README

SECTION 1: FILES

This readme contains information about how to use the files downloaded in this folder. The folder should contain:

- `README.md`
- `project_#_tfidf_matrix.pkl` file – this is the TFIDF (see section 2) matrix of the uploaded data.
- `project_#_training_#.pkl` file – this is the model itself (see section 3), trained on the most recent labeled data
- `project_#_labeled_data.csv` – this file contains all labeled data, with the original text, unique ID, and assigned label.
- `project_#_labels.csv` – This file contains the mapping between label name and ID.
- `project_#_vectorizer.pkl` – This file contains the fitted TFIDF Vectorizer (see section 2). This can be used to fit new data to the same format as the training data (see section 4 for examples)

NOTE: All models and TFIDF files use Scikit-Learn version 0.19.0. If you are using a different version, be careful!

SECTION 2: TFIDF

A. What is TFIDF?

Term Frequency Inverse Document Frequency is a way of quantifying text data. For each document (piece of text, in this case one tweet, news article, etc.) a count is created for the number of times each term (word) appears. This count is then scaled down by the amount of times the word appears in all of the documents (so common words like “the” are given a lower score). The result is a set of numerical features representing each document by the words in it.

Note that this method of quantifying text is a bag of words approach, and does not take the location or co-occurrence of words or phrases within a document into account.

B. File format

The text data loaded into SMART is preprocessed with Scikit-Learn’s TFIDFVectorizer with the parameters:

- max_df: 0.995 (only keep those terms with document frequency lower than this value)
- min_df: 0.005 (only keep those terms with document frequency higher than this value)
- stop_words: English (Automatically remove words like “the”, “at”, “and”, etc.)

The data is then placed into a dictionary of unique id to values (ex: { “tweet1” : [0, 0.5, 0.8, ….], “tweet2”: [0.56, 0.0, 0.99, …..] }). It is saved as a pickle file with the .pkl ending.

SECTION 3: THE MODEL

A. Source

The models used in this application are build using Scikit-Learn libraries. The four options are:

- Logistic Regression
  - Parameters: class*weight: balanced, solver: lbfgs, multi*class: multinomial
- Support Vector Machine
  - Parameters: default
- Random Forest
  - Parameters: default
- Gaussian Naïve Bayes
  - Parameters: default

B. File Format

All models are saved as pickle (.pkl) files through Scikit-Learn’s joblib library.

SECTION 4: HOW TO RUN

A. Preprocessing new data for the model to predict

The following sample code can be used to preprocess new data and get it in a format that is usable by the saved model.

```
with open(<<project_#_vectorizer.pkl>>,"rb") as tfidf_vectorizer:
    # load the vectorizer
    tfidf_transformer = pickle.load(tfidf_vectorizer)

new_data = ["This is new data","that needs to be formatted", "In the same way that th
e other data was before"]

# apply it to new data
transformed_data = tfidf_transformer.transform(new_data)
```

Note: to see what words the fields in the transformed TFIDF matrix are based off of, just call `print(tfidf_transformer.vocablulary_)`. This will print a dictionary of the words that were used and their counts in the training data.

B. Applying the model to predict new data

The following sample code can be used to read in the zipped files and get predictions for the unlabeled data. (The `<<>>>` and `#` are placeholders for your file names and project number).

NOTE: the model will predict labels as a number. The project_#_labels.csv gives the mapping between label text and ID.

```
import pandas as pd
import pickle
from sklearn.externals import joblib

# read in the TFIDF matrix and the labeled data
labeled_frame = pd.read_csv(<<project_#_labeled_data.csv>>)
with open(<<project_#_tfidf_matrix.pkl>>,"rb") as tfidf_file:
    tfidf_dict = pickle.load(tfidf_file)

# Subset the TFIDF matrix by the unlabeled data and get as 2D list
labeled_ids = labeled_frame["ID"].tolist()
unlabeled = [tfidf_dict[key] for key in tfidf_dict if key not in labeled_ids]

# read in the model from the pickle file
model = joblib.load(<<project_#_training.pkl>>)

# apply the model to the unlabeled data
predictions = model.predict(unlabeled)

# print the results
print(predictions)

# open the label dictionary to translate label ID's to their corrosponding text
labels_frame = pd.read_csv(<<project_#_labels.csv>>)
label_dict = labels_frame.set_index("Label_ID").to_dict()["Name"]

# get the predictions as actual labels
predictions = [label_dict[pred] for pred in predictions]
print(predictions)
```