Assignment 2: Neural Network From Scratch

CS 403

Shaan Vaidya
150050004

Feature Engineering:
First, the data histograms were plotted for each column to see how the data is distributed. This showed that for the 'native.country' field, almost all of the records were for 'United-States'. So the native.country field was removed from consideration.

Also, education and education-num represent the exact same field. 'education-num' is just a numerically coded 'education' column. The values were also assigned in a way that higher value is for higher education. So, education was also dropped.

Encoding categorical fields:
One-hot encoding was used for categorical fields. This was done from scratch and not using the 'get dummies' function to include the possibility that some category that occurs in test data but not in train data.

Missing values:
In the one-hot encoding of '?' values, all the categories were set to zero as there is no info about the missing data.

Code:

- The 'clean data' function is for all preprocessing of the data, including normalisation.
- For test data, means and stds of train data were used.
- MyNeuralNetwork class has been created.
- Most functions have self explanatory names.
- For weights of neurons, matrices have been used.
- Mini Batch Gradient descent has been implemented
- The error function is the general squared difference error
- tanh has been used as the activation function
- Input layer is the 0^{th} layer of the network and output the last
- Two hidden layers have been used, but can be changed as required
- Extra files for 'means and stds' and 'biases' have also needed to be created.
- The weight and bias matrices have been converted to arrays and appended to a file for writing.

Comparison with other standard methods:

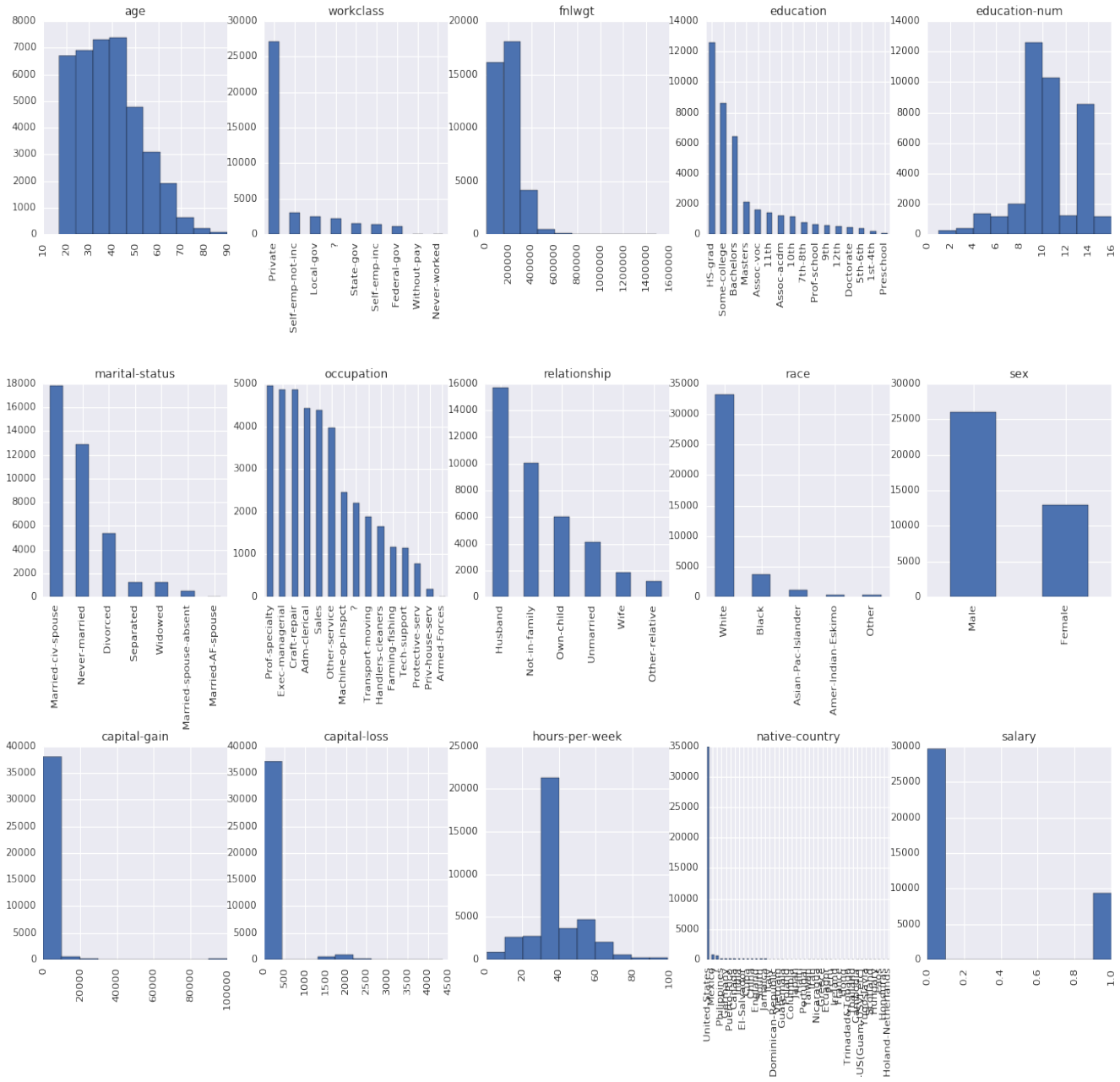
- The standard methods used were Logistic regression, Gaussian Naive Bayes and Random Forest Classifier.
- Neural net did better than all the three, RFC was the closest.
- Scores:
  - **Neural Net : 0.80455** (on the leaderboard, later run improved to 0.81 rank~41)
  - **LR : 0.77237**
  - **GNB : 0.74480**
  - **RFC : 0.78004**

Observations:

- Changing the activation function from sigmoid to tanh strangely improved the performance, probably just a random incident
- Used different sizes of hidden layers, worked better
- Increasing the number of hidden layers improved performance as expected

Plots:

Plotted the data and its frequency histograms



Code for plots:

```
fig = plt.figure(figsize=(20,15))
cols = 5
rows = int(float(data.shape[1]) / cols)
for i, column in enumerate(data.columns):
    ax = fig.add_subplot(rows, cols, i + 1)
    ax.set_title(column)
    if data.dtypes[column] == np.object:
        data[column].value_counts().plot(kind="bar", axes=ax)
    else:
        data[column].hist(axes=ax)
        plt.xticks(rotation="vertical")
plt.subplots_adjust(hspace=0.7, wspace=0.2)
plt.show()
```

Citation for matplotlib code:
https://www.valentinmihov.com/2015/04/17/adult-income-data-set/