Imputation Document

Step1: Import the required packages

```
#Importing Packages
import os
import pandas as pd
import numpy as np
```

Step 2 : Import the dataset

```
#Importing Dataset
dt=pd.read_excel('LUAD_Clinical_1.xlsx',index_col=0)
```

Step 3: Print the count of categorical and numerical columns

```
numerical_feats = dt.dtypes[dt.dtypes != "object"].index
print("Number of Numerical features: ",
len(numerical_feats))

categorical_feats = dt.dtypes[dt.dtypes ==
"object"].index
print("Number of Categorical features: ",
len(categorical_feats))
```

Step 4: Extract the categorical columns

```
print(dt[categorical_feats].columns)
```

Step 5: One Hot Encoding Categorical Features into Numerical Features

```
dt_copy = pd.get_dummies(dt, columns=['American Joint
Committee on Cancer Metastasis Stage Code',
       'Neoplasm Disease Lymph Node Stage American Joint
Committee on Cancer Code',
       'Neoplasm Disease Stage American Joint Committee
on Cancer Code3',
```

```
       'American Joint Committee on Cancer Publication
Version Type',
       'American Joint Committee on Cancer Tumor Stage
Code', 'Cancer Type',
       'Cancer Type Detailed', 'Disease Free Status',
'Form completion date',
       'Neoplasm Histologic Type Name',
       'Neoadjuvant Therapy Type Administered Prior To
Resection Text',
       'Prior Cancer Diagnosis Occurence', 'ICD-10
Classification',
       'International Classification of Diseases for
Oncology, Third Edition ICD-O-3 Histology Code',
       'International Classification of Diseases for
Oncology, Third Edition ICD-O-3 Site Code',
       'Informed consent verified', 'Is FFPE', 'Oncotree
Code',
       'Overall Survival Status', 'Primary Tumor Site',
       'Tissue Prospective Collection Indicator', 'Race
Category',
       'Tissue Retrospective Collection Indicator',
'Sample Type', 'Sex',
       'Tissue Source Site', 'Person Neoplasm Status',
'Vial number',
       'Patient\'s Vital Status'])
```

Step 6: Checking if there are any null values in the dataset

```
dt.isnull().values.any()
```

Step 7: Dropping null values where the all the records are null in a columns.

```
dt = dt.dropna(axis='columns',how='all')
```

Step 8: Imputation of both numerical and categorical columns

If there are missing values in any columns, the categorical values are imputed with most frequent value in column and numerical values are imputed with mean of the column.

```
import pandas as pd
import numpy as np

from sklearn.base import TransformerMixin

class DataFrameImputer(TransformerMixin):

    def __init__(self):
        """Impute missing values.

        Columns of dtype object are imputed with the most
frequent value
        in column.

        Columns of other types are imputed with mean of
column.

        """
    def fit(self, X, y=None):

        self.fill =
pd.Series([X[c].value_counts().index[0]
            if X[c].dtype == np.dtype('O') else
X[c].mean() for c in X],
            index=X.columns)

        return self

    def transform(self, X, y=None):
        return X.fillna(self.fill)
xt= DataFrameImputer().fit_transform(dt)
```

Step 9 : Check again if there are any null values.

```
xt.isnull().values.any()
```

Step 10: Export it to an excel file.

```
xt.to_excel('LUSC_clinical_final.xlsx')
```