```
import pandas as pd
import numpy as np
```

Import dataset

```
from google.colab import files
files.upload()
```

Choose Files Housing_dataset.zip

- **Housing_dataset.zip**(application/x-zip-compressed) - 11983 bytes, last modified: 4/5/2024 - 100% done

```
Saving Housing_dataset.zip to Housing_dataset.zip
```


```
{'Housing_dataset.zip': b'PK\x03\x04-
\x00\x00\x00\x08\x00H[5O\x8b\x8bs\x97\xff\xff\xff\xff\xff\xff\xff\xff\x0f\x00\x14\x00Ho
[\xae1\xb9m\xfd7\xe0\x99\x14\n"\xf5\xfe\xec\xc4F\xd2@\xdc\x0e\xecN`d\xfe\x03\t\xd7\xa2\
_\xbf\xfd\xfd_\xaf\x7f\xfc\xed\xf5\xcb\x7f\xfc\xf5\xf5\x97_\xff\xf9\xfa\xc7/\x7fy\xfd\x
~\xf9\xfd\xd7\xbf\xbf\xfe\xed\xf5_\xff\xfc\xfd\x97\xdf_\x7f\xfb\xeb_\xfe\xf7\xcf\x7f\n\
Y\xdf\xa1\x12 \x95n\x00I\xe5U\x9b\xffh\xa9\x06`?n\xbf\xa8\xbe\xdb\x80\xe8oI/\x95w\x19
\xdaw\x90\xfa\x96fT\xc8[\xae
\xfan\x11\xe4\x19M\xe9]\x89\x125\xe2\xe3j\x1ft\x94\x8c\xb3t\xa35\xe57\xfe\x18\x8a\xea+\
\x9a\x91\x03!\xec\'\x94I\x81\xdd\xbf\x80m<\x9bjj\xfdz\xb6X\xeb\xcb\xd8\x1a\xf1\xc7T\xea
\xd5\x1d\x87<\x8ao\xad\x0fA\xa1\xc7\xda\xae<jFh\xec\xf6\x87\x94\rr\x81\xb1\x1f\xc1\x1f!
<\x16!b\xed\x0f\x84s\x98o;\xd8\xbf\xb6d\xbf1\xd93\x93\xfe\x85\xd1
\xbav\x8f@\xb1\xf3\xa8\x13Rc?
\ti\xd1\x8eT\x1c\xa4\xb7\xf2M\x88I\x83\xda\x8f\x83\x9d\x1d<\x91w\xc8\xd1.\xff<L\xcc\x90
\xaa\xe6vbT\x93\x80\xd2\x8d\x1d;\x06\xee\xb6g`h\x1b<\xb5?
f\xa9\x17\x9ef\x13X\xfb\'b,\x87\xa9P\xc8\x10\r\xdcNt\x95\xd9\xb2\x86ta\x88\xfd\xf2\xd6]
\xa6\x0f\x8c\xd5\x1dl\x8do\x93\xc2oJ\\\xe0MC)E\x17\x84\xf4\xd6\xd2\xe5r\x9bDj\x04\x18\x
\xc9d\xc9\xce\x9e\xfc4\xf5[\xe0+\x8c\x98\xe9\x1d\x8aY\xe2\xb3\x0b\x9a\xf2\x89Q\xaai\xd8
\x8a\x87,j\xcc\x18Z)\xa5\xb0\x0f\x94\x0e\x83h\xb5_\x9e\x8b\xb1\xc2\xee"\xe2\x89~\x9el\x
b\x86\xc5\x04G\xa7\xb7\xa6\xd5\xfc\x02\xbb|\xc5M\x12G\xe9z\xda\xdf\x14&\xce4s0\x13i\x9f
8%\x10\x83C\xbf;\x15\xbc\xd9l\xc3\x80\xd6\xa9\xb8q\xd3)\t\x0f\xa0\x99\x17;\x1e\xbe$\xc9
s\x95\xf1G\x81\xaf\xb6PB\xeb\x00?
\x11\xf7\xee\xe7i\xeds\x9e\x0fJ5so(\xf1g\x94D\xc73\xb8\x8f\xd7\xb5~I\xbd\x9f%B\xadgR\xd
(\x08C\x1be\xbb\x97\x9d\x12\xd8B\xfa\xa0rE)|\n\xf6\xbc\xd4 \xecW_\x0eB\xd3\xf1N?
~\xbc\x91\x86\x19j\x89\xd1@\xf5O\x04\xa053\xa7\xa9\xbe!\xcf\xd1\xf8m\xfcD(H\x8a\x1e\xc1
(p\xa2{\xb0\xb7\xac\xea &\x17fH\x13?
\x83_W\xa0\x97L\x92\xe0\xd2\xfa\xad@\xe5\xbeh\xb1c\x1eth0\xc1G\xc8\x06\xc1\x19\xb4\xc0(
\xb49\x89\xb1\xab\x81i
\x93\x8c\xfd\xdf8\xb5\xd9k\xe1\xdb\xb3\x7fqj\xc4L\xc6\x853\xf0\xdf\x0br\x02\xa4&\x9fg\x
7\xf6V{\xc2\t\xa4\x98\x8c4D|\xeb\x91\xcc\xe7\x87w\xaf\xfet\xa4g\x86\xa1\x19W="|)\x82,\x
\xbaC\x02.0\xe6\xc3\xdaq\x19\xe0\xccp>t\xa1\x9e\xdc\x89\t\x85i\x82\x1b\n\x02GhD$*\xd4}\
<\xc7\x1b-
\x15\x1e9t\xf5\xd4\xb9v\xcf<\x10\xf3g\xae\xb9\xc1\x10\x0b\xf3\x1c\x08\xa4\x85\x89\xa0\
\xe7\xf9\x89\x90U\x17\x98j\xd1/\xfdY3\r\xd3\x9f\xad\x94\x90\x1d\xc5\x9c\\&\xfd\x10\x7f\
[L\x01f\x13\x0b\xe3\xad)x\xd9q(\xed1=NSh\xeeg\xac0xD\xc9s\x0c\x17\xd6t\x88\x01\xaf(\xb8
Cq\xa8\x9f&\xa5\xa2\x17\x04\xa4{\xaa\xfd\xe5\x82@\xcf\x1a\x1a\x14\x92\xe2\x0f(\x16\xf3\
\x0c\x05\xa7q\x85]-
=a\x04\xce\x14\xfd$3\xaf\xa6r6\x18\xfb%\x90\x90a\xc4LZ\xc1\xcd\x08?
\x85\x10\r\xe9r0\xb2!%\x82\xc4\x06\xe2p\x8b\xc0\xc2w\xce\x1d\xfe\xb2\xc6y\x1a\xdc\xe3\x
(v\xd5\x95\xfe\xa3=\xc5\xbc\x83\xd0&gZ0wx\xd4S|&>SR\xa0\xe7\x0b\xbc\x17r\x01\xe9
%_\xe0c\x98\xaf\n\xe7\x05?
\x81#\x0c%+\xf9\x86!\x88\x13:\xd9\xb2ata\x82P\x86V\x1b\xf68P\x9b-
\x8d\xe1d\xbdA\xb8\x93\xc2\x94\xd2\xaceH\xab;\x80\x1dA\xa1\x14\xfb\x05!\xe2\xe1Ed=\xcd)
\xef\xc8*\x80\x12V\x9b>
\xa6\xb3\x1b\x84\xc3n\r\xbe$1\x04\x1e\xe7\x8e!\xd1\x8d\xfd\x81\xc0,Ae\x18\x99>\xc1y\xec
(j\xd9\xd9\x81d\x92 /\x86\xdc\x89@H6:\xe0\x9c\x94Q\x1f\x18\xe5
\xb9\x80Xl\xd8\x85\x17\x83\xc7\xb7\xb1\xb4P7\xc2\x17\x98\x18\x16\t\xd6\xf30v\xbb\r5\x16
\xd0\x92S\xdeN$L\xbdfh\xb5\xe9\x92\x8b\xe9\xe2\x13\xc5|@\xd4!\xd3U^\xf1\n^\xf0\x1d\x86\
\xb6b:\x1b\x1e\xc1\xc1\\\x84\xa0(I\xc0\xcfwR\x82\x91\x02\x13\xb8\x9d\t\xe9\xed\x1a]z\xf
(w\xb1~\xde@J3\xe4\t#\x07\xe9@(\xf4\x17\x7fL5#$\xf8&\x08\xc9c\xd4\xa6>\xa9?
\xd3\xd8L\xdbk\x86A"=\x8d)\x1e\x96\xca*o\t\xf9\x93\x17\xdc*\x14\x9a\x99\x86\x14V\xed@\x
i3\xbb,F\xfc#\xf16Nd\x0f\xd4\x02\x12SV)m\xa4\x04\xd6\x01\x98+F\xe2\xc8Ii\xe9\xce\x17&#\
0\xcdU\x95\xa0\xb5D\xfdH\xda\xcd\xe7\xb2K\x1d\xe2\xeb|i\xfah\xdfT6\xbe\xb4F#\x99\x91o\x
\xd3\xdd\xc9\x18q\x92\x12\x13\xb5\x1e\xab\'v,\x15\n0\xfc5c\x99\xa1\xb4*\x0c\xff\xe3\x93
\xe1\x03R\xda$\xc5\x9e\xe4\x15$\xe9\x10<\xf8M7\x14\xb3\xee\xf6\xa1^\xa9db\xca\xfd\x02C\
\xb4
\x81\xf7\x01\x11\xf3\x08\xec_\x94\xbf":\x88d\x14S!\x18\xdb\x91\x8a\xce#e)\x0b_L\xef\xd3
0\xe3a\x8f\x03A\xc5d*\xbcT\xc2\xc6\x17a\xbbQgSH\x1a
1\xd0H~\x81d\x8b\xd6\x99\xa4\x07W2\xda\x00\xbfArA~C\xdb{\xf8\x01\xe4n\x8c\xbd^\xee(wj\x
\xde7\x90za\xacvH\x0bT\xb4\xb3\xb6o"\x97\xe0\x14\xa3q\xa0\r\xcd+,\x18\xb4\x07\x85\xee\x
8Z\x84.t~
{!\x9dlI\x12\xdb\xf9\x8e\xc0\x97>\xecl\xeai\xa1F\xbdo(\x93\x9a\x8a\xb7\x98c\xa6\x87x`\x
\x94\xb4\x81\x14\xd6\x02%=\x8d\x15\xaaq8\xad\x0f\n\x1a\x1a\x91@\xe7[\xa4\xd6\xad\xec&\x
[\xc1\xce\xeb\x15\xf4\xf0
\xeb\xa2\xecG\xfcz:\x89\xf1k\xf7\xd6\n\x00\x04\x81\xbd:y\xda\xf9n\x06;\xf6\xdb\xedL\xd5
"n9B\xb8\x9d&J\xf7n\\}/vC\xbb\x87\xbf\xa8\xe6M\x88\x94\xba\x17O6\x8c\x14\xe8\xcb1\x06f\
p\xf4\x9e\xd0=\x1e\xfb\r\xa6\xe1\x15\xe3\x86L2VYK\x8d\x96R\xe9Y\xf6\x99N\xealNb\x8e\x08
\x94\xb8\xb2\xef\xdb\xa9p_\xff\x9dF\x19\xa8\xb15\xbf\xa1#\x1e\x18\x8d\xc5r\xe9q\xadW\xa
*T\xf2\x0b}T\xcc\x92\xc8\x8a\xc1\xce\x19v\xc0\xc5\xfed\x91\xdb\x85\x0e\xb1\xb0\xbd\xa8\
[\xf1\xae\xe8\xd6\x9e\xca\xa9\xf7\xcdG&\x93\x9c-
\xd4\x1f\xde\x05~\x80d\x13\x01\xed\xac\xd1\x99WY\x17\x90\x06n\xf3\x7f\xe0G\xda\xe0m=P|\
(\xf6\xa7\xa30t\x13\xb6\x0b-
\xe8\x1d\x81\x87\n\xadT{[P\x90\xb9bg\x9d\xfd\x912r\xe1G\xf4\xbe\xb4\xdb\xc7\xed\xe2\xe9
\x18\'\xbe:T\x1f\xd1,\x00!3z\x1b\xbd\xfe\xd9*\x86:\xb3\xe2\xa8\x88\x1a\xccL\xa1\x10\xf0
(\xfd\x81\xd6Q4\xb5\x17\x88\xe8\xe0\x9df\x83\x0cn\xb4W2\x10\x18\x05\x19y{c\xf0\xd9\xebs
\xd0S\xdda\\\xaex\x89\x11\x12\xc9\xbe!\xe4aa\xf0\xa7\x7f\x17\xd17W\x87\xe7-
C\xc8\xec\x9fN\x18w\x9e\xbc3rG\xc9\x10\x8e\x08u\x94t26\xb3\xa7[\xdf\xa5\r\x08ve5\x14\x1
\x18\xe8\xc9W\xda\x0bs\xcbe\x05\x89\xcc\xfaU\x8a\xd2(\x91Y$C\xcfd\x87\xc1\x93lm6\x0e\xa
\xfc\x05\xe1\x12\x8b\xcc\xc2#\xf8E#of\x05\x81\xcfXQ\x95\x80\xef\x87\x02\xe47J\xf0!\x01\
\x0c\xffq\xc7(nW`\x0c={o\xdef\\
\xd0#\xab\xa6,j\xf3\x86\x8a\x9e\xd2\x02\x81\x1a\x16\xd4\x1e:\xfcF\x9b\x99\xbd\xac~\x82T
m\xcf\x8aU\xc1o\x84\xca&\x1b\xb6\x96\\
```

```
*\xea\x8d\x90\x9d\xf2t~ko\xd2\x0f\x10T\x0b`z\xa1\x01VVV6\xc9\xc1\xc5\xa3X0\xa2\xf8\xfe,
>\xdfYE\x07\xa3K\xc6\x01R0\x00\xc5\x16o\xedNDH\xf9x
\xc1\xee\xb4\xc8\x1d\xa1\xfa\xe0j`\xf7p\xfa\xba\xd6\x0f\x01hL\xa8>
{+\xdbC\xf7\xf4[\xc7\x84\xa5f\x7f`\xd9|\x95\xb8\x91P\xa1\x86\xd0\xb1T.\x18\x1d\x1a\x17m
\'\xf7p\x0f\x18\xf4d#\x98Dmi\xce6\x08\x1c\xe3\t\xe3>\x11qZ\xe7\x8c>G\xbec\x91\x05\x079\
c\xdb\x18\x7f\xf0\x91\xd8\xa1\xbeqL\xf0\x04F[\xca\xa3N,*\xc4\xe4\xf3\x01\x83q\x1ef\\/0\
S\xfcB\xdc\xfbq\x159\xb3Fo~\x84\xa3\xe8\x1f\x9b(a\x80
\x9d]\xfc7\xdf`\xb2\x8b\x9d\xe6\xcf\xec\x88`\xc0\xb7\x03\xa6\xe6\x01C;f\x04G_\x02B\xfd\
\xcf\xf44}\x824\xba\xa5\x9af\xee\xc68\x98\xa9,K3\xefP\xdb7\x11\xdd\xf7\x03\x88{&
<GF\x8b\xfd\x8d\x17lUEm\xaf\xc7\xf5J+D\x94\xfe\xd3\xe8\xfd\xad\x05\x89\x91\x1d\x83\x99\
<0\xe8\xcaK\xfa\xf6\xc1\x8c\xbe\xc0 \xb0\xcf\xa3Q\xd6-
yQ\x97\xf2\x05\xa3\xb0\xb5\xb7x\xcek\x83\xa0\x06\x83X\x8e5\x18\x16\x18\x96\x0b\x00w\rp\
<(m\xa8md{\n\xe7\xcd6\x14\xe5d\xbd\xb3%$\x1f:_(\xc1\xcb6\x17\x13t\x8c\xed\x0b\x1f\x84FM
\xf0I\xd0\x11\x8e\xf0\xbf0\x893\xb2\xdc\x1c\xd1`\x11\x80\xe91\x97\x16\xc1^\x0e\xa6cu\x1
%h\x9cFe\xfb\'\x08v0\xd49"\xac\xecxF\xb1c0\x8556\xf1}^\x00\xc9\x9897\x10\xa4\x0c\'g\x1b
\xb0\xe8O.\xce\xde\x9fw\xaam\x84\x14\xd4&P\xfcDZ;\xa4\x1d\xc3k"\xed\x19\xe8\xab7\x0c\xf
ha\xa4\xba\x82\xe0)D\xe87DUY\xd8T\xfd\x01\xc9\x81\xed\xb1\xe8a\xe2\x95\x83\x92\xda\x03\
\xb9\x8a,>\xda+\x8c%\xea\xfe\xd9\x17H\x84\xe6\xe1\xe4\x82\xcc4\x02\xf5\xd6\n\x91\xedE\x
\xbf\x855r\x7f\x1b\xe2#\xaa\xa2B\x1f\xa8>\xcbN\x98#H^\x81\xa63\x89j\xa9\xa9>$\xb0\xfa\x
"\r\xedl\xf1\x8a!\xcco\xca\xd8*\xe5\xf9M3\x95\x97\xf3\xa0g\x01\xad\xa7?
\x92R8\x8c3\x92\x89\xd2nl\x8d\xe2M\x05\xa8\xd6\xb2\xf3\xf2\x0b"z\x7f\x90Oh\xf8\xab\t\x9
&\xb1;;\xaf\xd2\xd0\x8a\xf2\xcc\x0cVM\x1bH\xf1\x84W}\xeb\r\xa4R\xe6+\xc7\xd5g\xefe\xcai
\x94\xfd\xbd\xc6}\x03*\x0c^\xeaX\xd6\x99\x06A,\xe40\xeb\xcf\xa7,\x1e\x08
hf\xcd"e6\x96\xa3-E\xb9\x89\xcf\x1b\xc2\xd87\xf8\\S\xac\xbc\xe5\x03\x05
\xf9\x07\x10\x9c\xb5~2.!v\xce\x9eo\x18\x06\x8d\xec(\xab\xea\xcd[B\x1f\x90\xe18\x07yR.\x
\xbe\x1ad\x87(\xcc!\xe5?\x80\x80\x178w-
\x08R\x90(\xcc>\xdde\x89N\x9fkI\xc4\x9c\x01\xf2\x87(:q\xe4\x9c\\\xa9\xee\x97\xe2\x85\x8
@/t\x85k\xf45\x03\xbe\xda\xe3+\xfe\xfd\n\xb1\xec\x96;P\xa6\x03\xda\xdc\xa7Olw\x81\xc8\x
\xcd\xd7\xe4N-\xb6\xba\xfa\\\xdf\x02\x12\xb1\x90\x18\x9d\x179\xe4\r\xe2\xc9\xd5?
\xddv\xa6\xd2\xd1sz\x80pKG$%U\xf1\xd3\x0bL\xa61\x95w\xa2\x11\xab\x18\x03\xb5\x90\x85\x1
\x86Bp\x10\x8c\xe3\xf9\x0e\x96\x1c@\xd2\x02\xc2\xcd\x8e\xfe\x06\xddM\x80\xc3)\xf4\x04IK
\x18\x7f\xb5\x9d\'\xc9\xe7\t\xe0\xf4dr$\xfa\xfa\xa3\x15A\xd1Q\xc6Y\x1dS\xcbHC~c\xe4@\xf
{f\xdc\x81\x7f\x85I\xdeI\xa7}\x94a\xf1\xf7\xd1\'\xfa\x9c\x08+y\xde\xc8\xed\xb41\xbd,Q\x
P7\x0bH\xaa\xe8gj\xb3?
\x1a\xa3l\xbeY\xfcCH\xe1\xdd\x86\x87\x10\xcc\xa5\xadg\xa1c\x9c\xb8\x9d\x9bQ%\xc2\xd5\xd
\x7fP*#\n\x89s\xd6>\xa2\x95\xef\xc4\x19\xdb\x940l\x8c\xd5\x8b\xcf\xbbq1\xe3\xe8h1\xdb\x
\xd8o7\xb6_
oy\x12\x82\xa9{\xaaZ\xa4\x96\xd0\xfa\xb0q\x85(\xe8h`#\x99\xfd\xa9\xc8\r%r\xd7\x83\x8f\x
5d\xb0\x19\xeb\xb0\xaaZX\xca8\x85.\xb7\x90~`\x0c[\xc7\xfe\xfc\xa7\x8a\x9d*\xa9\x1d\x10:
R\xea\xf6x\xd1A\xdb\xf0\xf3\xd8.\xed_W`\xf4\xd4\xdawJ\xe0\x00\x90\x92z\xd3g\xca\x08\xbf
\xf2\xbcaz\xdc"\x8c\xd3^\x9fMTW!\xe8\xf1\xca\xf5 \xa5|-
\xd8(q3yh\xced\xce$z\x0cEE\xd4\x82\xd4\xf3@0\x85\xbe\xd1\x02c\xf7\xd7[\xae\x0cq\xcd=\xe
wo\x00\xfa\x18\x89-
\xc0\x11l*c\xafem\xc3\x91\xcf\xc2\xb9\xb1\x93\x10t=\xcfM9\xab{Ee\x15\\\xb7RZ\x03\xba\xd
a\xe5\xbe\xd4\xf6\xaa<\x8e9\x12e\x81\xa8\xbe\xcfd>\xbe\x16\xb7\xdb\xc5\xc66f\x86\xb8\xf
\xc1\xbb+\xcc \xa5\xe1\xe5\xd5\xc8N\xd1\x13\xa7?
\xf7\xdc\xb0<{\x81)o\xef0\xe0w\x8e5>\x9d&k\x8cS\x99\xb9l\xf61\xf6\xc1\x0b\xfd\xe9o\x90\
\x08sFN\x89\xdd\xf3\xf6t\xda\xdb\x87o\x03\xc9\xaax\x82A5\xf7\x13\x06\xeb\x010\xbf\n\x9c
<GLoa#\xfa\x1a\xfa\xb1\xad\xc6\xae\xb06\xe2\xb4\xa2\xeb=C\x8f(\xf8J%\xe8\x9a6Z4r\xc8\xa
>i\x86\x1dv\x9bB(alZ\x0fl\xb7\xbeq\x04\xf6\xbc\xa1\xd9\x1c\xc9\xa8\x90\xb7\xd4Zc\x8b$r|
F\xff2E\xe5\xf7*\xac0\xc2\xd2w\x1f\xdbt\x05\xeetJiu\xdc\x1c\x06\xb3\x935\xb11AwW\x07&)\
Y(\x12\xe6\x1de\x83\xc8\xfcR\xc6\x91\xa3k\xba%\xc6\x8a\x8f\xc8\xb1\xc6I=\x89o\x8d\n%\x1
\x05M\xf5\xd1A\xe4\x1eFrV^\xd8P\xa3\x90M9Ii\x18\xcb`\xf7;r\x11gv\xd9\xfb{=\x14\xb5G\xaf
\xe8\xc9Z\xee\xcd*\xd4\xf9\x16\x1c\xef*.\x0c\x85\xed\xc9\xe1\xc6\x17]\xea\x8d/\x99R\xeb
u\x92\x0f\x14\xf5u\xe1|A}c/\x86\xa2\xdd\xca{\xfa\x9e27\xd6Zn\xaf\x19\xebY\xbaohBK\xd3E\
{\x1f\xc6\xba\xc4\x94\x86N\xb0\x0bO\xde\xd9\xb2\xd1R\xf2X\xb3\x16\xf3\xee\xed\xe7<\xbe\
_e\x88Lb\x8a\xffr\xa6\x8c\xd9\xee\xc2\xb4J\xdas\xa29 n\xff\xbel,\x9fh\xed
\x06M{\x8f%\xc9m/\x03\xa0\xd2\x8c\xcb\x0e\xf2$6\xccj\xad\xde\x7f}\x8aq\x18\xa8~q\xb9\xe
\xeb\xbcl&$\xecH\xdeF\xfe\xc5_{K\xa8\x15F\x8aom[\x12Q\xabR\xf0\xbc3\x8b\x9c)\x88\x9ao\x
[Q\xc8\xbe\x1c%\xf2\x0bV\xe8\xfe\x98\x08\xaalI<\x97\x15A\t\x8c\x97\xdc\xc2\xa1\x18"\xdd
\x90\xa3\x0c\x90\xd27s=\xbd\xc2\x1aYE2\xc0\xbf\xec3\x94=\xa7\x88\x85\xb9\x14\xd8\xf8\x9
\x18\xc4\xc5[\xeea\xb7i\xa5\x8d\x869?
\x8f\xd0\xe7\x085\xcb\x01\xc2G\xe4\xdf\x17\xd1\xf4Z\xb1\x81\xc8\xf2\x1f\xf8\x98\xcd\xa9
```


Read dataset

```
df = pd.read_csv("/content/Housing_dataset.zip")
```

```
df
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	Na
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	Na
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.90	7.

506 rows × 14 columns

Basic Operation

```
df.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	NaN

Next steps: Generate code with df View recommended plots

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     486 non-null    float64
 1   ZN       486 non-null    float64
 2   INDUS    486 non-null    float64
 3   CHAS     486 non-null    float64
 4   NOX      506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      486 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    int64  
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    486 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

```
df.describe()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE
count	486.000000	486.000000	486.000000	486.000000	506.000000	506.000000	486.000000
mean	3.611874	11.211934	11.083992	0.069959	0.554695	6.284634	68.518519
std	8.720192	23.388876	6.835896	0.255340	0.115878	0.702617	27.999513
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000
25%	0.081900	0.000000	5.190000	0.000000	0.449000	5.885500	45.175000
50%	0.253715	0.000000	9.690000	0.000000	0.538000	6.208500	76.800000
75%	3.560263	12.500000	18.100000	0.000000	0.624000	6.623500	93.975000
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000

Data Preprocessing 1)Check null values and handle it

```
df.isna()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	False	False	False	False	False	False	False	False	False	False	False	False	Fal
1	False	False	False	False	False	False	False	False	False	False	False	False	Fal
2	False	False	False	False	False	False	False	False	False	False	False	False	Fal
3	False	False	False	False	False	False	False	False	False	False	False	False	Fal
4	False	False	False	False	False	False	False	False	False	False	False	False	Tr
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	False	False	False	False	False	False	False	False	False	False	False	False	Tr
502	False	False	False	False	False	False	False	False	False	False	False	False	Fal
503	False	False	False	False	False	False	False	False	False	False	False	False	Fal
504	False	False	False	False	False	False	False	False	False	False	False	False	Fal
505	False	False	False	False	False	False	True	False	False	False	False	False	Fal

506 rows × 14 columns

```
df.isna().sum()
```

```
CRIM       20
ZN         20
INDUS      20
CHAS       20
NOX         0
RM          0
AGE        20
DIS         0
RAD         0
TAX         0
PTRATIO     0
B           0
LSTAT      20
MEDV        0
dtype: int64
```

```
df = df.replace(np.NaN, 0)
```

```
df
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTA
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.9
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.1
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.0
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.9
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	0.0
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.0
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.6
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.4
505	0.04741	0.0	11.93	0.0	0.573	6.030	0.0	2.5050	1	273	21.0	396.90	7.8

506 rows × 14 columns

Next steps: Generate code with df View recommended plots

```
df.isna().sum()
```

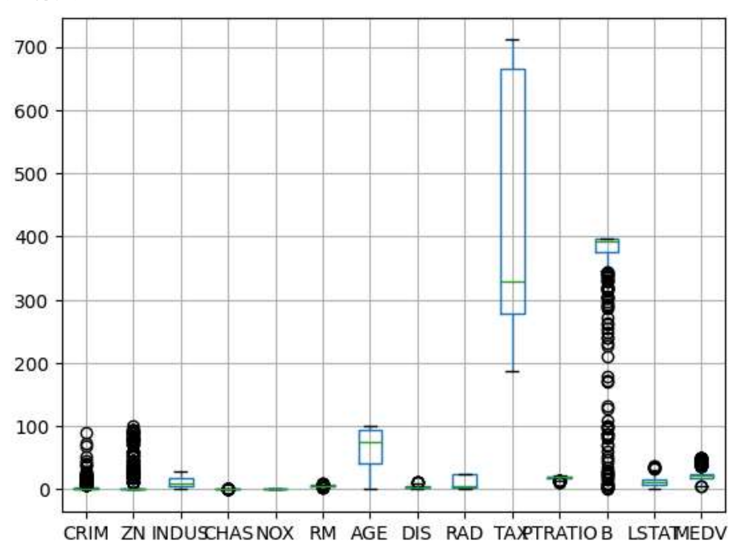
```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NOX        0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
MEDV       0
dtype: int64
```

Check Outliers

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df.boxplot()
```

```
<Axes: >
```




```
Q1 = df['MEDV'].quantile(0.25)
```

```
Q3 = df['MEDV'].quantile(0.75)
IQR = Q3 - Q1
Lower_limit = Q1 - 1.5 * IQR
Upper_limit = Q3 + 1.5 * IQR
print(f'Q1 = {Q1}, Q3 = {Q3}, IQR = {IQR}, Lower_limit = {Lower_limit},Upper_limit = {Upper_limit}')
```

```
Q1 = 17.025, Q3 = 25.0, IQR = 7.975000000000001, Lower_limit = 5.0624999999999964,Upper_limit = 36.962500000000006
```

```
outliers_MEDV=[]
for i in df.MEDV:
    if i<Lower_limit or i>Upper_limit:
        outliers_MEDV.append(i)
print("outliers are",outliers_MEDV)
```

```
outliers are [38.7, 43.8, 41.3, 50.0, 50.0, 50.0, 50.0, 37.2, 39.8, 37.9, 50.0, 37.0, 50.0, 42.3, 48.5, 50.0, 44.8, 50.0, 37.6, 46.7, 4
```

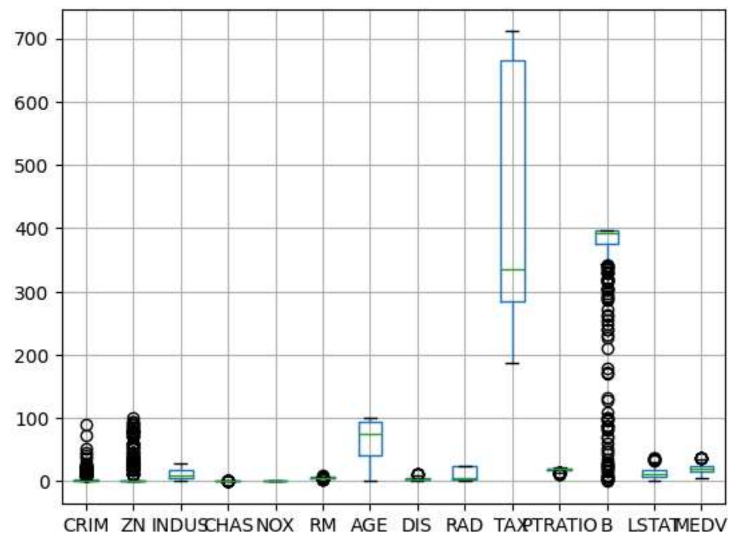
```
df[df.MEDV<Lower_limit].index
```

```
Index([398, 405], dtype='int64')
```

```
df1 = df.drop(df[(df['MEDV'] < Lower_limit) | (df['MEDV'] > Upper_limit)].index)
```

```
df1.boxplot()
```

```
<Axes: >
```




```
outliers_MEDV=[]
for i in df1.MEDV:
   if i<Lower_limit or i>Upper_limit:
       outliers_MEDV.append(i)
print("outliers are",outliers_MEDV)
```

```
outliers are []
```

```
df1
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	0.
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.
505	0.04741	0.0	11.93	0.0	0.573	6.030	0.0	2.5050	1	273	21.0	396.90	7.

466 rows × 14 columns

Next steps: Generate code with df1 View recommended plots

Preaparing the data for training the model

```
X = df.drop(['MEDV'], axis = 1)
Y = df['MEDV']
```

```
X
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	0.
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.
505	0.04741	0.0	11.93	0.0	0.573	6.030	0.0	2.5050	1	273	21.0	396.90	7.

506 rows × 13 columns

Next steps: Generate code with X View recommended plots

```
Y
```

```
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
       ...
501    22.4
502    20.6
503    23.9
504    22.0
505    11.9
Name: MEDV, Length: 506, dtype: float64
```

Splitting the data into training and testing sets

```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, Ytraiytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
ytrain_predn, Ytest = train_test_split(X, Y, test_size =0.2,random_state = 0)
```

```
import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
```

```
model=lm.fit(Xtrain, Ytrain)
```

```
model
```

```
▾ LinearRegression
LinearRegression()
```

Predict the y_pred for all values of train_x and test_x

```
Ytrain_pred = lm.predict(Xtrain)
Ytest_pred = lm.predict(Xtest)
Ytrain_pred
```

```
       24.85066407, 19.63467407, 29.88611991,  9.38276177, 24.76151505,
       16.72954149, 16.29385216, 22.77008753, 21.26802212, 12.2364709 ,
       24.64477302, 27.71019776, 22.37135579, 12.05768659, 24.34053293,
       26.22963857, 25.26554617, 27.5573375 , 25.38605534, 23.06402212,
       20.60323418, 35.90032876, 20.8846604 , 35.83504182, 25.53810828,
       20.18211477, 15.19432166, 31.81754202, 20.9929256 , 27.62499348,
       17.56075001, 32.93774989, 14.19953109,  2.05446008, 19.57716258,
       13.39959974, 37.06963824, 16.02843169, 14.22645918, 25.8004773 ,
       23.17938523, 17.51289677, 31.03928846, 29.67567958, 27.664875  ,
       24.87954992, 27.0939266 , 26.40862538, 11.1643275 , 20.36781195,
       10.0546563 , 17.26624222, 11.97746837, 27.32233663, 14.86882989,
       15.98655082, 28.32889291, 14.0393481 , 21.1535252 , 12.66673814,
       16.60941748, 23.44062773, 20.67404959, 14.38333681, 21.1686395 ,
       13.76862585, 25.66766384, 12.53439405, 35.14087573, 15.35455039,
       43.2661169 , 31.27520414, 34.39926511, 21.58897794, 15.39880006,
       26.85768714, 28.60521699, 13.42512446, 26.26735857, 35.59014297,
       16.12321718, 12.1104995 , 34.04506216, 35.60373817, 17.48828129,
       25.33999395, 20.60520108, 24.06997649, 18.77252598, 26.9130493 ,
       -4.01040623, 20.54287878, 37.41339975, 34.74735709, 24.63091938,
       25.91858311, 19.44854638, 21.10224807, 16.19845928, 17.06790664,
       20.38162681, 27.64453456, 19.60519903,  7.24242967, 16.00033535,
       31.73236706, 35.45008203, 15.42885644, 18.77768003, 26.32877708,
        5.66978469, 21.18273113, 22.75936134, 15.50615652, 17.80964805,
       22.81298897, 26.78805495, 25.83084156, 37.22114199, 14.19665739,
       28.23285926, 24.85620232, 20.73994597, 38.30287036, 22.29143643,
       23.24430272, 22.34183176, 10.95022824, 19.50231212, 32.85036084,
       24.38133064, 17.27463319, 32.50636784, 22.68303606, 28.23715359,
       31.71958986, 36.30336114, 21.66627169, 23.20527639, 22.97726689,
       31.84758343, 22.51864962, 17.9820902 , 21.40804716, 28.84399091,
       22.76519651, 22.05554279, 17.40672403, 17.30260312, 16.67622643,
       16.44614716, 17.91832318, 31.75872911, 22.57702341, 17.48740869,
       18.64901264, 33.9937244 , 13.93997981, 25.52249105, 16.63719249,
```

```
       28.42512571, 31.56316075, 24.45335221, 36.08996401, 17.99040572,
       19.63518719, 18.83950626, 41.31553357, 25.01876563, 18.73759211,
       33.53881   , 23.08223101, 19.05046237, 22.94236564])
```

```
Ytrain
```

```
220    26.7
71     21.7
240    22.0
6      22.9
417    10.4
       ...
323    18.5
192    36.4
117    19.2
47     16.6
172    23.1
```