(2) Prepare a data quality plan for the cleaned CSV file

data quality plan:

Feature	Data Quality Issue	Handling Strategy
RiskPerformance	none	Do nothing
ExternalRiskEstimate	Missing value(4 rows)	Imputation
MSinceOldestTradeOpen	Outliers(high)&MissingValues(34rows)	Imputation
MSinceMostRecentTradeOpen	Outliers (high)	Bring closer to bounds
AverageMInFile	Outliers (high)	Bring closer to bounds
NumSatisfactoryTrades	Outliers (high)	Bring closer to bounds
NumTrades60Ever2DerogPubRec	none	Do nothing
NumTrades90Ever2DerogPubRec	No new info	Remove column
PercentTradesNeverDelq	Outliers (low)	Do nothing
MSinceMostRecentDelq	none	Do nothing
MaxDelq2PublicRecLast12M	none	Do nothing
MaxDelqEver	none	Do nothing
NumTotalTrades	Outliers (high)	Bring closer to bounds
NumTradesOpeninLast12M	Outliers (high)	Bring closer to bounds
PercentInstallTrades	Outliers (high)	Bring closer to bounds
MSinceMostRecentInqexcl7days	none	Do nothing
NumInqLast6M	none	Do nothing
NumInqLast6Mexcl7days	No new info	Remove column
NetFractionRevolvingBurden	MissingValues(18rows)	Imputation
NetFractionInstallBurden	Missing Value(34.5%)	Remove Column
NumRevolvingTradesWBalance	Outliers(high)&MissingValues(17rows)	Imputation
NumInstallTradesWBalance	none	Do nothing
NumBank2NatlTradesWHighUtilization	Outliers(high)&MissingValues(54rows)	Imputation
PercentTradesWBalance	Missing value(5 rows)	Imputation

Apply your solutions

In [88]:

```
#Delete features
try:
    mySample_cleaned1 = mySample_cleaned1.drop(['NumTrades90Ever2DerogPubRec'
], axis=1)
    print(mySample_cleaned1.shape)
except:
    print("NumTrades90Ever2DerogPubRec already deleted")
    print(mySample_cleaned1.shape)
```

```
(944, 23)
```

In [89]:

```
#Delete features
try:
    mySample_cleaned1 = mySample_cleaned1.drop(['NumInqLast6Mexcl7days'], axis
=1)
    print(mySample_cleaned1.shape)
except:
    print("NumInqLast6Mexcl7days already deleted")
    print(mySample_cleaned1.shape)
```

```
(944, 22)
```

In [90]:

```
try:
    mySample_cleaned1 = mySample_cleaned1.drop(['NetFractionInstallBurden'], a
xis=1)
    print(mySample_cleaned1.shape)
except:
    print("NetFractionInstallBurden already deleted")
    print(mySample_cleaned1.shape)
```

```
(944, 21)
```

In [91]:

```
#Setting upper limit for outliers for MSinceOldestTradeOpen
UpperBound = 450 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['MSinceOldestTradeOpen'] > UpperBound,
'MSinceOldestTradeOpen'] = UpperBound
#mySample_cleaned1['MSinceOldestTradeOpen'].plot(kind='box')
```

In [92]:

```
#Setting upper limit for outliers for MSinceMostRecentTradeOpen
UpperBound = 25 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['MSinceMostRecentTradeOpen'] > UpperBo
und, 'MSinceMostRecentTradeOpen'] = UpperBound
#mySample_cleaned1['MSinceMostRecentTradeOpen'].plot(kind='box')
```

In [93]:

```
#Setting upper limit for outliers for AverageMInFile
UpperBound = 150 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['AverageMInFile'] > UpperBound, 'Avera
geMInFile'] = UpperBound
#mySample_cleaned1['AverageMInFile'].plot(kind='box')
```

In [94]:
```
#Setting upper limit for outliers for NumSatisfactoryTrades
UpperBound = 53 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['NumSatisfactoryTrades'] > UpperBound,
'NumSatisfactoryTrades'] = UpperBound
#mySample_cleaned1['NumSatisfactoryTrades'].plot(kind='box')
```

In [95]:
```
#Setting upper limit for outliers for NumTotalTrades
UpperBound = 55 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['NumTotalTrades'] > UpperBound, 'NumTo
talTrades'] = UpperBound
#mySample_cleaned1['NumTotalTrades'].plot(kind='box')
```

In [96]:
```
#Setting upper limit for outliers for NumTradesOpeninLast12M
UpperBound = 6 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['NumTradesOpeninLast12M'] > UpperBound
, 'NumTradesOpeninLast12M'] = UpperBound
#mySample_cleaned1['NumTradesOpeninLast12M'].plot(kind='box')
```

In [97]:
```
#Setting upper limit for outliers for PercentInstallTrades
UpperBound = 80 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['PercentInstallTrades'] > UpperBound,
'PercentInstallTrades'] = UpperBound
#mySample_cleaned1['PercentInstallTrades'].plot(kind='box')
```

In [98]:
```
#Setting upper limit for outliers for NumRevolvingTradesWBalance
UpperBound = 10 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['NumRevolvingTradesWBalance'] > UpperB
ound, 'NumRevolvingTradesWBalance'] = UpperBound
#mySample_cleaned1['NumRevolvingTradesWBalance'].plot(kind='box')
```

In [99]:
```
#Setting upper limit for outliers for NumBank2NatlTradesWHighUtilization
UpperBound = 5 #Upper limit from boxplot
mySample_cleaned1.loc[mySample_cleaned1['NumBank2NatlTradesWHighUtilization']
> UpperBound, 'NumBank2NatlTradesWHighUtilization'] = UpperBound
#mySample_cleaned1['NumBank2NatlTradesWHighUtilization'].plot(kind='box')
```

In [100]:
```
#Replace -9 and -8 with average values for ExternalRiskEstimate
mean = mySample_cleaned1.ExternalRiskEstimate.mean()
if len(mySample[mySample['ExternalRiskEstimate'] == -8]) > 0:
    mySample_cleaned1.ExternalRiskEstimate.replace(-8, mean, inplace=True)
if len(mySample[mySample['ExternalRiskEstimate'] == -9]) > 0:
    mySample_cleaned1.ExternalRiskEstimate.replace(-9, mean, inplace=True)
```

In [101]:
```
#Replace -9 and -8 with average values for MSinceOldestTradeOpen
mean = mySample_cleaned1.MSinceOldestTradeOpen.mean()
if len(mySample[mySample['MSinceOldestTradeOpen'] == -8]) > 0:
    mySample_cleaned1.MSinceOldestTradeOpen.replace(-8, mean, inplace=True)
if len(mySample[mySample['MSinceOldestTradeOpen'] == -9]) > 0:
    mySample_cleaned1.MSinceOldestTradeOpen.replace(-9, mean, inplace=True)
```

In [102]:

```
#Replace -9 and -8 with average values for NetFractionRevolvingBurden
mean = mySample_cleaned1.NetFractionRevolvingBurden.mean()
if len(mySample[mySample['NetFractionRevolvingBurden'] == -8]) > 0:
    mySample_cleaned1.NetFractionRevolvingBurden.replace(-8, mean, inplace=True)
if len(mySample[mySample['NetFractionRevolvingBurden'] == -9]) > 0:
    mySample_cleaned1.NetFractionRevolvingBurden.replace(-9, mean, inplace=True)
```

In [103]:

```
#Replace -9 and -8 with average values for NumRevolvingTradesWBalance
mean = mySample_cleaned1.NumRevolvingTradesWBalance.mean()
if len(mySample[mySample['NumRevolvingTradesWBalance'] == -8]) > 0:
    mySample_cleaned1.NumRevolvingTradesWBalance.replace(-8, mean, inplace=True)
if len(mySample[mySample['NumRevolvingTradesWBalance'] == -9]) > 0:
    mySample_cleaned1.NumRevolvingTradesWBalance.replace(-9, mean, inplace=True)
```

In [104]:

```
#Replace -9 and -8 with average values for NumBank2NatlTradesWHighUtilization
mean = mySample_cleaned1.NumBank2NatlTradesWHighUtilization.mean()
if len(mySample[mySample['NumBank2NatlTradesWHighUtilization'] == -8]) > 0:
    mySample_cleaned1.NumBank2NatlTradesWHighUtilization.replace(-8, mean, inplace=True)
if len(mySample[mySample['NumBank2NatlTradesWHighUtilization'] == -9]) > 0:
    mySample_cleaned1.NumBank2NatlTradesWHighUtilization.replace(-9, mean, inplace=True)
```

In [105]:

```
#Replace -9 and -8 with average values for PercentTradesWBalance
mean = mySample_cleaned1.PercentTradesWBalance.mean()
if len(mySample[mySample['PercentTradesWBalance'] == -8]) > 0:
    mySample_cleaned1.PercentTradesWBalance.replace(-8, mean, inplace=True)
if len(mySample[mySample['PercentTradesWBalance'] == -9]) > 0:
    mySample_cleaned1.PercentTradesWBalance.replace(-9, mean, inplace=True)
```

Cleaned data results:

In [106]:
```
mySample_cleaned1.select_dtypes(['int64', 'float64']).describe().T
```

Out[106]:

	count	mean	std	min	25%	
ExternalRiskEstimate	944.0	72.080459	9.713999	43.0	65.0	72.000
MSinceOldestTradeOpen	944.0	201.985490	95.364868	2.0	138.0	189.00
MSinceMostRecentTradeOpen	944.0	8.210805	7.053578	0.0	3.0	6.0000
AverageMInFile	944.0	78.352754	31.469245	6.0	58.0	76.000
NumSatisfactoryTrades	944.0	21.515890	11.752203	1.0	13.0	20.000
PercentTradesNeverDelq	944.0	92.345339	11.425976	33.0	89.0	97.000
MaxDelq2PublicRecLast12M	944.0	5.733051	1.696709	0.0	5.0	6.0000
MaxDelqEver	944.0	6.358051	1.893500	2.0	6.0	6.0000
NumTotalTrades	944.0	22.880297	12.812353	0.0	14.0	21.000
NumTradesOpeninLast12M	944.0	1.919492	1.709521	0.0	1.0	2.0000
PercentInstallTrades	944.0	33.269068	17.228723	0.0	21.0	32.000
NetFractionRevolvingBurden	944.0	34.375516	28.370896	0.0	8.0	31.000
NumRevolvingTradesWBalance	944.0	3.962678	2.532658	0.0	2.0	3.7510
NumBank2NatlTradesWHighUtilization	944.0	1.025426	1.262881	0.0	0.0	1.0000
PercentTradesWBalance	944.0	66.088443	21.761848	0.0	50.0	67.000

Cleaned categorical features:

In [107]:
```
mySample_cleaned1.select_dtypes(['category']).describe().T
```

Out[107]:

	count	unique	top	freq
NumTrades60Ever2DerogPubRec	307	4	Never	183
MSinceMostRecentDelq	944	7	Unknown	460
MSinceMostRecentInqexcl7days	944	6	Never	431
NumInqLast6M	944	6	unknown	367
NumInstallTradesWBalance	944	7	1	277

Save cleaned data to new csv