EE393 Term Project

Data Classification and Pattern Recognition Using Dataset IRIS

Ercument Burak Tokman
Computer Engineering
S013090

Outline

1.Introduction

Iris is perhaps the best known database to be found in the pattern recognition literature.

The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems as an example of linear discriminant analysis.

Fisher's paper is a classic in the field and is referenced frequently to this day. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

In this project we are going to use Iris Dataset with 6 different machine learning algorithms and visualize the data and compare the results. Algorithms are listed below:

Logistic Regression **(LR)**
Linear Discriminant Analysis **(LDA)**
K-Nearest Neighbors **(KNN)**
Classification and Regression Trees **(CART)**
Gaussian Naive Bayes **(NB)**
Support Vector Machines **(SVM)**

2.Examination of the Dataset

We are going to take a look at the data a few different ways:

1. **Dimensions of the dataset:** (150, 5)
2. **Peek at the data itself.**

Below is the first 10 instance of the data;

	sepal-length	*sepal-width*	*petal-length*	*petal-width*	*class*
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

3. Statistical summary of all attributes.

Now we can take a look at a summary of each attribute.
This includes the count, mean, the min and max values as well as some percentiles

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

4. Breakdown of the data by the class variable.

Take a look at the number of instances (rows) that belong to each class. This can be interpreted as an absolute count.

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

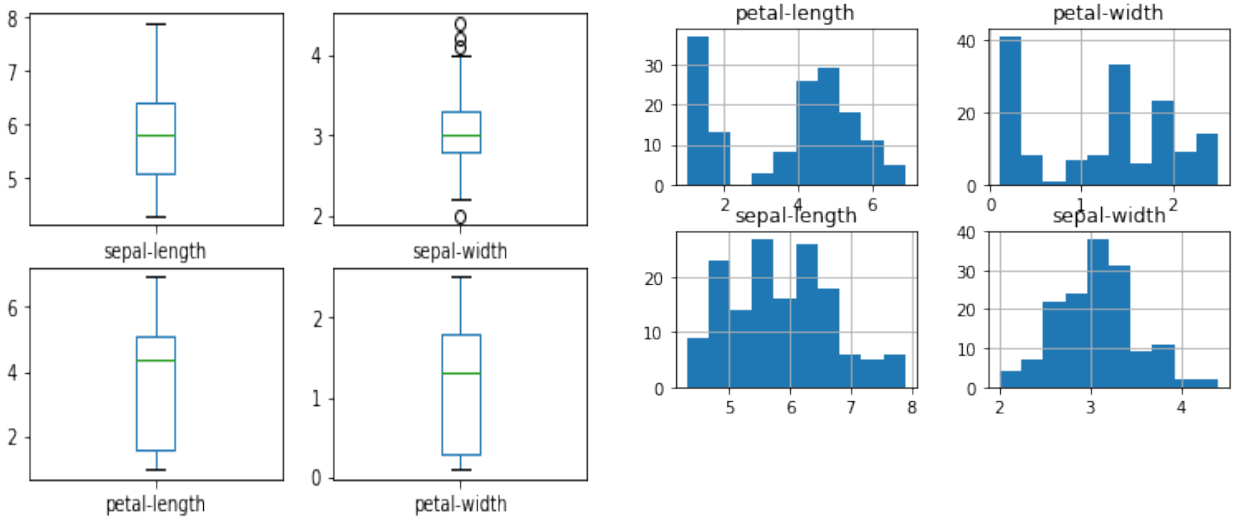
3.Visualization of the Data

Statistics is good to have an idea about the dataset but we need to extend that with some visualizations.We are going to look at two types of plots:

- Univariate plots to better understand each attribute.
- Multivariate plots to better understand the relationships between attributes.

3.1 - Univariate Plots

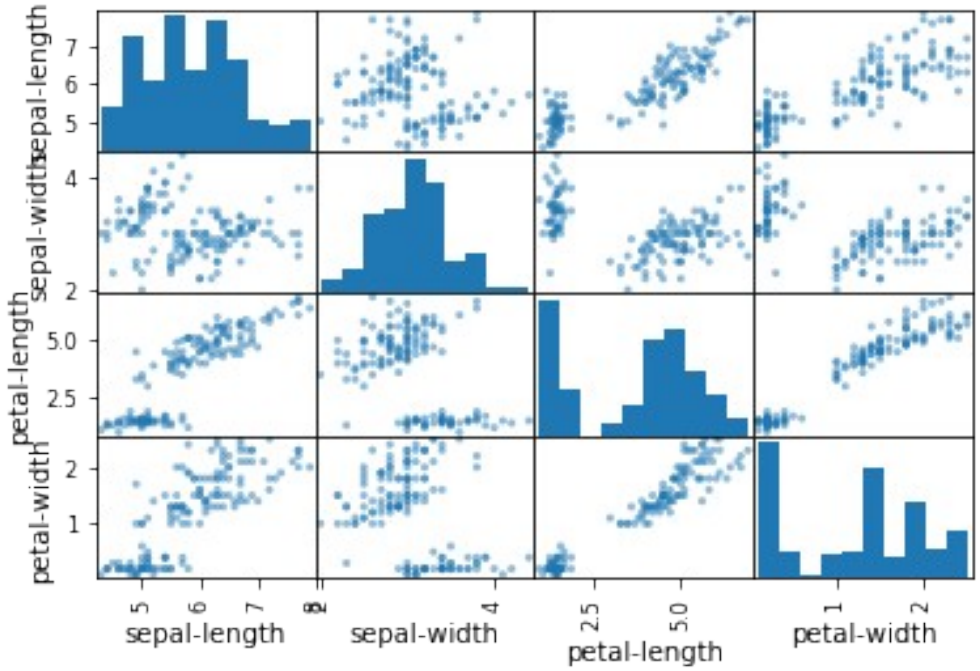
We start with some univariate plots, that is, plots of each individual variable. Given that the input variables are numeric, we can create box and whisker plots of each **(left)** and also create a histogram of each input variable to get an idea of the distribution **(right)**



It looks like perhaps two of the input variables have a Gaussian distribution. This is useful to note as we can use algorithms that can exploit this assumption.

3.2 - Multivariate Plots

Multivariate plots are usefull to see the interactions between the variables. First, let's look at scatterplots of all pairs of attributes.This can be helpful to spot structured relationships between input variables.



4.Evaluating Machine Learning Algorithms

After examining the data and visualizing it now its time to create some models of the data and estimate their accuracy on unseen data.

- Here is what we are going to cover in this step:
- Separate out a validation dataset.
- Set-up the test harness to use 10-fold cross validation.
- Build 5 different models to predict species from flower measurements Select the best model.

4.1 - Create a Validation Dataset

We need to know that the model we created is any good.
Later, we will use statistical methods to estimate the accuracy of the models that we create on unseen data. We also want a more concrete estimate of the accuracy of the best model on unseen data by evaluating it on actual unseen data.
That is, we are going to hold back some data that the algorithms will not get to see and we will use this data to get a second and independent idea of how accurate the best model might actually be.
We will split the loaded dataset into two, 80% of which we will use to train our models and 20% that we will hold back as a validation dataset.

4.2 – Test Harness

We will use 10-fold cross validation to estimate accuracy.
This will split our dataset into 10 parts, train on 9 and test on 1 and repeat for all combinations of train-test splits.

We are using the metric of ‘accuracy‘ to evaluate models. This is a ratio of the number of correctly predicted instances in divided by the total number of instances in the dataset multiplied by 100 to give a percentage (e.g. 95% accurate). We will be using the scoring variable when we run build and evaluate each model next.

5.Building Models

We don’t know which algorithms would be good on this problem or what configurations to use. We get an idea from the plots that some of the classes are partially linearly separable in some dimensions, so we are expecting generally good results.
Let’s evaluate 6 different algorithms:

Logistic Regression **(LR)**
Linear Discriminant Analysis **(LDA)**
K-Nearest Neighbors **(KNN)**
Classification and Regression Trees **(CART)**
Gaussian Naive Bayes **(NB)**
Support Vector Machines **(SVM)**

This is a good mixture of simple linear (LR and LDA), nonlinear (KNN, CART, NB and SVM) algorithms. We reset the random number seed before each run to ensure that the

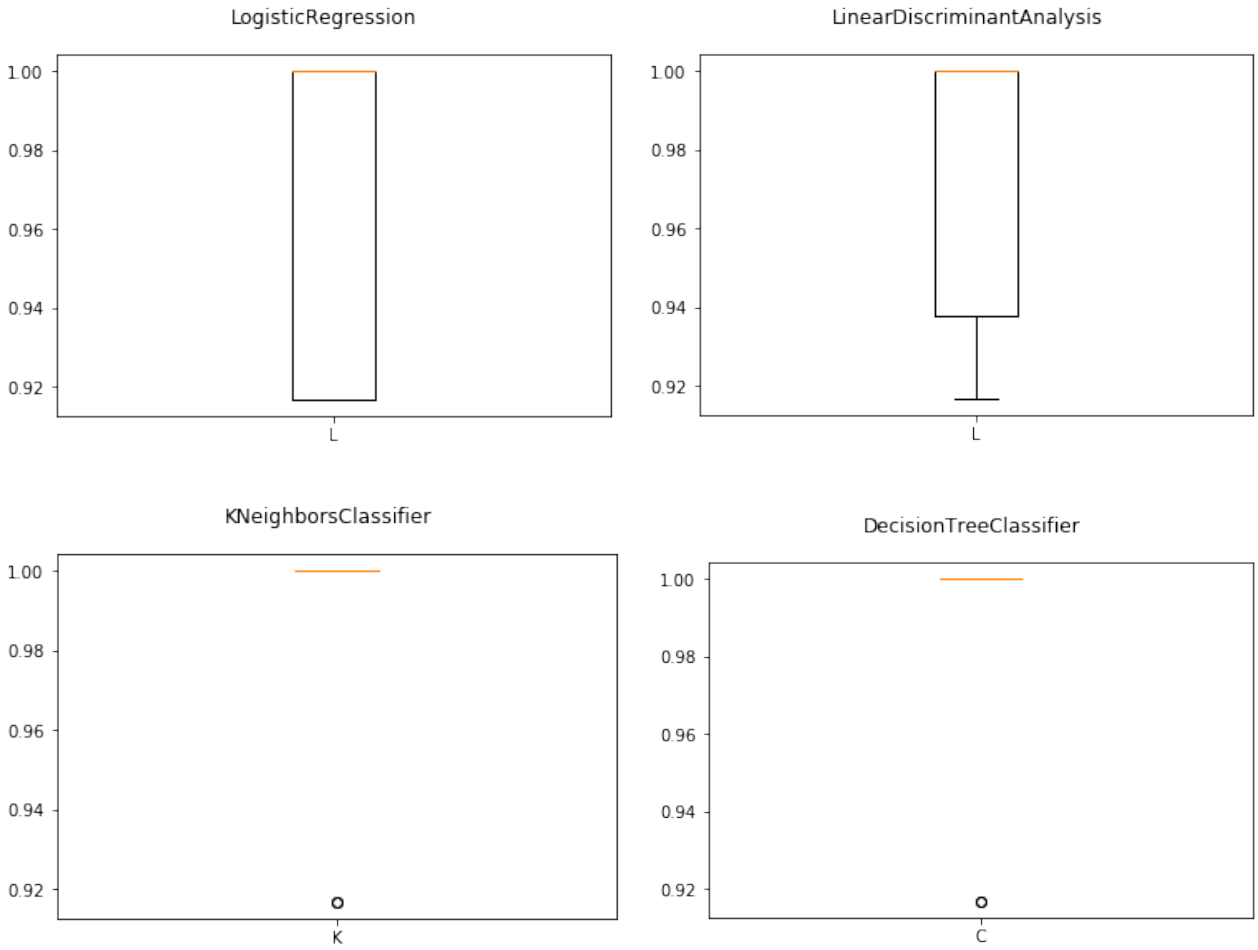
evaluation of each algorithm is performed using exactly the same data splits. It ensures the results are directly comparable.

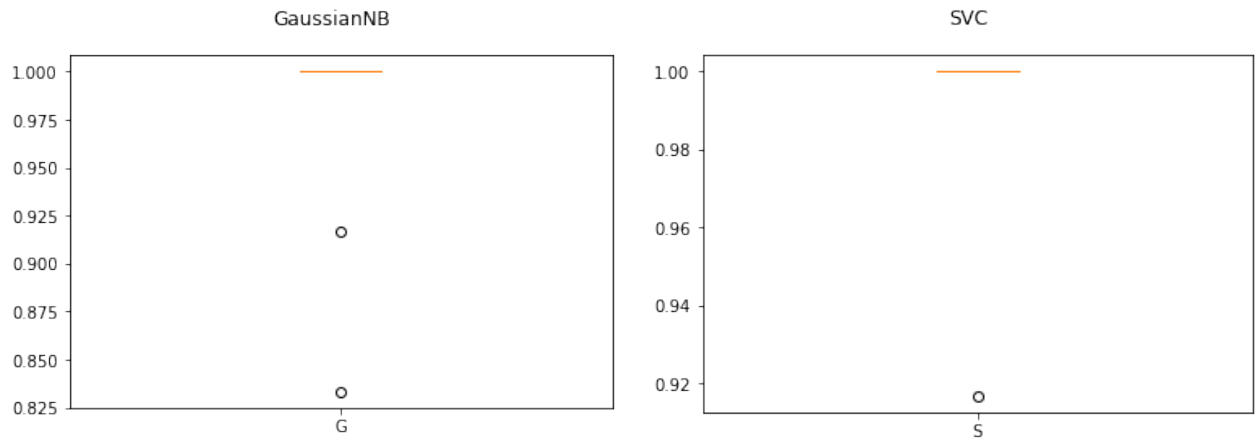
6.Results

6.1 – Table

	Score Mean	Score STD
Linear Regression	0.966667	0.040825
Linear Discriminant Analysis	0.975000	0.038188
K-Nearest Neighbors	0.983333	0.033333
Classification and Regression Trees	0.983333	0.033333
Gaussian Naive Bayes	0.975000	0.053359
Support Vector Machines	0.991667	0.025000

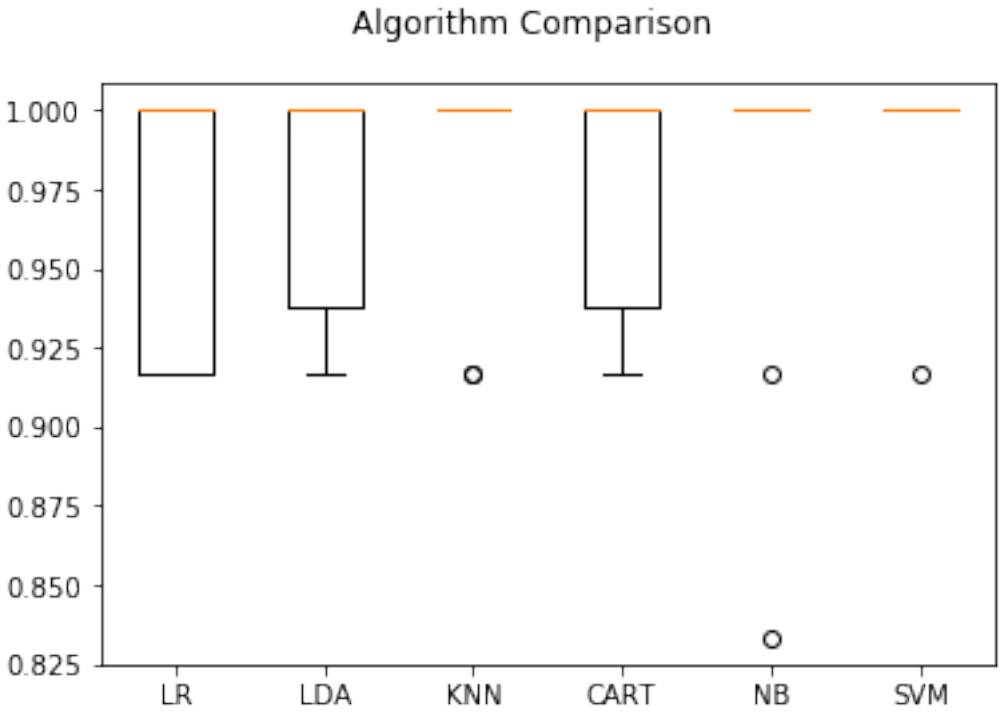
6.2 – Graphs





6.3 – Comparing Algorithms

We have 6 models and accuracy estimations for each.We need to compare the models to each other and select the most accurate. We can see that it looks like KNN has the largest estimated accuracy score.
We can also create a plot of the model evaluation results and compare the spread and the mean accuracy of each model. There is a population of accuracy measures for each algorithm because each algorithm was evaluated 10 times (10 fold cross validation).



Graph shows that the box and whisker plots are squashed at the top of the range, with many samples achieving 100% accuracy.

7.Making Future Predictions & Conclusion

The KNN algorithm was the most accurate model that we tested. Now we want to get an idea of the accuracy of the model on our validation set.

This will give us an independent final check on the accuracy of the best model. It is valuable to keep a validation set just in case you made a slip during training, such as overfitting to the training set or a data leak. Both will result in an overly optimistic result.

We can run the KNN model directly on the validation set and summarize the results as a final accuracy score, a confusion matrix and a classification report.

```
0.9
[[ 7  0  0]
 [ 0 11  1]
 [ 0  2  9]]
```

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	7
Iris-versicolor	0.85	0.92	0.88	12
Iris-virginica	0.90	0.82	0.86	11
avg / total	0.90	0.90	0.90	30

We can see that the accuracy is 0.9 or 90%. The confusion matrix provides an indication of the three errors made. Finally, the classification report provides a breakdown of each class by precision, recall, f1-score and support showing excellent results (granted the validation dataset was small).

8.References

UCI Machine Learning Repository Iris Data Set - **UCI**
https://archive.ics.uci.edu/ml/datasets/iris

Iris Data Analysis and Machine Learning(Python), **Aditya D. Bhat**
https://www.kaggle.com/adityabhat24/iris-data-analysis-and-machine-learning-python

Python Data Visualizations, **Ben Hamner**
https://www.kaggle.com/benhamner/python-data-visualizations

A Complete Guide to K-Nearest-Neighbors with Applications in Python and R, **Kevin Zakka**
https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/

Linear Discriminant Analysis – Bit by Bit, **Sebastian Raschka**
http://sebastianraschka.com/Articles/2014_python_lda.html

Learning Data Science: Day 21 - Decision Tree on Iris Dataset, **Haydar Ali Ismail**
https://medium.com/@haydar_ai/learning-data-science-day-21-decision-tree-on-iris-dataset-267f3219a7fa

Support Vector Machines, **João Neto**
http://www.di.fc.ul.pt/~jpn/r/svm/svm.html