Week 5 Project: IS607

Alexander Satz

October 3, 2014

A data set was downloaded from https://www.ebi.ac.uk/chembl/sarfari/kinasesarfari/. This data set contains 30,016 potency measurements of compouds against various protein kinases. The data set derives from thousand of publications and Chembl internal sources.

I sought to answer the following question: Do compounds with high potentcy in some targets end up being assayed more often, i.e. are 'potent' cmpds followed-up?

1. Open the file

```
> library("dplyr")
> library("tidyr")
> library("ggvis")
> file <- read.table(file = "/Users/alexandersatz/Documents/Cuny/IS607/week5/ks_bioactivity.
> bioact.1 <-tbl_df(file)
```

2. As shown below, there are 1447 types of measurements. This needs to be cleaned up before we can classify potency. Some values are log, some -log and some are not logs at all. Each row is an observation, and so there is no need for 'pivoting' etcetera.

```
> summarise(bioact.1,
+           numberdatatype = n_distinct(ACTIVITY_TYPE))

Source: local data frame [1 x 1]

  numberdatatype
1           1447

> activity.type <- group_by(bioact.1, ACTIVITY_TYPE)
> types.df <-summarise(activity.type,
+           number = n())
> types.df

Source: local data frame [1,447 x 2]

   ACTIVITY_TYPE number
```

```
1   -Delta G obs      2
2     -Log alpha      1
3         -Log C      7
4      -Log EC50     21
5      -Log IC25      1
6      -Log IC50      7
7   -Log IC50(M)      4
8        -Log KB      1
9        -Log KD      3
10     -Log KD50      4
..           ...    ...
```

Additionaly there are numerious scales being used including nM and uM, stated in both lower and uppercase (see below). Before all values can be converted to log units, this will need to be standardized.

```
> activity.units <- group_by(bioact.1, STANDARD_UNIT)
> units.df <-summarise(activity.units,
+                      number = n())
> units.df <- data.frame(units.df)
> head(units.df)
```

```
                                   STANDARD_UNIT number
1                                                 47944
2                               (mg of CPT) kg-1      1
3      (nM of XMP formed) hr-1 (mg of protein)-1      2
4                              (ug of base) ml-1      4
5 (ug of cross-linked protein) (mg of protein)-1      3
6                                            /hr      3
```

First we tackle those values already present on a log10 scale. We run a grepl search for 'log'. The result includes nonsensical outliers and measurments such as 'logD' and 'logP' which are not activity measurements. These rows are removed by additional filter() using text matching. Lastly, the Log value is converted to an integer because we want there to be a limited number of potency 'levels'. The data frame still has many columns as we haven't decided what to get rid of yet. NA values also exist. NA values may derive from assays where a 'value' for that pariticular compound could not be calculated. These values should not be in the database. I will remove them later.

```
> bioact.log <- filter(bioact.1, grepl("log", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.log <- filter(bioact.log, ! grepl("log2", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.log <- filter(bioact.log, ! grepl("logp", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.log <- filter(bioact.log, ! grepl("logd", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.log <- filter(bioact.log, ! grepl("GI50", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.logged1 <- mutate(bioact.log,
+                          LOG.ACT = as.integer(abs(STANDARD_VALUE)))
```

```
> bioact.logged1 <-arrange(bioact.logged1, desc(LOG.ACT))
> bioact.na <- (bioact.logged1[is.na(bioact.logged1$LOG.ACT),])
> bioact.logged1

Source: local data frame [2,317 x 14]

   ACTIVITY_ID DOM_ID                NAME ASSAY_TYPE COMPOUND_ID ACTIVITY_TYPE
1      2582768   1553          hEGFR_1553          B       35820      Log IC50
2      2711279     NA      Starlite ADMET          A      374400       log KOA
3      2582767   1553          hEGFR_1553          B      328106      Log IC50
4      2383811   1553          hEGFR_1553          B      271122      Log IC50
5      2383814   1553          hEGFR_1553          B      299622      Log IC50
6      2383291   1553          hEGFR_1553          B      294475      Log IC50
7      2383328   1553          hEGFR_1553          B      328216      Log IC50
8      2437590     NA      Starlite ADMET          A      229760       log KOA
9      2436949     NA Starlite Functional          F      464859      Log EC50
10     2446933     NA      Starlite ADMET          A         415        Log k'
..         ...    ...                 ...        ...         ...           ...
Variables not shown: RELATION (fctr), STANDARD_VALUE (dbl), STANDARD_UNIT
  (fctr), ACTIVITY_COMMENT (fctr), CHEMBL_ACTIVITY_ID (int), CHEMBL_ASSAY_ID
  (int), PUBMED_ID (int), LOG.ACT (int)

>
```

Next I need to deal with 'values' measured not on a log scale. These can have either nM or uM scales. First I pull out all values that are NOT 'log' values via a grepl match, then I divide this data into those that are on the nM and uM scales.

```
> bioact.notlog <-filter(bioact.1, ! agrepl("log", ACTIVITY_TYPE, ignore.case = TRUE))
> bioact.loguM <-bioact.notlog %>%
+   filter(grepl("um", STANDARD_UNIT, ignore.case = TRUE)) %>%
+   mutate(LOG.ACT = as.integer(-log10((STANDARD_VALUE/1000000))))
> bioact.loguM <-arrange(bioact.loguM, desc(LOG.ACT))
> #bioact.loguM$LOG.ACT  ## looks great and values range from 4-6 mainly, so the right range
>
> bioact.lognM <-bioact.notlog %>%
+   filter(grepl("nm", STANDARD_UNIT, ignore.case = TRUE)) %>%
+   mutate(LOG.ACT = as.integer(-log10((STANDARD_VALUE/1000000000))))
> bioact.lognM <-arrange(bioact.lognM, desc(LOG.ACT))
> #bioact.lognM$LOG.ACT  ## looks good
```

Now combine the 3 dataframes. I have 11000 entries. The final product can be inspected to see that the calculated value LOG.ACT matches the expected value! Last, NA values are removed as we know from above inspection that they are 'garbage' in the data set.

```
> biact2 <- rbind(bioact.logged1, bioact.loguM, bioact.lognM)
> biact2 <- select(biact2, COMPOUND_ID, STANDARD_UNIT, STANDARD_VALUE, LOG.ACT, DOM_ID, NAME
> biact2 <-arrange(biact2, desc(LOG.ACT))
> head(biact2, 10)

Source: local data frame [10 x 6]

   COMPOUND_ID STANDARD_UNIT STANDARD_VALUE LOG.ACT DOM_ID                NAME
1           59            nM       8.00e-06      14     NA Starlite Functional
2      1532440            uM       4.07e-07      12     NA Starlite Functional
3      1377737            uM       1.36e-07      12     NA Starlite Functional
4      1494120            uM       4.07e-07      12     NA Starlite Functional
5      1458022            uM       4.07e-07      12     NA Starlite Functional
6       216933            nM       8.80e-04      12   1950          hABL1_1950
7       408247            nM       7.30e-04      12   1950          hABL1_1950
8       223228            nM       2.50e-04      12     NA Starlite Functional
9       223228            nM       3.00e-04      12     NA Starlite Functional
10      264189            nM       6.60e-04      12   1950          hABL1_1950

> biact3 <- biact2[!is.na(biact2$LOG.ACT),]
```

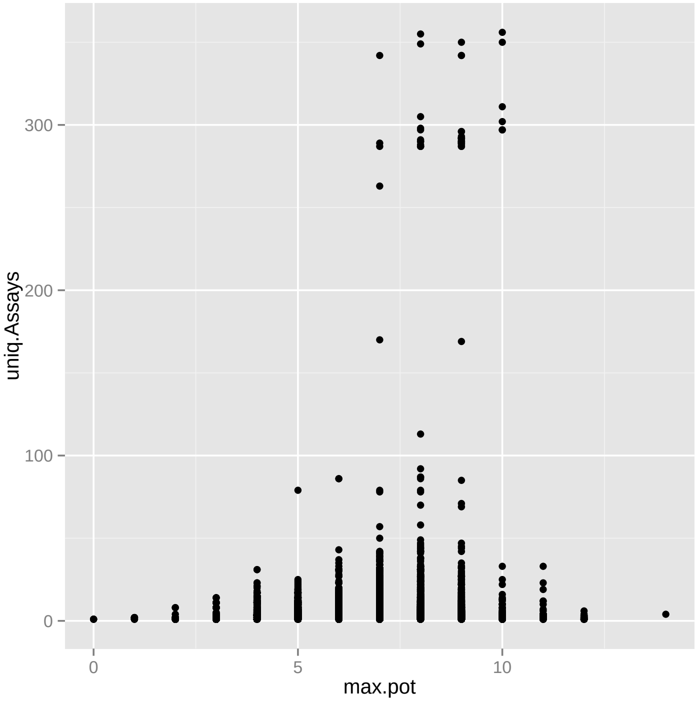
3. Now we group by 'compound id' and determine the max potency of each cmpd in any assay.

```
> biact.cmp <- group_by(biact3, COMPOUND_ID)
> bioact.sum <- summarize(biact.cmp,
+                         max.pot = max(LOG.ACT),
+                         tot.Assays = n(),
+                         uniq.Assays = n_distinct(NAME)
+                         )
> bioact.sum <-arrange(bioact.sum, desc(max.pot))
> bioact.sum

Source: local data frame [49,475 x 4]

   COMPOUND_ID max.pot tot.Assays uniq.Assays
1           59      14         83           4
2           76      12       1087           3
3          129      12       1311           4
4          135      12        125           2
5          633      12         73           3
6         1563      12        142           3
7       216933      12          2           1
8       223228      12        687           3
9       264189      12          5           1
10      406721      12          2           1
..         ...     ...        ...         ...
```

4. We plot potency of each cmpd (its max potency in any assay) versus number of the unique assays the compound was run in.

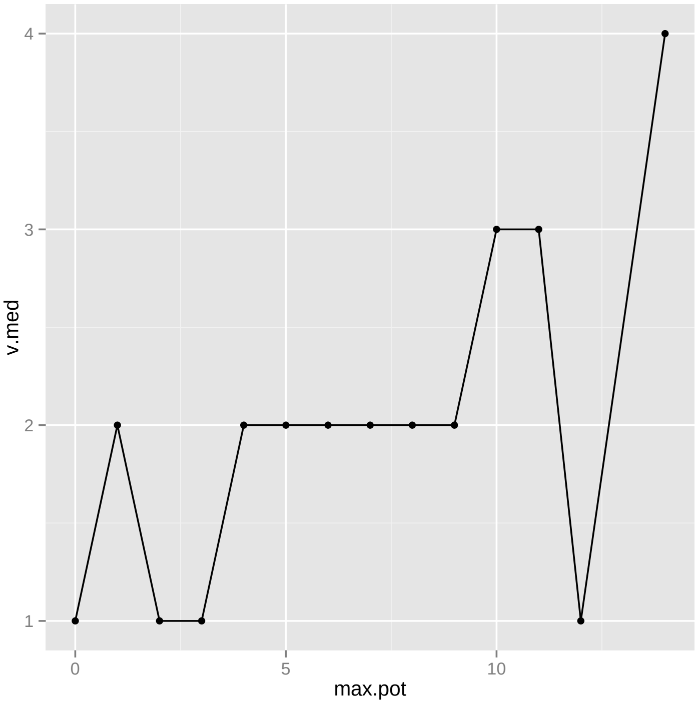


From the figure above it can be observed that some relatively potent cmpds (Log values of 7-10) have been tested in a large number of different assays (>300). These cmpds are likely 'standards' and there are relatively few of them. The smoothed line and transparent points available in the ggvis package more clealry show this, however the ggvis pkg appear to be incompaible with sweave.

5. We want to know if the median number of unique assays done increases with potency of the compounds. We use the median and not the average as the 'standards' will heavily influence the mean. The purpose of this is to determine if 'potent' compds are followed up? *Or is the kinase database more of a data dump?*

```
> biact.group <- group_by(bioact.sum, max.pot)
> uniq.assays<- summarise(biact.group,
+                          v.avg = mean(uniq.Assays),
+                          v.med = median(uniq.Assays))
```

A plot of binned potency versus median number of unique assays run is then



provided:

We see that cmpds with higher potencies do not seem to be consistently run in a variety of different assays. Indeed, most compounds are only tested in 2 different assays and this appears independent of potentcy level. Generally, the important range of log potency is between 5 and 9. This area of the plot is flat.