STAT652 Final Project

Xinyi Lu

2022-03-10

```
library(tidyverse)
library(randomForest)
library(rpart)
library(vip)
```

```
stack_overflow <- readRDS(url("https://ericwfox.github.io/data/stack_overflow.rds"))
head(stack_overflow)
```

```
## # A tibble: 6 x 21
##   country       salary years_coded_job open_source hobby company_size_nu~ remote
##   <chr>          <dbl>           <dbl> <lgl>       <lgl>            <dbl> <chr>
## 1 United Kingd~  114.               20 TRUE        TRUE             10000 Not r~
## 2 United Kingd~  100                20 FALSE       TRUE              5000 Remote
## 3 United States  130                20 TRUE        TRUE              1000 Remote
## 4 United States   82.5               3 FALSE       TRUE             10000 Not r~
## 5 United States  175                16 FALSE       TRUE             10000 Not r~
## 6 Germany         64.5               4 FALSE       FALSE             1000 Not r~
## # ... with 14 more variables: career_satisfaction <dbl>, data_scientist <lgl>,
## #   database_administrator <lgl>, desktop_applications_developer <lgl>,
## #   developer_with_stats_math_background <lgl>, dev_ops <lgl>,
## #   embedded_developer <lgl>, graphic_designer <lgl>,
## #   graphics_programming <lgl>, machine_learning_specialist <lgl>,
## #   mobile_developer <lgl>, quality_assurance_engineer <lgl>,
## #   systems_administrator <lgl>, web_developer <lgl>
```

1. Exploratory Data Analysis

```
summary(stack_overflow$salary)
```

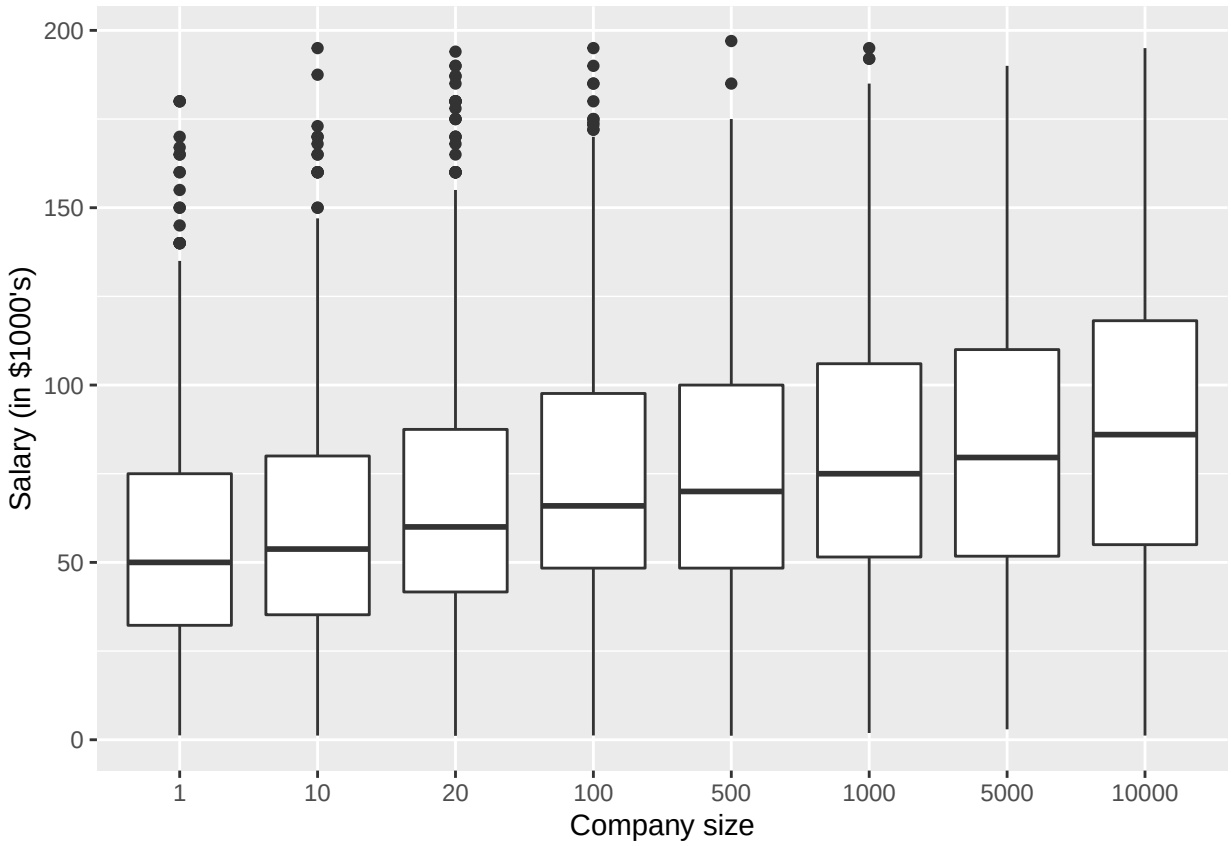
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.101  45.000  65.000  72.204 100.000 197.000
```

```
hist(stack_overflow$salary, xlab = "Salary (in thousands of dollars)",
     main = "Histogram of Salary" )
```

The response variable salary has mean 72.204, median 65, minimum 1.101 and maximum 197 in thousand dollars.

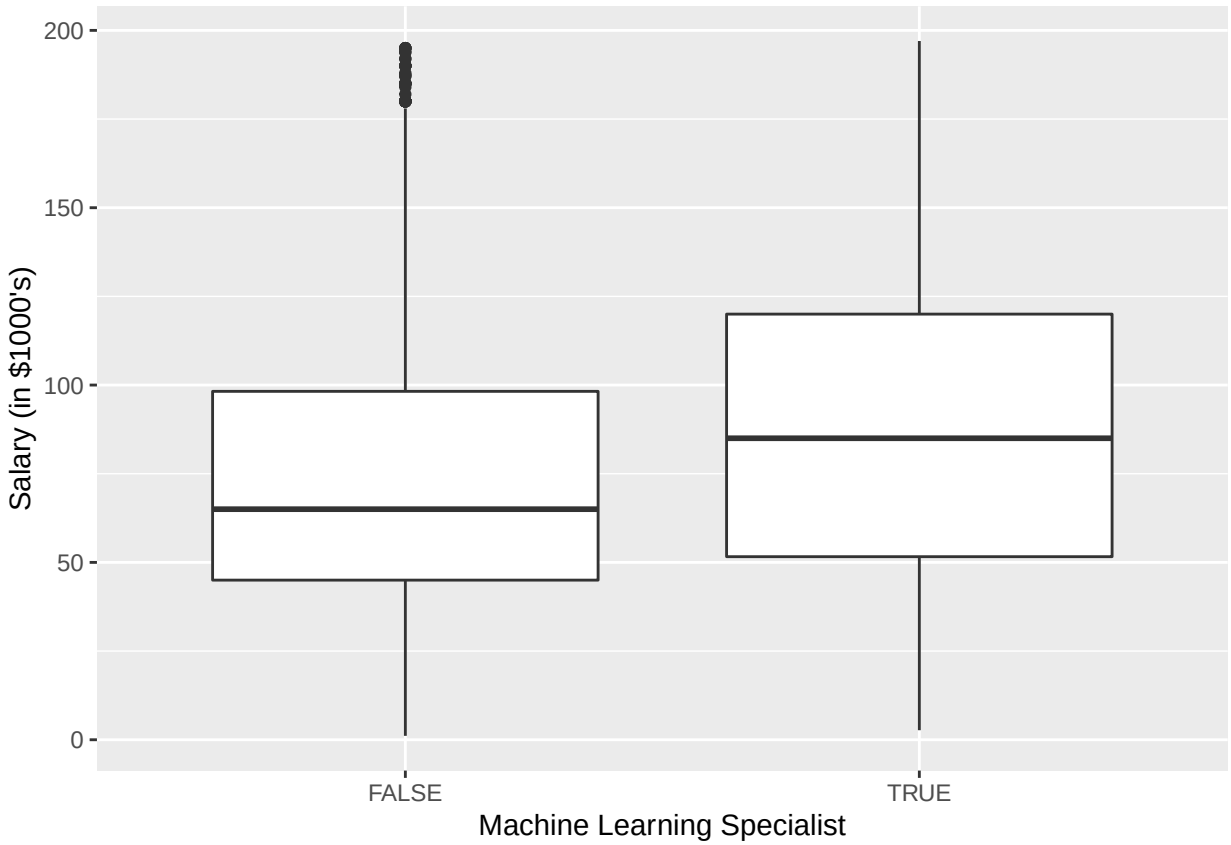
From the histogram of response variable salary, we can see an right skewed distribution with a high fequency of 0 salary. We can use log transformation for salary.

```
ggplot(stack_overflow, aes(x=factor(company_size_number), y=salary)) +
  geom_boxplot() +
  labs(x="Company size", y="Salary (in $1000's)")
```



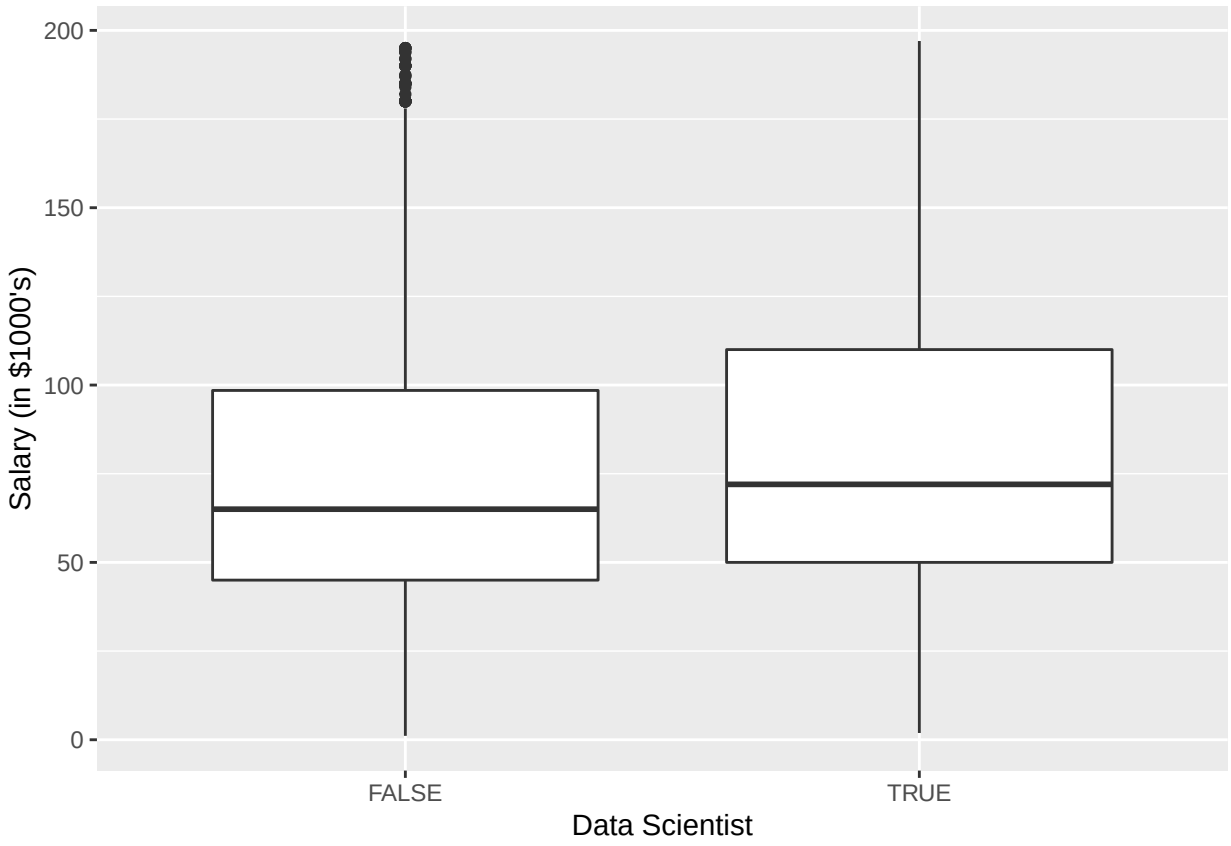
From the box plot of salary - company size, we can see that as the size of Stack Overflow developers' company increased, the median salary of Stack Overflow developer increased slightly.

```
ggplot(stack_overflow, aes(x = machine_learning_specialist, y = salary)) +
  geom_boxplot() +
  labs(x="Machine Learning Specialist", y="Salary (in $1000's)")
```



From the box plot of salary - machine learning specialist, we can see that Stack Overflow developers' who are machine learning specialist, have higher median salary than developers are not machine learning specialist, but there are salary overlap of the middle 50% between machine learning specialist and others.

```
ggplot(stack_overflow, aes(x = data_scientist, y = salary)) +
  geom_boxplot() +
  labs(x="Data Scientist", y="Salary (in $1000's)")
```



From the box plot of salary - data scientist, we can see that Stack Overflow developers' who are data scientist, have higher median salary than developers are not data scientist, but there are salary overlap of the middle 50% between data scientist and others.

2. Cross-Validation

a

Randomly split the stack_overflow data set in a 70% training and 30% test set. Make sure to use set.seed() so that your results are reproducible.

```
set.seed(123)
n <- nrow(stack_overflow)
train_index <- sample(x = 1:n, size = round(n * 0.7))
stack_overflow_train <- stack_overflow[train_index, ]
stack_overflow_test <- stack_overflow[-train_index, ]
```

b

Use lm() to fit a multiple linear regression model on the training set, with salary as the response, and all other variables as predictors. Next, use the step() function to select a reduced set of variables, and print the regression output (coefficient table) with the summary() function. Additionally, try using the vip() function to make a variable importance plot that ranks predictors according to the absolute value of the t-test statistic for each coefficient.

```
# multiple linear regression full model on training set
train_lm <- lm(salary ~ ., data = stack_overflow_train)
# summary(model_train)
length(coef(train_lm))
```

```
## [1] 24
```

```
# reduced multiple linear regression model on training set
step_train_lm <- step(train_lm, trace = F)
length(coef(step_train_lm))
```

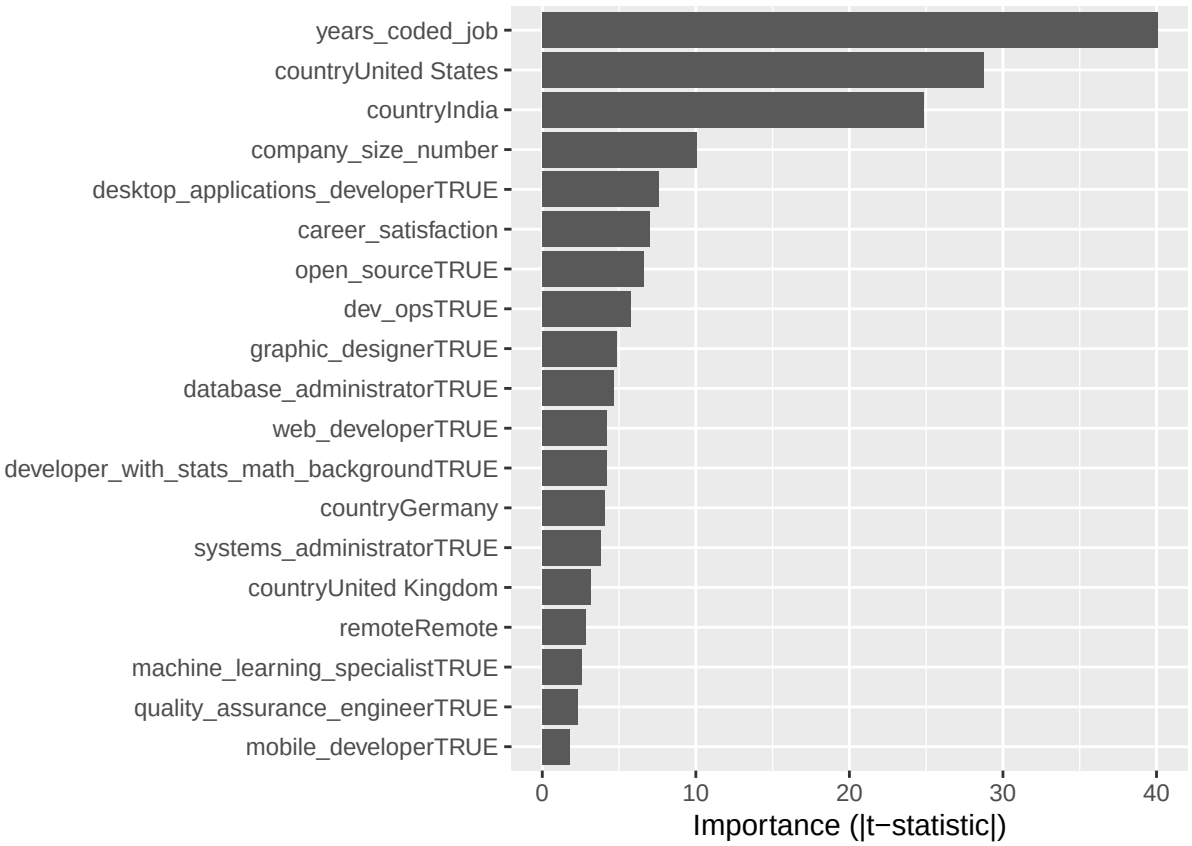
```
## [1] 20
```

```
summary(step_train_lm)
```

```
##
## Call:
## lm(formula = salary ~ country + years_coded_job + open_source +
##     company_size_number + remote + career_satisfaction + database_administrator +
##     desktop_applications_developer + developer_with_stats_math_background +
##     dev_ops + graphic_designer + machine_learning_specialist +
##     mobile_developer + quality_assurance_engineer + systems_administrator +
##     web_developer, data = stack_overflow_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -113.246  -13.408   -1.307   11.968  146.002
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             3.197e+01  2.041e+00  15.661  < 2e-16
## countryGermany                         -5.846e+00  1.437e+00  -4.069 4.79e-05
## countryIndia                           -3.953e+01  1.590e+00 -24.866  < 2e-16
## countryUnited Kingdom                  -4.308e+00  1.363e+00  -3.161 0.001581
## countryUnited States                    3.520e+01  1.224e+00  28.769  < 2e-16
## years_coded_job                         2.335e+00  5.826e-02  40.074  < 2e-16
## open_sourceTRUE                         4.719e+00  7.153e-01   6.598 4.62e-11
## company_size_number                     8.978e-04  8.903e-05  10.085  < 2e-16
## remoteRemote                            3.134e+00  1.106e+00   2.832 0.004640
## career_satisfaction                     1.402e+00  1.994e-01   7.031 2.33e-12
## database_administratorTRUE             -5.068e+00  1.087e+00  -4.662 3.21e-06
## desktop_applications_developerTRUE     -5.819e+00  7.655e-01  -7.602 3.48e-14
## developer_with_stats_math_backgroundTRUE  4.641e+00  1.111e+00   4.176 3.02e-05
## dev_opsTRUE                             6.162e+00  1.075e+00   5.731 1.06e-08
## graphic_designerTRUE                   -9.907e+00  2.045e+00  -4.844 1.31e-06
## machine_learning_specialistTRUE         5.057e+00  1.943e+00   2.603 0.009274
```

```
## mobile_developerTRUE                        1.538e+00  8.532e-01   1.802 0.071551
## quality_assurance_engineerTRUE             -4.109e+00  1.790e+00  -2.295 0.021752
## systems_administratorTRUE                  -4.639e+00  1.223e+00  -3.792 0.000151
## web_developerTRUE                          -3.258e+00  7.775e-01  -4.190 2.84e-05
##
## (Intercept)                                ***
## countryGermany                             ***
## countryIndia                               ***
## countryUnited Kingdom                      **
## countryUnited States                       ***
## years_coded_job                            ***
## open_sourceTRUE                            ***
## company_size_number                        ***
## remoteRemote                               **
## career_satisfaction                        ***
## database_administratorTRUE                 ***
## desktop_applications_developerTRUE         ***
## developer_with_stats_math_backgroundTRUE   ***
## dev_opsTRUE                                ***
## graphic_designerTRUE                       ***
## machine_learning_specialistTRUE            **
## mobile_developerTRUE                       .
## quality_assurance_engineerTRUE             *
## systems_administratorTRUE                  ***
## web_developerTRUE                          ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.14 on 4874 degrees of freedom
## Multiple R-squared:  0.6682, Adjusted R-squared:  0.6669
## F-statistic: 516.7 on 19 and 4874 DF,  p-value: < 2.2e-16
```

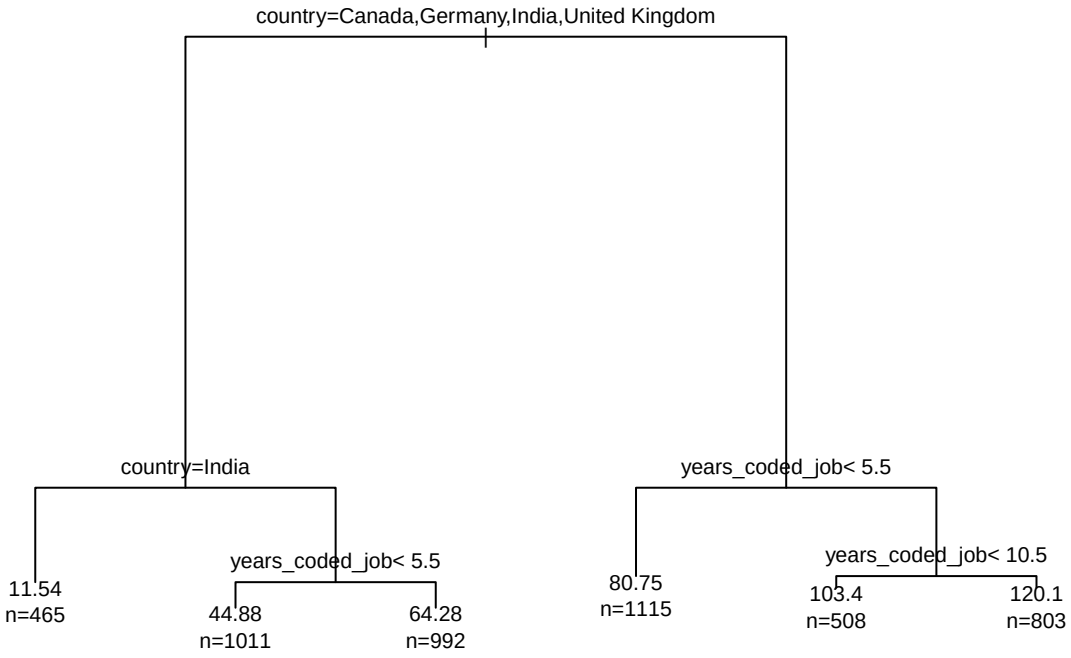
```
# variable importance plot that ranks predictors according to the absolute value of the t-test statisti
vip(step_train_lm, num_features = 19, geom = "col", include_type = TRUE)
```



c

Use rpart() to fit a regression tree on the training set, with salary as the response, and all other variables as predictors. Make a plot of the resulting regression tree. Note that if you set the argument pretty = 0 in the text() function the actual category names will be displayed in the tree.

```
# regression tree model on training set
t1 <- rpart(salary ~ . ,
            data = stack_overflow_train,
            method = "anova")
par(cex=0.7, xpd=NA)
plot(t1)
text(t1, use.n = TRUE, pretty = 0)
```



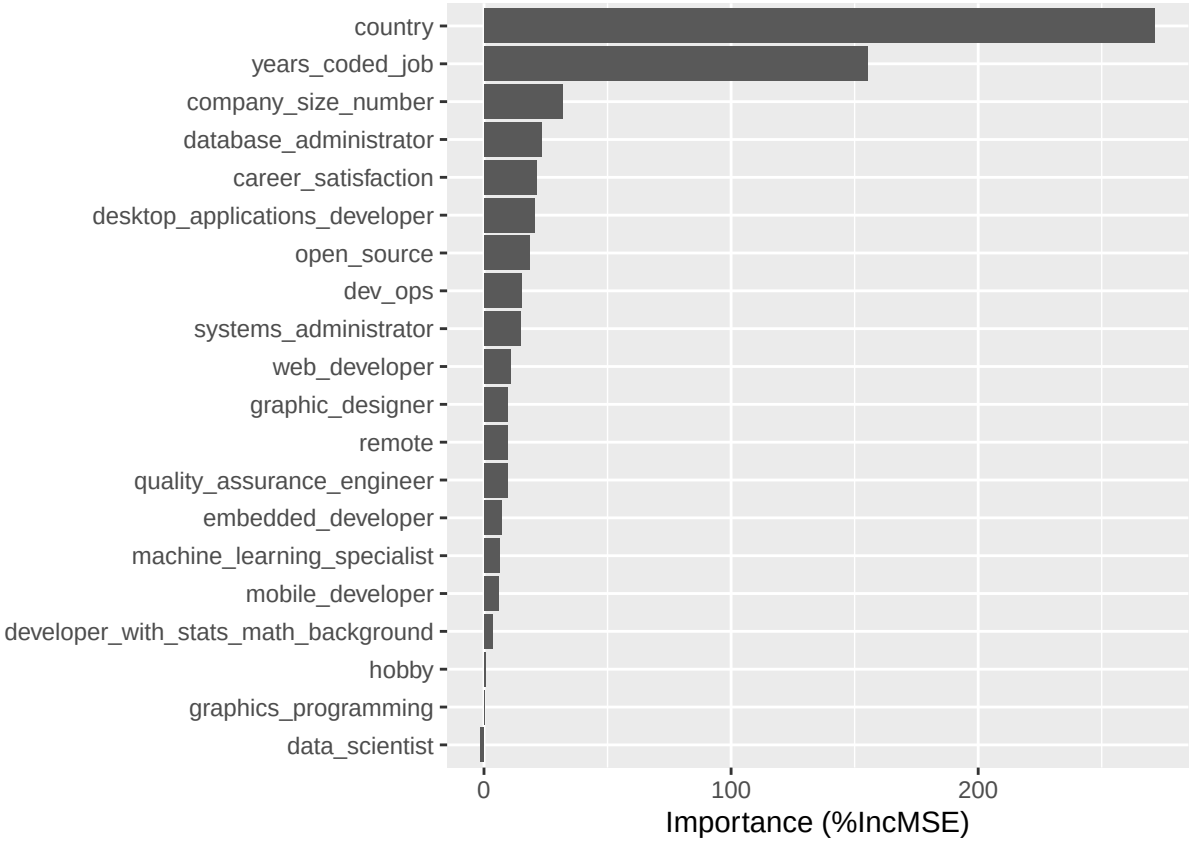
d

Use randomForest() to fit a random forest model on the training set, with salary as the response, and all other variables as predictors. Make a variable importance plot.

```
# random forest model on training set
rf2 <- randomForest(salary ~ ., data = stack_overflow_train, importance = TRUE)
rf2
```

```
##
## Call:
##  randomForest(formula = salary ~ ., data = stack_overflow_train,      importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##           Mean of squared residuals: 527.5412
##                     % Var explained: 67.17
```

```
# variable importance plot of random forest model
vip(rf2, num_features = 20, geom = "col", include_type = TRUE)
```



e

Make predictions on the test set and compute the RMSE and R^2 for the three models (multiple linear regression, regression tree, and random forests). Comment on the cross-validation results, and discuss the strengths and weaknesses of each model in terms of predictive performance and interpretability.

```
# functions of compute RMSE and R^2
RMSE <- function(y, y_hat) {
  sqrt(mean((y - y_hat)^2))
  }
r2 <- function(y, y_hat) {
  1-(sum((y - y_hat)^2) / sum((y - mean(y))^2))
  }
```

```
# stepwise multiple linear regression
pred1 <- predict(step_train_lm, newdata = stack_overflow_test)
RMSE1 <- RMSE(stack_overflow_test$salary, pred1)
R2_1 <- r2(stack_overflow_test$salary, pred1)
cor(stack_overflow_test$salary, pred1)^2
```

```
## [1] 0.6710302
```

```
# regression tree
pred2 <- predict(t1, newdata = stack_overflow_test)
RMSE2 <- RMSE(stack_overflow_test$salary, pred2)
```

```r
R2_2 <- r2(stack_overflow_test$salary, pred2)
cor(stack_overflow_test$salary, pred2)^2
```

```
## [1] 0.6330377
```

```r
# random forests
pred3 <- predict(rf2, newdata = stack_overflow_test)
RMSE3 <- RMSE(stack_overflow_test$salary, pred3)
R2_3 <- r2(stack_overflow_test$salary, pred3)
cor(stack_overflow_test$salary, pred3)^2
```

```
## [1] 0.6747733
```

```r
data.frame(models = c("Stepwise MLR", "Regression Tree", "Random Forests"),
           RMSE = c(RMSE1, RMSE2, RMSE3), R_square = c(R2_1, R2_2, R2_3))
```

```
##            models     RMSE  R_square
## 1    Stepwise MLR 23.00178 0.6709627
## 2 Regression Tree 24.29320 0.6329781
## 3  Random Forests 22.89302 0.6740667
```

From the above table, random forests model has lowest RMSE and highest R square, so it is the best preformence model. For the random forests model, the interpretation of the RMSE is that, when applied to withheld data, the predictions for salary are, on average, about $23,000 off from the actual Sale_Price. That is, the average error in predicting Sale_Price is about $23,000. The interpretation of R square is that about 67% of the variability in salary, on the test set, is explained by predictions from the linear regression model.

Strengths and weaknesses of each model:

Random Forest Regression is most accurate in these three models, while Multiple Linear Regression and Regression Tree are earier to interpretate.

The stepwise multiple linear regression use AIC criterion, and it has rigorous theoretical justification in information theory. It does the variable selection and selects the model that performs the best according to its criterion. Using MLR with fewer variables is easier to know the relationship between predictors and response, or interpret and explain. However, sometime a large number of predictors may overfit the data and perform poorly when making predictions for new, or future values of the response variable.

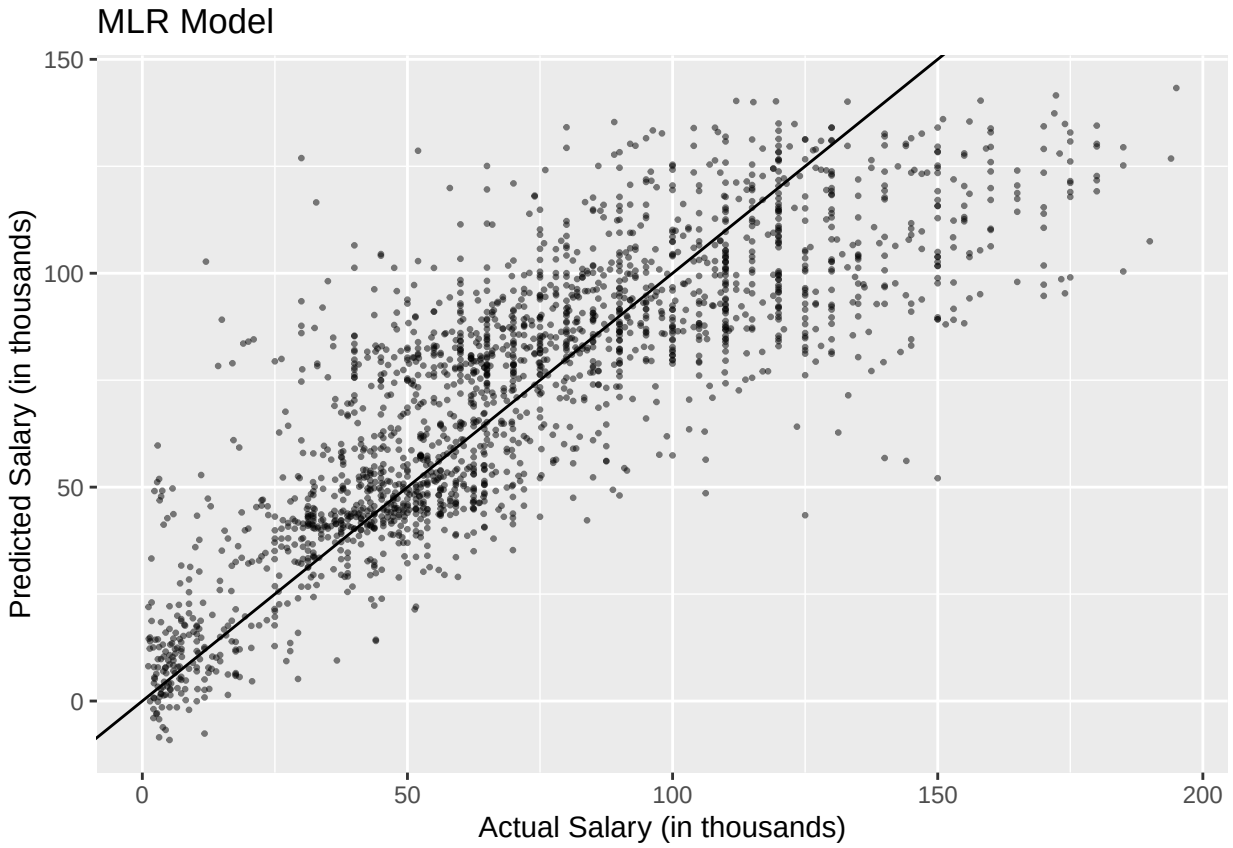
Regression Tree are simple and useful for interpretation. It is cheaper to collect data on fewer variables. In this caes, it shows the most important predictors in the model, which are "country" and "years_coded_job", but it typically is not competitive with the best statistical learning methods in terms of prediction accuracy.

Random Forest Regression is a robust algorithmapproach. It produces better results, work well on large datasets and it can often result in dramatic improvements in prediction accuracy, at the expense of some loss in interpretation. Unlike linear regression, or individual decision trees, we do not know what the relationships between the response and predictor variables. We can use variable importance plots to interpretate the random forests.

f

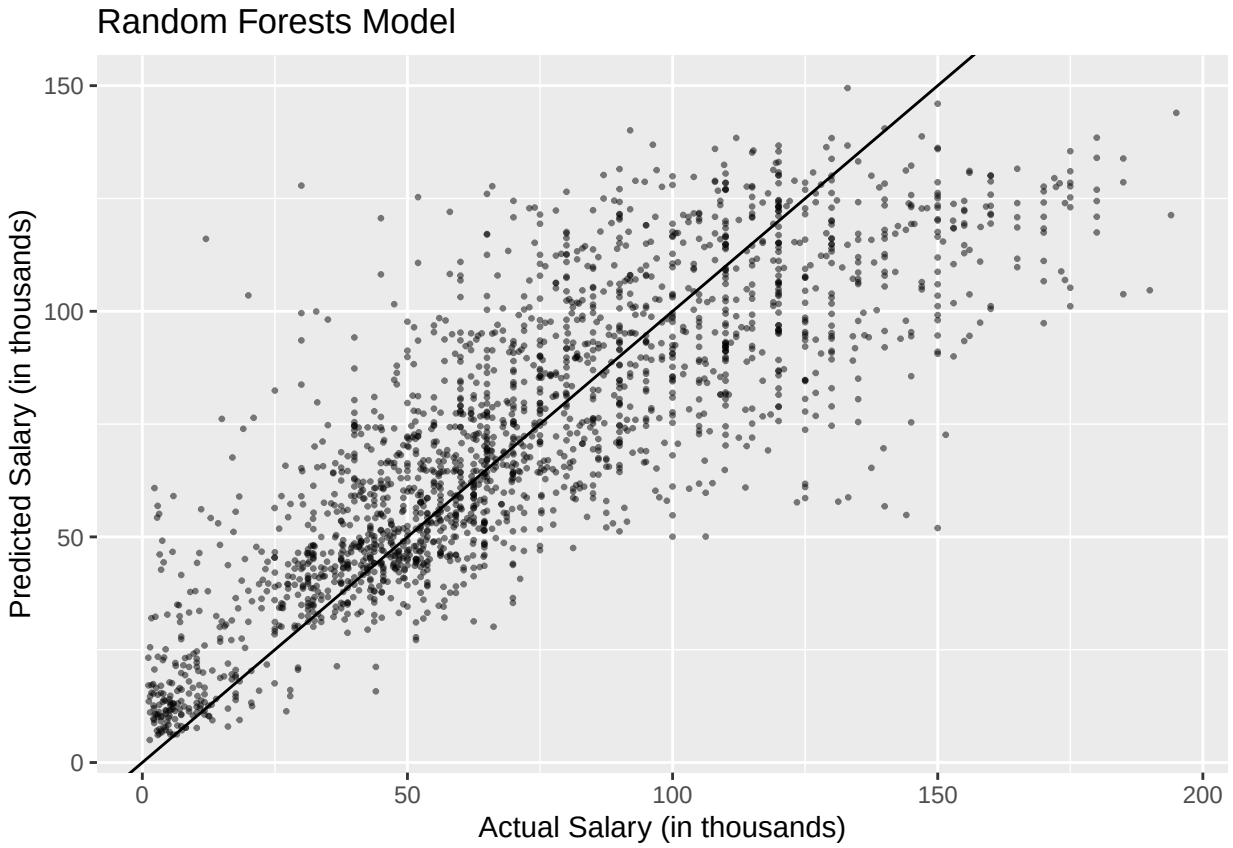
Make plots of the predicted versus actual values on the test set for each of the three models; add the 1-1 reference line to each plot. Comment on why the patterns in the plot of the predicted versus actual values for the regression tree model look different than the random forest and linear regression models?

```
ggplot(aes(salary, pred1), data = stack_overflow_test) +
  geom_point(size = 0.5, alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary (in thousands)") +
  ylab("Predicted Salary (in thousands)") +
  ggtitle("MLR Model")
```




```
ggplot(aes(salary, pred2), data = stack_overflow_test) +
  geom_point(size = 0.5, alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary (in thousands)") +
  ylab("Predicted Salary (in thousands)") +
  ggtitle("Regression Tree Model")
```


```
ggplot(aes(salary, pred3), data = stack_overflow_test) +
  geom_point(size = 0.5, alpha = 0.5) +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary (in thousands)") +
  ylab("Predicted Salary (in thousands)") +
  ggtitle("Random Forests Model")
```



The plots of the predicted versus actual values on the test set, with the 1-1 line superimposed. We can see that the plot of Random Forests and stepwise MLR are nearly the same.

The outputs of Regression Tree are discrete. For each group of predicted salary, the mean was calculated, while the actualy salary in testing set is not discrete. MLR and Random Forests' outputs are continuous, that's why the plots of Regreesion Tree and another two are different.