“ BAA- MACHINE LEARNING ”

USING – PYTHON

By – SIDDHESH POWAR

Modified Sample Data used for Analysis

```
csv imported successfully
```

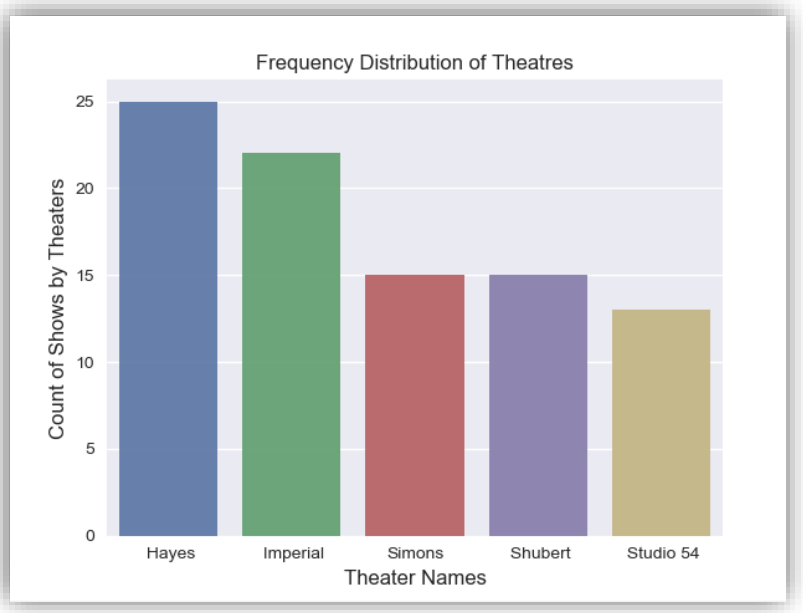
Out[30]:

nds	grosses_thousands	sub_thousands	group_thousands	single_thousands	avg_ticket	top_ticket	capacity_percent	run_time_mins	rev_per_seat	city_theater
16	1027	226	63	738	62.03	200	86	150	65.7	Hayes
16	1065	418	114	533	60.16	200	80	140	59.6	Hayes
16	717	380	42	294	55.95	200	90	155	79.8	Imperial
16	577	370	31	176	74.69	200	87	140	63.4	Hayes
16	835	402	131	302	77.34	200	81	140	78.4	Imperial
10	730	147	91	492	78.78	200	73	145	90.2	Hayes
16	1007	475	71	461	76.87	200	80	145	66.3	Imperial
32	2901	390	297	2214	75.74	200	75	135	67.8	Studio 54
16	1018	441	132	444	75.52	200	62	165	63.4	Imperial
16	734	412	51	270	74.86	200	65	230	68.1	Simons
16	935	433	79	423	69.40	200	65	135	70.6	Hayes
16	698	422	43	233	73.17	200	66	160	70.6	Hayes
16	1233	305	128	799	74.30	200	68	150	70.6	Shubert
16	882	433	149	300	71.03	200	73	150	63.4	Simons

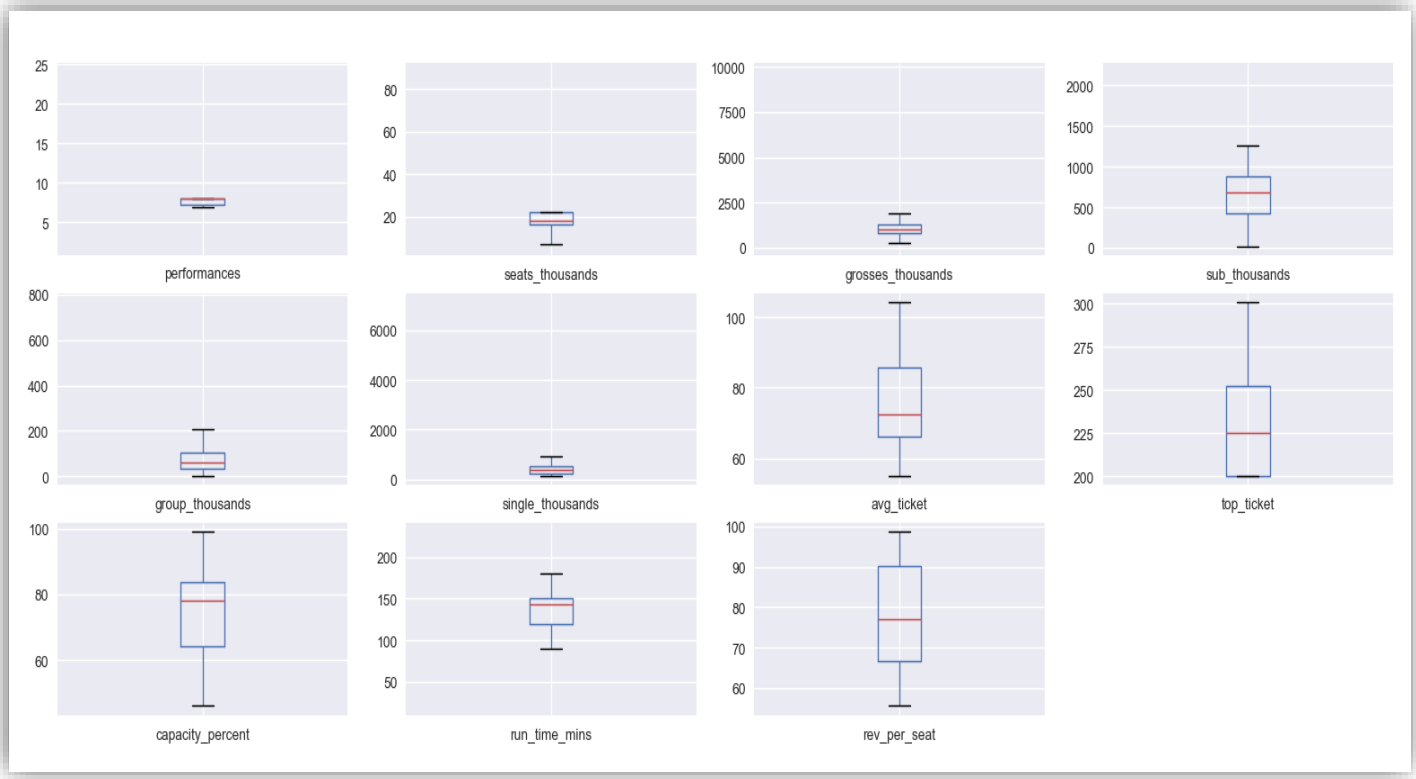
Exploratory Data Analysis EDA: Data Info

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 12 columns):
performances         90 non-null int64
seats_thousands      90 non-null int64
grosses_thousands    90 non-null int64
sub_thousands        90 non-null int64
group_thousands      90 non-null int64
single_thousands     90 non-null int64
avg_ticket           90 non-null float64
top_ticket           90 non-null int64
capacity_percent     90 non-null int64
run_time_mins        90 non-null int64
rev_per_seat         90 non-null float64
city_theater         90 non-null object
dtypes: float64(2), int64(9), object(1)
memory usage: 8.5+ KB
```

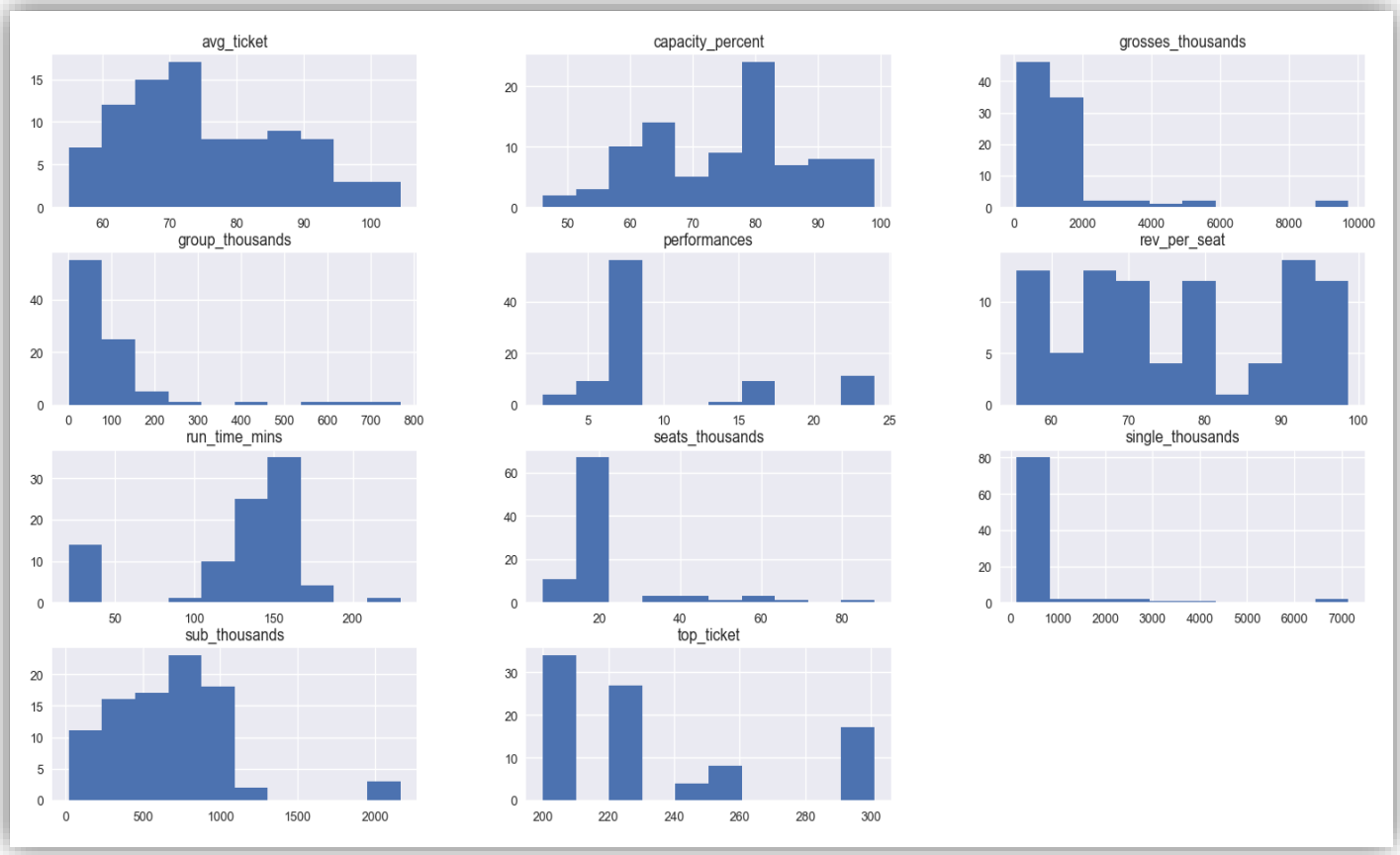
EDA: Plotting Frequency of Shows by each Theater



EDA: Univariate Analysis Box and Whisker Plot



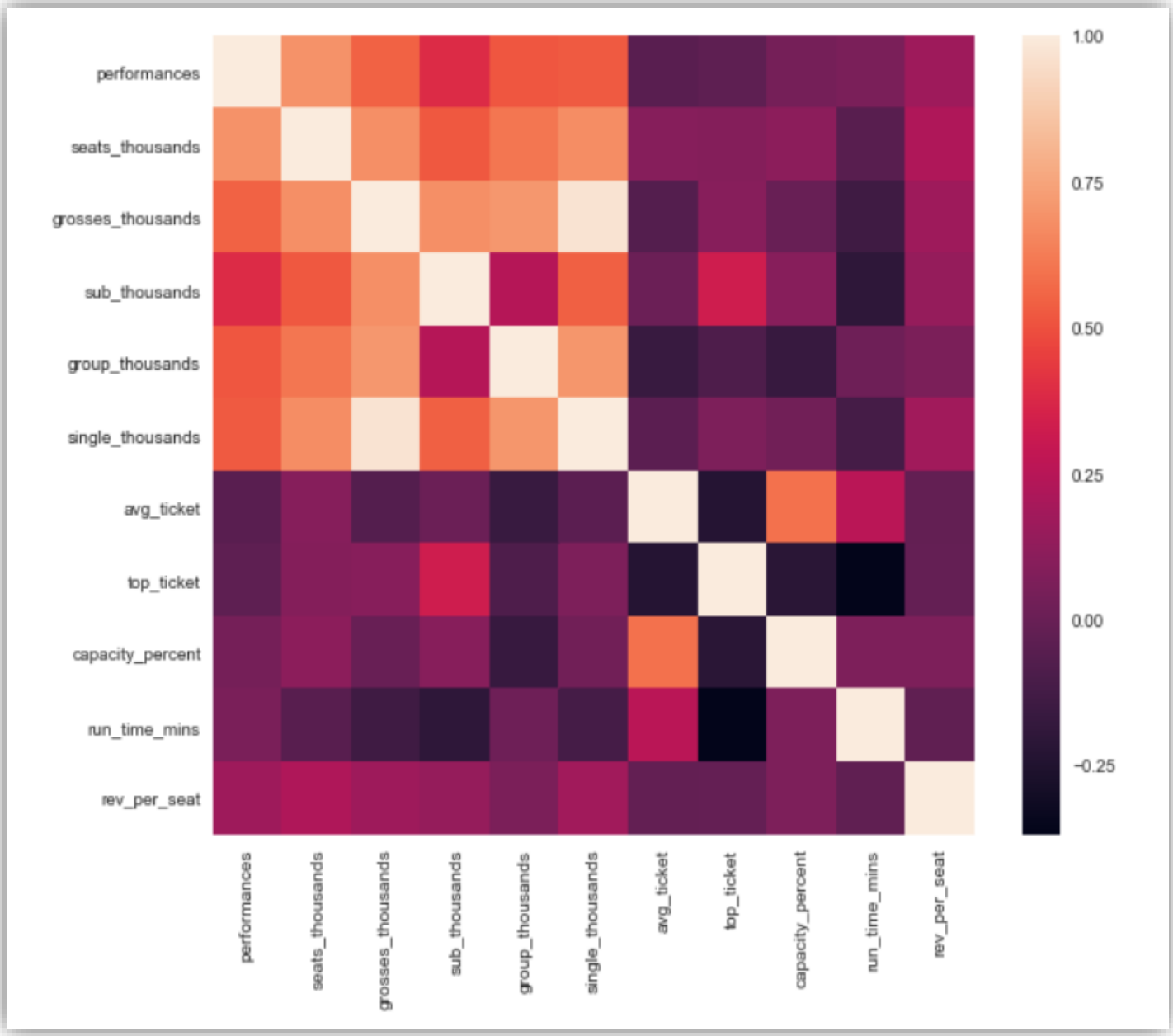
EDA: Univariate Plot Histogram



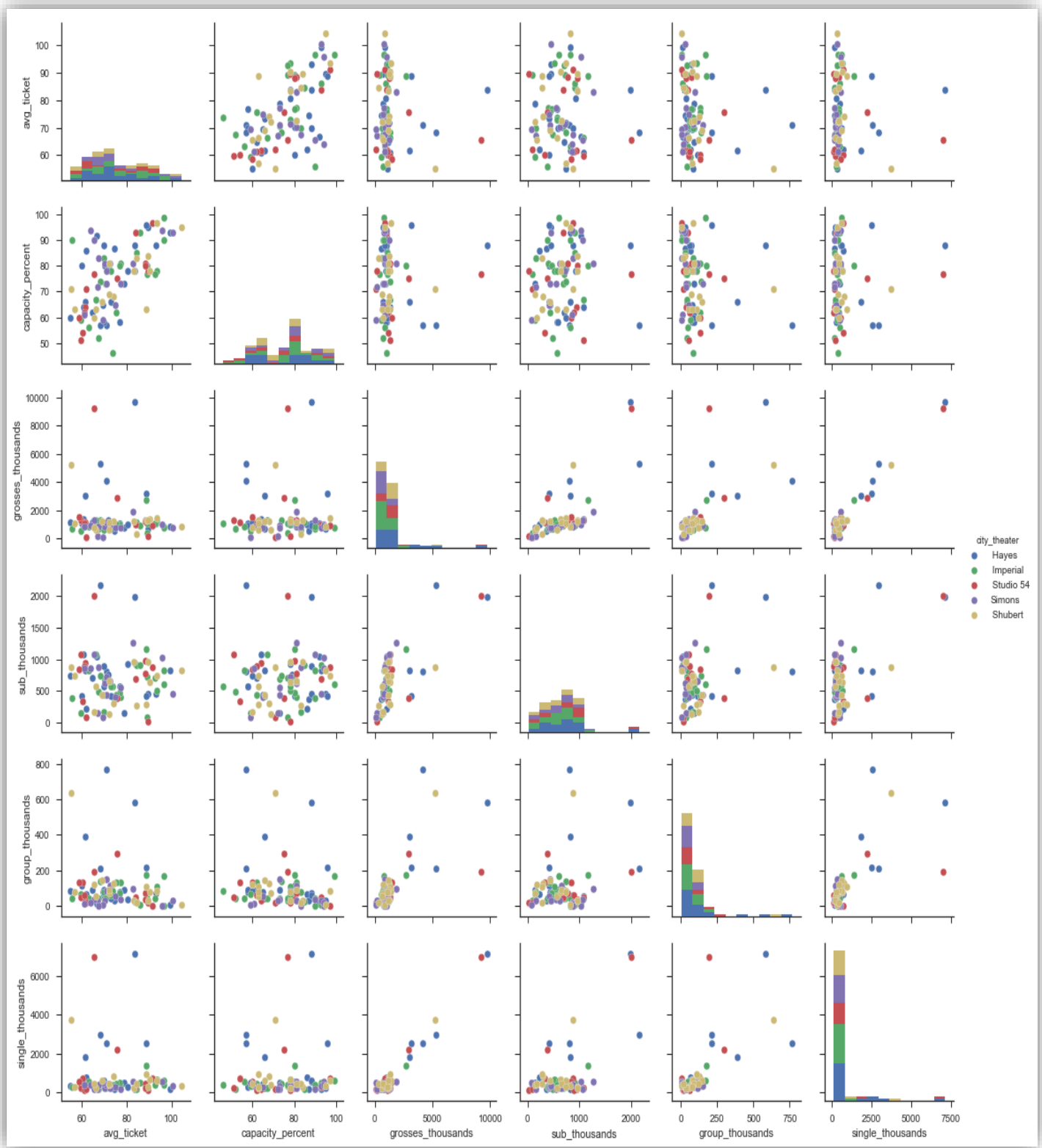
Correlation Matrix

0	1	0.347533	0.398948	0.455743	0.0729144	-0.233402	-0.731222	0.477978	-0.442621	0.0151847
1	0.347533	1	-0.284056	0.571003	-0.285483	0.38248	-0.362842	0.642578	0.252556	0.190047
2	0.398948	-0.284056	1	-0.523649	0.152937	-0.139176	-0.0928948	0.0162655	-0.434016	-0.383585
3	0.455743	0.571003	-0.523649	1	-0.225343	-0.227577	-0.481548	0.473286	0.279258	0.44665
4	0.0729144	-0.285483	0.152937	-0.225343	1	-0.104438	-0.147477	-0.523283	-0.614603	-0.189916
5	-0.233402	0.38248	-0.139176	-0.227577	-0.104438	1	-0.0302517	0.41764	0.205851	0.0950844
6	-0.731222	-0.362842	-0.0928948	-0.481548	-0.147477	-0.0302517	1	-0.49444	0.381407	-0.353652
7	0.477978	0.642578	0.0162655	0.473286	-0.523283	0.41764	-0.49444	1	0.375873	0.417863
8	-0.442621	0.252556	-0.434016	0.279258	-0.614603	0.205851	0.381407	0.375873	1	0.150421
9	0.0151847	0.190047	-0.383585	0.44665	-0.189916	0.0950844	-0.353652	0.417863	0.150421	1

Correlation Heatmap for all Parameters



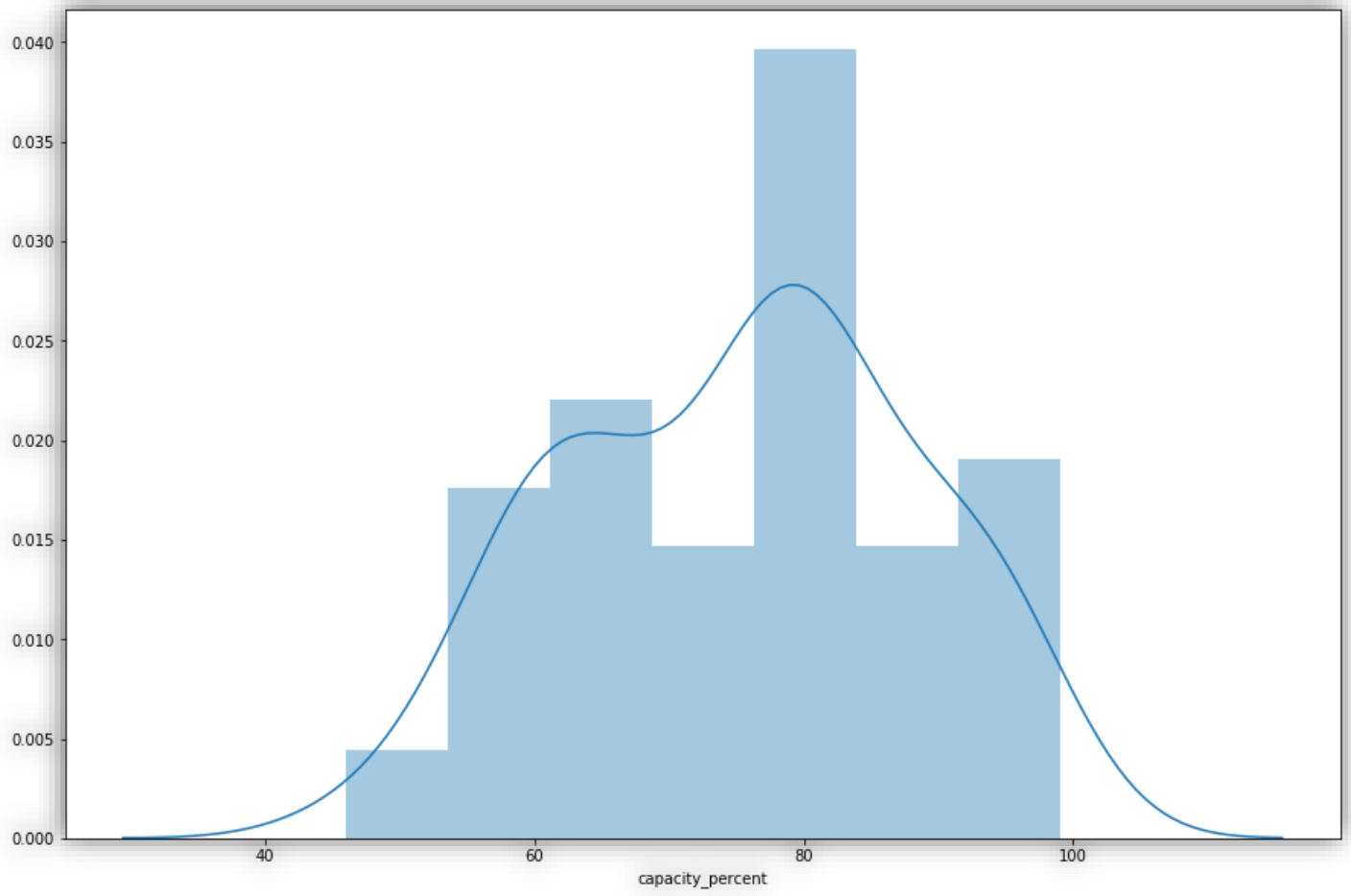
Pair Plots


avg_ticket
capacity_percent
grosses_thousands
sub_thousands
group_thousands
single_thousands
city_theater
Hayes
Imperial
Studio 54
Simons
Shubert

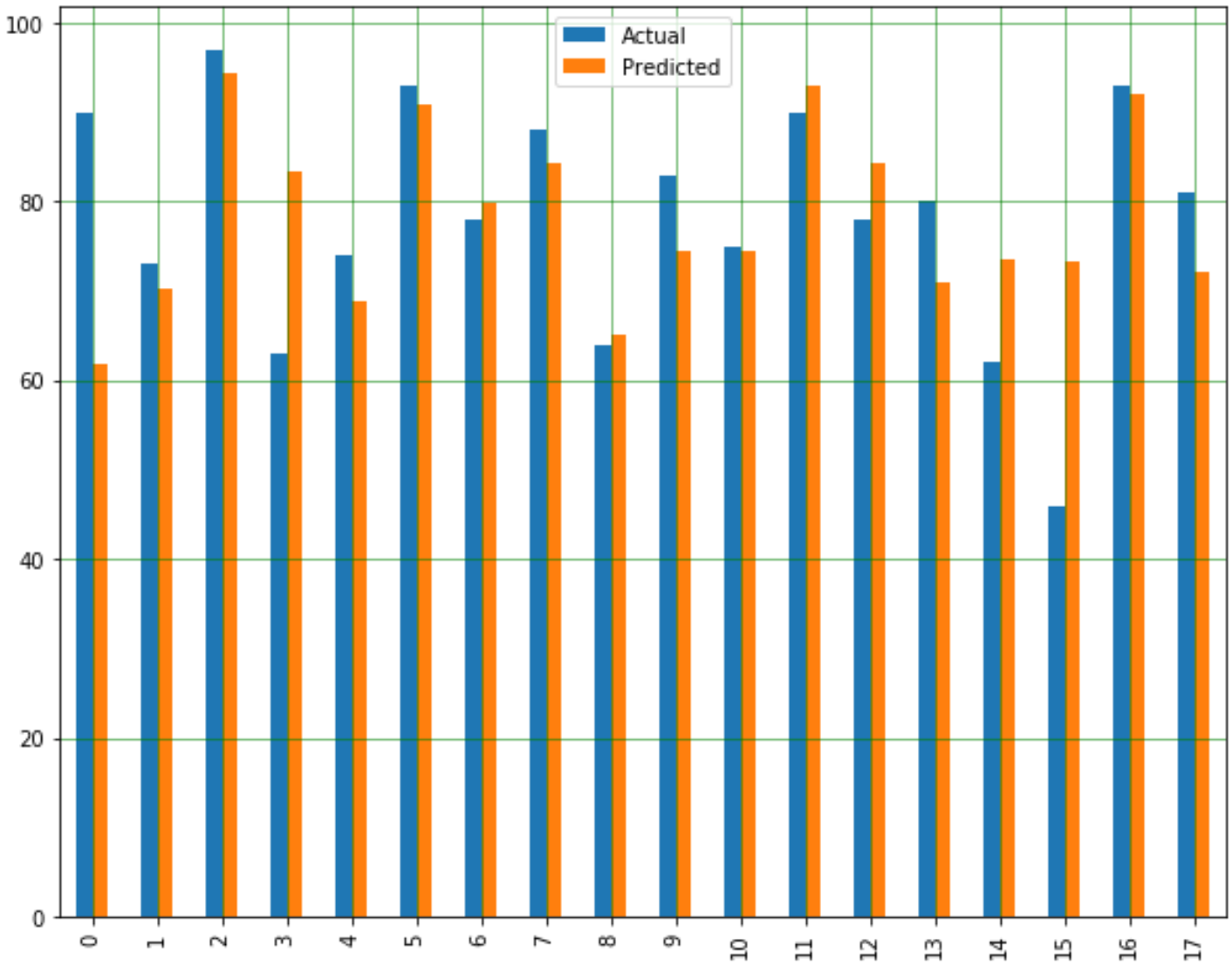
Machine Learning: Linear Regression Data Distribution

Out[16]:

	avg_ticket	capacity_percent	grosses_thousands	sub_thousands	group_thousands	single_thousands
count	90.000000	90.000000	90.000000	90.000000	90.000000	90.00000
mean	75.383111	75.488889	1365.411111	676.233333	92.677778	681.30000
std	12.222814	12.783631	1504.228787	386.548602	125.030048	1149.12706
min	55.000000	46.000000	84.000000	20.000000	1.000000	129.00000
25%	66.355000	64.250000	830.500000	424.750000	32.000000	242.00000
50%	72.565000	78.000000	1031.500000	679.000000	61.500000	373.50000
75%	85.725000	83.750000	1256.000000	877.500000	103.500000	529.00000
max	104.340000	99.000000	9711.000000	2165.000000	768.000000	7142.00000



Actual VS Predictions



Performance Evaluation of the Model

```
In [42]: from sklearn.linear_model import LinearRegression
         accuracy = regressor.score(X_test,y_test)
         print(accuracy*100,'%')

         89.05404442607621 %
```