**Assignment 03 Date: 6th March, 2019 Last Date of submission: 18th March, 2019**

**Goal of this assignment is to do hands-on on different vectorization methods (sparse and dense vectors) through a text classification exercise, using both linear and non-linear models.** Mail **a python 3.x Jupyter notebook** with **mail subject as "assignment_03_teamXX". Please Indicate team SRNs at the top of the notebook**

1. A review dataset is in CSV file dataset.csv
2. **X** is the text reviews and **y** is 'stars' for categorization problem. Ignore other attributes for this assignment.
3. You will see that there are review ratings (stars) of 1-5 for each text review. Drop all the reviews with rating 3. Update reviews with rating 1 & 2 as 0 rating. Update Reviews with 4 and 5 as 1 rating. We are doing this to make it a binary classification. Make your dataset balanced by dropping reviews of the class that is more in number. Now you will be left with around 3137 reviews of each class.
4. You will proceed with Train and Test Split strategy of the above dataset. Make sure that you do not have an unnecessarily large vocabulary when you vectorize these.

**Case 1:** This is the **baseline case of linear classification using extracted features**. In this, you are assuming that the emotions and sentiments in each review have a direct bearing on the rating. Based on this assumption, we will extract features out of every review i.e. 8 emotion valence using **textacy.lexicon_methods.emotional_valence** and sentiment using **sentiment.polarity** of TextBlob API. Once these features are extracted, you can do linear classification using SVM (use ScikitLearn library)

**Case 2:** In this case, you will use a **sparse vector representation of each review** and then can do the linear classification again using SVM (use ScikitLearn library). For making a sparse vector representation of each review, you have to fit an instance of ScikitLearn's TFIDFVectorizer class on the total set of reviews in the corpus created after step 4 above. As a next step ( scikitlearn has a fit-transform paradigm), you need to transform these reviews into sparse vector of size vocabulary. The scikitlearn API will do it for you . Having done that, you can do **linear classification using SVM** (use ScikitLearn library).

**Case 3:** Next, we will do a **dense vector strategy with SVD** ( singular value decomposition) by extending your work in the previous case. Use SVD ( again SciKitLearn API) to create dense vector representation of each review text and then again do **linear classification using SVM** (use ScikitLearn library)

**Case 4:** So far, we have NOT used **non-linear classification** at all. Now we will use ANN for this case and use Keras Library. We will use **Word2Vec (You may use Google vector file and you need to take care of word that is not part of Google Corpus) to create dense vector representation of each review** and then use that as input into an Artificial Neural Network. Build a non -deep Neural Network to save time.

**Case 5 (Optional)** : You can extend the case 4 by changing few lines of the Keras Model with CNN specific code. CNN is good for extracting features.

Finally comment about your findings on the accuracy by these methods.

**Note :** You refer to resources available in the net. Most of the above are template code.