Machine Learning & Data Mining
Set 5

Caltech CS/CNS/EE 155
February 2025

Policies

- Due 11:59 PM PST, February 21st on Gradescope.
- You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus.
- In this course, we will be using Google Colab for code submissions. You will need a Google account.
- You are allowed to use up to 48 late hours across the entire term. Late hours must be used in units of whole hours. Specify the total number of hours you have used when submitting the assignment.
- Students are expected to complete homework assignments based on their understanding of the course material. Student can use LLMs as a resource (e.g., helping with debugging, or grammar checking), but the assignments (including code) should be principally authored by the student.

Submission Instructions

- Submit your report as a single .pdf file to Gradescope, under "Set 5 Report".
- In the report, **include any images generated by your code** along with your answers to the questions.
- Submit your code by **sharing a link in your report** to your Google Colab notebook for each problem (see naming instructions below). Make sure to set sharing permissions to at least "Anyone with the link can view". **Links that can not be run by TAs will not be counted as turned in.** Check your links in an incognito window before submitting to be sure.
- For instructions specifically pertaining to the Gradescope submission process, see https://www.gradescope.com/get_started#student-submission.

Google Colab Instructions

For each notebook, you need to save a copy to your drive.

1. Open the github preview of the notebook, and click the icon to open the colab preview.
2. On the colab preview, go to File → Save a copy in Drive.
3. Edit your file name to "lastname_firstname_originaltitle", e.g."yue_yisong_3_notebook_part1.ipynb"

1 SVD and PCA [35 Points]

Problem A [3 points]:

Solution A:
$XX^T = (U\Sigma V^T)(V\Sigma U^T) = U\Sigma^2 U^T$.
$\lambda_i(XX^T) = \sigma_i^2(X)$
Columns of U are principal components and eigenvalues of XX^T are σ_i^2.

Problem B [4 points]:

Solution B:
$y^T X X^T y = (X^T y)^T (X^T y) = \|X^T y\|^2 \geq 0$,
so XX^T *is positive semi-definite.*
The eigenvalues of XX^T *are non-negative because* XX^T *is positive semi-definite.*

Problem C [5 points]:

Solution C:
$Tr(AB) = \sum_{i,j} A_{ij}B_{ji} = Tr(BA)$.
$Tr(ABC) = \sum_i (ABC)_{i,i} = \sum_{i,j,k} A_{i,j}B_{j,k}C_{k,i}$
$Tr(BCA) = \sum_{j,k,i} B_{j,k}C_{k,i}A_{i,j}$

Extends to $Tr(ABC) = Tr(CAB) = Tr(BCA)$.
Trace is invariant under cyclic permutations.

Problem D [3 points]:

Solution D:
Truncated SVD: $Nk + k + Nk = k(2N + 1)$. *Full* X: N^2.
Efficient when $k(2N + 1) < N^2 \Rightarrow k < \frac{N^2}{2N+1}$.
Uses $k(2N + 1)$ *values, efficient when* $k < \frac{N^2}{2N+1}$.

Problem E [3 points]: .

Solution E:

The singular value decomposition of X is $X = U\Sigma V^T$, where: - U is a $D \times D$ orthogonal matrix ($U^T U = I_{D\times D}$), - Σ is a $D \times N$ rectangular diagonal matrix with non-zero singular values $\sigma_1, \sigma_2, \ldots, \sigma_N$ on the diagonal, - V is an $N \times N$ orthogonal matrix ($V^T V = I_{N\times N}$). Since rank$(X) = N$, Σ can be partitioned as:

$$\Sigma = \begin{bmatrix} \Sigma' \\ \mathbf{0} \end{bmatrix}$$

where $\Sigma' = diag(\sigma_1, \sigma_2, \ldots, \sigma_N)$ is $N \times N$, and $\mathbf{0}$ is a $(D - N) \times N$ zero matrix.

$$U = \begin{bmatrix} U' & U'' \end{bmatrix}$$

where U' is $D \times N$ (first N columns), and U'' is $D \times (D - N)$.

$$U\Sigma = \begin{bmatrix} U' & U'' \end{bmatrix} \begin{bmatrix} \Sigma' \\ \mathbf{0} \end{bmatrix} = U'\Sigma' + U''\mathbf{0} = U'\Sigma'$$

Since $U''\mathbf{0} = 0$, it follows that $U\Sigma = U'\Sigma'$.

Problem F [3 points]:

Solution F:
U' is $D \times N$ with $D > N$, so it is not a square matrix. A matrix A is orthogonal if $AA^T = A^TA = I$. For U': - ${U'}^T U' = I_{N\times N}$ (since the columns of U' are orthonormal, being a subset of U's columns), - $U'{U'}^T$ is a $D \times D$ matrix, but its rank is at most $N < D$, so $U'{U'}^T \neq I_{D\times D}$. Since $U'{U'}^T \neq I_{D\times D}$, U' does not satisfy the full orthogonality condition and is not orthogonal.

Problem G [4 points]:

Solution G:
U' consists of the first N columns of U, an orthogonal matrix where $U^TU = I_{D\times D}$. Thus, U'^TU' is the $N \times N$ top-left submatrix of $U^TU = I_{N\times N}$.

$U'U'^T$ is a $D \times D$ matrix representing the projection onto the column space of U'. Since U' has $N < D$ columns, the rank of $U'U'^T$ is N, which is less than D. Hence, $U'U'^T \neq I_{D\times D}$.

The columns of U' are orthonormal because they are a subset of the columns of U, which are orthonormal by the definition of an orthogonal matrix.

Problem H [4 points]:

Solution H:

$X = U\Sigma^{+}V^{T}$, where Σ is $N \times N$ and invertible. Let $X^{+} = V\Sigma^{-1}U^{T}$. $XX^{+} = (U\Sigma V^{T})(V\Sigma^{-1}U^{T}) = U\Sigma\Sigma^{-1}U^{T} = UU^{T}$, $XX^{+}X = (UU^{T})(U\Sigma V^{T}) = U\Sigma V^{T} = X$, since UU^{T} acts as the identity on X's column space. $X^{+} = V\Sigma^{-1}U^{T}$ is the pseudoinverse.

Problem I [4 points]:

Solution I:

$$X^T X = (V\Sigma U^T)(U\Sigma V^T) = V\Sigma^2 V^T$$

$$since U^T U = I$$

$$(X^T X)^{-1} = (V\Sigma^2 V^T)^{-1} = V\Sigma^{-2} V^T$$

$$X^+ = (X^T X)^{-1} X^T = (V\Sigma^{-2} V^T)(V\Sigma U^T) = V\Sigma^{-1} U^T$$

Problem J [2 points]:

Solution J:
$X^+ = V\Sigma^{-1}U^T$: *Condition number is* $\kappa(\Sigma) = \frac{\sigma_1}{\sigma_N}$.

$X^+ = (X^TX)^{-1}X^T$: $X^TX = V\Sigma^2V^T$, *so* $\kappa(X^TX) = \frac{\sigma_1^2}{\sigma_N^2} = \left(\frac{\sigma_1}{\sigma_N}\right)^2$.

Expression 2 is more prone to errors because its condition number is the square of Expression 1's, thus increasing sensitivity to perturbations.

2 Matrix Factorization [30 Points]

Problem A [5 points]:

Solution A:

$$\min_{U,V} \quad \frac{\lambda}{2}\Big(\|U\|_F^2 + \|V\|_F^2\Big) \; + \; \frac{1}{2}\sum_{i,j}\big(y_{ij} \; - \; u_i^\top v_j\big)^2,$$

$$\frac{\lambda}{2}\|u_i\|^2 \quad + \quad \frac{1}{2}\sum_{j}\big(y_{ij} - u_i^\top v_j\big)^2.$$

Taking the derivative with respect to u_i

The derivative of $\frac{\lambda}{2}\|u_i\|^2 \to \lambda\, u_i$.

The derivative of $\frac{1}{2}\sum_j (y_{ij} - u_i^\top v_j)^2 \to$

$$\sum_j \big(u_i^\top v_j - y_{ij}\big)\, v_j.$$

The gradient

$$\frac{\partial}{\partial u_i} \; = \; \lambda\, u_i \; + \; \sum_j \big(u_i^\top v_j - y_{ij}\big)\, v_j.$$

$$\frac{\lambda}{2}\|v_j\|^2 \quad + \quad \frac{1}{2}\sum_{i}\big(y_{ij} - u_i^\top v_j\big)^2.$$

Taking the derivative with respect to v_j:

The derivative of $\frac{\lambda}{2}\|v_j\|^2 \to \lambda\, v_j$.
The derivative of $\frac{1}{2}\sum_i (y_{ij} - u_i^\top v_j)^2 \to$

$$\sum_i \big(u_i^\top v_j - y_{ij}\big)\, u_i.$$

The gradient

$$\frac{\partial}{\partial v_j} \; = \; \lambda\, v_j \; + \; \sum_i \big(u_i^\top v_j - y_{ij}\big)\, u_i.$$

Problem B [5 points]:

Solution B:

$$\sum_j \left(u_i^\top v_j - y_{ij}\right) v_j \;+\; \lambda\, u_i \;=\; 0.$$

Rearrange terms:

$$\underbrace{\Big(\sum_j v_j\, v_j^\top\Big)}_{V^\top V}\, u_i \;+\; \lambda\, u_i \;=\; \sum_j y_{ij}\, v_j,$$

which can be written in matrix form as

$$\left(V^\top V + \lambda\, I\right) u_i \;=\; \sum_j y_{ij}\, v_j.$$

Thus,

$$u_i \;=\; \left(V^\top V + \lambda\, I\right)^{-1} \sum_j y_{ij}\, v_j.$$

$$\sum_i \left(u_i^\top v_j - y_{ij}\right) u_i \;+\; \lambda\, v_j \;=\; 0,$$

which gives

$$\left(U^\top U + \lambda\, I\right) v_j \;=\; \sum_i y_{ij}\, u_i,$$

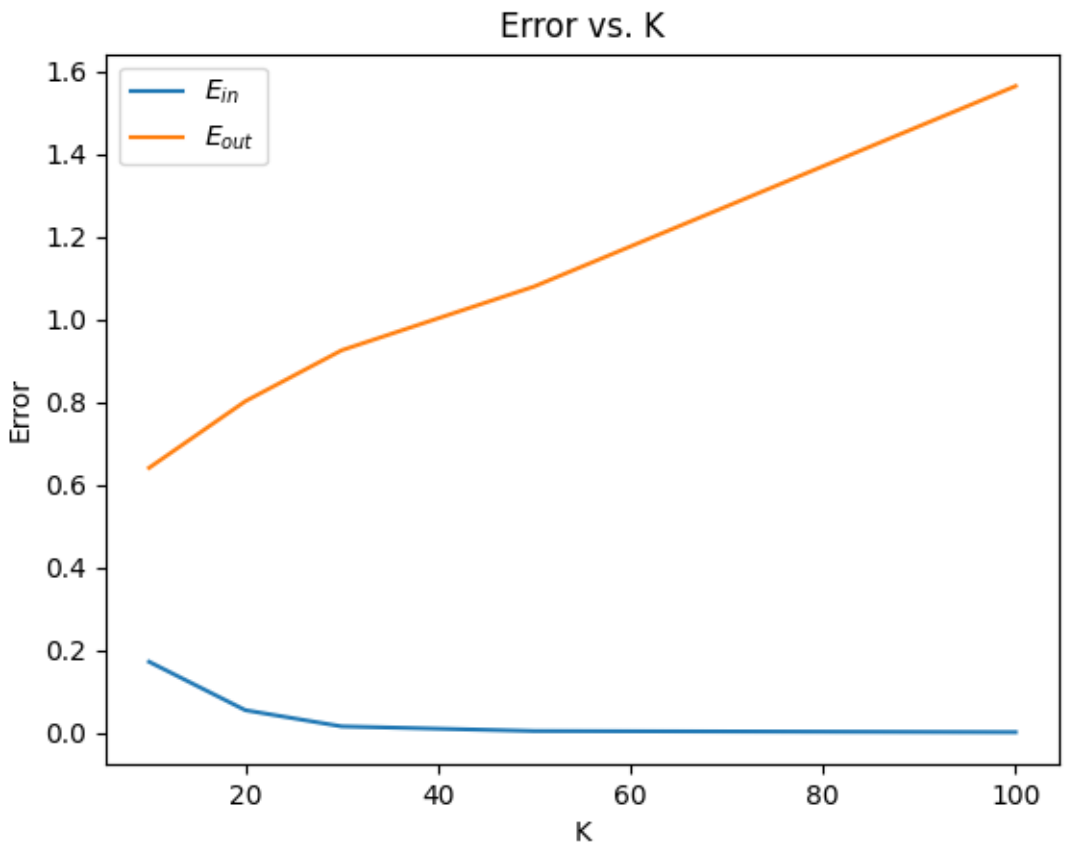
and thus

$$v_j \;=\; \left(U^\top U + \lambda\, I\right)^{-1} \sum_i y_{ij}\, u_i.$$

Problem C [10 points]:

Solution C: *Code*

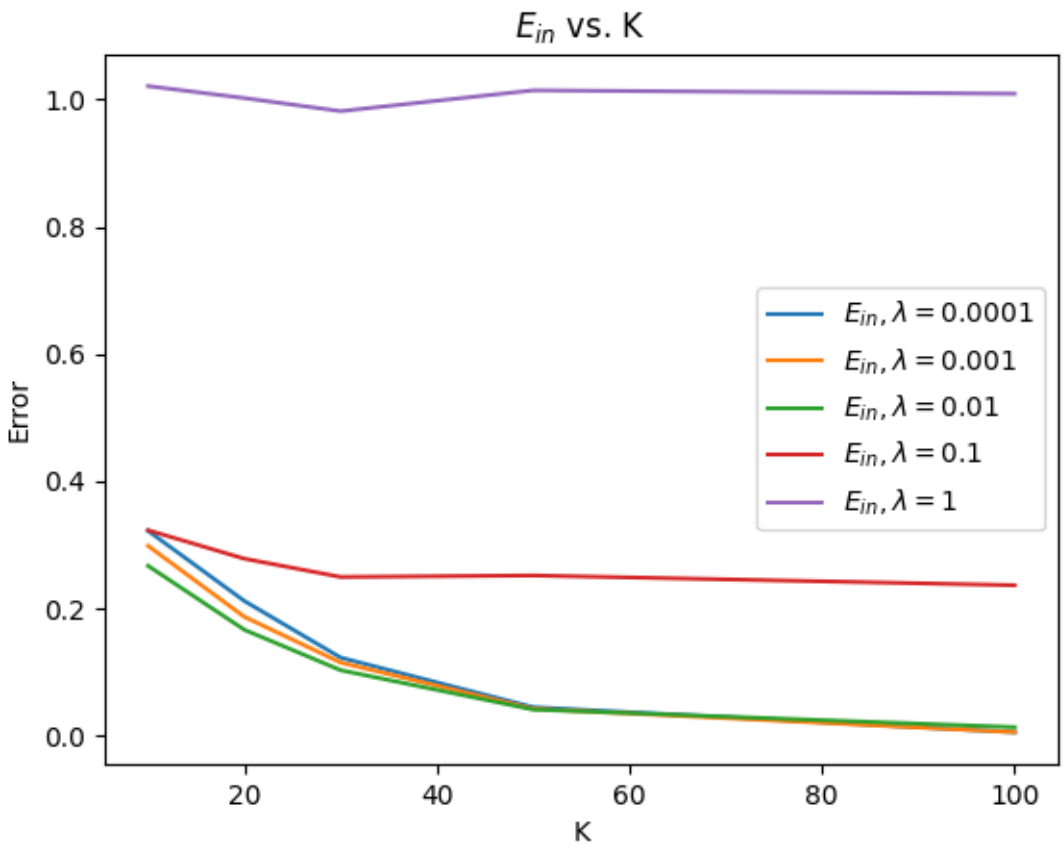
Problem D [5 points]:



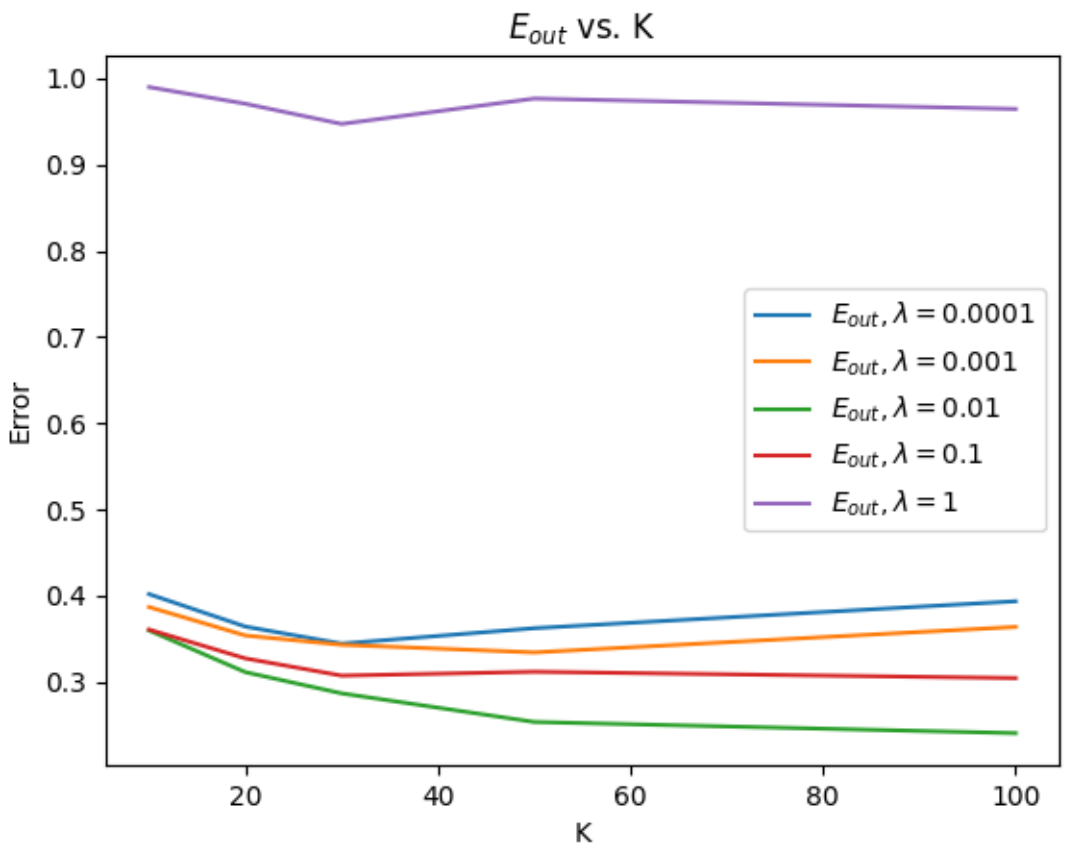
Solution D:

As k increases, the model gains capacity to better fit the training data, causing in E in to decrease. However, as k becomes too large, the model starts to overfit, memorizing training-specific noise, which raises out E out . This reflects the classic bias–variance tradeoff.

Problem E [5 points]:



Solution E:



As k grows, the model tends to overfit unless there is sufficient regularization to tame its complexity. Small values fit the training data extremely well but overfit, causing E out to climb. Larger prevents overfitting but may lead to underfitting, keeping both E in and E out higher.

3 Word2Vec Principles [35 Points]

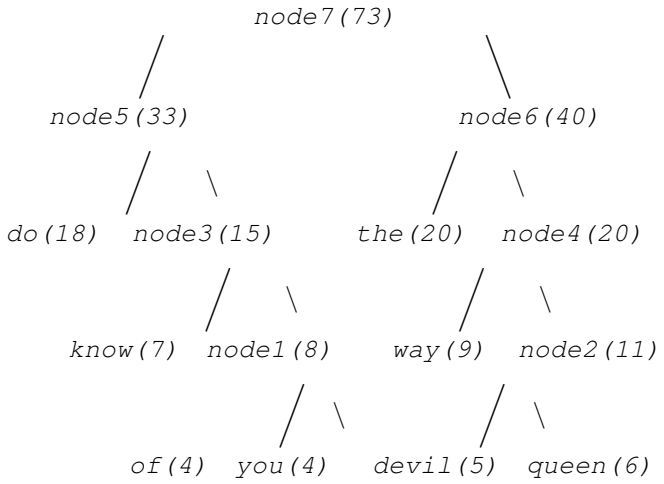
Problem A [5 points]:

Solution A: *The gradient descent for Skip–gram with a full softmax requires computing* $\nabla_{v_{w_O}} \big(-\log p(w_O \mid w_I)\big)$ *and* $\nabla_{v'_{w_I}} \big(-\log p(w_O \mid w_I)\big)$ *by summing over all* W *words in the vocabulary inside the softmax normalizer. Hence, each training example costs* $O(W)$ *just to update the parameters, making it expensive for large* W*.*

Problem B [10 points]:

Solution B: *To avoid summing over all words, one can use a* hierarchical softmax *with a binary tree. Each word is stored as a leaf node, and computing* $\log p(w_O \mid w_I)$ *then requires only evaluating the nodes along the path from the root to the leaf for* w_O*. By building a Huffman tree (which assigns shorter paths to more frequent words), the* average *path length, and therefore the average computational cost, is minimized. In contrast, a perfectly balanced tree of depth 3 assigns each of the 8 words a path length of 3, so its average representation length is 3; the Huffman tree, however, can yield a lower expected length by giving shorter paths to frequent words.*

A possible Huffman tree for the 8 words is:



Problem C [3 points]:

Solution C: *In principle, increasing the embedding dimension D gives the model more capacity to fit the data, thereby reducing the training objective. However, using a very large D risks overfitting, increases computational and storage costs, and often yields diminishing returns beyond a certain point.*

Problem D [10 points]:

Solution D: *Code*

Problem E [2 points]:

Solution E: *308 x 10*

Problem F [2 points]:

Solution F: *10 x 308*

Problem G [1 points]:

Solution G:

Pair(near, read), Similarity: 0.93040055
Pair(read, near), Similarity: 0.93040055
Pair(grows, sam), Similarity: 0.928273
Pair(sam, grows), Similarity: 0.928273
Pair(fingers, kite), Similarity: 0.92789584
Pair(kite, fingers), Similarity: 0.92789584
Pair(eleven, something), Similarity: 0.9250636
Pair(something, eleven), Similarity: 0.9250636
Pair(look, dish), Similarity: 0.91938096
Pair(dish, look), Similarity: 0.91938096
Pair(star, cold), Similarity: 0.91268593
Pair(cold, star), Similarity: 0.91268593
Pair(bad, swish), Similarity: 0.9005769
Pair(swish, bad), Similarity: 0.9005769
Pair(gump, cow), Similarity: 0.90050745
Pair(cow, gump), Similarity: 0.90050745
Pair(know, fat), Similarity: 0.8936275
Pair(fat, know), Similarity: 0.8936275
Pair(how, bump), Similarity: 0.891194
Pair(bump, how), Similarity: 0.891194
Pair(very, kite), Similarity: 0.8880101
Pair(more, did), Similarity: 0.88716364
Pair(did, more), Similarity: 0.88716364
Pair(and, all), Similarity: 0.8849913
Pair(all, and), Similarity: 0.8849913
Pair(feet, at), Similarity: 0.8845756
Pair(at, feet), Similarity: 0.8845756
Pair(way, and), Similarity: 0.8841738
Pair(them, hop), Similarity: 0.8830408
Pair(hop, them), Similarity: 0.8830408

Problem H [2 points]:

Solution H: *A key pattern is that words that frequently appear near each other in Dr.Seuss's text end up with high cosine similarity (e.g "near" and "read")*