# Supervised Bipartite Graph Method

May 11, 2018

## Work flow

- We want to construct a bipartite graph with Drug entities and Protein entities
- We can add a new drug entity and search those protein entities in the graph within Euclidean distance  radius. And predict them as interactive to this new drug.

## Heterogeneity problem

- Drugs are expressed with chemical graph structure and Proteins are expressed with sequence structures
- They are fundamentally different, especially in dimension.
- Euclidean distance does not apply to search neighbor points within a specific radius.

## Solution

- Using two mapping systems to embed heterogeneous points into a unified Euclidean space presenting the bipartite graph. The function below will map drugs to a one-dimension Euclidean space.

$$f(x_{new}) = \sum_{j=1}^{n1} \alpha_j k_u(x_{new}, x_j)$$

  where $n1$ is the sample size of training points, $\alpha$'s are weights, and $k_u$ is a similarity function.

- Of course, we need another one in similar form to deal with proteins.

$$g(y_{new}) = \sum_{j=1}^{n2} \beta_j k_v(y_{new}, y_j)$$

## Solution

- We want to map drugs and proteins into $d$-dimension space, so we need $d$ $f$'s and $g$'s.
- A new drug sequence $X_{new}$ can be translated as

$$(f_1(X_{new}), \ldots, f_d(X_{new}))$$

- A new protein sequence $Y_{new}$ can be translated as

$$(g_1(Y_{new}), \ldots, g_d(Y_{new}))$$

- Now all of them are in the same dimension, and can be used to compute Euclidean distance between.

## Criterion

- The training criterion is to minimize

$$\frac{\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}^T \begin{bmatrix} K_u D_u K_u & -K_U A_{uv} K_v \\ -K_v A_{uv}^T K_u & K_v D_v K_v \end{bmatrix} \begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix} + \lambda_1 \alpha^T K_u \alpha + \lambda_2 \beta^T K_v \beta}{\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}^T \begin{bmatrix} K_u K_u & 0 \\ 0 & K_v K_v \boldsymbol{\beta} \end{bmatrix} \begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}}$$

  where
  - $(K_u)_{ij} = k_u(x_i, x_j)$, $i, j = 1, \ldots, n_1$
  - $(K_v)_{ij} = k_v(y_i, y_j)$, $i, j = 1, \ldots, n_2$
  - $A_{uv}$ is interaction matrix, $(A_{uv})_{ij} = 1$ if $x_i$ and $y_j$ has interaction
  - $D_u$ and $D_v$ are diagonal matrices, $D_{ii}$ is the number of interactions with drug or protein $i$ involved

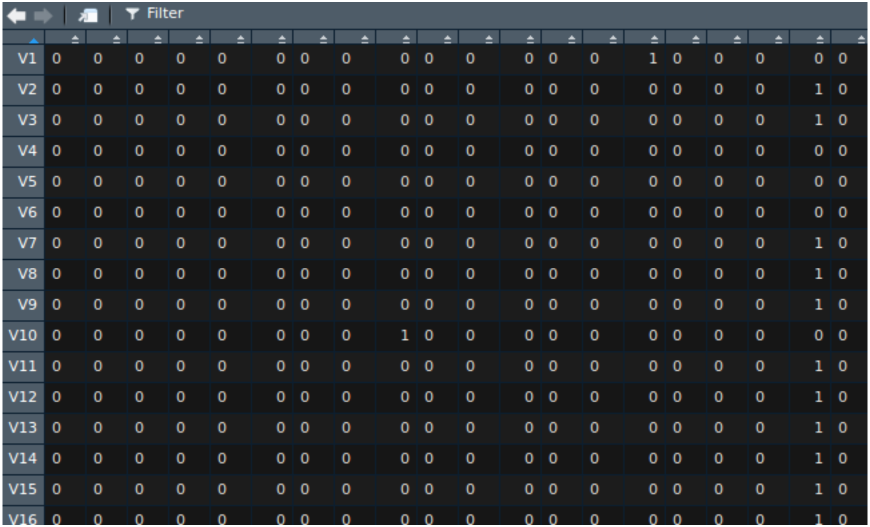
## Goal

- After being well trained, the mapping system can make sure a drug entity and a protein entity with interaction will be close in the destination space and those without interaction will be distant from each other.
- Then we can map a new drug to the space and look for its neighbor points with $\delta$ radius as its potential interactive protein.

## Experiment

- We used 200 drugs and 44 proteins in our experiment.
- $100 \times 44$ data were used as training set and another $100 \times 44$ were used as test set.
- Three matrices were used as input: cosine similarity matrix of drugs, cosine similarity of proteins and interaction matrix.

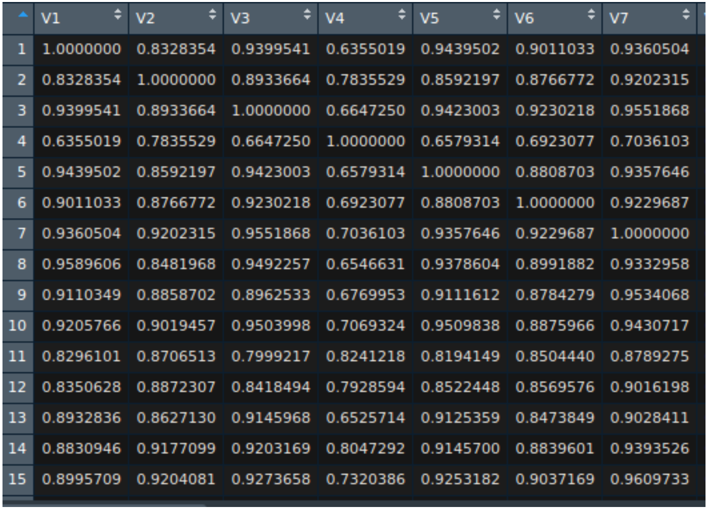
# Data



| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| V1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| V2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| V5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| V6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| V7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| V11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| V16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Figure: Interaction matrix

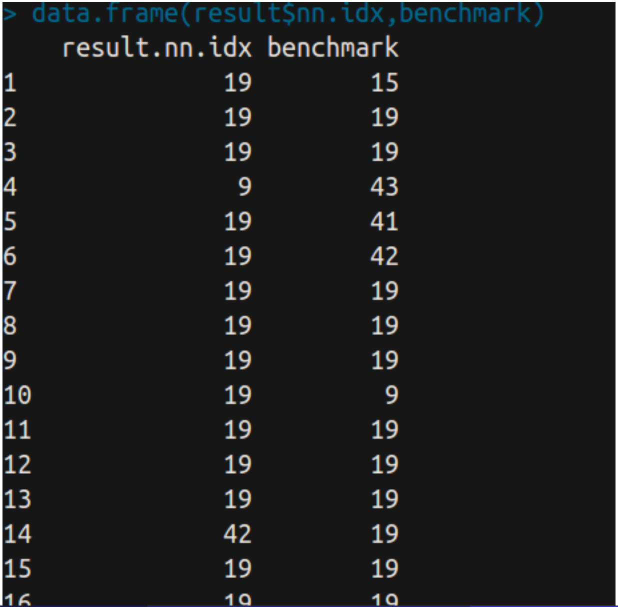
## Data



| | V1 | V2 | V3 | V4 | V5 | V6 | V7 |
|---|---|---|---|---|---|---|---|
| 1 | 1.0000000 | 0.8328354 | 0.9399541 | 0.6355019 | 0.9439502 | 0.9011033 | 0.9360504 |
| 2 | 0.8328354 | 1.0000000 | 0.8933664 | 0.7835529 | 0.8592197 | 0.8766772 | 0.9202315 |
| 3 | 0.9399541 | 0.8933664 | 1.0000000 | 0.6647250 | 0.9423003 | 0.9230218 | 0.9551868 |
| 4 | 0.6355019 | 0.7835529 | 0.6647250 | 1.0000000 | 0.6579314 | 0.6923077 | 0.7036103 |
| 5 | 0.9439502 | 0.8592197 | 0.9423003 | 0.6579314 | 1.0000000 | 0.8808703 | 0.9357646 |
| 6 | 0.9011033 | 0.8766772 | 0.9230218 | 0.6923077 | 0.8808703 | 1.0000000 | 0.9229687 |
| 7 | 0.9360504 | 0.9202315 | 0.9551868 | 0.7036103 | 0.9357646 | 0.9229687 | 1.0000000 |
| 8 | 0.9589606 | 0.8481968 | 0.9492257 | 0.6546631 | 0.9378604 | 0.8991882 | 0.9332958 |
| 9 | 0.9110349 | 0.8858702 | 0.8962533 | 0.6769953 | 0.9111612 | 0.8784279 | 0.9534068 |
| 10 | 0.9205766 | 0.9019457 | 0.9503998 | 0.7069324 | 0.9509838 | 0.8875966 | 0.9430717 |
| 11 | 0.8296101 | 0.8706513 | 0.7999217 | 0.8241218 | 0.8194149 | 0.8504440 | 0.8789275 |
| 12 | 0.8350628 | 0.8872307 | 0.8418494 | 0.7928594 | 0.8522448 | 0.8569576 | 0.9016198 |
| 13 | 0.8932836 | 0.8627130 | 0.9145968 | 0.6525714 | 0.9125359 | 0.8473849 | 0.9028411 |
| 14 | 0.8830946 | 0.9177099 | 0.9203169 | 0.8047292 | 0.9145700 | 0.8839601 | 0.9393526 |
| 15 | 0.8995709 | 0.9204081 | 0.9273658 | 0.7320386 | 0.9253182 | 0.9037169 | 0.9609733 |

Figure: Similarity matrix

# Prediction



```
> data.frame(result$nn.idx,benchmark)
   result.nn.idx benchmark
1             19        15
2             19        19
3             19        19
4              9        43
5             19        41
6             19        42
7             19        19
8             19        19
9             19        19
10            19         9
11            19        19
12            19        19
13            19        19
14            42        19
15            19        19
16            19        19
```

## Rate of Correct predictions

$$R = 0.64$$

## Q & A

Questions?

Thanks!