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

color = [
    '#ADCE74',
    '#A2738C',
    '#82DBD8',
    '#D3C09A',
    '#EC8F6A',
    '#6BBA62',
    '#F3D516',
    '#FFCB3C',
    '#FF677D',
    '#EADADA',
    '#999B84'
]

df = pd.read_csv('RealEstateAU_1000_Samples.csv')
df.head()

print(df.shape)
print(df.duplicated().sum())
tabela = pd.DataFrame({
    'Unique':df.nunique(),
    'Null':df.isna().sum(),
    'NullPercent':df.isna().sum() / len(df),
    'Types':df.dtypes.values
})
display(tabela)

df.head(2)

# eliminado colunas
df.drop(['index','latitude','longitude','state','location_type','location_name',
'address_1','breadcrumb','RunDate'],axis=1,inplace=True)

colunas = ['category_name','listing_agency','city']
for i in colunas:
    df[i] = df[i].str.upper()

colunas = ['address']
for col in colunas:
    df[col] = df[col].fillna('Unknow')

colunas = ['building_size','preferred_size','land_size']
for col in colunas:
    df[col] = df[col].str.replace('m²','').str.replace('ha','')

print('ok')

colunas = df['listing_agency'].str.split('-',1,expand=True)
df['agency_names'] = colunas[0]
df['agency_names2'] = colunas[1]
df['agency_names2'] = df['agency_names2'].str.replace(' ','')
df['agency_names2'] = df['agency_names2'].replace('   ','Unknown')
df['agency_names2'] = df['agency_names2'].replace( '    ','Unknown')
# espaçamento existente na linha
df['agency_names'] = df['agency_names'].replace( 'FOR SALE BY OWNER
','FOR SALE BY OWNER')


df['price'] = df['price'].replace('$1.15m','$1.150000')
```

```
df['price'] = df['price'].replace('JUST LIKE THAT: UNDER CONTRACT IN 7
DAYS','JUST LIKE THAT: UNDER CONTRACT IN SEVEN DAYS')
df['price'] = df['price'].replace('Offers over $1.2m','Offers over $1.200000')
df['price'] = df['price'].replace('Auction Wednesday 1st of June
2022','Auction')
df['price'] = df['price'].replace('Auction 8th June on site','Auction')
df['price'] = df['price'].replace('Auction - Wednesday 15th June 2022 at
5.30pm','Auction')
df['price'] = df['price'].replace('AUCTION: Saturday 4th Jun @11am On-
Site',"AUCTION")
df['price'] = df['price'].replace('JUST LIKE THAT: UNDER CONTRACT IN 5
DAYS','JUST LIKE THAT: UNDER CONTRACT IN FIVE DAYS')
df['price'] = df['price'].replace('Offers Over $500,000 - Offers by 6.30pm
22/6/22','Offers Over $500,000')
df['price'] = df['price'].replace('Negotiable Above 1.5M','Negotiable Above
1.500000')
df['price'] = df['price'].replace('$147 000','$147,000')
df['price'] = df['price'].replace('$369 000','$369,000')
df['price'] = df['price'].replace('$545,000 ( over 1000sqm of land)','$545,000 (
over thousand sqm of land)')
df['price'] = df['price'].replace('$450 000','$450,000')

df['property_type'].unique()

colunas = df['price'].str.split('-',1,expand=True)
df['price'] = colunas[0]
df['priceConsidered'] = colunas[1]

colunas = df['price'].str.split(' ',1,expand=True)
df['price'] = colunas[0]
df['priceConsidered'] = colunas[1]

# removendo números
df['priceCondition'] = df['priceConsidered'].replace(r'[0-9]','',regex=True)
df['priceCondition'] = df['priceCondition'].str.replace('$','',regex=True)
df['priceCondition'] =
df['priceCondition'].str.replace('!','',regex=True).str.replace(' ,','',regex=Tr
ue)
df['priceCondition'] = df['priceCondition'].str.upper()
# removendo letras
df['priceConsidered'] = df['priceConsidered'].replace('[^0-9]','',regex=True)

# removendo letras
df['price'] = df['price'].replace('[^0-9]','',regex=True)
# removendo símbolo
df['price'] = df['price'].replace('$','',regex=True).replace('-$','',regex=True)

df['price'] = df['price'].replace('',np.nan)
df['price'] = df['price'].fillna(df['priceConsidered'])
df['price'] = pd.to_numeric(df['price'], errors='coerce')

df['land_size'] = pd.to_numeric(df['land_size'], errors='coerce')
df['preferred_size'] = pd.to_numeric(df['preferred_size'], errors='coerce')
df['building_size'] = pd.to_numeric(df['building_size'], errors='coerce')

# new data
df =  df[(df['property_type'] != 'Lifestyle') & (df['property_type'] !=
'Warehouse')]

'''
not to exclude the missing data, as it could identify which of the columns had
more null values,
but other than that, I used Pandas' pivot_table to identify each attribute by
property type,
```

```
the index of the property_type column, is linked to the number of rooms existing
in them,
each column below represents the number of bathrooms, parking, land size and
building
'''
pd.pivot_table(df,index=['property_type','bedroom_count'],

values=['bathroom_count','parking_count','building_size','land_size','preferred_
size']).style.background_gradient(axis=0)

'''
here we have a table with the sum of bedrooms, bathrooms and parking in each
property by classification
Colors indicate the maximum number of attributes a property contains in the
data.
'''
pd.pivot_table(df,index=['product_depth','property_type'],

values=['bedroom_count','bathroom_count','parking_count'],aggfunc='sum').style.b
ackground_gradient(axis=0)

plt.figure(figsize=(14,7))
sns.histplot(x=df['price'])
plt.ticklabel_format(style='plain')


df['property_type'].value_counts().plot.pie(autopct='%.2f',radius=3,
textprops={'size':14},
                                            explode=(0,0,0,0,0,0,2,3,4,5,6),
colors=color)
plt.axis('off')
plt.show()

df['product_depth'].value_counts().plot.pie(autopct='%.2f', explode=(0,0,0,1),
textprops={'size':16}
                                            ,radius=3,colors=color)
plt.axis('off')
plt.show()


df['bedroom_count'].value_counts().plot.pie(autopct='%.2f',
explode=(0,0,0,0,0,0,1,2,3,4)
                                            ,radius=3,colors=color,
textprops={'size':16})
plt.axis('off')
plt.show()

df['bathroom_count'].value_counts().plot.pie(autopct='%.2f',radius=3,
explode=(0,0,0,0,1), colors=color, textprops={'size':16})
plt.axis('off')
plt.show()

df['parking_count'].value_counts().plot.pie(autopct='%.2f',radius=3,explode=(0,0
,0,0,0,0,0,0.1,1,2,3),
                                            colors=color, textprops={'size':16})
plt.axis('off')
plt.show()



plt.figure(figsize=(14,24))
ax = sns.barplot(y=df['property_type'],x=df['price'],
hue=df['product_depth'],ci=None, palette=color)
plt.yticks(fontsize=16)
```

```
plt.ylabel(None)
plt.legend(fontsize=14)
for i in ax.containers:
    ax.bar_label(i, fontsize=16,fmt='%d')
plt.ticklabel_format(style='plain',axis='x')
plt.show()


data = df['property_type'].unique()
plt.figure(figsize=(14,47))
for i,col in enumerate(data):
    ax = plt.subplot(13,1,i + 1)
    sns.lineplot(x='land_size',y='price', data=df[df['property_type']==col],
hue=df['product_depth'],ci=None)
    plt.title(f'{col}: Price | Product Depth', fontsize=14, fontweight='bold')
    plt.legend(fontsize=14,loc=2, bbox_to_anchor=(1.05,1))
    plt.ticklabel_format(style='plain')
plt.tight_layout()
plt.show()


colunas = ['land_size','preferred_size']
data = df['property_type'].unique()

plt.figure(figsize=(14,47))
for i,col in enumerate(data):
    for d in colunas:
        ax = plt.subplot(13,1,i + 1)
        sns.lineplot(x=d,y='price', data=df[df['property_type']==col],
label=f'{d}',ci=None)
    plt.title(f'{col}: Price | Property Type', fontsize=14, fontweight='bold')
    plt.xlabel(None)
    plt.legend(fontsize=14)
    plt.ticklabel_format(style='plain')
plt.tight_layout()
plt.show()


colunas = ['land_size','building_size']
data = df['property_type'].unique()

plt.figure(figsize=(14,47))
for i,col in enumerate(data):
    for d in colunas:
        ax = plt.subplot(13,1,i + 1)
        sns.lineplot(x=d,y='price', data=df[df['property_type']==col],
label=f'{d}',ci=None)
    plt.title(f'{col}: Price | Property Type',fontsize=14, fontweight='bold')
    plt.xlabel(None)
    plt.legend(fontsize=14)
    plt.ticklabel_format(style='plain')
plt.tight_layout()
plt.show()


plt.figure(figsize=(14,7))
sns.countplot(y=df['product_depth'],palette=color)
plt.ylabel(None)
plt.yticks(fontsize=14)
plt.show()


'''
average price per property depth
```

```
'''

plt.figure(figsize=(14,7))
ax = sns.barplot(y=df['product_depth'],x=df['price'],ci=None,palette=color)
plt.ylabel(None)
plt.yticks(fontsize=14)
plt.ticklabel_format(style='plain',axis='x')
for i in ax.containers:
    ax.bar_label(i, fontsize=14)


plt.figure(figsize=(14,7))
sns.countplot(y=df['property_type'],palette=color)
plt.yticks(fontsize=14)
plt.ylabel(None)
plt.show()


'''
average price by type of property
'''

plt.figure(figsize=(14,7))
ax = sns.barplot(y=df['property_type'],x=df['price'],ci=None,palette=color)
plt.yticks(fontsize=14)
plt.ylabel(None)
plt.ticklabel_format(style='plain',axis='x')
for i in ax.containers:
    ax.bar_label(i, fontsize=16,fmt='%d')


plt.figure(figsize=(14,20))
sns.countplot(y=df['city'],order=df['city'].value_counts().index,palette=color)
plt.yticks(fontsize=14)
plt.ylabel(None)
plt.show()


plt.figure(figsize=(14,20))
ax = sns.barplot(y=df['city'], x=df['price'], ci=None,palette=color)
plt.yticks(fontsize=14)
plt.ylabel(None)
for i in ax.containers:
    ax.bar_label(i,fontsize=16,fmt='%d')


df[df['price'] == 38900000.0].T


plt.figure(figsize=(14,20))
sns.countplot(y=df['agency_names'],order=df['agency_names'].value_counts().index
,palette=color)
plt.yticks(fontsize=12)
plt.ylabel(None)
plt.show()


plt.figure(figsize=(14,20))
sns.countplot(y=df['agency_names2'],
order=df['agency_names2'].value_counts().index,palette=color)
plt.yticks(fontsize=12)
plt.ylabel(None)
plt.show()
```

```
df.dropna(inplace=True)


# looking for location
from geopy.geocoders import Nominatim
from geopy.extra.rate_limiter import RateLimiter

geolocator = Nominatim(user_agent = 'Busca')

geocode = RateLimiter(geolocator.geocode,min_delay_seconds=1)

df['location'] = df['city'].apply(geocode)

df['lat'] = df['location'].apply(lambda loc: str(loc.latitude) if loc else None)

df['long'] = df['location'].apply(lambda loc: str(loc.longitude) if loc else
None)

df['lat'] = df['lat'].astype(float)
df['long'] = df['long'].astype(float)

import plotly.express as px
fig = px.scatter_mapbox(df, lat="lat", lon="long",color="price",
                  color_continuous_scale=px.colors.cyclical.IceFire,
size_max=15, zoom=10)
fig.show()
```