NB', 'Log. regr.']
    plt.figure(1, figsize=(9,5))
    n, bins, patches = plt.hist(values, color=colors, label=labels)
    plt.xticks(bins)
    plt.grid(color='darkgrey', lw = 0.5, axis='y')
    plt.ylabel('Number of videos')
    plt.xlabel('Absolute error')
    plt.legend()
    plt.show()


    #base prediction twitter training
    #sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',twitter,'Twitter', 'base','Base prediction')
    #sg=predictionFunction(SGDClassifier(),'SGDC',twitter,'Twitter','base','Base prediction')
    #co=predictionFunction(ComplementNB(),'Comp. NB',twitter,'Twitter','base','Base prediction')
    #mu=predictionFunction(MultinomialNB(),'Mult. NB',twitter,'Twitter','base','Base prediction')
    #lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',twitter,'Twitter','base','Base prediction')

    #base prediction combined training
    #sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YTtwitter,'Youtube + Twitter','base','Base prediction')
    #sg=predictionFunction(SGDClassifier(),'SGDC',YTtwitter,'Youtube + Twitter','base','Base prediction')
```

```
#co=predictionFunction(ComplementNB(),'Comp. NB',YTtwitter,'Youtube + Twitter','base','Base prediction')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YTtwitter,'Youtube + Twitter','base','Base prediction')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YTtwitter,'Youtube + Twitter','base','Base prediction')

#prediction 2 Youtube training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YT,'Youtube','p2','Counting neutral comments as negative')
#sg=predictionFunction(SGDClassifier(),'SGDC',YT,'Youtube','p2','Counting neutral comments as negative')
#co=predictionFunction(ComplementNB(),'Comp. NB',YT,'Youtube','p2','Counting neutral comments as negative')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YT,'Youtube','p2','Counting neutral comments as negative')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YT,'Youtube','p2','Counting neutral comments
# as negative')

#prediction 2 Twitter training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',twitter,'Twitter','p2','Counting neutral comments as negative')
#sg=predictionFunction(SGDClassifier(),'SGDC',twitter,'Twitter','p2','Counting neutral comments as negative')
#co=predictionFunction(ComplementNB(),'Comp. NB',twitter,'Twitter','p2', 'Counting neutral comments as negative')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',twitter,'Twitter','p2', 'Counting neutral comments as negative')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',twitter, 'Twitter','p2','Counting neutral comments
# as negative')

#prediction 2 combined training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YTtwitter, 'Youtube + Twitter','p2','Counting neutral comments
# as negative')
#sg=predictionFunction(SGDClassifier(),'SGDC',YTtwitter,'Youtube + Twitter','p2', 'Counting neutral comments as negative')
#co=predictionFunction(ComplementNB(),'Comp. NB',YTtwitter,'Youtube + Twitter','p2','Counting neutral comments as negative')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YTtwitter,'Youtube + Twitter','p2','Counting neutral comments as negative')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YTtwitter, 'Youtube + Twitter','p2',
# 'Counting neutral comments as negative')

#prediction 3 Youtube training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YT,'Youtube','p3',
# 'Counting half of neutral comments as positive')
#sg=predictionFunction(SGDClassifier(),'SGDC',YT,'Youtube','p3',
# 'Counting half of neutral comments as positive')
#co=predictionFunction(ComplementNB(),'Comp. NB',YT,'Youtube','p3',
# 'Counting half of neutral comments as positive')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YT,'Youtube','p3',
# 'Counting half of neutral comments as positive')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YT,'Youtube','p3',
# 'Counting half of neutral comments as positive')

#prediction 3 Twitter training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',twitter,'Twitter','p3',
# 'Counting half of neutral comments as positive')
#sg=predictionFunction(SGDClassifier(),'SGDC',twitter,'Twitter','p3',
# 'Counting half of neutral comments as positive')
#co=predictionFunction(ComplementNB(),'Comp. NB',twitter,'Twitter','p3',
# 'Counting half of neutral comments as positive')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',twitter,'Twitter','p3',
# 'Counting half of neutral comments as positive')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',twitter,'Twitter','p3',
# 'Counting half of neutral comments as positive')

#prediction 3 combined training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YTtwitter,'Youtube + Twitter','p3','Counting half of
# neutral comments as positive')
#sg=predictionFunction(SGDClassifier(),'SGDC',YTtwitter,'Youtube + Twitter','p3','Counting half of neutral
# comments as positive')
#co=predictionFunction(ComplementNB(),'Comp. NB',YTtwitter,'Youtube + Twitter','p3',
# 'Counting half of neutral comments as positive')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YTtwitter,'Youtube + Twitter','p3',
# 'Counting half of neutral comments as positive')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YTtwitter,'Youtube + Twitter','p3',
# 'Counting half of neutral comments as positive')

#prediction 4 Youtube training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YT,'Youtube','p4',
# 'Counting all of neutral comments as positive')
#sg=predictionFunction(SGDClassifier(),'SGDC',YT,'Youtube','p4',
# 'Counting all of neutral comments as positive')
#co=predictionFunction(ComplementNB(),'Comp. NB',YT,'Youtube','p4',
# 'Counting all of neutral comments as positive')
#mu=predictionFunction(MultinomialNB(),'Mult. NB',YT,'Youtube','p4',
# 'Counting all of neutral comments as positive')
#lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.', YT,'Youtube','p4','Counting all of
# neutral comments as positive')

#prediction 4 Twitter training
#sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',twitter,'Twitter','p4', 'Counting all of neutral
# comments as positive')
```

```
136         #sg=predictionFunction(SGDClassifier(),'SGDC',twitter,'Twitter','p4', 'Counting all of neutral comments as positive')
137         #co=predictionFunction(ComplementNB(),'Comp. NB',twitter,'Twitter','p4', 'Counting all of neutral comments as positive')
138         #mu=predictionFunction(MultinomialNB(),'Mult. NB',twitter,'Twitter','p4', 'Counting all of neutral comments as positive')
139         #lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',twitter,'Twitter','p4','Counting all of
140         # neutral comments as positive')
141
142         #prediction 4 combined training
143         #sv=predictionFunction(svm.LinearSVC(max_iter=1000),'SVM',YTtwitter,'Youtube + Twitter','p4','Counting all of
144         # neutral comments as positive')
145         #sg=predictionFunction(SGDClassifier(),'SGDC',YTtwitter,'Youtube + Twitter','p4','Counting all of
146         # neutral comments as positive')
147         #co=predictionFunction(ComplementNB(),'Comp. NB',YTtwitter,'Youtube + Twitter','p4','Counting all of
148         # neutral comments as positive')
149         #mu=predictionFunction(MultinomialNB(),'Mult. NB',YTtwitter,'Youtube + Twitter', 'p4','Counting all of
150         # neutral comments as positive')
151         #lo=predictionFunction(LogisticRegression(max_iter=1000),'Log. regr.',YTtwitter, 'Youtube + Twitter','p4',
152         # 'Counting all of neutral comments as positive')
153
154     #predictionFunction runs the training, prediction and presents the results
155     def predictionFunction(model, modelname, dataAdrs, dataSetName, prediction, predictionName):
156
157         data = []
158         data_sentiment = []
159
160         with open(dataAdrs, encoding="utf8") as yt_f:
161             reader = csv.reader(yt_f, delimiter=';')
162             i = 0
163             for row in reader:
164                 i += 1
165                 data.append(row[0])  # text
166                 data_sentiment.append(row[1])  # sentiment
167
168         vectorizer = CountVectorizer(
169             analyzer='word',
170             lowercase=False,
171         )
172
173         features = vectorizer.fit_transform(data)
174
175         X_train, X_test, y_train, y_test = train_test_split(
176             features,
177             data_sentiment,
178             train_size=0.80,
179             random_state=1234)
180
181         model = model.fit(X=X_train, y=y_train)
182         y_pred = model.predict(X_test)
183
184         #predict training dataset
185         yhat = model.predict(X_test)
186         # evaluate accuracy
187         trainingDataAccuracy = accuracy_score(y_test, yhat)
188
189         videoSentiments = {}
190         commentLists = {}
191
192         with open(YoutubeTrendingDatasetCommentsFilepath,
193                   encoding="utf8") as GBUSCommentsFile:
194             reader = csv.reader(GBUSCommentsFile, delimiter=';')
195             for row in reader:
196                 if len(row[0]) > 11:
197                     continue
198                 if row[0] not in videoSentiments:
199                     videoSentiments[row[0]] = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
200                 if row[0] not in commentLists:
201                     commentLists[row[0]] = [row[1]]
202                 cl = commentLists[row[0]]
203                 cl.append(row[1])
204                 commentLists[row[0]] = cl
205
206         noComments = 0
207         totPcount = 0
208         totNeutCount = 0
209         totNegCount = 0
210         for vid in commentLists:
211             youtubeFeatures = vectorizer.transform(commentLists[vid])
212             predictions = model.predict(youtubeFeatures)
213             sentimentScores = videoSentiments[vid]
214
215             for comment in range(0, len(commentLists[vid])):
```

```
216                 if predictions[comment] == '4':
217                     sentimentScores[0] += 1.0
218                     totPcount += 1
219                 if predictions[comment] == '2':
220                     sentimentScores[1] += 1.0
221                     totNeutCount += 1
222                 elif predictions[comment] == '0':
223                     sentimentScores[2] += 1.0
224                     totNegCount += 1
225                 noComments +=1
226             videoSentiments[vid] = sentimentScores
227 
228         with open(YoutubeTrendingDatasetVideosFilepath,
229                   encoding="utf8") as GBUSVideosFile:
230             reader = csv.reader(GBUSVideosFile, delimiter=',')
231             for row in reader:
232                 if row[0] in videoSentiments:
233                     setVideo = videoSentiments[row[0]]
234                     setVideo[3] = float(row[2])  # likes
235                     setVideo[4] = float(row[3])  # dislikes
236 
237                     try:
238                         # actual like ratio
239                         setVideo[6] = setVideo[3] / (setVideo[3] + setVideo[4])
240                         # predicted like ratio
241                         if prediction == 'base':
242                             setVideo[5] = setVideo[0]/(setVideo[0]+setVideo[2])
243                         if prediction == 'p2':
244                             setVideo[5]=setVideo[0]/(setVideo[0]+setVideo[1]+setVideo[2])
245                         if prediction == 'p3':
246                             setVideo[5]=(setVideo[0]+0.5*setVideo[1])/(setVideo[0]+setVideo[1]+setVideo[2])
247                         if prediction == 'p4':
248                             setVideo[5]=(setVideo[0]+1*setVideo[1])/(setVideo[0]+setVideo[1]+setVideo[2])
249                         videoSentiments[row[0]] = setVideo
250                     except: # incase only neutral comments for base prediction
251                         del videoSentiments[row[0]]
252 
253         actualRatioSequential = []
254         predictSequential = []
255 
256         for key in videoSentiments:
257             actualRatioSequential.append(videoSentiments[key][6])
258             predictSequential.append(videoSentiments[key][5])
259 
260         mae = 0
261         differences = []
262         absdifferences = []
263 
264         for i in range(0, len(actualRatioSequential)):
265             mae += abs(predictSequential[i] - actualRatioSequential[i])
266             differences.append(predictSequential[i] - actualRatioSequential[i])
267             absdifferences.append(abs(predictSequential[i] - actualRatioSequential[i]))
268 
269         mae = mae / len(actualRatioSequential)
270         print(modelname, dataSetName, predictionName)
271         print("training dataset accuracy: ", trainingDataAccuracy)
272         print("number of samples in testing dataset: ", len(differences))
273         pearsonR, pValue = stats.pearsonr(predictSequential, actualRatioSequential)
274         print("Pearson correlation: ", pearsonR, "Two tailed p-value", pValue)
275         print("Mean absolute error: ", mae)
276         print("SD of likeratio differences:", numpy.std(differences, ddof=1))
277         print("average like ratio testing dataset: ", numpy.average(actualRatioSequential))
278         print("SD like ratio testing dataset",numpy.std(actualRatioSequential, ddof=1))
279         print("number of comments testing dataset: ", noComments)
280         print("Positive comments training dataset: ", totPcount)
281         print("Neutral comments testing dataset:", totNeutCount)
282         print("Negative comments testing dataset: ", totNegCount)
283         print("-----------------------------------------------------------------------------------------------------------")
284         return absdifferences # list of differences between predicted & actual like proportions
285 
286 main()
```

TRITA-EECS-EX-2021:411