Assignment #4

Elements of Machine Learning

Saarland University – Winter Semester 2024/25

Rabin Adhikari
7072310
`raad00002@stud.uni-saarland.de`

Dhimitrios Duka
7059153
`dhdu00001@stud.uni-saarland.de`

3 Problem 3 (Dimensionality Reduction)

3.1 What information does the first principal component capture in terms of the data variance and the data explaining?

The first principal component is the direction in which the data varies the most. In other words, it's the direction that captures the most variance in the data. Projecting the data points onto the dimension of the principal components maximizes the spread of the data compared to projecting them onto any other dimensions. This ensures that distinctions between data points are more clearly observed, making the principal component particularly effective for linear transformations. Furthermore, assuming that the underlying data is linearly distributed, the first principal component represents the closest line to the data.

3.2 Calculate the first principal component.

We can imagine the provided data as a matrix where each row represents a data point and each column represents a feature.

$$\begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix}$$

First, we have to normalize the data. To perform normalization, first we compute the per feature mean μ_i and per feature standart deviation σ_i.

$$\begin{aligned} \mu_1 &= \frac{1+2+3}{3} = 2 \\ \mu_2 &= \frac{1+2+3}{3} = 2 \\ \sigma_1 &= \sqrt{\frac{1}{2}(1+0+1)} = 1 \\ \sigma_2 &= \sqrt{\frac{1}{2}(1+0+1)} = 1 \end{aligned} \tag{1}$$

Applying the normalization formula, we get the following matrix:

$$X = \begin{pmatrix} -1 & -1 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}$$

Next, we compute the covariance matrix.

$$\begin{aligned}\Sigma &= \frac{1}{n-1}X^TX \\ &= \frac{1}{2}\begin{pmatrix}-1 & -1\\ 0 & 0\\ 1 & 1\end{pmatrix}^T\begin{pmatrix}-1 & -1\\ 0 & 0\\ 1 & 1\end{pmatrix} \\ &= \begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}\end{aligned} \tag{2}$$

Next, we have to compute the eigenvectors and eigenvalues of the covariance matrix.

$$det\begin{pmatrix}1-\lambda & 1\\ 1 & 1-\lambda\end{pmatrix} = 0$$

$$(1-\lambda)^2 - 1 = 0 \tag{3}$$

Solving the equation above, we get $\lambda_1 = 0$ and $\lambda_2 = 2$. Since $\lambda_2 > \lambda_1$, we will chose λ_2 as the first principal component.

$$\begin{aligned}(\Sigma - 2I)v = 0\\ \begin{pmatrix}-1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} = 0\end{aligned} \tag{4}$$

Solving the equation above, we get the following eigenvector: $v_2 = \begin{pmatrix}1\\ 1\end{pmatrix}$. Normalizing v_2, we get: $v_2 = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\ 1\end{pmatrix}$.

3.3 Can PCA be used to reduce the dimensionality of a highly nonlinear dataset? Explain.

PCA is not used particularly for the highly non-linear data set because the results are often unsatisfactory. This is because PCA is fundamentally a linear technique. Its goal is to identify linear combinations of features that capture the maximum variance in the data by finding orthogonal directions that represent the directions with the highest variance. However, for nonlinear datasets, PCA struggles to capture the complex relationships between features, leading to the loss of important information and failing to preserve the dataset's underlying structure. A simple example is the spiral dataset. Suppose we have a spiral dataset in 2D and apply PCA to reduce it to 1D. The data would end up being projected into a single line, therefore losing its initial structure.

3.4 When might be sensible to chain two different dimensionality reduction algorithms? You can support your answer with an example.

Applying t-SNE to a large dataset is computationally expensive as it computes pairwise conditional probabilities for each data point. The solution to this problem is to use a combination of t-SNE and PCA. First, we use PCA to reduce the dimensions to a reasonable number of features, and after that, we run t-SNE to further reduce the dimensionality of the data.

Additionally, when working with a large number of features, t-SNE may capture noise in the dataset. Therefore, it's better to first reduce the dimensionality to a reasonable level using PCA before applying t-SNE, especially when the feature count is very high.

3.5 How can you assess the effectiveness of a dimensionality reduction algorithm, used as a preprocessing step, on your dataset by considering the accuracy or error of a downstream model?

To evaluate the effectiveness of a dimensionality reduction algorithm as a preprocessing step, we first train a model using the full, original dataset and evaluate its performance on a downstream task

$\mathcal{X}$. Next, we apply the dimensionality reduction technique, ensuring that principal components or equivalent features are calculated only from the training split, while the validation and test splits are transformed using these components. The model is then retrained on the reduced dataset, and its performance is compared to the vanilla model. Dimensionality reduction is added to the pipeline if the performance of the model trained on this dataset is comparable to or better than the original model, as even similar results can justify its use due to improved computational efficiency. However, since the performance of the model may depend on the splits as well, we may need to do this a few more times to be confident whether to use dimensionality reduction in the pipeline or not.