# CSC 546/746

## Assignment 4

## (25 points)

1. In this exercise, you will build a logistic regression model to predict whether a student gets admitted into a university. Suppose that you want to determine each applicant's chance of admission based on their results on two exams. You have historical data from previous applicants that you can use as a training set for logistic regression. For each training example, you have the applicant's scores on two exams and the admissions decision. Your task is to build a classification model that estimates an applicant's probability of admission based the scores from those two exams.

2. **Setup:**
    a. (1 point) Download “hw04_data.csv” from the Blackboard and upload (or copy) to your Jupyter Notebook folder. Create a new Jupyter Notebook project and name it as “hw02.ipynb”.
    b. (2 points) Import necessary libraries (import matplotlib.pylot as plt, otherwise the provided code will not work)
    c. (2 points) Load the data from “hw04_data.csv” into a Pandas data frame. Put the first two columns of data (exam scores) into “X” and the last column of data (admission decisions) into “y”. (**Note:** you should assign data from the Pandas variable. Converting the Pandas variable to a Numpy variable is not allowed)

3. **Visualization:**
    a. (1 point) Copy the following code to your project.

```
plt.figure(figsize=(10, 10))
# Plot the data for the “Not admitted” entries
plt.plot(X[y==0, 0], X[y==0, 1], "bs", label = "Not admitted")

# Plot the data for the “Admitted” entries with the style of
green triangles.
# You need to complete the code here

plt.legend(loc="best")

# Add labels for the coordinates
# You need to complete the code here

plt.show()
```

    b. (3 point) Complete the missing code and visualize the data.

4. **Train the Logistic Regression model**:
    a. (1 points) Splitting the dataset into the Training set (80%) and Test set (20%). Set the random state to 42.
    b. (1 points) Training the LogisticRegression model with the “lbfgs” solver. Fit the training data to the model (Assign the model to the variable “log_reg”).
5. **Visualization the decision boundary:**
    a. (1 point) Copy the code from the cell for question 3 and add the following code before the command “plt.show()”.

    ```
    left_right = np.array([20, 100])
    # If you save your model to a variable other than “log_reg”,
    you need to update the following command accordingly.
    boundary = -(log_reg.coef_[0][0] * left_right +
    log_reg.intercept_[0]) / log_reg.coef_[0][1]
    plt.plot(left_right, boundary, "k--", linewidth=3)
    ```

    b. (1 point) Observe the result and answer the following question in comments:
    Did the decision boundary perfectly separate the admitted entries from the unadmitted entries?

6. **Evaluation:**
    a. (2 points) For a student with an Exam 1 score of 45 and an Exam 2 score of 85, estimate whether this student get admitted or not. Print out the probability of the prediction too.
    b. (1 points) Print out the prediction results of the test dataset.
    c. (1 points) Print out the score of the model.
    d. (1 points) Print out the confusion matrix.
    e. (3 points) Calculate the Precision, Recall, and F1Score for this model. Put your answers in the comments or a markdown cell. (Formulas are required)
7. **(4 points) Analysis:**
    Create a new cell by the end of hw04.ipynb and discuss the model in this cell. You could write down any thoughts about this model. Be creative!
    Some questions you may consider:
    a. What do you think about the performance of this model?
    b. Do you think the linear decision boundary fits the model?
    c. What could you do to improve the performance of the model?
8. Submit “hw04.ipynb” to the Blackboard.