Dealing with NAN values

7 ways to handle missing values in the dataset:

1.Deleting Rows with missing values

2.Impute missing values for continuous variable

3.mpute missing values for categorical variable

4.Other Imputation Methods

5.Using Algorithms that support missing values

6.Prediction of missing values

7.Imputation using Deep Learning Library — Datawig

In [2]:
```
import pandas as pd,numpy as np,os
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
os.chdir('D:\machine learning\Raw data')
```

In [10]:
```
os.listdir()
```

Out[10]:
```
['a SQL',
 'Advertising.csv',
 'automobile_data.sas7bdat',
 'Automobile_data.xlsx',
 'Automobile_data2.csv',
 'Automobile_data2.xlsx',
 'bank_data.sas7bdat',
 'bigmart_data.csv',
 'Book1.xlsx',
 'carsnew2.xlsx',
 'casnew.csv',
 'churn.csv',
 'churn.xlsx',
 'churn2.csv',
 'churn_data.pickle',
 'churn_data.xlsx',
 'chur_12.xlsx',
 'cleaned data',
 'concrete_data.csv',
 'Covid_data.xlsx',
 'CREDIT_DISCOVERY_FOR_DS.csv',
 'data.csv',
 'dubai_refreshments_final.sas7bdat',
 'employees.csv',
 'employee_detail.sas7bdat',
 'german.data.txt',
 'german_credit_data.csv',
 'Gold.xlsx',
 'House Price.csv',
 'House_Price_Scoring.csv',
 'machine learning',
 'MANJU.csv',
 'marks',
 'merging',
 'nortel.csv',
 'payroll2.csv',
 'Problem Statement.docx',
 'state gdp',
 'test.csv',
 'Titanic_data.csv',
 'train.csv',
 'user devise',
 'user_usage.xlsx']
```

In [12]:
```
df_credit=pd.read_csv('german_credit_data.csv')
```

In [13]:
```
df_credit.head()
```

Out[13]:

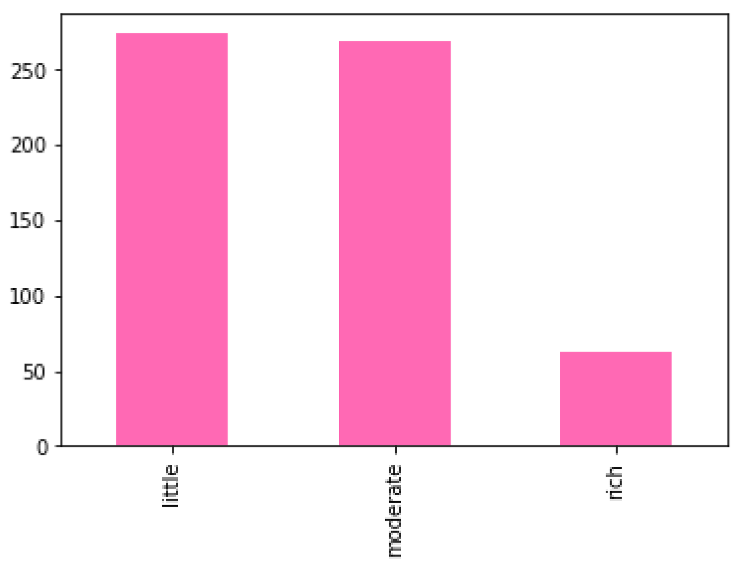
	Unnamed: 0	Age	Sex	Job	Housing	Saving accounts	Checking_account	Credit_amount	Duration	Purpose	Risk
0	0	67	male	2	own	NaN	little	1169	6	radio/TV	good
1	1	22	female	2	own	little	moderate	5951	48	radio/TV	bad
2	2	49	male	1	own	little	NaN	2096	12	education	good
3	3	45	male	2	free	little	little	7882	42	furniture/equipment	good
4	4	53	male	2	free	little	little	4870	24	car	bad

In [14]:
```
df_credit.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Unnamed: 0        1000 non-null   int64
 1   Age               1000 non-null   int64
 2   Sex               1000 non-null   object
 3   Job               1000 non-null   int64
 4   Housing           1000 non-null   object
 5   Saving accounts   817 non-null    object
 6   Checking_account  606 non-null    object
 7   Credit_amount     1000 non-null   int64
 8   Duration          1000 non-null   int64
 9   Purpose           1000 non-null   object
 10  Risk              1000 non-null   object
dtypes: int64(5), object(6)
memory usage: 86.1+ KB
```

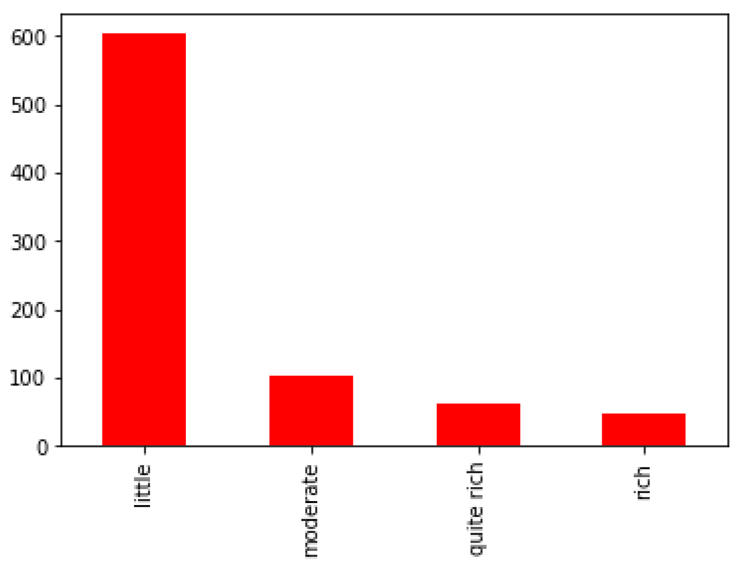
In [15]:
```
df_credit['Checking_account'].value_counts().plot.bar(color='hotpink')
```

Out[15]: <AxesSubplot:>



In [16]:
```
df_credit['Saving accounts'].value_counts().plot.bar(color='r')
```

Out[16]: <AxesSubplot:>



In [17]:
```
def impute_nan(df,variable):
    most_frequent_category=df[variable].mode()[0]
    df[variable].fillna(most_frequent_category,inplace=True)
```

In [18]:
```
   for feature in ['Checking_account','Saving accounts']:
        impute_nan(df_credit,feature)
```

In [19]:
```
df_credit.isnull().sum()
```

Out[19]:
```
Unnamed: 0          0
Age                 0
Sex                 0
Job                 0
Housing             0
Saving accounts     0
Checking_account    0
Credit_amount       0
Duration            0
Purpose             0
Risk                0
dtype: int64
```

In [ ]: