Problem Statement -Write a python program to create a data frame by uploading a csv file and carry out the basic operation of numpy such as finding the maximum value from the data set indexing and slicing of the data frame and to find the shape and dimension of the data framework .

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
data = pd.read_csv('/content/Iris.csv')
numeric_data = data.drop(columns=['Id',
'Species'])
array_data = numeric_data.to_numpy()
print("Shape of the array:", array_data.shape)
print("Dimensions of the array:", array_data.ndim)
print("Data type of the array:", array_data.dtype)
print("First few rows of the array:")
print(array_data[:5])
print("Value at row 0, column 1:", array_data[0, 1])
print("Sliced array (row 0 to 2, column 0 to 2):")
print(array_data[:3, :3])
array_sum = np.sum(array_data)
print("Sum of all elements in the array:",
array_sum)
array_diff = np.diff(array_data, axis=0)
print("Difference between consecutive elements in
each row:")
print(array_diff[:5])
array_product = np.prod(array_data)
print("Product of all elements in the array:",
array_product)
array_mean = np.mean(array_data, axis=0)
print("Mean of each column in the array:")
print(array_mean)
sns.pairplot(data, hue='Species')
plt.show()
```

Problem Statement - Write a python program to carry out a visualization for each feature separately.

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names
plt.figure(figsize=(12, 6))
for i in range(X.shape[1]):
    plt.subplot(2, 2, i+1)
    sns.histplot(X[:, i], kde=True, color='skyblue')
    plt.title(feature_names[i])
plt.tight_layout()
plt.show()
sns.pairplot(sns.load_dataset('iris'), hue='2es')
plt.show()
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
plt.figure(figsize=(8, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1],
hue=target_names[y], palette='viridis')
plt.title('PCA Visualization')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

Problem Statement -Write a python program to implement logistic regression on California_housing dataset.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.linear_model import
LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,
confusion_matrix
df =
pd.read_csv('/content/sample_data/california_housi
ng_train.csv')
df.dropna(inplace=True)
X = df.drop('median_house_value', axis=1
y = df['median_house_value']
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = LogisticRegression()
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Problem Statement -Write a python program to implement ID3 algorithm using entropy in decision tree.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,
confusion_matrix
df =
pd.read_csv('/content/sample_data/california_housi
ng_train.csv')
df.dropna(inplace=True)
X = df.drop('median_house_value', axis=1)
y = df['median_house_value']
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model =
DecisionTreeClassifier(criterion='entropy')
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Confusion Matrix:")
```

Problem Statement -Write a python program to implement CART algo for decision tree.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,
confusion_matrix
df =
pd.read_csv('/content/sample_data/california_housi
ng_train.csv')
df.dropna(inplace=True)
X = df.drop('median_house_value', axis=1
y = df['median_house_value'
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = DecisionTreeClassifier(criterion='gini')
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Problem Statement -Write a python program to implement SVM using linear kernel on iris.csv.

```
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.preprocessing import
StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import
classification_report, accuracy_score
url = "https://archive.ics.uci.edu/ml/machine-
learning-databases/iris/iris.data"
column_names = ['sepal_length', 'sepal_width',
'petal_length', 'petal_width', 'species']
iris = pd.read_csv(url, header=None,
names=column_names)
print(iris.head())
X = iris.iloc[:, :-1].values
y = iris.iloc[:, -1].values
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=42)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
svm = SVC(kernel='sigmoid', random_state=42)
svm.fit(X_train, y_train)
y_pred = svm.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
print(classification_report(y_test, y_pred))
```

Problem Statement - Write a python program to implement SVM using sigmoid kernel on iris.csv and write a python program to implement k-NN on iris.csv with k=3

```
 import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.preprocessing import
StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import
classification_report, accuracy_score
url = "https://archive.ics.uci.edu/ml/machine-
learning-databases/iris/iris.data"
column_names = ['sepal_length', 'sepal_width',
'petal_length', 'petal_width', 'species']
iris = pd.read_csv(url, header=None,
names=column_names)
print(iris.head())
X = iris.iloc[:, :-1].values  # all columns except
the last one
y = iris.iloc[:, -1].values   # the last column
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3,
random_state=42)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
svm = SVC(kernel='sigmoid', random_state=42)
svm.fit(X_train, y_train)
y_pred = svm.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
print(classification_report(y_test, y_pred))
```