CS-E-106: Data Modeling

Assignment 5

Instructor: Hakan Gogtas
Submitted by: *Saurabh Kulkarni*

Due Date: *11/04/2019*

```r
library(ggplot2)
library(MASS)
library(lattice)
```

Solution 1:

(a)

(1)

$$X'X = \begin{pmatrix} n & \sum X_i \\ \sum X_i & \sum X_i^2 \end{pmatrix}$$

Using rule 5.22: $a = n$ $b = c = \sum X_i$ $d = \sum X_i^2$

$D = n \sum X_i^2 - (\sum X_i)(\sum X_i = n[\sum X_i^2 - \frac{(\sum X_i)^2}{n}]) = n \sum (X_i - \bar{X})^2$

$$(X'X)^{-1} = \begin{pmatrix} \frac{\sum X_i^2}{n \sum (X_i - \bar{X})^2} & \frac{-\sum X_i}{n \sum (X_i - \bar{X})^2} \\ \frac{-\sum X_i}{n \sum (X_i - \bar{X})^2} & \frac{n}{n \sum (X_i - \bar{X})^2} \end{pmatrix}$$

However, $\sum X_i = n\bar{X}$ and $\sum (X_i - \bar{X})^2 = \sum X_i^2 - n\bar{X}^2$

$$(X'X)^{-1} = \begin{pmatrix} \frac{1}{n} + \frac{\bar{X}^2}{\sum (X_i - \bar{X})^2} & \frac{-\bar{X}}{\sum (X_i - \bar{X})^2} \\ \frac{-\bar{X}}{\sum (X_i - \bar{X})^2} & \frac{1}{\sum (X_i - \bar{X})^2} \end{pmatrix}$$

(2)

$nb_0 + b_1 \sum X_i = \sum Y_i$

$b_0 \sum X_i + b_1 \sum X_i^2 = \sum X_i Y_i$

$\implies X'Xb = X'Y$

$\implies b = (X'X)^{-1} X'Y$

(3)

Normal error regression model: $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$

$$Y = \begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ . \\ Y_n \end{pmatrix}$$

$$X = \begin{pmatrix} 1 & X_1 \\ 1 & X_1 \\ . & . \\ . & . \\ . & . \\ 1 & X_n \end{pmatrix}$$

$$\beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$$

$$\epsilon = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ . \\ . \\ . \\ \epsilon_n \end{pmatrix}$$

$$\begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ . \\ Y_n \end{pmatrix} = \begin{pmatrix} 1 & X_1 \\ 1 & X_1 \\ . & . \\ . & . \\ . & . \\ 1 & X_n \end{pmatrix}$$

$$\begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ . \\ . \\ . \\ \epsilon_n \end{pmatrix}$$

$$\therefore \hat{Y} = X\beta$$

(4)

$$\hat{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ . \\ Y_n \end{pmatrix}$$

$= Xb = X(X'X)^{-1}X'Y = HY$

$\implies H = X(X'X)^{-1}X'$

(5)

$$e = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ . \\ . \\ . \\ \epsilon_n \end{pmatrix} = Y - \hat{Y} = Y - X'b$$

$$SSE = e'e = (Y - Xb)'(Y - Xb) = Y'Y - b'X'Y$$

(6)

$$\sigma^2 b = \sigma^2 (X'X)^{-1}$$

$$\sigma^2 b = \begin{pmatrix} \frac{\sigma^2}{n} + \frac{\sigma^2 \bar{X}^2}{\sum (X_i - \bar{X})^2} & \frac{-\bar{X}\sigma^2}{\sum (X_i - \bar{X})^2} \\ \frac{-\bar{X}\sigma^2}{\sum (X_i - \bar{X})^2} & \frac{\sigma^2}{\sum (X_i - \bar{X})^2} \end{pmatrix}$$

$$s^2 b = MSE(X'X)^{-1} = \begin{pmatrix} \frac{MSE}{n} + \frac{MSE\bar{X}^2}{\sum (X_i - \bar{X})^2} & \frac{-\bar{X}MSE}{\sum (X_i - \bar{X})^2} \\ \frac{-\bar{X}MSE}{\sum (X_i - \bar{X})^2} & \frac{MSE}{\sum (X_i - \bar{X})^2} \end{pmatrix}$$

(7)

$$s^2 pred = MSE(1 + X_h'(X'X)^{-1} X_h)$$

At $X_h = 30$,

$s^2 pred = MSE(1 + 30^2 (X'X)^{-1})$

(b)

$$\sigma^2 b = \begin{pmatrix} \sigma^2(b_0) & \sigma(b_0, b_1) \\ \sigma(b_0, b_1) & \sigma^2(b_1) \end{pmatrix}$$

Thus, from part(a)(6):

$s^2(b_0) = \frac{MSE}{n} + \frac{MSE\bar{X}^2}{\sum (X_i - \bar{X})^2}$

$s(b_0, b_1) = \frac{-\bar{X}MSE}{\sum (X_i - \bar{X})^2}$

$s^2(b_1) = \frac{MSE}{\sum (X_i - \bar{X})^2}$

(c)

From part(a)(5),

$$SSE = e'e = (Y - Xb)'(Y - Xb) = Y'Y - b'X'Y$$

$b'X' = (Xb)' = \hat{Y}' = (HY)'$

From Hat matrix part(a)(3-4):

$b'X' = (HY)'$

H is symmetric, so H' = H. Hence,

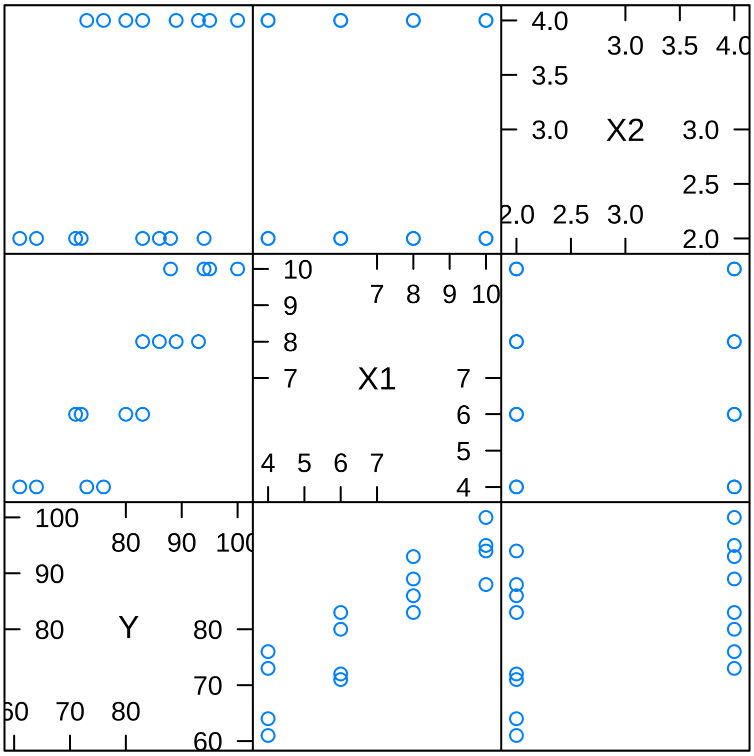
$b'X' = Y'H$

$\implies SSE = Y'(I - H)Y$

Solution 2:

(a)

```
brand_data = read.csv("Brand Preference.csv")
splom(brand_data)
```

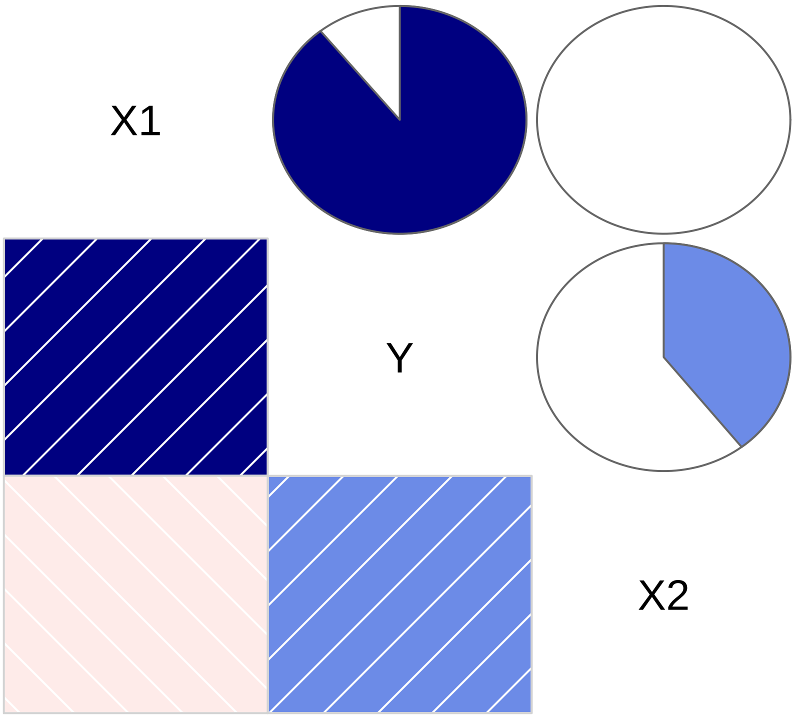


Scatter Plot Matrix

```r
library(corrgram)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus

##
## Attaching package: 'corrgram'

## The following object is masked from 'package:lattice':
##
##     panel.fill
```

```r
corrgram(brand_data, order=TRUE, lower.panel=panel.shade,
  upper.panel=panel.pie, text.panel=panel.txt, oma=c(3,3,3,15),
  main="Correlogram")
```




```r
cor(brand_data)
```

```
##            Y        X1        X2
## Y  1.0000000 0.8923929 0.3945807
## X1 0.8923929 1.0000000 0.0000000
## X2 0.3945807 0.0000000 1.0000000
```

Interpretation:

We can see a linear relationship between X1 and Y ($r \approx 0.9$). However, there seems to be little correlation between X2 and Y, or X2 and X1 either.

(b)

```r
lm_brand = lm(Y~., data=brand_data)
summary(lm_brand)
```

```
##
## Call:
## lm(formula = Y ~ ., data = brand_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.400 -1.762  0.025  1.587  4.200
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.6500     2.9961  12.566 1.20e-08 ***
## X1            4.4250     0.3011  14.695 1.78e-09 ***
## X2            4.3750     0.6733   6.498 2.01e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.693 on 13 degrees of freedom
## Multiple R-squared:  0.9521, Adjusted R-squared:  0.9447
## F-statistic: 129.1 on 2 and 13 DF,  p-value: 2.658e-09
```

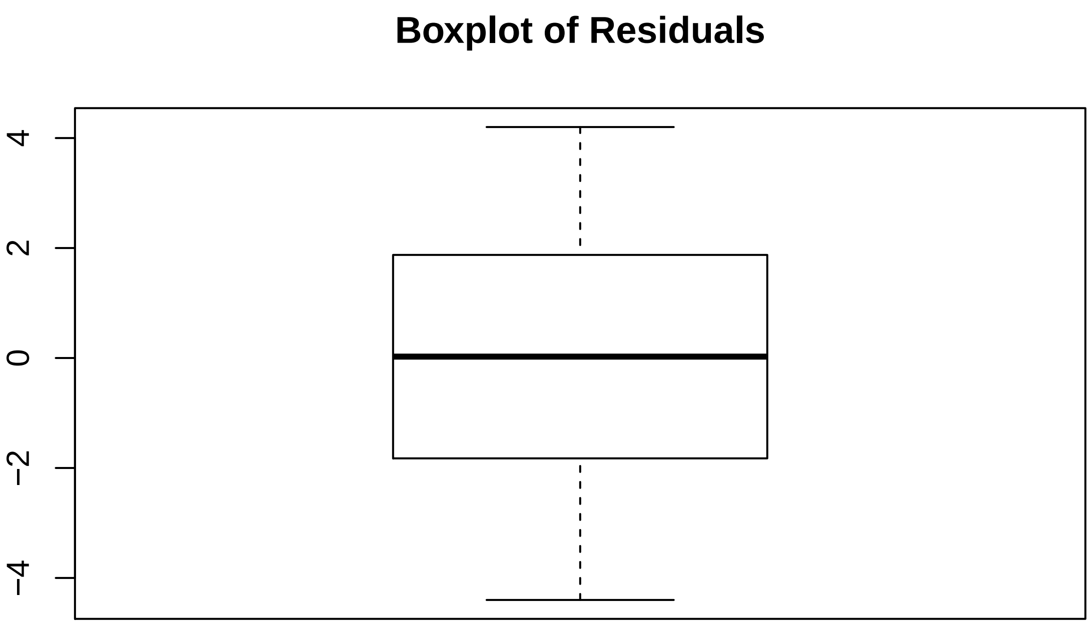
Estimated Regression Function: $Y = 37.65 + 4.425 * X1 + 4.375 * X1$

Interpretation:

Based on the regression function, none of the β's seem to be zero. β_2 does has a higher standard error and a greater p-value, which means X1 is more correlated to Y compared to X2. Also, the model is a very good fit ($R^2 = 0.95$).

(c)

```
ei = lm_brand$residuals
boxplot(ei)
title(main="Boxplot of Residuals")
```

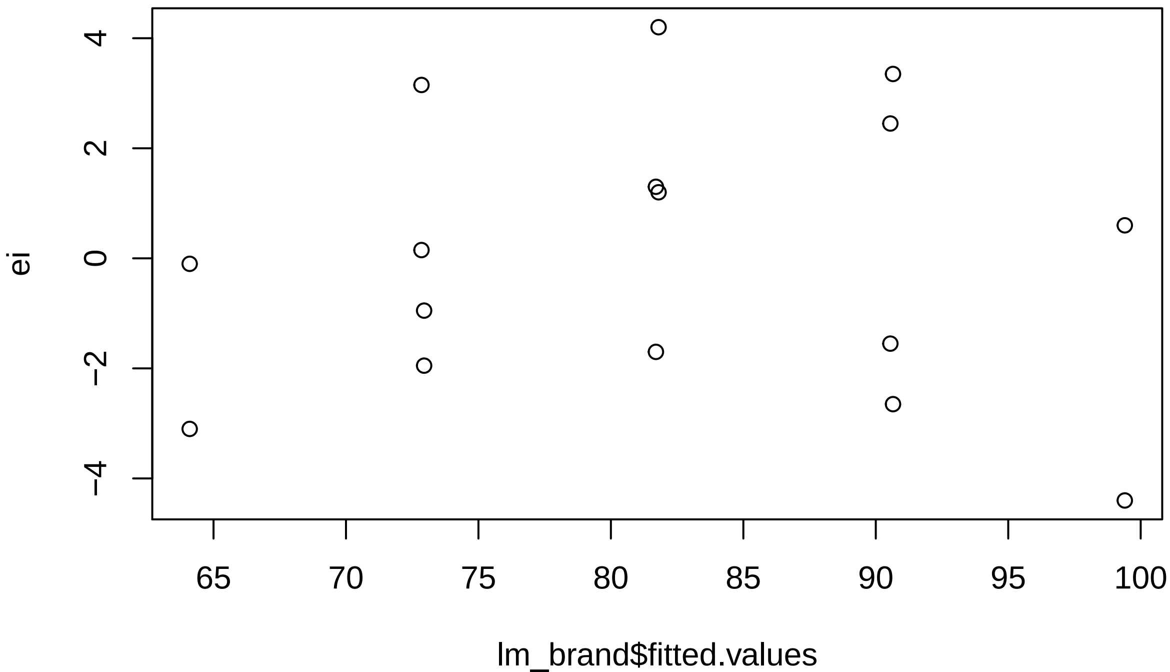


Interpretation:

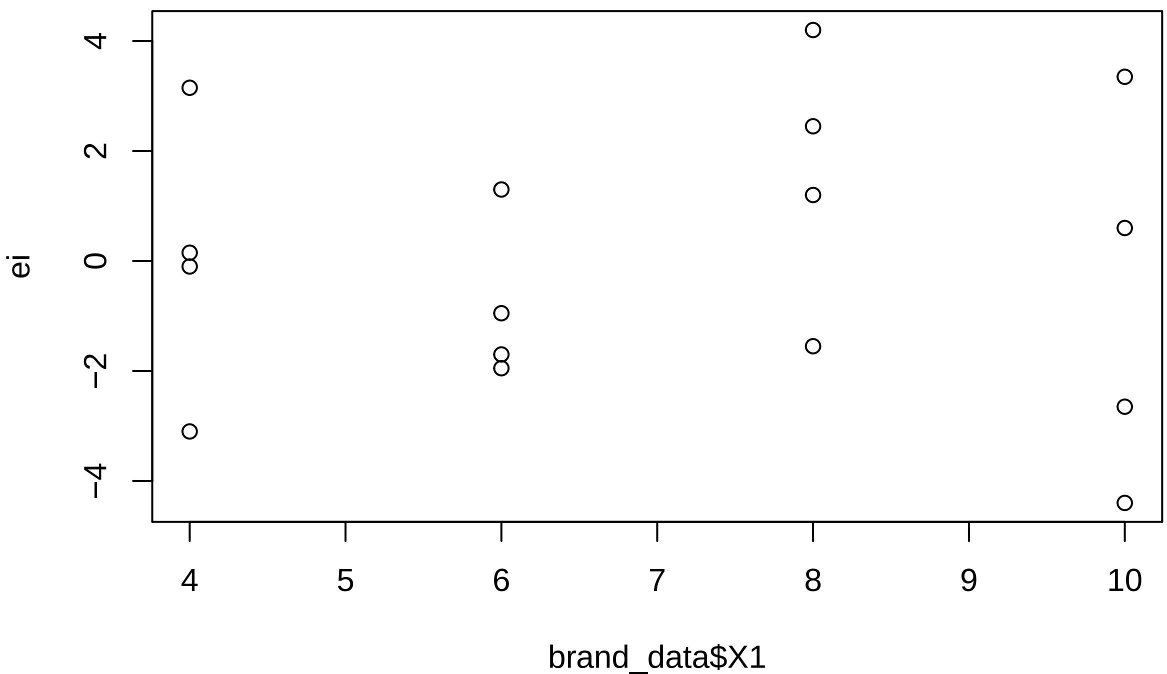
We see that we don't have any outliers in the error term based on the box plot. Also, it seems to be evenly spread around 0.

(d)

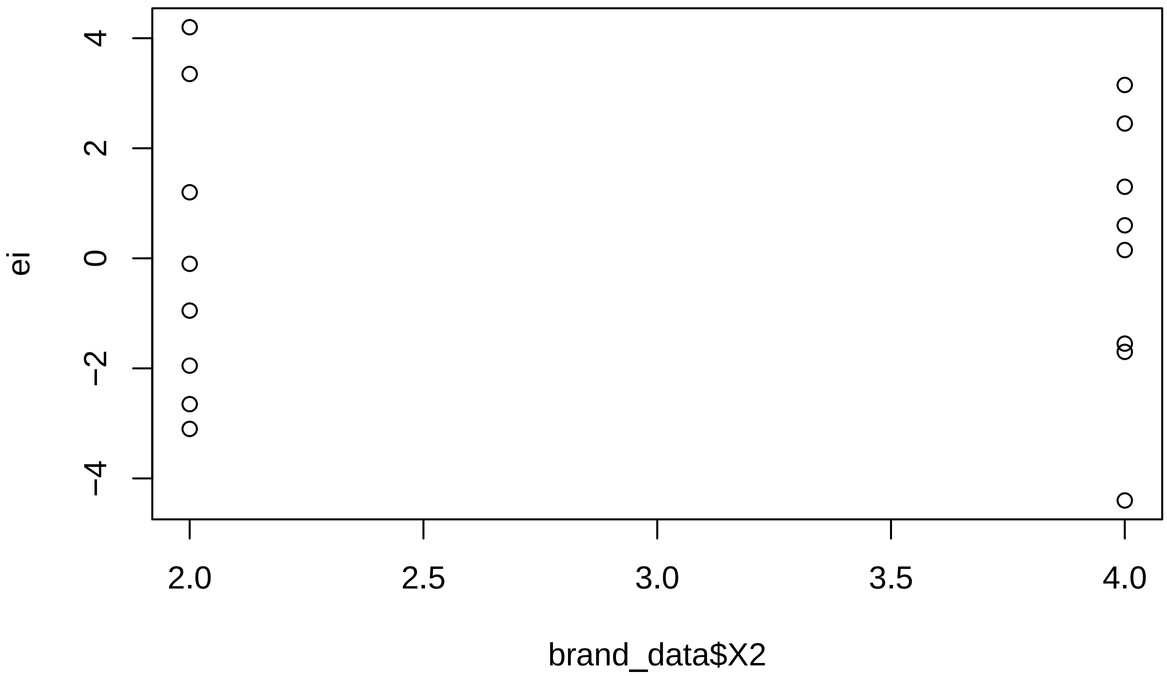
```
plot(lm_brand$fitted.values, ei)
```



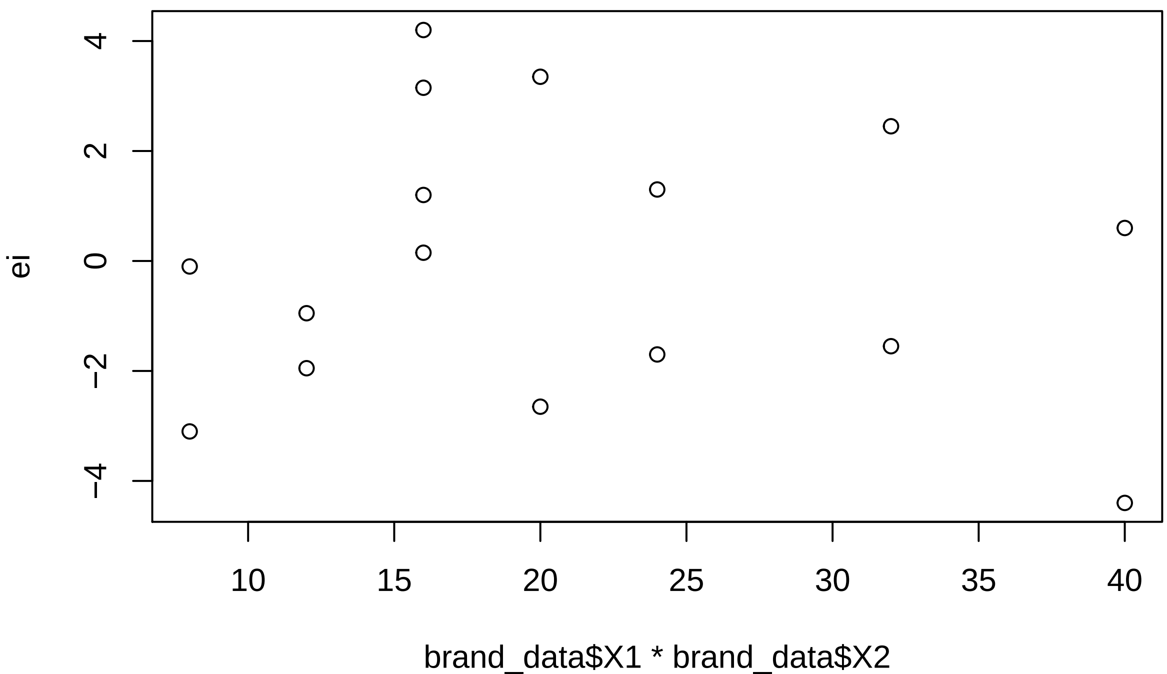

```
plot(brand_data$X1, ei)
```




```
plot(brand_data$X2, ei)
```




```
plot(brand_data$X1*brand_data$X2, ei)
```

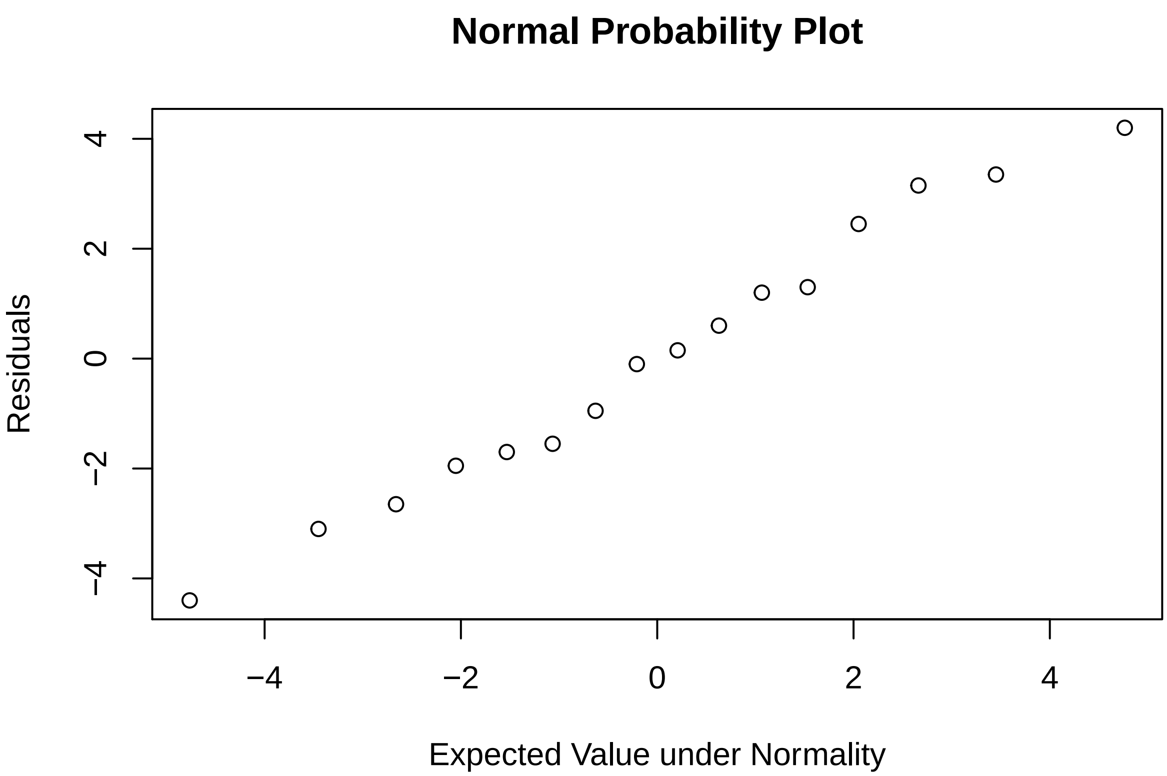



```
df=brand_data
rse=2.693

ri = rank(ei)
n = nrow(df)
zr = (ri-0.375)/(n+0.25)

#residual standard error from summary(lm) above
zr1 = rse*qnorm(zr)
```

```r
print(cor.test(zr1, ei))
```

```
##
##  Pearson's product-moment correlation
##
## data:  zr1 and ei
## t = 31.285, df = 14, p-value = 2.338e-14
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9791573 0.9976086
## sample estimates:
##       cor
## 0.9929238
```

```r
plot(zr1, ei, xlab="Expected Value under Normality",ylab="Residuals")
title(main="Normal Probability Plot")
```



Interpretation:

Residual Plots: The residuals appear to be equally spread and have no distinct patterns. We can say that there is contant variance in the error term.

Normal Probability Plot: The plot seems to be almost linear, which means that the error is in agreement with the normality.

(e)

Null Hypothesis: H_0: Error variance is constant Alternate Hypothesis: H_1: Error variance is not constant

```r
ei2 = ei^2
f = lm(ei2~brand_data$X1+brand_data$X2)
summary(f)
```

```
##
## Call:
## lm(formula = ei2 ~ brand_data$X1 + brand_data$X2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -7.724 -3.732 -1.961  2.987 11.276
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.1588     6.8599   0.169    0.868
## brand_data$X1   0.9175     0.6894   1.331    0.206
## brand_data$X2  -0.5625     1.5416  -0.365    0.721
##
## Residual standard error: 6.167 on 13 degrees of freedom
## Multiple R-squared:  0.1278, Adjusted R-squared:  -0.006434
## F-statistic: 0.9521 on 2 and 13 DF,  p-value: 0.4113
```

```r
#to find SSE(R) and SSR(R)
anova(f)
```

```
## Analysis of Variance Table
##
## Response: ei2
##               Df Sum Sq Mean Sq F value Pr(>F)
## brand_data$X1  1  67.34  67.344  1.7710 0.2061
## brand_data$X2  1   5.06   5.063  0.1331 0.7211
## Residuals     13 494.35  38.027
```

```r
#to find SSE(F) and SSR(F)
anova(lm_brand)
```

```
## Analysis of Variance Table
##
## Response: Y
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## X1         1 1566.45 1566.45 215.947 1.778e-09 ***
## X2         1  306.25  306.25  42.219 2.011e-05 ***
## Residuals 13   94.30    7.25
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
SSR_R = 67.34+5.06
SSE_R = 494.35

SSR_F = 1566.45+306.25
SSE_F= 94.30

n = nrow(brand_data)

#chi-squared: [SSR(R)/2] / [SSE(F)/n]^2
chiTest = (SSR_R/2) / ((SSE_F/n))^2
print(chiTest)
```

```
## [1] 1.042138
```

```r
#p
chi = qchisq(1-0.05,1)
print(chi)
```

```
## [1] 3.841459
```

Decision Rule:

- If $chiTest \leq \chi^2(1-\alpha, 1)$, conclude H_0: constant error variance
- If $chiTest > \chi^2(1-\alpha, 1)$, conclude H_1: non-constant error variance

Result:

Since $1.042138 \leq 3.841459$ i.e. $chiTest \leq \chi^2(1-\alpha, 1)$, we conclude H_0. The error variance is constant.

Solution 3:

(a)

Hypothesis:

$H_0 : \beta_k = 0$

$H_a : \beta_k \neq 0$

```r
df_brand = lm_brand$df.residual
alpha = 0.01
anova(lm_brand)
```

```
## Analysis of Variance Table
##
## Response: Y
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## X1         1 1566.45 1566.45 215.947 1.778e-09 ***
## X2         1  306.25  306.25  42.219 2.011e-05 ***
## Residuals 13   94.30    7.25
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
SSR_X1 = anova(lm_brand)[1,2]
SSR_X2 = anova(lm_brand)[2,2]
SSE = anova(lm_brand)[3,2]
MSR_X1 = SSR_X1/1
MSR_X2 = SSR_X2/1
MSE = SSE/df_brand

F_star_X1 = MSR_X1/MSE
print(F_star_X1)
```

```
## [1] 215.9475
```

```r
F_star_X2 = MSR_X2/MSE
print(F_star_X2)
```

```
## [1] 42.21898
```

```r
FTest = qf(1-alpha, 1, df_brand)
print(FTest)
```

```
## [1] 9.073806
```

Decision Rule:

If $F^* \leq FTest$, conclude H_0

If $F^* > FTest$, conclude H_a

Result:

Since $F^*_{X1} and F^*_{X2}$ are both $> FTest$, we conclude H_a i.e. both the β_k are neq 0. Thus, there exists a linear relation.

(b)

```
alpha = 0.01
g = length(lm_brand$coefficients)
confint(lm_brand, level = 1-alpha/g)
```

```
##                0.167 %  99.833 %
## (Intercept) 26.912447 48.387553
## X1           3.345835  5.504165
## X2           1.961914  6.788086
```

Interpretation:

Family confidence coefficient means that the obtained confidence intervals, for several β_k, are simultaneously accurate with a confidence coefficient of $1 - \alpha = 99\%$.

(c)

```
Xh<-data.frame(X1=5, X2=4)
predict(lm_brand, Xh,se.fit=TRUE,interval="confidence",level=1-alpha)
```

```
## $fit
##      fit      lwr      upr
## 1 77.275 73.88111 80.66889
##
## $se.fit
## [1] 1.126687
##
## $df
## [1] 13
##
## $residual.scale
## [1] 2.693297
```

Interpretation:

This means that the $E[Y_h]$ for the observations in X_h are within the obtained interval with a confidence coeficient of $1 - \alpha = 99\%$, where all observations in X_h are seen by our model.

(d)

```
Xh<-data.frame(X1=5, X2=4)
predict(lm_brand, Xh,se.fit=TRUE,interval="prediction",level=0.99)
```

```
## $fit
##      fit      lwr      upr
## 1 77.275 68.48077 86.06923
##
## $se.fit
## [1] 1.126687
##
```

```
## $df
## [1] 13
##
## $residual.scale
## [1] 2.693297
```

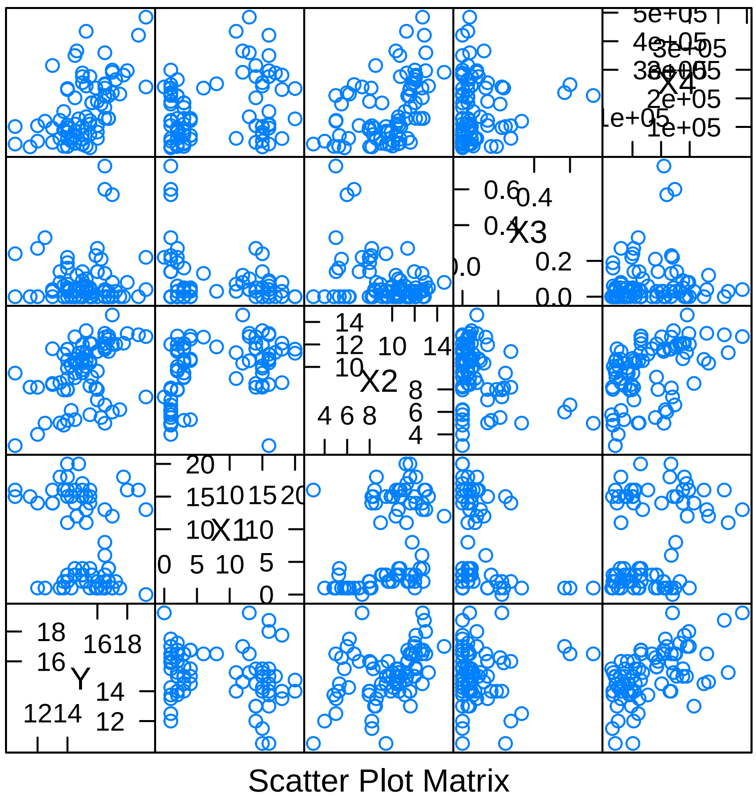
Interpretation:

This means that the $E[Y_h]$ for the observations in X_h are within the obtained interval with a confidence coeficient of $1 - \alpha = 99$ %, where all observations in X_h are new.

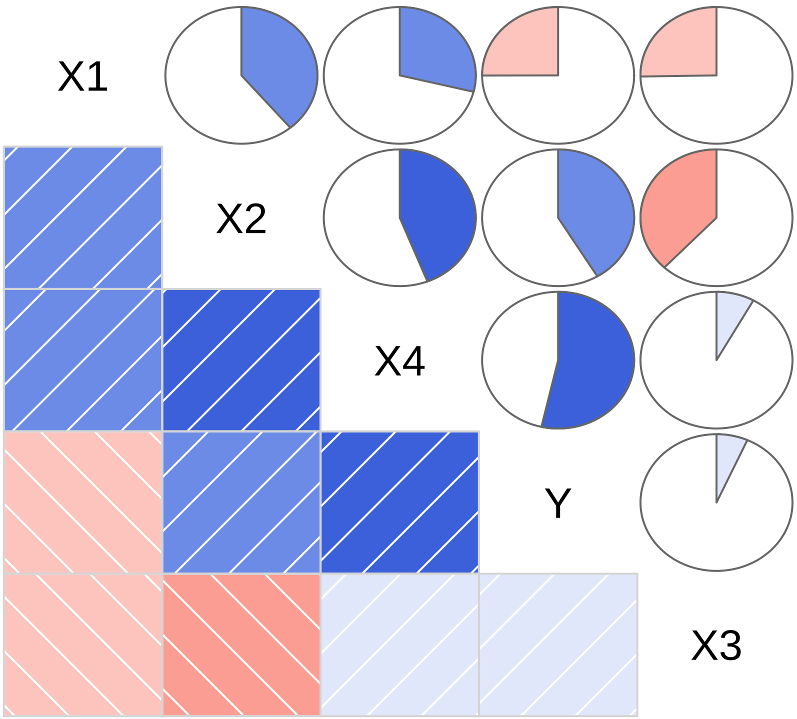
Solution 4:

(a)

```r
properties_data = read.csv("Commercial Properties.csv")
par(mfrow=c(1,1))
splom(properties_data, order=TRUE, oma=c(3,3,3,15))
```




```r
corrgram(properties_data, order=TRUE, lower.panel=panel.shade,
  upper.panel=panel.pie, text.panel=panel.txt, oma=c(3,3,3,15),
  main="Correlogram")
```




```r
cor(properties_data)
```

```
##              Y         X1         X2          X3         X4
## Y   1.00000000 -0.2502846  0.4137872  0.06652647 0.53526237
## X1 -0.25028456  1.0000000  0.3888264 -0.25266347 0.28858350
## X2  0.41378716  0.3888264  1.0000000 -0.37976174 0.44069713
## X3  0.06652647 -0.2526635 -0.3797617  1.00000000 0.08061073
## X4  0.53526237  0.2885835  0.4406971  0.08061073 1.00000000
```

Interpretation:

- There seems to be no 1:1 correlation in the data between any of the variables.
- The highest 1:1 correlation being between X4 and Y.
- We can see some clusters of data points in the plots for X1 and X3, showing that they are not equally spread.

(b)

```r
lm_prop = lm(Y~., data=properties_data)
summary(lm_prop)
```

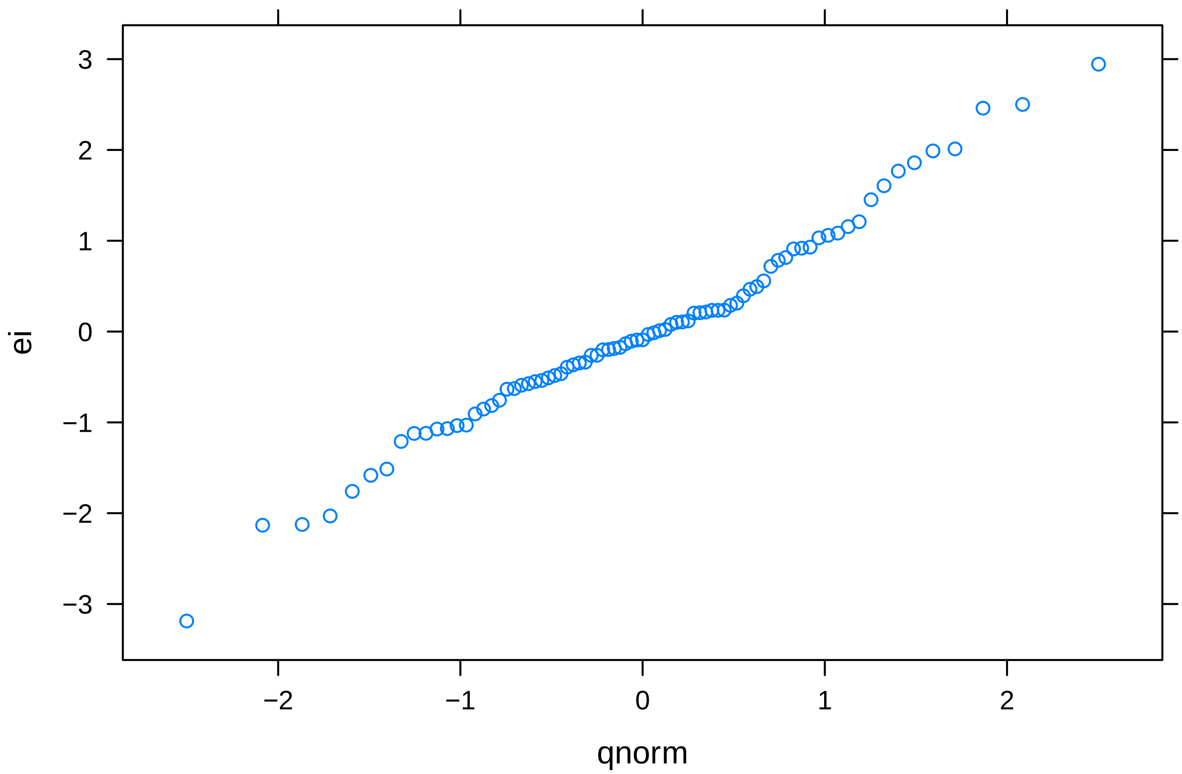
```
##
## Call:
## lm(formula = Y ~ ., data = properties_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1872 -0.5911 -0.0910  0.5579  2.9441
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.220e+01  5.780e-01  21.110  < 2e-16 ***
## X1          -1.420e-01  2.134e-02  -6.655 3.89e-09 ***
```

```
## X2           2.820e-01  6.317e-02   4.464 2.75e-05 ***
## X3           6.193e-01  1.087e+00   0.570     0.57
## X4           7.924e-06  1.385e-06   5.722 1.98e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.137 on 76 degrees of freedom
## Multiple R-squared:  0.5847, Adjusted R-squared:  0.5629
## F-statistic: 26.76 on 4 and 76 DF,  p-value: 7.272e-14
```

Regression Function: $Y = 12.2 - 0.142 * X1 + 0.282 * X2 + 0.6193 * X3 + 7.924e - 06 * X4$

(c)

```r
ei = lm_prop$residuals
qqmath(ei)
```



Interpretation: The QQ plot seems to be almost linear, which means that the error is in agreement with the normality. Distribution is fairly linear.

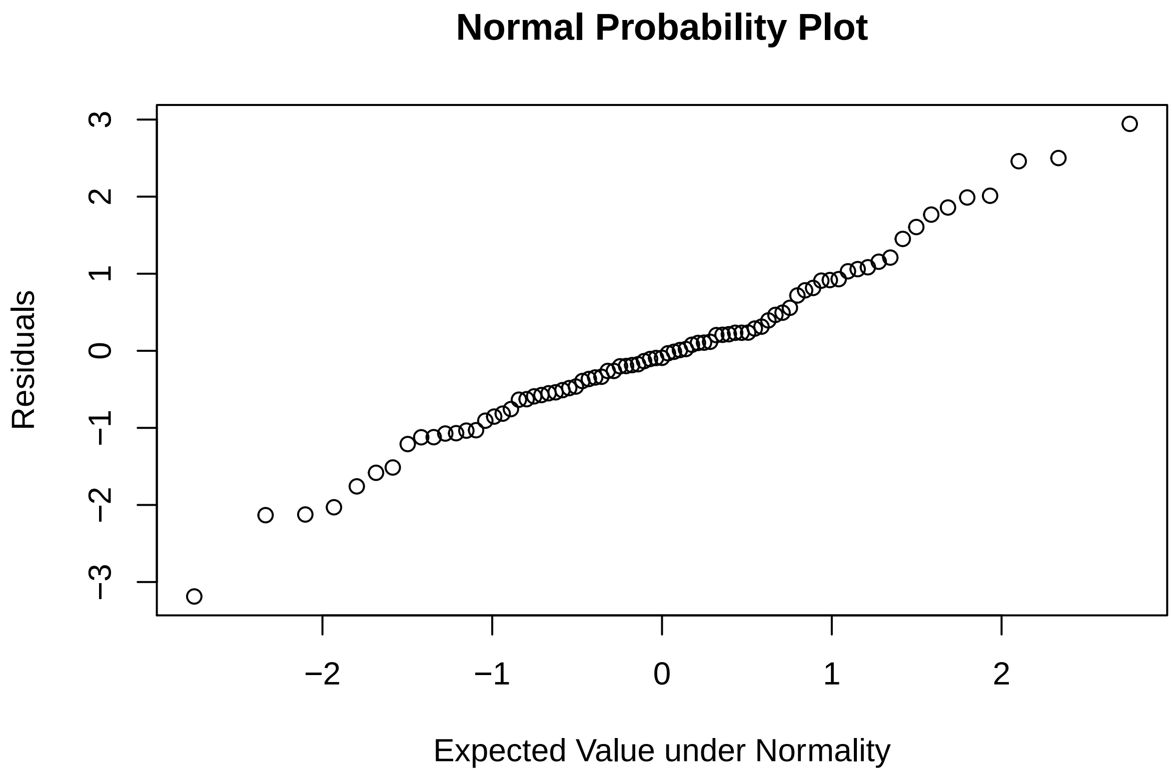
(d)

```r
df = properties_data
rse = 1.137

ri = rank(ei)
n = nrow(df)
zr = (ri-0.375)/(n+0.25)

#residual standard error from summary(lm) above
zr1 = rse*qnorm(zr)
```

```r
print(cor.test(zr1, ei))
```

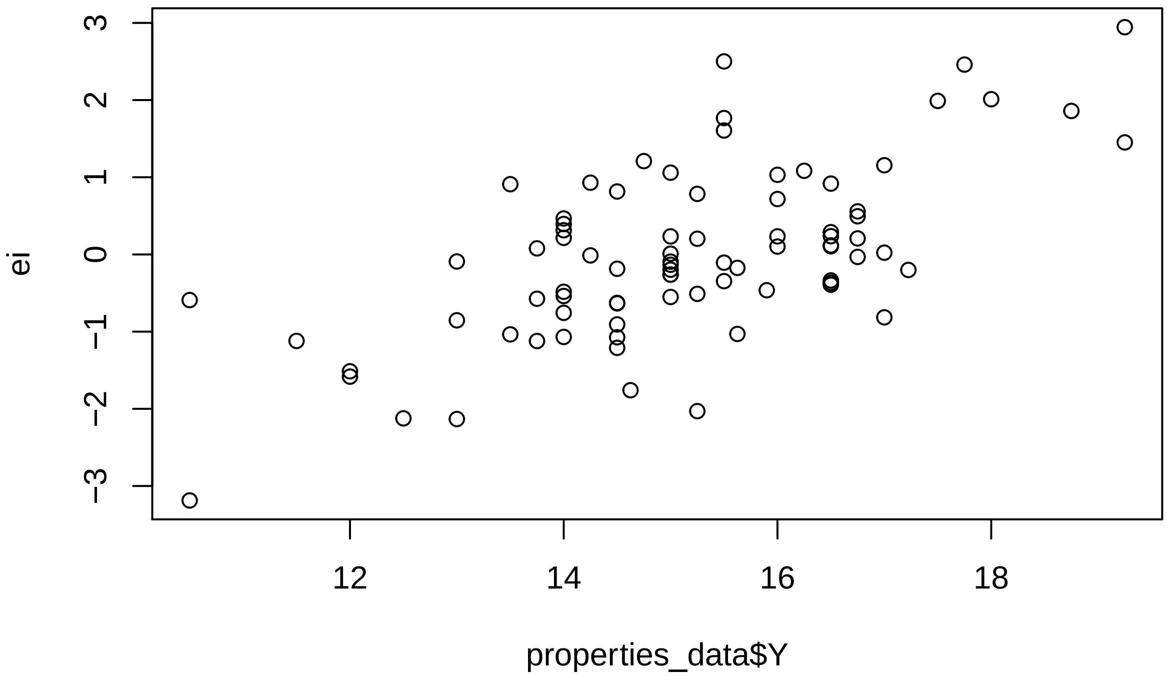
```
##
##  Pearson's product-moment correlation
##
## data:  zr1 and ei
## t = 64.593, df = 79, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9854874 0.9940009
## sample estimates:
##      cor
## 0.990665
```

```r
plot(zr1, ei, xlab="Expected Value under Normality",ylab="Residuals")
title(main="Normal Probability Plot")
```

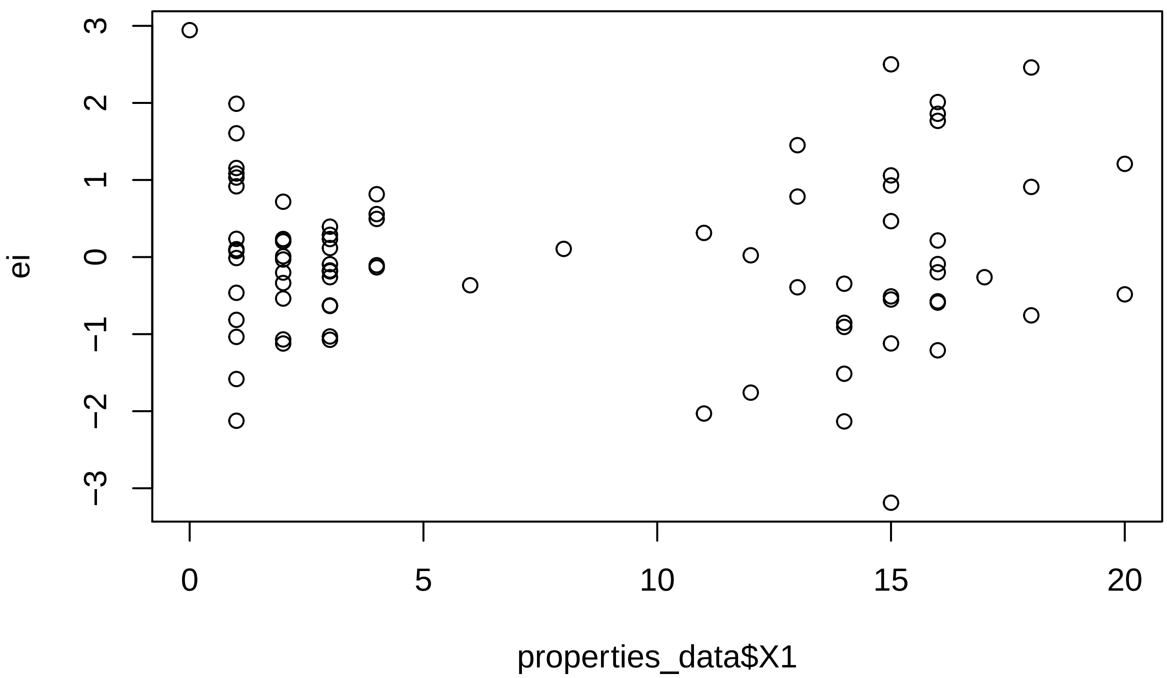


Interpretation: The normal probability plot also seems to be almost linear, which means that the error is in agreement with the normality.

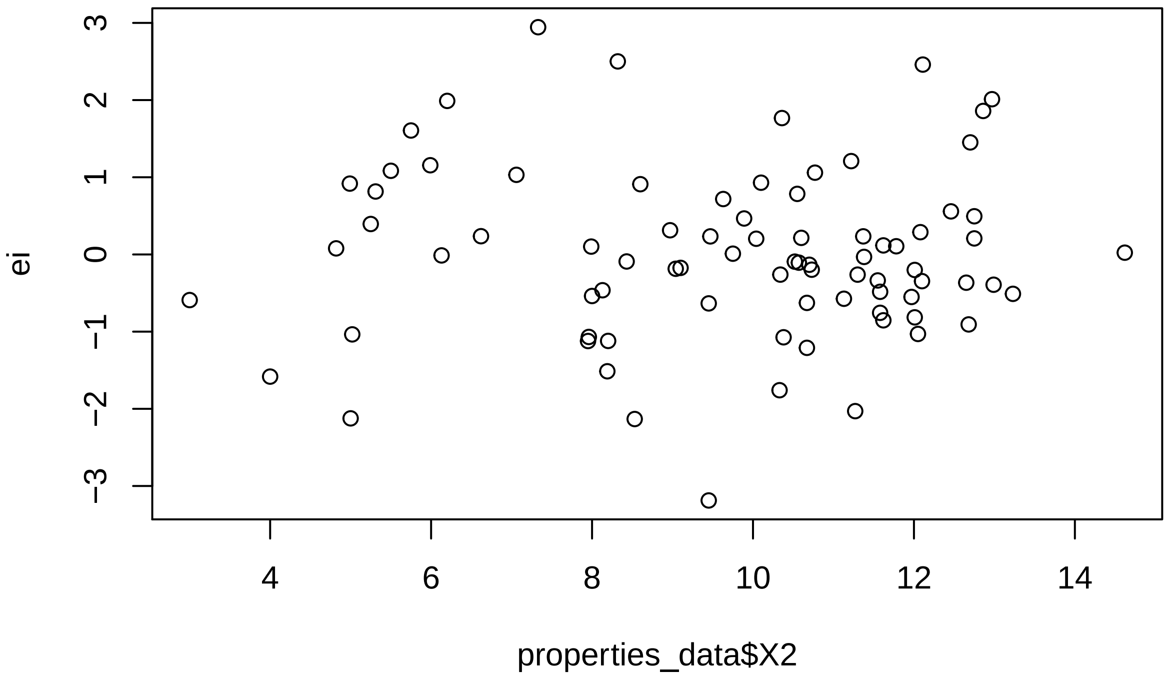
```r
plot(properties_data$Y, ei)
```



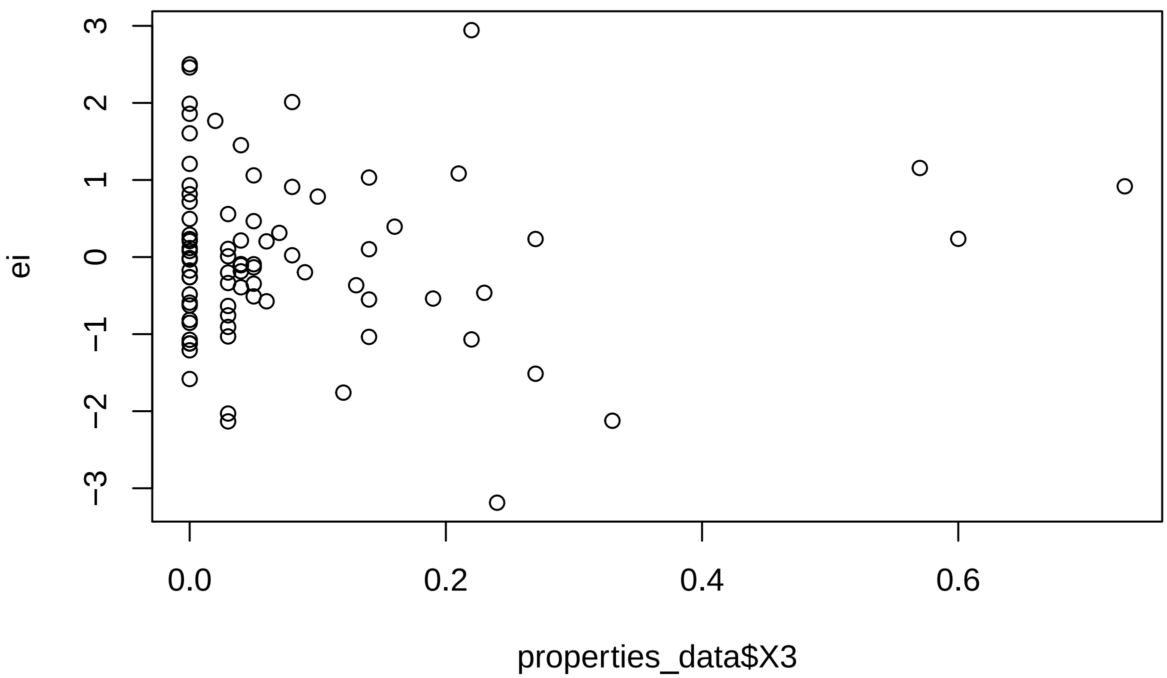

```
plot(properties_data$X1, ei)
```



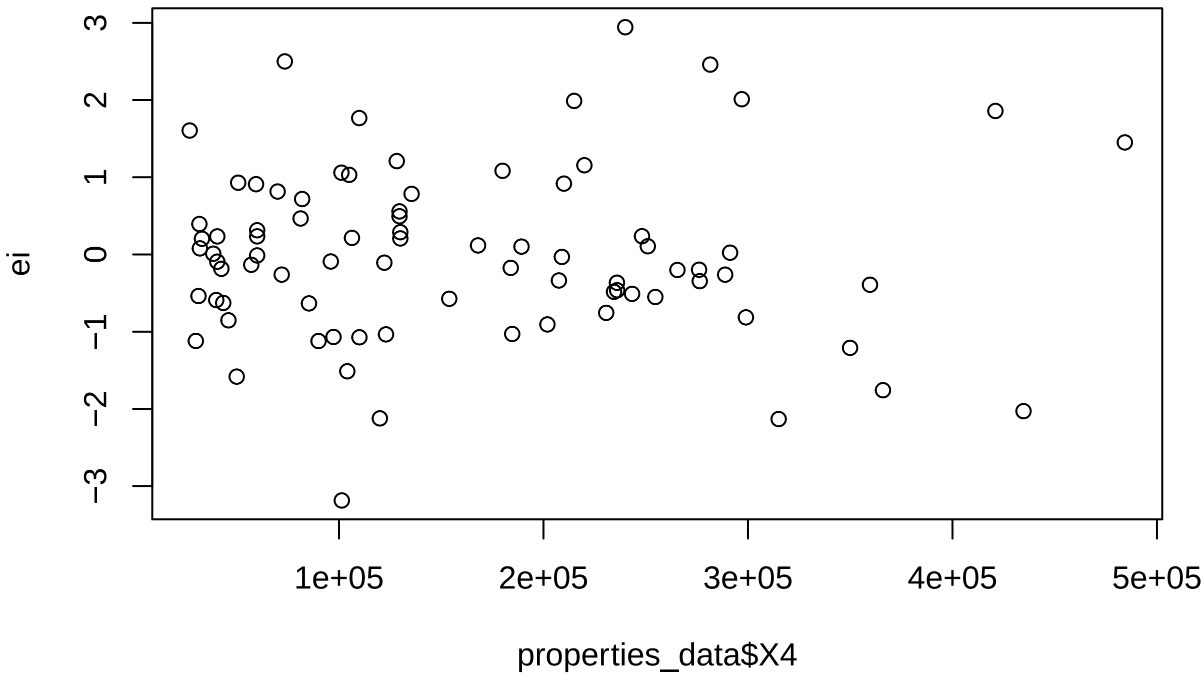

```
plot(properties_data$X2, ei)
```



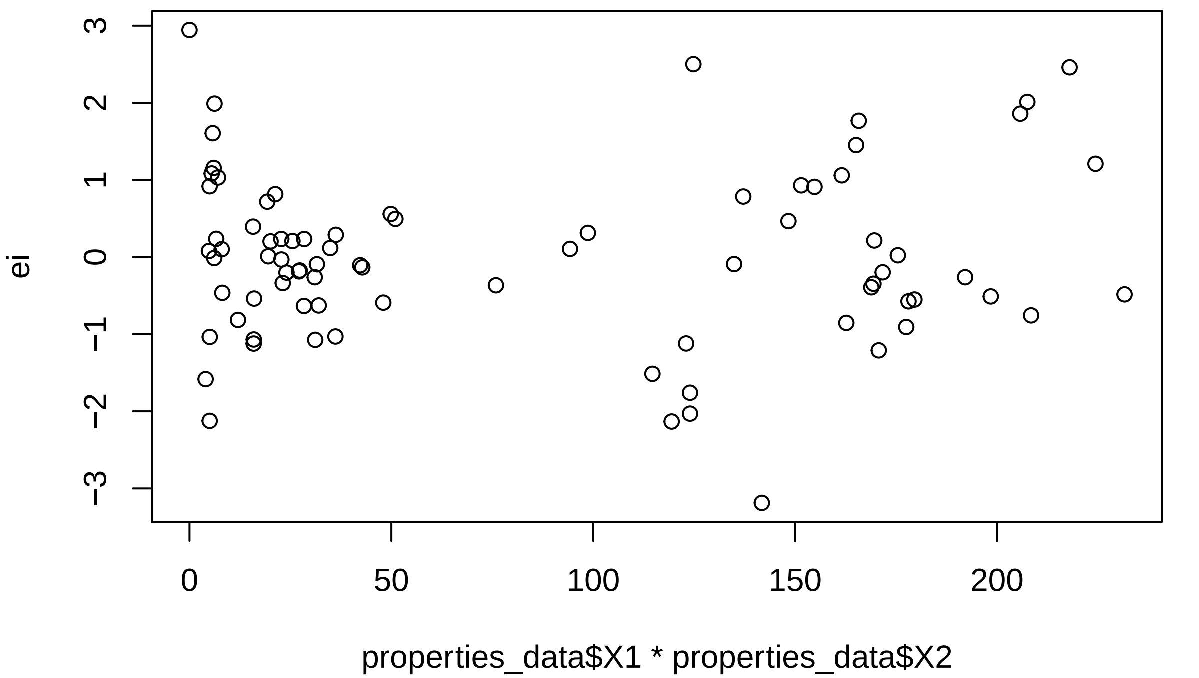

```r
plot(properties_data$X3, ei)
```



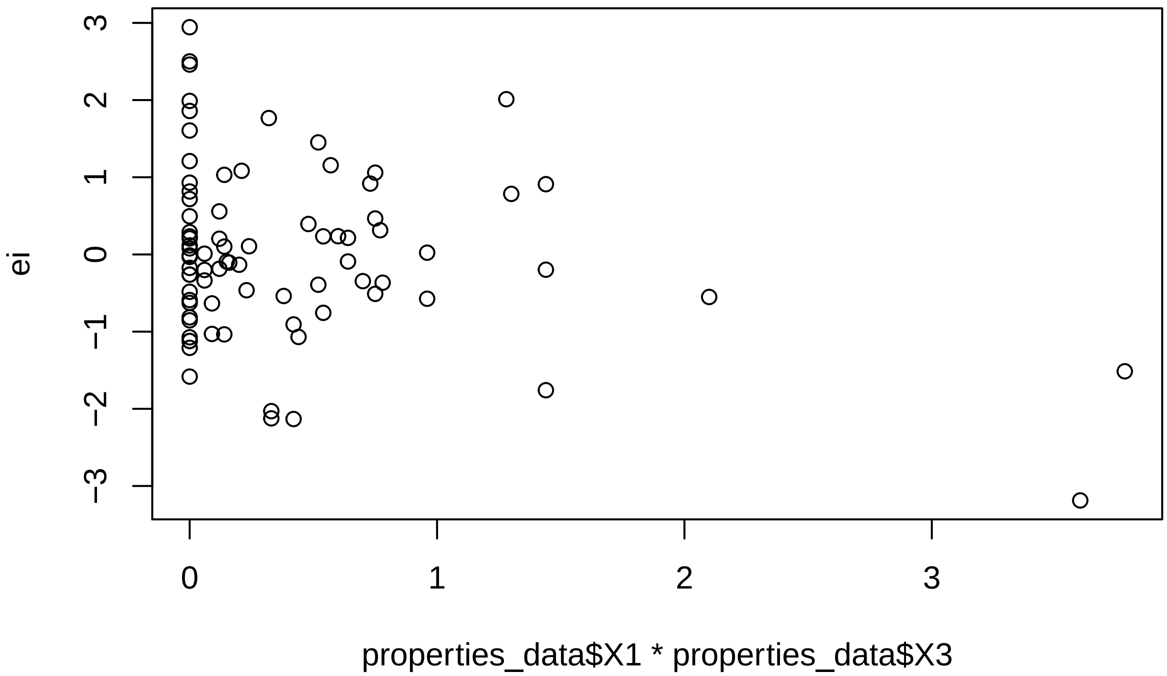

```r
plot(properties_data$X4, ei)
```



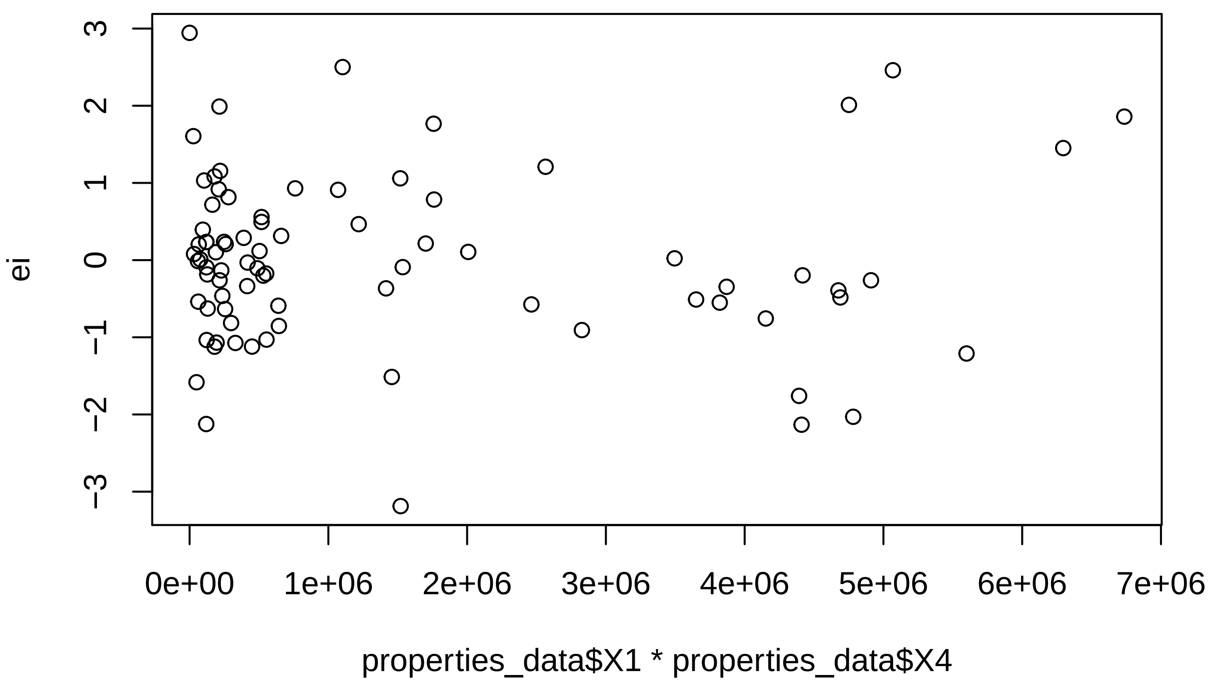

```
plot(properties_data$X1*properties_data$X2, ei)
```



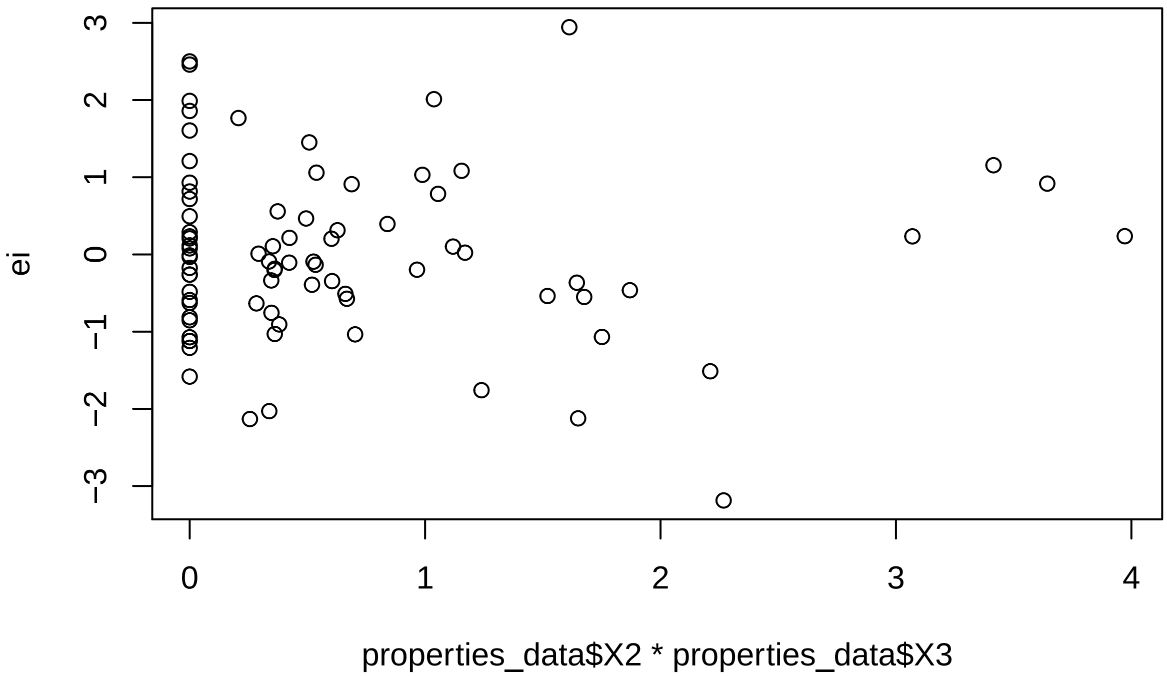

```
plot(properties_data$X1*properties_data$X3, ei)
```



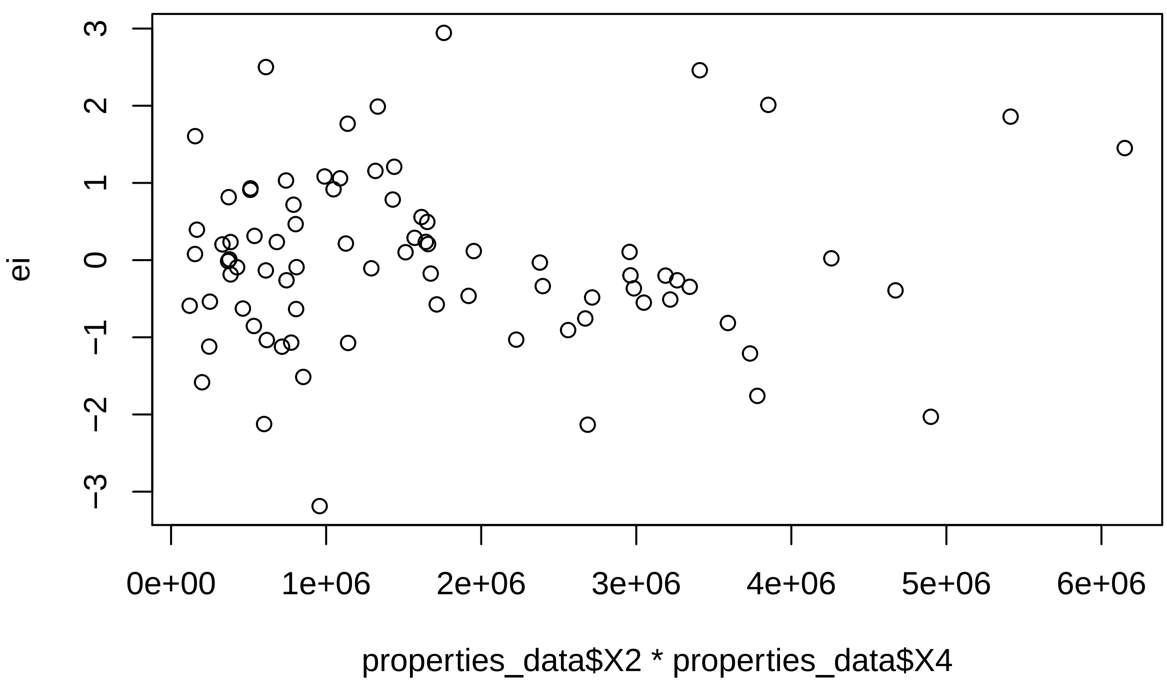

```
plot(properties_data$X1*properties_data$X4, ei)
```



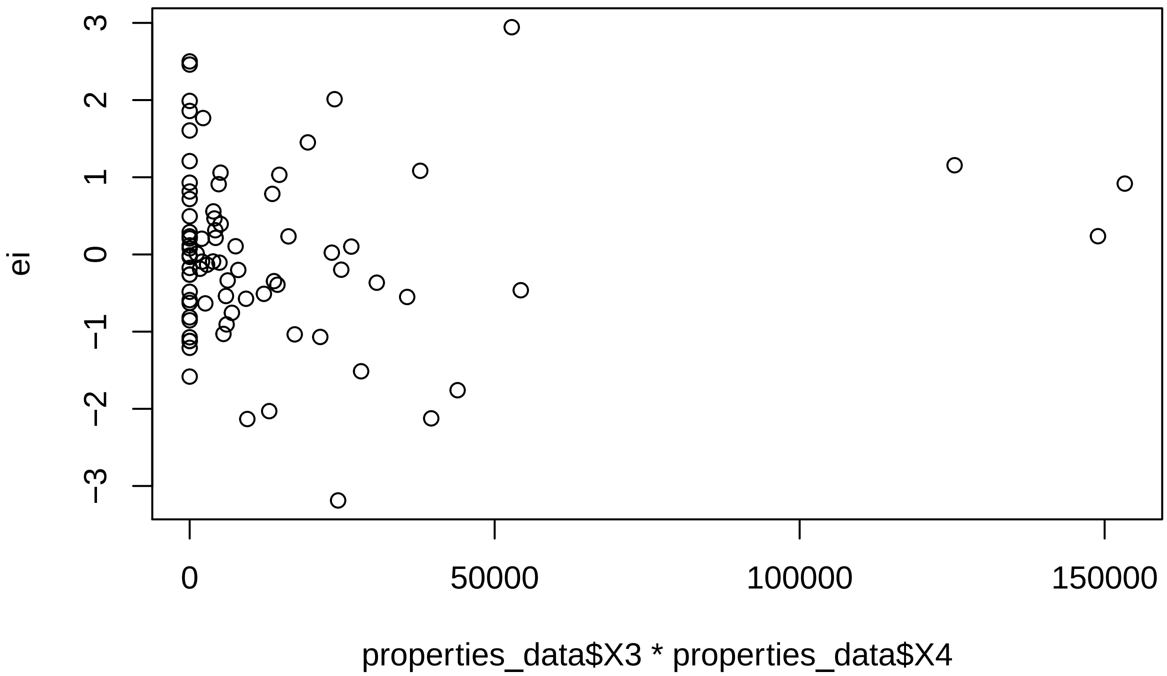

```
plot(properties_data$X2*properties_data$X3, ei)
```




```
plot(properties_data$X2*properties_data$X4, ei)
```




```
plot(properties_data$X3*properties_data$X4, ei)
```



Interpretation: - There is a linear pattern between residuals and Y, which could suggest that our linear model is not a good fit through out the data. - We can see some clusters of data points in the plots for X1 and X3 and their respective interactions. - For X2 and X4 and their interactions, the error variance is constant.

(e)

```
ei = lm_prop$residuals
fitted_df = data.frame(fitted_values = lm_prop$fitted.values)
df = data.frame(cbind(properties_data, fitted_df, ei))
df = df[order(df$fitted_values),]
```

Null Hypothesis: H_0: Error variance is constant Alternate Hypothesis: H_1: Error variance is not constant

```
df1 = df[1:40,]
df2 = df[41:nrow(df),]

med1 = median(df1$ei)
med2 = median(df2$ei)

#n1
n1 = nrow(df1)
print(n1)
```

```
## [1] 40
```

```
#n2
n2 = nrow(df2)
print(n2)
```

```
## [1] 41
```

```
d1 = abs(df1$ei-med1)
d2 = abs(df2$ei-med2)

#calculate means for our answer
```

```r
mean_d1 = mean(d1)
print(mean_d1)
```

```
## [1] 0.8695662
```

```r
mean_d2 = mean(d2)
print(mean_d2)
```

```
## [1] 0.7793035
```

```r
s2 = (var(d1)*(n1-1)+var(d2)*(n2-1))/(n1+n2-2)
print(s2)
```

```
## [1] 0.5411876
```

```r
#calculate s
s = sqrt(s2)
print(s)
```

```
## [1] 0.7356545
```

```r
#testStastic = (mean.d1 - mean.d2) / (s * sqrt((1/n1)+1/n2)
testStastic = (mean_d1-mean_d2)/(s*sqrt((1/n1)+(1/n2)))
print(testStastic)
```

```
## [1] 0.5520951
```

```r
t = qt(1-0.05, 118)
print(t)
```

```
## [1] 1.65787
```

Decision Rule:

- If $|testStatistic| \leq t(1-\alpha/2, n-2)$, conclude H_0: constant error variance
- If $|testStatistic| > t(1-\alpha/2, n-2)$, conclude H_1: non-constant error variance

Result:

Since $|0.5520951| \leq 1.65787$ i.e. $|testStatistic| \leq t(1-\alpha/2, n-2)$, we conclude H_0. The error variance is constant and thus does not vary with X.

Solution 5:

(a)

```r
X1 = c(4.0,6.0,12.0)
X2 = c(10.0,11.5,12.5)
X3 = c(0.1,0,0.32)
X4 = c(80000,120000,340000)
Xh = data.frame(X1, X2, X3, X4)

alpha = 0.05
g = nrow(Xh)
```

```r
predict(lm_prop, Xh, se.fit=TRUE, interval="prediction", level=1-alpha)
```

```
## $fit
##        fit      lwr      upr
## 1 15.14850 12.85249 17.44450
## 2 15.54249 13.24504 17.83994
```

```
## 3 16.91384 14.53469 19.29299
##
## $se.fit
##         1         2         3
## 0.1908982 0.1952287 0.3666570
##
## $df
## [1] 76
##
## $residual.scale
## [1] 1.136885
```

Interpretation:

We see that the intervals are not too wide for the individual prediction intervals. We get an R^2 of 0.5847 which gives a somewhat of a good prediction.

```
CI = predict(lm_prop, Xh, se.fit=TRUE, interval="confidence", level=1-alpha/g)
CI
```

```
## $fit
##        fit      lwr      upr
## 1 15.14850 14.68115 15.61584
## 2 15.54249 15.06455 16.02043
## 3 16.91384 16.01622 17.81146
##
## $se.fit
##         1         2         3
## 0.1908982 0.1952287 0.3666570
##
## $df
## [1] 76
##
## $residual.scale
## [1] 1.136885
```

Bonferroni and Working-Hotelling Method:

```
p = length(lm_prop$coefficients)
n = nrow(properties_data)
B = qt(1-alpha/(2*g), n-p)
W = sqrt(2*qf(1-alpha, g, n-p))
s = CI$se.fit
Yh = CI$fit[,1]

est_resp_CI = t(
rbind(
"Xh" = t(Xh),
"fit" = array(Yh),
"lower.B" = array(Yh-B*s),
"upper.B" = array(Yh+B*s),
"lower.W" = array(Yh-W*s),
"upper.W" = array(Yh+W*s)
)
)

est_resp_CI
```

```
##       X1   X2   X3     X4      fit  lower.B  upper.B  lower.W  upper.W
## [1,]   4 10.0 0.10  80000 15.14850 14.68115 15.61584 14.70284 15.59415
## [2,]   6 11.5 0.00 120000 15.54249 15.06455 16.02043 15.08673 15.99825
## [3,]  12 12.5 0.32 340000 16.91384 16.01622 17.81146 16.05788 17.76980
```