SVM Regression

Ethan Huynh and Ryan Gagliardi

Load packages and data.

```
library(e1071)
library(MASS)
df <- read.csv('diamonds.csv')
df$cut <- factor(df$cut)
df$color <- factor(df$color)
df$clarity <- factor(df$clarity)
```

Divide into train, test, validate

Cut the dataset from 50000 rows to 10000 rows randomly Then divide into train, test, and validate sets

```
set.seed(6164)
i <- sample(1:nrow(df), 0.2*nrow(df), replace=FALSE)
df2 <- df[i,]

spec <- c(train=.6, test=.2, validate=.2)
i2 <- sample(cut(1:nrow(df2),nrow(df2)*cumsum(c(0,spec)), labels=names(spec)))
train <- df2[i2=="train",]
test <- df2[i2=="test",]
vald <- df2[i2=="validate",]
```

Data exploration

Let’s explore the data with R functions and plots.

```
str(df)
```

```
## 'data.frame':    53940 obs. of  11 variables:
##  $ X      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Factor w/ 5 levels "Fair","Good",..: 3 4 2 4 2 5 5 5 1 5 ...
##  $ color  : Factor w/ 7 levels "D","E","F","G",..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Factor w/ 8 levels "I1","IF","SI1",..: 4 3 5 6 4 8 7 3 6 5 ...
##  $ depth  : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num  55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int  326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

```
dim(df)
```

```
## [1] 53940    11
```

```
head(df)
```

	X <int>	carat <dbl>	cut <fct>	color <fct>	clarity <fct>	depth <dbl>	table <dbl>	price <int>	x <dbl>
1	1	0.23	Ideal	E	SI2	61.5	55	326	3.95
2	2	0.21	Premium	E	SI1	59.8	61	326	3.89
3	3	0.23	Good	E	VS1	56.9	65	327	4.05
4	4	0.29	Premium	I	VS2	62.4	58	334	4.20
5	5	0.31	Good	J	SI2	63.3	58	335	4.34
6	6	0.24	Very Good	J	VVS2	62.8	57	336	3.94

6 rows | 1-10 of 12 columns

Try linear regression

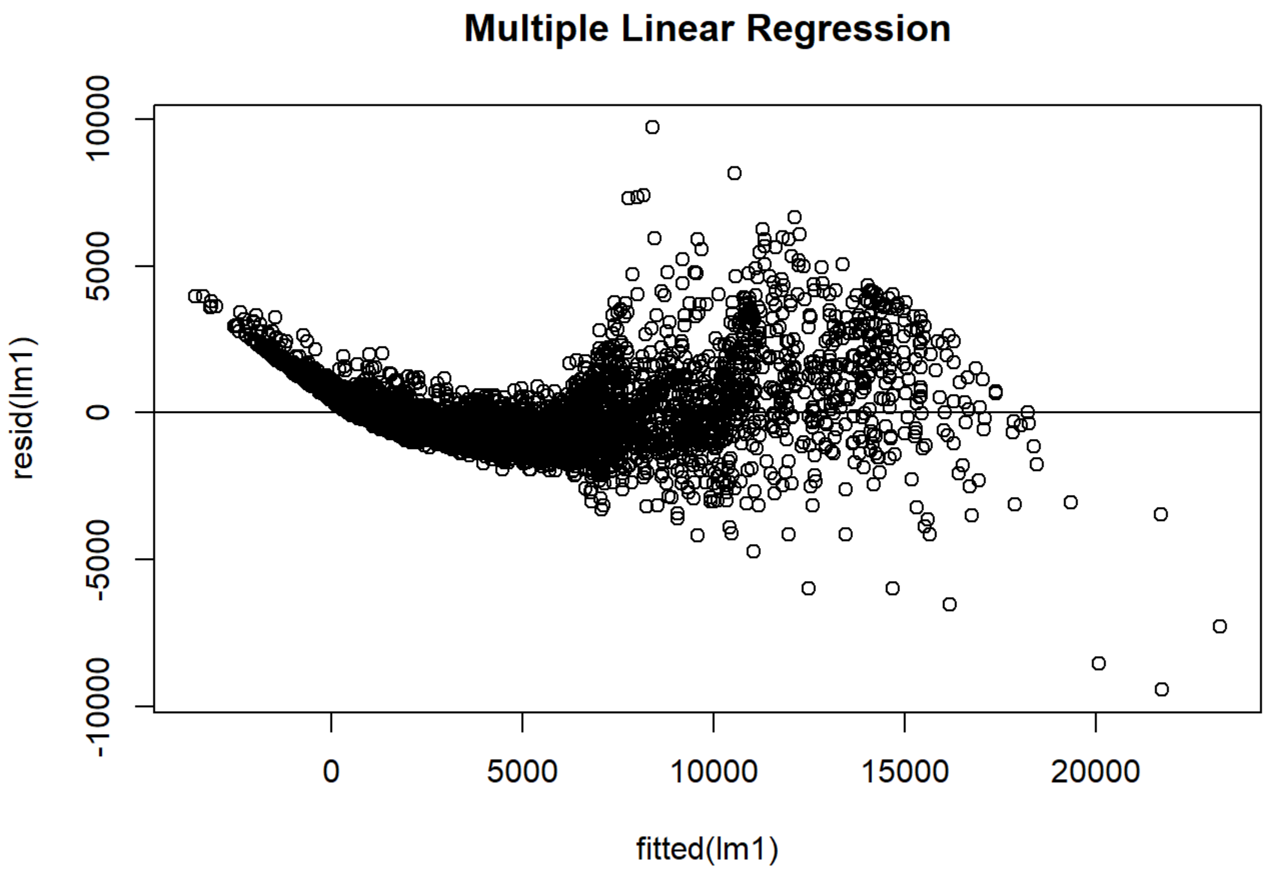
```
lm1 <- lm(price~carat+cut+color+clarity, data=train)
pred <- predict(lm1, newdata=test)
cor_lm1 <- cor(pred, test$price)
mse_lm1 <- mean((pred-test$price)^2)
print(paste('correlation:', cor_lm1))
```

```
## [1] "correlation: 0.954954252674521"
```

```
print(paste('mse:', mse_lm1))
```

```
## [1] "mse: 1411873.88028662"
```

```
plot(fitted(lm1), resid(lm1), main = "Multiple Linear Regression")
abline(0,0)
```



Try a linear kernel

```
svm1 <- svm(price~carat+cut+color+clarity, data=train, kernel="linear", cost=10, scale=TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = price ~ carat + cut + color + clarity, data = train,
##     kernel = "linear", cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  linear
##        cost:  10
##       gamma:  0.05263158
##     epsilon:  0.1
##
##
## Number of Support Vectors:  3709
```

```
pred <- predict(svm1, newdata=test)
cor_svm1 <- cor(pred, test$price)
mse_svm1 <- mean((pred - test$price)^2)
print(paste('correlation:', cor_svm1))
```

```
## [1] "correlation: 0.953739672285428"
```

```
print(paste('mse:', mse_svm1))
```

```
## [1] "mse: 1483027.07936527"
```

Tune

```
tune_svm1 <- tune(svm, price~carat+cut+color+clarity, data=vald, kernel="linear"
                  ,ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune_svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##    10
##
## - best performance: 1779915
##
## - Detailed performance results:
##    cost   error dispersion
## 1 1e-03 4214561   698204.5
## 2 1e-02 2294278   634759.8
## 3 1e-01 1925718   586994.4
## 4 1e+00 1789970   520623.5
## 5 5e+00 1782006   517130.6
## 6 1e+01 1779915   514074.2
## 7 1e+02 1780373   514531.5
```

Evaluate on best linear svm

```
pred <- predict(tune_svm1$best.model, newdata=test)
cor_svm1_tune <- cor(pred, test$price)
mse_svm1_tune <- mean((pred - test$price)^2)
print(paste('correlation:', cor_svm1_tune))
```

```
## [1] "correlation: 0.953599454070172"
```

```
print(paste('mse:', mse_svm1_tune))
```

```
## [1] "mse: 1528525.90730627"
```

Try a polynomial kernel

```
svm2 <- svm(price~carat+cut+color+clarity, data=train, kernel="polynomial", cost=10, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = price ~ carat + cut + color + clarity, data = train,
##     kernel = "polynomial", cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##       gamma:  0.05263158
##      coef.0:  0
##     epsilon:  0.1
##
##
## Number of Support Vectors:  2043
```

```
pred <- predict(svm2, newdata=test)
cor_svm2 <- cor(pred, test$price)
mse_svm2 <- mean((pred - test$price)^2)
print(paste('correlation:', cor_svm2))
```

```
## [1] "correlation: 0.96842913324452"
```

```
print(paste('mse:', mse_svm2))
```

```
## [1] "mse: 991646.99260546"
```

Try a radial kernel

```
svm3 <- svm(price~carat+cut+color+clarity, data=train, kernel="radial", cost=10, gamma=1, scale=
TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = price ~ carat + cut + color + clarity, data = train,
##     kernel = "radial", cost = 10, gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  10
##       gamma:  1
##     epsilon:  0.1
##
##
## Number of Support Vectors:  1799
```

```
pred <- predict(svm3, newdata=test)
cor_svm3 <- cor(pred, test$price)
mse_svm3 <- mean((pred - test$price)^2)
print(paste('correlation:', cor_svm3))
```

```
## [1] "correlation: 0.97602067104327"
```

```
print(paste('mse:', mse_svm3))
```

```
## [1] "mse: 756603.969680508"
```

Tune hyperperameters

```
set.seed(6164)
tune.out <- tune(svm, price~carat+cut+color+clarity, data=vald, kernel="radial",
                 ranges=list(cost=c(0.1,1,10,100,1000),
                             gamma=c(0.5,1,2,3,4)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10   0.5
##
## - best performance: 630171.2
##
## - Detailed performance results:
##     cost gamma      error dispersion
## 1  1e-01   0.5  2675946.0   598106.1
## 2  1e+00   0.5   781702.4   148929.7
## 3  1e+01   0.5   630171.2   147514.9
## 4  1e+02   0.5   811036.3   164220.4
## 5  1e+03   0.5  1524322.2   672550.2
## 6  1e-01   1.0  6460936.3  1279012.7
## 7  1e+00   1.0  1485277.7   322913.5
## 8  1e+01   1.0  1220489.7   242253.4
## 9  1e+02   1.0  1437360.8   363334.7
## 10 1e+03   1.0  2832305.3  1592424.1
## 11 1e-01   2.0 11298033.7  1820977.6
## 12 1e+00   2.0  3903961.1   815033.0
## 13 1e+01   2.0  2798059.9   663087.4
## 14 1e+02   2.0  3098164.2  1010991.9
## 15 1e+03   2.0  3688224.6  1690294.1
## 16 1e-01   3.0 12433616.7  1922858.4
## 17 1e+00   3.0  5111599.3  1034683.5
## 18 1e+01   3.0  3687532.8   888769.9
## 19 1e+02   3.0  3858319.8  1134995.6
## 20 1e+03   3.0  4820777.3  2303266.7
## 21 1e-01   4.0 12874287.3  1957537.4
## 22 1e+00   4.0  5653567.1  1112598.2
## 23 1e+01   4.0  4139672.7   960718.9
## 24 1e+02   4.0  4300903.1  1118461.8
## 25 1e+03   4.0  5623796.5  3413394.0
```

```
svm4 <- svm(price~carat+cut+color+clarity, data=train, kernel="radial", cost=100, gamma=0.5, sca
le=TRUE)
summary(svm4)
```

```
##
## Call:
## svm(formula = price ~ carat + cut + color + clarity, data = train,
##     kernel = "radial", cost = 100, gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  100
##       gamma:  0.5
##     epsilon:  0.1
##
##
## Number of Support Vectors:  1729
```

```
pred <- predict(svm4, newdata=test)
cor_svm4 <- cor(pred, test$price)
mse_svm4 <- mean((pred - test$price)^2)
print(paste('correlation:', cor_svm4))
```

```
## [1] "correlation: 0.974356953810741"
```

```
print(paste('mse:', mse_svm4))
```

```
## [1] "mse: 808415.736695084"
```

Best method

Radial was the best method that was tried, however only marginally better than the others. The predictors are very correlated to the target in this case which means that no matter which method is used, a fantastic correlation will be generated. The data was situated in a way that made Radial slightly edge out the other options but the difference was so small that it hardly matters in reality.