{

"cells": [

{

"cell\_type": "raw",

"metadata": {},

"source": [

"Renting out their unused room is the basic concept behind the Airbnb with that the traveller can get cheap accomodation and owner of the house can moentize their unused rooms. It is gaining momentum and Airbnb wants to gain more market share by showing to the prospective owners that how can much they can earn from their property.\n",

"\n",

"For that we need to create a good analytics solution that they can show the earning potential to the prospective customers."

]

},

{

"cell\_type": "raw",

"metadata": {},

"source": [

"Definition of variables\n",

"\n",

"room\_id: A unique number identifying an Airbnb listing. The listing has a URL on the Airbnb web site of http://airbnb.com/rooms/room\_id\n",

"\n",

"host\_id: A unique number identifying an Airbnb host. The hostâ€™s page has a URL on the Airbnb web site of http://airbnb.com/users/show/host\_id\n",

"\n",

"room\_type: One of â€œEntire home/aptâ€, â€œPrivate roomâ€, or â€œShared roomâ€\n",

"\n",

"borough: A subregion of the city or search area for which the survey is carried out. The borough is taken from a shapefile of the city that is obtained independently of the Airbnb web site. For some cities, there is no borough information; for others the borough may be a number. If you have better shapefiles for a city of interest, please send them to me.\n",

"\n",

"neighborhood: As with borough: a subregion of the city or search area for which the survey is carried out. For cities that have both, a neighbourhood is smaller than a borough. For some cities there is no neighbourhood information.\n",

"\n",

"reviews: The number of reviews that a listing has received. Airbnb has said that 70% of visits end up with a review, so the number of reviews can be used to estimate the number of visits. Note that such an estimate will not be reliable for an individual listing (especially as reviews occasionally vanish from the site), but over a city as a whole it should be a useful metric of traffic.\n",

"\n",

"overall\_satisfaction: The average rating (out of five) that the listing has received from those visitors who left a review.\n",

"accommodates: The number of guests a listing can accommodate.\n",

"\n",

"bedrooms: The number of bedrooms a listing offers.\n",

"\n",

"price: The price (in $US) for a night stay. In early surveys, there may be some values that were recorded by month.\n",

"\n",

"minstay: The minimum stay for a visit, as posted by the host.\n",

"\n",

"latitude and longitude: The latitude and longitude of the listing as posted on the Airbnb site: this may be off by a few \n",

"hundred metres. I do not have a way to track individual listing locations with\n",

"\n",

"last\_modified: the date and time that the values were read from the Airbnb web site."

]

},

{

"cell\_type": "code",

"execution\_count": 3,

"metadata": {},

"outputs": [],

"source": [

"import pandas as pd\n",

"import matplotlib.pyplot as plt\n",

"%matplotlib inline"

]

},

{

"cell\_type": "code",

"execution\_count": 4,

"metadata": {},

"outputs": [],

"source": [

"airbnb = pd.read\_csv('2017\_07\_22.csv')"

]

},

{

"cell\_type": "code",

"execution\_count": 5,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>room\_type</th>\n",

" <th>country</th>\n",

" <th>city</th>\n",

" <th>borough</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>name</th>\n",

" <th>last\_modified</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" <th>location</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>10176931</td>\n",

" <td>1476</td>\n",

" <td>49180562</td>\n",

" <td>Shared room</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>De Pijp / Rivierenbuurt</td>\n",

" <td>7</td>\n",

" <td>4.5</td>\n",

" <td>2</td>\n",

" <td>1.0</td>\n",

" <td>NaN</td>\n",

" <td>156.0</td>\n",

" <td>NaN</td>\n",

" <td>Red Light/ Canal view apartment (Shared)</td>\n",

" <td>2017-07-23 13:06:27.391699</td>\n",

" <td>52.356209</td>\n",

" <td>4.887491</td>\n",

" <td>0101000020E610000033FAD170CA8C13403BC5AA41982D...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>8935871</td>\n",

" <td>1476</td>\n",

" <td>46718394</td>\n",

" <td>Shared room</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Centrum West</td>\n",

" <td>45</td>\n",

" <td>4.5</td>\n",

" <td>4</td>\n",

" <td>1.0</td>\n",

" <td>NaN</td>\n",

" <td>126.0</td>\n",

" <td>NaN</td>\n",

" <td>Sunny and Cozy Living room in quite neighbours</td>\n",

" <td>2017-07-23 13:06:23.607187</td>\n",

" <td>52.378518</td>\n",

" <td>4.896120</td>\n",

" <td>0101000020E6100000842A357BA095134042791F477330...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>14011697</td>\n",

" <td>1476</td>\n",

" <td>10346595</td>\n",

" <td>Shared room</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Watergraafsmeer</td>\n",

" <td>1</td>\n",

" <td>0.0</td>\n",

" <td>3</td>\n",

" <td>1.0</td>\n",

" <td>NaN</td>\n",

" <td>132.0</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>2017-07-23 13:06:23.603546</td>\n",

" <td>52.338811</td>\n",

" <td>4.943592</td>\n",

" <td>0101000020E6100000A51133FB3CC613403543AA285E2B...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>6137978</td>\n",

" <td>1476</td>\n",

" <td>8685430</td>\n",

" <td>Shared room</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Centrum West</td>\n",

" <td>7</td>\n",

" <td>5.0</td>\n",

" <td>4</td>\n",

" <td>1.0</td>\n",

" <td>NaN</td>\n",

" <td>121.0</td>\n",

" <td>NaN</td>\n",

" <td>Canal boat RIDE in Amsterdam</td>\n",

" <td>2017-07-23 13:06:22.689787</td>\n",

" <td>52.376319</td>\n",

" <td>4.890028</td>\n",

" <td>0101000020E6100000DF180280638F134085EE92382B30...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>18630616</td>\n",

" <td>1476</td>\n",

" <td>70191803</td>\n",

" <td>Shared room</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>De Baarsjes / Oud West</td>\n",

" <td>1</td>\n",

" <td>0.0</td>\n",

" <td>2</td>\n",

" <td>1.0</td>\n",

" <td>NaN</td>\n",

" <td>93.0</td>\n",

" <td>NaN</td>\n",

" <td>One room for rent in a three room appartment</td>\n",

" <td>2017-07-23 13:06:19.681469</td>\n",

" <td>52.370384</td>\n",

" <td>4.852873</td>\n",

" <td>0101000020E6100000CD902A8A57691340187B2FBE682F...</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id room\_type country city borough \\\n",

"0 10176931 1476 49180562 Shared room NaN Amsterdam NaN \n",

"1 8935871 1476 46718394 Shared room NaN Amsterdam NaN \n",

"2 14011697 1476 10346595 Shared room NaN Amsterdam NaN \n",

"3 6137978 1476 8685430 Shared room NaN Amsterdam NaN \n",

"4 18630616 1476 70191803 Shared room NaN Amsterdam NaN \n",

"\n",

" neighborhood reviews overall\_satisfaction accommodates \\\n",

"0 De Pijp / Rivierenbuurt 7 4.5 2 \n",

"1 Centrum West 45 4.5 4 \n",

"2 Watergraafsmeer 1 0.0 3 \n",

"3 Centrum West 7 5.0 4 \n",

"4 De Baarsjes / Oud West 1 0.0 2 \n",

"\n",

" bedrooms bathrooms price minstay \\\n",

"0 1.0 NaN 156.0 NaN \n",

"1 1.0 NaN 126.0 NaN \n",

"2 1.0 NaN 132.0 NaN \n",

"3 1.0 NaN 121.0 NaN \n",

"4 1.0 NaN 93.0 NaN \n",

"\n",

" name last\_modified \\\n",

"0 Red Light/ Canal view apartment (Shared) 2017-07-23 13:06:27.391699 \n",

"1 Sunny and Cozy Living room in quite neighbours 2017-07-23 13:06:23.607187 \n",

"2 Amsterdam 2017-07-23 13:06:23.603546 \n",

"3 Canal boat RIDE in Amsterdam 2017-07-23 13:06:22.689787 \n",

"4 One room for rent in a three room appartment 2017-07-23 13:06:19.681469 \n",

"\n",

" latitude longitude location \n",

"0 52.356209 4.887491 0101000020E610000033FAD170CA8C13403BC5AA41982D... \n",

"1 52.378518 4.896120 0101000020E6100000842A357BA095134042791F477330... \n",

"2 52.338811 4.943592 0101000020E6100000A51133FB3CC613403543AA285E2B... \n",

"3 52.376319 4.890028 0101000020E6100000DF180280638F134085EE92382B30... \n",

"4 52.370384 4.852873 0101000020E6100000CD902A8A57691340187B2FBE682F... "

]

},

"execution\_count": 5,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.head()"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Entire home/apt 14978\n",

"Private room 3682\n",

"Shared room 63\n",

"Name: room\_type, dtype: int64"

]

},

"execution\_count": 6,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.room\_type.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 7,

"metadata": {},

"outputs": [],

"source": [

"#Dropping Shared rooms because of their low volume\n",

"airbnb = airbnb[airbnb.room\_type != 'Shared room']"

]

},

{

"cell\_type": "code",

"execution\_count": 8,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Entire home/apt 14978\n",

"Private room 3682\n",

"Name: room\_type, dtype: int64"

]

},

"execution\_count": 8,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"# Shared Room is dropped\n",

"airbnb.room\_type.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 9,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"De Baarsjes / Oud West 3280\n",

"De Pijp / Rivierenbuurt 2371\n",

"Centrum West 2217\n",

"Centrum Oost 1724\n",

"Westerpark 1424\n",

"Noord-West / Noord-Midden 1416\n",

"Oud Oost 1167\n",

"Bos en Lommer 985\n",

"Oostelijk Havengebied / Indische Buurt 916\n",

"Watergraafsmeer 515\n",

"Oud Noord 492\n",

"Ijburg / Eiland Zeeburg 378\n",

"Slotervaart 346\n",

"Buitenveldert / Zuidas 248\n",

"Noord West 241\n",

"Noord Oost 221\n",

"Geuzenveld / Slotermeer 192\n",

"Osdorp 161\n",

"De Aker / Nieuw Sloten 114\n",

"Bijlmer Centrum 98\n",

"Bijlmer Oost 97\n",

"Gaasperdam / Driemond 42\n",

"Westpoort 15\n",

"Name: neighborhood, dtype: int64"

]

},

"execution\_count": 9,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"# It shows where the most popular neighborhood also it is saying near which neighborhood most of the rooms are situated\n",

"# We can do analysis like the most preferred neighborhood or near which neighborhood most of the visitor coming\n",

"# or whether having a particular type of neighborhood helps you get better rent or reviews or rating etc.\n",

"airbnb.neighborhood.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 10,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"0 2975\n",

"1 1500\n",

"2 1238\n",

"3 1100\n",

"4 924\n",

"5 874\n",

"6 737\n",

"7 680\n",

"8 589\n",

"9 527\n",

"11 497\n",

"10 474\n",

"12 395\n",

"14 362\n",

"13 340\n",

"15 308\n",

"16 300\n",

"17 261\n",

"18 236\n",

"20 226\n",

"19 213\n",

"21 189\n",

"22 166\n",

"23 158\n",

"24 153\n",

"27 137\n",

"25 133\n",

"26 125\n",

"29 112\n",

"28 112\n",

" ... \n",

"334 1\n",

"191 1\n",

"223 1\n",

"303 1\n",

"344 1\n",

"279 1\n",

"247 1\n",

"231 1\n",

"290 1\n",

"354 1\n",

"179 1\n",

"211 1\n",

"323 1\n",

"180 1\n",

"212 1\n",

"228 1\n",

"276 1\n",

"292 1\n",

"452 1\n",

"532 1\n",

"277 1\n",

"309 1\n",

"405 1\n",

"166 1\n",

"230 1\n",

"310 1\n",

"390 1\n",

"447 1\n",

"215 1\n",

"183 1\n",

"Name: reviews, Length: 283, dtype: int64"

]

},

"execution\_count": 10,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.reviews.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 11,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 7697\n",

"0.0 5721\n",

"4.5 4543\n",

"4.0 573\n",

"3.5 105\n",

"3.0 19\n",

"2.5 1\n",

"1.0 1\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 11,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 12,

"metadata": {},

"outputs": [],

"source": [

"#Finding those properties where overall\_satisfaction is zero and exploring them"

]

},

{

"cell\_type": "code",

"execution\_count": 13,

"metadata": {},

"outputs": [],

"source": [

"airbnb\_0\_satisfaction = airbnb[airbnb.overall\_satisfaction == 0]"

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>country</th>\n",

" <th>borough</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>count</th>\n",

" <td>5.721000e+03</td>\n",

" <td>5721.0</td>\n",

" <td>5.721000e+03</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>5721.000000</td>\n",

" <td>5721.0</td>\n",

" <td>5721.000000</td>\n",

" <td>5721.000000</td>\n",

" <td>0.0</td>\n",

" <td>5721.000000</td>\n",

" <td>0.0</td>\n",

" <td>5721.000000</td>\n",

" <td>5721.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>mean</th>\n",

" <td>1.472219e+07</td>\n",

" <td>1476.0</td>\n",

" <td>4.674155e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.700926</td>\n",

" <td>0.0</td>\n",

" <td>3.024821</td>\n",

" <td>1.554623</td>\n",

" <td>NaN</td>\n",

" <td>175.620696</td>\n",

" <td>NaN</td>\n",

" <td>52.364361</td>\n",

" <td>4.889534</td>\n",

" </tr>\n",

" <tr>\n",

" <th>std</th>\n",

" <td>5.209774e+06</td>\n",

" <td>0.0</td>\n",

" <td>4.424869e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.814133</td>\n",

" <td>0.0</td>\n",

" <td>1.352056</td>\n",

" <td>0.949345</td>\n",

" <td>NaN</td>\n",

" <td>140.020220</td>\n",

" <td>NaN</td>\n",

" <td>0.015795</td>\n",

" <td>0.036742</td>\n",

" </tr>\n",

" <tr>\n",

" <th>min</th>\n",

" <td>2.542800e+04</td>\n",

" <td>1476.0</td>\n",

" <td>2.234000e+03</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.000000</td>\n",

" <td>0.0</td>\n",

" <td>1.000000</td>\n",

" <td>0.000000</td>\n",

" <td>NaN</td>\n",

" <td>22.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.296200</td>\n",

" <td>4.771083</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25%</th>\n",

" <td>1.150164e+07</td>\n",

" <td>1476.0</td>\n",

" <td>9.749992e+06</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.000000</td>\n",

" <td>0.0</td>\n",

" <td>2.000000</td>\n",

" <td>1.000000</td>\n",

" <td>NaN</td>\n",

" <td>114.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.353951</td>\n",

" <td>4.862918</td>\n",

" </tr>\n",

" <tr>\n",

" <th>50%</th>\n",

" <td>1.643578e+07</td>\n",

" <td>1476.0</td>\n",

" <td>2.962380e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.000000</td>\n",

" <td>0.0</td>\n",

" <td>2.000000</td>\n",

" <td>1.000000</td>\n",

" <td>NaN</td>\n",

" <td>150.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.363567</td>\n",

" <td>4.886634</td>\n",

" </tr>\n",

" <tr>\n",

" <th>75%</th>\n",

" <td>1.917347e+07</td>\n",

" <td>1476.0</td>\n",

" <td>7.652614e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>1.000000</td>\n",

" <td>0.0</td>\n",

" <td>4.000000</td>\n",

" <td>2.000000</td>\n",

" <td>NaN</td>\n",

" <td>209.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.374183</td>\n",

" <td>4.909861</td>\n",

" </tr>\n",

" <tr>\n",

" <th>max</th>\n",

" <td>2.000373e+07</td>\n",

" <td>1476.0</td>\n",

" <td>1.418319e+08</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>5.000000</td>\n",

" <td>0.0</td>\n",

" <td>16.000000</td>\n",

" <td>10.000000</td>\n",

" <td>NaN</td>\n",

" <td>6000.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.424980</td>\n",

" <td>5.013075</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id country borough reviews \\\n",

"count 5.721000e+03 5721.0 5.721000e+03 0.0 0.0 5721.000000 \n",

"mean 1.472219e+07 1476.0 4.674155e+07 NaN NaN 0.700926 \n",

"std 5.209774e+06 0.0 4.424869e+07 NaN NaN 0.814133 \n",

"min 2.542800e+04 1476.0 2.234000e+03 NaN NaN 0.000000 \n",

"25% 1.150164e+07 1476.0 9.749992e+06 NaN NaN 0.000000 \n",

"50% 1.643578e+07 1476.0 2.962380e+07 NaN NaN 0.000000 \n",

"75% 1.917347e+07 1476.0 7.652614e+07 NaN NaN 1.000000 \n",

"max 2.000373e+07 1476.0 1.418319e+08 NaN NaN 5.000000 \n",

"\n",

" overall\_satisfaction accommodates bedrooms bathrooms \\\n",

"count 5721.0 5721.000000 5721.000000 0.0 \n",

"mean 0.0 3.024821 1.554623 NaN \n",

"std 0.0 1.352056 0.949345 NaN \n",

"min 0.0 1.000000 0.000000 NaN \n",

"25% 0.0 2.000000 1.000000 NaN \n",

"50% 0.0 2.000000 1.000000 NaN \n",

"75% 0.0 4.000000 2.000000 NaN \n",

"max 0.0 16.000000 10.000000 NaN \n",

"\n",

" price minstay latitude longitude \n",

"count 5721.000000 0.0 5721.000000 5721.000000 \n",

"mean 175.620696 NaN 52.364361 4.889534 \n",

"std 140.020220 NaN 0.015795 0.036742 \n",

"min 22.000000 NaN 52.296200 4.771083 \n",

"25% 114.000000 NaN 52.353951 4.862918 \n",

"50% 150.000000 NaN 52.363567 4.886634 \n",

"75% 209.000000 NaN 52.374183 4.909861 \n",

"max 6000.000000 NaN 52.424980 5.013075 "

]

},

"execution\_count": 14,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb\_0\_satisfaction.describe()"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 621\n",

"0.0 483\n",

"4.5 411\n",

"4.0 56\n",

"3.5 8\n",

"3.0 4\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 15,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb[airbnb.accommodates == 3].overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 16,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 1787\n",

"0.0 1487\n",

"4.5 1033\n",

"4.0 127\n",

"3.5 16\n",

"3.0 6\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 16,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb[airbnb.bedrooms ==2].overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 1634\n",

"0.0 1169\n",

"4.5 881\n",

"4.0 104\n",

"3.5 12\n",

"3.0 4\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 17,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"# Rating based on the price range\n",

"airbnb[(airbnb.price >= 150) & (airbnb.price <=180)].overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 18,

"metadata": {},

"outputs": [],

"source": [

"# Based on above exploration, inserting overall satisfaction value of 4 where it is missing\n",

"airbnb\_imputed\_satisfaction = airbnb"

]

},

{

"cell\_type": "code",

"execution\_count": 19,

"metadata": {},

"outputs": [],

"source": [

"# Imputing the value 4 in overall satisfaction where the value is present as 0\n",

"airbnb\_imputed\_satisfaction.overall\_satisfaction = airbnb\_imputed\_satisfaction.overall\_satisfaction.replace(0,4)"

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 7697\n",

"4.0 6294\n",

"4.5 4543\n",

"3.5 105\n",

"3.0 19\n",

"1.0 1\n",

"2.5 1\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 20,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb\_imputed\_satisfaction.overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 21,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"5.0 7697\n",

"4.0 6294\n",

"4.5 4543\n",

"3.5 105\n",

"3.0 19\n",

"1.0 1\n",

"2.5 1\n",

"Name: overall\_satisfaction, dtype: int64"

]

},

"execution\_count": 21,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.overall\_satisfaction.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 22,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"2 9996\n",

"4 5563\n",

"3 1583\n",

"6 475\n",

"5 470\n",

"1 355\n",

"8 105\n",

"7 52\n",

"16 18\n",

"10 16\n",

"12 9\n",

"9 8\n",

"14 6\n",

"11 2\n",

"13 1\n",

"17 1\n",

"Name: accommodates, dtype: int64"

]

},

"execution\_count": 22,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.accommodates.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 23,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"119.0 1023\n",

"180.0 999\n",

"144.0 886\n",

"150.0 621\n",

"132.0 587\n",

"108.0 561\n",

"96.0 518\n",

"114.0 509\n",

"118.0 508\n",

"240.0 495\n",

"156.0 455\n",

"102.0 428\n",

"90.0 403\n",

"168.0 394\n",

"300.0 379\n",

"95.0 338\n",

"210.0 319\n",

"84.0 292\n",

"162.0 280\n",

"179.0 272\n",

"192.0 267\n",

"138.0 266\n",

"216.0 234\n",

"107.0 229\n",

"143.0 206\n",

"78.0 190\n",

"270.0 189\n",

"72.0 188\n",

"126.0 184\n",

"174.0 180\n",

" ... \n",

"310.0 1\n",

"762.0 1\n",

"451.0 1\n",

"398.0 1\n",

"755.0 1\n",

"339.0 1\n",

"279.0 1\n",

"482.0 1\n",

"445.0 1\n",

"467.0 1\n",

"392.0 1\n",

"417.0 1\n",

"366.0 1\n",

"344.0 1\n",

"416.0 1\n",

"373.0 1\n",

"406.0 1\n",

"22.0 1\n",

"534.0 1\n",

"1170.0 1\n",

"492.0 1\n",

"1078.0 1\n",

"502.0 1\n",

"367.0 1\n",

"1428.0 1\n",

"1016.0 1\n",

"328.0 1\n",

"1134.0 1\n",

"585.0 1\n",

"413.0 1\n",

"Name: price, Length: 419, dtype: int64"

]

},

"execution\_count": 23,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.price.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 24,

"metadata": {},

"outputs": [],

"source": [

"import numpy as np"

]

},

{

"cell\_type": "code",

"execution\_count": 25,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"1.0 0.591533\n",

"2.0 0.238800\n",

"3.0 0.077385\n",

"0.0 0.061844\n",

"4.0 0.025348\n",

"5.0 0.003323\n",

"6.0 0.001018\n",

"10.0 0.000268\n",

"7.0 0.000214\n",

"8.0 0.000161\n",

"9.0 0.000107\n",

"Name: bedrooms, dtype: float64"

]

},

"execution\_count": 25,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.bedrooms.value\_counts()/airbnb.bedrooms.count()"

]

},

{

"cell\_type": "code",

"execution\_count": 26,

"metadata": {},

"outputs": [],

"source": [

"# Dropping values where bedrooms are equal to zero\n",

"airbnb = airbnb[airbnb.bedrooms != 0]"

]

},

{

"cell\_type": "code",

"execution\_count": 27,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"1.0 11038\n",

"2.0 4456\n",

"3.0 1444\n",

"4.0 473\n",

"5.0 62\n",

"6.0 19\n",

"10.0 5\n",

"7.0 4\n",

"8.0 3\n",

"9.0 2\n",

"Name: bedrooms, dtype: int64"

]

},

"execution\_count": 27,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#Values without zero bedrooms\n",

"airbnb.bedrooms.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 28,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"1.0 11038\n",

"2.0 4456\n",

"3.0 1444\n",

"4.0 473\n",

"5.0 62\n",

"6.0 19\n",

"10.0 5\n",

"7.0 4\n",

"8.0 3\n",

"9.0 2\n",

"Name: bedrooms, dtype: int64"

]

},

"execution\_count": 28,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#values with zero bedrooms\n",

"airbnb.bedrooms.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 29,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"<class 'pandas.core.frame.DataFrame'>\n",

"Int64Index: 17506 entries, 33 to 18722\n",

"Data columns (total 20 columns):\n",

"room\_id 17506 non-null int64\n",

"survey\_id 17506 non-null int64\n",

"host\_id 17506 non-null int64\n",

"room\_type 17506 non-null object\n",

"country 0 non-null float64\n",

"city 17506 non-null object\n",

"borough 0 non-null float64\n",

"neighborhood 17506 non-null object\n",

"reviews 17506 non-null int64\n",

"overall\_satisfaction 17506 non-null float64\n",

"accommodates 17506 non-null int64\n",

"bedrooms 17506 non-null float64\n",

"bathrooms 0 non-null float64\n",

"price 17506 non-null float64\n",

"minstay 0 non-null float64\n",

"name 17460 non-null object\n",

"last\_modified 17506 non-null object\n",

"latitude 17506 non-null float64\n",

"longitude 17506 non-null float64\n",

"location 17506 non-null object\n",

"dtypes: float64(9), int64(5), object(6)\n",

"memory usage: 2.8+ MB\n"

]

}

],

"source": [

"airbnb.info()"

]

},

{

"cell\_type": "code",

"execution\_count": 30,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>country</th>\n",

" <th>borough</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>count</th>\n",

" <td>1.750600e+04</td>\n",

" <td>17506.0</td>\n",

" <td>1.750600e+04</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>17506.000000</td>\n",

" <td>17506.000000</td>\n",

" <td>17506.000000</td>\n",

" <td>17506.000000</td>\n",

" <td>0.0</td>\n",

" <td>17506.000000</td>\n",

" <td>0.0</td>\n",

" <td>17506.000000</td>\n",

" <td>17506.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>mean</th>\n",

" <td>1.108287e+07</td>\n",

" <td>1476.0</td>\n",

" <td>3.509582e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>16.368959</td>\n",

" <td>4.531932</td>\n",

" <td>2.960642</td>\n",

" <td>1.526220</td>\n",

" <td>NaN</td>\n",

" <td>169.137267</td>\n",

" <td>NaN</td>\n",

" <td>52.365207</td>\n",

" <td>4.888416</td>\n",

" </tr>\n",

" <tr>\n",

" <th>std</th>\n",

" <td>6.080450e+06</td>\n",

" <td>0.0</td>\n",

" <td>3.710345e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>32.796906</td>\n",

" <td>0.441955</td>\n",

" <td>1.333545</td>\n",

" <td>0.825627</td>\n",

" <td>NaN</td>\n",

" <td>111.009236</td>\n",

" <td>NaN</td>\n",

" <td>0.015112</td>\n",

" <td>0.034722</td>\n",

" </tr>\n",

" <tr>\n",

" <th>min</th>\n",

" <td>2.818000e+03</td>\n",

" <td>1476.0</td>\n",

" <td>2.234000e+03</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>0.000000</td>\n",

" <td>1.000000</td>\n",

" <td>1.000000</td>\n",

" <td>1.000000</td>\n",

" <td>NaN</td>\n",

" <td>12.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.296200</td>\n",

" <td>4.763264</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25%</th>\n",

" <td>5.923722e+06</td>\n",

" <td>1476.0</td>\n",

" <td>7.004309e+06</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>2.000000</td>\n",

" <td>4.000000</td>\n",

" <td>2.000000</td>\n",

" <td>1.000000</td>\n",

" <td>NaN</td>\n",

" <td>113.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.355178</td>\n",

" <td>4.863899</td>\n",

" </tr>\n",

" <tr>\n",

" <th>50%</th>\n",

" <td>1.210997e+07</td>\n",

" <td>1476.0</td>\n",

" <td>1.956281e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>6.000000</td>\n",

" <td>4.500000</td>\n",

" <td>2.000000</td>\n",

" <td>1.000000</td>\n",

" <td>NaN</td>\n",

" <td>144.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.364507</td>\n",

" <td>4.885528</td>\n",

" </tr>\n",

" <tr>\n",

" <th>75%</th>\n",

" <td>1.643984e+07</td>\n",

" <td>1476.0</td>\n",

" <td>5.089235e+07</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>17.000000</td>\n",

" <td>5.000000</td>\n",

" <td>4.000000</td>\n",

" <td>2.000000</td>\n",

" <td>NaN</td>\n",

" <td>196.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.374604</td>\n",

" <td>4.907448</td>\n",

" </tr>\n",

" <tr>\n",

" <th>max</th>\n",

" <td>2.000373e+07</td>\n",

" <td>1476.0</td>\n",

" <td>1.418319e+08</td>\n",

" <td>NaN</td>\n",

" <td>NaN</td>\n",

" <td>532.000000</td>\n",

" <td>5.000000</td>\n",

" <td>17.000000</td>\n",

" <td>10.000000</td>\n",

" <td>NaN</td>\n",

" <td>6000.000000</td>\n",

" <td>NaN</td>\n",

" <td>52.424980</td>\n",

" <td>5.027689</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id country borough reviews \\\n",

"count 1.750600e+04 17506.0 1.750600e+04 0.0 0.0 17506.000000 \n",

"mean 1.108287e+07 1476.0 3.509582e+07 NaN NaN 16.368959 \n",

"std 6.080450e+06 0.0 3.710345e+07 NaN NaN 32.796906 \n",

"min 2.818000e+03 1476.0 2.234000e+03 NaN NaN 0.000000 \n",

"25% 5.923722e+06 1476.0 7.004309e+06 NaN NaN 2.000000 \n",

"50% 1.210997e+07 1476.0 1.956281e+07 NaN NaN 6.000000 \n",

"75% 1.643984e+07 1476.0 5.089235e+07 NaN NaN 17.000000 \n",

"max 2.000373e+07 1476.0 1.418319e+08 NaN NaN 532.000000 \n",

"\n",

" overall\_satisfaction accommodates bedrooms bathrooms \\\n",

"count 17506.000000 17506.000000 17506.000000 0.0 \n",

"mean 4.531932 2.960642 1.526220 NaN \n",

"std 0.441955 1.333545 0.825627 NaN \n",

"min 1.000000 1.000000 1.000000 NaN \n",

"25% 4.000000 2.000000 1.000000 NaN \n",

"50% 4.500000 2.000000 1.000000 NaN \n",

"75% 5.000000 4.000000 2.000000 NaN \n",

"max 5.000000 17.000000 10.000000 NaN \n",

"\n",

" price minstay latitude longitude \n",

"count 17506.000000 0.0 17506.000000 17506.000000 \n",

"mean 169.137267 NaN 52.365207 4.888416 \n",

"std 111.009236 NaN 0.015112 0.034722 \n",

"min 12.000000 NaN 52.296200 4.763264 \n",

"25% 113.000000 NaN 52.355178 4.863899 \n",

"50% 144.000000 NaN 52.364507 4.885528 \n",

"75% 196.000000 NaN 52.374604 4.907448 \n",

"max 6000.000000 NaN 52.424980 5.027689 "

]

},

"execution\_count": 30,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.describe()"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"# Data Visualization"

]

},

{

"cell\_type": "code",

"execution\_count": 31,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>room\_type</th>\n",

" <th>country</th>\n",

" <th>city</th>\n",

" <th>borough</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>name</th>\n",

" <th>last\_modified</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" <th>location</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>10201214</td>\n",

" <td>1476</td>\n",

" <td>8497487</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Ijburg / Eiland Zeeburg</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" <td>NaN</td>\n",

" <td>763.0</td>\n",

" <td>NaN</td>\n",

" <td>Beautiful Watervilla in Amsterdam</td>\n",

" <td>2017-07-23 13:02:10.468528</td>\n",

" <td>52.348254</td>\n",

" <td>5.001477</td>\n",

" <td>0101000020E6100000E7C8CA2F83011440C0594A96932C...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>3119298</td>\n",

" <td>1476</td>\n",

" <td>15847782</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Westerpark</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" <td>NaN</td>\n",

" <td>445.0</td>\n",

" <td>NaN</td>\n",

" <td>Modern 5-8 person apartment</td>\n",

" <td>2017-07-23 12:58:15.945759</td>\n",

" <td>52.377581</td>\n",

" <td>4.873119</td>\n",

" <td>0101000020E61000009D103AE8127E1340A54BFF925430...</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id room\_type country city \\\n",

"33 10201214 1476 8497487 Entire home/apt NaN Amsterdam \n",

"34 3119298 1476 15847782 Entire home/apt NaN Amsterdam \n",

"\n",

" borough neighborhood reviews overall\_satisfaction \\\n",

"33 NaN Ijburg / Eiland Zeeburg 19 5.0 \n",

"34 NaN Westerpark 1 4.0 \n",

"\n",

" accommodates bedrooms bathrooms price minstay \\\n",

"33 8 4.0 NaN 763.0 NaN \n",

"34 8 3.0 NaN 445.0 NaN \n",

"\n",

" name last\_modified latitude \\\n",

"33 Beautiful Watervilla in Amsterdam 2017-07-23 13:02:10.468528 52.348254 \n",

"34 Modern 5-8 person apartment 2017-07-23 12:58:15.945759 52.377581 \n",

"\n",

" longitude location \n",

"33 5.001477 0101000020E6100000E7C8CA2F83011440C0594A96932C... \n",

"34 4.873119 0101000020E61000009D103AE8127E1340A54BFF925430... "

]

},

"execution\_count": 31,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 32,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([7292., 3231., 1943., 1235., 836., 577., 411., 321., 232.,\n",

" 196., 159., 115., 91., 77., 82., 51., 49., 45.,\n",

" 44., 57.]),\n",

" array([ 0., 5., 10., 15., 20., 25., 30., 35., 40., 45., 50.,\n",

" 55., 60., 65., 70., 75., 80., 85., 90., 95., 100.]),\n",

" <a list of 20 Patch objects>)"

]

},

"execution\_count": 32,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#airbnb.reviews.plot(kind = 'hist')\n",

"plt.hist(airbnb.reviews,bins=20,range=(0,100))"

]

},

{

"cell\_type": "code",

"execution\_count": 33,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([1.870e+02, 2.858e+03, 5.970e+03, 4.259e+03, 2.002e+03, 7.720e+02,\n",

" 6.360e+02, 3.180e+02, 1.560e+02, 1.530e+02, 6.200e+01, 3.800e+01,\n",

" 1.000e+01, 1.600e+01, 2.000e+01, 1.200e+01, 6.000e+00, 1.000e+00,\n",

" 5.000e+00, 3.000e+00]),\n",

" array([ 0., 50., 100., 150., 200., 250., 300., 350., 400.,\n",

" 450., 500., 550., 600., 650., 700., 750., 800., 850.,\n",

" 900., 950., 1000.]),\n",

" <a list of 20 Patch objects>)"

]

},

"execution\_count": 33,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb.price,bins=20,range=(0,1000))"

]

},

{

"cell\_type": "code",

"execution\_count": 34,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([3.250e+02, 9.105e+03, 1.489e+03, 5.437e+03, 4.640e+02, 4.710e+02,\n",

" 5.200e+01, 1.050e+02, 8.000e+00, 1.500e+01, 2.000e+00, 8.000e+00,\n",

" 0.000e+00, 6.000e+00, 0.000e+00, 1.900e+01]),\n",

" array([ 1., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12., 13.,\n",

" 14., 15., 16., 17.]),\n",

" <a list of 16 Patch objects>)"

]

},

"execution\_count": 34,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb.accommodates,bins=16)"

]

},

{

"cell\_type": "code",

"execution\_count": 35,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([1.000e+00, 0.000e+00, 0.000e+00, 1.000e+00, 0.000e+00, 1.500e+01,\n",

" 9.500e+01, 5.902e+03, 4.226e+03, 7.266e+03]),\n",

" array([1. , 1.4, 1.8, 2.2, 2.6, 3. , 3.4, 3.8, 4.2, 4.6, 5. ]),\n",

" <a list of 10 Patch objects>)"

]

},

"execution\_count": 35,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb.overall\_satisfaction)"

]

},

{

"cell\_type": "code",

"execution\_count": 36,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([ 6., 0., 43., 0., 0., 3291., 0., 2489., 0.,\n",

" 4401.]),\n",

" array([3. , 3.2, 3.4, 3.6, 3.8, 4. , 4.2, 4.4, 4.6, 4.8, 5. ]),\n",

" <a list of 10 Patch objects>)"

]

},

"execution\_count": 36,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb[(airbnb.price >= 100) & (airbnb.price <= 200)].overall\_satisfaction)"

]

},

{

"cell\_type": "code",

"execution\_count": 37,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([ 2., 0., 0., 78., 0., 0., 61., 0., 0., 46.]),\n",

" array([3.5 , 3.65, 3.8 , 3.95, 4.1 , 4.25, 4.4 , 4.55, 4.7 , 4.85, 5. ]),\n",

" <a list of 10 Patch objects>)"

]

},

"execution\_count": 37,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb[airbnb.price < 50].overall\_satisfaction)"

]

},

{

"cell\_type": "code",

"execution\_count": 38,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([1.000e+00, 0.000e+00, 4.000e+00, 0.000e+00, 1.700e+01, 0.000e+00,\n",

" 1.550e+03, 0.000e+00, 7.850e+02, 1.874e+03]),\n",

" array([2.5 , 2.75, 3. , 3.25, 3.5 , 3.75, 4. , 4.25, 4.5 , 4.75, 5. ]),\n",

" <a list of 10 Patch objects>)"

]

},

"execution\_count": 38,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb[airbnb.price > 200].overall\_satisfaction)"

]

},

{

"cell\_type": "code",

"execution\_count": 39,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(array([ 1., 0., 0., 0., 5., 0., 402., 0., 198., 481.]),\n",

" array([2.5 , 2.75, 3. , 3.25, 3.5 , 3.75, 4. , 4.25, 4.5 , 4.75, 5. ]),\n",

" <a list of 10 Patch objects>)"

]

},

"execution\_count": 39,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.hist(airbnb[airbnb.price > 300].overall\_satisfaction)"

]

},

{

"cell\_type": "code",

"execution\_count": 40,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.collections.PathCollection at 0x13941115fd0>"

]

},

"execution\_count": 40,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.scatter(airbnb.price, airbnb.reviews)"

]

},

{

"cell\_type": "code",

"execution\_count": 41,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.collections.PathCollection at 0x139411b13c8>"

]

},

"execution\_count": 41,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.scatter(airbnb.accommodates,airbnb.price)"

]

},

{

"cell\_type": "code",

"execution\_count": 42,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.collections.PathCollection at 0x1394120f828>"

]

},

"execution\_count": 42,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.scatter(airbnb.overall\_satisfaction,airbnb.price)"

]

},

{

"cell\_type": "code",

"execution\_count": 43,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.collections.PathCollection at 0x13941282400>"

]

},

"execution\_count": 43,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"plt.scatter(airbnb.bedrooms,airbnb.price)"

]

},

{

"cell\_type": "code",

"execution\_count": 44,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.axes.\_subplots.AxesSubplot at 0x139412b2dd8>"

]

},

"execution\_count": 44,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"airbnb.plot()"

]

},

{

"cell\_type": "code",

"execution\_count": 45,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<matplotlib.axes.\_subplots.AxesSubplot at 0x139413941d0>"

]

},

"execution\_count": 45,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 1440x360 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#Plotting neighborhood count\n",

"airbnb.neighborhood.value\_counts().plot(kind='bar',figsize=(20,5))"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"airbnb.neighborhood.value\_counts().plot(kind='barh')"

]

},

{

"cell\_type": "code",

"execution\_count": 46,

"metadata": {},

"outputs": [],

"source": [

"# Removing rows which are above Price range of 1500\n",

"airbnb[airbnb.price > 1500].count()\n",

"airbnb = airbnb[airbnb.price <1500]"

]

},

{

"cell\_type": "code",

"execution\_count": 47,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"180.0 961\n",

"119.0 948\n",

"144.0 819\n",

"150.0 583\n",

"132.0 549\n",

"108.0 502\n",

"240.0 487\n",

"114.0 469\n",

"96.0 455\n",

"118.0 452\n",

"156.0 430\n",

"168.0 381\n",

"102.0 378\n",

"300.0 371\n",

"90.0 368\n",

"210.0 316\n",

"95.0 305\n",

"84.0 271\n",

"179.0 262\n",

"162.0 259\n",

"192.0 256\n",

"138.0 243\n",

"216.0 231\n",

"107.0 212\n",

"143.0 189\n",

"270.0 187\n",

"78.0 175\n",

"72.0 174\n",

"126.0 170\n",

"174.0 165\n",

" ... \n",

"521.0 1\n",

"443.0 1\n",

"366.0 1\n",

"1199.0 1\n",

"507.0 1\n",

"373.0 1\n",

"406.0 1\n",

"444.0 1\n",

"1078.0 1\n",

"1016.0 1\n",

"1134.0 1\n",

"467.0 1\n",

"301.0 1\n",

"1050.0 1\n",

"328.0 1\n",

"567.0 1\n",

"337.0 1\n",

"398.0 1\n",

"451.0 1\n",

"457.0 1\n",

"23.0 1\n",

"310.0 1\n",

"29.0 1\n",

"1386.0 1\n",

"1319.0 1\n",

"659.0 1\n",

"371.0 1\n",

"416.0 1\n",

"756.0 1\n",

"413.0 1\n",

"Name: price, Length: 413, dtype: int64"

]

},

"execution\_count": 47,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.price.value\_counts()"

]

},

{

"cell\_type": "code",

"execution\_count": 48,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.model\_selection import train\_test\_split\n",

"from sklearn.linear\_model import LinearRegression"

]

},

{

"cell\_type": "code",

"execution\_count": 50,

"metadata": {},

"outputs": [

{

"ename": "NameError",

"evalue": "name 'X' is not defined",

"output\_type": "error",

"traceback": [

"\u001b[1;31m---------------------------------------------------------------------------\u001b[0m",

"\u001b[1;31mNameError\u001b[0m Traceback (most recent call last)",

"\u001b[1;32m<ipython-input-50-ae5cd47bda90>\u001b[0m in \u001b[0;36m<module>\u001b[1;34m\u001b[0m\n\u001b[1;32m----> 1\u001b[1;33m \u001b[0mX\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mhead\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0m",

"\u001b[1;31mNameError\u001b[0m: name 'X' is not defined"

]

}

],

"source": [

"X.head()"

]

},

{

"cell\_type": "code",

"execution\_count": 60,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>room\_type</th>\n",

" <th>country</th>\n",

" <th>city</th>\n",

" <th>borough</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>name</th>\n",

" <th>last\_modified</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" <th>location</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>10201214</td>\n",

" <td>1476</td>\n",

" <td>8497487</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Ijburg / Eiland Zeeburg</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" <td>NaN</td>\n",

" <td>763.0</td>\n",

" <td>NaN</td>\n",

" <td>Beautiful Watervilla in Amsterdam</td>\n",

" <td>2017-07-23 13:02:10.468528</td>\n",

" <td>52.348254</td>\n",

" <td>5.001477</td>\n",

" <td>0101000020E6100000E7C8CA2F83011440C0594A96932C...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>3119298</td>\n",

" <td>1476</td>\n",

" <td>15847782</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Westerpark</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" <td>NaN</td>\n",

" <td>445.0</td>\n",

" <td>NaN</td>\n",

" <td>Modern 5-8 person apartment</td>\n",

" <td>2017-07-23 12:58:15.945759</td>\n",

" <td>52.377581</td>\n",

" <td>4.873119</td>\n",

" <td>0101000020E61000009D103AE8127E1340A54BFF925430...</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id room\_type country city \\\n",

"33 10201214 1476 8497487 Entire home/apt NaN Amsterdam \n",

"34 3119298 1476 15847782 Entire home/apt NaN Amsterdam \n",

"\n",

" borough neighborhood reviews overall\_satisfaction \\\n",

"33 NaN Ijburg / Eiland Zeeburg 19 5.0 \n",

"34 NaN Westerpark 1 4.0 \n",

"\n",

" accommodates bedrooms bathrooms price minstay \\\n",

"33 8 4.0 NaN 763.0 NaN \n",

"34 8 3.0 NaN 445.0 NaN \n",

"\n",

" name last\_modified latitude \\\n",

"33 Beautiful Watervilla in Amsterdam 2017-07-23 13:02:10.468528 52.348254 \n",

"34 Modern 5-8 person apartment 2017-07-23 12:58:15.945759 52.377581 \n",

"\n",

" longitude location \n",

"33 5.001477 0101000020E6100000E7C8CA2F83011440C0594A96932C... \n",

"34 4.873119 0101000020E61000009D103AE8127E1340A54BFF925430... "

]

},

"execution\_count": 60,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 56,

"metadata": {},

"outputs": [],

"source": [

"X = airbnb.iloc[:,[8,9,10,11,]]"

]

},

{

"cell\_type": "code",

"execution\_count": 63,

"metadata": {},

"outputs": [],

"source": [

"Y = airbnb.price"

]

},

{

"cell\_type": "code",

"execution\_count": 65,

"metadata": {},

"outputs": [],

"source": [

"X\_train,X\_test,y\_train,y\_test = train\_test\_split(X.values,Y.values,test\_size = .3,random\_state=1)"

]

},

{

"cell\_type": "code",

"execution\_count": 66,

"metadata": {},

"outputs": [],

"source": [

"lm = LinearRegression()"

]

},

{

"cell\_type": "code",

"execution\_count": 67,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)"

]

},

"execution\_count": 67,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"lm.fit(X\_train,y\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 68,

"metadata": {},

"outputs": [],

"source": [

"y\_pred = lm.predict(X\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 69,

"metadata": {},

"outputs": [],

"source": [

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

]

},

{

"cell\_type": "code",

"execution\_count": 70,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"49.760048108236774"

]

},

"execution\_count": 70,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"mean\_absolute\_error(y\_test,y\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 71,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"77.0308769078892"

]

},

"execution\_count": 71,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.sqrt(mean\_squared\_error(y\_test,y\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": 72,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([-0.15730951, 13.06118553, 26.17337276, 29.26738688])"

]

},

"execution\_count": 72,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"lm.coef\_"

]

},

{

"cell\_type": "code",

"execution\_count": 73,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"<bound method RegressorMixin.score of LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)>\n"

]

}

],

"source": [

"print(lm.score)"

]

},

{

"cell\_type": "code",

"execution\_count": 74,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"0.3555865319381407"

]

},

"execution\_count": 74,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"r2\_score(y\_test,y\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 75,

"metadata": {},

"outputs": [],

"source": [

"# Using one hot encoding to transform categorical values"

]

},

{

"cell\_type": "code",

"execution\_count": 76,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>room\_type</th>\n",

" <th>country</th>\n",

" <th>city</th>\n",

" <th>borough</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>name</th>\n",

" <th>last\_modified</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" <th>location</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>10201214</td>\n",

" <td>1476</td>\n",

" <td>8497487</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Ijburg / Eiland Zeeburg</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" <td>NaN</td>\n",

" <td>763.0</td>\n",

" <td>NaN</td>\n",

" <td>Beautiful Watervilla in Amsterdam</td>\n",

" <td>2017-07-23 13:02:10.468528</td>\n",

" <td>52.348254</td>\n",

" <td>5.001477</td>\n",

" <td>0101000020E6100000E7C8CA2F83011440C0594A96932C...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>3119298</td>\n",

" <td>1476</td>\n",

" <td>15847782</td>\n",

" <td>Entire home/apt</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>Westerpark</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" <td>NaN</td>\n",

" <td>445.0</td>\n",

" <td>NaN</td>\n",

" <td>Modern 5-8 person apartment</td>\n",

" <td>2017-07-23 12:58:15.945759</td>\n",

" <td>52.377581</td>\n",

" <td>4.873119</td>\n",

" <td>0101000020E61000009D103AE8127E1340A54BFF925430...</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id room\_type country city \\\n",

"33 10201214 1476 8497487 Entire home/apt NaN Amsterdam \n",

"34 3119298 1476 15847782 Entire home/apt NaN Amsterdam \n",

"\n",

" borough neighborhood reviews overall\_satisfaction \\\n",

"33 NaN Ijburg / Eiland Zeeburg 19 5.0 \n",

"34 NaN Westerpark 1 4.0 \n",

"\n",

" accommodates bedrooms bathrooms price minstay \\\n",

"33 8 4.0 NaN 763.0 NaN \n",

"34 8 3.0 NaN 445.0 NaN \n",

"\n",

" name last\_modified latitude \\\n",

"33 Beautiful Watervilla in Amsterdam 2017-07-23 13:02:10.468528 52.348254 \n",

"34 Modern 5-8 person apartment 2017-07-23 12:58:15.945759 52.377581 \n",

"\n",

" longitude location \n",

"33 5.001477 0101000020E6100000E7C8CA2F83011440C0594A96932C... \n",

"34 4.873119 0101000020E61000009D103AE8127E1340A54BFF925430... "

]

},

"execution\_count": 76,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 77,

"metadata": {},

"outputs": [],

"source": [

"# Because one hot encoding only takes numerical value, first need to convert the label into numeric and then use \n",

"# one hot encoding"

]

},

{

"cell\_type": "code",

"execution\_count": 78,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.preprocessing import LabelEncoder"

]

},

{

"cell\_type": "code",

"execution\_count": 79,

"metadata": {},

"outputs": [],

"source": [

"le = LabelEncoder()"

]

},

{

"cell\_type": "code",

"execution\_count": 80,

"metadata": {},

"outputs": [],

"source": [

"airbnb['room\_type'] = le.fit\_transform(airbnb['room\_type'])"

]

},

{

"cell\_type": "code",

"execution\_count": 81,

"metadata": {},

"outputs": [],

"source": [

"airbnb['neighborhood'] = le.fit\_transform(airbnb['neighborhood'])"

]

},

{

"cell\_type": "code",

"execution\_count": 82,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_id</th>\n",

" <th>survey\_id</th>\n",

" <th>host\_id</th>\n",

" <th>room\_type</th>\n",

" <th>country</th>\n",

" <th>city</th>\n",

" <th>borough</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" <th>bathrooms</th>\n",

" <th>price</th>\n",

" <th>minstay</th>\n",

" <th>name</th>\n",

" <th>last\_modified</th>\n",

" <th>latitude</th>\n",

" <th>longitude</th>\n",

" <th>location</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>10201214</td>\n",

" <td>1476</td>\n",

" <td>8497487</td>\n",

" <td>0</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>11</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" <td>NaN</td>\n",

" <td>763.0</td>\n",

" <td>NaN</td>\n",

" <td>Beautiful Watervilla in Amsterdam</td>\n",

" <td>2017-07-23 13:02:10.468528</td>\n",

" <td>52.348254</td>\n",

" <td>5.001477</td>\n",

" <td>0101000020E6100000E7C8CA2F83011440C0594A96932C...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>3119298</td>\n",

" <td>1476</td>\n",

" <td>15847782</td>\n",

" <td>0</td>\n",

" <td>NaN</td>\n",

" <td>Amsterdam</td>\n",

" <td>NaN</td>\n",

" <td>21</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" <td>NaN</td>\n",

" <td>445.0</td>\n",

" <td>NaN</td>\n",

" <td>Modern 5-8 person apartment</td>\n",

" <td>2017-07-23 12:58:15.945759</td>\n",

" <td>52.377581</td>\n",

" <td>4.873119</td>\n",

" <td>0101000020E61000009D103AE8127E1340A54BFF925430...</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_id survey\_id host\_id room\_type country city borough \\\n",

"33 10201214 1476 8497487 0 NaN Amsterdam NaN \n",

"34 3119298 1476 15847782 0 NaN Amsterdam NaN \n",

"\n",

" neighborhood reviews overall\_satisfaction accommodates bedrooms \\\n",

"33 11 19 5.0 8 4.0 \n",

"34 21 1 4.0 8 3.0 \n",

"\n",

" bathrooms price minstay name \\\n",

"33 NaN 763.0 NaN Beautiful Watervilla in Amsterdam \n",

"34 NaN 445.0 NaN Modern 5-8 person apartment \n",

"\n",

" last\_modified latitude longitude \\\n",

"33 2017-07-23 13:02:10.468528 52.348254 5.001477 \n",

"34 2017-07-23 12:58:15.945759 52.377581 4.873119 \n",

"\n",

" location \n",

"33 0101000020E6100000E7C8CA2F83011440C0594A96932C... \n",

"34 0101000020E61000009D103AE8127E1340A54BFF925430... "

]

},

"execution\_count": 82,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"airbnb.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 83,

"metadata": {},

"outputs": [],

"source": [

"# Above we have converted the lables, now we'll use one hot encoding to create new variables"

]

},

{

"cell\_type": "code",

"execution\_count": 84,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.preprocessing import OneHotEncoder"

]

},

{

"cell\_type": "code",

"execution\_count": 85,

"metadata": {},

"outputs": [],

"source": [

"ohc = OneHotEncoder(categorical\_features=[0,1])"

]

},

{

"cell\_type": "code",

"execution\_count": 86,

"metadata": {},

"outputs": [],

"source": [

"X1 = airbnb.iloc[:,[3,7,8,9,10,11]]"

]

},

{

"cell\_type": "code",

"execution\_count": 87,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_type</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>0</td>\n",

" <td>11</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>0</td>\n",

" <td>21</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_type neighborhood reviews overall\_satisfaction accommodates \\\n",

"33 0 11 19 5.0 8 \n",

"34 0 21 1 4.0 8 \n",

"\n",

" bedrooms \n",

"33 4.0 \n",

"34 3.0 "

]

},

"execution\_count": 87,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"X1.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 88,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\preprocessing\\\_encoders.py:415: FutureWarning: The handling of integer data will change in version 0.22. Currently, the categories are determined based on the range [0, max(values)], while in the future they will be determined based on the unique values.\n",

"If you want the future behaviour and silence this warning, you can specify \"categories='auto'\".\n",

"In case you used a LabelEncoder before this OneHotEncoder to convert the categories to integers, then you can now use the OneHotEncoder directly.\n",

" warnings.warn(msg, FutureWarning)\n",

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\preprocessing\\\_encoders.py:451: DeprecationWarning: The 'categorical\_features' keyword is deprecated in version 0.20 and will be removed in 0.22. You can use the ColumnTransformer instead.\n",

" \"use the ColumnTransformer instead.\", DeprecationWarning)\n"

]

}

],

"source": [

"X1 = ohc.fit\_transform(X1).toarray()"

]

},

{

"cell\_type": "code",

"execution\_count": 89,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[1. , 0. , 0. , ..., 5. , 8. , 4. ],\n",

" [1. , 0. , 0. , ..., 4. , 8. , 3. ],\n",

" [1. , 0. , 0. , ..., 5. , 4. , 3. ],\n",

" ...,\n",

" [0. , 1. , 0. , ..., 4. , 1. , 1. ],\n",

" [0. , 1. , 0. , ..., 4.5, 2. , 1. ],\n",

" [0. , 1. , 0. , ..., 4.5, 2. , 1. ]])"

]

},

"execution\_count": 89,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"X1"

]

},

{

"cell\_type": "code",

"execution\_count": 90,

"metadata": {},

"outputs": [],

"source": [

"Y1 = airbnb.price"

]

},

{

"cell\_type": "code",

"execution\_count": 91,

"metadata": {},

"outputs": [],

"source": [

"X1\_train,X1\_test,y1\_train,y1\_test = train\_test\_split(X1,Y1.values,test\_size=.3,random\_state=3)"

]

},

{

"cell\_type": "code",

"execution\_count": 92,

"metadata": {},

"outputs": [],

"source": [

"lm1 = LinearRegression()"

]

},

{

"cell\_type": "code",

"execution\_count": 93,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)"

]

},

"execution\_count": 93,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"lm1.fit(X1\_train, y1\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 94,

"metadata": {},

"outputs": [],

"source": [

"y1\_pred = lm1.predict(X1\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 95,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"0.45014378238653396"

]

},

"execution\_count": 95,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"r2\_score(y1\_test,y1\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 96,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"45.06464151273021"

]

},

"execution\_count": 96,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"mean\_absolute\_error(y1\_test,y1\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 97,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"71.54915621231251"

]

},

"execution\_count": 97,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.sqrt(mean\_squared\_error(y1\_test,y1\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": 98,

"metadata": {},

"outputs": [],

"source": [

"# Creating log linear regression"

]

},

{

"cell\_type": "code",

"execution\_count": 99,

"metadata": {},

"outputs": [],

"source": [

"X\_log = airbnb.iloc[:,[3,7,8,9,10,11]]"

]

},

{

"cell\_type": "code",

"execution\_count": 100,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_type</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>0</td>\n",

" <td>11</td>\n",

" <td>19</td>\n",

" <td>5.0</td>\n",

" <td>8</td>\n",

" <td>4.0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>0</td>\n",

" <td>21</td>\n",

" <td>1</td>\n",

" <td>4.0</td>\n",

" <td>8</td>\n",

" <td>3.0</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_type neighborhood reviews overall\_satisfaction accommodates \\\n",

"33 0 11 19 5.0 8 \n",

"34 0 21 1 4.0 8 \n",

"\n",

" bedrooms \n",

"33 4.0 \n",

"34 3.0 "

]

},

"execution\_count": 100,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"X\_log.head(2)"

]

},

{

"cell\_type": "code",

"execution\_count": 101,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\ipykernel\_launcher.py:1: SettingWithCopyWarning: \n",

"A value is trying to be set on a copy of a slice from a DataFrame.\n",

"Try using .loc[row\_indexer,col\_indexer] = value instead\n",

"\n",

"See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy\n",

" \"\"\"Entry point for launching an IPython kernel.\n",

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\ipykernel\_launcher.py:2: SettingWithCopyWarning: \n",

"A value is trying to be set on a copy of a slice from a DataFrame.\n",

"Try using .loc[row\_indexer,col\_indexer] = value instead\n",

"\n",

"See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy\n",

" \n",

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\ipykernel\_launcher.py:3: SettingWithCopyWarning: \n",

"A value is trying to be set on a copy of a slice from a DataFrame.\n",

"Try using .loc[row\_indexer,col\_indexer] = value instead\n",

"\n",

"See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy\n",

" This is separate from the ipykernel package so we can avoid doing imports until\n"

]

}

],

"source": [

"X\_log['reviews'] = np.log10(X\_log['reviews']+1)\n",

"X\_log['accommodates'] = np.log10(X\_log['accommodates'])\n",

"X\_log['bedrooms'] = np.log10(X\_log['bedrooms'])"

]

},

{

"cell\_type": "code",

"execution\_count": 102,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>room\_type</th>\n",

" <th>neighborhood</th>\n",

" <th>reviews</th>\n",

" <th>overall\_satisfaction</th>\n",

" <th>accommodates</th>\n",

" <th>bedrooms</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>33</th>\n",

" <td>0</td>\n",

" <td>11</td>\n",

" <td>1.301030</td>\n",

" <td>5.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>34</th>\n",

" <td>0</td>\n",

" <td>21</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>35</th>\n",

" <td>0</td>\n",

" <td>5</td>\n",

" <td>1.041393</td>\n",

" <td>5.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>36</th>\n",

" <td>0</td>\n",

" <td>20</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>37</th>\n",

" <td>0</td>\n",

" <td>5</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>38</th>\n",

" <td>0</td>\n",

" <td>14</td>\n",

" <td>0.477121</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>39</th>\n",

" <td>0</td>\n",

" <td>8</td>\n",

" <td>1.380211</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>40</th>\n",

" <td>0</td>\n",

" <td>6</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>41</th>\n",

" <td>0</td>\n",

" <td>8</td>\n",

" <td>1.255273</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>42</th>\n",

" <td>0</td>\n",

" <td>4</td>\n",

" <td>1.204120</td>\n",

" <td>5.0</td>\n",

" <td>0.778151</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" room\_type neighborhood reviews overall\_satisfaction accommodates \\\n",

"33 0 11 1.301030 5.0 0.903090 \n",

"34 0 21 0.301030 4.0 0.903090 \n",

"35 0 5 1.041393 5.0 0.602060 \n",

"36 0 20 0.301030 4.0 0.903090 \n",

"37 0 5 0.000000 4.0 0.602060 \n",

"38 0 14 0.477121 4.0 0.602060 \n",

"39 0 8 1.380211 4.5 0.602060 \n",

"40 0 6 0.000000 4.0 0.602060 \n",

"41 0 8 1.255273 4.5 0.602060 \n",

"42 0 4 1.204120 5.0 0.778151 \n",

"\n",

" bedrooms \n",

"33 0.602060 \n",

"34 0.477121 \n",

"35 0.477121 \n",

"36 0.477121 \n",

"37 0.602060 \n",

"38 0.301030 \n",

"39 0.602060 \n",

"40 0.301030 \n",

"41 0.301030 \n",

"42 0.602060 "

]

},

"execution\_count": 102,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"X\_log.head(10)"

]

},

{

"cell\_type": "code",

"execution\_count": 103,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\preprocessing\\\_encoders.py:415: FutureWarning: The handling of integer data will change in version 0.22. Currently, the categories are determined based on the range [0, max(values)], while in the future they will be determined based on the unique values.\n",

"If you want the future behaviour and silence this warning, you can specify \"categories='auto'\".\n",

"In case you used a LabelEncoder before this OneHotEncoder to convert the categories to integers, then you can now use the OneHotEncoder directly.\n",

" warnings.warn(msg, FutureWarning)\n",

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\preprocessing\\\_encoders.py:451: DeprecationWarning: The 'categorical\_features' keyword is deprecated in version 0.20 and will be removed in 0.22. You can use the ColumnTransformer instead.\n",

" \"use the ColumnTransformer instead.\", DeprecationWarning)\n"

]

}

],

"source": [

"X\_log = ohc.fit\_transform(X\_log).toarray()"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": 104,

"metadata": {},

"outputs": [],

"source": [

"xlog\_train,xlog\_test,ylog\_train,ylog\_test = train\_test\_split(X\_log,Y.values,test\_size=.3,random\_state=5)"

]

},

{

"cell\_type": "code",

"execution\_count": 105,

"metadata": {},

"outputs": [],

"source": [

"lm\_log = LinearRegression()"

]

},

{

"cell\_type": "code",

"execution\_count": 106,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)"

]

},

"execution\_count": 106,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"lm\_log.fit(xlog\_train,ylog\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 107,

"metadata": {},

"outputs": [],

"source": [

"ylog\_pred = lm\_log.predict(xlog\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 108,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"0.3734280548648793"

]

},

"execution\_count": 108,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"r2\_score(ylog\_test,ylog\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 109,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"47.035993144163015"

]

},

"execution\_count": 109,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"mean\_absolute\_error(ylog\_test,ylog\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 110,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"79.67425390092829"

]

},

"execution\_count": 110,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": 111,

"metadata": {},

"outputs": [],

"source": [

"xlog\_df = pd.DataFrame(X\_log)"

]

},

{

"cell\_type": "code",

"execution\_count": 112,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>0</th>\n",

" <th>1</th>\n",

" <th>2</th>\n",

" <th>3</th>\n",

" <th>4</th>\n",

" <th>5</th>\n",

" <th>6</th>\n",

" <th>7</th>\n",

" <th>8</th>\n",

" <th>9</th>\n",

" <th>...</th>\n",

" <th>19</th>\n",

" <th>20</th>\n",

" <th>21</th>\n",

" <th>22</th>\n",

" <th>23</th>\n",

" <th>24</th>\n",

" <th>25</th>\n",

" <th>26</th>\n",

" <th>27</th>\n",

" <th>28</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.301030</td>\n",

" <td>5.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.041393</td>\n",

" <td>5.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>5</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.477121</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>6</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.380211</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>7</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>8</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.255273</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.204120</td>\n",

" <td>5.0</td>\n",

" <td>0.778151</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>10</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.698970</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>11</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>12</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.414973</td>\n",

" <td>4.5</td>\n",

" <td>0.477121</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>13</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.477121</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>14</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>15</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>16</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.255273</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.342423</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.301030</td>\n",

" </tr>\n",

" <tr>\n",

" <th>18</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.079181</td>\n",

" <td>5.0</td>\n",

" <td>0.778151</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>19</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.778151</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>20</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.146128</td>\n",

" <td>4.5</td>\n",

" <td>0.778151</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>21</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>1.799341</td>\n",

" <td>5.0</td>\n",

" <td>1.204120</td>\n",

" <td>1.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>22</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.778151</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>23</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.342423</td>\n",

" <td>5.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>24</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.698970</td>\n",

" </tr>\n",

" <tr>\n",

" <th>26</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.230449</td>\n",

" <td>4.5</td>\n",

" <td>1.000000</td>\n",

" <td>0.602060</td>\n",

" </tr>\n",

" <tr>\n",

" <th>27</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>28</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.477121</td>\n",

" <td>4.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.698970</td>\n",

" </tr>\n",

" <tr>\n",

" <th>29</th>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>...</th>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17471</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17472</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.000000</td>\n",

" <td>4.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17473</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.176091</td>\n",

" <td>4.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17474</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.698970</td>\n",

" <td>4.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17475</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17476</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17477</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.544068</td>\n",

" <td>5.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17478</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.602060</td>\n",

" <td>3.5</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17479</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.763428</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17480</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17481</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.079181</td>\n",

" <td>3.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17482</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.380211</td>\n",

" <td>5.0</td>\n",

" <td>0.477121</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17483</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.698970</td>\n",

" <td>4.5</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17484</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.602060</td>\n",

" <td>4.5</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17485</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17486</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>2.206826</td>\n",

" <td>4.5</td>\n",

" <td>0.602060</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17487</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17488</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17489</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.255273</td>\n",

" <td>4.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17490</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.176091</td>\n",

" <td>5.0</td>\n",

" <td>0.903090</td>\n",

" <td>0.477121</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17491</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.799341</td>\n",

" <td>4.5</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17492</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>2.075547</td>\n",

" <td>5.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17493</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.301030</td>\n",

" <td>4.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17494</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>2.450249</td>\n",

" <td>5.0</td>\n",

" <td>0.477121</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17495</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.778151</td>\n",

" <td>5.0</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17496</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.041393</td>\n",

" <td>5.0</td>\n",

" <td>0.477121</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17497</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.845098</td>\n",

" <td>5.0</td>\n",

" <td>0.602060</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17498</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.000000</td>\n",

" <td>4.0</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17499</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.146128</td>\n",

" <td>4.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>17500</th>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>1.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>...</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>0.0</td>\n",

" <td>2.056905</td>\n",

" <td>4.5</td>\n",

" <td>0.301030</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>17501 rows Ã— 29 columns</p>\n",

"</div>"

],

"text/plain": [

" 0 1 2 3 4 5 6 7 8 9 ... 19 20 21 \\\n",

"0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"1 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"2 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"3 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"4 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"5 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"6 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"7 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 ... 0.0 0.0 0.0 \n",

"8 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"9 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"10 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"11 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"12 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"13 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"14 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"15 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 1.0 0.0 \n",

"16 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"18 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"19 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"20 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"21 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"22 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"23 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"24 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"25 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"26 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"27 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 1.0 0.0 0.0 \n",

"28 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 ... 0.0 0.0 0.0 \n",

"29 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"... ... ... ... ... ... ... ... ... ... ... ... ... ... ... \n",

"17471 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17472 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17473 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17474 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 1.0 \n",

"17475 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 ... 0.0 0.0 0.0 \n",

"17476 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17477 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17478 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 1.0 0.0 \n",

"17479 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17480 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17481 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17482 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17483 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17484 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17485 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17486 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17487 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17488 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17489 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17490 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17491 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17492 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 1.0 \n",

"17493 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17494 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 ... 0.0 0.0 0.0 \n",

"17495 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17496 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17497 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17498 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17499 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"17500 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 ... 0.0 0.0 0.0 \n",

"\n",

" 22 23 24 25 26 27 28 \n",

"0 0.0 0.0 0.0 1.301030 5.0 0.903090 0.602060 \n",

"1 0.0 1.0 0.0 0.301030 4.0 0.903090 0.477121 \n",

"2 0.0 0.0 0.0 1.041393 5.0 0.602060 0.477121 \n",

"3 1.0 0.0 0.0 0.301030 4.0 0.903090 0.477121 \n",

"4 0.0 0.0 0.0 0.000000 4.0 0.602060 0.602060 \n",

"5 0.0 0.0 0.0 0.477121 4.0 0.602060 0.301030 \n",

"6 0.0 0.0 0.0 1.380211 4.5 0.602060 0.602060 \n",

"7 0.0 0.0 0.0 0.000000 4.0 0.602060 0.301030 \n",

"8 0.0 0.0 0.0 1.255273 4.5 0.602060 0.301030 \n",

"9 0.0 0.0 0.0 1.204120 5.0 0.778151 0.602060 \n",

"10 0.0 0.0 0.0 0.000000 4.0 0.698970 0.301030 \n",

"11 0.0 0.0 0.0 0.000000 4.0 0.602060 0.301030 \n",

"12 0.0 0.0 0.0 1.414973 4.5 0.477121 0.000000 \n",

"13 0.0 0.0 0.0 0.477121 4.0 0.602060 0.602060 \n",

"14 0.0 0.0 0.0 0.000000 4.0 0.602060 0.477121 \n",

"15 0.0 0.0 0.0 0.000000 4.0 0.602060 0.301030 \n",

"16 0.0 0.0 0.0 1.255273 4.5 0.602060 0.301030 \n",

"17 0.0 0.0 0.0 1.342423 4.0 0.602060 0.301030 \n",

"18 0.0 0.0 0.0 1.079181 5.0 0.778151 0.602060 \n",

"19 0.0 0.0 0.0 0.301030 4.0 0.778151 0.477121 \n",

"20 0.0 0.0 0.0 1.146128 4.5 0.778151 0.602060 \n",

"21 0.0 1.0 0.0 1.799341 5.0 1.204120 1.000000 \n",

"22 0.0 0.0 0.0 0.301030 4.0 0.778151 0.477121 \n",

"23 0.0 0.0 0.0 1.342423 5.0 0.602060 0.477121 \n",

"24 1.0 0.0 0.0 0.000000 4.0 0.602060 0.477121 \n",

"25 0.0 0.0 0.0 0.000000 4.0 0.602060 0.698970 \n",

"26 0.0 0.0 0.0 1.230449 4.5 1.000000 0.602060 \n",

"27 0.0 0.0 0.0 0.000000 4.0 0.602060 0.477121 \n",

"28 0.0 0.0 0.0 0.477121 4.0 0.903090 0.698970 \n",

"29 0.0 0.0 0.0 0.301030 4.0 0.301030 0.000000 \n",

"... ... ... ... ... ... ... ... \n",

"17471 0.0 0.0 0.0 0.000000 4.0 0.301030 0.000000 \n",

"17472 0.0 0.0 0.0 1.000000 4.5 0.301030 0.000000 \n",

"17473 0.0 0.0 0.0 1.176091 4.5 0.301030 0.000000 \n",

"17474 0.0 0.0 0.0 0.698970 4.0 0.301030 0.000000 \n",

"17475 0.0 0.0 0.0 0.000000 4.0 0.000000 0.000000 \n",

"17476 0.0 0.0 0.0 0.301030 4.0 0.301030 0.000000 \n",

"17477 0.0 0.0 0.0 1.544068 5.0 0.000000 0.000000 \n",

"17478 0.0 0.0 0.0 0.602060 3.5 0.000000 0.000000 \n",

"17479 0.0 0.0 0.0 1.763428 4.0 0.000000 0.000000 \n",

"17480 0.0 0.0 0.0 0.000000 4.0 0.000000 0.000000 \n",

"17481 0.0 0.0 0.0 1.079181 3.5 0.301030 0.000000 \n",

"17482 0.0 0.0 0.0 1.380211 5.0 0.477121 0.000000 \n",

"17483 0.0 0.0 0.0 0.698970 4.5 0.000000 0.000000 \n",

"17484 0.0 0.0 0.0 0.602060 4.5 0.000000 0.000000 \n",

"17485 0.0 0.0 0.0 0.301030 4.0 0.301030 0.000000 \n",

"17486 0.0 0.0 0.0 2.206826 4.5 0.602060 0.000000 \n",

"17487 0.0 0.0 0.0 0.000000 4.0 0.000000 0.000000 \n",

"17488 0.0 0.0 0.0 0.000000 4.0 0.000000 0.000000 \n",

"17489 0.0 0.0 0.0 1.255273 4.5 0.301030 0.000000 \n",

"17490 0.0 0.0 0.0 1.176091 5.0 0.903090 0.477121 \n",

"17491 0.0 0.0 0.0 1.799341 4.5 0.000000 0.000000 \n",

"17492 0.0 0.0 0.0 2.075547 5.0 0.301030 0.000000 \n",

"17493 0.0 0.0 0.0 0.301030 4.0 0.602060 0.000000 \n",

"17494 0.0 0.0 0.0 2.450249 5.0 0.477121 0.000000 \n",

"17495 0.0 0.0 0.0 0.778151 5.0 0.301030 0.000000 \n",

"17496 0.0 0.0 0.0 1.041393 5.0 0.477121 0.000000 \n",

"17497 0.0 0.0 0.0 0.845098 5.0 0.602060 0.000000 \n",

"17498 0.0 0.0 0.0 0.000000 4.0 0.000000 0.000000 \n",

"17499 0.0 0.0 0.0 1.146128 4.5 0.301030 0.000000 \n",

"17500 0.0 0.0 0.0 2.056905 4.5 0.301030 0.000000 \n",

"\n",

"[17501 rows x 29 columns]"

]

},

"execution\_count": 112,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"xlog\_df"

]

},

{

"cell\_type": "code",

"execution\_count": 113,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.tree import DecisionTreeRegressor"

]

},

{

"cell\_type": "code",

"execution\_count": 114,

"metadata": {},

"outputs": [],

"source": [

"dt = DecisionTreeRegressor()"

]

},

{

"cell\_type": "code",

"execution\_count": 115,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"DecisionTreeRegressor(criterion='mse', max\_depth=None, max\_features=None,\n",

" max\_leaf\_nodes=None, min\_impurity\_decrease=0.0,\n",

" min\_impurity\_split=None, min\_samples\_leaf=1,\n",

" min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,\n",

" presort=False, random\_state=None, splitter='best')"

]

},

"execution\_count": 115,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"dt.fit(xlog\_train,ylog\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 116,

"metadata": {},

"outputs": [],

"source": [

"ylog\_pred = dt.predict(xlog\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 117,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"52.70946154269602"

]

},

"execution\_count": 117,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"mean\_absolute\_error(ylog\_test,ylog\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 118,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"90.60194537324536"

]

},

"execution\_count": 118,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": 119,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"52.94523707611961\n",

"91.01618162337459\n"

]

}

],

"source": [

"#Decision tree regression using cirterin as friedman\_mse\n",

"dt = DecisionTreeRegressor(criterion=\"friedman\_mse\")\n",

"dt.fit(xlog\_train,ylog\_train)\n",

"ylog\_pred = dt.predict(xlog\_test)\n",

"print(mean\_absolute\_error(ylog\_test,ylog\_pred))\n",

"print(np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred)))"

]

},

{

"cell\_type": "code",

"execution\_count": 120,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"52.348124166825365\n",

"88.61729431893679\n"

]

}

],

"source": [

"#Decision tree regression using cirterion as mae\n",

"dt = DecisionTreeRegressor(criterion=\"mae\")\n",

"dt.fit(xlog\_train,ylog\_train)\n",

"ylog\_pred = dt.predict(xlog\_test)\n",

"print(mean\_absolute\_error(ylog\_test,ylog\_pred))\n",

"print(np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred)))"

]

},

{

"cell\_type": "code",

"execution\_count": 121,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"44.93696438773567\n",

"79.85195666865407\n"

]

}

],

"source": [

"#Decision tree regression using cirterion as mae using max\_depth=3 and min\_sample\_leaf=5\n",

"dt = DecisionTreeRegressor(criterion=\"mae\",max\_depth=10,min\_samples\_leaf=20)\n",

"dt.fit(xlog\_train,ylog\_train)\n",

"ylog\_pred = dt.predict(xlog\_test)\n",

"print(mean\_absolute\_error(ylog\_test,ylog\_pred))\n",

"print(np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred)))"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": [

"# Cross Validation to better understand the accuracy on decision tree"

]

},

{

"cell\_type": "code",

"execution\_count": 149,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.model\_selection import cross\_val\_score\n",

"from sklearn.linear\_model import LinearRegression"

]

},

{

"cell\_type": "code",

"execution\_count": 150,

"metadata": {},

"outputs": [],

"source": [

"lm= LinearRegression()"

]

},

{

"cell\_type": "code",

"execution\_count": 151,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)\n"

]

}

],

"source": [

"print(lm)"

]

},

{

"cell\_type": "code",

"execution\_count": 152,

"metadata": {},

"outputs": [],

"source": [

"scores = cross\_val\_score(lm, X,Y,cv=10,scoring='neg\_mean\_absolute\_error')"

]

},

{

"cell\_type": "code",

"execution\_count": 154,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[162.75865697 61.10659055 46.93939108 37.75342272 37.9775013\n",

" 32.13648873 35.29429125 56.83694596 33.75246922 60.76165455]\n"

]

}

],

"source": [

"print(-scores)"

]

},

{

"cell\_type": "code",

"execution\_count": 155,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"56.531741233723736\n"

]

}

],

"source": [

"print(-scores.mean())"

]

},

{

"cell\_type": "code",

"execution\_count": 148,

"metadata": {},

"outputs": [],

"source": [

"# Cross Validation to better understand the accuracy on decision tree"

]

},

{

"cell\_type": "code",

"execution\_count": 134,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.model\_selection import cross\_val\_score"

]

},

{

"cell\_type": "code",

"execution\_count": 158,

"metadata": {},

"outputs": [],

"source": [

"dt = DecisionTreeRegressor(criterion=\"mae\",max\_depth=10,min\_samples\_leaf=20)\n",

"scores = cross\_val\_score(dt, X,Y,cv=10,scoring='neg\_mean\_absolute\_error')"

]

},

{

"cell\_type": "code",

"execution\_count": 159,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[174.8195317 82.09342857 50.02485714 41.15742857 38.746\n",

" 29.88 29.69971429 48.826 32.32657143 52.248 ]\n"

]

}

],

"source": [

"print(-scores)"

]

},

{

"cell\_type": "code",

"execution\_count": 160,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"57.982153169617376"

]

},

"execution\_count": 160,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"-scores.mean()"

]

},

{

"cell\_type": "code",

"execution\_count": 142,

"metadata": {},

"outputs": [],

"source": [

"from sklearn.model\_selection import GridSearchCV"

]

},

{

"cell\_type": "code",

"execution\_count": 143,

"metadata": {},

"outputs": [],

"source": [

"max\_depth\_range = list(range(1,5))"

]

},

{

"cell\_type": "code",

"execution\_count": 144,

"metadata": {},

"outputs": [],

"source": [

"param\_grid = dict(max\_depth = max\_depth\_range)"

]

},

{

"cell\_type": "code",

"execution\_count": 145,

"metadata": {},

"outputs": [],

"source": [

"grid = GridSearchCV(dt, param\_grid, cv=10,scoring='neg\_mean\_absolute\_error',return\_train\_score=False)"

]

},

{

"cell\_type": "code",

"execution\_count": 146,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\model\_selection\\\_search.py:813: DeprecationWarning: The default of the `iid` parameter will change from True to False in version 0.22 and will be removed in 0.24. This will change numeric results when test-set sizes are unequal.\n",

" DeprecationWarning)\n"

]

},

{

"data": {

"text/plain": [

"GridSearchCV(cv=10, error\_score='raise-deprecating',\n",

" estimator=DecisionTreeRegressor(criterion='mae', max\_depth=10,\n",

" max\_features=None,\n",

" max\_leaf\_nodes=None,\n",

" min\_impurity\_decrease=0.0,\n",

" min\_impurity\_split=None,\n",

" min\_samples\_leaf=20,\n",

" min\_samples\_split=2,\n",

" min\_weight\_fraction\_leaf=0.0,\n",

" presort=False, random\_state=None,\n",

" splitter='best'),\n",

" iid='warn', n\_jobs=None, param\_grid={'max\_depth': [1, 2, 3, 4]},\n",

" pre\_dispatch='2\*n\_jobs', refit=True, return\_train\_score=False,\n",

" scoring='neg\_mean\_absolute\_error', verbose=0)"

]

},

"execution\_count": 146,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"grid.fit(X,Y)"

]

},

{

"cell\_type": "code",

"execution\_count": 147,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"-59.136820753099826\n",

"{'max\_depth': 4}\n",

"DecisionTreeRegressor(criterion='mae', max\_depth=4, max\_features=None,\n",

" max\_leaf\_nodes=None, min\_impurity\_decrease=0.0,\n",

" min\_impurity\_split=None, min\_samples\_leaf=20,\n",

" min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,\n",

" presort=False, random\_state=None, splitter='best')\n"

]

}

],

"source": [

"print(grid.best\_score\_)\n",

"print(grid.best\_params\_)\n",

"print(grid.best\_estimator\_)"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": 122,

"metadata": {},

"outputs": [],

"source": [

"#tyring with Random Forest Regressor\n",

"from sklearn.ensemble import RandomForestRegressor"

]

},

{

"cell\_type": "code",

"execution\_count": 123,

"metadata": {},

"outputs": [],

"source": [

"rf = RandomForestRegressor()"

]

},

{

"cell\_type": "code",

"execution\_count": 124,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\ensemble\\forest.py:245: FutureWarning: The default value of n\_estimators will change from 10 in version 0.20 to 100 in 0.22.\n",

" \"10 in version 0.20 to 100 in 0.22.\", FutureWarning)\n"

]

},

{

"data": {

"text/plain": [

"RandomForestRegressor(bootstrap=True, criterion='mse', max\_depth=None,\n",

" max\_features='auto', max\_leaf\_nodes=None,\n",

" min\_impurity\_decrease=0.0, min\_impurity\_split=None,\n",

" min\_samples\_leaf=1, min\_samples\_split=2,\n",

" min\_weight\_fraction\_leaf=0.0, n\_estimators=10,\n",

" n\_jobs=None, oob\_score=False, random\_state=None,\n",

" verbose=0, warm\_start=False)"

]

},

"execution\_count": 124,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"rf.fit(xlog\_train,ylog\_train)"

]

},

{

"cell\_type": "code",

"execution\_count": 125,

"metadata": {},

"outputs": [],

"source": [

"ylog\_pred = rf.predict(xlog\_test)"

]

},

{

"cell\_type": "code",

"execution\_count": 126,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"49.13658699860956"

]

},

"execution\_count": 126,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"mean\_absolute\_error(ylog\_test,ylog\_pred)"

]

},

{

"cell\_type": "code",

"execution\_count": 127,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"83.41102545906939"

]

},

"execution\_count": 127,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": 128,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\Users\\abagarwa\\AppData\\Local\\Continuum\\anaconda3\\lib\\site-packages\\sklearn\\ensemble\\forest.py:245: FutureWarning: The default value of n\_estimators will change from 10 in version 0.20 to 100 in 0.22.\n",

" \"10 in version 0.20 to 100 in 0.22.\", FutureWarning)\n"

]

},

{

"data": {

"text/plain": [

"81.94737778398587"

]

},

"execution\_count": 128,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"rf = RandomForestRegressor(criterion='mae')\n",

"rf.fit(xlog\_train,ylog\_train)\n",

"ylog\_pred = rf.predict(xlog\_test)\n",

"mean\_absolute\_error(ylog\_test,ylog\_pred)\n",

"np.sqrt(mean\_squared\_error(ylog\_test,ylog\_pred))"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.7.3"

}

},

"nbformat": 4,

"nbformat\_minor": 2

}