CSE1015 – Machine Learning Essentials

LAB – 6

Pinni Venkata Abhiram
20BAI1132

Experiment – 6 KNN Classification and Naïve bayes Algorithm

20BAI1132

Pinni Venkata Abhiram

Introduction

Building a KNN Classification model and a Naïve Bayes Classification model using sklearn modules to predict the category of the flower out of three categories i.e. Iris-setosa, Iris-virginica, Iris-versicolor.

The fields in the data are - 'Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm', 'Species'.

The model is built using sklearn knn classification module , sklearn GaussianNB module and various plots for observations provided by seaborn module and confusion matrix , accuracy score for final accuracy and confusion matrix.

Methodology

The methodology used in here is the concepts of KNN Classification and Naïve Bayes Classification

KNN Classification

K Nearest Neighbor algorithm falls under the Supervised Learning category and is used for classification (most commonly) and regression. It is a versatile algorithm also used for imputing missing values and resampling datasets. As the name (K Nearest Neighbor) suggests it considers K Nearest Neighbors (Data points) to predict the class or continuous value for the new Datapoint.

k-NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can

improve its accuracy dramatically. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.

Naïve Bayes Classification

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

The Naive Bayes classification algorithm is a probabilistic classifier. It is based on probability models that incorporate strong independence assumptions. The independence assumptions often do not have an impact on reality. Therefore they are considered as naive. You can derive probability models by using Bayes' theorem (credited to Thomas Bayes). Depending on the nature of the probability model, you can train the Naive Bayes algorithm in a supervised learning setting.

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

Dataset

The dataset used is a .csv file which contains the following columns

'Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm', 'Species' All the rows are having a float value.

We use 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm', for the development of the model and we predict the Species i.e. the species of flower given.

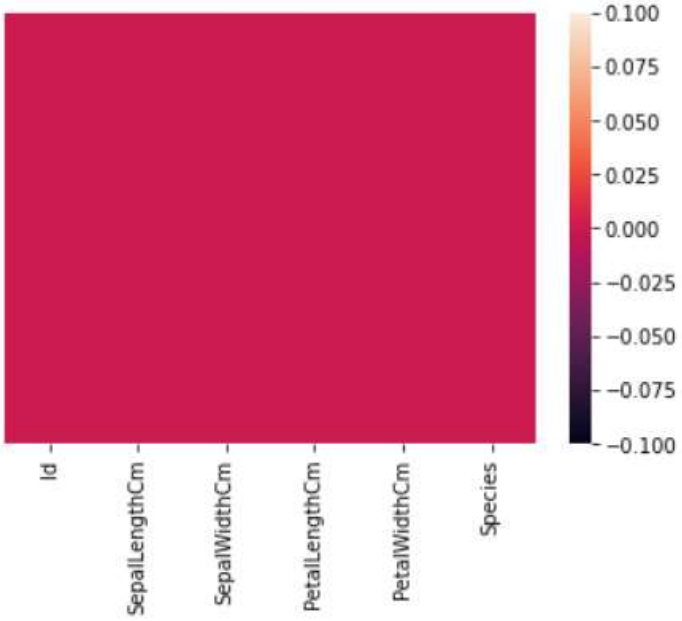
The dataset has 150 rows and 6 columns of data and we split 25% of the data for testing and 75% of data for training purpose.

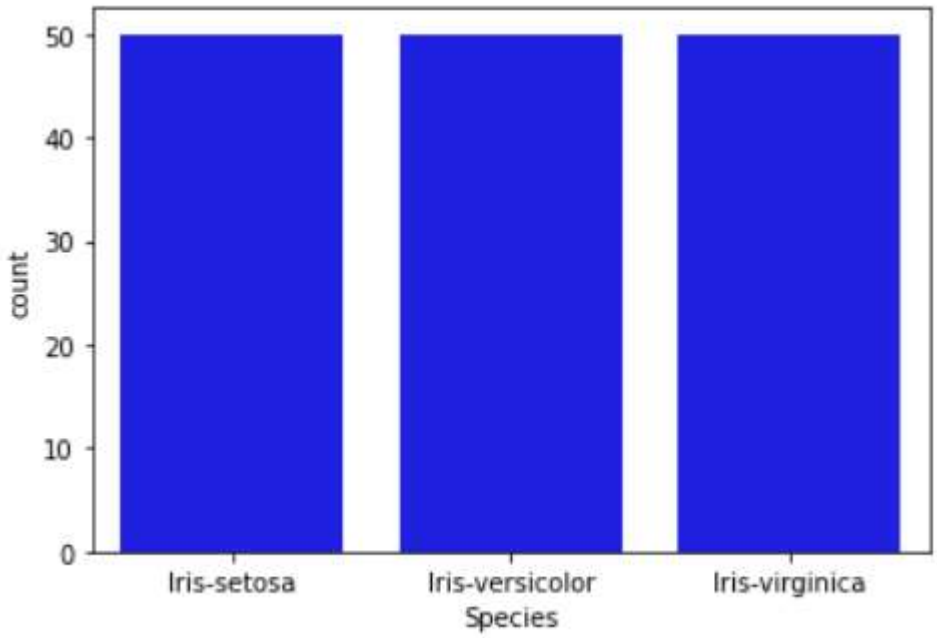
	A	B	C	D	E	F
1	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
2	1	5.1	3.5	1.4	0.2	Iris-setosa
3	2	4.9	3	1.4	0.2	Iris-setosa
4	3	4.7	3.2	1.3	0.2	Iris-setosa
5	4	4.6	3.1	1.5	0.2	Iris-setosa
6	5	5	3.6	1.4	0.2	Iris-setosa
7	6	5.4	3.9	1.7	0.4	Iris-setosa
8	7	4.6	3.4	1.4	0.3	Iris-setosa
9	8	5	3.4	1.5	0.2	Iris-setosa
10	9	4.4	2.9	1.4	0.2	Iris-setosa
11	10	4.9	3.1	1.5	0.1	Iris-setosa
12	11	5.4	3.7	1.5	0.2	Iris-setosa
13	12	4.8	3.4	1.6	0.2	Iris-setosa
14	13	4.8	3	1.4	0.1	Iris-setosa
15	14	4.3	3	1.1	0.1	Iris-setosa

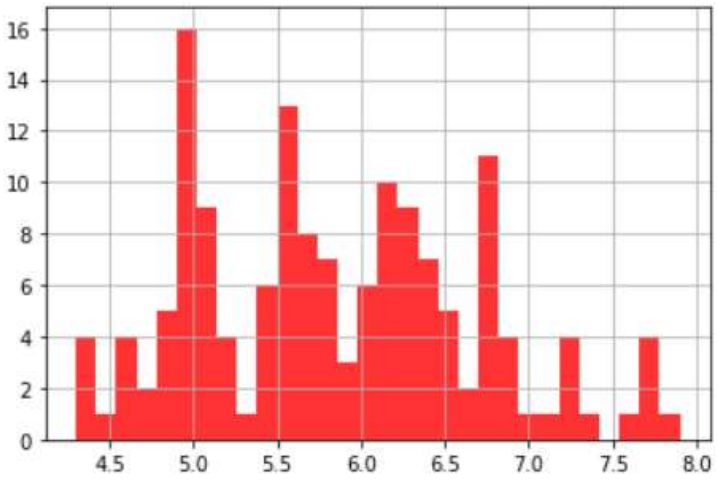
Experiments and Results

The experiment required numpy , pandas , seaborn , sklearn , matplotlib libraries for analysis and model building.

The data is imported using pandas then we make plots based on the data using seaborn for our better understanding







And many more plots.

We check if the data has any NaN values i.e. missing values and we try to remove them , this is the data cleaning process.

```
df.isnull().sum()
```

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

There are no rows with NaN values so the dataset is ready to be used

We use the sklearn library to divide the data into training set and test set.

```
x_train , x_test, y_train , y_test = tsplit(X,y,train_size=0.75,random_state=1)
```

Then we perform the KNN Classification using the training dataset and we predict the values using test set.

```
y_pred = classifier.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 2, 1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 0, 2, 1, 0, 0, 1, 2, 1, 2, 1, 2, 2, 0, 1, 0])
```

Then we draw the confusion matrix and get the accuracy score using the sklearn library and get the final accuracy score.

```
cm = confusion_matrix(y_test, y_pred)
cm
```

```
array([[13,  0,  0],
       [ 0, 16,  0],
       [ 0,  0,  9]], dtype=int64)
```

```
accuracy = accuracy_score(y_test, y_pred)*100
print('Accuracy of our model is equal ' + str(round(accuracy, 2)) + ' %.')
```

```
Accuracy of our model is equal 100.0 %.
```

Accuracy of the Knn model is 100% , which means the correct model for this type of question is the Knn classification model.

Naïve Bayes Classification

We get the naïve bayes classifier from the sklearn module and train the classifier using the train datasets.

Naive bayes classification model

```
naive_bayes = naive_bayes_algo()
```

Fitting the model with our training datasets

```
naive_bayes.fit(x_train, y_train)
```

```
GaussianNB()
```

Prediction of values using the test datasets and drawing the confusion matrix and accuracy scores.

```
y_pred = naive_bayes.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 2, 1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       2, 0, 2, 1, 0, 0, 1, 2, 1, 2, 1, 2, 2, 0, 1, 0])
```

Confusion Matrix and Accuracy scores

```
cm = confusion_matrix(y_test, y_pred)
cm
```

```
array([[13,  0,  0],
       [ 0, 15,  1],
       [ 0,  0,  9]], dtype=int64)
```

```
accuracy = accuracy_score(y_test, y_pred)*100
print('Accuracy of our model is equal ' + str(round(accuracy, 2)) + ' %.')
```

```
Accuracy of our model is equal 97.37 %.
```

Finally we can say that the knn model is 100% accurate and naïve bayes model is 97.37% accurate when we use the test dataset to predict when the model is built with 75% of the data.

Conclusion

The Knn Classification model and Naïve bayes model of Machine Learning is successfully implemented and can classify the given flower when details are given using Python Libraries and Jupyter Notebook interface.

References

https://www.kaggle.com/skalskip/iris-data-visualization-and-knn-classification/data

https://scikit-learn.org/stable/modules/naive_bayes.html

https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

https://towardsdatascience.com/introduction-to-na%C3%AFve-bayes-classifier-fa59e3e24aaf

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning