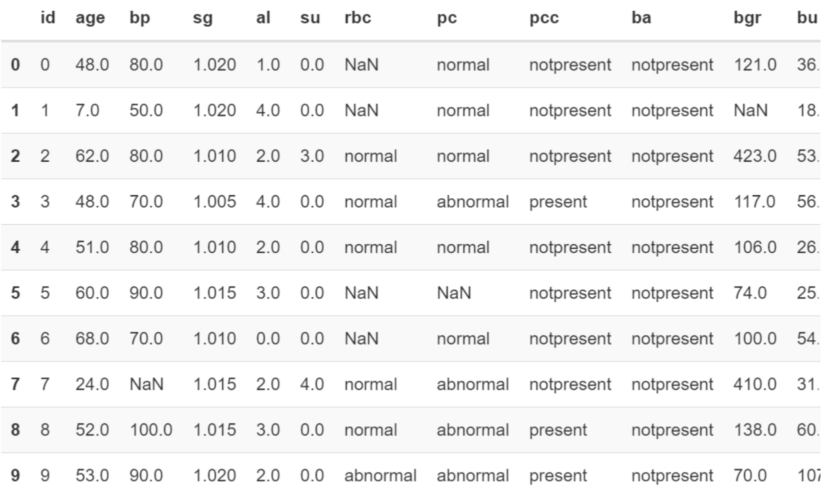
Data Collection and Preprocessing Phase

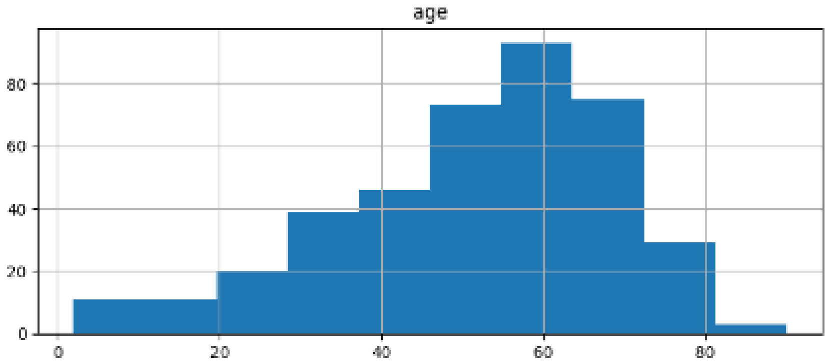
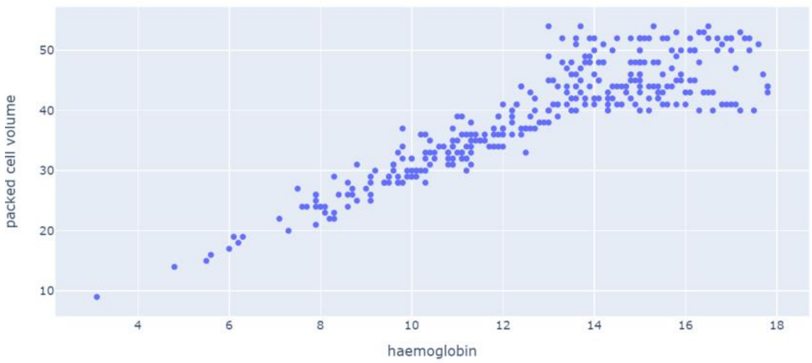
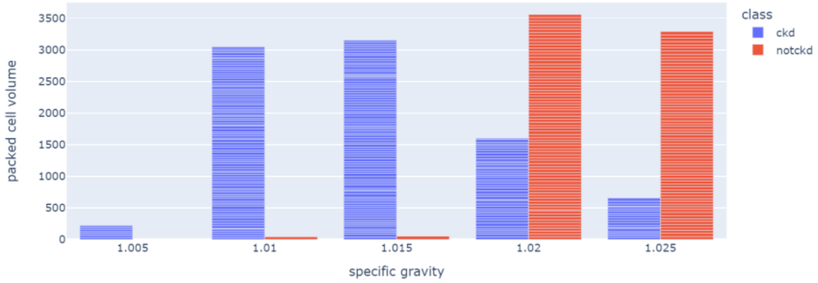
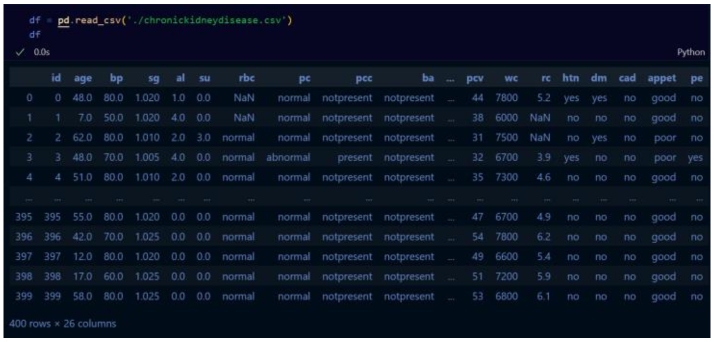
Date	5 July 2024
Team ID	SWTID1720082525
Project Title	Early Prediction of Chronic Kidney Disease Using Machine Learning
Maximum Marks	6 Marks

Data Exploration and Preprocessing Template

Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Section	**Description**
Data Overview	Dimension: 400 rows × 26 columns Descriptive statistics: 

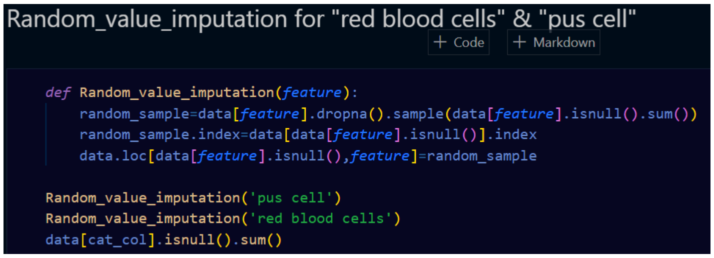
	id	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	36.
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	18.
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	53.
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	56.
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	106.0	26.
5	5	60.0	90.0	1.015	3.0	0.0	NaN	NaN	notpresent	notpresent	74.0	25.
6	6	68.0	70.0	1.010	0.0	0.0	NaN	normal	notpresent	notpresent	100.0	54.
7	7	24.0	NaN	1.015	2.0	4.0	normal	abnormal	notpresent	notpresent	410.0	31.
8	8	52.0	100.0	1.015	3.0	0.0	normal	abnormal	present	notpresent	138.0	60.
9	9	53.0	90.0	1.020	2.0	0.0	abnormal	abnormal	present	notpresent	70.0	107

Univariate Analysis	
Bivariate Analysis	
Multivariate Analysis	
Outliers and Anomalies	-
Data Preprocessing Code Screenshots	
Loading Data	

```python
df = pd.read_csv('./chronickidneydisease.csv')
df
```

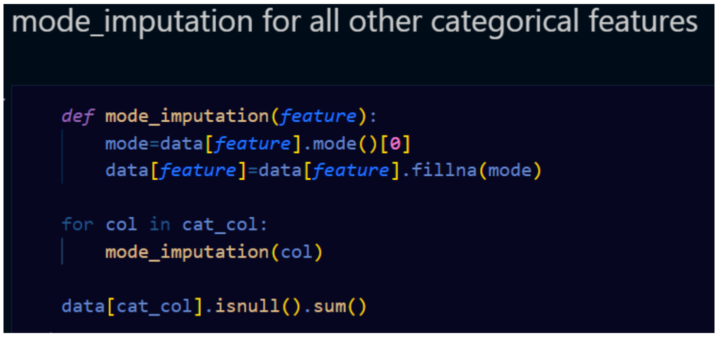
```
✓ 0.0s                                                                                                  Python

      id   age    bp     sg   al   su     rbc        pc         pcc          ba  ...  pcv    wc   rc  htn  dm  cad  appet   pe
  0    0  48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent  notpresent  ...   44  7800  5.2  yes  yes  no   good   no
  1    1   7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent  notpresent  ...   38  6000  NaN   no   no  no   good   no
  2    2  62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent  notpresent  ...   31  7500  NaN   no  yes  no   poor   no
  3    3  48.0  70.0  1.005  4.0  0.0  normal  abnormal     present  notpresent  ...   32  6700  3.9  yes   no  no   poor  yes
  4    4  51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent  notpresent  ...   35  7300  4.6   no   no  no   good   no
...  ...   ...   ...    ...  ...  ...     ...       ...         ...         ...  ...  ...   ...  ...  ...  ...  ...    ...  ...
395  395  55.0  80.0  1.020  0.0  0.0  normal    normal  notpresent  notpresent  ...   47  6700  4.9   no   no  no   good   no
396  396  42.0  70.0  1.025  0.0  0.0  normal    normal  notpresent  notpresent  ...   54  7800  6.2   no   no  no   good   no
397  397  12.0  80.0  1.020  0.0  0.0  normal    normal  notpresent  notpresent  ...   49  6600  5.4   no   no  no   good   no
398  398  17.0  60.0  1.025  0.0  0.0  normal    normal  notpresent  notpresent  ...   51  7200  5.9   no   no  no   good   no
399  399  58.0  80.0  1.025  0.0  0.0  normal    normal  notpresent  notpresent  ...   53  6800  6.1   no   no  no   good   no

400 rows × 26 columns
```

<table>
<tr>
<td>Handling Missing Data</td>
<td>



Random_value_imputation for "red blood cells" & "pus cell"

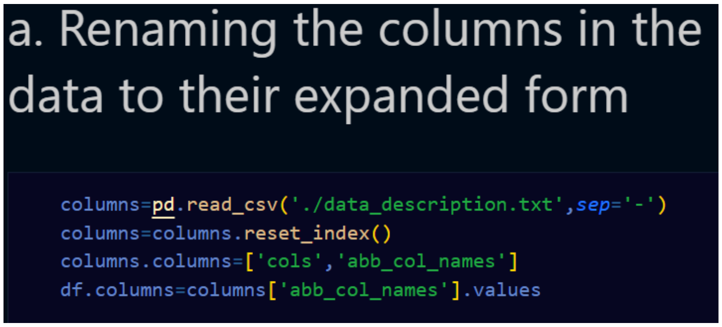
```
def Random_value_imputation(feature):
    random_sample=data[feature].dropna().sample(data[feature].isnull().sum())
    random_sample.index=data[data[feature].isnull()].index
    data.loc[data[feature].isnull(),feature]=random_sample

Random_value_imputation('pus cell')
Random_value_imputation('red blood cells')
data[cat_col].isnull().sum()
```



mode_imputation for all other categorical features

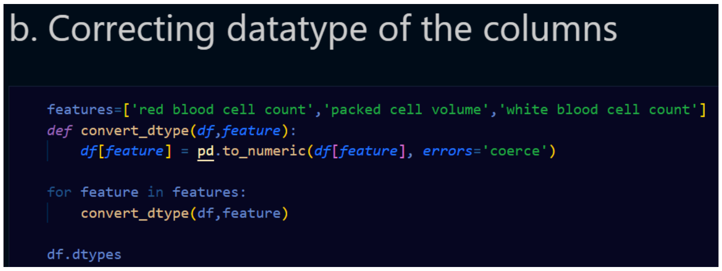
```
def mode_imputation(feature):
    mode=data[feature].mode()[0]
    data[feature]=data[feature].fillna(mode)

for col in cat_col:
    mode_imputation(col)

data[cat_col].isnull().sum()
```

</td>
</tr>
<tr>
<td>Data Transformation</td>
<td>



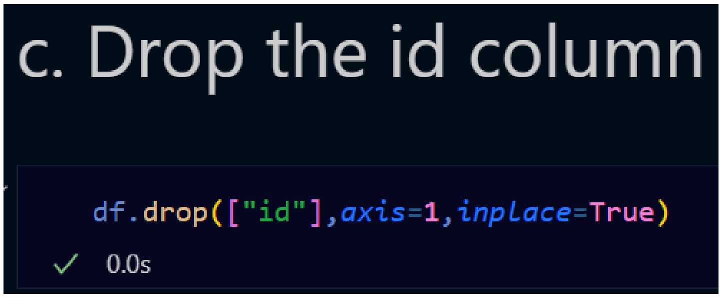
a. Renaming the columns in the data to their expanded form

```
columns=pd.read_csv('./data_description.txt',sep='-')
columns=columns.reset_index()
columns.columns=['cols','abb_col_names']
df.columns=columns['abb_col_names'].values
```



b. Correcting datatype of the columns

```
features=['red blood cell count','packed cell volume','white blood cell count']
def convert_dtype(df,feature):
    df[feature] = pd.to_numeric(df[feature], errors='coerce')

for feature in features:
    convert_dtype(df,feature)

df.dtypes
```

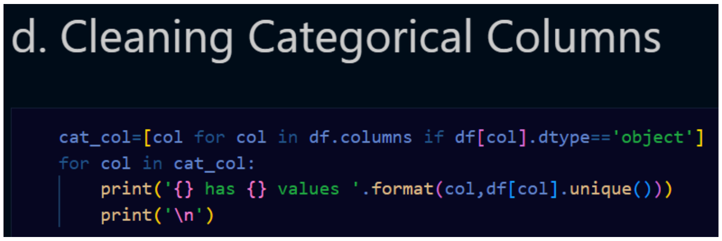


c. Drop the id column

```
df.drop(["id"],axis=1,inplace=True)
```

✓ 0.0s

</td>
</tr>
</table>

	 d. Cleaning Categorical Columns cat_col=[col for col in df.columns if df[col].dtype=='object'] for col in cat_col: print('{} has {} values '.format(col,df[col].unique())) print('\n')
Feature Engineering	Attached the codes in final submission.
Save Processed Data	-