Regression Methods

Ashwin Bharathwaj Sekar

This projects uses a dataset which contains designation levels & salaries of employees. I will be implementing various regression methods to fit a line to the data, in order to understand the salary expectation of one prospect who has a few years of experience after becoming a Region Manager in a bank

The focus of this project lies in the implementation of the models using R packages & visualizing the predicted values as compared to the original data points. So, I have chosen a simple dataset. In an actual life cycle of a model building, there are hundreds of variables. The pre-processing phase includes treatment of outliers & missing values, correlation & multi-collinearity checks, information value based feature selection etc.

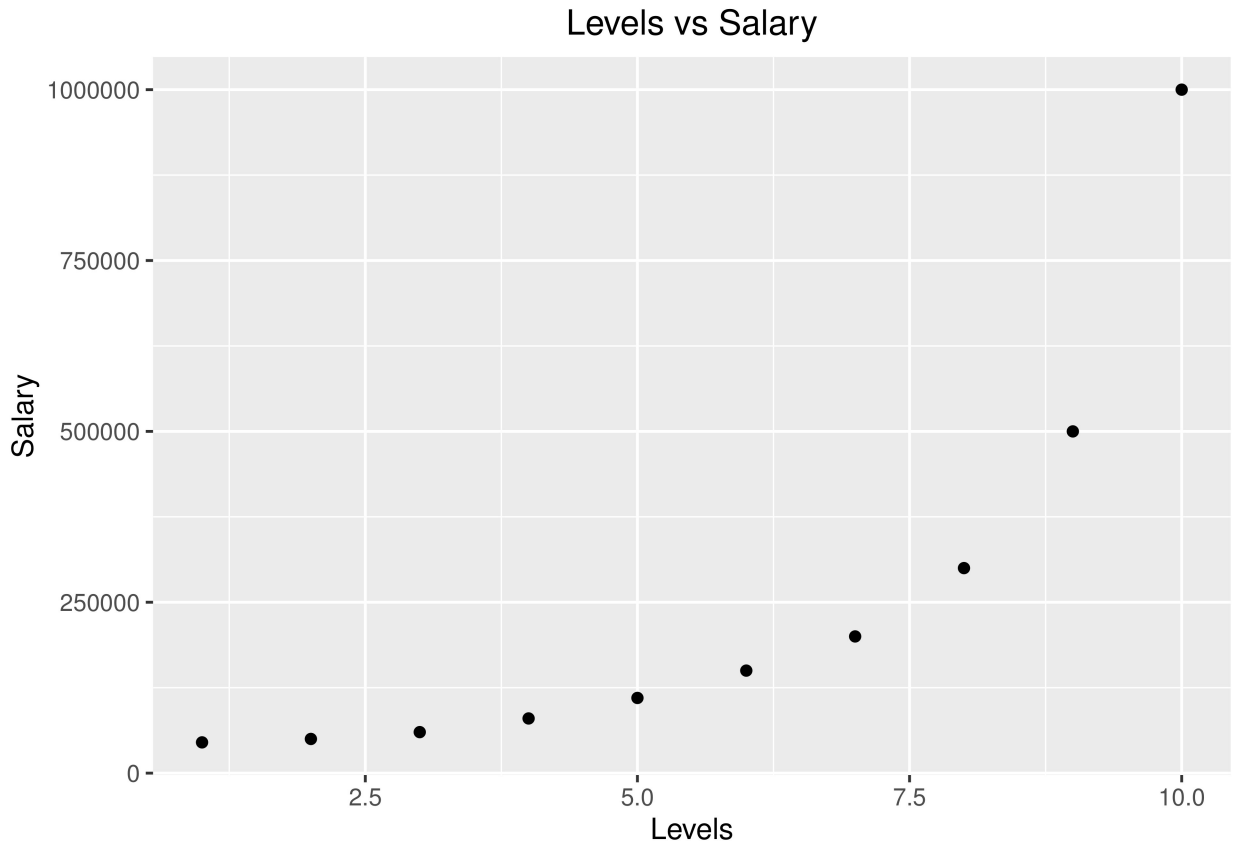
LOADING THE DATASET

```
library(caTools)
library(ggplot2)
library(ggpubr)
library(ElemStatLearn)
library(readr)

# Loading the dataset
dataset <- read_csv("Position_Salaries.csv")
print(head(dataset,10))
```

```
## # A tibble: 10 x 3
##    Position          Level  Salary
##    <chr>             <dbl>   <dbl>
##  1 Business Analyst      1   45000
##  2 Junior Consultant     2   50000
##  3 Senior Consultant     3   60000
##  4 Manager               4   80000
##  5 Country Manager       5  110000
##  6 Region Manager        6  150000
##  7 Partner               7  200000
##  8 Senior Partner        8  300000
##  9 C-level               9  500000
## 10 CEO                  10 1000000
```

```
#Visualizing the salary rates at various designation levels
ggplot()+ geom_point(aes(x=dataset$Level,y=dataset$Salary),colour='black') +
  ggtitle('Levels vs Salary')+xlab('Levels')+ylab('Salary') +
  theme_gray() + theme(plot.title = element_text(hjust = 0.5))
```




```
#Adding non-linear variations of the designation levels

dataset$Level2=dataset$Level*dataset$Level
dataset$Level3=dataset$Level2*dataset$Level
dataset$Level4=dataset$Level2*dataset$Level2
```

SIMPLE LINEAR AND POLYNOMIAL REGRESSION

We created three non-linear variations of *levels*. Since we noticed that the salary variations along levels looked non-linear, our intuition is that a polynomial model will fit the data better. However, one should be careful about over-fitting since higher degree polynomial regressions can be excessively flexible to capture the noise from the training data.

```
# Creating the simple linear regresion model
linear_regressor=lm(formula=Salary~Level,data=dataset)
summary(linear_regressor)
```

```
##
## Call:
## lm(formula = Salary ~ Level, data = dataset)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -170818 -129720  -40379   65856  386545
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -195333     124790  -1.565  0.15615
## Level          80879      20112   4.021  0.00383 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 182700 on 8 degrees of freedom
## Multiple R-squared:  0.669,  Adjusted R-squared:  0.6277
## F-statistic: 16.17 on 1 and 8 DF,  p-value: 0.003833
```

```
# Creating the polynomial regression model
poly_regressor=lm(formula=Salary~Level+Level2+Level3+Level4,data=dataset)
summary(poly_regressor)
```

```
##
## Call:
## lm(formula = Salary ~ Level + Level2 + Level3 + Level4, data = dataset)
##
## Residuals:
##      1      2      3      4      5      6      7      8      9     10
##  -8357  18240   1358 -14633 -11725   6725  15997  10006 -28695  11084
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  184166.7    67768.0   2.718  0.04189 *
## Level       -211002.3    76382.2  -2.762  0.03972 *
## Level2        94765.4    26454.2   3.582  0.01584 *
## Level3       -15463.3     3535.0  -4.374  0.00719 **
## Level4          890.2      159.8   5.570  0.00257 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20510 on 5 degrees of freedom
## Multiple R-squared:  0.9974, Adjusted R-squared:  0.9953
## F-statistic: 478.1 on 4 and 5 DF,  p-value: 1.213e-06
```

```
# Predictions
lin_pred=predict(linear_regressor,newdata=dataset)
poly_pred=predict(poly_regressor,newdata=dataset)
dataset_final=cbind(dataset,lin_pred,poly_pred)


lin_pred_chk=predict(linear_regressor,data.frame(Level=6.5))
poly_pred_chk=predict(poly_regressor,data.frame(Level=6.5,Level2=6.5^2,Level3=6.5^3,Level4=6.5^4))

# Predicted salary expectation for the prospect in consideration (Level ~6.5)
print(lin_pred_chk)
```

```
##        1
## 330378.8
```

```r
print(poly_pred_chk)
```

```
##        1
## 158862.5
```

The residual standard errors & adjusted R^2 are much better for the polynomial regressor, as expected.

Visualizing the results

```r
colors <- c("Original" = "black", "LinReg" = "red", "PolyReg" = "blue")

ggplot()+ geom_point(aes(x=dataset_final$Level,y=dataset_final$Salary, colour='Original'))+
  geom_line(aes(x=dataset_final$Level,y=dataset_final$lin_pred, colour='LinReg')) +
  geom_line(aes(x=dataset_final$Level,y=dataset_final$poly_pred, colour='PolyReg')) +
  ggtitle('Levels vs Salary') + labs(x = "Levels",y = "Salary",colour = "Legend") +
  scale_colour_manual(name = 'Models', values = colors) +
  theme_gray() + theme(plot.title = element_text(hjust = 0.5)) + theme(legend.position = "bottom")
```

SUPPORT VECTOR MACHINE REGRESSION

```r
library(e1071)
# Removing the categorical variables
```

```r
dataset=dataset[,2:5]

# Creating the SVR model
svr_regressor=svm(formula=Salary~.,data=dataset,type='eps-regression')
summary(svr_regressor)
```

```
##
## Call:
## svm(formula = Salary ~ ., data = dataset, type = "eps-regression")
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.3333333
##     epsilon:  0.1
##
##
## Number of Support Vectors:  4
```

```r
# Predictions
svr_pred=predict(svr_regressor,newdata=dataset)
dataset_final=cbind(dataset_final,svr_pred)

# Predicted salary expectation for the prospect in consideration (Level ~6.5)
svr_pred_chk=predict(svr_regressor,data.frame(Level=6.5,Level2=6.5^2,Level3=6.5^3,Level4=6.5^4))
print(svr_pred_chk)
```

```
##        1
## 200114.4
```

Visualizing the results

```r
colors <- c("Points" = 'black', "SVR" = "green", "LinReg" = "red", "PolyReg" = "blue")

ggplot()+ geom_point(aes(x=dataset_final$Level,y=dataset_final$Salary, colour='Points'))+
  geom_line(aes(x=dataset_final$Level,y=dataset_final$lin_pred, colour='LinReg')) +
  geom_line(aes(x=dataset_final$Level,y=dataset_final$poly_pred, colour='PolyReg')) +
  geom_line(aes(x=dataset_final$Level,y=svr_pred,colour='SVR')) +
  ggtitle('Levels vs Salary') + labs(x = "Levels",y = "Salary",colour = "Legend") +
  scale_colour_manual(name = 'Models', values = colors) +
  theme_gray() + theme(plot.title = element_text(hjust = 0.5)) + theme(legend.position = "bottom")
```

DECISION TREE

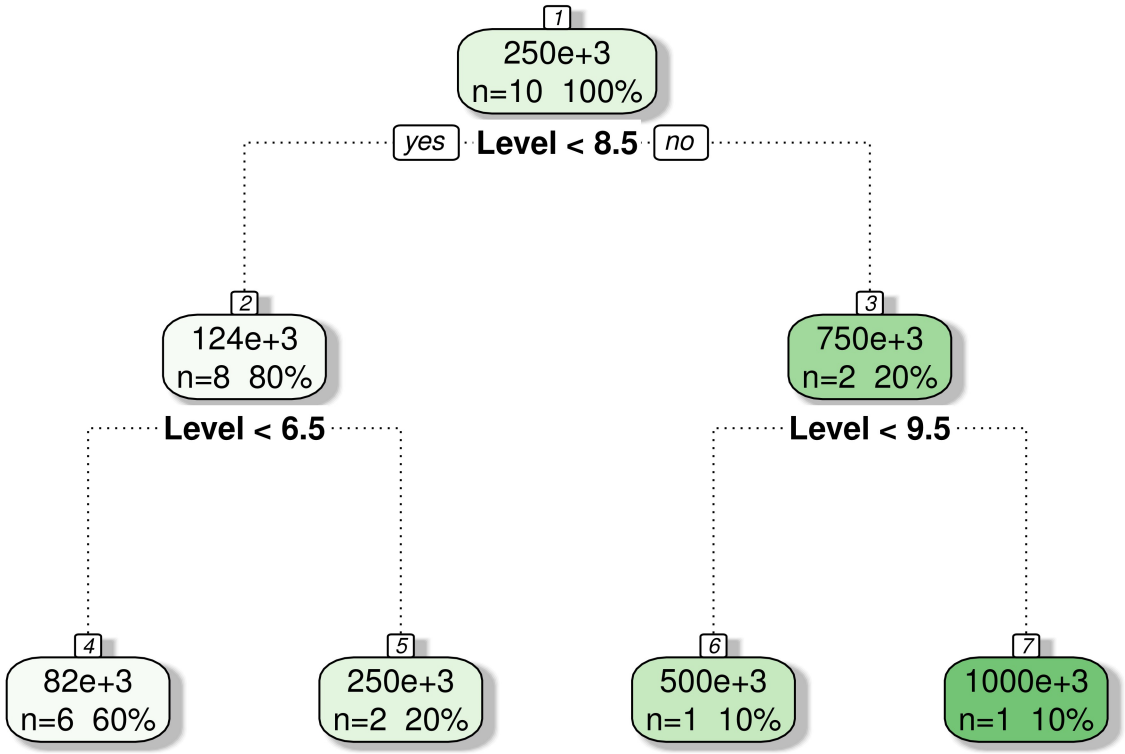
```
library(rpart)
library(rattle)
library(rpart.plot)
library(RColorBrewer)

# Creating the Decision trees
dtree=rpart(formula=Salary~Level,data=dataset,control=rpart.control(minsplit = 1))
summary(dtree)
```

```
## Call:
## rpart(formula = Salary ~ Level, data = dataset, control = rpart.control(minsplit = 1))
##   n= 10
##
##           CP nsplit  rel error   xerror      xstd
## 1 0.77638626      0 1.00000000 1.234568 0.7835133
## 2 0.15496716      1 0.22361374 1.161237 0.7912625
## 3 0.05217357      2 0.06864658 1.132188 0.7950588
## 4 0.01000000      3 0.01647301 1.132188 0.7950588
##
## Variable importance
## Level
##   100
##
```

```
## Node number 1: 10 observations,    complexity param=0.7763863
##   mean=249500, MSE=8.066225e+10
##   left son=2 (8 obs) right son=3 (2 obs)
##   Primary splits:
##       Level < 8.5 to the left,  improve=0.7763863, (0 missing)
##
## Node number 2: 8 observations,    complexity param=0.05217357
##   mean=124375, MSE=6.921484e+09
##   left son=4 (6 obs) right son=5 (2 obs)
##   Primary splits:
##       Level < 6.5 to the left,  improve=0.7600316, (0 missing)
##
## Node number 3: 2 observations,    complexity param=0.1549672
##   mean=750000, MSE=6.25e+10
##   left son=6 (1 obs) right son=7 (1 obs)
##   Primary splits:
##       Level < 9.5 to the left,  improve=1, (0 missing)
##
## Node number 4: 6 observations
##   mean=82500, MSE=1.38125e+09
##
## Node number 5: 2 observations
##   mean=250000, MSE=2.5e+09
##
## Node number 6: 1 observations
##   mean=500000, MSE=0
##
## Node number 7: 1 observations
##   mean=1000000, MSE=0
```

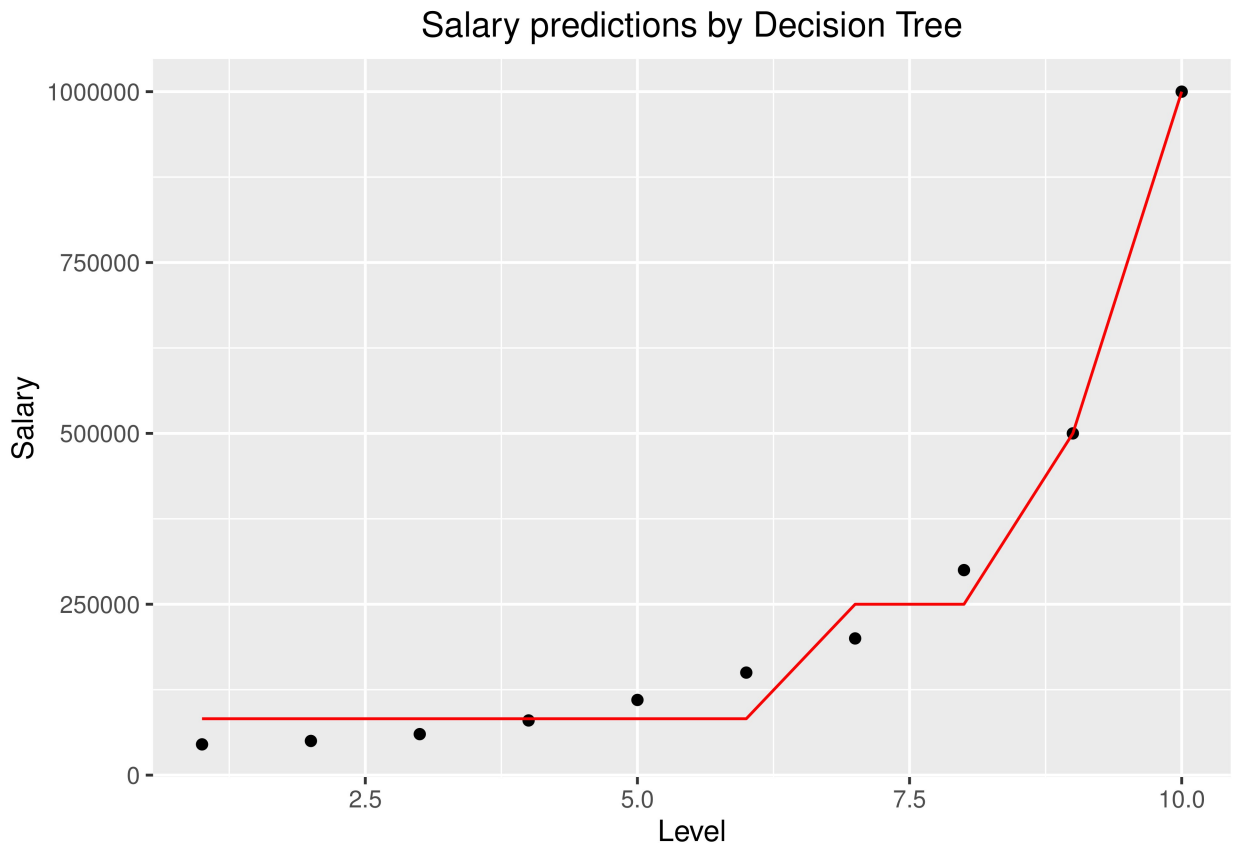
```
fancyRpartPlot(dtree, caption = NULL)
```




```
# Predictions
dtree_pred=predict(dtree,newdata=dataset)
dataset_final=cbind(dataset_final,dtree_pred)
```

Visualizing the results

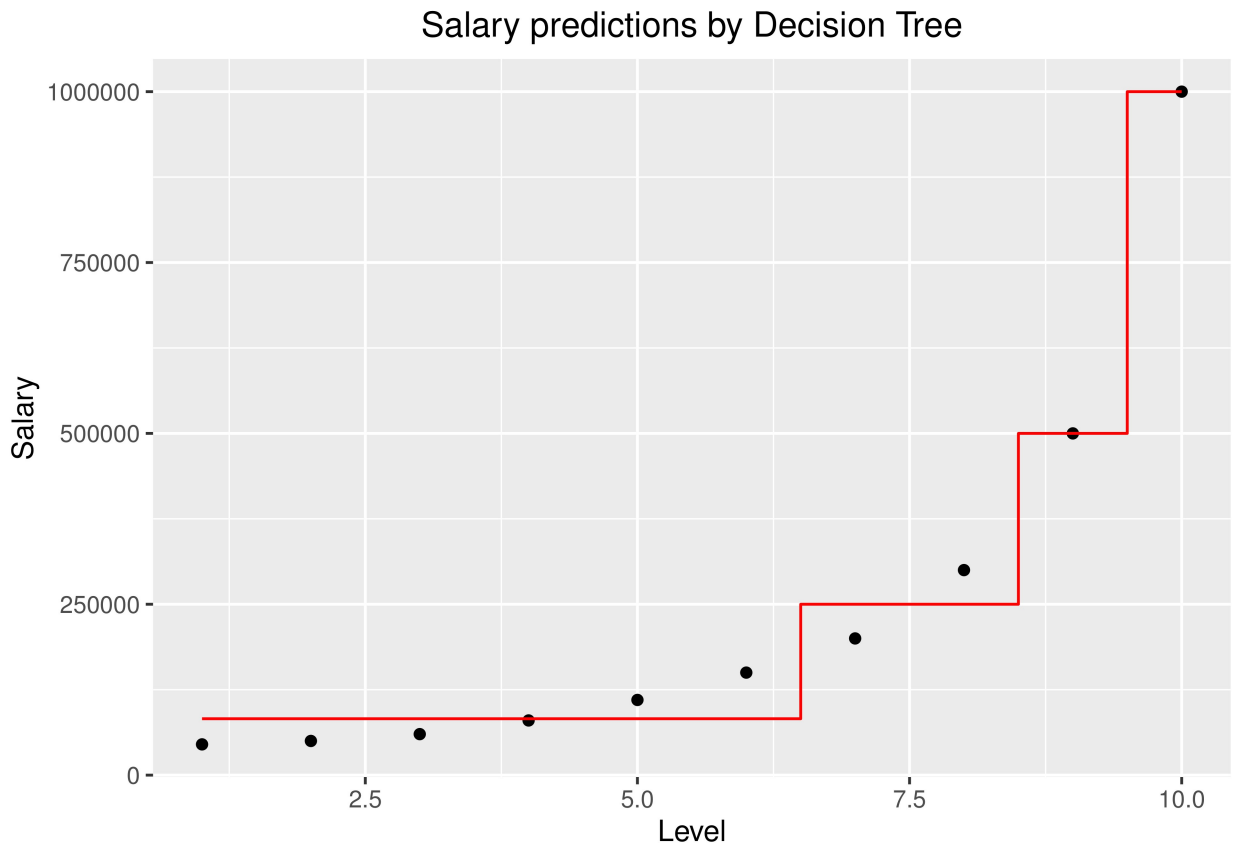
```
ggplot()+geom_point(aes(x=dataset$Level,y=dataset$Salary),colour='black') +
  geom_line(aes(x=dataset_final$Level,y=dataset_final$dtree_pred),colour='red') +
  xlab('Level') + ylab('Salary') +ggtitle('Salary predictions by Decision Tree') +
  theme_gray() + theme(plot.title = element_text(hjust = 0.5))
```



The prediction line above does not represent the true split points of the decision tree since we don't have enough dependant levels. In order to increase the resolution in the decision tree predictions, we create a dummy list of Levels to predict their salaries.

```
# Increase resolution in the plot by creating dummy dependent variables between 1 to 10 Level
x_grid=seq(min(dataset$Level), max(dataset$Level), 0.0001)

ggplot()+geom_point(aes(x=dataset$Level,y=dataset$Salary),colour='black') +
  geom_line(aes(x=x_grid,y=predict(dtree,newdata=data.frame(Level=x_grid))),colour='red')+
  xlab('Level') + ylab('Salary') +ggtitle('Salary predictions by Decision Tree') +
  theme_gray() + theme(plot.title = element_text(hjust = 0.5))
```



RANDOM FOREST REGRESSION

```
library(randomForest)
dataset=read.csv("Position_Salaries.csv")
# Creating the Random Forest regressor with 700 trees
rf_regressor=randomForest(x=dataset[2],y=dataset$Salary,ntree=700)
summary(rf_regressor)
```

```
##                 Length Class  Mode
## call               4    -none- call
## type               1    -none- character
## predicted         10    -none- numeric
## mse              700    -none- numeric
## rsq              700    -none- numeric
## oob.times         10    -none- numeric
## importance         1    -none- numeric
## importanceSD       0    -none- NULL
## localImportance    0    -none- NULL
## proximity          0    -none- NULL
## ntree              1    -none- numeric
## mtry               1    -none- numeric
## forest            11    -none- list
## coefs              0    -none- NULL
## y                 10    -none- numeric
## test               0    -none- NULL
```

```
## inbag             0    -none- NULL
```

```
# Predictions
rf_pred=predict(rf_regressor,newdata=dataset)
dataset_final=cbind(dataset_final,rf_pred)
```

Visualizing the results

```
# Increase resolution in the plot by creating dummy dependent variables between 1 to 10 Level
x_grid=seq(min(dataset$Level), max(dataset$Level), 0.0001)

colors <- c("Points" = 'black', "D-Tree" = "red", "Random Forest" = "blue")
library(ggthemes)

ggplot()+geom_point(aes(x=dataset$Level,y=dataset$Salary, colour='Points')) +
  geom_line(aes(x=x_grid,y=predict(rf_regressor,newdata=data.frame(Level=x_grid)),colour='D-Tree')) +
  geom_line(aes(x=x_grid,y=predict(dtree,newdata=data.frame(Level=x_grid)),colour='Random Forest')) +
  labs(x='Levels', y='Salary', colour='Legend') + ggtitle('Salary predictions by Random Forest/Decision
  theme_gray() + theme(plot.title = element_text(hjust = 0.5)) + theme(legend.position = "bottom")
```