



MITx 6.86x

# Machine Learning with Python-From Linear Models to Deep Learning

[Course](#)

[Progress](#)

[Dates](#)

[Discussion](#)

[Resources](#)



[Course](#) / [Unit 2. Nonlinear Classification, Linear regression,...](#) / [Project 2: Dig](#)

< Previous



## 10. Kernel Methods

 Bookmark this page

Project due Mar 15, 2023 08:59 -03 Past due

As you can see, implementing a direct mapping to the high-dimensional features is a lot more expensive than an even higher dimensional feature mapping.) This is where the kernel trick becomes useful.

Recall the kernel perceptron algorithm we learned in the lecture. The weights  $\theta$  can be represented as a combination of features:

$$\theta = \sum_{i=1}^n \alpha^{(i)} y^{(i)} \phi(x^{(i)})$$

In the softmax regression formulation, we can also apply this representation of the weights.

$$\theta_j = \sum_{i=1}^n \alpha_j^{(i)} \phi(x^{(i)}).$$

$$h(x) = \frac{1}{\sum_{j=1}^k e^{[\theta_j \cdot \phi(x)/\tau] - c}} \begin{bmatrix} e^{[\theta_1 \cdot \phi(x)/\tau] - c} \\ e^{[\theta_2 \cdot \phi(x)/\tau] - c} \\ \vdots \\ e^{[\theta_k \cdot \phi(x)/\tau] - c} \end{bmatrix}$$

$$h(x) = \frac{1}{\sum_{j=1}^k e^{[\sum_{i=1}^n \alpha_j^{(i)} y^{(i)} \phi(x^{(i)}) \cdot \phi(x)/\tau] - c}} \begin{bmatrix} e^{[\sum_{i=1}^n \alpha_1^{(i)} y^{(i)} \phi(x^{(i)}) \cdot \phi(x)/\tau] - c} \\ e^{[\sum_{i=1}^n \alpha_2^{(i)} y^{(i)} \phi(x^{(i)}) \cdot \phi(x)/\tau] - c} \\ \vdots \\ e^{[\sum_{i=1}^n \alpha_k^{(i)} y^{(i)} \phi(x^{(i)}) \cdot \phi(x)/\tau] - c} \end{bmatrix}$$

We actually do not need the real mapping  $\phi(x)$ , but the inner product between two feature vectors  $\phi(x_i) \cdot \phi(x)$ , where  $x_i$  is a point in the training set and  $x$  is the new data point for which we want to compute the probability. If we can create a kernel function  $K(x, y) = \phi(x) \cdot \phi(y)$ , for any two points  $x$  and  $y$ , then we can kernelize our softmax regression algorithm.

**You will be working in the files `part1/main.py` and `part1/kernel.py` in this problem set.**

```

4         compute the polynomial kernel between two matrices X and Y::
5          $K(x, y) = (\langle x, y \rangle + c)^p$ 
6         for each pair of rows x in X and y in Y.
7
8     Args:
9         X - (n, d) NumPy array (n datapoints each with d features)
10        Y - (m, d) NumPy array (m datapoints each with d features)
11        c - a coefficient to trade off high-order and low-order terms
12        p - the degree of the polynomial kernel
13
14    Returns:
15        kernel_matrix - (n, m) Numpy array containing the kernel matrix

```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

Submit

You have used 0 of 25 attempts

## Gaussian RBF Kernel

1 point possible (graded)

Another commonly used kernel is the Gaussian RBF kernel. Similarly, write a function `rbf_kernel(X, Y, gamma)` that takes two matrices `X` and `Y` and computes the RBF kernel  $K(x, y) = \exp(-\gamma ||x - y||^2)$  for every pair of rows

**Available Functions:** You have access to the NumPy python library as `np`

```

1
2 def rbf_kernel(X, Y, gamma):
3     """
4         Compute the Gaussian RBF kernel between two matrices X and Y::
5          $K(x, y) = \exp(-\gamma ||x - y||^2)$ 
6         for each pair of rows x in X and y in Y.
7
8     Args:
9         X - (n, d) NumPy array (n datapoints each with d features)
10        Y - (m, d) NumPy array (m datapoints each with d features)
11        gamma - the gamma parameter of gaussian function (scalar)
12
13    Returns:
14        kernel_matrix - (n, m) Numpy array containing the kernel matrix
15    """

```

Press ESC then TAB or click outside of the code editor to exit

Unanswered

Submit

You have used 0 of 25 attempts

kernelized features.

In the next project, you will apply neural networks to this task.

[< Previous](#)[Next >](#)

## Discussion



## edX

[About](#)[Affiliates](#)[edX for Business](#)[Open edX](#)[Careers](#)[News](#)

## Legal

[Terms of Service & Honor Code](#)[Privacy Policy](#)[Accessibility Policy](#)[Trademark Policy](#)[Sitemap](#)[Cookie Policy](#)[Do Not Sell My Personal Information](#)

## Connect

[Blog](#)[Contact Us](#)[Help Center](#)[Security](#)[Media Kit](#)