# Unraveling Stories from Your Massive Datasets Through Machine Learning

Venkat N Gudivada
Department of Computer Science
East Carolina University
Greenville, North Carolina
USA

# Outline

# Reference for This Tutorial

- Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. An Introduction to Statistical Learning with Applications in R, Springer, 2013.
- For more theoretical exposition, see Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, 2009.

# Resources for Machine Learning

- Kaggle
- RStudio in the cloud, for dummies
- RStudio in the Cloud I: Amazon Web Services
- Top 10 Machine Learning Algorithms
- 35 Free Online Books on Machine Learning
- BigML

# Big Data

- Big Data: Promises and Problems
- Five Vs: Volume, Velocity, Variety, Veracity, Value
- Data sparsity in feature space
- The Top 10 AI And Machine Learning Use Cases Everyone Should Know About
- 8 Inspirational Applications of Deep Learning
- Non-technical Introduction to Machine Learning

# Machine Learning

- Data is everywhere
- Generating actionable insights from data
- Artificial Intelligence, SQL Analytics, Data Warehousing, Pattern Recognition, Data Mining, Machine Learning, Data Science
- Self-driving cars

# Supervised vs. Unsupervised Learning

- Supervised – learn from labeled examples
- Unsupervised – learn without labeled examples
- Semi-supervised

# Parametric vs. Nonparametric Models

- Model is assumed and parameters are learned from the training data
- Model is discovered
- Model interpretability
- Prediction accuracy

# Training and Testing

- Training and test error
- Bias and variance trade-off
- Resampling and bootstrapping
- Validation Set Approach
- Cross-Validation (CV) – Leave One Out Cross Validation (LOOCV) and $k-$fold validation

# Classification and Regression

- Classification: Bayes classifiers, K-means, Nearest Neighbor, Linear Discriminant Analysis, Support Vector Machines (SVM), Tree-based methods (decision trees, Bagging, Random Forests, Boosting)
- Regression: linear (simple and multiple), nonlinear regression (polynomial, step functions, basis functions, regression splines, smoothing splines, local regression, generalized additive models (GAMs))

# Simple and Multiple Linear Regression

*AllData 2017*

*23 April 2017*

```
# long R commands wrap around after 80 characters
library(knitr)
```

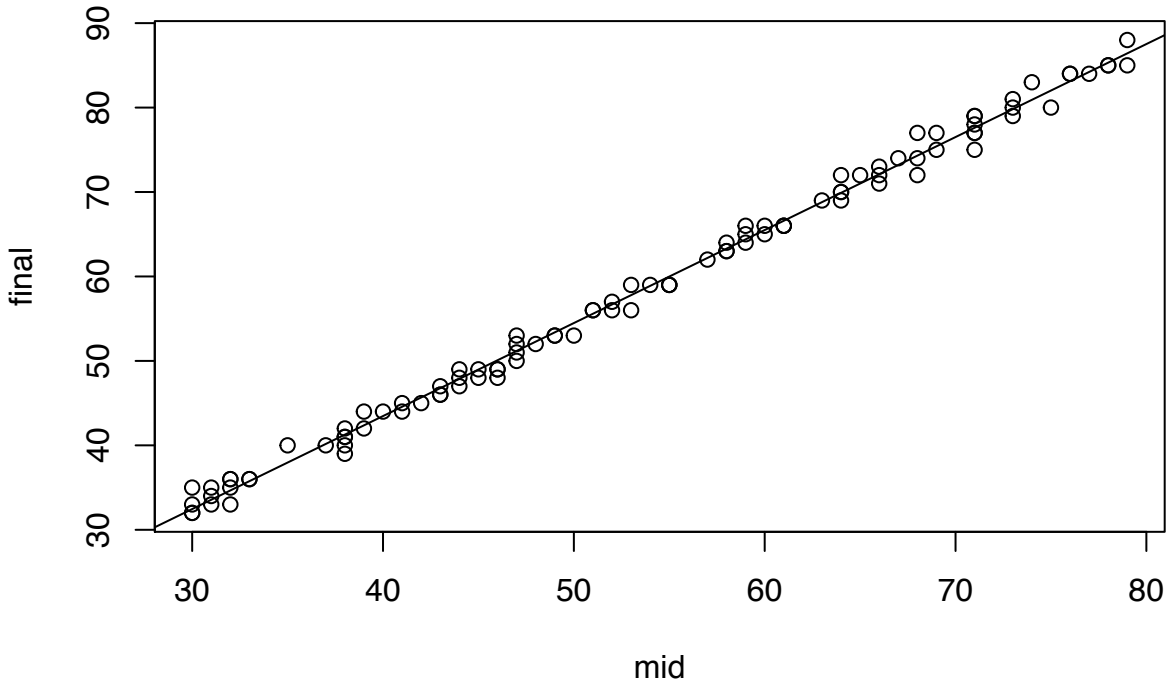
```
## Warning: package 'knitr' was built under R version 3.3.2
```

```
opts_chunk$set(tidy.opts = list(width.cutoff = 80), tidy = TRUE)
```

## Simple Linear Regression/ Univariate Regression

dataset: a subset of midterm exam scores and the corresponding final exam scores

```
# generate 100 midterm exam scores in the range (30,80) which are uniformly
# distributed

mid <- floor(runif(100, min = 30, max = 80))

# generate 100 final exam scores based on the midterm scores. add zero-mean
# random error. overall, students did 10% better if we exclude the random error

final <- floor(1.1 * mid + 10 * rnorm(100, mean = 0, sd = 0.1))


par(mfrow = c(1, 1))

# generate scatterplot
plot(mid, final)

# build simple linear model
exam.model1 <- lm(final ~ mid)

# superimpose the regression line on the scatterplot
abline(exam.model1)
```




```
# explore the model using summary() command
summary(exam.model1)
```

```
##
## Call:
## lm(formula = final ~ mid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6086 -0.6242 -0.1840  0.7147  2.6952
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.583458   0.416862    -1.4    0.165
## mid          1.101297   0.007513   146.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.095 on 98 degrees of freedom
## Multiple R-squared:  0.9955, Adjusted R-squared:  0.9954
## F-statistic: 2.149e+04 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
# compute confidence intervals
confint(exam.model1)
```

```
##                 2.5 %   97.5 %
## (Intercept) -1.410707 0.243791
## mid          1.086389 1.116206
```

```
# generate diagnostic plots
par(mfrow = c(2, 2))
plot(exam.model1)

# divide the dataset (100 instances/observations) into training set (60%) and
```

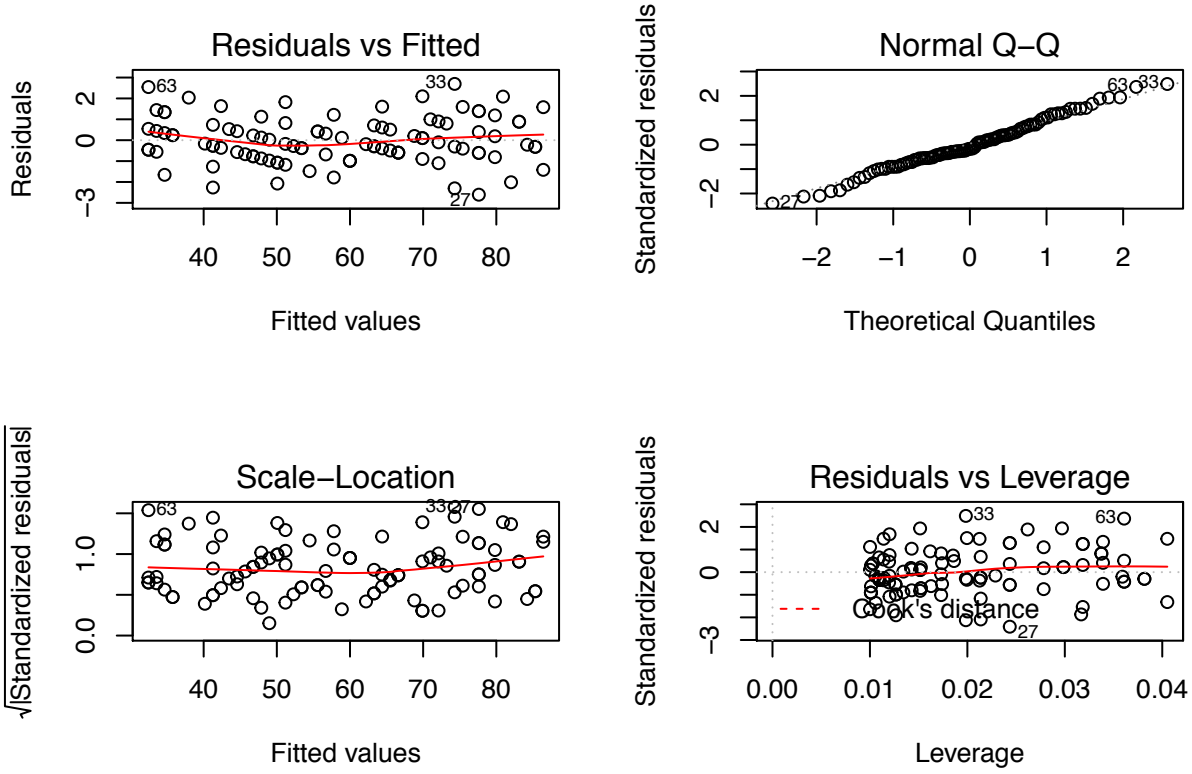
```
# test set (40%)

# depending on your installation, you may not have all the required packages. if
# so, install packages as needed.

# install.packages('combinat', dependencies = c('Depends', 'Suggests'))

# install.packages('klaR', dependencies = c('Depends', 'Suggests'))

# load packages
library(MASS)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.2
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```




```
library(klaR)

# create a data frame from mid and final vectors
exam.data <- data.frame(mid, final)

# 60% - 40% train/test split of the dataset
trainIndex <- createDataPartition(exam.data$final, p = 0.6, list = FALSE)
trainData <- exam.data[trainIndex, ]
testData <- exam.data[-trainIndex, ]
```

```r
# train a simple linear model using the training data
exam.model2 <- lm(final ~ mid, data = trainData)

# explore the model using summary() command
summary(exam.model2)
```

```
##
## Call:
## lm(formula = final ~ mid, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3115 -0.7833 -0.1088  0.8927  2.4876
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.48401    0.56683  -0.854    0.397
## mid          1.09988    0.01026 107.235   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.182 on 60 degrees of freedom
## Multiple R-squared:  0.9948, Adjusted R-squared:  0.9947
## F-statistic: 1.15e+04 on 1 and 60 DF,  p-value: < 2.2e-16
```

```r
# did exam.model2 fit the data better than exam.model1?

# test how well exam.model2 performs on the test data by making predictions using
# all rows of test data on column 1 (midterm exam scores). there are 38 instances
# in testData

# prediction with confidence intervals
predict(exam.model2, data.frame(mid = c(testData[, 1])), interval = "confidence")
```

```
##         fit      lwr      upr
## 1  41.31149 40.87720 41.74577
## 2  32.51244 31.94806 33.07681
## 3  77.60758 77.13619 78.07897
## 4  50.11054 49.77504 50.44605
## 5  55.60995 55.30597 55.91392
## 6  50.11054 49.77504 50.44605
## 7  40.21161 39.76228 40.66094
## 8  77.60758 77.13619 78.07897
## 9  64.40900 64.08664 64.73136
## 10 79.80734 79.30364 80.31104
## 11 85.30675 84.71751 85.89598
## 12 66.60876 66.26933 66.94819
## 13 74.30793 73.88218 74.73369
## 14 32.51244 31.94806 33.07681
## 15 63.30912 62.99364 63.62460
## 16 68.80853 68.44814 69.16891
## 17 44.61113 44.21899 45.00327
## 18 83.10699 82.55266 83.66131
## 19 38.01184 37.53121 38.49248
## 20 33.61232 33.06520 34.15943
```

```
## 21 72.10817 71.71044 72.50590
## 22 55.60995 55.30597 55.91392
## 23 51.21042 50.88355 51.53730
## 24 85.30675 84.71751 85.89598
## 25 69.90841 69.53629 70.28053
## 26 56.70983 56.40835 57.01132
## 27 62.20924 61.89942 62.51905
## 28 41.31149 40.87720 41.74577
## 29 75.40782 74.96728 75.84836
## 30 41.31149 40.87720 41.74577
## 31 69.90841 69.53629 70.28053
## 32 47.91078 47.55508 48.26648
## 33 56.70983 56.40835 57.01132
## 34 60.00947 59.70711 60.31184
## 35 52.31030 51.99097 52.62964
## 36 35.81208 35.29877 36.32539
## 37 79.80734 79.30364 80.31104
## 38 63.30912 62.99364 63.62460
```

```r
# the first five true final exam scores in the testData are: 39, 85, 54, 70, 32.
# predicted values by the linear model are: 39.04576, 82.86149, 53.28587,
# 69.716777, 32.47340

# prediction with prediction intervals
predict(exam.model2, data.frame(mid = c(testData[, 1])), interval = "prediction")
```

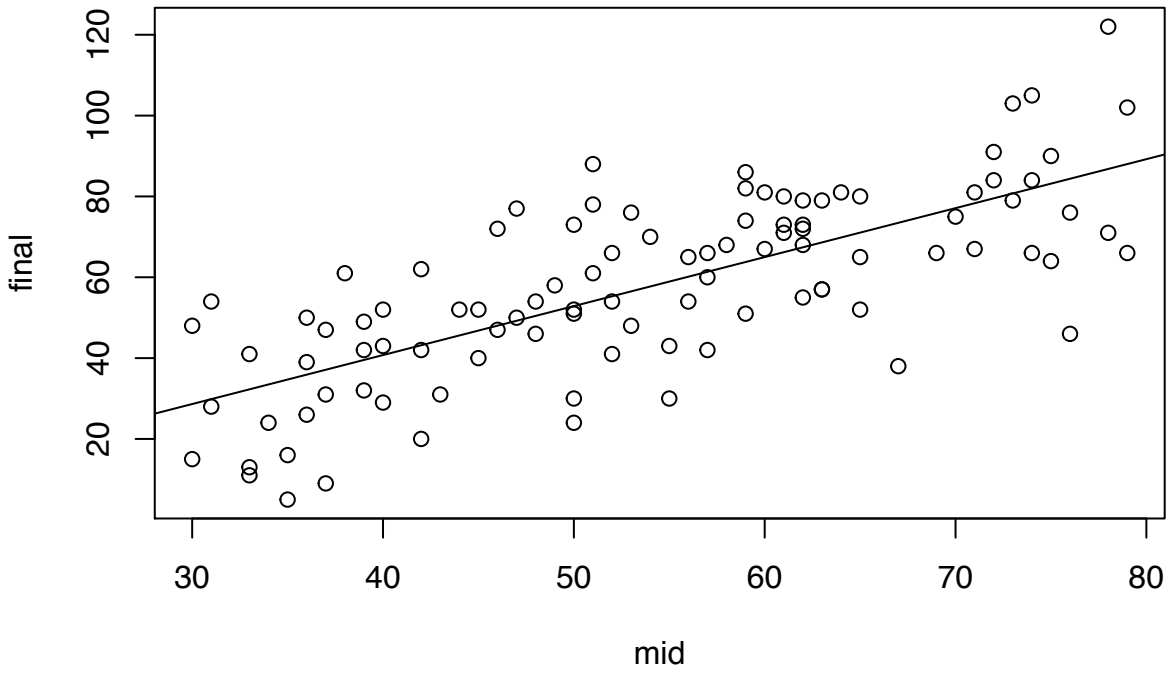
```
##         fit      lwr      upr
## 1  41.31149 38.90721 43.71576
## 2  32.51244 30.08130 34.94358
## 3  77.60758 75.19633 80.01883
## 4  50.11054 47.72213 52.49895
## 5  55.60995 53.22576 57.99413
## 6  50.11054 47.72213 52.49895
## 7  40.21161 37.80457 42.61864
## 8  77.60758 75.19633 80.01883
## 9  64.40900 62.02240 66.79560
## 10 79.80734 77.38957 82.22512
## 11 85.30675 82.86972 87.74378
## 12 66.60876 64.21980 68.99773
## 13 74.30793 71.90519 76.71068
## 14 32.51244 30.08130 34.94358
## 15 63.30912 60.92344 65.69480
## 16 68.80853 66.41650 71.20056
## 17 44.61113 42.21411 47.00815
## 18 83.10699 80.67816 85.53582
## 19 38.01184 35.59877 40.42492
## 20 33.61232 31.18513 36.03951
## 21 72.10817 69.71023 74.50611
## 22 55.60995 53.22576 57.99413
## 23 51.21042 48.82321 53.59763
## 24 85.30675 82.86972 87.74378
## 25 69.90841 67.51458 72.30223
## 26 56.70983 54.32596 59.09370
## 27 62.20924 59.82430 64.59417
## 28 41.31149 38.90721 43.71576
```

```
## 29 75.40782 73.00240 77.81323
## 30 41.31149 38.90721 43.71576
## 31 69.90841 67.51458 72.30223
## 32 47.91078 45.51945 50.30211
## 33 56.70983 54.32596 59.09370
## 34 60.00947 57.62550 62.39345
## 35 52.31030 49.92411 54.69649
## 36 35.81208 33.39228 38.23188
## 37 79.80734 77.38957 82.22512
## 38 63.30912 60.92344 65.69480
```

```
# next, we generate final exam score with greater standard deviation and examine
# its effect on RSE and R^2


# same as before
mid <- floor(runif(100, min = 30, max = 80))
mid
```

```
##   [1] 59 40 65 48 52 39 47 36 40 57 33 70 53 51 56 74 33 75 35 55 31 35 37
##  [24] 71 74 52 61 48 60 53 52 30 62 78 72 42 38 57 42 79 50 33 46 61 43 62
##  [47] 50 50 72 56 62 51 75 36 40 71 65 57 55 79 44 63 37 58 67 76 37 51 49
##  [70] 54 62 61 63 73 45 59 50 39 47 34 42 30 78 63 73 46 74 69 36 59 39 76
##  [93] 60 31 59 64 62 65 45 50
```

```
# SD for the final exam scores is much greater now
final <- floor(1.1 * mid + 10 * rnorm(100, mean = 0, sd = 1.6))


par(mfrow = c(1, 1))

# generate scatterplot
plot(mid, final)

# build simple linear model
exam.model3 <- lm(final ~ mid)

# superimpose the regression line on the scatterplot
abline(exam.model3)
```




```
# explore the model using summary() command
summary(exam.model3)
```

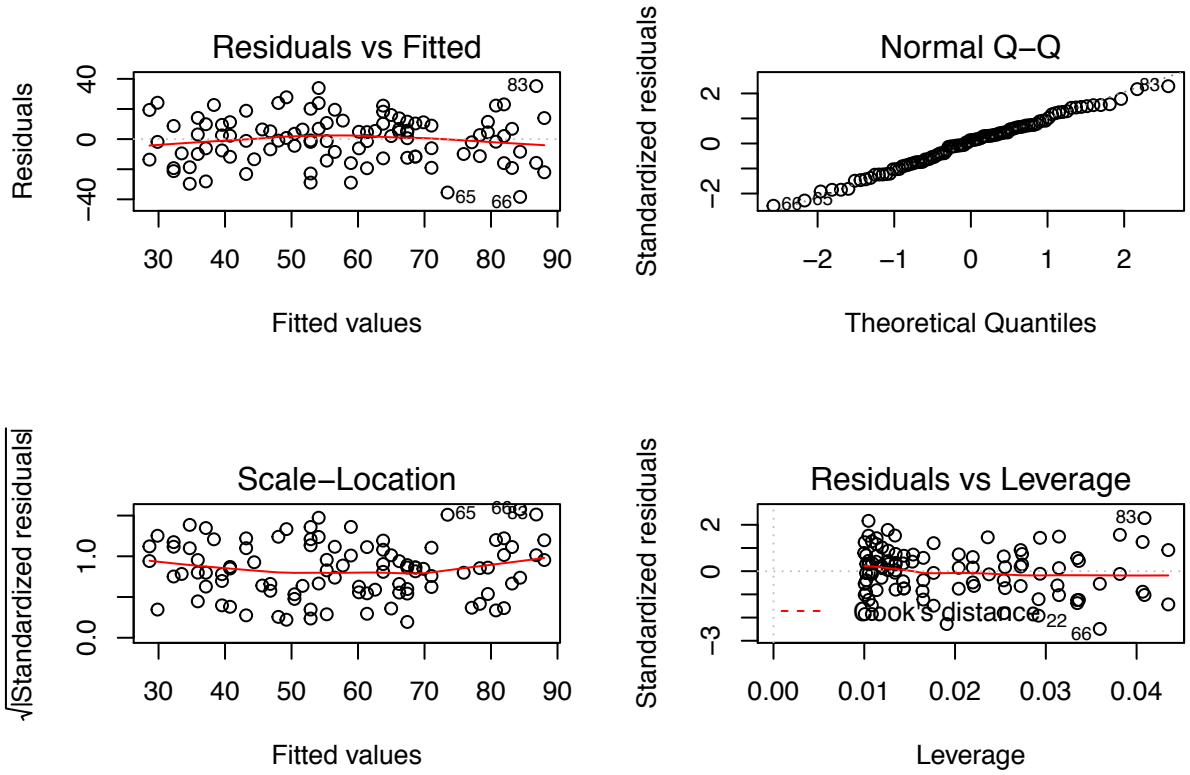
```
##
## Call:
## lm(formula = final ~ mid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.377 -11.624   2.029  10.458  35.200
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -7.7158     6.3911  -1.207     0.23
## mid           1.2117     0.1148  10.554   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.71 on 98 degrees of freedom
## Multiple R-squared:  0.532,  Adjusted R-squared:  0.5272
## F-statistic: 111.4 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
# notice lower t-value for mid (from 149.868 to 10.706); increased RSE (from
# 1.157 to 15.92); and decreased R^2 (from 0.9957 to 0.5391)

# compute confidence intervals
confint(exam.model3)
```

```
##                 2.5 %   97.5 %
## (Intercept) -20.398775 4.967212
## mid           0.983911 1.439580
```

```
# previous range for mid was (1.082978 1.1127141) is increased to the range
# (0.9249675, 1.345877)
```

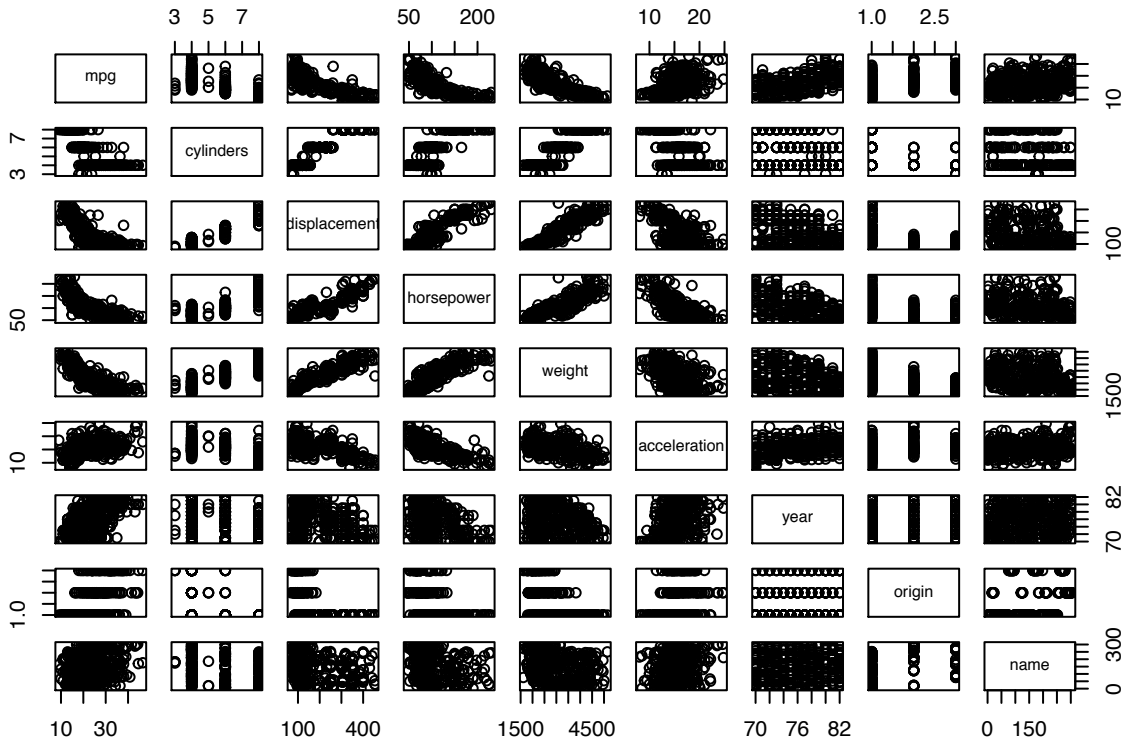
```r
# generate diagnostic plots
par(mfrow = c(2, 2))
plot(exam.model3)
```




## Multiple Linear Regression

Multiple linear regression on the Auto dataset.

```r
# problem 9, page 122. Gareth James, Daniela Witten, Trevor Hastie and Robert
# Tibshirani. An Introduction to Statistical Learning with Applications in R,
# Springer, 2013

library(ISLR)

# (a) scatterplot matrix which includes all of the variables in the data set
pairs(Auto)
```




```
# (b) matrix of correlations between the variables. exclude the name variable,
# which is qualitative
cor(subset(Auto, select = -name))
```

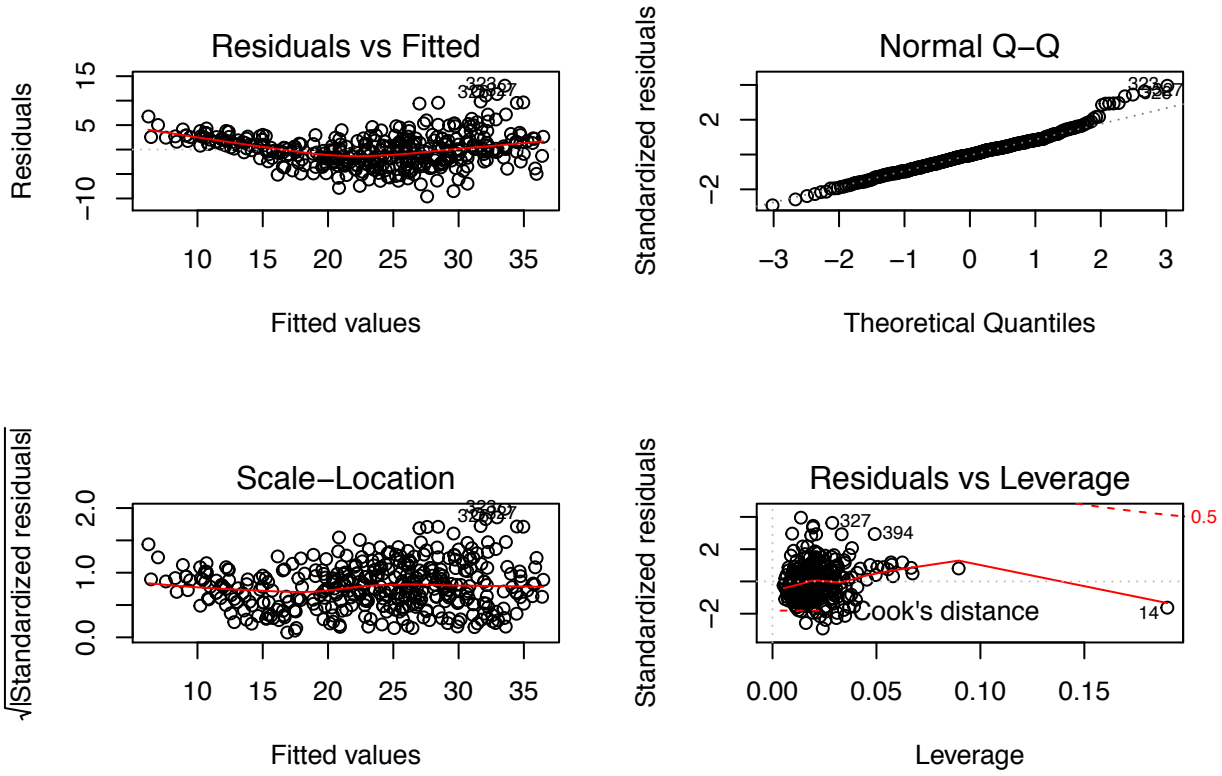
```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
##              acceleration       year     origin
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

```
# (c) multiple linear regression with mpg as the response and all other variables
# except name as the predictors
lmModel1 <- lm(mpg ~ . - name, data = Auto)

# examine residuals, coefficients, standard error, t values, and p-values
summary(lmModel1)
```

```
##
```

```
## Call:
## lm(formula = mpg ~ . - name, data = Auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```r
# (i) does a relationship exists between the predictors and the response?  reject
# null hypothesis based on F-statistic is far from 1, with a small p-value

# which predictors appear to have a statistically significant relationship to the
# response?  based on p-values, displacement, weight, year, and origin have a
# statistically significant whereas, cylinders, horsepower, and acceleration are
# not significant

# (iii) what does the coefficient for the year variable suggest?  regression
# coefficient for year is 0.750773. every one year, mpg increases by an amount
# equal to this coefficient

# (d) diagnostic plots of linear regression fit
par(mfrow = c(2, 2))
plot(lmModel1)
```




```
# from the leverage plot, point 14 has high leverage

# explore possible outliers using plot of studentized residuals
plot(predict(lmModel1), rstudent(lmModel1))
# notice that there are data with a value greater than 3. possible outliers exist


# (e) linear model with interaction effects. from the correlation matrix, two
# highest correlated pairs are cylinders and displacement; displacement and
# weight
attach(Auto)
```

```
## The following object is masked from package:ggplot2:
##
##     mpg
```

```
lmModel2 <- lm(mpg ~ cylinders * displacement + displacement * weight)
summary(lmModel2)
```

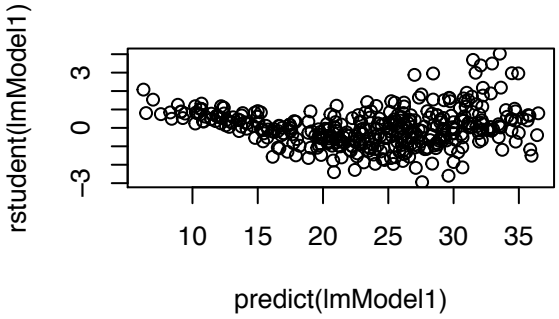
```
##
## Call:
## lm(formula = mpg ~ cylinders * displacement + displacement *
##     weight)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.2934  -2.5184  -0.3476   1.8399  17.7723
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)             5.262e+01  2.237e+00  23.519  < 2e-16 ***
## cylinders               7.606e-01  7.669e-01   0.992    0.322
## displacement           -7.351e-02  1.669e-02  -4.403 1.38e-05 ***
## weight                 -9.888e-03  1.329e-03  -7.438 6.69e-13 ***
## cylinders:displacement -2.986e-03  3.426e-03  -0.872    0.384
## displacement:weight     2.128e-05  5.002e-06   4.254 2.64e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.103 on 386 degrees of freedom
## Multiple R-squared:  0.7272, Adjusted R-squared:  0.7237
## F-statistic: 205.8 on 5 and 386 DF,  p-value: < 2.2e-16
```

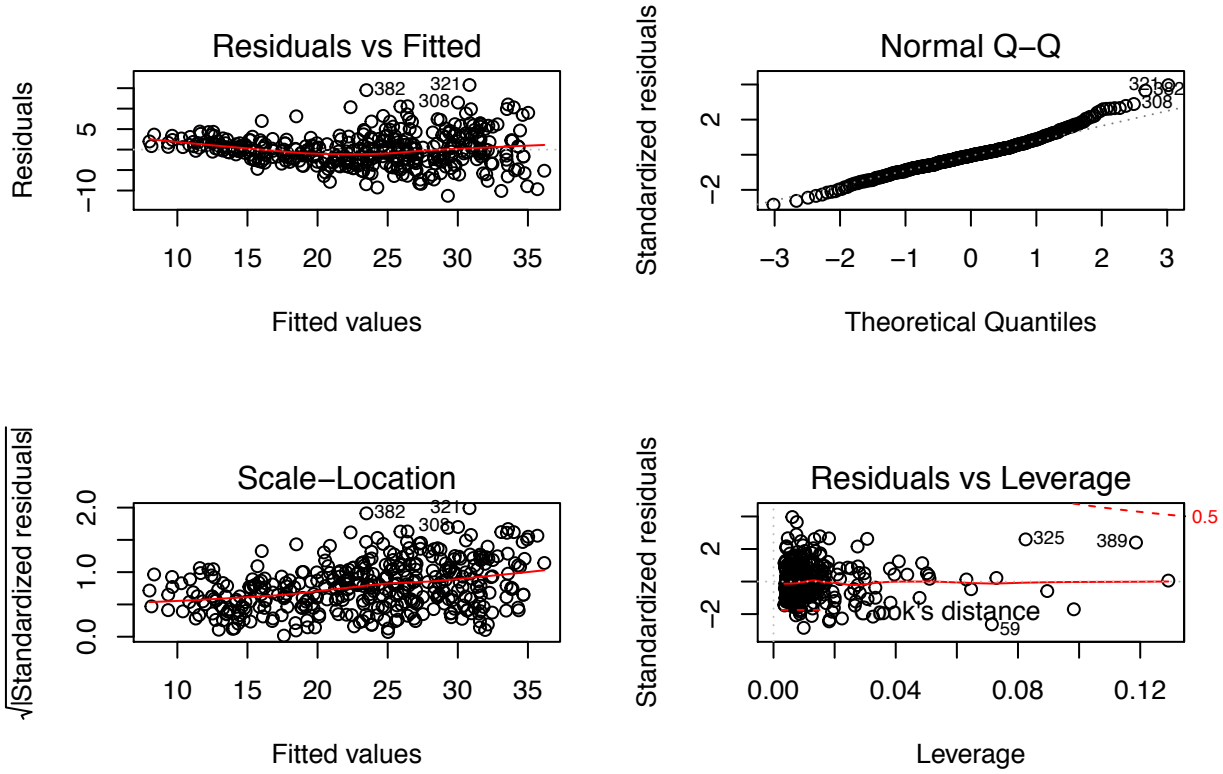
```
# (f) apply transformations on the predictor variables and construct a linear
# regression model

lmModel3 <- lm(mpg ~ log(weight) + sqrt(horsepower) + acceleration + I(acceleration^2))
summary(lmModel3)
```

```
##
## Call:
## lm(formula = mpg ~ log(weight) + sqrt(horsepower) + acceleration +
##     I(acceleration^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.2932  -2.5082  -0.2237   2.0237  15.7650
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       178.30303   10.80451  16.503  < 2e-16 ***
## log(weight)       -14.74259    1.73994  -8.473 5.06e-16 ***
## sqrt(horsepower)   -1.85192    0.36005  -5.144 4.29e-07 ***
## acceleration       -2.19890    0.63903  -3.441 0.000643 ***
## I(acceleration^2)   0.06139    0.01857   3.305 0.001037 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.99 on 387 degrees of freedom
## Multiple R-squared:  0.7414, Adjusted R-squared:  0.7387
## F-statistic: 277.3 on 4 and 387 DF,  p-value: < 2.2e-16
```

```
# diagnostic plots of linear regression fit
par(mfrow = c(2, 2))
```



```
plot(lmModel3)
```

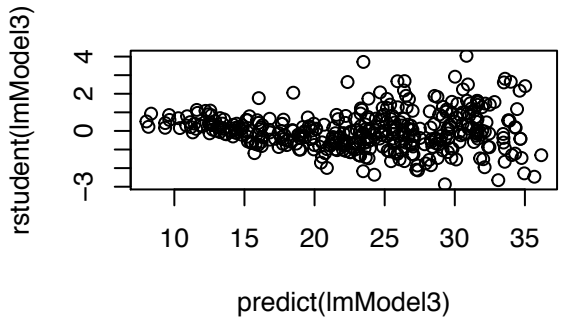



```
# based on p-values, log(weight), sqrt(horsepower), and acceleration^2 have
# statistical significance. leverage plot indicates more than three points with
# high leverage. residuals vs fitted plot indicates heteroskedasticity
# (unconstant variance over mean). Q-Q plot indicates some unnormality of the
# residuals

# studentized residuals displays potential outliers (>3)
plot(predict(lmModel3), rstudent(lmModel3))


# note from the correlation matrix, displacement, horsepower and weight show a
# similar nonlinear pattern against our response mpg. this nonlinear pattern is
# very close to a log form, try log(mpg) as the response variable

lmModel3 <- lm(log(mpg) ~ cylinders + displacement + horsepower + weight + acceleration +
    year + origin, data = Auto)

summary(lmModel3)
```

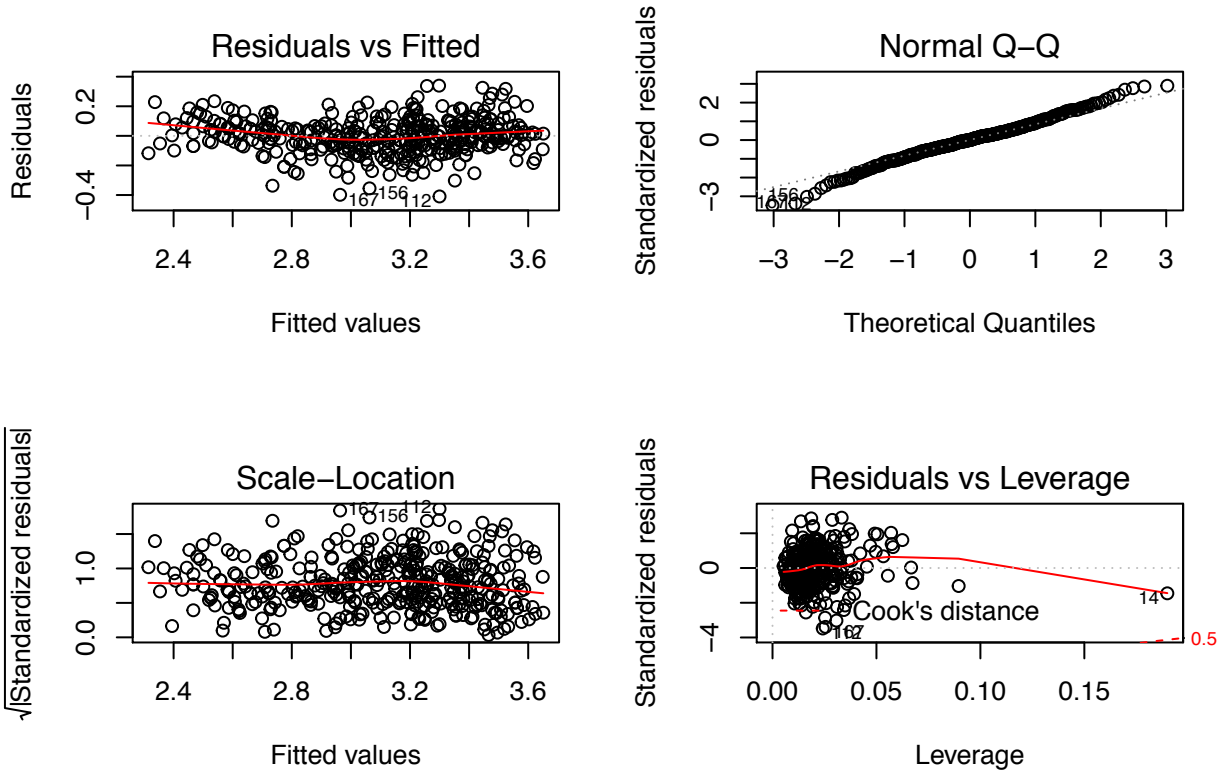
```
##
## Call:
## lm(formula = log(mpg) ~ cylinders + displacement + horsepower +
##     weight + acceleration + year + origin, data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40955 -0.06533  0.00079  0.06785  0.33925
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.751e+00  1.662e-01  10.533  < 2e-16 ***
## cylinders    -2.795e-02  1.157e-02  -2.415  0.01619 *
## displacement  6.362e-04  2.690e-04   2.365  0.01852 *
## horsepower   -1.475e-03  4.935e-04  -2.989  0.00298 **
## weight       -2.551e-04  2.334e-05 -10.931  < 2e-16 ***
## acceleration -1.348e-03  3.538e-03  -0.381  0.70339
## year          2.958e-02  1.824e-03  16.211  < 2e-16 ***
## origin        4.071e-02  9.955e-03   4.089 5.28e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1191 on 384 degrees of freedom
## Multiple R-squared:  0.8795, Adjusted R-squared:  0.8773
## F-statistic: 400.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

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




```
plot(lmModel3)
```




```
plot(predict(lmModel3), rstudent(lmModel3))
```