Linear Models

CS 4375 - Intro to Machine Learning

Dr. Karen Mazidi

Author: Leo Nguyen -ldn190002

Date: Sep 25, 2022

Linear Regression Models

- The linear regression model work by finding the relationship between the target value-y and the predictor value-x. This linear relationship can be defined by parameter w and b; which w is the slope line-quantifying the amount that y changes with changes in x; and b is an intercept.
- Strengths of linear regression: It is a relative simple algorithm, it work well when data have linear pattern, and it has low variance
- Weaknesses of linear regression: It is high bias because it assumes a linear shape of data

Physicochemical Properties of Protein Tertiary Structure Data Set

Citation:

ABV - Indian Institute of Information Technology & Management, Gwalior, MP, India. UCI Machine Learning Repository [http://archive.ics.uci.edu/ml (http://archive.ics.uci.edu/ml)]. Irvine, CA: University of California, School of Information and Computer Science.

Attribute Information:

- RMSD-Size of the residue.
- F1 - Total surface area.
- F2 - Non polar exposed area.
- F3 - Fractional area of exposed non polar residue.
- F4 - Fractional area of exposed non polar part of residue.
- F5 - Molecular mass weighted exposed area.
- F6 - Average deviation from standard exposed area of residue.
- F7 - Euclidean distance.
- F8 - Secondary structure penalty.
- F9 - Spacial Distribution constraints (N,K Value).

Load the data

```
df <- read.csv("data/CASP.csv", header=TRUE)
str(df)
```

```
## 'data.frame':    45730 obs. of  10 variables:
##  $ RMSD: num  17.28 6.02 9.28 15.85 7.96 ...
##  $ F1  : num  13558 6192 7726 8425 7461 ...
##  $ F2  : num  4305 1623 1726 2368 1737 ...
##  $ F3  : num  0.318 0.262 0.223 0.281 0.233 ...
##  $ F4  : num  162.2 53.4 67.3 67.8 52.4 ...
##  $ F5  : num  1872791 803447 1075648 1210472 1021020 ...
##  $ F6  : num  215.4 87.2 81.8 109.4 94.5 ...
##  $ F7  : num  4288 3329 2981 3248 2814 ...
##  $ F8  : int  102 39 29 70 41 15 70 74 39 26 ...
##  $ F9  : num  27 38.5 38.8 39.1 39.9 ...
```

Divide into 80/20 train/test

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Data Exploration

1. Look at the first 10 rows in training data

```
head(train, n=10)
```

```
##         RMSD       F1      F2      F3       F4        F5       F6      F7  F8
## 40784  4.750  7085.06 1713.77 0.24188  71.0054  964112.9  94.5983 3096.25  44
## 40854 16.247 10964.20 3078.91 0.28081 119.8650 1608625.6 141.0520 3745.58 139
## 41964 16.083  6926.30 2033.94 0.29365  69.2471  956051.2  99.5273 3180.47  37
## 15241  1.600 15273.60 4826.20 0.31598 160.1420 2116707.0 233.5890 5339.67 200
## 33702  4.240  6999.15 1648.37 0.23551  70.6835  949550.4  93.3331 3183.40  51
## 35716  2.988  8493.93 3658.40 0.43070  80.7255 1202937.5 138.3440 3407.53  41
## 17487  5.824  4984.98 1395.35 0.27991  43.4339  686591.4  63.0595 2810.20  25
## 15220  2.849  9389.75 2156.41 0.22965 114.0400 1306068.5 128.4600 4221.14  51
## 19838  1.861 15649.50 5251.22 0.33555 218.4740 2175659.5 256.4370 5372.32 287
## 2622  12.809  4120.18 1641.61 0.39843  38.3952  555175.6  56.8550 1401.72  52
##            F9
## 40784 36.6171
## 40854 36.0505
## 41964 34.9845
## 15241 26.5049
## 33702 37.6965
## 35716 37.1324
## 17487 41.5209
## 15220 34.1424
## 19838 24.3666
## 2622  44.2281
```

2. View the summary of entire training data set

```
summary(train)
```

```
##       RMSD              F1              F2               F3        
##  Min.   : 0.000   Min.   : 2392   Min.   :  403.5   Min.   :0.09362  
##  1st Qu.: 2.309   1st Qu.: 6939   1st Qu.: 1979.1   1st Qu.:0.25854  
##  Median : 5.010   Median : 8896   Median : 2666.4   Median :0.29985  
##  Mean   : 7.746   Mean   : 9873   Mean   : 3014.8   Mean   :0.30211  
##  3rd Qu.:13.389   3rd Qu.:12136   3rd Qu.: 3783.4   3rd Qu.:0.34260  
##  Max.   :20.999   Max.   :40035   Max.   :15061.6   Max.   :0.57769  
##        F4               F5                F6               F7        
##  Min.   : 10.69   Min.   : 319490   Min.   : 31.97   Min.   :     0  
##  1st Qu.: 63.59   1st Qu.: 954062   1st Qu.: 94.60   1st Qu.:  3165  
##  Median : 87.66   Median :1236808   Median :126.20   Median :  3840  
##  Mean   :103.59   Mean   :1368628   Mean   :145.59   Mean   :  3982  
##  3rd Qu.:133.87   3rd Qu.:1691223   3rd Qu.:181.36   3rd Qu.:  4645  
##  Max.   :369.32   Max.   :5472011   Max.   :598.41   Max.   :105948  
##        F8               F9        
##  Min.   :  0.00   Min.   :15.23  
##  1st Qu.: 31.00   1st Qu.:30.39  
##  Median : 54.00   Median :35.29  
##  Mean   : 69.95   Mean   :34.51  
##  3rd Qu.: 91.00   3rd Qu.:38.86  
##  Max.   :350.00   Max.   :55.07
```

3. Count number of NA value by column

```
colSums(is.na(train))
```

```
## RMSD   F1   F2   F3   F4   F5   F6   F7   F8   F9 
##    0    0    0    0    0    0    0    0    0    0
```

4. Check the correlation of all column in training data

- Based on the number we can see that F1 is highly correlation with F2, F4, F5, F6, F9

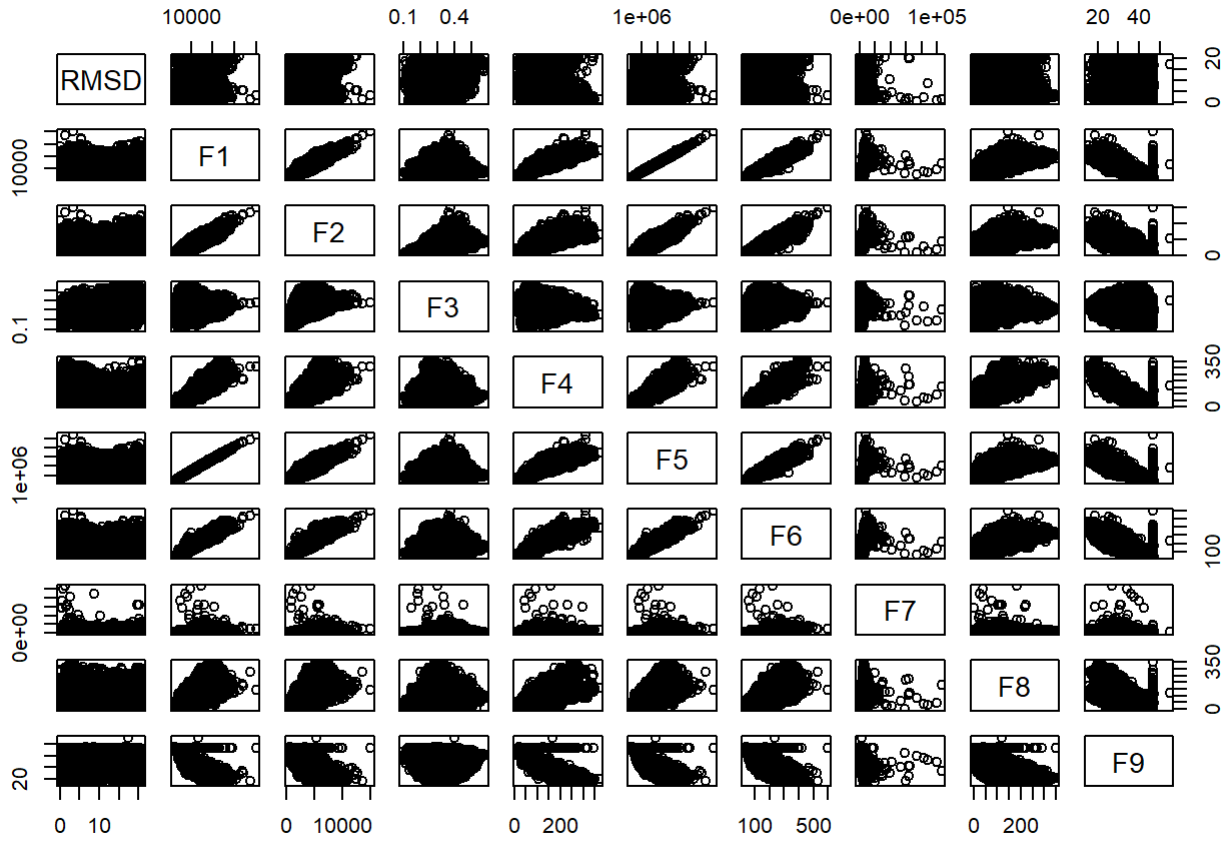
```
cor(train)
```

```
##              RMSD          F1         F2          F3          F4          F5
## RMSD  1.000000000 -0.01149595  0.1620261  0.37722744 -0.16683685 -0.01034643
## F1   -0.011495953  1.00000000  0.9061435  0.12558257  0.93175812  0.99819519
## F2    0.162026065  0.90614351  1.0000000  0.50305999  0.79388071  0.90247716
## F3    0.377227438  0.12558257  0.5030600  1.00000000  0.03130315  0.12199103
## F4   -0.166836853  0.93175812  0.7938807  0.03130315  1.00000000  0.92644583
## F5   -0.010346432  0.99819519  0.9024772  0.12199103  0.92644583  1.00000000
## F6   -0.032750179  0.96759157  0.9072965  0.19883974  0.93902201  0.96184477
## F7   -0.003228492  0.56641558  0.5266875  0.08042454  0.49927458  0.56596977
## F8    0.003019686  0.65155381  0.5848816  0.09619772  0.67717518  0.64317528
## F9    0.058850933 -0.90097656 -0.7898221 -0.07039026 -0.89274895 -0.90042132
##               F6           F7           F8          F9
## RMSD -0.03275018 -0.003228492  0.003019686  0.05885093
## F1    0.96759157  0.566415585  0.651553810 -0.90097656
## F2    0.90729647  0.526687482  0.584881581 -0.78982213
## F3    0.19883974  0.080424541  0.096197720 -0.07039026
## F4    0.93902201  0.499274582  0.677175178 -0.89274895
## F5    0.96184477  0.565969768  0.643175285 -0.90042132
## F6    1.00000000  0.549312415  0.663242948 -0.88509884
## F7    0.54931241  1.000000000  0.360079385 -0.53716251
## F8    0.66324295  0.360079385  1.000000000 -0.63935345
## F9   -0.88509884 -0.537162515 -0.639353454  1.00000000
```

5. Quick visualization on correlation

- These plots confirm the highly correlation of F1 and other attributes mentioned above

```
pairs(train)
```



6. Summary() for F1 and F5 in training data

Summary for F1 - Total surface area.

```
summary(train$F1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2392    6939    8896    9873   12136   40035
```

Summary for F5 - Molecular mass weighted exposed area.

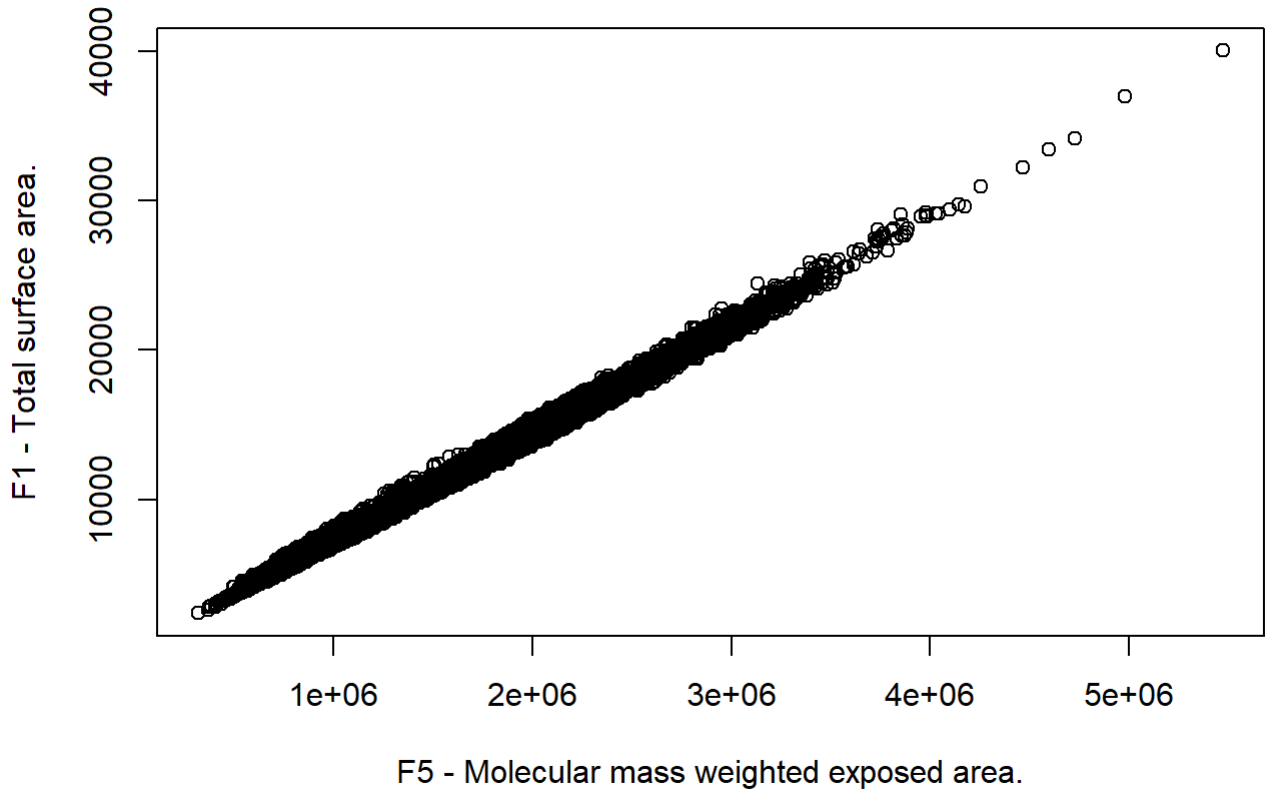
```
summary(train$F5)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  319490  954062 1236808 1368628 1691223 5472011
```

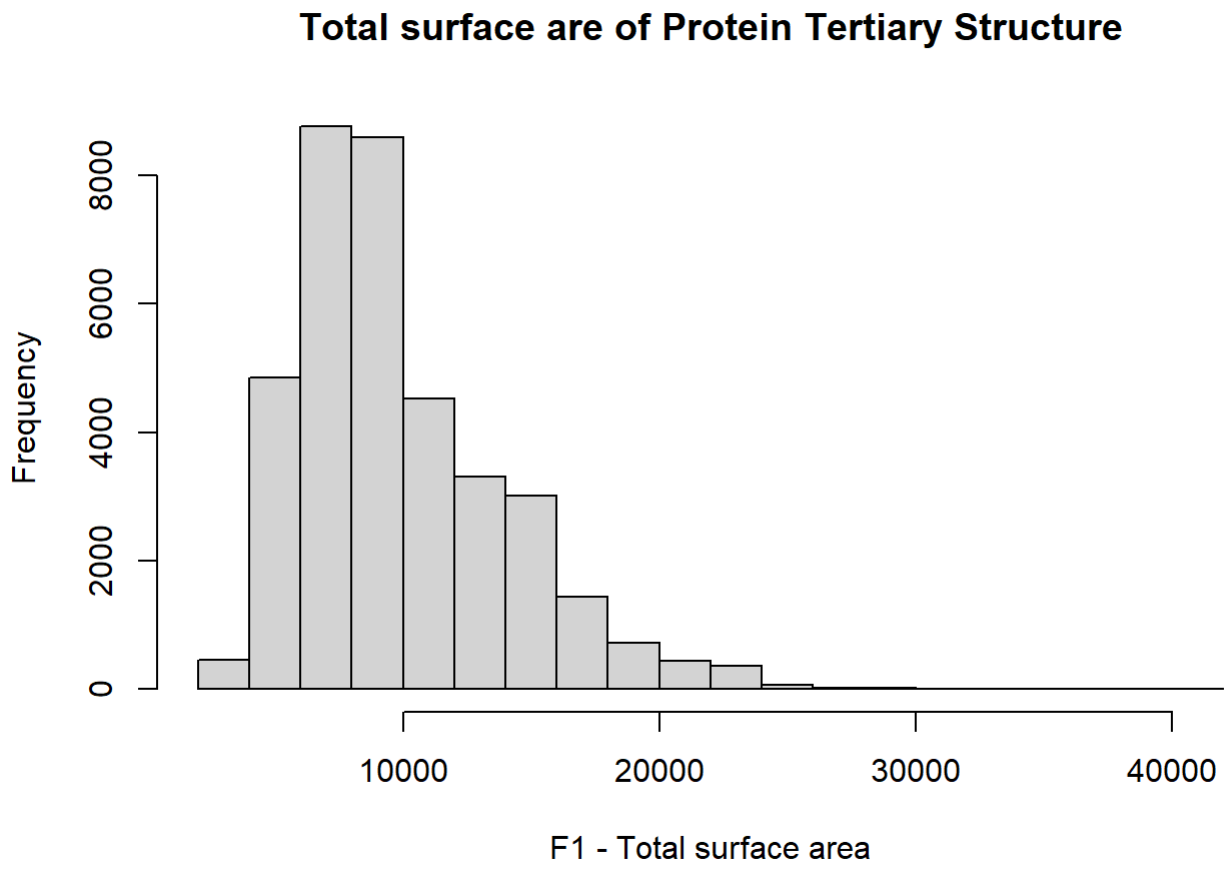
Data Visualization using informative graph

1. Plot show the relationship between F1 - Total surface area and F5 - Molecular mass weighted exposed area.

```
plot(train$F5,train$F1 ,xlab = "F5 - Molecular mass weighted exposed area.", ylab = "F1 - Total
 surface area." )
```




```
hist(train$F1, xlab = "F1 - Total surface area", main = "Total surface are of Protein Tertiary S
tructure")
```



Simple Linear Regression Model (One Predictor)

```
lm1 <- lm(F1~F5, data = train)
summary(lm1)
```

```
##
## Call:
## lm(formula = F1 ~ F5, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1003.87  -154.59   -14.84   133.81  1913.76
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.472e+01  3.344e+00   13.38   <2e-16 ***
## F5          7.181e-03  2.259e-06 3179.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 243.6 on 36582 degrees of freedom
## Multiple R-squared:  0.9964, Adjusted R-squared:  0.9964
## F-statistic: 1.011e+07 on 1 and 36582 DF,  p-value: < 2.2e-16
```

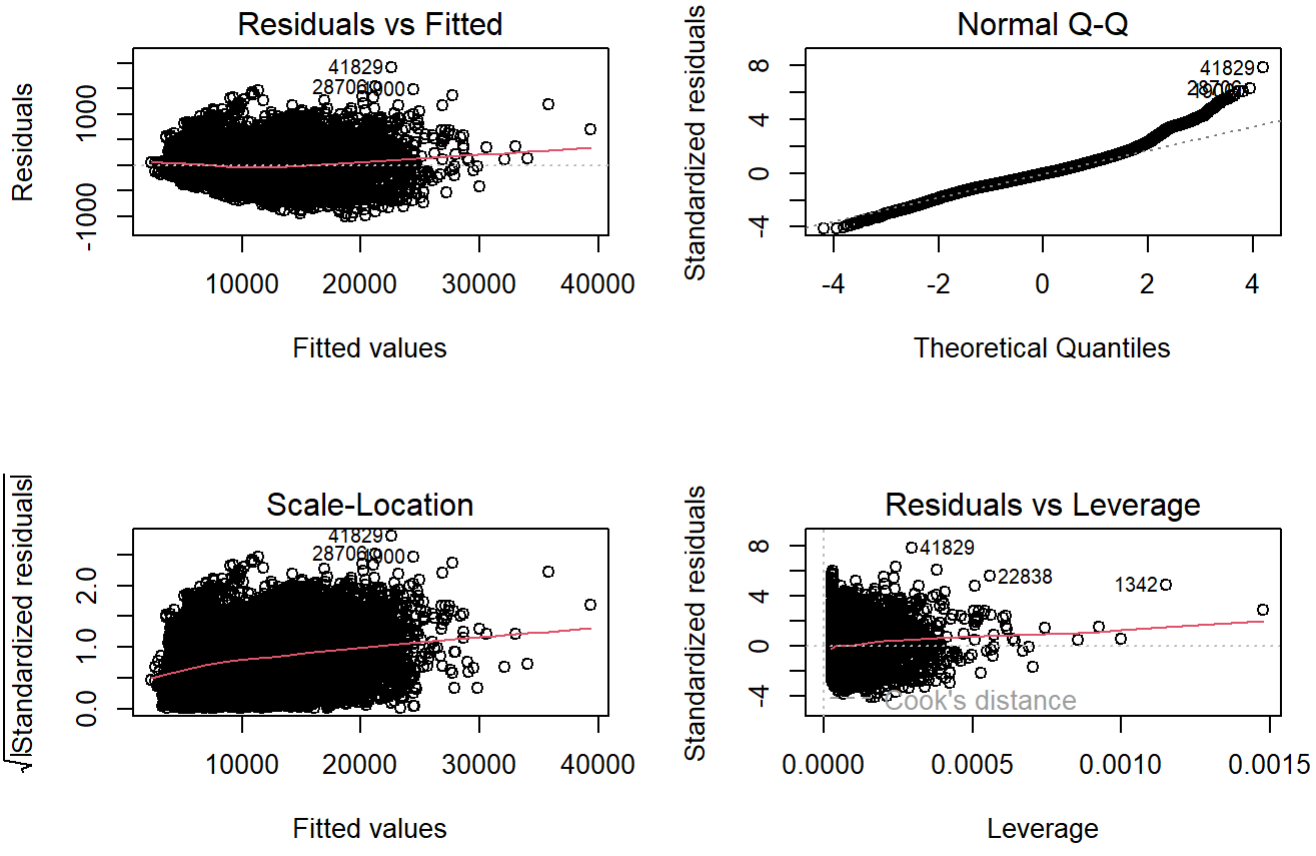
Model Summary Explanation

- The model was build to predict the target column F1 - Total surface area using the predictor F5 - Molecular mass weighted exposed area using the training data which contain 80% observations from original data set The Estimate column in Coefficients section provide us 2 important values w=4.472e+1, b=7.181e-03. It mean for each unit F5 increase, we expect 7.181e-03 unit of F1 more. The 3 asterisks in train$F5 indicate that F5 is a good predictor.The p value is also very small which mean we can reject the null hypothesis –> good model.
- The RSE = 243.6 is also small compare the the value of F1 which has the mean = 9783 (See detail on F1 summary above) and this RSE value was calculated from a large number of sample 36582 which make it more reliable. All this let us know that the model is a good fit. The R-square = 0.9964 which is very close to 1 mean almost all the variation in F1 can be predicted by F5.
- Finally, the F-statistic = 1.011e07 > 1 and low p-value also confirm the confidence in the model

Plotting the residuals

Plotting

```
par(mfrow=c(2,2))
plot(lm1)
```



Residual Plots Explanation 1. Plot 1: Residuals vs Fitted - The plot show that we have a fairly horizontal line without distinct patterns which is good indicator that we don't have non-linear relationships. Notice that, we don't have a lot of residual on the far right which mean we don't have much data when the value of F1 and F5 become bigger which was confirm in the graph above.

2. Plot 2: Normal Q-Q

- This plot show us if the residuals are normally distributed. We expect a fairly straight diagonal line following the dashed line. We can see that from 1 to 3 quarter, the line acting exactly what we want: the residuals are lined up well and close the dash line. However, moving forward the Q4, the residuals start to deviate from the dash-line as we don't have much data in that region.

3. Plot 3: Scale-Location

- This plot show if residuals are spread equally along the ranges of predictors. Based on the result, we have a fairly horizontal line and most of residuals equally spread around the line which indicate a good equal variance on the data. As from previous as, we don't have much residuals on the far right as lack of data

4. Plot 4: Residuals vs Leverage

- This plot will indicate leverage points which are influencing the regression line. We can see we also have fairly horizontal line which indicate the result would not be much different if we include or exclude the extreme values from analysis.

Multiple Linear Regression Model (Multiple Predictors)

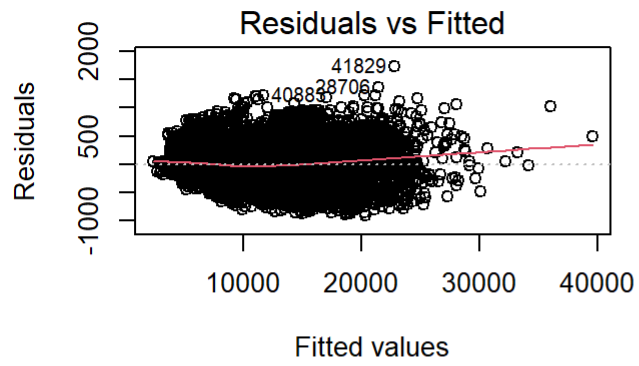
Create a model using F2 - Non polar exposed area and F5 - Molecular mass weighted exposed area as predictors

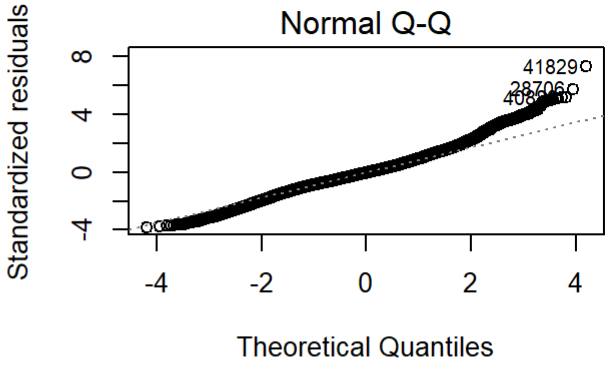
```
lm2 <- lm(F1~F2+F5, data=train)
summary(lm2)
```

```
##
## Call:
## lm(formula = F1 ~ F2 + F5, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -913.78 -148.94  -12.66  130.66 1737.10
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.951e+01  3.294e+00   18.07   <2e-16 ***
## F2          7.921e-02  1.980e-03   40.00   <2e-16 ***
## F5          6.996e-03  5.133e-06 1362.90   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 238.4 on 36581 degrees of freedom
## Multiple R-squared:  0.9965, Adjusted R-squared:  0.9965
## F-statistic: 5.275e+06 on 2 and 36581 DF,  p-value: < 2.2e-16
```

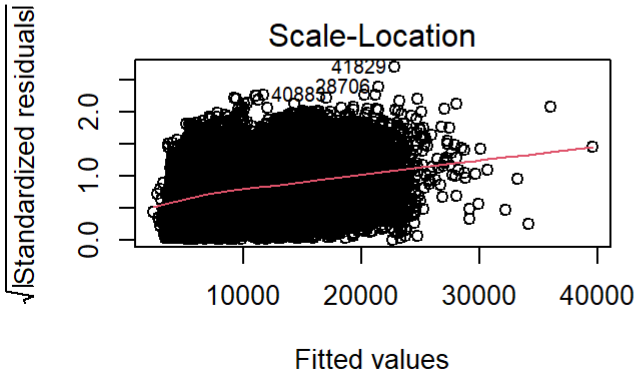
Plotting the residuals for lm2

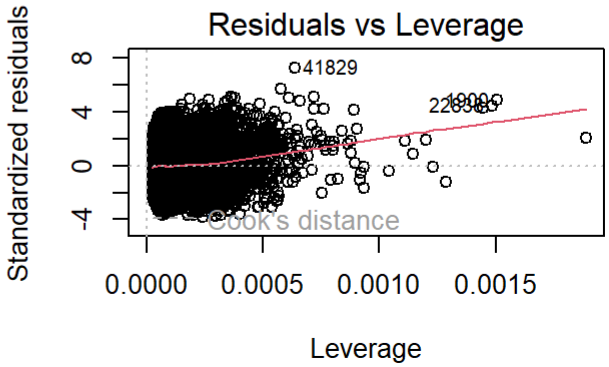
```
par(mfrow=c(2,2))
plot(lm2)
```





###





Build 3rd Linear Regression Model using all other column as predictors

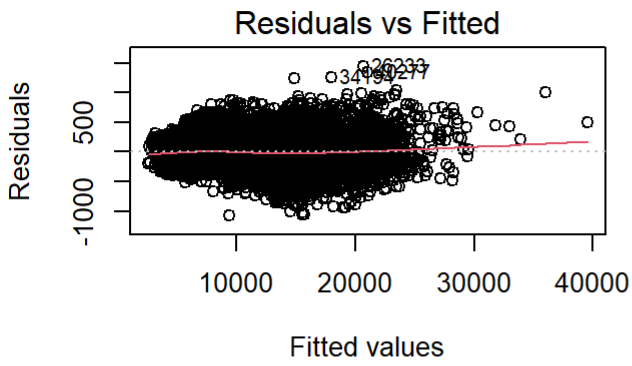
Build and output the summary

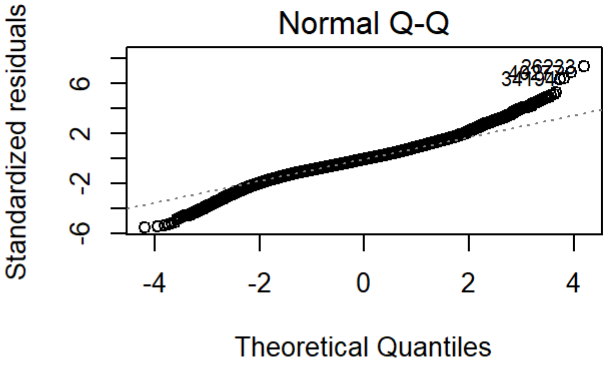
```
lm3 <- lm(F1~., data=train)
summary(lm3)
```

```
##
## Call:
## lm(formula = F1 ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1075.61  -120.50    -7.78   111.87  1444.92
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.032e+03  2.444e+01  42.239  < 2e-16 ***
## RMSD         2.185e+00  1.980e-01  11.037  < 2e-16 ***
## F2           3.525e-01  4.418e-03  79.800  < 2e-16 ***
## F3          -3.237e+03  4.215e+01 -76.811  < 2e-16 ***
## F4           2.841e+00  7.121e-02  39.887  < 2e-16 ***
## F5           5.706e-03  1.174e-05 485.976  < 2e-16 ***
## F6           3.510e+00  7.197e-02  48.779  < 2e-16 ***
## F7           2.507e-03  6.601e-04   3.797 0.000147 ***
## F8           5.114e-01  2.519e-02  20.300  < 2e-16 ***
## F9           2.263e+00  4.279e-01   5.290 1.23e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 196.3 on 36574 degrees of freedom
## Multiple R-squared:  0.9977, Adjusted R-squared:  0.9977
## F-statistic: 1.731e+06 on 9 and 36574 DF,  p-value: < 2.2e-16
```

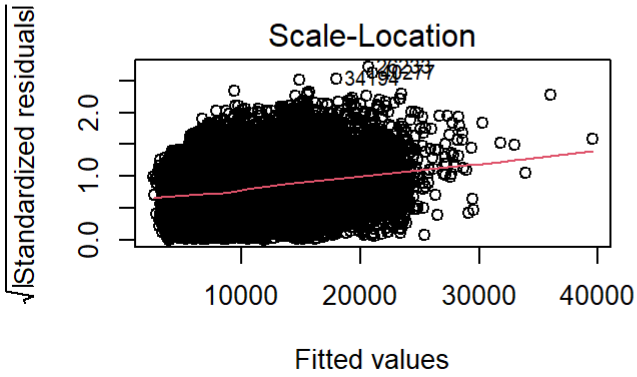
Plotting residuals for lm3

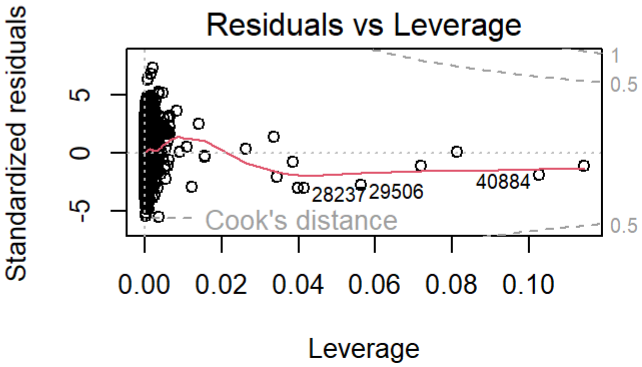
```
par(mfrow=c(2,2))
plot(lm3)
```





###





Comparing the summary result of 3 models

- Among 3 models, the lm3 provide the best result. It is because lm3 have the biggest R-square value (0.9977) compare to other model. It also has the lowest RSE (196.3) compare to other model. The lm3 achieve all that while maintain other indicator such p-value low and relative close to other model.

Comparing 3 models using correlation metric and mse

1. Calculate correlation and mse for lm1

```
pred1 <- predict(lm1, newdata=test)
cor1 <- cor(pred1, test$F1)
mse1 <- mean((pred1-test$F1)^2)
rmse1 <- sqrt(mse1)
print(paste('correlation:', cor1))
```

```
## [1] "correlation: 0.998144843343622"
```

```
print(paste('mse:', mse1))
```

```
## [1] "mse: 61338.2754796574"
```

```
print(paste('rmse:', rmse1))
```

```
## [1] "rmse: 247.665652603782"
```

2. Calculate correlation and mse for lm2

```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$F1)
mse2 <- mean((pred2-test$F1)^2)
rmse2 <- sqrt(mse2)
print(paste('correlation:', cor2))
```

```
## [1] "correlation: 0.998235386664316"
```

```
print(paste('mse:', mse2))
```

```
## [1] "mse: 58338.8962783763"
```

```
print(paste('rmse:', rmse2))
```

```
## [1] "rmse: 241.534461885621"
```

3. Calculate correlation and mse for lm3

```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$F1)
mse3 <- mean((pred3-test$F1)^2)
rmse3 <- sqrt(mse3)
print(paste('correlation:', cor3))
```

```
## [1] "correlation: 0.998818194110965"
```

```
print(paste('mse:', mse3))
```

```
## [1] "mse: 39080.7784530179"
```

```
print(paste('rmse:', rmse3))
```

```
## [1] "rmse: 197.688589587305"
```

Conclusion

- Based on the results above, the best model can be ranked as lm3 > lm2 > lm1. The ranking was very inline with the order of correlation and mse values. Even before, we calculated these values, we could guess who is the winner based on the summary output. These calculation can be served as a confirmation of the result.