Importing Libraries and Dataset

Firstly we have to import libraries :

1. Pandas – To load the Dataframe
2. Matplotlib – To visualize the data features i.e. barplot
3. Seaborn – To see the correlation between features using heatmap

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

data = pd.read_csv("LoanApprovalPrediction.csv")
data.head(5)
```

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

Data Preprocessing and Visualization

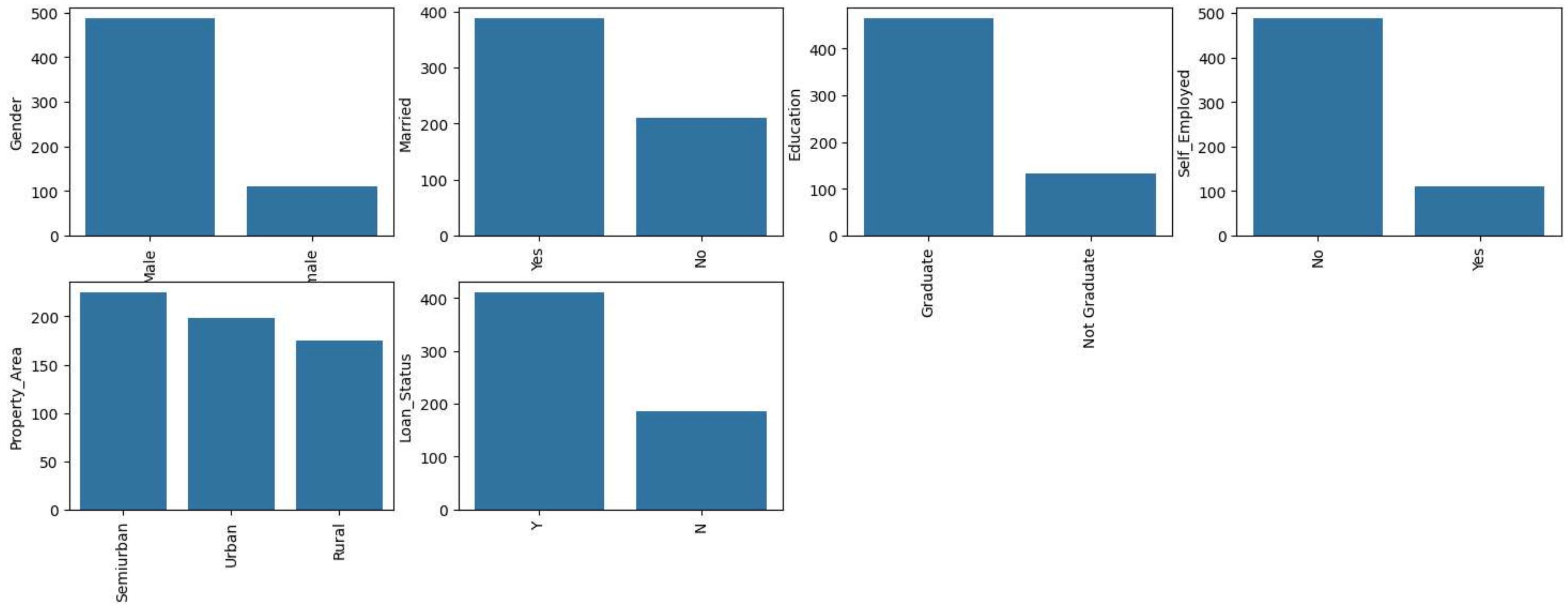
Get the number of columns of object datatype.

```
obj = (data.dtypes == 'object')
print("Categorical variables:",len(list(obj[obj].index)))
```

```
Categorical variables: 7
```

```
# As Loan_ID is completely unique and not correlated with any of the other column,
# So we will drop it using .drop() function.
# Dropping Loan_ID column
data.drop(['Loan_ID'],axis=1,inplace=True)
```

```
# Visualize all the unique values in columns using barplot.
# This will simply show which value is dominating as per our dataset.
obj = (data.dtypes == 'object')
object_cols = list(obj[obj].index)
plt.figure(figsize=(18,36))
index = 1

for col in object_cols:
  y = data[col].value_counts()
  plt.subplot(11,4,index)
  plt.xticks(rotation=90)
  sns.barplot(x=list(y.index), y=y)
  index +=1
```



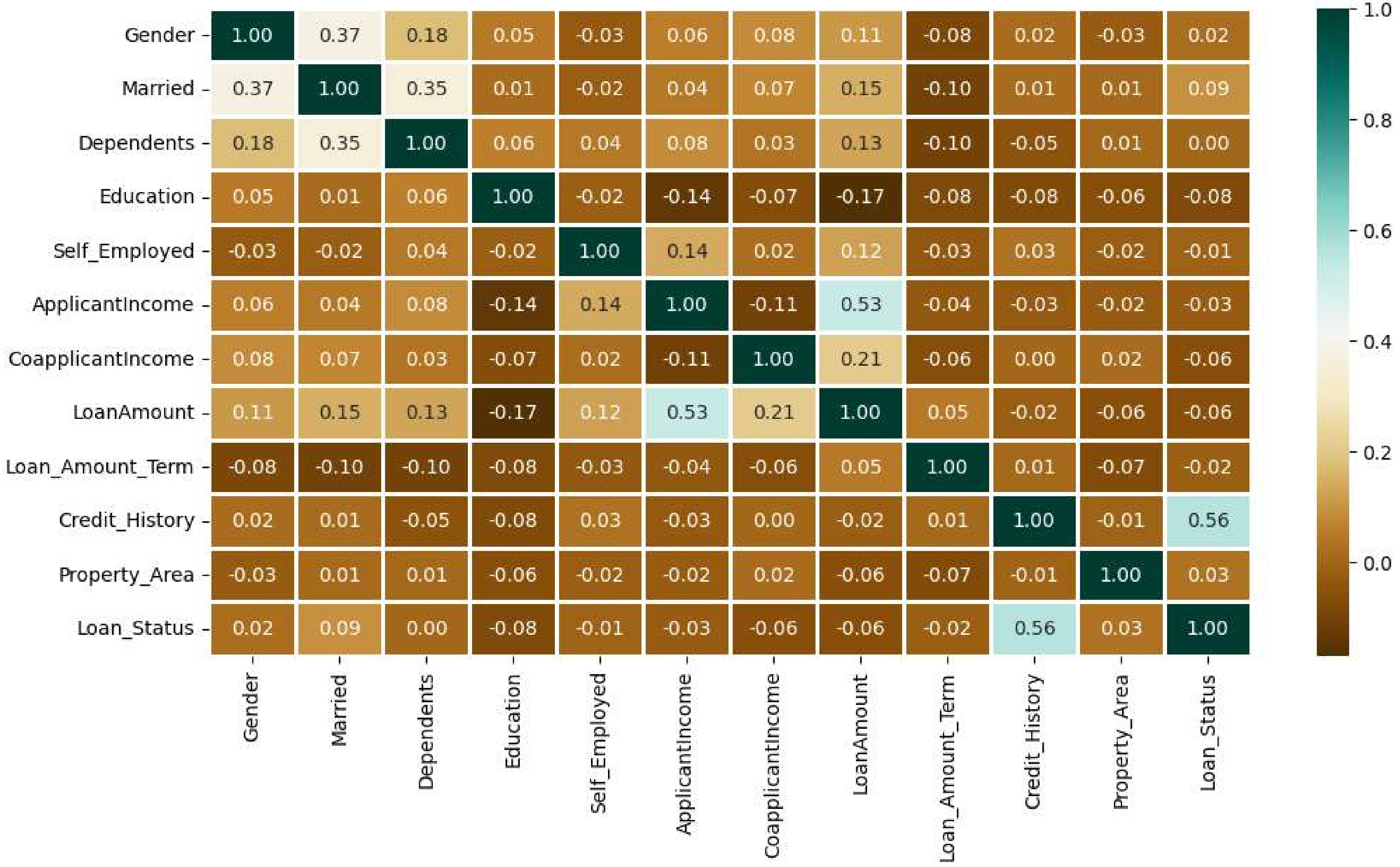
```
# As all the categorical values are binary so we can use Label Encoder for all
# such columns and the values will change into int datatype.

# Import label encoder
from sklearn import preprocessing

# label_encoder object knows how
# to understand word labels.
label_encoder = preprocessing.LabelEncoder()
obj = (data.dtypes == 'object')
for col in list(obj[obj].index):
  data[col] = label_encoder.fit_transform(data[col])
```

```
# Again check the object datatype columns. Let’s find out if there is still any left.

# To find the number of columns with
# datatype==object
obj = (data.dtypes == 'object')
print("Categorical variables:",len(list(obj[obj].index)))
```

```
Categorical variables: 0
```

```
plt.figure(figsize=(12,6))

sns.heatmap(data.corr(),cmap='BrBG',fmt='.2f',
            linewidths=2,annot=True)
```

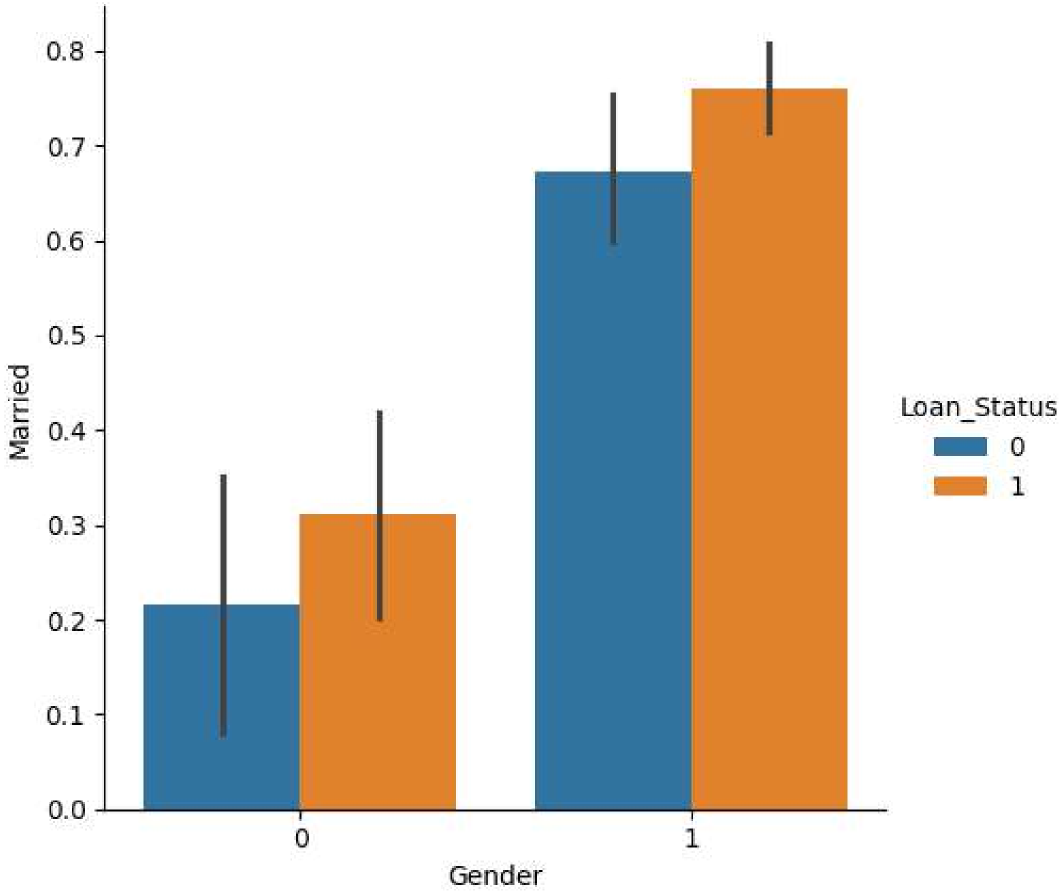
```
<Axes: >
```



The above heatmap is showing the correlation between Loan Amount and ApplicantIncome. It also shows that Credit_History has a high impact on Loan_Status.

```
# use Catplot to visualize the plot for the Gender, and Marital Status of the applicant.
sns.catplot(x="Gender", y="Married",
            hue="Loan_Status",
            kind="bar",
            data=data)
```

```
<seaborn.axisgrid.FacetGrid at 0x7866f85c2f80>
```



```
# find out if there is any missing values in the dataset using below code.
for col in data.columns:
  data[col] = data[col].fillna(data[col].mean())
data.isna().sum()
```

```
Gender              0
Married             0
Dependents          0
Education           0
Self_Employed       0
ApplicantIncome     0
CoapplicantIncome   0
LoanAmount          0
Loan_Amount_Term    0
Credit_History      0
Property_Area       0
Loan_Status         0
dtype: int64
```

Splitting Dataset

```
from sklearn.model_selection import train_test_split

X = data.drop(['Loan_Status'],axis=1)
Y = data['Loan_Status']
print(X.shape,Y.shape )

X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
                                                    test_size=0.4,
                                                    random_state=1)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

```
(598, 11) (598,)
((358, 11), (240, 11), (358,), (240,))
```

Model Training and Evaluation

As this is a classification problem so we will be using these models :

1. KNeighborsClassifiers

2. RandomForestClassifiers
3. Support Vector Classifiers (SVC)
4. Logistics Regression

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression

from sklearn import metrics

knn = KNeighborsClassifier(n_neighbors=3)
rfc = RandomForestClassifier(n_estimators = 7,
                            criterion = 'entropy',
                            random_state =7)
svc = SVC()
lc = LogisticRegression()

# making predictions on the training set
for clf in (rfc, knn, svc,lc):
    clf.fit(X_train, Y_train)
    Y_pred = clf.predict(X_train)
    print("Accuracy score of ",
        clf.__class__.__name__,
        "=",100*metrics.accuracy_score(Y_train,
                                        Y_pred))
```