Karim Khairat

Assignment 1 Report Machine Learning

Classification:

The classification variant implemented was Abalone-2, a two-class classification where young $\leq$ 10 rings and old > 10 rings. The rings are chosen to judge age as one can often approximate an abalone's age by counting the number of rings and adding 1.5. making it a two-class problem also avoids the possible issues with data for certain ring values being scarce, which would lead to inaccurate predictions resulting in a poor performing model.

Choice of Similarity Metric:

Initially the implementation included two similarity metrics, Euclidean distance, and cosine similarity. However, as the development progressed and testing ensued it was found that cosine similarity was performing very poorly as the cosine similarity was consistently found to be above 0.9. Due to the poor performance of cosine similarity, it was removed and only Euclidean distance remained. Euclidean Distance did the opposite to cosine similarity and proved itself in being an asset as a similarity metric. This was a result of the varying results between the different instances. Euclidean distance isn't without its flaws, as it can often perform poorly when used in conjunction with sparse data, however this is not the case with the abalone dataset.

Validation Framework – Hold-Out:

The Hold-out method was used as it is lighter on resources than cross-fold validation and it can be easily reproduced. However, it does have the disadvantage of not being able to find the ideal training set as the training set is decided after shuffling with a seed. The implementation splits the dataset into 20% testing and 80%training. This is done as it allows for quicker processing with little performance trade off (approximately 0.01 decrease in accuracy when compared to a 50:50 split while reducing processing time by 37%). This was also done to avoid potentially overfitting the data. The seed set is 55, a number which was randomly selected while implementing the Hold-Out method. A seed is important as it allows for the data to be reproduced.

Voting Methods:

There were two voting methods implemented, majority class and inverse linear distance. The two methods were implemented as they provide two different strategies to solving the same problem. However, of the two the recommended voting method would be inverse linear distance as it weighs the closer instances more heavily than the ones further away, thus it becomes more accurate when dealing with larger sets of neighbours. This can be proven within the program by changing the voting method called in evaluate, which is by default set to inverse linear distance.

Evaluation Metrics:

Two evaluation metrics were Implemented, those being accuracy and specificity. Specificity was chosen as it gives the true negative rate. The true negative rate is the proportion of negatives, in the case of the dataset young abalone, that are correctly identified as negatives. For the case of specificity, old abalone were treated as positives, while young abalone were treated as negatives. Accuracy, calculates the proportion of correctly identified abalone. This is useful in knowing the overall performance of the implementation, whereas specificity is more important when one wants to be cautious of incorrectly classifying the data. Consequently, accuracy is recommended in most cases, unless incorrect classification could lead to an unfavourable outcome (i.e. death), in which case specificity is recommended.

Data Representation:

The data was left mostly untouched; however, the rings count was left out during distance as the rings directly related to the classes we were testing for (old or young). Additionally, The Sexes were changed from characters to 1 for male, 2 for female and 3 for infant. This was done as it is more likely for an infant to have less than 11 rings than it is for males and females. However, it should be noted that one cannot classify based on sex alone as there were infants who had 11 or more rings. This makes the sex a valuable piece of information which would assist our model in correctly predicting the class.

Deciding K:

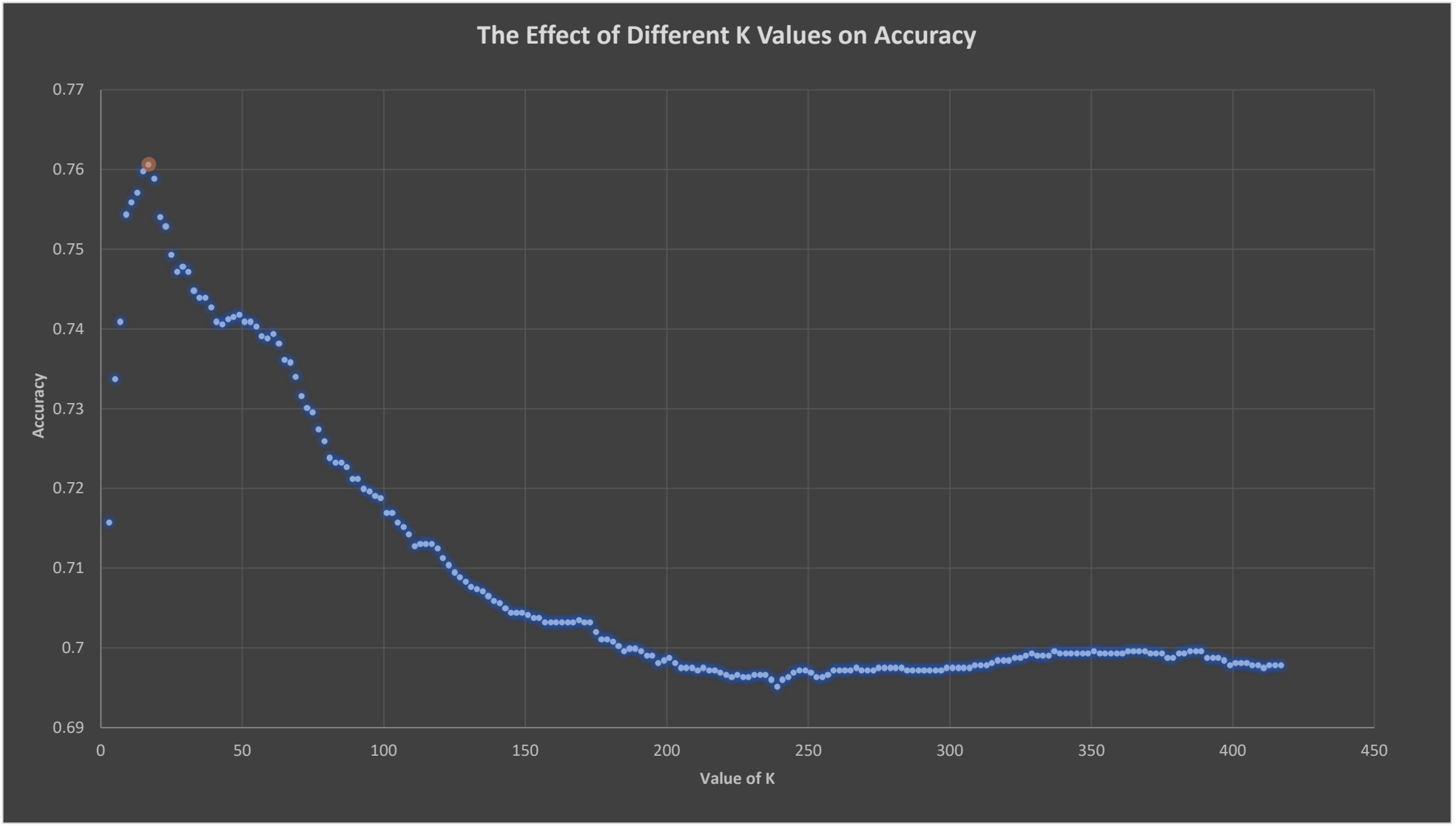
The decision to set k=17 is a result of testing every odd number of k values from 3 to half the length of the training data. This was done by altering the evaluate function to take in a value of k, then iterating through all the possible odd values of k between [3: ½* training data length] and adding the results to a csv file. This method is less feasible with larger datasets due to the amount of processing time required, however a possible solution would be finding the k values between [3: training data length^½]. It is also a valuable method for finding k as it allows for the easy identification of the optimal k value using a visual aid. This method of implemented using the following lines of code:

```
fp= open("k_results.csv",'wb')
aba= preprocess_data("abalone.data")
k= 3
while k<= len(aba[1])/2:
    #altered to take in k values!
    ev= evaluate(aba,"accuracy",k)
    wr = csv.writer(fp,dialect = 'excel')
    wr.writerow([k,ev])
    k+=2
fp.close()
```

Code used to find the optimal k value.

This method took approximately 42minutes. The results were plotted into a scatterplot making the best performing value for k easily identifiable. The scatterplot, which is presented on the following page, shows a rapid increase in accuracy until k = 17 and then a rapid decrease until it stabilises and fluctuates below accuracy = 0.7. It is important to note that k is chosen to be odd to prevent the implementation of tiebreaker scenarios.

Karim Khairat



Scatter Plot showing the effect of the different (odd) k values on accuracy. Optimal k value is highlighted.

Conclusion:

Overall, using the value of k =17, accuracy as the evaluation metric, Euclidean distance as the distance metric and inverse linear distance as the voting method, should yield the most useful results and are therefore the recommended and barring accuracy, are the default in the evaluation function.