Introduction:

Try to find out how MLB players' weight affect the time between their debut and Tommy John surgery date

Methods:

Gather data from Lahman's baseball database's 'People' table and list of players who underwent Tommy John surgery from Wikipedia

Using SQL Server to prepare the data

Using Python(Spyder) to perform poisson regression, for the data linear regression is actually better. But chose poisson for self-training and diversity

In [1]:

```
import pandas as pd
import pyodbc

#Gain data from SQL server, tables was imported into SQL Server from two excel file
#excel file can be found in the same repository
sql_conn = pyodbc.connect('''DRIVER={ODBC Driver 13 for SQL Server};
                            SERVER=ALLENHO\MSSQLSERVER002;
                            DATABASE=TommyJohn;
                            Trusted_Connection=yes''')
query = '''
select distinct t.Player, t.Position, t.Throws, t.date_of_surgery, p.weight, datediff(day, p.debut, t.date_of_surgery) as daydiff
from TJ$ t
join People$ p
on t.Player=concat(nameFirst,' ',nameLast)
where p.weight is not null and datediff(day, p.debut, t.date_of_surgery)>0 and datediff(day, p.debut, t.date_of_surgery)<7000
order by t.Player;
;
```

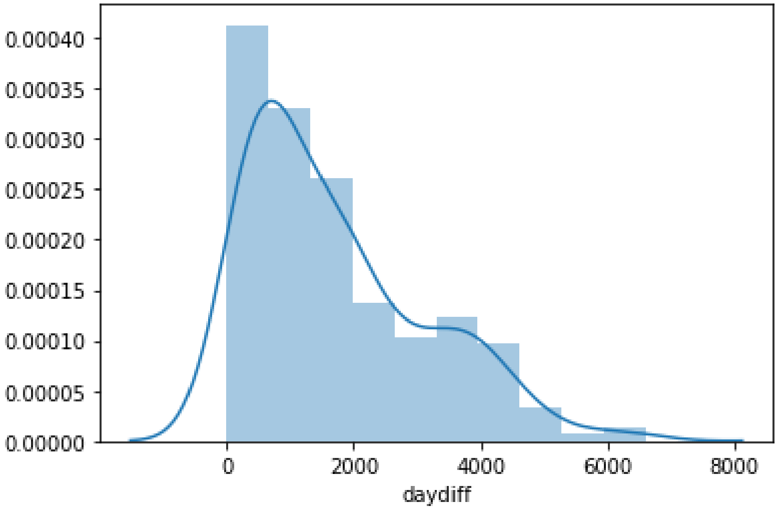
```
'''
df = pd.read_sql(query, sql_conn)

print(df.head())
```

```
            Player        Position Throws date_of_surgery  weight  dayd
iff
0    A. J. Burnett         Pitcher  Right      2003-04-01   230.0     1
323
1    A. J. Griffin         Pitcher  Right      2014-04-30   230.0
675
2    Aaron Barrett         Pitcher  Right      2015-09-03   230.0
521
3      Aaron Hicks  Center Fielder  Right      2019-10-01   202.0     2
374
4  Adam Wainwright         Pitcher  Right      2011-02-28   235.0     1
996
```

In [2]:

```
# Import libraries
import seaborn as sns
import matplotlib.pyplot as plt
# Plot sat variable
sns.distplot(df['daydiff'])
# Display the plot
plt.show()
```



In [3]:

```
# Import libraries
import statsmodels.api as sm
from statsmodels.formula.api import glm
import numpy as np
# Fit Poisson regression of sat by width
model = glm('daydiff ~ weight', data = df, family = sm.families.Poisson
()).fit()
# Display model results
print(model.summary())
```

```
                 Generalized Linear Model Regression Results

=======================================================================
=======
Dep. Variable:                daydiff   No. Observations:
    221
Model:                            GLM   Df Residuals:
    219
Model Family:                 Poisson   Df Model:
      1
Link Function:                    log   Scale:
1.0000
Method:                          IRLS   Log-Likelihood:            -1.2
957e+05
```

```
057e+05
Date:                    Tue, 04 Aug 2020   Deviance:                     2.3
918e+05
Time:                            21:05:29   Pearson chi2:
2.40e+05
No. Iterations:                         5

Covariance Type:                nonrobust

======================================================================
=======
                 coef    std err          z      P>|z|      [0.025
0.975]
----------------------------------------------------------------------
-------
Intercept      8.6950      0.016    528.132      0.000       8.663
  8.727
weight        -0.0059    7.8e-05    -75.072      0.000      -0.006
-0.006
======================================================================
=======
```

In [4]:

```python
# Compute average weight
mean_weight = np.mean(df['weight'])
# Print the compute mean
print('Average width: ', round(mean_weight, 3))
# Extract coefficients
intercept, slope = model.params
# Compute the estimated mean of y (lambda) at the average width
est_lambda = np.exp(intercept) * np.exp(slope * mean_weight)
# Print estimated mean of y
print('Estimated mean of y at average weight: ', round(est_lambda, 3))
```

```
Average width:  212.665
Estimated mean of y at average weight:  1720.251
```

In [5]:

```python
# Compute and print he multiplicative effect
print(np.exp(slope))
# Compute confidence intervals for the coefficients
```

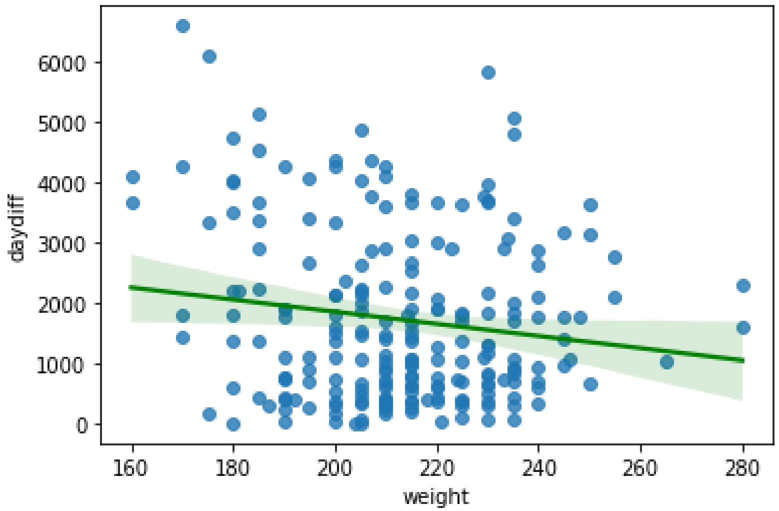
```
model_ci = model.conf_int()
# Compute and print the confidence intervals for the multiplicative eff
ect on the mean
print(np.exp(model_ci))
```

```
0.9941637034397474
                     0            1
Intercept  5783.532173  6169.089943
weight        0.994012     0.994316
```

With one unit increase of weight, the mean response of the time between their debut and Tommy John surgery date will multiply by 0.9941637034397474, which imply 0.6% decrease.

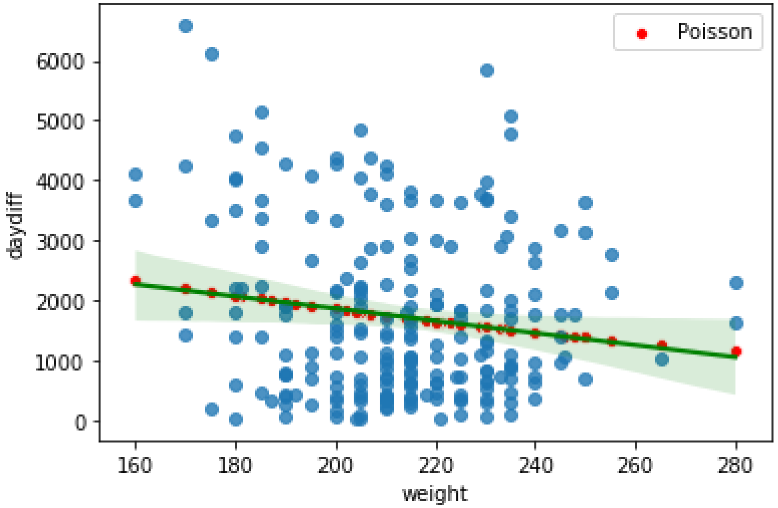
In [6]:

```
# Plot the data points and linear model fit
sns.regplot('weight', 'daydiff', data = df,
            y_jitter = 0.3,
            fit_reg = True,
            line_kws = {'color':'green',
                        'label':'LM fit'})
# Print plot
plt.show()
```



```
In [7]: # Add fitted values to the fit_values column of dataframe
        df['fit_values'] = model.fittedvalues

        # Poisson regression fitted values
        sns.scatterplot('weight','fit_values', data = df,
                   color = 'red', label = 'Poisson')
        # Plot the data points and linear model fit
        sns.regplot('weight', 'daydiff', data = df,
                    y_jitter = 0.3,
                    fit_reg = True,
                    line_kws = {'color':'green',
                                'label':'LM fit'})
        # Print plot
        plt.show()
```



Conclusion: it really didn't matter much for the weight of players on the time between their debut and Tommy John surgery date, but we can slightly imply they are negatively correlated.