Initial Question:

Predicting Gorkha Earthquake Damage

Final Question:

Predicting the predefined level of damage to the buildings caused by the 2015 Gorkha Earthquake in Nepal based on geographic location and construction so that, Nepal Government can take appropriate measures for housing reconstruction

Problem Description:

We're trying to predict the ordinal variable damage_grade, which represents a level of damage to the building that was hit by the earthquake.

There are 3 grades of the damage:

1. represents low damage
2. represents a medium amount of damage
3. represents almost complete destruction

In [ ]:
```
# Importing libraries

import numpy as np
import pandas as pd
from pathlib import Path
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading the Data

In [8]:
```
DATA_DIR = Path('C://Users/ManojKumar Chalamala/Desktop/Capstone')
```

In [9]:
```
train_values = pd.read_csv(DATA_DIR / 'train_values.csv', index_col='building_id')
train_labels = pd.read_csv(DATA_DIR / 'train_labels.csv', index_col='building_id')
```

Examine the Data

In [10]:
```
train_values.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 260601 entries, 802906 to 747594
Data columns (total 38 columns):
geo_level_1_id                          260601 non-null int64
geo_level_2_id                          260601 non-null int64
geo_level_3_id                          260601 non-null int64
count_floors_pre_eq                     260601 non-null int64
age                                     260601 non-null int64
area_percentage                         260601 non-null int64
height_percentage                       260601 non-null int64
land_surface_condition                  260601 non-null object
foundation_type                         260601 non-null object
roof_type                               260601 non-null object
ground_floor_type                       260601 non-null object
other_floor_type                        260601 non-null object
position                                260601 non-null object
plan_configuration                      260601 non-null object
has_superstructure_adobe_mud            260601 non-null int64
has_superstructure_mud_mortar_stone     260601 non-null int64
has_superstructure_stone_flag           260601 non-null int64
has_superstructure_cement_mortar_stone  260601 non-null int64
has_superstructure_mud_mortar_brick     260601 non-null int64
has_superstructure_cement_mortar_brick  260601 non-null int64
has_superstructure_timber               260601 non-null int64
has_superstructure_bamboo               260601 non-null int64
has_superstructure_rc_non_engineered    260601 non-null int64
has_superstructure_rc_engineered        260601 non-null int64
has_superstructure_other                260601 non-null int64
legal_ownership_status                  260601 non-null object
count_families                          260601 non-null int64
has_secondary_use                       260601 non-null int64
has_secondary_use_agriculture           260601 non-null int64
has_secondary_use_hotel                 260601 non-null int64
has_secondary_use_rental                260601 non-null int64
has_secondary_use_institution           260601 non-null int64
has_secondary_use_school                260601 non-null int64
has_secondary_use_industry              260601 non-null int64
has_secondary_use_health_post           260601 non-null int64
has_secondary_use_gov_office            260601 non-null int64
has_secondary_use_use_police            260601 non-null int64
has_secondary_use_other                 260601 non-null int64
dtypes: int64(30), object(8)
memory usage: 77.5+ MB
```

In [20]:
```
train_labels.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 260601 entries, 802906 to 747594
Data columns (total 1 columns):
damage_grade    260601 non-null int64
dtypes: int64(1)
memory usage: 4.0 MB
```

In [19]:
```
train_values.shape
```

Out[19]: (260601, 38)

In [21]:
```
train_labels.shape
```

Out[21]: (260601, 1)

Top and Bottom of the Data

In [22]:
```
train_values.head()
```

Out[22]:

building_id	geo_level_1_id	geo_level_2_id	geo_level_3_id	count_floors_pre_eq	age	area_perce
802906	6	487	12198	2	30	
28830	8	900	2812	2	10	
94947	21	363	8973	2	10	
590882	22	418	10694	2	10	
201944	11	131	1488	3	30	

5 rows × 38 columns

In [23]:
```
train_labels.head()
```

Out[23]:

building_id	damage_grade
802906	3
28830	2
94947	3
590882	2
201944	3

In [24]:
```
train_values.tail()
```

Out[24]:

building_id	geo_level_1_id	geo_level_2_id	geo_level_3_id	count_floors_pre_eq	age	area_perce
688636	25	1335	1621	1	55	
669485	17	715	2060	2	0	
602512	17	51	8163	3	55	
151409	26	39	1851	2	10	
747594	21	9	9101	3	10	

5 rows × 38 columns

In [25]:
```
train_labels.tail()
```

Out[25]:

building_id	damage_grade
688636	2
669485	3
602512	3
151409	2
747594	3

Explore the Data

In [26]:
```
train_values.describe()
```

Out[26]:

	geo_level_1_id	geo_level_2_id	geo_level_3_id	count_floors_pre_eq	age	area_p
count	260601.000000	260601.000000	260601.000000	260601.000000	260601.000000	260(
mean	13.900353	701.074685	6257.876148	2.129723	26.535029	
std	8.033617	412.710734	3646.369645	0.727665	73.565937	
min	0.000000	0.000000	0.000000	1.000000	0.000000	
25%	7.000000	350.000000	3073.000000	2.000000	10.000000	
50%	12.000000	702.000000	6270.000000	2.000000	15.000000	
75%	21.000000	1050.000000	9412.000000	2.000000	30.000000	
max	30.000000	1427.000000	12567.000000	9.000000	995.000000	

8 rows × 30 columns

In [62]:
```
# Explore the Statistical features of data
train_values.describe().T.style.background_gradient(cmap='Set2',axis=0)
```

Out[62]:

	count	mean	std	min	25%	50%	75
geo_level_1_id	260601	13.9004	8.03362	0	7	12	2
geo_level_2_id	260601	701.075	412.711	0	350	702	105
geo_level_3_id	260601	6257.88	3646.37	0	3073	6270	941
count_floors_pre_eq	260601	2.12972	0.727665	1	2	2	
age	260601	26.535	73.5659	0	10	15	3
area_percentage	260601	8.01805	4.39223	1	5	7	
height_percentage	260601	5.43437	1.91842	2	4	5	
has_superstructure_adobe_mud	260601	0.0886451	0.284231	0	0	0	
has_superstructure_mud_mortar_stone	260601	0.761935	0.4259	0	1	1	
has_superstructure_stone_flag	260601	0.0343322	0.182081	0	0	0	
has_superstructure_cement_mortar_stone	260601	0.0182348	0.1338	0	0	0	
has_superstructure_mud_mortar_brick	260601	0.068154	0.25201	0	0	0	
has_superstructure_cement_mortar_brick	260601	0.0752683	0.263824	0	0	0	
has_superstructure_timber	260601	0.254988	0.435855	0	0	0	
has_superstructure_bamboo	260601	0.0850112	0.278899	0	0	0	
has_superstructure_rc_non_engineered	260601	0.04259	0.201931	0	0	0	
has_superstructure_rc_engineered	260601	0.0158595	0.124932	0	0	0	
has_superstructure_other	260601	0.0149846	0.121491	0	0	0	
count_families	260601	0.983949	0.418389	0	1	1	
has_secondary_use	260601	0.11188	0.315219	0	0	0	
has_secondary_use_agriculture	260601	0.0643781	0.245426	0	0	0	
has_secondary_use_hotel	260601	0.0336261	0.180265	0	0	0	
has_secondary_use_rental	260601	0.00810051	0.0896377	0	0	0	
has_secondary_use_institution	260601	0.000940135	0.0306473	0	0	0	
has_secondary_use_school	260601	0.000360705	0.0189888	0	0	0	
has_secondary_use_industry	260601	0.0010706	0.0327026	0	0	0	
has_secondary_use_health_post	260601	0.000188027	0.013711	0	0	0	
has_secondary_use_gov_office	260601	0.000145817	0.0120746	0	0	0	
has_secondary_use_use_police	260601	8.82575e-05	0.00939415	0	0	0	
has_secondary_use_other	260601	0.00511894	0.0713635	0	0	0	

In [60]:
```
# checking the types of varibles in the dataset(int,float,object)
dtypes=pd.DataFrame(train_values.dtypes,columns=["Data Type"])
dtypes["Unique Values"]=train_values.nunique()
dtypes["Null Values"]=train_values.isnull().sum()
dtypes.style.background_gradient(cmap='Set2',axis=0)
```

Out[60]:

	Data Type	Unique Values	Null Values
geo_level_1_id	int64	31	0
geo_level_2_id	int64	1414	0
geo_level_3_id	int64	11595	0
count_floors_pre_eq	int64	9	0
age	int64	42	0
area_percentage	int64	84	0
height_percentage	int64	27	0
land_surface_condition	object	3	0
foundation_type	object	5	0
roof_type	object	3	0
ground_floor_type	object	5	0
other_floor_type	object	4	0
position	object	4	0
plan_configuration	object	10	0
has_superstructure_adobe_mud	int64	2	0
has_superstructure_mud_mortar_stone	int64	2	0
has_superstructure_stone_flag	int64	2	0
has_superstructure_cement_mortar_stone	int64	2	0
has_superstructure_mud_mortar_brick	int64	2	0
has_superstructure_cement_mortar_brick	int64	2	0
has_superstructure_timber	int64	2	0
has_superstructure_bamboo	int64	2	0
has_superstructure_rc_non_engineered	int64	2	0
has_superstructure_rc_engineered	int64	2	0
has_superstructure_other	int64	2	0
legal_ownership_status	object	4	0
count_families	int64	10	0
has_secondary_use	int64	2	0
has_secondary_use_agriculture	int64	2	0
has_secondary_use_hotel	int64	2	0
has_secondary_use_rental	int64	2	0
has_secondary_use_institution	int64	2	0
has_secondary_use_school	int64	2	0
has_secondary_use_industry	int64	2	0
has_secondary_use_health_post	int64	2	0

	Data Type	Unique Values	Null Values
has_secondary_use_gov_office	int64	2	0
has_secondary_use_use_police	int64	2	0
has_secondary_use_other	int64	2	0

In [40]:
```
damage_level = train_labels['damage_grade']
damage_level.unique()
```

Out[40]:
```
array([3, 2, 1], dtype=int64)
```

In [41]:
```
damage_level.value_counts()
```

Out[41]:
```
2    148259
3     87218
1     25124
Name: damage_grade, dtype: int64
```

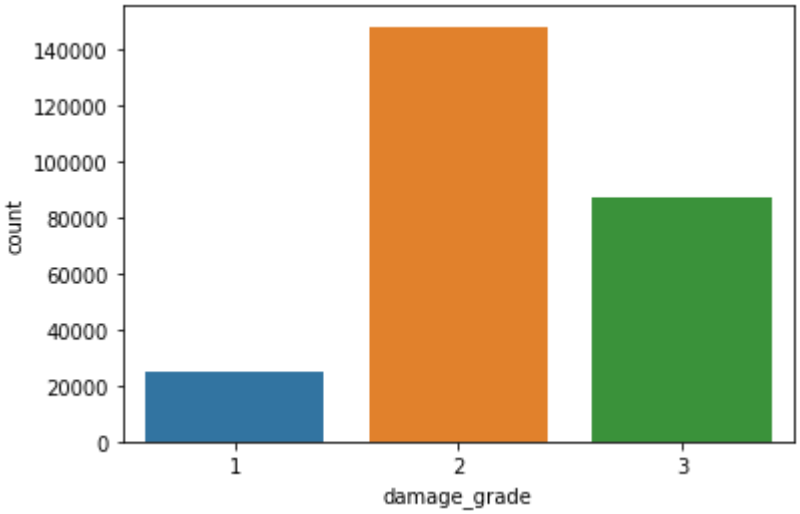
In [42]:
```
train_values.isna().any()
```

Out[42]:
```
geo_level_1_id                          False
geo_level_2_id                          False
geo_level_3_id                          False
count_floors_pre_eq                     False
age                                     False
area_percentage                         False
height_percentage                       False
land_surface_condition                  False
foundation_type                         False
roof_type                               False
ground_floor_type                       False
other_floor_type                        False
position                                False
plan_configuration                      False
has_superstructure_adobe_mud            False
has_superstructure_mud_mortar_stone     False
has_superstructure_stone_flag           False
has_superstructure_cement_mortar_stone  False
has_superstructure_mud_mortar_brick     False
has_superstructure_cement_mortar_brick  False
has_superstructure_timber               False
has_superstructure_bamboo               False
has_superstructure_rc_non_engineered    False
has_superstructure_rc_engineered        False
has_superstructure_other                False
legal_ownership_status                  False
count_families                          False
has_secondary_use                       False
has_secondary_use_agriculture           False
has_secondary_use_hotel                 False
has_secondary_use_rental                False
has_secondary_use_institution           False
has_secondary_use_school                False
has_secondary_use_industry              False
has_secondary_use_health_post           False
has_secondary_use_gov_office            False
has_secondary_use_use_police            False
has_secondary_use_other                 False
dtype: bool
```

Visualize the Data

In [44]:
```
sns.countplot(train_labels['damage_grade'])
```

Out[44]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x225dff1f4c8>
```

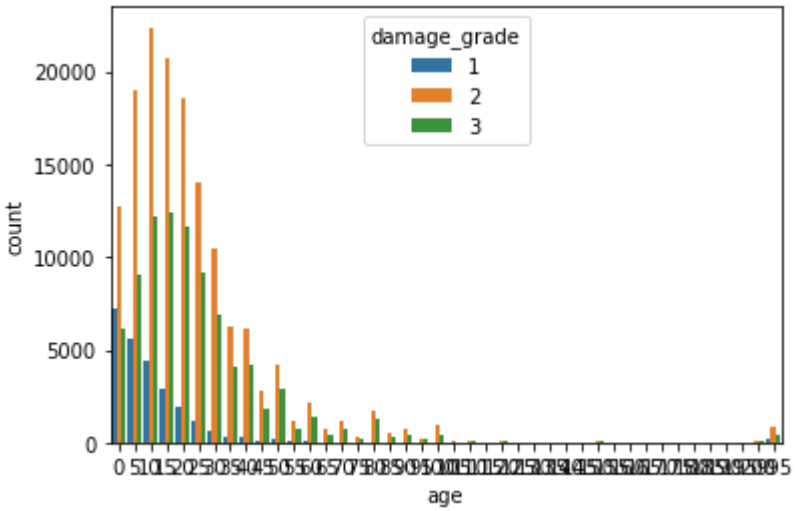


The majority of buildings almost 50% of buildings are falling under damage_grade level 2 represents medium level of damage

Visualize if age of building is affecting the damage

In [58]:
```
sns.countplot(x = train_values["age"], hue = train_labels["damage_grade"])
```

Out[58]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x225e8aea7c8>
```

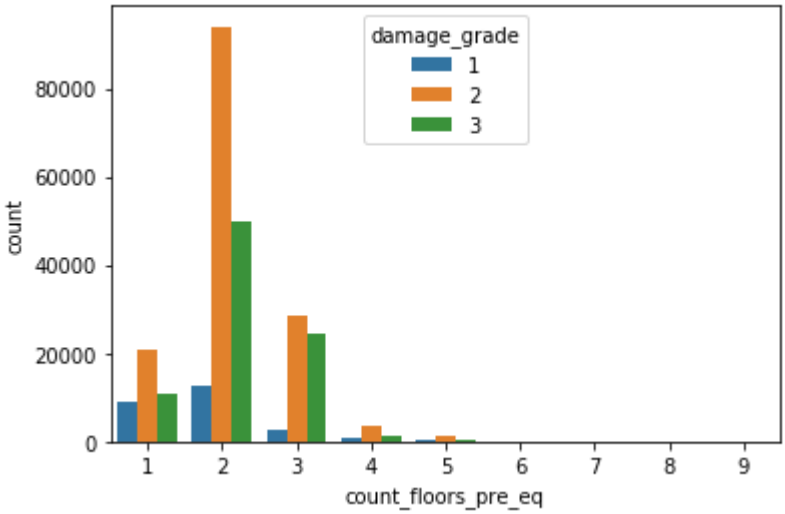


Visualize if no. of floors of a building affecting the level of damage

Majority of the buildings are falling under 0 to 50 years Most of the buildings has medium level of damage. So despite the age of building the level of damage is almost medium

In [59]:
```
sns.countplot(x = train_values["count_floors_pre_eq"], hue = train_labels["dam
age_grade"])
```

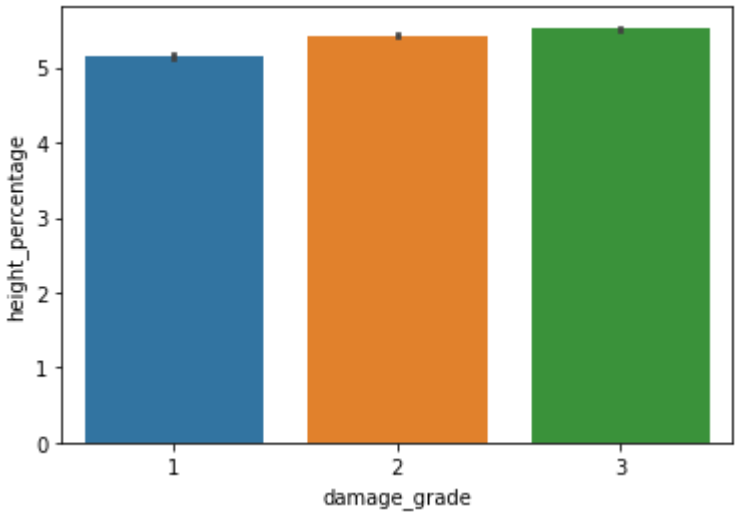
Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x225edccd348>



In [ ]:
```
Majority of the buildings are falling under 1 to 3 floors
Most of the buildings has medium level of damage. So despite the floor count t
he level of damage is almost medium
```

In [69]:
```
sns.barplot(y = train_values["height_percentage"], x = train_labels["damage_gr
ade"])
```

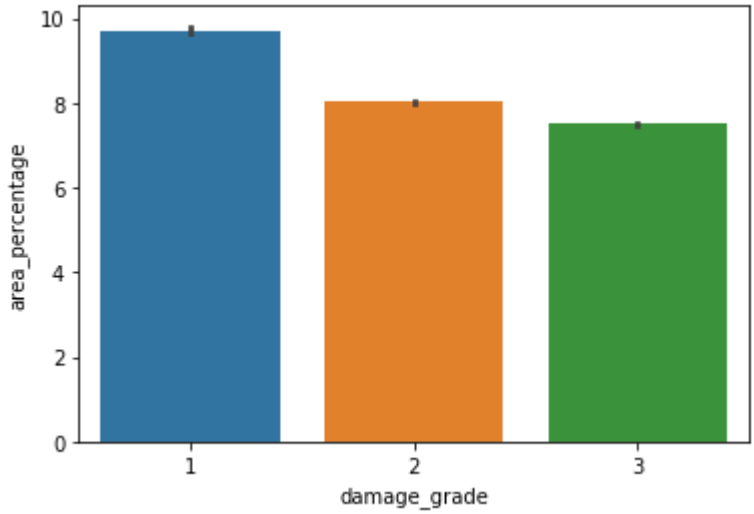
Out[69]: <matplotlib.axes._subplots.AxesSubplot at 0x225f100ae48>



From the above chart it infers higher the buildings it tends to get more damaged

In [70]:
```
sns.barplot(y = train_values["area_percentage"], x = train_labels["damage_grade"])
```

Out[70]: <matplotlib.axes._subplots.AxesSubplot at 0x225f1096c88>

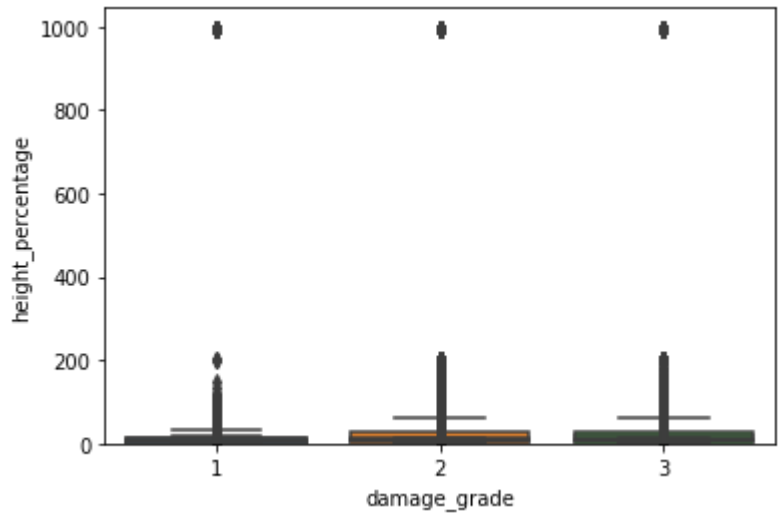


From the above chart it infers lesser the building area it tends to get more damaged

Visualize to see if any outliers in the dataset

In [57]:
```
sns.boxplot(x = train_labels['damage_grade'], y = train_values['age'])
sns.boxplot(x = train_labels['damage_grade'], y = train_values['count_floors_pre_eq'])
sns.boxplot(x = train_labels['damage_grade'], y = train_values['area_percentage'])
sns.boxplot(x = train_labels['damage_grade'], y = train_values['height_percentage'])
```

Out[57]: <matplotlib.axes._subplots.AxesSubplot at 0x225e8a25888>



Box plot infers that the data consists of many outliers