# Neural Networks

# Overview

This project analyzes census data in an attempt to predict household earnings. The original dataset was downloaded from [1994 Census Data](https://archive.ics.uci.edu/ml/datasets/Census+Income) which contains data from the 1994 census. The dataset contains 48k instances with 14 variables. The data was placed into a public folder in DropBox and can be accessed [here](https://www.dropbox.com/s/qhwnpwn2jka2b3z/adult.data?dl=1)

The general approach to the analysis was:

* Clean up data by identifying and correcting missing values and outliers.
* Look for correlations and trends in the data
* Convert columns with object type to numeric type
* The goal was to run multiple Multi-layer Perceptron classifier (MLP) models to train neural networks to classify people as earning more or less than $50k/year

# **Dataset Key Points**

* There are 2399 rows that have 'missing' data (denoted by ? in a field value) that were dropped from the training data
* The following columns appear to have the highest correlation to income: age education-num relationship hours-per-week

# Inspect the Dataset

* Review the dataset to understand the overall structure and content of the data.
* Look at the columns and data types. Neural networks need numeric input to work so any non-numeric columns will need to be converted to numeric values in some manner.
* Look at the ranges of data to look for outliers or bad data.

# Prepare Training and Test Data

* Separate the input data into a set of independent variables (**X**) and the dependend variable (**Y**). **X** = the set of all columns except 'income'. **Y** = the set of 'income' values.
* All values will be scaled using the sklearn **StandardScaler** to ensure that features with large numeric values do not dominate the model.

# Create Neural Networks

Train neural networks to classify people as earning more or less than $50k/year.

# Results table for MLP models with Combinations of Hyperparameters

Multiple MLP models were created using the parameters Hidden Nodes, Activation Function, Learning Rate, Max Iterations. Accuracy reported for test and training datasets is the F1 accuracy score retrieved from the **metrics.classification\_report()** function.

| **Run #** | **Hidden Nodes** | **Activation Function** | **Learning Rate** | **Max Iterations** | **Training Accuracy** | **Test Accuracy** |
| --- | --- | --- | --- | --- | --- | --- |
| #1 | 4,5,4 | logistic | 0.0001 | 500 | 0.85 | 0.85 |
| #2 | 40,50,40,10 | tanh | 0.01 | 250 | 0.90 | 0.84 |
| #3 | 30,20,10 | relu | 0.01 | 1000 | 0.89 | 0.84 |
| #4 | 50,25 | relu | 0.001 | 500 | 0.93 | 0.82 |
| #5 | 3,4,2 | logistic | 1.1 | 2000 | 0.75 | 0. 75 |

# Conclusion

Some of our observations are below:

* This dataset can be used to predict income classified as > 50k or <= 50k with a reasonably high level of accuracy.
* The training accuracy tends to be higher than the testing accuracy.
* Higher node counts seem to produce slightly better accuracy with this data.

Our MLP models are achieving an average of 86.4% and 82% on training and test accuracy, respectively, across 5 runs with different parameters.