Twitter User Gender Classification

Final project report

Akshay Karki (avk1063@rit.edu)
Sahana Ajit Murthy (sam2738@rit.edu)

ABSTRACT

28% of the world's population uses social media as a means of interaction, to let their feelings and opinions be known and also, as a way of getting to know new people. So, given the massive amounts of data generated, it is only common that researchers are spending enormous amount of time and energy to extract more and more information from what people say and do.

Out of all the social media platforms available, Twitter is widely popular among data scientists since the data is freely available with many user account attributes. But, among these attributes, the gender attribute is optional and many users tend to skip them. Hence, researchers are trying to combine existing features to infer the gender of a user based on their attributes like profile description, user name and full name. Our project is based on trying to infer the gender of a Twitter user given his/her other profile features.

1. INTRODUCTION

Social Network Analysis finds applications in areas such as Epidemic detection, Crime detection, Marketing, and many others. User Gender Identification in Social Networks is a vital task to curb the increased Cyber Crime caused by fake profiles, for marketing gender-specific products, and for analyzing difference of opinions between men and women in politics, social concerns, and world theories. Besides, profiles may belong to a business or a brand, forming the third category. So, we intend to analyze user profile information and identify the gender or if it is a business page.

We narrowed down our options of Social Networks to Twitter since its data is openly available for analysis. We are working on a Twitter dataset[10] obtained from Kaggle for User Gender Identificaton. The dataset contains various attributes about a Twitter user such as a username, profile description, tweet, etc. The dataset contains about 20000 rows each with a unique id and username. We aim to determine whether the profile belongs to a male, female or a business by analyzing the tweets and the user profile information. Further, we would like to determine if there exists certain words or phrases that are strong predictors of the gender.

The rest of the report is divided as per follows: Section 2 contains the background and related work. This section talks about some of the papers we have reviewed related to the topic of this project. Section 3 talks about the methodology implemented by us. This includes an in-depth explanation of what algorithms were used and how they were implemented with the features that we generated. Section 4 includes the results that we obtain after performing the experiments using different approaches. We talk about the challenges that we faced and the things we enjoyed about the project in section 5. We conclude our report with section 6.

2. BACKGROUND AND RELATED WORK

To tackle the problem of fake profiles on Twitter, a method of using writeprint identification is suggested in [7]. It is the process of extracting linguistic features from anonymous text to identify the author. This is possible each person has his/her way of writing which is different from others. They use a data crawler to collect twitter messages from 30 different accounts. The crawler used was Python Twitter Tools, a software which takes in an account name and produces the list of tweets for that particular user.

Just like we would need to perform data cleaning, the authors in [7] perform data cleaning and also POS (part of speech) tagging using the Stanford POS tagger. With the help of this tagger, they are able to select an appropriate set of features, since, it significantly affects the learning process and the outputs we get.

There are two important stages after the set of features has been extracted from each tweet as a feature vector. The first stage is the learning stage where the classification model is fed the feature vectors to generate a classification model. Initially about 200 tweets, with user name labels, are used as training data for the classification model. The next stage is the testing/prediction stage where the classification model, built using the feature vectors of given tweets, is fed with previously unseen data. The output at this stage is the pre-

.

dicted author of the input tweet.

A method similar to what we propose to implement has been done by Fernandez et al.,[6]. The authors perform different combinations of characteristics, such as user name, profile picture, etc., to analyze the performance of their system. The dataset only consists of users belonging to a particular city. From this complete dataset, only 10% has been chosen for analysis. The training and testing split is 70-30 of the selected dataset.The labeling process is done manually for each and every account. The accounts which were unknown, were ignored from consideration.

The method not only analyzes the name and profile description, but also the profile picture of each account. The name of every account was extracted through the Twitter API and depending on popular statistics, it was classified as belonging to either male or female. The authors then create a count of male and female descriptors based on gender roles to segregate men and women. Then, using a popular face detection API, the authors analyze the profile picture associated with the account.

By generating a set of features, each time a method was implemented, the authors are able to mix and match the features obtained to test several different classification models. Different classifiers such as Decision Tree and Support Vector Machines are used to observe the effect of different set of features on the accuracy of the model and also estimate which set of words or images could be used to segregate the men from women. In each classifier, the model proposed outperforms the other similar methodologies indicating a reasonably sound method.

Adopting a similar approach done by Fernandez et al.,[6], Burger et al., [2] also use a variety of features to identify the gender of the user. The dataset chosen has about 22 tweets per user, but it also has a high variance. The dataset is split up into 3 sets, namely training, development and testing and the authors ensure that each tweet is in only one of the three subsets. The accounts linked to other social media networks are carefully studied since they contain a lot of the target attributes desired by the authors including demographic information. During an initial iteration of the dataset, the authors observe that the data has almost equal binary split among males and females.

To handle dynamic nature of screen names and profile descriptions, which may keep changing repeatedly, the authors come up with an innovative way of storing all these values as a set. They also set an empty set for users who leave their descriptions or full names to be blank. This is a really nice way of handling incomplete data for this scenario and one which we may look to implement in our model since we are sure to come across this scenario when we work on our dataset. Also, to reduce time and space complexity, the authors perform one time feature generation to convert a feature into an integer codeword.

For the evaluation of the model that they have generated, the authors first train the data using the training set. They then evaluate the data using the development set. Only after satisfactory performance by the best models, the testing

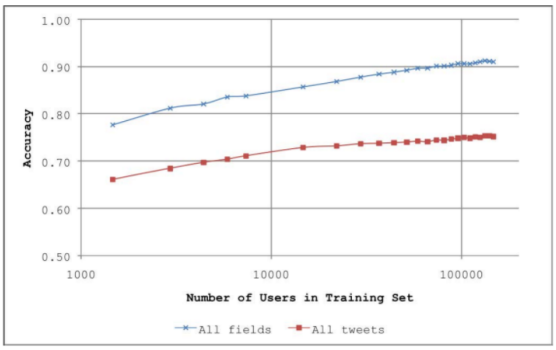

Figure 8: Performance increases when training with more users

Figure 1: Accuracy of model as compared to number of users [2]

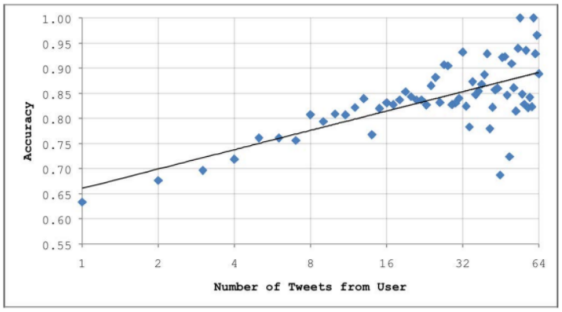

Figure 9: Performance increases with more tweets from target user

Figure 2: Accuracy of model as compared to number of tweets from a user [2]

data is exposed to the models to see how well they perform. They also compare the classifier's efficiency to a human and for this purpose they use Amazon Mechanical Turk to allow a large number of anonymous workers to complete simple on-line tasks.

An interesting observation is that as the number of tweets increases so does the accuracy of the model for both, all fields (which includes, the full name, description and the tweet) and just tweet only classifier. In almost all test cases, the system outperformed the human counterparts.

To identify the author's gender based on text found on the Internet, Cheng et al. [3] propose various features that act as gender indicators, design a set of measures to predict the author 's gender based on the short text messages and finally, design classifiers and optimize their parameters.

The problem considered in this paper [3] is that of a binary classification where in given a short anonymous piece of text e, assign it to one of the two classes: {male, female}. This problem is slightly different from our problem in that we also have a third class indicating if the twitter account belongs to a business or a brand based on various attributes.

The gender identification process [3] is broken down into 4 important steps:

1. Forming a dataset by collecting text messages

2. Identifying specific features that act as gender indicators.

3. Automatic extraction of the feature values from the text messages.

4. Developing a classification model to identify the gender of the author.

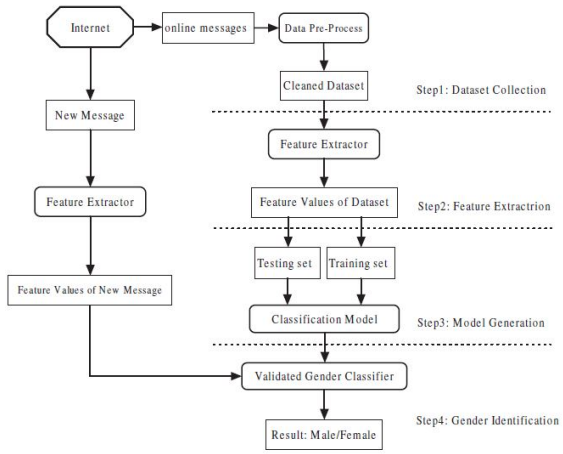

Figure 3: Gender Identification process [3]

Cheng et al. [3] have considered two datasets with each having different nature of texts: One contains messages from the newsgroup which has a neutral descriptive language whereas the other is based on personal emails which display the true nature of the author.

Reuters newgroup dataset, obtained from Reuters online databases, was pre-processed as follows:

1. After retrieving the authors' information, documents were categorized based on the gender.

2. Unnecessary information such as date and time were removed from the texts.

3. XML formatting was removed.

4. Retained only those messages with word count between 200 and 1000.

5. Removed the messages with too many quotation marks, setting the threshold of quote count to 0.002.

Enron email dataset contained 517,431 emails from about 150 users which includes business communication, personal messages, and technical information. The pre-processing steps followed to reduce the dataset to 8970 emails with 108 authors is as follows:

1. The emails were extracted from the sent items of each sender and they were categorized based on the gender.

2. The emails were cleaned by removing the header, signature and reply messages.

3. Duplicated or Carbon copied emails were removed.

4. Only emails with word count between 50 and 1000 were retained.

545 features were designed [3] which they classify into the below five categories:

- Character-based
- Word-based
- Syntactic
- Structure-based
- Function words

The feature extractor generated a 545-Dimension vector to depict values of the 545 features. This feature extraction was implemented in Python.

The paper [3] applies three classifier algorithms to solve this problem: Bayesian based logistic regression, AdaBoost Decision tree and Support Vector Machine (SVM). They implemented these classifiers in MATLAB and performed training and evaluation using 10-fold cross validation method. The results [3] showed that SVM outperformed the other two models with an accuracy of 76.75% and 82.23% for the two datasets respectively. These results also implied that it is comparatively difficult to distinguish genders when the tone is neutral such as in the news dataset.

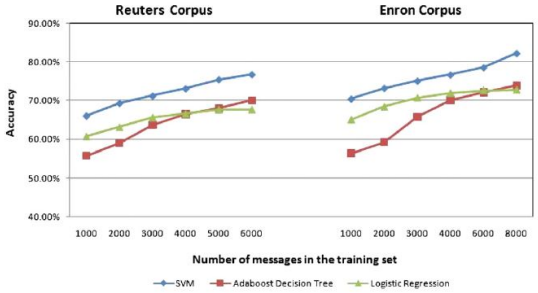

Figure 4: Accuracy comparison of the three classifiers used [3]

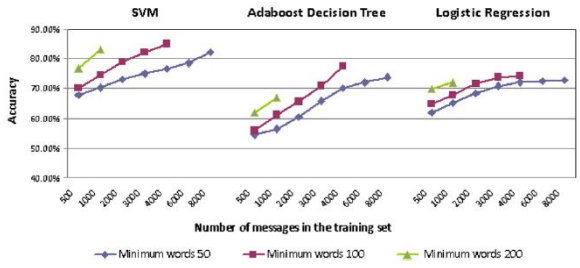

Figure 5: Classifier Accuracy as a function of number of words in the email and number of messages [3]

It is concluded by Cheng et al. [3] conclude that men and women differ in their writing and classification performance was the best for SVM and the accuracy increases with increase in the number of training samples and the number of words in each sample.

Related work by Vicente et al. [11] utilize the unstructured information provided in the user's profile to identify the gender of Twitter users. The profile information is extracted and represented as a set of features which are then used in both supervised and unsupervised learning algorithms to identify the gender of the user. The process carried out in this paper [11] to perform the gender classification task is explained below.

Dictionary Formation:
A dictionary with around 8444 names, that were found on the list of most used baby names in the US Social Security Administration Official Website, were compiled. The dictionary considered different spellings of similar names to be distinct. Also, it only focused on compiling names that are not unisex, so it is easy to distinguish between male and female names. The dictionary consisted of the following information: Name, Gender and Number of occurrences.

Data Set and Feature Extraction:
The dataset was obtained by extracting tweets posted in the month of December 2014 using the Twitter API. The authors [11] make use of only the "Screen name" and "User name" attributes from the user profiles for this task. They use an algorithm for extracting the screen and user names of users. Their model makes use of 192 features. It works as follows:

1. Read user name and screen name
2. Check for those names in the dictionary
3. If names are not found:
    - Remove repeated vowels in the names
    - Check for the modified names in the dictionary
    - If still not found, Replace the "leet speak" characters with original ones.
    - Find the modified names in the dictionary
4. Add all the found names to a list
5. For each of the found names, find gender-related features, set a threshold for each feature, and add them to another list. Return this list of found features.

The dataset was labeled by verifying the user names or screen names from the dictionary and then confirming based on the user's profile picture. If names were not found in the dictionary or if the user had no profile picture or had a celebrity in its place, such users were discarded. Also, the users who tweeted using the Twitter API, instead of from web or mobile, were considered to be bots and their profiles were discarded too. The chosen profiles were validated with information in the users' blogs, if it existed.

Gender Classification: This paper [11] experimented with a couple of supervised learning methods such as Multinomial Naive Bayes, Logistic Regression and Support Vector Machines and unsupervised learning algorithms such as Fuzzy c-means and K-Means clustering. Except for the Fuzzy c-Means method, which was applied using the Fuzzy logic toolkit in Python, all the learning algorithms were applied using Weka.

In Supervised learning, the model was trained and evaluated using 5-fold cross-validation and the results are captured in Table 1. In unsupervised learning, both the clustering techniques used Euclidean distance and set the number of clusters to 2. The results are as captured in Table 2.

	Lazy features		Greedy features	
	Accuracy	kappa	Accuracy	kappa
Logistic Regression	94.5%	0.89	93.7 %	0.87
Multinomial Naive Bayes	**97.2%**	**0.94**	**97.2%**	**0.94**
Support Vector Machines	96.0%	0.92	96.4%	0.93

Table 1: Classification accuracy for supervised methods [11]

	Lazy features		Greedy features	
	labelled	all data	labelled	all data
*k*Means clustering	74.9%	71.2%	75.1%	67.3%
Fuzzy c-Means	84.9%	87.3%	93.1%	**96.0%**

Table 2: Classification Accuracy for unsupervised methods [11]

Alowibdi et al. [1] focus on exploring different profile features of a twitter user such as first name, user name and profile colors to identify a user's gender rather than using tweets. An empirical study was performed to determine the strengths and weaknesses of these features in identifying a user's gender. They also designed an approach for feature reduction and gender prediction based on Phonemes.

The process followed by the authors [1] to carry out this task is as below:

1. From a vast number of twitter profiles, user profile information such as first names, user names, background colors, text colors, link colors, sidebar fill and border colors, were collected to form a dataset. The number of unique colors were reduced from 17 million to 512 by applying color quantization and sorting techniques.
2. Labeled the original genders for all profiles by finding information from their other Social networking sites.
3. Characters in different languages were converted to English characters using the Google Input tool.
4. Conversion of first names and user names to Phoneme sequences. Their phoneme set consists of 40 that belongs to one of the three lexical stresses, namely, No stress, primary stress, and secondary stress. Name to Phoneme conversion was done using LOGIOS lexicon tool.
5. Training and testing of classification models for gender identification is done using KNIME.

The dataset consisted of 194,293 Twitter profiles of which 104,535 profiles belonged to males and 89,758 profiles belonged to females. Data was sampled to have equal proportion of male and female profiles. The names were then preprocessed to remove white spaces, last names, punctuations and stop words. Both first names and user names are then converted to Phoneme sequences which are in turn

converted to n-grams. The n-grams and the profile colors act as inputs to the classifiers.

Experiments were performed for two kinds of input feature-sets: Phoneme-based feature set where phonemes were converted into n-grams and Word-frequency based feature set, where names were directly converted to n-grams. For both the cases, three different classifier algorithms, Naive Bayes, Decision Tree and Naive Bayes Decision Tree, were trained and evaluated. They [1] conclude that the first names are more effective as compared to profile names and colors in identifying a Twitter user's gender. The results obtained were as below:

	1-gram	2-gram	3-gram	4-gram	5-gram
Without phonemes (n-gram applied to characters of names)					
NB	NA	65.3	67.0	69.2	75.1
DT	NA	68.2	69.3	72.0	76.3
NB-Tree	NA	**69.3**	**70.7**	**74.0**	**78.3**
With phonemes (n-gram applied to set of phonemes)					
NB	65.2	65.3	66.0	NA	NA
DT	**78.5**	**79.2**	**82.5**	NA	NA

Table 3: Gender classification accuracy for profile names [1]

	1-gram	2-gram	3-gram	4-gram	5-gram
Without phonemes (n-gram applied to characters of names)					
NB	NA	55.3	56.0	57.2	58.0
DT	NA	**55.7**	**56.9**	**58.2**	**59.6**
NB-Tree	NA	53.2	54.0	56.0	58.0
With phonemes (n-gram applied to set of phonemes)					
NB	55.2	56.0	55.0	NA	NA
DT	**68.5**	**70.2**	**75.2**	NA	NA

Table 4: Gender Classification accuracy for user names [1]

	1 color	2 colors	3 colors	4 colors	5 colors
Without Applying Color Quantization and Sorting					
NB	58.0	59.3	61.1	61.1	61.2
DT	**58.9**	**61.2**	63.3	63.1	63.3
NB-Tree	58.0	60.3	**64.7**	**66.2**	**65.7**
With Applying Color Quantization and Sorting					
NB	60.2	61.0	62.0	62.0	63.0
DT	59.0	62.8	65.3	65.0	64.7
NB-Tree	**70.3**	**71.0**	**73.2**	**73.7**	**74.0**

Table 5: Gender Classification Accuracy for Profile colors [1]

Miller et al.[8] have studied the Twitter user gender identification using 1 through 5-gram features extracted from the tweet texts. They employ Naive Bayes and Perceptron classification algorithms which are innately stream-oriented, since they perform only a single pass on the data and then store the representations of the models compactly. They consider stream algorithms as opposed to batch-training methods to handle the huge amounts of twitter traffic.

Their results show that Naive Bayes has a better performance than Perceptron for most of the metrics considered like Accuracy, Recall, Precision, F-Measure and Balanced Accuracy. This implies that Probabilistic Modeling is more suited for our gender identification task. Besides, both algorithms performed better when a higher number of features represented tweets, but that increases the computational complexity.

An interesting method put forth by Ciot et al. [4] is the assessment of hidden attribute inferences on Twitter data for non-English based content. They primarily focus on languages with difficult orthography such as Japanese and French. The method, which the authors claim to be the first approach in trying to infer latent attributes, makes use of Amazon Mechanical Turk (AMT), similar to the work of Burger et al. [2], for labeling users with their genders and uses Support Vector Machines (SVM) for classification purposes.

The authors focus specially on the French language since unlike English, it has a lot of syntax based mechanisms for identifying the gender of the user. Despite a poor performance by the SVM, the combined classifier manages to have an 8% increase in the accuracy over only SVM classification. The model, however, fails to improve on the existing accuracy of other models while trying to classify the Japanese language using the SVM.

Similar to the work done by Miller et al.[8], Deitrick et al.[5] studied the use of a stream-based classification algorithm to identify genders of Twitter users based on their tweets. They use Modified Balanced Winnow, a neural network, to perform this task. It belongs to the class of Neural networks known as Mistake Driven Online Learners where in learning models get updated only when classification mistakes are encountered.

Modified Balanced Winnow was experimented by first using all the 9170 features and then with the selected 53 features. It was found that the accuracy increased by a great extent with fewer number of features. While the former case obtained an accuracy of 82%, the latter achieved an accuracy of 98%. They conclude that Modified Balanced Winnow is an effective model for determining a Twitter user's gender and using only the most significant features helps improve the accuracy.

Using statistical name characteristics to infer the gender of the user has been proposed and implemented by Mueller et al. [9]. The authors have implemented a new classifier named NamChar, which assigns the gender to the user based on the first name, which has been extracted using various characteristics. The dataset used is in the figure below. The classifier NamChar used by the authors takes into consideration a number of features to determine the gender of the user. They only consider the two genders, male and female,

to split the data into. An interesting conclusion they come to is that having a larger database of names to compare whether the given user is male or female, has no effect on the accuracy of the classifier.

Gender	Census	nam_dict
Male first name	888	18,204
Mostly male name	–	915
Male name if first part	–	7
Female first name	3,944	17,328
Mostly female name	–	722
Female name if first part	–	8
Unisex first name	331	8,329
Total number of names	5,163	45,513

Figure 6: Dictionaries of names used

3. METHODOLOGY USED

3.1 Data set used

The data set for our project was obtained from Kaggle[10]. The dataset contains about 20,050 records, each with a set of attributes like user profile description, sidebar color, tweet etc. Each record represents a user and all his/her profile attributes on Twitter.

After the data has been cleaned, we split the data into training and testing data set with 66% of data being split as training data and the rest as testing data. We use the train_test_split function of the model_selection module of sklearn library, which takes in two sets as parameters. The first parameter is the set of all selected input attributes from the entire data set and the second is the associated class values. The function returns 4 sets of data. The first set is the list of selected input attributes in the training data, the second set consists of its corresponding class values. Similarly, the third set consists of the selected input attributes in the testing data while the fourth set has its corresponding target values.

3.2 Approach used

3.2.1 Data cleaning

Like any data mining process, our first step involved cleaning the data. We first get rid of any observation which has null values for all its attributes. We do not find any such observations, since all rows have some non-null values. We, then drop row values in which our taregt variable gender of a user is missing or empty. This is done since the gender attribute is the one we will be comparing our predicted output with, so, we will need all values to be not null. The size of the dataset reduces to 19953 records, indicating 98 records had missing or null values.

Furthermore, we also eliminate records in which the value of the gender attribute is unknown. Doing this reduces the size of the data set to 18836, signifying 1117 records had 'unknown' value in the gender column. This is done since we only want to classify our data set into either male, female or brand.

Our next step required us to deal with cleaning the text in profile description and tweets column. This involved converting all text to lower case, stripping all punctuation strings before and after every word, removing HTML code, white spaces, special characters and numbers found in between the text. Once we are done with this, we are left with clean data, so to speak, which has only 3 gender values and can be processed. We then check the gender confidence column present in our data set. This confidence is represented as a value between 0 to 1, with 1 indicating full confidence of the judge in specifying the gender of that particular user. We get rid of records in which the gender confidence is not 1. This is done because we only want to deal with records which have absolutely correctly classified gender, since we will be evaluating our model based on this.

3.2.2 Feature Selection and Extraction

We first split the data set into training and testing data based on only one attribute. First, we use only tweet to build a classification model and then only user profile description is used for classification. We then combine two attributes, tweets and profile description, to create a new feature. The data containing only this new feature is used to split our data set into training and testing data set and also used in the classification of gender.

Next, we select 5 important features to be used during the classification process. These are tweet, profile description, sidebar color, link color and profile names. These features are chosen among the many others present, since we feel that these are the most indicative of a user's gender. We also experiment by removing one of the 5 features and evaluate the performances of classifier models.

Once the attributes to be used to train the classifier models are chosen, we had to also extract the features from text strings for attributes such as tweets and profile descriptions so as to feed them into the classifiers. For this, we used countVectorizer from feature extraction package in sklearn and used it to convert input text strings into word tokens and then transformed them into the word token count matrices. When we used 4 or 5 features, we had to stack and combine countVectorizer returned matrices for each of those features and then feed them into classifiers. Besides, we also used LabelEncoder from sklearn.preprocessing package to encode the target class labels.

3.3 Algorithms used

We build 7 classification models for classifying the given data set and check their accuracies. The models are:

1. Naive Bayes Classification
2. Stochastic Gradient Descent Classifier
3. Multilayer Perceptron Classifier
4. Random Forest Classifier
5. Decision Tree Classification
6. K-Nearest Neighbors Classifier

7. Support Vector Machines

All the above given models are implemented using the scikit-learn library in Python. For each of the models, we first train the respective model using our training data set generated earlier. Then, the accuracy of the model is computed using the testing data set. We also generate a confusion matrix to keep track of the number of correctly and incorrectly classified records. The results for each of these models are reported in the section 4.

3.3.1 Naive Bayes Classification

We use MultinominalNB function from the naive_bayes module of sklearn library for Naive Bayes Classification. It uses discrete features for classification. The parameters it takes in are alpha, used for parameter smoothing, fit_prior, which checks whether to use prior probabilities, and class_prior, which checks whether to use prior probabilities for class values. We set all of them to their default values.

The data that is inputted is training data set for all attributes and a data set containing just the output values. We use predict method to classify the test data set and use these predictions to compute the confusion matrix and plot it. Using the score method, we can get the accuracy of the model. We need to supply the testing data set for this as well.

3.3.2 Stochastic Gradient Descent Classifier

We use the SGDClassifier function from the linear_model module of sklearn library. Here the model is updated each time the function looks at the estimated gradient loss of a new sample. There are lots of parameters which can be passed to this function, but we keep all values to their default ones. We pass the training data set and the class value list through the model to train it. Then the accuracy of the model is computed using the testing data set. The confusion matrix is plotted using the module confusion_matrix from sklearn library.

3.3.3 Multilayer Perceptron Classifier

We use the MLPClassifier function from the neural_network module of sklearn library. The function uses the stochastic gradient descent. It has a lot of parameters to tune, but we decided to keep all parameters to their default values. We only pass the training data set and the class value list as parameters. Then, we perform prediction by passing the testing data as a parameter to the predict method. The value we get here is used to generate the confusion matrix. The accuracy of the model is computed using the testing data set and list of actual class values.

3.3.4 Random Forest Classifier

We use the RandomForestClassifier function from the ensemble module of sklearn library. Even though a lot of parameters are available to tune, we keep everything to default. We first train the model by passing the training data set and the class value list as parameters. We predict how the model will perform on testing data by passing the testing data as parameter to the predict method. This score helps us in building the confusion matrix. Then the accuracy of the model is calculated using the actual values of target variable in the testing data set.

3.3.5 Decision Tree Classification

For the Decision Tree Classification, we use the DecisionTreeClassifier function available in the tree module of the sklearn library. It uses the same algorithm as in the decision tree classifier. Training data set and the list of target variables are passed as parameters to train the model. The predict method is used to predict how the model performs on unseen data, which we have stored as testing data and also to build the confusion matrix. So, we pass the testing data set built earlier as parameter to the predict method. The accuracy score of the model is computed using the score method. We pass the testing data set and the list of target values as parameters here.

3.3.6 K-Nearest Neighbors Classifier

We use the KNeighborsClassifier function from the neighbors module of the sklearn library. The kNN classifier algorithm is implemented inside the function. We keep the parameters available in the function to the default values. We pass the training data set and the class value list as parameters to train the model. Then the accuracy of the model is computed using the testing data set and the list of actual target values. The confusion matrix is plotted using the module confusion_matrix from sklearn. We get the values for the confusion matrix by using the predict method which takes as argument the testing data set.

3.3.7 Support Vector Machines

We use the SVC method from svm module of the sklearn library. The model training is done using the 'fit'method which takes in as parameters the training data set and list of target values. The predict method is used to predict the accuracy of model on unseen data. It takes the testing data set as a parameter. The score we obtain from the predict method is used to build the confusion matrix. The accuracy of the model is computed using the testing data set and the list of actual target values.

4. EXPERIMENTAL RESULTS

4.1 Accuracy of each model

4.1.1 Using 5 features

The 5 features we use are, tweets, profile description, sidebar color, link color and name. Using these 5 features, we see that the Naive Bayes classifier performs the best classification with an accuracy of 66.48%. Multilayer Perceptron Classifier, takes a lot of time during computation, but still comes up second best at 65.55%. The worst performance is by Support Vector Machines which only manages to accurately guess the gender 38.94% of the time. This is slightly better than randomly guessing the gender which results in an accuracy of 33.34%.

	Naïve Bayes Classification	Stochastic Gradient Descent	Multilayer Perceptron	Random Forest Classifier	Decision Tree Classification	K-Nearest Neighbors Classifier	Support Vector Machines
Using 5 features	66.48%	63.59%	65.55%	54.26%	55.77%	46.03%	38.94%

Table 6: Accuracy of all models using 5 features

4.1.2 *Using 4 features*

The 4 features used are tweets, profile description, sidebar color and link color. The results are quite similar to what we obtain in 4.1.1. Naive Bayes Classification manages to beat all other classifiers with an accuracy of 65.23%. Multilayer Perceptron Classifier comes second best again with 63.01%, while Support Vector Machines perform the worst with an accuracy of 38.94%. We can infer from this result that the SVM performs worst among all the 7 classifiers since the data is not linearly separable.

	Naïve Bayes Classification	Stochastic Gradient Descent	Multilayer Perceptron	Random Forest Classifier	Decision Tree Classification	K-Nearest Neighbors Classifier	Support Vector Machines
Using 4 features	65.23%	62.31%	63.01%	52.44%	54.26%	44.84%	38.94%

Table 7: Accuracy of all models using 4 features

4.1.3 *Combining tweets and profile description*

We also run our tests on the 7 classifiers using a feature combining just tweets and profile descriptions. We find that even though the performance of all classification models decreases, Naive Bayes is still the best performing model coming in with an accuracy of 63.89%. The worst performing model is Support Vector Machines with an accuracy of 38.38%. From these results, we can make out that tweets and profile description are the most indicative attributes for predicting the gender of any user.

	Naïve Bayes Classification	Stochastic Gradient Descent	Multilayer Perceptron	Random Forest Classifier	Decision Tree Classification	K-Nearest Neighbors Classifier	Support Vector Machines
Combining tweets and profile description	63.89%	60.92%	60.48%	51.25%	51.55%	42.82%	38.38%

Table 8: Accuracy of all models using tweets and profile descriptions

4.1.4 *Only tweet text*

We then run our tests on the 7 classifiers on just one attribute, tweets. The performance decreases as compared to results in 4.1.3. Naive Bayes, as expected, comes up with the best performance at 55.11% while Support Vector Machines have an accuracy of 38.90% and come up last among all the models. An interesting point to notice here is that there is a slight increase in the performance of the SVM model as compared to 4.1.3. But the improvement is insignificant since it is still the poorest performing model among the ones we have chosen.

4.1.5 *Only profile description*

Just as we perform tests in 4.1.4 using tweet texts, we also perform them for only profile descriptions. We find

	Naïve Bayes Classification	Stochastic Gradient Descent	Multilayer Perceptron	Random Forest Classifier	Decision Tree Classification	K-Nearest Neighbors Classifier	Support Vector Machines
Only tweets	55.11%	52.44%	52.70%	47.37%	47.93%	43.90%	38.90%

Table 9: Accuracy of all models using only tweet texts

that the accuracy of Naive Bayes Classification is highest with 58.26%. The worst performing model is Support Vector Machines with an accuracy of 39.02%. We can also note that this accuracy obtained by the SVM model is the best it has obtained in all our tests. This signifies that using only profile descriptions to classify user gender would result in an improved performance if we used the SVM model. Besides, we can observe that profile description plays a better role at determining gender than a user's tweet since all the models have an improved accuracy as compared to the results obtained in 4.1.4

	Naïve Bayes Classification	Stochastic Gradient Descent	Multilayer Perceptron	Random Forest Classifier	Decision Tree Classification	K-Nearest Neighbors Classifier	Support Vector Machines
Only profile descriptions	58.26%	56.22%	54.15%	49.61%	51.91%	44.43%	39.02%

Table 10: Accuracy of all models using only profile descriptions

4.2 Confusion matrix evaluation

The confusion matrix is built for all the models using the predictions obtained from the predict method in scikit-learn and comparing it against the true class labels as mentioned in section 3.3. The confusion-matrix method from the metrics module of scikit-learn library is used to compute it. We use the imshow function from the pyplot module of Matplotlib library and pass the confusion matrix that we got earlier as its parameter. We plot the confusion matrix for all approaches which include using 5 features, using 4 features, combining tweets and profile descriptions, using only tweets and using only profile descriptions. Each of the approach generates a different confusion matrix.

For the 5 features approach, we find that the accuracy of the Naive Bayes classification is highest. This is also reflected in its confusion matrix, since it manages to correctly predict 3121 records out of the 4694 records present in the testing data. Individually looking at each gender, we find that it still has the highest accuracy among all the models. Unsurprisingly, we find that the lowest correctly predicted records belong to the SVM model.

Using the 4 features approach, we find that the Naive Bayes classifier still has the best count of correctly classified genders of users among all 7 models. This is represented by its confusion matrix. Looking at individual genders, for male and brands, the Multilayer Perceptron classifier has the second highest count. For the female gender, Decision Tree classifier has the next best accuracy coming in after the Naive Bayes classifier. But since it performs on male and brand gender, the overall performance takes a hit.

We obtain similar results for our other approaches. The Naive Bayes classifier manages to outperform other models

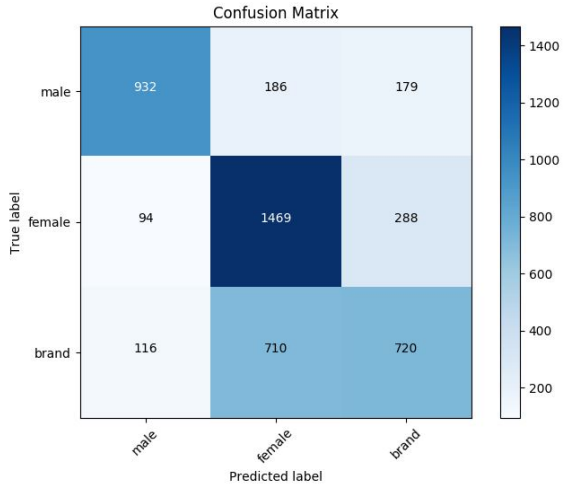

Figure 7: Confusion matrix for the Naive Bayes classifier using 5 features

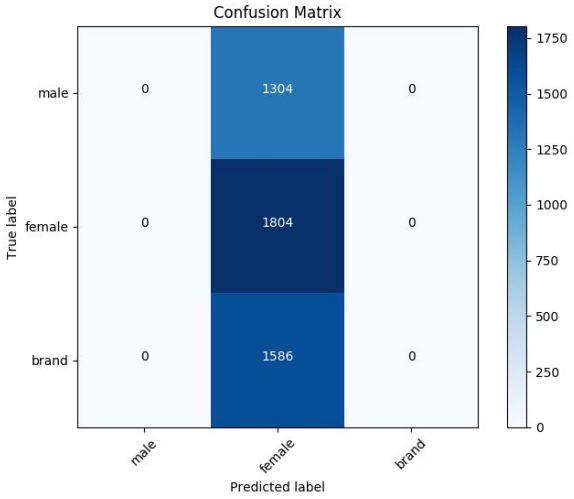

Figure 8: Confusion matrix for the SVM model using 5 features

even though its accuracy drops as compared to the previous two approaches. SVM model is the worst performing model, but that may be attributed to the linearly inseparable data. A surprising thing to note is that SVM model has its best performance when we use the only profile description approach. The count of correctly classified records increases as compared to the only tweets approach.

4.3 Frequency of words associated with each gender

Another goal of our project was to determine if certain words in English are strong predictors of a gender or a brand. To determine this, we first use the stopwords corpus from the NLTK library which contains a list of stopwords, such as for, the, of, etc. These words are then removed from the tweet of every user. The tweets are those which have been through the data cleaning process mentioned in section 3.2.1. This gives us words which might be important. Processing this

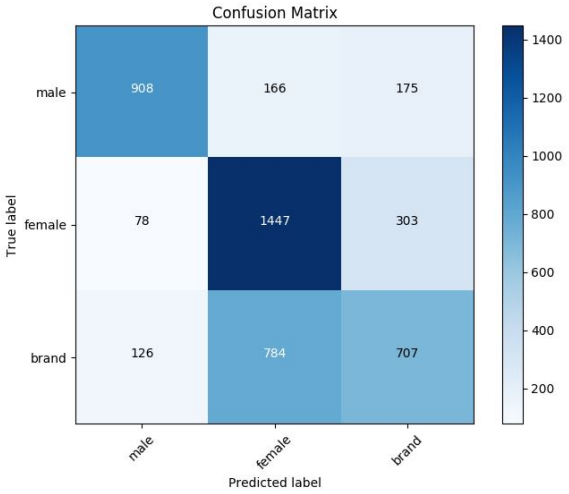

Figure 9: Confusion matrix for Naive Bayes classifier using 4 features

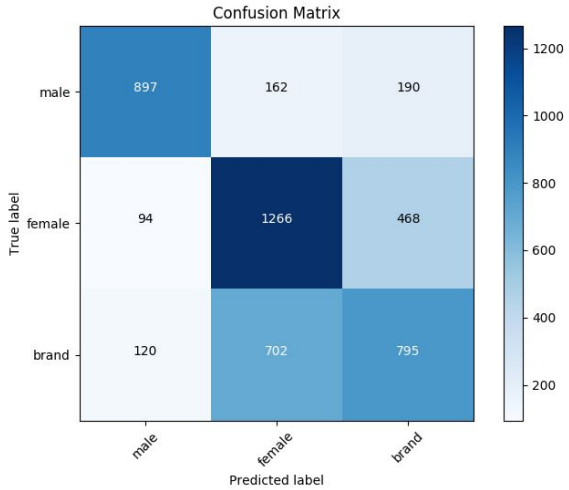

Figure 10: Confusion matrix for Multilayer Perceptron classifier using 4 features

set of words gives us which words are commonly associated with which gender or what type of words are commonly used by each gender and brands while tweeting.

We retrieve the list of 20 most frequent words for each gender and also brands. We then generate a bar graph projection showing the frequency of each word for a particular gender and brands. We take the 20 most frequently occurring words and plot it for all three classes of profiles using the plot function of the pyplot module in Matplotlib library. The parameters passed are kind of plot, which in our case is bar graphs, and the alignment, which is horizontal in our case.
It can be noticed that 'weather' is the most mentioned word by brands and men and women have many words in common and are mostly different from the words used by brands.

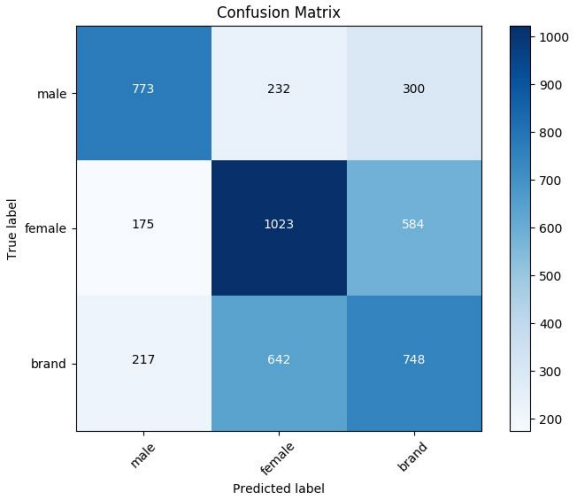

Figure 11: Confusion matrix for Decision Tree classifier using 4 features

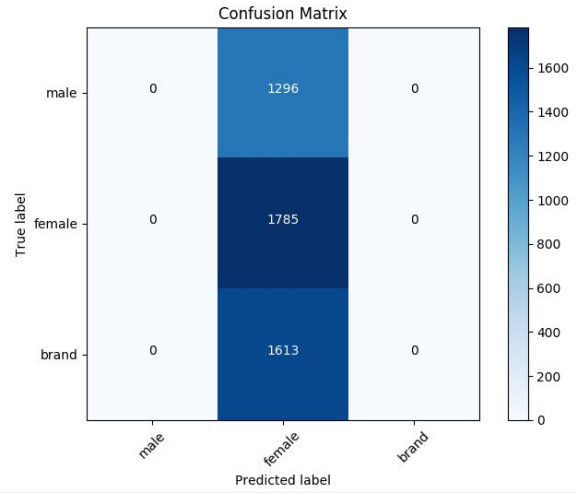

Figure 12: Confusion matrix for SVM model using only tweets

5. DISCUSSION

This was an interesting project to work on for many reasons. First of all, it is quite impressive to know how differently men, women and brands upload posts on Social Media. Also, how determining a user's gender on Social Media can help companies advertise gender-based products to the relevant users or how it can help to keep a check on Cyber crimes by recognizing fake gender profiles.

Second, working on this project helped us learn how to build classifier models when there are attributes with long text strings, as it involved more of text related tasks such as cleaning the text, processing it by tokenizing the text into word vectors and transforming it into a matrix representation for classifiers to easily understand the text and train on it. Furthermore, we were able to apply most of the concepts learned in class to this project and evaluate the performances of multiple classifier models on this data and find the best

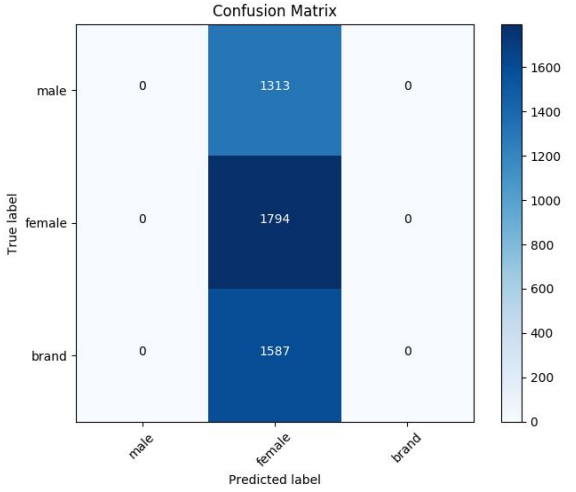

Figure 13: Confusion matrix for SVM model using only profile descriptions

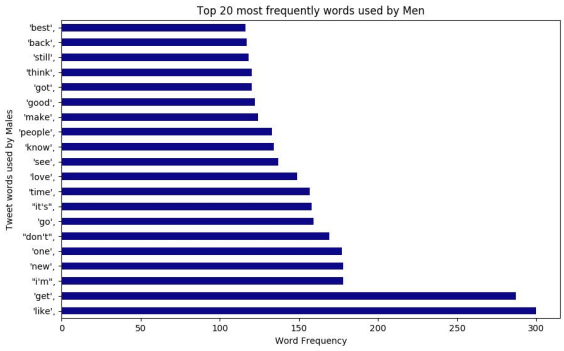

Figure 14: Frequency of words used by Males

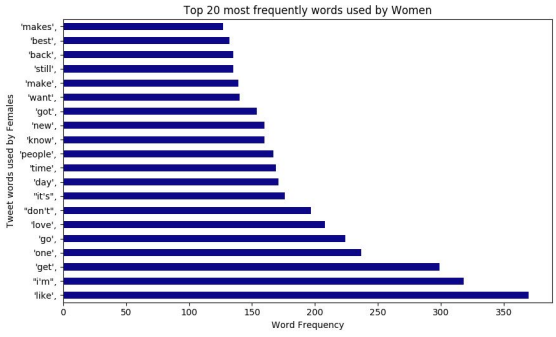

Figure 15: Frequency of words used by Females

one. This definitely enhanced our knowledge in the area of text processing and about various classifier models.

Few of the challenges that we encountered while working on this project include:

- Initially determining how to process the text to feed them into classifier models or which packages provide functionalities to process text and which one to choose.

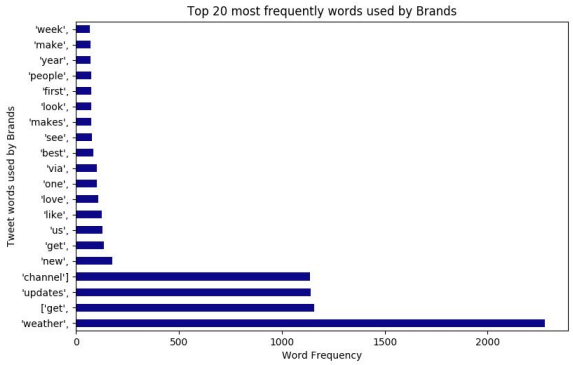

Figure 16: Frequency of words used by Brands

- To learn how much of text cleaning was required.
- To determine which features help determine the gender best and which features can be combined.
- To figure out how to transform multiple features into word token count matrices using countVectorizer.
- Which classifiers to train our data on.

However, we made our way through these obstacles and learned a great deal from it. We would like to give credits to sklearn's online documentation which was of good help while coding. Furthermore, the accuracy we obtained is data dependent. If the data contained some more relevant features, we could have achieved better results. This paves the way for future work discussed in the next section.

6. CONCLUSION

The project aimed to classify gender of a Twitter user based on various attributes such as profile description, user name and many more. We first clean the data using string operations and regular expressions. Once the data is cleaned, we split the data into training and testing data sets. Next, features are generated by combining existing attributes. Our approach is based on the number of features used during the classification process.

We use 7 classifier models during our classification stage. All the 7 models make use of the Machine learning methods that scikit-learn library provides. For each of the 7 models, we use different approaches such as using 5 features, or using just profile description etc., This results in varying outcomes but we find that Naive Bayes classifier performs the best classification while Support Vector Machine performs the worst. We also build confusion matrices for each approach for each model.

Wanting to further improve our work in this project, we determine most frequently used words in tweets by each gender and also brands which are projected onto bar graphs. The bar graphs are built using the Matplotlib library. This helps us associate which gender uses what types of words while tweeting.

Future scope in this project might include using user names as a feature along with these existing features to classify user gender. Implementing our own algorithms might lead to an improved performance in the results. Another interesting direction for future work would be to include more number of tweets, the number of tweets a user posts and also use hashtags as features by collecting data on our own to help classify a Twitter user's gender

References

[1] J. S. Alowibdi, U. A. Buy, and P. Yu. "Empirical Evaluation of Profile Characteristics for Gender Classification on Twitter". In: *2013 12th International Conference on Machine Learning and Applications*. Vol. 1. Dec. 2013, pp. 365–369. DOI: 10.1109/ICMLA.2013.74.

[2] John D. Burger et al. "Discriminating Gender on Twitter". In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, 2011, pp. 1301–1309. URL: http://aclanthology.coli.uni-saarland.de/pdf/D/D11/D11-1120.pdf.

[3] Na Cheng, R. Chandramouli, and K. P. Subbalakshmi. "Author Gender Identification from Text". In: *Digit. Investig.* 8.1 (July 2011), pp. 78–88. ISSN: 1742-2876. DOI: 10.1016/j.diin.2011.04.002. URL: http://dx.doi.org/10.1016/j.diin.2011.04.002.

[4] Morgane Ciot, Morgan Sonderegger, and Derek Ruths. "Gender Inference of Twitter Users in Non-English Contexts." In: *EMNLP*. ACL, 2013, pp. 1136–1145. ISBN: 978-1-937284-97-8. URL: http://dblp.uni-trier.de/db/conf/emnlp/emnlp2013.html#CiotSR13.

[5] W. Deitrick et al. "Gender Identification on Twitter Using the Modified Balanced Winnow". In: *Communications and Network*. Vol. 4. 3. June 2012, pp. 143–148. DOI: 10.4236/cn.2012.43023.

[6] D. Fernandez, D. Moctezuma, and O. S. Siordia. "Features combination for gender recognition on Twitter users". In: *2016 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. Nov. 2016, pp. 1–6. DOI: 10.1109/ROPEC.2016.7830623.

[7] S. Keretna, A. Hossny, and D. Creighton. "Recognising User Identity in Twitter Social Networks via Text Mining". In: *2013 IEEE International Conference on Systems, Man, and Cybernetics*. Oct. 2013, pp. 3079–3082. DOI: 10.1109/SMC.2013.525.

[8] Z. Miller, B. Dickinson, and W. Hu. "Gender Prediction on Twitter Using Stream Algorithms with N-Gram Character Features". In: *International Journal of Intelligence Science*. Vol. 2. 4A. Sept. 2012, pp. 143–148. DOI: 10.4236/ijis.2012.224019.

[9] Juergen Mueller and Gerd Stumme. "Gender Inference Using Statistical Name Characteristics in Twitter". In: *Proceedings of the The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016, Data Science 2016*. MISNC, SI, DS 2016. Union, NJ, USA: ACM, 2016, 47:1–47:8. ISBN: 978-1-4503-4129-5. DOI: 10.1145/2955129.2955182. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2955129.2955182.

[10] *Twitter user data*. URL: https://www.kaggle.com/crowdflower/twitter-user-gender-classification/data.

[11] M. Vicente, F. Batista, and J. P. Carvalho. "Twitter gender classification using user unstructured information". In: *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. Aug. 2015, pp. 1–7.