University
of San Diego®
SHILEY-MARCOS
SCHOOL OF ENGINEERING

Assignment 5.1 Exercises

For this assignment, you will conduct programming tasks in Google Colab and answer provided questions in a Google Doc or MS Word document. You will submit **2** documents to Blackboard:

1. A PDF file converted from Google Colab.
2. A PDF file converted from your Assignment 5.1 Short Answer Template Word Doc (document is linked in Blackboard assignment prompt).
    - This template has a copy of all short answer questions for this assignment.

Movie Data
For this assignment, you will work with the movie data. This data used here are a set of 25,000 movie reviews. For linear classification, this dataset gives you an example of using regularization on a dataset with many examples and many features (words).

The dataset for this assignment comes from this publication, Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*.

As you go through the assignment, you will create several tables and figures. After you complete the programming section, use the tables and figures you generated to answer the following questions.

Programming Section (Complete this section in Google Colab)

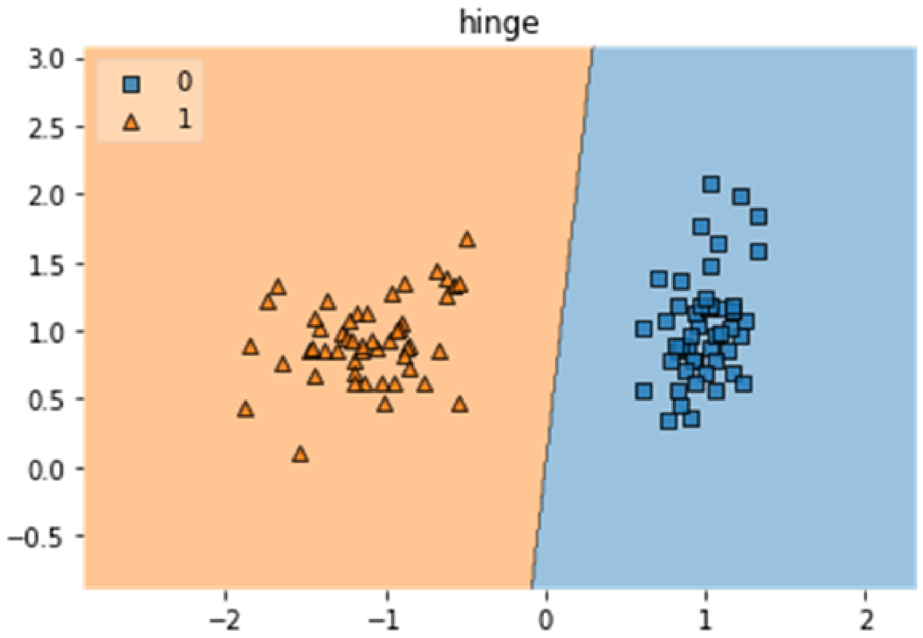
Linear Classification:
Install the package *mlextend*: http://rasbt.github.io/mlxtend/installation/

Next, you will do a few exercises to visualize the difference between the different linear classifiers.

Generate classification data using make_classification from sklearn.datasets:

```
X, y = make_classification(n_features=2, n_redundant=0,
n_informative=2,random_state=1, n_clusters_per_class=1)
```

Use SGDClassifier to train classifiers using different loss functions: log, hinge, and perceptron. Visualize the trained classifiers using `plot_decision_regions(X, y, clf=model, legend=2)` from the *mlextend* package. Plot the decision region for each of the three loss functions: a sample plot for hinge loss appears below.



Now, create a larger classification dataset. You will use cross_val_score from scikit-learn and compare this to bootstrap_scores from *mlextend*.

Set up the simulated data as follows:

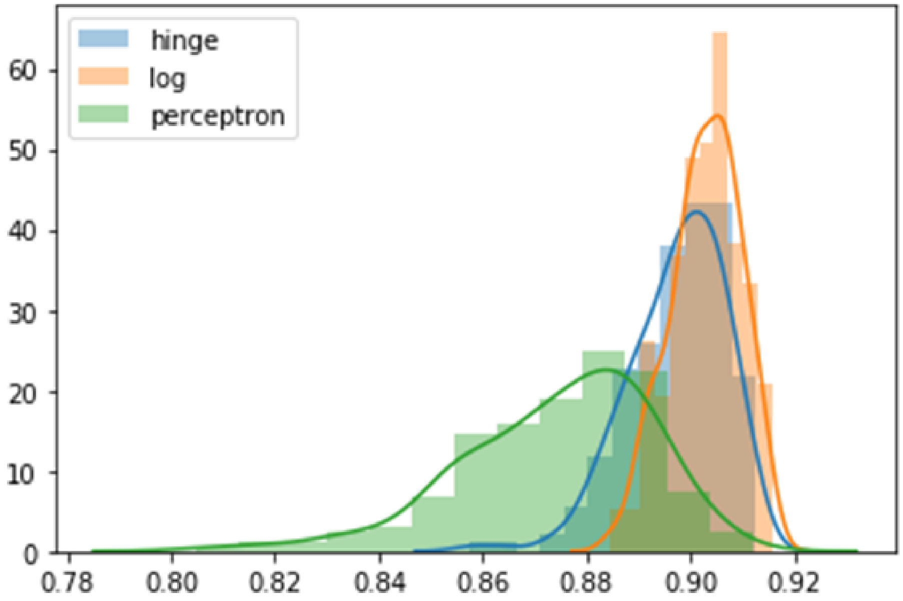
```
X, y = make_classification(n_samples=10000,n_features=20,
n_redundant=0, n_informative=20,random_state=1,
n_clusters_per_class=1)
```

With your classifier in a variable called model (using SGGClassifier, as before), you can get the accuracies as follows:

```
scores = cross_val_score(model, X, y, cv=5,scoring=scoring)
bootstrap_scores = bootstrap_point632_score(model, X, y,
method='oob')
```

Create a table with the average of each cross-validation score and the average of the bootstrap scores.

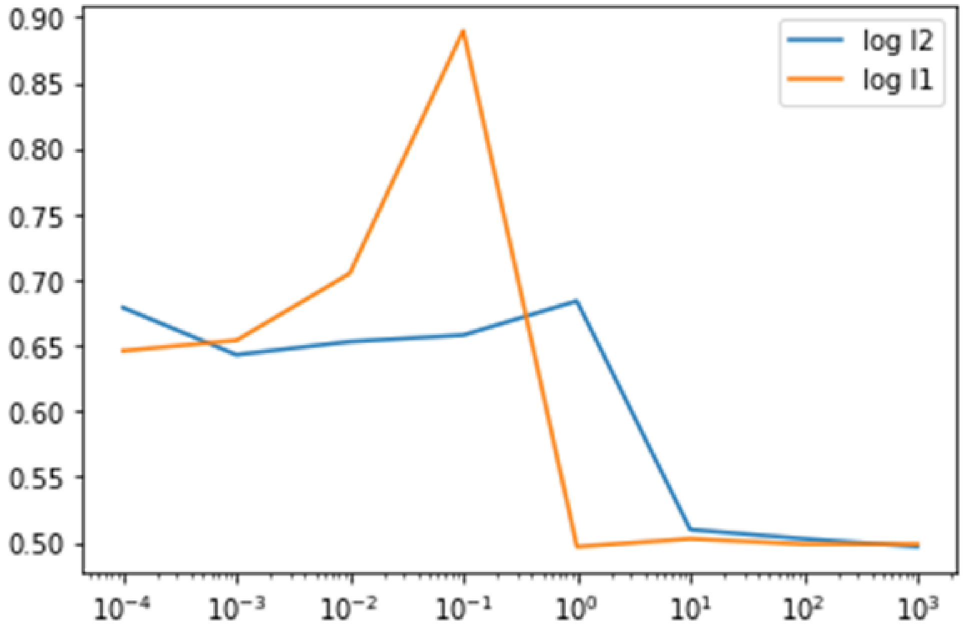
Finally, use distplot from the *seaborn* package to plot your bootstrap samples.



Finally, you will look at the importance of setting the regularization parameter. Create a database with only two informative features:

```
X, y = make_classification(n_samples=1000,n_features=2000,
n_redundant=0, n_informative=2,random_state=1,
n_clusters_per_class=1)
```

Train a regularized classifier using 'log' as the loss function. Try both types of regularization 'l1' and 'l2' and sweep alpha over a range from: [0.0001,0.001,0.01,0.1,1,10,100,1000]. Use *fivefold* cross-validation to measure accuracy. Create a table of accuracy indexed by alpha, and use this table to plot the accuracy for both types of regularization. Your plot should look like this:



Written Questions **(Complete these in the Assignment 5.1 Short Answer Template, located in your assignment prompt)**

(Short answer, 2-3 sentences each):

- In the simulated data section, you used the bootstrap to show you the distribution of the estimated accuracy of your classifier. You plotted this for three different classification methods. Based on this plot, do you think any of the three classification methods is significantly better than the other methods (on this particular dataset)?
- When you had the simulated data and looked for the best regularization parameter, one method (l1 or l2) had a clear advantage over the other. When you used the same code and methods on the text data, was one method of regularization clearly better than the other? Why do you think this was the case? Elaborate.