Dragon Real Estate - Price Predictor

```
import pandas as pd
```

```
housing = pd.read_csv("data.csv")
```

```
housing.head()
```

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

```
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
CRIM       506 non-null float64
ZN         506 non-null float64
INDUS      506 non-null float64
CHAS       506 non-null int64
NOX        506 non-null float64
RM         501 non-null float64
AGE        506 non-null float64
DIS        506 non-null float64
RAD        506 non-null int64
TAX        506 non-null int64
PTRATIO    506 non-null float64
B          506 non-null float64
LSTAT      506 non-null float64
MEDV       506 non-null float64
dtypes: float64(11), int64(3)
memory usage: 55.4 KB
```

```
housing['CHAS'].value_counts()
```

```
0    471
1     35
Name: CHAS, dtype: int64
```

```
housing.describe()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	A
count	506.000000	506.000000	506.000000	506.000000	506.000000	501.000000	506.000
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284341	68.5749
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.705587	28.1488
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.9000
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.884000	45.0250
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208000	77.5000
75%	3.677082	12.500000	18.100000	0.000000	0.624000	6.625000	94.0750
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000

```
%matplotlib inline
```

```
# # For plotting histogram
# import matplotlib.pyplot as plt
# housing.hist(bins=50, figsize=(20, 15))
```

Looking for Correlations

```
corr_matrix = housing.corr()
corr_matrix['MEDV'].sort_values(ascending=False)
```

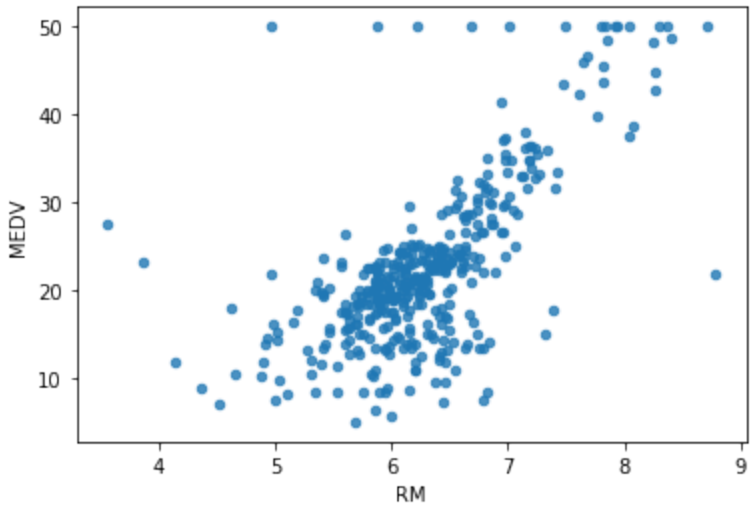
```
MEDV       1.000000
RM         0.680857
B          0.361761
ZN         0.339741
DIS        0.240451
CHAS       0.205066
AGE       -0.364596
RAD       -0.374693
CRIM      -0.393715
NOX       -0.422873
TAX       -0.456657
INDUS     -0.473516
PTRATIO   -0.493534
LSTAT     -0.740494
Name: MEDV, dtype: float64
```

```
# from pandas.plotting import scatter_matrix
# attributes = ["MEDV", "RM", "ZN", "LSTAT"]
```

```
# scatter_matrix(housing[attributes], figsize = (12,8))
```

```
housing.plot(kind="scatter", x="RM", y="MEDV", alpha=0.8)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xcbdc8f0>
```



Missing Attributes

```
a = housing.dropna(subset=["RM"])
```

```
(399, 13)
```

```
housing.drop("RM", axis=1).shape
```

```
(404, 12)
```

```
median = housing["RM"].median()
```

```
housing["RM"].fillna(median)
```

```
254    6.108
348    6.635
476    6.484
321    6.376
326    6.312
       ...
155    6.152
423    6.103
98     7.820
455    6.525
216    5.888
Name: RM, Length: 404, dtype: float64
```

```
housing.shape
```

```
(404, 13)
```

```
housing.describe()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	A
count	404.000000	404.000000	404.000000	404.000000	404.000000	399.000000	404.000
mean	3.602814	10.836634	11.344950	0.069307	0.558064	6.279481	69.039
std	8.099383	22.150636	6.877817	0.254290	0.116875	0.716784	28.258
min	0.006320	0.000000	0.740000	0.000000	0.389000	3.561000	2.900
25%	0.086963	0.000000	5.190000	0.000000	0.453000	5.876500	44.850
50%	0.286735	0.000000	9.900000	0.000000	0.538000	6.209000	78.200
75%	3.731923	12.500000	18.100000	0.000000	0.631000	6.630500	94.100
max	73.534100	100.000000	27.740000	1.000000	0.871000	8.780000	100.000

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy="median")
imputer.fit(housing)
```

```
SimpleImputer(add_indicator=False, copy=True, fill_value=None,
              missing_values=nan, strategy='median', verbose=0)
```

```
imputer.statistics_
```

```
array([2.86735e-01, 0.00000e+00, 9.90000e+00, 0.00000e+00, 5.38000e-01,
       6.20900e+00, 7.82000e+01, 3.12220e+00, 5.00000e+00, 3.37000e+02,
       1.90000e+01, 3.90955e+02, 1.15700e+01])
```

```
X = imputer.transform(housing)
```

```
housing_tr = pd.DataFrame(X, columns=housing.columns)
```

```
housing_tr.describe()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	A
count	404.000000	404.000000	404.000000	404.000000	404.000000	404.000000	404.0000
mean	3.602814	10.836634	11.344950	0.069307	0.558064	6.278609	69.0398
std	8.099383	22.150636	6.877817	0.254290	0.116875	0.712366	28.2582
min	0.006320	0.000000	0.740000	0.000000	0.389000	3.561000	2.9000
25%	0.086963	0.000000	5.190000	0.000000	0.453000	5.878750	44.8500
50%	0.286735	0.000000	9.900000	0.000000	0.538000	6.209000	78.2000
75%	3.731923	12.500000	18.100000	0.000000	0.631000	6.630000	94.1000
max	73.534100	100.000000	27.740000	1.000000	0.871000	8.780000	100.0000