In [1]:

```
import os
import pandas as pd
from sklearn.metrics import r2_score
from rfpimp import permutation_importances
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:

```
#os.chdir('/Users/apple/Desktop/Fall2019/ANLY503/HW/Project')
```

In [4]:

```
# read in dataset
team_df = pd.read_csv("teams_stat.csv")
team_df.head(3)
```

Out[4]:

	Unnamed: 0	gmDate	gmTime	seasTyp	offLNm1	offFNm1	offLNm2	offFNm2	teamAbbr
0	0	2017-10-17	08:00	Regular	Forte	Brian	Smith	Michael	BOS
1	1	2017-10-17	08:00	Regular	Forte	Brian	Smith	Michael	CLE
2	2	2017-10-17	10:30	Regular	Maddox	Tre	Garretson	Ron	HOU

3 rows × 122 columns

In [5]:

```
train_cols = ['teamMin',
 'teamDayOff',
 'teamAST',
 'teamTO',
 'teamSTL',
 'teamBLK',
 'teamPF',
 'teamFGA',
 'teamFG%',
 'team2PA',
 'team2P%',
 'team3PA',
 'team3P%',
 'teamFTA',
 'teamFT%',
 'teamORB',
 'teamDRB',
 'teamTRB']
target_col = 'teamRslt'
```

In [3]:

```
# Create Features & Target
X = team_df[train_cols]
y = team_df[target_col].apply(lambda x: 1 if x=='Win' else 0) # Target Encoding
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size = 0.8, ran
dom_state = 0)
```

In [84]:

```
# train model
rf = RandomForestRegressor(n_estimators = 100,
                           oob_score = True,
                           bootstrap = True,
                           random_state = 0)
rf.fit(X_train, y_train)
```

Out[84]:

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=Non
e,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=No
ne,
           oob_score=True, random_state=0, verbose=0, warm_start=Fal
se)
```

In [85]:

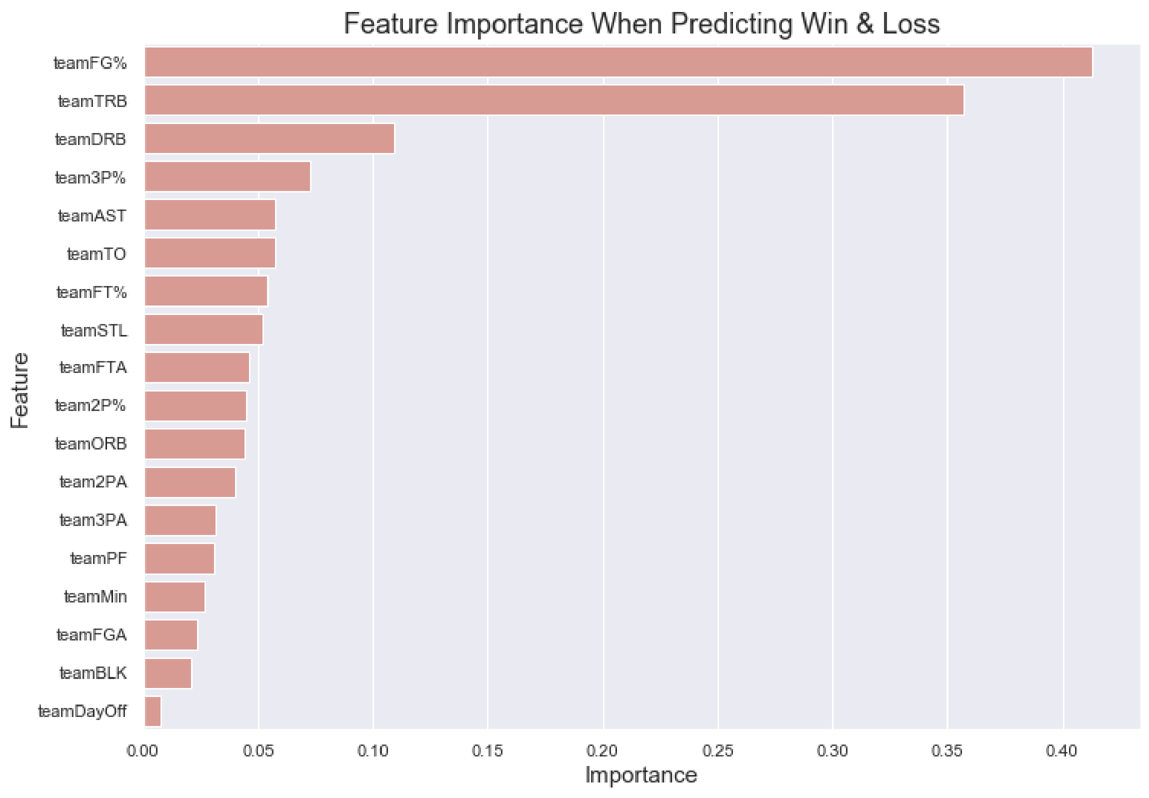
```
# plot feature importance
def r2(rf, X_train, y_train):
    return r2_score(y_train, rf.predict(X_train))

perm_imp_rfpimp = permutation_importances(rf, X_train, y_train, r2)
imp_plot = perm_imp_rfpimp.reset_index()
imp_plot.head()
```

Out[85]:

	Feature	Importance
0	teamFG%	0.413169
1	teamTRB	0.356923
2	teamDRB	0.109244
3	team3P%	0.072524
4	teamAST	0.057284

In [87]:

```
# Plot Feature Importance
sns.set(rc={'figure.figsize':(11.7, 8.27)})
fig, ax = plt.subplots( nrows=1, ncols=1 )
ax= sns.barplot(x='Importance', y="Feature", data=imp_plot,color="salmon",  satu
ration=.5)
ax.set_title('Feature Importance When Predicting Win & Loss').set_fontsize(18)
ax.set_xlabel('Importance').set_fontsize(15)
ax.set_ylabel('Feature').set_fontsize(15)
fig.savefig('Result_Prediction_Feature_Importance')   # save the figure to file
```



In [ ]: