# Programming Project 1: Logistic Regression

Deadline: January 29, 2024

In this project, you are given a training set and a test set which contain information about salaries of employees in a company. You will need to complete the following coding tasks:

1. Do some data cleaning and analysis on the given datasets.
2. Build and train a logistic regression model to predict if an employee has a salary higher than $\$50K$. You will need to implement the gradient ascent method to train the model.
3. Report data analysis and the training performance of your model.

## 1 Preliminaries

**Data Description.** You are given a training set stored in the *train_data.csv* file and a test set stored in the *test_data.csv* file. In these files, each row represents a different employee (or data samples), and the columns include information such as age, education, etc. In total, there are 14 features corresponding to the first 14 columns in the datasets. The last *income* column shows if each employee has an income $\leq 50K$ or $> 50K$, which represents true labels of the samples. The training set contains $26,049$ data samples and the test set contains $6,512$ samples.

**Required Libraries.** To complete your tasks, you will need to install the following libraries:

- `numpy`: To install via Anaconda, run the command: `conda install numpy`
- `pandas`: To install via Anaconda, run the command: `conda install pandas`
- `scikit-learn`: To install via Anaconda, run the command: `conda install scikit-learn`
- `matplotlib`: To install via Anaconda, run the command: `conda install matplotlib`
- `seaborn`: To install via Anaconda, run the command: `conda install seaborn`

## 2 Task 1: Data Cleaning and Analysis (30 points)

In this task, you are provided a file with starter code in it, *proj1_data_analysis.ipynb*. You have to implement three methods in this file:

1. **(15 pts)** `clean_data(train_data, test_data)`: In this method, you need to write codes to:

(a) Fill in missing values: There are missing entries that have a value of "?"; you need to replace these "?" values with suitable values.

(b) Find entries with strange values and replace them with suitable values: For example, in the *capital.gain* column, there are entries with a value of 99999.0, which is completely out of the normal range of values in this column. These entries can be considered as outliers that need to be addressed.

(c) Drop unnecessary columns from the datasets, including *education.num* and *fnlwgt*.

(d) Simplify the *race* and *education* columns. In particular, in the *race* column, find all entries with values in ['Black', 'Asian-Pac-Islander', 'Amer-Indian-Eskimo', 'Other'] and replace them with the same value of 'Other'. In the *education* column, find all entries with values in ['11th', '9th', '7th-8th', '5th-6th', '10th', '1st-4th', 'Preschool', '12th'] and replace them with the same value of 'School'.

(e) Convert labels into binary values: that is, in the *income* column, replace the value of $\leq 50K$ with the value of 0 and replace the value of $> 50K$ with the value of 1.

(f) Lastly, save the processed data into *cleaned_train_data.csv* and *cleaned_test_data.csv* files.

2. **(5 pts)** `plot_numeric_feature_correlation(train_data)`: In this method, you need to write codes to compute and plot the heatmap showing the correlation among numeric features and true labels using the training set. To complete this task, you can use the `pandas.DataFrame.corr` method to compute the correlation and the `seaborn.heatmap` method to plot the heatmap. The plotted figure should look similar to the figure below:

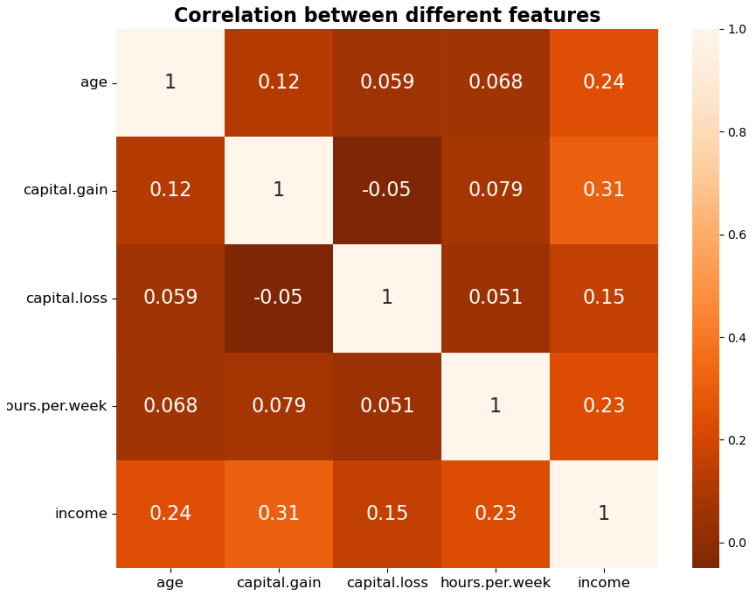
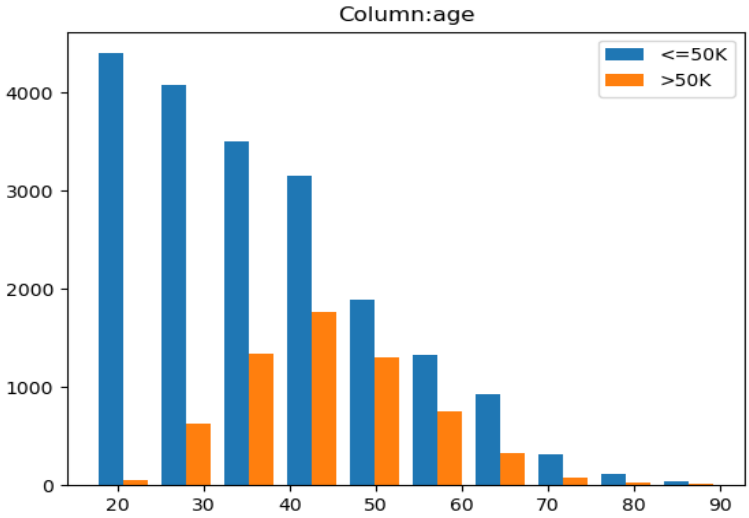



Figure 1: Correlation Heatmap

3. **(10 pts)** `plot_histogram(train_data)`: In this method, you need to write codes to plot histograms showing the number of samples in the training set that have income $\leq 50K$ and $> 50K$ for every column. For example, the histogram for the *age* column should look like:




## 3 Task 2: Implement Logistic Regression Model (60 points)

You are provided a file with starter code, *proj1_classification.ipynb*. You have to build and train a logistic regression model to predict if an employee has a salary $\leq 50K$ or $> 50K$ using this starter-code file. You have to implement:

1. **(40 pts)** Class `LogisticRegression(object)`: In this class, we already provided a completed `__init__()` function which provide all necessary inputs to train the model, including:

   - Training set: `(self.X_train, self.y_train)`
   - Test set: `(self.X_test, self.y_test)`
   - Learning rate: `self.learn_rate`
   - Number of gradient ascent iterations `self.n_iters`

   In this `__init__()` function, we also initialized the model weights `self.weight`. You need to update these weights during the gradient ascent process. Essentially, you will implement the following three methods to train the logistic regression model using gradient ascent.

   (a) `predict_prob(self, samples)`: The input of this method is a set of samples, named `samples`. You will use the current model weights `self.weight` to compute and return the predicted probability that each sample in `samples` has the income $> 50K$.

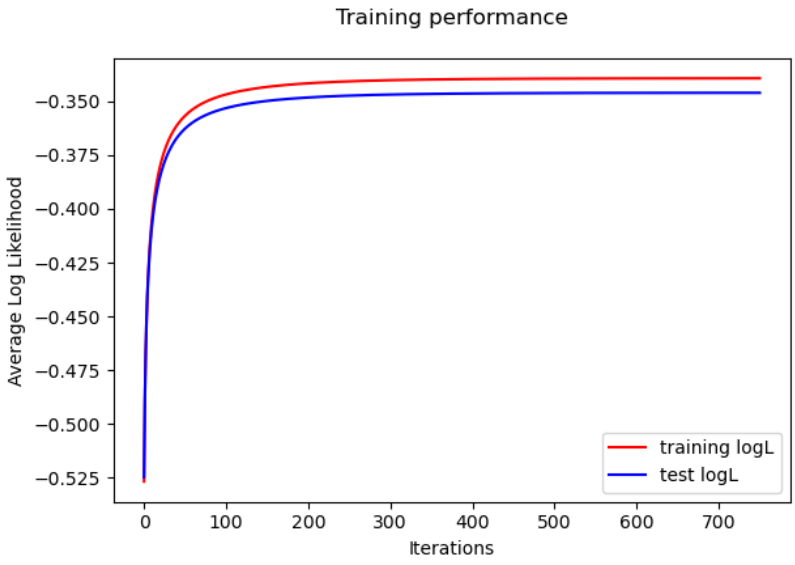
   (b) `compute_gradient(self, )`: In this method, you are going to compute and return the gradient of the average log-likelihood objective (computed based on the training set) with respect to the model weights `self.weight`.

(c) `gradient_ascent(self, )`: In this method, you will implement gradient ascent to train the parameters `self.weight` of the logistic regression model using the training set.

In addition, after every gradient ascent step of updating the model weights, you need to compute the prediction accuracy and average log-likelihood objective values for both the training and test sets based on the updated weights. You will store these values in `self.accuracy_train`, `self.accuracy_test`, `self.log_likelihood_train`, and `self.log_likelihood_test` correspondingly. Note that these variables are already initialized as empty lists in the `__init__()` function.

2. **(5 pts)** Method `preprocess_data(X_train, X_test)`: In this method, you will do some preprocessing on the datasets including: (i) encoding categorical features using the `OneHotEncoder` class; and (ii) normalizing numerical features using the `StandardScaler()` class. These two classes are imported from the `scikit-learn` library.

3. **(5 pts)** Method `plot_log_likelihood_performance(log_likelihood_train, log_likelihood_test)`: In this method, you are going to plot the average log-likelihood curves for the training and test sets during the training process (i.e., gradient ascent iterations). The input is the `(log_likelihood_train, log_likelihood_test)` lists that you computed in the method `gradient_ascent(self, )`. The plotted figure should look similar to the following figure:




4. **(5 pts)** Method `plot_accuracy_performance(accuracy_train, accuracy_test)`: In this method, you are going to plot the prediction accuracy for the training and test sets during the training process. The input is the `(accuracy_train, accuracy_test)` lists that you computed in the method `gradient_ascent(self, )`.

5. **(5 pts)** Method `plot_roc(X_train, y_train, X_test , y_test, learner)`: In this method, you will plot the ROC curve showing the performance of the trained logistic regression model `learner` on the input training and test sets.

## 4 Task 3: Write Report (10 points)

Finally, you will write a report on the results of your data analysis and model performance. Your report should include the following results:

1. The correlation heatmap plotted using your `plot_numeric_feature_correlation()` method. Write 2-3 sentences summarizing your observations regarding the correlations between numerical features as well as between these features and the *income* labels.

2. The histograms with respect to the *education* and *hours.per.weak* columns, plotted using your `plot_histogram(train_data)` method. Write 2-3 sentences summarizing your observations regarding these histograms.

3. The average log-likelihood figures using your `plot_log_likelihood_performance()` method and prediction accuracy figures using your `plot_accuracy_performance()` method when the learning rate is `learn_rate`= 0.05 and `learn_rate`= 0.75, respectively. Write 2-3 sentences summarizing your observations regarding these results.

4. The ROC figures using your `plot_roc()` methods when the learning rate is `learn_rate`= 0.05 and `learn_rate`= 0.75, respectively.

Save your report in a file named *proj1_report.pdf*.

## 5 Submission

You will need to submit the following files: (i) *proj1_data_analysis.ipynb*; (ii) *proj1_classification.ipynb*; and (iii) *proj1_report.pdf* on Canvas.