Regression and smoothing

Dr. Justin Wishart

Load the data

We will be using a dataset of Melbourne house prices. Let's first load the data and see what is in the table. This dataset was downloaded from Kaggle and the data was released under the CC BY-NC-SA 4.0 license.

```
melbdata <- read.csv("Melbourne_housing_FULL.csv", header = TRUE)
dim(melbdata)
```

```
## [1] 34857    21
```

```
colnames(melbdata)
```

```
##  [1] "Suburb"        "Address"       "Rooms"         "Type"
##  [5] "Price"         "Method"        "SellerG"       "Date"
##  [9] "Distance"      "Postcode"      "Bedroom2"      "Bathroom"
## [13] "Car"           "Landsize"      "BuildingArea"  "YearBuilt"
## [17] "CouncilArea"   "Lattitude"     "Longtitude"    "Regionname"
## [21] "Propertycount"
```

Now let's subset to only look at house prices in St. Kilda (and remove cases that have the price variable missing).

```
st.kilda <- subset(melbdata, Suburb == "St Kilda")
dim(st.kilda)
```

```
## [1] 374  21
```

```
sum(is.na(st.kilda$Price))
```

```
## [1] 71
```

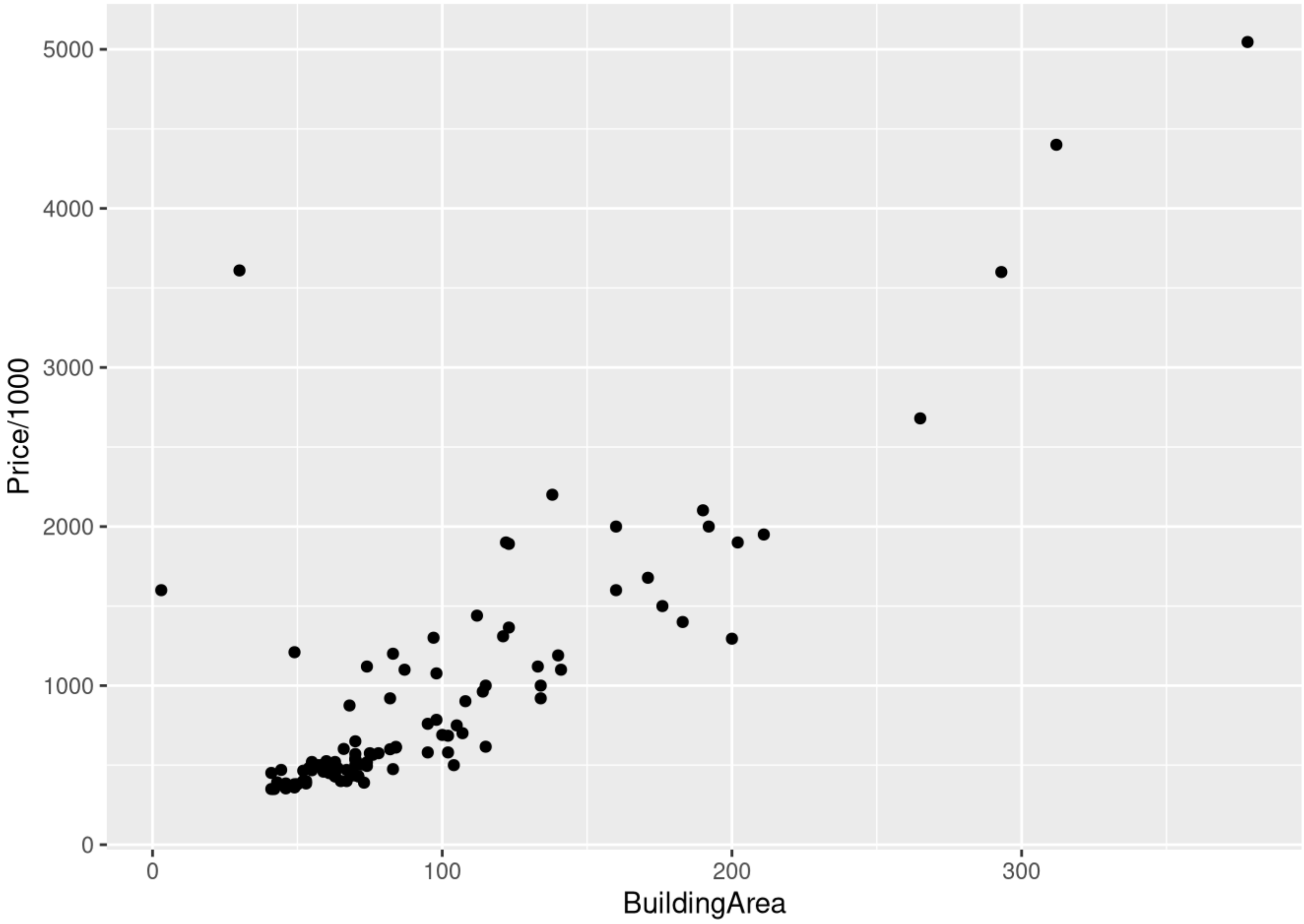
```
st.kilda <- st.kilda[!is.na(st.kilda$Price), ]
dim(st.kilda)
```

```
## [1] 303  21
```

Let's plot this using the `ggplot2` package.

```
library(ggplot2)
ggplot(st.kilda, aes(x = BuildingArea, y = Price/1000)) + geom_point()
```

```
## Warning: Removed 202 rows containing missing values (geom_point).
```



First regression model

```
lmfit <- lm(Price ~ BuildingArea, data = st.kilda)
summary(lmfit)
```

```
##
## Call:
## lm(formula = Price ~ BuildingArea, data = st.kilda)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -817415 -201614  -85181   19895 3403199
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -129484.0    91775.9  -1.411    0.161
## BuildingArea   11209.5      799.8  14.015   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 490300 on 99 degrees of freedom
##   (202 observations deleted due to missingness)
## Multiple R-squared:  0.6649, Adjusted R-squared:  0.6615
## F-statistic: 196.4 on 1 and 99 DF,  p-value: < 2.2e-16
```

```
new <- data.frame(BuildingArea = 100)
predict(lmfit, new, interval = "confidence")
```

```
##        fit      lwr     upr
## 1 991465.5 894562.7 1088368
```

Remove outliers

To improve our regression model, let’s try to remove the two points that look like outliers (and expensive places) and do the `lm()` fit again.

```
# Below the | operator does elementwise OR to check either of two conditions
outliers <- which(st.kilda$BuildingArea < 40 | st.kilda$Price > 1000000)
st.kilda.sub <- st.kilda[-outliers,]
lmfit2 <- lm(Price ~ BuildingArea, data = st.kilda.sub)
summary(lmfit2)
```

```
##
## Call:
## lm(formula = Price ~ BuildingArea, data = st.kilda.sub)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -237692  -42042   -5300   41304  356235
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  105236.3    39221.4   2.683  0.00915 **
## BuildingArea   6081.3      519.1  11.714  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 97560 on 68 degrees of freedom
##   (163 observations deleted due to missingness)
## Multiple R-squared:  0.6686, Adjusted R-squared:  0.6638
## F-statistic: 137.2 on 1 and 68 DF,  p-value: < 2.2e-16
```

Multivariate regression

Now let’s add building type as a predictor in our model and see if it improves the prediction.

```
lmfit3 <- lm(Price ~ Type + BuildingArea, data = st.kilda.sub)
summary(lmfit3)
```

```
##
## Call:
## lm(formula = Price ~ Type + BuildingArea, data = st.kilda.sub)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -198007  -45267   -8502   42380  244298
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   380647.9    65467.0   5.814 1.86e-07 ***
## Typeu        -228120.2    46402.8  -4.916 6.00e-06 ***
## BuildingArea    5245.0      479.5  10.937  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 84250 on 67 degrees of freedom
##   (163 observations deleted due to missingness)
## Multiple R-squared:  0.7565, Adjusted R-squared:  0.7492
## F-statistic: 104.1 on 2 and 67 DF,  p-value: < 2.2e-16
```

```
new <- data.frame(BuildingArea = c(100,100), Type = c("u", "h")) # Cant predict townh
ouse since none exist in original data after filtering.
predict(lmfit3, new)
```

```
##        1        2
## 677027.3 905147.5
```

Spline smoothing

```
st.kilda.noNA <- subset(st.kilda.sub, !is.na(BuildingArea) & !is.na(Price))
dim(st.kilda.noNA)
```

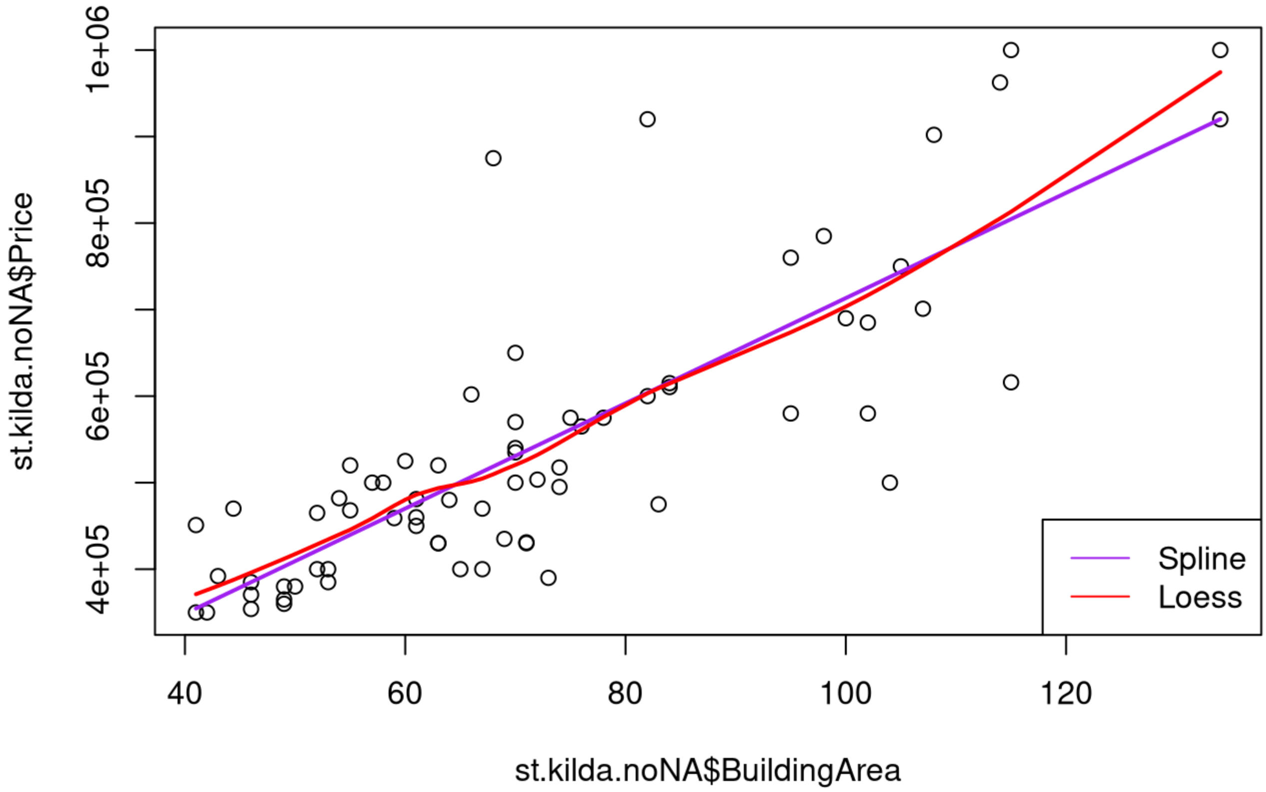
```
## [1] 70 21
```

```
spline.fit <- smooth.spline(x=st.kilda.noNA$BuildingArea, y = st.kilda.noNA$Price)
# Or use with
spline.fit2 <- with(st.kilda.noNA, smooth.spline(x = BuildingArea, y = Price))
```

Loess smoothing

```
lo1.fit <- loess(Price ~ BuildingArea, data = st.kilda.noNA)

plot(st.kilda.noNA$BuildingArea, st.kilda.noNA$Price)
lines(spline.fit, col = "purple", lwd =2)
lines(sort(st.kilda.noNA$BuildingArea), predict(lo1.fit, sort(st.kilda.noNA$BuildingA
rea)), col = "red", lwd =2)
legend("bottomright",c("Spline","Loess"),
      lty=c(1,1), col=c("purple", "red"))
```




```
# We can also use the loess fit to do prediction
predict(lo1.fit, 100)
```

```
## [1] 703532.9
```