In [1]:
```
# Loading necessary libraries
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
from datetime import datetime
from datetime import time

import plotly
import plotly.graph_objs as go
import chart_studio.plotly as py
from chart_studio.plotly import plot, iplot
import plotly.express as px

# Setting pandas to display columns
pd.set_option('display.max_columns', None)
```

In [2]:
```
# Loading end of drive file
nfl_small2_end_of_drive = pd.read_csv('nfl_small_end_of_drive.csv',index
_col=1,\
    dtype= {'ARI' : 'str','ATL' : 'str', 'BAL' : 'str', 'BUF' : 'str',
'CAR' : 'str',
            'CHI' : 'str', 'CIN' : 'str', 'CLE' : 'str', 'DAL' : 'str',
'DEN' : 'str',
            'DET' : 'str', 'GB' : 'str', 'HOU' : 'str', 'IND' : 'str',
'JAX' : 'str',
            'KC' : 'str', 'LA' : 'str', 'LAC' : 'str', 'MIA' : 'str',
'MIN' : 'str',
            'NE' : 'str', 'NO' : 'str', 'NYG' : 'str','NYJ' : 'str', 'O
AK' : 'str',
            'PHI' : 'str', 'PIT' : 'str','SEA' : 'str', 'SF' : 'str',
'TB' : 'str',
            'TEN' : 'str', 'WAS' : 'str'})

# Getting teams to add team matrix for easier filtering
teams = list(nfl_small2_end_of_drive.groupby('posteam').sum().index)

# Creating Home and Away Win Columns
nfl_small2_end_of_drive['home_team_win'] = np.where(nfl_small2_end_of_dr
ive['total_home_score']\
                                > nfl_small2_end_of_drive['total_away_scor
e'],1,0)

nfl_small2_end_of_drive['away_team_win'] = np.where(nfl_small2_end_of_dr
ive['total_home_score']\
                                < nfl_small2_end_of_drive['total_away_scor
e'],1,0)
```

In [103]:
```
# Loading end of game file
nfl_end_of_game = pd.read_csv('nfl_end_of_game.csv',index_col=0)

# Adding numeric win group to allow for correlation
win_dict = {'10+ Wins':2,'Between 6 and 9':1,'5 or Less':0}
nfl_end_of_game['win_group_num'] = nfl_end_of_game['binned'].map(win_dic
t)
```

In [104]:
```
# Team wins overall for sorting
team_wins_overall = []
years = [2015,2016,2017,2018]
year_list = []
team_list = []

for team in teams:
    for year in years:
        home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive
['end_of_game'] == 1) &\
                                                (nfl_small2_end_of_drive
['year'] == year) &\
                                                (nfl_small2_end_of_drive
[team] == 'H')
                                               ]['home_team_win'])

        away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive
['end_of_game'] == 1) &\
                                                (nfl_small2_end_of_drive
['year'] == year) &\
                                                (nfl_small2_end_of_drive
[team] == 'A')
                                               ]['away_team_win'])

        # Due to lack of data, normalizing 2018 to 16 week season
        if year == 2018:
            all_wins = np.round((home_wins + away_wins) * 16/14,0)
            team_wins_overall.append(round(all_wins,0))
            year_list.append(year)
            team_list.append(team)

        else:
            all_wins = home_wins + away_wins
            team_wins_overall.append(round(all_wins,0))
            year_list.append(year)
            team_list.append(team)

team_by_wins = pd.DataFrame({'team':team_list,'wins':team_wins_overall,
'year':year_list}).sort_values(by=['wins'])

# Creating new agg win team list to resort the dataframe by agg win tota
ls
team_win_list = list(team_by_wins.groupby('team').agg({'wins':'sum'}).so
rt_values(by=['wins']).index)

# Team wins overall for sorting by agg win totals
team_wins_overall = []
years = [2015,2016,2017,2018]
year_list = []
team_list = []

for team in team_win_list:
    for year in years:
        home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive
['end_of_game'] == 1) &\
                                                (nfl_small2_end_of_drive
```

```
['year'] == year) &\
                                                    (nfl_small2_end_of_drive
[team] == 'H')
                                                   ]['home_team_win'])

        away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive
['end_of_game'] == 1) &\
                                                    (nfl_small2_end_of_drive
['year'] == year) &\
                                                    (nfl_small2_end_of_drive
[team] == 'A')
                                                   ]['away_team_win'])

        # Due to lack of data, normalizing 2018 to 16 week season
        if year == 2018:
            all_wins = np.round((home_wins + away_wins) * 16/14,0)
            team_wins_overall.append(round(all_wins,0))
            year_list.append(year)
            team_list.append(team)

        else:
            all_wins = home_wins + away_wins
            team_wins_overall.append(round(all_wins,0))
            year_list.append(year)
            team_list.append(team)

team_by_wins = pd.DataFrame({'team':team_list,'wins':team_wins_overall,
'year':year_list})
```

In [105]:
```
# Creating agg Weather DF
nfl_small2_end_of_drive['Time (EST)'] = pd.to_datetime(nfl_small2_end_of
_drive['Time (EST)'])
def agg_func_for_weather(x):
    values = {
        'Temperature (°C)':x['Temperature (°C)'].mean(),
        'Air Pressure (hPa)':x['Air Pressure (hPa)'].mean(),
        'City':x['City'].iloc[0],
        'Field':x['Field'].iloc[0],
        'Dewpoint (°C)':x['Dewpoint (°C)'].mean(),
        'Precipitation (mm)':x['Precipitation (mm)'].mean(),
        'Wind Speed (km/h)':x['Wind Speed (km/h)'].mean(),
        'Roof':x['Roof'].iloc[0]
    }
    return pd.Series(values)


weather_avg_df = nfl_small2_end_of_drive.groupby(['game_id']).apply(lamb
da x: agg_func_for_weather(x))
weather_avg_df = weather_avg_df.fillna(0)
```

In [106]:
```
# Merging End of Game DF with Weather DF
nfl_df = nfl_end_of_game.merge(weather_avg_df, how='left', left_on=['gam
e_id'], right_on=['game_id'])
```

In [107]:
```
# Adding Weather Categorical Variables
import re

def my_func(x):
    line = re.sub('ideal', '', x)
    if len(line) == 2:
        return 'ideal'
    return line


nfl_df['temp_bin'] = pd.cut(x=nfl_df['Temperature (°C)'], bins=[-np.inf,15,25,np.inf], labels=['<15°C', 'ideal', '>25°C'])
nfl_df['wind_bin'] = pd.cut(x=nfl_df['Wind Speed (km/h)'], bins=[-1,15,np.inf], labels=['<=15km/h', '>15km/h'])
nfl_df['rain_bin'] = pd.cut(x=nfl_df['Precipitation (mm)'], bins=[-1,0.5,np.inf], labels=['<=0.5mm/h', '>0.5mm/h'])
nfl_df['weather_bin'] = nfl_df['temp_bin'].astype('str') + ' ' + nfl_df['wind_bin'].astype('str') + ' ' + nfl_df['rain_bin'].astype('str')
nfl_df['weather_bin'] = nfl_df['weather_bin'].apply(lambda x: my_func(x))
```

Weather Effects on NFL Play-By-Play Performance

Table of Contents:

1. Motivation (http://localhost:8888/notebooks/Final%20Project%20Notebook.ipynb#Motivation)
2. Background on American Football (http://localhost:8888/notebooks/Final%20Project%20Notebook.ipynb#Background-on-American-Football)
3. Data Sources (http://localhost:8888/notebooks/Final%20Project%20Notebook.ipynb#The-Data)
4. Data Cleaning and Combining (http://localhost:8888/notebooks/Final%20Project%20Notebook.ipynb#Cleaning-and-Combining)
5. The Analysis (http://localhost:8888/notebooks/Final%20Project%20Notebook.ipynb#The-Analysis)
6. Conclusion

Motivation

A common thought as it pertains to the affects of weather on NFL games is that weather plays an important factor on how teams choose play calling and how they perform given certain weather conditions. We are aiming to understand how weather changes intragame affect how teams perform in the moment and ultimately help determine the outcome.

Background on American Football

American Football is played between two teams in which each team is looking to advance a football down a 100 yard field to score a Touchdown (7 pts) or Field Goal (3 pts). Both teams are on the field at the same time, and the team advancing the football is called the **Offense (blue)** and the team trying to keep the Offense from advancing the football is call the **Defense (red).**

In [ ]:

The Game

Each game is made up of 4 15 minute quarters. Teams will flip a coin to see who starts with the ball at the beginning of the game, and the team that does not start with the ball at the beginning of the game will get the ball at halftime. After the first two quarters, we have Halftime, where the teams will take a short break.

A Series

The team on offense has four plays, or downs, to get 10 yards and reset their down counter. If a team does not get 10 yards on 4 downs, or gives the ball to the other defending team (turnover), then that team turns the ball over. A series ends when a team has turned the ball over or gotten ten or more yards.

A Drive

A Drive is a collection of sequential series run by the offense until they have either score or turned the ball over (which also happens at halftime).

Playing conditions

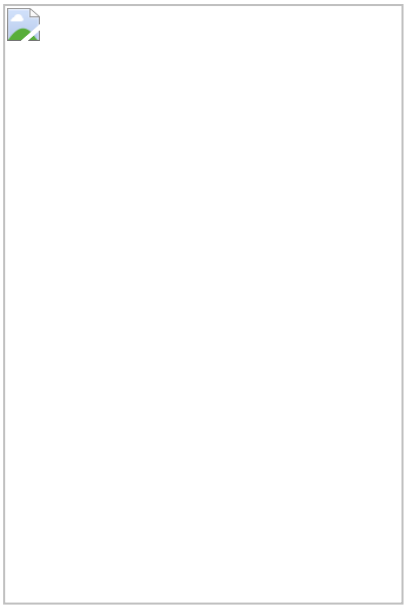
Teams play in stadiums seating between 60 and 80k people. They range in design, but most fit into three categories; Open Air (or exposed to outside weather) or Enclosed (not exposed to outside weather) or Retractable (where the roof can open if the team chooses to do so).

The field can be made up of a couple types of surfaces:
-Natural Grass
-Artifical Turf (Used mostly for Enclosed Stadiums, but also in some Open Air Stadiums)

In [ ]:

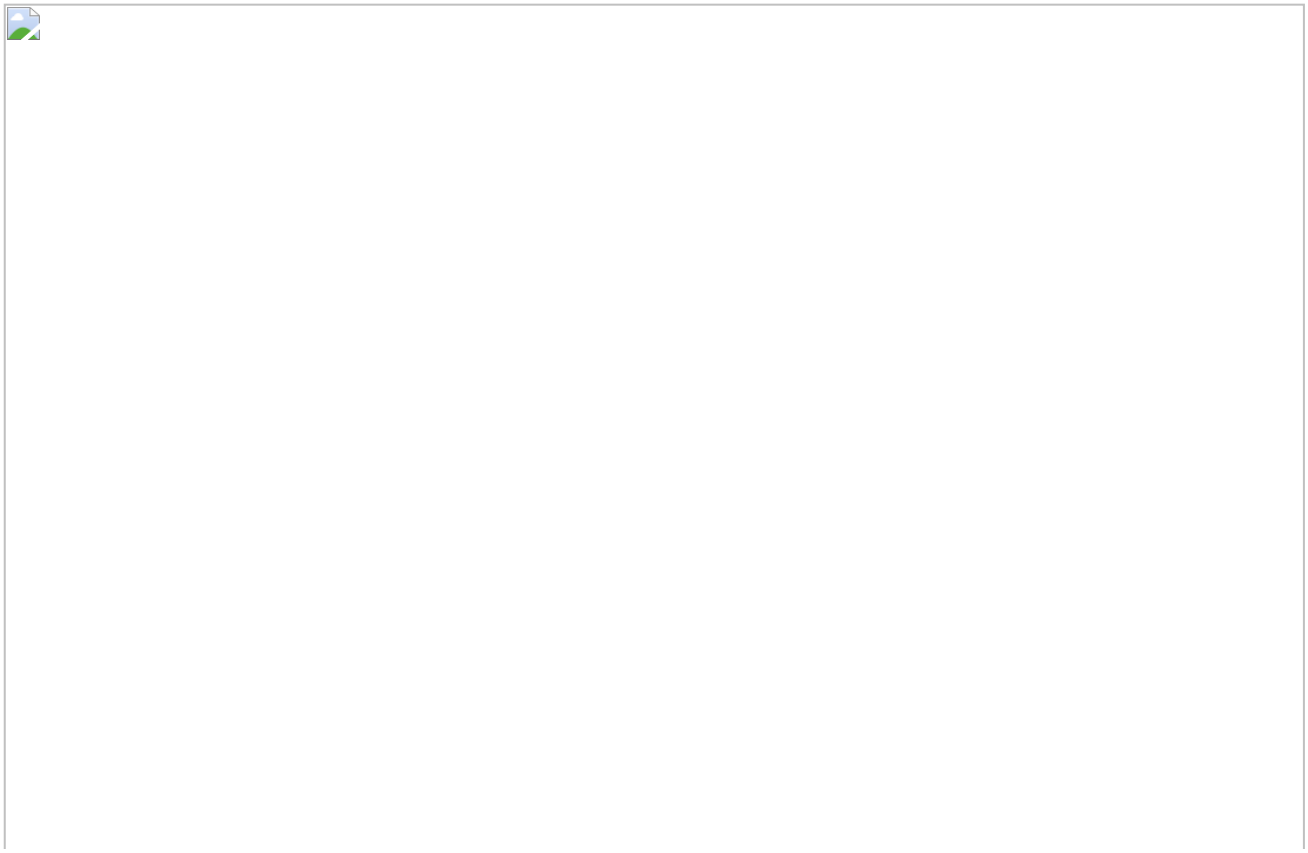
The NFL

The NFL, or National Football League, is the American league for American Football, and considered the most prevalant in the world. There are 32 teams from across the continental USA split into two conferences call the American Football Conference (AFC) and National Football Conference (NFC).



In [ ]:

The Teams



The Data

NFL Play-By-Play:

The NFL Play-By-Play data downloaded from Kaggle (https://www.kaggle.com/maxhorowitz/nflplaybyplay2009to2016#NFL%20Play%20by%20Play%202009-2018%20(v5).csv) is structured exactly as it sounds, and has a plethora of variables for each individual play in an NFL preseason and regular season from 2009-2018. We only used data from 2015-2018 for our analysis. While the NFL Play-by-Play data initially had 255 columns, we removed over 200 to give us a more manageable dataset of just metrics we chose to analyze. Most metrics removed were related to player specific stats, and we are only analyzing on the team level.

Data Attributes:

1. game_id - identifier using year/month/day/game to identify single games
2. game_date - Date game took place
3. game_seconds_remaining - Seconds left in game, used to approximate actual time in EST
4. home_team - Team/Location for game (except when games are in London of Mexico City)
5. away_team - Team playing Home Team
6. posteam - Team with position of the ball
7. drive - Drive of the Game (Incremental)
8. ydsnet - Yards gained so far on a drive. Includes penalties
9. desc - Play description. Used to removed "bad plays" (ex: coin tosses, pentalties, kickoffs)
10. play_type - What type of play was run. (ex: pass, run, kick)
11. yards_gained - Actual yards gain during play. Excludes penalties
12. pass_length - Length of Pass completion
13. field_goal_result - Made or missed Field Goal on play
14. total_home_score - Running score for home team in game
15. total_away_score - Running score for away team in game
16. field_goal_attempt - Field Goal attempted on play

Meteostat Weather:

Weather data was pulled from Meteostat API, which uses multiple sources to aggregate their data (Deutscher Wetterdienst - Open Data, Deutscher Wetterdienst - Climate Data Center, NOAA - National Weather Service, NOAA - Global Historical Climatology Network). First, finding a station for each city is necessary, which is done neatly by pinging their 'search stations' with each city name necessary. Some of these cities, such as Foxborough and Green Bay, did not have enough data in their system, so proxies had to be used for close cities that were larger. After this stage, the API was utilized to receive historical data in a JSON format at an hourly time stamp. Some data was not exactly hourly, so the added_time column was added. This JSON was then transformed into a dataframe and then appended to one another so that there was one large dataframe with all of the weather in it.

Data Attributes:

1. Time (GMT) - Time measurement was taken in Greenwich Mean Time (GMT)
2. Temperature (°C) - Temperature at Time (GMT)

3. Dewpoint (°C) - Dewpoint at Time (GMT)
4. Humidity (%) - Relative humidity at Time (GMT)
5. Precipitation (mm) - Amount of precipitation at the given Time (GMT)
6. Wind Speed (km/h) - Wind speed at Time (GMT)
7. Wind Direction (deg) - Wind direction at Time (GMT)
8. Air Pressure (hPa) - Barometric air pressure at Time (GMT)
9. Team Abbreviation - NFL abbreviation of team (NE, CLE, SEA, etc.) **Manually Added to combine with NFL data
10. City - City in which the temperature is being measured (some cities do not have enough data in the meteostat system so proxies were used, such as NE and GB).
11. Field - What kind of grass or turf is in the stadium **Manually Added from Wiki
12. Roof - Fixed, open, or retractable **Manually Added from Wiki
13. added_time - If there was no time point in the meteostat system at each hour, a time point was added into the dataframe then had the data interpolated. This column keeps track of which rows were interpolated.
14. Time (EST) - Time (GMT) -4h

Cleaning and Combining

NFL Play-By-Play:
The first step for cleaning the NFL data was to understand the missing values from all fields in our smaller dataframe. This was done using a null count function to understand what percentage of values in fields were null. Then each field was explored separately to understand what needed to be fixed. Logical nulls, such as pass yards on run plays, were field with nulls, and other fields used fill forward techniques after sorting the records by plays in order. We also had to add actual Game Time to the data so we could combine with weather, so we scraped an NFL stat website for every year (2015-2018) to get Start Time.

Weather Data:

Combining the Two:

Creating Aggregate Datasets:
Since we cared more about analyzing weather effects on wins, we needed to create a couple aggregate flags and datasets to make that analysis easier. The first dataset we created was an end of drive summary, which was done by finding the last play of every drive and summing up metrics from that drive. We also added some team specific columns that identified if they were home or away for a particular game to allow for further aggregation at the team level.

The second aggregate dataset we created was for end of game stats. We used this more heavily, as team stats at end of games could be more easily compared to if that team won or not. This was done using an custom agg function that summed up the variable fields we cared about and groupby/apply to append to an initial end of game summary df. We had to break each team stat up separately since most important variables were bifurcated by Home and Away, but were able to melt the dataframe to give each team their own row for each game they played in.

The Analysis

In [108]:
```
# Agg Wins by Team, 4 years
from plotly.graph_objs import *
fig = go.Figure(data=[go.Bar(name='Total', x = team_by_wins['team'], y =
team_by_wins['wins'])])

layout = go.Layout(
    title = 'Total Wins By Team (2015 - 2018)',
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)
)

fig.add_shape(type="rect",
              x0=-0.5,
              y0=0,
              x1=6.4,
              y1=27,
              line=dict(color="Red",
                        width=3))

fig.add_shape(type="rect",
              x0=6.6,
              y0=0,
              x1=20.4,
              y1=35,
              line=dict(color="Yellow",
                        width=3))
fig.add_shape(type="rect",
              x0=20.6,
              y0=0,
              x1=31.5,
              y1=50,
              line=dict(color="Green",
                        width=3))


fig.update_layout(
    title={
        'text':'<b>'+'Total Wins by Team'+'</b>'+'<br>(2015 thru 2018)',
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
    annotations=[
        dict(x=3, y=30, text="Worse than Average Teams", xref="x", yref=
"y", showarrow=False),
        dict(x=13.5, y=38, text="Average Teams", xref="x", yref="y", sho
warrow=False),
        dict(x=25, y=45, text="Better than Average Teams", xref="x", yre
f="y", showarrow=False),
                ])
fig

#Add year legend?
```

Over the course of the 4 years from 2015-2018, there are some clear better than average, middle of the road, and worse than average teams.

In [224]:
```
# Team wins by year
team_wins_2015 = []
teams = list(team_by_wins['team'].unique())
for team in teams:
    home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_drive['ye
ar'] == 2015) &\
                                            (nfl_small2_end_of_drive[tea
m] == 'H')
                                           ]['home_team_win'])

    away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                                (nfl_small2_end_of_driv
e['year'] == 2015) &\
                                                (nfl_small2_end_of_driv
e[team] == 'A')
                                           ]['away_team_win'])

    all_wins = home_wins + away_wins

    team_wins_2015.append(round(all_wins,0))


team_wins_2016 = []
for team in teams:
    home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_drive['ye
ar'] == 2016) &\
                                            (nfl_small2_end_of_drive[tea
m] == 'H')
                                           ]['home_team_win'])

    away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                                (nfl_small2_end_of_driv
e['year'] == 2016) &\
                                                (nfl_small2_end_of_driv
e[team] == 'A')
                                           ]['away_team_win'])

    all_wins = home_wins + away_wins

    team_wins_2016.append(round(all_wins,0))

team_wins_2017 = []
for team in teams:
    home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_drive['ye
ar'] == 2017) &\
                                            (nfl_small2_end_of_drive[tea
m] == 'H')
                                           ]['home_team_win'])
```

```
    away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_driv
e['year'] == 2017) &\
                                            (nfl_small2_end_of_driv
e[team] == 'A')
                                           ]['away_team_win'])

    all_wins = home_wins + away_wins

    team_wins_2017.append(round(all_wins,0))

team_wins_2018 = []
for team in teams:
    home_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_drive['ye
ar'] == 2018) &\
                                            (nfl_small2_end_of_drive[tea
m] == 'H')
                                           ]['home_team_win'])

    away_wins = sum(nfl_small2_end_of_drive[(nfl_small2_end_of_drive['en
d_of_game'] == 1) &\
                                            (nfl_small2_end_of_driv
e['year'] == 2018) &\
                                            (nfl_small2_end_of_driv
e[team] == 'A')
                                           ]['away_team_win'])

    all_wins = (home_wins + away_wins)*16/14

    team_wins_2018.append(round(all_wins,0))

# Chart of Total Wins by Year
fig = go.Figure(data=[go.Bar(name='2015', x = teams, y = team_wins_2015
),\
                      go.Bar(name='2016', x = teams, y = team_wins_2016
),\
                      go.Bar(name='2017', x = teams, y = team_wins_2017
),\
                      go.Bar(name='2018', x = teams, y = team_wins_2018
)])

layout = go.Layout(
    title = 'Total Wins By Year (2015 - 2018)',
    font=dict(size=8),
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)

)

fig.update_layout(
    title={
        'text':'<b>'+'Total Wins by Year'+'</b>'+'<br>(2015 thru 2018)',
        'y':0.9,
```

```
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
        barmode='group',width=500, height=550,font=dict(size=8),)
fig
```

In [110]:
```
# Diving into LA Rams
x = list(nfl_end_of_game[(nfl_end_of_game['Team']=='LA')& (nfl_end_of_ga
me['win']==1)].groupby('year')['win'].count().index.astype('str'))
y = nfl_end_of_game[(nfl_end_of_game['Team']=='LA')& (nfl_end_of_game['w
in']==1)].groupby('year')['win'].count().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Total', x = x, y = y))

layout = go.Layout(
    title = 'Total Wins for LA (2015 - 2018)',
    xaxis= dict(title= 'Year',dtick=1,ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)
)

fig.update_layout(
    title={
        'text':'<b>'+'Total Wins for LA (2015 - 2018)'+'</b>'+'<br>(2015
thru 2018)',
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},xaxis_type='category',width=500, height=550,
    annotations=[
        dict(x=2, y=10, text="Sean McVay joined the Rams coaching staff"
, xref="x", yref="y", showarrow=True, ay=-50, arrowhead=3, arrowwidth=3
)],
    font=dict(size=8)
)

# fig.add_annotation(text='Sean McVay',xref='x1', yref='y1',x=3,y=10,sec
ondary_y=True)

fig
```

We also see that while there are consistently good and consistently bad teams, most remain fairly competitive and win/loss totals can shift drastically year to year. For that reason, we need to bucket teams based off yearly performance and cannot compare results for one team year over year.

In [111]:
```
# Grouping Teams into win buckets by year
labels = ['5 or Less', 'Between 6 and 9','10+ Wins']
team_by_wins['binned'] = pd.cut(team_by_wins['wins'], bins=3, labels=lab
els)


x2015 = team_by_wins[team_by_wins['year']==2015].groupby('binned').count
().index
y2015 = team_by_wins[team_by_wins['year']==2015].groupby('binned').count
()['team'].values

x2016 = team_by_wins[team_by_wins['year']==2016].groupby('binned').count
().index
y2016 = team_by_wins[team_by_wins['year']==2016].groupby('binned').count
()['team'].values

x2017 = team_by_wins[team_by_wins['year']==2017].groupby('binned').count
().index
y2017 = team_by_wins[team_by_wins['year']==2017].groupby('binned').count
()['team'].values

x2018 = team_by_wins[team_by_wins['year']==2018].groupby('binned').count
().index
y2018 = team_by_wins[team_by_wins['year']==2018].groupby('binned').count
()['team'].values

fig = go.Figure(data=[go.Bar(name='2015', x = x2015, y = y2015),\
                      go.Bar(name='2016', x = x2016, y = y2016),\
                      go.Bar(name='2017', x = x2017, y = y2017),\
                      go.Bar(name='2018', x = x2018, y = y2018)])

layout = go.Layout(
    title = 'Teams by Win Bucket',
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)

)

fig.update_layout(
    title={
        'text':'<b>'+'Teams by Win Bucket'+'</b>'+'<br>(2015 thru 2018)'
,
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
        barmode='group',width=1000, height=600)
fig
```

As expected, year over year, we have very consistent groups across teams from low performers to high. We will use these to genericize actual teams into performance groups. Then we can compare across years more easily, given the limited data we have.

In [112]:
```
# Finding correlation between some key variables and wins

corr_cols = ['yards','yards_against', 'run_plays', 'run_plays_against',
'pass_plays',\
       'pass_plays_against','yard_diff','to_for', 'to_against', 'to_dif
f','fg_rate',\
        'fg_rate_against','pass_yds_per_at','pass_run_ratio','run_yds_pe
r_at',
             'pass_yds_per_at_against', 'run_yds_per_at_against']

corr_df = nfl_end_of_game[corr_cols].apply(lambda x: x.corr(nfl_end_of_g
ame.win)).sort_values(kind="quicksort")

colors = ['lightslategray',] * 17
colors[0] = 'crimson'
colors[16] = 'crimson'
colors[9] = 'crimson'
colors[13] = 'crimson'
colors[12] = 'crimson'

fig = go.Figure(data=[go.Bar(name='Corr', x = corr_df.index, y = corr_df
.values,marker_color=colors)])

layout = go.Layout(
    title = 'Variables by Win Correlation',
    xaxis= dict(title= 'Variable',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Correlation',ticklen= 5,zeroline= False)
)

fig.update_layout(
    title={
        'text':'<b>'+'Variables by Win Correlation'+'</b>'+'<br>(Based o
n Individual Game Results)',
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})
fig
```

Looking at correlation to wins for certain variables, we can see some clear variables we would love to assess weather impacts on. The variables below were chosen due to some of the others being directly correlated. For instance, to_diff is just to_against minus to_for, so both of those are covered by to_diff.

Variables we will explore further are:

1. **pass_run_ratio** = Passes / Runs per Game
2. **to_diff** = Turnover Differential
3. **fg_rate** = Field Goal Success Rate (doesn't seem highly correlated, but could be more greatly affected by weather)
4. **yard_diff** = Total Yards minus Total Yards by Opponent
5. **pass_yards_per_at** = Pass Yards per Attempt

In [113]:
```
# Create categorical variable and order for binned
nfl_end_of_game['binned'] = pd.Categorical(nfl_end_of_game['binned'], \
                                           categories=['5 or Less','Between
6 and 9','10+ Wins'],\
                                           ordered=True)

fig = go.Figure()

fig.add_trace(go.Bar(name='pass to run',x=nfl_end_of_game.groupby('binne
d').agg({'pass_run_ratio':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'pass_run_
ratio':'mean'})['pass_run_ratio'])
              )

fig.update_layout(
    title={
        'text':'<b>'+'Pass to Run Ratio by Team Wins'+'</b>'+'<br>(Pass
 Plays / Run Plays)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})

fig
```

Unsurprisingly, the better teams tend to be more balanced offensively and split their plays more between passes and runs. I wouldn't be surprised if this also manifests itself into more wins during non-ideal weather conditions.

In [114]:
```
# TO Margin
fig = go.Figure()

fig.add_trace(go.Bar(name='TO Margin',x=nfl_end_of_game.groupby('binned'
).agg({'to_diff':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'to_diff':
'mean'})['to_diff'])
              )

fig.update_layout(
    title={
        'text':'<b>'+'Turnover Margin by Team Wins'+'</b>'+'<br>(Giveawa
ys Minus Takeways)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})

fig
```

Also, not surprising, better teams tend to hold onto the ball more as well. As a casual football fan, I know the team that has a higher turnover differential has a better shot at winning the game.

In [115]:
```
nfl_end_of_game.head(3)
```

Out[115]:

	game_id	Team	team_against	score	score_against	yards	yards_against	run_plays	run_p
0	2015091000	NE	PIT	28	21	366.0	429.0	23.0	
1	2015091300	CHI	GB	23	31	416.0	336.0	33.0	
2	2015091301	LA	SEA	33	29	354.0	338.0	26.0	

In [116]:
```
# Field Goal Rate
fig = go.Figure()

fig.add_trace(go.Bar(name='FG Rate',x=nfl_end_of_game.groupby('binned').
agg({'fg_rate':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'fg_rate':
'mean'})['fg_rate'])
             )

fig.update_layout(
    title={
        'text':'<b>'+'Field Goal Success Rate'+'</b>'+'<br>(Field Goals
 Made versus Attempted)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})

fig
```

Every Win Group performs very well as it relates to making Field Goals. The slight edge for better teams may manifest itself in an extra win here and there.

In [117]:
```
# Yard Differential
from plotly.subplots import make_subplots
fig = make_subplots(specs=[[{"secondary_y": True}]])

fig.add_trace(go.Bar(name='Yard Diff',x=nfl_end_of_game.groupby('binned'
).agg({'yard_diff':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'yard_dif
f':'mean'})['yard_diff']),
                    secondary_y=False
              )

fig.add_trace(go.Scatter(name='Yards Gained',x=nfl_end_of_game.groupby(
'binned').agg({'yards':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'yards':'m
ean'})['yards'],yaxis='y2'),
                        secondary_y=True
              )

fig.add_trace(go.Scatter(name='Yards Given Up',x=nfl_end_of_game.groupby
('binned').agg({'yards_against':'mean'}).index,
                     y=nfl_end_of_game.groupby('binned').agg({'yards_aga
inst':'mean'})['yards_against']*-1,yaxis='y2'),
                        secondary_y=True
              )

fig.update_layout(
    title={
        'text':'<b>'+'Game Average Yard Differential'+'</b>'+'<br>(Yards
Gained Minus Yard Given Up)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'}

    )

fig
```

Yard differential is also correlate well with Win Group, but these low differences may not play a huge factor given teams gain on average XXX yards.

Exploring Effects of Weather

We pulled weather data using an API on Meteostat for locations of all NFL games in our dataset. Locations and game timing were taken into account when joining with play-by-play data. **I did not do this** For instance, when the Colts played the Jaguars in London on October 2nd 2016, weather was pulled for London during the time the game was happening **I did not do this**. Since weather data is on hour increments, we assumed the same weather lasted an entire hour.

Other Assumptions:

1. Games played in a dome were considered "ideal conditions" and assumed weather has zero effect
2. TBD

Looking for Trends Broadly

In [118]:
```
# Comparing Temperature to NFL Stat Variables

# Setting win column to string
nfl_df['win'] = nfl_df['win'].astype('str')

#Adding color scale for win versus loss
cols = nfl_df['win'].map({'1.0': 'rgb(0, 97, 252)', '0.0': 'rgb(252, 55,
0)'})

fig = make_subplots(rows=5, cols=3,
                    subplot_titles = ('Temperature', 'Wind', 'Precipitati
on'))

# Temp plots
fig.append_trace(go.Scatter(name="Pass/Run Ratio",x=nfl_df["Temperature
 (°C)"], y=nfl_df["pass_run_ratio"], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=1,col=1)

fig.append_trace(go.Scatter(name="Turnover Difference",x=nfl_df["Tempera
ture (°C)"], y=nfl_df["to_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=2,col=1)

fig.append_trace(go.Scatter(name="Field Goal Rate",x=nfl_df["Temperature
(°C)"], y=nfl_df["fg_rate"], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=3,col=1)

fig.append_trace(go.Scatter(name="Yard Difference",x=nfl_df["Temperature
(°C)"], y=nfl_df["yard_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=4,col=1)

fig.append_trace(go.Scatter(name="Pass Yards Per Attempt",x=nfl_df["Temp
erature (°C)"], y=nfl_df['pass_yds_per_at'], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=5,col=1)

fig.update_xaxes(title_text="Temperature (°C)", row=1, col=1)
fig.update_xaxes(title_text="Temperature (°C)", row=2, col=1)
fig.update_xaxes(title_text="Temperature (°C)", row=3, col=1)
fig.update_xaxes(title_text="Temperature (°C)", row=4, col=1)
fig.update_xaxes(title_text="Temperature (°C)", row=5, col=1)

fig.update_yaxes(title_text="Pass/Run Ratio", row=1, col=1)
fig.update_yaxes(title_text="Turnover Difference", row=2, col=1)
fig.update_yaxes(title_text="Field Goal Rate", row=3, col=1)
fig.update_yaxes(title_text="Yard Difference", row=4, col=1)
fig.update_yaxes(title_text="Pass Yards Per Attempt", row=5, col=1)


# Wind plots
fig.append_trace(go.Scatter(name="Pass/Run Ratio",x=nfl_df["Wind Speed
 (km/h)"], y=nfl_df["pass_run_ratio"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
```

```
w=1,col=2)

fig.append_trace(go.Scatter(name="Turnover Difference",x=nfl_df["Wind Sp
eed (km/h)"], y=nfl_df["to_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=2,col=2)

fig.append_trace(go.Scatter(name="Field Goal Rate",x=nfl_df["Wind Speed
 (km/h)"], y=nfl_df["fg_rate"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=3,col=2)

fig.append_trace(go.Scatter(name="Yard Difference",x=nfl_df["Wind Speed
 (km/h)"], y=nfl_df["yard_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=4,col=2)

fig.append_trace(go.Scatter(name="Pass Yards Per Attempt",x=nfl_df["Wind
Speed (km/h)"], y=nfl_df['pass_yds_per_at'], mode="markers",
                            marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=5,col=2)

fig.update_xaxes(title_text="Wind Speed (km/h)", row=1, col=2)
fig.update_xaxes(title_text="Wind Speed (km/h)", row=2, col=2)
fig.update_xaxes(title_text="Wind Speed (km/h)", row=3, col=2)
fig.update_xaxes(title_text="Wind Speed (km/h)", row=4, col=2)
fig.update_xaxes(title_text="Wind Speed (km/h)", row=5, col=2)

# fig.update_yaxes(title_text="Pass/Run Ratio", row=1, col=2)
# fig.update_yaxes(title_text="Turnover Difference", row=2, col=2)
# fig.update_yaxes(title_text="Field Goal Rate", row=3, col=2)
# fig.update_yaxes(title_text="Yard Difference", row=4, col=2)


# Precip plots
fig.append_trace(go.Scatter(name="Pass/Run Ratio",x=nfl_df["Precipitatio
n (mm)"], y=nfl_df["pass_run_ratio"], mode="markers",
                            marker=dict(color=cols, showscale=False),show
legend=True), row=1,col=3)

fig.append_trace(go.Scatter(name="Turnover Difference",x=nfl_df["Precipi
tation (mm)"], y=nfl_df["to_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=2,col=3)

fig.append_trace(go.Scatter(name="Field Goal Rate",x=nfl_df["Precipitati
on (mm)"], y=nfl_df["fg_rate"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=3,col=3)

fig.append_trace(go.Scatter(name="Yard Difference",x=nfl_df["Precipitati
on (mm)"], y=nfl_df["yard_diff"], mode="markers",
                            marker=dict(color=cols, showscale=False)), ro
w=4,col=3)

fig.append_trace(go.Scatter(name="Pass Yards Per Attempt",x=nfl_df["Prec
ipitation (mm)"], y=nfl_df['pass_yds_per_at'], mode="markers",
```

```
                              marker=dict(color=cols, showscale=False),lege
ndgroup='Wins'), row=5,col=3)

fig.update_xaxes(title_text="Precipitation (mm)", row=1, col=3)
fig.update_xaxes(title_text="Precipitation (mm)", row=2, col=3)
fig.update_xaxes(title_text="Precipitation (mm)", row=3, col=3)
fig.update_xaxes(title_text="Precipitation (mm)", row=4, col=3)
fig.update_xaxes(title_text="Precipitation (mm)", row=5, col=3)

# fig.update_yaxes(title_text="Pass/Run Ratio", row=1, col=3)
# fig.update_yaxes(title_text="Turnover Difference", row=2, col=3)
# fig.update_yaxes(title_text="Field Goal Rate", row=3, col=3)
# fig.update_yaxes(title_text="Yard Difference", row=4, col=3)


fig.update_layout(title= {'text':'<b>'+'Weather Effects on Football Gam
e'+'</b>',
                              'y':.95,
                              'x':0.5,
                              'xanchor': 'center',
                              'yanchor': 'top'}, height=1000, showlegend=F
alse)

fig['layout'].update(annotations=[
    dict(x=23.70357, y=10, xref='x1', yref='y1', text='Loss', showarrow=
True,
         arrowhead=7,
         ax=10, ay=-20, arrowcolor='rgb(252, 55, 0)'),
    dict(x=15.66793, y=6.33, xref='x1', yref='y1', text='Win', showarrow
=True,
         arrowhead=7,
         ax=-10, ay=-40, arrowcolor='rgb(0, 97, 252)')])

fig.show()
```

Inpsecting our mix of scatterplots above, we don't see any obvious trends related to the effects weather has on different game stats. This is partially because our weather only affects a small portion of games.

Pass/Run Ratio does seem to be somewhat correlated with Precipitation and is observed to decrease as the amount of rain increases. This intuitively makes sense, given a wet ball would be harder to throw and catch. This may drive a change to the outcome of the game due to better teams generally doing a better job at balancing running and passing. Given the limited amount of games played in the rain in this dataset, it is difficult to make this statement without much weight behind it.

Wind Speed may also have an effect on Field Goal Success Rate, given the scatter seems to "drag" further down the x axis (high wind speeds). We need to explore this further.

Defining Broader Weather Variable Categories

General trends are harder to visualize, so we create aggregate categories for each weather variable. The bins were chosen based on multiple conditions.

Explanation of Temperature: The ideal temperature binning was fit to approximate 'hot', 'cold', and 'nice' weather, which yields a good binning system, given that the football season occurs at the end of the year when there should be less 'hot' weather.

Explanation of Wind Speed: The binning was chosen based with an attempt to equally bin the wind. According to the Beaufort scale, anything above 15 km/h is above a gentle breeze, which dictates a pretty strong wind which was classified as the stronger wind force. Anything above this region would change the mechanics of throwing a football, or kicking a field goal.

Precipitation: For rain it was quite simple. There was such little data with lots of rain, so it was binned at 0.5 mm/h which is characterized by the USGS as between a light rain and a drizzle, which may seem low, however was binned this way in order to allow for some data to be taken for 'raining', and the person who wrote this section is from sunny Southern California and thinks this is a lot of water.

In [119]:
```
# Weather Variable Histograms
fig = make_subplots(rows=3, cols=1)
trace0 = go.Histogram(x=nfl_df["Precipitation (mm)"], histnorm='probabil
ity')
trace1 = go.Histogram(x=nfl_df["Temperature (°C)"], histnorm='probabilit
y')
trace2 = go.Histogram(x=nfl_df["Wind Speed (km/h)"], histnorm='probabili
ty')
fig.append_trace(trace0, 1,1)
fig.append_trace(trace1, 2,1)
fig.append_trace(trace2, 3,1)
fig.update_xaxes(title_text="Precipitation (mm)", row=1, col=1)
fig.update_xaxes(title_text="Temperature (°C)", row=2, col=1)
fig.update_xaxes(title_text="Wind Speed (km/h)", row=3, col=1)
fig.update_layout(title={
        'text':'<b>'+'Histogram of Weather Data'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'}, font=dict(size=8),autosize=False, height=600,
width=500, showlegend=False)
fig.show()
```

In [120]:
```
# Weather Bin Histograms
fig = make_subplots(rows=3, cols=1)

trace0 = go.Histogram(x=nfl_df["rain_bin"], histnorm='probability')
trace1 = go.Histogram(x=nfl_df["temp_bin"], histnorm='probability')
trace2 = go.Histogram(x=nfl_df["wind_bin"], histnorm='probability')

fig.append_trace(trace0, 1,1)
fig.append_trace(trace1, 2,1)
fig.append_trace(trace2, 3,1)

fig.update_xaxes(title_text="Rain bins", row=1, col=1)
fig.update_xaxes(title_text="Temperature bins", row=2, col=1)
fig.update_xaxes(title_text="Wind Speed bins", row=3, col=1)

fig.update_layout(title={
        'text':'<b>'+'Binning of Weather Data'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'}, font=dict(size=8),autosize=False, height=600,
width=500, showlegend=False)

fig.show()
```

Looking for Trends Locally

In this next section, we will dive into some specific variables and see if weather impacts are felt enough to potentially change the outcome of a game.

Analysis on Run to Pass Ratio

In [25]:
```
# Pass/Run Ratio by Precipitation Bucket
x = nfl_df.groupby('rain_bin')['pass_run_ratio'].mean().index
y = nfl_df.groupby('rain_bin')['pass_run_ratio'].mean().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Pass/Run Ratio',x=x, y=y)
              )

fig.update_layout(
    title={
        'text':'<b>'+'Affects of Rain on Pass/Run Ratio'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,width=550,hei
ght=400)
fig.update_xaxes(title_text="Amount of Rain")
fig.update_yaxes(title_text="Pass/Run Ratio")

fig.update_yaxes(range=[1, 1.7])

fig
```

In [26]:
```
#Pass and Run Play Average

x = nfl_df.groupby('rain_bin')['pass_plays'].mean().index
y = nfl_df.groupby('rain_bin')['pass_plays'].mean().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Average Pass Plays',x=x, y=y)
              )


x = nfl_df.groupby('rain_bin')['run_plays'].mean().index
y = nfl_df.groupby('rain_bin')['run_plays'].mean().values

fig.add_trace(go.Bar(name='Average Run Plays',x=x, y=y)
              )


fig.update_layout(
    title={
        'text':'<b>'+'Affects of Rain on Number of Pass/Run Plays per Ga
me'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,width=550,hei
ght=400,
        legend=dict(
            x=.25,
            y=.95,
            traceorder='normal',
            font=dict(size=7), bgcolor='rgba(0,0,0,0)'
        )
)

fig.update_xaxes(title_text="Amount of Rain")
fig.update_yaxes(title_text="Average number of plays per game")

fig.update_yaxes(range=[0, 40])

fig
```

While small, we do see some effects from rain to our Pass/Run ratio. Our results are showing that the ratio will generally decrease (meaning more runs or less passes) as the gets worse.

The effects appear to be driven by a decrease in overall pass plays in a game, which makes sense with our assumption that teams pass less as weather deteriorates

In [40]:
```
# Effects of Rain on Win Groups
x1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('rain_bin')['pass_run
_ratio'].mean().index
y1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('rain_bin')['pass_run
_ratio'].mean().values

x2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('rain_bin')['pa
ss_run_ratio'].mean().index
y2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('rain_bin')['pa
ss_run_ratio'].mean().values

x3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('rain_bin')['pass_run_
ratio'].mean().index
y3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('rain_bin')['pass_run_
ratio'].mean().values

fig = go.Figure(data=[go.Bar(name='5 or Less', x = x1, y = y1),\
                      go.Bar(name='Between 6 and 9', x = x2, y = y2),\
                      go.Bar(name='10+ Wins', x = x3, y = y3)]
                )

layout = go.Layout(
    title = 'Pass to Run Ratio by Win Bucket',
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)

)

fig.update_layout(
    title={
        'text':'<b>'+'Pass to Run Ratio by Win Bucket'+'</b>'+'<br>',
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},
        barmode='group',width=1000, height=600)
fig
```

Since better teams (10+ Wins) tend to run the ball more consistently, their pass/run ratio remains less effected by rain. Suprisingly, teams who struggle to win see an increase in pass/run ratio, which would most likely correlate to being behind and having to take risks.

Analysis on Turnover Differential (TO_DIFF)

****How do we compare this given they average to zero?****

In [220]:
```
# Rain affects on Turnovers

x1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('rain_bin')['to_again
st'].mean().index
y1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('rain_bin')['to_again
st'].mean().values

x2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('rain_bin')['to
_against'].mean().index
y2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('rain_bin')['to
_against'].mean().values

x3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('rain_bin')['to_agains
t'].mean().index
y3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('rain_bin')['to_agains
t'].mean().values

fig = go.Figure(data=[go.Bar(name='5 or Less', x = x1, y = y1),\
                      go.Bar(name='Between 6 and 9', x = x2, y = y2),\
                      go.Bar(name='10+ Wins', x = x3, y = y3)]
                )

layout = go.Layout(
    title = 'Turnovers by Win Group',
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)

)

fig.update_layout(
    title={
        'text':'<b>'+'Turnovers Against by Rain Group'+'</b>'+'<br>'+'(A
verage Per Game)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        yaxis=dict(range=[0,4.5]),
        barmode='group',width=500, height=550,
        legend=dict(
            x=.7,
            y=1,
            traceorder='normal',
            font=dict(size=7), bgcolor='rgba(0,0,0,0)'
        ))
fig
```

In [221]:
```
# Wind affects on Turnovers

x1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('wind_bin')['to_again
st'].mean().index
y1 = nfl_df[nfl_df['binned']=='5 or Less'].groupby('wind_bin')['to_again
st'].mean().values

x2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('wind_bin')['to
_against'].mean().index
y2 = nfl_df[nfl_df['binned']=='Between 6 and 9'].groupby('wind_bin')['to
_against'].mean().values

x3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('wind_bin')['to_agains
t'].mean().index
y3 = nfl_df[nfl_df['binned']=='10+ Wins'].groupby('wind_bin')['to_agains
t'].mean().values

fig = go.Figure(data=[go.Bar(name='5 or Less', x = x1, y = y1),\
                      go.Bar(name='Between 6 and 9', x = x2, y = y2),\
                      go.Bar(name='10+ Wins', x = x3, y = y3)]
                )

layout = go.Layout(
    title = 'Turnovers by Wind Group',
    xaxis= dict(title= 'Team',ticklen= 1,zeroline= False),
    yaxis= dict(title= 'Wins',ticklen= 5,zeroline= False)

)

fig.update_layout(
    title={
        'text':'<b>'+'Turnovers Against by Wind Group'+'</b>'+'<br>'+'(A
verage Per Game)',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        yaxis=dict(range=[0,4.5]),
        barmode='group',width=500, height=550,
        legend=dict(
            x=.7,
            y=1,
            traceorder='normal',
            font=dict(size=7), bgcolor='rgba(0,0,0,0)'
        ))
fig
```

Better teams tend to capitalize on worse weather by getting their opponents to turn the ball over more often. Wind doesn't appear to play any factor in the turnover battle.

Analysis on Field Goals

In [174]:
```
# FG Rate by Precipitation Bucket
x = nfl_df.groupby('rain_bin')['fg_rate'].mean().index
y = nfl_df.groupby('rain_bin')['fg_rate'].mean().values

from plotly.subplots import make_subplots
fig = make_subplots(specs=[[{"secondary_y": True}]])

fig.add_trace(go.Bar(name='Field Goal Success Rate',x=x, y=y,text=y),
              secondary_y=False
             )

fig.update_traces(texttemplate='%{text:%}', textposition='outside')

x = nfl_df.groupby('rain_bin')['fg_at'].mean().index
y = nfl_df.groupby('rain_bin')['fg_at'].mean().values

fig.add_trace(go.Scatter(name='FG Attempts',x=x, y=y,yaxis='y2'),
                         secondary_y=True
             )

fig.update_layout(
    title={
        'text':'<b>'+'Affects of Rain on Field Goal Success Rate'+'</b>'
,
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        width=500,height=400,
        yaxis=dict(tickformat="%",range=[0,1],showgrid=False,title='Fiel
d Goal Success Rate'),
        yaxis2=dict(range=[0,3],title='Avg FG Attempts'),
        legend=dict(
            x=.55,
            y=1,
            traceorder='normal',
            font=dict(size=7), bgcolor='rgba(0,0,0,0)'
        ))

fig.update_xaxes(title_text="Amount of Rain")

fig
```

In [175]:
```
# FG Rate by Precipitation Bucket
x = nfl_df.groupby('wind_bin')['fg_rate'].mean().index
y = nfl_df.groupby('wind_bin')['fg_rate'].mean().values

from plotly.subplots import make_subplots
fig = make_subplots(specs=[[{"secondary_y": True}]])

fig.add_trace(go.Bar(name='Field Goal Success Rate',x=x, y=y,text=y),
              secondary_y=False
             )

fig.update_traces(texttemplate='%{text:%}', textposition='outside')

x = nfl_df.groupby('wind_bin')['fg_at'].mean().index
y = nfl_df.groupby('wind_bin')['fg_at'].mean().values

fig.add_trace(go.Scatter(name='FG Attempts',x=x, y=y,yaxis='y2'),
                         secondary_y=True
             )

fig.update_layout(
    title={
        'text':'<b>'+'Affects of Wind on Field Goal Success Rate'+'</b>'
,
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        width=500,height=400,
        yaxis=dict(tickformat="%",range=[0,1],showgrid=False,title='Fiel
d Goal Success Rate'),
        yaxis2=dict(range=[0,3],title='Avg FG Attempts'),
        legend=dict(
            x=.55,
            y=1,
            traceorder='normal',
            font=dict(size=7), bgcolor='rgba(0,0,0,0)'
        ))

fig.update_xaxes(title_text="Amount of Wind")

fig
```

In [176]:
```
# Rain affects on Field Goal Distance

x = nfl_df.groupby('wind_bin')['avg_kick_dist'].mean().index
y = nfl_df.groupby('wind_bin')['avg_kick_dist'].mean().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Wind',x=x, y=y)
              )




fig.update_layout(
    title={
        'text':'<b>'+'Affects of Wind on Field Goal Distance'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        width=550,height=400,barmode='group')

fig.update_xaxes(title_text="Amount of Rain")
fig.update_yaxes(title_text="Number of Field Goal Attempts per Game")

# fig.update_yaxes(range=[.5, 1.7])

fig
```

Rain seems to play a larger factor in a teams ability to make field goals than wind. What's also interesting is teams appear to attempt fewer field goals in rainy games. So not only are they more risk averse, but that risk aversion doesn't seem to help them.

Average Yards Per Pass Attempt

In [178]:
```
nfl_df.head(2)
```

Out[178]:

	game_id	Team	team_against	score	score_against	yards	yards_against	run_plays	run_p
0	2015091000	NE	PIT	28	21	366.0	429.0	23.0	
1	2015091300	CHI	GB	23	31	416.0	336.0	33.0	

In [223]:
```
# Rain affects on Pass Yards Attempt

x = nfl_df.groupby('wind_bin')['pass_yds_per_at'].mean().index
y = nfl_df.groupby('wind_bin')['pass_yds_per_at'].mean().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Wind',x=x, y=y)
              )

fig.update_layout(
    title={
        'text':'<b>'+'Affects of Wind on Pass Yards Per Attempt'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        width=550,height=400,barmode='group')

fig.update_xaxes(title_text="Amount of Wind")
fig.update_yaxes(title_text="Average Pass Yard per Attempt")

# fig.update_yaxes(range=[.5, 1.7])

fig
```

In [181]:
```
# Rain affects on Pass Yards Attempt

x = nfl_df.groupby('rain_bin')['pass_yds_per_at'].mean().index
y = nfl_df.groupby('rain_bin')['pass_yds_per_at'].mean().values

fig = go.Figure()

fig.add_trace(go.Bar(name='Wind',x=x, y=y)
              )


fig.update_layout(
    title={
        'text':'<b>'+'Affects of Wind on Pass Yards Per Attempt'+'</b>',
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'},font=dict(size=8),autosize=False,
        width=550,height=400,barmode='group')

fig.update_xaxes(title_text="Amount of Rain")
fig.update_yaxes(title_text="Number of Field Goal Attempts per Game")

# fig.update_yaxes(range=[.5, 1.7])

fig
```

Conclusion

In [ ]: