Codes and other relevant explanations for supervised learning (Part 1)

GitHub link for the code:

https://github.com/Sabyasachi123276/ML-using-Python/blob/main/SVM(different_kernels)_KNN_NaiveBayes_SupervisedLearning(Part_1).ipynb

Dataset Description:

The Iris dataset was used in R.A. Fisher's classic 1936 paper. It can also be found in the UCI Machine Learning Repository. It includes three iris species with 50 samples each: Iris setosa, Iris versicolor, and Iris virginica, as well as some properties of each flower. Iris Setosa has index values of 0-49, Iris Versicolor has index values of 50-99, and Iris Virginica has index values of 100-149.

NumPy :

- It is a numeric Python module that provides fast math functions for calculations.
- It is used to read data in NumPy arrays and for manipulation purposes.

Pandas :

- It is used to read and write different files.
- Data manipulation can be done easily with data frames.

scikit-learn / sklearn:

- In Python, sklearn is a machine learning package that includes a lot of ML algorithms.
- Here, we are using some of its modules like train_test_split, DecisionTreeClassifier, and accuracy_score.

Matplotlib:

Matplotlib is a visualization library in Python. It is built on NumPy arrays and consists of several plots like line, bar, scatter, histogram, etc.

Pyplot:

Pyplot makes Matplotlib work like Matlab.

Seaborn:

Seaborn is a library mainly used for statistical plotting in Python. It is built on top of Matplotlib and provides beautiful default styles and color palettes to make statistical plots more attractive.

iloc() function: The Pandas library in Python provides the iloc() function, which is a widely used method for data analysis and manipulation. It stands for "integer location" and is primarily used for accessing and retrieving data from Pandas DataFrame objects using integer-based indexing.

Standard Scaler: It performs the normalization operation.

scaler.fit(): Similar to fitting a model, we can fit our scaling to the data using the fit method on the training set. This simply estimates the mean and standard deviation.

scaler.transform(): To apply the scaling, we can use the transform method, which will subtract the mean and divide by the standard deviation.

GaussianNB: Gaussian Naive Bayes (GaussianNB) can perform online updates to model parameters via partial_fit.

SVC (Support Vector Classifier): SVC is a specific implementation of the Support Vector Machine algorithm that is designed specifically for classification tasks. In other words, SVC is an SVM used for classification. It seeks to find the hyperplane that best separates the data points into different classes.

KNeighborsClassifier: KNeighborsClassifier implements the k-nearest neighbors vote.

classifier.fit(): The learning algorithm / estimator instance is first fitted to the model, i.e., it must *learn* from the model. This is done by passing our training set to the fit method.

classifier.predict(): The predict() will perform a prediction for each test instance, and it usually accepts only a single input.

confusion_matrix: Confusion matrix represents the prediction summary in matrix form. It shows how many predictions are correct and incorrect per class.

seaborn.heatmap() / sns.heatmap(): Plot rectangular data as a color-encoded matrix.

Parameters

(i) data: Here, the data input is the ‘result’.

(ii) annot: bool or rectangular dataset, optional.

If True, write the data value in each cell. If an array-like object has the same shape as data, then use this to annotate the heatmap instead of the data. Note that DataFrames will match on position, not index.

In this example, in each confusion matrix, correctly and incorrectly classified test sample numbers are shown.

(iii) fmt: string formatting code to use when adding annotations.

G or g (General Format) suffix in Python: It can be interpreted as the total number of digits before the decimal; otherwise, it represents the total number of zeros after the decimal point but before the significant digits.

In this example, fmt = 'g' is used.

(iv) xticklabels, yticklabels: Here, sequentially, three classes, 'Setosa', 'Versicolor', and 'Virginica' are marked.

classification_report: Build a text report showing the main classification metrics.

(i) Precision: TP / (TP + FP)

(ii) Recall / Sensitivity: TP / (TP + FN)

(iii) F1 Score = (2 * Precision * Recall) / (Precision + Recall)

True Positive (TP): A true positive is an outcome where the model correctly predicts the positive class.

True Negative (TN): A true negative is an outcome where the model correctly predicts the negative class.

False Positive (FP): A false positive is an outcome where the model incorrectly predicts the positive class.

False Negative (FN): A false negative is an outcome where the model *incorrectly* predicts the *negative* class.

Support: Support is a measure of the number of times an item set appears in a dataset. Here, the item set is a test data sample.

Macro Average: Average the precision, recall, and F1 scores across all classes to get the final macro-averaged precision, recall, and F1 scores, respectively. You may do the hand calculations and cross-check the output you get.

Weighted Average:

Weighted Average of Precision = [Sum(Individual Class Precision * Individual Class Support)] / Total Support.

In a similar way, you can do it for the rest of the recall and F1 score. You may do the hand calculations and cross-check the output you get.

accuracy_score: In Python, the accuracy_score function of the sklearn.metrics package calculates the accuracy score for a set of predicted labels against the true labels.

Accuracy: (TP + TN) / (TP + TN + FP + FN)

The IRIS data used in the example has a total of 150 data samples that comprise 50 from each class of IRIS Setosa, IRIS Virginica, and IRIS Versicolor.

During the training and test split, 70% was used for training and 30% for testing purposes.

30% of 150 is 45. That is the reason the total support count is 45.

For instance, depending on the output, among the 45, there can be 11 Iris Setosa, 17 Iris Virginica, and 17 Iris Versicolor. The same number will reflect during the prediction evaluations in the confusion matrix.