Final Project: Predictive Analysis for Wage Data set from ISLR2 package in R

MATH 40028/50028: Statistical Learning

March 22, 2022

ACADEMIC INTEGRITY: Every student should complete the project by their own. A project report having high degree of similarity with work by any other student, or with any other document (e.g., found online) is considered plagiarism, and will not be accepted. The minimal consequence is that the student will receive the project score of 0, and the best possible overall course grade will be D. Additional consequences are described at http://www.kent.edu/policyreg/administrative-policy-regarding-student-cheating-and-plagiarism (http://www.kent.edu/policyreg/administrative-policy-regarding-student-cheating-and-plagiarism) and will be strictly enforced.

Instruction

Goal: The goal of the final project is to apply the statistical learning methods discussed in this course to perform predictive analysis of real-life data. You will need to identify prediction problem(s), carry out necessary exploratory data analysis, perform predictive analysis using statistical learning methods, assess the performance, and communicate the results in a report.

Report: Use this Rmd file as a template. Edit the file by adding your project title in the YAML, and including necessary information in the four sections: (1) Introduction, (2) Statistical learning strategies and methods, (3) Predictive analysis and results, and (4) Conclusion.

Submission: Please submit your project report as a PDF file (4-8 pages, flexible) to Canvas by **11:59 p.m. on May 4**. The PDF file should be generated by "knitting" the Rmd file. You may choose to first generate an HTML file (by changing the output format in the YAML to `output: html_document`) and then convert it to PDF. **20 points will be deducted if the submitted files are in wrong format.**

Grade: The project will be graded based on your ability to (1) recognize and define prediction problems, (2) identify potentially useful statistical learning methods, (3) perform the predictive analysis and assess the performance, (4) document the analysis procedure (with R code) and clearly present the results, and (5) draw valid conclusions supported by the analysis.

Datasets: You may consider (but are not restricted) to use the following packages/datasets.

- `ISLR2` (https://cran.rstudio.com/web/packages/ISLR2/ISLR2.pdf): datasets used in the *Introduction to Statistical Learning* textbook
- `dslabs` (https://cran.r-project.org/web/packages/dslabs/dslabs.pdf)

Introduction [15 points]

Loading data and packages required

The first step is to load the necessary packages and data set. The “ISLR2” package, which contains the data, must be loaded in order to conduct this analysis. `randomForest` and `caret` are two additional packages added for the analysis’s efficiency. ‘Wage’ data will be used in the analysis. This data description is as follows: For a sample of 3000 male workers in the Mid Atlantic region, the “Wage” data collection includes information on salaries and other relevant statistics.

```
# loading required packages
library(caret)
library(e1071)
library(ISLR2)
library(tidyverse)
library(ggcorrplot)
library(randomForest)

# loading data to working space
wage_data <- ISLR2::Wage
```

Target Population and sampling techniques used

The workforce in the Mid-Atlantic region, as shown by the wage data, would be the target group in this scenario. The model built using this data set would predict the workers in the Mid-Atlantic region with comparable demographics and job-related attributes. Predictive analysis uses a variety of sampling techniques to choose this data from a broader population.

1. In a simple random sampling strategy, each observation in the population has an equal chance of being chosen. This strategy is typically used when the population size is known and there are small datasets.
2. Divide the overall population into distinct subgroups or strata based on a set of criteria, such as age, gender, or income, and then randomly choose a sample from each stratum. This tactic is useful when the general population is diverse and the subgroups all possess distinctive characteristics.
3. The population is separated into clusters or groups based on numerous factors, such as location. The study is then conducted on a sample of these clusters that are selected at random. Every individual in the selected clusters is included in the analysis. This method can be useful when it is difficult or expensive to identify and choose individuals from the community.

Possible bias in predictive analysis.

Potential bias is defined as any systematic error or deviation from the true significance in an investigation or study that has the potential to compromise the validity, correctness, or dependability of the findings. Some frequent reasons for potential bias in predictive analysis include the following:

1. Sampling bias occurs when conclusions about the population are drawn after an analysis of a sample that is not representative of the population from which it was drawn.
2. Improper measurements or information collection methods that result in results that are erroneous or imprecise are known as measurement bias.
3. Selection bias results in an unrepresentative sample when the criteria used to include or reject study participants are skewed.

Predictive Problem

The goal is to predict a worker's comparable income or compensation based on a variety of demographic and employment related factors, including age, education, experience, and occupation. The prediction challenge, to put it another way, is to develop an algorithm that can accurately estimate a worker's salary based on their professional and private characteristics.

This can be viewed as a regression problem where a set of input variables (demographic and job-related traits) must be utilized for predicting a continuous output variable (salary). Therefore, our goal variable from this set of data will be the integer variable "wage," which holds details about the worker's wage.

Splitting of the data

The data will then be divided into training and testing sets. To determine how effectively a model generalizes to new, untested data, predictive modeling divides a data set into training and test sets. The data set is frequently randomly split into two sets: a training set and a test set. A test set is used to evaluate the model's performance once the structure of the model has been fitted using the set of training data. How well the model predicts the outcome variables for the test data during testing determines how well it is evaluated.

The algorithm learns during training by observing patterns in the training data. The model's ability to generalize to fresh, untested data is determined by how well it performs on the test data set. To build an index that will be utilized to divide the wage_data, we will use the caret's `createDataPartition()` function. We divided the data by 80/20, meaning that 80 percent of the data will be used for training and the remaining 20 percent for testing. This is seen below.

```
set.seed(12345)
# index that split the data
train_index <-
  createDataPartition(wage_data$wage, p = 0.8, list = FALSE)
# train and test data
training_data <- wage_data[train_index, ]
testing_data <- wage_data[-train_index, ]
```

Statistical learning strategies and methods [35 points]

exploratory data analysis on the training data set

We do exploratory data analysis, or EDA, on the data used for training in this analysis in order to comprehend the distribution of the variables, identify any outliers or missing values, and determine the relationships between the predictors and response variables. The `boxplot.stats()` and `is.na()` routines are used to check for outliers and missing data, respectively, as illustrated below.

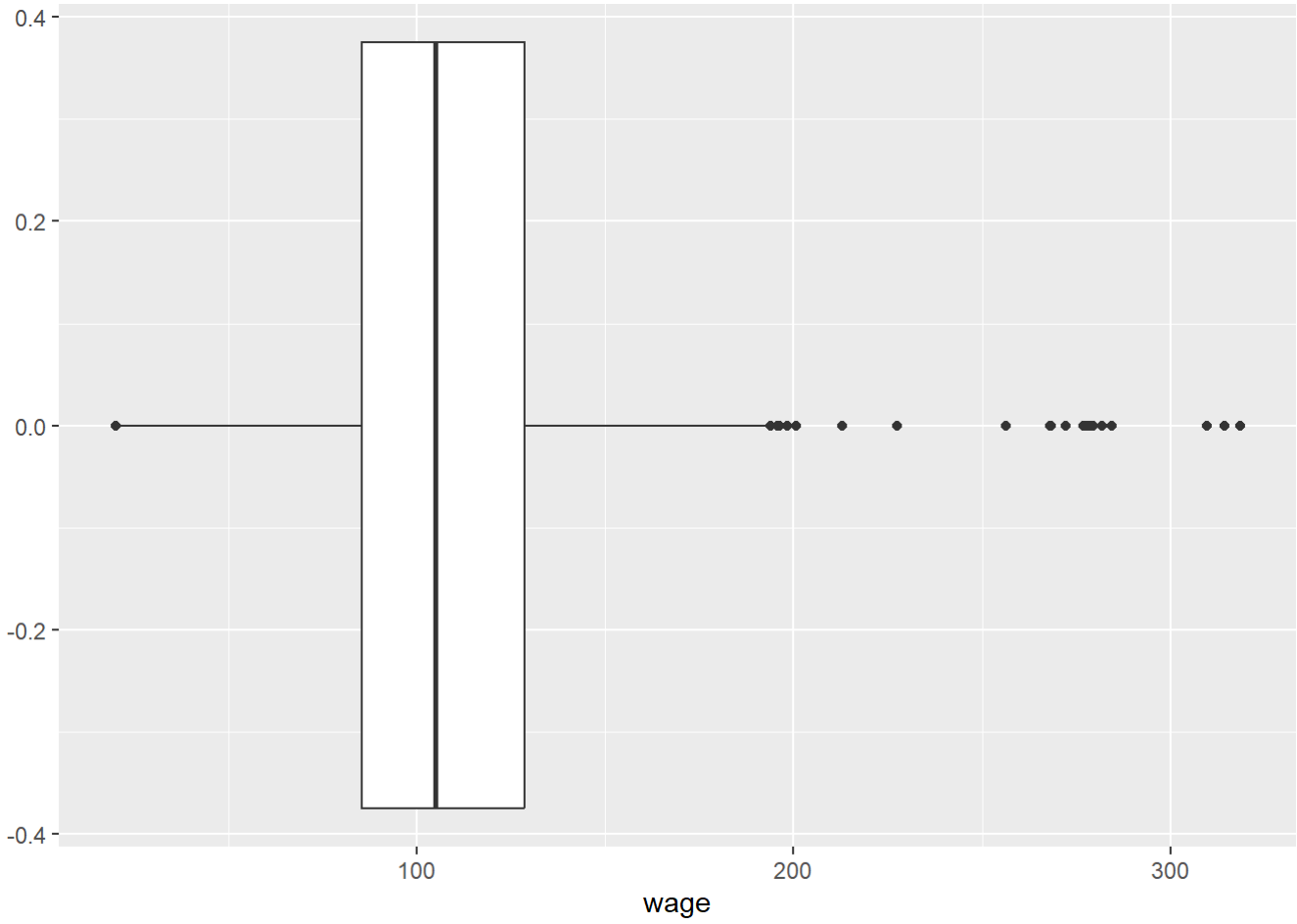
```
# check the structure of data
str(training_data)
```

```
## 'data.frame':    2402 obs. of  11 variables:
##  $ year      : int  2006 2004 2003 2003 2005 2008 2009 2006 2004 2007 ...
##  $ age       : int  18 24 45 43 50 54 44 41 52 45 ...
##  $ maritl    : Factor w/ 5 levels "1. Never Married",..: 1 1 2 2 4 2 2 1 2 4 ...
##  $ race      : Factor w/ 4 levels "1. White","2. Black",..: 1 1 1 3 1 1 4 2 1 1 ...
##  $ education : Factor w/ 5 levels "1. < HS Grad",..: 1 4 3 4 2 4 3 3 2 3 ...
##  $ region    : Factor w/ 9 levels "1. New England",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ jobclass  : Factor w/ 2 levels "1. Industrial",..: 1 2 1 2 2 2 1 2 2 2 ...
##  $ health    : Factor w/ 2 levels "1. <=Good","2. >=Very Good": 1 2 1 2 1 2 2 2 2 1 ...
##  $ health_ins: Factor w/ 2 levels "1. Yes","2. No": 2 2 1 1 1 1 1 1 1 1 ...
##  $ logwage   : num  4.32 4.26 4.88 5.04 4.32 ...
##  $ wage      : num  75 70.5 131 154.7 75 ...
```

```
# check for missing data
table(is.na(training_data))
```

```
##
## FALSE
## 26422
```

```
# check for outliers on the target variable wage
# use a boxplot
training_data %>%
  ggplot(aes(wage)) +
  geom_boxplot()
```

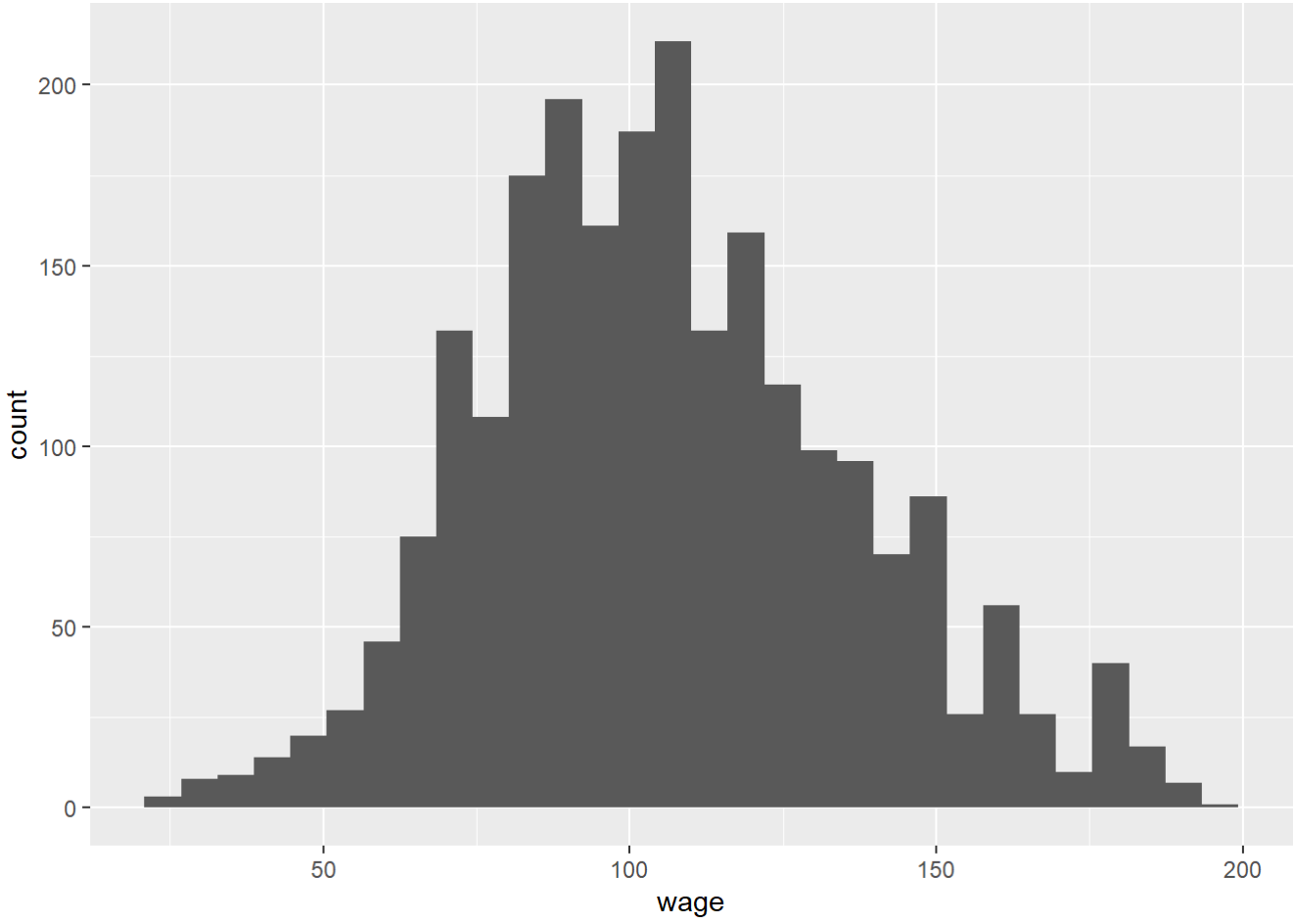


```
# our plot shows availability of outliers
# remove them using boxplot.stats()
outliers <- boxplot.stats(training_data$wage)$out

# filter out the outliers
training_data <- training_data %>%
  filter(!wage %in% outliers)

# check if the target variable is normally distributed
# using a histogram
training_data %>%
  ggplot(aes(wage)) +
  geom_histogram()
```

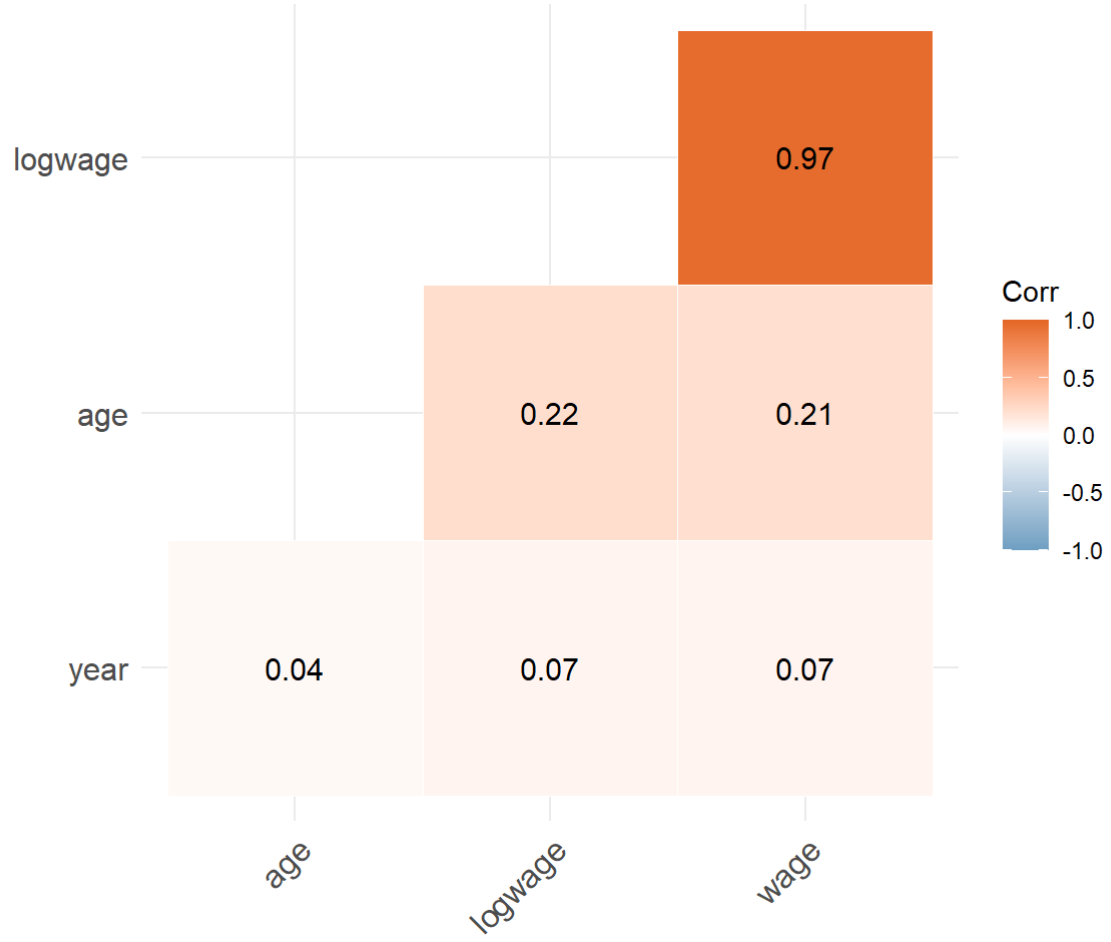
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# the target variable is normally distributed

# examine the correlation
# use a correlation matrix
num_vars <- sapply(training_data, is.numeric)
train_dat_num <- training_data[, num_vars]
corr_matrix <- cor(train_dat_num)

# Create a correlation heatmap
ggcorrplot(
  corr_matrix,
  hc.order = TRUE,
  type = "lower",
  outline.col = "white",
  colors = c("#6D9EC1", "white", "#E46726"),
  lab = TRUE
)
```




```
# explore interaction between dependent and independent variables
training_data %>%
  ggplot(aes(x = wage, age, color = education)) +
  geom_point() +
  facet_wrap( ~ race) +
  labs(title = 'A scatterplot showing distribution of wage on age, education and race', x =
'Target variable-Wage', y =
         'Age')
```

Using the `str()` function, we analyzed the data's structure, verified that there were no missing values, and then we chose the outliers in our target variable and removed these values from the data. According to the produced histogram, our target variable is regularly distributed. Using the correlation matrix, we can then see that the variable `logwage` exhibits a large negative correlation, whilst the others exhibit little to no association.Finally, we used a scatter plot to examine if any factors were dependent on pay. We found that these variables did indeed depend on the target variable. As a result, we can draw the conclusion that the independent variables in our data rely on the desired variable.

Statistical learning approach alongside other methods for feature engineering techniques

The process of turning unprocessed data into features that machine learning algorithms can exploit is known as feature engineering. By choosing the most pertinent features, decreasing noise, and enhancing the appearance of the data, feature engineering aims to enhance the effectiveness of machine learning models.

For feature engineering, a variety of statistical learning methods and techniques can be applied:

1. By identifying the most significant features that account for the majority of the variance in the data, principal component analysis (PCA), a statistical technique, can minimize the dimensionality of the data. A set of variables that are correlated can be converted into a set of unrelated variables using PCA, and these uncorrelated variables can subsequently be used as features in machine learning techniques.

2. Feature scaling is the procedure of altering the data's scale such that it is comparable. This is significant because the size of the input data has an impact on many machine learning methods. By using techniques like normalization, standardization, or log transformation, features can be scaled.

3. One-Hot Encoding: One-hot encoding is a method for converting categorical data into numerical information that can be used as a feature by machine learning algorithms. To do this, a binary feature is developed for each distinct value of the categorical variable.

4. The process of extracting new features from already existing features is known as feature extraction. Techniques like text mining, image processing, and signal processing can be used to do this.

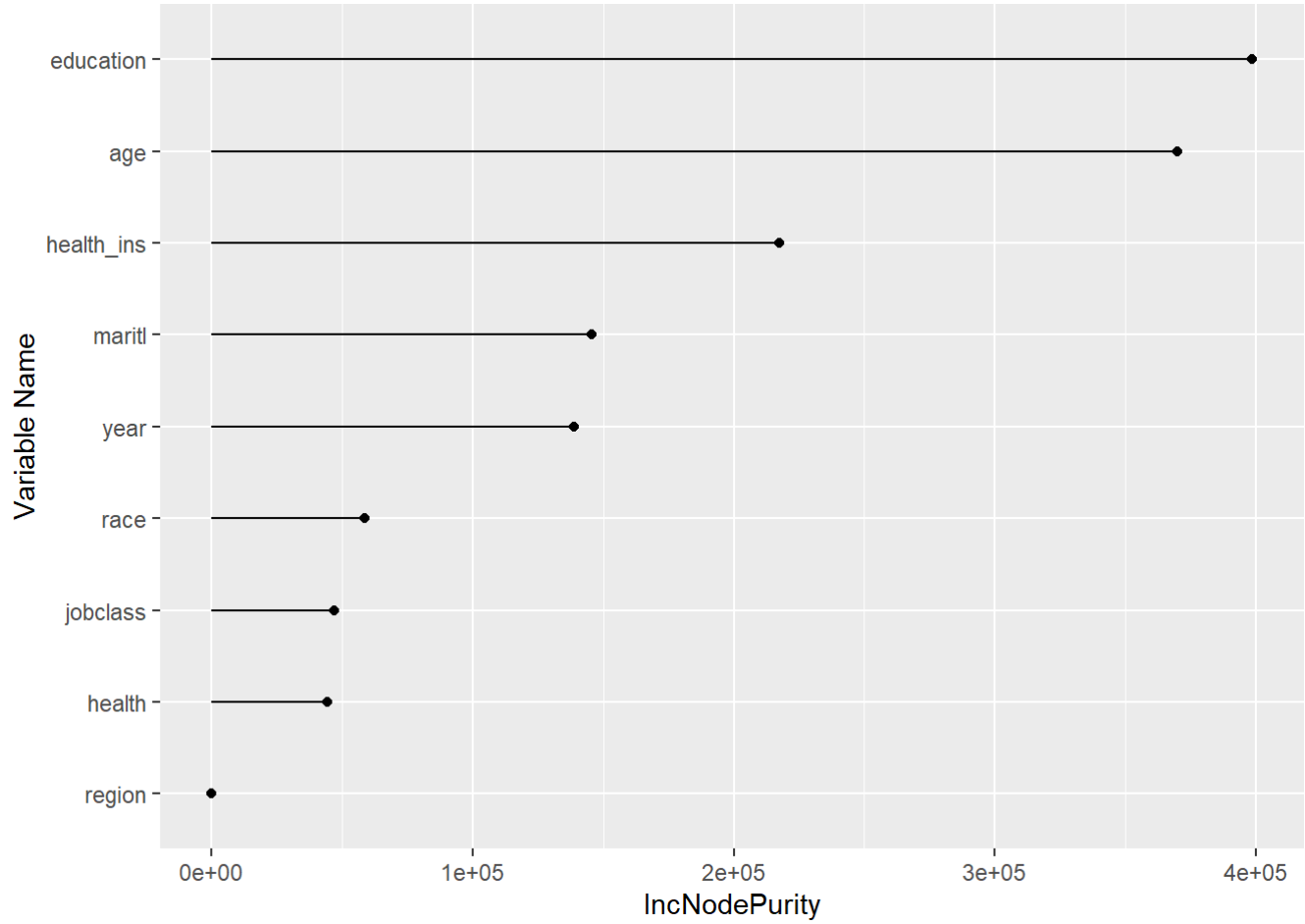
Therefore in our case we perform feature selection on our data so that we can use it to train a final machine learning model that will lead to best predictions.

```
set.seed(12345)
#train random forest model and calculate feature importance
rf <- randomForest(x= training_data[,1:9],y= training_data[,11])
var_imp <- varImp(rf, scale = FALSE)
#sort the score in decreasing order
var_imp_df <- data.frame(cbind(variable = rownames(var_imp), score = var_imp[,1]))
var_imp_df$score <- as.double(var_imp_df$score)
var_imp_df[order(var_imp_df$score,decreasing = TRUE),]
```

	variable <chr>	score <dbl>
5	education	398584.58
2	age	370100.69
9	health_ins	217567.24
3	maritl	145551.88
1	year	138746.31
4	race	58636.84
7	jobclass	46938.51
8	health	44123.11
6	region	0.00

9 rows

```
# create a plot for the data frame above
ggplot(var_imp_df, aes(x=reorder(variable, score), y=score)) +
  geom_point() +
  geom_segment(aes(x=variable,xend=variable,y=0,yend=score)) +
  ylab("IncNodePurity") +
  xlab("Variable Name") +
  coord_flip()
```



The algorithm determines the relative weights of each feature in a random forest model after training it on a dataset. The importance ratings are returned as a data frame after being sorted in decreasing order. The features shown in the dataframe were then shown in a plot. `Education` , `age` , `health_ins` , `year` , `maritl` , `race` , `jobclass` , and `health` are the plot variables that can be viewed as independent factors for the target variable. The final machine learning model will be based on these variables.

application of statistical learning techniques to forecasting under their presumptive assumptions.

When using statistical learning techniques to solve a prediction problem, it is important to take into account their various underlying assumptions. For continuous goal variables with a relationship that is linear between the given input variables and the target, linear regression, for instance, is appropriate. Target variables that are binary or categorical respond well to logistic regression. Random forests can accommodate inter variable interactions, whereas decision trees presume unbiased variables as inputs with a non-linear connection to the target. While neural networks are appropriate for complicated and non linear interactions between input parameters and target, support vector machine models also assume independently input indicators with a non linear connection to the target.

Predictive analysis and results [35 points]

Steps in the Statistical Learning Procedure for Predictive Analysis

For the procedure for predictive analysis for the wage data, we are going to use the Random Forest method to create a model that will help us to determine a worker's equivalent wage given a set of demographic and job-related characteristics using the `randomForest()` function.A number of parameters, including the number of

trees, the maximum depth of each tree, and the minimum number of observations per leaf, are passed into the function along with the training data as input. This is done as shown below.

```
# create a model using randomforest method
# formula used

formula <-
  as.formula("wage~year+age+maritl+race+education+jobclass+health+health_ins")
set.seed(12345)

# create the model
rf_model <-
  randomForest(
    formula,
    data = training_data,
    ntree = 500,
    mtry = 3
  )

# predict using the model
predict_rf_model <- predict(rf_model, newdata = testing_data)
```

In the above lines of codes, we started by identifying the model's variables and created an object called `formula` . This object contains the target variable that we want to predict using the other independent variables. The model created is a Random Forest Learning model. The model helps us determine workers wage given other characteristics. Using the models we predicted values using the `predict` function.

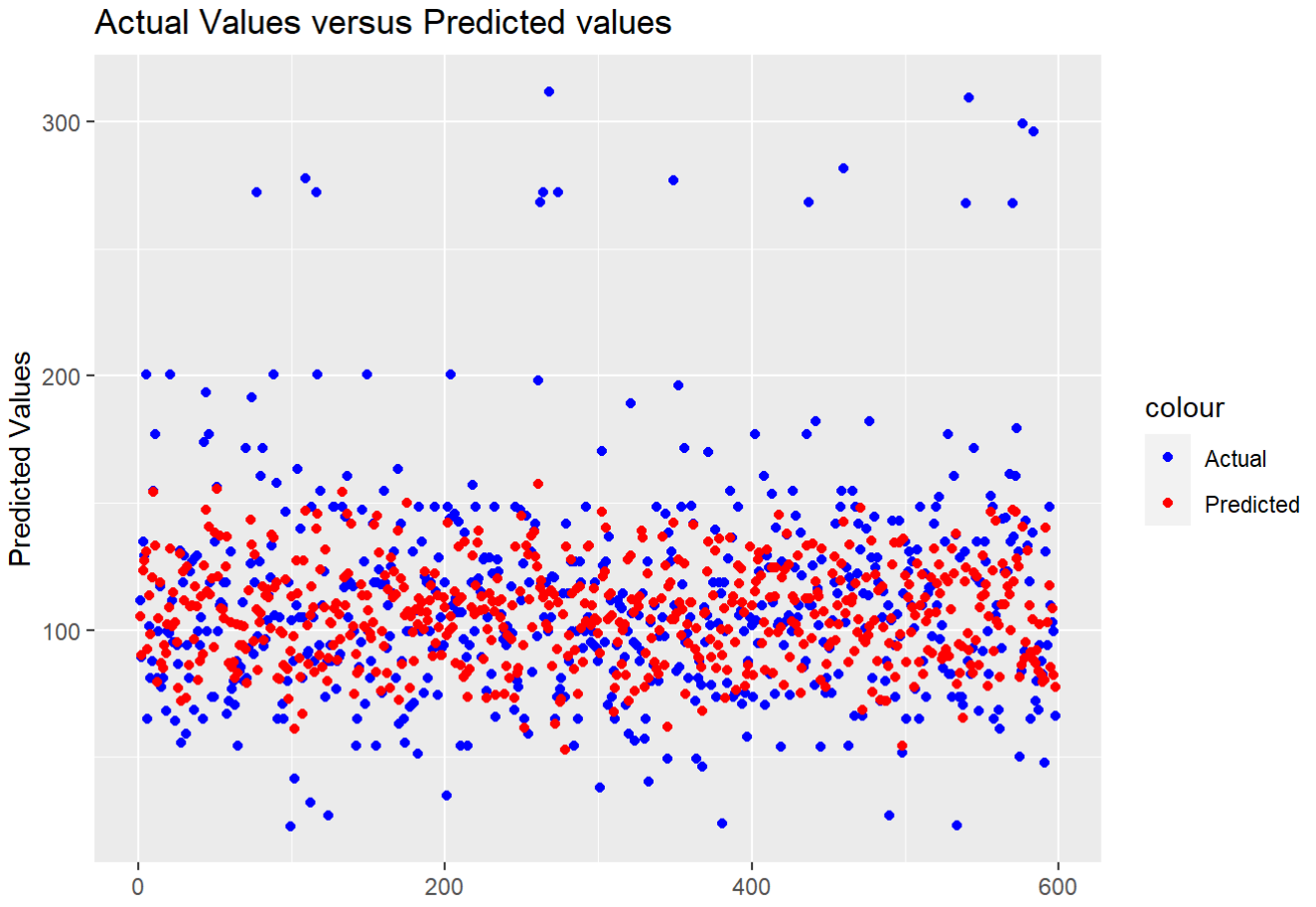
Analyzing the effectiveness of statistical learning methods with test data

To check the effectiveness of the model, we evaluate the performance of the random forest model using the Root Mean Square Error(RMSE), Mean Square Error(MSE), and Mean Absolute Error(MAE).

```
# evaluate the peformance of the model
rmse <- RMSE(predict_rf_model, testing_data$wage)
rsq <- R2(predict_rf_model, testing_data$wage)
MAE <- mean(abs(testing_data$wage - predict_rf_model))
cat('Mean Square Error:',rsq, '\nRoot Mean Square Error:',rmse ,'\nAverage Absolute Differenc
e:', MAE)
```

```
## Mean Square Error: 0.2878711
## Root Mean Square Error: 36.02434
## Average Absolute Difference: 23.12154
```

```
# 5. Visualize results using plots
tibble(x_value =seq_along(testing_data$wage) ,predicted_value = predict_rf_model, actual_valu
e = testing_data$wage) %>%
  ggplot(aes(x = x_value, y = actual_value))+
  geom_point(aes(color = 'Actual'))+
  geom_point(aes(x = x_value, y = predicted_value, color = 'Predicted'))+
  scale_color_manual(values = c('Actual'= 'blue', "Predicted"="red"))+
  labs(title = 'Actual Values versus Predicted values',x = '',y= 'Predicted Values')
```



Finally using the `RMSE()` function we calculated the performance metric, in this case the root mean square error. The squared discrepancies between the anticipated values and the actual values are averaged to get the MSE. The MSE in this instance is 0.2878711, which indicates that the anticipated values and actual values diverge by an average of 0.2878711 squared units. Better model performance is indicated by a lower MSE.

The standard deviation of the variations between the predicted values and the actual values is represented by the RMSE, which is the square root of the MSE. The RMSE in this instance is 36.024337, which indicates that the anticipated and actual values diverge by an average of 36.024337 units.

Better model performance is shown by a lower RMSE. The average absolute difference between expected and actual values is known as the MAE which is 23.1215372. The R2 number is the percentage of the target variable's variance that the model accounts for. You can assess the efficacy of the Random Forest approach for your particular problem by comparing these metrics to those derived from other models.

Conclusion [15 points]

Scope and generalizability of predictive analysis

We can draw the following conclusions regarding the effectiveness of the Random Forest model based on the evaluation metrics:

1. 0.2878711 is the Mean Square Error (MSE): This shows that the target variable's average squared difference between projected values and actual values is 0.2878711. This indicates that, generally speaking, the model's predictions are fairly accurate.
2. The average difference between the target variable's predicted and actual values is 36.024337, according to the Root Mean Square Error (RMSE), which is the square root of the MSE. Given the range of values in the target variable, despite this figure being a little large, it is still reasonable.

Potential limitations and possibilities for improvement.

The aforementioned random forest model might have shortcomings and room for development. The quality of the data used to train the model could be one restriction because errors, missing values, or outliers could have an impact on the model's precision. The selection of characteristics used to train the model could also have limitations because doing so could make the model less effective by using redundant or unnecessary features. For instance, changing the quantity of trees, their depth, and the number of variables chosen for each split can help to maximize the performance of the model.

Another potential drawback of the model is overfitting, which can be overcome by strategies like cross-validation, early halting, or regularization.Given that the model might not perform well on fresh datasets or populations, generalizability should also be taken into account. Given that random forest models are frequently regarded as "black-box" models, interpreting the model's predictions might be difficult as well. The most significant features can be identified using feature significance algorithms, but further study is required to make random forest models easier to understand.

Addressing these potential restrictions and assessing the model's performance on independent test datasets may be beneficial in enhancing the model's overall performance.