Akanksha Bansal **Assignment 1**
Data Mining **Jan 28, 2014**

Problem 1 Social science is an academic discipline concerned with society and the relationships among individuals within a society, which often rely primarily on empirical approaches. Social scientists commonly combine quantitative and qualitative approaches as part of a multi-strategy design in order to improve their understanding of this field. Thus hypothesis in this field are likely to be vague. Also there are not many ways to prove them. Data Mining has provided this field with a way where various hypothesis can be proved. Now social scientists can observe human behavior at the degree of granularity and variability which was not possible before. Also, big data makes now has made it feasible for machines to ask interesting questions which might even human might not even have considered. Because machine might be abel to find patterns between data which is practically not possible by any human. Then the presence of huge amount of data in the various fields has opened avenues of studying numerous aspects of this field. Data mining helps in formation of theories as well because now scientists have access to data which they had not even thought about before.

An example of above given things can be : If one tries to study the frequency with which words have been used in literature in the past couple of decades, it indicates that the usage of the word 'love' has reduced significantly. Now the presence of this data itself is not something a social scientists could have considered possible in a couple of decades ago. Based of this information now maybe they can hypothesize that maybe people are less verbose in expressing their feelings nowadays.

Problem 2 Probability that the distance between two vectors is $1 = \binom{1000}{1} * 2^{1000-1} * \frac{2}{2}$

Probability that the distance between two vectors is $2 = \binom{1000}{2} * 2^{1000-2} * \frac{2^2}{2}$

Probability that the distance between two vectors is $i =$ choosing the i bits which will be different * No of ways in which the rest $1000 - i$ bits can be set * No of ways in which the selected bits can be set

$=\binom{1000}{i} * 2^{1000-1}$

This value can be approximated to a Normal Distribution $Y = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-1}{2\sigma^2}(x-\mu)^2}$ where

$\mu = n * p = \frac{1000}{2} = 500$

$\sigma = \sqrt{np(1-p)} = \sqrt{1000\frac{1}{2*2}} = 15.81$

$P\left(495 < X < 505) = P(495 - 500 < X - \mu < 505 - 500) = P(\frac{495-500}{15.81} < \frac{X-\mu}{\sigma} < \frac{505-500}{15.81}\right)$

Replacing $\frac{X-\mu}{\sigma}$ with Z.

$P\,(495 < X < 505) = P\,(-0.32 < Z < 0.32) = 0.251$

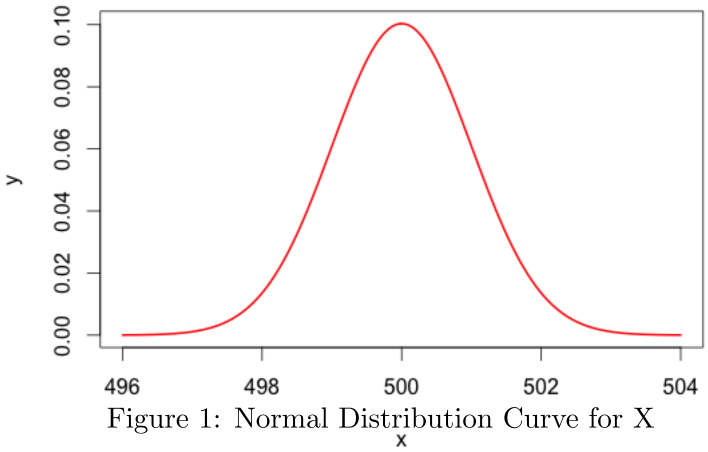

Figure 1: Normal Distribution Curve for X

```
$x = seq(-4, 4, length = 200) + 500;$
$y = 1/sqrt(2 * pi * 15.81) * exp(-(x - 500)^2/2);$
$plot(x, y, type = "l", lwd = 2, col = "red");$
```

Algorithm 1: Code Snippet

Problem 3 Here is the Code snippet from R for Bivariate Normal Distribution

```
library(MASS)
#Case 1
bivn < −mvrnorm(1000, mu = c(0, 0), Sigma = matrix(c(1, 0, 0, 1), 2))
#kerneldensityestimate
bivn.kde < −kde2d(bivn[, 1], bivn[, 2], n = 50)
# Basic plot of results
contour(bivn.kde)
image(bivn.kde)
persp(bivn.kde, phi = 45, theta = 30)
#contour + image
image(bivn.kde); contour(bivn.kde, add = T)
# perspective with theta and phi
persp(bivn.kde, phi = 60, theta = 45, shade = .1, border = NA)
```

Algorithm 2: Code Snippet

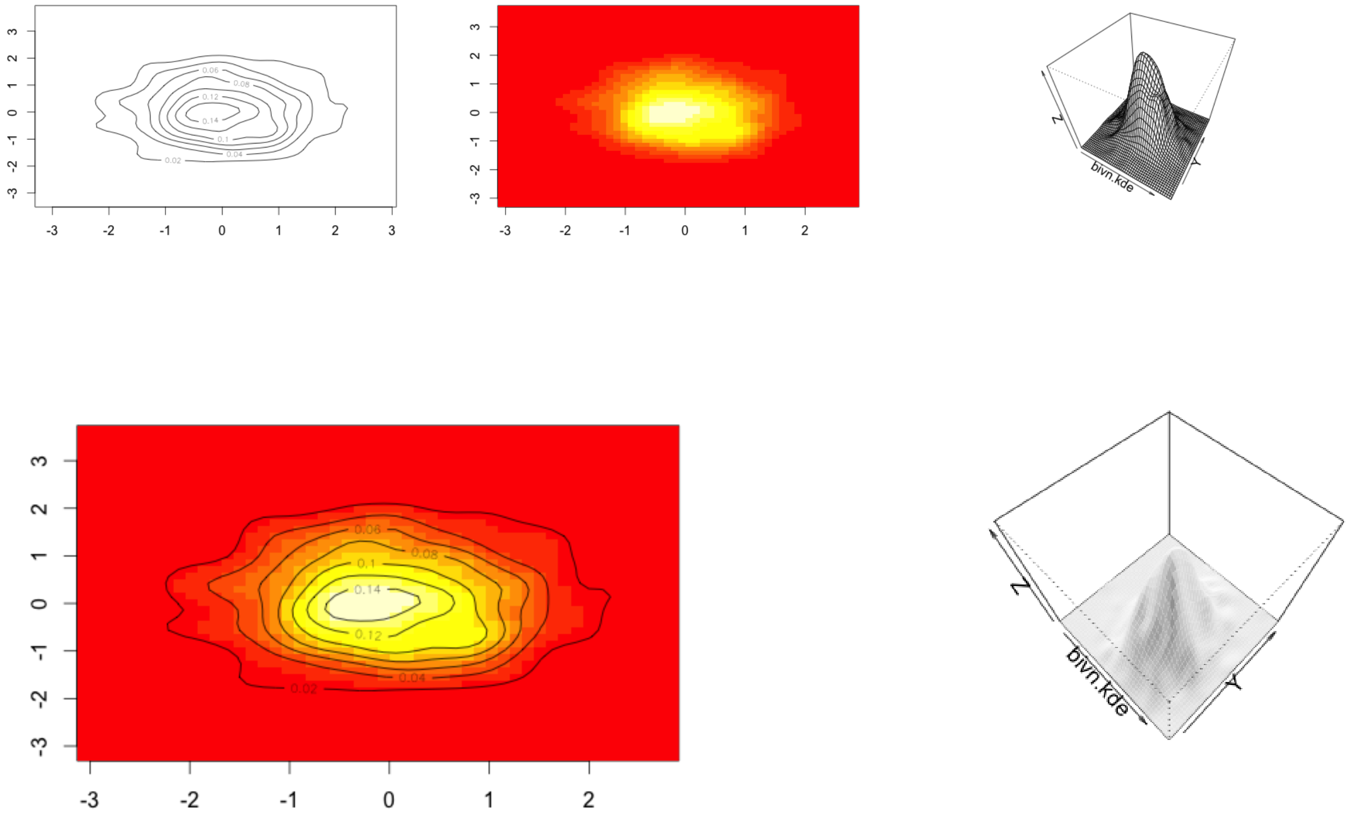

Figure 2: a. Contour b.Image c. persp image d. Contour + Image e. Final Persp

```
#Case ii
bivn < −mvrnorm(1000, mu = c(0, 0), Sigma = matrix(c(1, 0, 0, 4), 2))
bivn.kde < −kde2d(bivn[, 1], bivn[, 2], n = 50)
contour(bivn.kde)
image(bivn.kde)
persp(bivn.kde, phi = 45, theta = 30)
image(bivn.kde); contour(bivn.kde, add = T)
persp(bivn.kde, phi = 60, theta = 45, shade = .1, border = NA)
```

Algorithm 3: Code Snippet

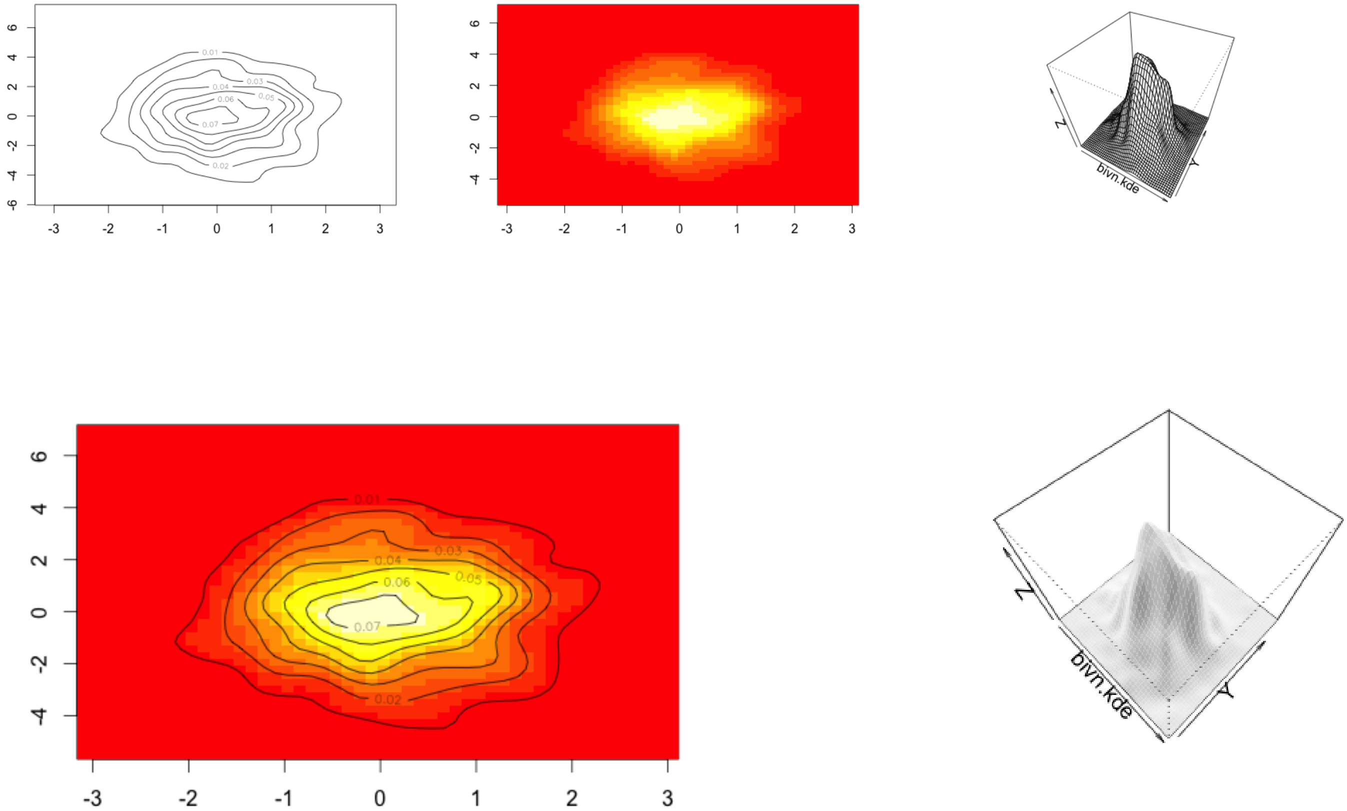

Figure 3: a. Contour b.Image c. persp image d. Contour + Image e. Final Persp

```
#Case iii
bivn < −mvrnorm(1000, mu = c(0, 0), Sigma = matrix(c(1, 1, 1, 4), 2))
bivn.kde < −kde2d(bivn[, 1], bivn[, 2], n = 50)
contour(bivn.kde)
image(bivn.kde)
persp(bivn.kde, phi = 45, theta = 30)
image(bivn.kde); contour(bivn.kde, add = T)
persp(bivn.kde, phi = 60, theta = 45, shade = .1, border = NA)
```

Algorithm 4: Code Snippet

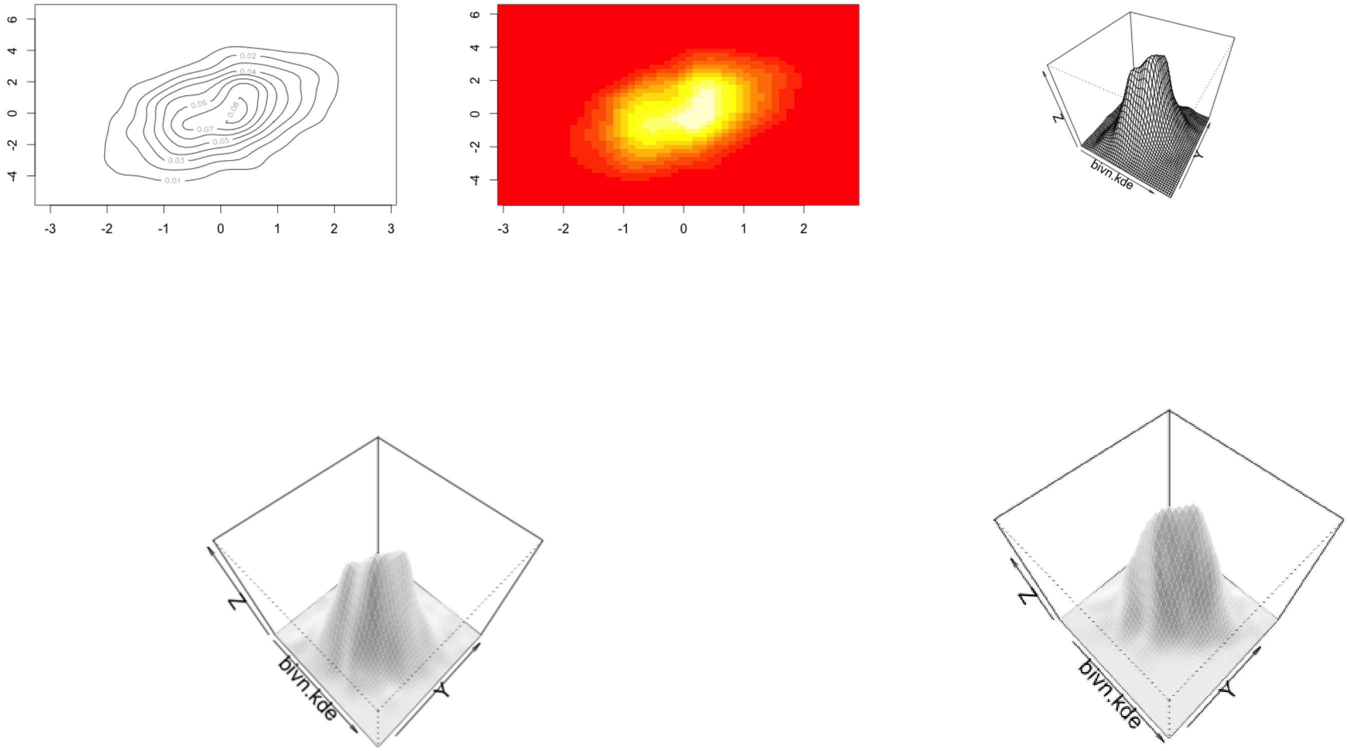

Figure 4: a. Contour b.Image c. persp image d. Contour + Image e. Final Persp

Problem 3c When Z is partitioned into X and Y, the conditional distribution of X given Y is

$X|Y = y \ \sim \ \mathcal{N}\left(\mu_x + \frac{\sigma_x}{\sigma_y}\rho(y - \mu_y),\, (1-\rho^2)\sigma_x^2\right).$

where ρ is the correlation coefficient between X and Y.

$\mu_x = 0, \mu_y = 0, \sigma_x = 1, \sigma_y = \sqrt{4} = 2, \rho = 0.5$

(i)In the given case $x = 1$,

$Y|X = x \ \sim \ \mathcal{N}\left(\mu_y + \frac{\sigma_y}{\sigma_x}\rho(x - \mu_x),\, (1-\rho^2)\sigma_y^2\right).$

$\Rightarrow Y|X = x \ \sim \ \mathcal{N}\left(\frac{2}{1} * \frac{1}{2} * 1,\, (1-(0.5)^2) * 2^2\right).$

$= Y|X = x \ \sim \ \mathcal{N}\left(1,\, 3\right).$

(ii)In the given case $y = 1$,

$\Rightarrow X|Y = y \ \sim \ \mathcal{N}\left(\frac{1}{2} * \frac{1}{2} * 1,\, (1-(\frac{1}{2})^2)\right).$

$= X|Y = y \ \sim \ \mathcal{N}\left(\frac{1}{4},\, \frac{3}{4}\right).$

Problem 4 $f(x \mid \mu, b) = \frac{1}{2*b} \exp\left(-\frac{|(x-\mu)|}{b}\right)$,

The corresponding probability density function for a sample of N independent identically distributed normal random variables is
$f(x_1, \ldots, x_n \mid \mu, b) = \prod_{i=1}^{N} f(x_i \mid \mu, b) = \left(\frac{1}{2b}\right)^N \exp\left(-\frac{\sum_{i=1}^{N}|(x_i-\mu)|}{b}\right)$,

This family of distributions has two parameters: μ, b, so we maximize the likelihood, over both parameters simultaneously, or if possible, individually.
$\mathcal{L}(\mu, b) = \log(f(x_1, \ldots, x_n \mid \mu, b))$
Since the logarithm is a continuous strictly increasing function over the range of the likelihood, the values which maximize the likelihood will also maximize its logarithm.
$\mathcal{L}(\mu, b) = (-N\log(2b)) - \left(\frac{\sum_{i=1}^{N}|(x_i-\mu)|}{b}\right)$

$\Rightarrow 0 = \frac{\partial}{\partial b}(-N\log(2b)) - \left(\frac{\sum_{i=1}^{N}|(x_i-\mu)|}{b}\right)$
$0 = \frac{-N}{b} + \frac{\sum_{i=1}^{N}|(x_i-\mu)|}{b^2}$
$\mathbf{b_{MLE}} = \frac{\sum_{i=1}^{N}|(x_i-\mu_{MLE})|}{n}$
$0 = \frac{\partial}{\partial \mu}\left(\frac{\sum_{i=1}^{N}|(x_i-\mu)|}{b}\right)$
$0 = \frac{\partial}{\partial \mu}\left(\sum_{i=1}^{N} \mid (x_i - \mu) \mid\right)$

This value will be zero when the half of the $x_i < \mu_{MLE}$ and the rest half $x_i > \mu_{MLE}$. Because

$$\frac{\partial \mid f \mid}{\partial f} = \begin{cases} -1 & \text{if } n < 0 \\ 1 & \text{if } n > 0 \end{cases}$$

$\mu_{MLE} = \mathcal{L}1\text{Median of } x$
The $\mathcal{L}1$ Median is defined to be any point which minimizes the sum of Euclidean distances to all points in the data set. The $\mathcal{L}1$ Median need not be one of the data. The $\mathcal{L}1$ reduces the standard univariate median. Any measurement X of the data set can be moved along the vector from L1 to X without changing the value of the median. The breakdown point of the $\mathcal{L}1$ median has been found to be 1/2. If we place just over 50% of the data at one point, then the median will always stay there. For example when just under 50% of the data is moved to infinity, the median remains in the vicinity of the majority of the data, since the bounded region resembles a point from infinity.

This value represents the case when $\sum_{i=1}^{N} \mid (x_i - \mu) \mid$ is minimized and not maximized. Because the curve of the $\mid x \mid$ indicates that the curve doesn't have any maximum. Thus the value will be maximized when $\mu = \infty$ However the original equation had the $\sum_{i=1}^{N} \mid (x_i - \mu) \mid$ with a negative sign, by minimizing the $\sum_{i=1}^{N} \mid (x_i - \mu) \mid$ we are in turn maximizing $\mathcal{L}(\mu, b)$.

Problem 5a Code Details for creating the box and scatter plots

```
boxdata <- with(Boston,boxplot(as.data.frame(
  Boston[,c('lstat','medv')]), main = "boxplot(LSTAT, MEDV)"))
identify(rep(1, length(Boston)), Boston, labels = seq_along(Boston))
#inserting the Labels of the cutoff
text(boxdata$stats[1][1],label=boxdata$stats[1][1])
plot(boxdata)

#Scatter plot LSAT on Y and MDEV in Y
plot(Boston$medv, Boston$lstat, main="LSTAT vs MEDV",xlab="MEDV", ylab="LSTAT")
#Showing the outliers in a different color
outlier.colors = (Boston$medv %in% boxdata$out)*1+(Boston$lstat %in% boxdata$out)*2
outlier.colors <- outlier.colors + 1
plot(Boston$medv, Boston$lstat, col=outlier.colors)

#Removing the outliers from the dataset to see the behavior of the model
medv<-Boston$medv[!Boston$medv %in% boxdata$out]
lstat<-Boston$lstat[!Boston$lstat %in% boxdata$out]
#box plot after removal of outliers
boxplot(as.data.frame(lstat), main = "tuned LSTAT",data=lstat)
boxplot(as.data.frame(medv), main = "tuned MEDV",data=medv)
```

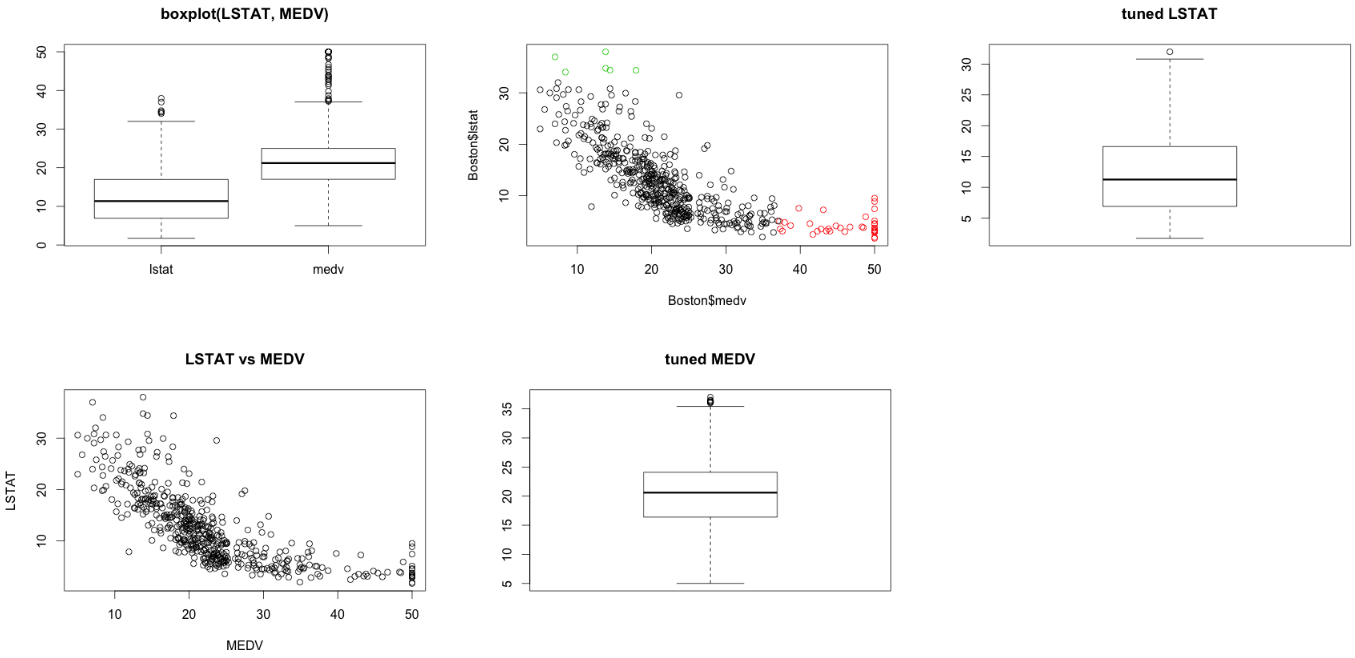

Figure 5: a. box plot b.Image with outliers highlighted c. Lsat with outliers removed d. ScatterPlot e. Medv with outliers removed

Problem 5b Model Genration with $lmedv = lstat + rm + crim + zn + chas$

```
Bostrain <- Boston[(1:300),]
Bostest <- Boston[(301:506),]
Bostrain$lmedv <- log(Bostrain$medv)
Bostest$lmedv <- log(Bostest$medv)

model <- lm(lmedv ~ lstat + rm + crim + zn + chas, data = Bostrain)
> model <- lm(lmedv ~ lstat + rm + crim + zn + chas, data = Bostrain)
> summary(model)
Call:
lm(formula = lmedv ~ lstat + rm + crim + zn + chas, data = Bostrain)

Residuals:
     Min       1Q   Median       3Q      Max
-0.34599 -0.08839 -0.01053  0.08300  0.58201

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.4307463  0.1150405  12.437  < 2e-16 ***
lstat       -0.0154217  0.0019305  -7.989 3.12e-14 ***
rm           0.2979299  0.0157242  18.947  < 2e-16 ***
crim        -0.0136857  0.0131904  -1.038  0.30033
zn           0.0003312  0.0003366   0.984  0.32593
chas         0.0728628  0.0278516   2.616  0.00935 **
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 0.1365 on 294 degrees of freedom
Multiple R-squared:  0.8235,  Adjusted R-squared:  0.8205
F-statistic: 274.4 on 5 and 294 DF,  p-value: < 2.2e-16

> mse(model$fitted.values,Bostrain$lmedv)
[1] 0.01825346
```

Based on the t values removing crim and zn can be tried

Some additional tests done

Testing the models behavior when *crim* is dropped from the model equation

```
> model2 <- lm(lmedv ~ lstat + rm + zn + chas, data = Bostrain)
> summary(model2)

Call:
lm(formula = lmedv ~ lstat + rm + zn + chas, data = Bostrain)

Residuals:
     Min       1Q   Median       3Q      Max
-0.35694 -0.08899 -0.00866  0.08584  0.59462

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.4391618  0.1147691  12.540  < 2e-16 ***
lstat       -0.0160199  0.0018426  -8.694 2.47e-16 ***
rm           0.2965788  0.0156722  18.924  < 2e-16 ***
```

```
zn            0.0003927  0.0003314   1.185   0.2370
chas          0.0701688  0.0277339   2.530   0.0119 *
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 0.1365 on 295 degrees of freedom
Multiple R-squared:  0.8229,   Adjusted R-squared:  0.8205
F-statistic: 342.7 on 4 and 295 DF,  p-value: < 2.2e-16

> mse(model2$fitted.values,Bostrain$lmedv)
[1] 0.01832029
```

Better results

Testing the models behavior when *zn* is dropped from the model equation too

```
> model2 <- lm(lmedv ~ lstat + rm + chas, data = Bostrain)
> summary(model2)

Call:
lm(formula = lmedv ~ lstat + rm + chas, data = Bostrain)

Residuals:
     Min        1Q    Median        3Q       Max
-0.36148  -0.08795  -0.00785   0.08745   0.59999

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.444879   0.114746  12.592   <2e-16 ***
lstat       -0.016580   0.001782  -9.304   <2e-16 ***
rm           0.297589   0.015660  19.004   <2e-16 ***
chas         0.068371   0.027711   2.467   0.0142 *
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 0.1366 on 296 degrees of freedom
Multiple R-squared:  0.8221,   Adjusted R-squared:  0.8202
F-statistic: 455.8 on 3 and 296 DF,  p-value: < 2.2e-16

> mse(model2$fitted.values,Bostrain$lmedv)
[1] 0.01840749
```

Based on the t-values it doesn't make sense to remove *chase* from the model but considering that it is a dummy value (as read from the variables details) removing it from the model can be tried.

```
> model2 <- lm(lmedv ~ lstat + rm, data = Bostrain)
> #testModel2<-predict.lm(model2, newdata = Bostest, se.fit=TRUE)
> summary(model2)

Call:
lm(formula = lmedv ~ lstat + rm, data = Bostrain)

Residuals:
     Min       1Q   Median       3Q      Max
-0.36741 -0.08730 -0.00947  0.08630  0.58912

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.423104   0.115382  12.334   <2e-16 ***
lstat       -0.016137   0.001788  -9.024   <2e-16 ***
rm           0.301199   0.015724  19.155   <2e-16 ***
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 0.1378 on 297 degrees of freedom
Multiple R-squared:  0.8184,  Adjusted R-squared:  0.8172
F-statistic: 669.2 on 2 and 297 DF,  p-value: < 2.2e-16

> mse(model2$fit,Bostrain$lmedv)
[1] 0.01878604
```

There is a slight increase in the MS value.

Part 5(b) Testing of the model with Test Data

```
testModel<-predict.lm(model, newdata = Bostest, se.fit=TRUE)
> summary(testModel)
               Length Class  Mode
fit            206    -none- numeric
se.fit         206    -none- numeric
df               1    -none- numeric
residual.scale   1    -none- numeric
> mse(testModel$fit,Bostest$lmedv)
[1] 0.1116501
```

The MSE obtained after running the model on the test data = 0.1116501. MSE generated when run over Train data = 0.01825346

Thus Final Model becomes : $LMEDV\ LSTAT + RM + CHAS$

Problem 5c Code for generating Residual plot for the model

```
library(car)
library(hydroGOF)
Boston$lmedv <- log(Boston$medv)
fit <- lm(lmedv ~ lstat + rm + crim + zn + chas, data = Boston)
> summary(fit, data = Boston)
Call:
lm(formula = lmedv ~ lstat + rm + crim + zn + chas, data = Boston)

Residuals:
     Min       1Q   Median       3Q      Max
-0.70215 -0.12208 -0.02339  0.10076  0.92747

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  2.6094450  0.1240140  21.042  < 2e-16 ***
lstat       -0.0314198  0.0019323 -16.260  < 2e-16 ***
rm           0.1343025  0.0174121   7.713 6.70e-14 ***
crim        -0.0103008  0.0012541  -8.213 1.85e-15 ***
zn           0.0004434  0.0004529   0.979    0.328
chas         0.1556354  0.0379311   4.103 4.76e-05 ***
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 0.2147 on 500 degrees of freedom
Multiple R-squared:  0.7267,  Adjusted R-squared:  0.724
F-statistic:   266 on 5 and 500 DF,  p-value: < 2.2e-16

> outlierTest(fit)
    rstudent unadjusted p-value Bonferonni p
413 4.451419         1.0532e-05    0.0053292
372 4.090101         5.0262e-05    0.0254330
375 4.051619         5.8976e-05    0.0298420

qqPlot(fit , main="QQ Plot")
leveragePlots(fit)
plot(fit)
```

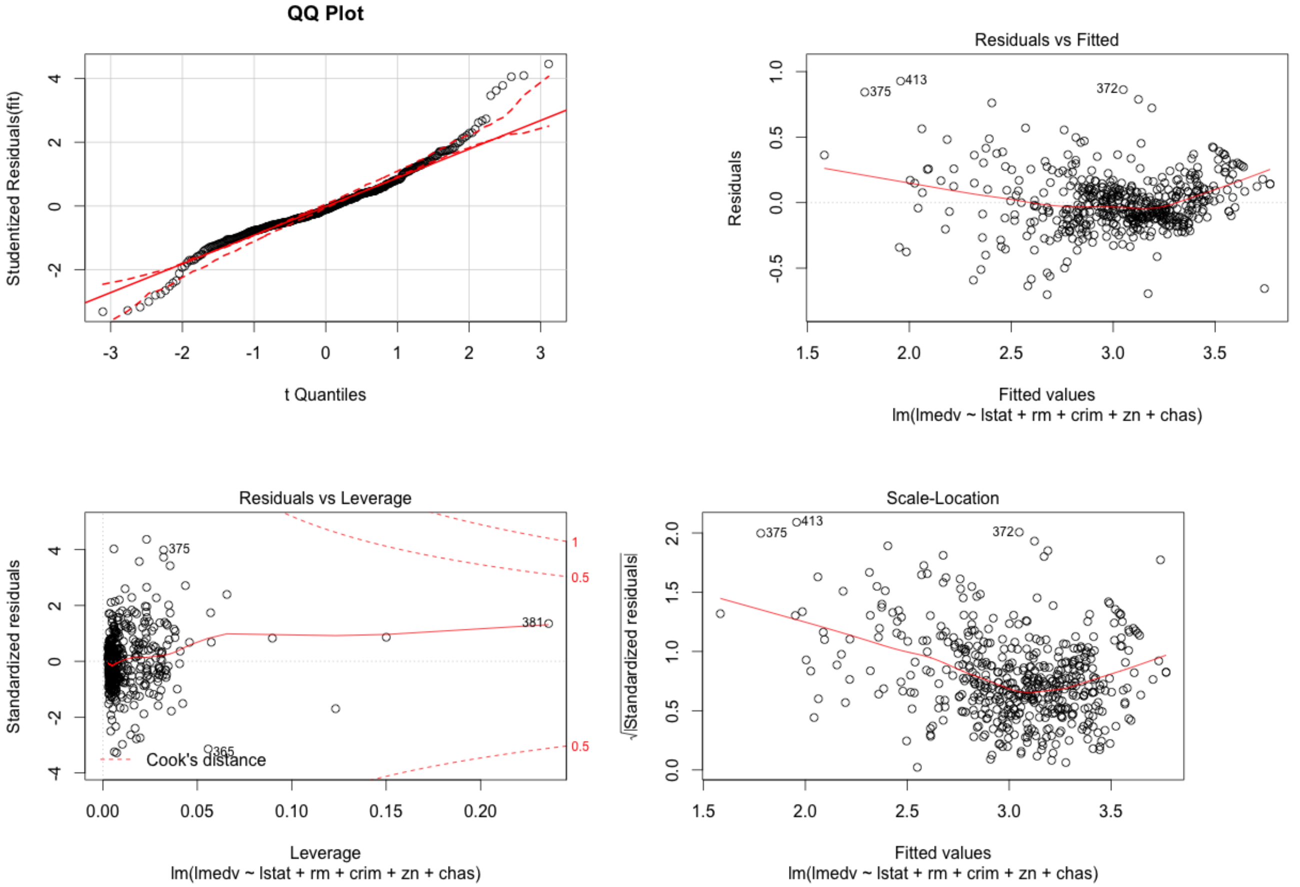

Figure 6:

It depends on what the applications of the model might be, and what is the permissible amount of error. The residual plot indicates that the data is centered around the lmedv value of 3.25. Around the value of lmedv=3.25 the model's error value going towards 0. But for other values of lmddv the error values are more. Therefore, If these values are not acceptable in the application, then MLR model is not the most ideal model for this problem. The train data's mse also indicates that the model did not perform as well as the test data. However if the error values are within range of being permissible MLR can be chosen.