Data Mining and Analysis: Fundamental Concepts and Algorithms

dataminingbook.info

Mohammed J. Zaki[1] Wagner Meira Jr.[2]

[1]Department of Computer Science
Rensselaer Polytechnic Institute, Troy, NY, USA

[2]Department of Computer Science
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Chapter 5: Kernel Methods

Input and Feature Space

For mining and analysis, it is important to find a suitable data representation. For example, for complex data such as text, sequences, images, and so on, we must typically extract or construct a set of attributes or features, so that we can represent the data instances as multivariate vectors.

Given a data instance $\mathbf{x}$ (e.g., a sequence), we need to find a mapping ϕ, so that $\phi(\mathbf{x})$ is the vector representation of $\mathbf{x}$.

Even when the input data is a numeric data matrix a nonlinear mapping ϕ may be used to discover nonlinear relationships.

The term *input space* refers to the data space for the input data $\mathbf{x}$ and *feature space* refers to the space of mapped vectors $\phi(\mathbf{x})$.

Sequence-based Features

Consider a dataset of DNA sequences over the alphabet $\Sigma = \{A, C, G, T\}$.

One simple feature space is to represent each sequence in terms of the probability distribution over symbols in Σ. That is, given a sequence $\mathbf{x}$ with length $|\mathbf{x}| = m$, the mapping into feature space is given as

$$\phi(\mathbf{x}) = \{P(A), P(C), P(G), P(T)\}$$

where $P(s) = \frac{n_s}{m}$ is the probability of observing symbol $s \in \Sigma$, and n_s is the number of times s appears in sequence $\mathbf{x}$.

For example, if $\mathbf{x} = ACAGCAGTA$, with $m = |\mathbf{x}| = 9$, since A occurs four times, C and G occur twice, and T occurs once, we have

$$\phi(\mathbf{x}) = (4/9, 2/9, 2/9, 1/9) = (0.44, 0.22, 0.22, 0.11)$$

We can compute larger feature spaces by considering, for example, the probability distribution over all substrings or words of size up to k over the alphabet Σ.

Nonlinear Features

Consider the mapping ϕ that takes as input a vector $\mathbf{x} = (x_1, x_2)^T \in \mathbb{R}^2$ and maps it to a "quadratic" feature space via the nonlinear mapping

$$\phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2)^T \in \mathbb{R}^3$$

For example, the point $\mathbf{x} = (5.9, 3)^T$ is mapped to the vector

$$\phi(\mathbf{x}) = (5.9^2, 3^2, \sqrt{2} \cdot 5.9 \cdot 3)^T = (34.81, 9, 25.03)^T$$

We can then apply well-known linear analysis methods in the feature space.

Kernel Method

Let $\mathcal{I}$ denote the input space, which can comprise any arbitrary set of objects, and let $\mathbf{D} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{I}$ be a dataset comprising n objects in the input space. Let $\phi : \mathcal{I} \to \mathcal{F}$ be a mapping from the input space $\mathcal{I}$ to the feature space $\mathcal{F}$.

Kernel methods avoid explicitly transforming each point $\mathbf{x}$ in the input space into the mapped point $\phi(\mathbf{x})$ in the feature space. Instead, the input objects are represented via their pairwise similarity values comprising the $n \times n$ *kernel matrix*, defined as

$$\mathbf{K} = \begin{pmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \cdots & K(\mathbf{x}_1, \mathbf{x}_n) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \cdots & K(\mathbf{x}_2, \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(\mathbf{x}_n, \mathbf{x}_1) & K(\mathbf{x}_n, \mathbf{x}_2) & \cdots & K(\mathbf{x}_n, \mathbf{x}_n) \end{pmatrix}$$

$K : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$ is a *kernel function* on any two points in input space, which should satisfy the condition

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Intuitively, we need to be able to compute the value of the dot product using the original input representation $\mathbf{x}$, without having recourse to the mapping $\phi(\mathbf{x})$.

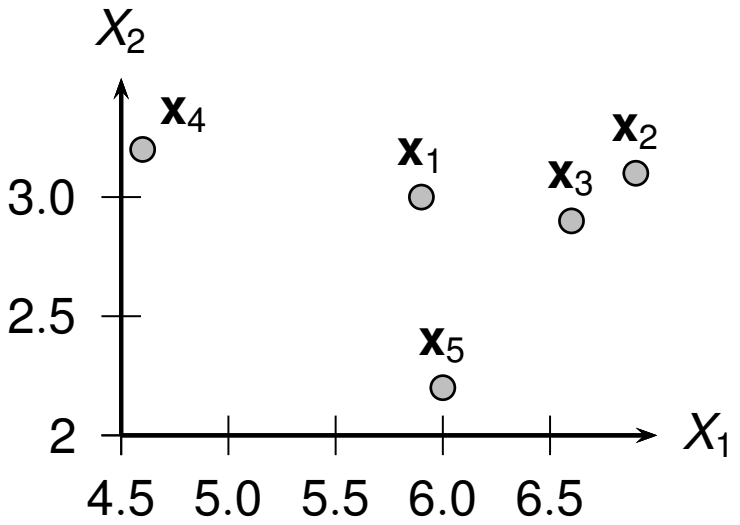
Linear Kernel

Let $\phi(\mathbf{x}) \rightarrow \mathbf{x}$ be the *identity kernel*. This leads to the *linear kernel*, which is simply the dot product between two input vectors:

$$\phi(\mathbf{x})^T \phi(\mathbf{y}) = \mathbf{x}^T \mathbf{y} = K(\mathbf{x}, \mathbf{y})$$

For example, if $\mathbf{x}_1 = \begin{pmatrix} 5.9 & 3 \end{pmatrix}^T$ and $\mathbf{x}_2 = \begin{pmatrix} 6.9 & 3.1 \end{pmatrix}^T$, then we have

$$K(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2 = 5.9 \times 6.9 + 3 \times 3.1 = 40.71 + 9.3 = 50.01$$



K	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$
$\mathbf{x}_1$	43.81	50.01	47.64	36.74	42.00
$\mathbf{x}_2$	50.01	57.22	54.53	41.66	48.22
$\mathbf{x}_3$	47.64	54.53	51.97	39.64	45.98
$\mathbf{x}_4$	36.74	41.66	39.64	31.40	34.64
$\mathbf{x}_5$	42.00	48.22	45.98	34.64	40.84

Kernel Trick

Many data mining methods can be *kernelized* that is, instead of mapping the input points into feature space, the data can be represented via the $n \times n$ kernel matrix $\mathbf{K}$, and all relevant analysis can be performed over $\mathbf{K}$.

This is done via the *kernel trick*, that is, show that the analysis task requires only dot products $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ in feature space, which can be replaced by the corresponding kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ that can be computed efficiently in input space.

Once the kernel matrix has been computed, we no longer even need the input points $\mathbf{x}_i$, as all operations involving only dot products in the feature space can be performed over the $n \times n$ kernel matrix $\mathbf{K}$.

Kernel Matrix

A function K is called a **positive semidefinite kernel** if and only if it is symmetric:

$$K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)$$

and the corresponding kernel matrix $\mathbf{K}$ for any subset $\mathbf{D} \subset \mathcal{I}$ is positive semidefinite, that is,

$$\mathbf{a}^T \mathbf{K} \mathbf{a} \geq 0, \text{ for all vectors } \mathbf{a} \in \mathbb{R}^n$$

which implies that

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \text{ for all } a_i \in \mathbb{R}, i \in [1, n]$$

Dot Products and Positive Semi-definite Kernels

Positive Semidefinite Kernel

If $K(\mathbf{x}_i, \mathbf{x}_j)$ represents the dot product $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ in some feature space, then K is a positive semidefinite kernel.

First, K is symmetric since the dot product is symmetric, which also implies that $\mathbf{K}$ is symmetric.

Second, $\mathbf{K}$ is positive semidefinite because

$$\begin{aligned}
\mathbf{a}^T \mathbf{K} \mathbf{a} &= \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \\
&= \left(\sum_{i=1}^{n} a_i \phi(\mathbf{x}_i) \right)^T \left(\sum_{j=1}^{n} a_j \phi(\mathbf{x}_j) \right) \\
&= \left\| \sum_{i=1}^{n} a_i \phi(\mathbf{x}_i) \right\|^2 \geq 0
\end{aligned}$$

Empirical Kernel Map

We now show that if we are given a positive semidefinite kernel $K : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$, then it corresponds to a dot product in some feature space $\mathcal{F}$.

Define the map ϕ as follows:

$$\phi(\mathbf{x}) = \Big((K(\mathbf{x}_1, \mathbf{x}), K(\mathbf{x}_2, \mathbf{x}), \ldots, K(\mathbf{x}_n, \mathbf{x})\Big)^T \in \mathbb{R}^n$$

The *empirical kernel map* is defined as

$$\phi(\mathbf{x}) = \mathbf{K}^{-1/2} \cdot \Big((K(\mathbf{x}_1, \mathbf{x}), K(\mathbf{x}_2, \mathbf{x}), \ldots, K(\mathbf{x}_n, \mathbf{x})\Big)^T \in \mathbb{R}^n$$

so that the dot product yields

$$\begin{aligned}
\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) &= \Big(\mathbf{K}^{-1/2}\, \mathbf{K}_i\Big)^T \Big(\mathbf{K}^{-1/2}\, \mathbf{K}_j\Big) \\
&= \mathbf{K}_i^T \left(\mathbf{K}^{-1/2} \mathbf{K}^{-1/2}\right) \mathbf{K}_j \\
&= \mathbf{K}_i^T\, \mathbf{K}^{-1}\, \mathbf{K}_j
\end{aligned}$$

where $\mathbf{K}_i$ is the ith column of $\mathbf{K}$.
Over all pairs of mapped points, we have

$$\left\{\mathbf{K}_i^T \mathbf{K}^{-1}\, \mathbf{K}_j\right\}_{i,j=1}^{n} = \mathbf{K}\, \mathbf{K}^{-1}\, \mathbf{K} = \mathbf{K}$$

Data-specific Mercer Kernel Map

The Mercer kernel map also corresponds to a dot product in feature space.

Since $\mathbf{K}$ is a symmetric positive semidefinite matrix, it has real and non-negative eigenvalues. It can be decomposed as follows:

$$\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

where $\mathbf{U}$ is the orthonormal matrix of eigenvectors $\mathbf{u}_i = (u_{i1}, u_{i2}, \ldots, u_{in})^T \in \mathbb{R}^n$ (for $i = 1, \ldots, n$), and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues, with both arranged in non-increasing order of the eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$:
The Mercer map ϕ is given as

$$\phi(\mathbf{x}_i) = \sqrt{\mathbf{\Lambda}}\mathbf{U}_i$$

where $\mathbf{U}_i$ is the ith row of $\mathbf{U}$.
The kernel value is simply the dot product between scaled rows of $\mathbf{U}$:

$$\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = \left(\sqrt{\mathbf{\Lambda}}\mathbf{U}_i\right)^T \left(\sqrt{\mathbf{\Lambda}}\mathbf{U}_j\right) = \mathbf{U}_i^T \mathbf{\Lambda} \mathbf{U}_j$$

Polynomial Kernel

Polynomial kernels are of two types: homogeneous or inhomogeneous.

Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. The (inhomogeneous) *polynomial kernel* is defined as

$$K_q(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y}) = (c + \mathbf{x}^T \mathbf{y})^q$$

where q is the degree of the polynomial, and $c \geq 0$ is some constant. When $c = 0$ we obtain the homogeneous kernel, comprising only degree q terms. When $c > 0$, the feature space is spanned by all products of at most q attributes.
This can be seen from the binomial expansion

$$K_q(\mathbf{x}, \mathbf{y}) = (c + \mathbf{x}^T \mathbf{y})^q = \sum_{k=1}^{q} \binom{q}{k} c^{q-k} \left(\mathbf{x}^T \mathbf{y}\right)^k$$

The most typical cases are the *linear* (with $q = 1$) and *quadratic* (with $q = 2$) kernels, given as

$$K_1(\mathbf{x}, \mathbf{y}) = c + \mathbf{x}^T \mathbf{y}$$
$$K_2(\mathbf{x}, \mathbf{y}) = (c + \mathbf{x}^T \mathbf{y})^2$$

Gaussian Kernel

The Gaussian kernel, also called the Gaussian radial basis function (RBF) kernel, is defined as

$$K(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right\}$$

where $\sigma > 0$ is the spread parameter that plays the same role as the standard deviation in a normal density function.

Note that $K(\mathbf{x}, \mathbf{x}) = 1$, and further that the kernel value is inversely related to the distance between the two points $\mathbf{x}$ and $\mathbf{y}$.

A feature space for the Gaussian kernel has infinite dimensionality.

Basic Kernel Operations in Feature Space

Basic data analysis tasks that can be performed solely via kernels, without instantiating $\phi(\mathbf{x})$.

Norm of a Point: We can compute the norm of a point $\phi(\mathbf{x})$ in feature space as follows:

$$\|\phi(\mathbf{x})\|^2 = \phi(\mathbf{x})^T\phi(\mathbf{x}) = K(\mathbf{x},\mathbf{x})$$

which implies that $\|\phi(\mathbf{x})\| = \sqrt{K(\mathbf{x},\mathbf{x})}$.

Distance between Points: The distance between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ is

$$\begin{aligned}\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 &= \|\phi(\mathbf{x}_i)\|^2 + \|\phi(\mathbf{x}_j)\|^2 - 2\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) \\ &= K(\mathbf{x}_i,\mathbf{x}_i) + K(\mathbf{x}_j,\mathbf{x}_j) - 2K(\mathbf{x}_i,\mathbf{x}_j)\end{aligned}$$

which implies that

$$\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\| = \sqrt{K(\mathbf{x}_i,\mathbf{x}_i) + K(\mathbf{x}_j,\mathbf{x}_j) - 2K(\mathbf{x}_i,\mathbf{x}_j)}$$

Basic Kernel Operations in Feature Space

Kernel Value as Similarity: We can rearrange the terms in

$$\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 = K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j)$$

to obtain

$$\frac{1}{2}\left(\|\phi(\mathbf{x}_i)\|^2 + \|\phi(\mathbf{x}_j)\|^2 - \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2\right) = K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

The more the distance $\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|$ between the two points in feature space, the less the kernel value, that is, the less the similarity.

Mean in Feature Space: The mean of the points in feature space is given as $\boldsymbol{\mu}_\phi = 1/n \sum_{i=1}^n \phi(\mathbf{x}_i)$. Thus, we cannot compute it explicitly. However, the the squared norm of the mean is:

$$\|\boldsymbol{\mu}_\phi\|^2 = \boldsymbol{\mu}_\phi^T\boldsymbol{\mu}_\phi = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) \tag{1}$$

The squared norm of the mean in feature space is simply the average of the values in the kernel matrix $\mathbf{K}$.

Basic Kernel Operations in Feature Space

Total Variance in Feature Space: The total variance in feature space is obtained by taking the average squared deviation of points from the mean in feature space:

$$\sigma_\phi^2 = \frac{1}{n}\sum_{i=1}^{n} \|\phi(\mathbf{x}_i) - \boldsymbol{\mu}_\phi\|^2 = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x}_i, \mathbf{x}_i) - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} K(\mathbf{x}_i, \mathbf{x}_j)$$

Centering in Feature Space We can center each point in feature space by subtracting the mean from it, as follows:

$$\hat{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \boldsymbol{\mu}_\phi$$

The kernel between centered points is given as

$$\hat{K}(\mathbf{x}_i, \mathbf{x}_j) = \hat{\phi}(\mathbf{x}_i)^T \hat{\phi}(\mathbf{x}_j)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) - \frac{1}{n}\sum_{k=1}^{n} K(\mathbf{x}_i, \mathbf{x}_k) - \frac{1}{n}\sum_{k=1}^{n} K(\mathbf{x}_j, \mathbf{x}_k) + \frac{1}{n^2}\sum_{a=1}^{n}\sum_{b=1}^{n} K(\mathbf{x}_a, \mathbf{x}_b)$$

More compactly, we have:

$$\hat{\mathbf{K}} = \left(\mathbf{I} - \frac{1}{n}\mathbf{1}_{n\times n}\right)\mathbf{K}\left(\mathbf{I} - \frac{1}{n}\mathbf{1}_{n\times n}\right)$$

where $\mathbf{1}_{n\times n}$ is the $n \times n$ matrix of ones.

Basic Kernel Operations in Feature Space

Normalizing in Feature Space: The dot product between normalized points in feature space corresponds to the cosine of the angle between them

$$\phi_n(\mathbf{x}_i)^T \phi_n(\mathbf{x}_j) = \frac{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}{\|\phi(\mathbf{x}_i)\| \cdot \|\phi(\mathbf{x}_j)\|} = \cos\theta$$

If the mapped points are both centered and normalized, then a dot product corresponds to the correlation between the two points in feature space.

The normalized kernel matrix, $\mathbf{K}_n$, can be computed using only the kernel function K, as

$$\mathbf{K}_n(\mathbf{x}_i, \mathbf{x}_j) = \frac{\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}{\|\phi(\mathbf{x}_i)\| \cdot \|\phi(\mathbf{x}_j)\|} = \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) \cdot K(\mathbf{x}_j, \mathbf{x}_j)}}$$

$\mathbf{K}_n$ has all diagonal elements as 1.

Spectrum Kernel for Strings

Given alphabet Σ, the l-spectrum feature map is the mapping $\phi: \Sigma^* \to \mathbb{R}^{|\Sigma|^l}$ from the set of substrings over Σ to the $|\Sigma|^l$-dimensional space representing the number of occurrences of all possible substrings of length l, defined as

$$\phi(\mathbf{x}) = \Big(\cdots, \#(\alpha), \cdots\Big)^T_{\alpha \in \Sigma^l}$$

where $\#(\alpha)$ is the number of occurrences of the l-length string α in $\mathbf{x}$.

The (full) spectrum map considers all lengths from $l = 0$ to $l = \infty$, leading to an infinite dimensional feature map $\phi : \Sigma^* \to \mathbb{R}^\infty$:

$$\phi(\mathbf{x}) = \Big(\cdots, \#(\alpha), \cdots\Big)^T_{\alpha \in \Sigma^*}$$

where $\#(\alpha)$ is the number of occurrences of the string α in $\mathbf{x}$.

The (l-)spectrum kernel between two strings $\mathbf{x}_i, \mathbf{x}_j$ is simply the dot product between their (l-)spectrum maps:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

The (full) spectrum kernel can be computed efficiently via suffix trees in $O(n + m)$ time for two strings of length n and m.

Diffusion Kernels on Graph Nodes

Let $\mathbf{S}$ be some symmetric similarity matrix between nodes of a graph $G = (V, E)$. For instance, $\mathbf{S}$ can be the (weighted) adjacency matrix $\mathbf{A}$ or the Laplacian matrix $\mathbf{L} = \mathbf{A} - \mathbf{\Delta}$ (or its negation), where $\mathbf{\Delta}$ is the degree matrix for an undirected graph G, defined as $\mathbf{\Delta}(i, i) = d_i$ and $\mathbf{\Delta}(i, j) = 0$ for all $i \neq j$, and d_i is the degree of node i.

Power Kernels: Summing up the product of the base similarities over all l-length paths between two nodes, we obtain the l-length similarity matrix $\mathbf{S}^{(l)}$, which is simply the lth power of $\mathbf{S}$, that is,

$$\mathbf{S}^{(l)} = \mathbf{S}^l$$

Even path lengths lead to positive semidefinite kernels, but odd path lengths are not guaranteed to do so, unless the base matrix $\mathbf{S}$ is itself a positive semidefinite matrix.

Power kernel $\mathbf{K}$ can be obtained via the eigen-decomposition of $\mathbf{S}^l$:

$$\mathbf{K} = \mathbf{S}^l = \left(\mathbf{U}\mathbf{\Lambda}\mathbf{U}^T\right)^l = \mathbf{U}\left(\mathbf{\Lambda}^l\right)\mathbf{U}^T$$

Exponential Diffusion Kernel

The exponential diffusion kernel we can obtain a new kernel between nodes of a graph by paths of all possible lengths, but damps the contribution of longer paths

$$\begin{aligned}\mathbf{K} &= \sum_{l=0}^{\infty} \frac{1}{l!}\beta^l \mathbf{S}^l \\ &= \mathbf{I} + \beta\mathbf{S} + \frac{1}{2!}\beta^2\mathbf{S}^2 + \frac{1}{3!}\beta^3\mathbf{S}^3 + \cdots \\ &= \exp\{\beta\mathbf{S}\}\end{aligned}$$

where β is a damping factor, and $\exp\{\beta\mathbf{S}\}$ is the matrix exponential. The series on the right hand side above converges for all $\beta \geq 0$.

Substituting $\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ the kernel can be computed as

$$\begin{aligned}\mathbf{K} &= \mathbf{I} + \beta\mathbf{S} + \frac{1}{2!}\beta^2\mathbf{S}^2 + \cdots \\ &= \mathbf{U}\begin{pmatrix} \exp\{\beta\lambda_1\} & 0 & \cdots & 0 \\ 0 & \exp\{\beta\lambda_2\} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \exp\{\beta\lambda_n\} \end{pmatrix}\mathbf{U}^T\end{aligned}$$

where λ_i is an eigenvalue of $\mathbf{S}$.

Von Neumann Diffusion Kernel

The *von Neumann diffusion kernel* is defined as

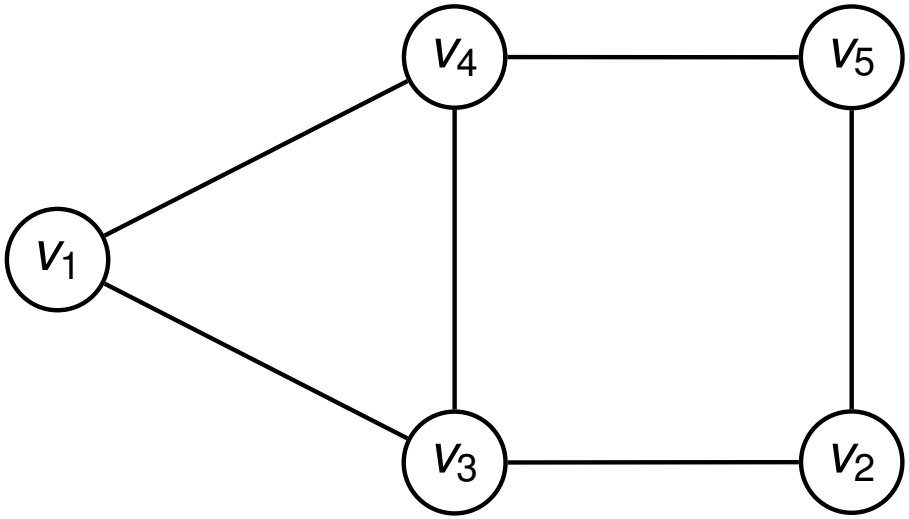
$$\mathbf{K} = \sum_{l=0}^{\infty} \beta^l \mathbf{S}^l$$

where $\beta \geq 0$. Expanding and rearranging the terms, we obtain

$$\mathbf{K} = (\mathbf{I} - \beta \mathbf{S})^{-1}$$

The kernel is guaranteed to be positive semidefinite if $|\beta| < 1/\rho(\mathbf{S})$, where $\rho(\mathbf{S}) = \max_i\{|\lambda_i|\}$ is called the *spectral radius* of $\mathbf{S}$, defined as the largest eigenvalue of $\mathbf{S}$ in absolute value.

Graph Diffusion Kernel: Example



Adjacency and degree matrices are given as

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \qquad \mathbf{\Delta} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

Graph Diffusion Kernel: Example

Let the base similarity matrix $\mathbf{S}$ be the negated Laplacian matrix

$$\mathbf{S} = -\mathbf{L} = \mathbf{A} - \mathbf{D} = \begin{pmatrix} -2 & 0 & 1 & 1 & 0 \\ 0 & -2 & 1 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 \\ 1 & 0 & 1 & -3 & 1 \\ 0 & 1 & 0 & 1 & -2 \end{pmatrix}$$

The eigenvalues of $\mathbf{S}$ are as follows:

$$\lambda_1 = 0 \qquad \lambda_2 = -1.38 \qquad \lambda_3 = -2.38 \qquad \lambda_4 = -3.62 \qquad \lambda_5 = -4.62$$

and the eigenvectors of $\mathbf{S}$ are

$$\mathbf{U} = \begin{pmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 & \mathbf{u}_4 & \mathbf{u}_5 \\ \hline 0.45 & -0.63 & 0.00 & 0.63 & 0.00 \\ 0.45 & 0.51 & -0.60 & 0.20 & -0.37 \\ 0.45 & -0.20 & -0.37 & -0.51 & 0.60 \\ 0.45 & -0.20 & 0.37 & -0.51 & -0.60 \\ 0.45 & 0.51 & 0.60 & 0.20 & 0.37 \end{pmatrix}$$

Graph Diffusion Kernel: Example

Assuming $\beta = 0.2$, the exponential diffusion kernel matrix is given as

$$\mathbf{K} = \exp\{0.2\mathbf{S}\} = \mathbf{U} \begin{pmatrix} \exp\{0.2\lambda_1\} & 0 & \cdots & 0 \\ 0 & \exp\{0.2\lambda_2\} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \exp\{0.2\lambda_n\} \end{pmatrix} \mathbf{U}^T$$

$$= \begin{pmatrix} 0.70 & 0.01 & 0.14 & 0.14 & 0.01 \\ 0.01 & 0.70 & 0.13 & 0.03 & 0.14 \\ 0.14 & 0.13 & 0.59 & 0.13 & 0.03 \\ 0.14 & 0.03 & 0.13 & 0.59 & 0.13 \\ 0.01 & 0.14 & 0.03 & 0.13 & 0.70 \end{pmatrix}$$

Assuming $\beta = 0.2$, the von Neumann kernel is given as

$$\mathbf{K} = \mathbf{U}(\mathbf{I} - 0.2\mathbf{\Lambda})^{-1}\mathbf{U}^T = \begin{pmatrix} 0.75 & 0.02 & 0.11 & 0.11 & 0.02 \\ 0.02 & 0.74 & 0.10 & 0.03 & 0.11 \\ 0.11 & 0.10 & 0.66 & 0.10 & 0.03 \\ 0.11 & 0.03 & 0.10 & 0.66 & 0.10 \\ 0.02 & 0.11 & 0.03 & 0.10 & 0.74 \end{pmatrix}$$