Program 1

```
import pandas as pd
import matplotlib.pyplot as plt
data={'year':list(range(2010,2021)),'Job
posting':[150,300,450,600,800,1200,1600,2100,2700,3400,4200]}
df=pd.DataFrame(data)
plt.plot(df['year'],df['Job posting'],marker='o')
plt.title('Trend of data science job posting')
plt.xlabel('year')
plt.ylabel('number of job posting')
plt.show()
```

Program 2

```
import matplotlib.pyplot as plt
roles=['Data Analyst','Data science','Data engineer']
posting=[100,50,200]
plt.bar(roles,posting,width=0.2)
plt.title('Distribution of various data science role')
plt.xlabel('Job')
plt.ylabel('no of posting')
plt.show()
```

Program 3

```
import pandas as pd
sd=pd.DataFrame(
      {
         "ID":[100,101,102],
         "Name":['Kumaran','Lingeswaran','Lohit'],
         "Age":[17,18,18]
      })
print('Structured Data\n',sd)
print('')
unsd='This is a unstructured data which contain audio, video,combination of data type.'
print('Unstructured Data\n',unsd,'\n')
semisd='{"Name":"Raju","Age":40,"Job":"Software engineer","Hobby":"Gamming"}'
print('Semistructured Data\n',semisd)
```

Program 4

```
from cryptography.fernet import Fernet
key= Fernet.generate_key()
f=Fernet(key)
token=f.encrypt(b"hi I am Lingeswaran")
token
b'...'
f.decrypt(token)
b'hi I am Lingeswaran'
key=Fernet.generate_key()
cipher_suite=Fernet(key)
plain_text=b'hi I am Lingeswaran'
cipher_text=cipher_suite.encrypt(plain_text)
decrypted_text=cipher_suite.decrypt(cipher_text)
print('Original data',plain_text)
print('Encrypted data',cipher_text)
print('Decrypted data',decrypted_text)
```

Program 5

```
import matplotlib.pyplot as cricket
Overs=list(range(5,51,5))
Indian_Score=[30,55,90,129,165,200,239,270,310,350]
Srilankan_Score=[25,70,90,120,140,170,195,220,255,279]
cricket.plot(Overs,Indian_Score)
cricket.plot(Overs,Srilankan_Score)
cricket.show()
cricket.title("INDIA Vs SRILANKA" )
cricket.xlabel(" Overs" )
cricket.ylabel(" Score" )
cricket.legend()
cricket.plot(Overs,Indian_Score,color=" green" ,label=" INDIA" )
cricket.plot(Overs,Srilankan_Score,color=" red" ,label=" SRILANKA" )
cricket.legend(loc=" center right" )
```

Program 6

```
import matplotlib.pyplot as hscmark
import numpy as np
Names = [ 'SHREE ',  'DEV ',  'KEERTHI ',  'PRIYA ',  'SHAN ',  'KUMARAN ']
xaxis = np.arange(len(Names))
Percentage_hsc = [96, 91, 94, 75, 45, 81]
hscmark.bar(Names, Percentage_hsc)
hscmark.xticks(xaxis, Names, rotation=45)
hscmark.xlabel( "Names of Pupil ")
hscmark.ylabel( "Percentage ")
hscmark.title( "Comparison of HSC Percentage ", fontsize=20, color= "green ")
hscmark.show()
```

Program 7

```
import matplotlib.pyplot as election
# Election data
labels = [ 'CANDIDATE 1 ',  'CANDIDATE 2 ',  'CANDIDATE 3 ',  'CANDIDATE 4 ']
Votes = [315, 130, 245, 210]
colors = [ 'green ',  'yellow ',  'red ',  'orange ']
explode = (0.2, 0, 0, 0)
# Plotting the pie chart
election.pie(Votes, labels=labels, colors=colors, explode=explode, autopct= '%0.2f%% ')
election.title( 'Election Results ')
election.show()
```

Program 8

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import gutenberg
# Download the gutenberg corpus if not already installed
nltk.download( 'gutenberg ')
nltk.download( 'punkt ') # Ensure the punkt tokenizer models are also downloaded
# Load the text from the gutenberg corpus
sample = gutenberg.raw( "austen-emma.txt ") # Change to an existing text from the corpus
# Tokenize the sample text
token = word_tokenize(sample)
```

```
# Create a list of the first 50 tokens
wlist = []
for i in range(50):
wlist.append(token[i])
# Calculate the frequency of each word in the list
wordfreq = [wlist.count(w) for w in wlist]
# Print the word-frequency pairs
print( "Pairs\n " + str(list(zip(wlist, wordfreq))))
```

Program 9

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load dataset
data = pd.read_csv( '/kaggle/input/stores-area-and-sales-data/Stores.csv ')

# Summary Statistics
print(data.describe())

# univariate analysis
data[ 'Store_Sales '].hist(bins=20)
plt.title( ' Store Sales Distribution ')
plt.show()

data[ 'Daily_Customer_Count '].hist(bins=20)
plt.title( 'customer count  ')
plt.show()

# Bivariate Analysis
sns.scatterplot(x= 'Store_Sales ', y= 'Daily_Customer_Count ', data=data)
plt.title( 'Sales vs customer count ')
plt.show()

sns.heatmap(data.corr(), annot=True, cmap= 'coolwarm ')
plt.title( 'Correlation Matrix ')
plt.show()
```

Program 10

```
import numpy as np
import pandas as pd
df=pd.read_csv('Social_Network_Ads.csv')
features=df.iloc[:,[2,3]].values
label=df.iloc[:,4].values
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
for i in range(1,401):
    x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2)
    model=LogisticRegression()
    model.fit(x_train,y_train)
    train_score=model.score(x_train,y_train)
    test_score=model.score(x_test,y_test)
    if test_score>train_score:
        print("Test {} Train{} Random State {}".format(test_score,train_score,i))

x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2,
finalModel=LogisticRegression()
finalModel.fit(x_train,y_train)
print(finalModel.score(x_train,y_train))
print(finalModel.score(x_test,y_test))
from sklearn.metrics import classification_report
print(classification_report(label,finalModel.predict(features)))
```

Program 11

```
import numpy as np
import pandas as pd
df=pd.read_csv('Salary_data.csv')
df.dropna(inplace=True)
df.describe()
features=df.iloc[:,[0]].values
label=df.iloc[:,[1]].values
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2,random_state=20)
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
```

```
model.score(x_train,y_train)
model.score(x_test,y_test)
yr_of_exp=float(input("Enter Years of Experience: "))
yr_of_exp_NP=np.array([[yr_of_exp]])
Salary=model.predict(yr_of_exp_NP)
print("Estimated Salary for {} years of experience is {}: " .format(yr_of_exp,Salary))
```