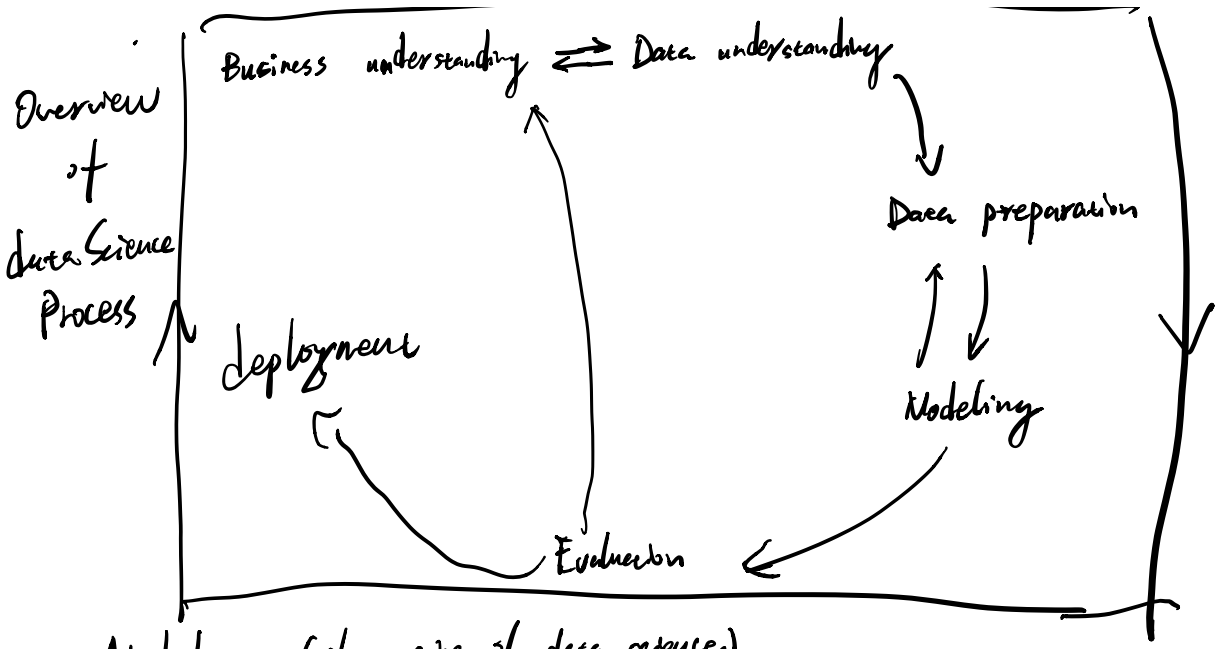
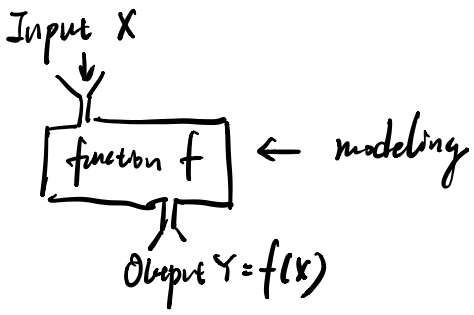


- Modeling: (the engine of data science)



- Evaluation: (the safety net of data science)
  evaluation should be built in automatically to the modeling process.
  state the evaluation framework first before getting into any modeling.

- Deployment: (model and analysis are nothing without action)
  When the model is shipped to a production system:
  - test, test, test !
  - Be proactive about QA and regular performance monitoring.

  When the analysis is delivered to people: (5-6 slides, visualization dashboards)
  - use data to tell story

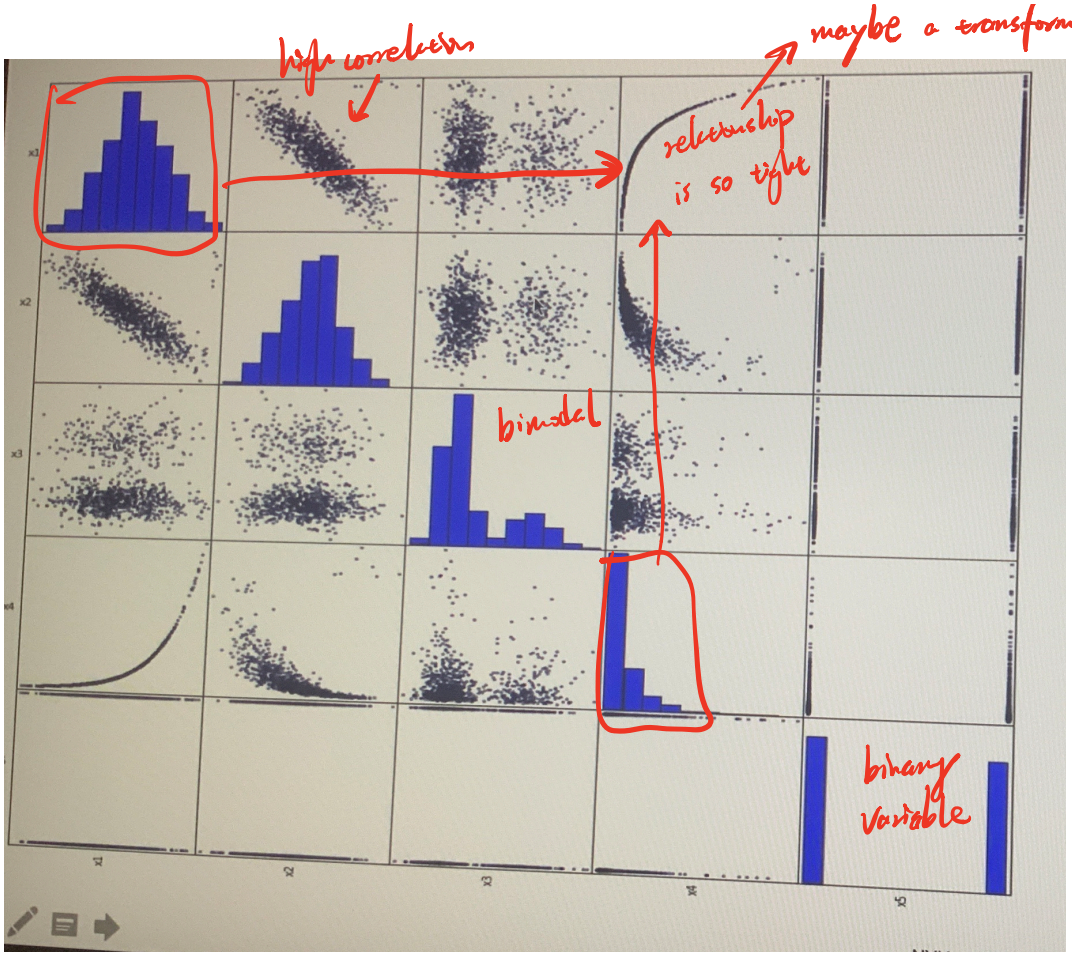
- connect your analysis to the audience's goals
- collect feedbacks

(models. or features usually in footnotes, focus on the result)

Data preparation and understanding is integral to great analysis.
It is well worth spending 80% of your time getting it right.

Always know the source:

- Internal ETL processes (extraction. transformer loading)
  someone else already structured and cleaned up for you,
  what you get is a output of table data.
- Production logging / sampling
  raw data. you can have more flexibility but more unwieldy
  data preparation and processing.
- Web scraping / API
  social network applications have their APIs.
- Survey / Panel
  collect the data from customers. use survey generator to collect
  data. it's not big data. but can solve your specific problems.



sample covariance. $cov(X,Y) = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X}_n)(Y_i - \bar{Y}_n)$

sample correlation $corr(X,Y) = \frac{cov(X,Y)}{S_X S_Y}$

$$= \frac{1}{N-1}\frac{\sum(X_i - \bar{X}_n)(Y_i - \bar{X}_n)}{S_X S_Y}$$

MI can work with both continuous and categorical data and Normalized MI is between 0 and 1. It does not give us the information of negative dependencies.

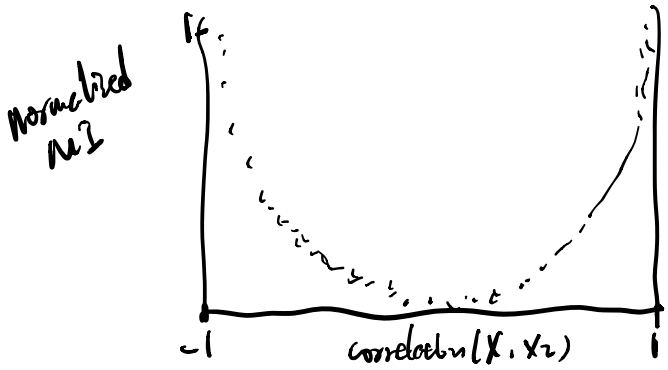
Mutual Information:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log\left(\frac{P(x,y)}{P(x)P(y)}\right)$$

If X, Y are independent. $P(X,Y) = P(X)P(Y)$, $\log(1) = 0$

If X, Y are completely dependent, F is at its maximum

Sklearn has functions for Mutual Information and Normalized Mutual Information. MI and correlation are monotonically related concepts. though MI is strictly positive that do not indicate negative dependencies

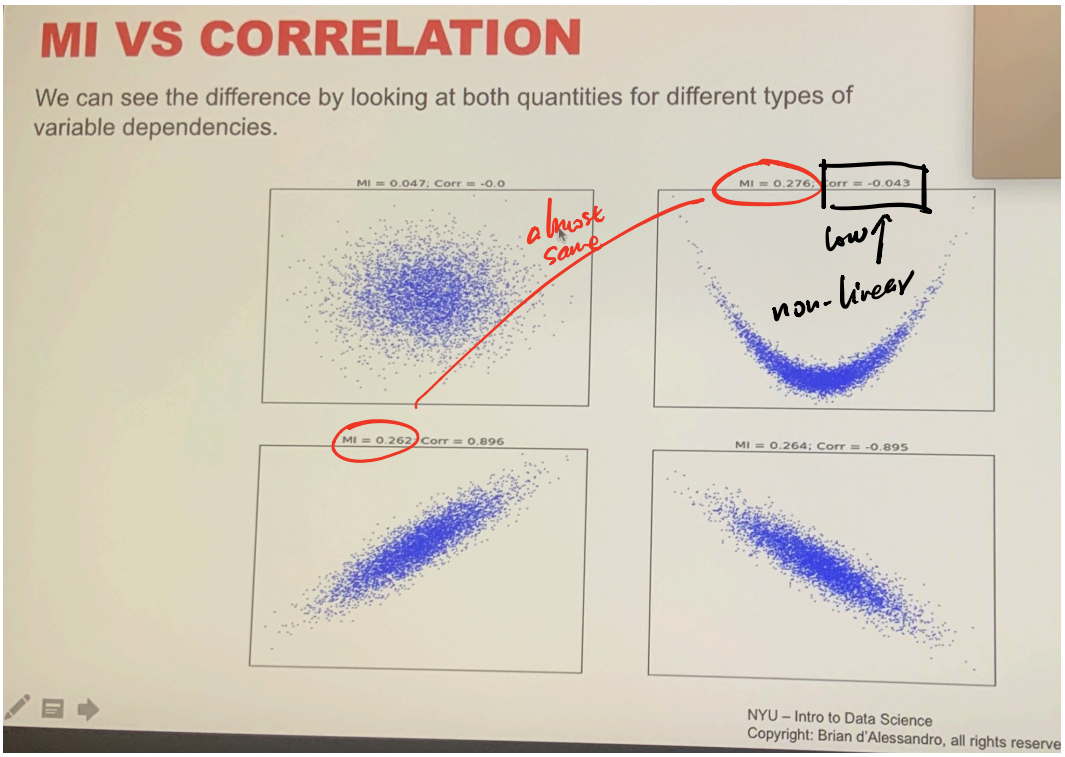


Mutual Information:

- can capture non-linear dependencies better
- works naturally with categorical data

Correlation coefficient:

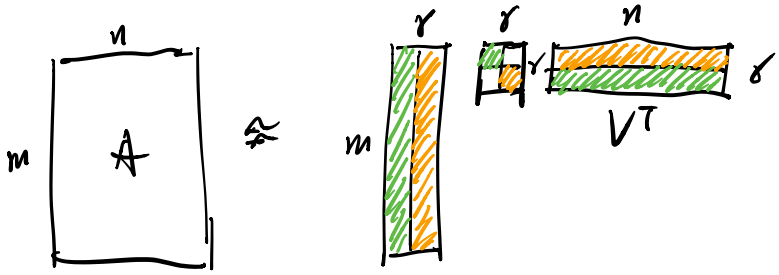
- can express negative dependencies
- well understood and intuitive. (easy to communicate)



Mutual Information can help to rank the features by calculating the MI between features and target. If MI≈0. we can get rid of feature.

Multivariate Structure (global structure)

$$A \approx U \Sigma V^T$$



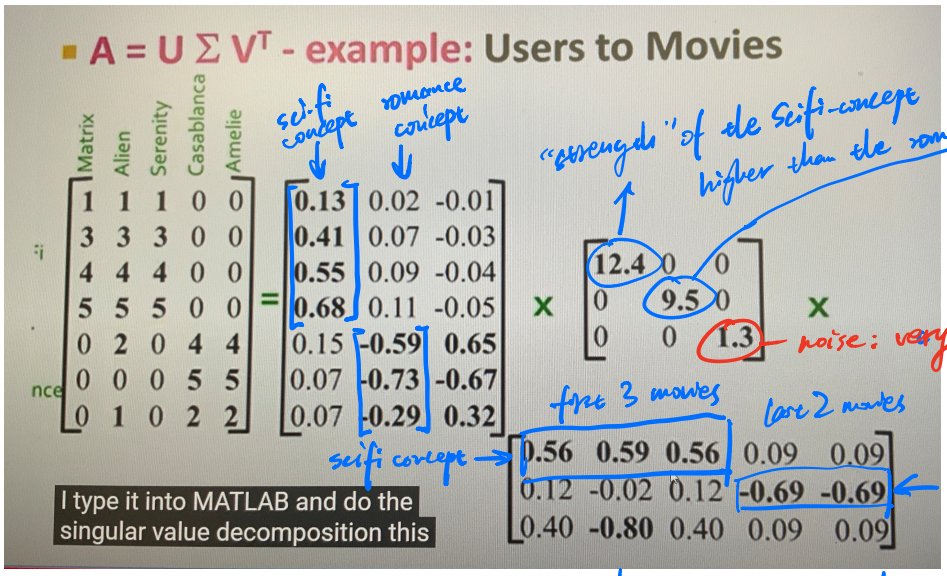
It's always possible to decompose a real matrix A into $A = U \Sigma V^T$ where:

1. U, Σ, V : unique
2. U, V : column orthonormal
   - $U^T U = I$. $V^T V = I$ (I: identity matrix)
   - (columns are orthogonal unit vectors)
3. Σ : diagonal
   - entries (singular values) are positive, and sorted in decreasing order ($\sigma_1 \geq \sigma_2 \geq \sigma_3 \cdots \geq 0$)

SVD - Example : User-to-movie

sci-fi movie someone

user	Matrix	Alien	Serenity	Casablanca	Amelia
1	1	1	1	0	0
2	3	3	3	0	0
3	4	4	4	0	0
4	5	5	5	0	0
5	0	2	0	4	4
6	0	0	0	5	5
7	0	1	0	2	2



U: user-to concept matrix. how much a user correspond to a given concept. ex: the 4-th user corresponds heavily to the scifi concept. while the 5-th user correspond heavily to the romance concept.

Σ: the strength of every concept (every value is positive)

V: the movie to concept matrix: how much every movie correspond to a given concept. we can see the first three movies heavily correspond to the scifi concept

U: left singular vectors. Each row contains k elements that correspond to the latent factors of each row of X. U is orthonormal. (user-to-concept matrix)

Σ: It's a diagonal matrix that holds the singular values (in decreasing order). The singular value tells us the weights that determine how each latent factor contributes to the matrix. (the strength matrix)

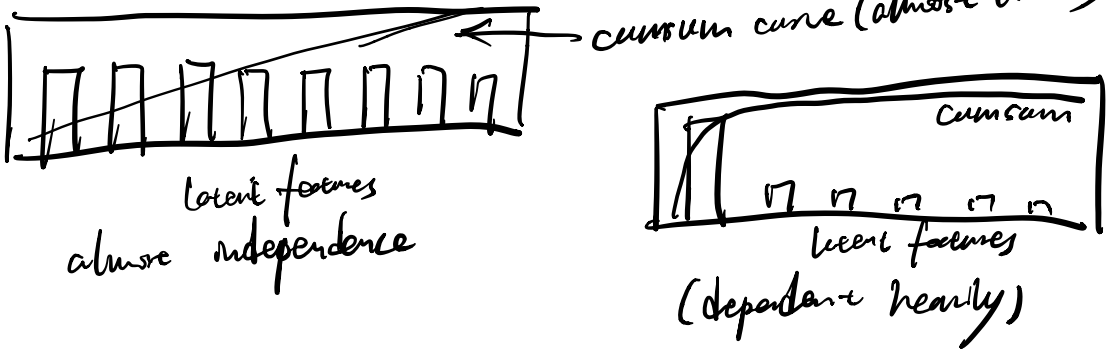
V^T: right singular vectors. Each row contains k elements that correspond to the latent factors of each column of X. V is orthonormal (movie-to-concept matrix)

corr = 0 $\Rightarrow \Sigma = \begin{bmatrix} 45 & 0 \\ 0 & 44 \end{bmatrix}$

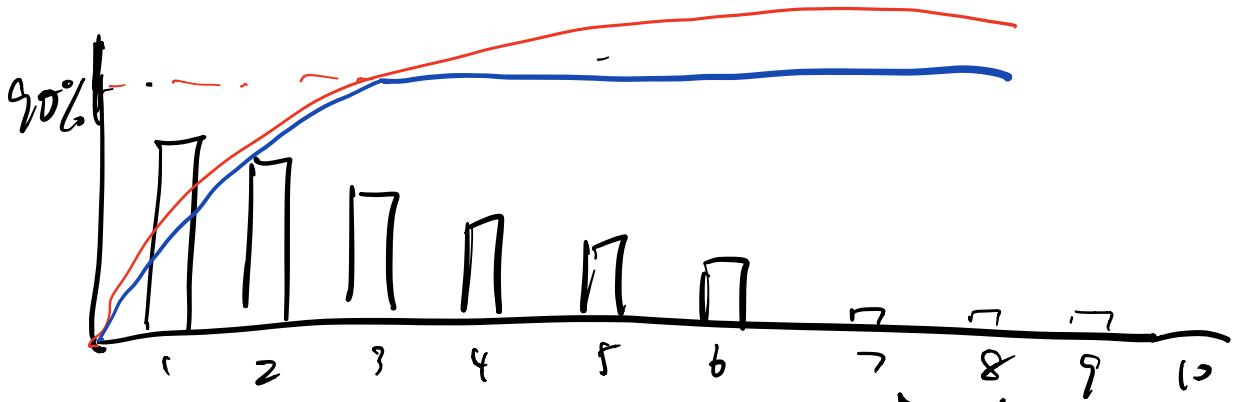
corr = 0.5 $\Rightarrow \Sigma = \begin{bmatrix} 53 & 0 \\ 0 & 32 \end{bmatrix}$

corr = 0.9 $\Rightarrow \Sigma = \begin{bmatrix} 62 & 0 \\ 0 & 5 \end{bmatrix}$

When it comes to more dimensions.



The rank of a matrix is the size of the number of independent columns of a matrix. The rank can be found by counting the number of singular value > 0



We can just take the first k dimensions of the singular matrix, which can lower the dimension without losing too much information.

these SVs are almost 0, so they add very little information. rank = 9

Low Rank Approximation

We can build a matrix X_k that approximate our original matrix by doing the following:

1. Compute the SVD of X
2. Truncate U, Σ and V to take only the k-highest columns & SVs
3. Multiply back together to get X_k

$$X_k = U_k \Sigma_k V_k^T \qquad k << M$$

$N \times M \quad N \times k \quad k \times k \quad (M \times k)^T$

Dimensionality Reduction:

- Compute rank-k SVD: $X_k = U_k \Sigma_k V_k^T$
- We can create an $N \times k$ reduced matrix: XV_k. This effectively becomes our new X matrix. and we can do analysis (cluster, modeling) with the reduced matrix.
- Project new data points to XV_k and make prediction.

Loss function: $L(f(x), y)$

Squared loss: $(f(x) - y)^2$

0-1 loss: $I(f(x) == y)$ (accuracy)

Logistic loss: $-[y \cdot \ln f(x) + (1-y) \ln(1 - f(x))]$

Hinge loss: $\max(0, 1 - f(x) \cdot y)$