Chapter 2: Descriptive Analytics

2.1 Working with DataFrames

2.1.1 Loading the dataset onto a DataFrame

```
import pandas as pd

pd.set_option('display.float_format', lambda x: '%.3f' % x)
```

```
pd.read_csv?
```

```
ipl_auction_df = pd.read_csv( 'IPL IMB381IPL2013.csv' )
```

```
type(ipl_auction_df)
```

```
pandas.core.frame.DataFrame
```

2.1.2 Displaying first few records of the DataFrame

```
pd.set_option('display.max_columns', 7)
```

```
ipl_auction_df.head(5)
```

	Sl.NO.	PLAYER NAME	AGE	...	AUCTION YEAR	BASE PRICE	SOLD PRICE
0	1	Abdulla, YA	2	...	2009	50000	50000
1	2	Abdur Razzak	2	...	2008	50000	50000
2	3	Agarkar, AB	2	...	2008	200000	350000
3	4	Ashwin, R	1	...	2011	100000	850000
4	5	Badrinath, S	2	...	2011	100000	800000

5 rows × 26 columns

2.1.3 Finding metadata of the DataFrame

```
list(ipl_auction_df.columns)
```

```
['Sl.NO.',
 'PLAYER NAME',
 'AGE',
 'COUNTRY',
 'TEAM',
 'PLAYING ROLE',
 'T-RUNS',
 'T-WKTS',
 'ODI-RUNS-S',
 'ODI-SR-B',
 'ODI-WKTS',
 'ODI-SR-BL',
 'CAPTAINCY EXP',
 'RUNS-S',
 'HS',
 'AVE',
 'SR-B',
 'SIXERS',
 'RUNS-C',
 'WKTS',
 'AVE-BL',
 'ECON',
 'SR-BL',
 'AUCTION YEAR',
 'BASE PRICE',
 'SOLD PRICE']
```

```
ipl_auction_df.head(5).transpose()
```

	0	1	2	3	4
Sl.NO.	1	2	3	4	5
PLAYER NAME	Abdulla, YA	Abdur Razzak	Agarkar, AB	Ashwin, R	Badrinath, S
AGE	2	2	2	1	2
COUNTRY	SA	BAN	IND	IND	IND
TEAM	KXIP	RCB	KKR	CSK	CSK
PLAYING ROLE	Allrounder	Bowler	Bowler	Bowler	Batsman
T-RUNS	0	214	571	284	63
T-WKTS	0	18	58	31	0
ODI-RUNS-S	0	657	1269	241	79
ODI-SR-B	0.000	71.410	80.620	84.560	45.930
ODI-WKTS	0	185	288	51	0
ODI-SR-BL	0.000	37.600	32.900	36.800	0.000
CAPTAINCY EXP	0	0	0	0	0
RUNS-S	0	0	167	58	1317
HS	0	0	39	11	71
AVE	0.000	0.000	18.560	5.800	32.930
SR-B	0.000	0.000	121.010	76.320	120.710
SIXERS	0	0	5	0	28
RUNS-C	307	29	1059	1125	0
WKTS	15	0	29	49	0
AVE-BL	20.470	0.000	36.520	22.960	0.000
ECON	8.900	14.500	8.810	6.230	0.000
SR-BL	13.930	0.000	24.900	22.140	0.000
AUCTION YEAR	2009	2008	2008	2011	2011
BASE PRICE	50000	50000	200000	100000	100000
SOLD PRICE	50000	50000	350000	850000	800000

```
ipl_auction_df.shape
```

```
(130, 26)
```

2.1.4 Finding Summary of the DataFrame

```
ipl_auction_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 130 entries, 0 to 129
Data columns (total 26 columns):
Sl.NO.           130 non-null int64
PLAYER NAME      130 non-null object
AGE              130 non-null int64
COUNTRY          130 non-null object
TEAM             130 non-null object
PLAYING ROLE     130 non-null object
T-RUNS           130 non-null int64
T-WKTS           130 non-null int64
ODI-RUNS-S       130 non-null int64
ODI-SR-B         130 non-null float64
ODI-WKTS         130 non-null int64
ODI-SR-BL        130 non-null float64
CAPTAINCY EXP    130 non-null int64
RUNS-S           130 non-null int64
HS               130 non-null int64
AVE              130 non-null float64
SR-B             130 non-null float64
SIXERS           130 non-null int64
RUNS-C           130 non-null int64
WKTS             130 non-null int64
AVE-BL           130 non-null float64
ECON             130 non-null float64
SR-BL            130 non-null float64
AUCTION YEAR     130 non-null int64
BASE PRICE       130 non-null int64
SOLD PRICE       130 non-null int64
dtypes: float64(7), int64(15), object(4)
memory usage: 26.5+ KB
```

2.1.5 Slicing and Indexing a dataframe

Selecting Rows by Indexes

```
ipl_auction_df[0:5]
```

	Sl.NO.	PLAYER NAME	AGE	...	AUCTION YEAR	BASE PRICE	SOLD PRICE
0	1	Abdulla, YA	2	...	2009	50000	50000
1	2	Abdur Razzak	2	...	2008	50000	50000
2	3	Agarkar, AB	2	...	2008	200000	350000
3	4	Ashwin, R	1	...	2011	100000	850000
4	5	Badrinath, S	2	...	2011	100000	800000

5 rows × 26 columns

```
ipl_auction_df[-5:]
```

	Sl.NO.	PLAYER NAME	AGE	...	AUCTION YEAR	BASE PRICE	SOLD PRICE
125	126	Yadav, AS	2	...	2010	50000	750000
126	127	Younis Khan	2	...	2008	225000	225000
127	128	Yuvraj Singh	2	...	2011	400000	1800000
128	129	Zaheer Khan	2	...	2008	200000	450000
129	130	Zoysa, DNT	2	...	2008	100000	110000

5 rows × 26 columns

Selecting Columns by Column Names

```
ipl_auction_df['PLAYER NAME'][0:5]
```

```
0      Abdulla, YA
1     Abdur Razzak
2      Agarkar, AB
3        Ashwin, R
4     Badrinath, S
Name: PLAYER NAME, dtype: object
```

```
ipl_auction_df[['PLAYER NAME', 'COUNTRY']][0:5]
```

	PLAYER NAME	COUNTRY
0	Abdulla, YA	SA
1	Abdur Razzak	BAN
2	Agarkar, AB	IND
3	Ashwin, R	IND
4	Badrinath, S	IND

Selecting Rows and Columns by indexes

```
ipl_auction_df.iloc[4:9, 1:4]
```

	PLAYER NAME	AGE	COUNTRY
4	Badrinath, S	2	IND
5	Bailey, GJ	2	AUS
6	Balaji, L	2	IND
7	Bollinger, DE	2	AUS
8	Botha, J	2	SA

2.1.6 Value Counts and Cross Tabulations

Finding Unique Occurances of Values in Columns

```
ipl_auction_df.COUNTRY.value_counts()
```

```
IND    53
AUS    22
SA     16
SL     12
PAK     9
NZ      7
WI      6
ENG     3
ZIM     1
BAN     1
Name: COUNTRY, dtype: int64
```

```
ipl_auction_df.COUNTRY.value_counts(normalize=True)*100
```

```
IND   40.769
AUS   16.923
SA    12.308
SL     9.231
PAK    6.923
NZ     5.385
WI     4.615
ENG    2.308
ZIM    0.769
BAN    0.769
Name: COUNTRY, dtype: float64
```

Cross-tabulation between two columns

```
pd.crosstab( ipl_auction_df['AGE'], ipl_auction_df['PLAYING ROLE'] )
```

PLAYING ROLE	Allrounder	Batsman	Bowler	W. Keeper
AGE				
1	4	5	7	0
2	25	21	29	11
3	6	13	8	1

2.1.7 Sorting dataframe by column values

```
ipl_auction_df[['PLAYER NAME', 'SOLD PRICE']].sort_values('SOLD PRICE')[0:5]
```

	PLAYER NAME	SOLD PRICE
73	Noffke, AA	20000
46	Kamran Khan	24000
0	Abdulla, YA	50000
1	Abdur Razzak	50000
118	Van der Merwe	50000

```
ipl_auction_df[['PLAYER NAME', 'SOLD PRICE']].sort_values('SOLD PRICE', ascendin
g = False)[0:5]
```

	PLAYER NAME	SOLD PRICE
93	Sehwag, V	1800000
127	Yuvraj Singh	1800000
50	Kohli, V	1800000
111	Tendulkar, SR	1800000
113	Tiwary, SS	1600000

2.1.8 Creating new columns

Which player got the maximum premium on the base price?

```
ipl_auction_df['premium'] = ipl_auction_df['SOLD PRICE'] - ipl_auction_df['BASE
 PRICE']
```

```
ipl_auction_df[['PLAYER NAME', 'BASE PRICE', 'SOLD PRICE', 'premium']][0:5]
```

	PLAYER NAME	BASE PRICE	SOLD PRICE	premium
0	Abdulla, YA	50000	50000	0
1	Abdur Razzak	50000	50000	0
2	Agarkar, AB	200000	350000	150000
3	Ashwin, R	100000	850000	750000
4	Badrinath, S	100000	800000	700000

Which players got the maximum premium offering on their base price?

```
ipl_auction_df[['PLAYER NAME',
                'BASE PRICE',
                'SOLD PRICE', 'premium']].sort_values('premium',
                                                      ascending = False)[0:5]
```

	PLAYER NAME	BASE PRICE	SOLD PRICE	premium
50	Kohli, V	150000	1800000	1650000
113	Tiwary, SS	100000	1600000	1500000
127	Yuvraj Singh	400000	1800000	1400000
111	Tendulkar, SR	400000	1800000	1400000
93	Sehwag, V	400000	1800000	1400000

2.1.9 Grouping and Aggregating

What is the average SOLD PRICE for each age category?

```
ipl_auction_df.groupby('AGE')['SOLD PRICE'].mean()
```

```
AGE
1    720250.000
2    484534.884
3    520178.571
Name: SOLD PRICE, dtype: float64
```

```
soldprice_by_age = ipl_auction_df.groupby('AGE')['SOLD PRICE'].mean().reset_inde
x()
soldprice_by_age
```

	AGE	SOLD PRICE
0	1	720250.000
1	2	484534.884
2	3	520178.571

Average SOLD PRICE for Different Playing Roles in Each Age Category?

```
soldprice_by_age_role = ipl_auction_df.groupby(['AGE', 'PLAYING ROLE'])['SOLD PR
ICE'].mean().reset_index()
soldprice_by_age_role
```

	AGE	PLAYING ROLE	SOLD PRICE
0	1	Allrounder	587500.000
1	1	Batsman	1110000.000
2	1	Bowler	517714.286
3	2	Allrounder	449400.000
4	2	Batsman	654761.905
5	2	Bowler	397931.034
6	2	W. Keeper	467727.273
7	3	Allrounder	766666.667
8	3	Batsman	457692.308
9	3	Bowler	414375.000
10	3	W. Keeper	700000.000

2.1.10 Joining dataframes

Compare the average auction price for different ages and playing roles.

```
soldprice_comparison = soldprice_by_age_role.merge( soldprice_by_age,
                                                    on = 'AGE',
                                                    how = 'outer')
```

```
soldprice_comparison
```

	AGE	PLAYING ROLE	SOLD PRICE_x	SOLD PRICE_y
0	1	Allrounder	587500.000	720250.000
1	1	Batsman	1110000.000	720250.000
2	1	Bowler	517714.286	720250.000
3	2	Allrounder	449400.000	484534.884
4	2	Batsman	654761.905	484534.884
5	2	Bowler	397931.034	484534.884
6	2	W. Keeper	467727.273	484534.884
7	3	Allrounder	766666.667	520178.571
8	3	Batsman	457692.308	520178.571
9	3	Bowler	414375.000	520178.571
10	3	W. Keeper	700000.000	520178.571

2.1.11 Re-naming columns

```
soldprice_comparison.rename( columns = { 'SOLD PRICE_x': 'SOLD_PRICE_AGE_ROLE',
                                         'SOLD PRICE_y': 'SOLD_PRICE_AGE' }, inplace = Tru
e  )
```

```
soldprice_comparison.head(5)
```

	AGE	PLAYING ROLE	SOLD_PRICE_AGE_ROLE	SOLD_PRICE_AGE
0	1	Allrounder	587500.000	720250.000
1	1	Batsman	1110000.000	720250.000
2	1	Bowler	517714.286	720250.000
3	2	Allrounder	449400.000	484534.884
4	2	Batsman	654761.905	484534.884

2.1.12 Applying Operations to multiple columns

Percentage change in SOLD PRICE

```
soldprice_comparison['change'] = soldprice_comparison.apply(lambda rec:
                               (rec.SOLD_PRICE_AGE_ROLE - rec.SOLD_PRICE_AGE) / rec.SOL
D_PRICE_AGE,
                               axis = 1)
```

```
soldprice_comparison
```

	AGE	PLAYING ROLE	SOLD_PRICE_AGE_ROLE	SOLD_PRICE_AGE	change
0	1	Allrounder	587500.000	720250.000	-0.184
1	1	Batsman	1110000.000	720250.000	0.541
2	1	Bowler	517714.286	720250.000	-0.281
3	2	Allrounder	449400.000	484534.884	-0.073
4	2	Batsman	654761.905	484534.884	0.351
5	2	Bowler	397931.034	484534.884	-0.179
6	2	W. Keeper	467727.273	484534.884	-0.035
7	3	Allrounder	766666.667	520178.571	0.474
8	3	Batsman	457692.308	520178.571	-0.120
9	3	Bowler	414375.000	520178.571	-0.203
10	3	W. Keeper	700000.000	520178.571	0.346

2.1.13 Filtering Records from Dataframe based on conditions

Which players have hit more then 80 sixes in the IPL tournament so far?

```
ipl_auction_df[ipl_auction_df['SIXERS'] > 80 ][['PLAYER NAME', 'SIXERS']]
```

	PLAYER NAME	SIXERS
26	Gayle, CH	129
28	Gilchrist, AC	86
82	Pathan, YK	81
88	Raina, SK	97
97	Sharma, RG	82

2.1.14 Removing a column

```
ipl_auction_df.drop( 'Sl.NO.', inplace = True, axis = 1)
ipl_auction_df.columns
```

```
Index(['PLAYER NAME', 'AGE', 'COUNTRY', 'TEAM', 'PLAYING ROLE', 'T-R
UNS',
       'T-WKTS', 'ODI-RUNS-S', 'ODI-SR-B', 'ODI-WKTS', 'ODI-SR-BL',
       'CAPTAINCY EXP', 'RUNS-S', 'HS', 'AVE', 'SR-B', 'SIXERS', 'RU
NS-C',
       'WKTS', 'AVE-BL', 'ECON', 'SR-BL', 'AUCTION YEAR', 'BASE PRIC
E',
       'SOLD PRICE', 'premium'],
      dtype='object')
```

2.2 Dealing With Missing Values

```
autos = pd.read_csv( 'auto-mpg.data',
                     sep= '\s+',
                     header = None)
autos.head( 5 )
```

	0	1	2	...	6	7	8
0	18.000	8	307.000	...	70	1	chevrolet chevelle malibu
1	15.000	8	350.000	...	70	1	buick skylark 320
2	18.000	8	318.000	...	70	1	plymouth satellite
3	16.000	8	304.000	...	70	1	amc rebel sst
4	17.000	8	302.000	...	70	1	ford torino

5 rows × 9 columns

```
autos.columns = ['mpg','cylinders', 'displacement',
                 'horsepower', 'weight', 'acceleration',
                 'year', 'origin', 'name']

autos.head( 5 )
```

	mpg	cylinders	displacement	...	year	origin	name
0	18.000	8	307.000	...	70	1	chevrolet chevelle malibu
1	15.000	8	350.000	...	70	1	buick skylark 320
2	18.000	8	318.000	...	70	1	plymouth satellite
3	16.000	8	304.000	...	70	1	amc rebel sst
4	17.000	8	302.000	...	70	1	ford torino

5 rows × 9 columns

Now, we will look at the schema of the datframe.

```
autos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
mpg             398 non-null float64
cylinders       398 non-null int64
displacement    398 non-null float64
horsepower      398 non-null object
weight          398 non-null float64
acceleration    398 non-null float64
year            398 non-null int64
origin          398 non-null int64
name            398 non-null object
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
```

```
autos["horsepower"] = pd.to_numeric( autos["horsepower"], errors = 'corece' )
autos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
mpg             398 non-null float64
cylinders       398 non-null int64
displacement    398 non-null float64
horsepower      392 non-null float64
weight          398 non-null float64
acceleration    398 non-null float64
year            398 non-null int64
origin          398 non-null int64
name            398 non-null object
dtypes: float64(5), int64(3), object(1)
memory usage: 28.1+ KB
```

```
autos[autos.horsepower.isnull()]
```

	mpg	cylinders	displacement	...	year	origin	name
32	25.000	4	98.000	...	71	1	ford pinto
126	21.000	6	200.000	...	74	1	ford maverick
330	40.900	4	85.000	...	80	2	renault lecar deluxe
336	23.600	4	140.000	...	80	1	ford mustang cobra
354	34.500	4	100.000	...	81	2	renault 18i
374	23.000	4	151.000	...	82	1	amc concord dl

6 rows × 9 columns

```
autos = autos.dropna(subset = ['horsepower'])
```

```
autos[autos.horsepower.isnull()]
```

	mpg	cylinders	displacement	...	year	origin	name

0 rows × 9 columns

2.3 Exploration using Visualization Plots

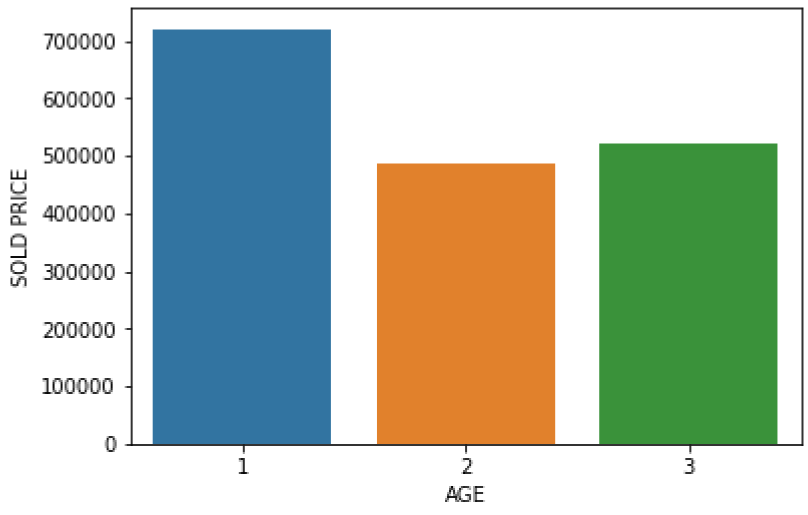
2.3.1 Drawing Plots

```
import matplotlib.pyplot as plt
import seaborn as sn
%matplotlib inline
```

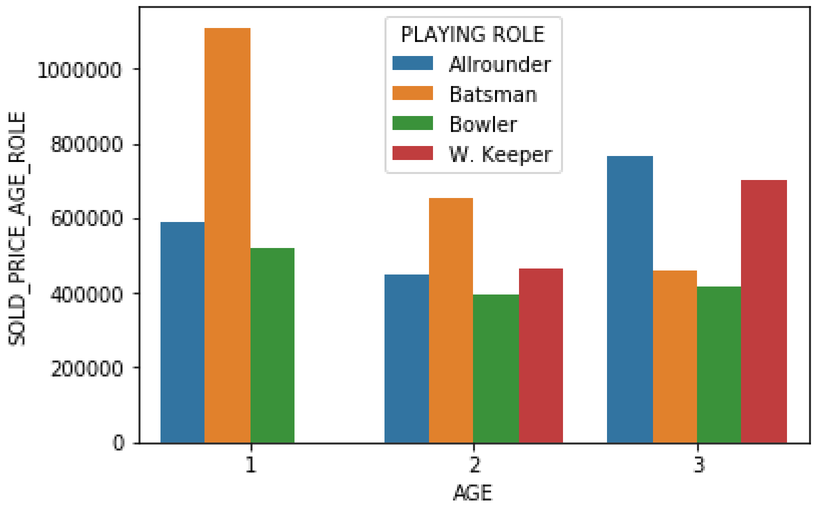
```
import warnings
warnings.filterwarnings('ignore')
```

2.3.2 Bar Plot

```
sn.barplot(x = 'AGE', y = 'SOLD PRICE', data = soldprice_by_age);
```

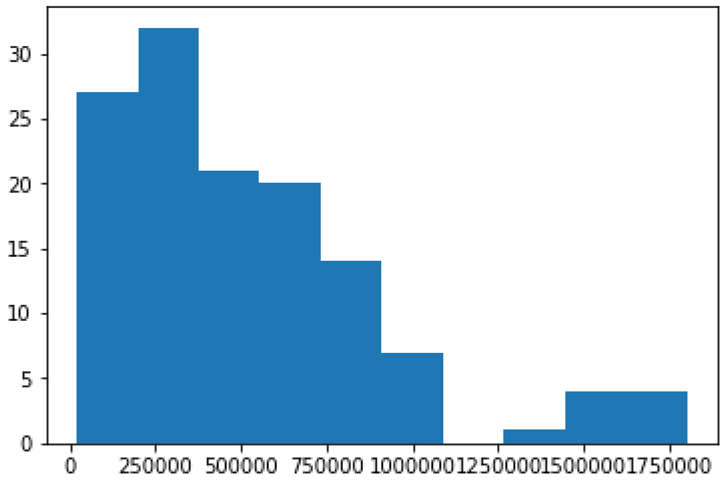


```
sn.barplot(x = 'AGE', y = 'SOLD_PRICE_AGE_ROLE', hue = 'PLAYING ROLE', data = so
ldprice_comparison);
```

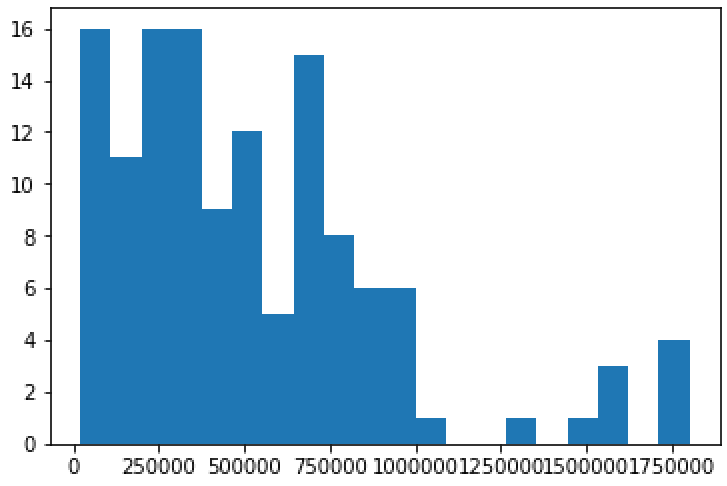


2.3.3 Histogram

```
plt.hist( ipl_auction_df['SOLD PRICE'] );
```

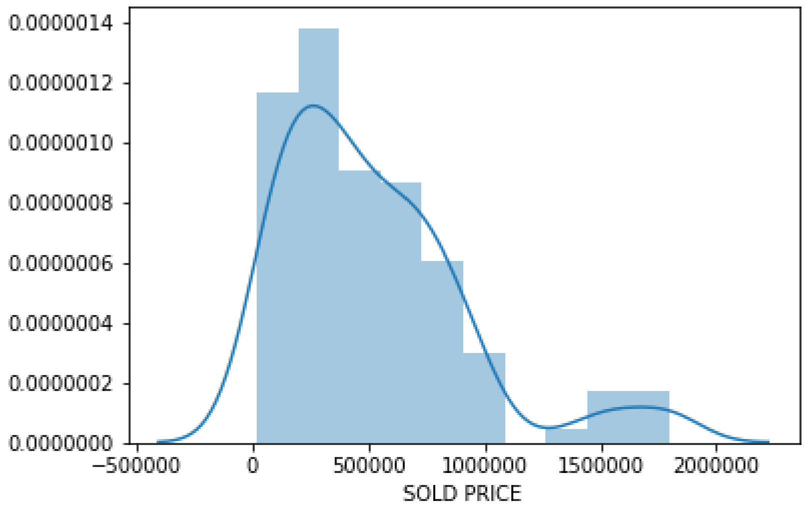


```
plt.hist( ipl_auction_df['SOLD PRICE'], bins = 20 );
```



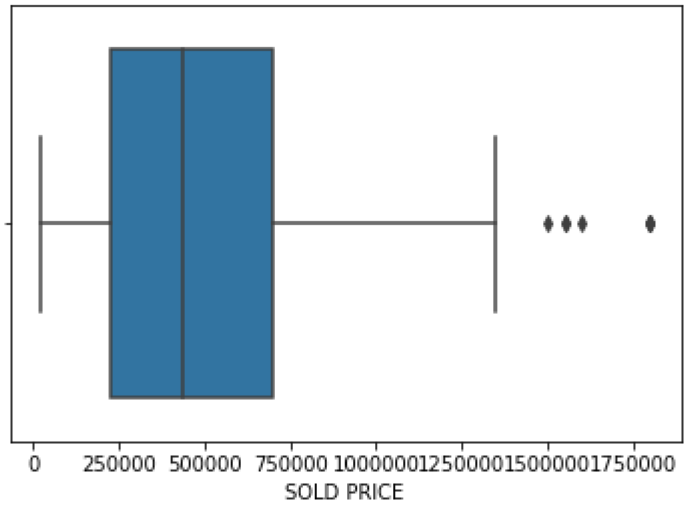
2.3.4 Distribution or Density plot

```
sn.distplot( ipl_auction_df['SOLD PRICE']);
```

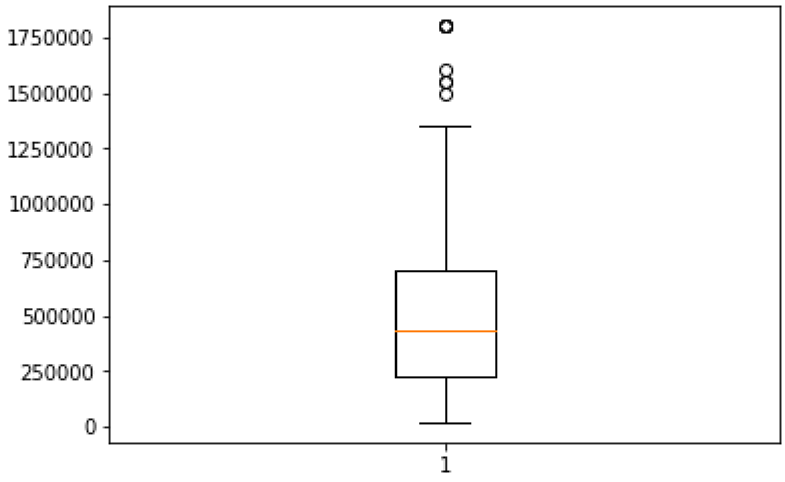


2.3.5 Box Plot

```
box = sn.boxplot(ipl_auction_df['SOLD PRICE']);
```



```
box = plt.boxplot(ipl_auction_df['SOLD PRICE']);
```



```
[item.get_ydata()[0] for item in box['caps']]
```

```
[20000.0, 1350000.0]
```

```
[item.get_ydata()[0] for item in box['whiskers']]
```

```
[225000.0, 700000.0]
```

```
[item.get_ydata()[0] for item in box['medians']]
```

```
[437500.0]
```

Who are outliers?

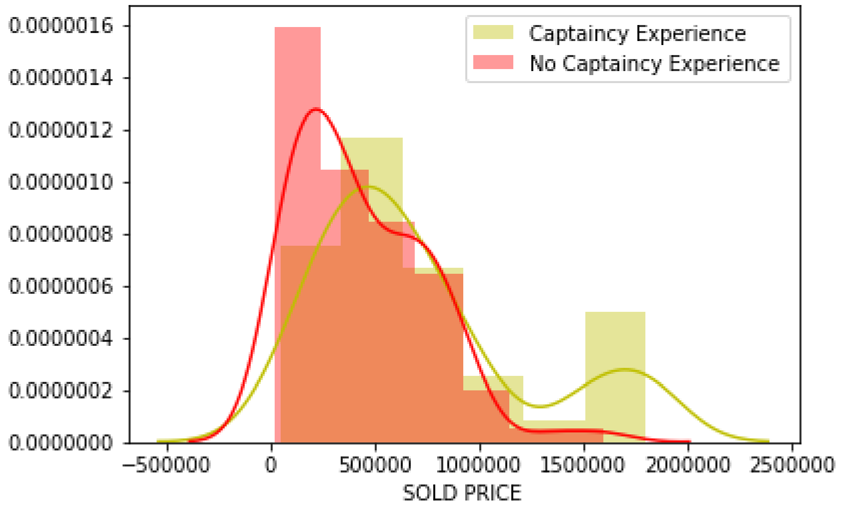
```
ipl_auction_df[ipl_auction_df['SOLD PRICE'] > 1350000.0][['PLAYER NAME',
                                                           'PLAYING ROLE',
                                                           'SOLD PRICE']]
```

	PLAYER NAME	PLAYING ROLE	SOLD PRICE
15	Dhoni, MS	W. Keeper	1500000
23	Flintoff, A	Allrounder	1550000
50	Kohli, V	Batsman	1800000
83	Pietersen, KP	Batsman	1550000
93	Sehwag, V	Batsman	1800000
111	Tendulkar, SR	Batsman	1800000
113	Tiwary, SS	Batsman	1600000
127	Yuvraj Singh	Batsman	1800000

2.3.6 Comparing Distributions

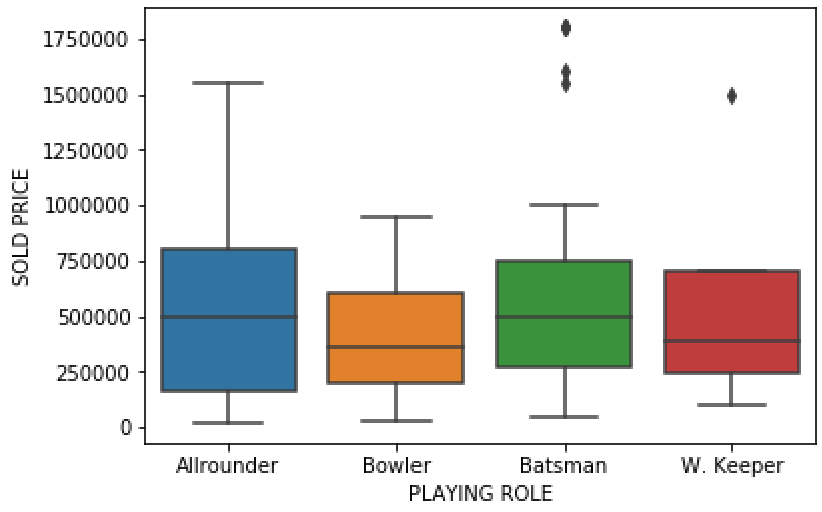
Using distribution plots

```
sn.distplot( ipl_auction_df[ipl_auction_df['CAPTAINCY EXP'] == 1]['SOLD PRICE'],
            color = 'y',
            label = 'Captaincy Experience')
sn.distplot( ipl_auction_df[ipl_auction_df['CAPTAINCY EXP'] == 0]['SOLD PRICE'],
            color = 'r',
            label = 'No Captaincy Experience');
plt.legend();
```



Using box plots

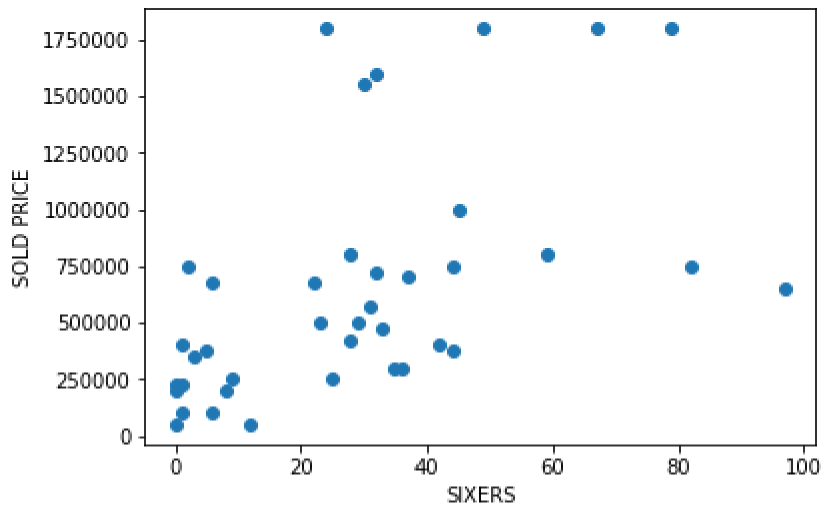
```
sn.boxplot(x = 'PLAYING ROLE', y = 'SOLD PRICE', data = ipl_auction_df);
```



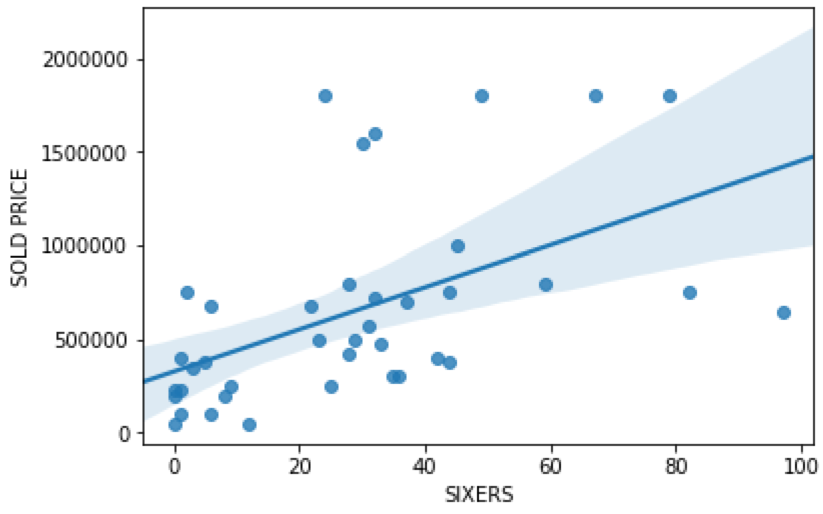
2.3.7 Scatter Plot

```
ipl_batsman_df = ipl_auction_df[ipl_auction_df['PLAYING ROLE'] == 'Batsman']
```

```
plt.scatter(x = ipl_batsman_df.SIXERS,
            y = ipl_batsman_df['SOLD PRICE']);
plt.xlabel('SIXERS')
plt.ylabel('SOLD PRICE');
```



```
sn.regplot( x = 'SIXERS',
            y = 'SOLD PRICE',
            data = ipl_batsman_df );
```

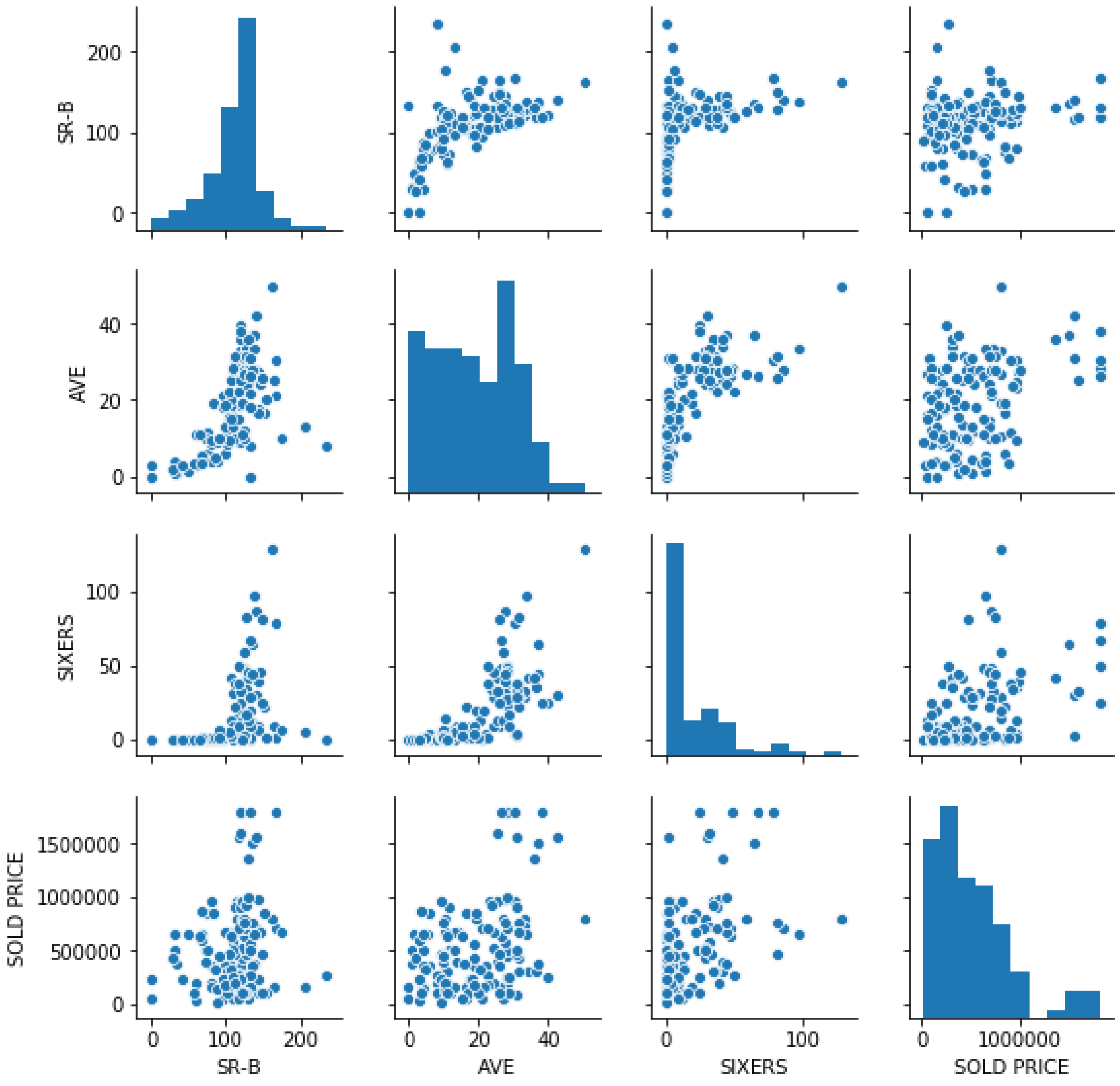


2.3.8 Pair Plot

```
influential_features = ['SR-B', 'AVE', 'SIXERS', 'SOLD PRICE']
```

```
sn.pairplot(ipl_auction_df[influential_features], size=2)
```

```
<seaborn.axisgrid.PairGrid at 0x1a1b188860>
```

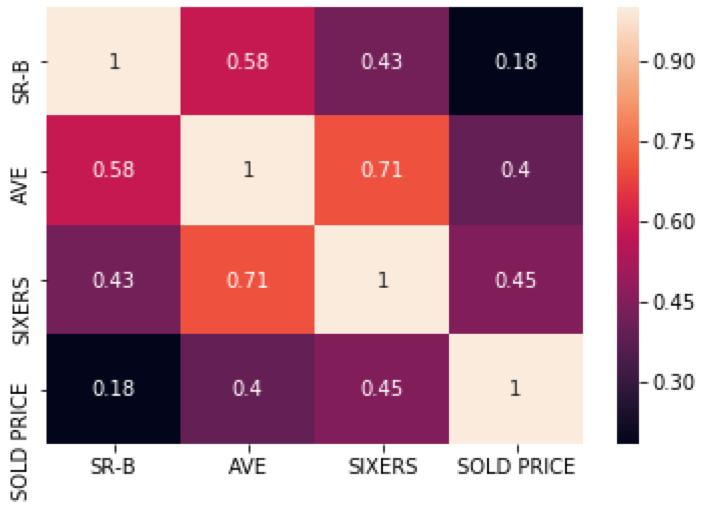


2.3.9 Correlations and Heatmaps

```
ipl_auction_df[influential_features].corr()
```

	SR-B	AVE	SIXERS	SOLD PRICE
SR-B	1.000	0.584	0.425	0.184
AVE	0.584	1.000	0.705	0.397
SIXERS	0.425	0.705	1.000	0.451
SOLD PRICE	0.184	0.397	0.451	1.000

```
sn.heatmap(ipl_auction_df[influential_features].corr(), annot=True);
```


SR-B
AVE
SIXERS
SOLD PRICE
SR-B
AVE
SIXERS
SOLD PRICE