Assignment 20 Solutions

1. What is the underlying concept of Support Vector Machines ?

Ans: The underlying concept of Support Vector Machines (SVMs) is to find the optimal hyperplane that maximizes the margin between two or more classes of data. By using a technique known as the kernel trick, SVMs are able to project data into higher-dimensional space to find a separating hyperplane that maximizes the margin between two classes of data. SVMs are powerful tools for classification, regression, outlier detection, and other machine learning tasks.

2. What is the concept of a support vector ?

Ans: A support vector is a vector in a vector space that, when added to the data set, can help optimize the performance of a supervised learning algorithm. Support vectors are used in machine learning algorithms such as support vector machines (SVMs) and support vector regression (SVR). The concept of a support vector is to find a hyperplane that best divides a dataset into classes. This hyperplane is known as the decision boundary, and the support vectors are the data points that are closest to it.

3. When using SVMs, why is it necessary to scale the inputs ?

Ans: Scaling the inputs is necessary when using SVMs because the algorithm is sensitive to the relative size of the inputs. Without scaling, features with large values will dominate the optimization process and the model will not be able to accurately capture the underlying relationship between the features and the target. Scaling the inputs ensures that all features are given the same weight in the optimization process, allowing the model to capture more subtle patterns.

4. When an SVM classifier classifies a case, can it output a confidence score? What about a percentage chance ?

Ans: Yes, an SVM classifier can output a confidence score. However, it cannot output a percentage chance as this type of information is not provided by the model.

5. Should you train a model on a training set with millions of instances and hundreds of features using the primal or dual form of the SVM problem ?

Ans: It depends on the complexity of the data set and the size of the training set. Generally, the dual form of the SVM problem is more efficient and can handle datasets with a large number of features and large number of instances better than the primal form. However, if the data set is too large or too complex, it might be better to use the primal form.

6. Let's say you've used an RBF kernel to train an SVM classifier, but it appears to underfit the training collection. Is it better to raise or lower (gamma)? What about the letter C ?

Ans: It is better to raise gamma and lower C. Increasing gamma will make the decision boundary more complex, allowing for more complex patterns to be learned by the SVM classifier. Reducing C will increase the regularization strength, which can help to prevent overfitting.

7. To solve the soft margin linear SVM classifier problem with an off-the-shelf QP solver, how should the QP parameters (H, f, A, and b) be set ?

Ans: The QP parameters (H, f, A, and b) should be set as follows:

H: The Hessian matrix - a square matrix of size nxn, where n is the number of features. The entries of H are the inner products of the feature vectors for each training example.

f: A vector of size n (the number of features) with the coefficients of the linear objective function.

A: A matrix of size n x m (where m is the number of constraints) with the coefficients of the linear constraints.

b: A vector of size m (the number of constraints) with the right-hand side of the linear constraints.

8. On a linearly separable dataset, train a LinearSVC. Then, using the same dataset, train an SVC and an SGDClassifier. See if you can get them to make a model that is similar to yours ?

Ans: Yes, it is possible to get the SVC and SGDClassifier to make a model that is similar to the LinearSVC. To do this, you will need to adjust the hyperparameters of each model, such as the regularization parameter C, the kernel type, and the margin width. You can also adjust other parameters such as the learning rate and the number of iterations. Once all of these parameters are tuned correctly, the models should produce similar results.

9. On the MNIST dataset, train an SVM classifier. You'll need to use one-versus-the-rest to assign all 10 digits because SVM classifiers are binary classifiers. To accelerate up the process, you might want to tune the hyperparameters using small validation sets. What level of precision can you achieve ?

Ans: The level of precision you can achieve with an SVM classifier on the MNIST dataset will depend on the hyperparameters you use. Generally, SVM classifiers can achieve high levels of accuracy on the MNIST dataset; in some cases, up to 99.7% accuracy has been reported. Tuning the hyperparameters for the SVM classifier can help you achieve even better accuracy.

10. On the California housing dataset, train an SVM regressor ?

Ans:

In [1]:

```
1 from sklearn.datasets import fetch_california_housing
```

In [2]:

```
housing = fetch_california_housing()
X = housing.data
y = housing.target
```

In [3]:

```
from sklearn.model_selection import train_test_split
```

In [4]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [5]:

```
from sklearn.preprocessing import StandardScaler
```

In [6]:

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [7]:

```
from sklearn.svm import SVR
```

In [8]:

```
svm_reg = SVR(kernel="linear")
svm_reg.fit(X_train_scaled, y_train)
```

Out[8]:

```
SVR(kernel='linear')
```

Note:- In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [10]:

```
from sklearn.metrics import mean_squared_error
```

In [11]:

```
y_pred = svm_reg.predict(X_test_scaled)
mse = mean_squared_error(y_test, y_pred)
print("MSE:", mse)
```

```
MSE: 0.5792242326685152
```

In [ ]:

```
1
```