predict-dev-salary-on-stack-overflow

gaargly

2023-01-30

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

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

1. Exploratory Data Analysis

We see that our response variable salary has a mean value of 72.204 (in thousands) with a standard deviation of 40.093 (in thousands).

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

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

```
sd(stack_overflow$salary)
```

```
## [1] 40.09345
```

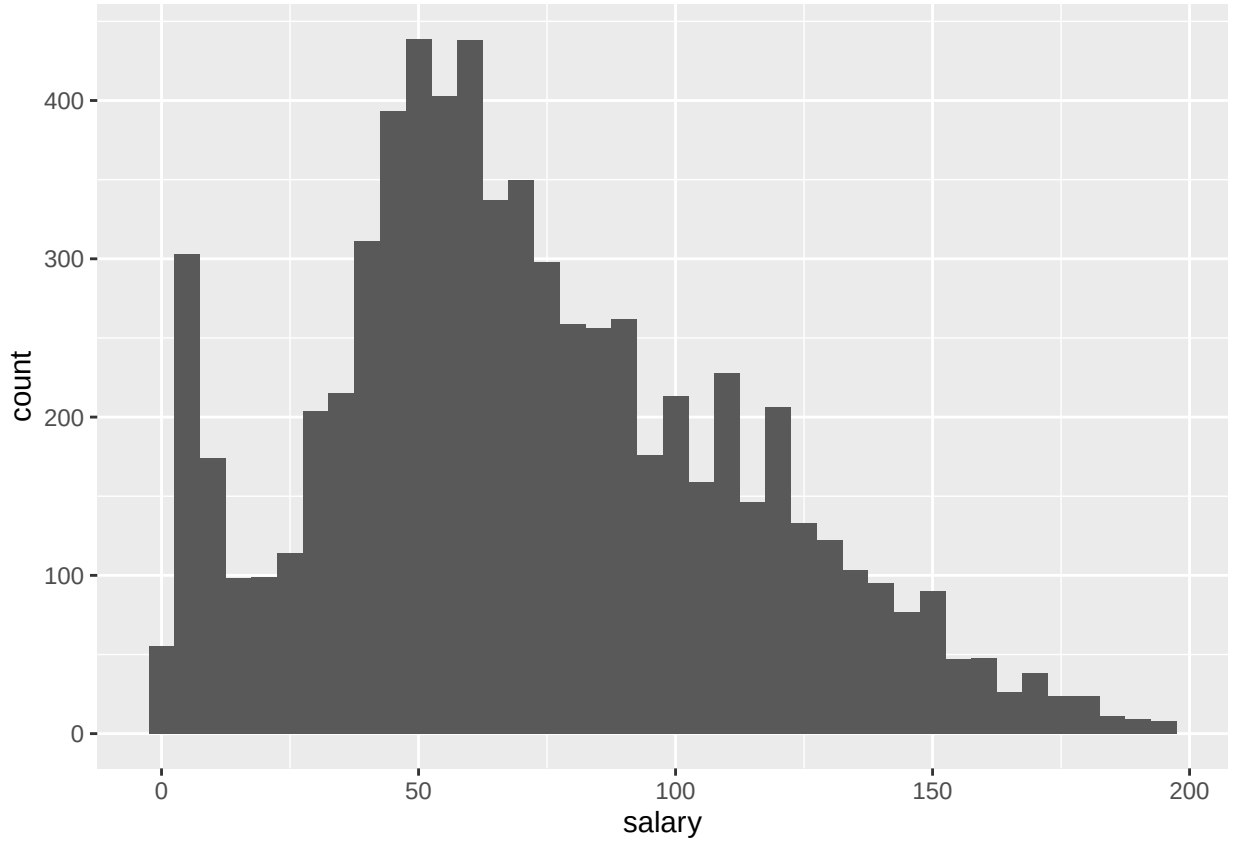
We also observe that there are 6991 values of salary with 1104 distinct values. Below, some additional percentiles of salary are also reported.

```
describe(stack_overflow$salary)
```

```
## stack_overflow$salary
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     6991        0     1104        1     72.2    45.22    7.341   20.682
##      .25      .50      .75      .90      .95
##   45.000   65.000  100.000  130.000  145.000
##
## lowest :   1.101   1.136   1.175   1.211   1.233
## highest: 192.000 193.750 194.000 195.000 197.000
```

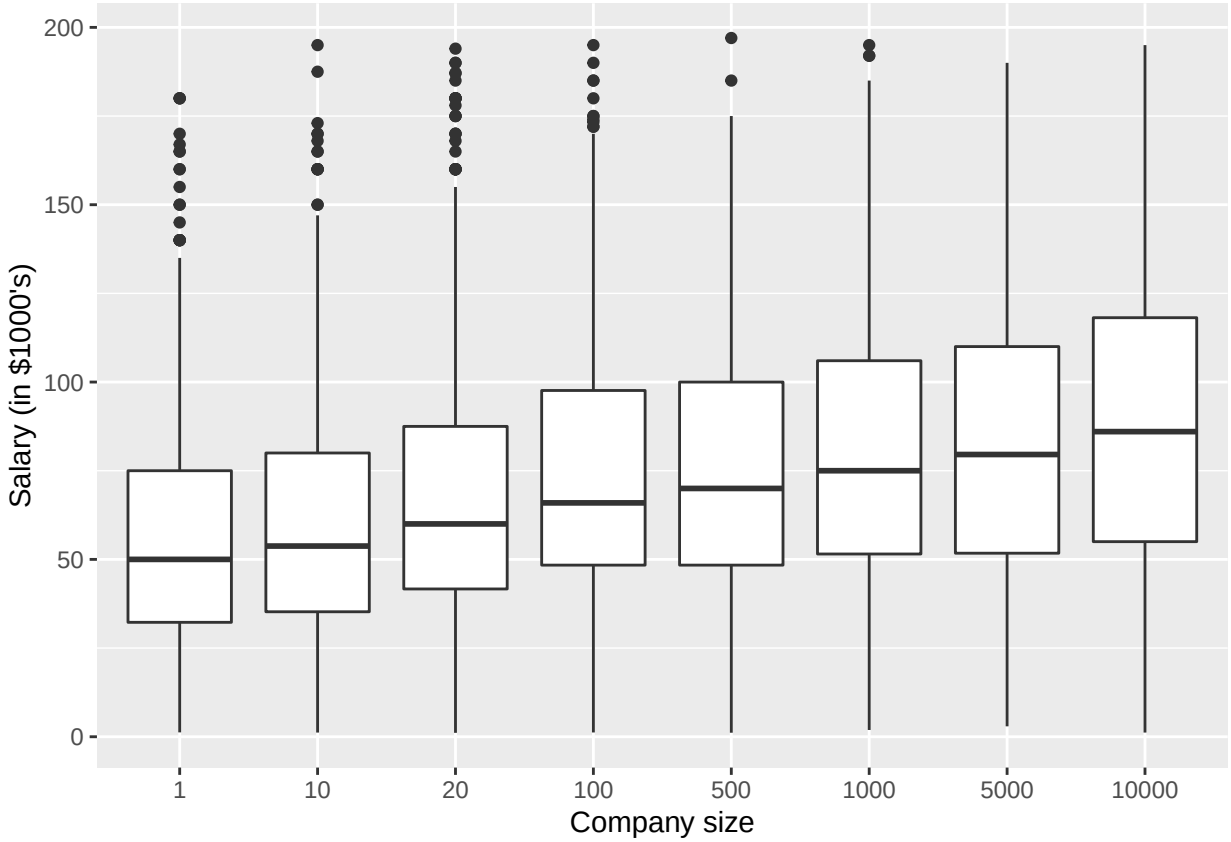
Per the below histogram, we see that our response variable salary has a right skew.

```
ggplot(stack_overflow, aes(x=salary)) + geom_histogram(binwidth=5)
```



It looks like bigger companies, i.e. companies with more employees, offer higher salaries on average, though there are outliers with very high salaries in companies with under 10,000 employees.

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

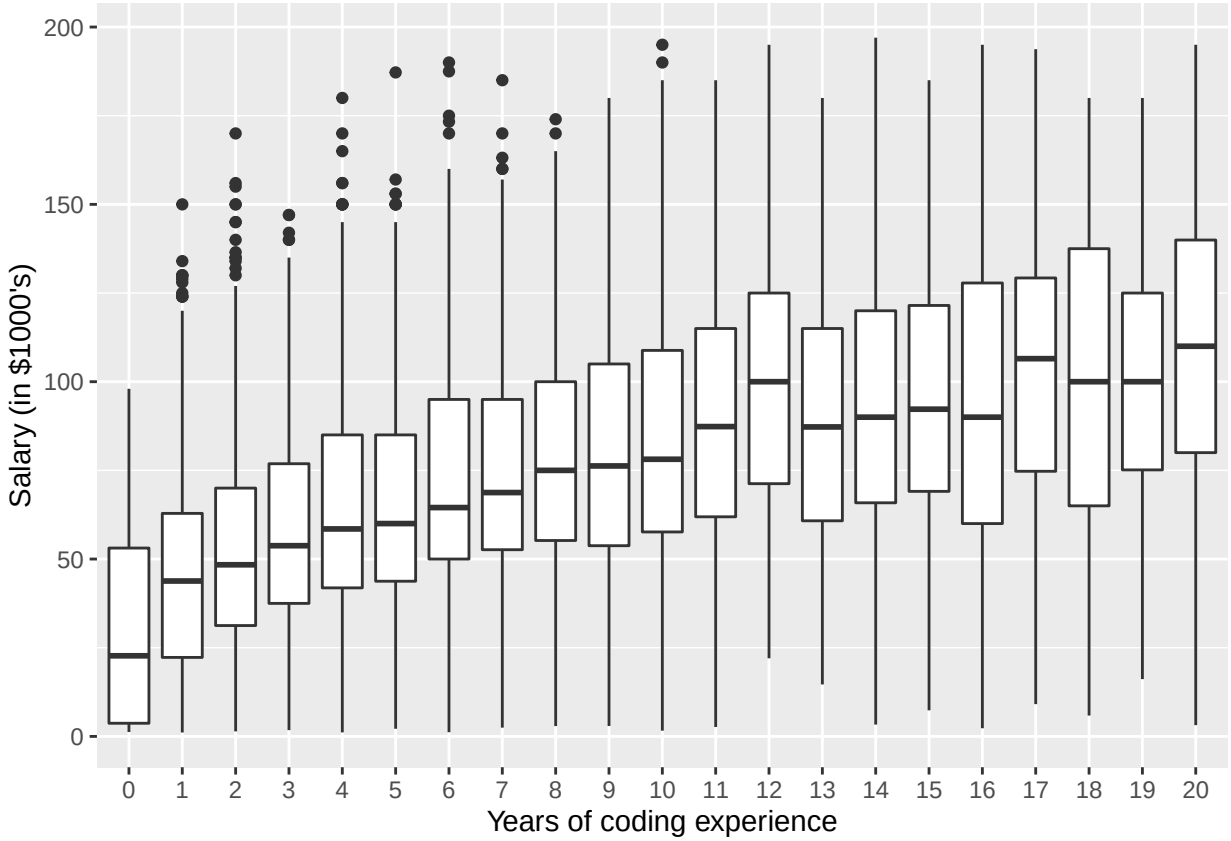



```
# Take a look at variables in our dataset
glimpse(stack_overflow)
```

```
## Rows: 6,991
## Columns: 21
## $ country                                <chr> "United Kingdom", "United Kingdom~
## $ salary                                 <dbl> 113.750, 100.000, 130.000, 82.500~
## $ years_coded_job                        <dbl> 20, 20, 20, 3, 16, 4, 1, 1, 20, 2~
## $ open_source                            <lgl> TRUE, FALSE, TRUE, FALSE, FALSE, ~
## $ hobby                                  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, FAL~
## $ company_size_number                    <dbl> 10000, 5000, 1000, 10000, 10000, ~
## $ remote                                 <chr> "Not remote", "Remote", "Remote",~
## $ career_satisfaction                    <dbl> 8, 8, 9, 5, 7, 9, 5, 8, 8, 10, 7,~
## $ data_scientist                         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ database_administrator                 <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ desktop_applications_developer         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ developer_with_stats_math_background   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ dev_ops                                <lgl> FALSE, FALSE, TRUE, FALSE, FALSE,~
## $ embedded_developer                     <lgl> FALSE, TRUE, TRUE, FALSE, FALSE, ~
## $ graphic_designer                       <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ graphics_programming                   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ machine_learning_specialist            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ mobile_developer                       <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ quality_assurance_engineer             <lgl> FALSE, FALSE, TRUE, FALSE, FALSE,~
## $ systems_administrator                  <lgl> FALSE, FALSE, FALSE, FALSE, FALSE~
## $ web_developer                          <lgl> FALSE, FALSE, TRUE, TRUE, TRUE, T~
```

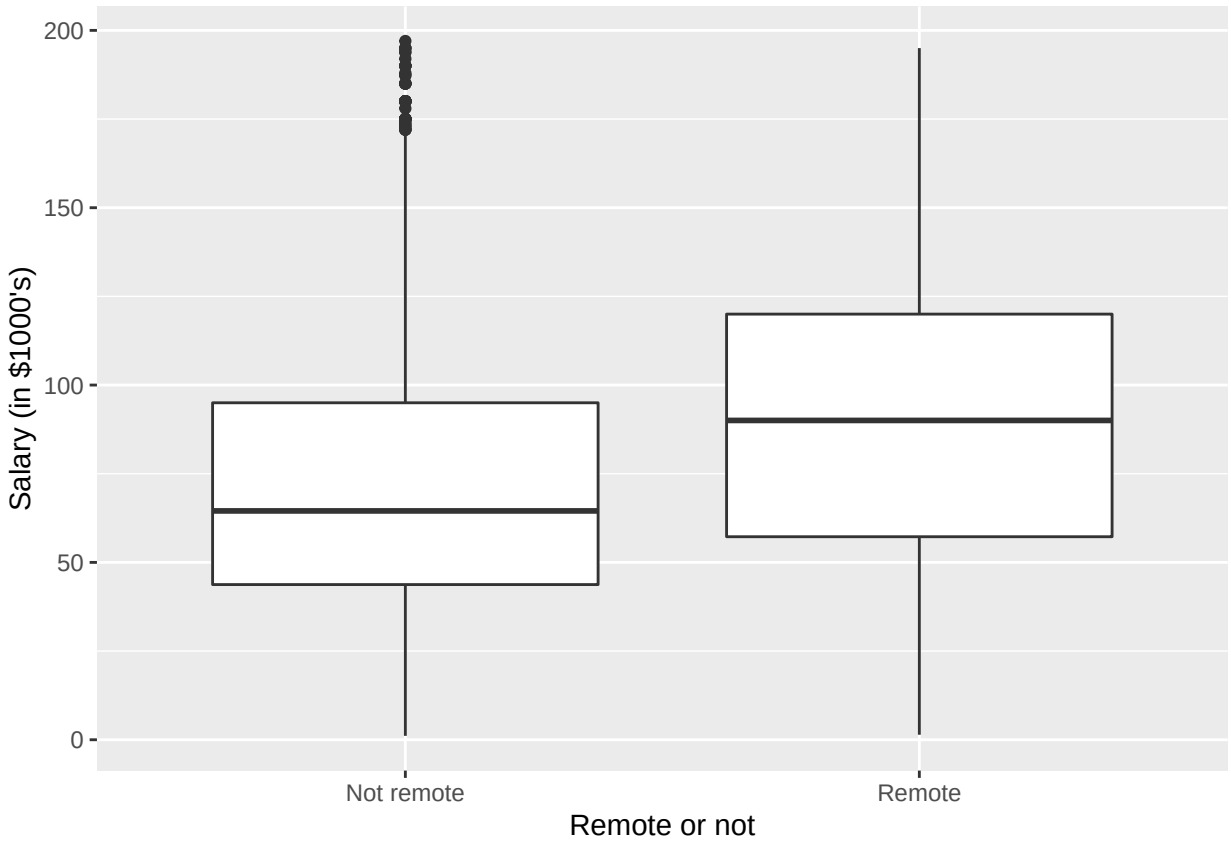
Years of coding experience generally correlates with higher salaries until year 13, at which point there is a surprising dip in average salaries until year 17 at which point salaries increase again. However, there is a dip in 18 and 19 years of experience, with an increase again at year 20. There are some outliers with very high salaries for those with 1 to 8 years of coding experience as well as 10 years of coding experience.

```
# Years of coding experience vs. Salary
ggplot(stack_overflow, aes(x = factor(years_coded_job), y = salary)) +
  geom_boxplot() +
  labs(x = "Years of coding experience", y = "Salary (in $1000's)")
```



Remote jobs offer higher salaries than non-remote jobs, though there are some outliers with very high salaries who work non-remote jobs.

```
# Remote or not vs. Salary
ggplot(stack_overflow, aes(x = factor(remote), y = salary)) +
  geom_boxplot() +
  labs(x = "Remote or not", y = "Salary (in $1000's)")
```



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(1)
n <- nrow(stack_overflow)
train_index <- sample(1:n, round(0.7*n))
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.

```
# Check dimensions of our dataset to know how many predictors we'll be working with (looks like 20 pred
dim(stack_overflow_train)
```

```
## [1] 4894   21
```

```r
# Fit MLR model with salary as the response and all other variables as predictors
lm_fit <- lm(salary ~ ., data = stack_overflow_train)

# Use step() function to select a reduced set of variables (backwards stepwise selection using the AIC)
lm_step_fit <- step(lm_fit, trace=F)

# Print the regression output (coefficient table)
summary(lm_step_fit)
```

```
##
## 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
## -114.769  -13.363   -1.217   11.945  146.770
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             3.199e+01  2.054e+00  15.576  < 2e-16
## countryGermany                         -5.859e+00  1.415e+00  -4.142 3.51e-05
## countryIndia                           -3.897e+01  1.583e+00 -24.619  < 2e-16
## countryUnited Kingdom                  -4.069e+00  1.354e+00  -3.006 0.002656
## countryUnited States                    3.560e+01  1.206e+00  29.516  < 2e-16
## years_coded_job                         2.363e+00  5.817e-02  40.614  < 2e-16
## open_sourceTRUE                         3.054e+00  7.137e-01   4.279 1.92e-05
## company_size_number                     8.955e-04  8.872e-05  10.094  < 2e-16
## remoteRemote                            3.521e+00  1.125e+00   3.130 0.001757
## career_satisfaction                     1.456e+00  2.000e-01   7.276 3.97e-13
## database_administratorTRUE             -5.148e+00  1.079e+00  -4.773 1.87e-06
## desktop_applications_developerTRUE     -5.415e+00  7.641e-01  -7.087 1.57e-12
## developer_with_stats_math_backgroundTRUE  4.042e+00  1.080e+00   3.744 0.000183
## dev_opsTRUE                             6.006e+00  1.062e+00   5.658 1.62e-08
## graphic_designerTRUE                   -8.710e+00  2.015e+00  -4.323 1.57e-05
## machine_learning_specialistTRUE         8.731e+00  1.929e+00   4.526 6.14e-06
## mobile_developerTRUE                    1.238e+00  8.586e-01   1.442 0.149253
## quality_assurance_engineerTRUE         -4.372e+00  1.809e+00  -2.418 0.015655
## systems_administratorTRUE              -4.789e+00  1.199e+00  -3.995 6.58e-05
## web_developerTRUE                      -4.024e+00  7.749e-01  -5.193 2.16e-07
##
## (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: 23.07 on 4874 degrees of freedom
## Multiple R-squared:  0.6702, Adjusted R-squared:  0.6689
## F-statistic: 521.2 on 19 and 4874 DF,  p-value: < 2.2e-16
```

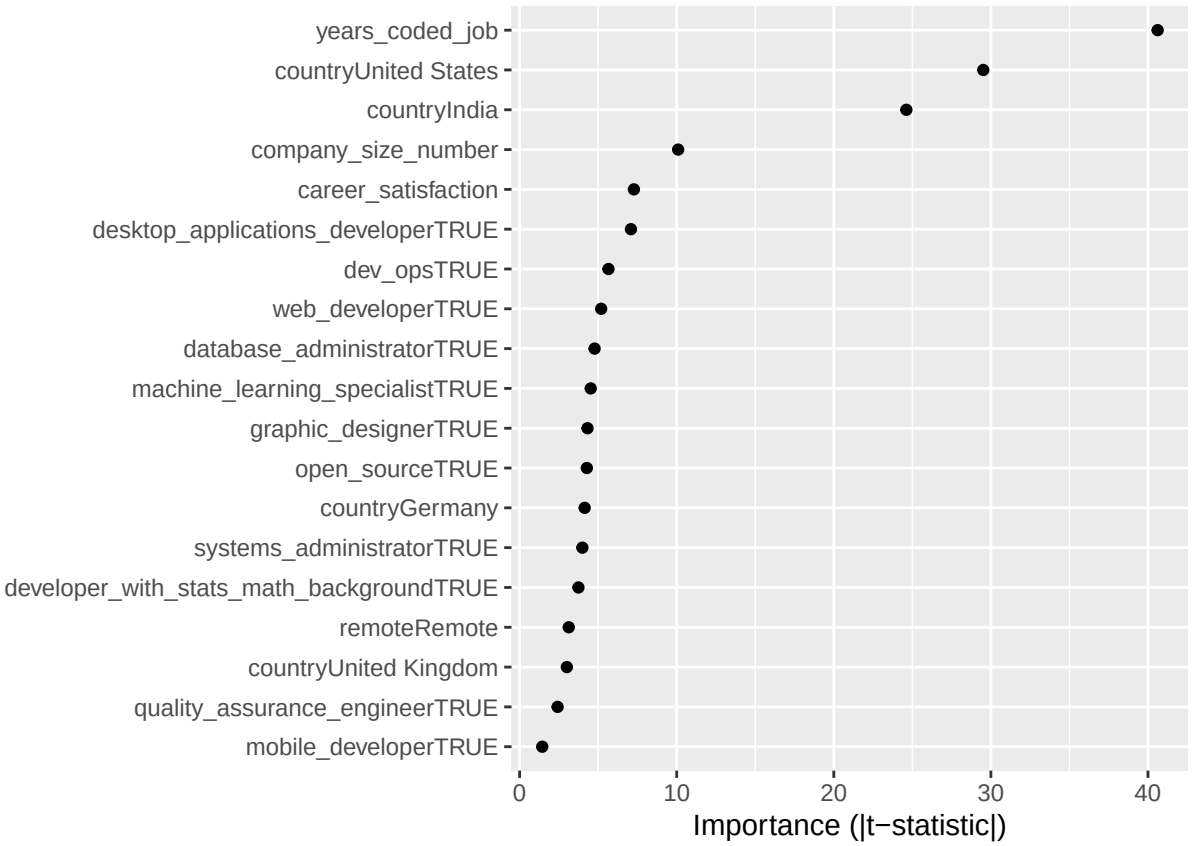
```
# Check number of coefficients
length(coef(lm_step_fit))
```

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

The model selected by backwards stepwise selection using the AIC has 20 coefficients, i.e. it has 19 predictors.

From our Variable Important Plot, we see that the years of coding experience is the most important predictor, with whether the developer is in the U.S. being the second-most important predictor, and whether the developer is in India being the third-most important predictor.

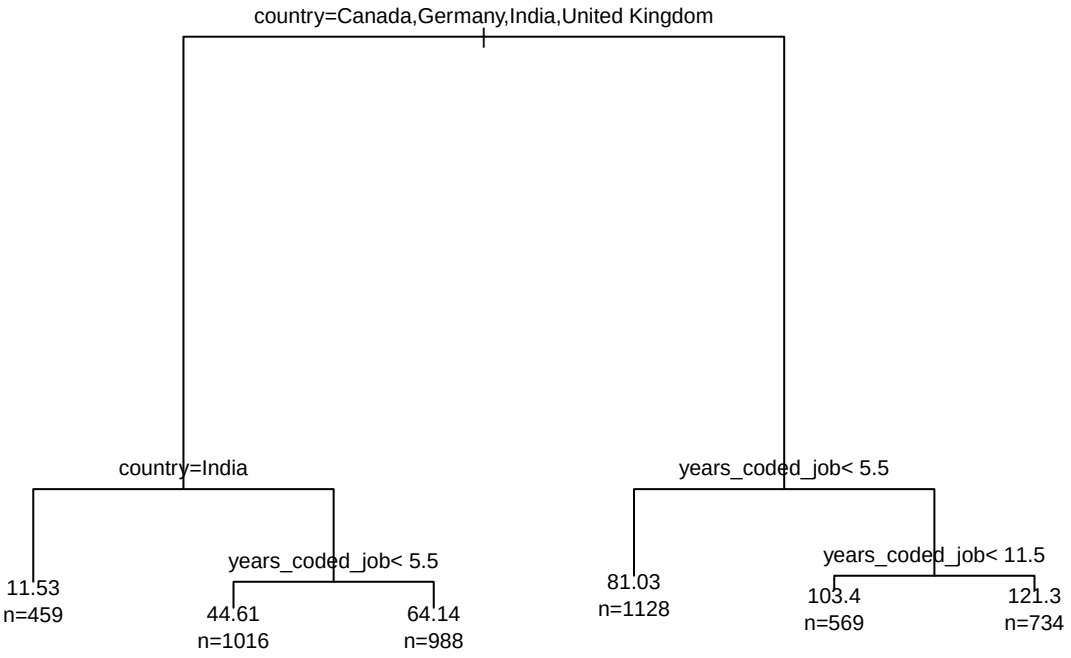
```
# Use vip() function to make a variable importance plot
vip(lm_step_fit, num_features = 19, geom = "point", include_type = TRUE)
```



(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.

```
# Fit regression tree model with salary as the response and all other variables as predictors
t1 <- rpart(salary ~ ., data = stack_overflow_train, method = "anova")
```

```
# Plot the regression tree
par(cex=0.7, xpd=NA)
plot(t1)
text(t1, use.n = TRUE, pretty = 0)
```



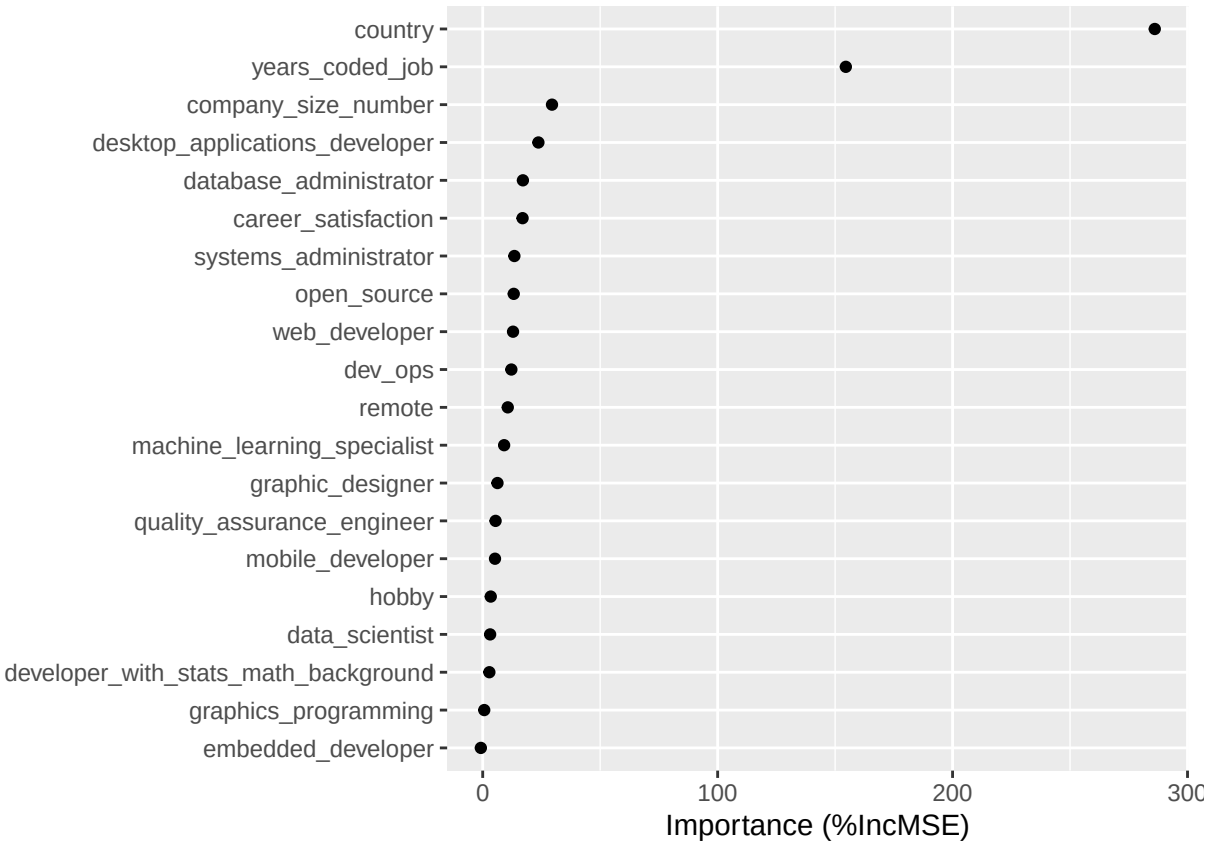
(d) Use randomForest() to fit a random forest model on the training set, with salary as the response, and all other variables as predictors. Make a variable importance plot.

```
# Use randomForest() to fit a random forest model on the training set, with salary as the response, and
rf1 <- randomForest(salary ~ ., data = stack_overflow_train, importance = TRUE)
rf1
```

```
##
## 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: 536.7693
##                     % Var explained: 66.61
```

Our Variable Importance Plot for our random forest model on the training set ranks country as the most important predictor with number of years of coding as the second most important predictor.

```
# Variable Importance Plot for for our random forest model on the training set
vip(rf1, num_features = 20, geom = "point", 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.

RMSE

```
## Make predictions on test set and compute RMSE
pred_lm <- predict(lm_step_fit, newdata = stack_overflow_test) # MLR model with backwards stepwise sele
pred_t1 <- predict(t1, newdata = stack_overflow_test) # Regression Tree model
pred_rf1 <- predict(rf1, newdata = stack_overflow_test) # Random Forest model

# Function to compute RMSE
RMSE <- function(y, y_hat) {
  sqrt(mean((y - y_hat)^2))
}

# Report RMSE of the MLR model chosen by backwards stepwise selection using the AIC
RMSE(stack_overflow_test$salary, pred_lm)
```

```
## [1] 23.13129
```

```
# Report RMSE of Regression Tree model
RMSE(stack_overflow_test$salary, pred_t1)
```

```
## [1] 24.1949
```

```
# Report RMSE of Random Forest model
RMSE(stack_overflow_test$salary, pred_rf1)
```

```
## [1] 22.72584
```

R^2

```
# Report adjusted R^2 of the MLR model chosen by backwards stepwise selection using the AIC
summary(lm_step_fit)$adj
```

```
## [1] 0.6688679
```

```
# Report R^2 of Regression Tree model
cor(stack_overflow_test$salary, pred_t1)^2
```

```
## [1] 0.6355715
```

```
# Report R^2 of Random Forest model
cor(stack_overflow_test$salary, pred_rf1)^2
```

```
## [1] 0.6786899
```

Comment on the cross-validation results, and discuss the strengths and weaknesses of each model in terms of predictive performance and interpretability.

MLR model: RMSE ~23.13, Adjusted-R^2 ~0.6689 which means ~67% of the variability in salary can be explained by the model Regression tree model: RMSE ~24.19, R^2 ~0.6356 which means ~64% of the variability in salary can be explained by the model Random forest model: RMSE ~22.73, R^2 ~0.6787 which means ~68% of the variability in salary can be explained by the model

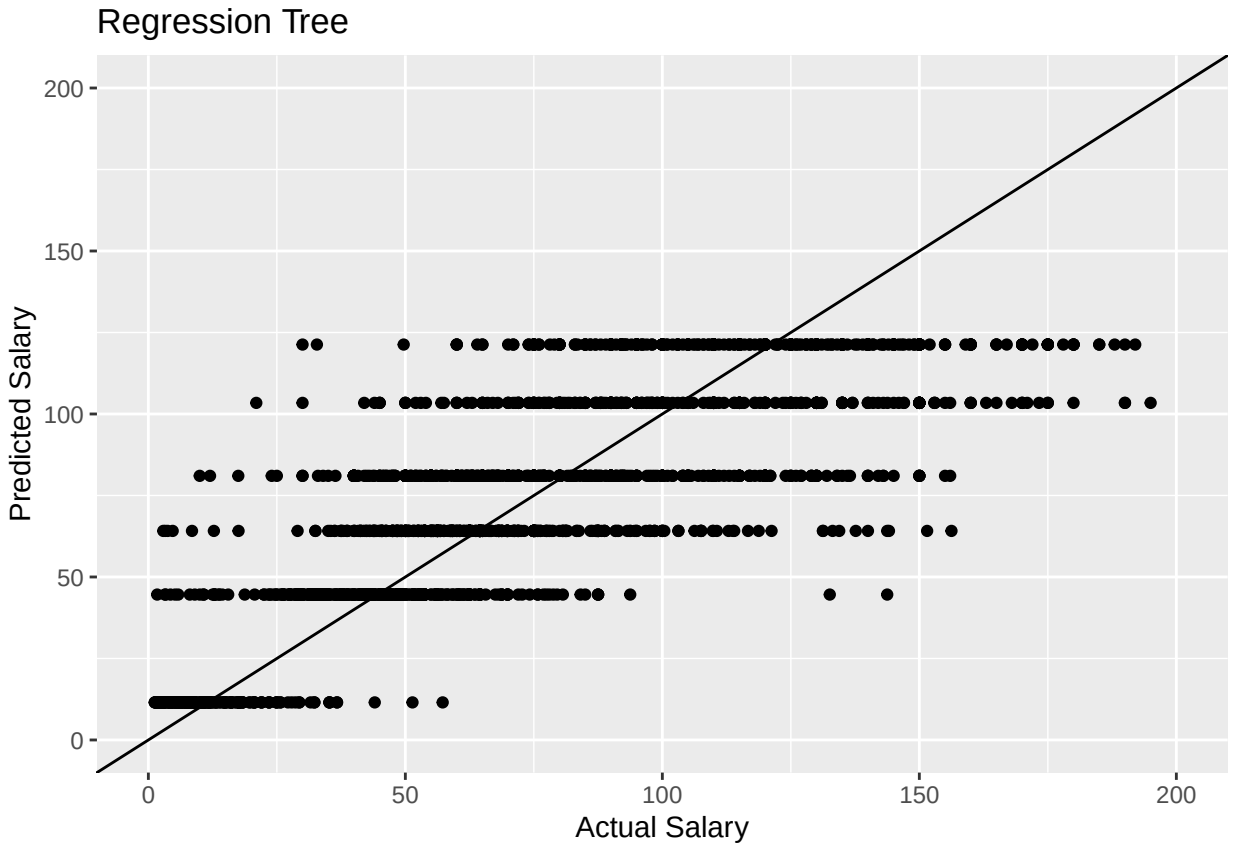
Each model offers different strengths and weaknesses in terms of predictive performance and interpretability. We notice that our random forest model has the lowest RMSE and the highest R^2 compared to our MLR and regression tree models. Although random forest models are useful when we have many predictors, interpretability is difficult because we are using bootstrap to produce multiple trees. On the other hand, our MLR model is easier to interpret as we can describe the effect of each predictor on the response with all other predictors held constant. Though with such a high number of predictors (19), we may be overfitting our data. We can consider options such as ridge or lasso regression to improve our variance-bias tradeoff. Our regression tree model is fairly easy to interpret as we can describe the splits in the tree fairly easily and demonstrate the relative importance of predictors visually, but we have the highest RMSE and lowest R^2 for this model.

(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.

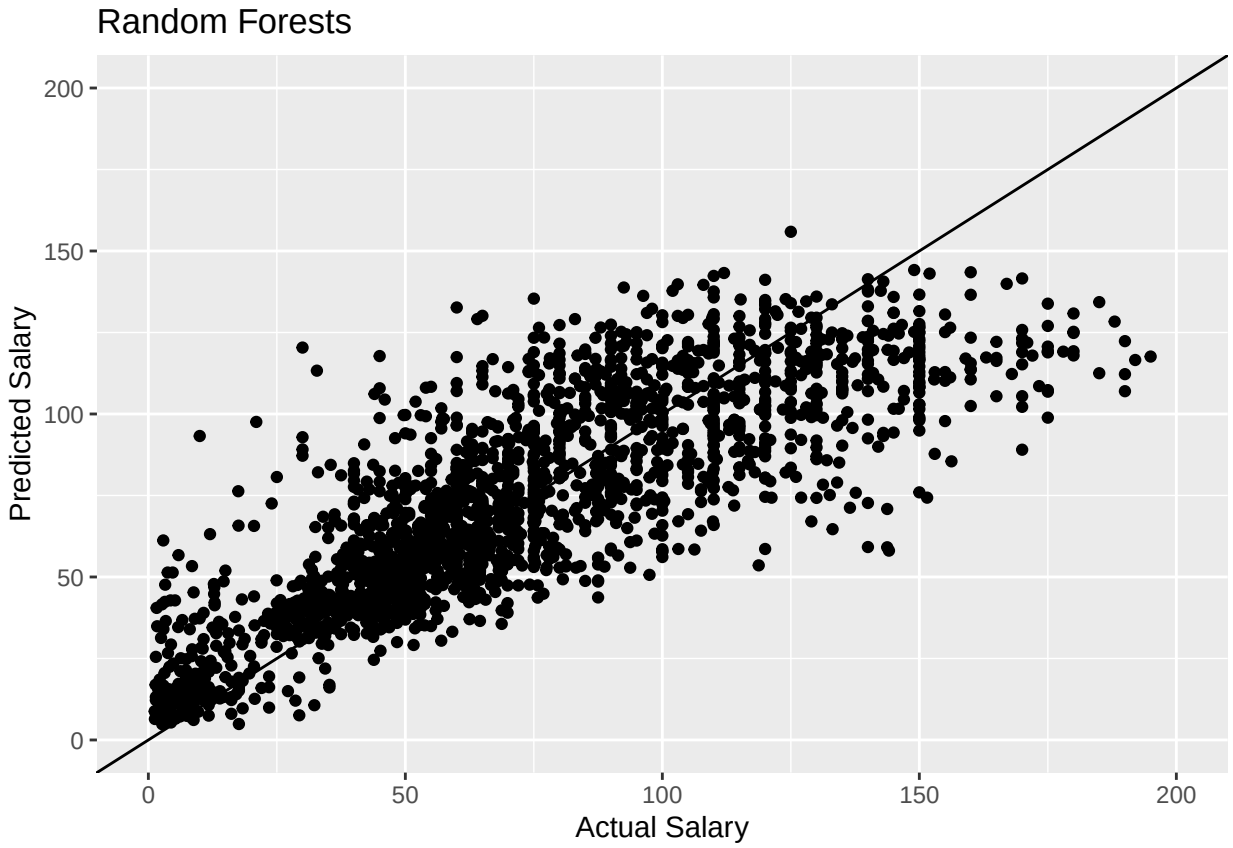
```r
pred_df <- data.frame(
  Actual = stack_overflow_test$salary,
  Pred_LM = pred_lm,
  Pred_T = pred_t1,
  Pred_RF = pred_rf1
)

# MLR
ggplot(pred_df, aes(x = Actual, y = Pred_LM)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary") + ylab("Predicted Salary") +
  ggtitle("Multiple Linear Regression") +
  xlim(0,200) + ylim(0,200)
```


```r
# Regression Tree
ggplot(pred_df, aes(x = Actual, y = Pred_T)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary") + ylab("Predicted Salary") +
  ggtitle("Regression Tree") +
  xlim(0,200) + ylim(0,200)
```




```
# Random Forests
ggplot(pred_df, aes(x = Actual, y = Pred_RF)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) +
  xlab("Actual Salary") + ylab("Predicted Salary") +
  ggtitle("Random Forests") +
  xlim(0,200) + ylim(0,200)
```



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?

We see a more typical scatterplot around the 1-1 reference line for our multiple linear regression model and random forest model. However, since the regression tree produces the mean of the response for all observations which fall under the terminal nodes, we see a different pattern in the plot of the predicted versus actual values for this model. The predicted values are not continuous and instead have step-like jumps in this plot for the regression tree.