Experiment No. 7 - Implement and test Multiclass SVM classifier

Support Vector Machines

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

- **Data** - Digit Dataset
- **Task** - Multi Class (0-9 digit)
- **Mathematical Model** - SVM
- **Loss Function**
- **Learning Algorithm**
- **Model Evaluation**

Algorithm

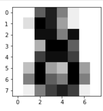
1. Import library and load data
2. SVM Classifier
3. Training the model
4. Prediction
5. Evaluation

1. Import library and load data

In [1]:
```
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn import svm
from IPython.display import Image

#Load the digits dataset
digits = datasets.load_digits()
```

In [2]:
```
#Display the first digit
plt.figure(1, figsize=(3, 3))
plt.imshow(digits.images[-1], cmap=plt.cm.gray_r, interpolation='neare
st')
plt.show()
```



In [3]:
```
print(digits.data)
print(digits.target)
```

```
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
[0 1 2 ... 8 9 8]
```

2. SVM Classifier

In [4]:
```
from sklearn.model_selection import train_test_split

classifier = svm.SVC()
classifier = svm.SVC(gamma=0.001, C=100)
```

3. Training the model

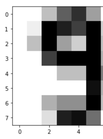
In [5]:
```
#x,y = digits.data[:-10], digits.target[:-10]
X = digits.data
Y = digits.target
x_train, x_test, y_train, y_test = train_test_split(X, Y, train_size =
0.9)
classifier.fit(x_train,y_train)
```

Out[5]:
```
SVC(C=100, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rb
f',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

4. Prediction

In [6]:
```
print(classifier.predict(digits.data[-5].reshape(1, -1)))
plt.imshow(digits.images[-5], cmap=plt.cm.gray_r, interpolation='neare
st')
plt.show()
```

```
[9]
```



5. Evaluation

In [7]:
```
from sklearn.metrics import accuracy_score

y_pred = classifier.predict(x_test)

print(accuracy_score(y_test, y_pred))
```

```
0.9722222222222222
```

Questions

1. Why Support vector machines are called as Kernel Machines?

Ans:- SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.

Due to the technique used by SVM it is called as Kernel Machines.

2. List the features of Support Vector Machine (SVM). Compare SVM classifier with logistic regression classifier?

Ans:- **Features of Support Vector Machines**

- Accuracy is higher
- More efficient because it uses a subset of training points
- Choose data points nearest to the hyperplane
- Works well on smaller cleaner datasets
- Uses kernel transformation

Difference between SVM classifer and Logistic regression classifier

- SVM tries to finds the "best" margin (distance between the line and the support vectors) that separates the classes and this reduces the risk of error on the data, while logistic regression does not, instead it can have different decision boundaries with different weights that are near the optimal point.
- SVM works well with unstructured and semi-structured data like text and images while logistic regression works with already identified independent variables.
- SVM is based on geometrical properties of the data while logistic regression is based on statistical approaches.
- The risk of overfitting is less in SVM, while Logistic regression is vulnerable to overfitting.

```
**3. Compare SVM and Neural Networks. **
```

Ans:- SVM tends to create one of more hyperplanes to separate out the data clusters. Hyperplanes are strictly linear when kernel trick not in use. Strictly linear makes SVM behaves like a neural without an activation function.

With the kernel trick, SVMs are roughly equivalent to feed forward neural networks with a non-linear activation function. Therefore, the difference between a SVM and a NN is in how they decide what these parameters should be set to.

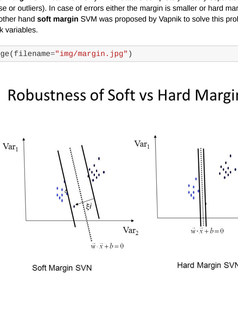
4. Explain slack variable, hard margin and soft margin.

Ans:- **Slack variables** are introduced to allow certain constraints to be violated. That is, certain training points will be allowed to be within the margin. We want the number of points within the margin to be as small as possible, and of course we want their penetration of the margin to be as small as possible.

Hard margin SVM can work only when data is completely linearly separable without any errors (noise or outliers). In case of errors either the margin is smaller or hard margin SVM fails. On the other hand **soft margin** SVM was proposed by Vapnik to solve this problem by introducing slack variables.

In [8]:
```
Image(filename="img/margin.jpg")
```

Out[8]:



5. State limitations of SVM classifier.

- Training time is higher for larger datasets.
- Less effective on noisier datasets with overlapping classes.
- In cases where number of features for each data point exceeds the number of training data sample, the SVM will under perform?
- As the support vector classifier works by putting data points, above and below the classifying hyper plane there is no probabilistic explanation for the classification.

References

[1] - https://pythonprogramming.net/support-vector-machine-svm-example-tutorial-scikit-learn-python/

Author Name:- Hemant Ghuge
LinkedIn:- https://www.linkedin.com/in/hemantghuge/
GitHub:- https://github.com/HemantGorakshGhuge