# CS 491/691 Project 3

Natural Language Processing

Please submit one zip file to Webcampus in a folder named Project3 (a zip file with all the applicable ".py" files as well as your README.txt). You are provided with "test_script.py". I will use the "test_script.py" to evaluate your implementations. Make sure you can run "test_script.py" without errors.

Refer to these resources for help with Doc2Vec:
https://medium.com/@mishra.thedeepak/doc2vec-simple-implementation-example-df2afbbfbad5

https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html#sphx-glr-auto-examples-tutorials-run-doc2vec-lee-py

## Load and clean the dataset (20 points)

`def load_data(fname):`
This function takes as input the filepath to the dataset. You should load the data from the file and return the stored data in whatever format you'd like. This function should return a list (or equivalent) of documents and a list of labels.

`def clean_data(documents):`
This function takes the loaded dataset as input, removes any invalid data (i.e. nan/NULL values or corrupted data), and changes the labels from values of either 0 or 4 values of either 0 or 1. Do other kinds of preprocessing you see fit, just make a note of it in your README.txt. The function should return the cleaned data and labels.

## Train the doc2vec model and transform the dataset (40 points)

`def train_doc2vec(cleaned_documents):`
This function should take your cleaned dataset, tokenize it using the nltk library, and train the doc2vec model from gensim. Please save the model to your computer's disk as well as return it from the function. Note that there are many hyperparameters. Feel free to play with them, but make note of your process in the README.txt

`def tokenize_data(cleaned_documents, d2v_model):`
This function should take your cleaned dataset and trained doc2vec model as input. Vectorize each document (tweet) in the dataset with the "infer_vector" command from the d2v_model. Return from the function a list (or equivalent) of vectorized documents.

## Train and test models and compute performance metrics (40 points)

def train(X_train, y_train):
This function should take the training split of your vectorized dataset and training split of your labels as input. Train at least 2 models, of your choice, from scikit-learn. Return from the function a dictionary where the key is the name of a model and the corresponding value is a trained model object.

def test(trained_models_dict, X_test, y_test):
This function should take as input your trained models dictionary, the test split of your vectorized dataset, and the test split of your labels as input. Please calculate the accuracy, balanced accuracy, and f1-score for each model on the test dataset. Return from the function a dictionary where the keys are the model names with the performance metric appended to the end (i.e. “decision_tree_classifier_accuracy” or “linear_svc_f1”) and the corresponding values are the computed performance metric value.