Supervised Classifiers Comparison

Kenneth Hansen - cph-kh415@cphbusiness.dk
Martin Høigaard Cupello - cph-mr221@cphbusiness.dk

May 7, 2021

Abstract

It is difficult to decide on which supervised machine learning classification model to implement on a dataset. This can be problem since choosing the wrong model may lead to inaccurate predictions. This article will compare a selection of models to determine which model is the most accurate. This would help on deciding which model to apply to a given classification dataset.

Contents

1 Introduction

This article will compare different machine learning classifiers, and try to determine which one is the most accurate for supervised classification. We will use the popular iris dataset to test different supervised machine learning algorithms, collect metrics, and finally test our findings against two other datasets to see if the results can be confirmed and compare accuracy scores to find the most accurate model.

Our hypothesis is that there exists a single classification model which provides the most accurate predictions on all datasets.

2 Supervised learning

Supervised learning is a type of artificial intelligence and also a machine learning algorithm. The algorithm takes an input variable x and an output y, to learn how the input gets the desired output Y = f(X). This is normally done with a training dataset to produce an accurate prediction. It can be divided into two groups: Classification, where the model assigns a category based on the given input, and regression, which is used to understand the relationship between dependent and independent variables.
This article will be comparing classification models.

2.1 KNN

KNN (K Nearest Neighbor) tries to determine which group the data belongs to, based on proximity to the nearest neighbors. KNN does not make any assumptions; it only uses input data which makes an educated guess of the output based on its closest neighbors. How to choose K: A good estimate to choose K is to take the square root of n, where n is the size of your training data. However, depending on the size of the dataset, it might not be the best.

2.2 Gaussian Naive Bayes and Multinomial Naive Bayes

These classifiers are called "probabilistic classifiers" and use statistics by adding the probability of each input parameter to determine which category the data is most likely to belong to. They are called "naive" since they all treat variables as independent of each other, but they use different algorithms to predict the result. The gaussian naive bayes uses a gaussian distribution, where the probability is calculated using the standard deviation and mean for each input. The multinomial naive bayes assumes that all distributions are multinomial, which means more than 2 input variables.

2.3 Decision Tree

The decision tree is a tree-like structure consisting of nodes and branches. The root node is the starting point which is the entire dataset. Then the node is split into decision nodes, based on the labels attributes. At the end of the tree are the terminal nodes which are the labels/classifications. This is better explained by using the picture below. Assume there is a dataset consisting of animal attributes. You are trying to determine the type/species of an animal, based on its attributes. First the model ask if it has feathers, and based on this decision node, the tree splits into several branches. Based on this decision, it is determined if the animal is a bird or not. The model then work its way down the tree until it reaches a terminal node with a classification.

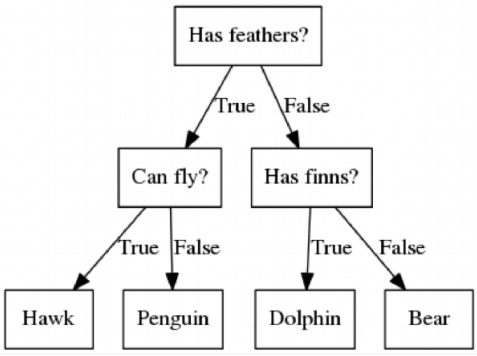


Figure 1: Decision Tree

3 Iris Dataset

Before starting the comparison, the dataset will be explained. This dataset contains information about the flower iris, and has 5 columns, sepal-length, sepal-width, petal-length and petal-width, and a classification column. With this information about the flower, it is possible to predict which species of flower the iris is.

3.1 Classifiers comparison

In this section we will try to find the optimal value for K in the KNN model. We will then apply the different models to our dataset, and list the accuracy scores. It should be noted that all models use the same training/test data split.

Figure 2: KNN classifier

```
success_rate = []
for i in range(3,40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train,y_train)
    pred_i = knn.predict(X_test)
    success_rate.append(accuracy_score(y_test, pred_i))
```

```
print(success_rate)
success_rate.index(max(success_rate))+3
```

```
[0.9, 0.9333333333333333, 0.9, 0.8666666666666667, 0.8666666666666667, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9333333333333
33, 0.9, 0.9, 0.9, 0.9, 0.9, 0.8666666666666667, 0.8666666666666667, 0.8666666666666667, 0.8333333333333334, 0.833333
3333333334, 0.8, 0.8, 0.8333333333333334, 0.9, 0.9, 0.8333333333333334, 0.8666666666666667, 0.8333333333333334, 0.833
3333333333334, 0.8333333333333334, 0.8333333333333334, 0.8333333333333334, 0.8333333333333334, 0.8333333333333334, 0.
8333333333333334]

4
```

The above picture shows that 4 is actually the most optimal value for K as it gives a higher success rate with a train/test split of 80%/20%.

Figure 3: KNN Accurracy Score

```
0.9333333333333333
```

Figure 4: Gaussian Accurracy Score

```
0.8333333333333334
```

Figure 5: Multinomial Accurracy Score

```
0.8333333333333334
```

Figure 6: Decision Tree Accurracy Score

```
0.9
```

3.2 Results

After optimizing and testing each classifier on the dataset, the results suggest that knn is the best classifier for a data set with few attributes and a size of 150 rows. To test our hypothesis, we will take a similar sized dataset and apply the same classifiers to this data.

4 Penguin Dataset

We are using the penguins data set for this experiment. On this dataset we will try to classify which species a penguin is, based on a penguins features. Since the penguins dataset are about double the size of the iris dataset (a little longer actually), we first take a sample of 45% of the full penguins dataset. This leaves us with 150 entries in the sample, close enough to say that it is similar to the length of the iris dataset. We can now begin applying the models.

4.1 Classifiers comparison

We will be using the same procedure as for the Iris dataset. First find optimal k value, then compute the accuracy scores for the different models.

Figure 7: KNN Accurracy Score

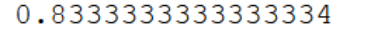


Figure 8: Gaussian Accurracy Score

Figure 9: Multinomial Accurracy Score

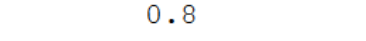


Figure 10: Decision Tree Accurracy Score

4.2 Results

After testing each classifier on the penguin data, it seems that the hypotheses is incorrect. It was assumed that a similar sized dataset would provide the same results, but the result shows that KNN is not the optimal model in this case. Gaussian Naive Bayes and Decision Tree is more accurate on the penguin dataset.

5 Wine Quality Dataset

The last test will be on the wine-quality dataset. It is larger than the iris and penguins dataset combined. Using this dataset, we will try to classify if the wine is red or white, based on the wines physico-chemical content and its quality. Our assumption is that since Decision Tree has performed consistently well on the two previous datasets (93,3% on penguins and 90% on the iris dataset), that it will also perform well on the wine-quality dataset, even though it is of a larger size.

5.1 Classifiers comparison

Using the same procedure as for the Iris and Penguins dataset we get:

Figure 11: KNN Accurracy Score

Figure 12: Gaussian Accurracy Score

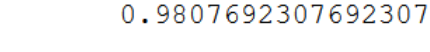


Figure 13: Multinomial Accurracy Score

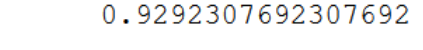
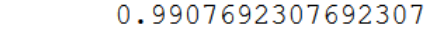


Figure 14: Decision Tree Accurracy Score



5.2 Results

The results from this dataset shows that the decision tree is the most optimal classifier in this case, while Multinomial Naive Bayes continues to be the worst of them all.

6 Results comparison

Here is shown the name and score for each model on each dataset (Iris, Penguins, Wine-Quality) and the total average is shown on the right.

KNN (93,3%, 83,3%, 94,9%)	$\approx$ 90,5%
Gaussian Naive Bayes (83,3%, 93,3%, 98%)	$\approx$ 91,5%
Multinomial Naive Bayes (83,3%, 80%, 93%)	$\approx$ 85,4%
Decision Tree (90,0%, 93,3%, 99,1%)	$\approx$ 94,1%

From these results it can be seen that Decision Tree performs well in most cases, though it is not the most accurate one in all cases. Worst of all in MultiNomial Naive Bayes which got the lowest score in all cases.

6.1 Future work

To further improve upon the results and find the optimal model, we can expand the experiments to include more of the supervised classification models, such as Support Vector Machine and Random Forest.

7 Conclusion

The result have proven the hypothesis incorrect. Given our results, we can conclude that there does not exist a single best classifier, though Decision Tree was a viable candidate. We can conclude this because there was no single classifier that had the highest accuracy score on all of the datasets. It is not possible to decide beforehand which one is the most optimal. We conclude that to find the best classifier for your dataset, you have to try different models and see which one provides the best result.

8 Bibliography

References

[1] Amey Band. How to find the optimal value of k in knn? `https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb`. 23 May 2020.

[2] Jason Brownlee. Naive bayes for machine learning. `https://machinelearningmastery.com/naive-bayes-for-machine-learning`. 15 August 2020.

[3] Jason Brownlee. Supervised and unsupervised machine learning algorithms. `https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms`. 20 August 2020.

[4] Nagesh Singh Chauhan. Decision tree algorithm, explained. `https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html`. Junuary 2020.

[5] Davuluri Hemanth Chowdary. Decision trees explained with a practical example. https://towardsai.net/p/programming/decision-trees-explained-with-a-practical-example-fe47872d3b53. 28 May 2020.

[6] P. Cortez. Wine quality data set. https://archive.ics.uci.edu/ml/datasets/wine+quality. 04 May 2021.

[7] IBM Cloud Education. Supervised learning. https://www.ibm.com/cloud/learn/supervised-learning. 9 August 2020.

[8] Onel Harrison. Machine learning basics with the k-nearest neighbors algorithm. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761. 10 September 2018.

[9] jlund3. Naive bayes for machine learning. https://stats.stackexchange.com/questions/33185/difference-between-naive-bayes-multinomial-naive-bayes. 22 july 2014.

[10] Prateek Majumder. Gaussian naive bayes. https://iq.opengenus.org/gaussian-naive-bayes. 5 May 2021.

[11] Madison Schott. K-nearest neighbors (knn) algorithm for machine learning? https://medium.com/capital-one-tech/k-nearest-neighbors-knn-algorithm-for-machine-learning-e883219c8f26. 22 April 2019.