```
import pandas as pd
import numpy as np

df=pd.read_csv(r'C:\Users\admin\Desktop\VNR\3-2\FMLT\External practice\Iris.csv')
df

df.shape
```

```
(150, 6)
```

```
df.columns
```

```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

```
df[['SepalLengthCm']]
```

	SepalLengthCm
0	5.1
1	4.9
2	4.7
3	4.6
4	5.0
...	...
145	6.7
146	6.3
147	6.5
148	6.2
149	5.9

150 rows × 1 columns

```
df.rename(columns={'Id':'ID'},inplace=True)
```

```
df.columns
```

```
Index(['ID', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

```
df1=pd.DataFrame({'id':[1,2,3],
                  'name':['a','b','c']})
df2=pd.DataFrame({'sal':['4k','5k','6k'],
                  'name':[50,60,70]})
bind_rows=pd.concat([df1,df2],axis=1)
bind_rows
```

	id	name	sal	name
0	1	a	4k	50
1	2	b	5k	60
2	3	c	6k	70

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

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

```
for i in df.columns:
    print(df[i].mean())
```

```
df.describe()
```

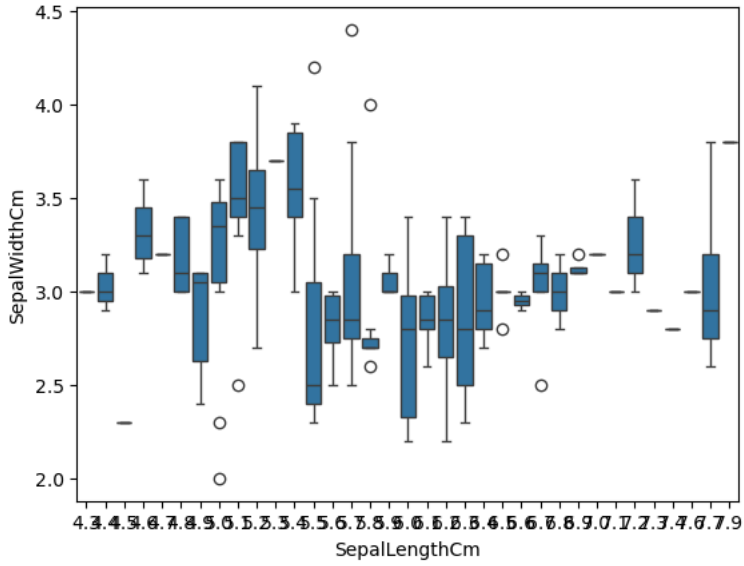
	ID	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

```
q1=np.percentile(df['SepalLengthCm'],25,method='midpoint')
q1
```

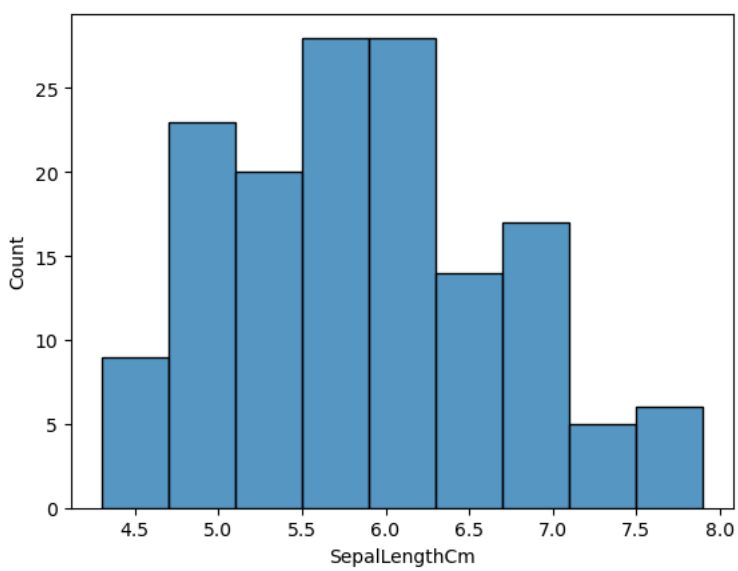
```
5.1
```

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.boxplot(x='SepalLengthCm',y='SepalWidthCm',data=df)
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```




```
sns.histplot(x='SepalLengthCm',data=df)
plt.show()
```



```
df4=pd.DataFrame({'val':[1,2,3,4,5,6,7,8,9,10]})
df4['new_val']=df4['val'].clip(lower=4,upper=8)
print(df4)
```

```
   val  new_val
0    1        4
1    2        4
2    3        4
3    4        4
4    5        5
5    6        6
6    7        7
7    8        8
8    9        8
9   10        8
```

```
df5=pd.DataFrame({'val':[1,np.nan,3,np.nan,5]})
df5['new_val']=df5['val'].fillna(df5['val'].mean())
df5
```

	val	new_val
0	1.0	1.0
1	NaN	3.0
2	3.0	3.0
3	NaN	3.0
4	5.0	5.0

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split,KFold
from sklearn.metrics import mean_squared_error
from sklearn.utils import resample
x=np.array([[1,2,3,4,5],
            [2,3,4,5,6],
            [3,4,5,6,7],
            [4,5,6,7,8],
            [5,6,7,8,9]])
y=np.array([10,20,30,40,50])
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
fitting=LinearRegression()
fitting.fit(x_train,y_train)
testing=fitting.predict(x_test)
ans=mean_squared_error(y_test,testing)
ans
```

```
5.048709793414476e-29
```

```
kf=KFold(n_splits=5,shuffle=True)
li=[]
for i,j in kf.split(x):
  x_train_kfold,x_test_kfold=x[i],x[j]
  y_train_kfold,y_test_kfold=y[i],y[j]
  fitting_kf=LinearRegression()
  fitting_kf.fit(x_train_kfold,y_train_kfold)
  testing_kf=fitting_kf.predict(x_test_kfold)
  ans_kf=mean_squared_error(y_test_kfold,testing_kf)
  li.append(ans_kf)
new_ans=np.mean(li)
print(new_ans)
```

```
6.05845175209737e-29
```

```
n=100
new_li=[]
for i in range(n):
    x_train_boo,y_train_boo=resample(x,y,random_state=42)
    model_boo=LinearRegression()
    model_boo.fit(x_train_boo,y_train_boo)
    result_boo=model_boo.predict(x_test)
    ans_boo=mean_squared_error(y_test,result_boo)
    new_li.append(ans_boo)
print(np.mean(new_li))
```

```
5.048709793414476e-29
```

```
#from week 4 approx
```

```
import pandas as pd
import numpy as np
```

```
df = pd.read_csv(r'C:\Users\admin\Desktop\VNR\3-2\FMLT\External practice\Iris.csv')
```

```
df
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

Week 4

capping of values

```
df['SepalLengthCm'].clip(2, 4) #lite
```

```
0      4.0
1      4.0
2      4.0
3      4.0
4      4.0
      ...
145    4.0
146    4.0
147    4.0
148    4.0
149    4.0
Name: SepalLengthCm, Length: 150, dtype: float64
```

```
from sklearn.impute import SimpleImputer
```

Week 5

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_squared_error
```

```
X = np.array(
    [
        [1,2,3,4,5],
        [2,3,4,5,6],
        [3,4,5,6,7],
        [4,5,6,7,8],
        [5,6,7,8,9]
    ]
)
y = np.array([10, 20, 30, 40, 50])
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
y_pred = lr.predict(X_test)
```

```
print(X_test, y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
print('Holdout method mse: ', mse)
```

```
[[2 3 4 5 6]] [20] [20.]
Holdout method mse:  0.0
```

K-Fold Cross Validation

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

kf = KFold(n_splits=2, shuffle=True, random_state=42)
lr1 = LinearRegression()
kfcv_scores = cross_val_score(lr1, X, y, cv=kf)
print(kfcv_scores.mean())
```

```
1.0
```

Bootstrap Sampling

```
#method 1 for linear regression
from sklearn.utils import resample
n_iterations = 100
mse_scores_bootstrap = []
for _ in range(n_iterations):
    Xb, yb = resample(X, y, random_state=42)
    mb = LinearRegression()
    mb.fit(X_train, y_train)
    y_pred_bs = mb.predict(X_test)
    mse_bs = mean_squared_error(y_test, y_pred_bs)
    mse_scores_bootstrap.append(mse_bs)

print(np.mean(mse_scores_bootstrap))

# method 2 for random forest classifier
```

```
0.0
```

```
#week 5 second part
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score, KFold, ShuffleSplit
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LinearRegression

# X = np.array(
#     [
#         [1,2,3,4,5],
#         [2,3,4,5,6],
#         [3,4,5,6,7],
#         [4,5,6,7,8],
#         [5,6,7,8,9]
#     ]
# )
# y = np.array([10, 20, 30, 40, 50])
X = df.iloc[:, 1:-1]
y = df.iloc[:, -1:]
```

```
#holdout
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)


holdout_score = rf.score(X_test, y_test)
print(holdout_score)
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
```

```
1.0
```

```
#K-fold
kf = KFold(n_splits=5, shuffle=True)
rf = RandomForestClassifier(random_state=42)
kf_scores = cross_val_score(rf, X, y, cv=kf)
print(kf_scores.mean())
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
0.9600000000000002
```

```
#Bootstrap
bs = ShuffleSplit(n_splits=5, test_size=0.2)
bs_scores = cross_val_score(rf, X ,y, cv=bs)
print(bs_scores.mean())
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
0.9800000000000001
```

Week 6

```
#classification
    #decision tree

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

X = df.iloc[:, 1:-1]
y = np.array(df.iloc[:, -1:])
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

dt = DecisionTreeClassifier(criterion='entropy')
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
dt_acc = accuracy_score(y_test, y_pred)
dt_pre = precision_score(y_test, y_pred, average='weighted')
dt_f1 = f1_score(y_test, y_pred, average='weighted')
dt_rec = recall_score(y_test, y_pred, average='weighted')

print(dt_acc, dt_pre, dt_rec, dt_f1)
```

```
0.9333333333333333 0.9422222222222222 0.9333333333333333 0.931547619047619
```

```
from sklearn.ensemble import RandomForestClassifier

dt = RandomForestClassifier()
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)
dt_acc = float(accuracy_score(y_test, y_pred))
dt_pre = float(precision_score(y_test, y_pred, average='weighted'))
dt_f1 = float(f1_score(y_test, y_pred, average='weighted'))
dt_rec = float(recall_score(y_test, y_pred, average='weighted'))

print(dt_acc, dt_pre, dt_rec, dt_f1)
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\base.py:1474: DataConversionWarning: A column-vector y was passed when a 1d array
  return fit_method(estimator, *args, **kwargs)
```

```
0.9666666666666667 0.9690476190476189 0.9666666666666667 0.9663035584604212
```

Week 6: Regression

```
y
```

```
       ['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']], dtype=object)
```

```
df
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

```
from sklearn.preprocessing import LabelEncoder
y_danger = LabelEncoder().fit_transform(y)
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\preprocessing\_label.py:114: DataConversionWarning: A column-vector y was passed
  y = column_or_1d(y, warn=True)
```

```
lrw6 = LinearRegression()
X_train, X_test, y_train, y_test = train_test_split(X, y_danger, test_size=0.2)
lrw6.fit(X_train, y_train)

lrw6.score(X_test, y_test)
```

```
0.9541117998395531
```

Week 6: Clustering

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3)

kmeans.fit(X)
```

```
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarni
  warnings.warn(
```

```
▾ KMeans
KMeans(n_clusters=3)
```

```
from sklearn.metrics import silhouette_score
from sklearn.metrics.cluster import contingency_matrix

print(kmeans.labels_)
s = silhouette_score(X, kmeans.labels_)
c = contingency_matrix(y, kmeans.labels_)

purity = np.sum(np.amax(c, axis=0))/np.sum(c)
print(s, purity)
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 0 2 2 2 2
 2 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2 2 0 2 2 2 0 2 2 2 0 2
 2 0]
0.5525919445499757 0.8933333333333333
```

Week 7: const and extrac

```
#Dummy coding categorical(nominal) variables.
dummies = pd.get_dummies(df.Species)

new_df = pd.concat([df,dummies], axis=1)
new_df = new_df.drop(['Species'], axis=1)
new_df
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Iris-setosa	Iris versicolo
0	1	5.1	3.5	1.4	0.2	True	Fals
1	2	4.9	3.0	1.4	0.2	True	Fals
2	3	4.7	3.2	1.3	0.2	True	Fals
3	4	4.6	3.1	1.5	0.2	True	Fals
4	5	5.0	3.6	1.4	0.2	True	Fals
...	...	...	...	...	...	...	.
145	146	6.7	3.0	5.2	2.3	False	Fals
146	147	6.3	2.5	5.0	1.9	False	Fals
147	148	6.5	3.0	5.2	2.0	False	Fals
148	149	6.2	3.4	5.4	2.3	False	Fals
149	150	5.9	3.0	5.1	1.8	False	Fals

```
#Encoding categorical(ordinal) variables.

from sklearn.preprocessing import LabelEncoder

df['New Species'] = LabelEncoder().fit_transform(df['Species'])

df
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species	New Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa	0
1	2	4.9	3.0	1.4	0.2	Iris-setosa	0
2	3	4.7	3.2	1.3	0.2	Iris-setosa	0
3	4	4.6	3.1	1.5	0.2	Iris-setosa	0
4	5	5.0	3.6	1.4	0.2	Iris-setosa	0
...	...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica	2

```
#Transforming numeric(continuous)features to categorical features

ratings = pd.DataFrame({'rating': [1, 4, 6, 8 ,3 ,2 ,6 ,7 ,8 ,9 , 4, 5, 3, 6,7,9, 10]})
ratings['category'] = pd.cut(ratings.rating, bins=[0, 4, 7, 10], labels=['bad', 'avg', 'good'])
ratings
```

	rating	category
0	1	bad
1	4	bad
2	6	avg
3	8	good
4	3	bad
5	2	bad
6	6	avg
7	7	avg
9	9	good

Week 7: Feature Extraction

```
# Principal Component Analysis (PCA)
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
print(X.shape)

new_X = pca.fit_transform(X)

# new_X = pca.transform(X)
print(new_X.shape)

print('pca: ', pca.explained_variance_ratio_)

# Linear Discriminant Analysis (LDA)
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

lda = LinearDiscriminantAnalysis(n_components=2)
x_r2 = lda.fit_transform(X, y)
print('lda: ', lda.explained_variance_ratio_)
# Feature Subset Selection
```

```
(150, 4)
(150, 2)
pca:  [0.92461621 0.05301557]
lda:  [0.99147248 0.00852752]
C:\Users\admin\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1300: DataConversionWarning: A column-vector y was passed whe
  y = column_or_1d(y, warn=True)
```

Week 7: Feature Subset Selection

```
import seaborn as sns
sns.pairplot(df.drop(['Id'], axis=1), hue='New Species', height=1)
```

```
<seaborn.axisgrid.PairGrid at 0x1889be2f680>
```