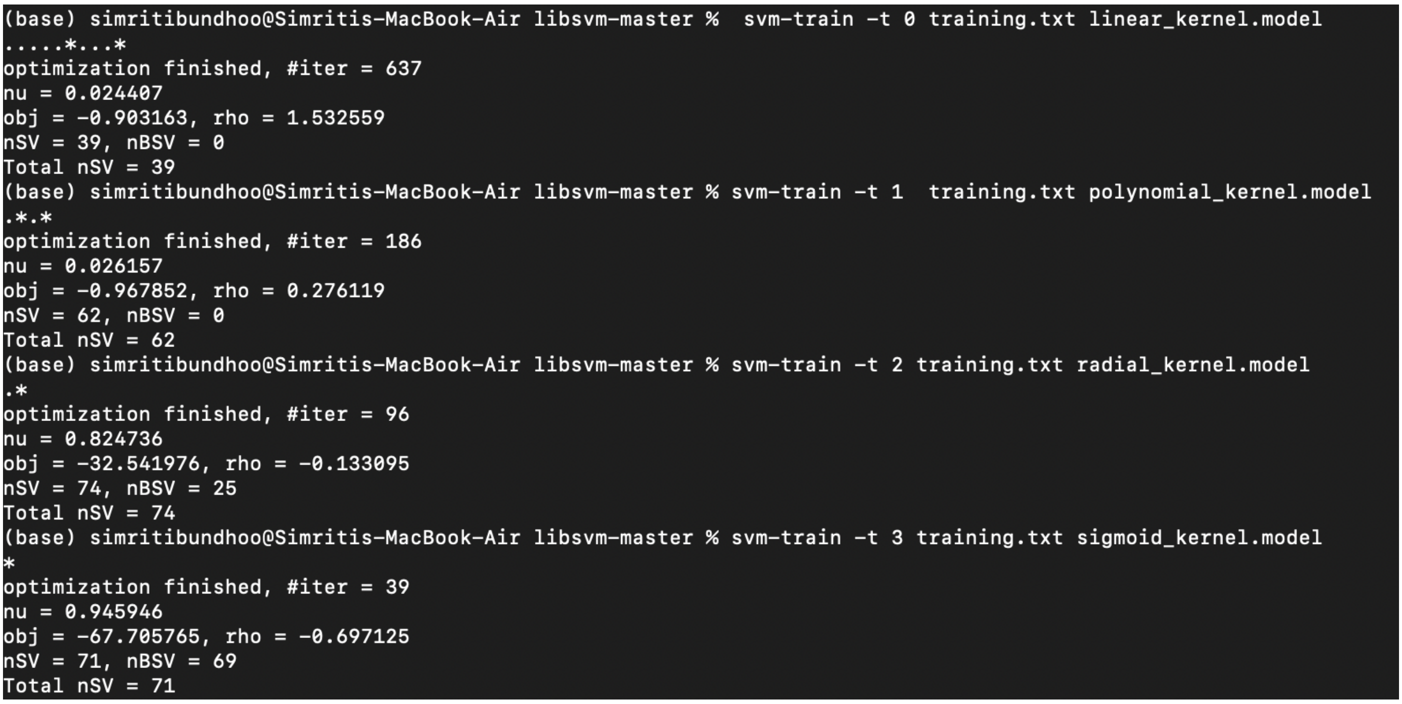
50.007 Machine Learning

Simriti Bundhoo 1006281

Homework 3

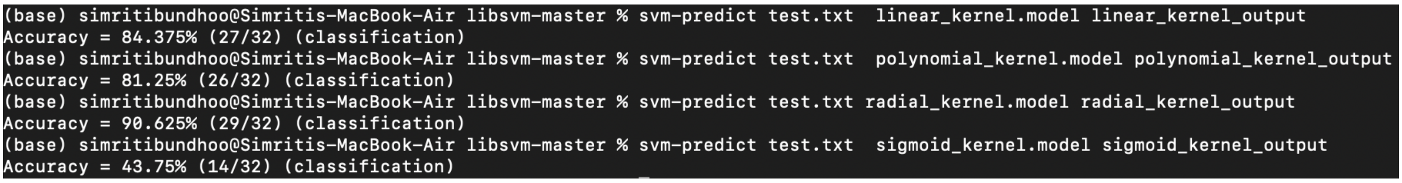
1

Training the kernel models:



```
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master %  svm-train -t 0 training.txt linear_kernel.model
.....*...*
optimization finished, #iter = 637
nu = 0.024407
obj = -0.903163, rho = 1.532559
nSV = 39, nBSV = 0
Total nSV = 39
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-train -t 1  training.txt polynomial_kernel.model
.*.*
optimization finished, #iter = 186
nu = 0.026157
obj = -0.967852, rho = 0.276119
nSV = 62, nBSV = 0
Total nSV = 62
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-train -t 2 training.txt radial_kernel.model
.*
optimization finished, #iter = 96
nu = 0.824736
obj = -32.541976, rho = -0.133095
nSV = 74, nBSV = 25
Total nSV = 74
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-train -t 3 training.txt sigmoid_kernel.model
*
optimization finished, #iter = 39
nu = 0.945946
obj = -67.705765, rho = -0.697125
nSV = 71, nBSV = 69
Total nSV = 71
```

Test accuracy for each kernel choice:



```
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-predict test.txt  linear_kernel.model linear_kernel_output
Accuracy = 84.375% (27/32) (classification)
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-predict test.txt  polynomial_kernel.model polynomial_kernel_output
Accuracy = 81.25% (26/32) (classification)
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-predict test.txt radial_kernel.model radial_kernel_output
Accuracy = 90.625% (29/32) (classification)
(base) simritibundhoo@Simritis-MacBook-Air libsvm-master % svm-predict test.txt  sigmoid_kernel.model sigmoid_kernel_output
Accuracy = 43.75% (14/32) (classification)
```

2

CASE 1: $K(x,z) = K_1(x,z)K_2(x,z)$
Since $K_1(x,z)$ and $K_2(x,z)$ are kernels, they can be re-written as inner products of vectors in some feature space:

$$K_1(x,z) = \phi_1(x)^T \times \phi_1(z)$$

$$K_2(x,z) = \phi_2(x)^T \times \phi_2(z)$$

Using the definition: $K(x,z) = K_1(x,z) \times K_2(x,z) = \phi_1(x)^T\phi_1(z) \times \phi_2(x)^T\phi_2(z)$. A new feature vector is defined by $\phi(x) = [\phi_1(x), \phi_2(x)]$.

Therefore,

$$\begin{aligned} K(x,z) &= \phi_1(x)^T\phi_1(z) \times \phi_2(x)^T\phi_2(z) \\ &= \phi(x)^T\phi_1(z) \times \phi(x)^T\phi_2(z) \\ &= (\phi(x)^T\phi_1(z)) \times (\phi(x)^T\phi_2(z)) \\ &= \phi(x)^T \times \phi(z) \end{aligned}$$

Since $\phi(x)$ is a valid feature vector in some feature space, **K(x, z) is also a valid kernel**.

CASE 2: $K(x,z) = aK_1(x,z) + bK_2(x,z)$
As in case 1, $K_1(x,z)$ and $K_2(x,z)$ can be re-written as inner products of vectors in some feature space:

$$K_1(x,z) = \phi_1(x)^T \times \phi_1(z)$$

$$K_2(x,z) = \phi_2(x)^T \times \phi_2(z)$$

Using the definition: $K(x,z) = aK_1(x,z) + bK_2(x,z) = a \times (\phi_1(x)^T \times \phi_1(z)) + b \times \phi_2(x)^T \times \phi_2(z)$. A new feature vector is defined by $\phi(x) = [a \times \phi_1(x), b \times \phi_2(x)]$.

Therefore,

$$\begin{aligned} K(x,z) &= a \times (\phi_1(x)^T\phi_1(z)) + b \times (\phi_2(x)^T\phi_2(z)) \\ &= (\phi(x)^T\phi_1(z)) + \phi((x)^T\phi_2(z)) \\ &= \phi(x)^T \times (\phi_1(z) + \phi_2(z)) \\ &= \phi(x)^T \times \phi(z) \end{aligned}$$

Since $\phi(x)$ is a valid feature vector in some feature space, **K(x, z) is also a valid kernel**.

CASE 3: $K(x,z) = aK_1(x,z) - bK_2(x,z)$
Following similar steps as in case 2, a new feature vector is defined by $\phi(x) = [a \times \phi_1(x), -b \times \phi_2(x)]$. Therefore, as in case 2, $K(x,z)$ will also be simplified to $K(x,z) = \phi(x)^T \times \phi(z)$. Since $\phi(x)$ is a valid feature vector in some feature space, **K(x, z) is also a valid kernel**.

CASE 4: $K(x,z) = f(x)f(z)$
Consider a kernel matrix K' where the (i,j)-th element is $K'(x_i, x_j) = f(x_i) \times f(x_j))$.
For K to be a valid kernel, K' must be symmetric and positive for any set of examples x_1, x_2, ..., x_m. However, K' may not be necessarily symmetric and positive semidefinite for all possible functions $f(x)$.

Counterexample: $f(x) = x$

In this case, $K' = [(x_1^2) \times (x_1x_2), (x_1x_2) \times (x_2^2)]$. The matrix K' is not necessarily symmetric and may not be positive semidefinite for all choices of x_1 and x_2.

Therefore, **K(x, z) is not guaranteed to be a valid kernel for all functions** $f(x)$.

3

Refer to the attached .ipynb file for the code. The answer for the most important five features is also included in the file.

4

$$y = w_0 + \sum_{i=1}^{n} w_i x_i + w_i x_i^2$$

Error function: $E_d = \frac{1}{2}\sum_j (y_j - y_j^*)^2$

To calculate this derivative, we need to consider each term in the sum separately. So we remove the summation. Calculating the partial derivative:

$$\frac{\partial E}{\partial w_i} = \frac{\partial \frac{1}{2}(y_j - y_j^*)^2}{\partial w_i} = 0$$

The terms involving y_j can be differentiated w.r.t. w_i :

$$\frac{\partial \frac{1}{2}(y_j - y_j^*)^2}{\partial w_i} = (y_j - y_j^*) \times \frac{\partial y_j}{\partial w_i}$$

The derivative of y_j w.r.t. w_i can be calculated by:

$$\frac{\partial y_j}{\partial w_i} = \frac{\partial (w_0 + w_i x_i + w_i x_i^2)}{\partial w_i} = x_i + x_i^2$$

This derivative is put back into the previous equation to obtain:

$$\frac{\partial E}{\partial w_i} = (y_j - y_j^*) \times (x_i + x_i^2)$$

The update rule for 1 term:

$$\delta w_i = -\eta \times \frac{\partial E}{\partial w_i} = -\eta \times (y_j - y_j^*) \times (x_i + x_i^2)$$

Thus, stochastic gradient weight update rule is:

$$w_i(new) = w_i(old) + \delta w_i = w_i(old) + \eta \sum_{ij} (y_j - y_j^*) \times (x_i + x_i^2)$$

5

Let's consider a Naive Bayes classifier with two binary features x_1 and x_2. For each feature, there are two possible values (0 and 1). The following parameters can be deduced:

$$x_1 : P(x_1 = 0|0), P(x_1 = 1|0), P(x_1 = 0|1), P(x_1 = 1|1)$$
$$x_2 : P(x_2 = 0|0), P(x_2 = 1|0), P(x_2 = 0|1), P(x_2 = 1|1)$$

Therefore, the total number of parameters for two features = 4 + 4 = 8

Let's now consider a Naive Bayes classifier with three binary features x_1, x_2 and x_3. The following parameters can be deduced:

$$x_1 : P(x_1 = 0|0), P(x_1 = 1|0), P(x_1 = 0|1), P(x_1 = 1|1)$$
$$x_2 : P(x_2 = 0|0), P(x_2 = 1|0), P(x_2 = 0|1), P(x_2 = 1|1)$$

$$x_3 : P(x_3 = 0|0), P(x_3 = 1|0), P(x_3 = 0|1), P(x_3 = 1|1)$$

Now, the total number of parameters for three features = 4 + 4 + 4 = 12

Let's now consider a Naive Bayes classifier with n binary features. A pattern can be observed from the above examples, 2 features create 8 parameters and 3 features create 12 parameters. Therefore, it can be deduced that for n features, $4n$ parameters can be obtained.