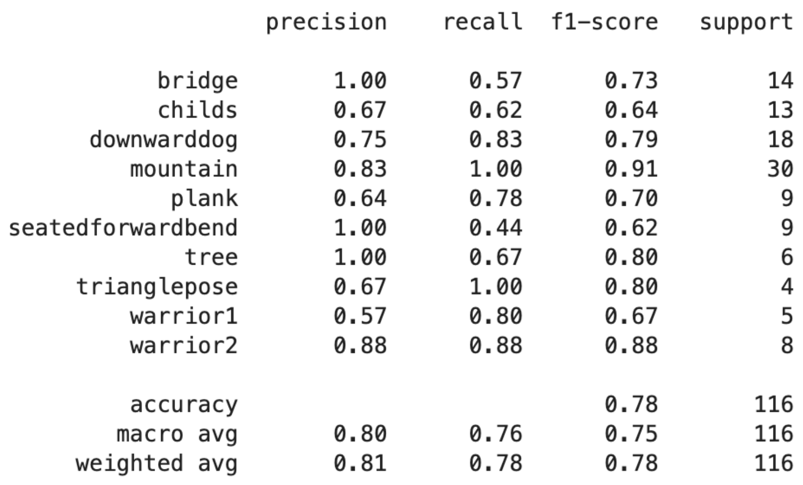
3. Implement a kernel density estimate (KDE) naive Bayes classifier and compare its performance to the Gaussian naive Bayes classifier. Recall that KDE has kernel bandwidth as a free parameter -- you can choose an arbitrary value for this, but a value in the range 5-25 is recommended. Discuss any differences you observe between the Gaussian and KDE naive Bayes classifiers.

<table>
<tr><th>Method</th><th>Classification results</th></tr>
<tr><td>KDE</td><td>

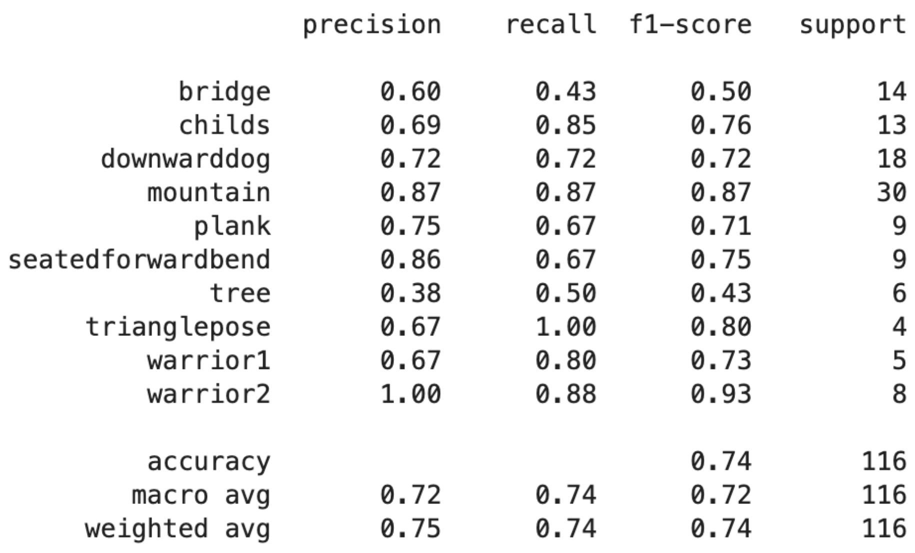


```
                   precision    recall  f1-score   support

           bridge       1.00      0.57      0.73        14
           childs       0.67      0.62      0.64        13
      downwarddog       0.75      0.83      0.79        18
         mountain       0.83      1.00      0.91        30
            plank       0.64      0.78      0.70         9
seatedforwardbend       1.00      0.44      0.62         9
             tree       1.00      0.67      0.80         6
    trianglepose       0.67      1.00      0.80         4
         warrior1       0.57      0.80      0.67         5
         warrior2       0.88      0.88      0.88         8

         accuracy                           0.78       116
        macro avg       0.80      0.76      0.75       116
     weighted avg       0.81      0.78      0.78       116
```

</td></tr>
<tr><td>Gaussian</td><td>



```
                   precision    recall  f1-score   support

           bridge       0.60      0.43      0.50        14
           childs       0.69      0.85      0.76        13
      downwarddog       0.72      0.72      0.72        18
         mountain       0.87      0.87      0.87        30
            plank       0.75      0.67      0.71         9
seatedforwardbend       0.86      0.67      0.75         9
             tree       0.38      0.50      0.43         6
    trianglepose       0.67      1.00      0.80         4
         warrior1       0.67      0.80      0.73         5
         warrior2       1.00      0.88      0.93         8

         accuracy                           0.74       116
        macro avg       0.72      0.74      0.72       116
     weighted avg       0.75      0.74      0.74       116
```

</td></tr>
</table>

The KDE naive bayes was implemented using kernel bandwidth = 15. The KDE classifier assumes arbitrary probability distribution and does not assume the shape of distribution. By contrast, the Gaussian naive bayes assumes a normal distribution.

The total accuracy of the KDE classifier (k=15) is only slightly higher than the Gaussian naive bayes (0.78 compared to 0.74) , which shows that a normal distribution is a good approximation for a majority of classes. However, the bandwidth parameter needs to be tuned to achieve a high accuracy.

The overall metrics (Precision, recall, F-score) predicted by the KDE classifier are higher than that of GNB. However, the F-score of each individual class is not always higher in the KDE classification. For example, the class ‘childs’ has f_score = 0.76 in GNB, and 0.64 in KDE. This shows that the shape of probability distribution of each class is different.

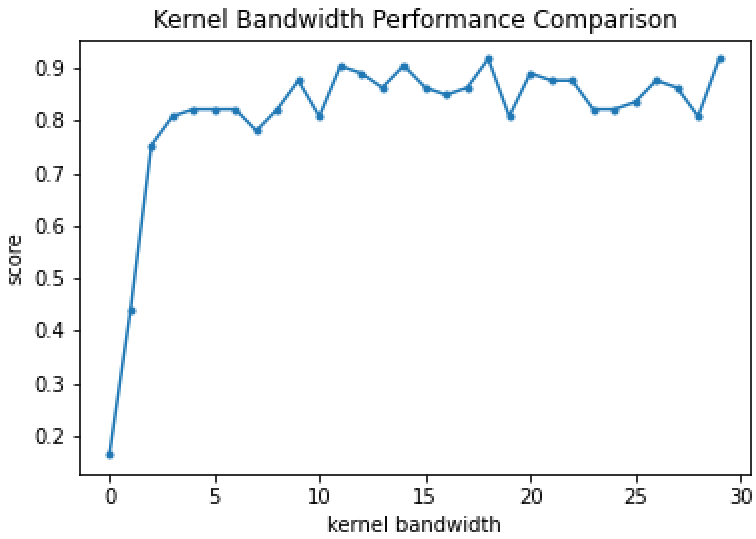
The time complexity of KDE is much higher than that of GNB, since in KDE, each test instance is compared against each instance in each class. The overall time complexity would be O(mn), where m is the number of instances in the training set, and n is the number of instances in test sets.

In comparison, in GNB, the average value of attributes in each class is computed, and each test instance is only compared once against the average instance in each class. The time complexity is O(cn), where c is the number of classes.

4. Instead of using an arbitrary kernel bandwidth for the KDE naive Bayes classifier, use random hold-out or cross-validation to choose the kernel bandwidth. Discuss how this changes the model performance compared to using an arbitrary kernel bandwidth.

A 10-fold cross validation was performed on the training set to tune the kernel bandwidth k. The performance on the validation set was compared.

Plot 1 displays that for kernel bandwidth values in the range 0-29 (Figure 1), which shows that k=18 outperforms other values on the validation set, and thus would maximize model accuracy. The implementation of 10-fold cross validation ensures a reliable evaluation of model accuracy and the random selection of a validation set avoids overfitting.



A small k causes a near-zero probability distribution everywhere except at previously-observed data points, which would result in overfitting and poor generalization. In contrast, a large k causes a more uniform distribution, ignores peaks in the real probability distribution, and compromises accuracy.

Tuning k allows the selection of an optimal k that maximizes the model performance and avoids overfitting compared to using an arbitrary kernel bandwidth, which is shown by the huge difference (68.9%) in the accuracy score obtained by the best k (k = 27) and worst k (k=0).

The best k (k=18) was then used to evaluate the performance of the model in the test dataset, which yields an moderately high accuracy score of 0.784.