In [30]:
```
#Importing the Libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline
```

In [94]:
```
#Importing the dataset and extracting the independent and dependent variables
df = pd.read_excel('/Users/AmrinSinghDhillon/Desktop/WIN PROJECTION.xlsx')
X = df.iloc[:, :-1].values
Y = df.iloc[:, 4].values
```

In [95]:
```
df
```

Out[95]:

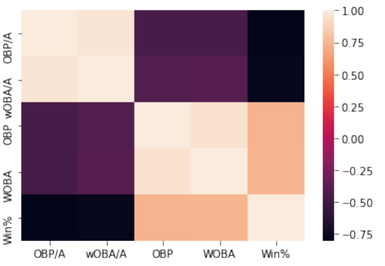
	OBP/A	wOBA/A	OBP	WOBA	Win%
0	0.370	0.346	0.369	0.322	0.471
1	0.257	0.238	0.342	0.301	0.667
2	0.389	0.366	0.347	0.316	0.375
3	0.352	0.319	0.313	0.273	0.259
4	0.366	0.340	0.370	0.358	0.635
...	...	...	...	...	...
1190	0.377	0.357	0.371	0.360	0.556
1191	0.381	0.364	0.420	0.411	0.632
1192	0.386	0.355	0.165	0.144	0.000
1193	0.405	0.372	0.327	0.294	0.294
1194	0.409	0.393	0.343	0.311	0.314

1195 rows × 5 columns

In [96]:
```
# Data Visualization
# Building the Correlation Matrix

sns.heatmap(df.corr())
```

Out[96]:
```
<AxesSubplot:>
```



In [97]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 0)
```

In [98]:
```
# Fitting Multiple Linear Regression to the Training Set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, Y_train)
```

Out[98]:
```
LinearRegression()
```

In [99]:
```
# Predicting the Test set results
Y_pred = regressor.predict(X_test)
Y_pred
```

Out[99]:
```
array([ 0.67137337,  0.2168172 ,  0.34906244,  0.3514095 ,  0.00328933,
        0.23523945,  0.50263915,  0.40148006,  0.450888  ,  0.60613003,
        0.53440034,  0.59566922,  0.562772  ,  0.56707993,  0.42113635,
        0.05927615,  0.56111657,  0.54808555,  0.47380325,  0.60696657,
        0.32899625,  0.71239991,  0.38410334,  0.21774671,  0.54117466,
        0.50254514,  0.29727995,  0.65584906,  0.60320996,  0.2822188 ,
        0.62933899,  0.5195144 ,  0.41813397,  0.60080589,  0.38276699,
        0.71330904, -0.17738373,  0.41377426,  0.38142284,  0.48503803,
        0.32002892,  0.50137446,  0.42166697,  0.39206135,  0.46584159,
        0.27770931,  0.41744291,  0.49272122,  0.5562451 ,  0.56878463,
        0.55464405,  0.59721223,  0.42368639,  0.43177234,  0.58318052,
        0.62423844,  0.33984847,  0.52886249,  0.4940466 ,  0.41560632,
        0.32636248,  0.45973146,  0.28991858,  0.68514469,  0.73005789,
        0.6755471 ,  0.4627083 ,  0.37815227,  0.68347596,  0.29683389,
        0.50667898,  0.4883094 ,  0.71147294,  0.66516156,  0.35923898,
        0.54131821,  0.59004702,  0.57904441,  0.60875575,  0.54818831,
        0.4588762 , -0.05815268,  0.44843474,  0.2861597 ,  0.42874438,
        0.38728439,  0.48606502,  0.49912043,  0.02506574,  0.5913014 ,
        0.65468014,  0.48730561,  0.57363057,  0.48026804,  0.4474844 ,
        0.3956878 ,  0.62743401,  0.51731901,  0.63917793, -0.04409421,
        0.51865461,  0.02634913,  0.52259591,  0.43258047,  0.47273761,
        0.277246  ,  0.55159054,  1.02417586,  0.52973741,  0.58032072,
        0.58656392,  0.58727552,  0.40377037,  0.45030728,  0.49402631,
        0.64561656,  0.29650316,  0.50344238,  0.09022441,  0.33822233,
        0.46140354,  0.60821284,  0.5832243 ,  0.40228133,  0.66494865,
        0.10910973,  0.47463941,  0.35983095,  0.57900872,  0.56644646,
       -0.13994031,  0.28017315,  0.28125742,  0.49878031,  0.47556208,
        0.6794441 ,  0.55755691,  0.48087383,  0.40624968,  0.65203589,
        0.24101484,  0.29008457,  0.4765687 ,  0.46485541,  0.36628674,
        0.4707133 ,  0.3938028 ,  0.40536486,  0.36863431,  0.36422588,
        0.22521929,  0.7824724 ,  0.4720189 ,  0.76972617,  0.60102705,
        0.57538155,  0.38485757,  0.6245913 ,  0.61587792,  0.6597548 ,
        0.44650251,  0.5330001 ,  0.47598659,  0.64689858,  0.43105785,
        0.48587178,  0.52916486,  0.30416248,  0.63044076,  0.42868019,
        0.2951297 ,  0.68600161,  0.46469495,  0.56381471,  0.37084732,
        0.33065832,  0.49226819,  0.3942613 ,  0.6594788 ,  0.30234656,
        0.45358092,  0.55225095,  0.61962105,  0.59342478,  0.40237749,
        0.71949548,  0.65760226,  0.41341749,  0.52765706,  0.42873865,
        0.59683439,  0.52230843,  0.65859542,  0.53710955,  0.83008661,
        0.50460045,  0.24947226,  0.48281353,  0.45710255,  0.61554034,
        0.36895453,  0.55843217,  0.50634352,  0.70870136,  0.18409739,
        0.51210113,  0.62973993,  0.36856374,  0.66950559,  0.46880813,
        0.5201429 ,  0.35937278,  0.59697229,  0.35152246,  0.48446665,
        0.60776087,  0.5250647 ,  0.53607278,  0.3523397 ,  0.40747396,
        0.33225163,  0.37216528,  0.40006248,  0.64225571,  0.55037953,
        0.36346249,  0.61050105,  0.3278604 ,  0.25142693,  0.37357155,
        0.65869238,  0.47593619,  0.44939549,  0.73799148,  0.50199797,
        0.44027499,  0.58555693,  0.42051464,  0.54336156])
```

In [100…
```
# Calculating the Coefficients
print(regressor.coef_)
```

```
[-1.35836195 -1.31809666  0.55923219  2.0718312 ]
```

In [101…
```
# Calculating the Intercept
print(regressor.intercept_)
```

```
0.5339290390514896
```

In [102…
```
# Calculating the R squared value
from sklearn.metrics import r2_score
r2_score(Y_test, Y_pred)
```

Out[102…
```
0.8350511143906414
```