Implementing Machine learning models on Iris Dataset.

Hari Mittapalli Follow
Dec 27, 2018 · 2 min read

We have discussed EDA on Iris Dataset in the last blog post. If you have not read it yet, have a look at my **last post**

In this post I will try to implement different machine learning models and see the differences between them and accuracy rates.

Without further delay let's get into action. As discussed in previous post Iris Data set contains the details of Iris flowers features such as Petal Length and Width and Sepal Length and Width, and these are of 3 categories which are virginic, versicolor and setosa. Our objective is to classify these objects to 3 categories based on the input features.

You can find the Jupyter notebook for the source code ***here*.**

Splitting Dataset:

Before implementing any model we need to split the dataset to *train* and *test* sets. We use ***train_test_split*** class from ***sklearn.model_selection*** library to split our dataset.

> *from sklearn.model_selection import train_test_split*
> *train,test=train_test_split(data,test_size=0.3)*

The above code will split the dataset to 70% as train and 30% as test datasets.

> *train.shape, test.shape*

```
((105, 5), (45, 5))
```

Now let's split the train and test sets further as input and output sets.

> *train_X=train[['Sepal Length',"Sepal Width","Petal length","Petal Width"]]*

```
train_y=train.Species
test_X=test[['Sepal Length',"Sepal Width","Petal length","Petal Width"]]
test_y=test.Species
```

1. Decission Trees Model:

First let's get started with *Decission Trees Model***.**

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. It uses *Entropy* and *Information Gain* to construct a decision tree.

Entropy

Entropy controls how a Decision Tree decides to ***split*** *the data. It actually effects how a* ***Decision Tree*** *draws its boundaries.*

Information Gain:

Information gain (IG) measures how much ***"information"*** *a feature gives us about the class.*

We need to import the ***DecisionTreeClassifier*** from ***sklearn.tree*** to implement our decision tree classifier model and also import ***metrics*** function to calculate accuracy score of the model.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
```

Let's build the Decision tree model on the train set.

```
dtmodel=DecisionTreeClassifier()
dtmodel.fit(train_X,train_y)
```

We can predict the output for the test dataset using predict() function. Let's do that.

```
dtpredict=dtmodel.predict(test_X)
```

And calculate the accuracy score of the model.

```
dtaccuracy=metrics.accuracy_score(dtpredict,test_y)
print("Decission Tree Model Accuracy is {}".format(dtaccuracy * 100))
```

```
Decission Tree Model Accuracy is 93.33333333333333
```

Accuracy score of our decision Tree model is 93.33%. Curious about knowing which are the wrongly predicted records?

> *test_preddf=test.copy()*
> *test_preddf['Predicted Species']=dtpredict*
> *wrongpred=test_preddf.loc[test['Species'] != dtpredict]*
> *wrongpred*

	Petal length	Petal Width	Sepal Length	Sepal Width	Species	Predicted Species
77	6.7	3.0	5.0	1.7	versicolor	virginica
133	6.3	2.8	5.1	1.5	virginica	versicolor
129	7.2	3.0	5.8	1.6	virginica	versicolor

these are 3 wrongly predicted records

As we can see in above pic 77th indexed record is actually v*ersicolor* Specie but predicted as *virginica* and other 2(133th,129th) records are of *virginica* but predicted as *versicolor.*

Will discuss about **Logistic Regression Model** and **SVM**(Support Vector Machine) Model in the coming posts.