NYCU Pattern Recognition, Homework 4

411551005 徐浩哲

Part. 1 Coding

1. Implement K-fold data partitioning.

Q1

```python
def cross_validation(X_train, shuffle=True, k=5, random_state=35):
    n_samples = X_train.shape[0]
    indices = np.arange(n_samples)
    if shuffle:
        rstate = np.random.RandomState(random_state)
        rstate.shuffle(indices)
    n_splits = k
    fold_sizes = np.full(n_splits, n_samples // n_splits, dtype=int)
    fold_sizes[: n_samples % n_splits] += 1
    KFold = []
    current = 0
    for fold_size in fold_sizes:
        start, stop = current, current + fold_size

        # get_test_mask function content merged into cross_validation
        test_index = indices[start:stop]
        test_mask = np.zeros(n_samples, dtype=bool)
        test_mask[test_index] = True

        train_index = indices[np.logical_not(test_mask)]
        test_index = indices[test_mask]
        # error = [x for x in train_index if x in test_index]
        # print(len(error))
        KFold.append([train_index, test_index])
        current = stop
    return KFold
```
[5]

```python
kfold_data = cross_validation(x_train, k=10)
assert len(kfold_data) == 10 # should contain 10 fold of data
assert len(kfold_data[0]) == 2 # each element should contain train fold and validation fold
assert kfold_data[0][1].shape[0] == 700 # The number of data in each validation fold should equal to training data divieded by K
```
[6]

2. Set the kernel parameter to 'rbf' and do grid search on the hyperparameters C and gamma to find the best values through cross-validation. Print the best hyperparameters you found. Note that we suggest using K=5 for the cross-validation.

Q2

```python
def gridsearch(x, y, kfold_data, candidate_C, candidate_gamma):
    history = []
    max_acc = 0
    n_gamma = len(candidate_gamma)
    candidate = [(c, g) for c in candidate_C for g in candidate_gamma]
    tmp_acc = []

    for i, (c, g) in enumerate(candidate):
        avg_acc = 0

        for j, (train, test) in enumerate(kfold_data):
            clf = SVC(C=c, kernel='rbf', gamma=g)
            clf.fit(x[train], y[train])
            y_pred = clf.predict(x[test])
            acc = accuracy_score(y[test], y_pred)
            avg_acc += acc

        avg_acc /= len(kfold_data)
        tmp_acc.append(avg_acc)

        if avg_acc > max_acc:
            best_C = c
            best_gamma = g
            max_acc = avg_acc

        if (i + 1) % n_gamma == 0:
            history.append(tmp_acc)
            tmp_acc = []

    return np.asarray(history), (best_C, best_gamma)

candidate_C = np.logspace(-4, 4, num=9)
candidate_gamma = np.logspace(-4, 4, num=9)
history, best_parameters = gridsearch(x_train, y_train, kfold_data, candidate_C, candidate_gamma)
print(f'Best parameters (C, gamma): {best_parameters}')
```

✓ 6520.8s Python

```
Best parameters (C, gamma): (1.0, 0.0001)
```

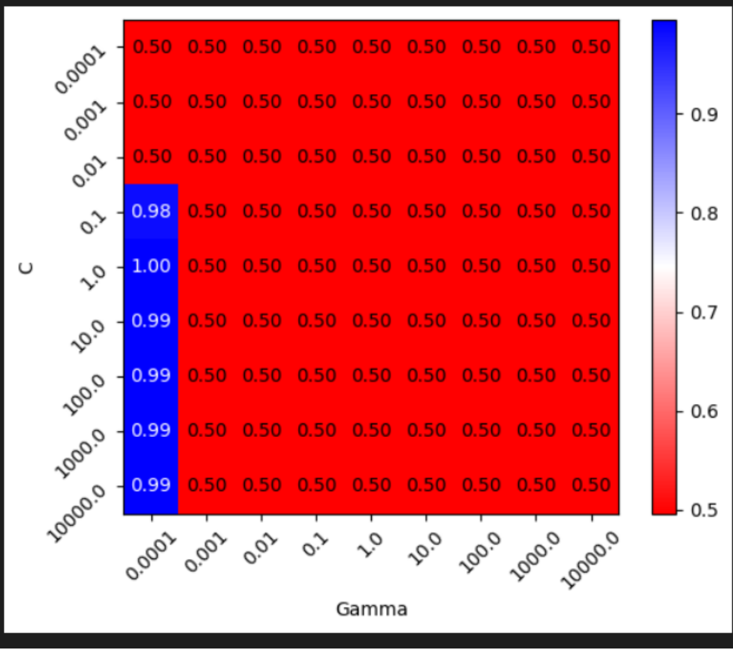
3. Plot the results of your SVM's grid search.

```python
from matplotlib.colors import LinearSegmentedColormap
cmap = LinearSegmentedColormap.from_list("custom", ["red", "white", "blue"])

plt.imshow(history, cmap= cmap)
plt.colorbar()

for i in range(history.shape[0]):
    for j in range(history.shape[1]):
        plt.text(j, i, f"{history[i, j]:.2f}", ha="center", va="center", color="black" if history[i,
plt.ylabel("C")
plt.xlabel("Gamma")
plt.yticks(np.arange(9), candidate_C, rotation = 45)
plt.xticks(np.arange(9), candidate_gamma, rotation = 45)
plt.show()
```

[9] ✓ 0.3s Python



4. Train your SVM model using the best hyperparameters found in Q2

```python
# Do Not Modify Below

best_model = SVC(C=best_parameters[0], gamma=best_parameters[1], kernel='rbf')
best_model.fit(x_train, y_train)

y_pred = best_model.predict(x_test)

print("Accuracy score: ", accuracy_score(y_pred, y_test))

# If your accuracy here > 0.9 then you will get full credit (20 points).
```

[10] ✓ 7.7s

```
Accuracy score:  0.995
```

Part. 2 Questions

1. Show that the kernel matrix $K = [k(x_n, x_m)]_{nm}$ should be positive semidefinite is the necessary and sufficient condition for $k(x, x')$ to be a valid kernel.

Necessity: (If $k(x, x')$ is a valid kernel, then K is positive semidefinite)
Let $k(x, x')$ be a valid kernel. By definition, a function $k(x, x')$ is a valid kernel if it satisfies the following conditions:

- $k(x, x') = k(x', x)$ (symmetry)
- $k(x, x')$ is a continuous function
- K, the kernel matrix, is positive semidefinite.

The first two properties follow from the definition of a kernel function. To show the third property, let's consider an arbitrary set of data points $\{x_1, x_2, \dots, x_n\}$ and the corresponding kernel matrix K. We will show that for any vector $v \in R^n, v^T * K * v \geq 0$.

By definition, the elements of the kernel matrix K are given by $K_{nm} = k(x_n, x_m)$. Let $v = [v_1, v_2, \dots, v_n]^T$ be an arbitrary vector in R^T. Then,

$$v^T * K * v = \sum_{i=1}^{n} \sum_{j=1}^{n} v_i * k(x_i, x_j) * v_j$$

Since $k(x, x')$ is a valid kernel, we can express it as an inner product in some feature space:

$$k(x, x') = \varphi(x)\text{^}T * \varphi(x')$$

Now, let's rewrite $v^T * K * v$ using the inner product representation:

$$v^T * K * v = \sum_{i=1}^{n} \sum_{j=1}^{n} v_i * \varphi(x_i)^T * \varphi(x_j) * v_j$$

Let $\Psi_i = \sqrt{v_i} * \varphi(x_i)$

$$v^T * K * v = \sum_{i=1}^{n} \sum_{j=1}^{n} \Psi_i^T * \Psi_j = ||\Psi_1 + \Psi_2 + \dots + \Psi_n||^2 \geq 0$$

Thus, the kernel matrix K is positive semidefinite.

Sufficiency: (If K is positive semidefinite, then $k(x, x')$ is a valid kernel)
Let K be a positive semidefinite kernel matrix. By definition, K is symmetric, and for any vector $v \in R^n, v^T * K * v \geq 0$. Our goal is to show that there exists a function $\varphi: R^d \rightarrow R^D$, such that for any $x, x' \in R^d, k(x, x') = \varphi(x)^T * \varphi(x')$

Since K is positive semidefinite, we can apply an eigen decomposition to K, such that $K = Q * \Lambda * Q^T$, where Q is an orthogonal matrix containing the eigenvectors, and Λ is a diagonal matrix containing the eigenvalues. Note that all eigenvalues are non-negative since K is positive semidefinite.

Now, we can express K as follows:

$$K = Q * \Lambda^{(1/2)} * \Lambda^{(1/2)} * Q^T$$

Let $\Phi = Q * \Lambda^{(1/2)}$, then $K = \Phi * \Phi^T$. By definition, Φ contains the feature vectors:

$$\Phi = [\varphi(x_1), \varphi(x_2), \dots, \varphi(x_n)]$$

Thus, for any x, x'

2. Given a valid kernel $k_1(x, x')$, explain that $\mathrm{k}(x, x') = exp\big(k_1(x, x')\big)$is also a valid kernel.

To show that $\mathrm{k}(x, x') = exp\big(k_1(x, x')\big)$ is a valid kernel, given that k_1(x, x') is a valid kernel, we can use the Taylor series expansion of the exponential function.

The Taylor series expansion of exp(x) is given by:

$$\exp(x) = \sum (x^n/n!)\,, for\ n = 0, 1, 2, \dots ..$$

Therefore, the exponential of a kernel function can be written as:

$$\exp\big(k_1(x, x')\big) = \sum (k_1(x, x')^n/n!)\,, for\ n = 0, 1, 2, \dots ..$$

Since k_1(x, x') is a valid kernel, it follows that $k_1(x, x')^n$ is also a valid kernel for any non-negative integer n. This is because taking a power of a valid kernel is equivalent to taking multiple products of the kernel, and the product of valid kernels is a valid kernel.

Moreover, the division by n! does not affect the validity of the kernel, as n! is just a scalar factor. Thus, each term in the Taylor series expansion is a valid kernel.

The sum of valid kernels is also a valid kernel. Therefore, since the Taylor series expansion is a sum of valid kernels, $\mathrm{k}(x, x') = exp\big(k_1(x, x')\big)$ is a valid kernel.

3. Prove functions are or are not valid kernels.
   - $\mathrm{k}(\mathrm{x}, \mathrm{x}') = \mathrm{k}_1(\mathrm{x}, \mathrm{x}') - 1$

     We can see that $k(x, x')$ is symmetric because $k_1(x, x')$is symmetric:

$$k(x, x') = k_1(x, x') - 1 = k_1(x', x) - 1 = k(x', x)$$

However, we need to check if $k(x, x')$ is positive semidefinite. Recall that a kernel K is positive semidefinite if for any set of points $x_1, \ldots, x_n$ and any real coefficients $c_1, \ldots, c_n$, the following inequality holds:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0$$

Now consider $k(x, x') = k_1(x, x') - 1$. Let's plug this into the inequality:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \left(k_1(x_i, x_j) - 1\right)$$

Rearrange the terms:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k_1(x_i, x_j) - \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j$$

We know the first term is non-negative since $k_1(x, x')$ is a valid kernel. However, we can't guarantee that the second term is non-negative.

Consider the following counterexample:

$$k_1(x, x') = x * x'$$

In this case, $k(x, x') = x * x' - 1$

Now consider the points $x_1 = 1$ and $x_2 = -1$.

Let $c_1 = c_2 = 1$

We have:

$$K(x_1, x_1) = k(1,1) = 0$$
$$K(x_1, x_2) = k(1, -1) = -2$$
$$K(x_2, x_1) = k(-1,1) = -2$$
$$K(x_2, x_2) = k(-1, -1) = 0$$

The resulting matrix K is:

$$\begin{matrix} 0 & -2 \\ -2 & 0 \end{matrix}$$

The eigenvalues of this matrix are 2 and -2, and since one of them is negative, this matrix is not positive semidefinite.

Thus, $k(x, x') = k_1(x, x') - 1$ is not guaranteed to be a valid kernel because it is not positive semidefinite for all valid kernels

$k_1(x, x')$.

- $k(\mathrm{x}, \mathrm{x}') = \mathrm{k}_1(\mathrm{x}, \mathrm{x}')^2 + \exp\left(\left||\mathrm{x}|\right|^2\right) * \exp\left(\left||\mathrm{x}'|\right|^2\right)$

Symmetry:
Since $k_1(x, x')$ is symmetric, we have:

$$k_1(x, x') = k_1(x', x)$$

Now, we need to show that $k(x, x') = k(x', x)$. Let's plug in the values:

$$k(\mathrm{x}, \mathrm{x}') = \mathrm{k}_1(\mathrm{x}, \mathrm{x}')^2 + \exp\left(\left||\mathrm{x}|\right|^2\right) * \exp\left(\left||\mathrm{x}'|\right|^2\right)$$

$$\mathrm{k}(\mathrm{x}', \mathrm{x}) = \mathrm{k}_1(\mathrm{x}', \mathrm{x})^2 + \exp\left(\left||\mathrm{x}'|\right|^2\right) * \exp\left(\left||\mathrm{x}|\right|^2\right)$$

Since $k_1(x, x') = k_1(x', x)$, we can see that $k_1(x, x')^2 = k_1(x', x)^2$. Also, $\exp\left(\left||x|\right|^2\right) * \exp\left(\left||x'|\right|^2\right) = \exp\left(\left||x'|\right|^2\right) * \exp\left(\left||x|\right|^2\right)$. Therefore, $\mathrm{k}(\mathrm{x}, \mathrm{x}') = \mathrm{k}(\mathrm{x}', \mathrm{x})$, and the kernel is symmetric.

Positive Semidefinite:
Recall that a kernel K is positive semidefinite if for any set of points $x_1, \ldots, x_n$ and any real coefficients $c_1, \ldots, c_n$, the following inequality holds:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0$$

We can rewrite the kernel as a sum of two kernels:

$$k(x, x') = k_1(x, x')^2 + \exp\left(\left||x|\right|^2\right) * \exp\left(\left||x'|\right|^2\right)$$

$$= k_2(x, x') + k_3(x, x')$$

Where:

$$k_2(x, x') = k_1(x, x')^2$$

$$\mathrm{k}_3(\mathrm{x}, \mathrm{x}') = \exp\left(\left||\mathrm{x}|\right|^2\right) * \exp\left(\left||\mathrm{x}'|\right|^2\right)$$

We know that the sum of two valid kernels is also a valid kernel. So, we need to prove that both $k_2(x, x') and k_3(x, x')$ are valid

kernels.
For $k_2(x, x')$, we have:
$$k_2(x, x') = k_1(x, x')^2$$
Since $k_1(x, x')$is a valid kernel, $k_1(x, x')^2$ is also a valid kernel, as the square of a valid kernel is also positive semidefinite.
For $k_3(x, x')$, we have:
$$k_3(x, x') = \exp\left(||x||^2\right) * \exp\left(||x'||^2\right)$$

Notice that $k_3(x, x')$ can be written as a product of two functions:
$$k_3(x, x') = f(x) * f(x')$$
Where:
$$f(x) = \exp\left(||x||^2\right)$$
The product of two functions in this form is a valid kernel, known as the "separable kernel."

- $k(x, x') = k_1(x, x')^2 + \exp\big(k_1(x, x')\big) - 1$

Symmetry:
Since $k_1(x, x')$ is symmetric, we have:
$$k_1(x, x') = k_1(x', x)$$
Now, we need to show that $k(x, x') = k(x', x)$. Let's plug in the values:
$$k(x, x') = k_1(x, x')^2 + \exp\big(k_1(x, x')\big) - 1$$
$$k(x', x) = k_1(x', x)^2 + \exp\big(k_1(x', x)\big) - 1$$

Since $k_1(x, x') = k_1(x', x)$, we can see that $k_1(x, x')^2 = k_1(x', x)^2$ and $\exp\big(k_1(x, x')\big) = \exp\big(k_1(x', x)\big)$.
Therefore, $k(x, x') = k(x', x)$, and the kernel is symmetric.

Positive Semidefinite:
Recall that a kernel K is positive semidefinite if for any set of

points $x_1, \ldots, x_n$ and any real coefficients $c_1, \ldots, c_n$, the following inequality holds:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j K(x_i, x_j) \geq 0$$

We can rewrite the kernel as a sum of three kernels:

$$k(x, x') = k_1(x, x')^2 + \exp(k_1(x, x')) - 1$$

$$= k_2(x, x') + k_3(x, x') - k_4(x, x')$$

Where:

$$k_2(x, x') = k_1(x, x')^2$$

$$k_3(x, x') = \exp(k_1(x, x'))$$

$$k_4(x, x') = 1$$

We know that the sum of valid kernels is also a valid kernel. So, we need to prove that $k_2(x, x')$, $k_3(x, x')$, and $-k_4(x, x')$ are valid kernels.

For $k_2(x, x')$, we have:

$$k_2(x, x') = k_1(x, x')^2$$

Since $k_1(x, x')$ is a valid kernel, $k_1(x, x')^2$ is also a valid kernel, as the square of a valid kernel is also positive semidefinite.

For $k_3(x, x')$, we have:

$$k_3(x, x') = \exp(k_1(x, x'))$$

Since $k_1(x, x')$ is a valid kernel, $\exp(k_1(x, x'))$ is also a valid kernel, as the exponential of a valid kernel is positive semidefinite.

For $-k_4(x, x')$, we have:

$$-k_4(x, x') = -1$$

This kernel is not positive semidefinite because the constant kernel with value -1 is not guaranteed to be positive semide

4. Consider the optimization problem

$$\text{minimize } f(x) = (x-2)^2$$

$$\text{subject to } g(x) = (x-3)(x-1) - 3 \leq 0$$

1. The Lagrangian function $L(x, \lambda)$ is given as:

$$L(x, \lambda) = f(x) + \lambda * g(x) = (x - 2)^2 + \lambda * \left((x - 3)(x - 1) - 3\right)$$

2. Now, let's find the gradient of $L(x, \lambda)$ with respect to x:
$$\nabla_x L(x, \lambda) = dL(x, \lambda)/dx = 2(x - 2) + \lambda * (2x - 4)$$

3. When $\nabla_x L(x, \lambda) = 0$
$$0 = 2(x - 2) + \lambda * (2x - 4)$$
Rearranging the terms:
$$\lambda * (2x - 4) = 2(2 - x)$$
$$2x - 4 = 2(2 - x)/\lambda$$
Now, solving for x in terms of λ:
$$x = (4 + 2/\lambda) / 2$$
4. Substitute this expression for x back into the Lagrangian L(x, λ) to obtain the dual function L(λ):
$$L(\lambda) = \left((4 + 2/\lambda)/2 - 2\right)^2 + \lambda * \left(\left((4 + 2/\lambda)/2 - 3\right)\left((4 + 2/\lambda)/2 - 1\right) - 3\right)$$
Now, the dual problem is to maximize the dual function L(λ), subject to λ ≥ 0:

maximize L(λ)
subject to λ ≥ 0