Machine Learning	**Homework 2**
CSCI 5622 Fall 2017	**Due Time Sep 29, 2017**

Submission Guidelines

- In order to download files required for the homework, clone `https://github.com/BoulderDS/csci_5622_hws`.
- For programming questions, submit python source files in a zip file.
- For other questions, submit a PDF file of no more than 4 pages.

All homework submissions are done through Moodle.

1 Logistic Regression (40 pts)

In this homework you'll implement a Logistic Regression classifier to take drawings of either an eight or a nine and output which number it corresponds to.

1.1 Programming questions (25 pts)

Finish `logreg.py` to achieve the following goals. You can use `tests.py` to test your code.

1. Finish the *sgd_update* function so that it performs stochastic gradient descent on the single training example and updates the weight vector correspondingly.
2. Finish the *sigmoid* function to return the output of applying the sigmoid function to the input parameter.
3. Finish the code in the *main* function to loop over the training data and perform stochastic gradient descent for the user-defined number of epochs.

1.2 Analysis (15 pts)

1. What is the role of the learning rate (eta) on the efficiency of convergence during training?
2. What is the role of the number of epochs on test accuracy?

2 Feature Engineering (40 pts)

In many practical machine learning problems, the raw data is not provided in a format that is easily understood by learning algorithms. To get the best performance from a machine learning model, certain dimensions or features need to be engineered by the programmer prior to the learning phase. These engineered features transform the data into a representation that is better understood by a machine. For example, if we want a machine learning model to learn about natural language, it may be better to feed in frequency counts of the words in a document, compared to just a plaintext string.

2.1 Programming questions (25 pts)

Finish `feature_eng.py` to achieve the following goals.

1. Add several custom feature transformers to accompany the *TextLengthTransformer* and add the custom features to the FeatureUnion. (hint: Use *TextLengthTransformer* as an example of how to do this.)
2. Use scikit-learn to add n-gram features (unigram, bigram, ...) to the FeatureUnion. (hint: look at "vectorizers" in scikit-learn, pay attention to the default choice of regular expression for the tokenizer.)

2.2 Analysis (15 pts)

1. What custom features did you add/try (other than n-grams)? How did those additional features affect the model performance? Why do you think those additional features helped/hurt the model performance?
2. What are unigrams, bigrams, and n-grams? When you added those features to the FeatureUnion, what happened to the model performance? Why do these features help/hurt?
3. **EXTRA CREDIT (extra 10 pts):** Replace the code that fits the *SGDClassifier* on the training data with scikit-learn's k-fold cross validation and use it to determine the best regularization constant *alpha* for the classifier. Why do we need cross validation to tune the *alpha* parameter? What did you determine was the best value for *alpha*? What experiments did you run and what were your cross validation results?

3 Gradient Descent Learning Rule for Multi-class Logistic Regression (20 pts)

We introduced binary classification using logistic regression in class. This framework can be extended to perform multi-class classification. Specifically, assuming the training set is $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$, $y_i \in \{1, 2, \ldots, C\}$, independent and identically distributed with

$$p(y = c|\boldsymbol{x}) = \frac{\exp(\boldsymbol{\beta}_c^T \boldsymbol{x})}{\sum_{c'=1}^{C} \exp(\boldsymbol{\beta}_{c'}^T \boldsymbol{x})}.$$

Note: since we are doing multi-class logistic regression, we have a different weight vector $\boldsymbol{\beta}_k$ for each class c.

- Derive the negative log likelihood for multi-class logistic regression.
- The gradient descent learning rule for optimizing weight vectors generalizes to the following form: $\beta_j^{t+1} = \beta_j^t - \eta \nabla \beta_j^t$ where η is the learning rate. Find the $\nabla \beta_{c,j}$ (the parameter for feature x_j in class c) for a multi-class logistic regression model.