1.Demonstrate the following data preprocessing tasks using python libraries.

a) Loading the dataset

b) Identifying the dependent and independent variables

c) Dealing with missing data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

dataset = pd.read_csv("suv_data.csv")
dataset.head()
```

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

```
x = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

bool_series=pd.isnull(dataset["Gender"])
dataset[bool_series]
bool_series=pd.notnull(dataset["Gender"])
dataset[bool_series]
dataset[10:25]
```

	User ID	Gender	Age	EstimatedSalary	Purchased
10	15570769	Female	26	80000	0
11	15606274	Female	26	52000	0
12	15746139	Male	20	86000	0
13	15704987	Male	32	18000	0
14	15628972	Male	18	82000	0
15	15697686	Male	29	80000	0
16	15733883	Male	47	25000	1
17	15617482	Male	45	26000	1
18	15704583	Male	46	28000	1
19	15621083	Female	48	29000	1
20	15649487	Male	45	22000	1
21	15736760	Female	47	49000	1
22	15714658	Male	48	41000	1
23	15599081	Female	45	22000	1
24	15705113	Male	46	23000	1

```
new_data=dataset.dropna(axis=0,how='any')
```

new_data

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

```
dataset.replace(to_replace=np.nan,value=-99)
dataset["Gender"].fillna("No Gender",inplace=True)
```

dataset

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

```
print("Old data frame length:", len(dataset))
```

```
print("New data frame length:", len(dataset))
```

```
print("Number of rows with at least 1 NA value:", len(dataset)-len(new_data))
```

```
Old data frame length: 400
New data frame length: 400
Number of rows with at least 1 NA value: 0
```

```
new_df1=dataset.fillna(method="ffill")
```

```
new_df1
```

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

new_df3=dataset.dropna(how='all')

new_df3

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
...	...	...	...	...	...
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

400 rows × 5 columns

2. Demonstrate the following data preprocessing tasks using python library

a) Dealing with categorical data

b) Scaling the features

c) Splitting dataset into Training and Testing Sets

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('Titanic-Dataset.csv')
print(data)
```

```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           ...       ...     ...
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

                                                  Name     Sex   Age  SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                               Heikkinen, Miss. Laina  female  26.0      0
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                             Allen, Mr. William Henry    male  35.0      0
..                                                 ...     ...   ...    ...
886                              Montvila, Rev. Juozas    male  27.0      0
887                       Graham, Miss. Margaret Edith  female  19.0      0
888           Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                              Behr, Mr. Karl Howell    male  26.0      0
890                                Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     ...               ...      ...   ...      ...
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

```
# extract dependent and independent features
```

```
x = data.drop('Survived', axis = 1)
y = data['Survived']
print(x)
print(y)
```

```
     PassengerId  Pclass                                               Name  \
0              1       3                            Braund, Mr. Owen Harris
1              2       1  Cumings, Mrs. John Bradley (Florence Briggs Th...
2              3       3                             Heikkinen, Miss. Laina
3              4       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)
4              5       3                           Allen, Mr. William Henry
..           ...     ...                                                ...
886          887       2                              Montvila, Rev. Juozas
887          888       1                       Graham, Miss. Margaret Edith
888          889       3           Johnston, Miss. Catherine Helen "Carrie"
889          890       1                              Behr, Mr. Karl Howell
890          891       3                                Dooley, Mr. Patrick

        Sex   Age  SibSp  Parch            Ticket     Fare Cabin Embarked
0      male  22.0      1      0         A/5 21171   7.2500   NaN        S
1    female  38.0      1      0          PC 17599  71.2833   C85        C
2    female  26.0      0      0  STON/O2. 3101282   7.9250   NaN        S
3    female  35.0      1      0            113803  53.1000  C123        S
4      male  35.0      0      0            373450   8.0500   NaN        S
..      ...   ...    ...    ...               ...      ...   ...      ...
886    male  27.0      0      0            211536  13.0000   NaN        S
887  female  19.0      0      0            112053  30.0000   B42        S
888  female   NaN      1      2        W./C. 6607  23.4500   NaN        S
889    male  26.0      0      0            111369  30.0000  C148        C
890    male  32.0      0      0            370376   7.7500   NaN        Q

[891 rows x 11 columns]
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
```

```
x.drop(['Name', 'Ticket', 'Cabin'],axis = 1, inplace = True)
print(x)
```

```
     PassengerId  Pclass     Sex   Age  SibSp  Parch     Fare Embarked
0              1       3    male  22.0      1      0   7.2500        S
1              2       1  female  38.0      1      0  71.2833        C
2              3       3  female  26.0      0      0   7.9250        S
3              4       1  female  35.0      1      0  53.1000        S
4              5       3    male  35.0      0      0   8.0500        S
..           ...     ...     ...   ...    ...    ...      ...      ...
886          887       2    male  27.0      0      0  13.0000        S
887          888       1  female  19.0      0      0  30.0000        S
888          889       3  female   NaN      1      2  23.4500        S
889          890       1    male  26.0      0      0  30.0000        C
890          891       3    male  32.0      0      0   7.7500        Q

[891 rows x 8 columns]
```

x['Age'] = x['Age'].fillna(x['Age'].mean())

print(x)

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	male	22.000000	1	0	7.2500	S
1	2	1	female	38.000000	1	0	71.2833	C
2	3	3	female	26.000000	0	0	7.9250	S
3	4	1	female	35.000000	1	0	53.1000	S
4	5	3	male	35.000000	0	0	8.0500	S
..	...	...	...	...	...	...	...	...
886	887	2	male	27.000000	0	0	13.0000	S
887	888	1	female	19.000000	0	0	30.0000	S
888	889	3	female	29.699118	1	2	23.4500	S
889	890	1	male	26.000000	0	0	30.0000	C
890	891	3	male	32.000000	0	0	7.7500	Q

[891 rows x 8 columns]

x['Embarked'] = x['Embarked'].fillna(x['Embarked'].mode()[0])

print(x)

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	male	22.000000	1	0	7.2500	S
1	2	1	female	38.000000	1	0	71.2833	C
2	3	3	female	26.000000	0	0	7.9250	S
3	4	1	female	35.000000	1	0	53.1000	S
4	5	3	male	35.000000	0	0	8.0500	S
..	...	...	...	...	...	...	...	...
886	887	2	male	27.000000	0	0	13.0000	S
887	888	1	female	19.000000	0	0	30.0000	S
888	889	3	female	29.699118	1	2	23.4500	S
889	890	1	male	26.000000	0	0	30.0000	C
890	891	3	male	32.000000	0	0	7.7500	Q

[891 rows x 8 columns]

x = pd.get_dummies(x, columns = ['Sex', 'Embarked'],prefix = ['Sex', 'Embarked'],drop_first = True)

print(x)

```
     PassengerId  Pclass        Age  SibSp  Parch     Fare  Sex_male  \
0              1       3  22.000000      1      0   7.2500      True
1              2       1  38.000000      1      0  71.2833     False
2              3       3  26.000000      0      0   7.9250     False
3              4       1  35.000000      1      0  53.1000     False
4              5       3  35.000000      0      0   8.0500      True
..           ...     ...        ...    ...    ...      ...       ...
886          887       2  27.000000      0      0  13.0000      True
887          888       1  19.000000      0      0  30.0000     False
888          889       3  29.699118      1      2  23.4500     False
889          890       1  26.000000      0      0  30.0000      True
890          891       3  32.000000      0      0   7.7500      True

     Embarked_Q  Embarked_S
0         False        True
1         False       False
2         False        True
3         False        True
4         False        True
..          ...         ...
886       False        True
887       False        True
888       False        True
889       False       False
890        True       False

[891 rows x 9 columns]
```

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)

print(x_train)

```
     PassengerId  Pclass        Age  SibSp  Parch     Fare  Sex_male  \
140          141       3  29.699118      0      2  15.2458     False
439          440       2  31.000000      0      0  10.5000      True
817          818       2  31.000000      1      1  37.0042      True
378          379       3  20.000000      0      0   4.0125      True
491          492       3  21.000000      0      0   7.2500      True
..           ...     ...        ...    ...    ...      ...       ...
835          836       1  39.000000      1      1  83.1583     False
192          193       3  19.000000      1      0   7.8542     False
629          630       3  29.699118      0      0   7.7333      True
559          560       3  36.000000      1      0  17.4000     False
684          685       2  60.000000      1      1  39.0000      True

     Embarked_Q  Embarked_S
140       False       False
439       False        True
817       False       False
378       False       False
491       False        True
..          ...         ...
835       False       False
192       False        True
629        True       False
559       False        True
684       False        True

[712 rows x 9 columns]
```

print(y_train)

```
140    0
439    0
817    0
378    0
491    0
      ..
835    1
192    1
629    0
559    1
684    0
Name: Survived, Length: 712, dtype: int64
```

from sklearn.preprocessing import StandardScaler

std_x = StandardScaler()

x_train = std_x.fit_transform(x_train)

x_test = std_x.transform(x_test)

print(x_train)

```
[[-1.16343003e+00  8.19250590e-01 -2.82437263e-03 ... -1.37207547e+00
  -3.14269681e-01 -1.63985340e+00]
 [-1.26383402e-02 -3.80968381e-01  9.66293694e-02 ...  7.28822884e-01
  -3.14269681e-01  6.09810609e-01]
 [ 1.44220868e+00 -3.80968381e-01  9.66293694e-02 ...  7.28822884e-01
  -3.14269681e-01 -1.63985340e+00]
 ...
 [ 7.18633972e-01  8.19250590e-01 -2.82437263e-03 ...  7.28822884e-01
   3.18198052e+00 -1.63985340e+00]
 [ 4.49217857e-01  8.19250590e-01  4.78884313e-01 ... -1.37207547e+00
  -3.14269681e-01  6.09810609e-01]
 [ 9.30318063e-01 -3.80968381e-01  2.31370804e+00 ...  7.28822884e-01
  -3.14269681e-01  6.09810609e-01]]
```

3. Write Python code to select features in machine learning using Python.

```
from pandas import read_csv
from numpy import set_printoptions
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
from matplotlib import pyplot

path=r'diabetes.csv'
names=['preg','plas','pres','skin','test','mass','peds','age','class']
dataframe=read_csv(path,names=names)
dataframe.head()
```

	preg	plas	pres	skin	test	mass	peds	age	class
0	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
1	6	148	72	35	0	33.6	0.627	50	1
2	1	85	66	29	0	26.6	0.351	31	0
3	8	183	64	0	0	23.3	0.672	32	1
4	1	89	66	23	94	28.1	0.167	21	0

```
array=dataframe.values
x=array[:,0:8]
y=array[:,8]
print(x)
print(y)
```

```
[['Pregnancies' 'Glucose' 'BloodPressure' ... 'BMI'
  'DiabetesPedigreeFunction' 'Age']
 ['6' '148' '72' ... '33.6' '0.627' '50']
 ['1' '85' '66' ... '26.6' '0.351' '31']
 ...
 ['5' '121' '72' ... '26.2' '0.245' '30']
 ['1' '126' '60' ... '30.1' '0.349' '47']
 ['1' '93' '70' ... '30.4' '0.315' '23']]
['Outcome' '1' '0' '1' '0' '1' '0' '1' '0' '1' '1' '0' '1' '0' '1' '1' '1'
 '1' '1' '0' '1' '0' '0' '1' '1' '1' '1' '1' '0' '0' '0' '0' '1' '0' '0'
 '0' '0' '0' '1' '1' '1' '0' '0' '0' '1' '0' '1' '0' '0' '1' '0' '0' '0'
 '0' '1' '0' '0' '1' '0' '0' '0' '0' '1' '0' '0' '0' '1' '0' '1' '0' '0'
 '1' '0' '1' '0' '0' '0' '0' '0' '1' '0' '0' '0' '0' '0' '1' '0' '0' '0'
 '1' '0' '0' '0' '0' '1' '0' '0' '0' '0' '0' '1' '1' '0' '0' '0' '0' '0'
 '0' '0' '0' '1' '1' '1' '0' '0' '1' '1' '1' '0' '0' '0' '1' '0' '0' '0'
 '1' '1' '0' '0' '1' '1' '1' '1' '1' '0' '0' '0' '0' '0' '0' '0' '0' '0'
 '0' '1' '0' '0' '0' '0' '0' '0' '0' '0' '1' '0' '1' '1' '0' '0' '0' '1'
 '0' '0' '0' '0' '1' '1' '0' '0' '0' '0' '1' '1' '0' '0' '0' '1' '0' '1'
 '0' '1' '0' '0' '0' '0' '0' '1' '1' '1' '1' '1' '0' '0' '1' '1' '0' '1'
 '0' '1' '1' '1' '0' '0' '0' '0' '0' '0' '1' '1' '0' '1' '0' '0' '0' '1'
 '1' '1' '1' '0' '1' '1' '1' '1' '0' '0' '0' '0' '0' '1' '0' '0' '1' '1'
 '0' '0' '0' '1' '1' '1' '1' '0' '0' '0' '1' '1' '0' '1' '0' '0' '0' '0'
 '0' '0' '0' '0' '1' '1' '0' '0' '0' '1' '0' '1' '0' '1' '0' '1' '0' '0'
 '0' '1' '1' '0' '0' '0' '0' '0' '1' '0' '0' '0' '1' '0' '0' '1' '1' '0'
 '0' '1' '0' '0' '0' '1' '1' '1' '0' '0' '1' '0' '1' '0' '1' '1' '0' '1'
 '0' '0' '1' '0' '1' '1' '0' '0' '1' '0' '1' '0' '0' '1' '0' '1' '0' '1'
```

x_train,x_test,y_train,y_test,=train_test_split(x,y,test_size=0.33,random_state=1)

fs= SelectKBest(score_func=f_classif,k='all')

fs.fit(x_train,y_train)

x_train_fs=fs.transform(x_train)

x_test_fs=fs.transform(x_test)

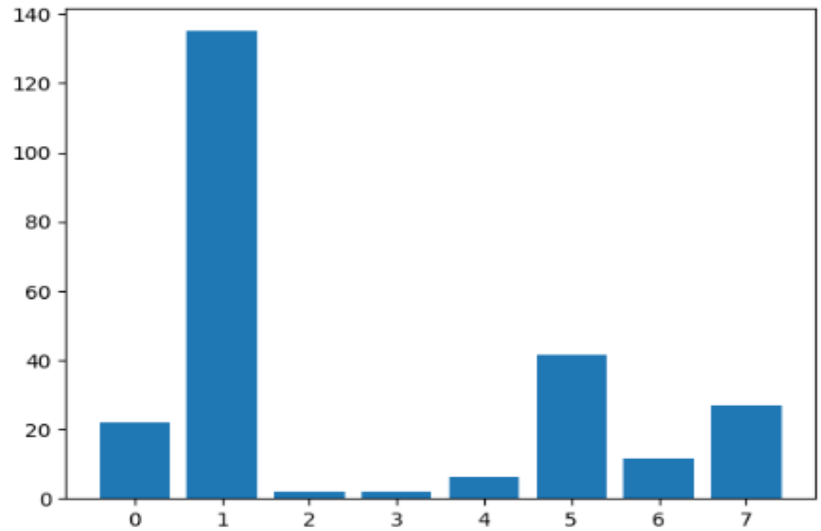
for i in range(len(fs.scores_)):

  print('feature %d:%f'%(i,fs.scores_[i]))

pyplot.bar([i for i in range(len(fs.scores_))],fs.scores_)

pyplot.show()

```
feature 0:21.952926
feature 1:134.995132
feature 2:1.868609
feature 3:2.141532
feature 4:6.303769
feature 5:41.691993
feature 6:11.759834
feature 7:26.831139
```



4. Write Python code to load the data from a CSV file and select the top 10 features using the chi-squared test. The selected features are to be printed on the console.

```
import numpy as np
 import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_selection import chi2


df=pd.read_csv('loandata.csv')
df.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	

```
df=df[['Gender','Married','Dependents','Education','Self_Employed','Credit_History','Property_Area','Loan_Status']]
df.head()
```

	Gender	Married	Dependents	Education	Self_Employed	Credit_History	Property_Area	Loan_Status
0	Male	No	0	Graduate	No	1.0	Urban	Y
1	Male	Yes	1	Graduate	No	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	1.0	Urban	Y
4	Male	No	0	Graduate	No	1.0	Urban	Y

```
#label encoding
from sklearn.preprocessing import LabelEncoder
for col in df.columns:
    le=LabelEncoder()
    df[col]=le.fit_transform(df[col])
df.head()
```

	Gender	Married	Dependents	Education	Self_Employed	Credit_History	Property_Area	Loan_Status
0	1	0	0	0	0	1	2	1
1	1	1	1	0	0	1	0	0
2	1	1	0	0	1	1	2	1
3	1	1	0	1	0	1	2	1
4	1	0	0	0	0	1	2	1

```
x=df.iloc[:,0:6]
y=df.iloc[:,-1]


f_score=chi2(x,y)
f_score
p_value=pd.Series(f_score[1], index=x.columns)
p_value.sort_values(ascending=False,inplace=True)
p_value
p_value.plot(kind="bar")
plt.xlabel("Features", fontsize=20)
plt.ylabel("p_values", fontsize=20)
plt.title("chi squared test base on p value")
plt.show()
```