BAlRifai_Assignment2

July 6, 2021

[996]:
```python
import pandas as pd
import numpy as np
from datetime import datetime
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
import scipy, pylab
import sqlite3 as sql
```

[997]:
```python
# Loading the lahman2014 dataset
lahman2014 = 'lahman2014.sqlite'
lahman2014_conn = sql.connect(lahman2014)
c = lahman2014_conn.cursor()
```

1 Part 1: Wrangling

1.0.1 Problem 1

[998]:
```python
# Query to get total winning percentage for each team from the Teams table,␣
↪grouping by year and teamID
query = "SELECT Teams.yearID, Teams.teamID, Teams.lgID, Teams.franchID, Teams.W␣
↪as totalWins, Teams.G as totalGames, CAST(Teams.W AS FLOAT) / CAST(Teams.G␣
↪AS FLOAT) * 100 as winPercentage FROM Teams GROUP BY Teams.yearID, Teams.
↪teamID ORDER BY Teams.yearID, Teams.teamID"
lahman2014_teams = pd.read_sql(query, lahman2014_conn)
lahman2014_teams
```

[998]:

	yearID	teamID	lgID	franchID	totalWins	totalGames	winPercentage
0	1871	BS1	NA	BNA	20	31	64.516129
1	1871	CH1	NA	CNA	19	28	67.857143
2	1871	CL1	NA	CFC	10	29	34.482759
3	1871	FW1	NA	KEK	7	19	36.842105
4	1871	NY2	NA	NNA	16	33	48.484848
…	…	…	…	…	…	…	…
2770	2014	SLN	NL	STL	90	162	55.555556
2771	2014	TBA	AL	TBD	77	162	47.530864
2772	2014	TEX	AL	TEX	67	162	41.358025
2773	2014	TOR	AL	TOR	83	162	51.234568

```
2774    2014    WAS   NL     WSN         96          162       59.259259

[2775 rows x 7 columns]
```

Description For the table above, I queried the sum of total wins (W) and total games (G) for each team for each year they played and then calculated their win percentage using the formula (number of wins / number of games * 100). This table includes a lot of extra teams that are NOT in the salaries table and will be ignored when joining the tables. This table also starts at the year 1871 while the salaries table starts at 1985. There are NO missing values in this table.

[999]:
```
# Query to get salaries for each team by year by using sum and grouping by year␣
↪and teamID again
query = "SELECT Salaries.yearID, Salaries.teamID, Salaries.lgID, sum(Salaries.
↪salary) as totalPayroll FROM Salaries GROUP BY Salaries.yearID, Salaries.
↪teamID ORDER BY Salaries.yearID, Salaries.teamID"
lahman2014_payrolls = pd.read_sql(query, lahman2014_conn)
lahman2014_payrolls
```

[999]:
```
     yearID teamID lgID  totalPayroll
0      1985    ATL   NL    14807000.0
1      1985    BAL   AL    11560712.0
2      1985    BOS   AL    10897560.0
3      1985    CAL   AL    14427894.0
4      1985    CHA   AL     9846178.0
..      …     …   …            …
855    2014    SLN   NL   120693000.0
856    2014    TBA   AL    72689100.0
857    2014    TEX   AL   112255059.0
858    2014    TOR   AL   109920100.0
859    2014    WAS   NL   131983680.0

[860 rows x 4 columns]
```

Description For the table above, I queried the sum of the salaries for each player in each team by year, resulting in a total payroll for each team over time. This table is useful in showing trends in pay over time (which can be seen by the sample is increasing). While this table has NO missing values, it does start much later in time than the Teams table, which will determine the type of join used for joining the two.

[1000]:
```
# Pandas right join into salaries (the smaller table) the values that match␣
↪yearID and teamID from the teams table
lahman2014_payrollwithwins = pd.merge(lahman2014_teams, lahman2014_payrolls, ␣
↪how='inner', left_on=['teamID','lgID', 'yearID'], right_on =␣
↪['teamID','lgID', 'yearID'])
lahman2014_payrollwithwins
```

```
[1000]:      yearID teamID lgID franchID  totalWins  totalGames  winPercentage  \
0      1985    ATL   NL      ATL         66         162      40.740741
1      1985    BAL   AL      BAL         83         161      51.552795
2      1985    BOS   AL      BOS         81         163      49.693252
3      1985    CAL   AL      ANA         90         162      55.555556
4      1985    CHA   AL      CHW         85         163      52.147239
..      …     …   …      …        …         …            …
853    2014    SLN   NL      STL         90         162      55.555556
854    2014    TBA   AL      TBD         77         162      47.530864
855    2014    TEX   AL      TEX         67         162      41.358025
856    2014    TOR   AL      TOR         83         162      51.234568
857    2014    WAS   NL      WSN         96         162      59.259259

     totalPayroll
0      14807000.0
1      11560712.0
2      10897560.0
3      14427894.0
4       9846178.0
..            …
853   120693000.0
854    72689100.0
855   112255059.0
856   109920100.0
857   131983680.0

[858 rows x 8 columns]
```

Description For the table above, I INNER joined the two previously created tables (salaries and win percentages over time) to show how payroll has affected win percentage and plot any relations in the coming parts. While the two tables separately DID NOT have missing data, there are many values that would not match up if I did a left/right join, leading to missing values because of the timeline difference between the tables. Using an inner join ensured that only exact matching records of ALL THREE teamID, lgID, and year were joined together, leading to no missing values. This was checked with pd.isnull.values.any() which returned false.

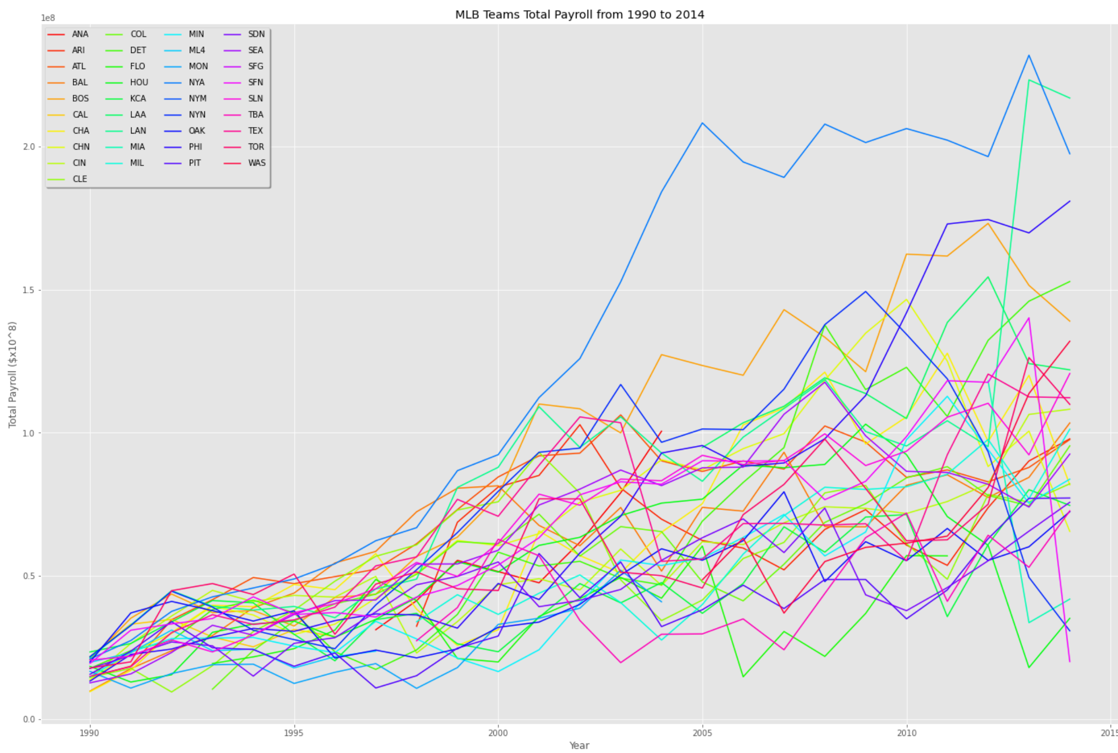
2 Part 2: Exploratory Data Analysis

2.1 2.1 Payroll Distribution

2.1.1 Problem 1

```
[1001]: # Get data past 1990
filtered_payrolls = lahman2014_payrolls[lahman2014_payrolls['yearID'] >= 1990].
 ↪reset_index()

# Group payrolls by teamID for plotting
```

```python
fig, ax = plt.subplots(figsize=(24,16))

# Set proper labels for time and payroll
ax.set_xlabel('Year')
ax.set_ylabel('Total Payroll ($x10^8)')

# Set more distinct colors
plt.gca().set_prop_cycle(plt.cycler('color', plt.cm.hsv(np.linspace(0, 1, 38))))

for label, group in filtered_payrolls.groupby(['teamID']):
    ax.plot(group['yearID'], group['totalPayroll'], label=label)

ax.legend(ncol=4,
          labelspacing=1.0,
          handletextpad=1.0, handlelength=2.0,
          fancybox=True, shadow=True)
plt.title("MLB Teams Total Payroll from 1990 to 2014")
plt.show()
```



2.1.2 Question 1

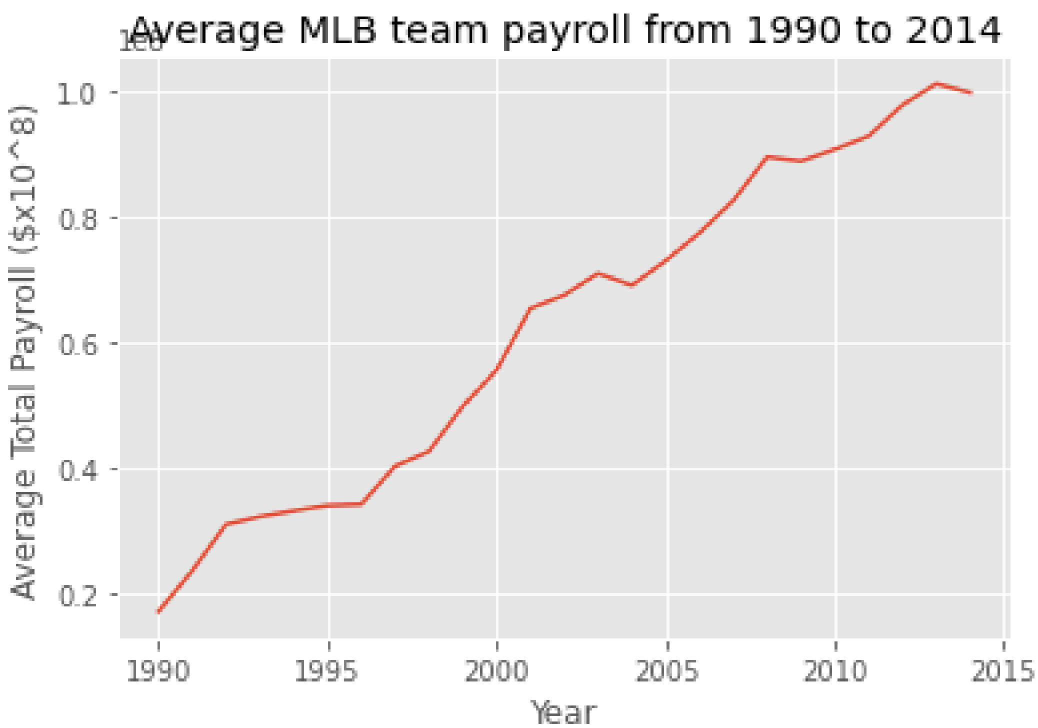
The most clear trend in the distribution of payrolls conditioned on time is the increase of payroll over time. Take the New York Mets (Dark Blue, highest line) for example, they started at around

\$0.25x10^8 total payroll in 1990 and their total team payroll is 8 TIMES that in 2014. Another trend that is seen is the spread between different teams in payrolls. While there is a heavy positive correlation between time and increased payroll, not all teams have increased at the same rate. The spread was much tighter in 1990 than it is in 2014, with some teams still making around the same that they made in 1990.

2.1.3 Problem 3

[1002]:
```
ax = filtered_payrolls.groupby('yearID').totalPayroll.mean().plot()
ax.set_xlabel("Year")
ax.set_ylabel("Average Total Payroll ($x10^8)")
plt.title("Average MLB team payroll from 1990 to 2014")
```

[1002]:
```
Text(0.5, 1.0, 'Average MLB team payroll from 1990 to 2014')
```



Description The plot above shows the previously mentioned trend that the mean payroll in the MLB has drastically increased since 1990. Although not all teams saw an increase in payroll, the mean is pulled higher by a few teams that saw a huge increase such as the New York Yankees.

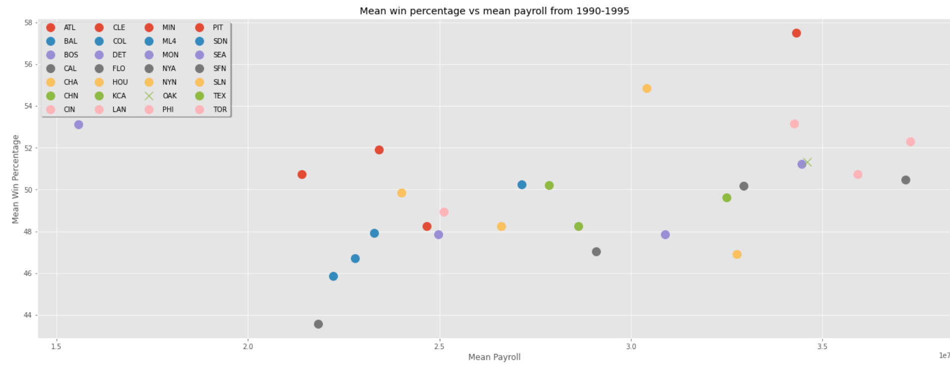
2.2 2.2 Correlation between payroll and winning percentage

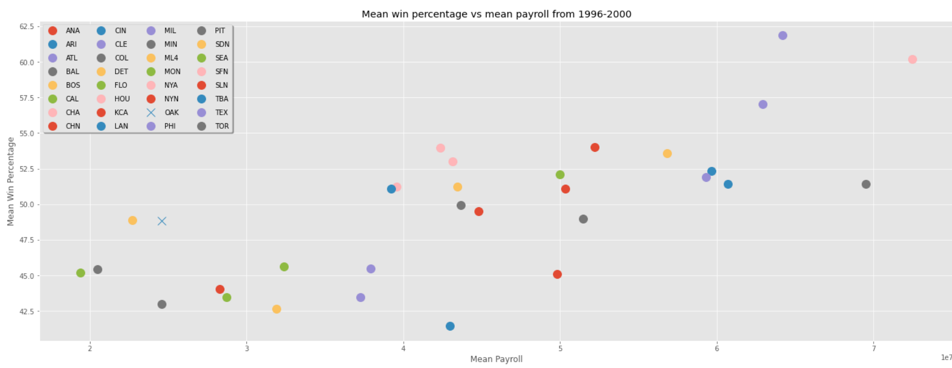
2.2.1 Problem 4

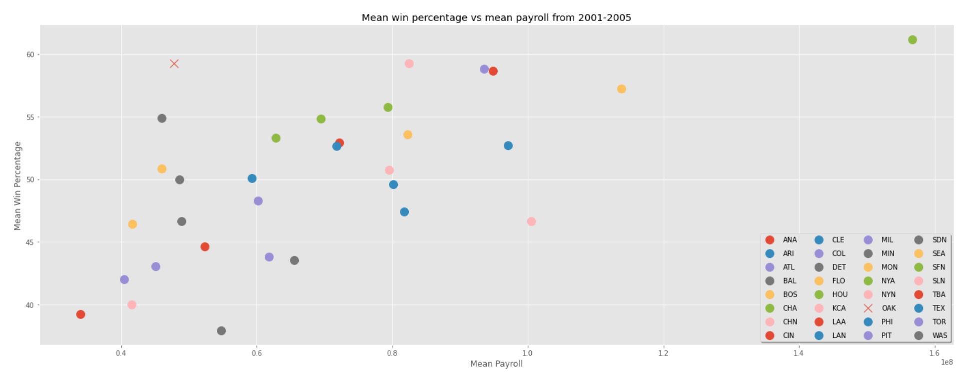
[1003]:
```
filtered_payrollswithwins =␣
↪lahman2014_payrollwithwins[lahman2014_payrollwithwins['yearID'] >= 1990].
↪reset_index()
binned_payrollswithwins= pd.DataFrame(filtered_payrollswithwins)
binned_payrollswithwins['bin'] = pd.cut(x=filtered_payrollswithwins['yearID'],␣
↪bins=[1989, 1995, 2000, 2005, 2010, 2015],
                     labels=['1990-1995', '1996-2000', '2001-2005',
                             '2006-2010', '2011-2015'])

#Retrieve each bin to create 5 subplots
payrollswithwin_1990 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'1990-1995')].reset_index().groupby('teamID')
payrollswithwin_1995 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'1996-2000')].reset_index().groupby('teamID')
payrollswithwin_2000 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2001-2005')].reset_index().groupby('teamID')
payrollswithwin_2005 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2006-2010')].reset_index().groupby('teamID')
payrollswithwin_2010 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2011-2015')].reset_index().groupby('teamID')

# For each bin (year range):
# 1. set title and axis titles
# 2. plot the group payroll mean and the group winpercentage mean
# 3. Create legend with markers
# 4. Mark OAK data by an X
fig, ax = plt.subplots(5, 1, figsize=(24,50))
for name, group in payrollswithwin_1990:
    ax[0].title.set_text("Mean win percentage vs mean payroll from 1990-1995")
    if (name == 'OAK'):
        ax[0].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[0].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[0].set_xlabel("Mean Payroll")
    ax[0].set_ylabel("Mean Win Percentage")
    ax[0].legend(ncol=4,
               labelspacing=1.0,
               handletextpad=1.0, handlelength=2.0,
               fancybox=True, shadow=True)

for name, group in payrollswithwin_1995:
```

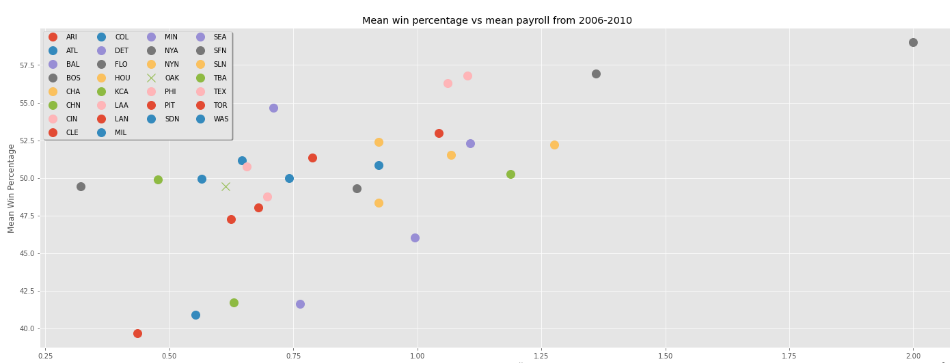
```
    ax[1].title.set_text("Mean win percentage vs mean payroll from 1996-2000")
    if (name == 'OAK'):
        ax[1].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[1].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[1].set_xlabel("Mean Payroll")
    ax[1].set_ylabel("Mean Win Percentage")
    ax[1].legend(ncol=4,
              labelspacing=1.0,
              handletextpad=1.0, handlelength=2.0,
              fancybox=True, shadow=True)

for name, group in payrollswithwin_2000:
    ax[2].title.set_text("Mean win percentage vs mean payroll from 2001-2005")
    if (name == 'OAK'):
        ax[2].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[2].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[2].set_xlabel("Mean Payroll")
    ax[2].set_ylabel("Mean Win Percentage")
    ax[2].legend(ncol=4,
              labelspacing=1.0,
              handletextpad=1.0, handlelength=2.0,
              fancybox=True, shadow=True)

for name, group in payrollswithwin_2005:
    ax[3].title.set_text("Mean win percentage vs mean payroll from 2006-2010")
    if (name == 'OAK'):
        ax[3].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[3].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[3].set_xlabel("Mean Payroll")
    ax[3].set_ylabel("Mean Win Percentage")
    ax[3].legend(ncol=4,
              labelspacing=1.0,
              handletextpad=1.0, handlelength=2.0,
              fancybox=True, shadow=True)

for name, group in payrollswithwin_2010:
    ax[4].title.set_text("Mean win percentage vs mean payroll from 2011-2015")
```

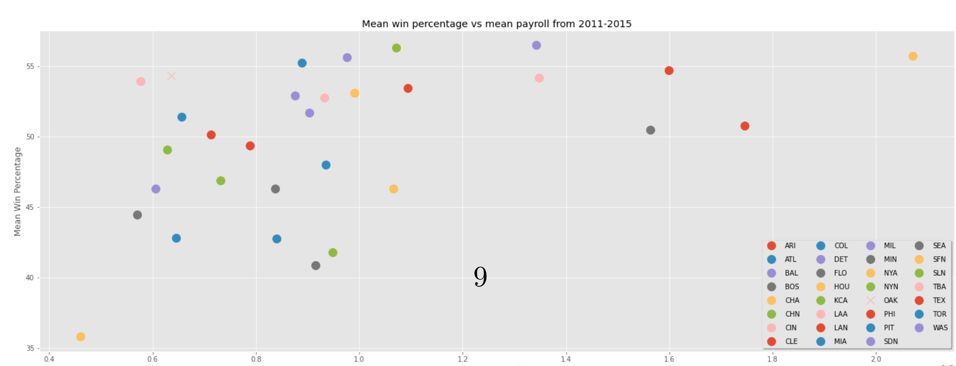
```
    if (name == 'OAK'):
        ax[4].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[4].plot(group.totalPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[4].set_xlabel("Mean Payroll")
    ax[4].set_ylabel("Mean Win Percentage")
    ax[4].legend(ncol=4,
               labelspacing=1.0,
               handletextpad=1.0, handlelength=2.0,
               fancybox=True, shadow=True)

plt.show()
```


Mean win percentage vs mean payroll from 1990-1995
Mean Win Percentage
Mean Payroll
1e7


Mean win percentage vs mean payroll from 1996-2000
Mean Win Percentage
Mean Payroll
1e7


Mean win percentage vs mean payroll from 2001-2005
Mean Win Percentage
Mean Payroll
1e8


Mean win percentage vs mean payroll from 2006-2010
Mean Win Percentage
Mean Payroll
1e8


Mean win percentage vs mean payroll from 2011-2015
Mean Win Percentage
Mean Payroll
1e8

2.2.2 Question 2

While the mean payrols have been rising a lot through the time periods, the mean win percentage has largely remained the same, with a limit of around 60% for most teams. I believe that there is a slight correlation between higher payrolls and win percentages, as can be seen most clearly in the 2006-2010 time period. But it is a very weak correlation since every time period, the mean win percentage does not keep increasing as mean payroll is increasing and there is a lot of variance in the plot.

Once again, the New York Yankees stand out nearly every year as having a high pay mean and high win percentage mean, showing that they are paying good money for good players.

The Oakland A's spending is marked by an "X" in each time period. While their mean pay has nearly doubled since 1990-1995 (going from about .35x10^8 to .65x10^8), their win percentage has remained around 50-60% and is usually an outlier in the lower paid teams for having a high win percentage, so the Moneyball year is, in fact, a thing.

3 Part 3: Data transformations

3.1 3.1 Standardizing across years

3.1.1 Problem 5

[1004]:
```
filtered_payrollswithwins =␣
↪lahman2014_payrollwithwins[lahman2014_payrollwithwins['yearID'] >= 1990].
↪reset_index()
standardized_payrollwithwins = pd.DataFrame(filtered_payrollswithwins)
standardized_values = []

# Get mean and standard deviation of each year (j)
payroll_mean = standardized_payrollwithwins.groupby('yearID')['totalPayroll'].
↪mean()
payroll_std = standardized_payrollwithwins.groupby('yearID')['totalPayroll'].
↪std()

for name, group in standardized_payrollwithwins.groupby('teamID'):
    year_group = group.groupby('yearID')['totalPayroll']
    for key, payroll in year_group:
        team_payroll_in_year = payroll.iloc[0]
        standardized_values.append((team_payroll_in_year - payroll_mean[key]) /␣
↪payroll_std[key])

standardized_payrollwithwins['standarizedPayroll'] = standardized_values
standardized_payrollwithwins
```

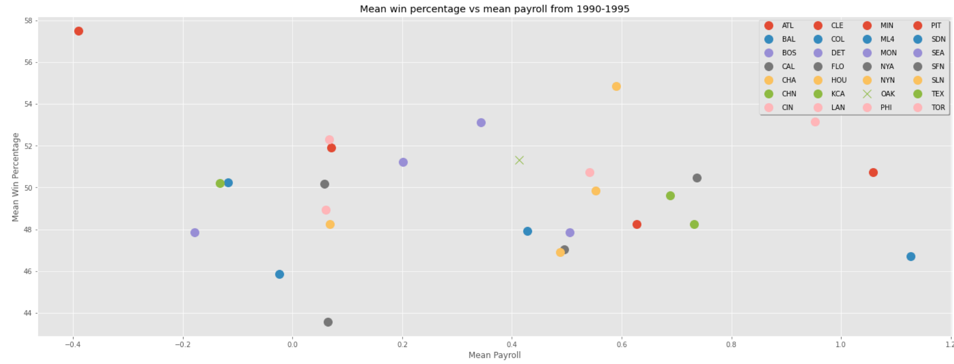
[1004]:
```
     index  yearID teamID lgID franchID  totalWins  totalGames  winPercentage  \
0      130    1990    ATL   NL      ATL         65         162      40.123457
1      131    1990    BAL   AL      BAL         76         161      47.204969
2      132    1990    BOS   AL      BOS         88         162      54.320988
3      133    1990    CAL   AL      ANA         80         162      49.382716
4      134    1990    CHA   AL      CHW         94         162      58.024691
..     …      …     …   …     …        …         …
723    853    2014    SLN   NL      STL         90         162      55.555556
724    854    2014    TBA   AL      TBD         77         162      47.530864
725    855    2014    TEX   AL      TEX         67         162      41.358025
726    856    2014    TOR   AL      TOR         83         162      51.234568
727    857    2014    WAS   NL      WSN         96         162      59.259259

     totalPayroll  standarizedPayroll
0      14555501.0           -0.698639
1       9680084.0           -0.086369
2      20558333.0            0.271410
3      21720000.0           -0.190214
4       9491500.0           -0.721244
..            …                  …
723   120693000.0           -0.769040
724    72689100.0           -0.709594
725   112255059.0           -0.459099
726   109920100.0            0.257062
727   131983680.0            0.704160

[728 rows x 10 columns]
```

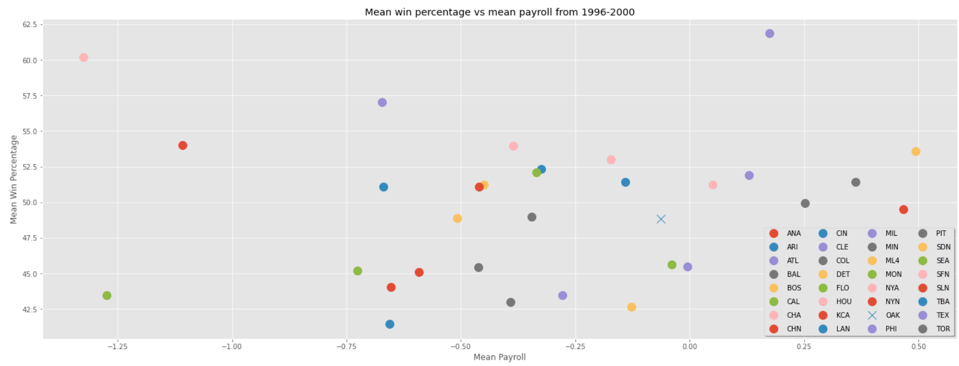
3.1.2 Problem 6

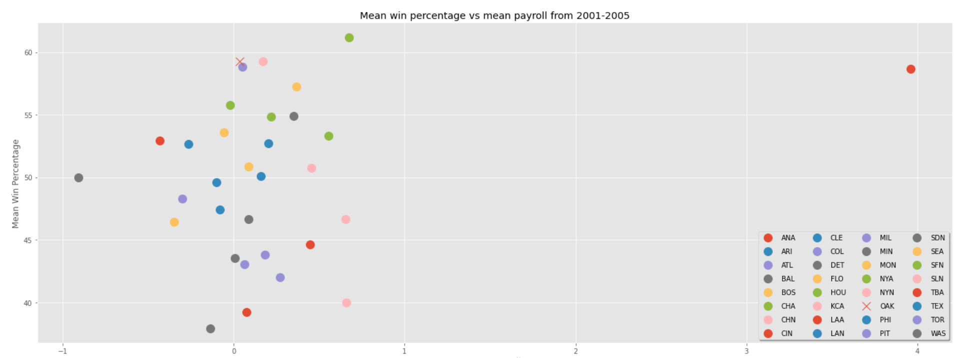
[1005]:
```python
binned_payrollswithwins = pd.DataFrame(standardized_payrollwithwins)
binned_payrollswithwins['bin'] = pd.
↪cut(x=standardized_payrollwithwins['yearID'], bins=[1989, 1995, 2000, 2005,␣
↪2010, 2015],
                     labels=['1990-1995', '1996-2000', '2001-2005',
                             '2006-2010', '2011-2015'])

#Retrieve each bin to create 5 subplots
payrollswithwin_1990 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'1990-1995')].reset_index().groupby('teamID')
payrollswithwin_1995 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'1996-2000')].reset_index().groupby('teamID')
payrollswithwin_2000 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2001-2005')].reset_index().groupby('teamID')
payrollswithwin_2005 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2006-2010')].reset_index().groupby('teamID')
```

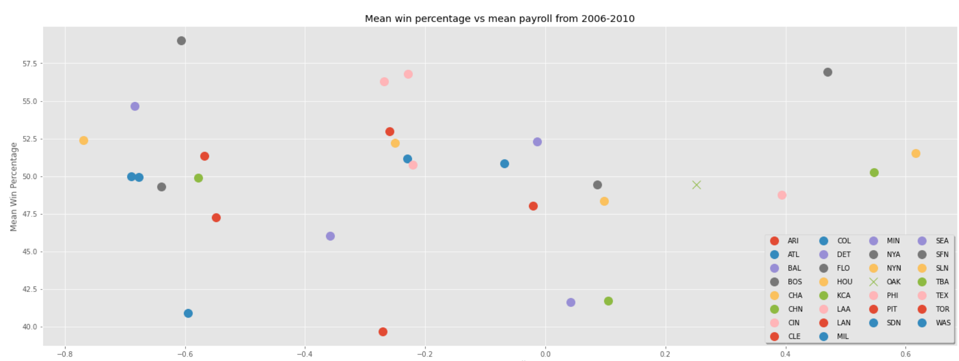
```
payrollswithwin_2010 = binned_payrollswithwins[(binned_payrollswithwins.bin ==␣
↪'2011-2015')].reset_index().groupby('teamID')

# For each bin (year range):
# 1. set title and axis titles
# 2. plot the group payroll mean and the group winpercentage mean
# 3. Create legend with markers
# 4. Mark OAK data by an X
fig, ax = plt.subplots(5, 1, figsize=(24,50))
for name, group in payrollswithwin_1990:
    ax[0].title.set_text("Mean win percentage vs mean payroll from 1990-1995")
    if (name == 'OAK'):
        ax[0].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[0].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[0].set_xlabel("Mean Payroll")
    ax[0].set_ylabel("Mean Win Percentage")
    ax[0].legend(ncol=4,
                labelspacing=1.0,
                handletextpad=1.0, handlelength=2.0,
                fancybox=True, shadow=True)

for name, group in payrollswithwin_1995:
    ax[1].title.set_text("Mean win percentage vs mean payroll from 1996-2000")
    if (name == 'OAK'):
        ax[1].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[1].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
↪marker='o', linestyle='', ms=12, label=name)
    ax[1].set_xlabel("Mean Payroll")
    ax[1].set_ylabel("Mean Win Percentage")
    ax[1].legend(ncol=4,
                labelspacing=1.0,
                handletextpad=1.0, handlelength=2.0,
                fancybox=True, shadow=True)

for name, group in payrollswithwin_2000:
    ax[2].title.set_text("Mean win percentage vs mean payroll from 2001-2005")
    if (name == 'OAK'):
        ax[2].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
↪marker='x', linestyle='', ms=12, label=name)
    else:
```

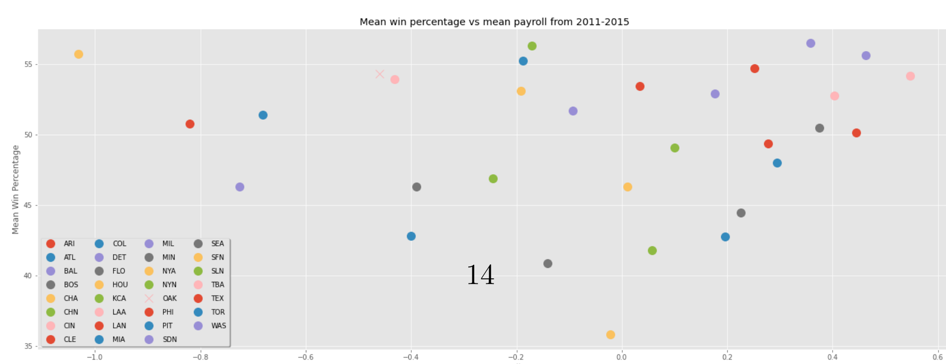
```
        ax[2].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
 ↪marker='o', linestyle='', ms=12, label=name)
    ax[2].set_xlabel("Mean Payroll")
    ax[2].set_ylabel("Mean Win Percentage")
    ax[2].legend(ncol=4,
               labelspacing=1.0,
               handletextpad=1.0, handlelength=2.0,
               fancybox=True, shadow=True)

for name, group in payrollswithwin_2005:
    ax[3].title.set_text("Mean win percentage vs mean payroll from 2006-2010")
    if (name == 'OAK'):
        ax[3].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
 ↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[3].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
 ↪marker='o', linestyle='', ms=12, label=name)
    ax[3].set_xlabel("Mean Payroll")
    ax[3].set_ylabel("Mean Win Percentage")
    ax[3].legend(ncol=4,
               labelspacing=1.0,
               handletextpad=1.0, handlelength=2.0,
               fancybox=True, shadow=True)

for name, group in payrollswithwin_2010:
    ax[4].title.set_text("Mean win percentage vs mean payroll from 2011-2015")
    if (name == 'OAK'):
        ax[4].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
 ↪marker='x', linestyle='', ms=12, label=name)
    else:
        ax[4].plot(group.standarizedPayroll.mean(), group.winPercentage.mean(),␣
 ↪marker='o', linestyle='', ms=12, label=name)
    ax[4].set_xlabel("Mean Payroll")
    ax[4].set_ylabel("Mean Win Percentage")
    ax[4].legend(ncol=4,
               labelspacing=1.0,
               handletextpad=1.0, handlelength=2.0,
               fancybox=True, shadow=True)

plt.show()
```


Mean win percentage vs mean payroll from 1990-1995
Mean Win Percentage
Mean Payroll


Mean win percentage vs mean payroll from 1996-2000
Mean Win Percentage
Mean Payroll


Mean win percentage vs mean payroll from 2001-2005
Mean Win Percentage
Mean Payroll


Mean win percentage vs mean payroll from 2006-2010
Mean Win Percentage
Mean Payroll


Mean win percentage vs mean payroll from 2011-2015
Mean Win Percentage
Mean Payroll

3.1.3 Question 3

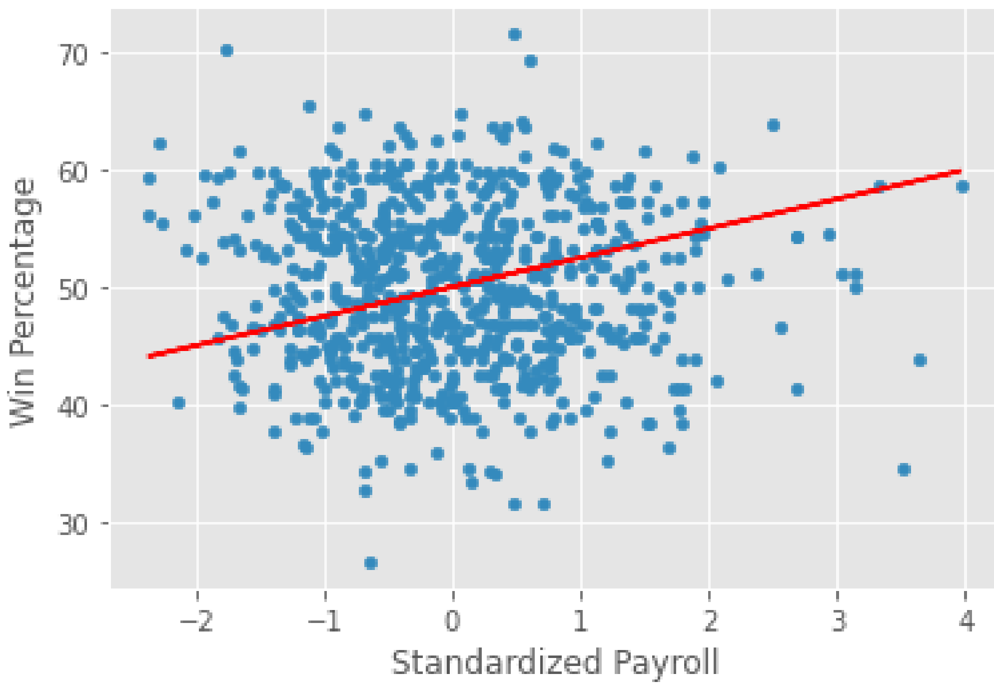
Standardizing the payrolls meant that for each year, the differences between teams is made more pronounced by subtracting the mean payroll and dividing by the standard deviation. This results in a smaller spead on the X axis, centering it around 0, clearly showing who spent more than average in that year and who spent less. The plots are also all similar since the payroll has been standardized instead of it being previously vastly different numbers on the X axis. Once again, a small positive correlation between payroll and win percentage can be seen, but for the most part, the data is very spread out with high variance, and does not lead to any solid conclusions on payroll having a large effect on wins.

3.2 3.2 Expected wins

3.2.1 Problem 7

[1007]:
```
# Expected win percentage using formula from documentation https://github.com/
↪cmsc320/summer2021/tree/main/project2
standardized_payrollwithwins['expected_win'] = standardized_payrollwithwins.
↪apply (lambda row: 50 + 2.5 * row['standarizedPayroll'], axis=1)

ax = standardized_payrollwithwins.plot.scatter(x='standarizedPayroll',␣
↪y='winPercentage')
standardized_payrollwithwins.plot(x='standarizedPayroll', y='expected_win',␣
↪color='Red', legend=False, ax=ax)
ax.set_xlabel("Standardized Payroll")
ax.set_ylabel("Win Percentage")
```

[1007]:
```
Text(0, 0.5, 'Win Percentage')
```

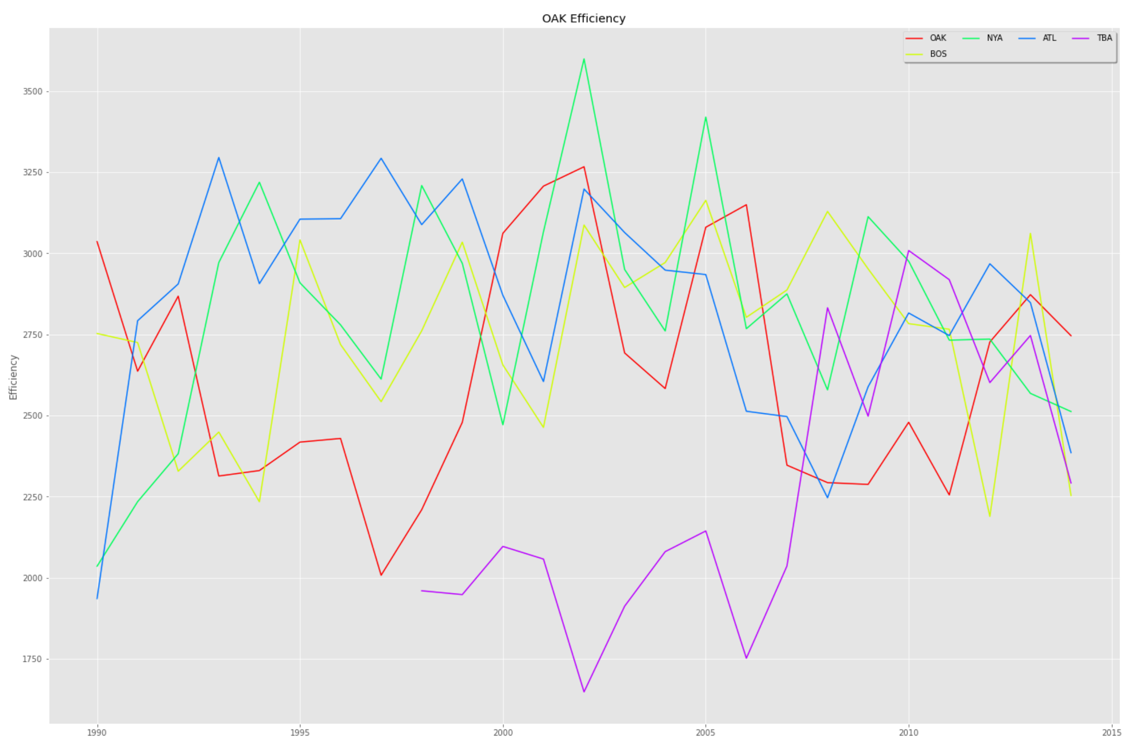


3.3 3.3 Spending efficiency

3.3.1 Problem 8

[1019]:
```
# efficiency using formula from documentation https://github.com/cmsc320/
↪summer2021/tree/main/project2
standardized_payrollwithwins['efficiency'] = standardized_payrollwithwins.apply␣
↪(lambda row: row['winPercentage'] * row['expected_win'], axis=1)

# Group payrolls by teamID for plotting
fig, ax = plt.subplots(figsize=(24,16))

# Set proper labels for time and payroll
ax.set_xlabel('Year')
ax.set_ylabel('Efficiency')

#Set more distinct
plt.gca().set_prop_cycle(plt.cycler('color', plt.cm.hsv(np.linspace(0, 1, 6))))

# Plotting Oakland As
for label, group in standardized_payrollwithwins.groupby(['teamID']):
    if (label == 'OAK'):
        ax.plot(group['yearID'], group['efficiency'], label=label)

# Plotting Boston
```

```
for label, group in standardized_payrollwithwins.groupby(['teamID']):
    if (label == 'BOS'):
        ax.plot(group['yearID'], group['efficiency'], label=label)

# Plotting New YOrk Yankees
for label, group in standardized_payrollwithwins.groupby(['teamID']):
    if (label == 'NYA'):
        ax.plot(group['yearID'], group['efficiency'], label=label)

# Plotting Atlanta
for label, group in standardized_payrollwithwins.groupby(['teamID']):
    if (label == 'ATL'):
        ax.plot(group['yearID'], group['efficiency'], label=label)

# Plotting Tampa Bay
for label, group in standardized_payrollwithwins.groupby(['teamID']):
    if (label == 'TBA'):
        ax.plot(group['yearID'], group['efficiency'], label=label)

        ax.legend(ncol=4,
           labelspacing=1.0,
           handletextpad=1.0, handlelength=2.0,
           fancybox=True, shadow=True)
plt.title("OAK Efficiency")
plt.show()
```



3.3.2 Question 4

The plot above shows a few key team's efficiency throughout the Moneyball period. This plot shows more than just increased win percentage as a function increased payroll as was shown for Questions 2 and 3. This plot shows how efficient a team is at reaching their expected win rate based on their pay. The more efficient they are, the more easily they can live up to their "expected pay worth." For example, a team that has a very high pay roll but a very low win percentage would be considered low efficiency and vice-versa.

The Oakland's efficiency during the Moneyball year (2002) is extremely high, almost matching the efficiency of the New York Yankees, yet at a lower budget. This shows that teams with lower payrolls can be as efficient as teams with higher payrolls, but there are still teams with lower payrolls that severely underperform, such as Tampa Bay, which can be seen in the first plot to be among the lowest. Although Tampa Bay is not as efficient as the other teams, their efficiency has increased over time despite not having a large change in payroll. In conclusion, payroll does have an effect on a teams win rate BUT it does not always result in an increased win rate if not spent efficiently like Oakland does.

```
[ ]:
```