Machine Learning Adult Wage Prediction

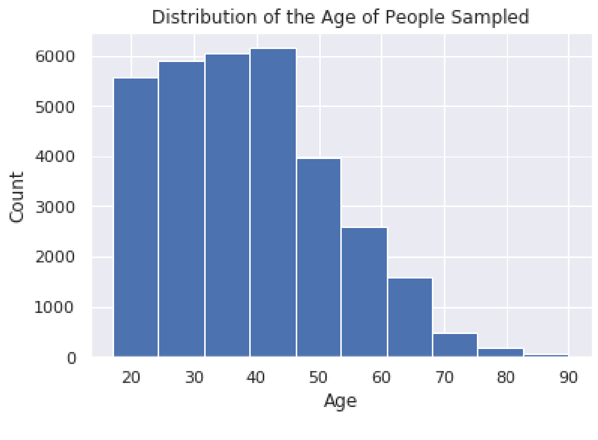
Objective:

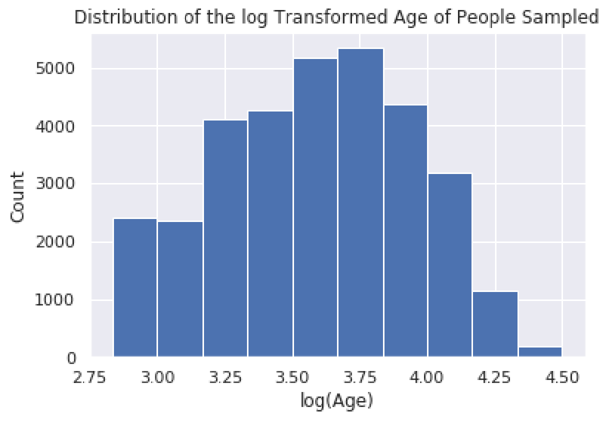
We worked with a popular machine learning (ML) data set called the adult wage data set (https://archive.ics.uci.edu/ml/data sets/Adult). The data set consists of the following features for each person: age, work class, weight of sample, education, number of years in education, marital status, occupation, relationship, race, sex, capital gain, capital loss, work hours, and native country. Based on these features, the challenge is to predict whether a person has a salary over US$ 50k. Our objective was to build and compare different models to determine the model with the highest validation score possible (without overfitting), and to explore different techniques which can improve the accuracy of these models.

Data Preprocessing:

1. **Removing unused features:**
    a. ‘fnlwgt’: sampling weight. We did not know the proper techniques to handle this feature.
    b. ‘education’: categorical data for education level. The information contained in this feature seemed to overlap with the ‘education-num’ feature (the number of years of school attended by a person).
    c. ‘native-country’: this categorical data was too unbalanced (the majority had the value “United-States”), and the information overlapped moderately with the ‘race’ feature.

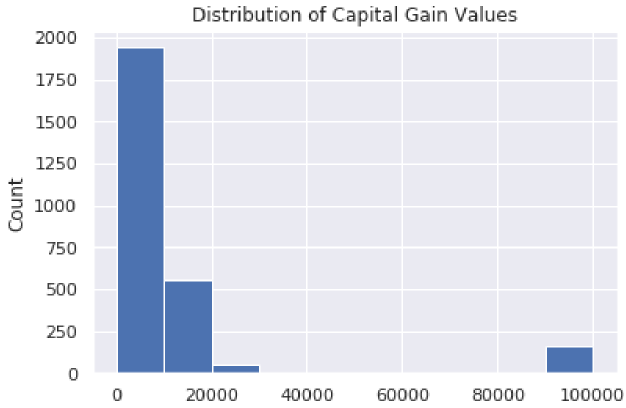
2. **Scaling numerical features:**
    a. ‘age’ was distributed with a bias to the left. Log transformation was applied to normalize the distribution. This would help some of the ML models, since the transformation evens out the Euclidean distances between data points within this feature, which makes it easier to find boundaries between the categories in Euclidean space. The accuracy of ML models improved slightly as a result.

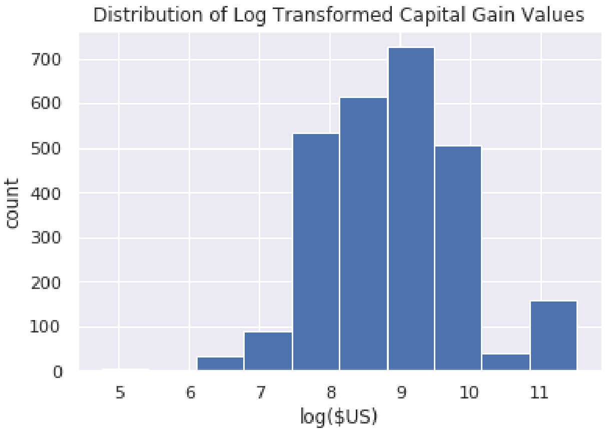




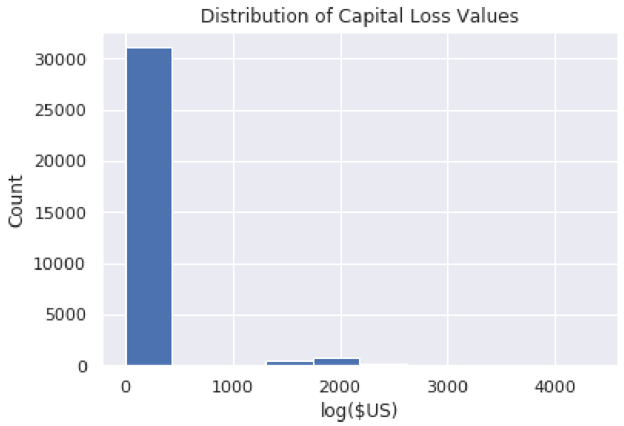
b. “education-num” values increment linearly. We thought that the number of years in education after graduating highschool should have more weight than the number of years prior, since a highschool diploma is usually needed to get a high-paying job. Hence, we multiplied by a factor of three all the “education-num” values greater than eight (which is the number of years it takes a person to graduate highschool in this data set). This transformation moderately improved the ML models’ prediction scores.

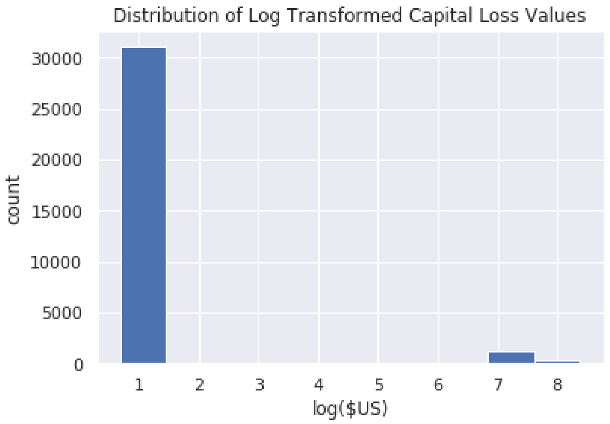
c. ‘capital-gain’ feature was distributed unevenly, largely due to most of the values being zero. Log transformation was applied to even out the distances between data points, similar to the transformation on ‘age’ feature. Note that values of zero were excluded in the histogram figures below to illustrate the point that log transformation more or less produced a normal distribution.





d. ‘capital-loss’ feature was more or less distributed evenly (excluding the zeroes), but log transformation still improved our models’ accuracies. We believe that this is due to the transformation widening the gap between those who had never lost money and those who had tried investing and lost money. This perhaps reflects the correlation between risk-taking and wage, or a correlation between wealthy financial background and wage (this logic may also apply to capital-gain feature). Unfortunately, we lack the resources to investigate further on this matter. Zero values were included in the following histograms.





3. **Creating dummy variables:** All the categorical features were transformed into dummy variables. This means that each unique item within the feature is transformed to its own column, where the value is set to either 0 or 1 corresponding to the feature's original value. For example, 'race' feature was divided into 'Amer-Indian-Eskimo', 'Asian-Pac-Islander', 'Black', 'White', and 'Other' features. For all the categorical features, small minorities were grouped together into one feature, since we wanted to avoid introducing sparsely populated features as much as possible; such features would not provide enough data for ML algorithms to train on, often resulting in overfitting.

4. **Undersampling:** There was an imbalance of data where the ratio of salary_under_50K to salary_over_50K was approximately 3:1. To even out the ratio, we sampled a fraction from salary_under_50K and concatenated it to salary_over_50K, forming a new balanced data set. From this data set, we extracted the features and responses for model training and validation.

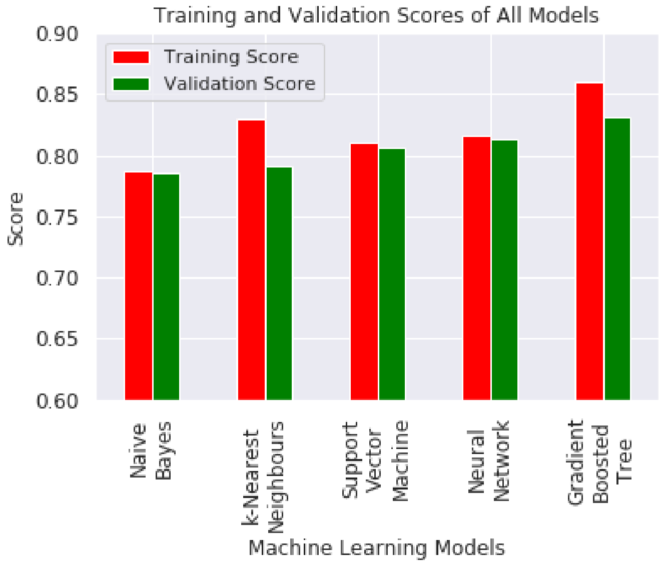
Training and Validating:

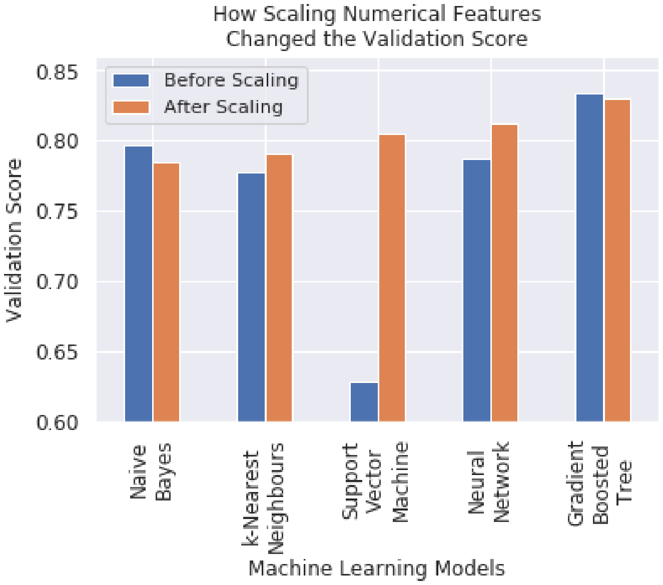
The adult wage data set was divided into a training set and a validation set: the former was used in training the models, and the latter was used to determine the accuracy of the models on data never before seen by the models. Five ML models were trained and validated:

1. Naive Bayes: Naive Bayes algorithm was run 50 times to get the average training and validation scores.
2. k-Nearst Neighbours: kNN algorithm was run 10 times to get the average training and validation score. In the pipeline, we applied MinMaxScaler to even out Euclidean distances, PCA to reduce the dimensions to 15 features, and KNeighborsClassifier with n_neighbors set to 8.

3. Support Vector Machine: SVM algorithm was run twice due to performance concerns. In the pipeline, we applied PCA to reduce the dimensions to 10 features, then used a radial basis function kernel to optimize the results.
4. Neural Network: MLPClassifier with two hidden layers (35 and 25 in size) was run 6 times. We set the activation function to ‘logistic’.
5. Gradient Boosted Tree: We used 30 estimators with a max depth of 15 with 30 minimum number of samples. This algorithm was run five times to get the average scores.

Results and Discussion:





We found that Naive Bayes model yielded the lowest score at around 0.78, and Gradient Boosted Tree model yielded the highest validation score at around 0.83 on a balanced data set. The results are summarized in the bar graph to the right.
The Gradient Boosted Tree model consistently produced the highest validation score while a variety of changes were made to the features, although the model suffered from large overfitting in comparison to other models. However, we believe the extent of overfitting by this model (difference of 0.03 between training and validation score) is within an acceptable range for practical purposes. Hence, we accept this model as the optimal model.
We also wanted to know whether our scaling techniques used on the numerical features had any effect on the models’ scores. The results from raw data versus scaled data are outlined in the bar graph to the right. The validation score of Support Vector Machine model improved greatly after scaling; Neural Network model improved slightly; the rest of the models were not significantly affected. These findings are surprising - we expected k-Nearest Neighbours model to improve the most with scaling due to scaling affecting its calculation of Euclidean distances, but this was not the case. Since Support Vector Machines also draw boundaries within the Euclidean space, it makes sense that scaling would affect its score. The

intricacies behind why our Support Vector Machine model improved so greatly is beyond the scope of our project. We are satisfied in knowing that scaling did result in some improvements in validation scores.

If time permitted, we could have researched more on oversampling. We used undersampling because it was simple to implement, but the cost was that the models had less data to train on. Oversampling would have avoided this problem, hence potentially producing better models.