Task 2 - Classification

**Task:**

Create a Review Sentiment Classifier using Consumer Reviews dataset. It can be either a Binary Classifier with 2 classes (Positive and Negative) or a Multi-class classifier with 3 classes (Positive, Negative and Neutral). You are free to choose your categories and their names.

A reference dataset is provided in the shared folder with features viz. Categories, reviews, ratings and their titles. The review sentiments are labelled as ratings and you are expected to use a supervised technique to classify reviews into two or three buckets as stated above.

*A separate test dataset will be given on 27th July(Saturday) which will be strictly used to evaluate test accuracy.*

**Solution:**

1. A Notebook that explains your thought process. This is similar to the previous report you submitted. Please write readable code and explain every step in detail. Make sure your NLP pipeline is explained clearly with an example sentence. We only understand plain English sentences.
2. An API that accepts a review text as input and predicts the sentiment for that new review text input. You can choose to host this API on a public remote server or send a video recording of your Postman test.

**Deadline:** 28th July 2019, 11:59pm.

**Optional Task:**

Scale up the size of dataset by writing a crawler engine that collects reviews of products from Amazon's e-commerce website. Do this if you are done with the given task and want to train/test on more data.

**Hints:**

1. CNNs and RNNs are great at classification tasks dealing with textual data.
2. Neural Networks only outperform conventional ML algorithms when trained on large volumes of data.
3. A lot of problems in systems leveraging learning algorithms arise due to problem of bias in the dataset. Class Imbalance in the dataset is one of the reasons why the learning algorithm gets biased. This will happen when you label your dataset after clustering.

**Points to Note**:

1. Choose an appropriate method to generate character/word/sentence level embeddings.
2. You are free to choose your performance metric(s). However there has to be a clear rationale behind your choice of performance metric(s) with reference to the learning algorithm used and which other options were explored.
3. Your model needs to be interpretable. Defining just the performance metric without ways to interpret, validate and test the model would not fetch good scores during evaluation.
4. You are free to choose your choice of platform for building the model - Google Colab, GCP, AWS, Intel AI Dev Cloud, local system, etc. However there has to a precise README.txt on your submission repository on how to run the code and relevant platform details. Please make it as easy as possible.
5. Source code needs to be well commented. A bad model with descriptive comments explaining the rationale behind each step will fetch more score during evaluation as compared to a great model with no comments.

All the best :)