Data Science Project Training Report

on

Machine Learning Domain Projects for Classification using Datasets

BACHELOR OF TECHNOLOGY

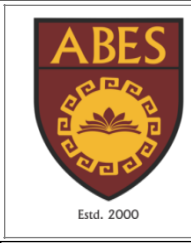
Session 2021-22
in
Information Technology

By

Ujjwal jain: 2000320130181
Utkarsh Kumar Srivastava: 2000320130183
Rudransh jain: 2000320130136

Ms. Monica Batra
ASSISTANT PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
ABES ENGINEERING COLLEGE, GHAZIABAD



AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

Student's Declaration

I hereby declare that the work being presented in this report entitled **"Iris Species Clustering"** is an authentic record of my / our own work carried out under the supervision of **Ms. Monica Batra**, **Assistant Professor, Information Technology.**

Date:

Signature of Group Members:
Ujjwal jain
Utkarsh Kumar Srivastava
Rudransh jain
IT-C

This is to certify that the above statement made by the candidate(s) is correct to the best of my knowledge.

Signature of HOD
Prof.(Dr.) Amit Sinha
Information Technology

Signature of Teacher
Ms. Monica Batra
Assistant Professor
Information Technology

Date:.....

Table of Contents

Introduction

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

Data Collection

- This dataset consists of 3 categories of species which is setosa, versicolor and virginica.
- We can find two kind of data which is CSV data and SQLITE database.
- Each iris species consists of 50 samples.
- The features of iris flower are Sepal Length in cm, Sepal Width in cm, Petal Length in cm and Petal Width in cm.

Data Pre processing

- We can read the both kind of data by using the below code

```
import pandas as pd
df=pd.read_csv("https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data")
df.columns=['sepal length','sepal width','petal length','petal width','class']
print(df)
```

```
     sepal length  sepal width  petal length  petal width           class
0             4.9          3.0           1.4          0.2     Iris-setosa
1             4.7          3.2           1.3          0.2     Iris-setosa
2             4.6          3.1           1.5          0.2     Iris-setosa
3             5.0          3.6           1.4          0.2     Iris-setosa
4             5.4          3.9           1.7          0.4     Iris-setosa
..            ...          ...           ...          ...             ...
144           6.7          3.0           5.2          2.3  Iris-virginica
145           6.3          2.5           5.0          1.9  Iris-virginica
146           6.5          3.0           5.2          2.0  Iris-virginica
147           6.2          3.4           5.4          2.3  Iris-virginica
148           5.9          3.0           5.1          1.8  Iris-virginica

[149 rows x 5 columns]
```

- **Pre-processed data**

```
print(df.shape)
print(df.size)
print(df.columns)
print(df.head())
print(df.tail)
print(df.info())
print(df.describe().T)
```

```
(149, 5)
745
Index(['sepal length', 'sepal width', 'petal length', 'petal width', 'class'], dtype='object')
   sepal length  sepal width  petal length  petal width        class
0           4.9          3.0           1.4          0.2  Iris-setosa
1           4.7          3.2           1.3          0.2  Iris-setosa
2           4.6          3.1           1.5          0.2  Iris-setosa
3           5.0          3.6           1.4          0.2  Iris-setosa
4           5.4          3.9           1.7          0.4  Iris-setosa
<bound method NDFrame.tail of      sepal length  sepal width  petal length  petal width           class
0             4.9          3.0           1.4          0.2     Iris-setosa
1             4.7          3.2           1.3          0.2     Iris-setosa
2             4.6          3.1           1.5          0.2     Iris-setosa
3             5.0          3.6           1.4          0.2     Iris-setosa
4             5.4          3.9           1.7          0.4     Iris-setosa
..            ...          ...           ...          ...             ...
144           6.7          3.0           5.2          2.3  Iris-virginica
145           6.3          2.5           5.0          1.9  Iris-virginica
146           6.5          3.0           5.2          2.0  Iris-virginica
147           6.2          3.4           5.4          2.3  Iris-virginica
148           5.9          3.0           5.1          1.8  Iris-virginica
```

Data Set Information

This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) 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.

Predicted attribute: class of iris plant.

Data Set Characteristics:	Multivariate	**Number of Instances:**	150	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	4	**Date Donated**	1988-07-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	4745234

To Find Missing Value

```
[3]  print(df.isna().sum())
     #To check missing value
```

```
sepal length    0
sepal width     0
petal length    0
petal width     0
class           0
dtype: int64
```

Exploratory Data Analysis (EDA)

- Let's group the data by species and do some descriptive statistics

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 149 entries, 0 to 148
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal length  149 non-null    float64
 1   sepal width   149 non-null    float64
 2   petal length  149 non-null    float64
 3   petal width   149 non-null    float64
 4   class         149 non-null    object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None
              count      mean       std  min  25%  50%  75%  max
sepal length  149.0  5.848322  0.828594  4.3  5.1  5.8  6.4  7.9
sepal width   149.0  3.051007  0.433499  2.0  2.8  3.0  3.3  4.4
petal length  149.0  3.774497  1.759651  1.0  1.6  4.4  5.1  6.9
petal width   149.0  1.205369  0.761292  0.1  0.3  1.3  1.8  2.5
```

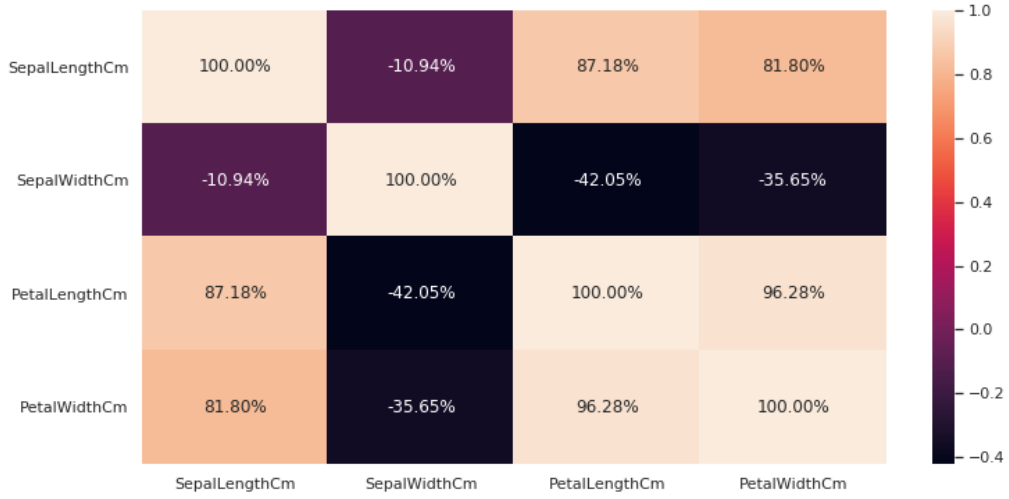
Feature Observations

- Let's plot the correlation between the features.
- Correlation between the features

```
features_correlation = iris_db.drop('Species', axis = 1).corr()
```

- Let's plot an insightful plot on feature correlation

```
sb.heatmap(data = features_correlation, annot = True, fmt = '.2%')
```



- Let's understand how the Petal length and Petal width contributes together to classify iris species.
- Petal Length vs Petal Width

  sb.relplot(x = 'PetalLengthCm', y = 'PetalWidthCm', data = iris_db, hue = 'Species', aspect = 2, height = 6)

- From the above scatter plot, setosa species clearly classified from other two species.
- Other two species can also be classified based on Petal Length and Petal Width easily.
- Let's understand how the Petal length and Sepal Length contributes together to classify iris species.
- Petal Length vs Sepal Length

  sb.relplot(x = 'PetalLengthCm', y = 'SepalLengthCm', data = iris_db, hue = 'Species', aspect = 2, height = 6)

- From the above plot, we can separate versicolor and virginica by means of Petal Length alone.

- Sepal Length does not contribute more here
- Let's understand how the Petal width and Sepal Length contributes together to classify iris species.
- Petal Width vs Sepal Length

  sb.relplot(x = 'PetalWidthCm', y = 'SepalLengthCm', data = iris_db, hue = 'Species', aspect = 2, height = 6)

- From the above plot, we can separate based on Petal Width alone.
- Sepal Length doesn't contribute more here.

Feature Selection

- From the above discussion, we can easily classify the iris species based on petal features than sepal features.is
- Selecting Iris Features

```
iris_features = iris_db.loc[:,['PetalLengthCm','PetalWidthCm']]
```

- Selecting Iris Species

```
iris_species = iris_db.loc[:,'species']
```

Building Machine Learning Model

- Based on above explorations on the data, we came to know that we can classify the iris species based on some conditions on Petal Length and PetalWidth.
- Let's build a **KMeansClustering** model to the data.
- Let's import all the libraries required to run clustering model.

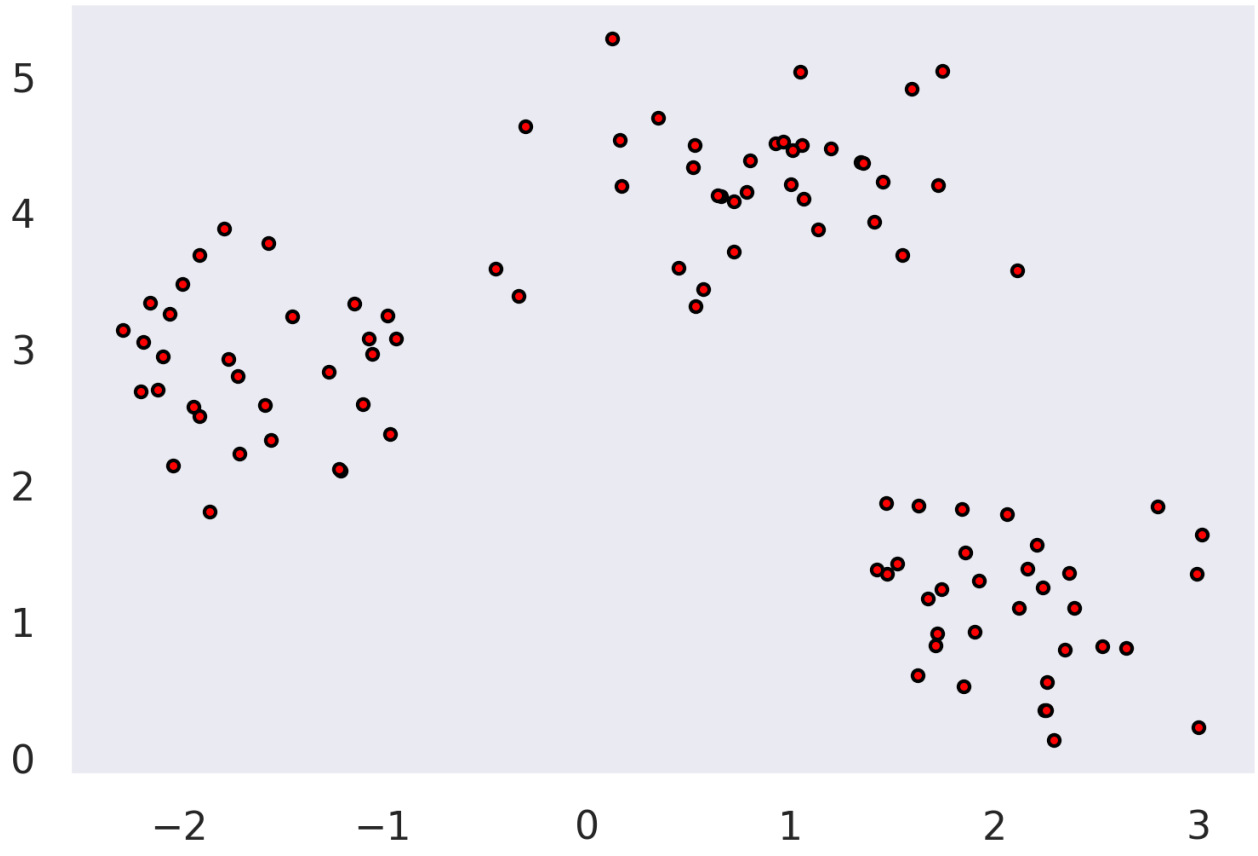
```
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from termcolor import colored
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import chi2
import scipy.stats as stats
from imblearn.over_sampling import RandomOverSampler
from sklearn.feature_selection import VarianceThreshold
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from scipy.stats import zscore
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import cross_val_score
sns.set(rc={"figure.dpi":300, 'savefig.dpi':300})
plt.rcParams['figure.dpi'] = 300
plt.rcParams['savefig.dpi'] = 300
pd.set_option('display.max_columns',29)
```

Model Performance

- Checking model performance by changing max_depth parameter and the criterion as gini.

```
from sklearn.datasets import make_blobs
X,y=make_blobs(n_samples=100,n_features=2,centers=3,cluster_std=0.5,shuffle=True,
               random_state=0)
import matplotlib.pyplot as plt
plt.scatter(X[:,0],X[:,1],c='red',edgecolor='black',marker='o',s=25)
plt.grid()
plt.show()
```

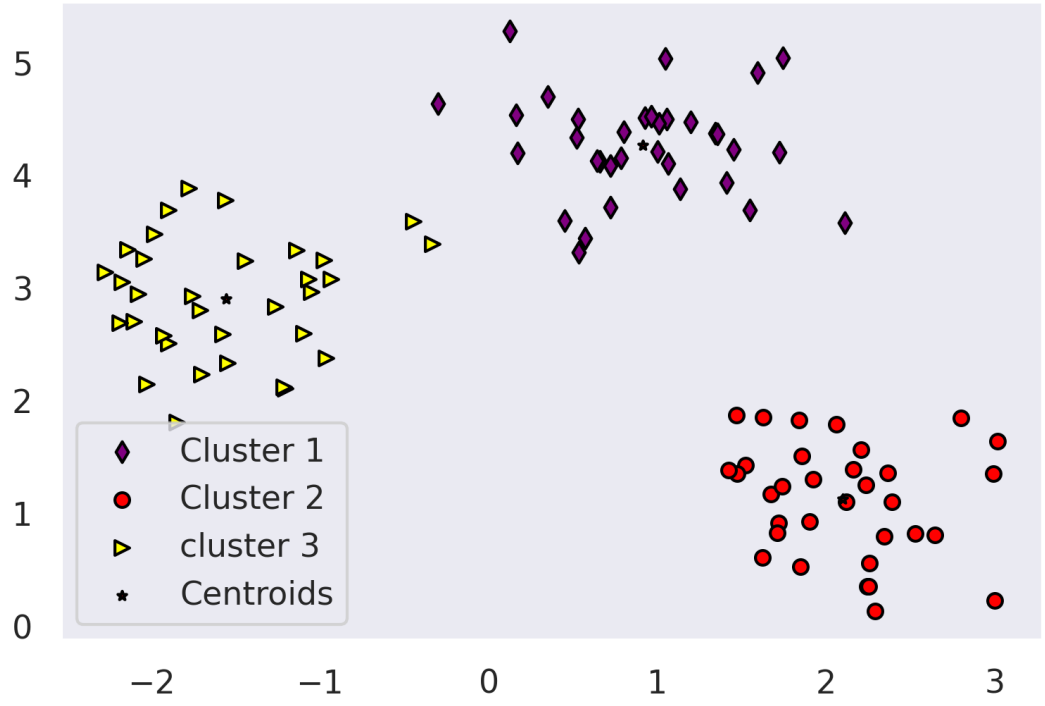
Output:-



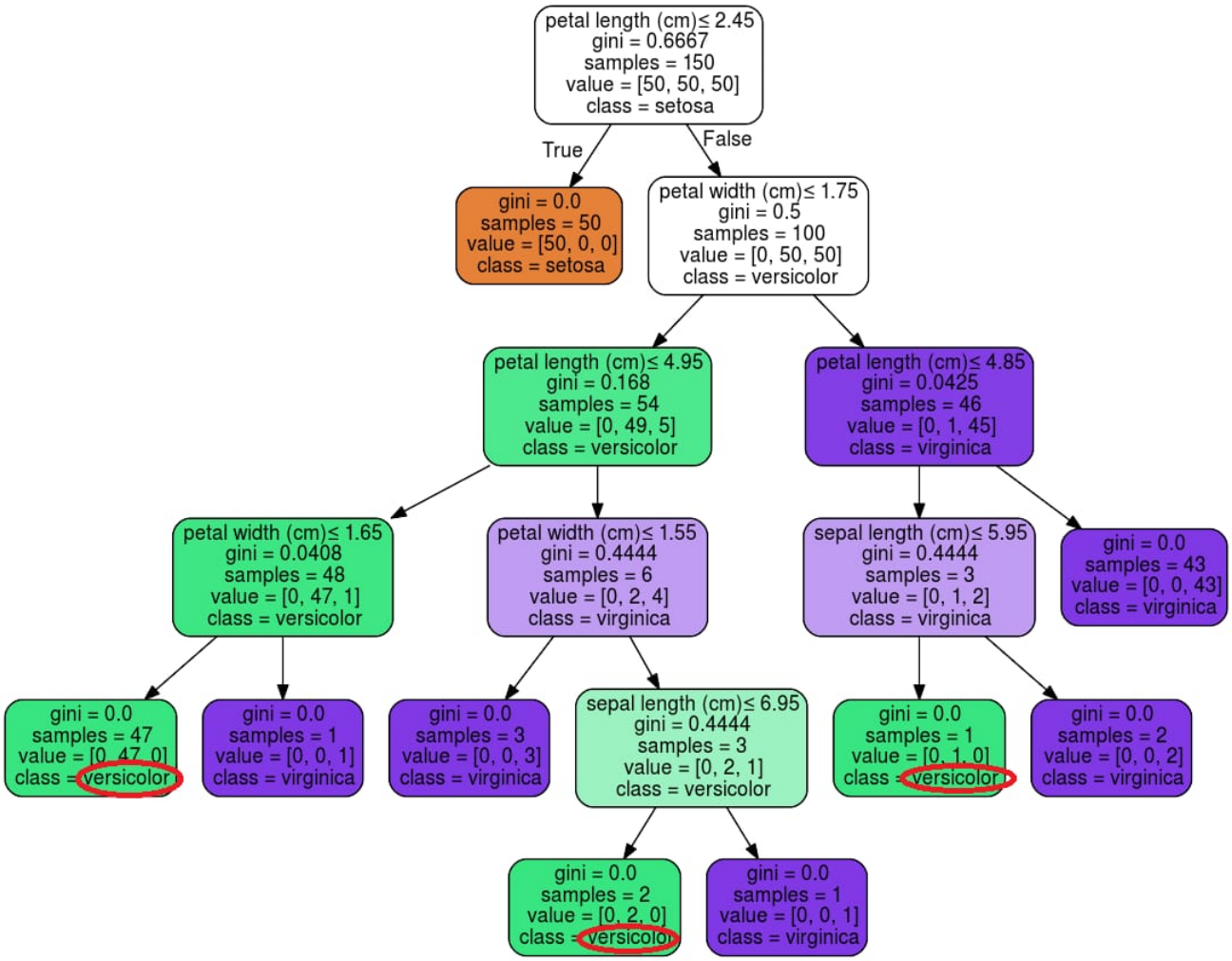
```
from sklearn.cluster import KMeans
km=KMeans(n_clusters=3,init='random',n_init=10,max_iter=300,tol=1e-04,random_state=0)
y_km=km.fit_predict(X)
```

```
plt.figure()
plt.scatter(X[y_km==0,0],X[y_km==0,1],s=25,c='purple',marker='d',edgecolor='black',
            label='Cluster 1')
plt.scatter(X[y_km==1,0],X[y_km==1,1],s=25,c='red',marker='o',edgecolor='black',
            label='Cluster 2')
plt.scatter(X[y_km==2,0],X[y_km==2,1],s=25,c='yellow',marker='>',edgecolor='black',
            label='cluster 3')
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],
            s=10,marker='*',color='red',
            edgecolor='black',label='Centroids')
plt.legend(scatterpoints=1)
plt.grid()
plt.show()
print('Distortion:%.2f'%km.inertia_)
```

Output:-



Build Optimal Model



The Decision Tree

Predictions

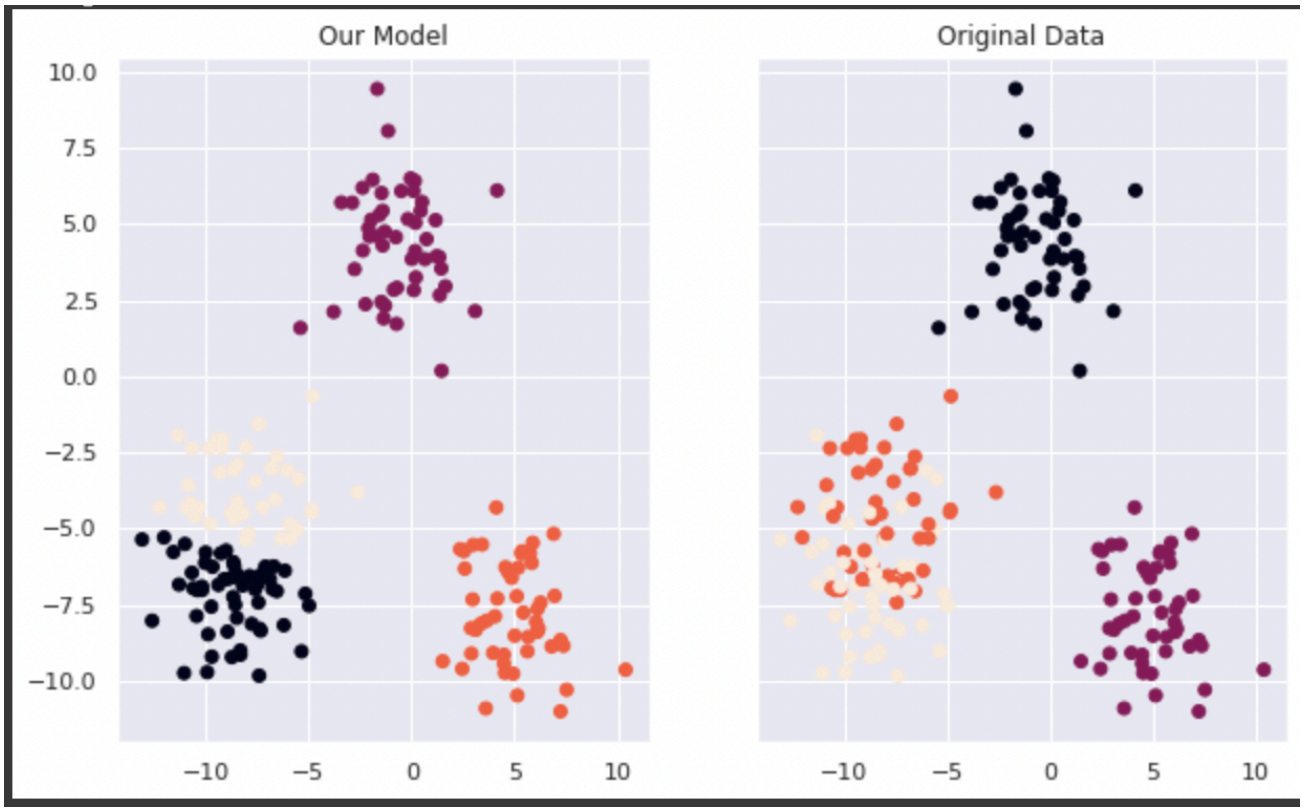
```
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True,figsize=(10,6))

ax1.set_title('Our Model')

ax1.scatter(raw_data[0][:,0], raw_data[0][:,1],c=model.labels_)

ax2.set_title('Original Data')

ax2.scatter(raw_data[0][:,0], raw_data[0][:,1],c=raw_data[1])
```



Conclusion

- Finally we finishes the iris classification project.
- We have built a K Means Clustering which performs well with given features petal length and petal width and having the **Training accuracy of 99.11%** and **Testing accuracy of 100%**.