Assignment 3

September 2020

Instructions

- This assignment should be completed individually.
- Do not look at solutions to this assignment or related ones on the Internet.
- The files related to the assignment are present in `lab3-rollno.zip` folder. Extract it and upload it on moodle in the same .zip format after completion and after replacing the string "rollno" with your actual roll number. For example, if you roll number is 00405036, then single zip folder that you will upload will be named "lab3-00405036.zip". Also collate all the CS337 based theory solutions into ONE pdf file named `answers.pdf`. Include `answers.pdf` inside the zip folder mentioned above and submit the zip folder.
- Answers to all subjective questions need to be placed in single pdf `answers.pdf` including all plots and figures and uploaded.
- Only add/modify code between `TODO` and `END TODO` unless specified otherwise. You must not import any additional libraries.
- Python files to submit - `perceptron.py`, `utils.py`, `binary_logistic_regression.py` and `multiclass_logistic_regression.py`
- This Assignment carries a total of 4 marks for CS337 Theory and 13 marks for CS335 Lab

1 Feature Design for Perceptron

This task requires using the One-vs-rest classifier implemented in Assignment 2 for dataset D2.

In the previous Assignment, we used the simplest possible features: value of each pixel. In this task, you have to design at most 5 features using the pixel values and achieve good accuracy. The features have to be designed keeping in mind the dataset it is being applied to (see the images in `data/D2/images.zip`.

As a concrete illustration, consider the data you have used. In each training point (a binary matrix corresponding to an image), consider the number of separate, connected regions of white pixels. This quantity, although it does not vary across shapes, is an example of a feature that is not directly available to the classifier from the per-pixel information. Further you can add features that cumulatively describe all the per pixel values. You can also try analysing edges and corners. Note that you are not allowed to import any libraries for this task, strictly adhere to the TODOs in the file.

1.1 CS335: Lab

(a) In `perceptron.py`, fill up the function `get_features(x)` which given a flattened list of pixel values for an image, to return the set of features. For other functions you can use the code from Assignment 2. Full marks will be given if test accuracy is at least 90%. (2 marks)

(b) Explain the features used and the motivation in the report (1 mark)

2 Logistic Regression

2.1 CS 337: Theoretical Problem

Consider the following formulation for extending the logistic regression to a multi-class classification setup-

$$P(Y = k|\mathbf{w}_k, \phi(\mathbf{x})) = \frac{e^{\mathbf{w}_k^T \phi(\mathbf{x})}}{\sum_{k=1}^{K} e^{\mathbf{w}_k^T \phi(\mathbf{x})}}$$

Here, each $\mathbf{w}_k$ is a class specific vector of dimension equal to number of features in $\phi(x)$. This formulation is called softmax. Note that this formulation is slightly different from the multi-class logistic mentioned in Lecture 9. Each of the weight vectors $\mathbf{w}_k$ are computed by optimizing the categorical cross-entropy loss function.

$$E(\mathbf{W}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_k^{(i)} \times log(P(Y = k|\mathbf{w}_k, \phi(\mathbf{x}^{(i)})))$$

Here $y^{(i)}$ is a K length one-hot vector representing the label of the i^{th} example, and $\mathbf{W}$ is a matrix having $\mathbf{w_k}$ as its rows. Note that $\mathbf{W}$ is a matrix of dimensions (num_features x num_classes).

(a) Show that cross entropy used to train a binary logistic regression (Eqn (3) of Lecture 9) is a special case of this. (1 mark)

(b) In this task, you need to find a vectorized expression for the gradient of $E(\mathbf{W})$ with respect to $\mathbf{W}$. (3 marks)
One approach to this problem is the following- first assume that you have access to a matrix Z of dimension $N \times K$, such that-

$$Z = \Phi(X)\mathbf{W}$$

Doing this simplifies the expression for error to be in terms of Z. Then compute the gradient of E wrt Z (this will be of dimension $N \times K$, and can be computed by first principles since E is a scalar), and Z wrt $\mathbf{W}$. Finally, use the chain rule to get the required gradient.

2.2 CS 335: Lab Problems

(a) In this problem, we will try to predict the popularity of a song based on its properties. The dataset `songs.csv` contains information about numerous songs from 1990-2010, with the last column indicating weather the song made it to Top 10 of the Billboard Hot 100 Chart(1 indicating it did, 0 otherwise). We will implement logistic regression for this binary classification problem. Perform the following tasks.

(i) Complete the function `split_data()` in `utils.py` to split the data such that the train set consist of all the songs up to and including the year 2009 and test set consist of songs that released in 2010. (0.5 marks)

(ii) Inside the function `load_data()` in `utils.py`, discard columns(up to 5) which you think won't be relevant in predicting the output. (0.5 marks)

(iii) Complete the functions `train()` and `predict()` in the `BinaryLogisticRegression` class in `binary_logistic_regression.py`. (2 marks)

Make sure you write an efficient vectorized implementation for each task. Note that since this is a binary classification task, you don't need the softmax formulation described above. You are allowed to change learning rate `lr` and maximum iterations `max_iter`.

(b) Report the test accuracy you obtained. Now, consider a model M which predicts 0 for any song. What accuracy does this model achieve on the test set? Do you think accuracy is a good evaluation metric here? Briefly justify your answer. (1 marks)

(c) Consider a different evaluation metric F_1 score, defined as the harmonic mean of precision and recall. Precision is defined as the fraction of positive outputs which are actually positive. Recall is defined as the fraction of actually positive samples predicted as positive. Formally, let's denote TP as true positives, FP as false positives and FN as false negatives. Then recall, precision and F_1 score are defined as follows:

$$\text{recall} = \frac{TP}{TP + FN}$$

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{F}_1 = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

Complete the function `f1_score()`. Report the F_1 score you obtain for test set. Also, report the F_1 score achieved by the model M described in the previous part. Do you think F_1 is a good evaluation metric for this task? Briefly justify your answer. (2 marks)

(d) Now, we will implement logistic regression for multi-class classification problem using the formulation described in section 2.1. We will use dataset `data/D1` for this task. Complete the functions `softmax()`, `train()` and `predict()` in the `LogisticRegression` class in `multiclass_logistic_regression.py`. Use the vectorized expression you derived in problem 2.1(b) to calculate the gradient of categorical cross-entropy loss function with respect to weights. Again, You are free to change the learning rate and maximum iterations. (3 marks)

(e) You have implemented both, logistic regression and perceptron for dataset $D1$. Which one achieves more test accuracy? Why? Informally justify your answers. (1 marks)