

# Data Scientist Salary Prediction

Le Thi Thanh Tam

9/5/2022

## Data Scientist Salary Prediction

```
rm(list=ls())
library(ggplot2)
# Load data
data <- read.csv('E:/ThanhTam_DA/Project/Prediction/Data Scientist Salary/data_model_2.csv')

# Transform columns
names(data)[names(data) == 'avg.salary.k.'] <- 'salary'
data$type.of.ownership = factor(data$type.of.ownership)
data$sector = factor(data$sector)
data$job.location = factor(data$job.location)
data$job_title_sim = factor(data$job_title_sim)
data$seniority_by_title = factor(data$seniority_by_title)
data$degree = factor(data$degree)

# Best variable selection
library(leaps)

## Warning: package 'leaps' was built under R version 4.1.3

regfit.fwd = regsubsets(salary~., data, nvmax = 30, method = "forward")
fwd.summary = summary(regfit.fwd)

fwd.summary$rsq

## [1] 0.1419651 0.2574784 0.3487904 0.4135071 0.4305654 0.4439358 0.4598250
## [8] 0.4889180 0.5152634 0.5264122 0.5386571 0.5475069 0.5546571 0.5596790
## [15] 0.5653519 0.5709860 0.5769100 0.5809774 0.5844411 0.5878818 0.5913969
## [22] 0.5946584 0.5979972 0.6008059 0.6046067 0.6071140 0.6095831 0.6116869
## [29] 0.6135834 0.6158089
```

- we see that the R2 statistic increases from 14 %, when only one variable is included in the model, to almost 62 %, when all variables are included. As expected, the R2 statistic increases monotonically as more variables are included.

```
# Plot RSS, adjusted R squared, Cp and BIC for all of the models
par(mfrow = c(2,2))
plot(fwd.summary$rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(fwd.summary$adjr2, xlab = "Number of Variables", ylab = "Adjusted R sqd", type = "l")
# Identify the location of the maximum point of Adjusted R squared
which.max(fwd.summary$adjr2) # 30
```

```
## [1] 30
```

```
# Plot a red dot to indicate the model with the largest adjusted R squared statistic
points(30, fwd.summary$adjr2[30], col = "red", cex = 2, pch = 20)
```

```
plot(fwd.summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
which.min(fwd.summary$cp) #30
```

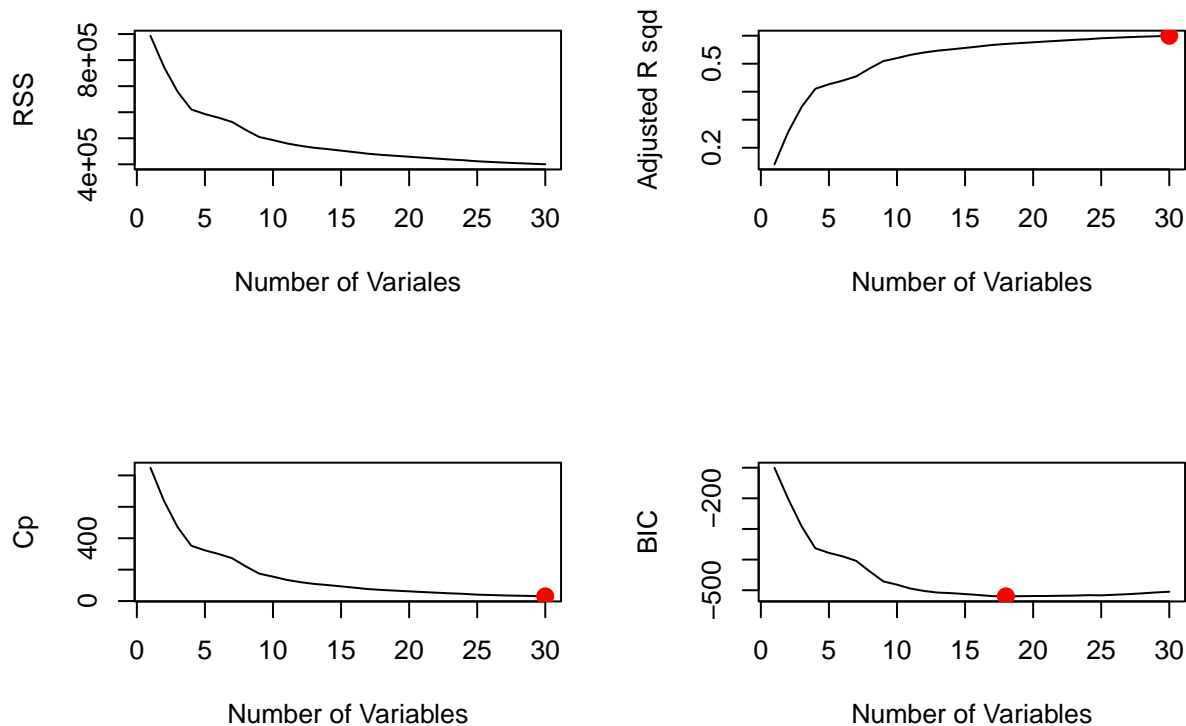
```
## [1] 30
```

```
points(30, fwd.summary$cp[30], col = "red", cex = 2, pch = 20)
```

```
plot(fwd.summary$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
which.min(fwd.summary$bic) # 18
```

```
## [1] 18
```

```
points(18, fwd.summary$bic[18], col = "red", cex = 2, pch = 20 )
```



- We see the best model with the highest Adjusted R squared and lowest Cp is 30-variable model.

```
# Check the coefficient estimates associated with the model
coef(regfit.fwd, 2)
```

```
##           (Intercept)      job_title_simDA seniority_by_titleSR
##           98.70957         -39.95864         27.88535
```

- For this data, the best one-variable model contains only **job\_title\_sim** for Data Analyst position, the best two-variable model additionally includes **seniority\_by\_title** with Senior level.
- However, to obtain the accuracy of that, we need to perform on the test set
- Next step I will use the the validation set approach to test and select the best model for this data and run the multiple linear regression

```
# write function for predict for regsubsets
predict.regsubsets=function(object,newdata,id,...){
  form=as.formula(object$call[[2]])
  mat=model.matrix(form,newdata)
  coefi=coef(object,id=id)
  xvars=names(coefi)
  mat[,xvars]%*%coefi
}

set.seed(1)
train=sample(c(TRUE,FALSE), nrow(data), rep=TRUE)
test=(!train)

regfit.best=regsubsets(salary~., data=data[train,], nvmax=30, method = "forward")
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 3 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
test.mat=model.matrix(salary~., data=data[test,])
val.errors=rep(NA, 30)
for(i in 1:30){
  coefi=coef(regfit.best, id=i)
  pred=test.mat[,names(coefi)]%*%coefi
  val.errors[i]=mean((data$salary[test]-pred)^2)
}
which.min(val.errors) #29
```

```
## [1] 23
```

```
val.errors # 995.8831
```

```
## [1] 1353.443 1350.942 1310.342 1501.777 1399.094 1397.928 1384.211 1382.629
## [9] 1390.316 1390.316 1390.381 1410.483 1411.305 1358.165 1193.508 1195.587
## [17] 1196.232 1195.864 1202.385 1181.803 1171.491 1159.845 1075.581 1084.768
## [25] 1084.304 1086.497 1090.953 1090.710 1083.671 1084.314
```

```
# perform on the full data
regfit.best=regsubsets(salary~., data=data, nvmax=30,method="forward")
coef(regfit.best, 29) # select the best 29-variables model
```

```
##           (Intercept) type.of.ownershipNonprofit
##      150.17574160      -15.78661700
##           sector4           sector5
##     -35.49952098      11.76052117
##           sector7           sector13
##     -46.11832206      8.61749668
##           sector18          sector19
##      17.72221970     -19.03778925
##           hourly      employer.provided
##     -11.74302734      42.71252574
##      job.locationAZ      job.locationCA
##     -18.08492549      22.98298272
##      job.locationCT      job.locationFL
##     -23.57062976     -16.32591917
##      job.locationGA      job.locationNM
##     -28.87810490     -40.88225180
##      job.locationTN              age
##     -18.96916595      0.06431561
##           python              sas
##      6.88794567      9.18734423
##           keras           pytorch
##      17.74438133     -8.38286690
##      job_title_simDA      job_title_simDE
##     -100.59759389     -67.23090906
##      job_title_simDS      job_title_simM
##     -60.25438536     -80.60934710
##      job_title_simMLE      job_title_simN
##     -47.52067898     -82.92112034
##      seniority_by_titleSR          num_comp
##      25.92271992      1.39228862
```

```
# note we already perform the best subsets selection on the full data and select the best 29-variables model
###-----\
#
```

- Our final best model after using validation set approach including 29 variables and the MSE is 995.8831

Consider the result from multiple linear regression

```
lm.fit=lm(salary~., data)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = salary ~ ., data = data)
##
```

```

## Residuals:
##      Min       1Q   Median       3Q      Max
## -106.085  -13.048   -0.471   10.372  115.062
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.347e+02  2.664e+01   5.057 5.57e-07 ***
## rating            8.327e-01  1.724e+00   0.483 0.629302
## type.of.ownershipHospital -1.552e+01  1.661e+01  -0.934 0.350551
## type.of.ownershipNonprofit -1.735e+01  1.468e+01  -1.182 0.237796
## type.of.ownershipOther    -6.771e+00  1.831e+01  -0.370 0.711682
## type.of.ownershipPrivate   2.316e+00  1.422e+01   0.163 0.870638
## type.of.ownershipPublic    4.701e+00  1.426e+01   0.330 0.741720
## type.of.ownershipSchool    1.004e+01  1.899e+01   0.529 0.597137
## type.of.ownershipSubsidiary 1.235e+01  1.499e+01   0.824 0.410300
## sector1            2.625e+00  2.738e+01   0.096 0.923653
## sector2            1.734e+01  1.414e+01   1.226 0.220635
## sector3            9.779e+00  2.879e+01   0.340 0.734201
## sector4           -2.964e+00  2.054e+01  -0.144 0.885295
## sector5            1.320e+01  1.253e+01   1.053 0.292678
## sector6            6.653e+00  1.298e+01   0.513 0.608309
## sector7           -4.901e+01  1.895e+01  -2.587 0.009914 **
## sector8            7.857e+00  1.775e+01   0.443 0.658264
## sector9           -4.247e+00  1.536e+01  -0.277 0.782236
## sector10           1.250e+01  1.344e+01   0.930 0.352551
## sector11           9.559e+00  1.534e+01   0.623 0.533448
## sector12           1.429e+01  1.345e+01   1.063 0.288345
## sector13           1.475e+01  1.278e+01   1.155 0.248672
## sector14           5.874e+00  1.283e+01   0.458 0.647187
## sector15           2.504e+00  1.372e+01   0.182 0.855260
## sector16           1.314e+01  1.630e+01   0.806 0.420459
## sector17           -2.422e+01  2.011e+01  -1.204 0.228962
## sector18           2.327e+01  1.737e+01   1.339 0.180952
## sector19           -1.667e+01  1.946e+01  -0.857 0.391951
## sector20           -1.275e+00  1.618e+01  -0.079 0.937184
## sector21           1.451e+01  1.431e+01   1.014 0.311083
## sector22           -2.630e+00  1.721e+01  -0.153 0.878565
## sector23           2.636e+00  1.583e+01   0.166 0.867821
## sector24           -3.825e+00  1.592e+01  -0.240 0.810247
## revenue            4.221e-01  1.139e+00   0.370 0.711166
## hourly            -1.045e+01  8.419e+00  -1.241 0.215147
## employer.provided    4.435e+01  9.991e+00   4.439 1.06e-05 ***
## job.locationAZ      -5.004e+00  1.288e+01  -0.388 0.697810
## job.locationCA       3.950e+01  9.962e+00   3.965 8.16e-05 ***
## job.locationCO       2.107e+01  1.262e+01   1.670 0.095499 .
## job.locationCT      -4.921e+00  1.479e+01  -0.333 0.739506
## job.locationDC       2.776e+01  1.246e+01   2.227 0.026285 *
## job.locationDE       2.409e+01  1.674e+01   1.439 0.150519
## job.locationFL      -1.440e+00  1.159e+01  -0.124 0.901209
## job.locationGA      -6.148e+00  1.436e+01  -0.428 0.668811
## job.locationIA       1.271e+01  1.502e+01   0.846 0.397678
## job.locationID       1.521e+01  1.954e+01   0.779 0.436499
## job.locationIL       2.094e+01  1.083e+01   1.934 0.053545 .
## job.locationIN       5.489e-01  1.285e+01   0.043 0.965941

```

|                         |            |           |         |          |     |
|-------------------------|------------|-----------|---------|----------|-----|
| ## job.locationKS       | -2.097e+01 | 1.932e+01 | -1.086  | 0.278070 |     |
| ## job.locationKY       | 4.508e+01  | 1.496e+01 | 3.013   | 0.002692 | **  |
| ## job.locationLA       | 3.280e+00  | 1.621e+01 | 0.202   | 0.839723 |     |
| ## job.locationMA       | 2.005e+01  | 1.023e+01 | 1.960   | 0.050399 | .   |
| ## job.locationMD       | 1.892e+01  | 1.046e+01 | 1.809   | 0.070858 | .   |
| ## job.locationMI       | 1.367e+01  | 1.471e+01 | 0.930   | 0.352927 |     |
| ## job.locationMN       | 2.882e+01  | 2.012e+01 | 1.432   | 0.152529 |     |
| ## job.locationMO       | 2.348e+01  | 1.329e+01 | 1.767   | 0.077778 | .   |
| ## job.locationNC       | 1.681e+01  | 1.152e+01 | 1.460   | 0.144876 |     |
| ## job.locationNE       | 1.162e+01  | 1.660e+01 | 0.700   | 0.484149 |     |
| ## job.locationNJ       | 2.469e+01  | 1.152e+01 | 2.144   | 0.032445 | *   |
| ## job.locationNM       | -2.497e+01 | 1.697e+01 | -1.472  | 0.141533 |     |
| ## job.locationNY       | 2.018e+01  | 1.026e+01 | 1.966   | 0.049715 | *   |
| ## job.locationOH       | 1.901e+01  | 1.190e+01 | 1.597   | 0.110770 |     |
| ## job.locationOR       | 4.806e+00  | 1.617e+01 | 0.297   | 0.766437 |     |
| ## job.locationPA       | 2.187e+01  | 1.099e+01 | 1.990   | 0.047019 | *   |
| ## job.locationRI       | 4.762e+01  | 2.726e+01 | 1.747   | 0.081142 | .   |
| ## job.locationSC       | -2.902e+00 | 2.603e+01 | -0.111  | 0.911259 |     |
| ## job.locationTN       | -6.470e+00 | 1.226e+01 | -0.528  | 0.597996 |     |
| ## job.locationTX       | 1.467e+01  | 1.107e+01 | 1.326   | 0.185272 |     |
| ## job.locationUT       | 2.863e+01  | 1.454e+01 | 1.968   | 0.049462 | *   |
| ## job.locationVA       | 1.046e+01  | 1.018e+01 | 1.027   | 0.304837 |     |
| ## job.locationWA       | 2.288e+01  | 1.257e+01 | 1.821   | 0.069070 | .   |
| ## job.locationWI       | 1.099e+01  | 1.287e+01 | 0.854   | 0.393240 |     |
| ## age                  | 5.544e-02  | 2.559e-02 | 2.167   | 0.030627 | *   |
| ## python               | 8.900e+00  | 2.462e+00 | 3.614   | 0.000325 | *** |
| ## spark                | -1.870e+00 | 3.223e+00 | -0.580  | 0.562019 |     |
| ## aws                  | 1.504e+00  | 2.580e+00 | 0.583   | 0.560180 |     |
| ## excel                | 6.642e-01  | 2.094e+00 | 0.317   | 0.751193 |     |
| ## sql                  | -4.398e+00 | 2.634e+00 | -1.670  | 0.095422 | .   |
| ## sas                  | 1.018e+01  | 3.913e+00 | 2.602   | 0.009483 | **  |
| ## keras                | 1.742e+01  | 6.726e+00 | 2.590   | 0.009818 | **  |
| ## pytorch              | -1.378e+01 | 5.912e+00 | -2.331  | 0.020045 | *   |
| ## scikit               | -1.835e+00 | 4.840e+00 | -0.379  | 0.704664 |     |
| ## tensor               | 4.263e+00  | 5.319e+00 | 0.801   | 0.423202 |     |
| ## hadoop               | 4.977e+00  | 3.408e+00 | 1.460   | 0.144682 |     |
| ## tableau              | -7.364e+00 | 3.148e+00 | -2.340  | 0.019613 | *   |
| ## bi                   | 6.228e+00  | 4.403e+00 | 1.414   | 0.157737 |     |
| ## flink                | -6.674e+00 | 8.994e+00 | -0.742  | 0.458344 |     |
| ## mongo                | 1.079e+01  | 5.002e+00 | 2.158   | 0.031294 | *   |
| ## google_an            | -1.711e+01 | 9.020e+00 | -1.897  | 0.058231 | .   |
| ## job_title_simDA      | -9.556e+01 | 8.003e+00 | -11.942 | < 2e-16  | *** |
| ## job_title_simDE      | -6.541e+01 | 7.900e+00 | -8.280  | 7.22e-16 | *** |
| ## job_title_simDS      | -5.874e+01 | 7.474e+00 | -7.860  | 1.63e-14 | *** |
| ## job_title_simM       | -7.507e+01 | 9.193e+00 | -8.166  | 1.70e-15 | *** |
| ## job_title_simMLE     | -4.311e+01 | 9.941e+00 | -4.336  | 1.68e-05 | *** |
| ## job_title_simN       | -7.944e+01 | 7.588e+00 | -10.469 | < 2e-16  | *** |
| ## seniority_by_titleN  | -1.176e+01 | 1.502e+01 | -0.783  | 0.434088 |     |
| ## seniority_by_titleSR | 1.507e+01  | 1.511e+01 | 0.997   | 0.319077 |     |
| ## degreeN              | -1.190e+00 | 2.378e+00 | -0.500  | 0.617020 |     |
| ## degreeP              | 1.588e+00  | 3.635e+00 | 0.437   | 0.662274 |     |
| ## desc_len             | -2.581e-04 | 7.173e-04 | -0.360  | 0.719105 |     |
| ## num_comp             | 1.721e+00  | 7.960e-01 | 2.162   | 0.030992 | *   |
| ## size                 | -2.140e+00 | 1.692e+00 | -1.264  | 0.206519 |     |

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.73 on 640 degrees of freedom
## Multiple R-squared:  0.6538, Adjusted R-squared:  0.5992
## F-statistic: 11.97 on 101 and 640 DF,  p-value: < 2.2e-16
```

```
## Compute variance inflation factors
library(car)
```

```
## Warning: package 'car' was built under R version 4.1.3
```

```
## Loading required package: carData
```

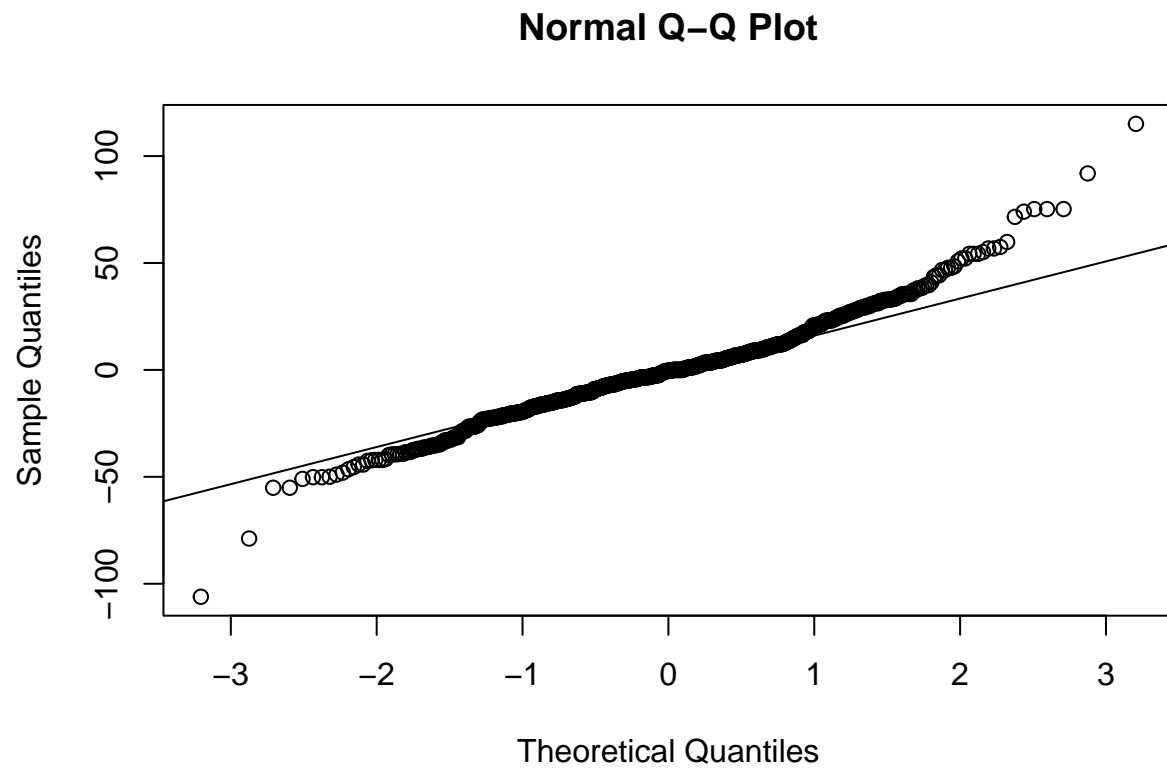
```
## Warning: package 'carData' was built under R version 4.1.3
```

```
vif(lm.fit)
```

```
##              GVIF Df GVIF^(1/(2*Df))
## rating          2.511465e+00  1      1.584760
## type.of.ownership 3.699815e+02  7      1.525603
## sector           1.992028e+05 24      1.289443
## revenue          2.623077e+00  1      1.619592
## hourly           2.923337e+00  1      1.709777
## employer.provided 2.944862e+00  1      1.716060
## job.location      1.198876e+04 36      1.139330
## age              2.497064e+00  1      1.580210
## python           1.991172e+00  1      1.411089
## spark            2.387316e+00  1      1.545094
## aws              1.587134e+00  1      1.259815
## excel            1.441610e+00  1      1.200671
## sql              2.283700e+00  1      1.511192
## sas              1.634918e+00  1      1.278639
## keras            2.238703e+00  1      1.496230
## pytorch          2.293789e+00  1      1.514526
## scikit           2.083344e+00  1      1.443379
## tensor           3.266694e+00  1      1.807400
## hadoop           2.130541e+00  1      1.459637
## tableau          2.084797e+00  1      1.443883
## bi               1.782754e+00  1      1.335198
## flink            1.417197e+00  1      1.190461
## mongo            1.562200e+00  1      1.249880
## google_an        1.984582e+00  1      1.408752
## job_title_sim     2.312312e+01  6      1.299184
## seniority_by_title 1.614244e+00  2      1.127177
## degree           2.892084e+00  2      1.304076
## desc_len         1.567474e+00  1      1.251988
## num_comp          1.597776e+00  1      1.264032
## size             2.903571e+00  1      1.703987
```

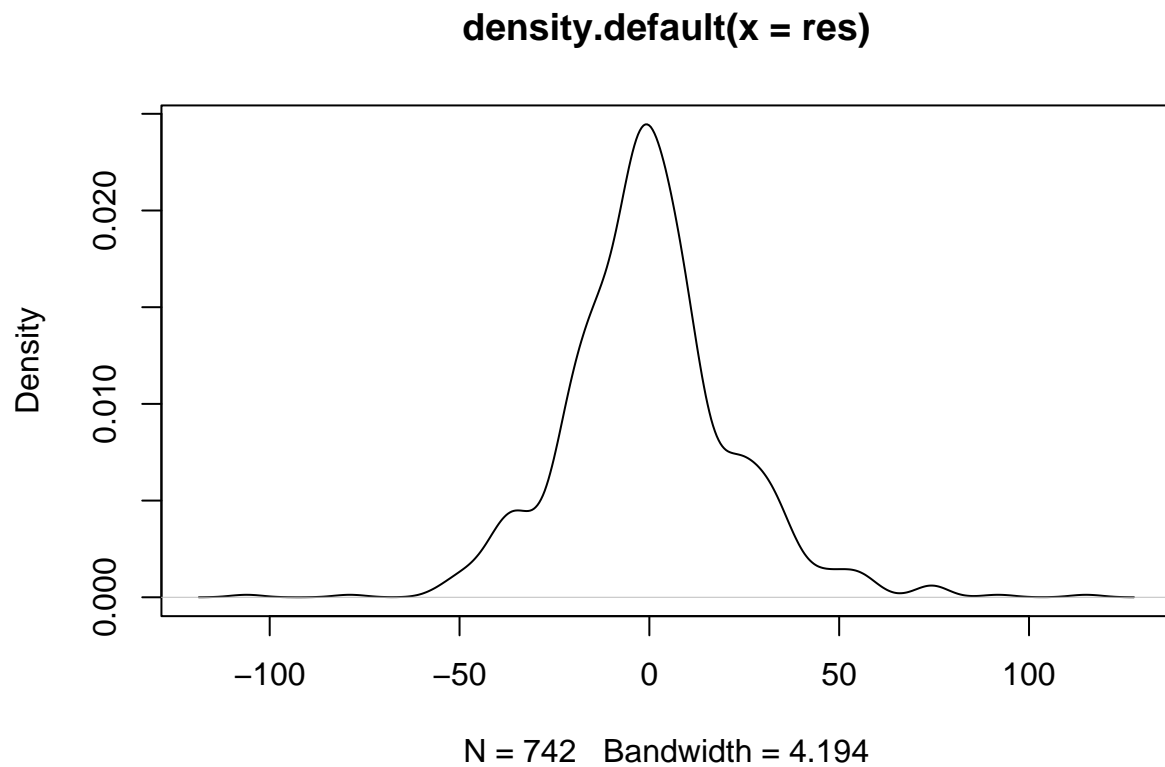
```
# get list residual
res <- resid(lm.fit)

## normal distribution
### create a q-q plot
qqnorm(res)
qqline(res) # add a straight diagonal line to the plot
```



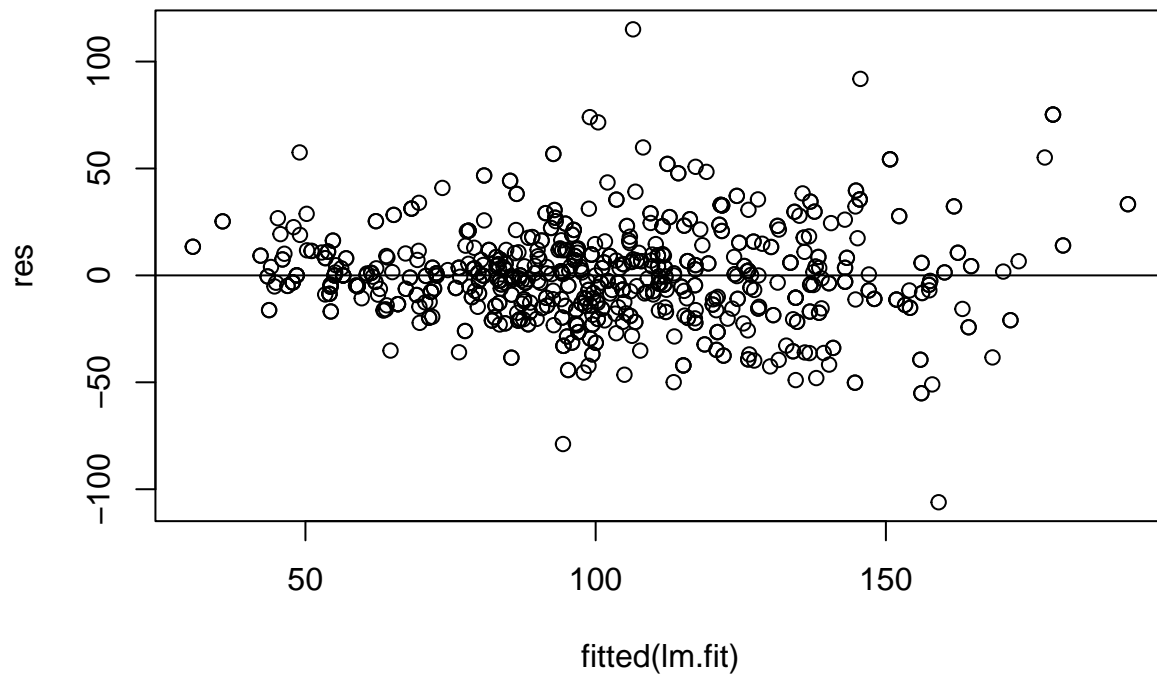
```
plot(density(res))
```





```
####-> we can see that the density plot roughly follows the bell shape (normal distribution)

## produce residual vs.fitted plot to visualizing heteroskedasticity
plot(fitted(lm.fit), res)
abline(0,0)
```



*#--> we can see the data showing heteroscedasticity. the residuals are observed to have unequal variance*

- we can see the data showing heteroscedasticity. the residuals are observed to have unequal variance

## Lasso

- Now I will perform the lasso \_ the techniques for shrinking the regression coefficients towards zero. By using this technique, hope the coefficient estimates can significantly reduce their variance for a more accurate prediction.
- Now I will perform the lasso in order to predict salary on this data

```
# install.packages("glmnet")
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.1.3
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-4
```

```

x=model.matrix(salary~., data)[,-1] # automatically transforms any qualitative variables into dummy variables
y=data$salary

# split the sample data into a training set and a test set in order to estimate the test error of the lasso model
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]

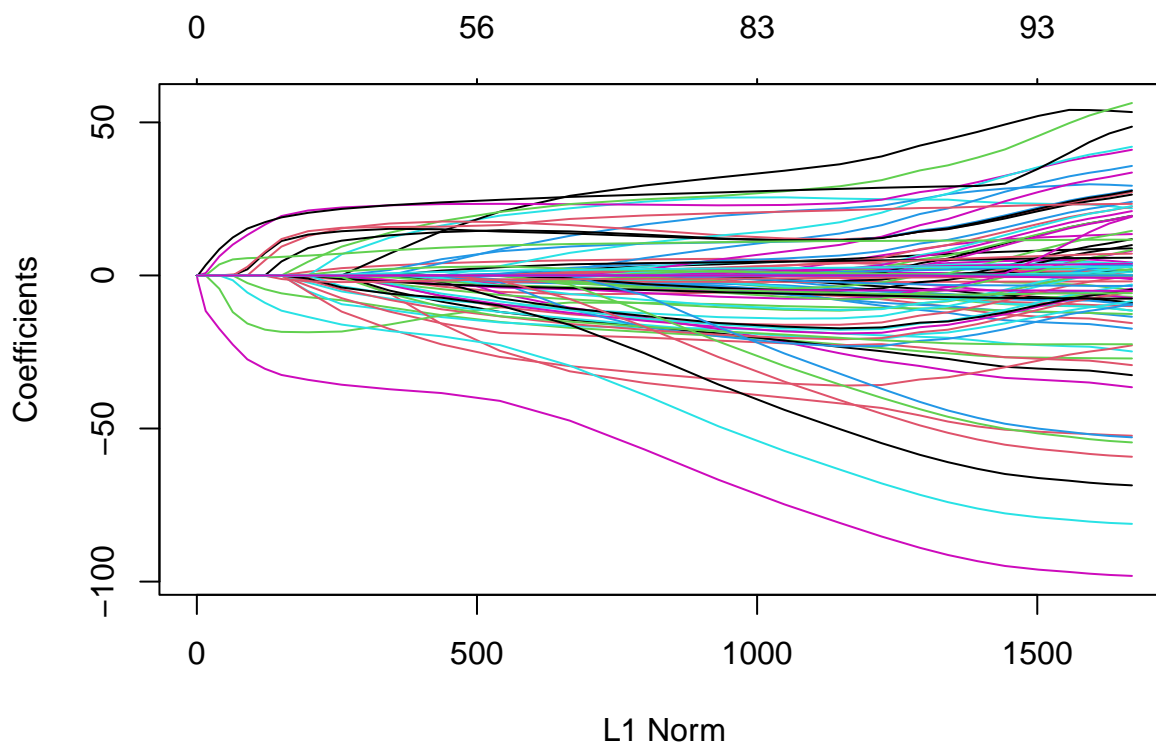
# now we fit a lasso model
grid=10^seq(10,-2,length=100)
lasso.mod=glmnet(x[train,], y[train], alpha=1, lambda = grid)
plot(lasso.mod)

```

```

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

```

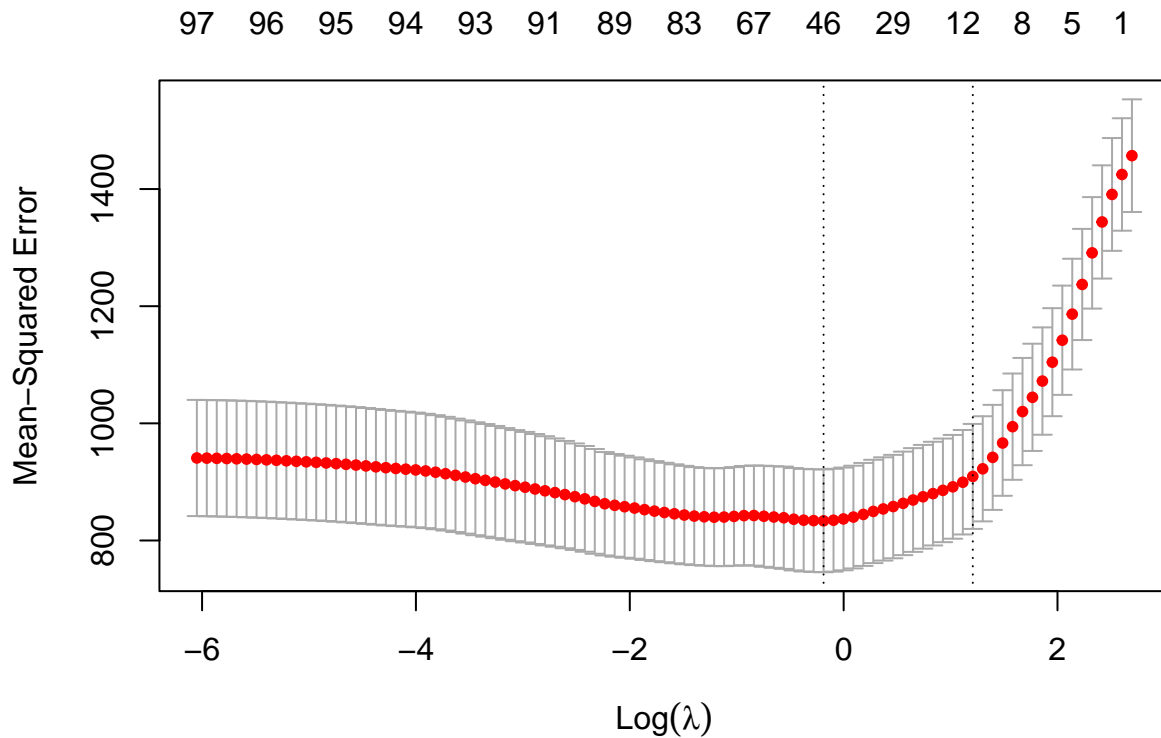


```

##-> we now see some of the coefficients will be exactly equal to zero

# perform cross-validation and compute the associated test error
set.seed(1)
cv.out=cv.glmnet(x[train,], y[train], alpha=1)
plot(cv.out)

```



```
bestlam=cv.out$lambda.min
lasso.pred=predict(lasso.mod, s=bestlam,newx=x[test,])
mean((lasso.pred-y.test)^2) # MSE = 762.544
```

```
## [1] 662.7368
```

*##-> This is substantially lower than the test set MSE of least squares*

- As expected, the lasso regression perform better than multiple linear regression compared MSE = 762.544 (mean square errors) in predicting the salary.

## RandomForest

```
# install.packages("randomForest")
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'

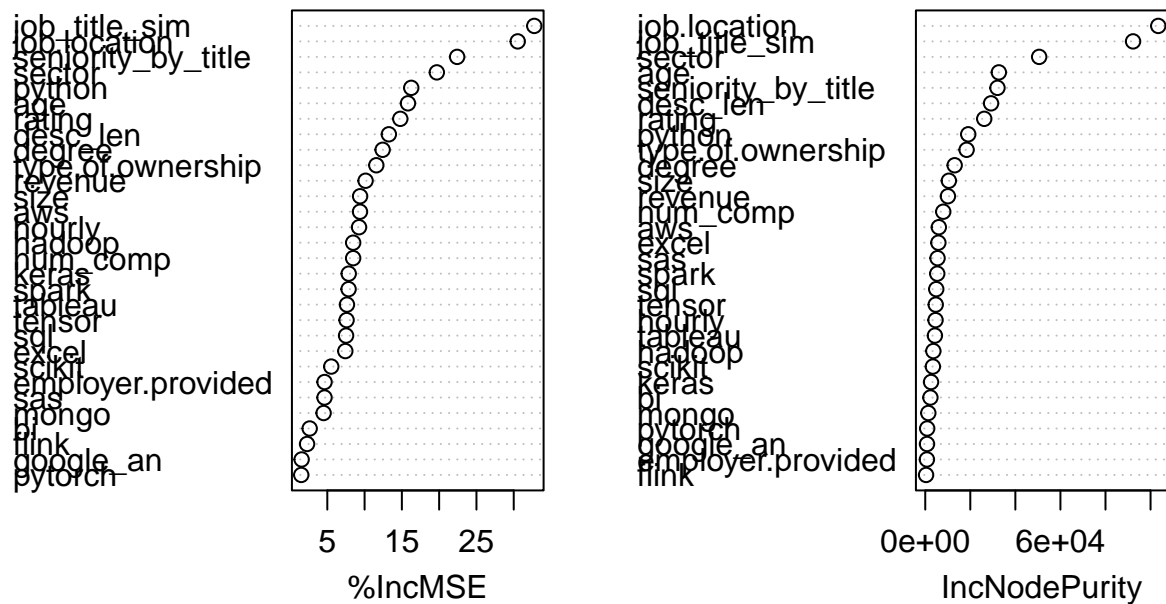
## The following object is masked from 'package:ggplot2':
##
##      margin

set.seed(1)
train=sample(1:nrow(data), nrow(data)/2)
data.test=data[-train,"salary"]
rf.data=randomForest(salary~., data=data, subset=train, mtry=6, importance=TRUE)
yhat.rf= predict(rf.data, newdata=data[-train,])
mean((yhat.rf-data.test)^2) # 575.7547

## [1] 510.7222

#plot of the importance measures
varImpPlot(rf.data)
```

rf.data



- The results indicate that across all of the trees considered in the random forest, the wealth level of the job title (job\_title\_sim) and job\_location are by far the most important variables.
- The Random Forest model far outperformed the other approaches on the test and validation sets.