December 13, 2020

Determining Wealth Using AI

Jesse McCarville-Schueths
CS 4300
University of Missouri-St. Louis

Introduction

Artificial Intelligence is a powerful tool to use for predictions. In a world were personal data is has value, it can be useful to determine the wealth of a person. This concept has potential for many applications from determining appropriate demographics for advertising to offering a fair starting salary to a college graduate. In this project, we will build a neural network for binary classification to determine whether a person makes over $50k a year. We will test a variety of structures of the neural network in order to determine which one will be appropriate for our task. We will accomplish this task using only 13 key features about a person. We will then build a more efficient neural network by determining which features negatively impact our accuracy. We will remove any unwanted features from our final model. This tool of artificial Intelligence is only a powerful tool when trained with appropriate data and in a appropriate model. By determining the best model for our data and removing unnecessary features, we will aim to be as accurate as possible with our final model.

Task: Salary Prediction
The prediction task is to determine whether a person makes over 50K a year.

About the Project Phases

I decided to switch datasets during phase II of the project. My original dataset was dealing with poker hands and was not happy with the heavily unbalanced results of the "poker hands" (due to basic statistics). I decided to switch to a dataset that dealt with determining the scores of math tests dependent on test prep, race, gender, parents education, etc. I had issues while running the neural network and would constantly get accuracy levels of around 1.3%. I tried manipulating the neural network (adding/removing layers/nodes, epochs, and changing the batch size). I spent too much time trying to fix my issue before I looked into the data and decided that it was going to require more manipulation before it would be ready for the neural network. I decided to switch my dataset again to this adult earnings dataset. This dataset is far more easier to work with since I am still a novice in this field of AI. Much of Phase I had to be reworked to be applied to this new dataset and to better understand the data.

Link to OverLeaf Project:
`https://www.overleaf.com/read/ykjkwwfzbmfs`
Link to Google Colab:
`https:`
`//colab.research.google.com/drive/1At35idERy-KKX6pu_AGMvb7UnQ2osCOy?usp=sharing`

The Dataset and Its Source

Each record is a person's atrribute listed below.

Attributes

1. Age: [continuous]
2. Workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
3. Final Weight (fnlwgt): [continuous]
4. Education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
5. Education-num: [continuous]
6. Marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
7. Relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
8. Race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
9. Sex: Female, Male
10. Capital-gain: [continuous]
11. Capital-loss: [continuous]
12. Hours-per-week: [continuous]
13. Native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad and Tobago, Peru, Hong, Holand-Netherlands
14. Earnings: greater than 50K, lesser than or equal to 50K

About

The dataset was sourced from UCI Machine Learning Repository. It was created by Ronny Kohavi and Barry Becker. Extraction was done by Barry Becker from the 1994 Census database.
Source: http://archive.ics.uci.edu/ml/datasets/Adult

Splitting the Data

Before splitting the data, the data was shuffled in order remove any predetermined "traits" in the order of the original dataset. The data was then split, since the dataset is large (n = 32,561), 30% of the data was used for training (n = 9768) the model and 70% of the data was used for validation (n= 22,793). To ensure the normalization of the training data didn't disturb the normalization of the validation data (and vice-versa), the data was normalized after the split.

Normalized Histograms

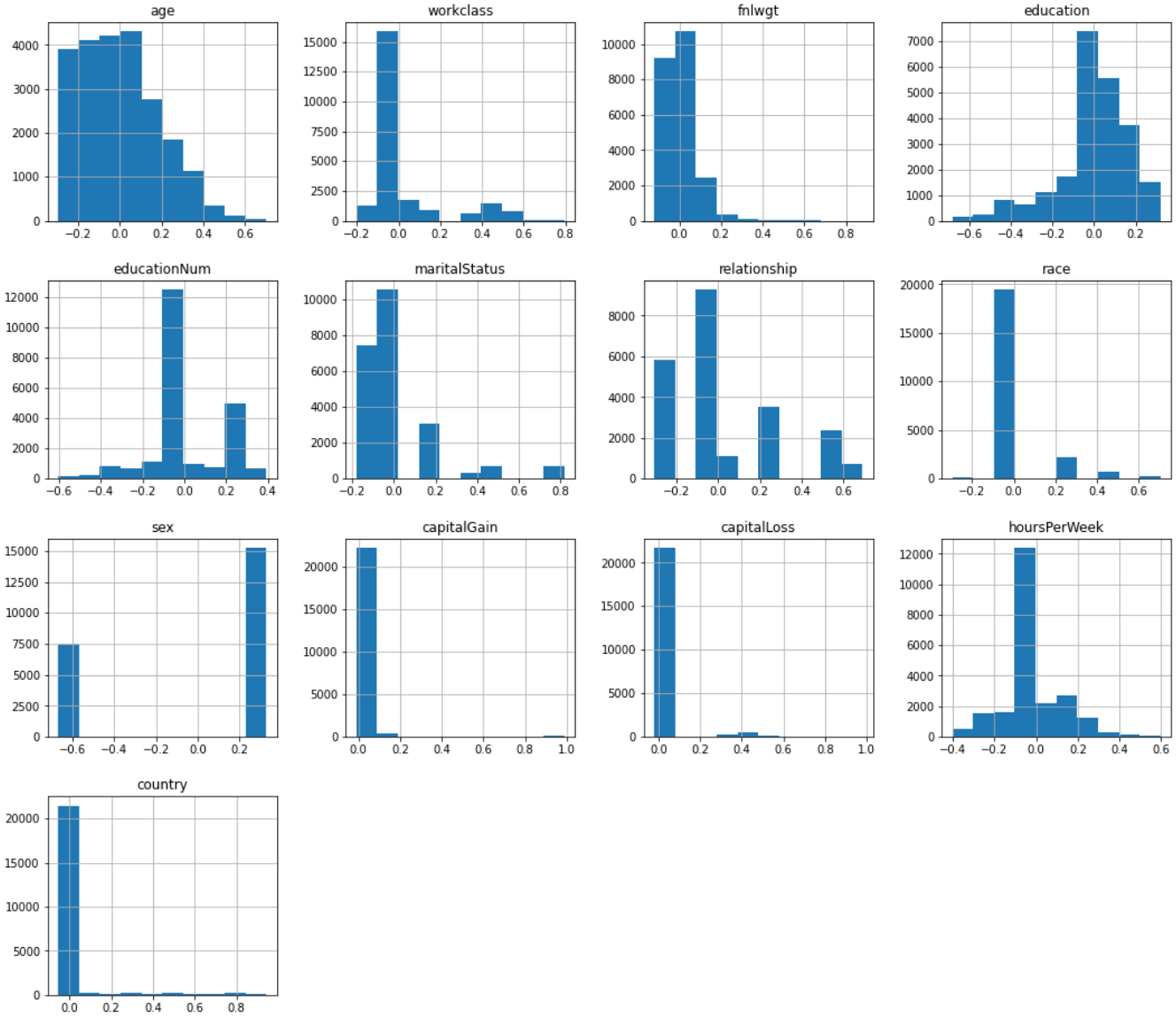

Figure 1: Histograms of Normalized Dataset

Options of Model Selections

All models were ran with 100 epochs and a batch size of 64. Since I am still a novice at this, I took the opportunity to run multiple alternative models to learn how each one effects the learning rates. Observing the Model Accuracy graph, it should be noted that the closer the line is to the top left hand corner, the more accurate the neural network is. Also it should be noted that in the Model Loss graph, the closer the line is to the lower right hand corner the less loss there is.

Base Model

This base model has a basic architecture of a single input and a single output layer.

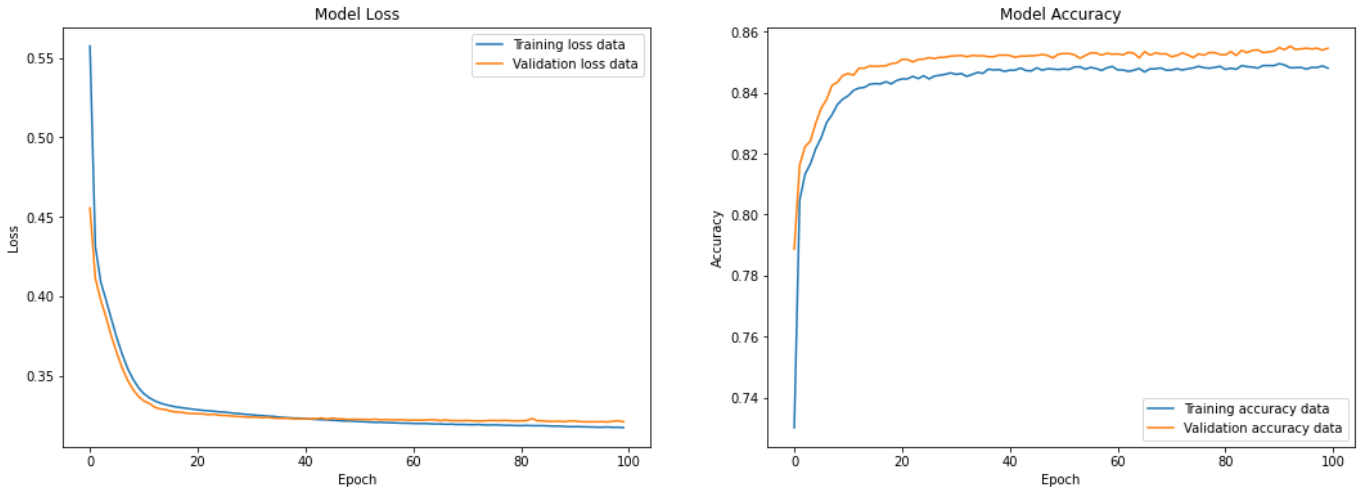

Figure 2: Learning Curve of Base Model

Model with Middle Layer

This model has an architecture of an input layer, a hidden layer, and an output layer.

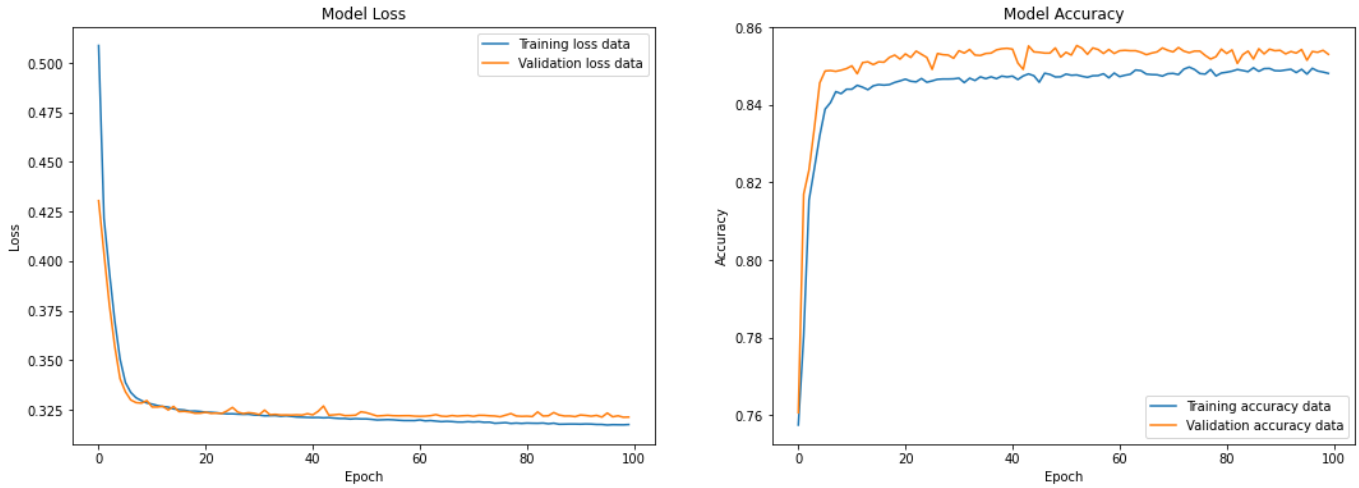

Figure 3: Learning Curve of Model with Hidden Layer

Model with 4 Layers

This model has an architecture of an input layer, two hidden layers, and an output layer.

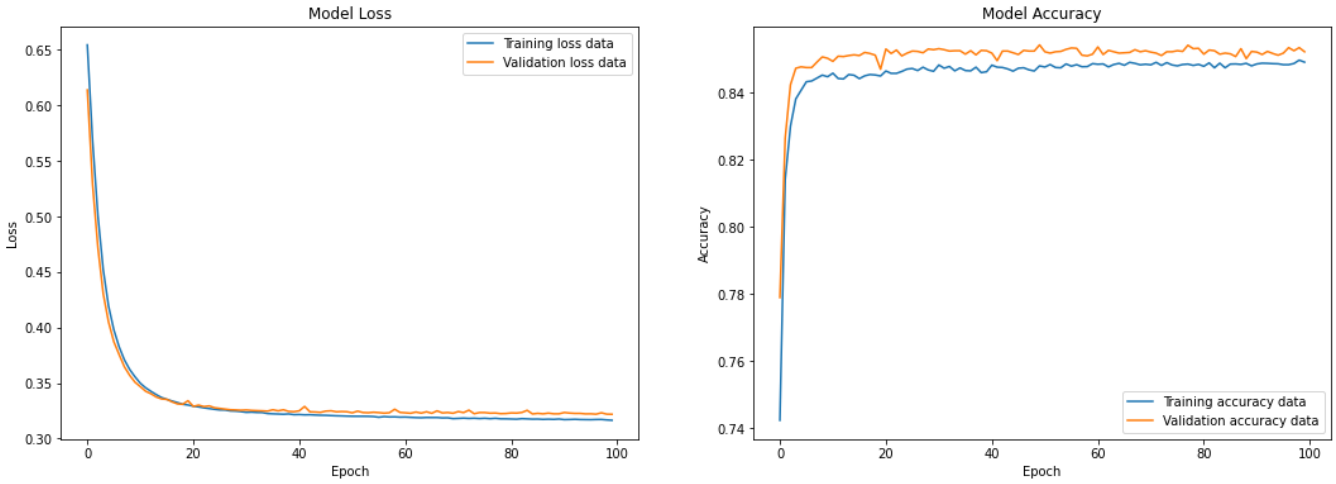

Figure 4: Learning Curve of Model with 2 Hidden Layers

Model with 6 Layers

This model has an architecture of an input layer, four hidden layers, and an output layer.

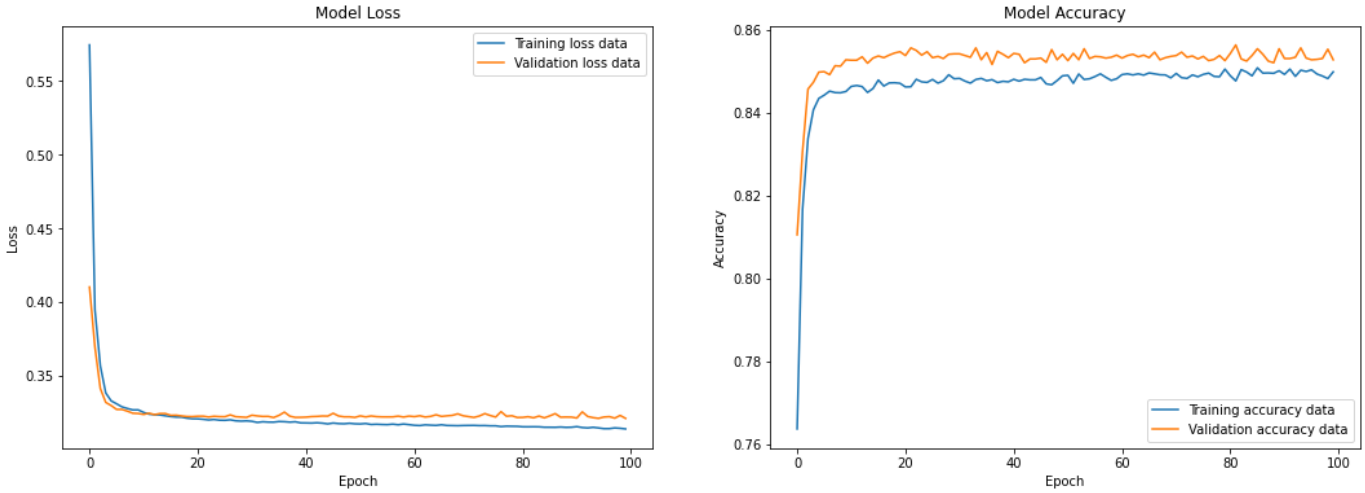

Figure 5: Learning Curve of Model with Hidden Layer

Model with Linear Activation as Last Layer

This model has an architecture of an input layer, a hidden layer, and an output layer with linear activation.

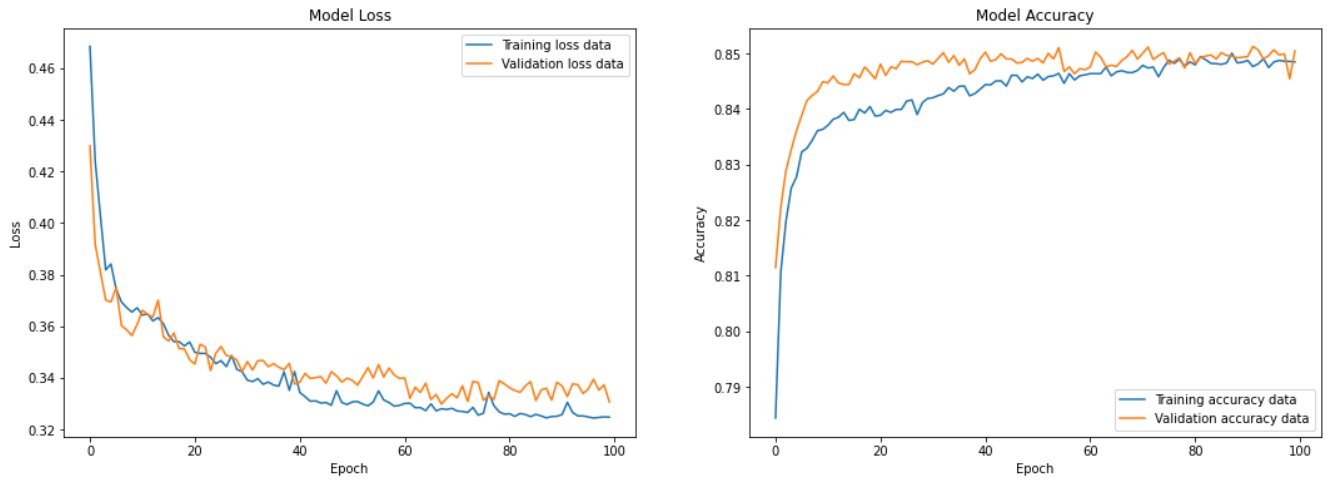

Figure 6: Learning Curve of Model with Linear Activation as Last Layer

Model with Linear Activation on All Layers

This model has an architecture of an input layer, a hidden, and an output layer with linear activation on all layers.

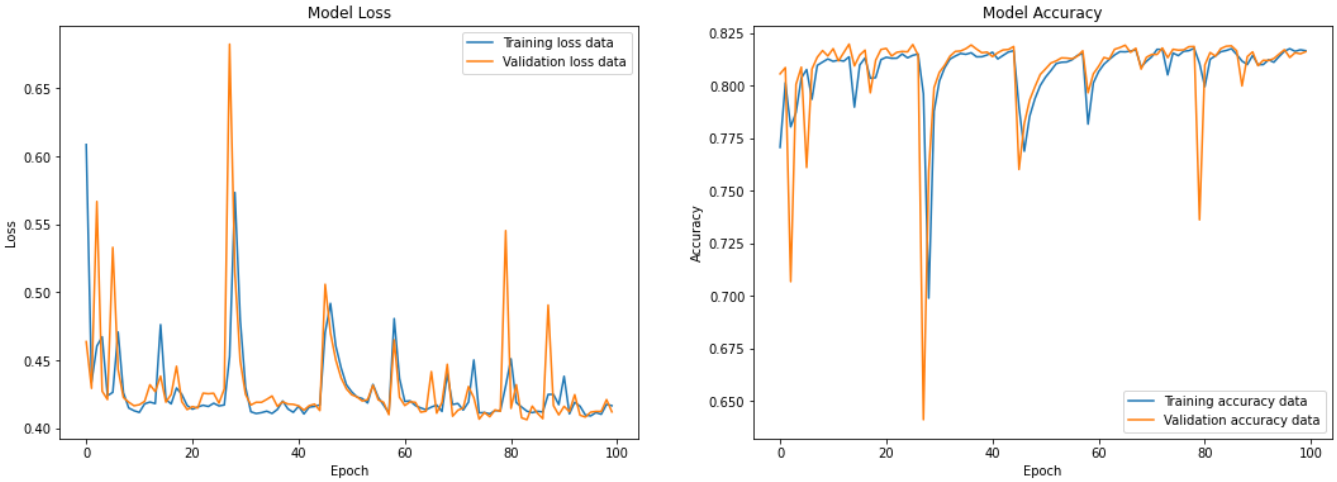

Figure 7: Learning Curve of Model with Linear Activation on All Layers

Model with Sigmoid Activation on All Layers

This model has an architecture of an input layer, a hidden, and an output layer with sigmoid activation on all layers.

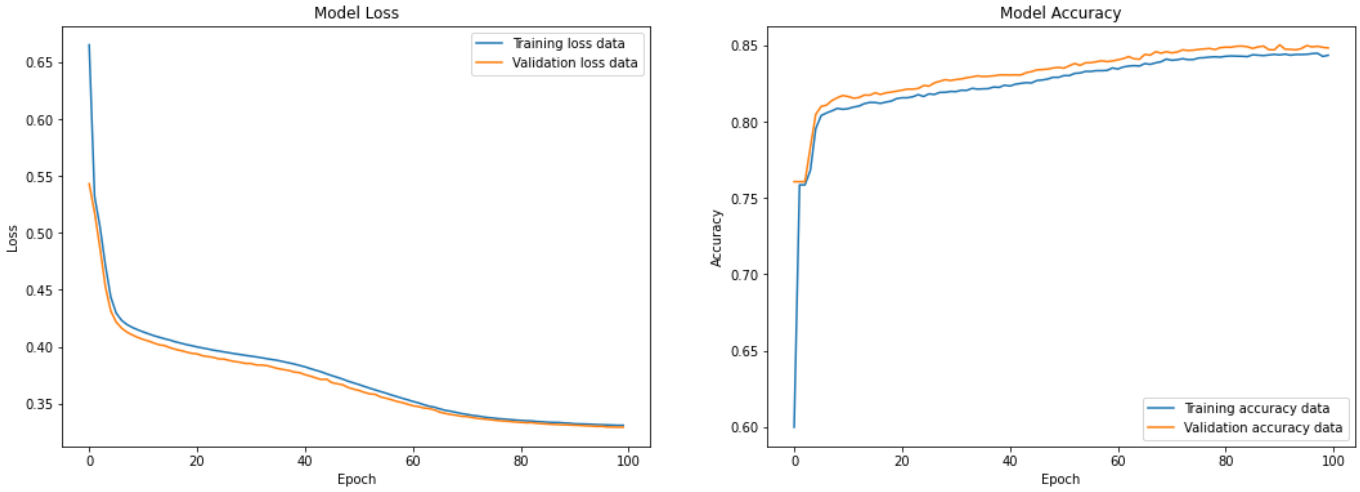

Figure 8: Learning Curve of Model with Sigmoid Activation on All Layers

Model with Overfitting

This model has an architecture of an input layer, a hidden, and an output layer with overfitting by a factor of 10 on the first and hidden layer.

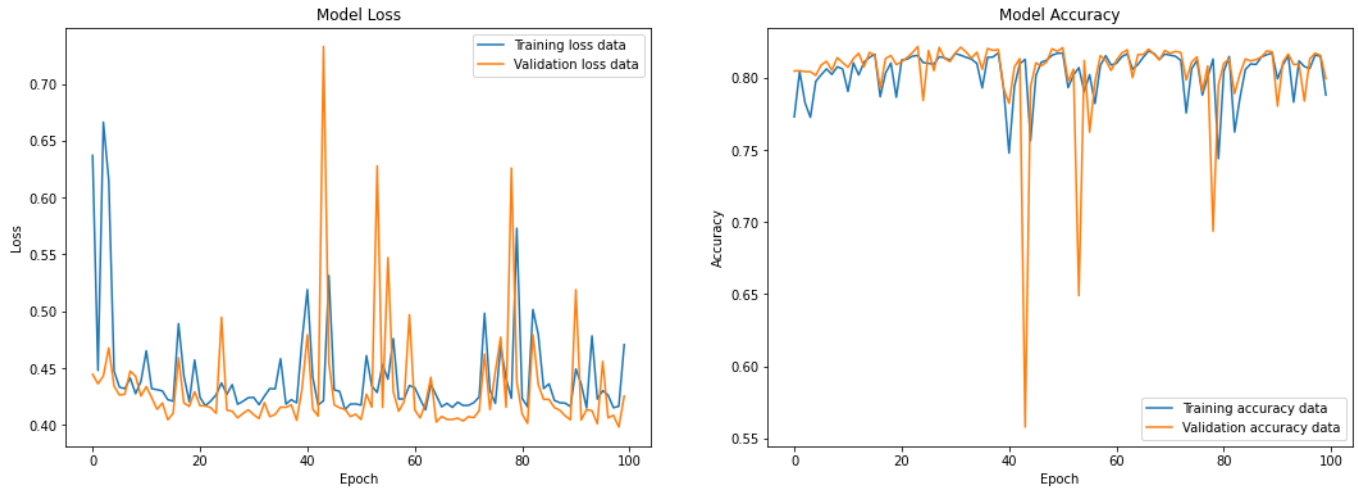

Figure 9: Learning Curve of Model with Overfitting

Discussion of Model Selection

When choosing an appropriate model, the different models had some differences. The over-fitted model lacks accuracy. This could be because the model contains extra capacity and random noise within the model can begin amplifying itself causing bad predictions to be passed along the nodes causing incorrect outputs. This would also explain why there is a higher loss percentage. Looking into the to Linear models, it can be observed that their loss it greater than the previous models that use relu activation. It should be noted that best accuracy was achieved when using a combination of relu and sigmoid activation functions along with multiple layers and multiple nodes (keeping in mind that more layers and node may lead to unwanted noise). Future testing will develop a better neural network model that will incorporate the correct amount of layers and nodes.

Model	Accuracy	Loss
Base	84.89%	31.68%
Single Hidden	84.88%	31.65%
Two Hidden	84.95%	31.53%
Four Hidden	85.03%	31.26%
Linear Last	84.95%	32.37%
Linear All	81.70%	41.11%
Sigmoid All	84.45%	33.04%
OverFitted	79.94%	43.12%

Figure 10: Summary table of Accuracy and Loss

After running multiple models, observing the Model Accuracy graphs and observing the Accuracy and Loss Table (above) from early test models, we can decide on an appropriate model to use for future iterative feature removal and selection. We will be using a four layer model. This model has an architecture of an input layer, two hidden layers, and an output layer. Relu activation will be used for the first three layers and sigmoid activation will be used for the output layer. This model will be using check-pointing and early stopping. It should be noted that the closer the line is to the top left hand corner, the more accurate the neural network is. Also it should be note that in the Model Loss graphs, the closer the line is to the lower right hand corner the less loss there is. This should be kept in mind for future comparisons. The approximate accuracy that one might expect is %85.01. It should be noted that this percentage is a bit higher than the percentage found in the table from figure 10. This is because the model was ran again with model check-pointing and early stopping. This accuracy is still similar to the previous accuracy of %85.03 percent. It is expected to receive slightly different results when running the same model multiple times. This model will be a baseline for our final results.

Feature importance

As of now, we have discovered a model that can do reasonably well on our validation set. We now have a good idea of the network architecture needed, the number of epochs needed, and how to perform model check-pointing and early stopping. To determine the important features of the model, all 13 inputs were run through a simple model using sigmoid activation individually. The accuracy results from each run were charted for comparison.

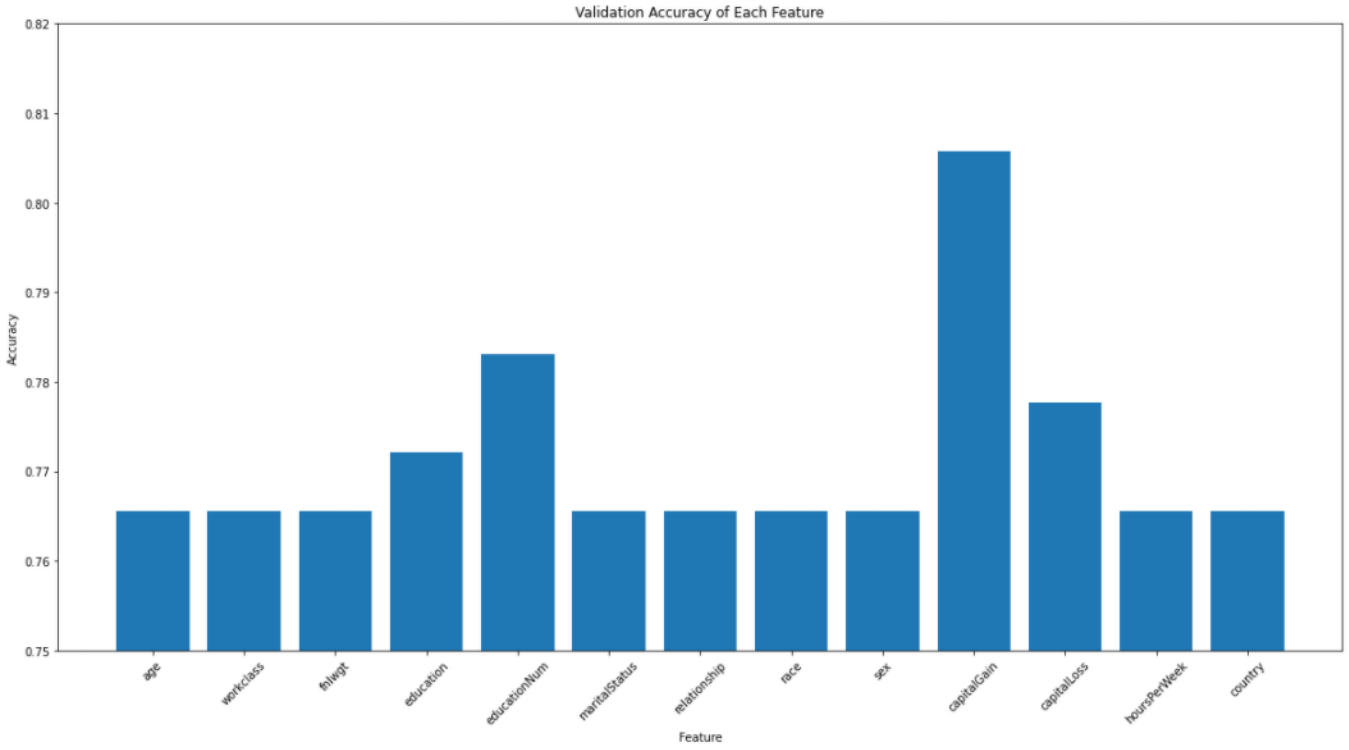

Figure 11: Bar Chart of Importance of Each Individual Feature

As we can see, our most important features are 'education', 'educationNum', 'capitalGain', and 'capitalLoss'. It would expected that removing these features may result in large dips to our accuracy results in the next test (feature removal).

Feature Removal

In order to determine the impact of each feature has on our accuracy. We will implement a simple Recursive Feature Elimination (RFE) technique to remove redundant or insignificant input features. After sorting our feature by importance. We will run a simple single layer model using sigmoid activation repeatedly. Each iteration we will remove a single feature and record the accuracy of the result. We will then chart our results to determine if any features should be removed. Please note that these accuracy's are from simple models with no hidden layers.

Removed Feature	Accuracy
age	81.96%
workclass	81.67%
sex	81.52%
race	81.51%
relationship	81.48%
fnlwgt	81.39%
maritalStatus	81.29%
hoursPerWeek	81.24%
country	81.07%
capitalLoss	80.89%
education	80.86%
educationNum	80.49%
capitalGain	80.22%

Figure 12: Table of Feature Removal

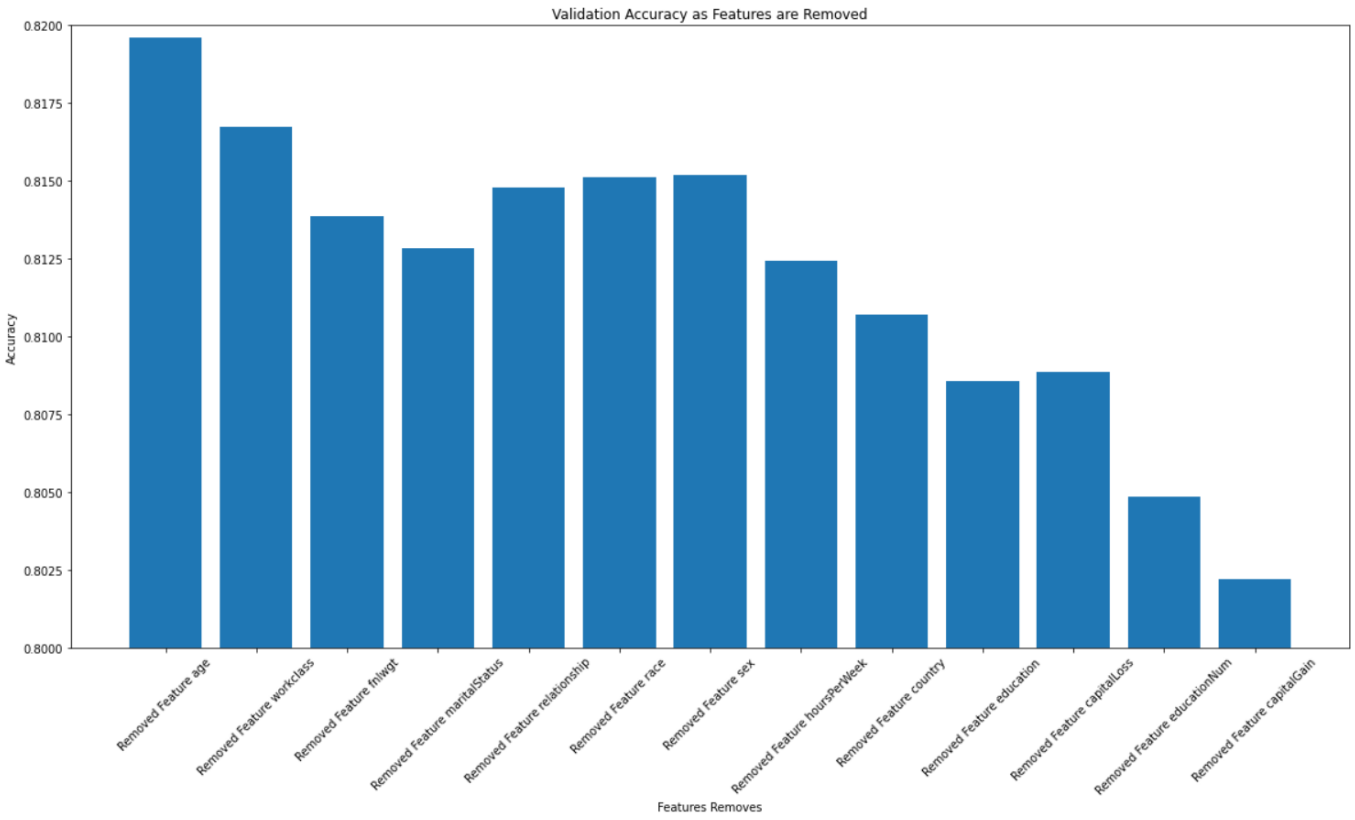

Figure 13: Bar Chart of Feature Removal

After observing the bar chart from the previous page. We can now see which features did not impact our accuracy significantly and which ones impacted our accuracy negatively. There appears to be no to little change in accuracy when removing 'race', 'sex', and 'capital loss'. It is surprising to find that capital loss did not impact our accuracy score as expected because it was found to be a important feature in our previous test. On the other hand, it was convenient to find that removing 'relationship' increased our accuracy. At this point, we will run two more models. One model will have 'race', 'sex', and 'relationship' removed and the other model will have just 'relationship' removed. Both these models will contain 2 hidden layers in preparation to compare to our original model with features removed.

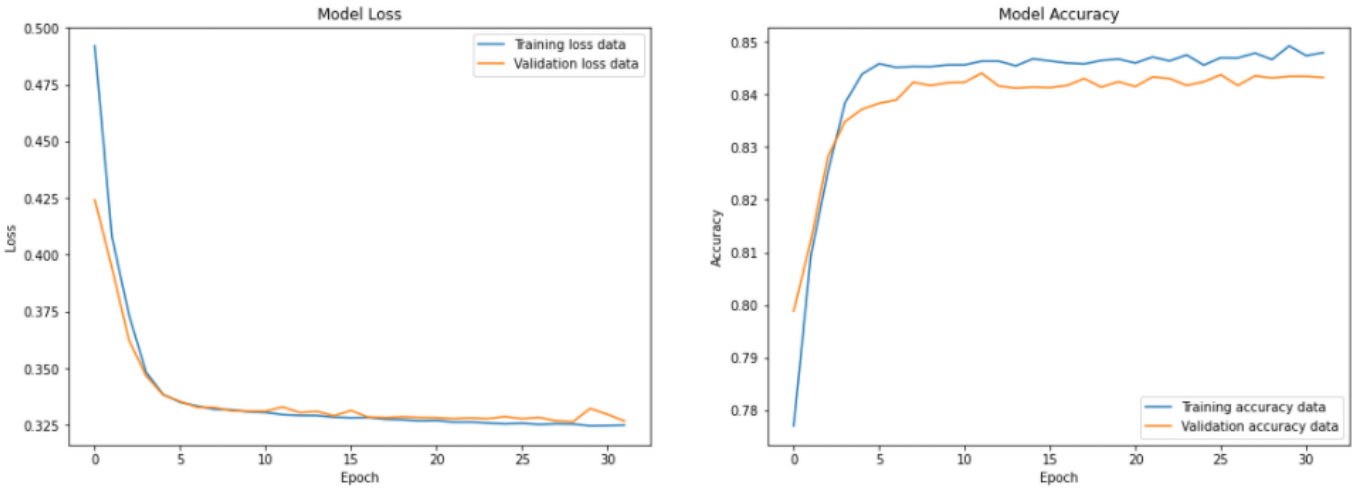

Figure 14: Learning Curves of 3 Features Removed

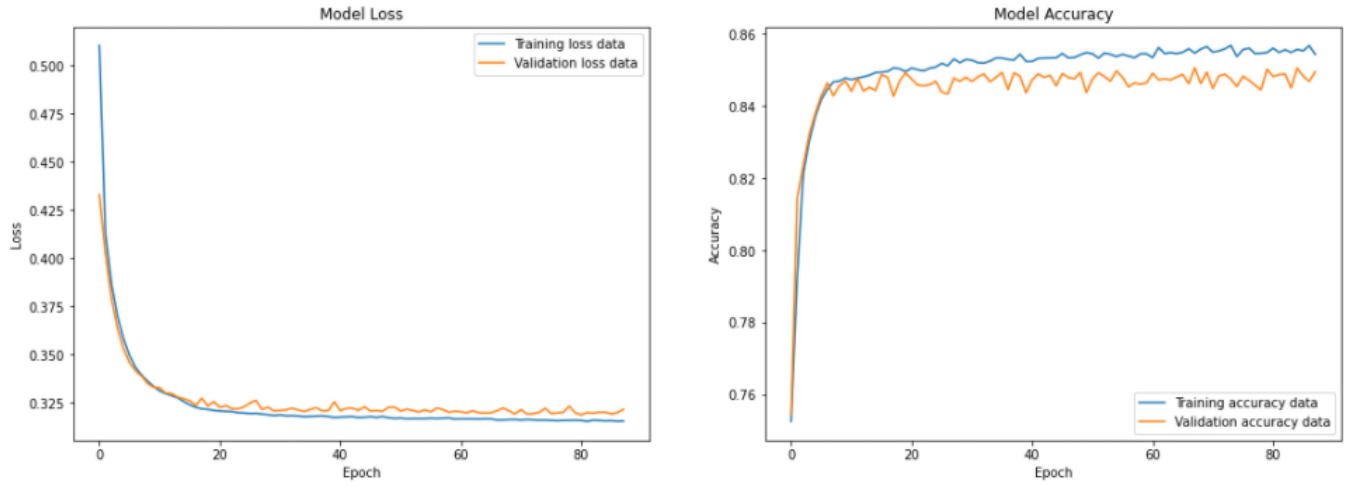

Figure 15: Learning Curves of Relationship Feature Removed

The accuracy of the model with relationship' removed was 85.41%. The accuracy of the model with relationship', 'race', and 'sex' removed was 84.82%. Clearly we have found our best model. Below, we compare our final model with 'relationship' removed against our original model.

Final Model and Conclusion

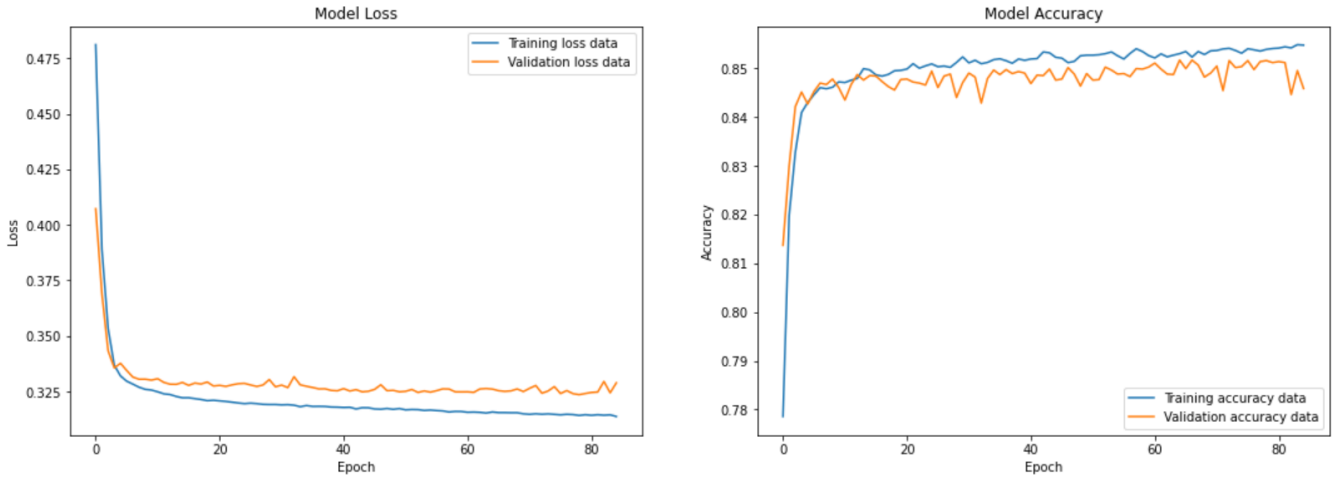

Figure 16: Learning Curves of Model with No Features Removed

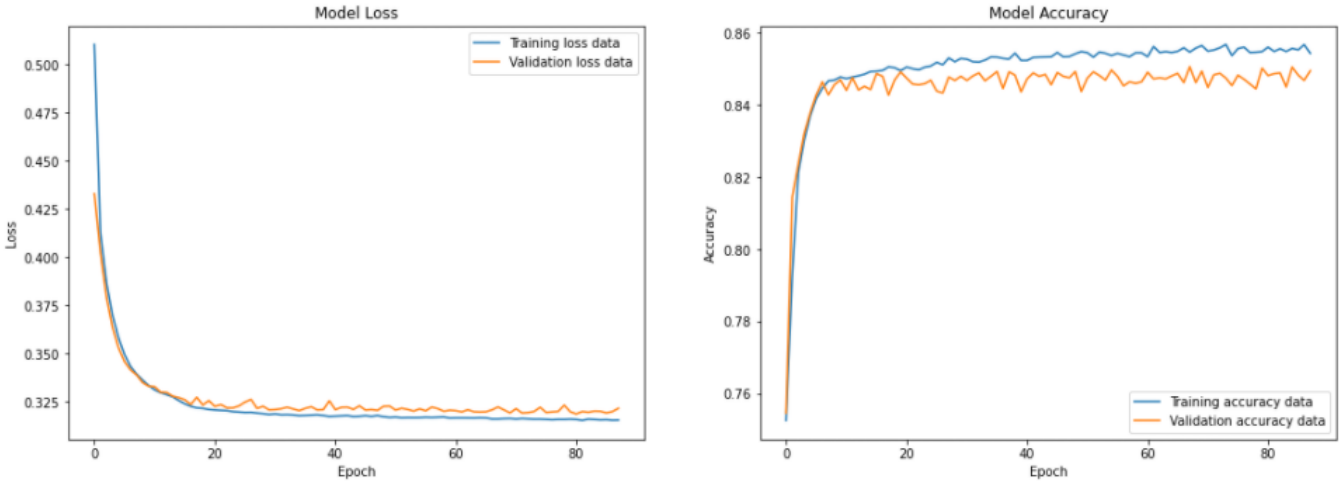

Figure 17: Learning Curves of Relationship Feature Removed Final Model

In conclusion, the learning curves of both the the model with 'relationship' removed and our original model look very similar. It is important to note that training and validation date graph are closer together in the model loss of the 'relationship' removed model. This is a good sign that we have created a better model. Our final accuracy has also improved from 85.01% to 85.41%. In this project, we built a neural network for binary classification to determine whether a person makes over $50k a year. We will tested a variety of neural networks and determined the appropriate one for our task. We accomplished this task using only 12 of the original 13 key features about a person. We built a more efficient neural network by determining the 'relationship' feature negatively impacted our accuracy. Trained with appropriate data and in a appropriate model, we determined the best model for our data and achieved an %85.41 accuracy with our final model.