Term Paper Presentation: Major League Baseball

Katie Adams

DSC 530

Summer 2021

Statistical Question

What Major League Baseball (MLB) team stats impact team wins the most?

In [184]:
```python
# Chapter 1:
## A minimum of 5 variables in your dataset used during your analysis
baseball_df=pd.read_csv("C:/Users/kadams/OneDrive - Suncor Energy Inc/DSC Masters Program/GitHub/DSC520/Assignments/dsc520
baseball_df.head()
## The variables that will be used in this analysis are: W, BB, OPS, 2B, HR, OBP, and SLG
```

Out[184]:

	Tm	League	National	American	Active	#Bat	BatAge	R/G	G	W	...	SLG	OPS	OPSPlus	TB	GDP	HBP	SH	SF	IBB
0	ARI	0	1.0	NaN	1	46.8	28.40	23.43	708	354	...	0.4198	0.7388	454	10301	490	262	152	173	
1	ATL	0	1.0	NaN	1	53.0	28.22	24.32	707	362	...	0.4296	0.7612	489	10267	524				
2	BAL	1	NaN	1.0	1	51.2	27.66	22.09	708	290	...	0.4226	0.7342	485	10250	532				
3	BOS	1	NaN	1.0	1	47.4	27.56	26.11	708	402	...	0.4464	0.7838	530	[illegible]					
				NaN	1	48.2	27.46	24.17	709	408	...	0.4230	0.7558	[illegible]						

Dataset Variables

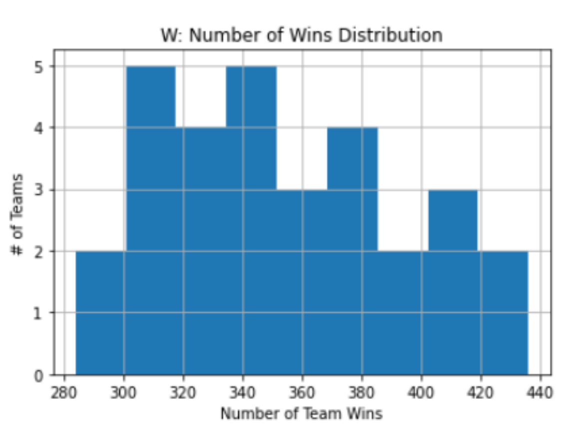
1. W: Number of Wins
2. BB: Bases on Balls/Walks
3. OPS: On Base+Slugging Percentage
4. Double: Doubles
5. HR: Home Runs
6. OBP: On Base Percentage
7. SLG: Slugging Percentage

Description of Dataset Variables

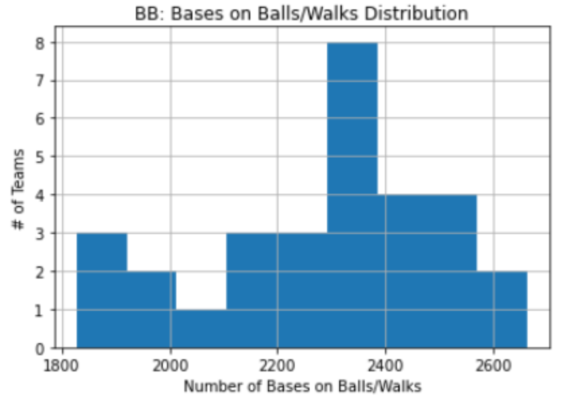
1. **W (Wins):** Number of Team Wins
2. **BB (Bases on Balls/Walks):** A walk (or base on balls) occurs when a pitcher throws four pitches out of the strike zone, none of which are swung at by the hitter. After refraining from swinging at four pitches out of the zone, the batter is awarded first base
3. **OPS (On Base+Slugging Percentage):** Adds on-base percentage and slugging percentage to get one number that unites the two. It's meant to combine how well a hitter can reach base, with how well he can hit for average and for power
4. **Doubles (Double):** When batter hits the ball into play and reaches second base without the help of an intervening error or attempt to put out another baserunner
5. **HR (Home Run):** When a batter hits a fair ball and scores on the play without being put out or without the benefit of an error
6. **OBP (On Base Percentage):** How frequently a batter reaches base per plate appearance (hits, walks and hit-by-pitches). Does not include errors, times reached on a fielder's choice or a dropped third strike
7. **SLG (Slugging Percentage):** Total number of bases a player records per at-bat (does not include walks and hit-by-pitches in its equation)

Histograms of Variables

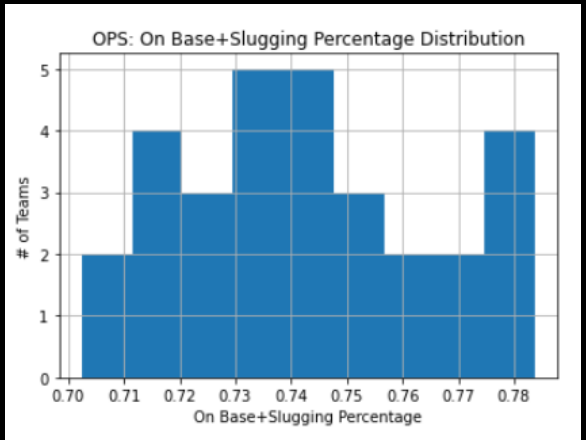
- In this particular dataset, all the values sit very close to each other, so the histograms do not show outliers clearly (if there are any)
- To double check for outliers, a boxplot was created to confirm the outliers that the histogram could not show easily



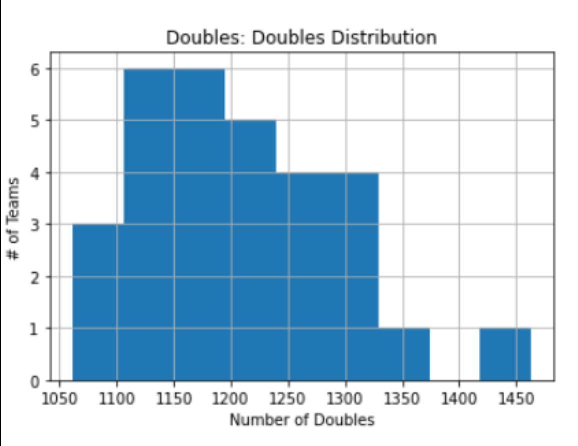

```python
## W: Number of Team Wins
baseball_df.W.hist(grid=True, bins=9)
plt.xlabel('Number of Team Wins')
plt.ylabel('# of Teams')
plt.title('W: Number of Wins Distribution')
```




```python
## BB: Bases on Balls/Walks
baseball_df.BB.hist(grid=True, bins=9)
plt.xlabel('Number of Bases on Balls/Walks')
plt.ylabel('# of Teams')
plt.title('BB: Bases on Balls/Walks Distribution')
```




```python
## OPS: On Base+Slugging Percentage
baseball_df.OPS.hist(grid=True, bins=9)
plt.xlabel('On Base+Slugging Percentage')
plt.ylabel('# of Teams')
plt.title('OPS: On Base+Slugging Percentage Distribution')
```



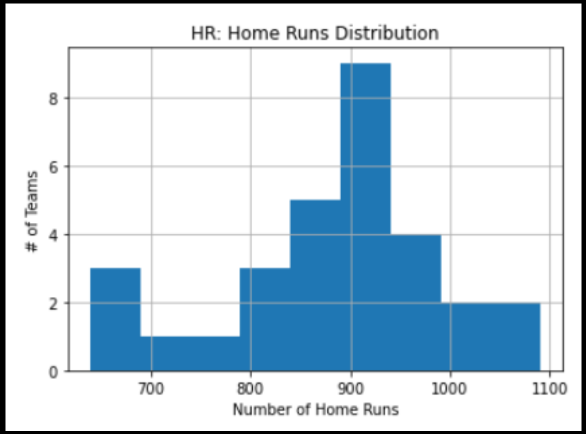

```python
## Doubles: Doubles
baseball_df.Doubles.hist(grid=True, bins=9)
plt.xlabel('Number of Doubles')
plt.ylabel('# of Teams')
plt.title('Doubles: Doubles Distribution')
```

Histograms of Variables

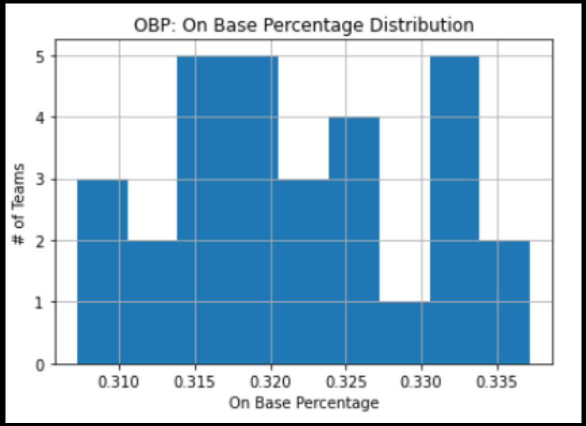
- In this particular dataset, all the values sit very close to each other, so the histograms do not show outliers clearly (if there are any)
- To double check for outliers, a boxplot was created to confirm the outliers that the histogram could not show easily

Histograms of Variables

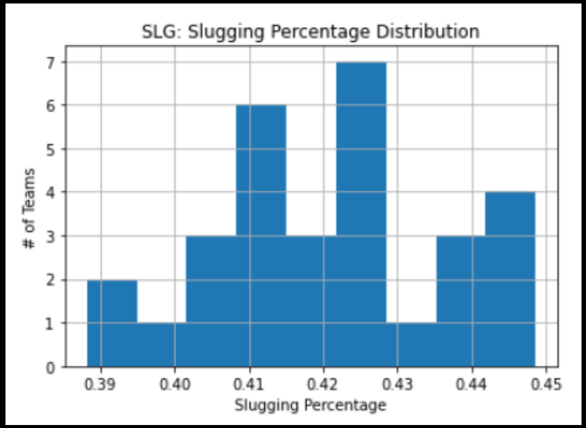
- In this particular dataset, all the values sit very close to each other, so the histograms do not show outliers clearly (if there are any)
- To double check for outliers, a boxplot was created to confirm the outliers that the histogram could not show easily




```
## HR: Home Runs
baseball_df.HR.hist(grid=True, bins=9)
plt.xlabel('Number of Home Runs')
plt.ylabel('# of Teams')
plt.title('HR: Home Runs Distribution')
```



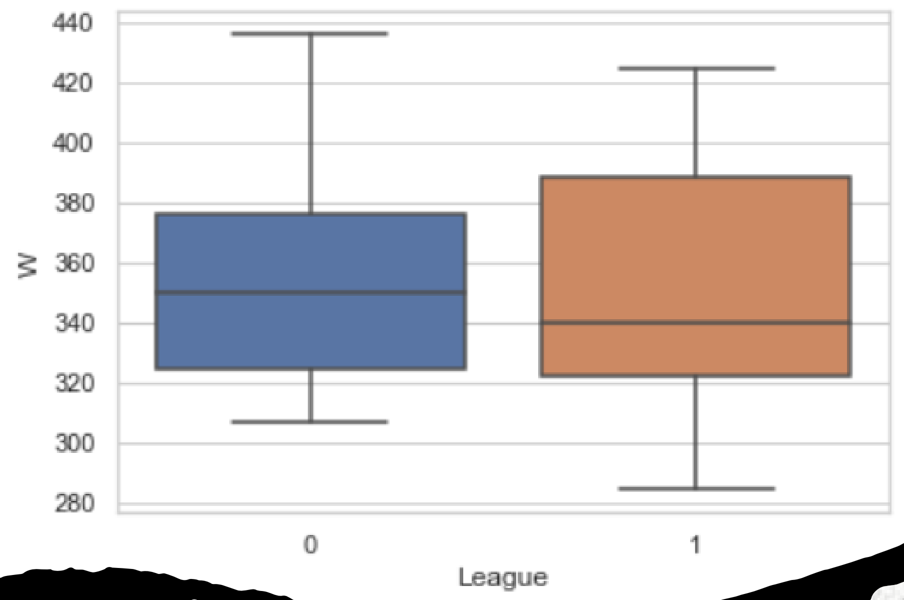

```
## OBP: On Base Percentage
baseball_df.OBP.hist(grid=True, bins=9)
plt.xlabel('On Base Percentage')
plt.ylabel('# of Teams')
plt.title('OBP: On Base Percentage Distribution')
```




```
## SLG: Slugging Percentage
baseball_df.SLG.hist(grid=True, bins=9)
plt.xlabel('Slugging Percentage')
plt.ylabel('# of Teams')
plt.title('SLG: Slugging Percentage Distribution')
```

Boxplot of MLB Leagues & Number of Wins

```
[190]: # Chapter 2:
       ## Identify any outliers and explain the reasoning for them being

       ## Boxplot: In these 5 years (2016-2020), there are no outliers i
       ## truely good over 5 years. You'd really only see outliers withi
       sns.set_theme(style="whitegrid")
       ax = sns.boxplot(x="League", y="W", data=baseball_df) #The Nation
```



- In this 5-year dataset (2016-2020), there are no outliers in this dataset because no team is going to be truly bad or truly good over a 5-year span
- Outliers would only be seen in single 1 season

Descriptive variable characteristics: Mean

These are the Mean values over 5 years:

```
Mean of Variables:
Wins is 353.8333333333333
Bases on Balls/Walks is 2286.3333333333335
On Base+Slugging Percentage is 0.7423533333333333
Doubles is 1208.9666666666667
Home Runs is 879.3333333333334
On Base Percentage is 0.3215533333333332
Slugging Percentage is 0.42079999999999995
```

```
print("These are the descriptive values of number of Wins over 5 years:")
print("")
## Mean - pg. 23
print("Mean of Variables:")
mean_W = baseball_df.W.mean()
print("Wins is", mean_W)
mean_BB= baseball_df.BB.mean()
print("Bases on Balls/Walks is", mean_BB)
mean_OPS= baseball_df.OPS.mean()
print("On Base+Slugging Percentage is", mean_OPS)
mean_Doubles= baseball_df.Doubles.mean()
print("Doubles is", mean_Doubles)
mean_HR= baseball_df.HR.mean()
print("Home Runs is", mean_HR)
mean_OBP= baseball_df.OBP.mean()
print("On Base Percentage is", mean_OBP)
mean_SLG= baseball_df.SLG.mean()
print("Slugging Percentage is", mean_SLG)
```

Descriptive variable characteristics: Mode

These are the Mode values over 5 years:

```
Mode of Variables:
Wins is 374
Bases on Balls/Walks is 2322
On Base+Slugging Percentage is 0.7808
Doubles is 1130
Home Runs is 814
On Base Percentage is 0.3146
Slugging Percentage is 0.4134
```

```python
print("")
print("Mode of Variables:")
mode_W = statistics.mode(baseball_df.W)
print("Wins is", mode_W)
mode_BB = statistics.mode(baseball_df.BB)
print("Bases on Balls/Walks is", mode_BB)
mode_OPS = statistics.mode(baseball_df.OPS)
print("On Base+Slugging Percentage is", mode_OPS)
mode_Doubles = statistics.mode(baseball_df.Doubles)
print("Doubles is", mode_Doubles)
mode_HR = statistics.mode(baseball_df.HR)
print("Home Runs is", mode_HR)
mode_OBP = statistics.mode(baseball_df.OBP)
print("On Base Percentage is", mode_OBP)
mode_SLG = statistics.mode(baseball_df.SLG)
print("Slugging Percentage is", mode_SLG)
```

Descriptive variable characteristics: Spread

These are the Spread values over 5 years:

```
Spread (Variance) of Variables:
Wins is 1712.4195402298851
Bases on Balls/Walks is 50080.16091954023
On Base+Slugging Percentage is 0.0005242418850574717
Doubles is 7740.791954022989
Home Runs is 11723.747126436778
On Base Percentage is 6.801291954022989e-05
Slugging Percentage is 0.0002489103448275863
```

```python
## Spread (variance) - pg. 23
print("")
print("Spread (Variance) of Variables:")
var_W = baseball_df.W.var()
print("Wins is", var_W)
var_BB= baseball_df.BB.var()
print("Bases on Balls/Walks is", var_BB)
var_OPS= baseball_df.OPS.var()
print("On Base+Slugging Percentage is", var_OPS)
var_Doubles= baseball_df.Doubles.var()
print("Doubles is", var_Doubles)
var_HR= baseball_df.HR.var()
print("Home Runs is", var_HR)
var_OBP= baseball_df.OBP.var()
print("On Base Percentage is", var_OBP)
var_SLG= baseball_df.SLG.var()
print("Slugging Percentage is", var_SLG)
```

Descriptive variable characteristics: Standard Deviation

These are the Standard Deviation values over 5 years:

```
Standard Deviation of Variables:
The standard deviation is 41.381391231203004
Bases on Balls/Walks is 223.7859712304152
On Base+Slugging Percentage is 0.022896329073837835
Doubles is 87.98177057790431
Home Runs is 108.27625375139638
On Base Percentage is 0.008246994576221685
Slugging Percentage is 0.015776892749448046
```

```python
## Standard Deviation
print("")
print("Standard Deviation of Variables:")
std_W = baseball_df.W.std()
print("The standard deviation is", std_W)
std_BB= baseball_df.BB.std()
print("Bases on Balls/Walks is", std_BB)
std_OPS= baseball_df.OPS.std()
print("On Base+Slugging Percentage is", std_OPS)
std_Doubles= baseball_df.Doubles.std()
print("Doubles is", std_Doubles)
std_HR= baseball_df.HR.std()
print("Home Runs is", std_HR)
std_OBP= baseball_df.OBP.std()
print("On Base Percentage is", std_OBP)
std_SLG= baseball_df.SLG.std()
print("Slugging Percentage is", std_SLG)
```

Descriptive variable characteristics: Tails

These are the Tail values over 5 years:

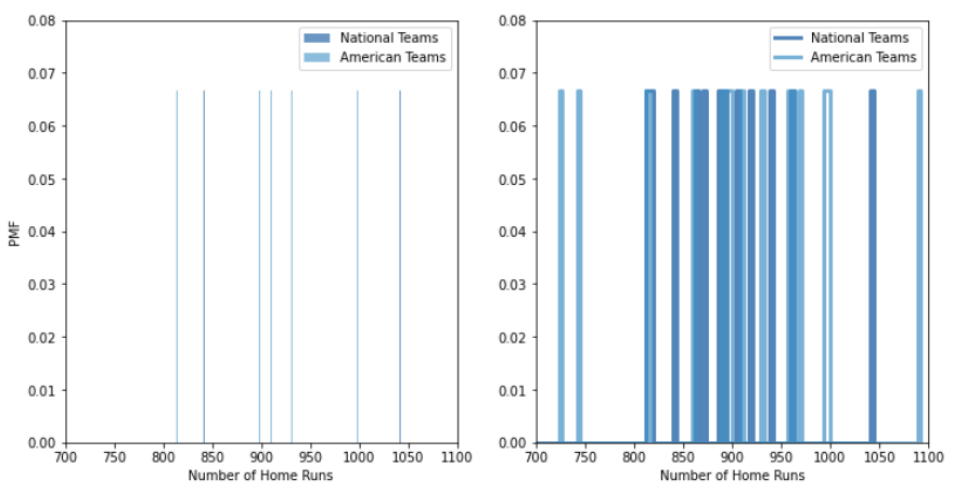
```
Minimum Tails of Variables:
Wins is 284
Bases on Balls/Walks is 1827
On Base+Slugging Percentage is 0.7024
Doubles is 1061
Home Runs is 639
On Base Percentage is 0.3072
Slugging Percentage is 0.3882

Maximum Tails of Variables:
Wins is 436
Bases on Balls/Walks is 2664
On Base+Slugging Percentage is 0.7838
Doubles is 1463
Home Runs is 1091
On Base Percentage is 0.3372
Slugging Percentage is 0.4486
```

```python
## Tails - pg. 25 (definition: The part of a distribution at the high or low extremes)
print("")
print("Minimum Tails of Variables:")
minvalue_W = min(baseball_df.W)
print("Wins is", minvalue_W)
minvalue_BB= min(baseball_df.BB)
print("Bases on Balls/Walks is", minvalue_BB)
minvalue_OPS= min(baseball_df.OPS)
print("On Base+Slugging Percentage is", minvalue_OPS)
minvalue_Doubles= min(baseball_df.Doubles)
print("Doubles is", minvalue_Doubles)
minvalue_HR= min(baseball_df.HR)
print("Home Runs is", minvalue_HR)
minvalue_OBP= min(baseball_df.OBP)
print("On Base Percentage is", minvalue_OBP)
minvalue_SLG= min(baseball_df.SLG)
print("Slugging Percentage is", minvalue_SLG)

print("")
print("Maximum Tails of Variables:")
maxvalue_W = max(baseball_df.W)
print("Wins is", maxvalue_W)
maxvalue_BB= max(baseball_df.BB)
print("Bases on Balls/Walks is", maxvalue_BB)
maxvalue_OPS= max(baseball_df.OPS)
print("On Base+Slugging Percentage is", maxvalue_OPS)
maxvalue_Doubles= max(baseball_df.Doubles)
print("Doubles is", maxvalue_Doubles)
maxvalue_HR= max(baseball_df.HR)
print("Home Runs is", maxvalue_HR)
maxvalue_OBP= max(baseball_df.OBP)
print("On Base Percentage is", maxvalue_OBP)
maxvalue_SLG= max(baseball_df.SLG)
print("Slugging Percentage is", maxvalue_SLG)
```

PMF: Comparing Two Scenarios

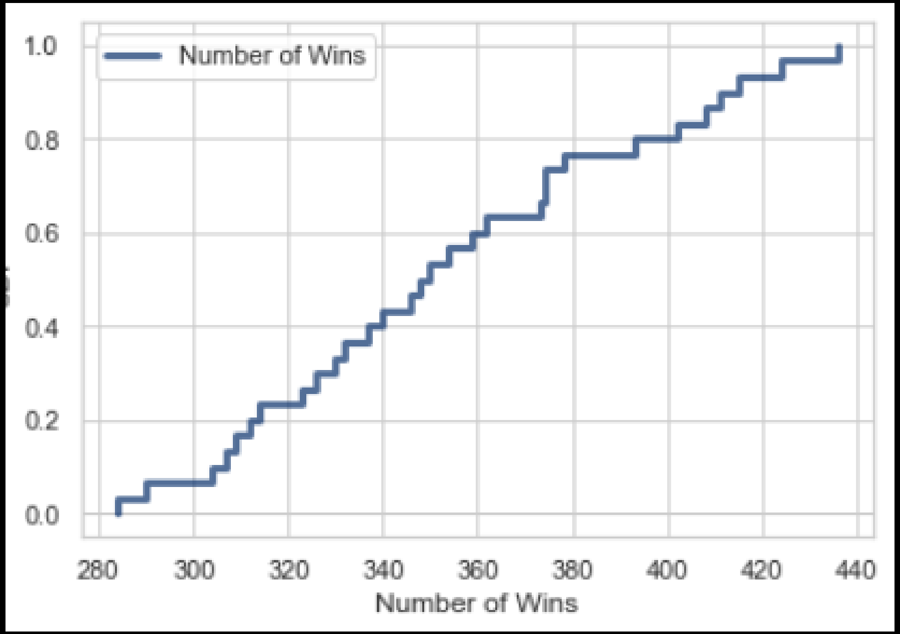
- In this analysis, two distributions of the sample are compared: The number of home runs in the MLB American and National Leagues
- Results: The PMF shows that over 5 years, there is very little difference in the number of home runs between the American and National Leagues

```python
# Filter the dataset by Teams on the National and American Leagues
NAT_Teams = baseball_df[baseball_df.League == 0] # 0 represents MLB National League
AMER_Teams = baseball_df[baseball_df.League == 1] # 1 represents MLB American League

# Create PMF of National and American Leagues
NAT_pmf = thinkstats2.Pmf(NAT_Teams.HR, label="National Teams")
AMER_pmf = thinkstats2.Pmf(AMER_Teams.HR, label="American Teams")

# plot Pmf bar graph
width=0.45
axis = [700, 1100, 0, 0.08]
thinkplot.PrePlot(2, cols=2)
thinkplot.Hist(NAT_pmf, align='right', width=width)
thinkplot.Hist(AMER_pmf, align='left', width=width)
thinkplot.Config(xlabel='Number of Home Runs', ylabel='PMF', axis=axis)

# plot Pmf step function
thinkplot.PrePlot(2)
thinkplot.SubPlot(2)
thinkplot.Pmfs([NAT_pmf, AMER_pmf])
thinkplot.Config(xlabel='Number of Home Runs', axis=axis)
```

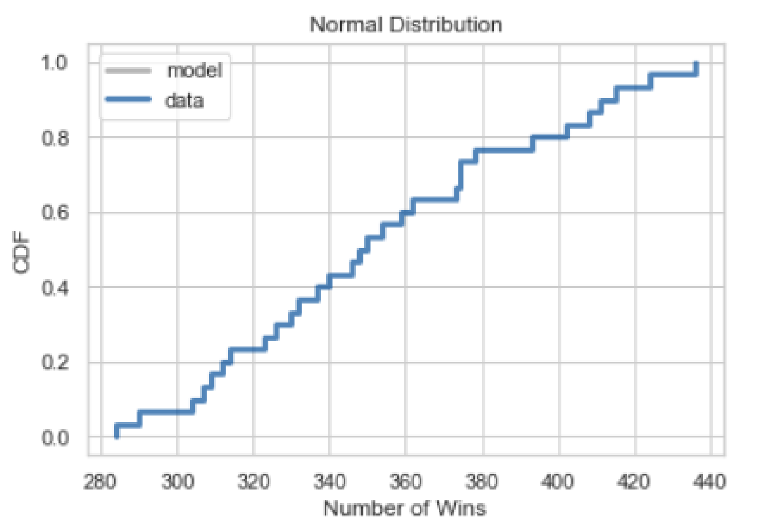


CDF With 1 Variable

- The CDF shows that the variable's (Team Wins, W) mode (374) is clearly visible, with about 10% of the Teams have less than 300 wins, about 10% have more than 420 wins, and about 90% have less than 420 wins over 5 years.
- The question of this term paper is "What Major League Baseball (MLB) team stats impact team wins the most?"
  - Results: So, in this case, since we are only looking at the CDF of 1 of the variables, it is not possible to answer the question at this point since it depends on comparing Wins to another variable in the dataset.



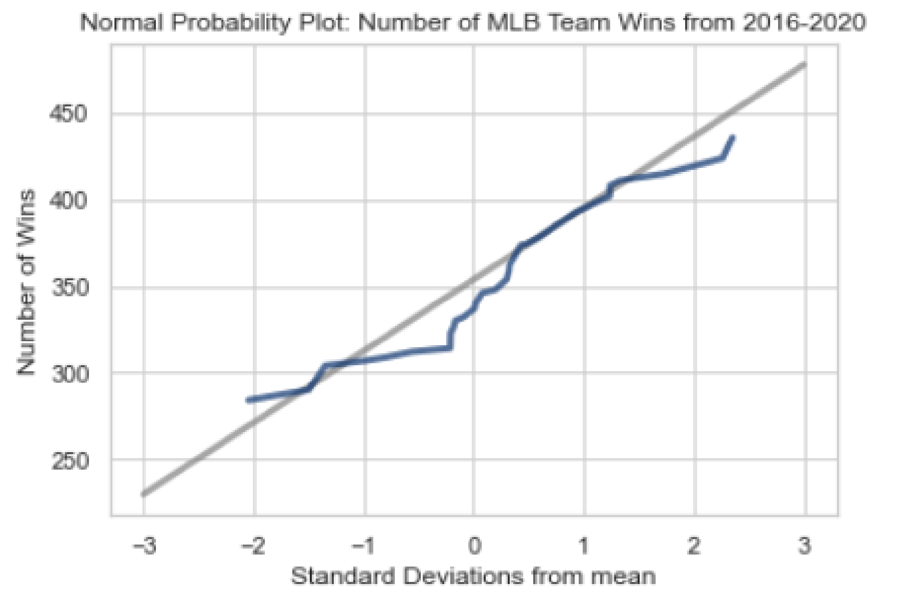

```python
## Cdf for the distribution of number of Wins in the baseball dataframe
cdf = thinkstats2.Cdf(baseball_df.W, label = "Number of Wins")

thinkplot.Cdf(cdf)
thinkplot.Show(xlabel="Number of Wins", ylabel="CDF")
```




```python
## Normal Distribution
## I'll use a normal model to fit the distribution of number of Wins from 2016-2020
wins = baseball_df.W

# estimate parameters: trimming outliers yields a better fit
mu, var = thinkstats2.TrimmedMeanVar(wins, p=0.01)
print('Mean is, Var is', mu, var)

# plot the model
sigma = np.sqrt(var)
print('Sigma is', sigma)
xs, ps = thinkstats2.RenderNormalCdf(mu, sigma, low=0, high=12.5)

thinkplot.Plot(xs, ps, label='model', color='0.6')

# plot the data
cdf = thinkstats2.Cdf(wins, label='data')

thinkplot.PrePlot(1)
thinkplot.Cdf(cdf)
thinkplot.Config(title='Normal Distribution',
                 xlabel='Number of Wins',
                 ylabel='CDF')
```

Analytical Distribution Plot

Normal Distribution




```
## Normal Probability Plot
wins = baseball_df.W

def MakeNormalPlot(wins):
    mean = wins.mean()
    std = wins.std()

    xs = [-3, 3]
    fxs, fys = thinkstats2.FitLine(xs, inter=mean, slope=std)
    thinkplot.Plot(fxs, fys, color="gray", label="model")

    xs, ys = thinkstats2.NormalProbability(wins)
    thinkplot.Plot(xs, ys, label="Number of Wins")
    thinkplot.Config(title='Normal Probability Plot: Number of MLB Team Wins from 2016-2020',
                 xlabel='Standard Deviations from mean',
                 ylabel='Number of Wins')

MakeNormalPlot(wins)
```

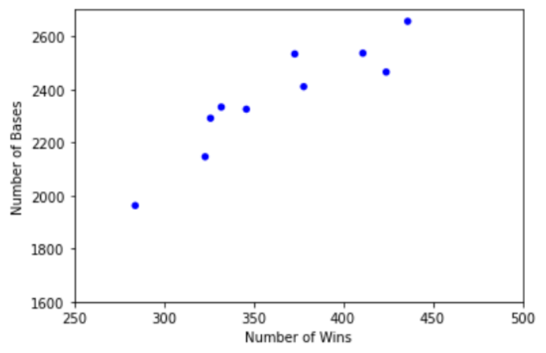
Analytical Distribution Plot

Normal Probability Plot

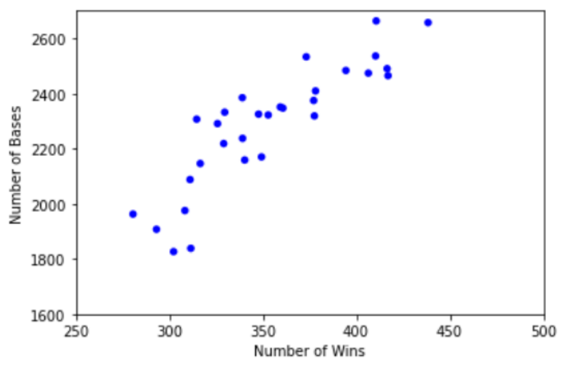
Analytical Distribution Plots

- `Results:` These two analytical distribution plots (Normal CDF, Normal Distribution, Normal Probability Plot) tell us that the dataset is normally distributed
- This normal distribution is to be expected as this was a large dataset (Central Limit Theorem)

```python
def SampleRows(df, nrows, replace=False):
    indices = np.random.choice(df.index, nrows, replace=replace)
    sample = df.loc[indices]
    return sample

sample = SampleRows(df, 10)
Wins, Bases = sample.W, sample.BB

thinkplot.Scatter(Wins, Bases, alpha=1)
thinkplot.Config(xlabel='Number of Wins',
                 ylabel='Number of Bases',
                 axis=[250, 500, 1600, 2700],
                 legend=False)
```



In [25]:
```python
def Jitter(values, jitter=0.5):
    n = len(values)
    return np.random.normal(0, jitter, n) + values

Wins = Jitter(Wins, 1.4)
Bases = Jitter(Bases, 0.5)

thinkplot.Scatter(Wins, Bases, alpha=1)
thinkplot.Config(xlabel='Number of Wins',
                 ylabel='Number of Bases',
                 axis=[250, 500, 1600, 2700],
                 legend=False)
```

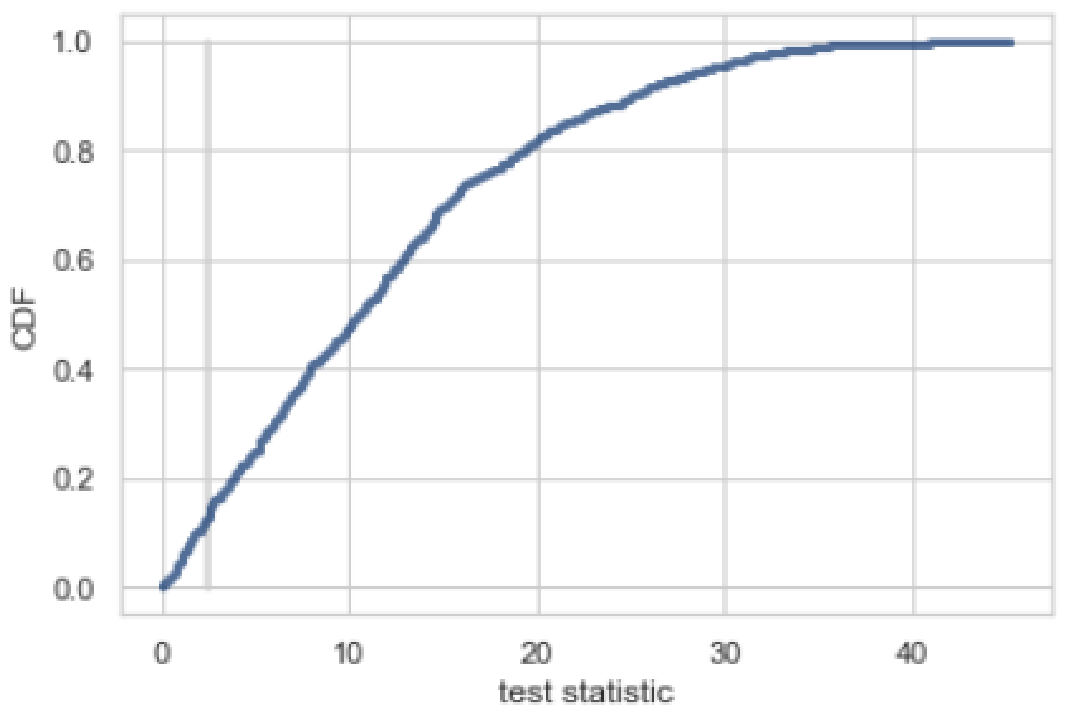


Scatterplots

- The covariance between Wins and BB: Balls/Walks is 7741.855555555556
- The Pearson's correlation between Wins and BB: Balls/Walks is 0.8648292618009141
- The Spearman's correlation between Wins and BB: Balls/Walks is 0.8861942428849995
- Results: Looking at the scatterplot, the relationship between Wins and Bases on Balls/Walks is linear
- Results: With a Pearson's correlation of 0.86, which means there's a positive correlation between Team Wins and Bases on Balls/Walks
- Results: In terms of causation, BB (Bases on Balls/Walks) is a way to get runs, and runs leads to more Team Wins

Hypothesis Test

The p-value is 0.877




```python
## To find the difference in wins between the National and American MLB Leagues:
NAT_Teams = baseball_df[baseball_df.League == 0] # 0 represents MLB National League
AMER_Teams = baseball_df[baseball_df.League == 1] # 1 represents MLB American League

## Extract the number of wins as NumPy arrays, and pass them into "data" variable to DiffMeansPermute below to compute p-value
data = NAT_Teams.W.values, AMER_Teams.W.values

ht = DiffMeansPermute(data)
pvalue = ht.PValue()
pvalue
print("The p-value is", pvalue)
## A p-value of 0.877 or 87% "means that we expect to see a difference as big as the observed effect about 87%
## of the time. So this effect is not statistically significant" (Downey, ___)
## 87% is bigger than 5%, which is the threshold of statisical significance. So, this effect is not statistically significant

## "HypothesisTest provides PlotCdf, which plots the distribution of the test statistic and the gray line indicating the
## observed effect size" (Downey, ____):

ht.PlotCdf()
thinkplot.Show(xlabel='test statistic',
               ylabel='CDF')
```

- Results: In looking at the difference in wins between the American and National Leagues, a p-value of 0.877 or 87% was found which "means that we expect to see a difference as big as the observed effect about 87%of the time. So, this effect is not statistically significant" (Downey, 2021)
- 87% is bigger than 5%, which is the threshold of statistical significance. So, this effect is not statistically significant

Regression Analysis

- Results: When we control for BB and Doubles, R-Squared increases from 0.75 (single regression) to 0.82 (multiple regression)
- These results suggest that the apparent difference in Team Wins being may be explained by the number of Bases on Walks/Balls and Doubles (strong correlation)

```
## Run a single regression with a variable, BB
formula = 'W ~ BB'
results_single = smf.ols(formula, data=baseball_df).fit()
results_single.summary()
```

OLS Regression Results

Dep. Variable:	W	R-squared:	0.748
Model:	OLS	Adj. R-squared:	0.739
Method:	Least Squares	F-statistic:	83.08
Date:	Tue, 10 Aug 2021	Prob (F-statistic):	7.15e-10
Time:	17:40:16	Log-Likelihood:	-133.07
No. Observations:	30	AIC:	270.1
Df Residuals:	28	BIC:	273.0
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-11.7969	40.299	-0.293	0.772	-94.346	70.752
BB	0.1599	0.018	9.115	0.000	0.124	0.196

Omnibus:	0.183	Durbin-Watson:	2.650
Prob(Omnibus):	0.913	Jarque-Bera (JB):	0.281
Skew:	-0.164	Prob(JB):	0.869
Kurtosis:	2.657	Cond. No.	2.40e+04

```
# Run multiple regression of Wins to BB: Bases on Balls/Walks and Doubles
formula = 'W ~ BB + Doubles'
results_multiple = smf.ols(formula, data=baseball_df).fit()
results_multiple.summary()
```

OLS Regression Results

Dep. Variable:	W	R-squared:	0.820
Model:	OLS	Adj. R-squared:	0.807
Method:	Least Squares	F-statistic:	61.58
Date:	Tue, 10 Aug 2021	Prob (F-statistic):	8.71e-11
Time:	17:43:59	Log-Likelihood:	-128.01
No. Observations:	30	AIC:	262.0
Df Residuals:	27	BIC:	266.2
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-127.8913	49.431	-2.587	0.015	-229.316	-26.467
BB	0.1372	0.017	8.266	0.000	0.103	0.171
Doubles	0.1390	0.042	3.294	0.003	0.052	0.226

Omnibus:	0.742	Durbin-Watson:	2.355
Prob(Omnibus):	0.690	Jarque-Bera (JB):	0.709
Skew:	-0.017	Prob(JB):	0.702
Kurtosis:	2.248	Cond. No.	3.86e+04

References

- Downey, A. (2011). *Think stats: Probability and statistics for programmers*. O'Reilly.