Choose Your Own Capstone Project

Megan Lambert

12/19/2019

Executive Summary

Introduction

The goal of this project is to choose our own publicly available dataset to explore and apply machine learning techniques. Many businesses and organizations use machine learning to predict certain variables about their customers using other known variables. These predictions can be used to gain insights into consumer behavior to help make business decisions.

For this project, I chose the Adult Census Income data from the University of California at Irvine's Machine Learning Repository. Ronny Kohavi and Barry Becker extracted the data from the 1994 Census Bureau database. In this dataset, each row represents one person and includes several other variables. We will use a combination of variables to predict whether the income of that user is less than or equal to $50k or greater than $50k.

Goal

The objective of this project is to train a machine learning algorithm to predict whether a person's income is less than or equal to $50k or greater than $50k based on other variables in the Adult Census Income dataset. Our algorithm will be evaluated based on accuracy of our predictions to the validation set. We will explore various combinations of predictors and different models to improve accuracy. Our goal is to find the model that gives us the highest accuracy.

Dataset Description

We are using the public Adult Census Income dataset from the UCI Machine Learning Repository which can be found here: https://www.kaggle.com/uciml/adult-census-income

The dataset has 32,561 rows and each one represents an individual person from the 1994 Census. This data is limited to individuals between the ages of 17 and 99 years old.

Our dataset includes 15 total variables, 9 are categorical and 6 are continuous:

Continuous Variables

age

fnlwgt

capital.loss

capital.gain

education.num

hours.per.week

Categorical Variables

sex

race

income

workclass

education

occupation

relationship

marital.status

native.country

We will describe the dataset further when we go over our data exploration and visualization steps in the Methods and Analysis section.

Summary of Steps Taken

Downloading the Adult Census Income Dataset and Installing r Packages

Our first step is to download UCI's Adult Census Income dataset from my github repository and install any packages we may need.

Data Exploration and Preprocessing

Next, we explore the structure and various features of the dataset to help determine which variables will be good predictors for our model. We perform cleaning/preprocessing steps to remove rows with missing values and remove two variables we believe will not be good predictors.

Create Train and Validation Sets

We will split the data into train and validation sets once. Then, we will split the training set once more so that we can test various models before testing our final model on our validation set.

Data Visualization

We explore and plot variables from our dataset so we can see which will make good predictors for our model.

Comparing Different Models

We train various types of machine learning models on our training dataset. We create a results table that helps us compare the overall accuracy of each model we try. Our goal is to choose the final model that gives us the highest accuracy in predicting income.

Testing the Final Model on our Validation Dataset

Once we identify our final model, we use our validation dataset to test our predictions and compare overall accuracy.

Methods and Analysis

Downloading the Dataset

The file can be downloaded from github at:

https://raw.githubusercontent.com/meganlambert/HarvardX__Capstone2/master/adult.csv

```
#Install packages we may need
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(matrixStats)) install.packages("matrixStats", repos = "http://cran.us.r-project.org")
if(!require(gbm)) install.packages("gbm", repos = "http://cran.us.r-project.org")
```

```
#Download the Adult Census Income file
census <- read.csv("https://raw.githubusercontent.com/meganlambert/HarvardX_Capstone2/master/adult.csv"
```

Data Exploration and Preprocessing

Now we need to explore the Adult Census Income dataset so we can help determine which variables might be good predictors for our model. We will also need to look for indicators that cleaning steps should be taken.

```
#Examine the first rows of the census dataset with headers
head(census)
```

```
##   age workclass fnlwgt    education education.num marital.status
## 1  90         ?  77053      HS-grad             9        Widowed
## 2  82   Private 132870      HS-grad             9        Widowed
## 3  66         ? 186061 Some-college            10        Widowed
## 4  54   Private 140359      7th-8th             4       Divorced
## 5  41   Private 264663 Some-college            10      Separated
## 6  34   Private 216864      HS-grad             9       Divorced
##          occupation  relationship  race    sex capital.gain capital.loss
## 1                 ? Not-in-family White Female            0         4356
## 2   Exec-managerial Not-in-family White Female            0         4356
## 3                 ?     Unmarried Black Female            0         4356
## 4 Machine-op-inspct     Unmarried White Female            0         3900
## 5    Prof-specialty     Own-child White Female            0         3900
## 6     Other-service     Unmarried White Female            0         3770
##   hours.per.week native.country income
## 1             40  United-States  <=50K
## 2             18  United-States  <=50K
## 3             40  United-States  <=50K
## 4             40  United-States  <=50K
## 5             40  United-States  <=50K
## 6             45  United-States  <=50K
```

Here we can see that the data has 32,561 observations of 15 variables.

```r
#Examine the structure of the dataset
str(census)
```

```
## 'data.frame':    32561 obs. of  15 variables:
##  $ age           : int  90 82 66 54 41 34 38 74 68 41 ...
##  $ workclass     : Factor w/ 9 levels "?","Federal-gov",..: 1 5 1 5 5 5 5 8 2 5 ...
##  $ fnlwgt        : int  77053 132870 186061 140359 264663 216864 150601 88638 422013 70037 ...
##  $ education     : Factor w/ 16 levels "10th","11th",..: 12 12 16 6 16 12 1 11 12 16 ...
##  $ education.num : int  9 9 10 4 10 9 6 16 9 10 ...
##  $ marital.status: Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 7 7 7 1 6 1 6 5 1 5 ...
##  $ occupation    : Factor w/ 15 levels "?","Adm-clerical",..: 1 5 1 8 11 9 2 11 11 4 ...
##  $ relationship  : Factor w/ 6 levels "Husband","Not-in-family",..: 2 2 5 5 4 5 5 3 2 5 ...
##  $ race          : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 5 5 3 5 5 5 5 5 5 5 ...
##  $ sex           : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 1 2 1 1 2 ...
##  $ capital.gain  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : int  4356 4356 4356 3900 3900 3770 3770 3683 3683 3004 ...
##  $ hours.per.week: int  40 18 40 40 40 45 40 20 40 60 ...
##  $ native.country: Factor w/ 42 levels "?","Cambodia",..: 40 40 40 40 40 40 40 40 40 1 ...
##  $ income        : Factor w/ 2 levels "<=50K",">50K": 1 1 1 1 1 1 1 2 1 2 ...
```

We can also see from the structure that 3 variables have missing values expressed as "?". These variables are: workclass, occupation, and native.country.

Data Cleaning

We will remove the 2399 rows that have "?" missing values so that our models run smoothly.

```r
#Remove rows with missing values
clean_census <- filter(census,
              !workclass == "?",
              !occupation == "?",
              !native.country == "?")
clean_census <- droplevels(clean_census)
```

We can view the structure of our cleaned dataset and see that we have 30,162 observations left.

```r
#Examine the structure of the dataset to confirm missing values are removed.
str(clean_census)
```

```
## 'data.frame':    30162 obs. of  15 variables:
##  $ age           : int  82 54 41 34 38 74 68 45 38 52 ...
##  $ workclass     : Factor w/ 7 levels "Federal-gov",..: 3 3 3 3 3 6 1 3 5 3 ...
##  $ fnlwgt        : int  132870 140359 264663 216864 150601 88638 422013 172274 164526 129177 ...
##  $ education     : Factor w/ 16 levels "10th","11th",..: 12 6 16 12 1 11 12 11 15 10 ...
##  $ education.num : int  9 4 10 9 6 16 9 16 15 13 ...
##  $ marital.status: Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 7 1 6 1 6 5 1 1 5 7 ...
##  $ occupation    : Factor w/ 14 levels "Adm-clerical",..: 4 7 10 8 1 10 10 10 10 8 ...
##  $ relationship  : Factor w/ 6 levels "Husband","Not-in-family",..: 2 5 4 5 5 3 2 5 2 2 ...
##  $ race          : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 5 5 5 5 5 5 5 3 5 5 ...
##  $ sex           : Factor w/ 2 levels "Female","Male": 1 1 1 1 2 1 1 1 2 1 ...
##  $ capital.gain  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ capital.loss  : int  4356 3900 3900 3770 3770 3683 3683 3004 2824 2824 ...
##  $ hours.per.week: int  18 40 40 45 40 20 40 35 45 20 ...
##  $ native.country: Factor w/ 41 levels "Cambodia","Canada",..: 39 39 39 39 39 39 39 39 39 39 ...
##  $ income        : Factor w/ 2 levels "<=50K",">50K": 1 1 1 1 1 2 1 2 2 2 ...
```

```
#View the summary statistics of the dataset.
summary(clean_census)
```

```
##       age                   workclass         fnlwgt       
##  Min.   :17.00   Federal-gov     :  943   Min.   :  13769  
##  1st Qu.:28.00   Local-gov       : 2067   1st Qu.: 117627  
##  Median :37.00   Private         :22286   Median : 178425  
##  Mean   :38.44   Self-emp-inc    : 1074   Mean   : 189794  
##  3rd Qu.:47.00   Self-emp-not-inc: 2499   3rd Qu.: 237629  
##  Max.   :90.00   State-gov       : 1279   Max.   :1484705  
##                  Without-pay     :   14                    
##         education    education.num                  marital.status 
##  HS-grad     :9840   Min.   : 1.00   Divorced             : 4214  
##  Some-college:6678   1st Qu.: 9.00   Married-AF-spouse    :   21  
##  Bachelors   :5044   Median :10.00   Married-civ-spouse   :14065  
##  Masters     :1627   Mean   :10.12   Married-spouse-absent:  370  
##  Assoc-voc   :1307   3rd Qu.:13.00   Never-married        : 9726  
##  11th        :1048   Max.   :16.00   Separated            :  939  
##  (Other)     :4618                   Widowed              :  827  
##            occupation           relationship                   race      
##  Prof-specialty :4038   Husband       :12463   Amer-Indian-Eskimo:  286  
##  Craft-repair   :4030   Not-in-family : 7726   Asian-Pac-Islander:  895  
##  Exec-managerial:3992   Other-relative:  889   Black             : 2817  
##  Adm-clerical   :3721   Own-child     : 4466   Other             :  231  
##  Sales          :3584   Unmarried     : 3212   White             :25933  
##  Other-service  :3212   Wife          : 1406                            
##  (Other)        :7585                                                   
##      sex          capital.gain    capital.loss     hours.per.week 
##  Female: 9782   Min.   :    0   Min.   :   0.00   Min.   : 1.00  
##  Male  :20380   1st Qu.:    0   1st Qu.:   0.00   1st Qu.:40.00  
##                 Median :    0   Median :   0.00   Median :40.00  
##                 Mean   : 1092   Mean   :  88.37   Mean   :40.93  
##                 3rd Qu.:    0   3rd Qu.:   0.00   3rd Qu.:45.00  
##                 Max.   :99999   Max.   :4356.00   Max.   :99.00  
##                                                                  
##        native.country    income     
##  United-States:27504   <=50K:22654  
##  Mexico       :  610   >50K : 7508  
##  Philippines  :  188                
##  Germany      :  128                
##  Puerto-Rico  :  109                
##  Canada       :  107                
##  (Other)      : 1516                
```

Here we can see that approximately 75.11% of our dataset has an income of less than or equal to $50k.

Removing Variables

We need to remove variables that will not be good predictors for our models. The fnlwgt variable is an estimated measure of the units of population that are representative of the observation. This will not be a relevant predictor for income so we will remove it.

We will also remove the education variable because we already have education.num, which is a numerical version of education that we can use as a predictor.

```
#Remove education and fnlwgt variables.
clean_census <- clean_census %>% select(-c(education,fnlwgt))
```

Create Train and Validation Sets

We will now split the dataset into train and validation sets.

```
#Splitting the dataset for validation and training.
set.seed(1,sample.kind = "Rounding")  #if using R3.5 or earlier set.seed(1)
test_index <- createDataPartition(clean_census$income, times = 1, p = 0.2, list = FALSE)
census_validation <- clean_census[test_index, ]
census_training <- clean_census[-test_index, ]
```

We will split the census_training dataset once more so we can use it to test out various models before choosing a final model.

```
#Splitting the census_training dataset again for testing
set.seed(10,sample.kind = "Rounding")  #if using R3.5 or earlier set.seed(10)
test_indexsplit <- createDataPartition(census_training$income, times = 1, p = 0.2, list = FALSE)
testing <- census_training[test_indexsplit, ]
training <- census_training[-test_indexsplit, ]
```

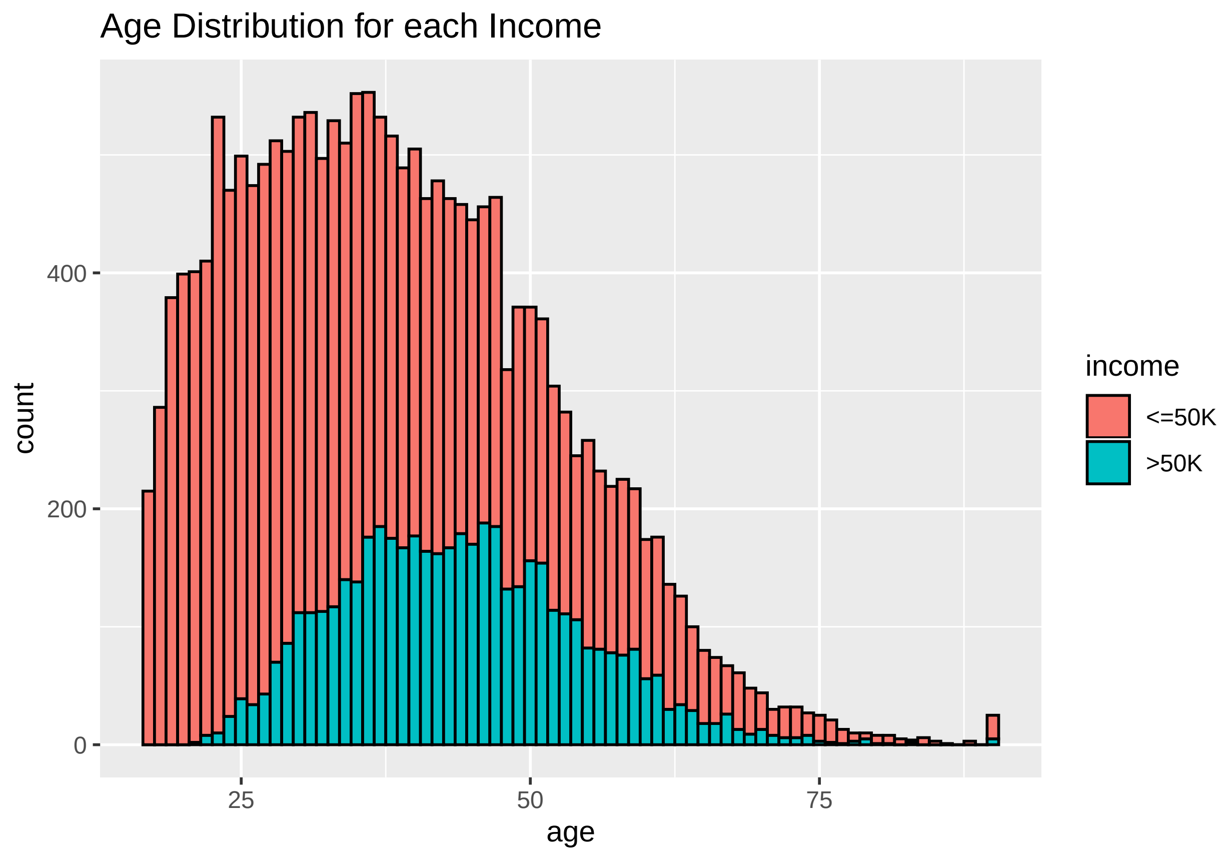
Data Visualization

Now we will explore and visualize several variables in our dataset to gain insights on which might make good predictors.

age

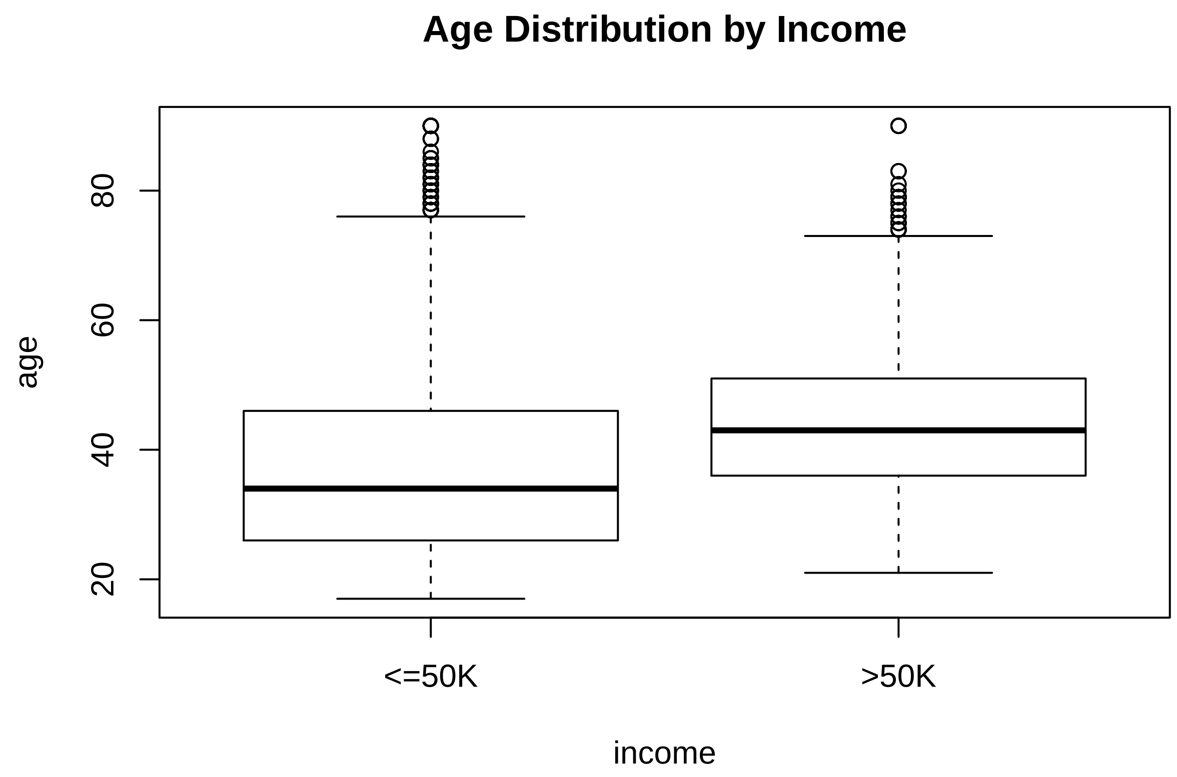
We can see a large amount of variability in the age attribute, which should make a good predictor for income.

```
#Make a histogram of the distribution of age for each income value
training %>% ggplot(aes(age)) +
  geom_histogram(aes(fill=income),color='black',binwidth=1) +
  labs(title= "Age Distribution for each Income")
```



We can see from our boxplot that the median age for individuals with greater than \$50k income is significantly higher than the median age for individuals with less than or equal to \$50k income.

```
#Make a boxplot of the distribution of age by income.
boxplot(age ~ income, data=training, main="Age Distribution by Income")
```



education.num

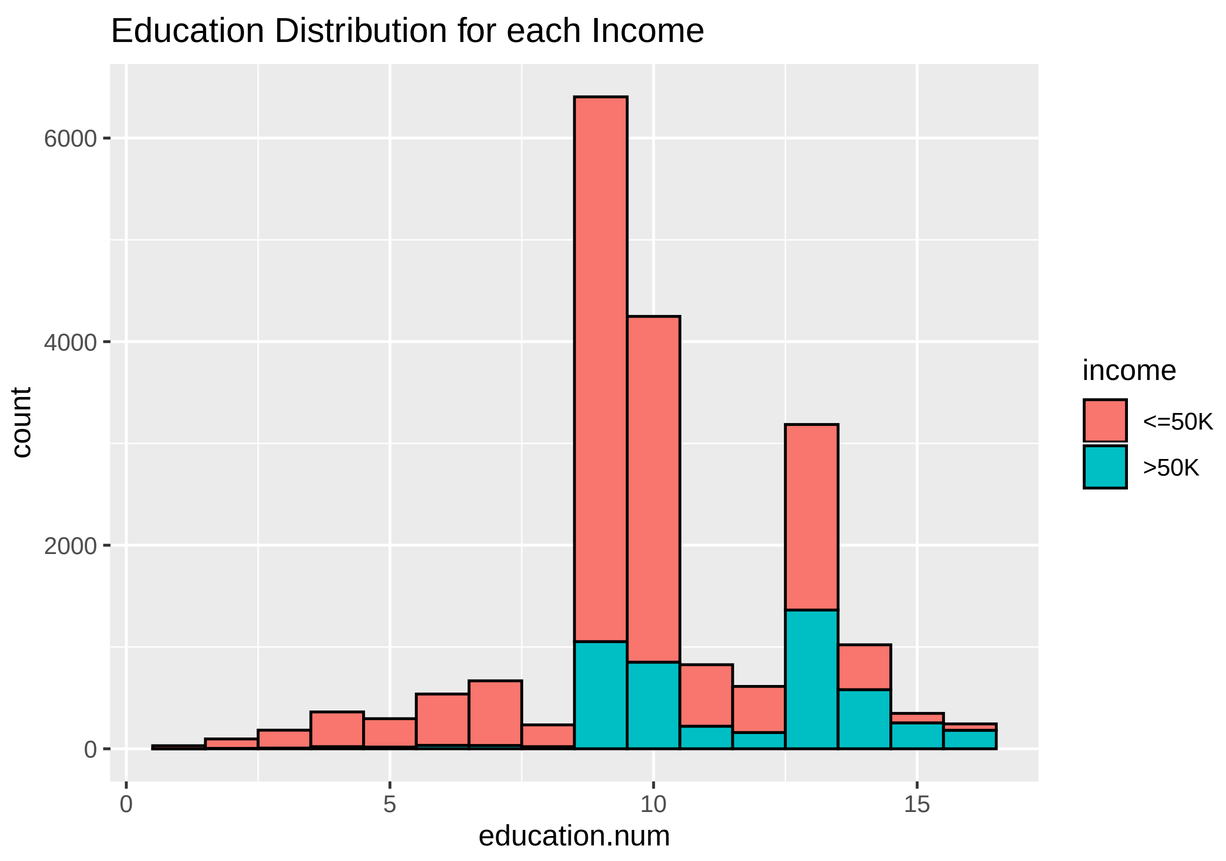
Here we can see that education.num ranges from 1 to 16, with a median of 10. These integers represent education levels with 1 being the lowest (Preschool) and 16 being the highest (Doctorate).

```
#View the summary statistics for the education.num variable
summary(training$education.num)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     9.0    10.0    10.1    12.0    16.0
```

Our histogram shows us that higher education levels tend to have a higher proportion of observations that have greater than $50k income.

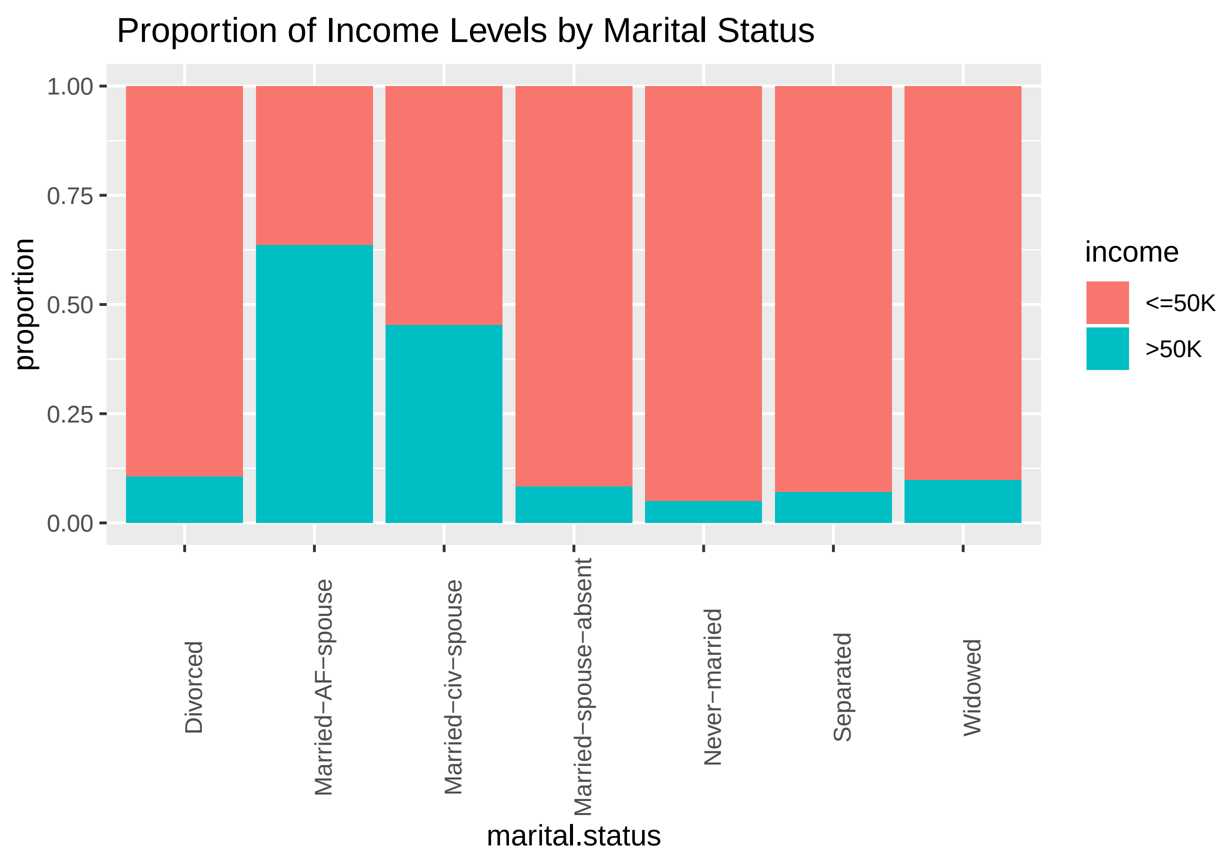
```
#Make a histogram of the distribution of education.num for each income value
training %>% ggplot(aes(education.num)) +
  geom_histogram(aes(fill=income),color='black',binwidth=1) +
  labs(title= "Education Distribution for each Income")
```



marital.status

Our bar graph shows that a higher proportion of married people with spouses that are not absent have incomes over $50k.

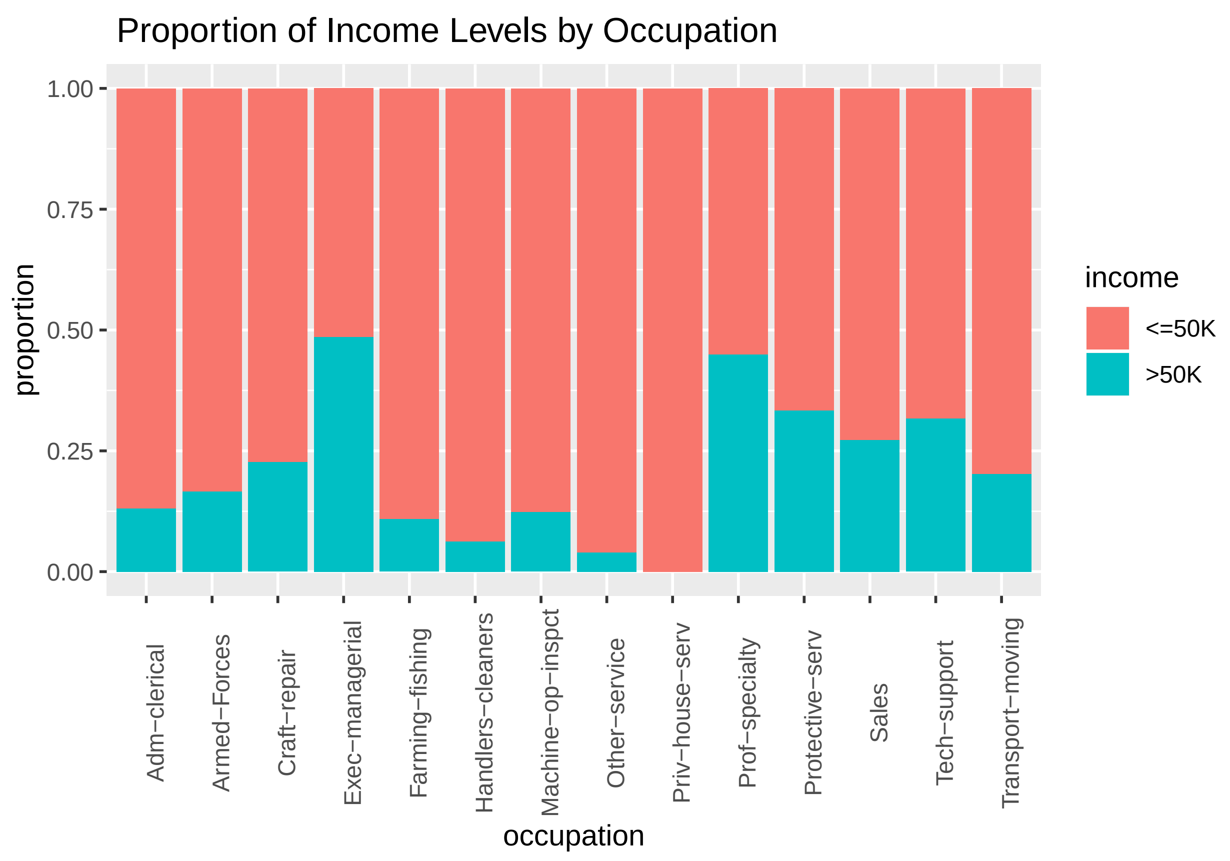
```r
#Make a bar graph to show the proportion of income levels by marital status
training %>% ggplot(aes(marital.status, fill = income)) +
  geom_bar(position = "fill") +
  labs(y = "proportion", title=" Proportion of Income Levels by Marital Status") +
  theme(axis.text.x = element_text(angle=90))
```



occupation

Here we can see that certain occupations have much a higher proportion of individuals making greater than \$50k than other occupations. This indicates that occupation will be a great predictor for our models. The occupations with the highest proportion of individuals making over \$50k are Exec-managerial, Prof-specialty, Protective-serv, and Tech-support.

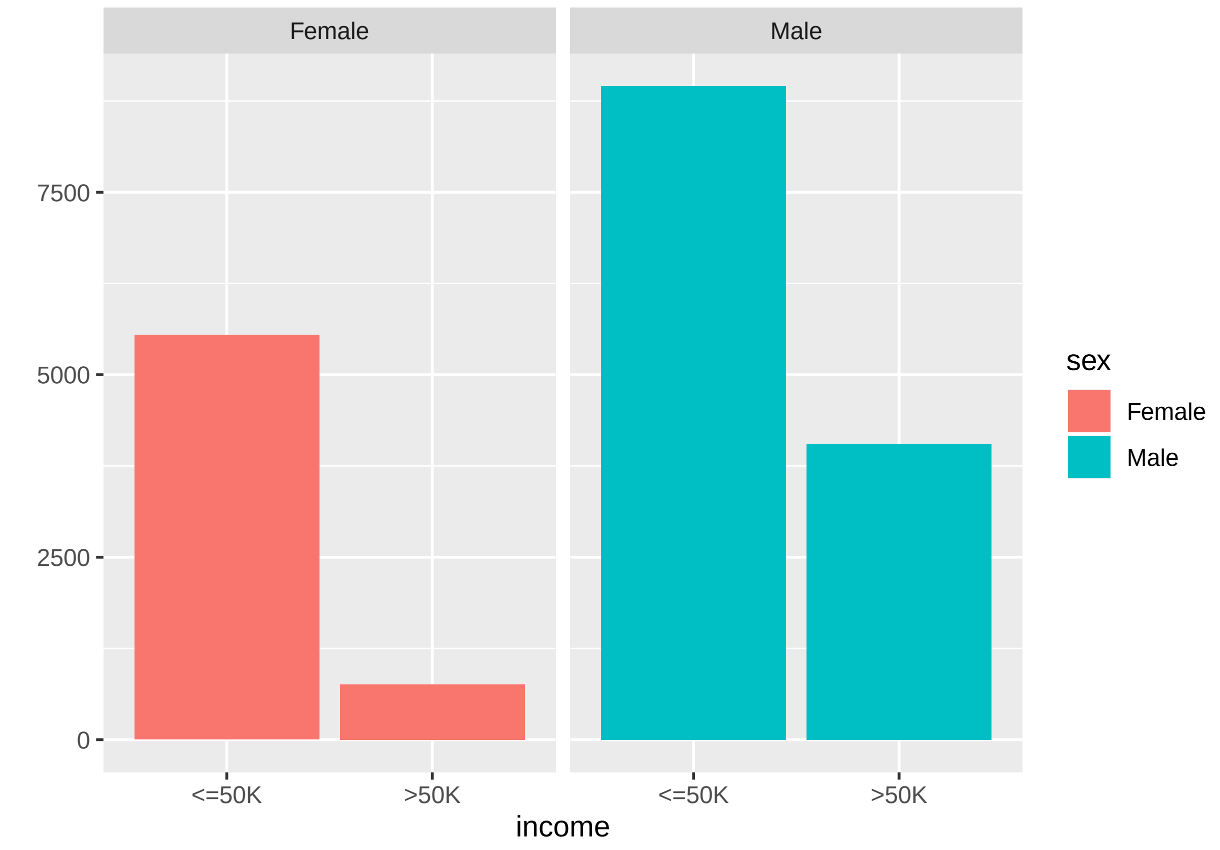
```r
#Make a bar graph showing the proportion of income levels by occupation
training %>% ggplot(aes(occupation, fill = income)) +
  geom_bar(position = "fill") +
  labs(y = "proportion", title=" Proportion of Income Levels by Occupation")  +
  theme(axis.text.x = element_text(angle=90))
```



sex

Here we can see that a higher proportion of males make greater than $50k than women. This shows that sex will be a good predictor for our models.

```r
#Plot the distribution of income levels by gender
qplot(income, data = training, fill = sex) + facet_grid (. ~ sex)
```



Near Zero Variance

We will use the nearZeroVar function from the caret package to see which variables need to be explored and possibly not used in our models.

None of our variables have zero variance. We can see that capital.gain, capital.loss, and native.country have low variance and must be explored.

```
#Use the nearZeroVar function to identify variables with zero or low variance
nearZeroVar(training, saveMetrics=TRUE)
```

```
##                  freqRatio percentUnique zeroVar   nzv
## age               1.001812    0.37301834   FALSE FALSE
## workclass         9.159949    0.03626567   FALSE FALSE
## education.num     1.507768    0.08289296   FALSE FALSE
## marital.status    1.436890    0.03626567   FALSE FALSE
## occupation        1.003107    0.07253134   FALSE FALSE
## relationship      1.605093    0.03108486   FALSE FALSE
## race              8.855996    0.02590405   FALSE FALSE
## sex               2.063810    0.01036162   FALSE FALSE
## capital.gain     84.531100    0.56988913   FALSE  TRUE
## capital.loss    158.577586    0.42482644   FALSE  TRUE
## hours.per.week    5.318527    0.47663455   FALSE FALSE
## native.country   47.479784    0.20723241   FALSE  TRUE
## income            3.017902    0.01036162   FALSE FALSE
```

Using summary statistics to explore capital.gain and capital.loss, we can see that the variable means for each level of income are very different. This means that despite low variance, these may still be good predictors.

```r
#View summary statistics for individuals with less than or equal to $50k income
summary (training[ training$income == "<=50K",
                       c("capital.gain", "capital.loss")])
```

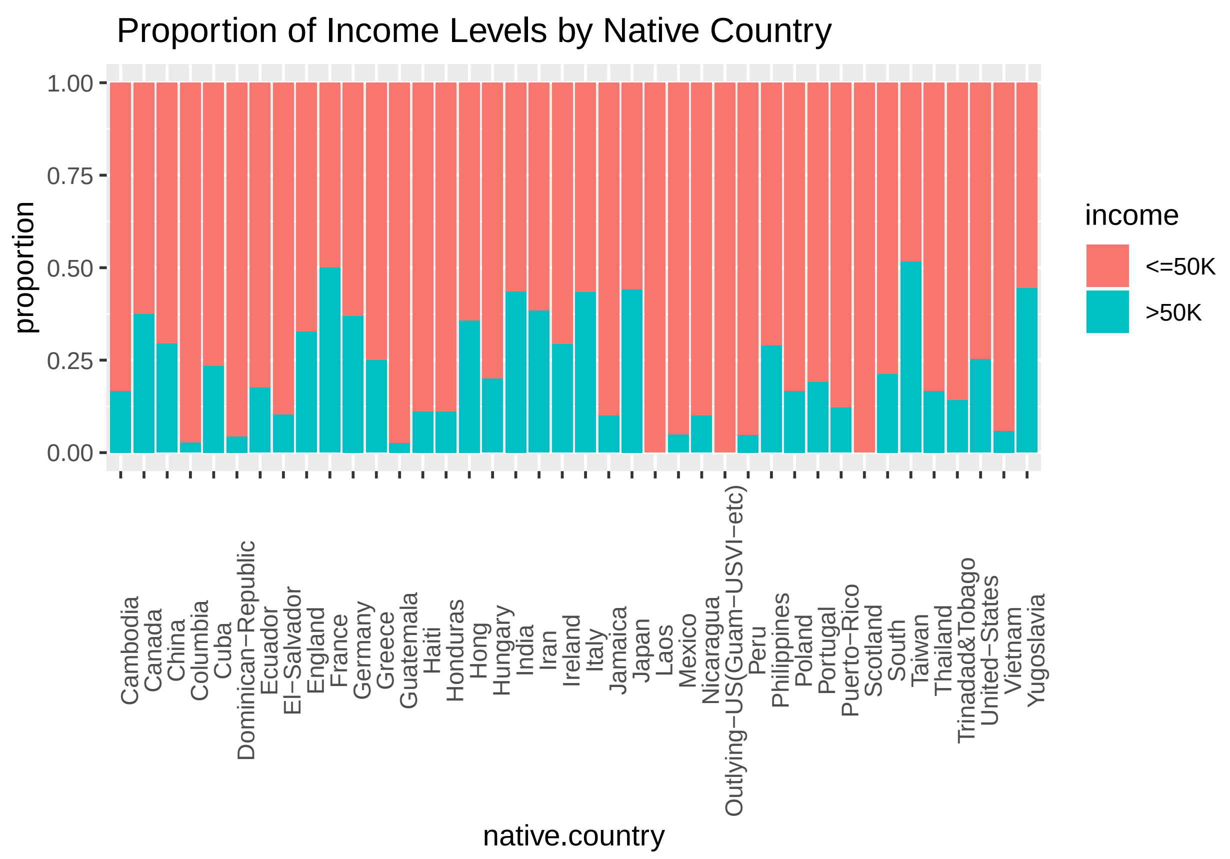
```
##   capital.gain      capital.loss
##  Min.   :    0.0   Min.   :   0.0
##  1st Qu.:    0.0   1st Qu.:   0.0
##  Median :    0.0   Median :   0.0
##  Mean   :  142.6   Mean   :  52.3
##  3rd Qu.:    0.0   3rd Qu.:   0.0
##  Max.   :41310.0   Max.   :4356.0
```

```r
#View summary statistics for individuals with greater than $50k income.
summary (training[ training$income == ">50K",
                       c("capital.gain", "capital.loss")])
```

```
##   capital.gain    capital.loss
##  Min.   :    0   Min.   :   0.0
##  1st Qu.:    0   1st Qu.:   0.0
##  Median :    0   Median :   0.0
##  Mean   : 4089   Mean   : 195.7
##  3rd Qu.:    0   3rd Qu.:   0.0
##  Max.   :99999   Max.   :3683.0
```

Despite low variance, we can see that the proportion of individuals making greater than $50k varies quite a bit by native country. This means native.country can be used as a predictor for income.

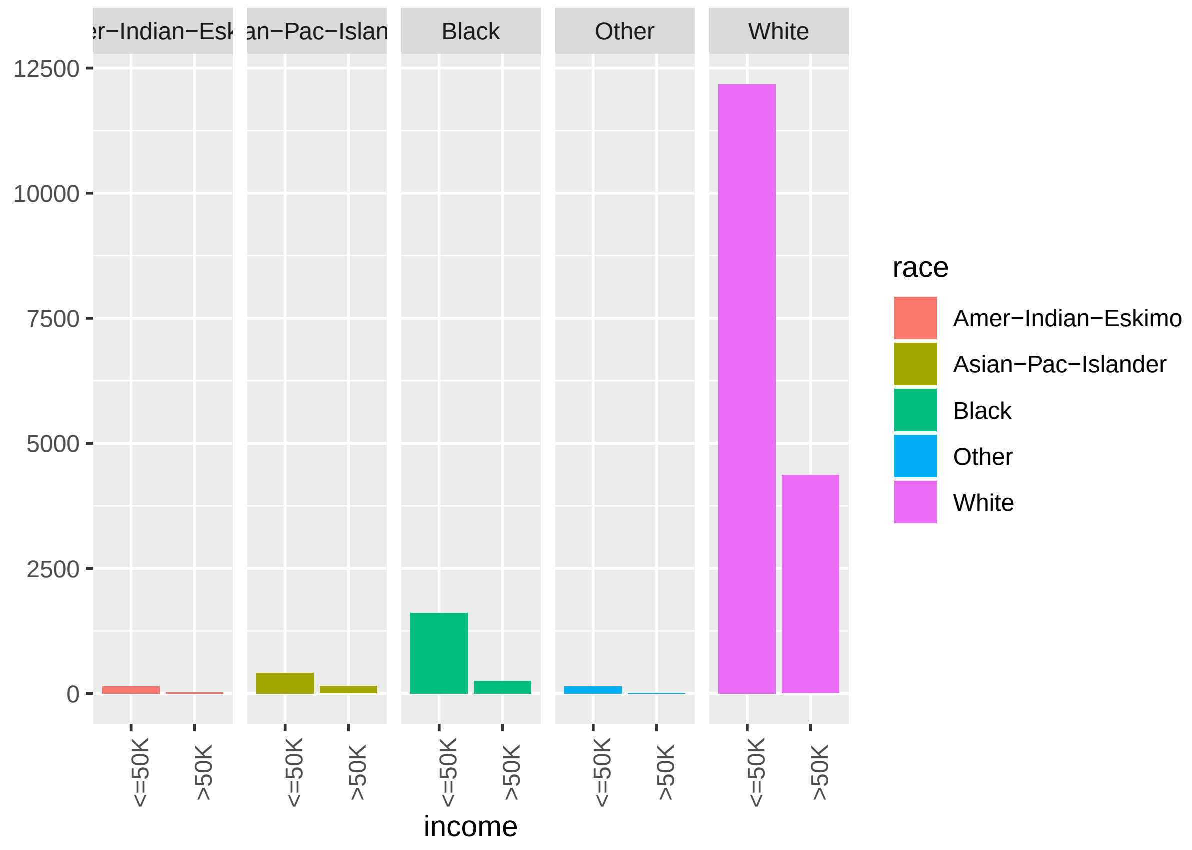
```r
#Make a bar graph to show the proportion of income levels by native.country
training %>% ggplot(aes(native.country, fill = income)) +
  geom_bar(position = "fill") +
  labs(y = "proportion", title=" Proportion of Income Levels by Native Country")  +
  theme(axis.text.x = element_text(angle=90))
```



race

Here we can see that some races have a higher proportion of individuals with an income greater than $50k. This indicates that race will be a good predictor for our model.

```
#Plot the distribution of income levels by race
qplot(income, data = training, fill = race) +
  facet_grid (. ~ race) +
  theme(axis.text.x = element_text(angle=90))
```



Machine Learning Models

Now we will begin to test various machine learning models to see which has the highest overall accuracy in predicting whether an individual has an income that is less than or equal to $50k or greater than $50k.

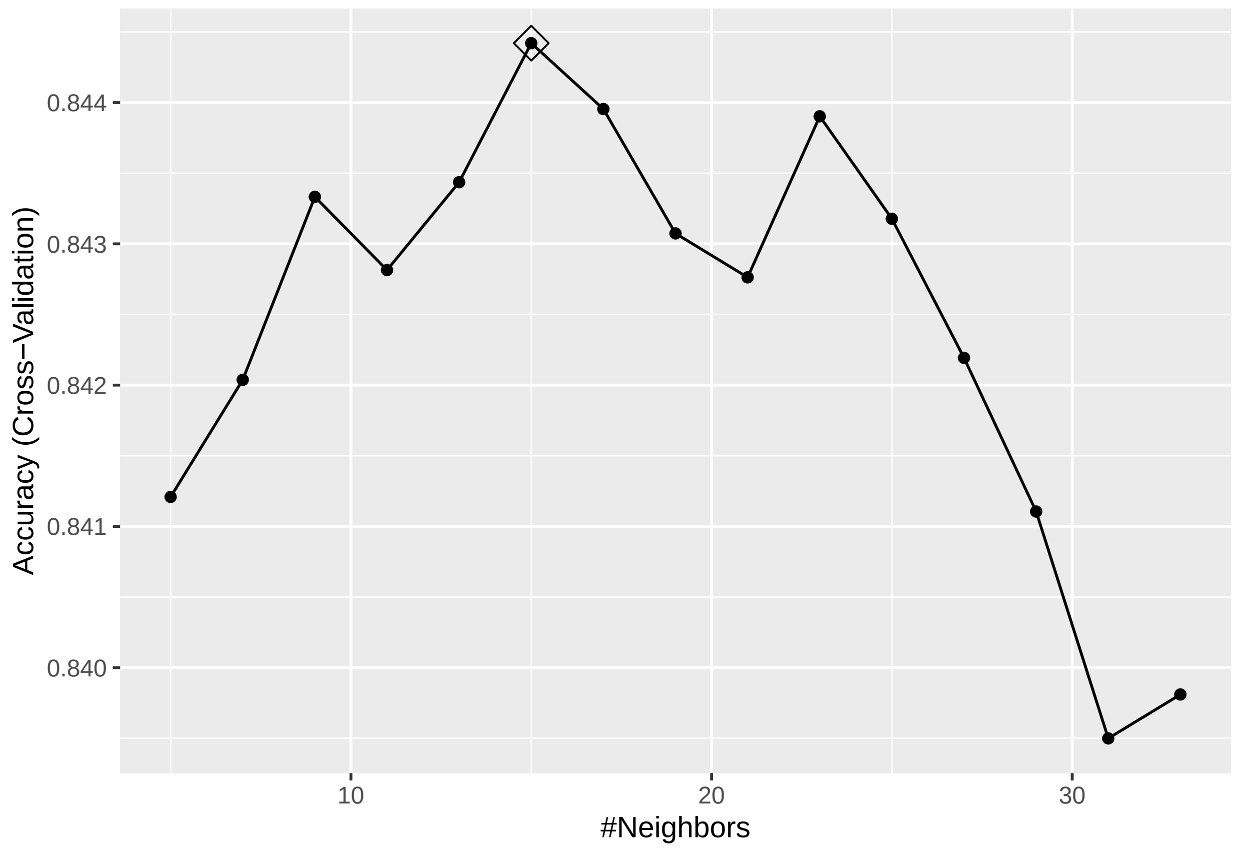
knn (K nearest neighbors) Model

We will first try a knn, or k nearest neighbors, model. It is based on a similarity concept and is similar to bin smoothing. Knn is also adaptable to multiple dimensions. It works by calculating the distance between observations based on the attributes. New data points, or observations, are predicted by looking at the k-nearest points and averaging them. Therefore, if the majority of K-neighbors belong to a certain class, the new observation also belongs to that same class.

Here we use k as our tuning parameter, which represents the number of neighbors to be considered. We use a 10-fold cross-validation to make our code run faster and to avoid overfitting. We will have 10 validation samples that use 10% of the observations in each sample that are used to create separate boosted models. The final model is an ensemble based on all of these models.

```
#Train a knn model on our training dataset optimizing k as the tuning parameter
set.seed(9,sample.kind = "Rounding")  #if using R3.5 or earlier set.seed(9)
#Using a 10 fold cross-validation method to make our code run faster.
control <- trainControl(method = "cv", number = 10, p = .9)
train_knn <- train(income ~ .,
                   method = "knn",
                   data = training,
                    tuneGrid = data.frame(k = seq(5,33,2)),
                    trControl = control)

#Highlighting the optimized k value on this plot
ggplot(train_knn, highlight = TRUE)
```



```
#Use this code to see that the best k value is 15
train_knn$bestTune
```

```
##    k
## 6 15
```

```
#Compute the accuracy of the knn model on the testing dataset
knn_accuracy <- confusionMatrix(predict(train_knn, testing, type = "raw"),
                testing$income)$overall["Accuracy"]

#Create a table to save our results for each model
accuracy_results <- tibble(method = "knn", Accuracy = knn_accuracy)
#View the knn accuracy results in our table
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745

gbm (Gradient Boosting Machines) Model

For our second model, we will try a gbm, or Gradient Boosting Machines model. This approach creates an ensemble where new models are added sequentially rather than simply averaging the predicted values of all the models. The Gradient Boosting Machine method builds an ensemble of shallow and successive trees where each learns and improves based on previous trees. We will also use a 10 fold cross-validation method here.

```
#Train a gbm model on our training dataset using 10-fold cross-validation
set.seed(2000,sample.kind = "Rounding") #if using R3.5 or earlier set.seed(2000)
#Using a 10 fold cross-validation method to make our code run faster
 trCtrl <- trainControl (method = "cv", number = 10)
#Train a gbm model
 train_gbm <- train (income~ .,
                 trControl = trCtrl,
                 method = "gbm",
                 preProc="zv",
                 data = training,
                 verbose = FALSE)

#Compute the accuracy of our gbm model on the testing dataset
gbm_accuracy <- confusionMatrix(predict(train_gbm, testing, type = "raw"),
                 testing$income)$overall["Accuracy"]

#Save the gbm accuracy results to our table
accuracy_results <- bind_rows(accuracy_results, tibble(method="gbm",
                                                       Accuracy = gbm_accuracy))
#View the gbm accuracy results in our table
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745
gbm	0.8593329

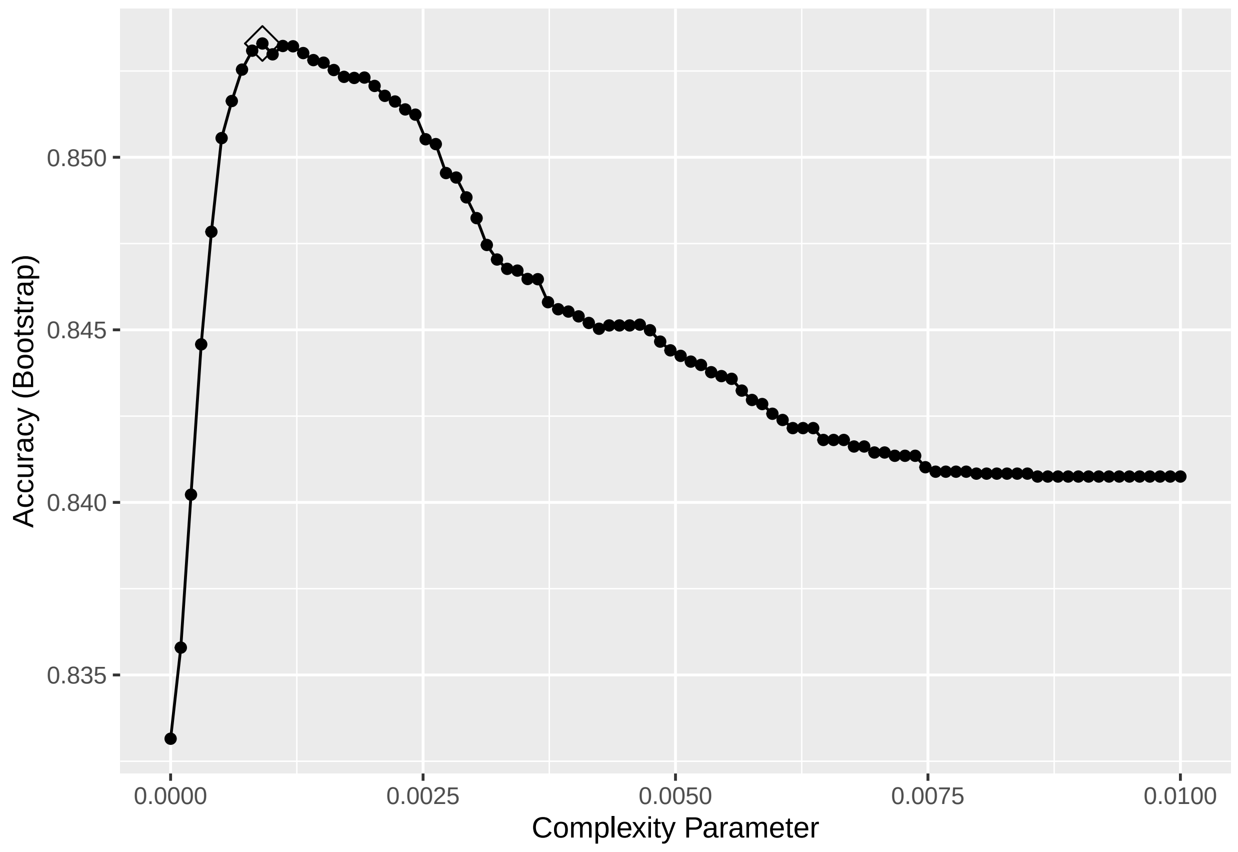
Classification Tree Model

For our third model, we will train a Classification Tree algorithm using the rpart method from the caret package. A tree can be described as a flow chart with yes or no questions and predictions at the ends that are called nodes. Decision trees are a type of supervised learning algorithm that work by partitioning the predictor space in order to predict an outcome, which in our case is income. The partitions are created recursively.

We will use cross-validation to choose the best cp (complexity parameter).

```
#Train a Classification Tree model using rpart and optimizing for the complexity parameter
set.seed(300,sample.kind = "Rounding")  #if using R3.5 or earlier set.seed(300)
train_rpart <- train(income ~ .,
                      method = "rpart",
                      tuneGrid = data.frame(cp = seq(0, 0.01, len=100)),
                      data = training)
```

```
#Highlight the optimized complexity parameter
ggplot(train_rpart, highlight=TRUE)
```



```
#Use this code to see the best cp value
train_rpart$bestTune
```

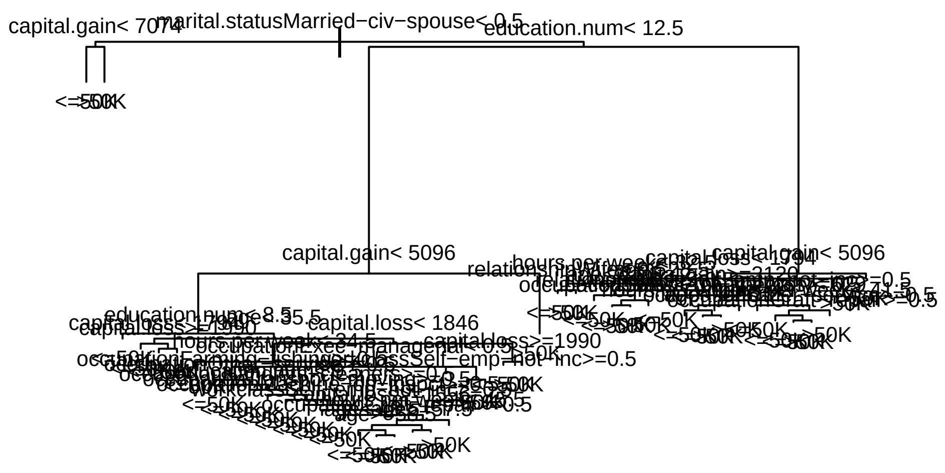
```
##              cp
## 10 0.0009090909
```

```
#Compute the accuracy of our Classification Tree model on the testing dataset
rpart_accuracy <- confusionMatrix(predict(train_rpart, testing),
                                  testing$income)$overall["Accuracy"]

#Save the Classification Tree model accuracy results to our table
accuracy_results <- bind_rows(accuracy_results,
                              tibble(method="rpart", Accuracy = rpart_accuracy))
#View the rpart accuracy results in our table
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745
gbm	0.8593329
rpart	0.8576756

```r
#View the text version of our classification tree
plot(train_rpart$finalModel, margin = 0.1)
text(train_rpart$finalModel, cex = 0.7)
```



Random Forest Model

For our last model, we will train a random forest model using the randomForest package in r. Random Forests take the average of multiple decision trees in order to improve predictions. Random Forests use the bootstrap to introduce randomness and ensure that individual trees are unique. They sample N observations with replacement from the training set to create a bootstrap training set. The second way that Random Forests introduce randomness is that each tree is built from its own randomly selected subset of features. This random selection process helps reduce the correlation between the trees. Finally, the Random Forest algorithm creates an ensemble by averaging the predictions of all the trees to form a final prediction.

```r
#Train a random forest model on our training dataset
set.seed(3,sample.kind = "Rounding")  #if using R3.5 or earlier set.seed(3)
train_rf <- randomForest(income ~ ., data = training)
#Compute the accuracy of our random forest model on the testing dataset
rf_accuracy <- confusionMatrix(predict(train_rf, testing),
                               testing$income)$overall["Accuracy"]

#Save the random forest accuracy results to our table
accuracy_results <- bind_rows(accuracy_results,
                              tibble(method="random forest", Accuracy = rf_accuracy))
```

```r
#View the random forest accuracy results in our table
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745
gbm	0.8593329
rpart	0.8576756
random forest	0.8601616

Testing the Final Model on our Validation Set

From our results table, we can see that our Random Forest model achieved the highest accuracy, so we will now test that model on our validation set.

```r
#Train our final random forest model on our census_training dataset
set.seed(3, sample.kind = "Rounding") #if using R3.5 or earlier set.seed(3)
final_train_rf <- randomForest(income ~ ., data = census_training)

#Compute the accuracy of our final random forest model on the validation set
final_accuracy <- confusionMatrix(predict(final_train_rf,
                                  census_validation),
                            census_validation$income)$overall["Accuracy"]

#Save the random forest accuracy results to our table.
accuracy_results <- bind_rows(accuracy_results,
                              tibble(method="Final Random Forest Model",
                                     Accuracy = final_accuracy))
#View the final random forest model accuracy results in our table
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745
gbm	0.8593329
rpart	0.8576756
random forest	0.8601616
Final Random Forest Model	0.8614288

Results

We can see from our results table that the random forest model has the highest overall accuracy of 0.8601 on the data we set aside for training and testing purposes. After choosing Random Forest as our final model, we tested our predictions on the validation dataset and achieved an overall accuracy of 0.8614 for predicting whether a person's income is less than or equal to $50k or greater than $50k.

```r
#Results
accuracy_results %>% knitr::kable()
```

method	Accuracy
knn	0.8489745
gbm	0.8593329
rpart	0.8576756
random forest	0.8601616
Final Random Forest Model	0.8614288

Conclusion

Summary

Our goal was to test various machine learning models to predict whether a person's income is less than or equal to $50k or greater than $50k. We wanted to see which model worked best, based on overall accuracy. We explored the public Adult Census Income dataset from the UCI Machine Learning Repository. We performed various cleaning and preprocessing steps such as removing rows with missing values. Then we used r code to visualize and plot different variables in the dataset. After training 4 different types of machine learning models, we found that our Random Forest model gave us the highest overall accuracy. We tested our final Random Forest model on our validation dataset and achieved an overall accuracy of 86.14%.

Limitations and Future Work

One of the main limitations of the dataset is that it is over 25 years old and is not representative of the current US Population. Therefore, if we were to make predictions based on this historical data, they wouldn't perform as well on current data. Since so many of the variables influencing income have changed in the past 25 years, it would be interesting to train new machine learning models on more recent census data to examine the differences.