

For this week's assignment we're going to use Dodgers Major League Baseball data from 2012. The data file you will be using is contained in the dodgers.csv file. I would like you to determine what night would be the best to run a marketing promotion to increase attendance. It is up to you if you decide to recommend a specific day or if you recommend a day of the week (e.g., Tuesdays) or month and day of the week (e.g., July Tuesdays). Use R and/or Python to accomplish this assignment. It is important to remember, there will be lots of ways to solve this problem. Explain your thought process and how you used various techniques to come up with your recommendation.

Reading in data

```
In [138]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
```

```
In [138]: #dodgers.csv file
dodgers_pd = pd.read_csv('C:/Users/phil1/OneDrive/Desktop/Predictive-630/data/dodgers/dodgers_pd.head()
```

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bobblehead
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	NO
1	APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	NO
2	APR	12	28328	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	NO
3	APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	NO
4	APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	NO

Identifying Missing Data

```
In [138]: #looking at the data when reviews.val.title is NULL --> empty review
dodgers_pd.isnull().sum()
```

```
Out[138]: month          0
day          0
attend       0
day_of_week  0
opponent     0
temp        15
skies        0
day_night    0
cap          0
shirt        0
fireworks    0
bobblehead   0
dtype: int64
```

There is no missing data in the DataFrame!

Building Promo Variable

In looking over the data and thinking about our problem statement, I wanted to build a derivative variable that denoted if any promotion was offered at a given Dodgers game. Our original dataset had a 'YES' or 'NO' marker for the various promotion items such as shirt, cap, etc. However, for the problem, I want to know plainly whether a promotion should be offered on a given day. Therefore, I just want a variable that marks the records with games that offered promotions or did not offer promotions.

```
In [139]: #function to determine if a promotion was overall offered at a Dodgers game
def mark_promo(df):
    #if any 'YES' values among the promotions, assign a 'YES' for the new column
    if df['cap'] == 'YES' or df['shirt'] == 'YES' or df['fireworks'] == 'YES' or df['fireworks'] == 'YES' or df['fireworks'] == 'YES':
        df['promo'] = 'YES'
    else:
        df['promo'] = 'NO'
    return df
```

```
In [139]: #apply the mark_promo function to the dataframe to create the new 'promo' column
dodgers_pd['promo'] = dodgers_pd.apply(mark_promo,axis=1)
```

Creating Promo & Day of Week Interaction Term

Similar to above, as related to the problem statement, now that we have a marker for games with and without promotions, I also want to explore the interaction between day of the week and offering a promotion. Therefore, I want to create a new column with those possibilities of day of the week and promotion offering to explore in the regression modeling.

```
In [139]: #creating a new column 'day_promo'
dodgers_pd['day_promo'] = dodgers_pd['day_of_week'] + dodgers_pd['promo']
```

```
Out[139]: dodgers_pd['day_promo'].head()
```

```
Out[139]: 0      TuesdayNO
1      WednesdayNO
2      ThursdayNO
3      FridayYES
4      SaturdayNO
Name: day_promo, dtype: object
```

```
In [139]: dodgers_pd['day_promo'].value_counts()
```

```
Out[139]: FridayYES      13
MondayNO      11
SaturdayNO      11
WednesdayNO      10
SundayNO      10
TuesdayYES      8
TuesdayNO      5
SundayYES      3
ThursdayNO      3
ThursdayYES      2
SaturdayYES      2
WednesdayYES      2
MondayYES      1
Name: day_promo, dtype: int64
```

Friday games with promotions hold the maximum number of data values, with 13 out of 81 games. The other leading values are days where promotions weren't offered. Therefore, we will have to do some digging to see which days will bring the highest increase in attendance when promotions are given.

From looking initially at these value counts, one would think that promotions should be offered on Fridays as they have the highest representation in the dataset. However, what is the change in attendance that they bring?

Descriptive Statistics

```
In [139]: #describing the overall dataframe
#retrieving rows and columns
print("The dimension of the table is: ",dodgers_pd.shape)
print("81 rows and 14 columns")
```

The dimension of the table is: (81, 14)

81 rows and 14 columns

```
In [139]: dodgers_pd.dtypes
```

```
Out[139]: month          object
day          int64
attend       int64
day_of_week  object
opponent     object
temp         int64
skies        object
day_night    object
cap          object
shirt        object
fireworks    object
bobblehead   object
promo        object
day_promo    object
dtype: object
```

Numerical variables --> day, attend, temp

Categorical variables --> month, day_of_week, opponent, skies, day_night, cap, shirt, fireworks, bobblehead, day_promo, promo

Describing the Date/Time Variables

```
In [139]: #looking at date features for prediction
dates_targets = dodgers_pd[['month', 'day', 'day_of_week', 'day_night']]
dates_targets
```

	month	day	day_of_week	day_night
0	APR	10	Tuesday	Day
1	APR	11	Wednesday	Night
2	APR	12	Thursday	Night
3	APR	13	Friday	Night
4	APR	14	Saturday	Night
...
76	SEP	29	Saturday	Night
77	SEP	30	Sunday	Day
78	OCT	1	Monday	Night
79	OCT	2	Tuesday	Night
80	OCT	3	Wednesday	Night

81 rows x 4 columns

```
In [139]: #describe the data
desc = dates_targets.describe()
desc
```

```
Out[139]: day
count      81.000000
mean      16.135802
std        9.605666
min         1.000000
25%        8.000000
50%       15.000000
75%       25.000000
max       31.000000
```

This result shows the summary statistics for the 'day' variable in the dataset, which just marks the day of the month for the corresponding Dodgers MLB game. Given that there are usually around 30-31 days in a month, these statistics don't give us very much information around the target variable. This variable will be more useful when combined with the other date/time variables!

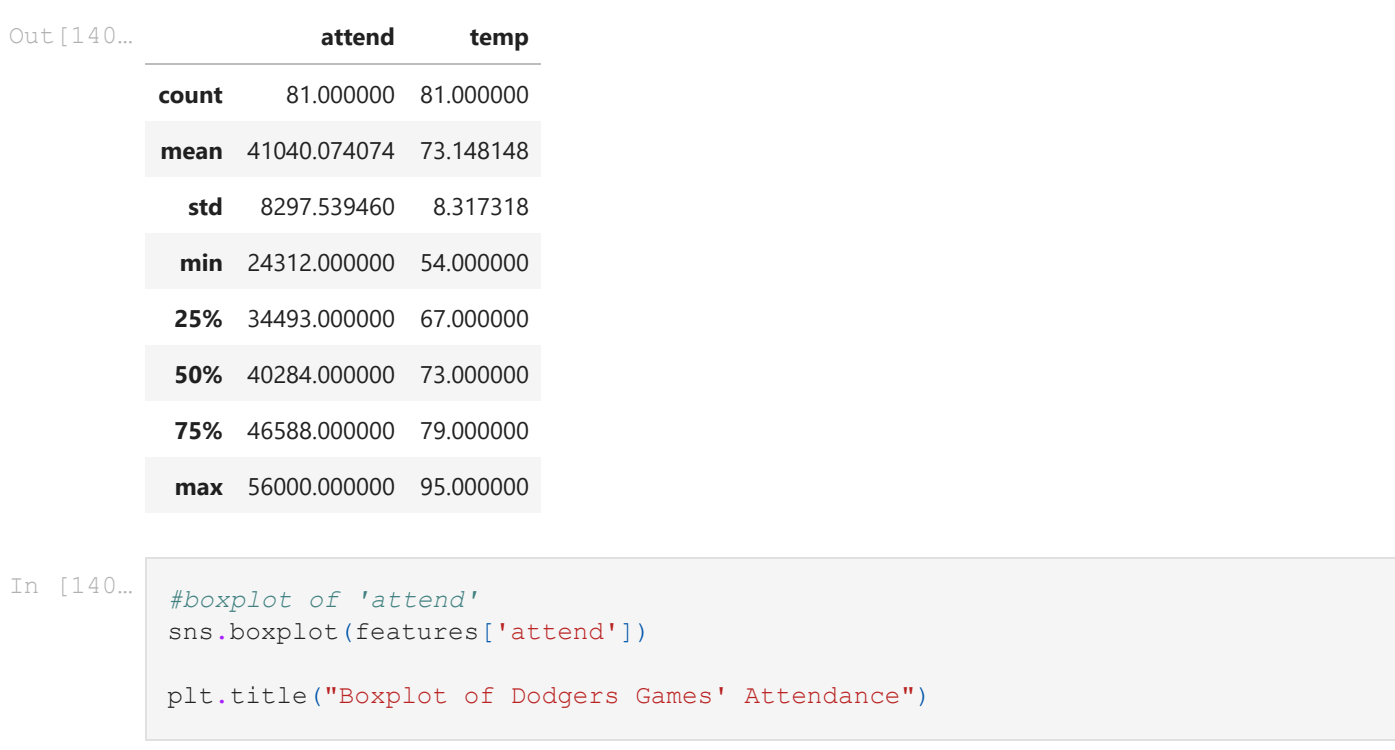
```
In [139]: dates_targets['day_night'].value_counts()
```

```
Out[139]: Night      66
Day       15
Name: day_night, dtype: int64
```

```
In [140]: #boxplot of 'day'
sns.boxplot(dates_targets['day'])
plt.title("Boxplot of Game Days in a Month")
```

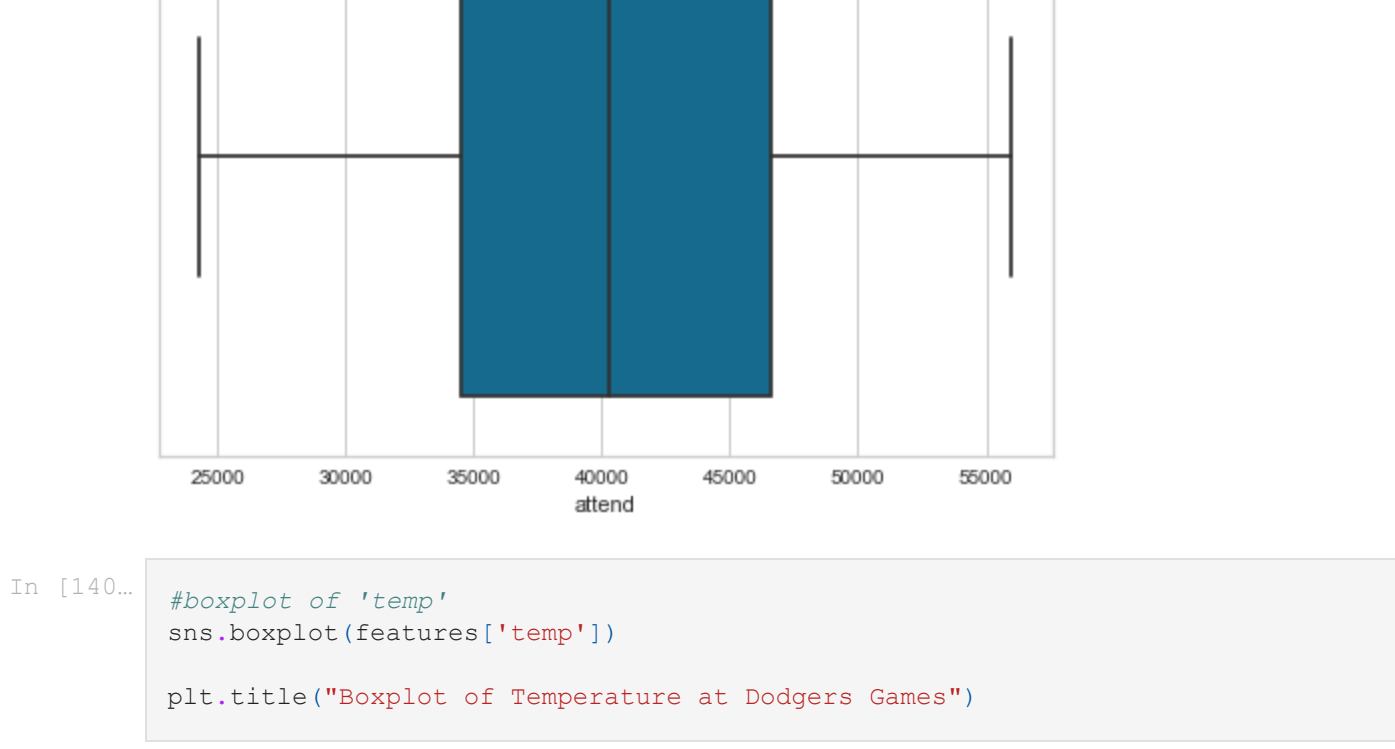
C:\Users\phil1\Anaconda3\lib\site-packages\seaborn\decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be "data", and passing other arguments without an explicit keyword will result in an error or misinterpretation.

warnings.warn(
Text(0.5, 1.0, "Boxplot of Game Days in a Month")



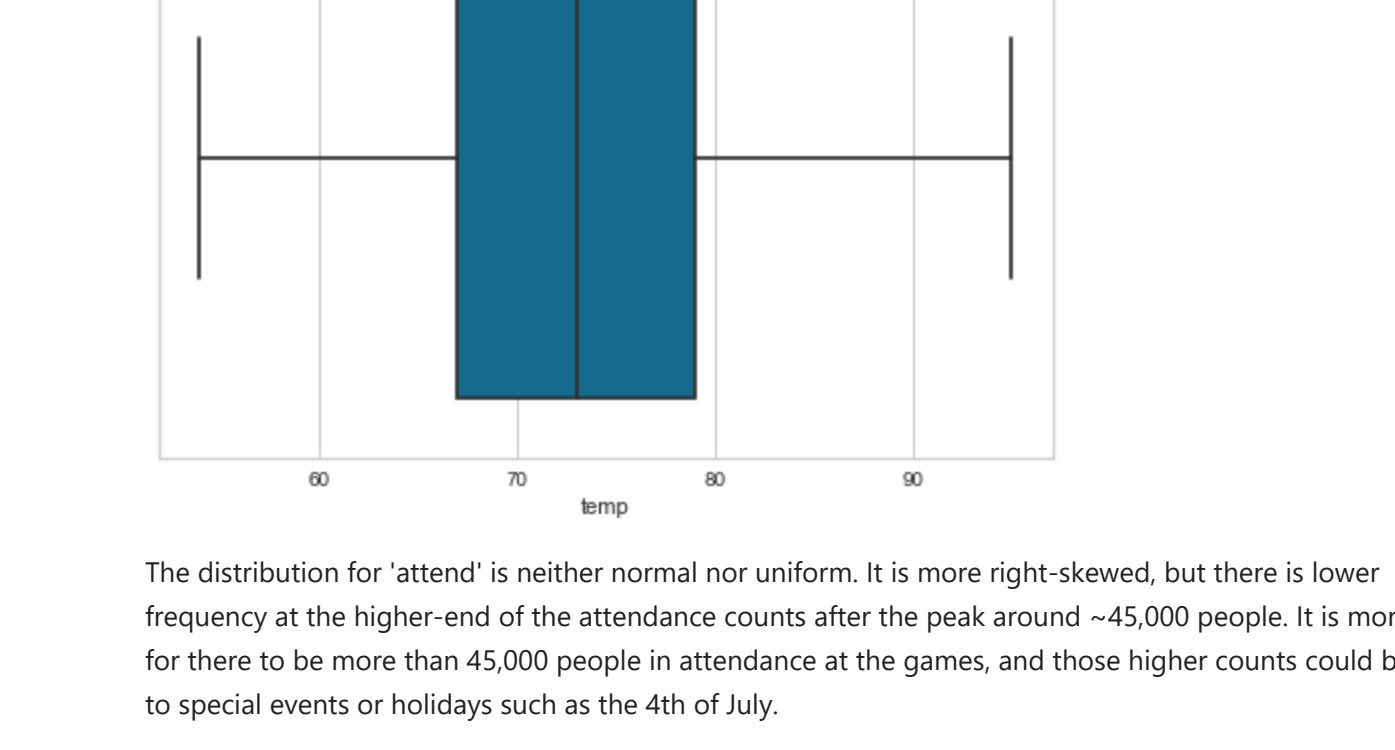
```
In [140]: #bar graph visualization for attendance counts per month
plt.bar(dodgers_pd['month'],dodgers_pd['attend'])
plt.xlabel('Month')
plt.ylabel('Attendance')
```

```
Out[140]: Text(0, 0.5, 'Attendance')
```



```
In [140]: #bar graph visualization for attendance counts for day of the week
plt.bar(dodgers_pd['day_of_week'],dodgers_pd['attend'])
plt.xlabel('Day of the Week')
plt.ylabel('Attendance')
```

```
Out[140]: Text(0, 0.5, 'Attendance')
```



Just from visualizing the other potential target variables of 'month', 'day_of_week' and 'day_night', it can be seen where most of the values in the dataset lie. The months as per the baseball season fall during summer and fall with the majority of games being in May and August. Also, with the days of the week, most of the games are held during the weekend from Friday to Sunday, with Tuesday being a weekday that holds majority popularity. Also, night games are much more popular than day games ... these classes are imbalanced.

The boxplot of 'day' is not uniform or normal; it has three peaks at the beginning of the months (~1-3), middle of the months (~day 15), and the end of the months (~29-31). It could be that each baseball season has a different schedule, so I'm not sure there is much significance to the frequency of the days during a month that are played and attended, since it depends on other teams and availability of sporting venues. As we are trying to gather results in relation to running a marketing promotion on a certain night in order to increase attendance, I am going to use 'month' and 'day_of_week' variables as the target variables for my regression since I think they will be the most influential in being able to provide valuable information around targeted marketing. Also, with marketing promotions especially in social media, it is more applicable to know a month and day of the week rather than just a day which can vary month to month.

Describing the Other Features

```
In [140]: features = dodgers_pd[['attend','opponent','temp','skies','cap','shirt','fireworks','bobblehead','promo']]
features
```

	attend	opponent	temp	skies	cap	shirt	fireworks	bobblehead	promo
0	56000	Pirates	67	Clear	NO	NO	NO	NO	NO
1	29729	Pirates	58	Cloudy	NO	NO	NO	NO	NO
2	28328	Pirates	57	Cloudy	NO	NO	NO	NO	NO
3	31601	Padres	54	Cloudy	NO	NO	YES	NO	YES
4	46549	Padres	57	Cloudy	NO	NO	NO	NO	NO
...
76	40724	Rockies	84	Cloudy	NO	NO	NO	NO	NO
77	35607	Rockies	95	Clear	NO	NO	NO	NO	NO
78	33624	Giants	86	Clear	NO	NO	NO	NO	NO
79	42473	Giants	83	Clear	NO	NO	NO	NO	NO
80	34014	Giants	82	Cloudy	NO	NO	NO	NO	NO

81 rows x 9 columns

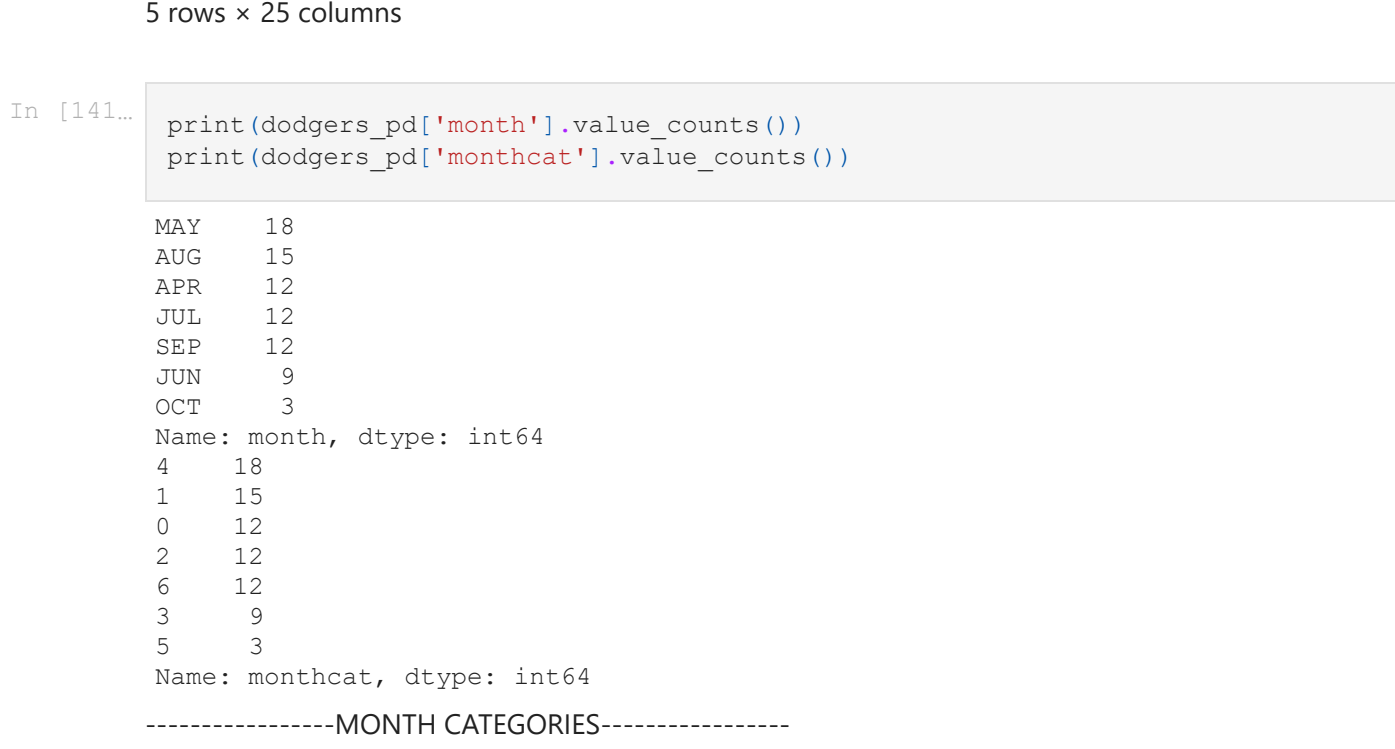
```
In [140]: desc_features = features.describe()
desc_features
```

```
Out[140]: attend      temp
count      81.000000      81.000000
mean    10400.074074    73.148148
std      8297.539460     8.317318
min       2431.000000    54.000000
25%     34493.000000    67.000000
50%     40284.000000    73.000000
75%     46588.000000    79.000000
max     56000.000000    95.000000
```

```
In [140]: #boxplot of 'attend'
sns.boxplot(features['attend'])
plt.title("Boxplot of Dodgers Games' Attendance")
```

C:\Users\phil1\Anaconda3\lib\site-packages\seaborn\decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be "data", and passing other arguments without an explicit keyword will result in an error or misinterpretation.

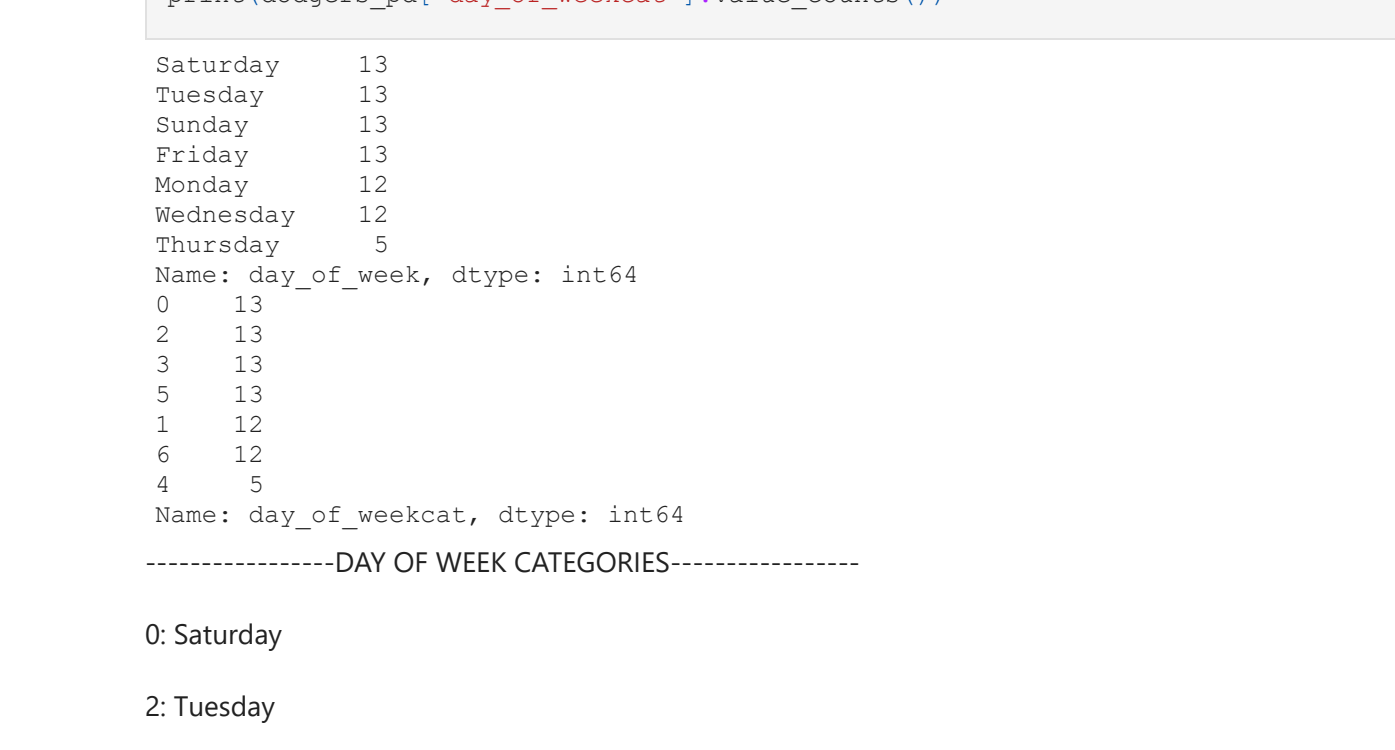
warnings.warn(
Text(0.5, 1.0, "Boxplot of Dodgers Games' Attendance")



```
In [140]: #boxplot of 'temp'
sns.boxplot(features['temp'])
plt.title("Boxplot of Temperature at Dodgers Games")
```

C:\Users\phil1\Anaconda3\lib\site-packages\seaborn\decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be "data", and passing other arguments without an explicit keyword will result in an error or misinterpretation.

warnings.warn(
Text(0.5, 1.0, "Boxplot of Temperature at Dodgers Games")



The distribution for 'attend' is neither normal nor uniform. It is more right-skewed, but there is lower frequency at the higher-end of the attendance counts after the peak around ~45,000 people. It is more rare for there to be more than 45,000 people in attendance at the games, and those higher counts could be due to special events or holidays such as the 4th of July.

The distribution of game temperatures is relatively normal! The concentration of temperatures are around 70-75 degrees Fahrenheit, which makes sense given the months that the games are held during. Summer to fall seasons range around mildly warm and hot temperatures, so this distribution makes sense with that weather.

```
In [140]: #value counts for categorical variables
print(features['opponent'].value_counts())
features['opponent'].value_counts().sort_values().plot(kind = 'barh')
```

```
Out[140]: <AxesSubplot>
Giants      9
Rockies     9
Padres      9
Snakes      9
Cardinals   7
Mets        4
Breweary    4
Reds        3
Nationals   3
White Sox   3
Marlins     3
Phillies    3
Braves      3
Cubs        3
Pirates     3
Astros      3
Name: opponent, dtype: int64
```

```
Out[140]: <AxesSubplot>
Chiefs      8
Rockies     8
Padres      8
Snakes      8
Cardinals   7
Mets        6
Angels      5
Pirates     5
Astros      5
White Sox   5
Marlins     5
Phillies    5
Braves      5
Cubs        5
Reds        5
Name: opponent, dtype: int64
```

```
In [140]: print(features['skies'].value_counts())
```

```
Out[140]: Clear      62
Cloudy    19
Name: skies, dtype: int64
```

```
In [140]: print(features['cap'].value_counts())
```

```
Out[140]: NO      79
YES      2
Name: cap, dtype: int64
```

```
In [141]: print(features['shirt'].value_counts())
```

```
Out[141]: NO      78
YES      3
Name: shirt, dtype: int64
```

```
In [141]: print(features['fireworks'].value_counts())
```

```
Out[141]: NO      67
YES      14
Name: fireworks, dtype: int64
```

```
In [141]: print(features['bobblehead'].value_counts())
```

```
Out[141]: NO      70
YES      11
Name: bobblehead, dtype: int64
```

```
In [141]: print(features['promo'].value_counts())
```

```
Out[141]: NO      51
YES      30
Name: promo, dtype: int64
```

The balance for the categorical variables with 'skies', 'cap', 'shirt', 'fireworks' and 'bobblehead' is quite imbalanced. There is a value that holds the majority, which would need to be handled when going into regression modeling. Most of these are in correspondence to promotional objects, and it seems that 'NO' is the majority value which shows that the past promotion items weren't so popularly taken at the baseball games.

Out of the 81 Dodgers games in the dataset, some sort of promotion was offered at 30 of them ... a little less than half. There is a higher amount where no promotions were offered, but it allows us to compare the days where promotions were and weren't offered and analyze the change in attendance in comparison.

Creating Label Encoders for Categorical Variables

In order to utilize categorical variables in our models, we need to encode them as numbers. The pieces of code below do exactly this!

```
In [141]: #month, day, attend, day_of_week, opponent, skies, day_night, cap, shirt, fireworks, bobblehead
cat_cols = ['month', 'day_of_week', 'opponent', 'skies', 'day_night', 'cap', 'shirt', 'fireworks', 'bobblehead', 'promo']
#turning categories into their numerical counterparts using LabelEncoder
for var in cat_cols:
    number = LabelEncoder()
    dodgers_pd[var+cat] = number.fit_transform(dodgers_pd[var].astype('str'))
```

```
Out[141]: dodgers_pd.head()
```

	month	day	attend	day_of_week	opponent	temp	skies	day_night	cap	shirt	fireworks	bobblehead	promo
0	APR	10	56000	Tuesday	Pirates	67	Clear	Day	NO	NO	NO	NO	NO
1	APR	11	29729	Wednesday	Pirates	58	Cloudy	Night	NO	NO	NO	NO	NO
2	APR	12	28328	Thursday	Pirates	57	Cloudy	Night	NO	NO	NO	NO	NO
3	APR	13	31601	Friday	Padres	54	Cloudy	Night	NO	NO	YES	NO	YES
4	APR	14	46549	Saturday	Padres	57	Cloudy	Night	NO	NO	NO	NO	NO

5 rows x 25 columns

```
In [141]: print(dodgers_pd['month'].value_counts())
print(dodgers_pd['monthcat'].value_counts())
```

```
Out[141]: MAY      18
AUG      15
APR      12
JUL      12
SEP      12
JUN       9
OCT       3
Name: month, dtype: int64
4      18
1      15
0      12
2      12
6      12
3      9
5      3
Name: monthcat, dtype: int64
```

```
4: MAY
1: AUG
0: APR
```

```
2: JUL
6: SEP
3: JUN
5: OCT
```

```
In [141]: print(dodgers_pd['day_of_week'].value_counts())
print(dodgers_pd['day_of_weekcat'].value_counts())
```

```
Out[141]: Saturday      13
Sunday      13
Friday      13
Monday      12
Wednesday      12
Thursday      5
Name: day_of_week, dtype: int64
2      13
3      13
5      13
1      12
6      12
4      5
Name: day_of_weekcat, dtype: int64
```

```
-----DAY OF WEEK CATEGORIES-----
0: Saturday
2: Tuesday
3: Sunday
5: Friday
1: Monday
6: Wednesday
4: Thursday
-----PROMOTION CATEGORIES-----
0: NO
1: YES
```

```
In [141]: print(dodgers_pd['day_promo'].value_counts())
print(dodgers_pd['day_promocat'].value_counts())
```

```
Out[141]: FridayYES      13
MondayNO      11
SaturdayNO      11
SundayNO      10
TuesdayYES      8
TuesdayNO      5
SundayYES      3
ThursdayNO      3
ThursdayYES      2
SaturdayYES      2
WednesdayYES      1
MondayYES      1
Name: day_promo, dtype: int64
0      13
1      11
3      11
5      10
11     8
10     5
9      5
6      3
7      3
4      2
8      2
2      1
12     1
Name: day_promocat, dtype: int64
```

Investigating Relationships between Variables

We want to explore the correlation between our features to ensure that we are not performing modeling on features which have a relationship and therefore an effect on each other. In regression, we purely want to see which features have an effect on our predictor variable, and by removing highly-correlated features or ones with multicollinearity, we can ensure that our features will represent the effect on the dependent variable and not on each other.

```
In [141]: #correlation plot
dodgers_pd_regress = dodgers_pd.copy()
print(dodgers_pd_regress.columns)
```

```
Out[141]: Index(['month', 'day', 'attend', 'day_of_week', 'opponent', 'temp', 'skies',
'day_night', 'cap', 'shirt', 'fireworks', 'bobblehead', 'promo',
'day_promo', 'monthcat', 'day_of_weekcat', 'opponentcat', 'skiescat',
'day_nightcat', 'capcat', 'shirtcat', 'fireworkscat', 'bobbleheadcat',
'day_promocat'],
dtype='object')
```

```
In [142]: #creating new dataframe with only needed columns for regression
dodgers_pd_regress = dodgers_pd_regress[['day', 'monthcat', 'day_of_weekcat', 'opponentcat', 'skiescat', 'day_nightcat', 'capcat', 'shirtcat', 'fireworkscat', 'bobbleheadcat', 'day_promocat']]
```

```
In [142]: #subset dataframe to only have night games --> where day_nightcat = 1
dodgers_pd_regress = dodgers_pd_regress[dodgers_pd_regress['day_nightcat'] == 1]
```

```
In [142]: #drop day_nightcat column after only using 'Night'
dodgers_pd_regress = dodgers_pd_regress.drop(['day_nightcat'],axis=1)
```

```
In [142]: print(dodgers_pd_regress.shape)
print("66 rows and 13 columns for Dodgers Night Games")
```

```
Out[142]: (66, 13)
66 rows and 13 columns for Dodgers Night Games
```

```
In [142]: corr = dodgers_pd_regress.corr()
corr.style.background_gradient(cmap='coolwarm')
```

```
Out[142]:
```

	day	monthcat	day_of_weekcat	opponentcat	skiescat	capcat	shirtcat	fireworkscat
day	1.000000	-0.181142	-0.125346	-0.130097	-0.039875	-0.168428	-0.156662	0.122948
monthcat	-0.181142	1.000000	-0.017610	-0.093765	-0.171474	-0.052585	-0.154053	-0.047171
day_of_weekcat	-0.125346	-0.017610	1.000000	-0.097733	0.130012	0.127088	0.019582	-0.552171
opponentcat	-0.130097	-0.093765	-0.097733	1.000000	-0.114155	0.119136	-0.125112	0.096481
skiescat	-0.039875	-0.171474	0.130012	-0.114155	1.000000	0.202548	-0.108253	-0.068048
capcat	-0.168428	-0.052585	0.127088	-0.125112	0.202548	1.000000	-0.021926	-0.064381
shirtcat	-0.156662	-0.154053	0.019582	-0.125112	-0.108253	-0.021926	1.000000	-0.091771
fireworkscat	0.122948	-0.047171	-0.552171	0.096481	-0.068048	-0.064358	-0.091725	1.000000
bobbleheadcat	0.172594	-0.168530	0.272458	0.090432	-0.091287	-0.055470	-0.079057	-0.232020
attend	0.018145	-0.082841	0.154318	0.000970	-0.154587	-0.102093	0.098326	0.012771
temp	-0.178932	0.473551	0.079533	0.				

[illegible]

a difference of 16,432.94 people. Therefore, I would recommend Thursdays as the best days to offer promotions at Dodgers games in order to increase attendance!