MH3510
Regression Analysis

NANYANG
TECHNOLOGICAL
UNIVERSITY
SINGAPORE

Group Project Report

Name	Matric. No
Gordon Tan Au Aun	U2240682F
Seah Kah Yen	U2240401D
Tan Dao Ze, Enric	U2240521H
Loh Jyn Ern Daryl	U2240990B
Zeng Yuzhi	U2240256K
Tan jing sheng	U2240634J
Choo Yi Ken	U2240710B
Tong Hao Kit	U2240130E
Javier Teh Ding Kiat	U2240471A

Table of Content

Modelling Linear Regression Model with R

1. Load Required Libraries

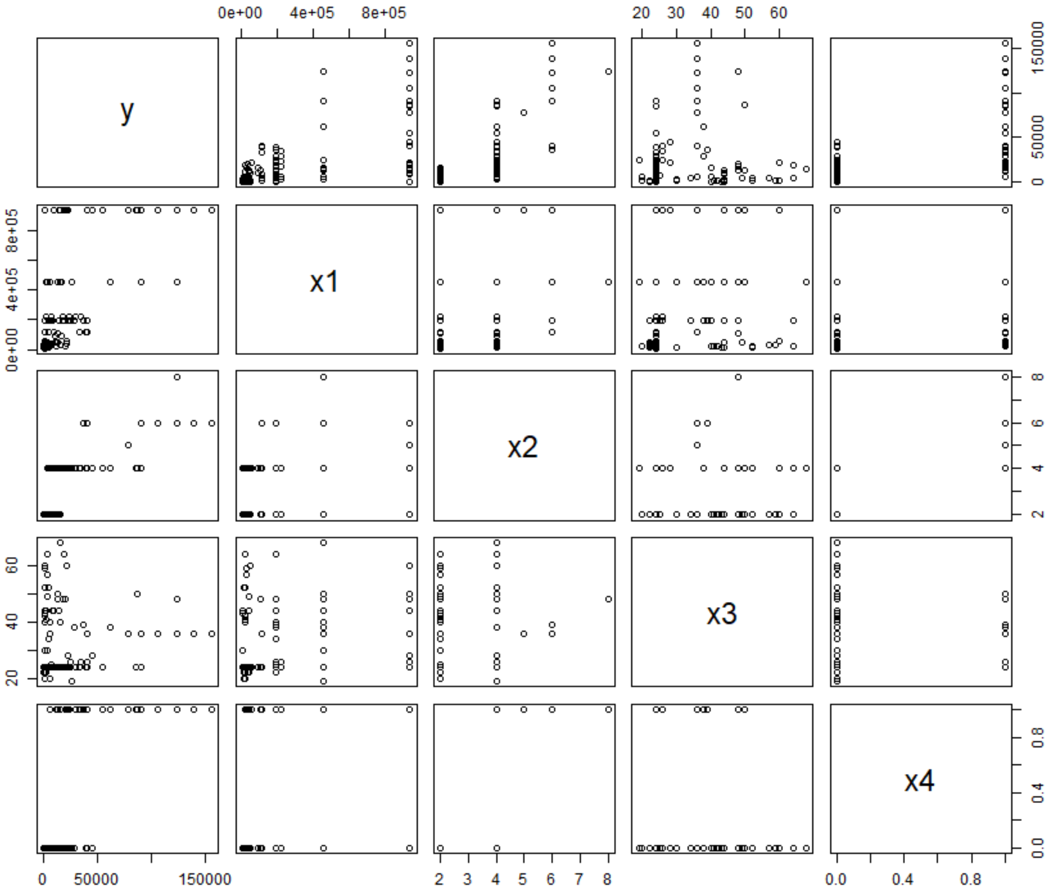
```
library(dplyr)
library(caret)
```

2. Data Preparation

```
# Load raw data
data_raw <- read.table('aadt.txt', header = FALSE)
df <- data.frame(y = data_raw$V1, x1 = data_raw$V2, x2 = data_raw$V3, x3 =
data_raw$V4, x4 = data_raw$V5)

# Convert x4 values of 2 to 0
df$x4[df$x4 == 2] <- 0

# Scale the data (optional)
df_scaled <- df
```

3. Exploratory Data Analysis

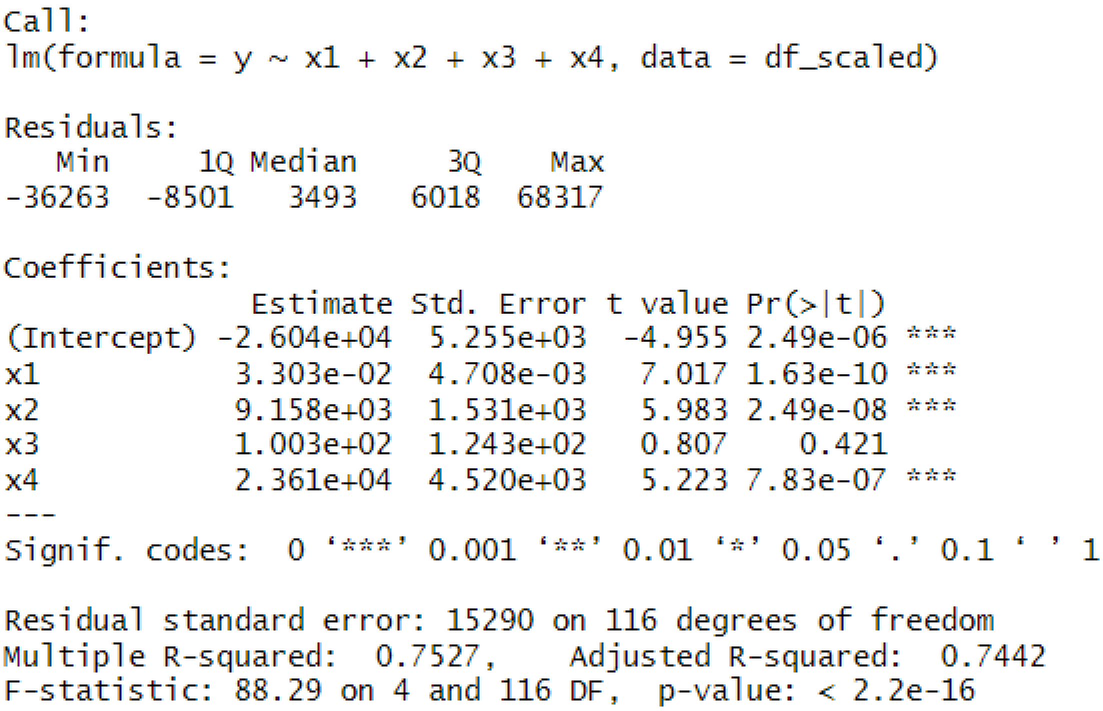
```
# Graphical display of the observed data
plot(df)
```



- **y vs. x1**: There seems to be some clustering or grouping in the data points. The relationship doesn't appear clearly linear.
- **y vs. x2**: The plot between y and x2 does not show a clear trend. The clustering observed here might imply the presence of some categorical elements or distinct groups within the dataset.
- **y vs. x3**: There appears to be minimal linear correlation between y and x3. The scatter is widespread, indicating that x3 might not be a significant predictor for y.
- **y vs. x4**: The points between y and x4 indicate a distinct grouping with many points overlapping at zero. This might indicate that x4 is binary or categorical.
- **x1 vs. x2**: There is no clear linear relationship between x1 and x2.
- **x2 vs. x3**: The scatter plot is widely spread, indicating that x2 and x3 are likely not correlated with each other.
- **x3 vs. x4**: There is significant correlation between x3 and x4.

4. Multiple Linear Regression Model

```
# Fit a multiple linear regression model
mlr <- lm(y ~ x1 + x2 + x3 + x4, data = df)
summary(mlr)
```

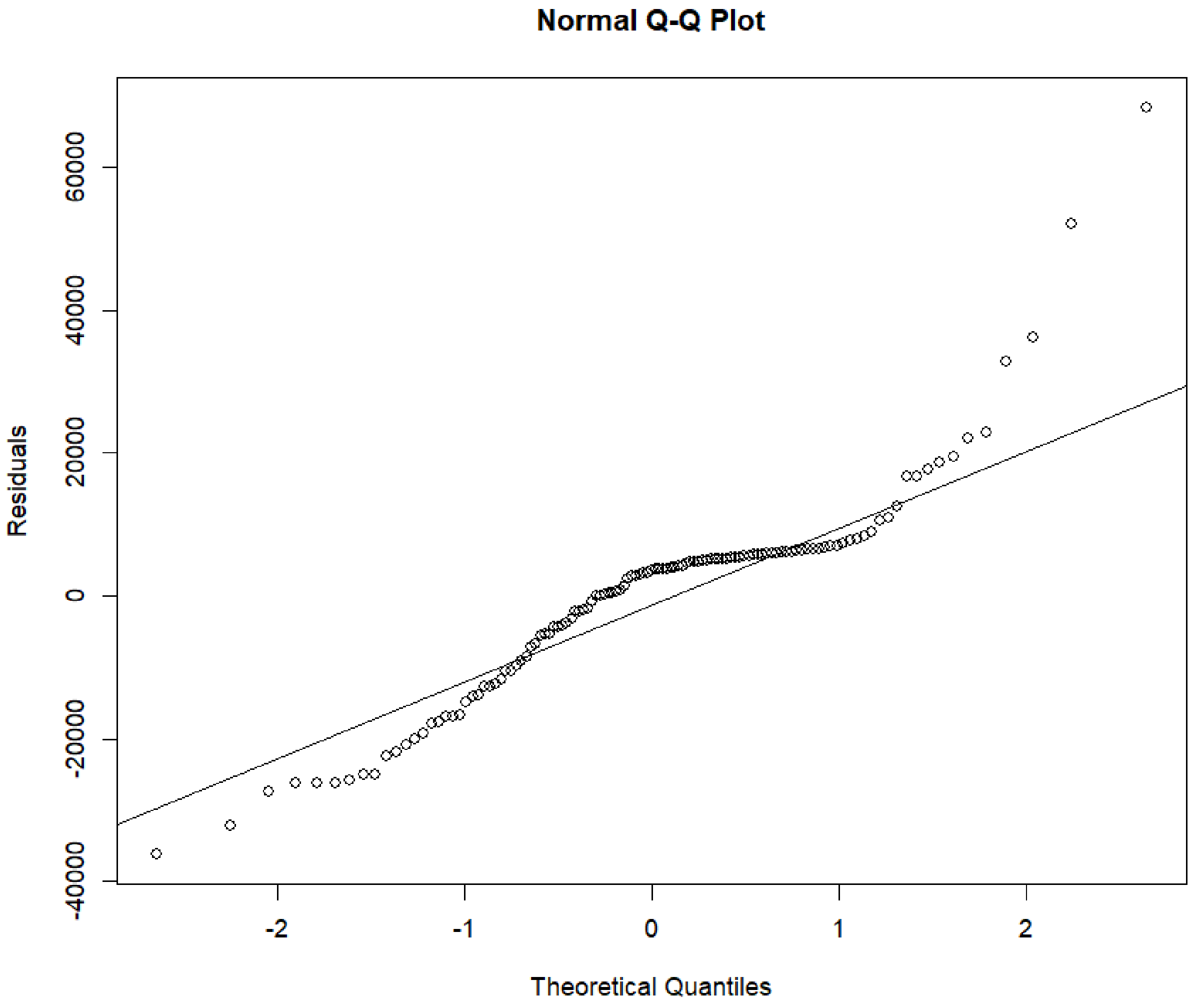


```
Call:
lm(formula = y ~ x1 + x2 + x3 + x4, data = df_scaled)

Residuals:
   Min     1Q Median     3Q    Max 
-36263  -8501   3493   6018  68317 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -2.604e+04  5.255e+03  -4.955 2.49e-06 ***
x1           3.303e-02  4.708e-03   7.017 1.63e-10 ***
x2           9.158e+03  1.531e+03   5.983 2.49e-08 ***
x3           1.003e+02  1.243e+02   0.807    0.421    
x4           2.361e+04  4.520e+03   5.223 7.83e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15290 on 116 degrees of freedom
Multiple R-squared:  0.7527,	Adjusted R-squared:  0.7442 
F-statistic: 88.29 on 4 and 116 DF,  p-value: < 2.2e-16
```

- **Coefficients:**
    - Coefficient of **x3** has a p-value of 0.421 which is greater than the significance level of 0.05, thus H_0: $\beta_3 = 0$ is not rejected. Hence, **x3** is the least significant variable to the model in terms of t-test
    - Coefficients of **x1**, **x2** and **x4** have p-values which are smaller than 0.05, thus H_0: $\beta_i = 0$ for i = 1, 2, 4 are rejected. Hence, **x1**, **x2** and **x4** are highly significant variables.
- **R^2 Statistic:**
    - Multiple R-squared = 0.7527 indicates that approximately 75.27% of the variability in the response variable **y** is explained by the model.
    - Adjusted R-squared = 0.7442 is slightly lower than the R-squared value, indicating that the model fits the data well, considering the number of predictors.
- **F-statistic**:
    - 88.29 on 4 and 116 degrees of freedom, with a p-value < 2.2e-16. This suggests that the model is significant overall, and at least one of the predictors is related to **y**

5. Normality Check of Residuals

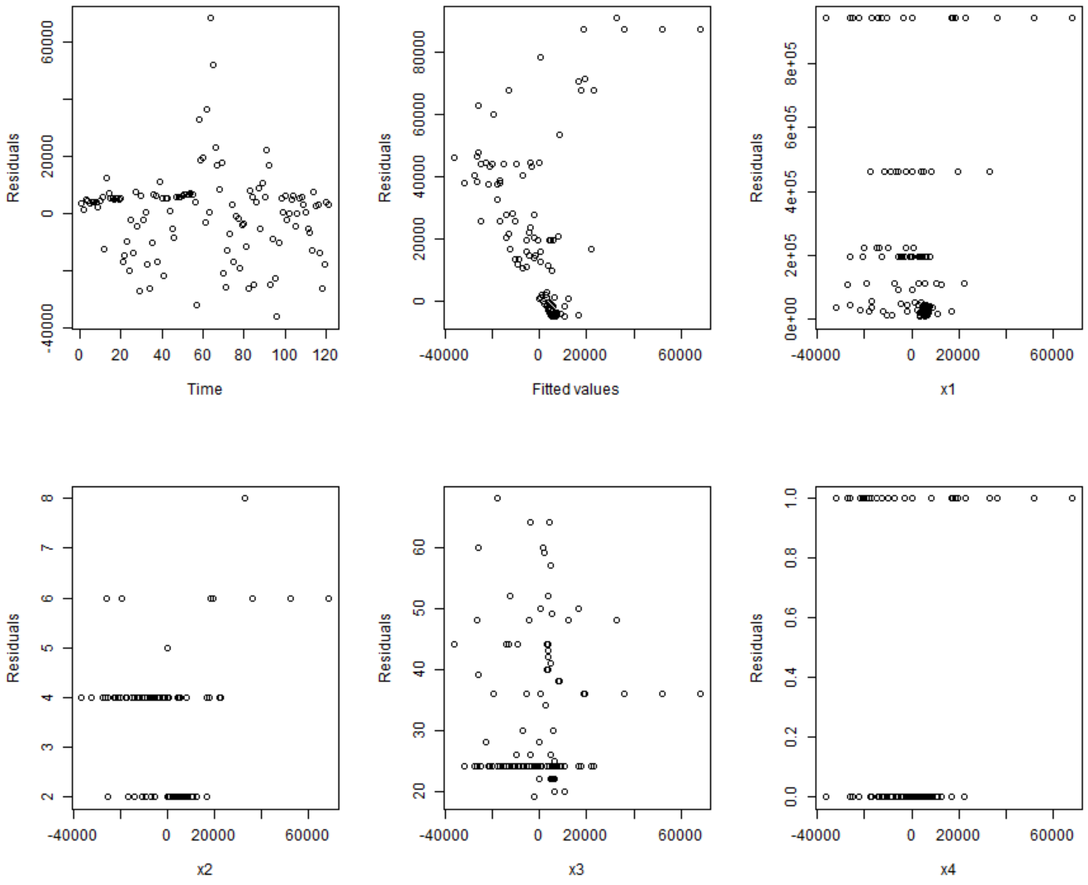
```
# Normality checking
qqnorm(residuals(mlr), ylab = 'Residuals')
qqline(residuals(mlr))
```



- Points generally follow the normal line on the QQ plot, except for some deviations in the upper tail.
- This suggests that the data mostly follows a normal distribution but has a few outliers in the upper tail.

6. Residual Analysis

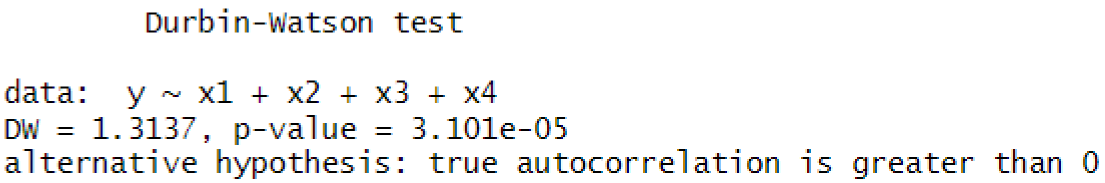
```
# Draw some plots of residuals
par(mfrow = c(2, 3))
plot(residuals(mlr), ylab = 'Residuals', xlab = 'Time')
plot(residuals(mlr), fitted(mlr), ylab = 'Residuals', xlab = 'Fitted values')
plot(residuals(mlr), df_scaled$x1, ylab = 'Residuals', xlab = 'x1')
plot(residuals(mlr), df_scaled$x2, ylab = 'Residuals', xlab = 'x2')
plot(residuals(mlr), df_scaled$x3, ylab = 'Residuals', xlab = 'x3')
plot(residuals(mlr), df_scaled$x4, ylab = 'Residuals', xlab = 'x4')
par(mfrow = c(1, 1))
```



- **Residuals vs. Time:** The points are somehow distributed around the zero line. There doesn't appear to be a strong trend over time, suggesting no clear temporal pattern.
- **Residuals vs. Fitted Values:** The plot shows a funnel-shaped pattern, which means that the variability of residuals is not constant across different levels of the fitted values.
- **Residuals vs. Predictors (x1, x2, x3, x4):** There are noticeable patterns and clusters, particularly for **x2** and **x3**, which may indicate non-linearity.The residuals for **x4** are clustered, suggesting **x4** might be categorical

7. Durbin-Watson Test

```
library(lmtest)
dwtest(y ~ x1 + x2 + x3 + x4, data = df)
```

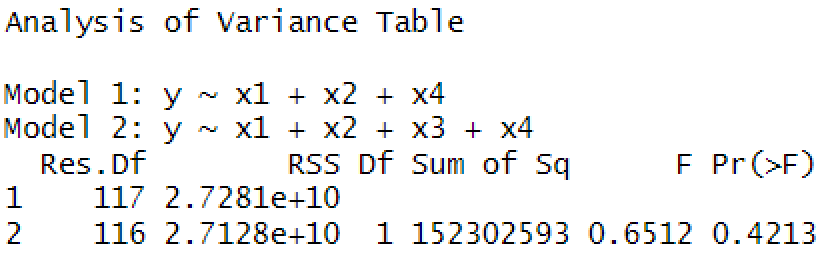


```
        Durbin-Watson test

data:  y ~ x1 + x2 + x3 + x4
DW = 1.3137, p-value = 3.101e-05
alternative hypothesis: true autocorrelation is greater than 0
```

- DW = 1.3137 suggests positive autocorrelation of residuals, meaning that consecutive residuals are correlated in a positive manner
- The p-value is **3.101e-05**, which is very small. The null hypothesis for the Durbin-Watson test is that there is no autocorrelation in the residuals. Since the p-value is much smaller than typical significance levels, we reject the null hypothesis. This indicates that autocorrelation is present in the residuals.

8. Model Comparison with F-tests

I.
```
# Model without x3
mlr1 <- lm(y ~ x1 + x2 + x4, data = df)
anova(mlr1, mlr)
```

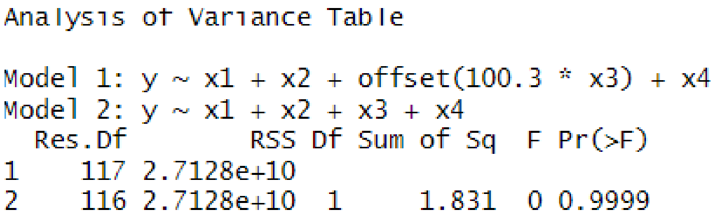


```
Analysis of Variance Table

Model 1: y ~ x1 + x2 + x4
Model 2: y ~ x1 + x2 + x3 + x4
  Res.Df        RSS Df Sum of Sq      F Pr(>F)
1    117 2.7281e+10
2    116 2.7128e+10  1 152302593 0.6512 0.4213
```

- The p-value is 0.4213. Since the p-value is greater than common significance levels, we fail to reject the null hypothesis, meaning that adding **x3** does not significantly improve the model.

II. # Model where $\hat{\beta}_3$ is constant using offset

```
mlr3 <- lm(y ~ x1 + x2 + offset(100.3 * x3) + x4, data = df)
summary(mlr3)
anova(mlr3, mlr)
```

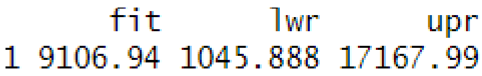


```
Analysis of Variance Table

Model 1: y ~ x1 + x2 + offset(100.3 * x3) + x4
Model 2: y ~ x1 + x2 + x3 + x4
  Res.Df        RSS Df Sum of Sq  F Pr(>F)
1    117 2.7128e+10
2    116 2.7128e+10  1     1.831  0 0.9999
```

- From the results of the MLR, we get $\hat{\beta}_3 = 100.3$
- To determine if setting a constant coefficient for **x3** improves model fitting, we specified 100.3 as an offset for β_3
- The p-value obtained is 0.9999. Since p-value is extremely high, we fail to reject the null hypothesis. It would be appropriate to treat **x3** as an offset or consider removing it altogether, as it does not add value in terms of improving model fit.

9. Prediction

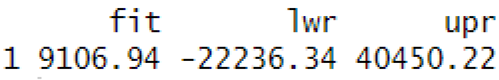
```
con <- data.frame(x1 = 50000, x2 = 3, x3 = 60, x4 = 0)

# Confidence interval prediction
predict(mlr, con, interval = 'confidence', level = 0.95)
```



```
      fit      lwr      upr
1 9106.94 1045.888 17167.99
```

- fit: The predicted value for the given input is 9106.94.
- lwr (Lower Bound): 1045.888 is the lower bound of the 95% confidence interval.
- upr (Upper Bound): 17167.99 is the upper bound of the 95% confidence interval.
- Interpretation: We are 95% confident that the mean response will fall between 1045.888 and 17167.99.

```
# Prediction interval prediction
predict(mlr, con, interval = 'prediction', level = 0.95)
```

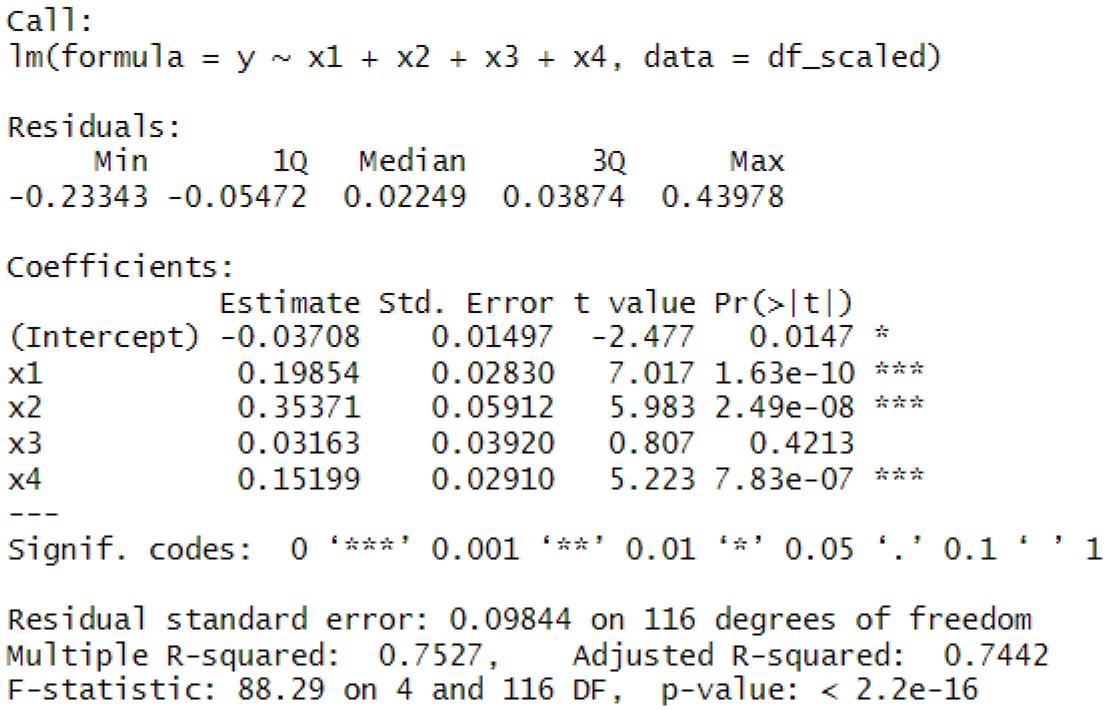


```
      fit       lwr      upr
1 9106.94 -22236.34 40450.22
```

- fit: The predicted value for the given input is 9106.94.
- lwr (Lower Bound): -22236.34 is the lower bound of the 95% prediction interval.
- upr (Upper Bound): 40450.22 is the upper bound of the 95% prediction interval.
- Interpretation: The prediction interval is very wide, ranging from -22236.34 to 40450.22, which suggests a lot of uncertainty in predicting individual responses.

10. Consideration of normalised data

```
# Normalised data
process <- preProcess(df, method=c("range"))
df_scaled <- predict(process, df)

mlr_scaled <- lm(y ~ x1+x2+x3+x4, data=df_scaled)
summary(mlr_scaled)
```



```
Call:
lm(formula = y ~ x1 + x2 + x3 + x4, data = df_scaled)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.23343 -0.05472  0.02249  0.03874  0.43978 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.03708    0.01497  -2.477   0.0147 *  
x1           0.19854    0.02830   7.017 1.63e-10 ***
x2           0.35371    0.05912   5.983 2.49e-08 ***
x3           0.03163    0.03920   0.807   0.4213    
x4           0.15199    0.02910   5.223 7.83e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09844 on 116 degrees of freedom
Multiple R-squared:  0.7527,	Adjusted R-squared:  0.7442 
F-statistic: 88.29 on 4 and 116 DF,  p-value: < 2.2e-16
```

- The result shows that the normalised data does not improve the model fitting. Therefore, normalised data is not necessary.

11. Annex A (source code)

```
library(dplyr)
library(caret)

# Graphical display of the observed data.
data_raw <- read.table('aadt.txt',header=FALSE)
df <-
data.frame(y=data_raw$V1,x1=data_raw$V2,x2=data_raw$V3,x3=data_raw$V4,x4=data_raw$V5)
df$x4[df$x4 == 2]<-0
process <- preProcess(df, method=c("range"))
df_scaled <- predict(process, df)
plot(df)
plot(df_scaled)

# Fit a multiple linear regression model for both scaled and non-scaled data.
mlr <- lm(y ~ x1+x2+x3+x4, data=df)
summary(mlr)
mlr_scaled <- lm(y ~ x1+x2+x3+x4, data=df_scaled)
summary(mlr_scaled)

# Normality checking.
qqnorm(residuals(mlr),ylab='Residuals')
qqline(residuals(mlr))
qqnorm(residuals(mlr_scaled),ylab='Residuals')
qqline(residuals(mlr_scaled))

# Draw some plots of residuals.
par(mfrow=c(2,3))
plot(residuals(mlr),ylab='Residuals',xlab='Time')
plot(residuals(mlr),fitted(mlr),ylab='Residuals',xlab='Fitted values')
plot(residuals(mlr),df$x1,ylab='Residuals',xlab='x1')
plot(residuals(mlr),df$x2,ylab='Residuals',xlab='x2')
plot(residuals(mlr),df$x3,ylab='Residuals',xlab='x3')
plot(residuals(mlr),df$x4,ylab='Residuals',xlab='x4')
par(mfrow=c(1,1))
# scaled data
par(mfrow=c(2,3))
plot(residuals(mlr_scaled),ylab='Residuals',xlab='Time')
plot(residuals(mlr_scaled),fitted(mlr),ylab='Residuals',xlab='Fitted values')
plot(residuals(mlr_scaled),df_scaled$x1,ylab='Residuals',xlab='x1')
plot(residuals(mlr_scaled),df_scaled$x2,ylab='Residuals',xlab='x2')
plot(residuals(mlr_scaled),df_scaled$x3,ylab='Residuals',xlab='x3')
plot(residuals(mlr_scaled),df_scaled$x4,ylab='Residuals',xlab='x4')
par(mfrow=c(1,1))

# Durbin-Watson tests.
# install.packages( "lmtest" )
library(lmtest)
dwtest(y ~ x1+x2+x3+x4, data=df)
dwtest(y ~ x1+x2+x3+x4, data=df_scaled)
```

```
# Some F-tests.
mlr1 <- lm(y ~ x1+x2+x4,data=df)  #remove x3 as insignificant from above
anova(mlr1,mlr)

mlr1_scaled <- lm(y ~ x1+x2+x4,data=df_scaled)  #remove x3 as insignificant from above
anova(mlr1_scaled,mlr_scaled)

mlr2 <- lm(y ~ x1+x2+offset(100.3*x3)+x4,data=df)  # from MLR results, B3^=100.3
summary(mlr2)
anova(mlr2,mlr)

mlr2_scaled <- lm(y ~ x1+x2+offset(0.03*x3)+x4,data=df_scaled)  # from MLR(scaled)
results, B3^=0.03163
summary(mlr2_scaled)
anova(mlr2_scaled,mlr_scaled)

# Predicting non-scaled inputs
con <- data.frame(x1=50000,x2=3,x3=60,x4=0)
predict(mlr,con,interval='confidence',level=0.95)
predict(mlr,con,interval='prediction',level=0.95)

# Predicting scaled inputs
# scaling new input
new_input <- data.frame(y=-1,x1=50000,x2=3,x3=60,x4=0)
df_input <- rbind(new_input,df)
process <- preProcess(df_input[c('x1','x2','x3','x4')], method=c("range"))
df_input_scaled <- predict(process, df_input)
## scaled input: x1=0.045286737 ,x2=0.1666667 ,x3=0.83673469  ,x4=0
to_predict <- data.frame(x1=0.045286737 ,x2=0.1666667 ,x3=0.83673469  ,x4=0)
predict(mlr_scaled,to_predict,interval='confidence',level=0.95)
predict(mlr_scaled,to_predict,interval='prediction',level=0.95)

predicted_val <- (0.05732971*(max(df['y'])-min(df['y'])))+min(df['y'])
predicted_lwr_conf <- (0.00543876*(max(df['y'])-min(df['y'])))+min(df['y'])
predicted_upr_conf <- (0.1092207*(max(df['y'])-min(df['y'])))+min(df['y'])
predicted_lwr_pred <- (-0.1444346 *(max(df['y'])-min(df['y'])))+min(df['y'])
predicted_upr_pred <- (0.259094*(max(df['y'])-min(df['y'])))+min(df['y'])
conf<-data.frame(fit=predicted_val,lwr=predicted_lwr_conf,upr=predicted_upr_conf)
pred<-data.frame(fit=predicted_val,lwr=predicted_lwr_pred,upr=predicted_upr_pred)
conf
pred
```