Python based machine learning using SKLearn

Zak Rowland

SKLearn is a machine learning library that you can use in Python.

The most popular dataset to understand how to use machine learning techniques is Fishers Iris dataset. Fischer's Iris dataset consists of **three classes**

1. Iris setosa
2. Iris Versicolour
3. Iris Virginica

It consists of **four different dimensions** of data. Each feature or dimension of data provides additional information. However, if there are too many dimensions of data are provided, you might get **curse of dimensionality**.

1. Sepal length
2. Sepal width
3. Petal length
4. Petal width

0. Import SKLearn and other libraries. You might need to install them as well.

    a) On command line, run the following.

        i. pip install sklearn

        ii. Run pip install matplotlib

        iii. Run pip install numpy

    b) In your Python file, you will need to include these libraries. You will need additional libraries for instance, you'll also need to import the LDA classifier. It'll error out, so you'll know if you're missing something.

    c) In your python file,

        i. import matplotlib as plt

        ii. from sklearn import datasets

        iii. from sklearn.neighbors import KNeighborsClassifier

        iv. from sklearn.cross_validation import train_test_split

1. Import the iris setosa dataset. This dataset is from Fischer. The link is for reference and information only.https://archive.ics.uci.edu/ml/datasets/iris Access the data through the command below.

    a) iris = datasets.load_iris()

b) Here is the usage information for datasets. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_iris.html

c) The iris.target will become your labels or your y axis in the step below. The iris.data will become your X data in the step below.

```
X = iris.data

y = iris.target
```

2. Use the KNN classifier to classify one dimension of data, as well as four dimensions of data. For the single dimension, use just the sepal length. See below for how to use KNN classifier in Python using SKLearn.

```
#This line will split your dataset for training and test

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20, shuffle=False)


#Call the actual classifier - In this case KNN. You can also import and call other classfiers.

classifier = KNeighborsClassifier(n_neighbors=7)


#Have the classfier fit the data

classifier.fit(X_train,y_train)


#Predict the data

y_pred = classifier.predict(X_test)
```

a) Provide screenshot of the precision, accuracy, and F1 scores for the single dimension of data. You can get these by calling:

Print(classifier_report(y_test,y_predict))

```
Report for single dimension:
C:\Users\zakro\AppData\Local\Programs\Python\Python38-32\lib\site-packages\sklearn\metrics\_classification.py:1221: U
ndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `z
ero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.30      0.46        30

    accuracy                           0.30        30
   macro avg       0.50      0.15      0.23        30
weighted avg       1.00      0.30      0.46        30
```

b) Provide screenshot of the precision, accuracy, and F1 scores for four dimensions of data. You can get these by calling:

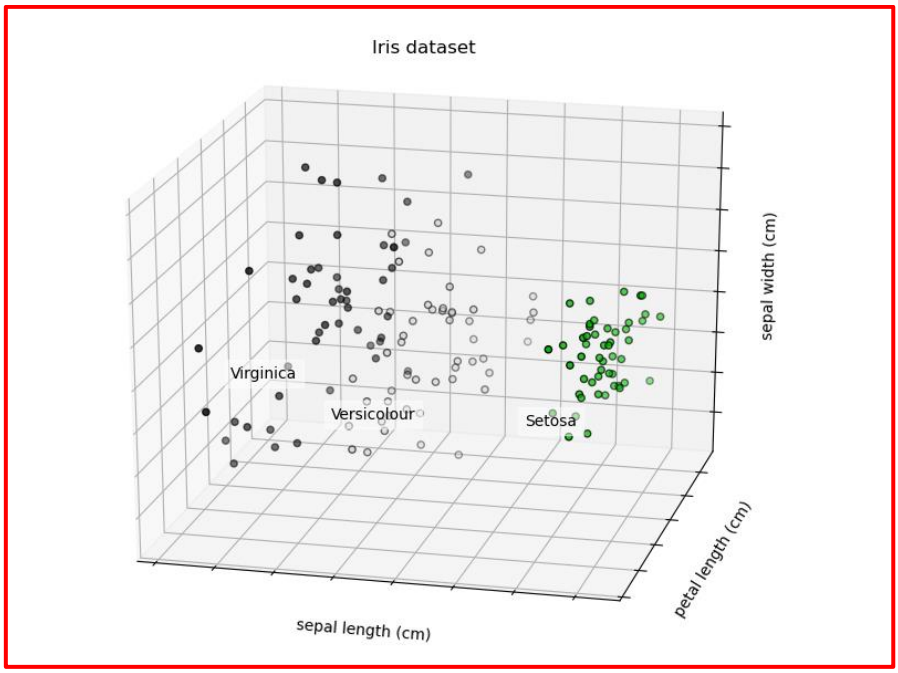
Print(classifier_report(y_test,y_predict))

```
Report for all four dimensions:
C:\Users\zakro\AppData\Local\Programs\Python\Python38-32\lib\site-packages\sklearn\metrics\_classification.py:1221: U
ndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `z
ero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.80      0.89        30

    accuracy                           0.80        30
   macro avg       0.50      0.40      0.44        30
weighted avg       1.00      0.80      0.89        30
```

c) Compare the results between one dimension of data and four dimension of data. What can you conclude about the dimensionality of data with respect to performance of classifier in 2a and 2b?

The precision wasn't affected however using four dimensions increased recall and f1-score. This means that the number of correctly identified positives (accuracy) increased.

3. Plot a 3D KNN cluster plot for the Iris setosa dataset (three classes and four dimensions)- Neatly title the axes and graph name, as well as provide a legend. Provide both the plot and code. You may need to import an additional library.



4. Provide the confusion matrix for the three class and for dimension case.

a) As an example, you may call the confusion matrix as follows. Test is your test dataset and predicted is your predicted. You may have named it differently.

confusion_matrix(test,predicted)

```
Confusion matrix:
[[ 0  0]
 [ 6 24]]
```

b) Please explain what the confusion matrix is.

The confusion matrix shows the accuracy of the classification by providing the number of true negatives, false positives, false negatives, and true positives. In my matrix, there are 6 false negatives and 24 true positives.

5. Now replace the KNN classifier and use the LDA classifier to classify the three classes and four dimensions of data. Reduce this to 2 dimensions. You can do this by replacing this line:

#Call the actual classifier - In this case KNN. You can also import and call other classfiers.

classifier = KNeighborsClassifier(n_neighbors=7)

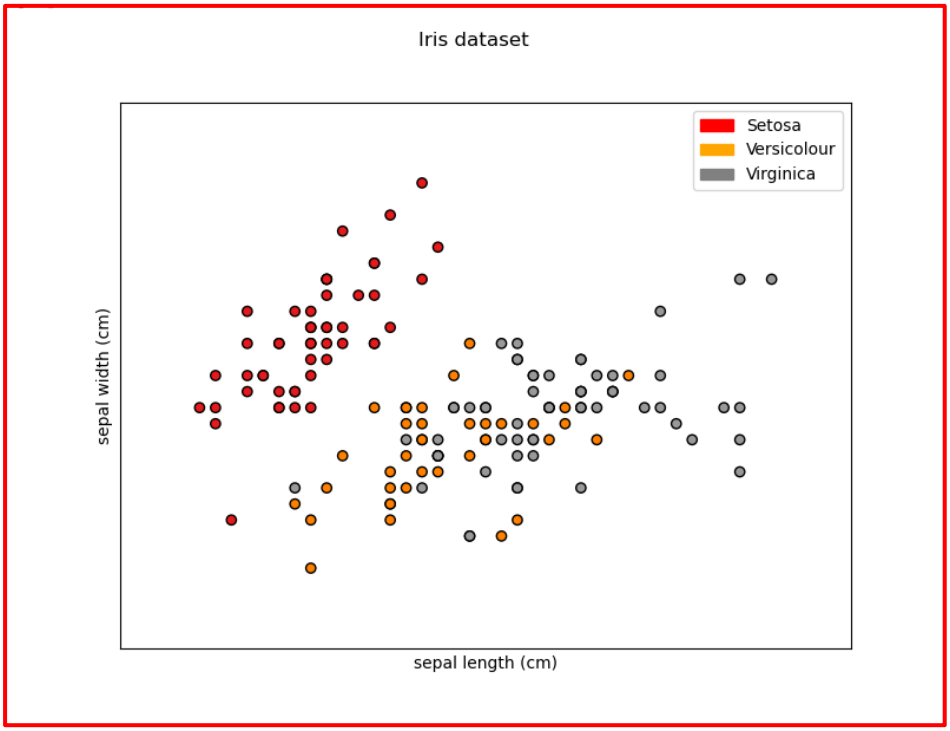
With

classifier = LinearDiscriminantAnalysis(n_components=2)

6. Linear Discriminant can be used to reduce the dimensions of data. Please describe two reasons why it might be useful to reduce the dimensions of data.

Two reasons to reduce the dimensions are to avoid the curse of dimensionality (too many dimensions,) and it reduces execution time and storage space. It can also make the data easier to visualize like in a 3D graph.

7. Provide the 2D graph of the Iris Setosa dataset using LDA classifier and code. Please put the title, axes, and graph name, as well as provide a legend.



8. Collect the same precision, accuracy, and F1 scores. Compare the LDA output to the KNN classifier. Is the performance of LDA similar to KNN?

```
Report for LDA 2 dimensions:
C:\Users\zakro\AppData\Local\Programs\Python\Python38-32\lib\site-packages\sklearn\metrics\_classification.py:1221: U
ndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `z
ero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         0
           2       1.00      0.90      0.95        30

    accuracy                           0.90        30
   macro avg       0.50      0.45      0.47        30
weighted avg       1.00      0.90      0.95        30

Confusion matrix:
[[ 0  0]
 [ 3 27]]
```

The performance is very similar, however LDA has slightly higher recall and f1 scores when compared to the four-dimensional KNN. As seen in the confusion matrix, there was three more true positive and three less false negatives. The accuracy is higher.

9. What is the difference between regression and classifier?

Classification is used to predict the output based on classes/labels for the data, and regression is used to predict a quantity or value from a continuous dataset.

10. What is the difference between supervised and unsupervised learning? Are LDA and KNN supervised or unsupervised?

Supervised learning is providing a set of input variables and specifying the output variables that are expected after going through the algorithm. Unsupervised learning is providing a set of input variables and not knowing the output variables, so the algorithm comes up with output based on the data given. LDA and KNN are both examples of supervised learning.

11. The classifiers we talked about have no contextual information. What is contextual information, and what model from lecture could be used to add contextual information?

Contextual information is information about the data in the dataset given. The classifiers used above rely only on the numbers, they don't care about the context. A neural network could be used to add contextual information.

12. We looked at KNN and LDA. Now using your existing code, substitute the classifier and fill in the table below.

a) To measure the time to algorithm you will need to import time

import time

b) Before the algorithm starts, get the current time.

start = time.time()

c) After the algorithm finishes, get the current time and subtract it.

end = time.time()

elapsed = end - start

Note: There are more accurate methods but this is simple and should still give you a good comparison.

Classifier	F1-score	Time to run algorithm
KNN	0.89	0.002998114s

LDA	0.95	0.0019989014s
Gaussian Naive Bayes	0.97	0.0019989014s
MLPC	0.57	0.2148675919s
AdaBoost	0.85	0.0939414501s

13. Based on the table that you generated, which method performed best?

Gaussian Naive Bayes has the highest F1-score and fastest time.

Please submit the following:

1) Paste your graphs and write your answers into this document and highlight your answers in red.

2) Provide code that you used for each part (Just submit the file).