Data Analyses and Study of Melting Point Data

Andrew Wang

9/22/2020

Introduction

In this report, we are analyzing a set of melting-point data, from the article “General Melting Point Prediction Based on a Diverse Compound Data Set and Artificial Neural Networks” by M. Karthikeyan. In this section, I am going to provide some background information on melting points, while also going over information, processes, and conclusions from Karthikeyan’s article. After that, I will analyze the data using R.

To start off with, the melting point is the temperature at which a compound will transition from a solid to a liquid state. The melting point of a compound is very important, and is controlled by both single molecules and molecular interactions. The melting point is important because it can be used to determine a variety of other data, such as purity, identity, boiling point, and solubility - which is very important in the pharmaceutical industry. Melting points can be estimated through the Molecular Similarity Principle, the idea that structurally similar molecules have similar properties. However, melting point is harder to predict than boiling point, due to solid-state properties, the unique properties that compounds in a solid state have. In general, melting point depends on a combination of molecular properties and molecule arrangement in a solid state.

It had been found in previous research that factors like hydrophilic and polar surface area, along with ring structures which provide increased rigidity, increase the melting point, while the opposite, nonpolar surface area and flexibility, decreases melting point. However, earlier research used a smaller dataset, with a different prediction model. This prediction model, known as a “Group Vector Space” method, was used to predict boiling points of various hydrocarbons. However, the new prediction model used by Karthikeyan differed in 3 key ways:

1. It has a larger and more diverse dataset. This simply means that the dataset used by Karthikeyan was larger than those used in previous studies.

2. Feature selection through principal component analysis. This study only selected components with large Eigenvalues (the factor by which value changes when a scalar is applied to it), so that the model would not be overwhelmed with many descriptors.

3. It used a neural network for model building. Before, scientists had used linear techniques, which would lead to misleading data in modeling nonlinear relationships.

Finally, training (2087), test (1043), and validation (1043) sets were selected, which will be used to training the neural network to make accurate predictions.

Methods and model construction

4173 structures were used in the training and development of the neural network, but structures with melting point variations of larger than 5 degrees Celsius or those containing heavy metals were excluded from the analysis. The table below breaks down some statistical data regarding the properties of the data.

mp	heavy.atoms	molar.mass	refractivity	logP	dipole.moment
Min. : 14.0	Min. : 6.00	Min. : 84.08	Min. : 1.992	Min. :-6.023	Min. : 0.000
1st Qu.:117.5	1st Qu.:17.00	1st Qu.:243.27	1st Qu.: 6.609	1st Qu.: 2.106	1st Qu.: 2.375
Median :161.5	Median :22.00	Median :308.33	Median : 8.310	Median : 3.221	Median : 3.811
Mean :165.3	Mean :22.24	Mean :317.41	Mean : 8.514	Mean : 3.337	Mean : 5.210
3rd Qu.:209.4	3rd Qu.:26.00	3rd Qu.:375.38	3rd Qu.:10.179	3rd Qu.: 4.529	3rd Qu.: 5.633
Max. :392.5	Max. :59.00	Max. :815.62	Max. :19.354	Max. :12.780	Max. :248.303

After this, 2D and 3D descriptors were made from the optimized structures of the external validation group. 2D descriptors included things like charge, molecular refractivity, van der Waals volume, and many others. 3D descriptors included potential energy, surface area, volume, shape descriptors, hence the name “3D”.

Finally models were constructed for 2D descriptors, 3D descriptors, and a combination of both 2D and 3D descriptors. All the analysis was conducted by R (to be covered later). The neural network was fed the training, test, and validation data (with more neural networking specifics in the article) and gave results for all three models.

Results, discussion, and conclusion

To determine the variables which caused the most variation in melting point, Karthikeyan used PCA, a mutlivariate data analysis technique to convert observation into linear variables. Based on the results from the PCA, below are the top three components for melting point variation.

Component 1: Responsible for 32.61% of variation. It is defined by size of the molecule, positive charge, polarizability on one side. On the other, it is defined by molecular and electronic energy. This corresponded with the concept that of smaller and less polarizable molecules as “hard”, and larger and more polarizable molecules as “soft”. Furthermore, this reinforced the idea that molecule size has a large impact on melting point.

Component 2: Responsible for 12.81% of variation. It is defined by polar and negative surface areas on one side. On the other, it is defined by fractional hydrophobic water accessible surface area. This corresponds with the belief that hydro/lipo-philic compounds are important in the prediction of melting points.

Component 3: Responsible for 7.61% of variation. It is defined by positively and negatively charged surface area on both sides.

This essentially means that the three variables with the largest impact on melting point are size, polarity, and surface area charge. Jointly, they are responsible for 53.03% of melting point variation. While it was expected that the 2D and 3D descriptors would yield similar results, this was not the case. 2D descriptors gave better results than their 3D counterparts, as 3D descriptors did not get as many relevant factors to melting point prediction as 2D descriptors did. 2D descriptors gave the best results for factors like correlation coefficient and absolute mean error.

However, there were kinds of compounds which the model struggeled to predict the melting point of. Among those are nonaromatic steroids, special small molecules, and special large molecules. Nonaromatic steroids, which are rigid and recieve a higher melting point prediction because of that, were often overestimated, the cause of which is still unknown. For small molecules, those which self-organized and whose bonds are stronger than average for their size got inaccurate melting point predictions. Finally, large molecules whose interactions were not on par with those of their physiochemically similar neighbors also received inaccurate melting point predictions.

To conclude, the model made by Karthikeyan has greater applicability, due to its input of a larger dataset. However, it still contains outliers, the two main reasons for outliers being intermolecular interactions and intramolecular interactions. Intermolecular interactions are a result of the arrangement of compounds in the solid state, which was not taken into account by single-molecule descriptors. Intramolecular interactions were underestimated by the model, as a more flexible compound is assumed to have a lower melting point.

Analysis with R

Data cleanup and setup

In this section, we are going over various functions and methods for cleaning data in R, which will allow us to better perform analysis later on it. First, we need to clean up and set up the R environment with the working directory. R script for reference later.

```
# clean up and setup
rm(list=ls()) # clean up any old stuff in R
setwd("C:/Users/hyper/OneDrive/Desktop/Desktop Folders/Programming/R/Assignments/Week
 4") # go to this folder
```

Note that your directory structure will be different than mine, so don’t directly copy the code here.

Load the library

Next, we want to load the library we’ll be using to clean and prepare the data. The library name is `dplyr` . However, you will need to install the library first with the command `install.packages("dplyr")` . After this, you can import the library with `library(dplyr)` .

```
library(dplyr)
```

Importing the csv file

After installing the library, it is time to import the csv file. First, make sure your csv file is in the same location as your specified working directory (defined above). This will make sure R knows where to go to find the csv file. After this, we run a few initial viewing functions to look at basic attributes of the dataframe, such as names and structure (str). I have commented out the head() and tail() function calls here, as I felt they made the report rather long, while not adding anything of large substance.

```
#input csv file
chemData <- read.csv("dirtyMPdata.csv")

#basic high-level structure
View(chemData)
str(chemData)
```

```
## 'data.frame':    4450 obs. of  18 variables:
##  $ ï..structure : chr  "O=C1Cc2ccccc21" "Clc1ccc(cc1)C1c2c(OC(N)=C1C#N)[nH][nH0]c2C
(F)(F)F" "O=C(OC)C(=Cc1ccccc1)Cc1ccccc1" "FC(F)(F)c1[nH0]cc2ccccc2c1" ...
##  $ mp           : num  14 20.5 27.5 30.5 31 31.5 32 32.5 33 34 ...
##  $ rings        : int  6 11 12 10 6 6 6 5 12 12 ...
##  $ heavy.atoms  : int  9 23 19 14 12 14 12 10 16 14 ...
##  $ single.bonds : int  9 20 22 10 16 19 14 12 21 16 ...
##  $ triple.bonds : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ reactive     : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ molar.mass   : num  118 341 252 197 162 ...
##  $ refractivity : num  3.56 7.73 7.8 4.73 4.65 ...
##  $ formal.charge: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ logP         : num  1.43 3.64 3.49 3.57 1.94 ...
##  $ tpsa         : num  17.1 87.7 26.3 12.9 26.3 ...
##  $ dipole.moment: num  2.72 6.32 1.91 5.2 1.76 ...
##  $ energy       : num  -57.8 -189.1 -123.5 -119.7 -83.5 ...
##  $ density      : num  0.959 1.296 0.946 1.195 0.978 ...
##  $ volume       : num  123 263 267 165 166 ...
##  $ PCA1         : num  15.55 1.92 2.98 11.03 10.97 ...
##  $ PCA2         : num  1.622 0.502 2.723 1.826 -0.905 ...
```

```
#head(chemData)
#tail(chemData)
summary(chemData)
```

```
##  ï..structure             mp             rings         heavy.atoms
##  Length:4450        Min.   : 14.0   Min.   : 0.000   Min.   : 6.00
##  Class :character   1st Qu.:117.5   1st Qu.: 6.000   1st Qu.:17.00
##  Mode  :character   Median :161.5   Median :11.000   Median :22.00
##                     Mean   :165.3   Mean   : 9.956   Mean   :22.24
##                     3rd Qu.:209.4   3rd Qu.:12.000   3rd Qu.:26.00
##                     Max.   :392.5   Max.   :36.000   Max.   :59.00
##
##   single.bonds     triple.bonds       reactive        molar.mass
##  Min.   :  4.00   Min.   :0.0000   Min.   :0.0000   Min.   : 84.08
##  1st Qu.: 19.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:243.27
##  Median : 24.00   Median :0.0000   Median :0.0000   Median :308.33
##  Mean   : 28.24   Mean   :0.1058   Mean   :0.2209   Mean   :317.41
##  3rd Qu.: 33.00   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:375.38
##  Max.   :119.00   Max.   :4.0000   Max.   :1.0000   Max.   :815.62
##
##   refractivity    formal.charge           logP             tpsa
##  Min.   : 1.992   Min.   :-3.00000   Min.   :-6.023   Min.   :  0.00
##  1st Qu.: 6.609   1st Qu.: 0.00000   1st Qu.: 2.106   1st Qu.: 38.33
##  Median : 8.310   Median : 0.00000   Median : 3.221   Median : 61.07
##  Mean   : 8.514   Mean   : 0.01236   Mean   : 3.337   Mean   : 67.50
##  3rd Qu.:10.179   3rd Qu.: 0.00000   3rd Qu.: 4.529   3rd Qu.: 88.06
##  Max.   :19.354   Max.   : 2.00000   Max.   :12.780   Max.   :413.24
##
##  dipole.moment         energy           density            volume
##  Min.   :  0.000   Min.   :-489.00   Min.   :0.7945   Min.   : 78.89
##  1st Qu.:  2.375   1st Qu.:-186.18   1st Qu.:0.9812   1st Qu.:230.77
##  Median :  3.811   Median :-150.42   Median :1.0540   Median :285.19
##  Mean   :  5.210   Mean   :-157.51   Mean   :1.0804   Mean   :294.47
##  3rd Qu.:  5.633   3rd Qu.:-118.36   3rd Qu.:1.1432   3rd Qu.:349.31
##  Max.   :248.303   Max.   : -44.35   Max.   :1.8956   Max.   :728.58
##                    NA's   :13
##       PCA1               PCA2
##  Min.   :-33.7975   Min.   :-19.9754
##  1st Qu.: -4.1177   1st Qu.: -2.8547
##  Median :  0.6655   Median :  0.3189
##  Mean   :  0.0000   Mean   :  0.0000
##  3rd Qu.:  4.6769   3rd Qu.:  3.2613
##  Max.   : 17.3486   Max.   : 12.4899
##
```

Fixing missing data values

One of the most, if not the most, important steps in data cleaning and preparation is the make sure you fix the missing values. While this can be done through deletion, it will remove lots of data and is generally not considered good practice. Instead, what we can do is make the missing data points the average of all other nonmissing data points. By doing this, we can keep all data rows (observations), while also keep any analysis we do mostly consistent. Below, I first do some printing to find where my empty values are, and I discover that they are all in the energy column. This makes my data cleaning much easier, as now I only need to clean up one column. After this, I check againt to make sure there are no more empty values.

```
#cleaning data only one column with any missing data
print(sum(is.na(chemData))) #debug - check out much missing data there is
```

```
## [1] 13
```

```
print(sum(is.na(chemData$energy))) #debug - check where it is
```

```
## [1] 13
```

```
chemData$energy <- ifelse(is.na(chemData$energy),mean(chemData$energy,na.rm = TRUE),che
mData$energy)
print(sum(is.na(chemData))) #debug - make sure no more data is missing
```

```
## [1] 0
```

Renaming the columns

Below, I rename a few columns which I thought the names of which were not clear. I used the `rename` function of `dplyr` . After this, I print out the names again to make sure the names have changed.

```
#rename a few columns
chemData <- rename(chemData, structure = ï..structure, melting.point = mp)
names(chemData)
```

```
##  [1] "structure"     "melting.point" "rings"         "heavy.atoms"  
##  [5] "single.bonds"  "triple.bonds"  "reactive"      "molar.mass"   
##  [9] "refractivity"  "formal.charge" "logP"          "tpsa"         
## [13] "dipole.moment" "energy"        "density"       "volume"       
## [17] "PCA1"          "PCA2"
```

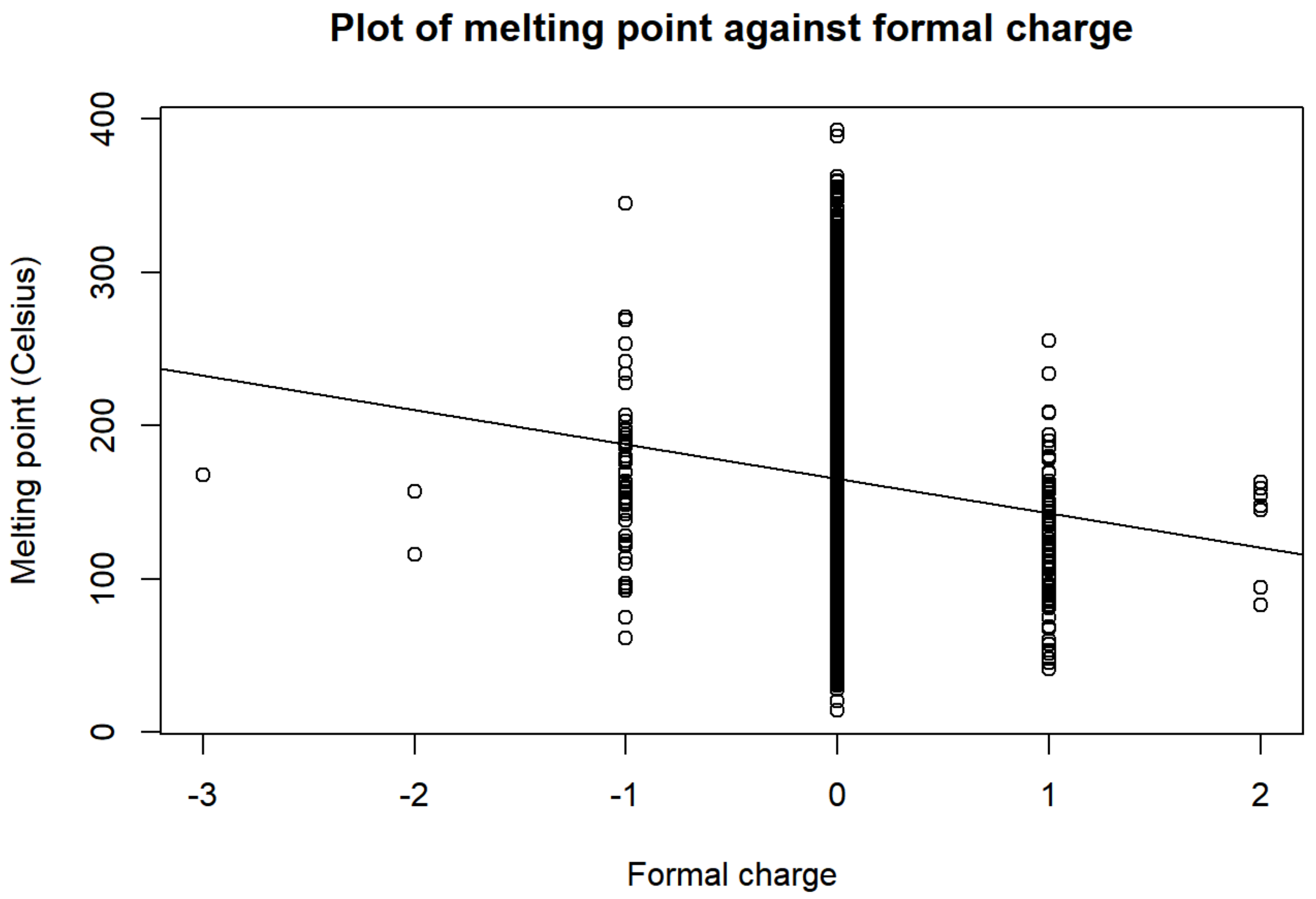
Linear Regressions

In this next portion, I conduct a linear regression on the formal charge, the volume, and the molar refractivity with regard to the melting point. The code below is split up into three portions, and they all follow essentially the same format. First, I create a variable to store a linear regression model relating melting point to one of the aforementioned variables. After that, I print it out. Next, I plot the two variables, and give them appropriate names. Finally, I used the existing linear regression model to plot a regression line.

```
#linear regressions
FCMP <- lm(chemData$melting.point~chemData$formal.charge) #create regression model
FCMP #print regression model
```

```
##
## Call:
## lm(formula = chemData$melting.point ~ chemData$formal.charge)
##
## Coefficients:
##            (Intercept)  chemData$formal.charge
##                 165.57                  -22.46
```

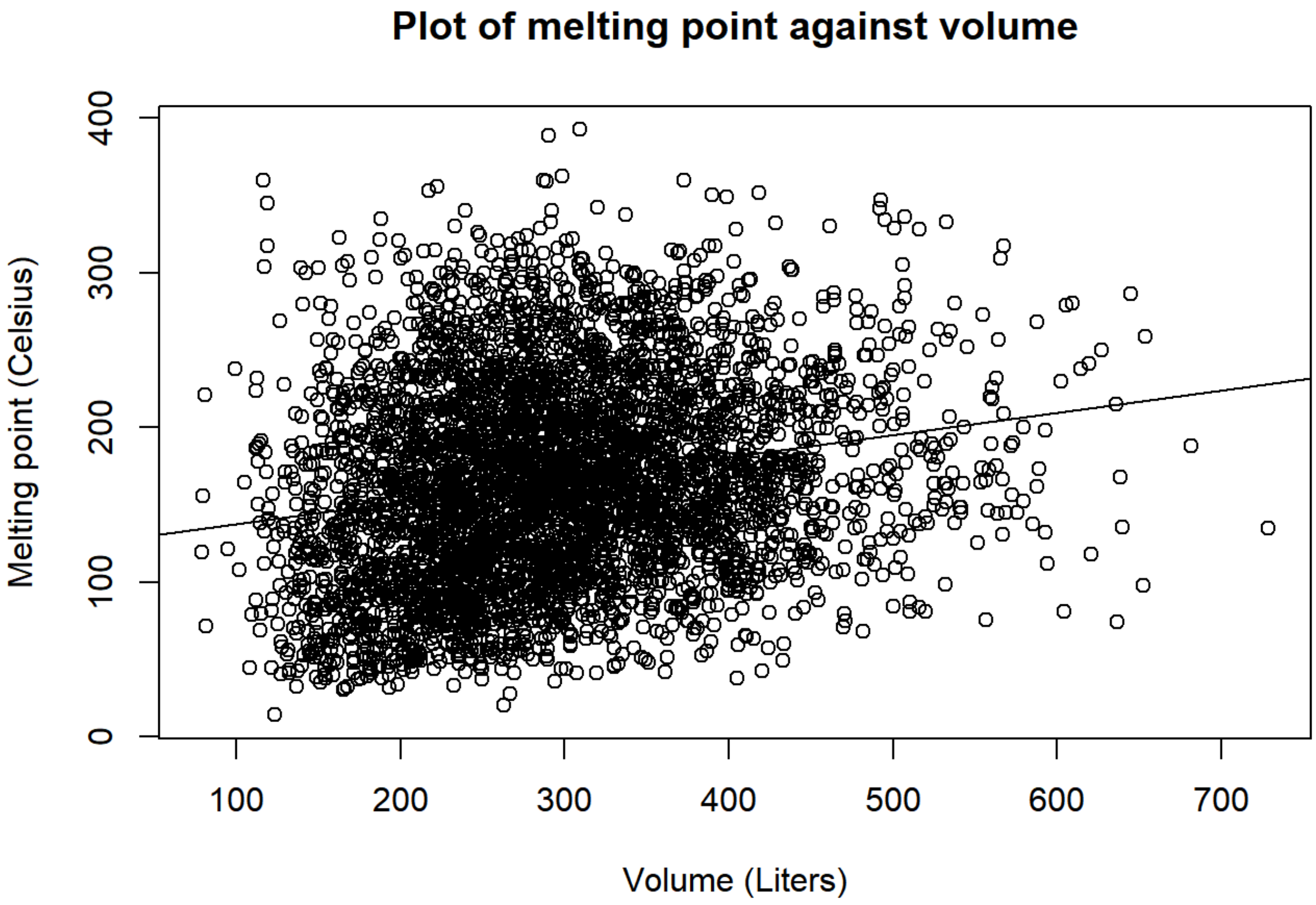
```
plot(chemData$melting.point~chemData$formal.charge, main = "Plot of melting point against formal charge", xlab = "Formal charge", ylab = "Melting point (Celsius)") #plot the two variables
abline(FCMP) #add regression line
```



```
VMP <- lm(chemData$melting.point~chemData$volume)
VMP
```

```
##
## Call:
## lm(formula = chemData$melting.point ~ chemData$volume)
##
## Coefficients:
##     (Intercept)  chemData$volume
##        122.9602           0.1437
```

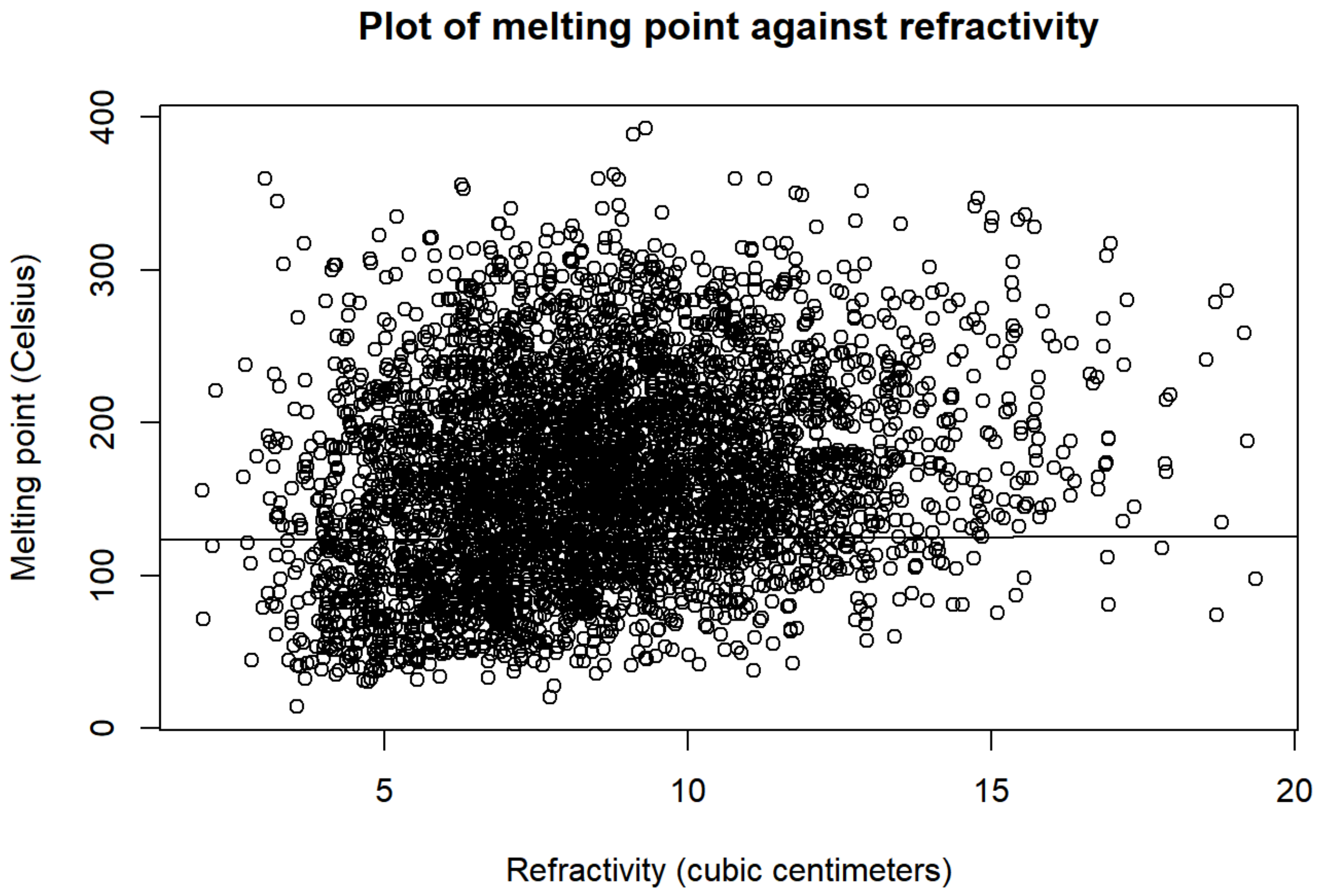
```
plot(chemData$melting.point~chemData$volume, main = "Plot of melting point against volu
me", xlab = "Volume (Liters)", ylab = "Melting point (Celsius)")
abline(VMP)
```



```
MRMP <- lm(chemData$melting.point~chemData$refractivity)
MRMP
```

```
##
## Call:
## lm(formula = chemData$melting.point ~ chemData$refractivity)
##
## Coefficients:
##           (Intercept)  chemData$refractivity
##               117.474                  5.616
```

```
plot(chemData$melting.point~chemData$refractivity, main = "Plot of melting point agains
t refractivity", xlab = "Refractivity (cubic centimeters)", ylab = "Melting point (Cels
ius)")
abline(VMP)
```



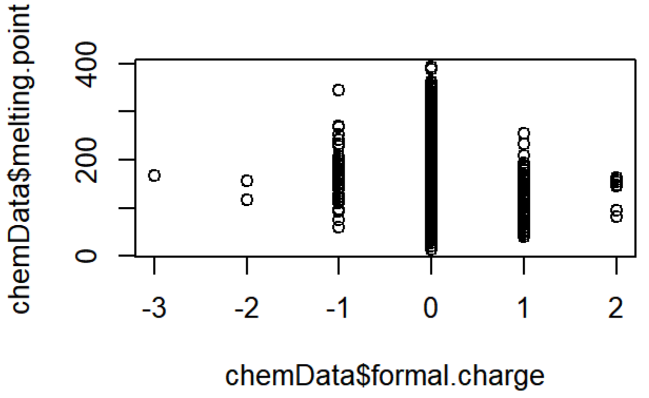
Multi-regression plotting

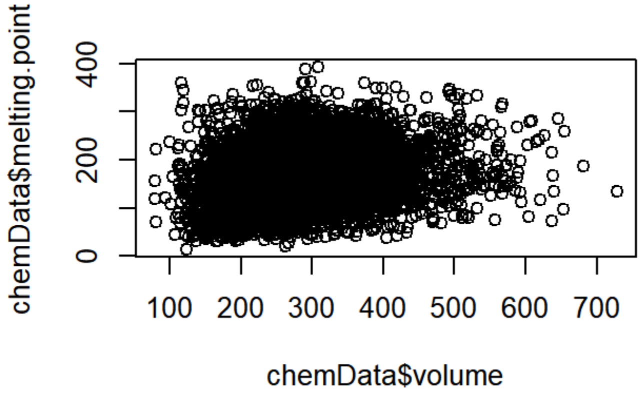
In this next part, I conduct a multiple linear regression, combining charge, volume and refractivity as three predictors for melting point. The code below essentially reflects the same code format as used for single-regression. First, I create a variable to store the regression. Next, I print it out. However, here I set up a new graphics grid so that I can store the 3 plots at one place instead of having 3 different plots. After setting up the new graphics grid, I graph my regression. Finally, I reset the graphics grid so that future graphs will be normal.

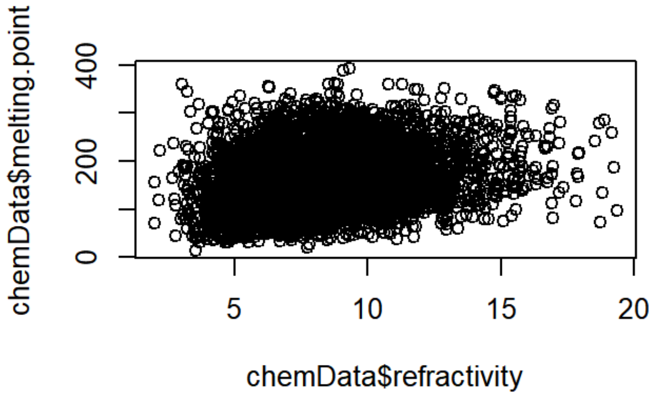
```
# multi-linear regression
multiMP <- lm(chemData$melting.point~chemData$formal.charge+chemData$volume+chemData$re
fractivity)
multiMP
```

```
##
## Call:
## lm(formula = chemData$melting.point ~ chemData$formal.charge +
##     chemData$volume + chemData$refractivity)
##
## Coefficients:
##            (Intercept)  chemData$formal.charge         chemData$volume
##               123.0117                -22.7805                 -0.7187
##  chemData$refractivity
##                29.8553
```

```
par(mfrow = c(2,2)) #setting up a graphics grid
plot(chemData$melting.point~(chemData$formal.charge+chemData$volume+chemData$refractivi
ty))
par(mfrow = c(1,1)) #reset graphics grid
```







BIC analysis

In this final portion of the code, I conduct a BIC, or Bayesian information criterion, analysis to look at the correlative relationship between volume and melting point. I want to determine to what degree a change in volume wil be reflected in a change in melting point. First, I set up a BIC analysis to compare the melting point to 1. This is important because it gives me a stable base for future comparisions. Next, I print out the value. After this I set up my second BIC analysis, which compares volume to melting point. I then print it out. Finally, I want to get the difference between these two values. While I could do the mental math to figure out how much larger the volume BIC is than the baseline BIC, I it is expressed more clearly if I print out the difference, which is what I do in the final line. The difference, 181, is very large, and this shows a close relationship between volume and melting point.

```
#BIC analysis volume -> melting point
baseMP <- BIC(lm(chemData$melting.point~1))
baseMP
```

```
## [1] 49674.4
```

```
volumeMP <- BIC(lm(chemData$melting.point~chemData$volume))
volumeMP
```

```
## [1] 49493.12
```

```
print(abs(volumeMP-baseMP))
```

```
## [1] 181.2809
```