# Notebook 1 Regression

Matthew Naughton

CS 4375.003

Portfolio: Kernel and Ensemble Methods

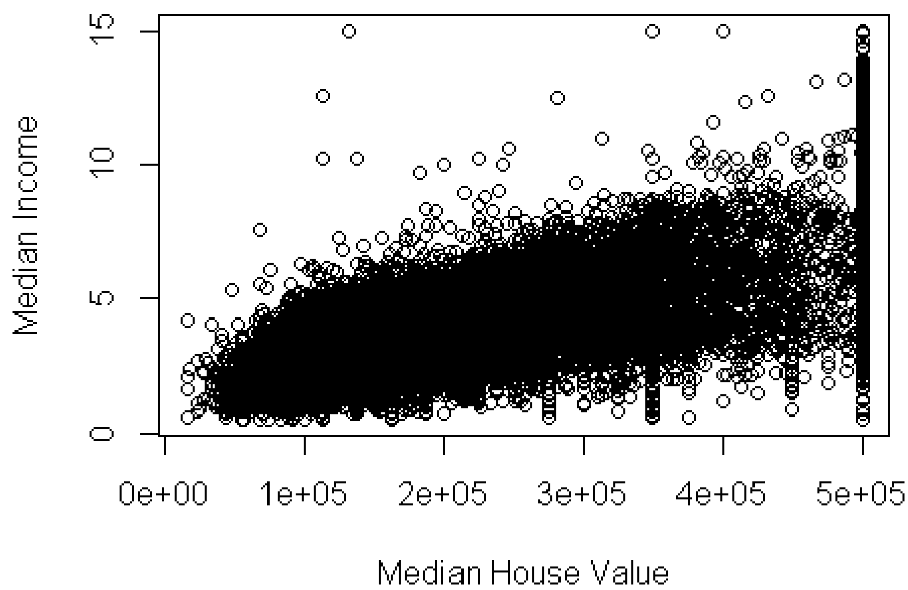
```
set.seed(1234)
df <- read.csv(file = 'housing.csv')

# install.packages("ggplot2") ##uncomment and run if not installed
# install.packages("e1071")   ##uncomment and run if not installed
# install.packages("MASS")   ##uncomment and run if not installed
library(ggplot2)
library(e1071)
library(MASS)

ocean_factor <- as.factor(df$ocean_proximity)
spec <- c(train=.6, test=.2, validate=.2)
i <- sample(cut(1:nrow(df),
                nrow(df)*cumsum(c(0,spec)), labels=names(spec)))
train <- df[i=="train",]
test <- df[i=="test",]
vald <- df[i=="validate",]
#i <- sample(1:nrow(df), nrow(df)*0.75, replace=FALSE)
#train <- df[i,]
#test <- df[-i,]

# Exploring the data:
head(df)

##   longitude latitude housing_median_age total_rooms total_bedrooms
population
## 1   -122.23    37.88                 41         880            129
322
## 2   -122.22    37.86                 21        7099           1106
2401
## 3   -122.24    37.85                 52        1467            190
496
## 4   -122.25    37.85                 52        1274            235
558
## 5   -122.25    37.85                 52        1627            280
565
## 6   -122.25    37.85                 52         919            213
413
##   households median_income median_house_value ocean_proximity
## 1        126        8.3252             452600        NEAR BAY
```

```
## 2       1138        8.3014              358500        NEAR BAY
## 3        177        7.2574              352100        NEAR BAY
## 4        219        5.6431              341300        NEAR BAY
## 5        259        3.8462              342200        NEAR BAY
## 6        193        4.0368              269700        NEAR BAY
```

```r
tail(df)
```

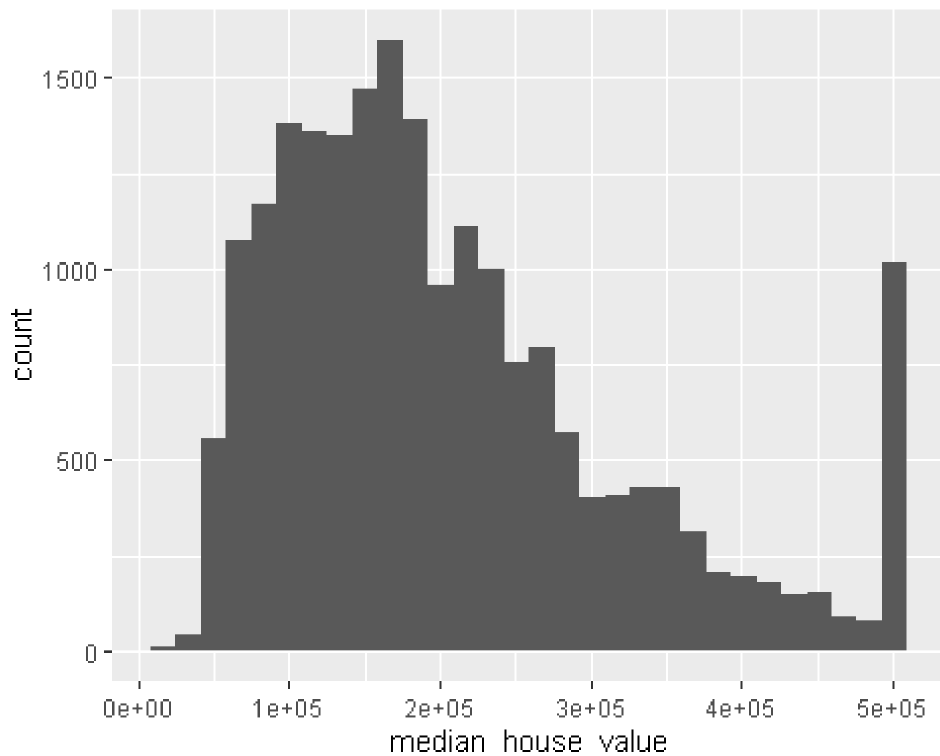
```
##       longitude latitude housing_median_age total_rooms total_bedrooms
## 20635   -121.56    39.27                 28        2332            395
## 20636   -121.09    39.48                 25        1665            374
## 20637   -121.21    39.49                 18         697            150
## 20638   -121.22    39.43                 17        2254            485
## 20639   -121.32    39.43                 18        1860            409
## 20640   -121.24    39.37                 16        2785            616
##       population households median_income median_house_value
ocean_proximity
## 20635       1041        344        3.7125             116800
INLAND
## 20636        845        330        1.5603              78100
INLAND
## 20637        356        114        2.5568              77100
INLAND
## 20638       1007        433        1.7000              92300
INLAND
## 20639        741        349        1.8672              84700
INLAND
## 20640       1387        530        2.3886              89400
INLAND
```

```r
plot(df$median_house_value, df$median_income, xlab="Median House Value",
ylab="Median Income")
```

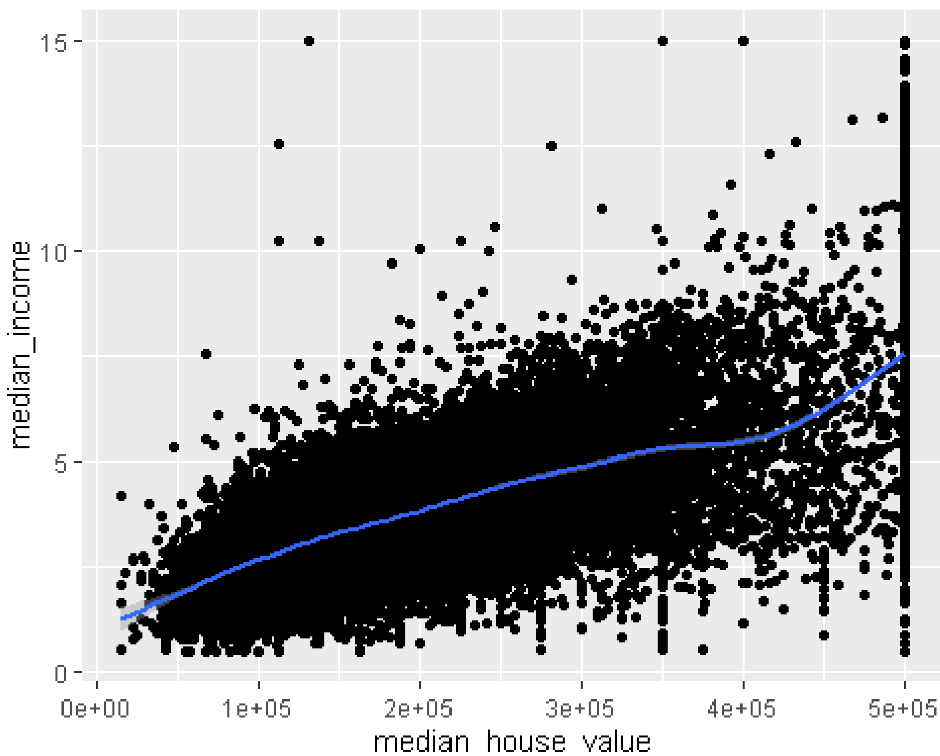



```
str(df)

## 'data.frame':    20640 obs. of  10 variables:
##  $ longitude         : num  -122 -122 -122 -122 -122 ...
##  $ latitude          : num  37.9 37.9 37.9 37.9 37.9 ...
##  $ housing_median_age: num  41 21 52 52 52 52 52 52 42 52 ...
##  $ total_rooms       : num  880 7099 1467 1274 1627 ...
##  $ total_bedrooms    : num  129 1106 190 235 280 ...
##  $ population        : num  322 2401 496 558 565 ...
##  $ households        : num  126 1138 177 219 259 ...
##  $ median_income     : num  8.33 8.3 7.26 5.64 3.85 ...
##  $ median_house_value: num  452600 358500 352100 341300 342200 ...
##  $ ocean_proximity   : chr  "NEAR BAY" "NEAR BAY" "NEAR BAY" "NEAR BAY"
...

names(df)

##  [1] "longitude"          "latitude"           "housing_median_age"
##  [4] "total_rooms"        "total_bedrooms"     "population"
##  [7] "households"         "median_income"      "median_house_value"
## [10] "ocean_proximity"

summary(df)

##    longitude         latitude     housing_median_age  total_rooms
##  Min.   :-124.3   Min.   :32.54   Min.   : 1.00      Min.   :    2
##  1st Qu.:-121.8   1st Qu.:33.93   1st Qu.:18.00      1st Qu.: 1448
##  Median :-118.5   Median :34.26   Median :29.00      Median : 2127
```

```
##  Mean   :-119.6   Mean   :35.63   Mean   :28.64      Mean   : 2636
##  3rd Qu.:-118.0   3rd Qu.:37.71   3rd Qu.:37.00      3rd Qu.: 3148
##  Max.   :-114.3   Max.   :41.95   Max.   :52.00      Max.   :39320
##
##  total_bedrooms     population      households     median_income
##  Min.   :   1.0   Min.   :    3   Min.   :   1.0   Min.   : 0.4999
##  1st Qu.: 296.0   1st Qu.:  787   1st Qu.: 280.0   1st Qu.: 2.5634
##  Median : 435.0   Median : 1166   Median : 409.0   Median : 3.5348
##  Mean   : 537.9   Mean   : 1425   Mean   : 499.5   Mean   : 3.8707
##  3rd Qu.: 647.0   3rd Qu.: 1725   3rd Qu.: 605.0   3rd Qu.: 4.7432
##  Max.   :6445.0   Max.   :35682   Max.   :6082.0   Max.   :15.0001
##  NA's   :207
##  median_house_value ocean_proximity
##  Min.   : 14999     Length:20640
##  1st Qu.:119600     Class :character
##  Median :179700     Mode  :character
##  Mean   :206856
##  3rd Qu.:264725
##  Max.   :500001
##

#range of value and income
range(df$median_house_value)

## [1]  14999 500001

range(df$median_income)

## [1]  0.4999 15.0001

ggplot(data=df)+geom_histogram(mapping = aes(x=median_house_value))

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(df, aes(x=median_house_value, y=median_income))+
  geom_point()+
  stat_smooth()

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```




```
#Correlation:
cor(df$median_house_value, df$median_income)

## [1] 0.6880752

# Linear regression
lm1 <- lm(median_house_value~median_income, data=train)
pred <- predict(lm1, newdata=test)
cor_lm1 <- cor(pred, test$median_house_value)
mse_lm1 <- mean((pred - test$median_house_value)^2)

# SVM linear kernel
svm1 <- svm(median_house_value~., data=train, kernel="linear", cost=10,
scale=TRUE)
summary(svm1)

##
## Call:
## svm(formula = median_house_value ~ ., data = train, kernel = "linear",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  linear
##        cost:  10
##       gamma:  0.07692308
```

```
##     epsilon:  0.1
##
##
## Number of Support Vectors:  9922
```

```r
# Evaluate linear kernel svm
pred <- predict(svm1, newdata=test)
cor_svm1 <- cor(pred, test$median_house_value[(1:length(pred))])
mse_svm1 <- mean((pred - test$median_house_value[(1:length(pred))])^2)

#table(pred, test$ocean_proximity[(1:length(pred))])
#mean(pred==test$ocean_proximity[(1:length(pred))])
#plot(svm1, test, median_income ~ median_house_value)

# Tune the svm
tune_svm1 <- tune(svm, median_house_value~median_income, data=vald,
kernel="linear",
                  ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10)))
```

```
## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length
```

```
## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length
```

```
## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
```

```
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
```

```
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length
```

```
## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length

## Warning in pred - true.y: longer object length is not a multiple of
shorter
## object length
```

```
summary(tune_svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   cost
##  0.001
##
## - best performance: 13516180713
##
## - Detailed performance results:
##    cost       error dispersion
## 1 1e-03 13516180713 3167630515
## 2 1e-02 14618991604 3668035847
## 3 1e-01 14755060141 3721498741
## 4 1e+00 14768185264 3722444558
## 5 5e+00 14772729803 3725927367
## 6 1e+01 14773799700 3726770641
```

```
pred <- predict(tune_svm1$best.model, newdata=test)
cor_svm1_tune <- cor(pred, test$median_house_value[(1:length(pred))])
mse_svm1_tune <- mean((pred - test$median_house_value[(1:length(pred))])^2)

# SVM polynomial kernel

svm2 <- svm(median_house_value~., data=train, kernel="polynomial", cost=10,
scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = median_house_value ~ ., data = train, kernel = "polynomial",
##     cost = 10, scale = TRUE)
##
##
```

```
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##       gamma:  0.07692308
##      coef.0:  0
##     epsilon:  0.1
##
##
## Number of Support Vectors:  9260
```

```
# Evaluate the polynomial kernel

pred <- predict(svm2, newdata=test)
cor_svm2 <- cor(pred, test$median_house_value[(1:length(pred))])
mse_svm2 <- mean((pred - test$median_house_value[(1:length(pred))])^2)

#table(pred2, test$ocean_proximity[(1:length(pred2))])
#mean(pred2==test$ocean_proximity[(1:length(pred2))])
#plot(svm2, test, median_income ~ median_house_value)

# SVM radial kernel

svm3 <- svm(median_house_value~., data=train, kernel="radial", cost=10,
gamma=1, scale=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = median_house_value ~ ., 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:  9073
```

```
# Evaluate radial kernel
pred <- predict(svm3, newdata=test)
cor_svm3 <- cor(pred, test$median_house_value[(1:length(pred))])
mse_svm3 <- mean((pred - test$median_house_value[(1:length(pred))])^2)

#table(pred, test$median_house_value[(1:length(pre3))])
```

```r
#mean(pred==test$median_house_value[(1:length(pred))])
#plot(svm3, test, median_house_value ~ median_income)

# Tuning the hyperparameters
tune.out <- tune(svm, median_house_value~., data=vald, kernel="radial",
                 ranges=list(cost=c(0.1,1,10),
                             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
##     1   0.5
##
## - best performance: 3643851125
##
## - Detailed performance results:
##    cost gamma       error dispersion
## 1   0.1   0.5  5556095639  702450336
## 2   1.0   0.5  3643851125  401025212
## 3  10.0   0.5  3710001847  439752018
## 4   0.1   1.0  7288940427  921340109
## 5   1.0   1.0  4338342266  494865720
## 6  10.0   1.0  4515736157  390752698
## 7   0.1   2.0  9371781168 1053655868
## 8   1.0   2.0  5745424123  711741863
## 9  10.0   2.0  5692900032  474132066
## 10  0.1   3.0 10615120576 1088078083
## 11  1.0   3.0  6778754620  845277922
## 12 10.0   3.0  6555410666  618531944
## 13  0.1   4.0 11440269200 1098435926
## 14  1.0   4.0  7614682842  938718401
## 15 10.0   4.0  7295446133  734854825

# Radial kernel with various cost and gamma values
svm4 <- svm(median_house_value~., data=train, kernel = "radial", cost=100,
gamma=0.5, scale=TRUE)
summary(svm4)

##
## Call:
## svm(formula = median_house_value ~ ., 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:  9020
```

```r
# Evaluate Radial kernel with various cost/gamma values
pred <- predict(svm4, newdata=test)
cor_svm4 <- cor(pred, test$median_house_value[(1:length(pred))])
mse_svm4 <- mean((pred - test$median_house_value[(1:length(pred))])^2)

#table(pred, test$ocean_proximity[(1:length(pred))])
#mean(pred==test$ocean_proximity[(1:length(pred))])
#plot(svm4, test, median_income ~ median_house_value)

# correlation and mse for linear model
cor_lm1
```

```
## [1] 0.6980403
```

```r
mse_lm1
```

```
## [1] 7052106439
```

```r
# svm1 through svm4 correlation and mse
cor_svm1
```

```
## [1] 0.3748647
```

```r
mse_svm1
```

```
## [1] 14472254737
```

```r
cor_svm2
```

```
## [1] 0.114215
```

```r
mse_svm2
```

```
## [1] 98206193678
```

```r
cor_svm3
```

```
## [1] 0.4225706
```

```r
mse_svm3
```

```
## [1] 14317898140
```

```r
cor_svm4
```

```
## [1] 0.4202958
```

```r
mse_svm4
## [1] 15123977772
#Overall, the linear model had the best correlation value, I believe this is
because the data is more linear and therefore attempts to polynomialize the
data would make it worse.
```