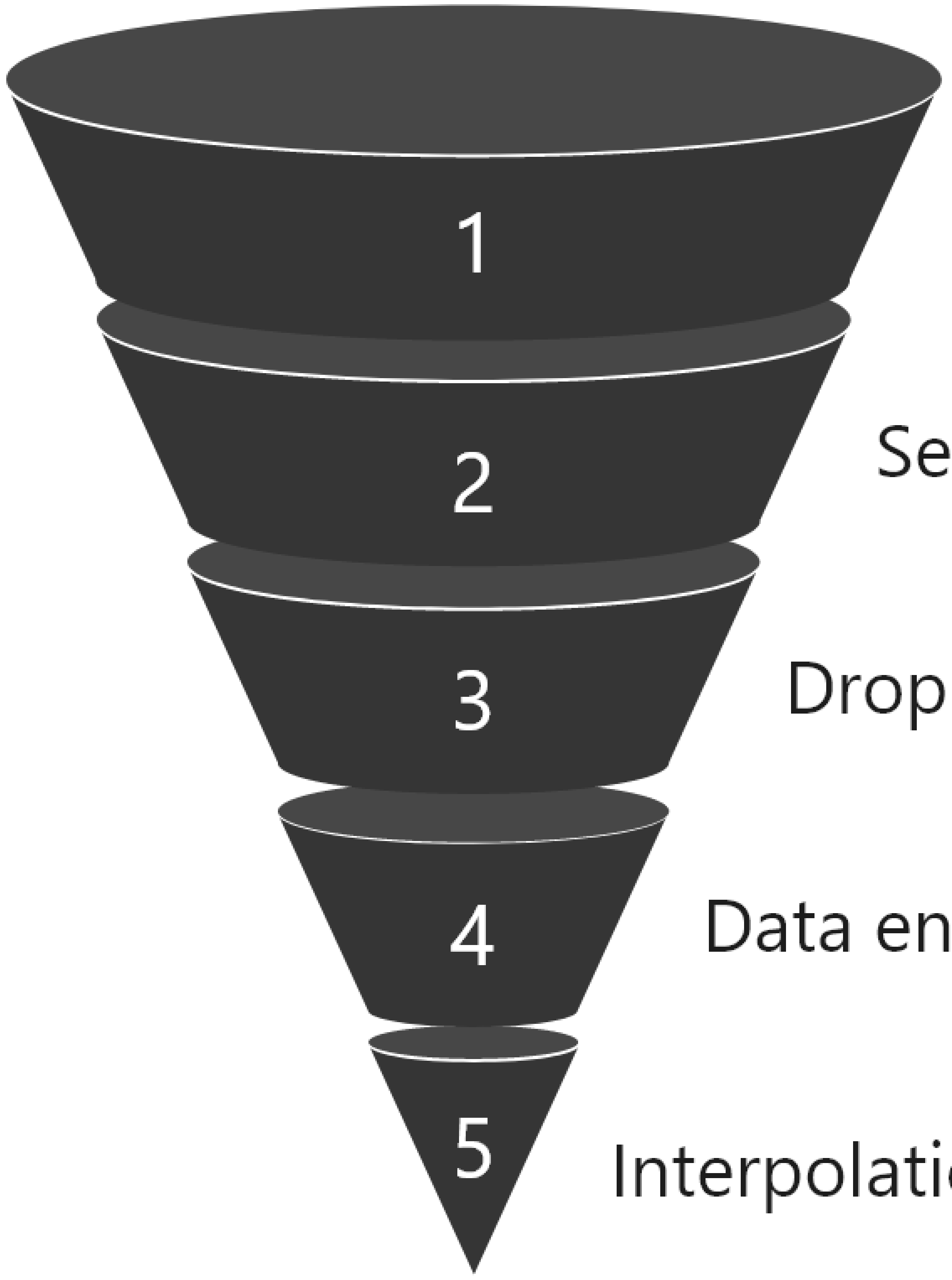
Python Group Assignment

Group - Swachh Data Sangathan

Amit, Jyoti , Tathagat, Soumyadeep

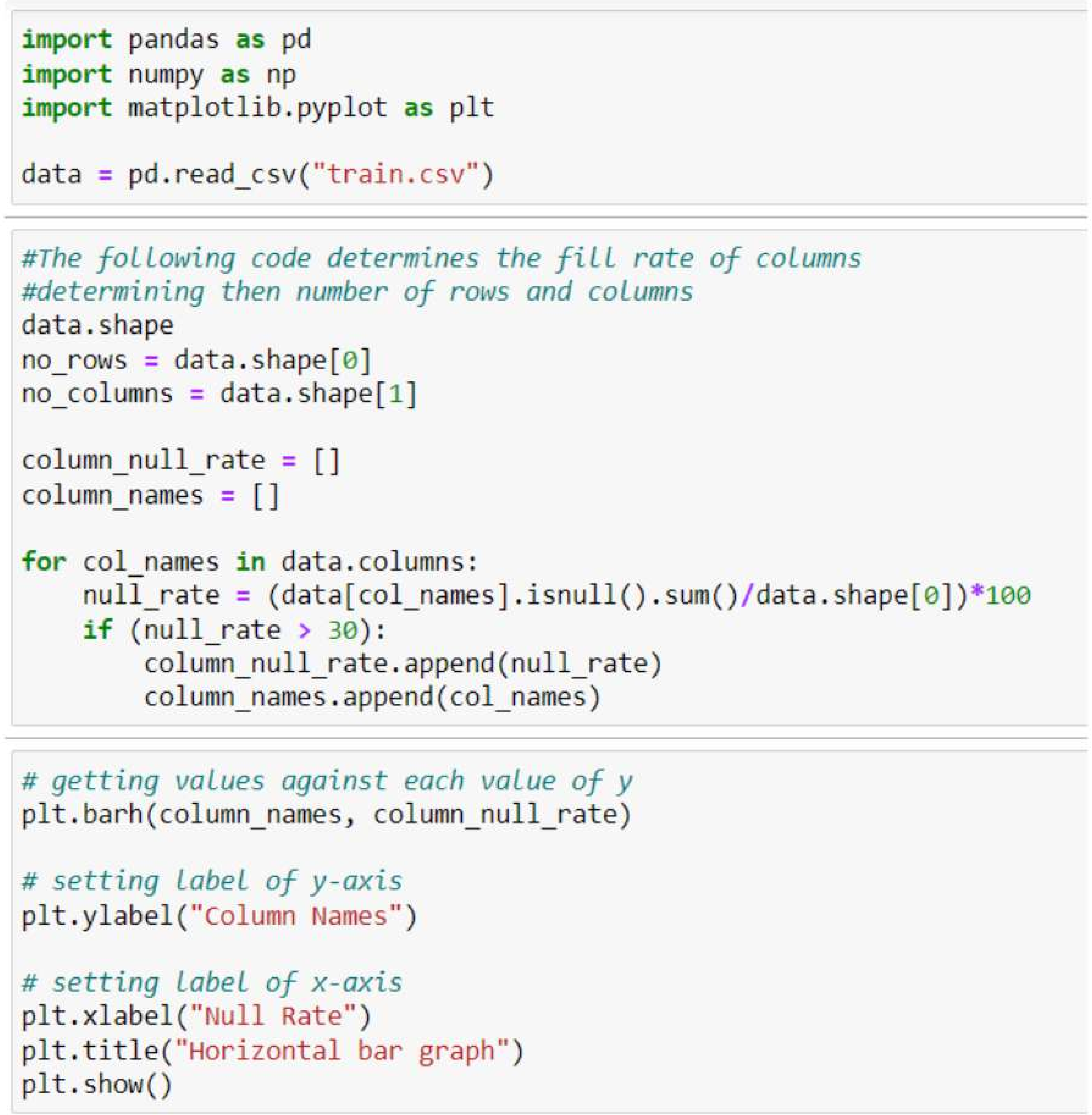
Proceed >

STEPS IN DATA CLEANING



1. Dimension Reduction Based on null values & correlation
2. Segregating Numerical & Categorical Data
3. Dropping irrelevant columns
4. Data encoding
5. Interpolation

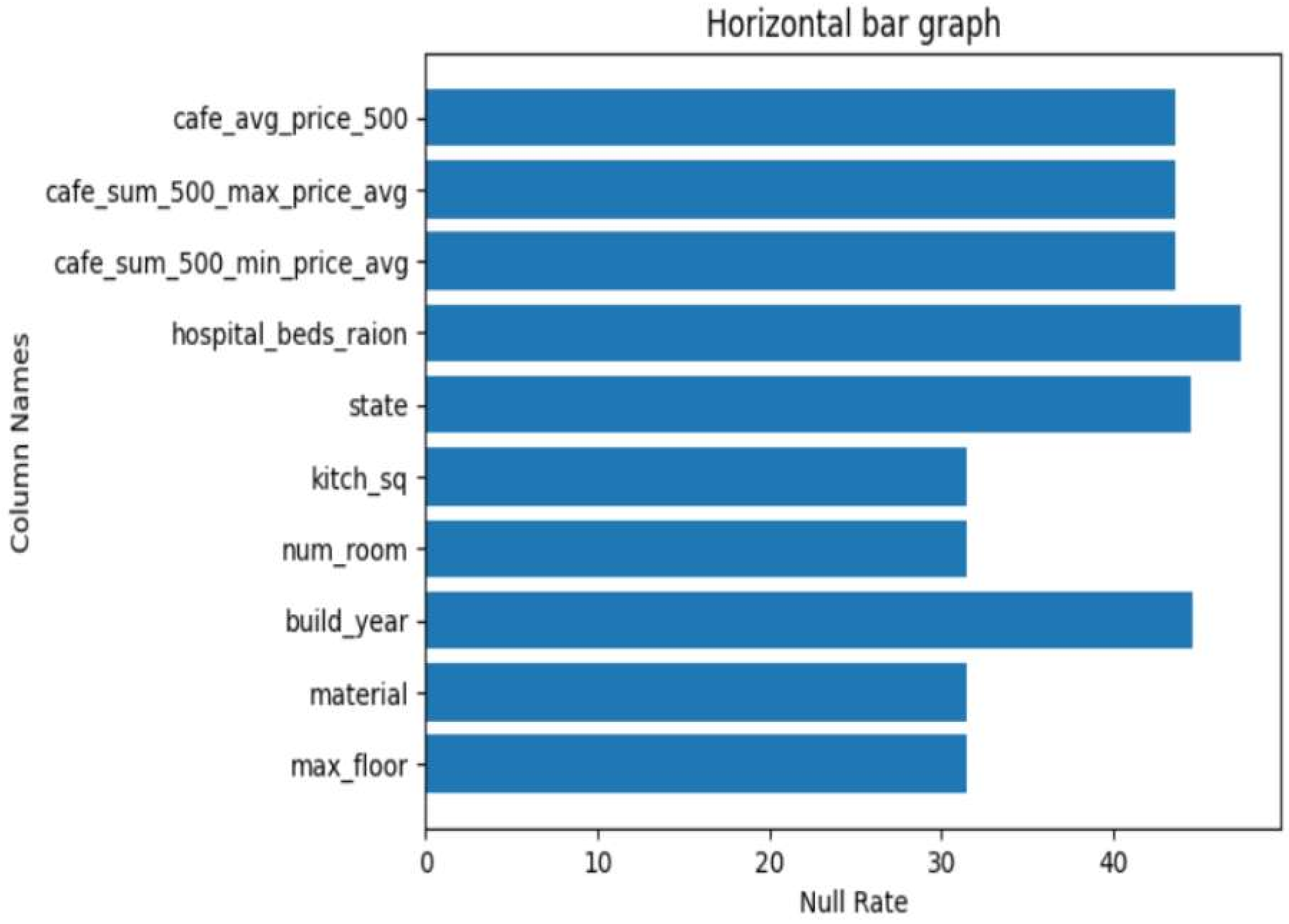
Dimension Reduction Based on null values & correlation



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

data = pd.read_csv("train.csv")
```

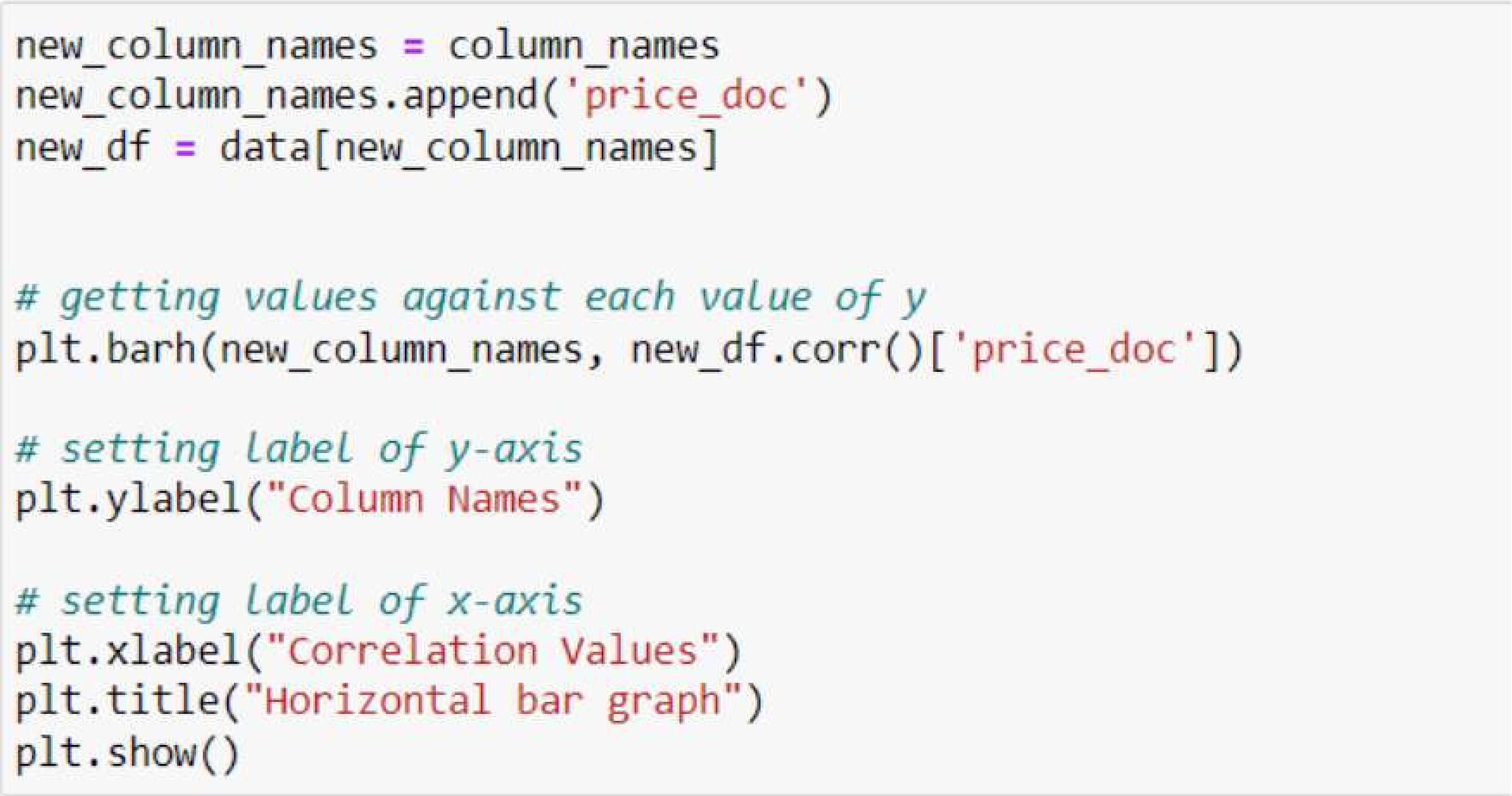
```python
#The following code determines the fill rate of columns
#determining then number of rows and columns
data.shape
no_rows = data.shape[0]
no_columns = data.shape[1]

column_null_rate = []
column_names = []

for col_names in data.columns:
    null_rate = (data[col_names].isnull().sum()/data.shape[0])*100
    if (null_rate > 30):
        column_null_rate.append(null_rate)
        column_names.append(col_names)
```

```python
# getting values against each value of y
plt.barh(column_names, column_null_rate)

# setting label of y-axis
plt.ylabel("Column Names")

# setting label of x-axis
plt.xlabel("Null Rate")
plt.title("Horizontal bar graph")
plt.show()
```

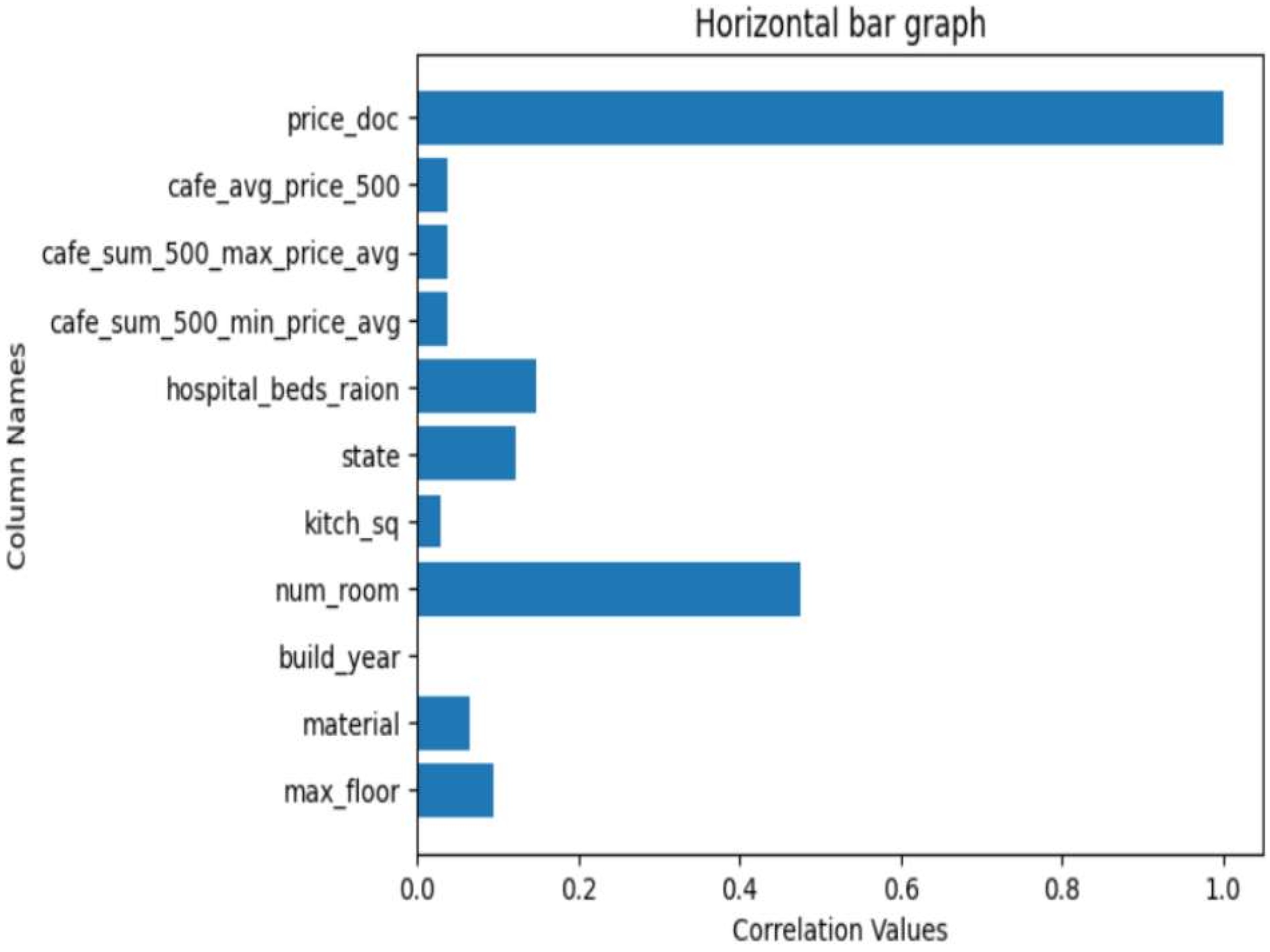


Dimension Reduction Based on null values & correlation

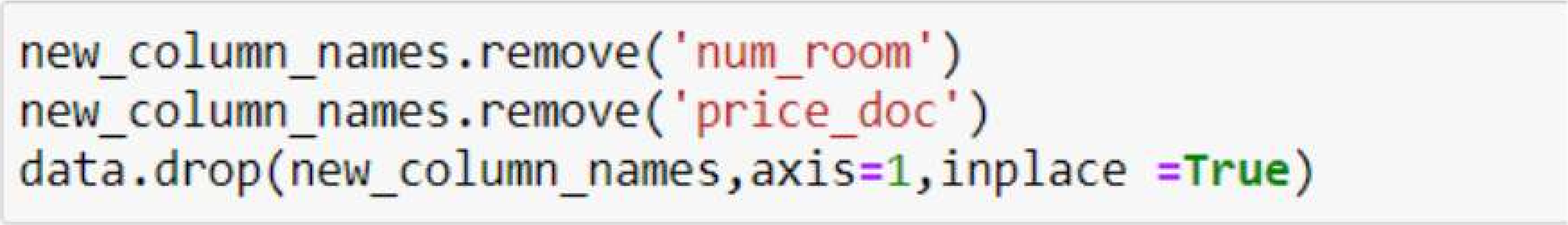
CREATING CORRELATION BAR GRAPH OF CERTAIN COLUMNS WITH RESPECT TO PRICE



```
new_column_names = column_names
new_column_names.append('price_doc')
new_df = data[new_column_names]


# getting values against each value of y
plt.barh(new_column_names, new_df.corr()['price_doc'])

# setting label of y-axis
plt.ylabel("Column Names")

# setting label of x-axis
plt.xlabel("Correlation Values")
plt.title("Horizontal bar graph")
plt.show()
```



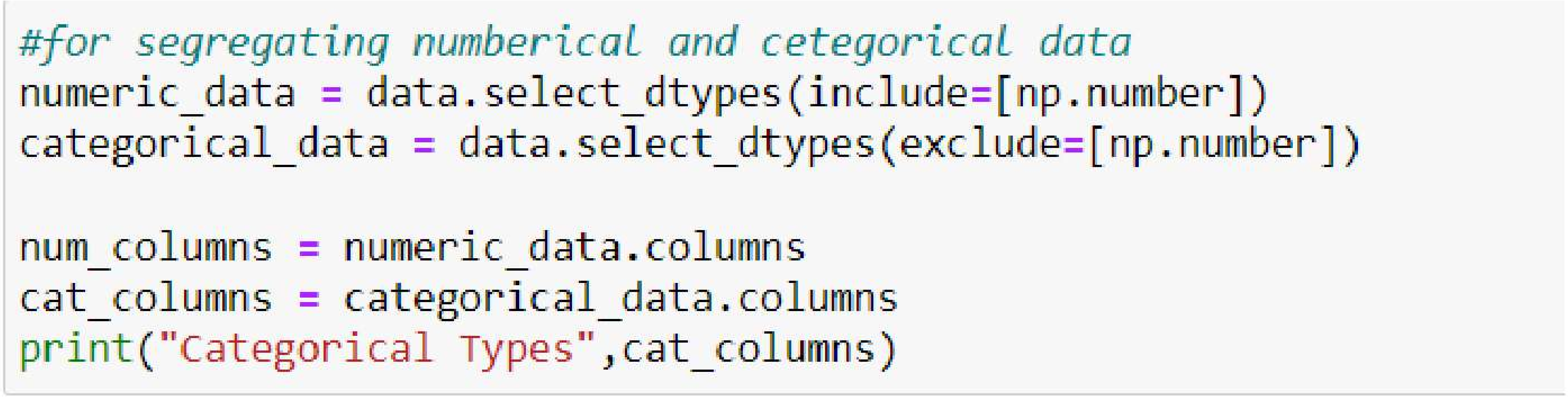
REMOVING COLUMNS WITH LOW CORRELATION AND HIGH NULL RATE



```
new_column_names.remove('num_room')
new_column_names.remove('price_doc')
data.drop(new_column_names,axis=1,inplace =True)
```

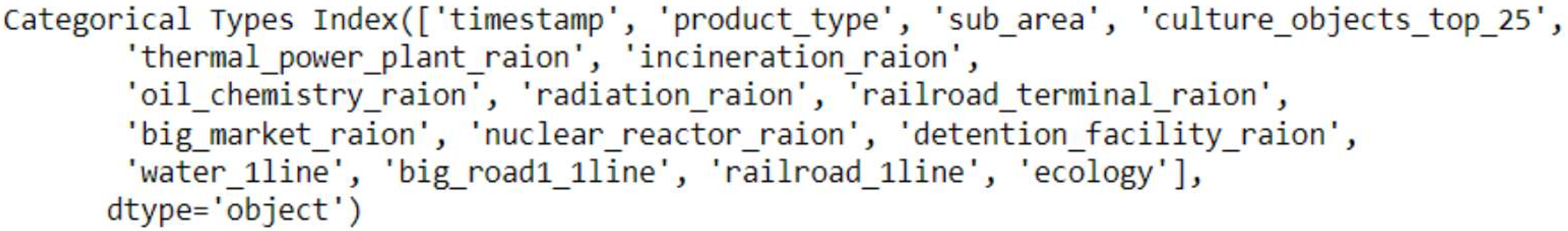
Segregating Numerical & Categorical Data

CODE FOR SEGREGATING NUMERICAL & CATEGORICAL DATA USING NUMPY



```
#for segregating numberical and cetegorical data
numeric_data = data.select_dtypes(include=[np.number])
categorical_data = data.select_dtypes(exclude=[np.number])

num_columns = numeric_data.columns
cat_columns = categorical_data.columns
print("Categorical Types",cat_columns)
```

OUTPUT



```
Categorical Types Index(['timestamp', 'product_type', 'sub_area', 'culture_objects_top_25',
       'thermal_power_plant_raion', 'incineration_raion',
       'oil_chemistry_raion', 'radiation_raion', 'railroad_terminal_raion',
       'big_market_raion', 'nuclear_reactor_raion', 'detention_facility_raion',
       'water_1line', 'big_road1_1line', 'railroad_1line', 'ecology'],
      dtype='object')
```

Data Encoding

PURPOSE OF DATA ENCODING

MACHINE TYPE
ASSEMBLY-LINE
MAINFRAME
SERVER

DATA CATEGORIES

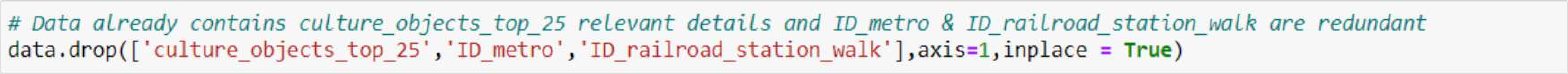
MACHINE TYPE
1
2
3

ENCODING OF DATA CATEGORIES WITH UNEQUAL WEIGHTAGE

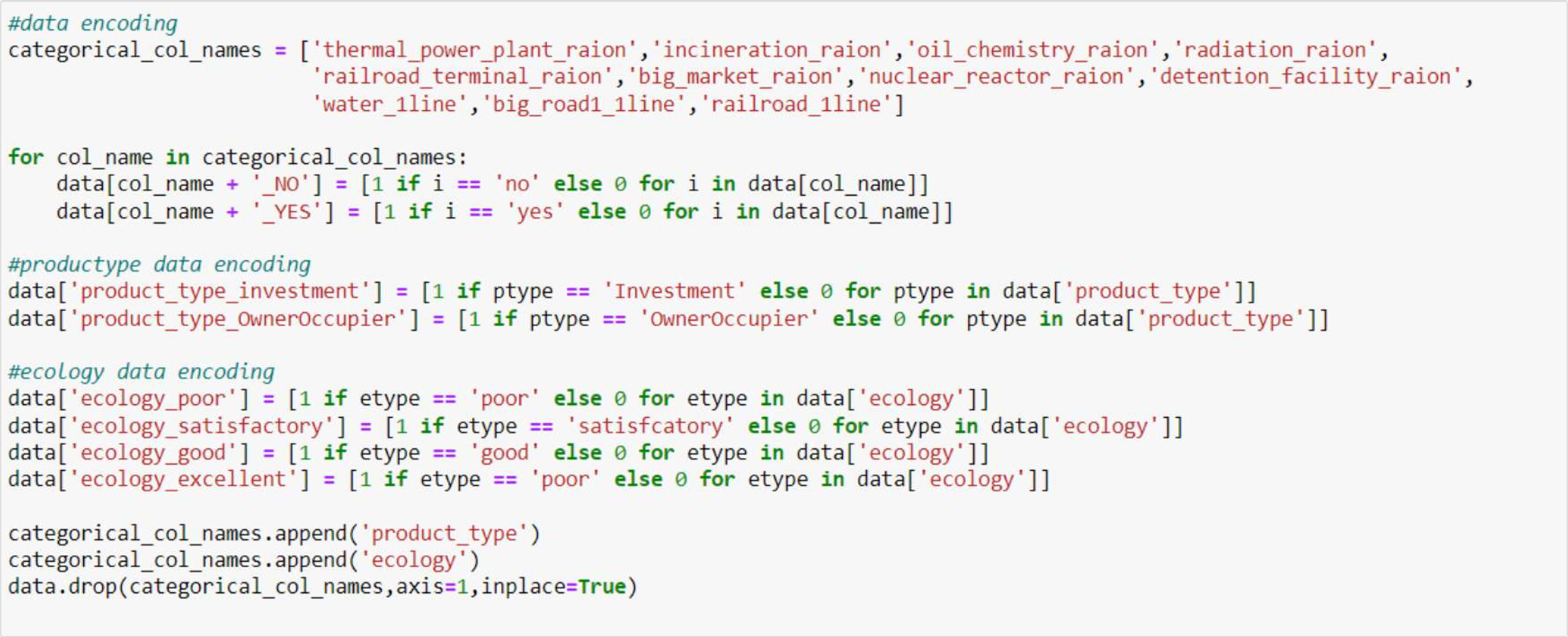
MACHINE TYPE_ ASSEMBLY LINE	MACHINE TYPE_ MAINFRAME	MACHINE TYPE_ SERVER
1	0	0
0	1	0
0	0	1

ENCODING OF DATA CATEGORIES WITH EQUAL WEIGHTAGE

Data Encoding

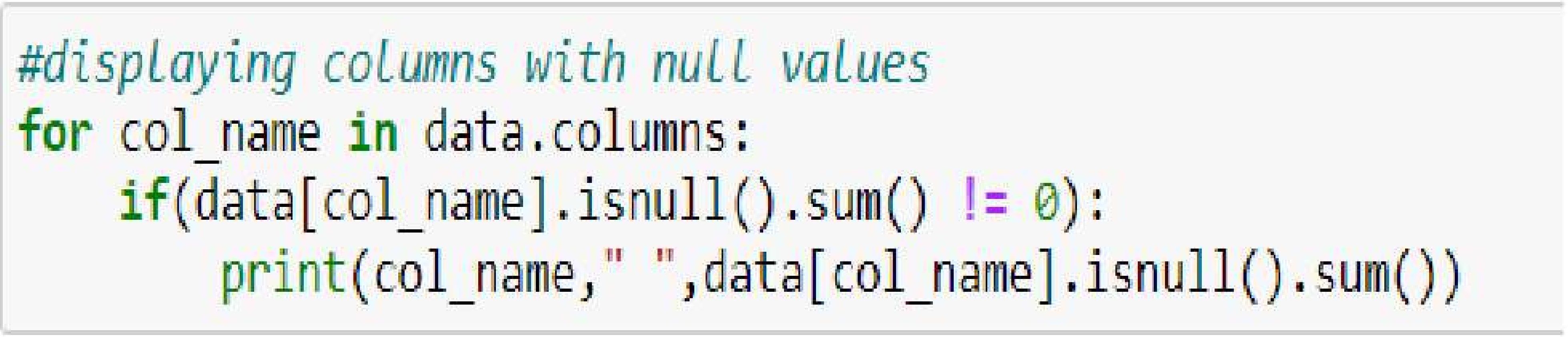


```python
# Data already contains culture_objects_top_25 relevant details and ID_metro & ID_railroad_station_walk are redundant
data.drop(['culture_objects_top_25','ID_metro','ID_railroad_station_walk'],axis=1,inplace = True)
```



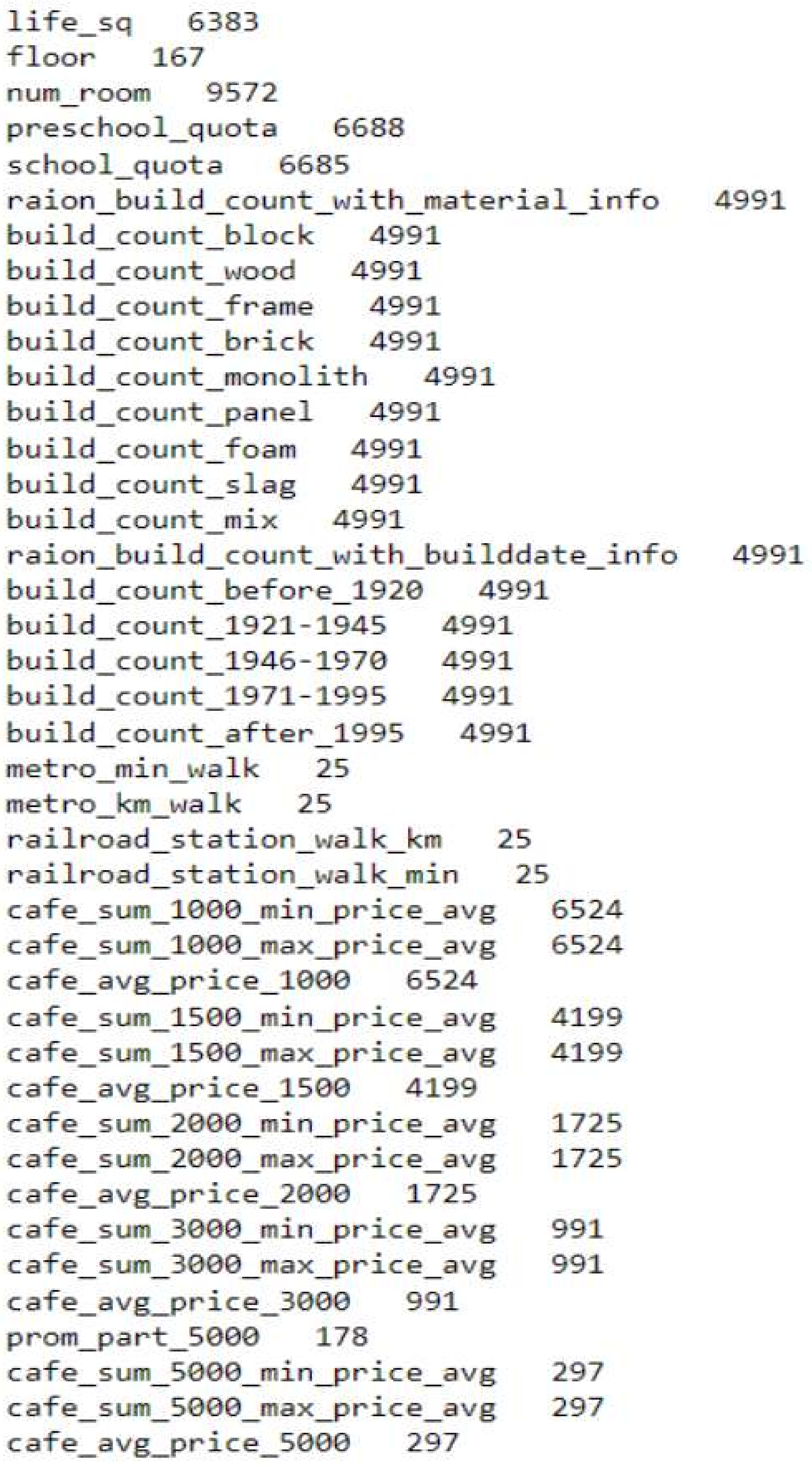
```python
#data encoding
categorical_col_names = ['thermal_power_plant_raion','incineration_raion','oil_chemistry_raion','radiation_raion',
                         'railroad_terminal_raion','big_market_raion','nuclear_reactor_raion','detention_facility_raion',
                         'water_1line','big_road1_1line','railroad_1line']

for col_name in categorical_col_names:
    data[col_name + '_NO'] = [1 if i == 'no' else 0 for i in data[col_name]]
    data[col_name + '_YES'] = [1 if i == 'yes' else 0 for i in data[col_name]]

#productype data encoding
data['product_type_investment'] = [1 if ptype == 'Investment' else 0 for ptype in data['product_type']]
data['product_type_OwnerOccupier'] = [1 if ptype == 'OwnerOccupier' else 0 for ptype in data['product_type']]

#ecology data encoding
data['ecology_poor'] = [1 if etype == 'poor' else 0 for etype in data['ecology']]
data['ecology_satisfactory'] = [1 if etype == 'satisfcatory' else 0 for etype in data['ecology']]
data['ecology_good'] = [1 if etype == 'good' else 0 for etype in data['ecology']]
data['ecology_excellent'] = [1 if etype == 'poor' else 0 for etype in data['ecology']]

categorical_col_names.append('product_type')
categorical_col_names.append('ecology')
data.drop(categorical_col_names,axis=1,inplace=True)
```

Interpolation



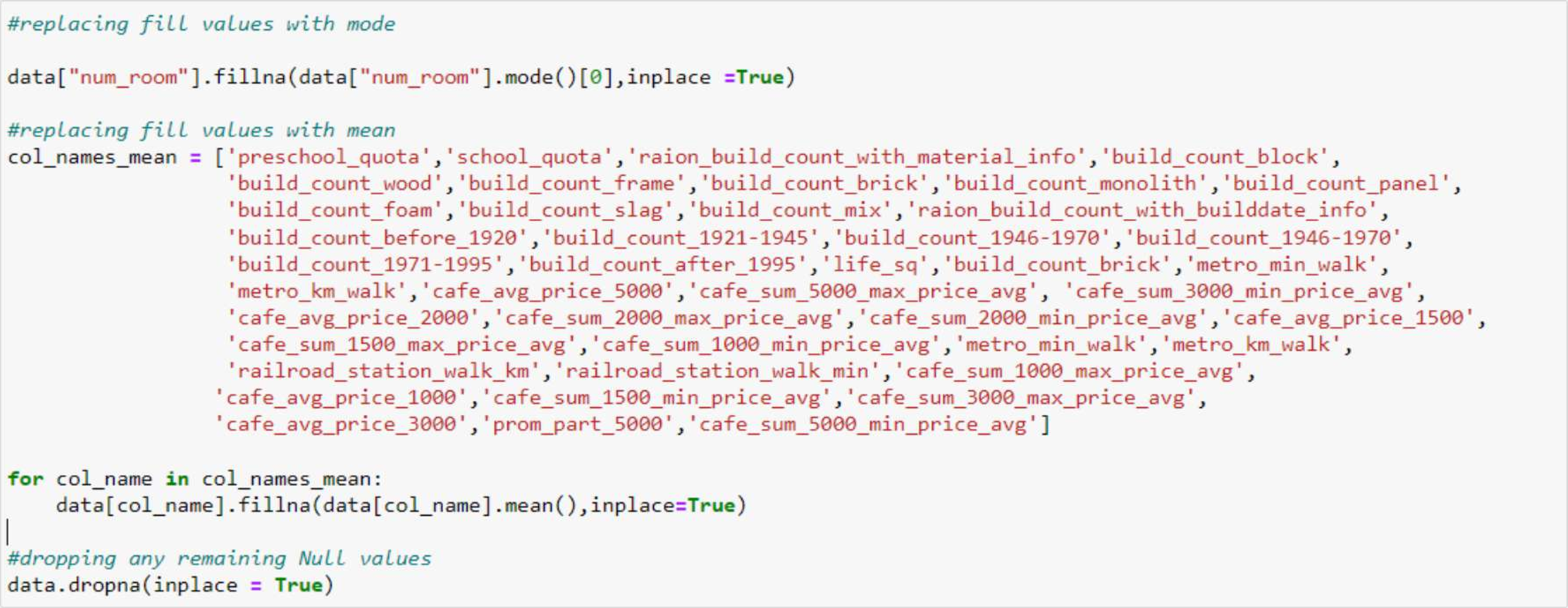
```
#displaying columns with null values
for col_name in data.columns:
    if(data[col_name].isnull().sum() != 0):
        print(col_name," ",data[col_name].isnull().sum())
```

THIS PIECE OF CODE DISPLAYS ALL COLUMNS WITH NULL VALUE



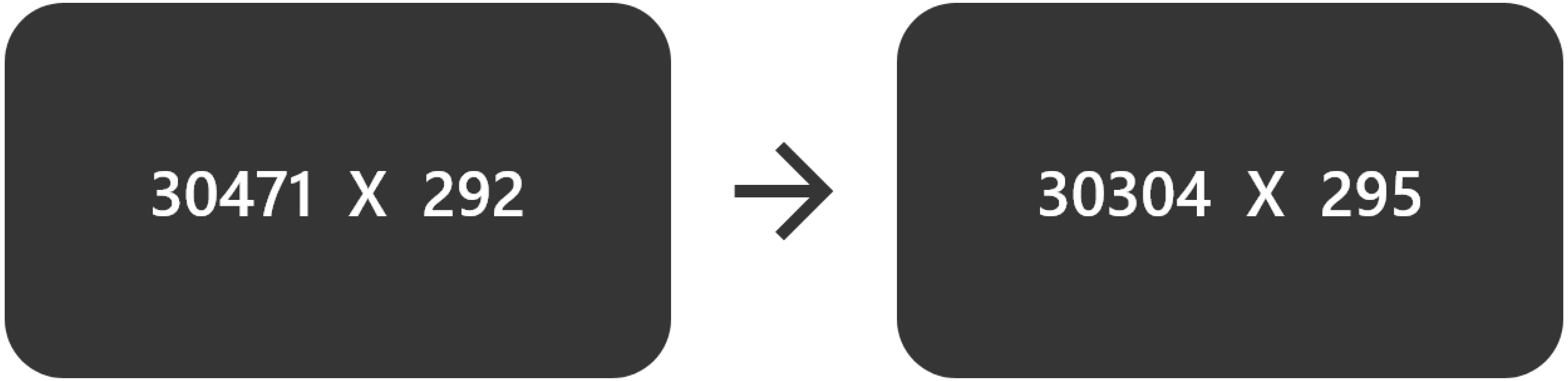
```
life_sq   6383
floor   167
num_room   9572
preschool_quota   6688
school_quota   6685
raion_build_count_with_material_info   4991
build_count_block   4991
build_count_wood   4991
build_count_frame   4991
build_count_brick   4991
build_count_monolith   4991
build_count_panel   4991
build_count_foam   4991
build_count_slag   4991
build_count_mix   4991
raion_build_count_with_builddate_info   4991
build_count_before_1920   4991
build_count_1921-1945   4991
build_count_1946-1970   4991
build_count_1971-1995   4991
build_count_after_1995   4991
metro_min_walk   25
metro_km_walk   25
railroad_station_walk_km   25
railroad_station_walk_min   25
cafe_sum_1000_min_price_avg   6524
cafe_sum_1000_max_price_avg   6524
cafe_avg_price_1000   6524
cafe_sum_1500_min_price_avg   4199
cafe_sum_1500_max_price_avg   4199
cafe_avg_price_1500   4199
cafe_sum_2000_min_price_avg   1725
cafe_sum_2000_max_price_avg   1725
cafe_avg_price_2000   1725
cafe_sum_3000_min_price_avg   991
cafe_sum_3000_max_price_avg   991
cafe_avg_price_3000   991
prom_part_5000   178
cafe_sum_5000_min_price_avg   297
cafe_sum_5000_max_price_avg   297
cafe_avg_price_5000   297
```

Interpolation



```
#replacing fill values with mode

data["num_room"].fillna(data["num_room"].mode()[0],inplace =True)

#replacing fill values with mean
col_names_mean = ['preschool_quota','school_quota','raion_build_count_with_material_info','build_count_block',
                  'build_count_wood','build_count_frame','build_count_brick','build_count_monolith','build_count_panel',
                  'build_count_foam','build_count_slag','build_count_mix','raion_build_count_with_builddate_info',
                  'build_count_before_1920','build_count_1921-1945','build_count_1946-1970','build_count_1946-1970',
                  'build_count_1971-1995','build_count_after_1995','life_sq','build_count_brick','metro_min_walk',
                  'metro_km_walk','cafe_avg_price_5000','cafe_sum_5000_max_price_avg', 'cafe_sum_3000_min_price_avg',
                  'cafe_avg_price_2000','cafe_sum_2000_max_price_avg','cafe_sum_2000_min_price_avg','cafe_avg_price_1500',
                  'cafe_sum_1500_max_price_avg','cafe_sum_1000_min_price_avg','metro_min_walk','metro_km_walk',
                  'railroad_station_walk_km','railroad_station_walk_min','cafe_sum_1000_max_price_avg',
                 'cafe_avg_price_1000','cafe_sum_1500_min_price_avg','cafe_sum_3000_max_price_avg',
                 'cafe_avg_price_3000','prom_part_5000','cafe_sum_5000_min_price_avg']

for col_name in col_names_mean:
    data[col_name].fillna(data[col_name].mean(),inplace=True)

#dropping any remaining Null values
data.dropna(inplace = True)
```

SUMMARY



30471 X 292 → 30304 X 295

1. DIMENSION REDUCTION BASED ON NULL RATE AND CORRELATION WITH PRICE SUCCESSFULL

2. SEGGREGATED CATEGORICAL DATA AND DONE DATA ENCODING SUCCESSFULLY

3. LOST 167 OBSERVATION DUE TO NULL VALUES PRESENT IN DIMENSION : FLOOR

4. DATA READY TO BE APPLIED FOR MACHINE LEARNING MODELS LIKE LINEAR REGRESSION.

FINISH

THANK YOU