**COIMBATORE INSTITUTE OF TECHNOLOGY**

**2. ANALYSIS ON BRAZIL HOUSE RENT DATA TO PREDICT HOUSE RENT**

- Nandhakumar T (1832035)

**AIM:**

To predict the house rent of Brazil based on other components in the dataset.

**DESCRIPTION:**

Rent of a house increase or decrease depends on various factors like area, location ,facility, pet, safe and security, etc. But not all factors were responsible to affect the house rent. The project aimed to predict the house rent(Brazil) from the given data. In order to predict the output we have to determine the key factors that affects the house rent. By using such factors , Better results(Rent) can be predicted using it. For this problem Multiple Linear Regression is Best to predict house rent.

**CODE:**

**#IMPORTING PACKAGES**

import numpy as np

import pandas as pd

import seaborn as sns

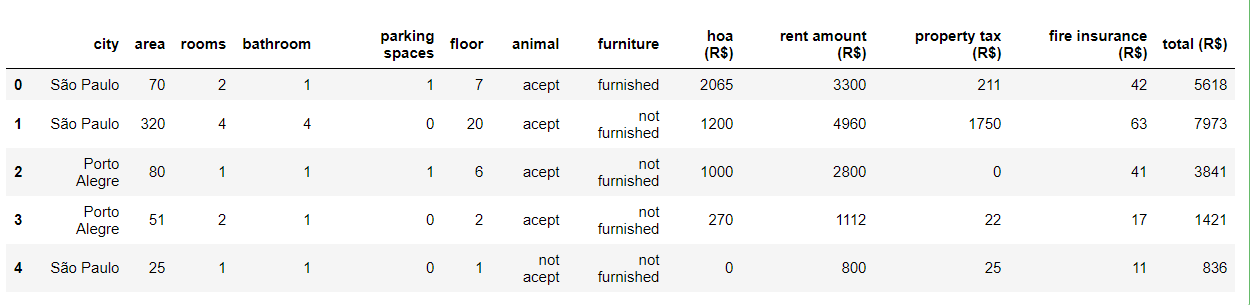
import matplotlib.pyplot as plt

**#READING DATA FROM LOCAL MACHINE AND STORE IT AS A DATAFRAME**

df = pd.read\_csv(r'C:/Users/THANGAVEL/Desktop/houses\_to\_rent.csv')

df.head()

**OUTPUT:**



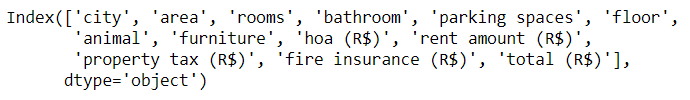
**#EXPLORATORY DATA ANALYSIS**

**#ALL COLUMNS IN DATASET**

all\_cols = df.columns

all\_cols

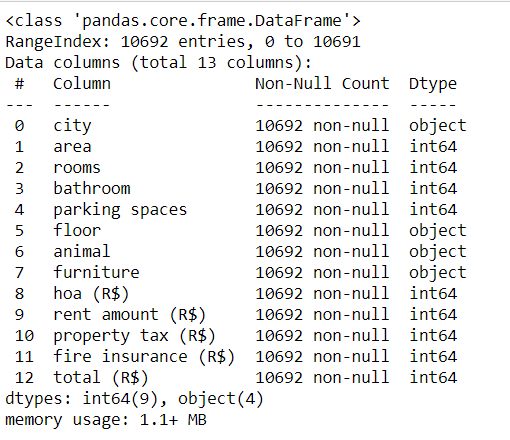
**OUTPUT:**



**#INFORMATION ABOUT THE DATA FRAME**

df.info()

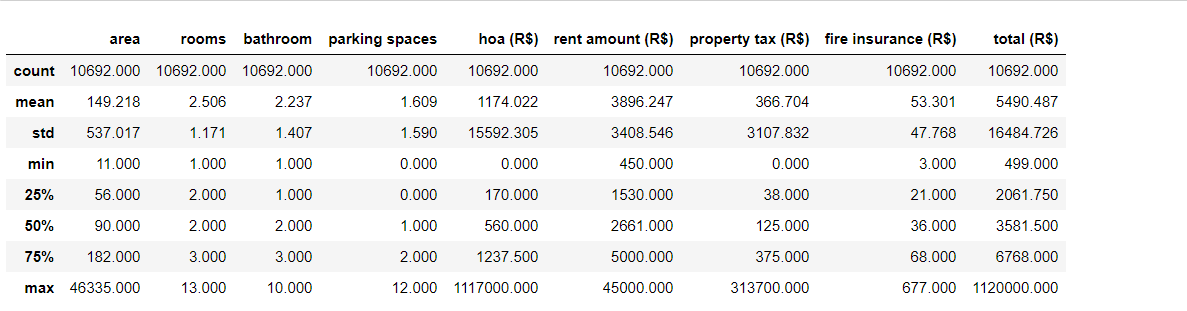
**OUTPUT:**



**#SUMMARY STATISTICS**

df.describe().round(3)

**OUTPUT:**



**#REPALCING MISSING VALUES**

cols = df.columns

cols = cols.map(lambda x: x.replace(' ','\_') if isinstance(x, (str)) else x)

df.columns = cols

**#CHANGE "$" FOR USE QUERIES**

df.rename(columns={'hoa\_(R$)' : 'hoa',

'rent\_amount\_(R$)' : 'rent\_amount',

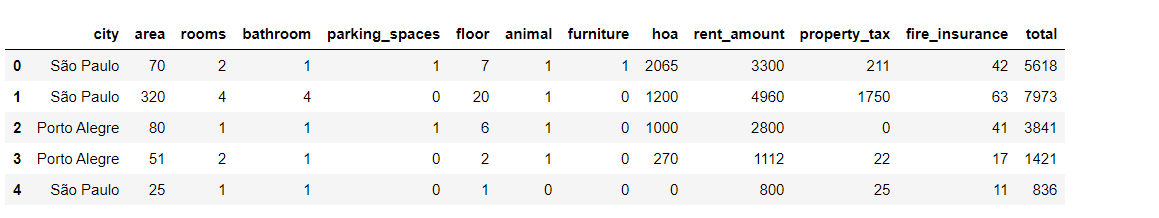
'property\_tax\_(R$)' : 'property\_tax',

'fire\_insurance\_(R$)' : 'fire\_insurance',

'total\_(R$)' : 'total'}, inplace = True)

df.head()

**OUTPUT:**



**#COUNT PLOT FOR FURNITURE**

fc = sns.countplot(df['furniture'], hue = df['city'])

fc.figure.set\_size\_inches(12, 8)

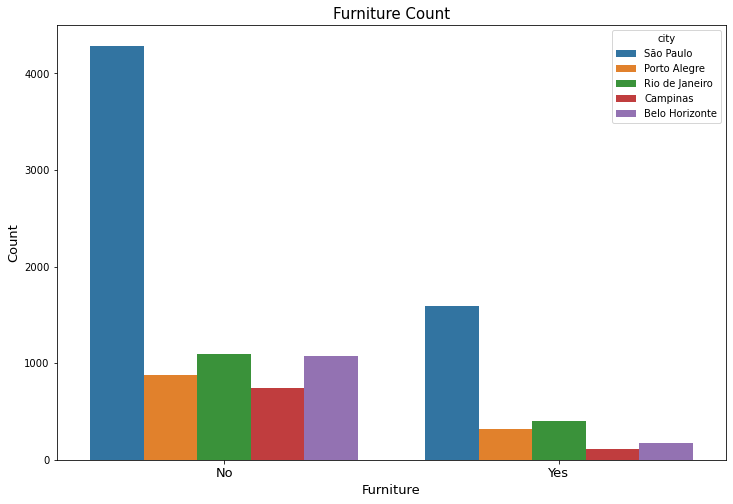
fc.set\_title('Furniture Count',fontsize=15)

fc.set\_xlabel('Furniture',fontsize=13)

fc.set\_ylabel('Count', fontsize=13)

fc.set\_xticklabels(['No','Yes'], fontsize=13)

**OUTPUT:**



**#BARPLOT FOR NUMBER OF ROOMS WITH SIZE OF AREA**

bs = sns.barplot(x='rooms', y='area', data = df, palette = 'dark')

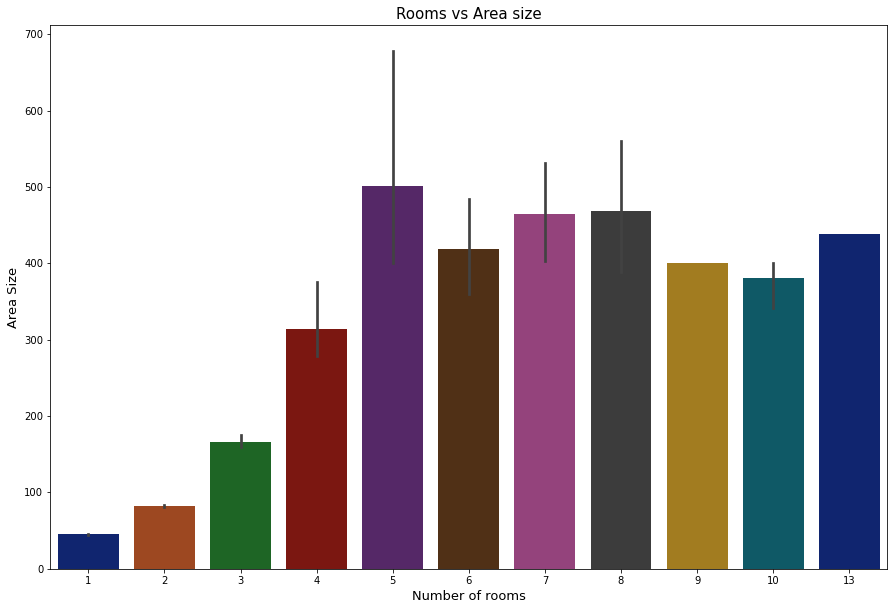
bs.figure.set\_size\_inches(15, 10)

bs.set\_title('Rooms vs Area size',fontsize=15)

bs.set\_xlabel('Number of rooms', fontsize=13)

bs.set\_ylabel('Area Size', fontsize=13)

**OUTPUT:**



**#SCATTER PLOT FOR TOTAL RENT VS HOA TAX**

df = df.drop(labels=df[(df['hoa'] > 300000)].index)

df = df.drop(labels=df[(df['total'] > 30000)].index)

th = sns.scatterplot(x = 'total', y = 'hoa', data = df)

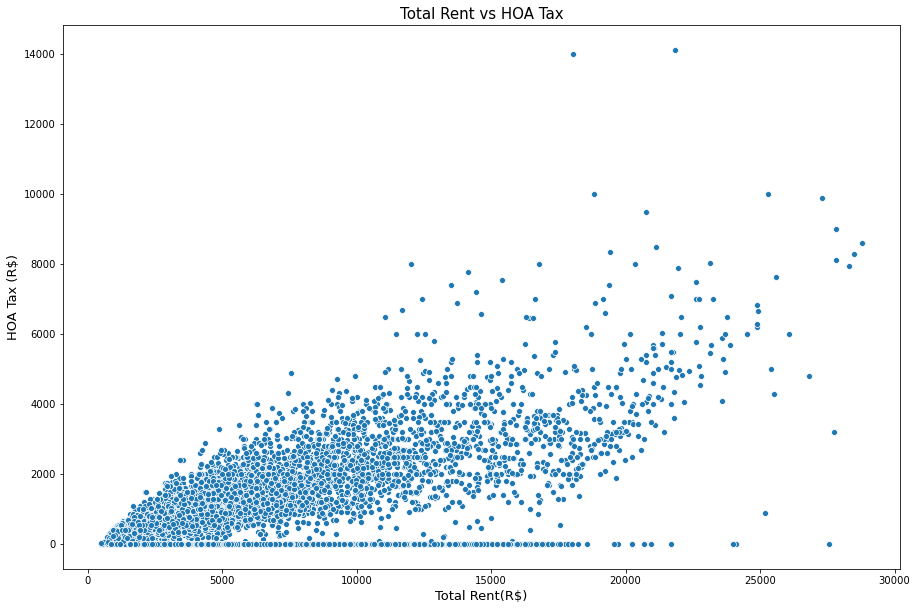
th.figure.set\_size\_inches(15, 10)

th.set\_title('Total Rent vs HOA Tax',fontsize=15)

th.set\_xlabel('Total Rent(R$)', fontsize=13)

th.set\_ylabel('HOA Tax (R$)', fontsize=13)

**OUTPUT:**



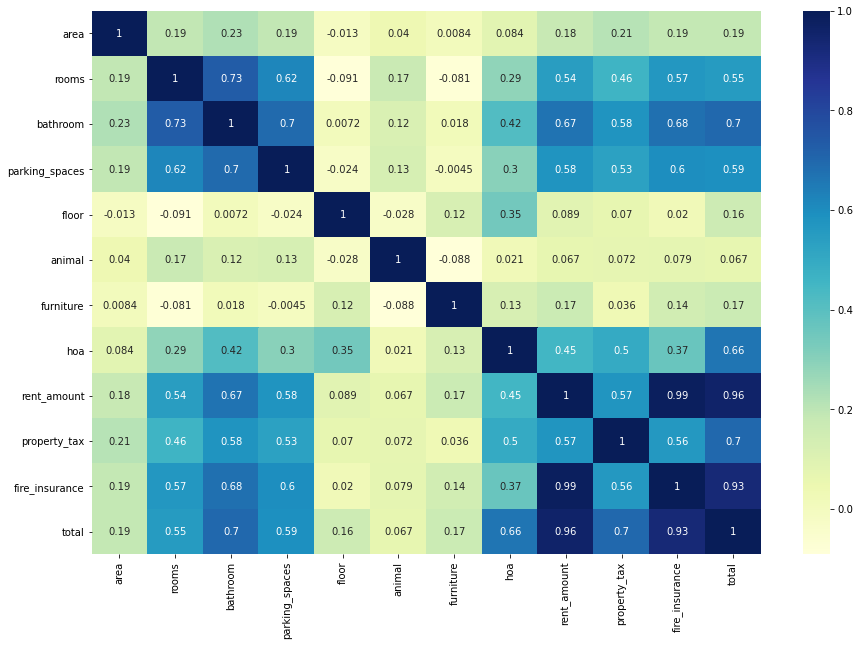
**#HEAT MAP TO FIND BETTER CORRELATED VALUES FOR TOTAL RENT**

cor = df.corr()

plt.figure(figsize=(15,10))

sns.heatmap(df.corr(), annot=True, cmap = 'YlGnBu')

**OUTPUT:**

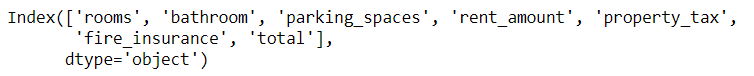


**#BATROOMS, HOA PROPERTY\_TAX, FIRE\_INSURANCE WERE MORE CORRELATED WITH RENT\_AMOUNT**

req\_cols = cor[cor.loc['rent\_amount']>0.5].T.columns

req\_cols

**OUTPUT:**



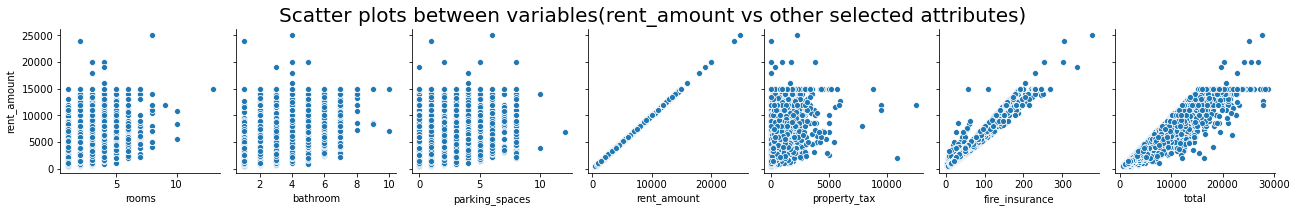
**#PAIRPLOT FOR RENT\_AMOUNT WITH REQ\_COLS**

ax = sns.pairplot(df, y\_vars='rent\_amount', x\_vars=req\_cols)

ax.fig.suptitle('Scatter plots between variables(rent\_amount vs other selected attributes)', fontsize=20, y=1.1)

ax

**OUTPUT:**



**#SELECTING X AND Y VALUES**

metrics = []

y = df['rent\_amount']

x = df[req\_cols]

**#IMPORTING PACKAGES FOR MODELS, SPLIT AND ACCURACY SCORES**

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

**#SPLITTING INTO TRAINING AND TEST DATA USIG TRAIN\_TEST\_SPLIT**

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.3, random\_state = 8)

**#FITTING AND TRAINING MODEL**

lr = LinearRegression()

lr.fit(x\_train, y\_train)

predict = lr.predict(x\_test)

**#ADD A CONSTANT AND LOOKING THE SUMMARY**

import statsmodels.api as sm

x\_train\_constant = sm.add\_constant(x\_train)

model\_sm = sm.OLS(y\_train, x\_train\_constant, hascont = True).fit()

print(model\_sm.summary())

#looking the metrics

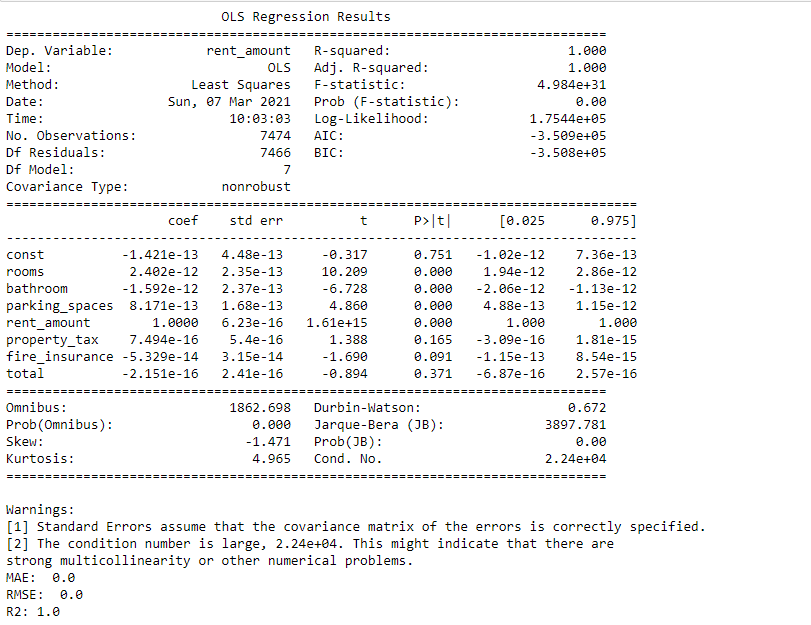
print('MAE: ', mean\_absolute\_error(y\_test, predict).round(3))

print('RMSE: ', np.sqrt(mean\_squared\_error(y\_test, predict)).round(3))

print('R2:', r2\_score(y\_test, predict).round(3))

metrics.append(np.sqrt(mean\_squared\_error(y\_test, predict)))

**OUTPUT:**



**INFERENCE:**

The test results shows that the r-square values(R2) is 1.00 with 0 RMSE and )MAE, which means the model is good enough to predict the house rent.

**RESULT:**

The value of r2(r-square) resembles 100% of accuracy score, which means the model is perfectly ready to predict the output(House rent). From my observation bathrooms, HOA property tax, fire insurance are the major factors that affects the house rent in Brazil. If this factors are high the rent of the house will be high, Otherwise the rent will decrease depends on the factors variation. In this Data Science Project I used Data Viz with seaborn and matplotlib to improve the understanding of some variables from the data, pandas to handle and analyzing some columns, and using statsmodels to have a statistical comprehesion of the dataset and verifying if i was can use the Linear Regression model. In addition, I put the data on a logarithmic scale, use the StandardScaler and I tried to group some data to reduce the multicollinearity and clear the data. Of course, I apply some sklearn modules to splitting the dataset, do prediction and know how efficient my model is.We were able to observe in the comparison of the results that the second attempt was the one with the best RMSE, but there was multicollinearity in the data and linear regression assumes no multicollinearity, so I had to change something in the model