Data Science Lab

Bi variate Analysis

Name: Manikandan P
RegNo: 2019202030

Data set: IPL

Importing required Modules:

In [38]:
```
import sklearn
import pandas
import seaborn
import matplotlib.pyplot as plt
import matplotlib
%matplotlib inline
```

In [39]:
```
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
```

Reading and Summarizing the SUV Data set:

In [3]:
```
ds=pandas.read_csv('matches.csv')
```

In [7]:
```
ds.head(10)
```

Out[7]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_
0	1	2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvra
1	2	2017	Pune	06-04-2017	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7	SPI
2	3	2017	Rajkot	07-04-2017	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10	C
3	4	2017	Indore	08-04-2017	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	6	GJ M
4	5	2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	0	KM
5	6	2017	Hyderabad	09-04-2017	Gujarat Lions	Sunrisers Hyderabad	Sunrisers Hyderabad	field	normal	0	Sunrisers Hyderabad	0	9	Rashi
6	7	2017	Mumbai	09-04-2017	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	0	4	I
7	8	2017	Indore	10-04-2017	Royal Challengers Bangalore	Kings XI Punjab	Royal Challengers Bangalore	bat	normal	0	Kings XI Punjab	0	8	A

In [8]:
```
ds.shape
```

Out[8]:
```
(756, 18)
```

In [9]:
```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               756 non-null    int64
 1   season           756 non-null    int64
 2   city             749 non-null    object
 3   date             756 non-null    object
 4   team1            756 non-null    object
 5   team2            756 non-null    object
 6   toss_winner      756 non-null    object
 7   toss_decision    756 non-null    object
 8   result           756 non-null    object
 9   dl_applied       756 non-null    int64
 10  winner           752 non-null    object
 11  win_by_runs      756 non-null    int64
 12  win_by_wickets   756 non-null    int64
 13  player_of_match  752 non-null    object
 14  venue            756 non-null    object
 15  umpire1          754 non-null    object
 16  umpire2          754 non-null    object
 17  umpire3          119 non-null    object
dtypes: int64(5), object(13)
memory usage: 106.4+ KB
```

In [6]:
```
ds.groupby('toss_decision').size()
```

Out[6]:
```
toss_decision
bat      293
field    463
dtype: int64
```

Cleaning the data set:

In [26]:
```
clean_ds=ds.drop(columns=['id', 'date', 'dl_applied', 'venue', 'player_of_match', 'umpire1', 'umpire2', 'umpire3' ], axis='1')
clean_ds.head(10)
```

Out[26]:

	season	city	team1	team2	toss_winner	toss_decision	result	winner	win_by_runs	win_by_wickets
0	2017	Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	Sunrisers Hyderabad	35	0
1	2017	Pune	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Rising Pune Supergiant	0	7
2	2017	Rajkot	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	Kolkata Knight Riders	0	10
3	2017	Indore	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	Kings XI Punjab	0	6
4	2017	Bangalore	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	Royal Challengers Bangalore	15	0
5	2017	Hyderabad	Gujarat Lions	Sunrisers Hyderabad	Sunrisers Hyderabad	field	normal	Sunrisers Hyderabad	0	9
6	2017	Mumbai	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	Mumbai Indians	0	4
7	2017	Indore	Royal Challengers Bangalore	Kings XI Punjab	Royal Challengers Bangalore	bat	normal	Kings XI Punjab	0	8
8	2017	Pune	Delhi Daredevils	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Delhi Daredevils	97	0
9	2017	Mumbai	Sunrisers Hyderabad	Mumbai Indians	Mumbai Indians	field	normal	Mumbai Indians	0	4

In [11]:
```
clean_ds.describe()
```

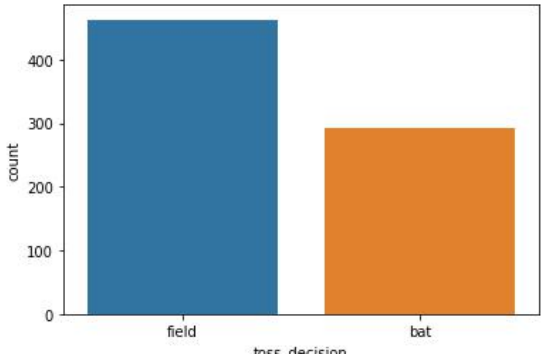
Out[11]:

	season	win_by_runs	win_by_wickets
count	756.000000	756.000000	756.000000
mean	2013.444444	13.283069	3.350529
std	3.366895	23.471144	3.387963
min	2008.000000	0.000000	0.000000
25%	2011.000000	0.000000	0.000000
50%	2013.000000	0.000000	4.000000
75%	2016.000000	19.000000	6.000000
max	2019.000000	146.000000	10.000000

Visualizing Data:

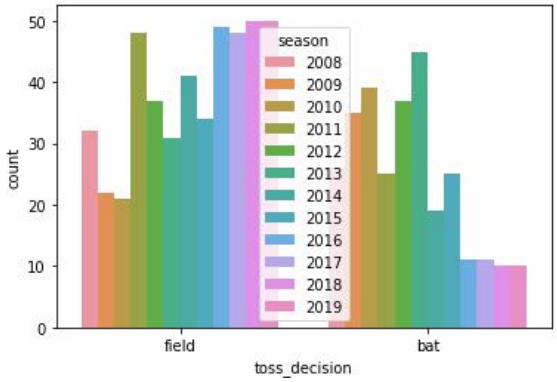
In [12]:
```
seaborn.countplot(x='toss_decision', data=clean_ds)
```

Out[12]: <AxesSubplot:xlabel='toss_decision', ylabel='count'>



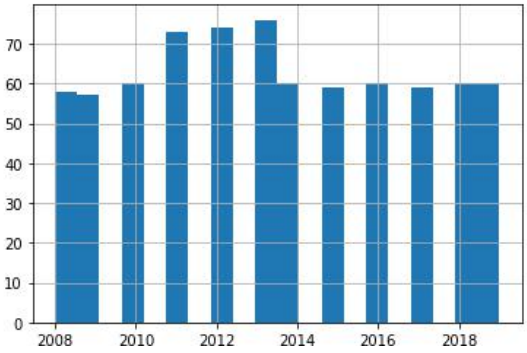
In [16]:
```
seaborn.countplot(x='toss_decision', hue='season', data=clean_ds)
```

Out[16]: <AxesSubplot:xlabel='toss_decision', ylabel='count'>



In [19]:
```
clean_ds['season'].hist(bins=20)
```

Out[19]: <AxesSubplot:>



Categorizing and Visualizing the dataset:

In [46]:
```
bin_toss = pandas.get_dummies (clean_ds ['toss_decision'])
bin_toss.head()
```

Out[46]:

	bat	field
0	0	1
1	0	1
2	0	1
3	0	1
4	1	0

In [27]:
```
home_win=[]
for i in range (0, len  (ds ['team1'])):
    if clean_ds ['team1'][i] == clean_ds ['winner'][i]:
        home_win.append (1);
    else:
        home_win.append (0);
home_win_df=pandas.DataFrame (data = home_win, columns = ['home_win'])
aug_ds = pandas.concat([clean_ds, home_win_df], axis = 1)
aug_ds.head()
```

Out[27]:

	season	city	team1	team2	toss_winner	toss_decision	result	winner	win_by_runs	win_by_wickets	home_win
0	2017	Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	Sunrisers Hyderabad	35	0	1
1	2017	Pune	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Rising Pune Supergiant	0	7	0
2	2017	Rajkot	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	Kolkata Knight Riders	0	10	0
3	2017	Indore	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	Kings XI Punjab	0	6	0
4	2017	Bangalore	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	Royal Challengers Bangalore	15	0	1

In [28]:
```
toss_match_win=[]
for i in range (0, len  (ds ['toss_winner'])):
    if clean_ds ['toss_winner'][i] == clean_ds ['winner'][i]:
        toss_match_win.append (1);
    else:
        toss_match_win.append (0);
toss_match_win_df=pandas.DataFrame (data = toss_match_win, columns = ['toss_match_win'])
aug_ds = pandas.concat([aug_ds, toss_match_win_df], axis = 1)
aug_ds.head()
```

Out[28]:

	season	city	team1	team2	toss_winner	toss_decision	result	winner	win_by_runs	win_by_wickets	home_win	toss_match_win
0	2017	Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	Sunrisers Hyderabad	35	0	1	0
1	2017	Pune	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Rising Pune Supergiant	0	7	0	1
2	2017	Rajkot	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	Kolkata Knight Riders	0	10	0	1
3	2017	Indore	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	Kings XI Punjab	0	6	0	1
4	2017	Bangalore	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	Royal Challengers Bangalore	15	0	1	1

In [30]:
```
first_bat_win=[]
for i in range (0, len  (ds ['win_by_runs'])):
    if clean_ds ['win_by_runs'][i] >0:
        first_bat_win.append (1);
    else:
        first_bat_win.append (0);
first_bat_win_df=pandas.DataFrame (data = first_bat_win, columns = ['first_bat_win'])
aug_ds = pandas.concat([aug_ds, first_bat_win_df], axis = 1)
aug_ds.head()
```

n	city	team1	team2	toss_winner	toss_decision	result	winner	win_by_runs	win_by_wickets	home_win	toss_match_win	first_bat_win
17	Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	Sunrisers Hyderabad	35	0	1	0	1
17	Pune	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Rising Pune Supergiant	0	7	0	1	0
17	Rajkot	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	Kolkata Knight Riders	0	10	0	1	0
17	Indore	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	Kings XI Punjab	0	6	0	1	0
17	Bangalore	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	Royal Challengers Bangalore	15	0	1	1	1

In [31]:
```
aug_ds=aug_ds.drop(columns=['win_by_runs', 'win_by_wickets'], axis = '1')
aug_ds.head()
```

Out[31]:

	season	city	team1	team2	toss_winner	toss_decision	result	winner	home_win	toss_match_win	first_bat_win
0	2017	Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	Sunrisers Hyderabad	1	0	1
1	2017	Pune	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	Rising Pune Supergiant	0	1	0
2	2017	Rajkot	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	Kolkata Knight Riders	0	1	0
3	2017	Indore	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	Kings XI Punjab	0	1	0
4	2017	Bangalore	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	Royal Challengers Bangalore	1	1	1

Final Dataset:

In [33]:
```
bin_res = pandas.get_dummies (aug_ds ['result'],drop_first = True)
bin_res.head ()
```

Out[33]:

	normal	tie
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0

In [48]:
```
fin_ds = pandas.concat ([aug_ds, bin_res, bin_toss], axis = 1)
fin_ds1 = fin_ds.drop (columns = ['season', 'city', 'team1', 'team2', 'toss_winner', 'toss_decision', 'winner', 'result'], axis =
fin_ds1.head()
```

Out[48]:

	home_win	toss_match_win	first_bat_win	normal	tie	bat	field
0	1	0	1	1	0	0	1
1	0	1	0	1	0	0	1
2	0	1	0	1	0	0	1
3	0	1	0	1	0	0	1
4	1	1	1	1	0	1	0

Predicting

In [51]:
```
Y = fin_ds1 ['home_win']
X = fin_ds1.drop (columns = ['home_win'], axis = 1)
X.head()
```

Out[51]:

	toss_match_win	first_bat_win	normal	tie	bat	field
0	0	1	1	0	0	1
1	1	0	1	0	0	1
2	1	0	1	0	0	1
3	1	0	1	0	0	1
4	1	1	1	0	1	0

In [52]:
```
Y.head()
```

Out[52]:
```
0    1
1    0
2    0
3    0
4    1
Name: home_win, dtype: int64
```

In [53]:
```
X_train, X_test, Y_train, Y_test = model_selection.train_test_split (X,Y, test_size = 0.2, random_state = 0)
model = LogisticRegression (solver = 'liblinear')
model.fit (X_train, Y_train)
```

Out[53]:
```
LogisticRegression(solver='liblinear')
```

\THANK YOU MAM !!\