```
# Importing Library
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
# Reading the training dataset in a dataframe using Pandas
df = pd.read_csv("train_u6lujuX_CVtuZ9i.csv")
# Reading the test dataset in a dataframe using Pandas
test = pd.read_csv("train_u6lujuX_CVtuZ9i.csv")
```

```
df.head(10)
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome
0	LP001002	Male	No	0	Graduate	No	5849
1	LP001003	Male	Yes	1	Graduate	No	4583
2	LP001005	Male	Yes	0	Graduate	Yes	3000
3	LP001006	Male	Yes	0	Not Graduate	No	2583
4	LP001008	Male	No	0	Graduate	No	6000
5	LP001011	Male	Yes	2	Graduate	Yes	5417
6	LP001013	Male	Yes	0	Not Graduate	No	2333
7	LP001014	Male	Yes	3+	Graduate	No	3036
8	LP001018	Male	Yes	2	Graduate	No	4006

```
#creating training dataset
df_length =len(df)
test_col = len(test.columns)
```

```
df.describe()
```

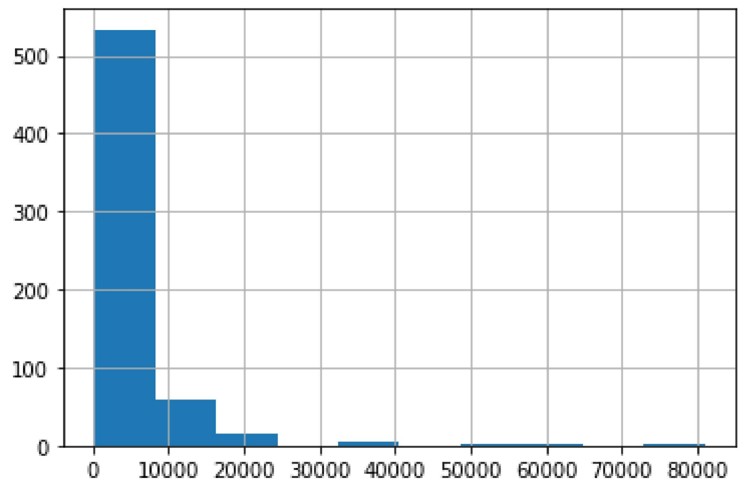
	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_Hist
count	614.000000	614.000000	592.000000	600.00000	564.000

```
df['Property_Area'].value_counts()
```

```
Semiurban    233
Urban        202
Rural        179
Name: Property_Area, dtype: int64
```

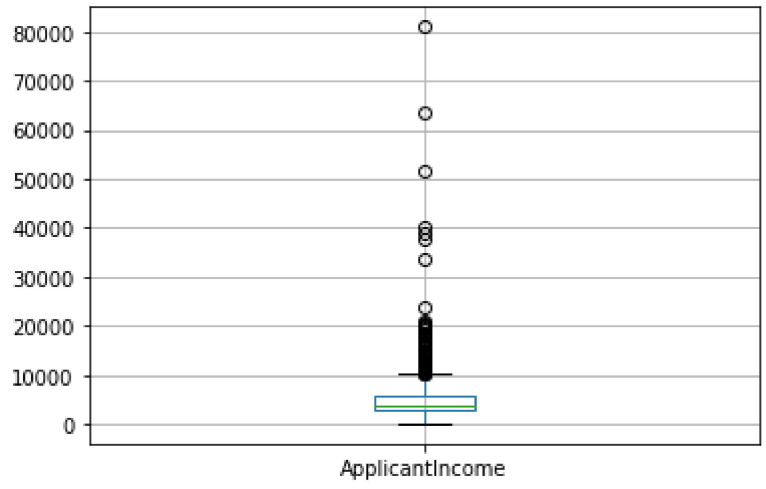
```
%matplotlib inline
df['ApplicantIncome'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb06379f1d0>
```



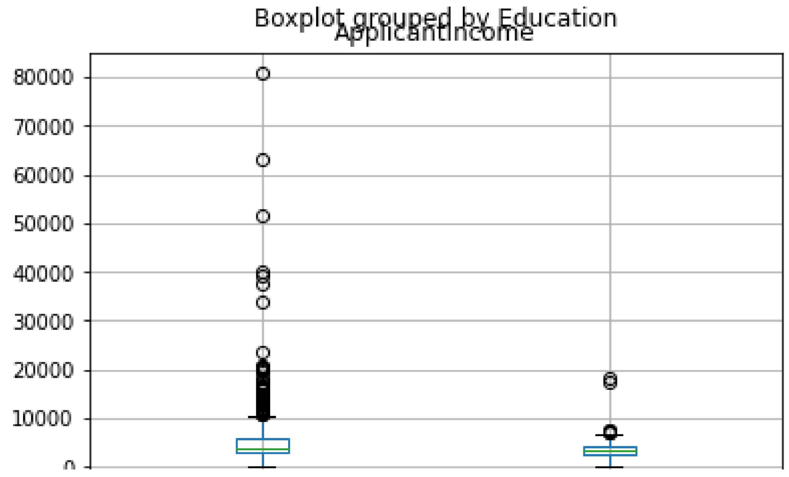
```
df.boxplot(column='ApplicantIncome')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb05b667f90>
```



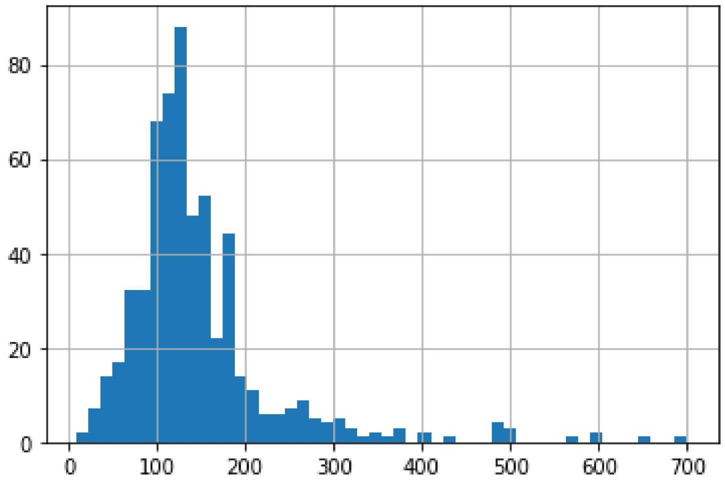
```
df.boxplot(column='ApplicantIncome', by = 'Education')
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83: VisibleDeprecationW
  return array(a, dtype, copy=False, order=order)
<matplotlib.axes._subplots.AxesSubplot at 0x7fb059d96850>
```



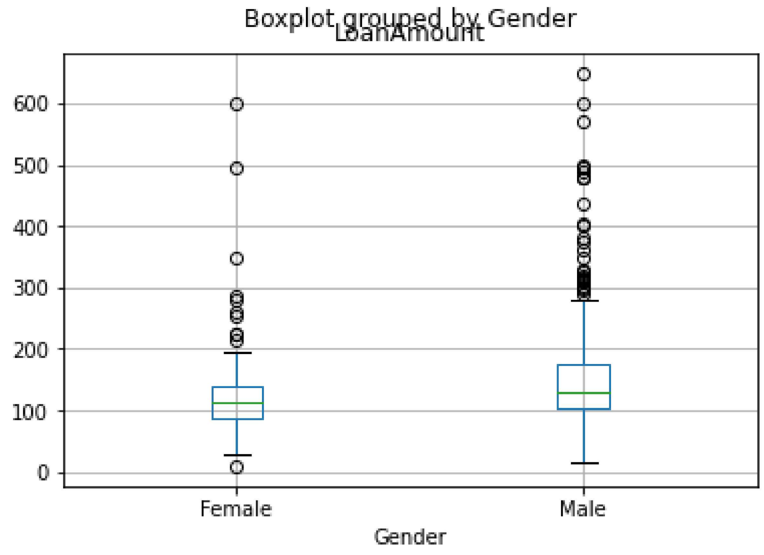
```
df['LoanAmount'].hist(bins=50)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb059d35750>
```



```
df.boxplot(column='LoanAmount', by = 'Gender')
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83: VisibleDeprecationW
  return array(a, dtype, copy=False, order=order)
<matplotlib.axes._subplots.AxesSubplot at 0x7fb059c3a110>
```



```
loan_approval = df['Loan_Status'].value_counts()['Y']
print(loan_approval)
```

```
422
```

```
pd.crosstab(df ['Credit_History'], df ['Loan_Status'], margins=True)
```

Loan_Status	N	Y	All
Credit_History			
0.0	82	7	89
1.0	97	378	475
All	179	385	564

```
def percentageConvert(ser):
    return ser/float(ser[-1])
# Loan approval rate for customers having Credit_History (1)
df=pd.crosstab(df ["Credit_History"], df ["Loan_Status"], margins=True).apply(percentageCo
loan_approval_with_Credit_1 = df['Y'][1]
print(loan_approval_with_Credit_1*100)
```

```
79.57894736842105
```

```
df['Y']
```

```
Credit_History
0.0    0.078652
1.0    0.795789
All    0.682624
Name: Y, dtype: float64
```

```
 # Perform log transformation of TotalIncome to make it closer to normal
df['LoanAmount_log'] = np.log(df['LoanAmount'])
# Looking at the distribtion of TotalIncome_log
df['LoanAmount_log'].hist(bins=20)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self,
key, method, tolerance)
   2897             try:
-> 2898                 return self._engine.get_loc(casted_key)
   2899             except KeyError as err:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'LoanAmount'
```

```
# Impute missing values for Gender
df['Gender'].fillna(df['Gender'].mode()[0],inplace=True)
# Impute missing values for Married
df['Married'].fillna(df['Married'].mode()[0],inplace=True)
# Impute missing values for Dependents
df['Dependents'].fillna(df['Dependents'].mode()[0],inplace=True)
# Impute missing values for Credit_History
df['Credit_History'].fillna(df['Credit_History'].mode()[0],inplace=True)
# Convert all non-numeric values to number
cat=['Gender','Married','Dependents','Education','Self_Employed','Credit_History','Propert
for var in cat:
    le = preprocessing.LabelEncoder()
    df[var]=le.fit_transform(df[var].astype('str'))
df.dtypes
```

```
    ---------------------------------------------------------------------------
    KeyError                                  Traceback (most recent call last)
    /usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self,
    key, method, tolerance)
       2897             try:
    -> 2898                 return self._engine.get_loc(casted_key)
       2899             except KeyError as err:
```

```
#Model Building
```

```
    pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()
```

```
df['Type']='Train'
test['Type']='Test'
fullData = pd.concat([df,test],axis=0, sort=True)
#Look at the available missing values in the dataset
fullData.isnull().sum()
```

```
    All                  614
    ApplicantIncome        3
    CoapplicantIncome      3
    Credit_History        53
    Dependents            18
    Education              3
    Gender                16
    LoanAmount            25
    Loan_Amount_Term      17
    Loan_ID                3
    Loan_Status            3
    Married                6
    N                    614
    Property_Area          3
    Self_Employed         35
    Type                   0
    Y                    614
    dtype: int64
```

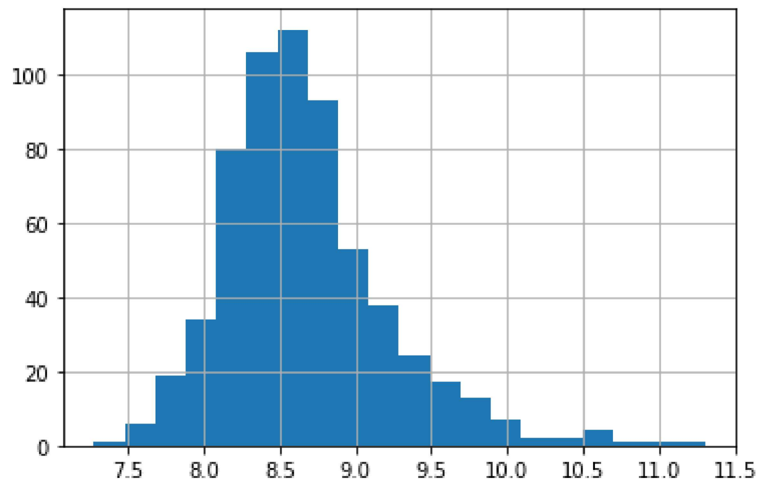
```
ID_col = ['Loan_ID']
target_col = ["Loan_Status"]
cat_cols = ['Credit_History','Dependents','Gender','Married','Education','Property_Area']
```

```
#Imputing Missing values with mean for continuous variable
fullData['LoanAmount'].fillna(fullData['LoanAmount'].mean(), inplace=True)
fullData['LoanAmount_log'].fillna(fullData['LoanAmount_log'].mean(), inplace=True)
fullData['Loan_Amount_Term'].fillna(fullData['Loan_Amount_Term'].mean(), inplace=True)
fullData['ApplicantIncome'].fillna(fullData['ApplicantIncome'].mean(), inplace=True)
fullData['CoapplicantIncome'].fillna(fullData['CoapplicantIncome'].mean(), inplace=True)
fullData['Gender'].fillna(fullData['Gender'].mode()[0], inplace=True)
fullData['Married'].fillna(fullData['Married'].mode()[0], inplace=True)
fullData['Dependents'].fillna(fullData['Dependents'].mode()[0], inplace=True)
fullData['Loan_Amount_Term'].fillna(fullData['Loan_Amount_Term'].mode()[0], inplace=True)
fullData['Credit_History'].fillna(fullData['Credit_History'].mode()[0], inplace=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self,
key, method, tolerance)
   2897             try:
-> 2898                 return self._engine.get_loc(casted_key)
   2899             except KeyError as err:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'LoanAmount_log'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
                          2 frames
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self,
key, method, tolerance)
   2898                 return self._engine.get_loc(casted_key)
   2899             except KeyError as err:
```

```
fullData['TotalIncome']=fullData['ApplicantIncome'] + fullData['CoapplicantIncome']
fullData['TotalIncome_log'] = np.log(fullData['TotalIncome'])
#Histogram for Total Income
fullData['TotalIncome_log'].hist(bins=20)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb0568da310>
```



```
for var in cat_cols:
    number = LabelEncoder()
    fullData[var] = number.fit_transform(fullData[var].astype('str'))
train_modified=fullData[fullData['Type']=='Train']
test_modified=fullData[fullData['Type']=='Test']
train_modified["Loan_Status"] = number.fit_transform(train_modified["Loan_Status"].astype(
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWarnin
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
```

```
from sklearn.linear_model import LogisticRegression
predictors_Logistic=['Credit_History','Education','Gender']
x_train = train_modified[list(predictors_Logistic)].values
y_train = train_modified["Loan_Status"].values
x_test=test_modified[list(predictors_Logistic)].values


model = LogisticRegression()
model.fit(x_train, y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-56-52d72ce63b87> in <module>()
      1 model = LogisticRegression()
----> 2 model.fit(x_train, y_train)

/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py in
fit(self, X, y, sample_weight)
   1556             raise ValueError("This solver needs samples of at least 2
classes"
   1557                              " in the data, but the data contains only one"
-> 1558                              " class: %r" % classes_[0])
   1559
   1560         if len(self.classes_) == 2:

ValueError: This solver needs samples of at least 2 classes in the data, but the
data contains only one class: 0
```