```
import warnings
warnings.filterwarnings('ignore')
```

K-Nearest-Neighbors

KNN falls in the supervised learning family of algorithms. Informally, this means that we are given a labelled data consiting of training observations (x,y) and would like to capture the relationship between x and y. More formally is to learn a function h:X→Y so that given an unseen observation x, h(x) can confidently predict the correspondir

In this module we will explore the inner workings of KNN, choosing the optimal K values and using KNN from sci

Overview

1.Read the problem statement.

2.Get the dataset.

3.Explore the dataset.

4.Pre-processing of dataset.

5.Visualization

6.Transform the dataset for building machine learning model.

7.Split data into train, test set.

7.Build Model.

8.Apply the model.

9.Evaluate the model.

10.Finding Optimal K value

11.Repeat 7,8,9 steps.

Problem statement

Dataset

The data set we'll be using is the Iris Flower Dataset which was first introduced in 1936 by the famous statisticia Fisher and consists of 50 observations from each of three species of Iris (Iris setosa, Iris virginica and Iris versic features were measured from each sample: the length and the width of the sepals and petals.

Attributes of the dataset: https://archive.ics.uci.edu/ml/datasets/Iris

Train the KNN algorithm to be able to distinguish the species from one another given the measurements of the features.

Question 1

Import the data set and print 10 random rows from the data set

```
from google.colab import drive
drive.mount('/content/drive')
```

```
import numpy as np
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from scipy.stats import zscore
from sklearn.preprocessing import Imputer
from sklearn.metrics import accuracy_score
import seaborn as sns
import os
%matplotlib inline
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=9473

Enter your authorization code:
··········
Mounted at /content/drive
```

```
data = pd.read_csv('/content/drive/My Drive/4aug/assign/Iris.csv')
```

```
data.sample(10)
```

	Sepal Length (in cm)	Sepal Width in (cm)	Petal length (in cm)	Petal width (in cm)	
40	5.0	3.5	1.3	0.3	
44	5.1	3.8	1.9	0.4	
120	6.9	3.2	5.7	2.3	Iri
130	7.4	2.8	6.1	1.9	Iri
9	4.9	3.1	1.5	0.1	
19	5.1	3.8	1.5	0.3	
7	5.0	3.4	1.5	0.2	
84	5.4	3.0	4.5	1.5	
1	4.9	3.0	1.4	0.2	

Data Pre-processing

Question 2 - Estimating missing values

*Its not good to remove the records having missing values all the time. We may end up loosing some data points will have to see how to replace those missing values with some estimated values (median) *

```
data.isnull().values.any()
#there are no missing values in dataframe
```

```
False
```

```
data['Sepal Length (in cm)'] = data['Sepal Length (in cm)'].fillna((data['Sepal Length (in cm)'].
```

Question 3 - Dealing with categorical data

Change all the classes to numericals (0to2).

```
dummy = pd.get_dummies(data['Class'])
data=pd.concat([data,dummy], axis=1)
```

```
data
```

17	5.1	3.5	1.4	0.3	...is-setosa	1	0	0
18	5.7	3.8	1.7	0.3	Iris-setosa	1	0	0
19	5.1	3.8	1.5	0.3	Iris-setosa	1	0	0
20	5.4	3.4	1.7	0.2	Iris-setosa	1	0	0
21	5.1	3.7	1.5	0.4	Iris-setosa	1	0	0
22	4.6	3.6	1.0	0.2	Iris-setosa	1	0	0
23	5.1	3.3	1.7	0.5	Iris-setosa	1	0	0
24	4.8	3.4	1.9	0.2	Iris-setosa	1	0	0
25	5.0	3.0	1.6	0.2	Iris-setosa	1	0	0
26	5.0	3.4	1.6	0.4	Iris-setosa	1	0	0
27	5.2	3.5	1.5	0.2	Iris-setosa	1	0	0
28	5.2	3.4	1.4	0.2	Iris-setosa	1	0	0
29	4.7	3.2	1.6	0.2	Iris-setosa	1	0	0
...	...	...	...	...	...	...	...	...
120	6.9	3.2	5.7	2.3	Iris-virginica	0	0	1
121	5.6	2.8	4.9	2.0	Iris-virginica	0	0	1
122	7.7	2.8	6.7	2.0	Iris-virginica	0	0	1
123	6.3	2.7	4.9	1.8	Iris-virginica	0	0	1
124	6.7	3.3	5.7	2.1	Iris-virginica	0	0	1
125	7.2	3.2	6.0	1.8	Iris-virginica	0	0	1
126	6.2	2.8	4.8	1.8	Iris-virginica	0	0	1
127	6.1	3.0	4.9	1.8	Iris-virginica	0	0	1
128	6.4	2.8	5.6	2.1	Iris-	0	0	1

					virginica			
129	7.2	3.0	5.8	1.6	Iris-virginica	0	0	1
130	7.4	2.8	6.1	1.9	Iris-virginica	0	0	1
131	7.9	3.8	6.4	2.0	Iris-virginica	0	0	1
132	6.4	2.8	5.6	2.2	Iris-virginica	0	0	1
133	6.3	2.8	5.1	1.5	Iris-virginica	0	0	1
134	6.1	2.6	5.6	1.4	Iris-virginica	0	0	1
135	7.7	3.0	6.1	2.3	Iris-virginica	0	0	1
136	6.3	3.4	5.6	2.4	Iris-virginica	0	0	1
137	6.4	3.1	5.5	1.8	Iris-virginica	0	0	1
138	6.0	3.0	4.8	1.8	Iris-virginica	0	0	1
139	6.9	3.1	5.4	2.1	Iris-virginica	0	0	1
140	6.7	3.1	5.6	2.4	Iris-virginica	0	0	1
141	6.9	3.1	5.1	2.3	Iris-virginica	0	0	1
142	5.8	2.7	5.1	1.9	Iris-virginica	0	0	1
143	6.8	3.2	5.9	2.3	Iris-virginica	0	0	1
144	6.7	3.3	5.7	2.5	Iris-virginica	0	0	1
145	6.7	3.0	5.2	2.3	Iris-virginica	0	0	1
					Iris-			

```
data.groupby(["Class"]).count()
```

	Sepal Length (in	Sepal Width	Petal length (in	Petal width	Iris-	Iris-

```
dataClass=data.drop(labels='Class', axis=1)
dataClassz= dataClass.apply(zscore)


        ..

dataClassz
```

	Sepal Length (in cm)	Sepal Width in (cm)	Petal length (in cm)	Petal width (in cm)	Iris-setosa	Iris-versicolor	Iris-virginica
0	-0.900681	1.032057	-1.341272	-1.312977	1.414214	-0.707107	-0.707107
1	-1.143017	-0.124958	-1.341272	-1.312977	1.414214	-0.707107	-0.707107
2	-1.385353	0.337848	-1.398138	-1.312977	1.414214	-0.707107	-0.707107
3	-1.506521	0.106445	-1.284407	-1.312977	1.414214	-0.707107	-0.707107
4	-1.021849	1.263460	-1.341272	-1.312977	1.414214	-0.707107	-0.707107
5	-0.537178	1.957669	-1.170675	-1.050031	1.414214	-0.707107	-0.707107
6	-1.506521	0.800654	-1.341272	-1.181504	1.414214	-0.707107	-0.707107
7	-1.021849	0.800654	-1.284407	-1.312977	1.414214	-0.707107	-0.707107
8	-1.748856	-0.356361	-1.341272	-1.312977	1.414214	-0.707107	-0.707107
9	-1.143017	0.106445	-1.284407	-1.444450	1.414214	-0.707107	-0.707107
10	-0.537178	1.494863	-1.284407	-1.312977	1.414214	-0.707107	-0.707107
11	-1.264185	0.800654	-1.227541	-1.312977	1.414214	-0.707107	-0.707107
12	-1.264185	-0.124958	-1.341272	-1.444450	1.414214	-0.707107	-0.707107
13	-1.870024	-0.124958	-1.511870	-1.444450	1.414214	-0.707107	-0.707107
14	-0.052506	2.189072	-1.455004	-1.312977	1.414214	-0.707107	-0.707107
15	-0.173674	3.114684	-1.284407	-1.050031	1.414214	-0.707107	-0.707107
16	-0.537178	1.957669	-1.398138	-1.050031	1.414214	-0.707107	-0.707107
17	-0.900681	1.032057	-1.341272	-1.181504	1.414214	-0.707107	-0.707107
18	-0.173674	1.726266	-1.170675	-1.181504	1.414214	-0.707107	-0.707107
19	-0.900681	1.726266	-1.284407	-1.181504	1.414214	-0.707107	-0.707107
20	-0.537178	0.800654	-1.170675	-1.312977	1.414214	-0.707107	-0.707107
21	-0.900681	1.494863	-1.284407	-1.050031	1.414214	-0.707107	-0.707107
22	-1.506521	1.263460	-1.568735	-1.312977	1.414214	-0.707107	-0.707107
23	-0.900681	0.569251	-1.170675	-0.918558	1.414214	-0.707107	-0.707107
24	-1.264185	0.800654	-1.056944	-1.312977	1.414214	-0.707107	-0.707107
25	-1.021849	-0.124958	-1.227541	-1.312977	1.414214	-0.707107	-0.707107
26	-1.021849	0.800654	-1.227541	-1.050031	1.414214	-0.707107	-0.707107
27	-0.779513	1.032057	-1.284407	-1.312977	1.414214	-0.707107	-0.707107
28	-0.779513	0.800654	-1.341272	-1.312977	1.414214	-0.707107	-0.707107
29	-1.385353	0.337848	-1.227541	-1.312977	1.414214	-0.707107	-0.707107
...	...	...	...	...	...	...	...

120	1.280340	0.337848	1.103953	1.447956	-0.707107	-0.707107	1.414214
121	-0.294842	-0.587764	0.649027	1.053537	-0.707107	-0.707107	1.414214
122	2.249683	-0.587764	1.672610	1.053537	-0.707107	-0.707107	1.414214
123	0.553333	-0.819166	0.649027	0.790591	-0.707107	-0.707107	1.414214
124	1.038005	0.569251	1.103953	1.185010	-0.707107	-0.707107	1.414214
125	1.643844	0.337848	1.274550	0.790591	-0.707107	-0.707107	1.414214
126	0.432165	-0.587764	0.592162	0.790591	-0.707107	-0.707107	1.414214
127	0.310998	-0.124958	0.649027	0.790591	-0.707107	-0.707107	1.414214
128	0.674501	-0.587764	1.047087	1.185010	-0.707107	-0.707107	1.414214
129	1.643844	-0.124958	1.160819	0.527645	-0.707107	-0.707107	1.414214
130	1.886180	-0.587764	1.331416	0.922064	-0.707107	-0.707107	1.414214
131	2.492019	1.726266	1.502013	1.053537	-0.707107	-0.707107	1.414214
132	0.674501	-0.587764	1.047087	1.316483	-0.707107	-0.707107	1.414214
133	0.553333	-0.587764	0.762759	0.396172	-0.707107	-0.707107	1.414214
134	0.310998	-1.050569	1.047087	0.264699	-0.707107	-0.707107	1.414214
135	2.249683	-0.124958	1.331416	1.447956	-0.707107	-0.707107	1.414214
136	0.553333	0.800654	1.047087	1.579429	-0.707107	-0.707107	1.414214
137	0.674501	0.106445	0.990221	0.790591	-0.707107	-0.707107	1.414214
138	0.189830	-0.124958	0.592162	0.790591	-0.707107	-0.707107	1.414214
139	1.280340	0.106445	0.933356	1.185010	-0.707107	-0.707107	1.414214
140	1.038005	0.106445	1.047087	1.579429	-0.707107	-0.707107	1.414214
141	1.280340	0.106445	0.762759	1.447956	-0.707107	-0.707107	1.414214
142	-0.052506	-0.819166	0.762759	0.922064	-0.707107	-0.707107	1.414214
143	1.159173	0.337848	1.217684	1.447956	-0.707107	-0.707107	1.414214
144	1.038005	0.569251	1.103953	1.710902	-0.707107	-0.707107	1.414214
145	1.038005	-0.124958	0.819624	1.447956	-0.707107	-0.707107	1.414214
146	0.553333	-1.281972	0.705893	0.922064	-0.707107	-0.707107	1.414214
147	0.795669	-0.124958	0.819624	1.053537	-0.707107	-0.707107	1.414214

```
target=data['Class']
```

```
target
```

```
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
5         Iris-setosa
6         Iris-setosa
7         Iris-setosa
8         Iris-setosa
9         Iris-setosa
10        Iris-setosa
11        Iris-setosa
12        Iris-setosa
13        Iris-setosa
14        Iris-setosa
15        Iris-setosa
16        Iris-setosa
17        Iris-setosa
18        Iris-setosa
19        Iris-setosa
20        Iris-setosa
21        Iris-setosa
22        Iris-setosa
23        Iris-setosa
24        Iris-setosa
25        Iris-setosa
26        Iris-setosa
27        Iris-setosa
28        Iris-setosa
29        Iris-setosa
            ...
120    Iris-virginica
121    Iris-virginica
122    Iris-virginica
123    Iris-virginica
124    Iris-virginica
125    Iris-virginica
126    Iris-virginica
127    Iris-virginica
128    Iris-virginica
129    Iris-virginica
130    Iris-virginica
131    Iris-virginica
132    Iris-virginica
133    Iris-virginica
134    Iris-virginica
135    Iris-virginica
136    Iris-virginica
137    Iris-virginica
138    Iris-virginica
139    Iris-virginica
140    Iris-virginica
141    Iris-virginica
142    Iris-virginica
143    Iris-virginica
144    Iris-virginica
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
```

```
Name: Class, Length: 150, dtype: object
```

```
x=  np.array(dataClassz)
```

```
x
```

```
array([[-0.90068117,  1.03205722, -1.3412724 , ...,  1.41421356,
        -0.70710678, -0.70710678],
       [-1.14301691, -0.1249576 , -1.3412724 , ...,  1.41421356,
        -0.70710678, -0.70710678],
       [-1.38535265,  0.33784833, -1.39813811, ...,  1.41421356,
        -0.70710678, -0.70710678],
       ...,
       [ 0.79566902, -0.1249576 ,  0.81962435, ..., -0.70710678,
        -0.70710678,  1.41421356],
       [ 0.4321654 ,  0.80065426,  0.93335575, ..., -0.70710678,
        -0.70710678,  1.41421356],
       [ 0.06866179, -0.1249576 ,  0.76275864, ..., -0.70710678,
        -0.70710678,  1.41421356]])
```

```
y=np.array(target)
```

```
y
```

```
array(['Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

Question 4

Observe the association of each independent variable with target variable and drop variables from feature set havi correlation in range -0.1 to 0.1 with target variable.

Question 5

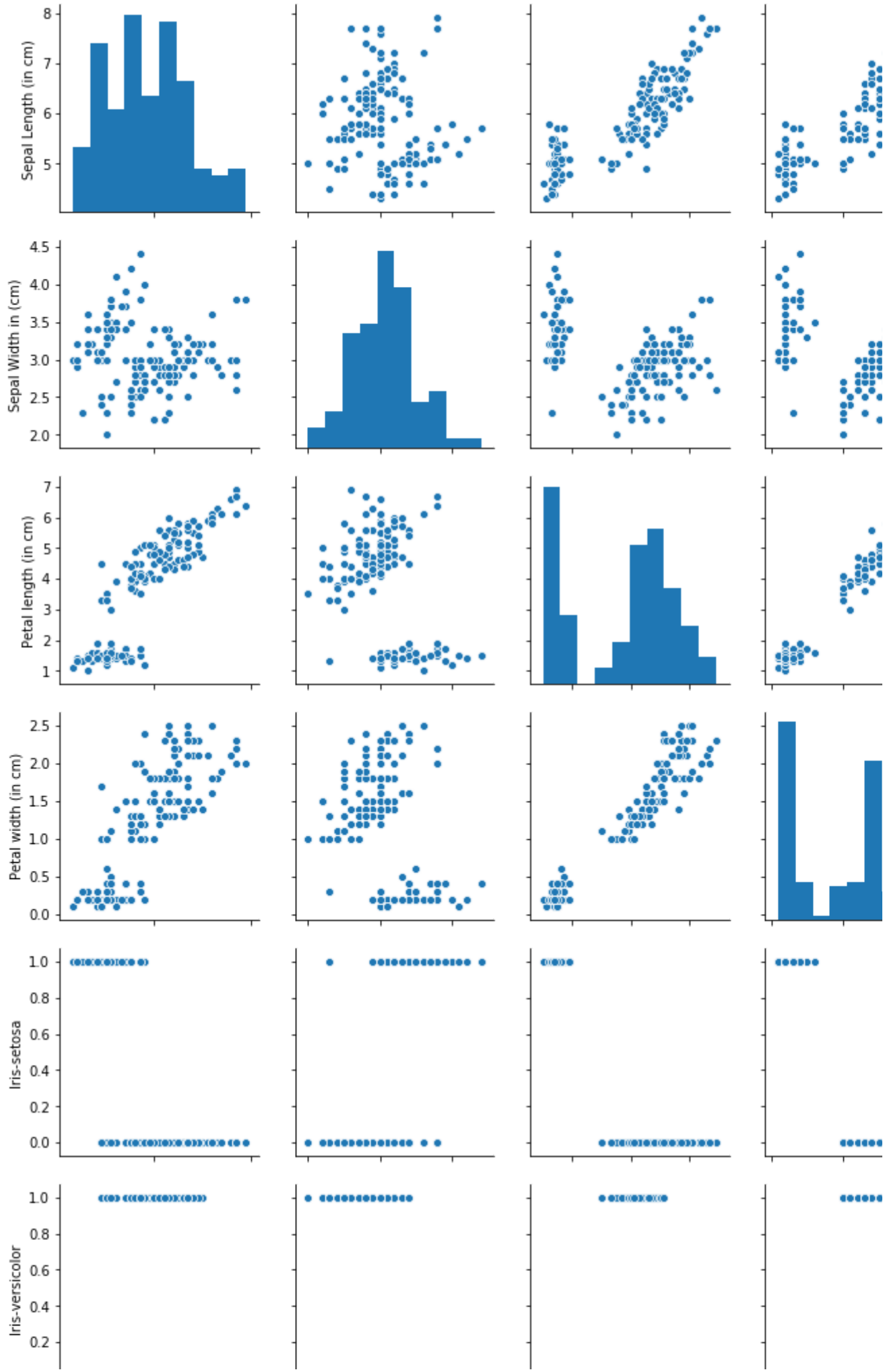
Observe the independent variables variance and drop such variables having no variance or almost zero variance(va 0.1). They will be having almost no influence on the classification.

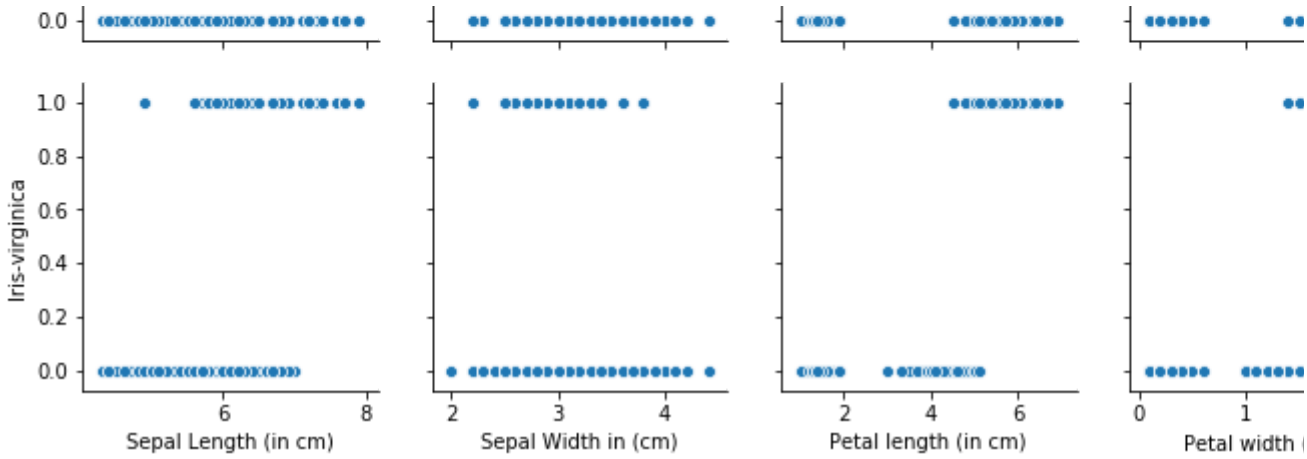
Question 6

Plot the scatter matrix for all the variables.

```
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x7fd6173ae278>
```





Split the dataset into training and test sets

Question 7

Split the dataset into training and test sets with 80-20 ratio.

```
from sklearn.model_selection import train_test_split
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.20, random_state=1)
```

```
NNH= KNeighborsClassifier(n_neighbors= 5 , weights = 'distance')
```

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

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='distance')
```

```
predicted_labels= NNH.predict(x_test)
```

```
NNH.score(x_test,y_test)
```

```
1.0
```

Question 8 - Model

*Build the model and train and test on training and test sets respectively using **scikit-learn**. Print the Accuracy of th with different values of **k=3,5,9**.*

Hint: For accuracy you can check **accuracy_score()** in scikit-learn

```
from sklearn import   metrics
```

```
print(metrics.confusion_matrix(y_test, predicted_labels))
```

```
[[11  0  0]
 [ 0 13  0]
 [ 0  0  6]]
```

Question 9 - Cross Validation

Run the KNN with no of neighbours to be 1,3,5..19 and *Find the* *optimal number of neighbours** from the above the Mis classification error

Hint:

Misclassification error (MSE) = 1 - Test accuracy score. Calculated MSE for each model with neighbours = 1,3,5. find the model with lowest MSE

Question 10

Plot misclassification error vs k (with k value on X-axis) using matplotlib.

Naive Bayes with Iris Data

```
#Load all required library
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
from sklearn import datasets
from sklearn.decomposition import PCA
```

Slice Iris data set for Independent variables and dependent variables

Please note 'Species' is my dependent variables, name it y and independent set data as X

```
#Check the dataset
print(y)
print(X)
```

Question 11

Find the distribution of target variable (Class)

And, Plot the distribution of target variable using histogram

```
#Drop Id variable from data
```

Question 12

Find Correlation among all variables and give your insights

```
#Please note, it's Require to remove correlated features because they are voted twice in the mode
## it can lead to over inflating importance.We will ignore it here
```

Split data in Training and test set in 80:20.

Question 13

Do Feature Scaling

```
# Use StandardScaler or similar methods
```

Question 14

Train and Fit NaiveBayes Model

```
#Fit the model
```

```
#Predict
```

Question 15

Print Accuracy and Confusion Matrix and Conclude your findings

```
# show Confusion Matrix
```

```
# show accuracy
```

```
#Show precision and Recall metrics
```