ResumeScanner

Muhammad Sualeh Alam

2025-04-23

Data Cleaning

Transforming the raw “resume_data” into a cleaned data format for further analysis.

```r
# Read old data
df_res <- read.csv("C:/Users/ihate/OneDrive/Documents/CSUEB Project/Resume
ATS Scanner Project/resume_data.csv", stringsAsFactors = FALSE)

# Remove rows with any missing values
df_clean <- na.omit(df_res)

# Drop columns
df_clean <- df_clean[ , !(names(df_clean) %in% c("job_description"))]

# Create a new column 'decision' based on 75 score threshold
df_clean$decision <- ifelse(df_clean$score > 75, "Hired", "Rejected")

#head(df_clean)

# After Data cleaning
print(dim(df_res) )
```

```
## [1] 232  11
```

```r
print(dim(df_clean))
```

```
## [1] 114  11
```

Creating a new Cleaned Dataset

Export it into a new **cleaned_data** file

```r
# Code is currently commented out otherwise each time its run it creates a
new file.
#write.csv(df_clean, "cleaned_data.csv", row.names = FALSE)
```

Model Building for MLR

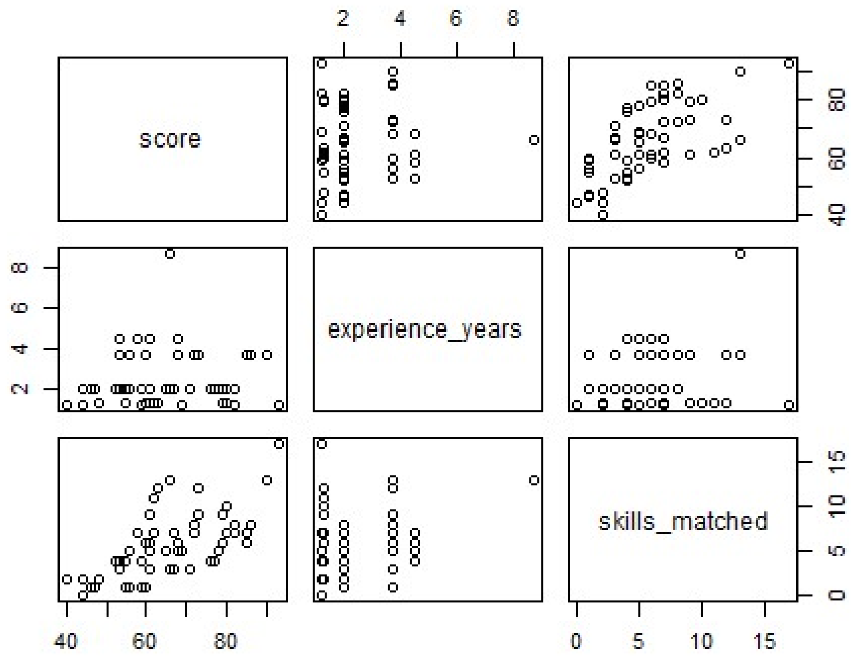
```r
df_resume <- read.csv("C:/Users/ihate/Downloads/f24/data/cleaned_data.csv") #
make sure this is in the wd

# Using One-hot Encoding to convert categorical into numbers
```

```
df_resume$fexperience_level <- as.integer(factor(df_resume$experience_level,
levels = c("Junior", "Mid-Level", "Senior"))) - 1
df_resume$fexperience_level = as.factor(df_resume$fexperience_level)
```

Multiple Linear Regression

```
pairs(score ~ experience_years+ skills_matched, data=df_resume)
```



```
library(car)
# Checking VIF for base model
vif(lm( (score) ~ (experience_years) + (skills_matched) + fexperience_level,
data=df_resume))
```

```
##                        GVIF Df GVIF^(1/(2*Df))
## experience_years  12.378041  1        3.518244
## skills_matched     1.328466  1        1.152591
## fexperience_level 13.421799  2        1.914047
```

From the VIFs value it seems there do exist multi-collinearity issue between 2 columns (experience_years and fexperience_level) as VIF > 10. So, for now we will not remove these columns because they are essential for our analysis, instead we will check Multi-collinearity at the end after complete transformation and on our final model.

This is a classic **model-building dilemma**, and it's all about balancing predictive power and model stability.

```r
base_model = lm( (score) ~ (experience_years) + (skills_matched) +
fexperience_level, data=df_resume)
summary(base_model)
```

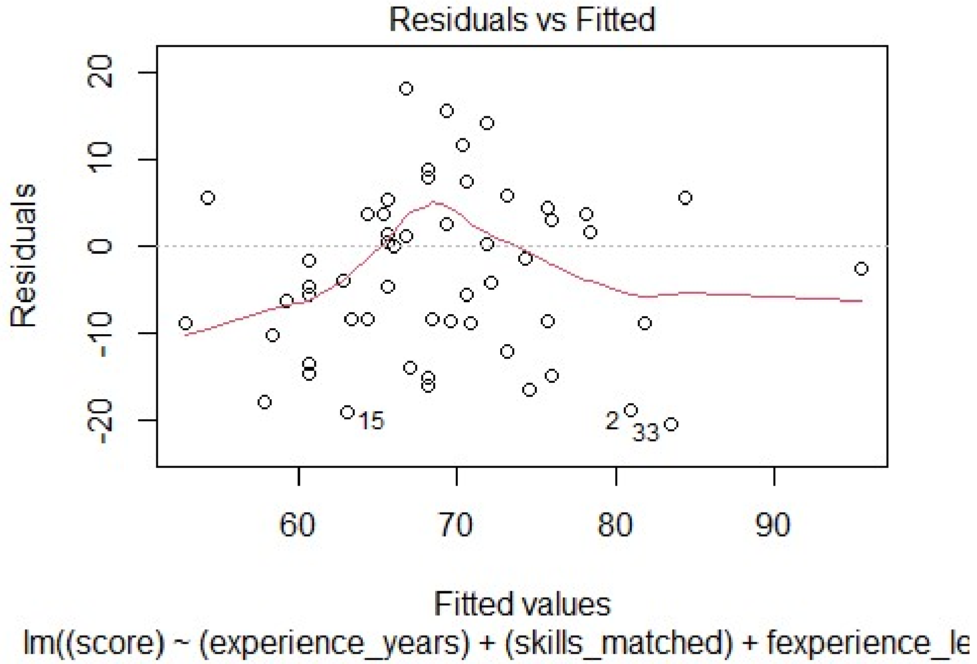
```
##
## Call:
## lm(formula = (score) ~ (experience_years) + (skills_matched) +
##     fexperience_level, data = df_resume)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.424  -5.347   1.393   7.379  18.178
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          45.4382     5.0984   8.912 1.28e-14 ***
## experience_years      6.3245     2.3679   2.671 0.008723 **
## skills_matched        2.5067     0.3211   7.806 3.87e-12 ***
## fexperience_level1 -16.8462     5.5790  -3.020 0.003153 **
## fexperience_level2 -67.3649    19.6786  -3.423 0.000873 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.505 on 109 degrees of freedom
## Multiple R-squared:  0.3598, Adjusted R-squared:  0.3363
## F-statistic: 15.32 on 4 and 109 DF,  p-value: 5.713e-10
```

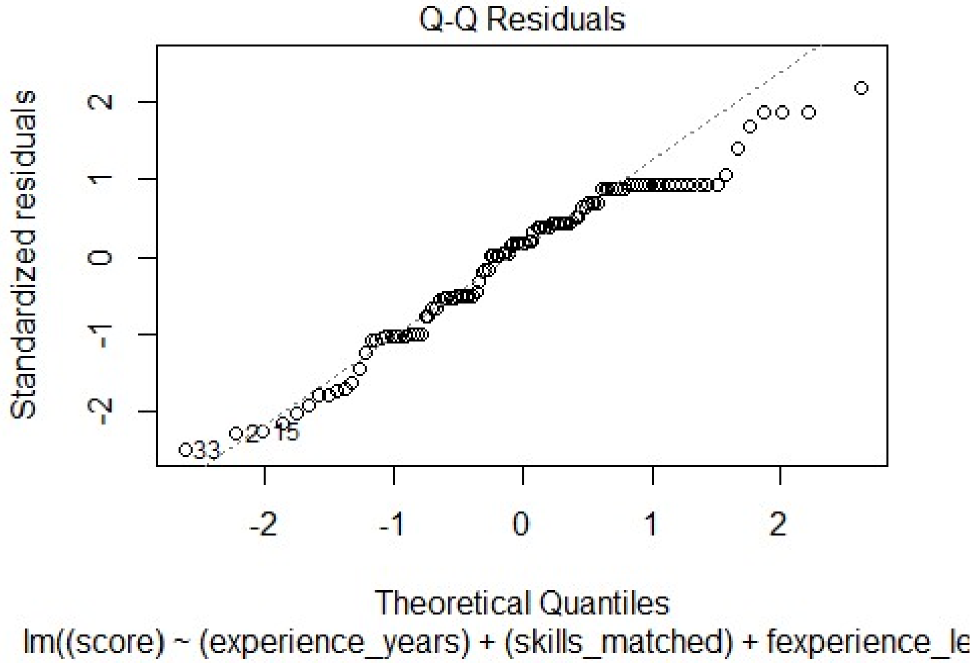
Conclusion:
All our predictors are highly significant at the $\alpha = 0.05$ significance level, so we will keep all of them in our model.

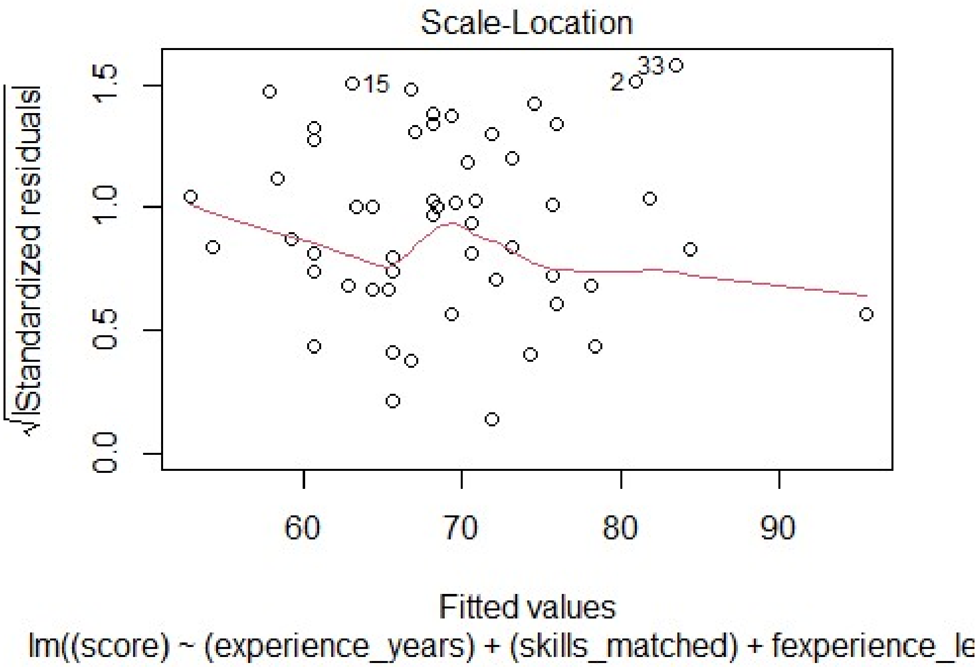
Our base model for Multiple Linear Regression gives us an:
$R^2_{adj} = 33.63\%.$
$R^2 = 35.98\%$

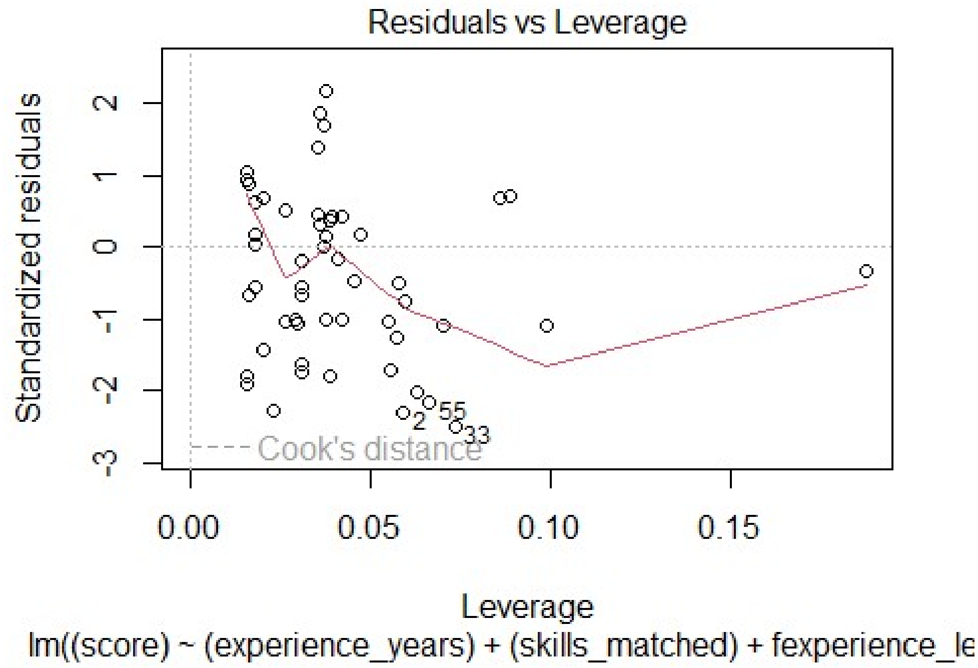
We will now check assumptions to see whether they are violated or not or we need to transform

```r
plot(base_model)
```


Residuals vs Fitted
Residuals
20
10
0
-10
-20
60
70
80
90
15
2
33
Fitted values
lm((score) ~ (experience_years) + (skills_matched) + fexperience_le


Q-Q Residuals
Standardized residuals
2
1
0
-1
-2
-2
-1
0
1
2
33
2
15
Theoretical Quantiles
lm((score) ~ (experience_years) + (skills_matched) + fexperience_le





```
shapiro.test(resid(base_model))

##
##  Shapiro-Wilk normality test
```

```
##
## data:  resid(base_model)
## W = 0.96028, p-value = 0.001873
```

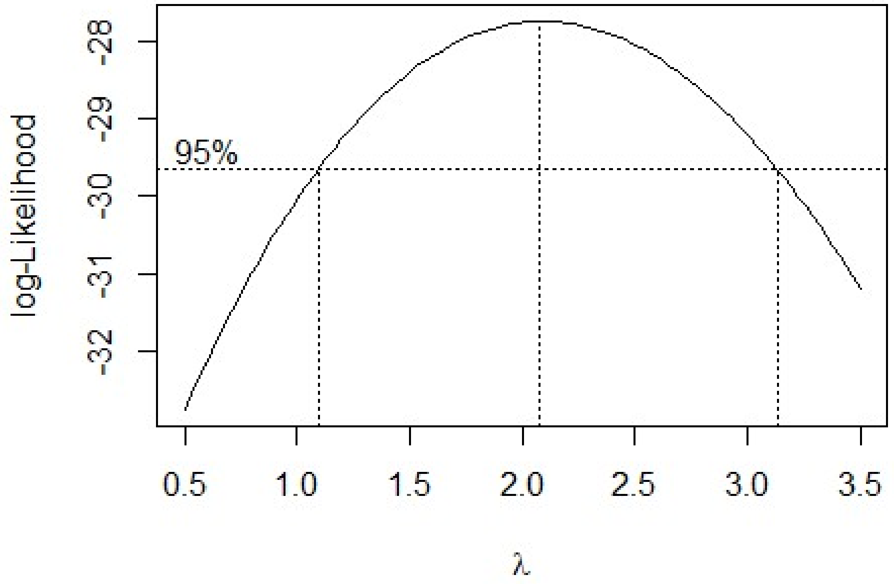
Observations:
From the plots, we can see that constant variance assumption seems satisfied as we can see a random scatter of points. Furthermore, the QQPlot also shows most of the points on the line but there are visible deviations seen at both ends.

Comment on Normality Assumption:
For further confirmation we use Shapiro-Wilk test for normality to confirm our assumption of normality is met or not. From the results, we can see that **p-value = 0.001873** is lesser than our α $(significance\ level)$ which means normality assumption is violated.

Now we will try to transform using BoxCox Transformation to stabilize Normality.

```
library(MASS)

boxcox(base_model, lambda=seq(0.5, 3.8, by=0.5))
```



```
summary(powerTransform(base_model))

## bcPower Transformation to Normality
##    Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## Y1    2.0888           2       1.0713       3.1064
##
## Likelihood ratio test that transformation parameter is equal to 0
##  (log transformation)
##                            LRT df       pval
## LR test, lambda = (0) 17.85535  1 2.3835e-05
##
## Likelihood ratio test that no transformation is needed
##                            LRT df     pval
## LR test, lambda = (1) 4.631032  1 0.031399
```

Results:
From the BoxCox transformation, it is suggested to use lambda=2, technically Square transformation on the Score variable.

Using BoxCox Transformation

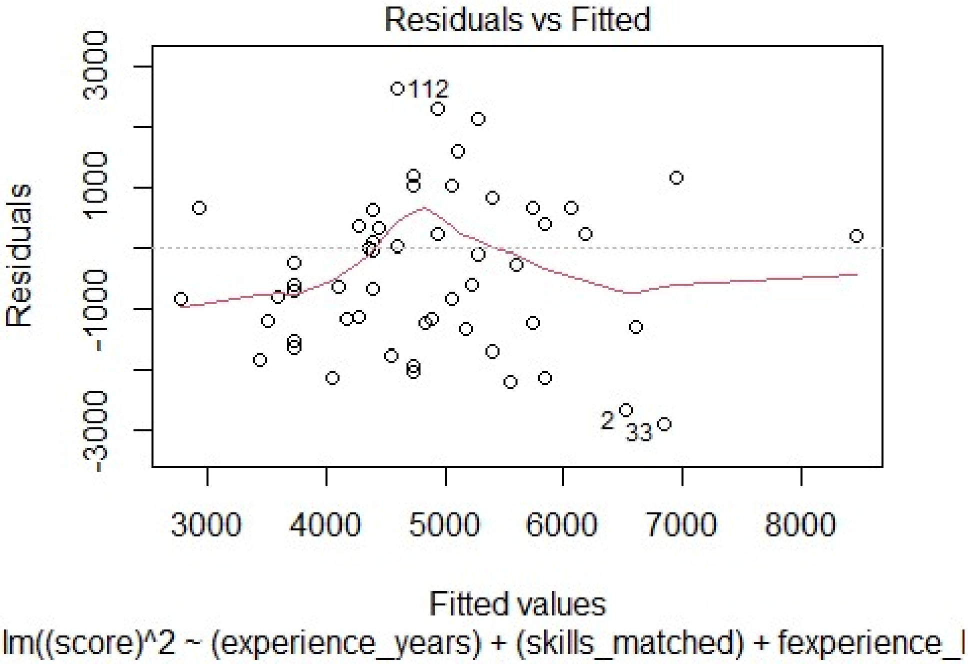
```
base_modelTransform = lm( (score)^2 ~ (experience_years) + (skills_matched) +
fexperience_level, data=df_resume)
summary(base_modelTransform)
```

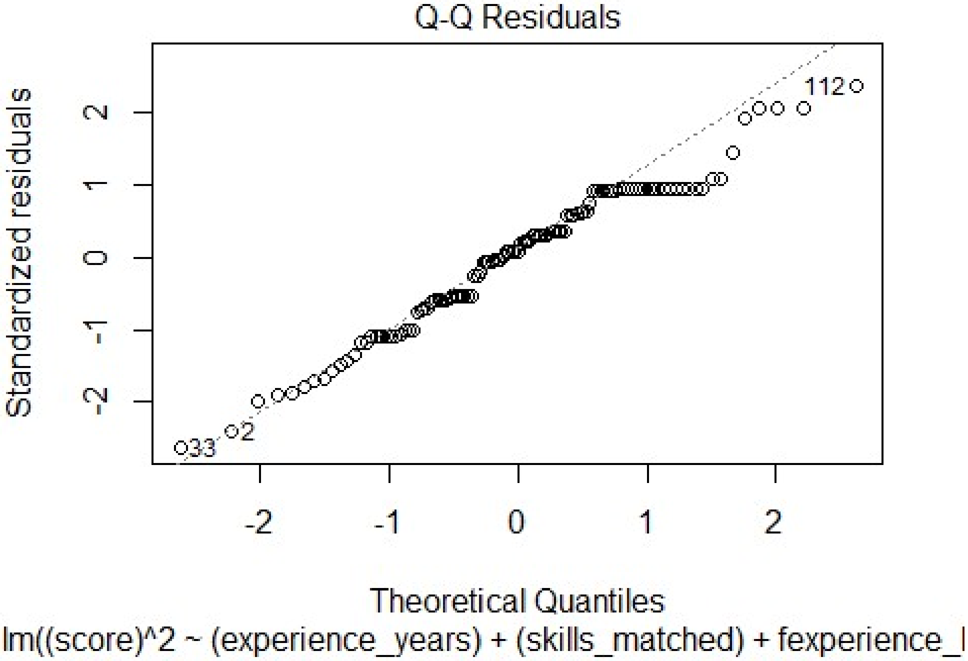
```
##
## Call:
## lm(formula = (score)^2 ~ (experience_years) + (skills_matched) +
##     fexperience_level, data = df_resume)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2876.81  -693.73    94.55  1020.59  2622.56
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1900.41     681.75   2.788  0.00627 **
## experience_years      745.30     316.64   2.354  0.02037 *
## skills_matched        334.48      42.94   7.789 4.21e-12 ***
## fexperience_level1  -2037.63     746.03  -2.731  0.00736 **
## fexperience_level2  -8414.03    2631.43  -3.198  0.00181 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1137 on 109 degrees of freedom
## Multiple R-squared:  0.3606, Adjusted R-squared:  0.3372
## F-statistic: 15.37 on 4 and 109 DF,  p-value: 5.345e-10
```

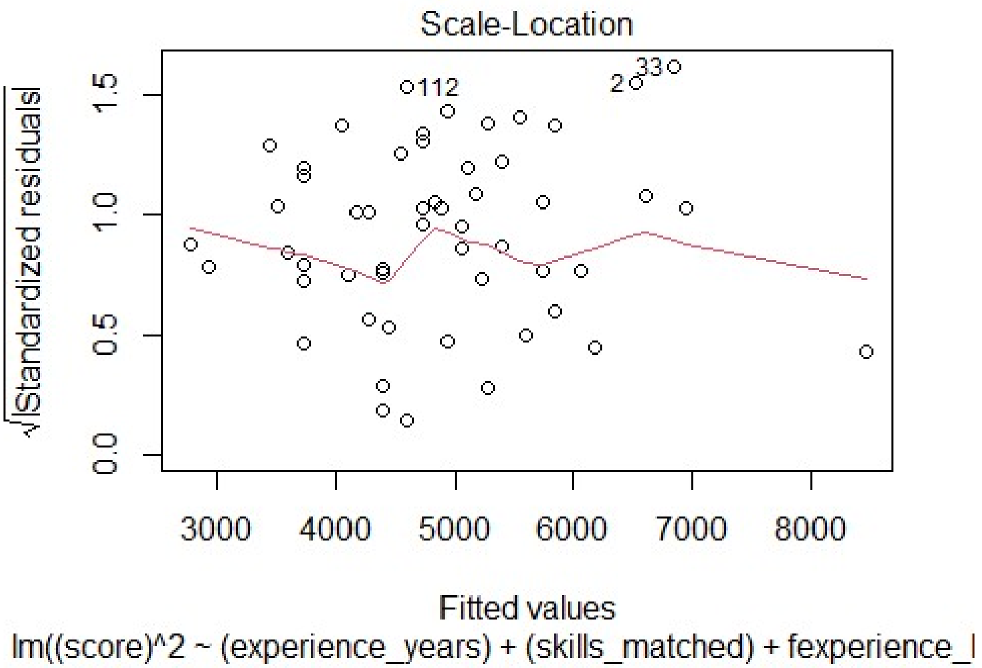
Conclusion:
After applying the BoxCox Transformation, not much improvement in the R^2_{adj}. After being transformed R^2_{adj}= 33.72% from the base model R^2_{adj} = 33.63%. This is a very slight improvement in the model after transformation.

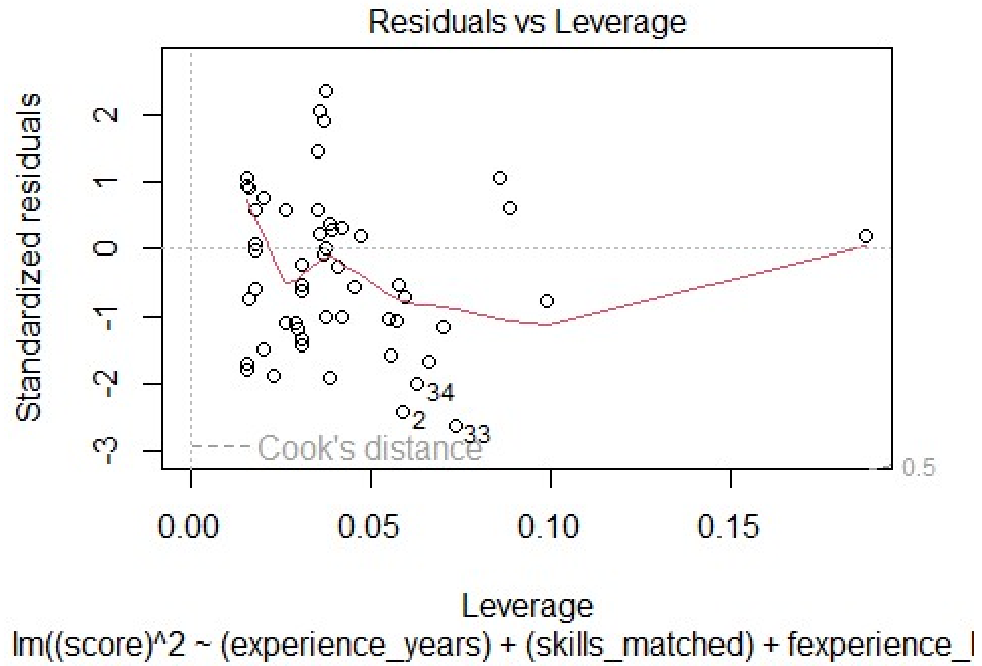
We will check the assumptions once again after Square transformation

```
plot(base_modelTransform)
```


Residuals vs Fitted
Residuals
3000
1000
-1000
-3000
112
2
33
3000
4000
5000
6000
7000
8000
Fitted values
lm((score)^2 ~ (experience_years) + (skills_matched) + fexperience_l


Q-Q Residuals
Standardized residuals
2
1
0
-1
-2
112
2
33
-2
-1
0
1
2
Theoretical Quantiles
lm((score)^2 ~ (experience_years) + (skills_matched) + fexperience_l





```
shapiro.test(resid(base_modelTransform))

##
##  Shapiro-Wilk normality test
```

```
##
## data:  resid(base_modelTransform)
## W = 0.97161, p-value = 0.01584
```

```r
cat("AIC Value of Transformed Model = ", AIC(base_modelTransform))
```

```
## AIC Value of Transformed Model =  1934.695
```

Comment on assumptions:
Even after using square transformation the Normality assumption is still not being satisfied. The QQPlot still shows major signs of deviation at tails. Furthermore, the Residual vs Fitted plot also shows problems of constant variance now as we can observe some signs of Fanning Pattern from the plot.

Furthermore, the AIC value of the transformed model also shows a very high-value.

Model Optimization on my own :)

We will try some more transformations to improve the R^2_{adj}

```r
# Using log1p because skills_matched has a 0 value so this will handle this.
lm3 <- lm( log(score) ~ log(experience_years) + log1p(skills_matched) +
fexperience_level, data=df_resume)
summary(lm3)
```

```
##
## Call:
## lm(formula = log(score) ~ log(experience_years) + log1p(skills_matched) +
##     fexperience_level, data = df_resume)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32241 -0.06833  0.02549  0.08347  0.22250
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            3.68231    0.06279  58.647  < 2e-16 ***
## log(experience_years)  0.21214    0.05867   3.616 0.000455 ***
## log1p(skills_matched)  0.25237    0.02599   9.712  < 2e-16 ***
## fexperience_level1    -0.22887    0.05635  -4.062  9.2e-05 ***
## fexperience_level2    -0.61880    0.16085  -3.847 0.000202 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1204 on 109 degrees of freedom
## Multiple R-squared:  0.4654, Adjusted R-squared:  0.4458
## F-statistic: 23.73 on 4 and 109 DF,  p-value: 3.954e-14
```

```r
cat("Final Model AIC Value= ", AIC(lm3))
```

```
## Final Model AIC Value=  -152.3034
```

Why This Is Statistically Justified:
In practice, **model performance > strict adherence to Box-Cox lambda**, especially if your goal is prediction.

Statistical Dilligence:
Box-Cox suggested a λ = 2, but empirical performance supports the log model.

Final Model:

$$log\left(\widehat{Score}\right) = log(ExperienceYears) + log(Skillsmatched) + Fexperiencelevel1 \\ + Fexperiencelevel1$$

Estimated Regression Model:

$$log\left(\widehat{Score}\right) = 3.682 + 0.212log(ExperienceYears) + 0.252log(Skillsmatched) \\ - 0.228Fexperiencelevel1 - 0.618Fexperiencelevel2$$

R^2**= 46.5%**
R^2_{adj} **=44.58%** (Significant improvement in the model)
AIC Value of Final Model= -152.3 (The lower the AIC the better the model performance)

R^2_{adj} =44.58% not amazing, but very reasonable for human behavioral modeling (resume scores are subjective!)

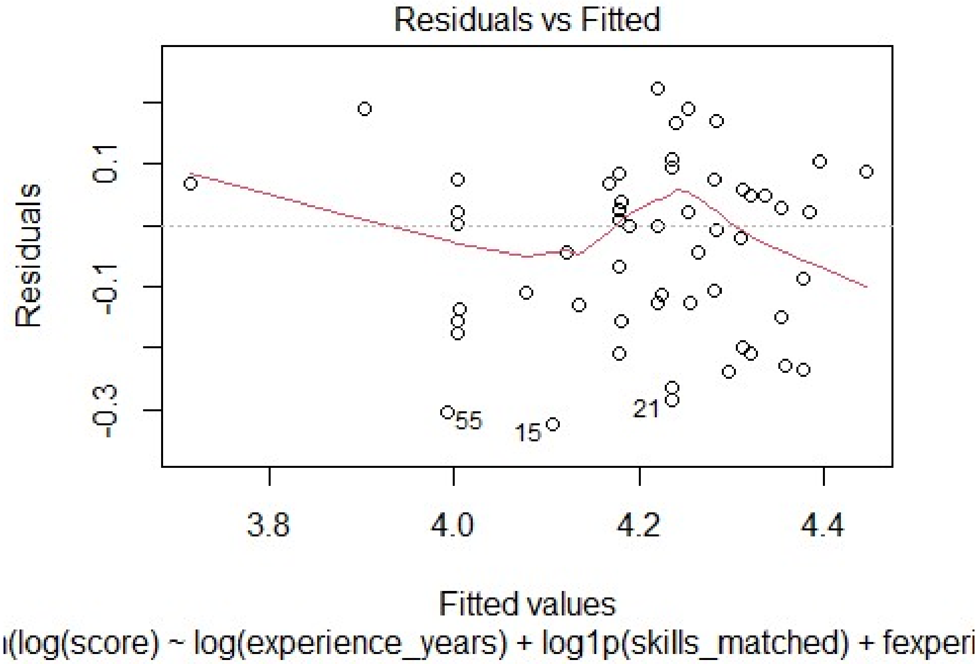
Interpretation of R^2_{adj}:

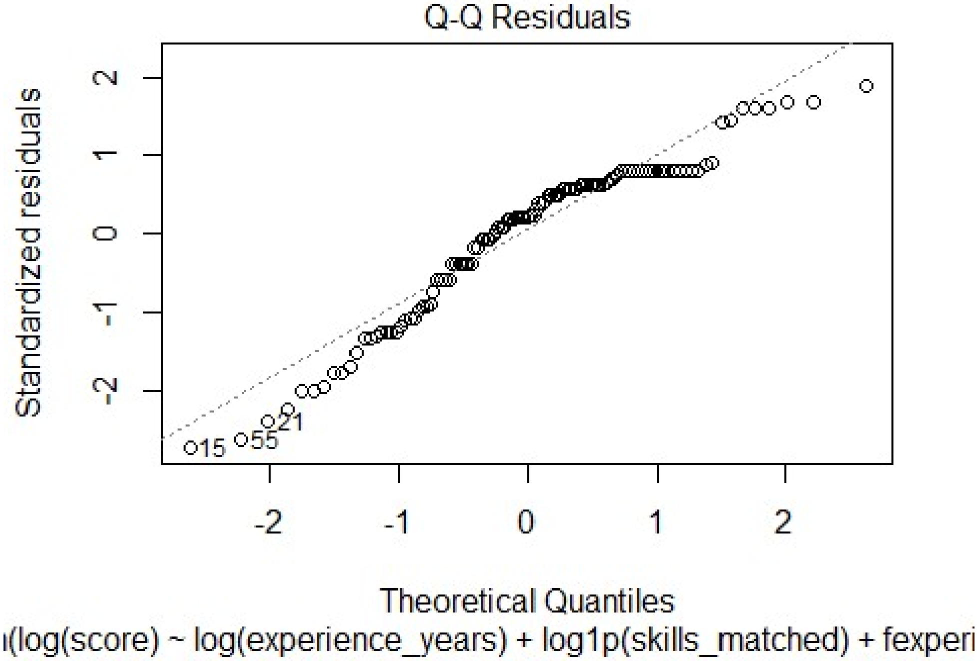
An R-squared of 0.4458 (or 44.58%) means that ~ 44% of the variation in the outcome variable can be attributed to the model's predictor(s). The remaining is attributed to other factors not included in the model or random variation.

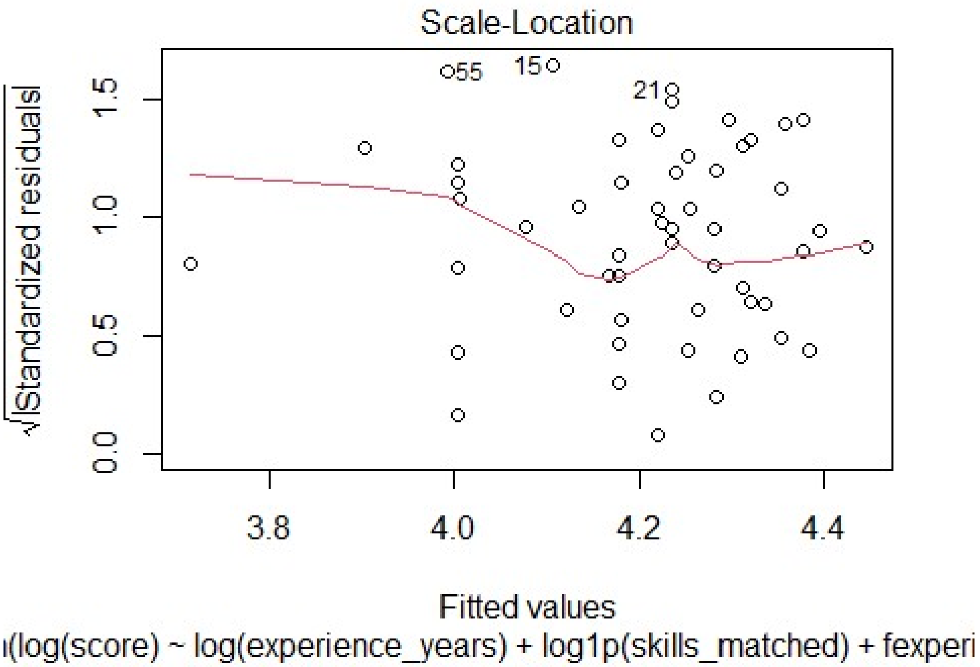
In real-world scenarios, especially with human-centric data like resume screening, 36% isn't bad.

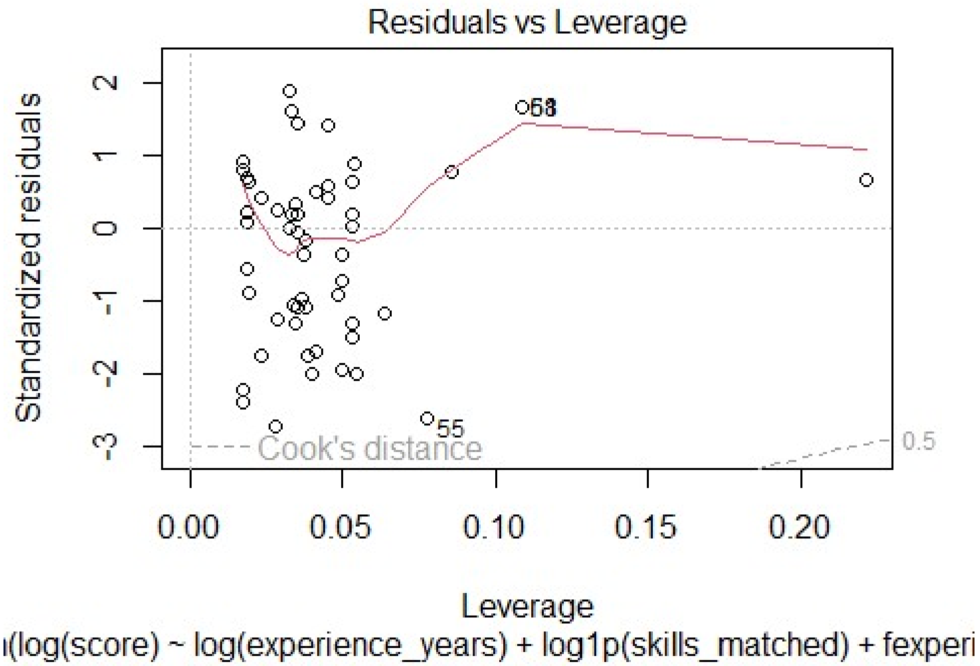
Checking assumptions again for this final model after log transformation.

```
plot(lm3)
```


Residuals vs Fitted
Residuals
0.1
-0.1
-0.3
3.8
4.0
4.2
4.4
55
15
21
Fitted values
ı(log(score) ~ log(experience_years) + log1p(skills_matched) + fexperi


Q-Q Residuals
Standardized residuals
2
1
0
-1
-2
-2
-1
0
1
2
15
55
21
Theoretical Quantiles
ı(log(score) ~ log(experience_years) + log1p(skills_matched) + fexperi





```
shapiro.test(resid(lm3))

## 
##  Shapiro-Wilk normality test
```

```
##
## data:  resid(lm3)
## W = 0.93543, p-value = 3.381e-05
```

```
# Checking VIF value for the transformed model
vif(lm3)
```

```
##                             GVIF Df GVIF^(1/(2*Df))
## log(experience_years) 5.144432  1        2.268134
## log1p(skills_matched) 1.203080  1        1.096850
## fexperience_level     5.543026  2        1.534393
```

Comment on assumptions:
Even after using log transformation the Normality assumption is still not satisfied. The QQPlot still shows some signs of deviation at tails. However, the Residual vs Fitted plot shows a random scatter of points and no fanning pattern is observed therefore, constant variance condition is satisfied in this transformation

Comment on Normality Assumption:
For further confirmation we use Shapiro-Wilk test for confirming our assumption of normality. From the results, we can see that p-value is still very small than our $\alpha\ (significance\ level)$ which means normality assumption is violated.

Normality Assumption Violation Discussion $(n1 >= 30)$
Yes, since we have 158 a decent-sized sample, we are in the "safe zone" for applying CLT to model inference. The CLT will allow us to rely on the regression estimates (coefficients and their significance) for large samples even if the residuals are not perfectly normal. So, the normality of residuals isn't as critical for large samples when you're performing regression.

Comment on Multi-collinearity:
Surprisingly, the multi-collinearity issue has also been fixed to a greater extent after the log transformation.

Final Comments on FINAL MODEL

It is far superior both in Adjusted R^2 and AIC, which suggests:
1. Better model fit
2. Better generalization
3. More appropriate transformation for your data

Logistic Regression WORKS

```
df_resume$fdecision <- as.integer(factor(df_resume$decision, levels =
c("Rejected", "Hired"))) - 1

glm3 <- glm(fdecision ~ (experience_years) + (skills_matched) +
fexperience_level, data=df_resume, family=binomial)
```

```r
summary(glm3)
```

```
##
## Call:
## glm(formula = fdecision ~ (experience_years) + (skills_matched) +
##     fexperience_level, family = binomial, data = df_resume)
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -4.8897     1.6899  -2.894 0.003809 **
## experience_years      1.5633     0.7249   2.157 0.031031 *
## skills_matched        0.3985     0.1132   3.520 0.000431 ***
## fexperience_level1   -5.4953     1.8469  -2.975 0.002925 **
## fexperience_level2  -29.5356  1455.4083  -0.020 0.983809
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 152.06  on 113  degrees of freedom
## Residual deviance: 126.88  on 109  degrees of freedom
## AIC: 136.88
##
## Number of Fisher Scoring iterations: 14
```

```r
predicted_probs <- predict(glm3, type = "response")

predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)
table(Predicted = predicted_classes, Actual = df_resume$fdecision)
```

```
##          Actual
## Predicted  0  1
##         0 60 24
##         1 10 20
```

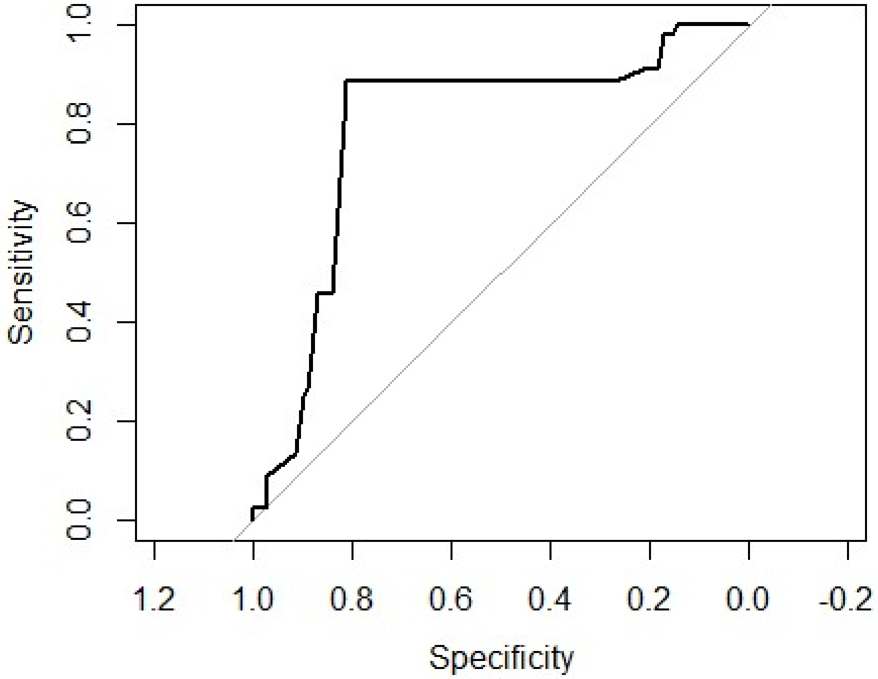
```r
# Accuracy
mean(predicted_classes == df_resume$fdecision)
```

```
## [1] 0.7017544
```

```r
library(pROC)
roc_obj <- roc(df_resume$fdecision, predicted_probs)
auc(roc_obj)
```

```
## Area under the curve: 0.7937
```

```r
plot(roc_obj)
```



Interpretation:

Overall Accuracy (70.2%): Decent — the model is correct ~70% of the time.

Sensitivity (45.5%): This is low — it's only correctly identifying 45.5% of the actual 1s (positives). So if predicting "yes" is crucial, this may be a problem.

Specificity (85.7%): Pretty good — it's doing well at predicting 0s.

AUC (0.7937): This is actually quite good.

AUC > 0.7 = decent

AUC > 0.8 = strong

So 0.79 is close to very good territory.

Final Thoughts and Recommendations

Feature Engineering: Since resume data can be sparse, more sophisticated feature engineering might help improve performance. Consider extracting more meaningful features from the text (e.g. sentiment analysis) or exploring the potential impact of other factors such as education level or job title.

Model Comparison: While the linear models and logistic regression perform well, it might be worth comparing the results with machine learning models like Decision Trees, Random

Forests, or Gradient Boosting (e.g., XGBoost) to see if they offer a significant improvement in predictive accuracy, especially for complex datasets like resumes.