In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind # (T test - This is a test for the null h
                                  #           identical average (expected)
                                  #            This test assumes that the p
import statistics as stats
import warnings
warnings.filterwarnings("ignore")
from scipy.stats import mannwhitneyu
```

In [2]:
```
file = "C:\\Users\\HP\\Desktop\\Untitled Folder 1\\passes.csv"

data = pd.read_csv(file,delimiter = ";")
```

In [3]:
```
# Display the data in dataframe format.
data
```

Out[3]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **0** | 11 | 72.0 | No |
| **1** | 11 | 91.0 | Yes |
| **2** | 12 | 82.0 | Yes |
| **3** | 12 | 86.0 | No |
| **4** | 13 | 82.0 | Yes |
| **...** | ... | ... | ... |
| **301** | 177 | 81.0 | Yes |
| **302** | 178 | 73.0 | No |
| **303** | 178 | 74.0 | No |
| **304** | 179 | 74.0 | Yes |
| **305** | 179 | 89.0 | No |

306 rows × 3 columns

In [ ]:

In [4]:
```
# This displays the first 5 data points of data.
data.head()
```

Out[4]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **0** | 11 | 72.0 | No |
| **1** | 11 | 91.0 | Yes |
| **2** | 12 | 82.0 | Yes |
| **3** | 12 | 86.0 | No |
| **4** | 13 | 82.0 | Yes |

In [5]:
```
# This displays the last 5 data points of data.
data.tail()
```

Out[5]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **301** | 177 | 81.0 | Yes |
| **302** | 178 | 73.0 | No |
| **303** | 178 | 74.0 | No |
| **304** | 179 | 74.0 | Yes |
| **305** | 179 | 89.0 | No |

In [6]:
```
# Here I converts the variable name of dataset.
df = data
```

In [7]:
```
df
```

Out[7]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **0** | 11 | 72.0 | No |
| **1** | 11 | 91.0 | Yes |
| **2** | 12 | 82.0 | Yes |
| **3** | 12 | 86.0 | No |
| **4** | 13 | 82.0 | Yes |
| **...** | ... | ... | ... |
| **301** | 177 | 81.0 | Yes |
| **302** | 178 | 73.0 | No |
| **303** | 178 | 74.0 | No |
| **304** | 179 | 74.0 | Yes |
| **305** | 179 | 89.0 | No |

306 rows × 3 columns

In [8]:
```
df["winner"].isin(["draw"]).count
```

Out[8]:
```
<bound method Series.count of 0      False
1      False
2      False
3      False
4      False
       ...
301    False
302    False
303    False
304    False
305    False
Name: winner, Length: 306, dtype: bool>
```

In [9]:
```
# Describe the data into rows and columns.
df.describe()
```

Out[9]:

| | game_id | passing_quote |
|---|---|---|
| **count** | 306.000000 | 304.000000 |
| **mean** | 95.000000 | 79.680921 |
| **std** | 49.138146 | 6.960058 |
| **min** | 11.000000 | 53.000000 |
| **25%** | 53.000000 | 75.000000 |
| **50%** | 95.000000 | 80.000000 |
| **75%** | 137.000000 | 85.000000 |
| **max** | 179.000000 | 92.000000 |

In [10]:
```
# Get summary of data.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 306 entries, 0 to 305
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   game_id        306 non-null    int64
 1   passing_quote  304 non-null    float64
 2   winner         304 non-null    object
dtypes: float64(1), int64(1), object(1)
memory usage: 7.3+ KB
```

In [11]:
```
# Separate data into two groups based on winners & loosers.

winners = df[df["winner"]=="Yes"]
loosers = df[df["winner"]=="No"]
draw = df[df["winner"]=="Draw"]
```

In [12]:
```
winners
```

Out[12]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **1** | 11 | 91.0 | Yes |
| **2** | 12 | 82.0 | Yes |
| **4** | 13 | 82.0 | Yes |
| **7** | 14 | 77.0 | Yes |
| **10** | 16 | 87.0 | Yes |
| **...** | ... | ... | ... |
| **288** | 171 | 90.0 | Yes |
| **291** | 172 | 77.0 | Yes |
| **299** | 176 | 91.0 | Yes |
| **301** | 177 | 81.0 | Yes |
| **304** | 179 | 74.0 | Yes |

114 rows × 3 columns

In [13]:
```
# total count of winners.
winners.count()

# total numbers of winners in 114 out of 304.
```

Out[13]:
```
game_id          114
passing_quote    114
winner           114
dtype: int64
```

In [14]:
```
loosers
```

Out[14]:

| | game_id | passing_quote | winner |
|---|---|---|---|
| **0** | 11 | 72.0 | No |
| **3** | 12 | 86.0 | No |
| **5** | 13 | 79.0 | No |
| **6** | 14 | 79.0 | No |
| **8** | 15 | 85.0 | No |
| **...** | ... | ... | ... |
| **298** | 176 | 76.0 | No |
| **300** | 177 | 78.0 | No |
| **302** | 178 | 73.0 | No |
| **303** | 178 | 74.0 | No |
| **305** | 179 | 89.0 | No |

190 rows × 3 columns

In [18]:
```
loosers.count()

# total number of loosers are 190 out of 304.
```

Out[18]:
```
game_id          190
passing_quote    190
winner           190
dtype: int64
```

## There are draw matches data shown.

In [19]:
```
draw
```

Out[19]:

| game_id | passing_quote | winner |
|---|---|---|

## That means total number of winners are less than total number of loosers as per giving dataset.

In [ ]:
```

```

In [20]:
```
# It combines the data of winners and loosers passeing_quote with the help

games = df.groupby("game_id")["passing_quote"].agg(list).reset_index()
```

In [21]:
```
games
```

Out[21]:

| | game_id | passing_quote |
|---|---|---|
| **0** | 11 | [72.0, 91.0] |
| **1** | 12 | [82.0, 86.0] |
| **2** | 13 | [82.0, 79.0] |
| **3** | 14 | [79.0, 77.0] |
| **4** | 15 | [85.0, 77.0] |
| **...** | ... | ... |
| **148** | 175 | [84.0, 76.0] |
| **149** | 176 | [76.0, 91.0] |
| **150** | 177 | [78.0, 81.0] |
| **151** | 178 | [73.0, 74.0] |
| **152** | 179 | [74.0, 89.0] |

153 rows × 2 columns

In [22]:
```
games.columns = ["game_id","passing_quotes"]
```

In [23]:
```
games.columns
```

Out[23]:
```
Index(['game_id', 'passing_quotes'], dtype='object')
```

In [24]:
```
df.game_id
```

Out[24]:
```
0       11
1       11
2       12
3       12
4       13
      ...
301    177
302    178
303    178
304    179
305    179
Name: game_id, Length: 306, dtype: int64
```

In [25]:
```
df.passing_quote
```

Out[25]:
```
0      72.0
1      91.0
2      82.0
3      86.0
4      82.0
       ...
301    81.0
302    73.0
303    74.0
304    74.0
305    89.0
Name: passing_quote, Length: 306, dtype: float64
```

In [ ]:
```

```

In [26]:
```
# Calculate the Difference between passing rates for each game.
games["diff"] = games["passing_quotes"].apply(lambda x: abs(x[0] - x[1]) if
```

In [27]:
```
# Calculated Difference of passing rate.
games["diff"]
```

Out[27]:
```
0      19.0
1       4.0
2       3.0
3       2.0
4       8.0
       ...
148     8.0
149    15.0
150     3.0
151     1.0
152    15.0
Name: diff, Length: 153, dtype: float64
```

In [28]:
```
games
```

Out[28]:

| | game_id | passing_quotes | diff |
|---|---|---|---|
| **0** | 11 | [72.0, 91.0] | 19.0 |
| **1** | 12 | [82.0, 86.0] | 4.0 |
| **2** | 13 | [82.0, 79.0] | 3.0 |
| **3** | 14 | [79.0, 77.0] | 2.0 |
| **4** | 15 | [85.0, 77.0] | 8.0 |
| **...** | ... | ... | ... |
| **148** | 175 | [84.0, 76.0] | 8.0 |
| **149** | 176 | [76.0, 91.0] | 15.0 |
| **150** | 177 | [78.0, 81.0] | 3.0 |
| **151** | 178 | [73.0, 74.0] | 1.0 |
| **152** | 179 | [74.0, 89.0] | 15.0 |

153 rows × 3 columns

### The difference is calculated with the help of lamda function.

example --- 2-1 = 1

## Compare example with datapoints.

***game_id ---------- passing_rate***

**11 ---------- [72.0, 91.0]**

***91 - 72 = 19***

***So the difference of 11th game passing rate is of 19 passes.***

In [ ]:
```

```

## So now extract passing rate for winners and loosers.

In [29]:
```
win_rate = winners["passing_quote"]
loss_rate = loosers["passing_quote"]
```

In [30]:
```
win_rate
```

Out[30]:
```
1      91.0
2      82.0
4      82.0
7      77.0
10     87.0
       ...
288    90.0
291    77.0
299    91.0
301    81.0
304    74.0
Name: passing_quote, Length: 114, dtype: float64
```

In [31]:
```
win_rate.mean()
```

Out[31]: 81.07894736842105

In [32]:
```
win_rate.median()
```

Out[32]: 83.0

In [33]:
```
win_rate.std()
```

Out[33]: 8.064062895748393

In [34]:
```
wmin = win_rate.min()
```

In [35]:
```
wmax = win_rate.max()
```

In [36]:
```
winrange_ = wmax-wmin
```

In [37]:
```
winrange_
```

Out[37]: 39.0

In [38]:
```
win_rate.count()
```

Out[38]: 114

In [39]:
```
loss_rate.count()
```

Out[39]: 190

In [40]:
```
loss_rate.mean()
```

Out[40]: 78.84210526315789

In [41]:
```
loss_rate.median()
```

Out[41]: 79.0

In [42]:
```
loss_rate.std()
```

Out[42]: 6.074172557969793

In [43]:
```
lmin = loss_rate.min()
```

In [44]:
```
lmax = loss_rate.max()
```

In [45]:
```
lossrange_ = lmax - lmin
```

In [46]:
```
lossrange_
```

Out[46]: 31.0

# Descriptive Statistics

In [47]:
```
win_stats = win_rate.describe()
loss_stats = loss_rate.describe()
diff_stats = games["diff"].describe()
```

## 1. Win Stats

In [48]:
```
win_stats
```

Out[48]:
```
count    114.000000
mean      81.078947
std        8.064063
min       53.000000
25%       76.250000
50%       83.000000
75%       87.000000
max       92.000000
Name: passing_quote, dtype: float64
```

## 2. Loss Stats

In [49]:
```
loss_stats
```

Out[49]:
```
count    190.000000
mean      78.842105
std        6.074173
min       59.000000
25%       75.000000
50%       79.000000
75%       83.000000
max       90.000000
Name: passing_quote, dtype: float64
```

## 3. Stats after Difference

In [50]:
```
diff_stats
```

Out[50]:
```
count    152.000000
mean       8.046053
std        5.796024
min        0.000000
25%        3.000000
50%        7.000000
75%       11.000000
max       29.000000
Name: diff, dtype: float64
```

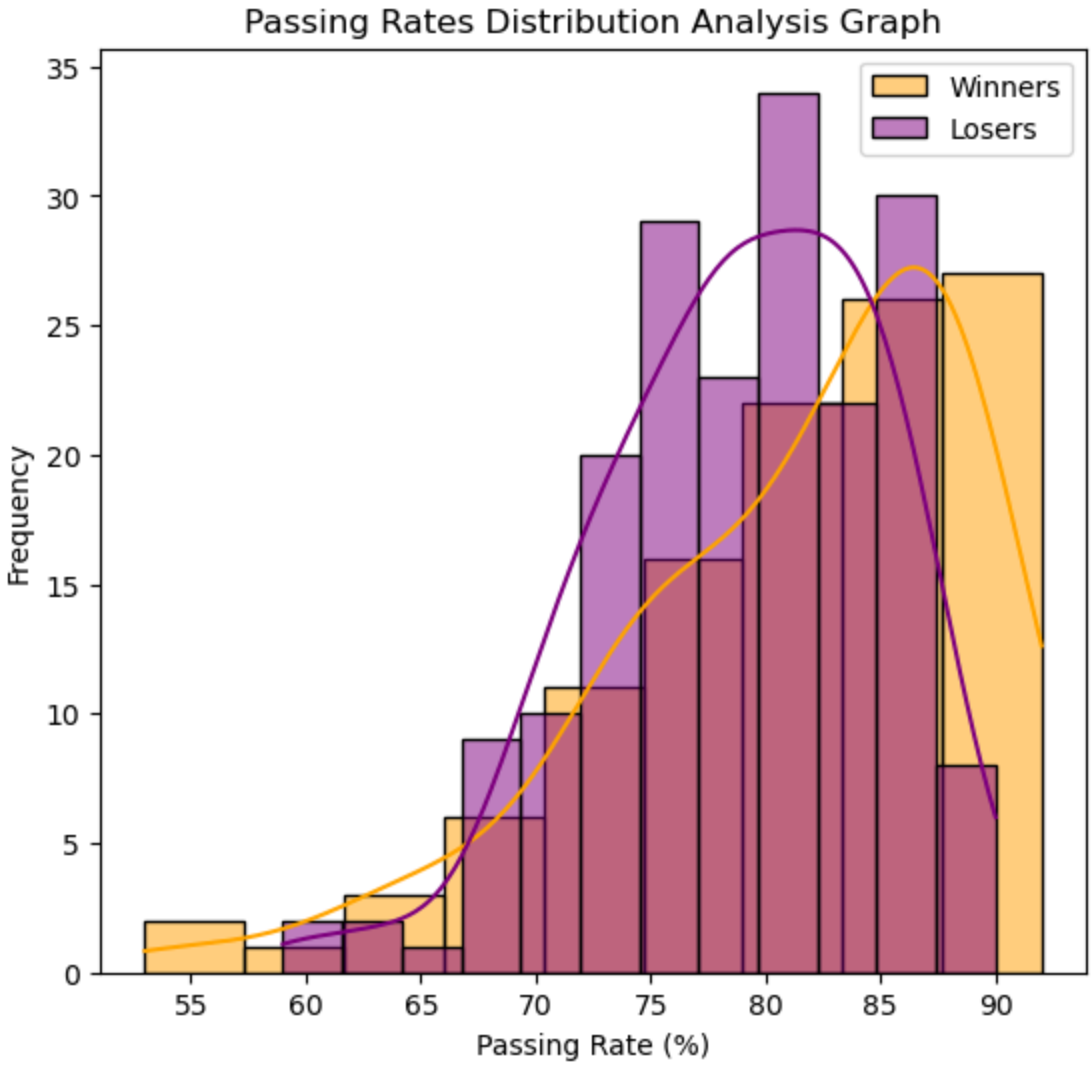
In [ ]:
```

```

# Visualization

## Plot passing rates for Winners and Loosers.

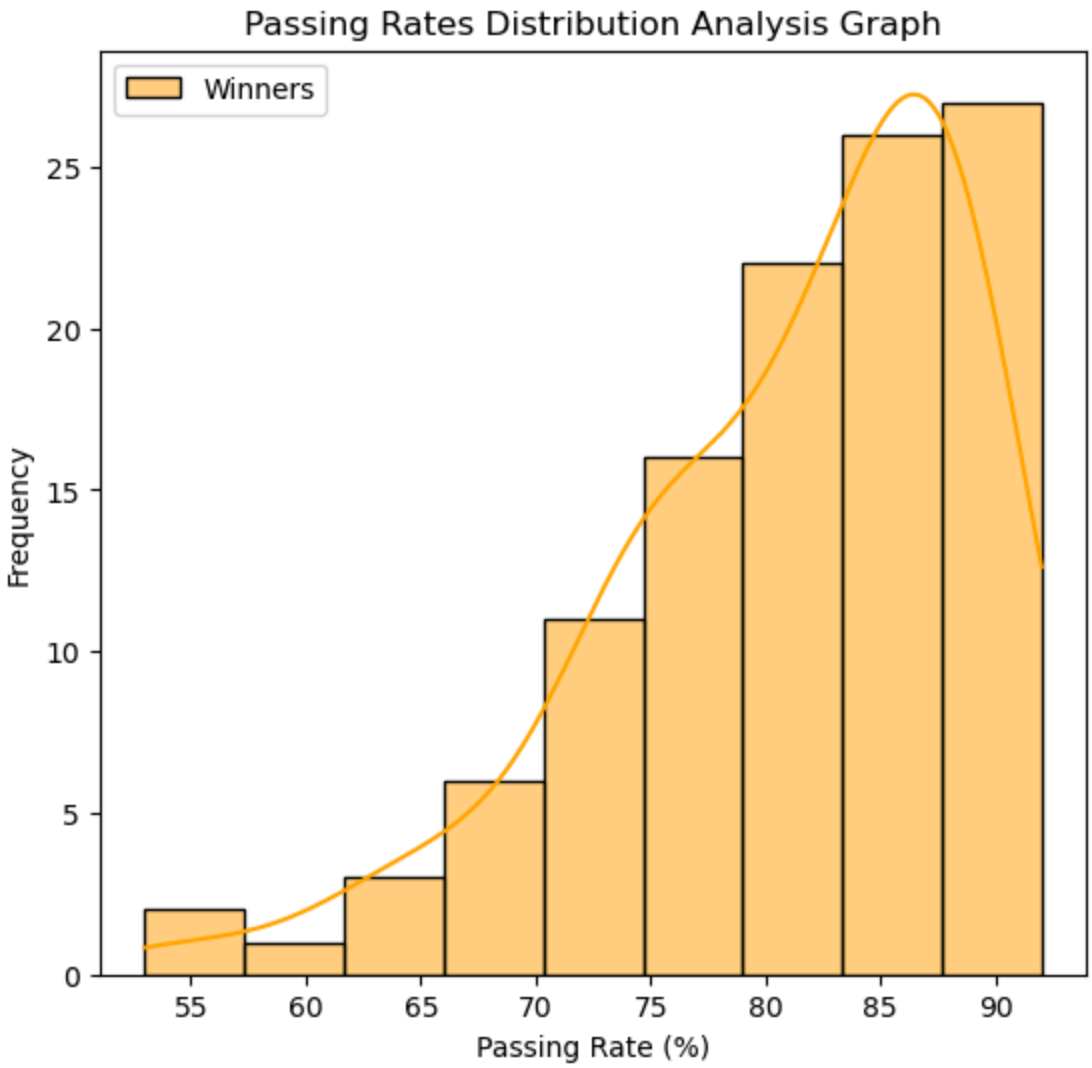
In [60]:
```
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)

sns.histplot(win_rate, kde=True, color='orange', label='Winners')
sns.histplot(loss_rate, kde=True, color='purple', label='Losers')

plt.legend()

plt.title('Passing Rates Distribution Analysis Graph')

plt.xlabel('Passing Rate (%)')
plt.ylabel('Frequency')
```

Out[60]:
```
Text(0, 0.5, 'Frequency')
```



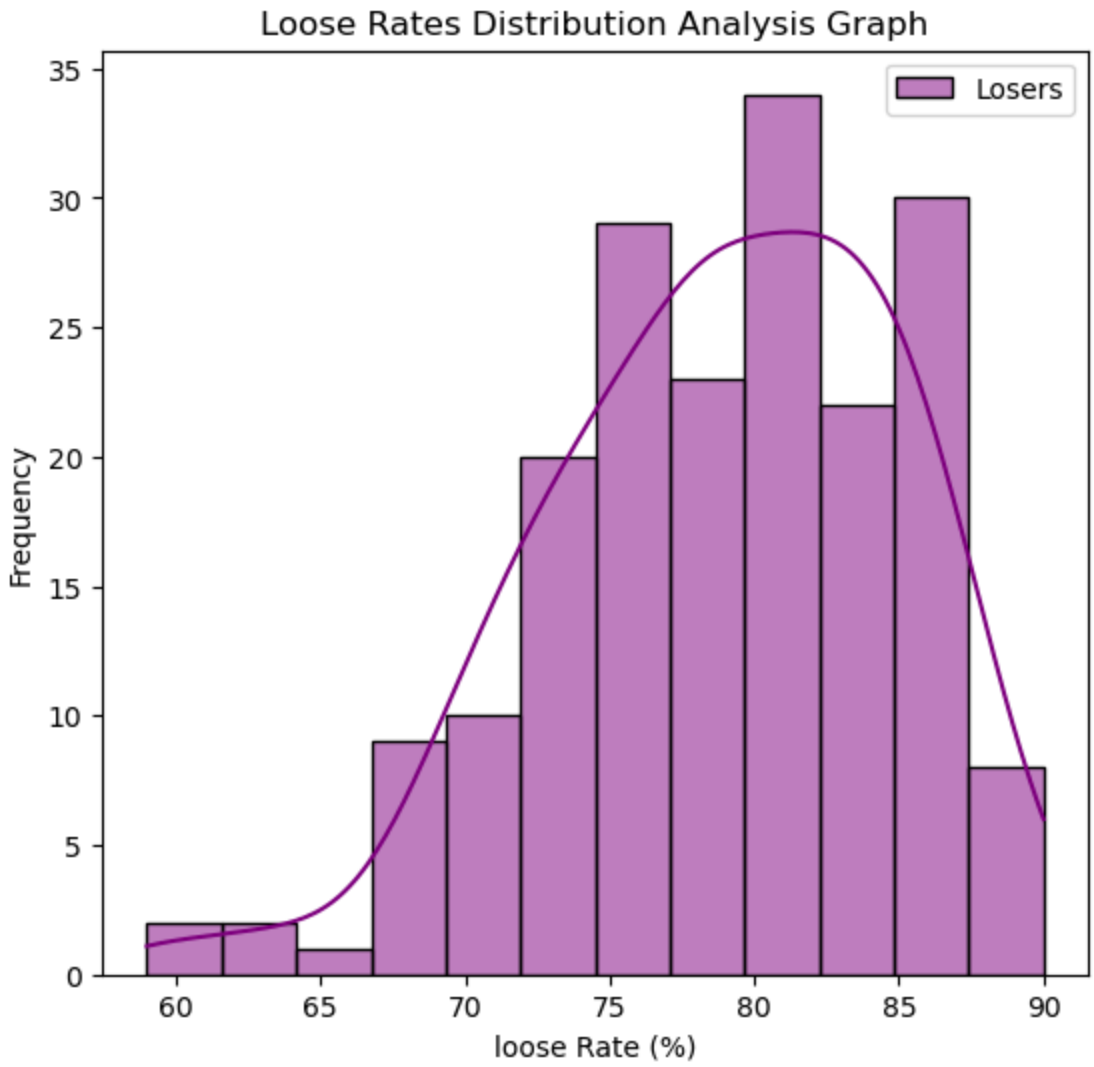
In [52]:
```
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)

sns.histplot(win_rate, kde=True, color='orange', label='Winners')

plt.legend()

plt.title('Passing Rates Distribution Analysis Graph')

plt.xlabel('Passing Rate (%)')
plt.ylabel('Frequency')
```

Out[52]:
```
Text(0, 0.5, 'Frequency')
```



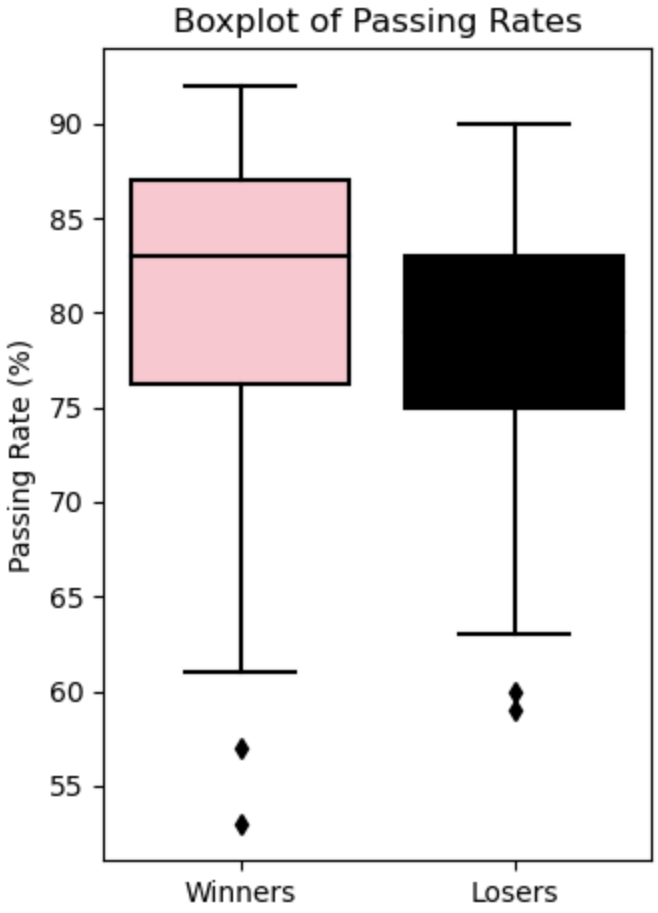
In [53]:
```
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)

sns.histplot(loss_rate, kde=True, color='purple', label='Losers')

plt.legend()

plt.title('Loose Rates Distribution Analysis Graph')

plt.xlabel('loose Rate (%)')
plt.ylabel('Frequency')
```

Out[53]:
```
Text(0, 0.5, 'Frequency')
```



## Boxplot Distribution to show the outliers between the data points.

In [54]:
```
plt.subplot(1, 2, 2)
sns.boxplot(data=[win_rate, loss_rate], palette=['pink', 'black'])
plt.xticks([0, 1], ['Winners', 'Losers'])
plt.title('Boxplot of Passing Rates')
plt.ylabel('Passing Rate (%)')

plt.tight_layout()
plt.show()
```




In [ ]:
```

```

## Hypothesis Testing

In [55]:
```
# Two-sample t-test for passing rates between winners and losers
t_stat_rate, p_val_rate = ttest_ind(win_rate, loss_rate, alternative='great
print(f"t-statistic for passing rates: {t_stat_rate} \n p-value: {p_val_rat
```

```
t-statistic for passing rates: 2.741802595638678
 p-value: 0.0032373825441299096
```

## What is T - Statistics ?

### In statistics, the t-statistic is the ratio of the difference in a number's estimated value from its assumed value to its standard error.

## What is p value in statistics ?

### In statistics, a p-value is defined as a number that indicates how likely you are to obtain a value that is at least equal to or more than the actual observation if the null hypothesis is correct.

In [56]:
```
# Prepare Data for T-tests.

draws = games[games['diff'] == 0]
non_draws = games[games['diff'] != 0]
```

### It show the draw passing rate

In [57]:
```
draws
```

Out[57]:

| | game_id | passing_quotes | diff |
|---|---|---|---|
| **33** | 47 | [69.0, 69.0] | 0.0 |
| **74** | 93 | [85.0, 85.0] | 0.0 |

In [58]:
```
non_draws
```

Out[58]:

| | game_id | passing_quotes | diff |
|---|---|---|---|
| **0** | 11 | [72.0, 91.0] | 19.0 |
| **1** | 12 | [82.0, 86.0] | 4.0 |
| **2** | 13 | [82.0, 79.0] | 3.0 |
| **3** | 14 | [79.0, 77.0] | 2.0 |
| **4** | 15 | [85.0, 77.0] | 8.0 |
| **...** | ... | ... | ... |
| **148** | 175 | [84.0, 76.0] | 8.0 |
| **149** | 176 | [76.0, 91.0] | 15.0 |
| **150** | 177 | [78.0, 81.0] | 3.0 |
| **151** | 178 | [73.0, 74.0] | 1.0 |
| **152** | 179 | [74.0, 89.0] | 15.0 |

151 rows × 3 columns

## Lets check the passing rate difference with retriving the T-test and P-Value.

In [59]:
```
# Two-sample t-test for difference in passing rates between games with winr
t_stat_diff, p_val_diff = ttest_ind(non_draws['diff'], draws['diff'], alter
print(f"t-statistic for differences in passing rates: {t_stat_diff}, \np-va
```

```
t-statistic for differences in passing rates: nan,
p-value: nan
```

In [ ]:
```

```