COMP.4220 - Machine Learning

Homework 4

student name:

1. Show that the logistic sigmoid function $\sigma(a) = \frac{1}{1+e^{-a}}$:

   (a) satisfies:

$$\sigma(-a) = 1 - \sigma(a).$$

   (b) and its inverse is given by

$$\sigma^{-1}(y) = \ln\{\frac{y}{1-y}\}.$$

(6 marks)

2. (Programming) In this section, you implement a Logistic Regression model and apply it to a synthetic data set. For this part, only work in `HW4_part1.py` file.

   (a) In the `LogisticRegression` class implement the `sigmoid` function.

   (b) In the `LogisticRegression` class complete the implementation of the `fit` function by adding the formula for prediction using the `sigmoid` function.

   (c) In the same class, within the `fit` function implement the gradient part.

   (d) In the same class, within the `fit` function implement the Hessian part.

   (e) In the same class, within the `fit` function finish the implementation by finding **w**. This line must make use of the gradinet and the Hessian calculated in previous steps.

   (f) in the same class, implement a function that returns the probability of classifying a new point **x** using the trained **w**.

   (g) In the same class, implement the `classify` function that uses a threshold over the probability value and returns a class label.

   (h) Use the `create_synthetic_data` function in the starter code to create a training data set for a two-class classification problem. Make sure you add outliers. The output should be x_{train} and t_{train} including 50 data points in each class with an extra 10 points as outliers.

   (i) Now generate 100 points for your testing set x^1_{test} and x^2_{test} for classes 1 and 2, respectively. The generated test set must be in range $[-5, 15]$. To avoid sparsity, instead of returning two 1D vectors, you should generate two coordinate matrices. Here is an example:

```
    a = np.array([0, 0.5, 1])
    b = np.array([0, 1])
    # your code should output
    A = np.array([[0, 0.5, 1],
                  [0, 0.5, 1]])
    B = np.array(([0, 0, 0],
                  [1, 1 1]])
```

   Then combine the two arrays into one array x_{test} with size $10,000 \times 2$ (i.e., 10,000 points with two coordinates). This is the number of points your classifier will be tested on.

   (j) Instantiate the `LogisticRegression` class and train it using the training set.

   (k) Use the trained model to classify the test set and set the prediction outputs into an array.

   (l) Plot your results by using the `scatter` function for the training set, and the `contourf` function for the test set. Make sure axes are limited to $[-5, 15]$.

Tips: (1) do not forget to add the bias term, (2) test your code multiple times and see if you notice cases when the algorithm fails to find a solution, and (3) see if you can control the number of iterations in cases when you need less or more steps before the training stops. (12 marks)

3. (Programming) In this section, you apply the Logistic Regression model from the `Scikit-learn` module to a small real-world data set. All the work related to this section is to be done in the `HW4_part2.py` file.

   (a) Import the data set using the following line and check the description of the data set by printing the description as we did in previous homework. Assign the outputs to $\mathbf{X}$ and $\mathbf{t}$.

   ```
           from sklearn.datasets import load_wine

   ```

   (b) Let's say you are interested in the samples 13 and 133, and want to know their class name. Print class names associated with those samples.

   (c) In previous homework, you used the `StandardScalar` to standardize the data set. This time import and use `MinMaxScalar` to the default $[0, 1]$ range. We do this because the features are real positive numbers and transforming them using `StandardScalar` generates numbers around mean zero which include negative values.

   (d) Using the `train_test_split` function from `Scikit-learn` assign 80% of the points to the training set $(X_{\text{train}}, t_{\text{train}})$ and the rest to the test set $(X_{\text{test}}, t_{\text{test}})$.

   (e) Instantiate the `LogisticRegression` class from `Scikit-learn` and train a model. Explain your tuning process and what parameters you used (e.g., solver, penalty, maximum number of iterations).

   (f) Use the test set, X_{test}, to predict, asign the output to $\mathbf{y}_{\text{pred}}$, and print out the `score` for the test set.

   (g) Import and use the `classification_report` function and print out a report by performing prediction on the whole data set. Include the target labels in the report.

   Tips: refer to the `Scikit-learn` documentation to learn about the functions and their features. (7 marks)