Handled Missing Values Part 3

July 20, 2023

0.0.1 Arbitrary Value Imputation

this technique was derived from kaggle competition It consists of replacing NAN by an arbitrary value

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```
df = pd.read_csv("C:\\Users\\ssart\\Downloads\\train.csv", usecols =␣
↪['Age','Fare','Survived'])
```

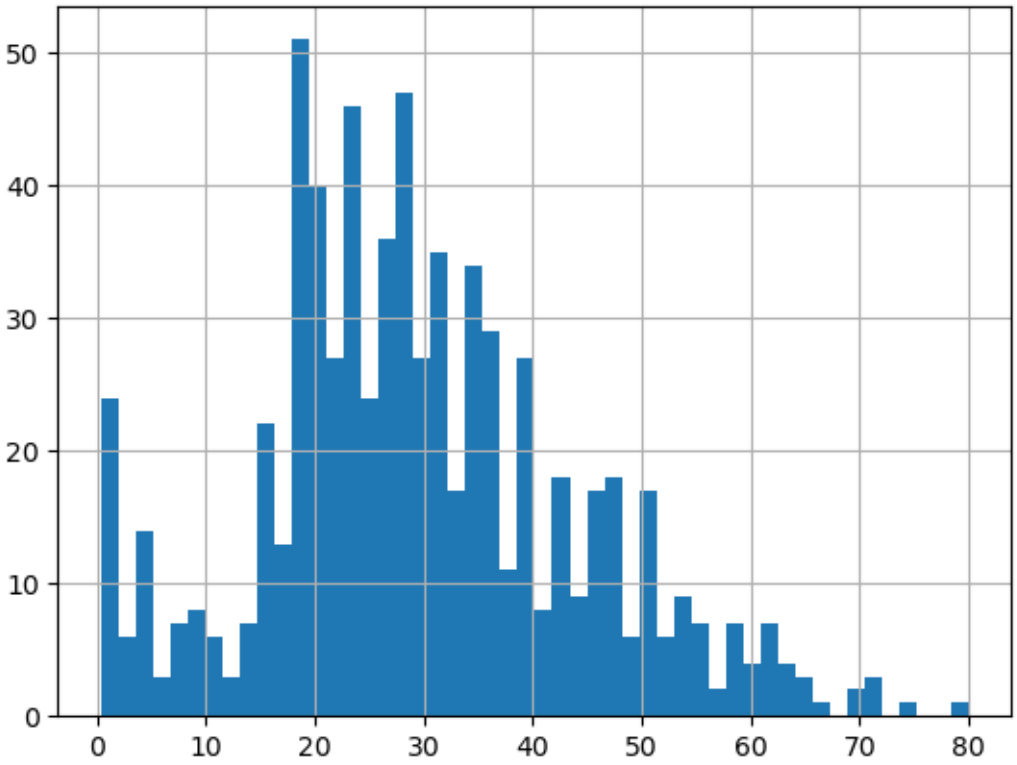
[3]:
```
df
```

[3]:
```
     Survived   Age     Fare
0           0  22.0   7.2500
1           1  38.0  71.2833
2           1  26.0   7.9250
3           1  35.0  53.1000
4           0  35.0   8.0500
..        …   …       …
886         0  27.0  13.0000
887         1  19.0  30.0000
888         0   NaN  23.4500
889         1  26.0  30.0000
890         0  32.0   7.7500

[891 rows x 3 columns]
```

[4]:
```
df['Age'].hist(bins =50)
```

[4]:
```
<AxesSubplot:>
```



[5]:
```
def impute_nan(df,variable):
    df[variable+"_zero"] = df[variable].fillna(0)
    df[variable+"_hundred"] = df[variable].fillna(100)
```

0.0.2 Advantages

- Easy to implement
- Captures the importance of missingess if there is one ### Disadvantages
- Distorts the original distribution of the variable
- If missingess is not important, it may mask the predictive power of the original variable by distorting its distribution
- Hard to decide which value to use ### How To Handle Categroical Missing Values ### Frequent Category Imputation

[6]:
```
import pandas as pd
```

[7]:
```
df = pd.read_csv("C:\\Users\\ssart\\Downloads\\House-Price.
↪csv",usecols=['BsmtQual','FireplaceQu','GarageType','SalePrice'])
```

[8]:
```
df.sample(8)
```

[8]:
	BsmtQual	FireplaceQu	GarageType	SalePrice
1272	TA	NaN	Attchd	137000
75	Gd	NaN	BuiltIn	91000
354	TA	Gd	Attchd	140000
1258	Gd	NaN	Attchd	190000
1350	TA	NaN	Detchd	200000
482	TA	Gd	Attchd	155000
1090	NaN	NaN	Detchd	92900
1009	TA	NaN	NaN	102000

[9]:
```
df.isnull().sum()
```

[9]:
```
BsmtQual        37
FireplaceQu    690
GarageType      81
SalePrice        0
dtype: int64
```

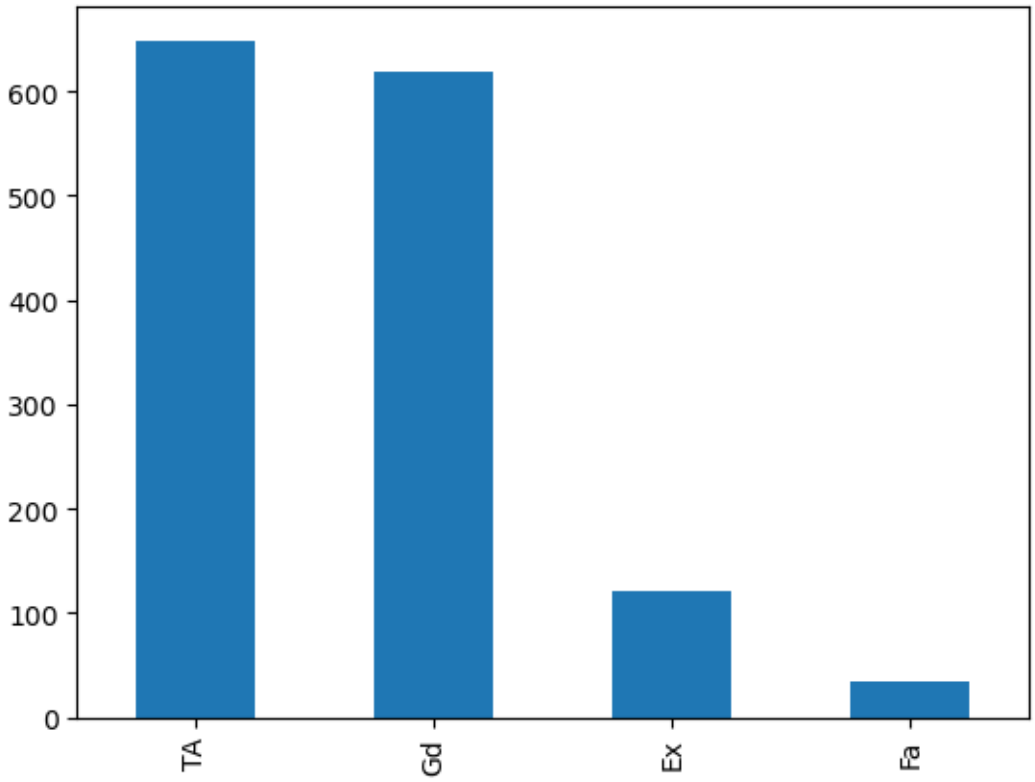
[10]:
```
df.isnull().mean().sort_values(ascending=True )
```

[10]:
```
SalePrice      0.000000
BsmtQual       0.025342
GarageType     0.055479
FireplaceQu    0.472603
dtype: float64
```

0.1 Compute the frequency with every feature

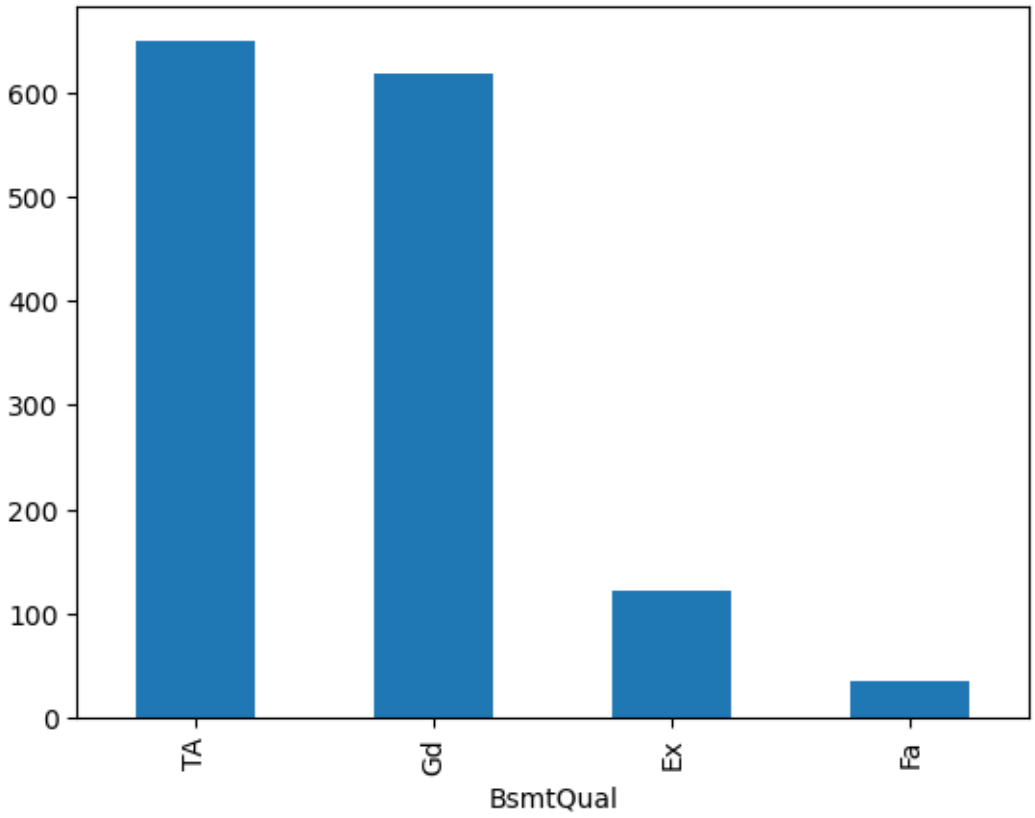
[11]:
```
df['BsmtQual'].value_counts().plot.bar()
```

[11]:
```
<AxesSubplot:>
```



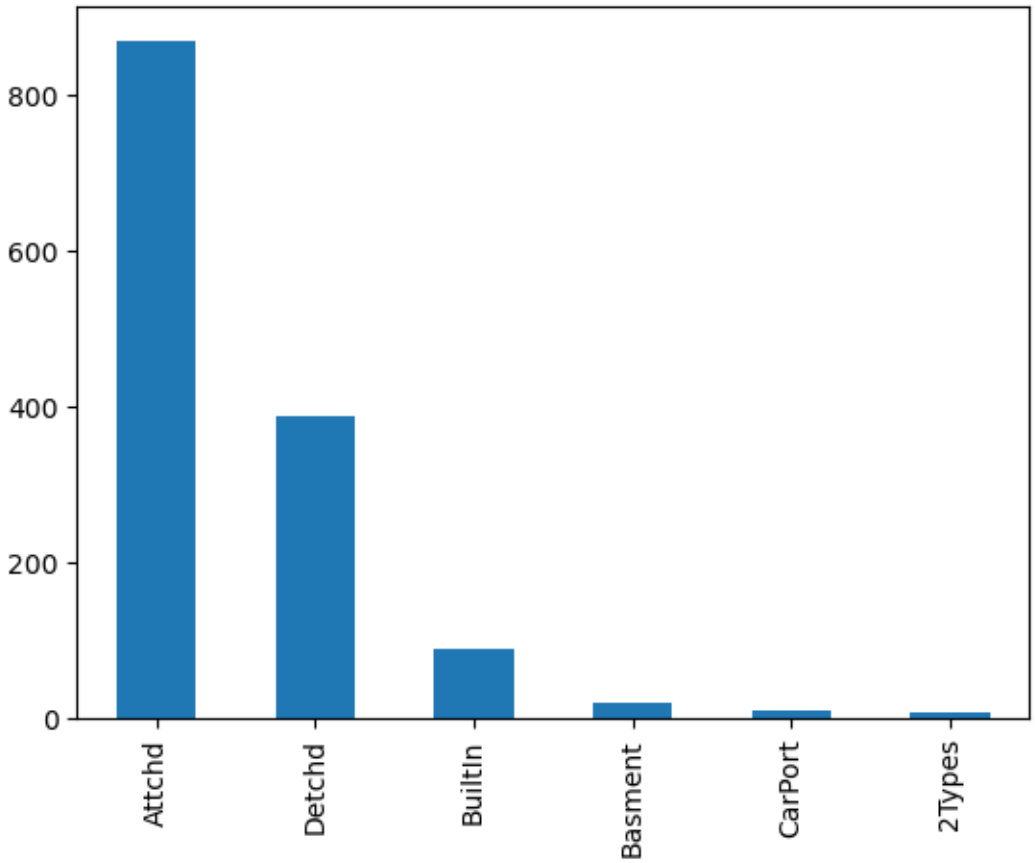
[12]:
```
df.groupby(['BsmtQual'])['BsmtQual'].count().sort_values(ascending=False).plot.
↪bar()
```

[12]: <AxesSubplot:xlabel='BsmtQual'>



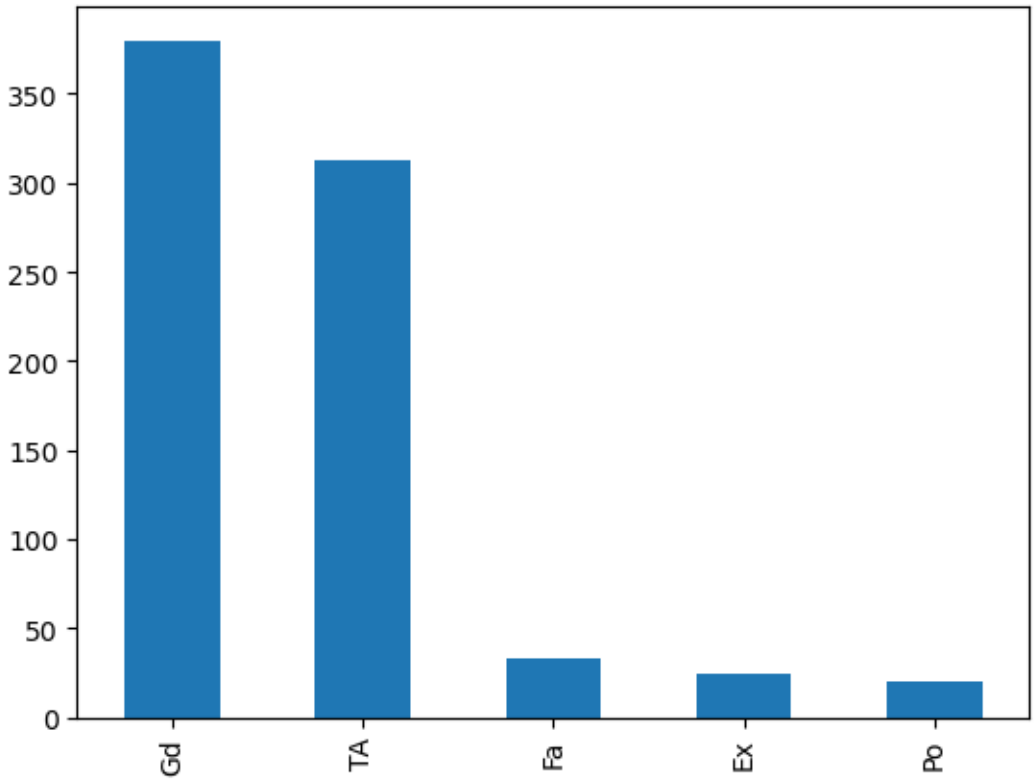
[13]:
```
df['GarageType'].value_counts().plot.bar()
```

[13]: <AxesSubplot:>



[14]:
```
df['FireplaceQu'].value_counts().plot.bar()
```

[14]: <AxesSubplot:>



[15]:
```
df['GarageType'].value_counts().index[0]
```

[15]: 'Attchd'

[16]:
```
df['GarageType'].mode()[0]
```

[16]: 'Attchd'

[17]:
```
def impute_nan(df,variable):
    most_frequent_varibale =df['GarageType'].value_counts().mode()[0]
    df[variable].fillna(most_frequent_varibale,inplace= True)
```

[18]:
```
for feature in ['BsmtQual','FireplaceQu','GarageType']:
    impute_nan(df,feature)
```

[19]:
```
df.sample(10)
```

[19]:

	BsmtQual	FireplaceQu	GarageType	SalePrice
1356	TA	6	Attchd	110000
358	TA	6	BuiltIn	130000
252	Gd	6	Attchd	173000
363	TA	6	Detchd	118000

```
838         Gd           6     Attchd     144000
1263        TA          Gd     Detchd     180500
888         TA          TA     Attchd     268000
440         Ex          Gd     Attchd     555000
8           TA          TA     Detchd     129900
1361        Ex          Gd     Attchd     260000
```

[20]:
```
df.isnull().mean()
```

[20]:
```
BsmtQual       0.0
FireplaceQu    0.0
GarageType     0.0
SalePrice      0.0
dtype: float64
```

0.1.1 Advantages

-Easy To implement -Fater way to implement ### Disadvantages -Since we are using the more frequent labels, it may use them in an over respresented way, if there are many nan's -It distorts the relation of the most frequent label ## Adding a variable to capture NAN

[21]:
```
df=pd.read_csv("C:\\Users\\ssart\\Downloads\\House-Price.
  ↪csv",usecols=['BsmtQual','FireplaceQu','GarageType','SalePrice'])
df.head()
```

[21]:
```
  BsmtQual FireplaceQu GarageType  SalePrice
0       Gd         NaN     Attchd     208500
1       Gd          TA     Attchd     181500
2       Gd          TA     Attchd     223500
3       TA          Gd     Detchd     140000
4       Gd          TA     Attchd     250000
```

[22]:
```
df['BsmtQual_var'] = np.where(df['BsmtQual'].isnull(),1,0)
```

[23]:
```
df.head()
```

[23]:
```
  BsmtQual FireplaceQu GarageType  SalePrice  BsmtQual_var
0       Gd         NaN     Attchd     208500             0
1       Gd          TA     Attchd     181500             0
2       Gd          TA     Attchd     223500             0
3       TA          Gd     Detchd     140000             0
4       Gd          TA     Attchd     250000             0
```

[24]:
```
feature=df['BsmtQual'].mode()[0]
```

[25]:
```
df['BsmtQual'].fillna(feature, inplace =True)
```

[26]:
```
df.head()
```

[26]:

	BsmtQual	FireplaceQu	GarageType	SalePrice	BsmtQual_var
0	Gd	NaN	Attchd	208500	0
1	Gd	TA	Attchd	181500	0
2	Gd	TA	Attchd	223500	0
3	TA	Gd	Detchd	140000	0
4	Gd	TA	Attchd	250000	0

[27]:
```
df['FireplaceQu_var'] = np.where(df['FireplaceQu'].isnull(),1,0)
feature=df['FireplaceQu'].mode()[0]
```

[28]:
```
df['FireplaceQu'].fillna(feature,inplace = True)
```

[29]:
```
df.head()
```

[29]:

	BsmtQual	FireplaceQu	GarageType	SalePrice	BsmtQual_var	FireplaceQu_var
0	Gd	Gd	Attchd	208500	0	1
1	Gd	TA	Attchd	181500	0	0
2	Gd	TA	Attchd	223500	0	0
3	TA	Gd	Detchd	140000	0	0
4	Gd	TA	Attchd	250000	0	0

[30]:
```
df.isnull().mean()
```

[30]:
```
BsmtQual           0.000000
FireplaceQu        0.000000
GarageType         0.055479
SalePrice          0.000000
BsmtQual_var       0.000000
FireplaceQu_var    0.000000
dtype: float64
```

0.1.2 Suppose if you have more frequent categories, we just replace NAN with a new category

[31]:
```
df = pd.read_csv("C:\\Users\\ssart\\Downloads\\House-Price.
↪csv",usecols=['BsmtQual','FireplaceQu','GarageType','SalePrice'])
df.head()
```

[31]:

	BsmtQual	FireplaceQu	GarageType	SalePrice
0	Gd	NaN	Attchd	208500
1	Gd	TA	Attchd	181500
2	Gd	TA	Attchd	223500
3	TA	Gd	Detchd	140000
4	Gd	TA	Attchd	250000

[32]:
```
def impute_nan(df,variable):
    df[variable+"newvar"] = np.where(df[variable].
↪isnull(),"missing",df[variable])
```

[33]:
```
for feature in ['BsmtQual','FireplaceQu','GarageType']:
    impute_nan(df,feature)
```

[34]:
```
df.head()
```

[34]:
```
  BsmtQual FireplaceQu GarageType  SalePrice BsmtQualnewvar FireplaceQunewvar  \
0       Gd          NaN     Attchd     208500             Gd            missing
1       Gd           TA     Attchd     181500             Gd                 TA
2       Gd           TA     Attchd     223500             Gd                 TA
3       TA           Gd     Detchd     140000             TA                 Gd
4       Gd           TA     Attchd     250000             Gd                 TA

  GarageTypenewvar
0           Attchd
1           Attchd
2           Attchd
3           Detchd
4           Attchd
```

[35]:
```
df=df.drop(['BsmtQual','FireplaceQu','GarageType'],axis = 1)
```

[36]:
```
df
```

[36]:
```
      SalePrice BsmtQualnewvar FireplaceQunewvar GarageTypenewvar
0        208500             Gd            missing           Attchd
1        181500             Gd                 TA           Attchd
2        223500             Gd                 TA           Attchd
3        140000             TA                 Gd           Detchd
4        250000             Gd                 TA           Attchd
…           …              …                  …              …
1455     175000             Gd                 TA           Attchd
1456     210000             Gd                 TA           Attchd
1457     266500             TA                 Gd           Attchd
1458     142125             TA            missing           Attchd
1459     147500             TA            missing           Attchd

[1460 rows x 4 columns]
```

[ ]:
```

```