

## Problem Statement

A real estate agent want help to predict the house price for regions in USA.He gave us the dataset to work on to use linear regression model.Create a model that helps him to estimate of what the house would sell for

## Import libraries

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: # To import dataset
df=pd.read_csv('bottle csv')
df
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3
165: DtypeWarning: Columns (47,73) have mixed types.Specify dtype option on i
mport or set low_memory=False.
    has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Out[2]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
0	1	1	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0000A-3	0	10.500	33.4400	NaN	25.64900	Na
1	1	2	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0008A-3	8	10.460	33.4400	NaN	25.65600	Na
2	1	3	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0010A-7	10	10.460	33.4370	NaN	25.65400	Na
3	1	4	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0019A-3	19	10.450	33.4200	NaN	25.64300	Na
4	1	5	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0020A-7	20	10.450	33.4210	NaN	25.64300	Na
...	...	...	...	...	...	...	...	...	...	...
864858	34404	864859	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0000A-7	0	18.744	33.4083	5.805	23.87055	108.7
864859	34404	864860	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0002A-3	2	18.744	33.4083	5.805	23.87072	108.7
864860	34404	864861	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0005A-3	5	18.692	33.4150	5.796	23.88911	108.4
864861	34404	864862	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0010A-3	10	18.161	33.4062	5.816	24.01426	107.7

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2S
864862	34404	864863	093.4 026.4	20- 1611SR- MX-310- 2239- 09340264- 0015A-3	15	17.533	33.3880	5.774	24.15297	105.6

864863 rows × 74 columns

```
In [3]: # To display top 10 rows  
df.head(10)
```

Out[3]:

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R
0	1	1	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0000A-3	0	10.50	33.440	NaN	25.649	NaN	...	
1	1	2	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0008A-3	8	10.46	33.440	NaN	25.656	NaN	...	
2	1	3	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0010A-7	10	10.46	33.437	NaN	25.654	NaN	...	
3	1	4	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0019A-3	19	10.45	33.420	NaN	25.643	NaN	...	
4	1	5	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0020A-7	20	10.45	33.421	NaN	25.643	NaN	...	
5	1	6	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0030A-7	30	10.45	33.431	NaN	25.651	NaN	...	
6	1	7	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0039A-3	39	10.45	33.440	NaN	25.658	NaN	...	
7	1	8	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0050A-7	50	10.24	33.424	NaN	25.682	NaN	...	
8	1	9	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0058A-3	58	10.06	33.420	NaN	25.710	NaN	...	

	Cst_Cnt	Btl_Cnt	Sta_ID	Depth_ID	Depthm	T_degC	Salnty	O2ml_L	STheta	O2Sat	...	R
9	1	10	054.0 056.0	19- 4903CR- HY-060- 0930- 05400560- 0075A-7	75	9.86	33.494	NaN	25.801	NaN	...	

10 rows × 74 columns

# Data Cleaning and Pre-Processing

In [4]: `df.info()`



```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 864863 entries, 0 to 864862
```

```
Data columns (total 74 columns):
```

#	Column	Non-Null Count	Dtype
0	Cst_Cnt	864863 non-null	int64
1	Btl_Cnt	864863 non-null	int64
2	Sta_ID	864863 non-null	object
3	Depth_ID	864863 non-null	object
4	Depthm	864863 non-null	int64
5	T_degC	853900 non-null	float64
6	Salnty	817509 non-null	float64
7	O2ml_L	696201 non-null	float64
8	STheta	812174 non-null	float64
9	O2Sat	661274 non-null	float64
10	Oxy_μmol/Kg	661268 non-null	float64
11	BtlNum	118667 non-null	float64
12	RecInd	864863 non-null	int64
13	T_prec	853900 non-null	float64
14	T_qual	23127 non-null	float64
15	S_prec	817509 non-null	float64
16	S_qual	74914 non-null	float64
17	P_qual	673755 non-null	float64
18	O_qual	184676 non-null	float64
19	SThtaq	65823 non-null	float64
20	O2Satq	217797 non-null	float64
21	ChlorA	225272 non-null	float64
22	Chlqua	639166 non-null	float64
23	Phaeop	225271 non-null	float64
24	Phaqua	639170 non-null	float64
25	PO4uM	413317 non-null	float64
26	PO4q	451786 non-null	float64
27	SiO3uM	354091 non-null	float64
28	SiO3qu	510866 non-null	float64
29	NO2uM	337576 non-null	float64
30	NO2q	529474 non-null	float64
31	NO3uM	337403 non-null	float64
32	NO3q	529933 non-null	float64
33	NH3uM	64962 non-null	float64
34	NH3q	808299 non-null	float64
35	C14As1	14432 non-null	float64
36	C14A1p	12760 non-null	float64
37	C14A1q	848605 non-null	float64
38	C14As2	14414 non-null	float64
39	C14A2p	12742 non-null	float64
40	C14A2q	848623 non-null	float64
41	DarkAs	22649 non-null	float64
42	DarkAp	20457 non-null	float64
43	DarkAq	840440 non-null	float64
44	MeanAs	22650 non-null	float64
45	MeanAp	20457 non-null	float64
46	MeanAq	840439 non-null	float64
47	IncTim	14437 non-null	object
48	LightP	18651 non-null	float64
49	R_Depth	864863 non-null	float64
50	R_TEMP	853900 non-null	float64
51	R_POTEMP	818816 non-null	float64

```

52  R_SALINITY          817509 non-null float64
53  R_SIGMA            812007 non-null float64
54  R_SVA              812092 non-null float64
55  R_DYNHT            818206 non-null float64
56  R_O2               696201 non-null float64
57  R_O2Sat            666448 non-null float64
58  R_SIO3             354099 non-null float64
59  R_PO4              413325 non-null float64
60  R_NO3              337411 non-null float64
61  R_NO2              337584 non-null float64
62  R_NH4              64982 non-null float64
63  R_CHLA             225276 non-null float64
64  R_PHAEO            225275 non-null float64
65  R_PRES             864863 non-null int64
66  R_SAMP             122006 non-null float64
67  DIC1               1999 non-null float64
68  DIC2               224 non-null float64
69  TA1                2084 non-null float64
70  TA2                234 non-null float64
71  pH2                10 non-null float64
72  pH1                84 non-null float64
73  DIC Quality Comment 55 non-null object
dtypes: float64(65), int64(5), object(4)
memory usage: 488.3+ MB

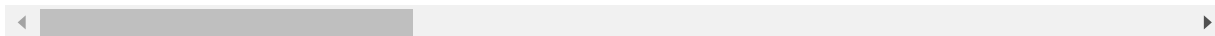
```

In [5]: `df.describe()`

Out[5]:

	Cst_Cnt	Btl_Cnt	Depthm	T_degC	Salnty	O2ml_
<b>count</b>	864863.000000	864863.000000	864863.000000	853900.000000	817509.000000	696201.000000
<b>mean</b>	17138.790958	432432.000000	226.831951	10.799677	33.840350	3.39246
<b>std</b>	10240.949817	249664.587267	316.050259	4.243825	0.461843	2.07325
<b>min</b>	1.000000	1.000000	0.000000	1.440000	28.431000	-0.01000
<b>25%</b>	8269.000000	216216.500000	46.000000	7.680000	33.488000	1.36000
<b>50%</b>	16848.000000	432432.000000	125.000000	10.060000	33.863000	3.44000
<b>75%</b>	26557.000000	648647.500000	300.000000	13.880000	34.196900	5.50000
<b>max</b>	34404.000000	864863.000000	5351.000000	31.140000	37.034000	11.13000

8 rows × 70 columns



```
In [6]: df.columns
```

```
Out[6]: Index(['Cst_Cnt', 'Btl_Cnt', 'Sta_ID', 'Depth_ID', 'Depthm', 'T_degC',  
              'Salnty', 'O2m1_L', 'STheta', 'O2Sat', 'Oxy_μmol/Kg', 'BtlNum',  
              'RecInd', 'T_prec', 'T_qual', 'S_prec', 'S_qual', 'P_qual', 'O_qual',  
              'SThetaq', 'O2Satq', 'ChlorA', 'Chlqua', 'Phaeop', 'Phaqua', 'PO4uM',  
              'PO4q', 'SiO3uM', 'SiO3qu', 'NO2uM', 'NO2q', 'NO3uM', 'NO3q', 'NH3uM',  
              'NH3q', 'C14As1', 'C14A1p', 'C14A1q', 'C14As2', 'C14A2p', 'C14A2q',  
              'DarkAs', 'DarkAp', 'DarkAq', 'MeanAs', 'MeanAp', 'MeanAq', 'IncTim',  
              'LightP', 'R_Depth', 'R_TEMP', 'R_POTEMP', 'R_SALINITY', 'R_SIGMA',  
              'R_SVA', 'R_DYNHT', 'R_O2', 'R_O2Sat', 'R_SIO3', 'R_PO4', 'R_NO3',  
              'R_NO2', 'R_NH4', 'R_CHLA', 'R_PHAEO', 'R_PRES', 'R_SAMP', 'DIC1',  
              'DIC2', 'TA1', 'TA2', 'pH2', 'pH1', 'DIC Quality Comment'],  
              dtype='object')
```

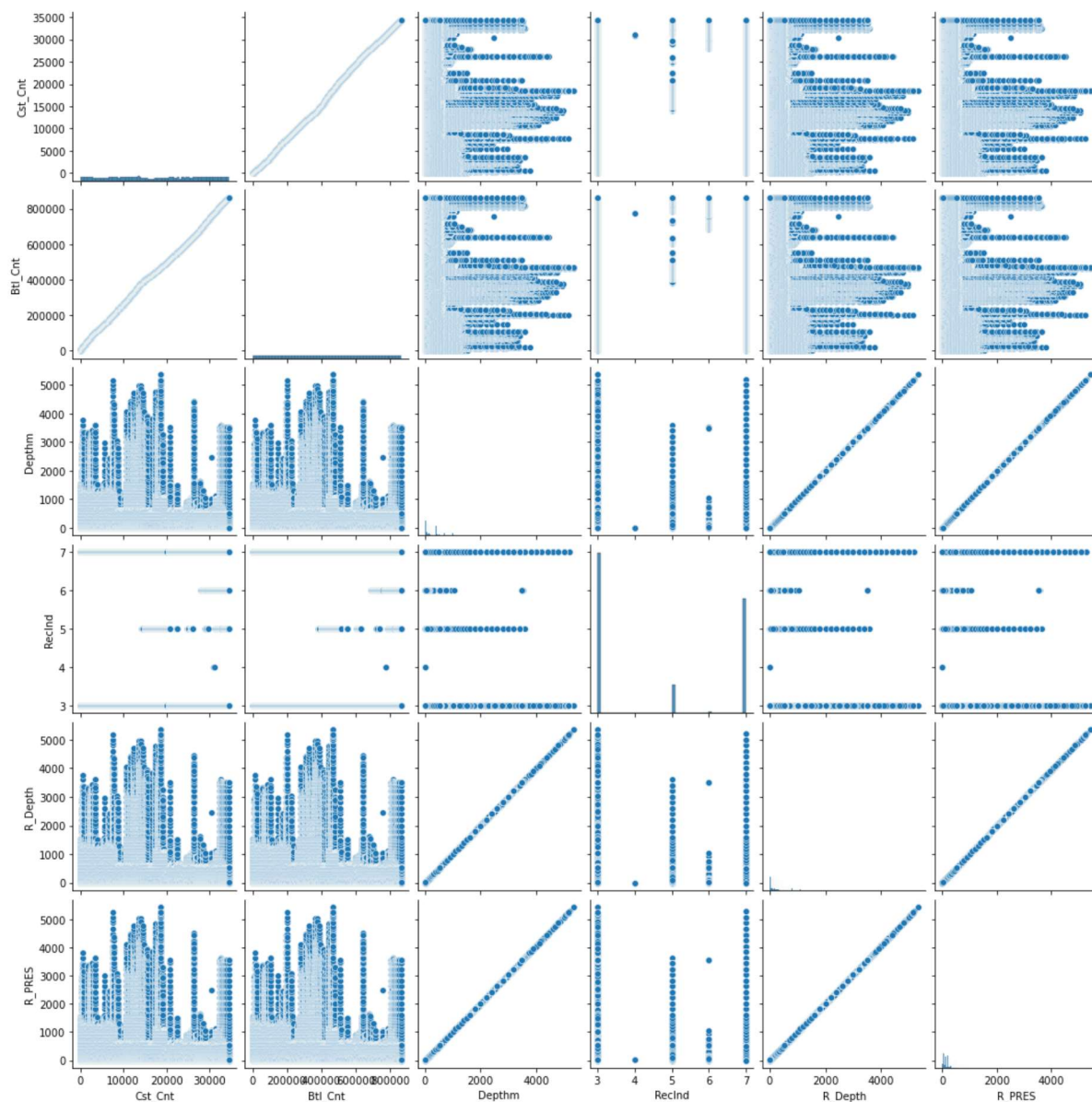
```
In [7]: a = df.dropna(axis='columns')  
a.columns
```

```
Out[7]: Index(['Cst_Cnt', 'Btl_Cnt', 'Sta_ID', 'Depth_ID', 'Depthm', 'RecInd',  
              'R_Depth', 'R_PRES'],  
              dtype='object')
```

## EDA and Visualization

```
In [8]: sns.pairplot(a)
```

```
Out[8]: <seaborn.axisgrid.PairGrid at 0x2638d039e20>
```

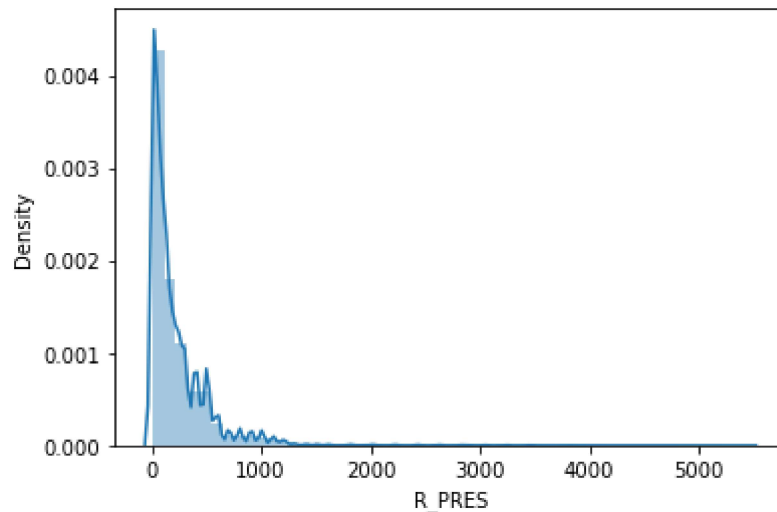


```
In [9]: sns.distplot(a['R_PRES'])
```

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

```
warnings.warn(msg, FutureWarning)
```

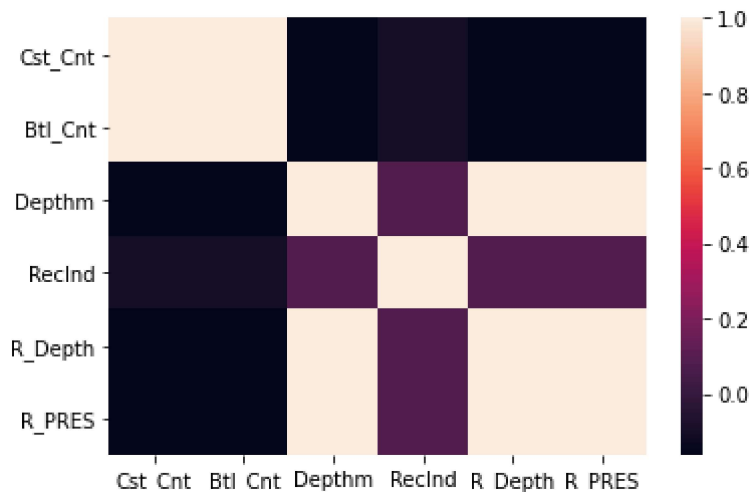
```
Out[9]: <AxesSubplot:xlabel='R_PRES', ylabel='Density'>
```



```
In [10]: a1=a[['Cst_Cnt', 'Btl_Cnt', 'Sta_ID', 'Depth_ID', 'Depthm', 'RecInd',  
              'R_Depth', 'R_PRES']]
```

```
In [11]: sns.heatmap(a1.corr())
```

```
Out[11]: <AxesSubplot:>
```



## To Train the Model - Model Building

We are going to train Linear Regression model. We need to split out data into two variables x

```
In [16]: x=a1[['Cst_Cnt', 'Btl_Cnt', 'R_Depth', 'R_PRES']]
y=a1['Depthm']
```

## To split my dataset into training and test data

```
In [18]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

```
In [19]: from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[19]: LinearRegression()

```
In [20]: print(lr.intercept_)
0.0015351219604440303
```

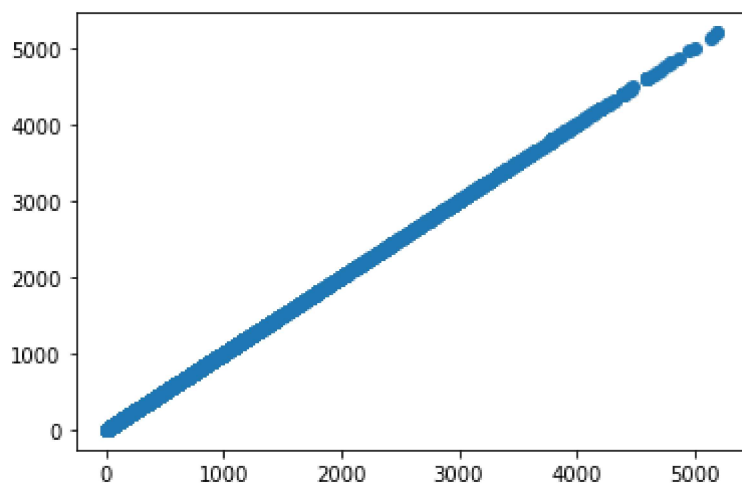
```
In [21]: coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[21]:

	Co-efficient
Cst_Cnt	1.606121e-06
Btl_Cnt	-6.920989e-08
R_Depth	1.000296e+00
R_PRES	-2.925724e-04

```
In [22]: prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[22]: <matplotlib.collections.PathCollection at 0x26394a15a00>



```
In [23]: print(lr.score(x_test,y_test))
0.9999999942244459
```

In [ ]: