**Predicting the Apartment Rent**

1. **Business problem/ Discovery Phase:**

This dataset comprises detailed information on apartment rentals, ideal for various machine learning tasks including clustering, classification, and regression. It features a comprehensive set of attributes that capture essential aspects of rental listings, such as:

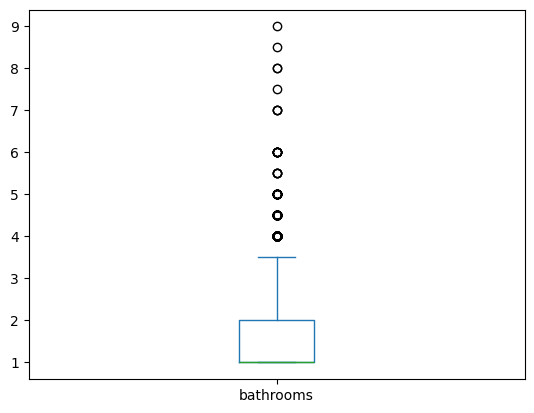
Identifiers & Location: Includes unique identifiers (id), geographic details (address, cityname, state, latitude, longitude), and the source of the classified listing.  
Property Details: Provides information on the apartment's category, title, body, amenities, number of bathrooms, bedrooms, and square\_feet (size of the apartment).  
Pricing Information: Contains multiple features related to pricing, including price (rental price), price\_display (displayed price), price\_type (price in USD), and fee.  
Additional Features: Indicates whether the apartment has a photo (has\_photo), whether pets are allowed (pets\_allowed), and other relevant details such as currency and time of listing creation.  
The dataset is well-cleaned, ensuring that critical columns like price and square feet are never empty. This makes it a robust resource for developing predictive models and performing in-depth analyses on rental trends and property characteristics.

1. **Data Collection:** Dataset taken from Kaggle.
2. **Data Cleaning and Preprocessing:**
3. Null Values:

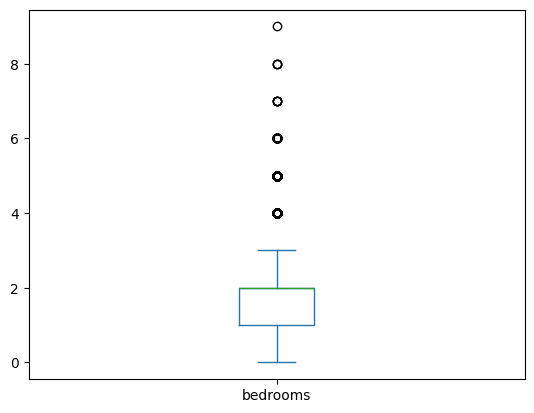
|  |
| --- |
| id int64 0 This column can also be dropped as it is just and identifier |
| category object 0 |
| title object 0 |
| body object 0 |
| amenities object 16044 -> Can be Imputed with Mode as it is a categorical data |
| bathrooms float64 63 -> Use Median to fill the Null values as outliers are present as can be seen in the box plots |
| bedrooms float64 124 -> Use Median to fill the Null values as outliers are present as can be seen in the box plots |
| currency object 0 |
| fee object 0 |
| has\_photo object 0 |
| pets\_allowed object 60424 -> This column can be dropped as more then 50% data is Null (Else can be imputed with Mode values as this is a categorical data |
| price float64 1 -> Contains outliers (Use Median to fill all null values) |
| price\_display object 1 Same as price column |
| price\_type object 0 |
| square\_feet int64 0 |
| address object 91549 -> This column can be dropped as more than 50% data is N/A and we cannot infer any information from this column. |
| cityname object 302 -> Can be filled with median values |
| state object 302 -> Can be filled with median values |
| latitude float64 25 -> Contains outliers (Use Median to fill all null values), this column can be dropped as it does not add much value |
| longitude float64 25 -> Contains outliers (Use Median to fill all null values), this column can be dropped as it does not add much value |
| source object 0 |
| time int64 0 |
|  |

Since. Bathroom, bedroom, price, latitude and longitude were numberical data we had to check for presence outliers using box plots.

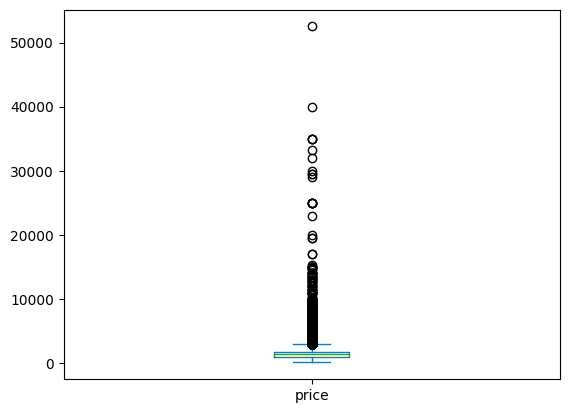
df['bathrooms'].plot(kind='box')



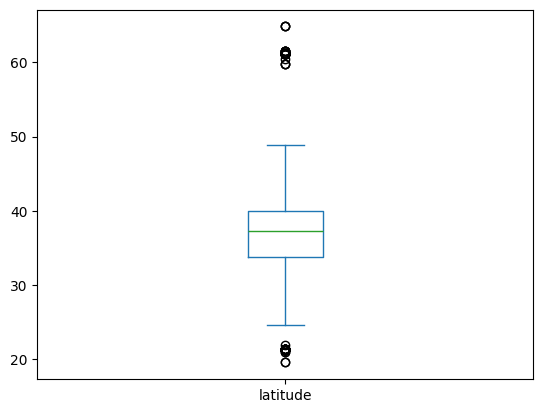
df['bedrooms'].plot(kind='box')



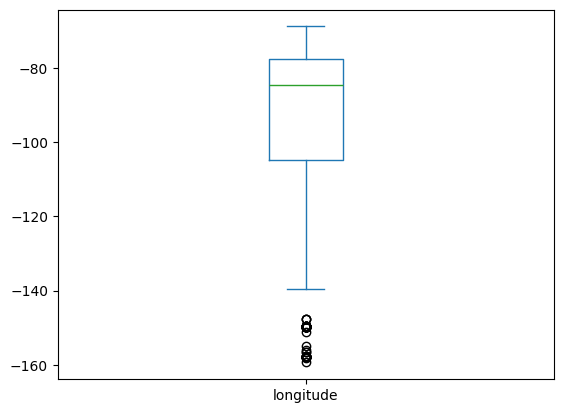
df['price'].plot(kind='box')



df['latitude'].plot(kind**=**'box')



df['longitude'].plot(kind**=**'box')



1. Duplicate Values

df.duplicated().sum() = np.int64(84) - > Drop these duplicate values

1. Unique Values:

id 99408

category 7

title 58503

body 94503

amenities 9827

bathrooms 16

bedrooms 10

currency 1

fee 2

has\_photo 3

pets\_allowed 4

price 3687

price\_display 3718

price\_type 3

square\_feet 2538

address 7771

cityname 2979

state 51

latitude 7212

longitude 7270

source 25

time 75360

1. **Exploratory Data Analysis**

Univariate Analysis of all Categorical Data:

The unique values from category are ['housing/rent/apartment', 'housing/rent/home', 'housing/rent/short\_term', 'housing/rent', 'housing/rent/condo', 'housing/rent/other', 'housing/rent/commercial/retail']

category

housing/rent/apartment 99347

housing/rent/commercial/retail 42

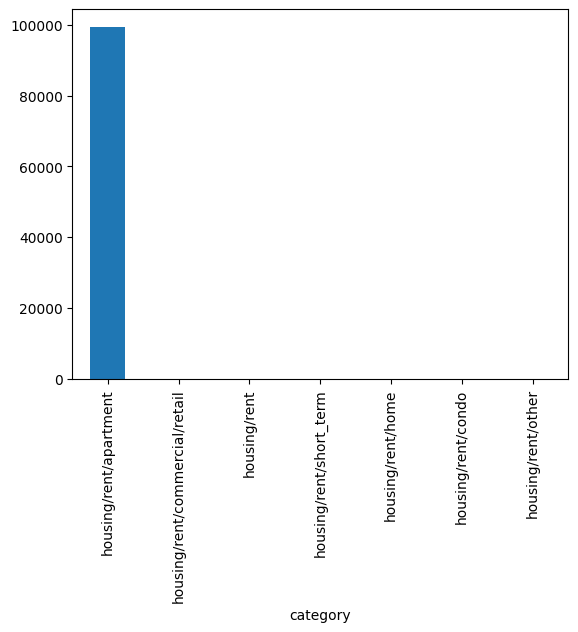
housing/rent 7

housing/rent/short\_term 4

housing/rent/home 4

housing/rent/condo 3

housing/rent/other 1

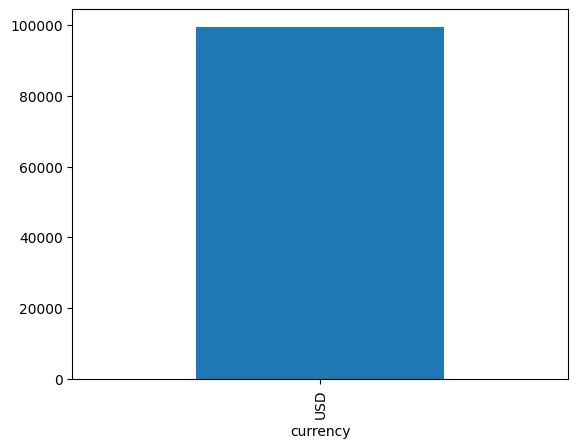


The unique values from currency are ['USD']

currency

USD 99408

Name: count, dtype: int64



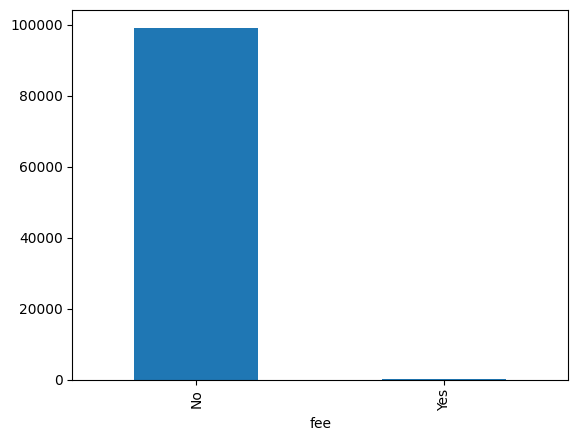
The unique values from fee are ['No' 'Yes']

fee

No 99207

Yes 201

Name: count, dtype: int64



The unique values from has\_photo are ['Thumbnail' 'No' 'Yes']

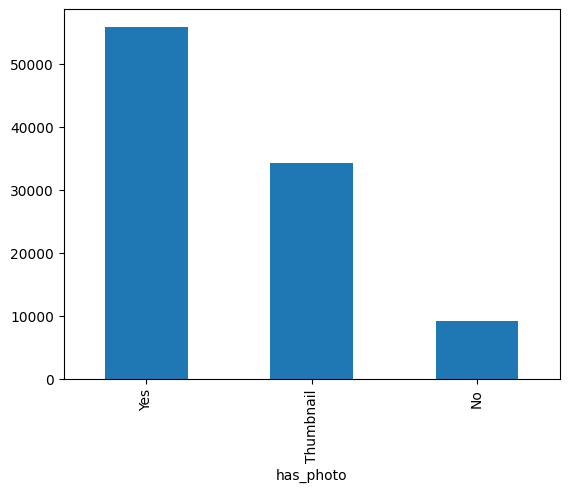
has\_photo

Yes 55908

Thumbnail 34302

No 9198

Name: count, dtype: int64



The unique values from pets\_allowed are ['Cats' 'Cats,Dogs' 'Dogs' 'Cats,Dogs,None']

pets\_allowed

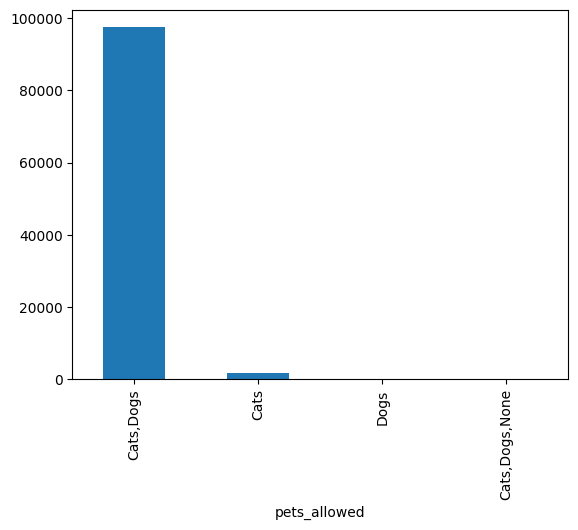
Cats,Dogs 97440

Cats 1840

Dogs 127

Cats,Dogs,None 1

Name: count, dtype: int64



The unique values from price type are ['Monthly' 'Weekly' 'Monthly|Weekly']

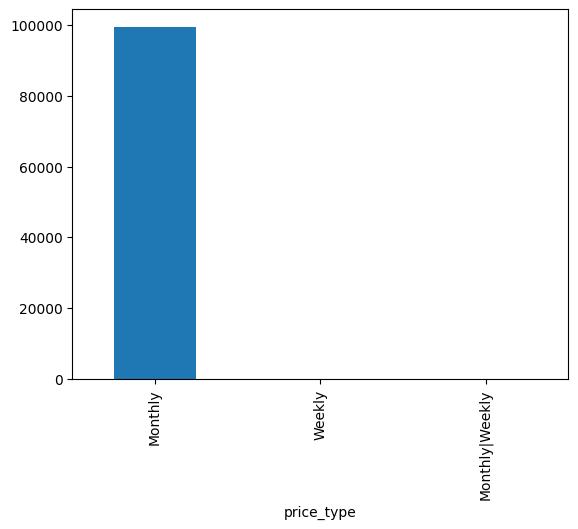
price\_type

Monthly 99404

Weekly 3

Monthly|Weekly 1

Name: count, dtype: int64



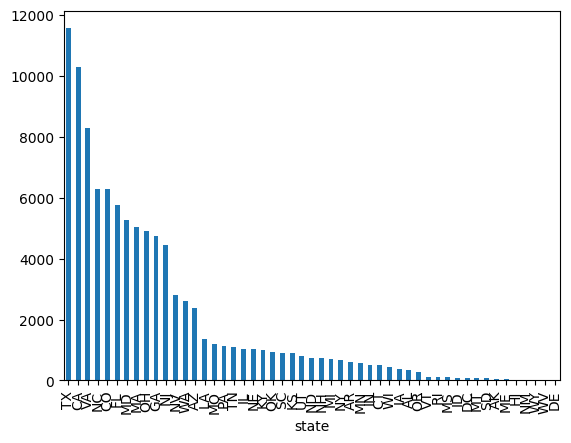
The unique values from state are ['CA' 'VA' 'NC' 'NM' 'CO' 'WV' 'GA' 'MA' 'DC' 'AZ' 'IA' 'WA' 'TX' 'IL'

'MS' 'OR' 'FL' 'MO' 'PA' 'WI' 'OK' 'UT' 'RI' 'NJ' 'IN' 'MD' 'OH' 'TN'

'ND' 'NE' 'AR' 'MI' 'MN' 'HI' 'ID' 'SC' 'KS' 'AL' 'SD' 'NY' 'KY' 'LA'

'AK' 'CT' 'NV' 'WY' 'VT' 'NH' 'MT' 'DE' 'ME']

state



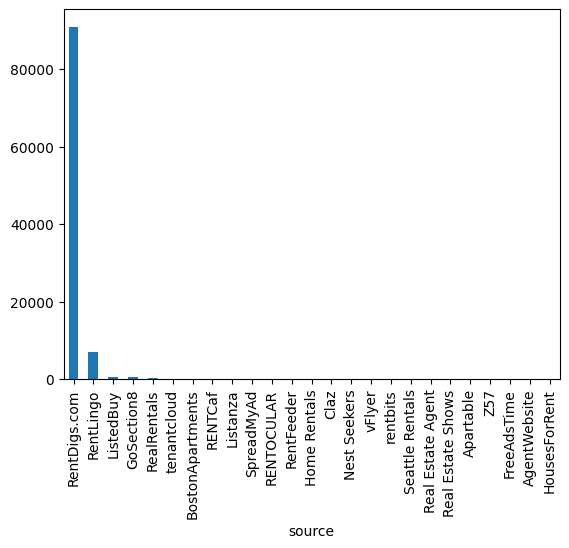
The unique values from source are ['RentLingo' 'ListedBuy' 'RentDigs.com' 'RENTCaf' 'GoSection8' 'Listanza'

'RealRentals' 'RENTOCULAR' 'tenantcloud' 'Real Estate Agent' 'rentbits'

'Home Rentals' 'Nest Seekers' 'RentFeeder' 'vFlyer' 'Claz'

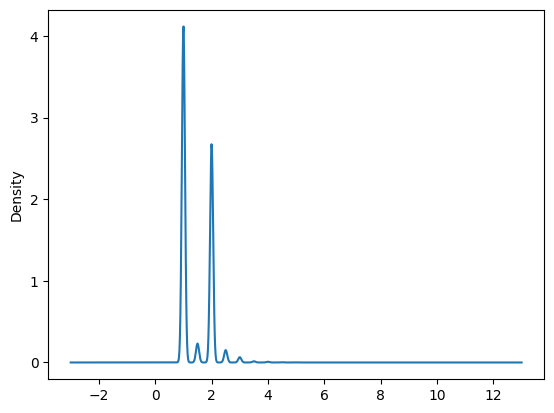
'Real Estate Shows' 'Seattle Rentals' 'BostonApartments' 'SpreadMyAd'

'Apartable' 'Z57' 'FreeAdsTime' 'AgentWebsite' 'HousesForRent']

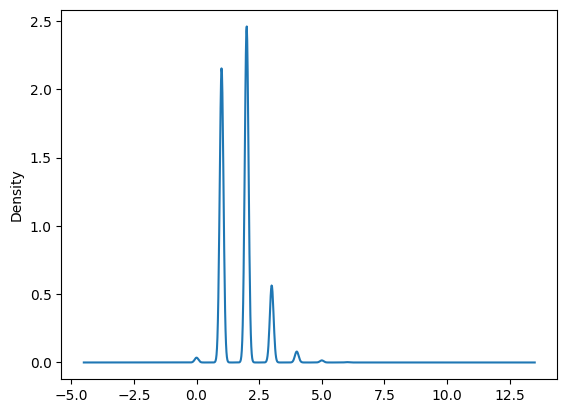


Univariate Analysis of all Numeical Data:

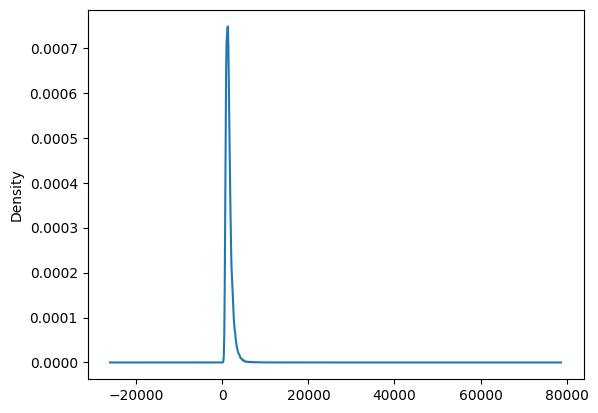
The kde values from bathrooms



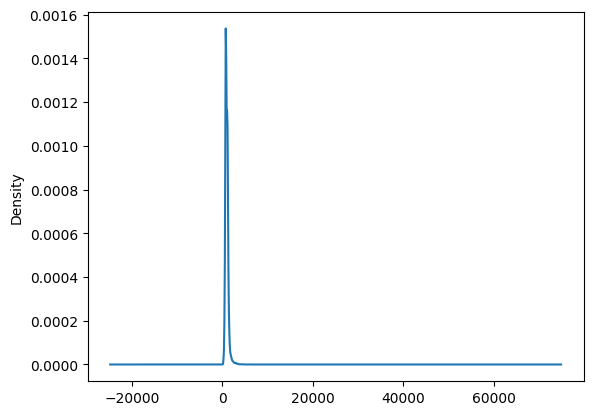
The kde values from bedrooms



The kde values from price



The kde values from square\_feet



Bivariate Analysis: See supporting document

1. **Feature Engineering:**

In this step, the required features are selected for model building using some feature selection techniques.

“Curse of Dimensionality” -> Reduction in number of features based on the number of unique values in each column, as it can take up too much space and crash out the code.

Also, as the performance increases the model performance may degrade.

Columns like ‘amenities’, ‘address’ and ‘cityname’ cannot be considered while performing the Univariate and Bivariate analysis as it leads to the code crashing.

We need to drop columns with large number of Unique values. i.e. ‘time’, ‘id’, ‘title’, ‘body’.

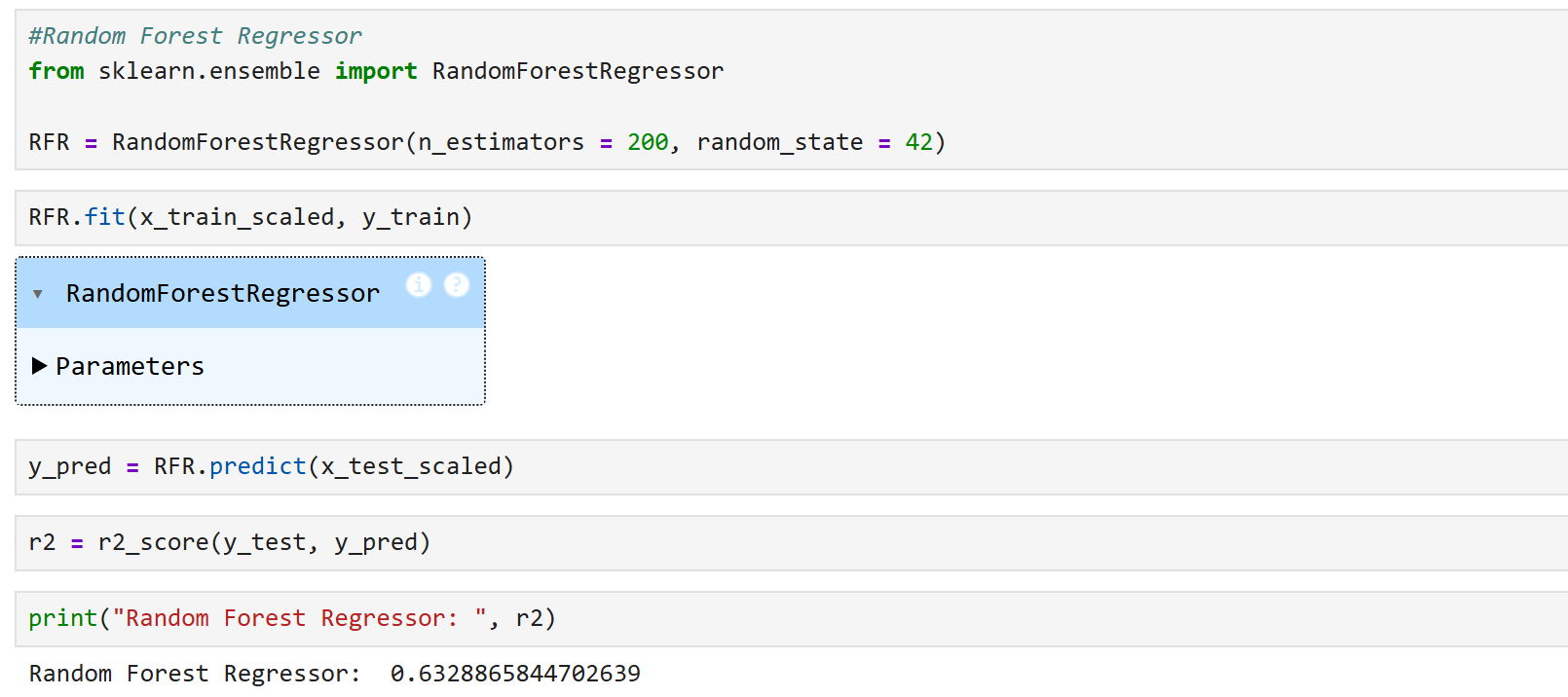
In feature Engineering we also perform Encoding to convert the Categorical data into numerical format for the system to understand and further we also perform data Standardization using StandardScalar or MinMaxScalar depending on the distribution of the data provided.

We apply standardization in ML models because the machine by default gives a higher priority to bigger numbers and it may not be necessary all the time, therefore standardization is important as it brings all data to a normal standard format, which helps the model do its job.

1. **Model Selection:** Since Model selection depends on the business problem on hand. Since we have to predict the apartment rent, it is a regression problem. We have to try using all the regression models in order to find the best performing model.

Using the Train-Test Split method

The best performing model was the Random Forest Regressor – 0.633



Cross Validation using K-folds

The best performing model was Random Forest Regressor

