Problem Statement

Identification of the gender of a Twitter user using the user's profile information

Problem Statement: With the growth of social media in recent years, there has been an increasing interest in the automatic characterisation of users based on the informal content they generate. Gender recognition is essential and critical for many applications in the commercial domains. Imagine that Twitter needs to push advertisements based on gender. As there are many fake accounts or accounts belonging to organisations, Twitter cannot rely on what the users themselves mention in their respective profile descriptions. Hence, Twitter would need to determine the gender of the profile based on user behaviour on the platform.

To enable this, you would need to train an algorithm to determine if a Twitter account belongs to a man or a woman or an organization. You need to build two models based on two different classification algorithms and compare the results. You may choose any of the algorithms that you have been introduced to, throughout the course. Moreover, feel free to proceed with any data preparation technique which suits the given dataset (whether it is covered in the course or not).

Dataset: You can find the dataset here.

Data Description:

The dataset contains 20,000 rows, each with a particular username, a random tweet, account profile and image, location, link, sidebar colour and other miscellaneous data.

The dataset contains the following fields:

Note: You can think of a golden account as a verified account and relevant columns have "gold" either as a prefix or suffix. Twitter verifies the accounts of famous organisations and people so that Twitter users can be sure if the account is fake or not.

Variable Name	**Description**	**Missing Values**
_unit_id	unique id for a user	no
_golden	whether the user was included in the gold standard for the model; TRUE or FALSE	no
_unit_state	state of the observation; one of finalised (for contributor-judged) or golden (for gold standard observations)	no
_trusted_judgments	number of trusted judgments (int); always 3 for non-golden, and what may be a unique id for gold standard observations	no
_last_judgment_at	date and time of last contributor judgment; blank for gold standard observations	yes
gender	one of male, female, or brand (for non-human profiles)	yes
gender:confidence	a float representing confidence in the provided gender	yes
profile_yn	"no" here seems to mean that the profile was meant to be part of the dataset but was not available when contributors went to judge it	no
profile_yn:confidence	confidence in the existence/non-existence of the profile	no
created	date and time when the profile was created	no

Variable Name	Description	Missing Values
description	the user's profile description	yes
fav_number	number of tweets the user has favourited	no
gender_gold	if the profile is golden, what is the gender?	yes
link_color	the link colour on the profile, as a hex value	no
name	the user's name	no
profile_yn_gold	whether the profile y/n value is golden	yes
profileimage	a link to the profile image	no
retweet_count	number of times the user has retweeted (or possibly, been retweeted)	no
sidebar_color	colour of the profile sidebar, as a hex value	no
text	text of a random one of the user's tweets	no
tweet_coord	if the user has location turned on, the coordinates as a string with the format "[latitude, longitude]"	yes
tweet_count	number of tweets that the user has posted	no
tweet_created	when the random tweet (in the text column) was created	no
tweet_id	the tweet id of the random tweet	no

Variable Name	Description	Missing Values
tweet_location	location of the tweet; does not seem to be particularly normalized	yes
user_timezone	the timezone of the user	yes

Note: Please feel free to refer to any additional content online, that you may feel, will help you in the completion of the assignment. However, please be informed that plagiarism, even for a considerable part of the code, will be strictly penalised. Standard libraries usage does not come under plagiarism.

Submission Guidelines

List of Deliverables

1. A fat jar containing all the required classes.
2. A zip file of the entire project.
3. An entire project report in pdf format. The project report must contain the following:
    1. **Data Processing Steps:**

- List of data issues found in the raw data. Presence of null values or any processing like removing irrelevant data or missing values etc. may be mentioned here. (Entire explanation is not required. Just list in bullet points)
- How did you tackle the issues mentioned above? Here, you will need to pick up each issue that you have mentioned above and clearly mention the issue, with a sample or an example, indicating the input you have and the expected output after processing. You need to paste the respective code snippet and the actual output for each issue in the report. For instance, if you are using any UDF function to process a column, then you need to just paste the UDF function, how are you invoking the function on the dataframe and finally the result of the column (5-10 rows) after calling the UDF function.

2. **Model Building :**
    - You need to clearly list all the different models you have used. For example decision tree, random forests etc. Note that different models correspond to different algorithms used for building a model. The models generated by tuning the hyperparameters are considered as one.
    - For each model, clearly explain what data (columns) is being used to train the model.
    - If you are using any hyperparameters while building the model, you need to mention and clearly explain how each parameter impacts the model (for every model you have used).

- In the aforementioned points, you need to paste the respective code snippets. Make sure that the code is clearly visible.(you can use some dark theme editor so that code is clearly visible)

3. **Evaluation Metrics:**
    1. For each model, you need to report the following performance metrics:
        1. Evaluation Scores: Accuracy, Precision, Recall, F1 Score.
        2. Confusion Matrix.
    2. For each model, check if the model is facing any issue of overfitting or underfitting. Report on how you checked this and also paste the code or any graphs(if present).
4. **Inferences & Suggestions:**

    You need to:

    0. Compare the results from the models and mention drawbacks and advantages for each of them.
    1. Suggest some improvisation techniques to those models.
    2. Choose from the two models present and justify why did you choose that particular model for the given problem statement.

Solution Approach

In this segment, you will see an approach to identify the gender of a Twitter user using their profile information.

Approach:

The first step is data preparation where you will need to input the twitter data and drop all the columns and rows which are irrelevant to our classification model.

1. While reading the data setting the mode to DROPMALFORMED will handle the data which isn't matching the schema or having an incorrect number of delimiters.
2. You are interested in classifying users as male, female or a brand which means the records where the gender is unknown have no use. Training the classifier with these records would make the classifier learn 'unknown' as one of the genders to classify.
3. There are mainly columns available in the Twitter data, and this entire data cannot be used in the classification model. The columns which are of interest are _unit_id, gender, description, text, tweet_count. All other columns which aren't useful can be dropped.
4. Records with Null entries need to be removed.

Once you have the data ready, the next step would be to split the data into training and testing data. This step is required since we aren't provided with two separate datasets for training and testing our results. Generally, the data is split in 70:30 ratio or 80:20 ratio.

To build the model you need a label column and features:

1. Using the String Indexer, the categorical attribute 'Gender' needs to be converted to numerical labels.
2. Using multiple transformations on the textual information to get necessary input features - Tokenizer, stopword Remover, stemmer, hashingtf and idf.

Finally, you build a model using a decision tree or random forest classifier and tune it using various hyperparameters like setImpurity, setMaxDepth, setNumTrees, etc. Using the MulticlassClassificationEvaluator you evaluate the classifier with various metrics like precision, recall, accuracy and F1score.

Note: In the presented solution, to perform any analysis we'll be creating a temp view of a dataframe by calling the method **CreateorReplaceTempView()**. The temp view imparts us SQL type functionalities, which make our lives easier while performing any SQL operations or analysis on a particular dataframe.

Note: The following approach is one of the many possible solutions and is not the only approach that always needs to be followed.