Male / Female

In [1]:

```
import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency
from scipy.stats import fisher_exact

data = pd.read_csv(filepath_or_buffer='../../../Archive/HTWTempRatios.csv')
```

In [2]:

```
#Compute a contingency table for men/women hitting the wall.
data["HTW"] = (data['DoS15km'] >= 0.25) | (data['DoS20km'] >= 0.25)
htw_tab = pd.crosstab(data['Gender'], data['HTW'])
```

In [3]:

```
htw_tab
```

Out[3]:

HTW	False	True
Gender		
F	132874	7535
M	254281	28806

In [4]:

```
oddsr_htw, p_htw = fisher_exact(htw_tab)
print("Male / Female OR for HTW: ", oddsr_htw)
print("p: ", p_htw)
```

```
Male / Female OR for HTW:  1.9976794316395454
p:  0.0
```

In [5]:

```
#Compute a contingecy table for negative (or equal) splits
#data["NegSplit"] =
splits_tab = pd.crosstab(data['Gender'], data['SplitRatio'] <= 1)
splits_tab
```

Out[5]:

SplitRatio	False	True
Gender		
F	127438	12971
M	254572	28515

In [6]:

```
oddsr, p = fisher_exact(splits_tab)
```

In [7]:

```
print("Male / Female OR for Negativ Split: ",oddsr)
print("p: ", p)
```

```
Male / Female OR for Negativ Split:  1.1004953938037632
p:  5.747684234938161e-18
```

In [ ]:

Age groups

In [1]:
```python
import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency
from scipy.stats import fisher_exact

data = pd.read_csv(filepath_or_buffer='../../../Archive/HTWTempRatios.csv')
```

In [2]:
```python
#Compute a contingency table for age groups hitting the wall.
data["HTW"] = (data['DoS15km'] >= 0.25) | (data['DoS20km'] >= 0.25)
data["AgeGroup"] = "None"

data.loc[data["Age"].between(17,29, inclusive='both'), 'AgeGroup'] = "17-29" #remove any with missing age
data.loc[data["Age"].between(30,39, inclusive='both'), 'AgeGroup'] = "30-39"
data.loc[data["Age"].between(40,49, inclusive='both'), 'AgeGroup'] = "40-49"
data.loc[data["Age"].between(50,59, inclusive='both'), 'AgeGroup'] = "50-59"
data.loc[data["Age"].between(60,99, inclusive='both'), 'AgeGroup'] = "60+" #remove any unrealisic outlier

# Show number of male/female runner per age group
mf_tab = pd.crosstab(data["AgeGroup"], data['Gender'])
mf_tab
```

Out[2]:

Gender	F	M
AgeGroup		
17-29	39300	49731
30-39	41320	84164
40-49	38938	85337
50-59	16488	45773
60+	3331	15941
None	1032	2141

Runners Hitting the Wall per age group

In [3]:
```python
df = data.loc[data["AgeGroup"] != "None"] # drop datapoints with missing or wrong age.
htw_tab = pd.crosstab(df["AgeGroup"], df['HTW'])
f_htw_tab = pd.crosstab((df.loc[df["Gender"] == "F"])["AgeGroup"], df['HTW'])
m_htw_tab = pd.crosstab((df.loc[df["Gender"] == "M"])["AgeGroup"], df['HTW'])
```

In [4]:
```python
# Number of females hitting the wall per age group
f_htw_tab
```

Out[4]:

HTW	False	True
AgeGroup		
17-29	36549	2751
30-39	39363	1957
40-49	37290	1648
50-59	15576	912
60+	3152	179

In [5]:
```python
c, p, dof, expected = chi2_contingency(f_htw_tab)
print("Chi-squared females HTW by age groups, p: ", p)
```

```
Chi-squared females HTW by age groups, p:  3.145193918645074e-72
```

In [6]:
```python
# Number of males hitting the wall per age group
m_htw_tab
```

Out[6]:

HTW	False	True
AgeGroup		
17-29	43284	6447
30-39	75567	8597
40-49	77699	7638
50-59	41528	4245
60+	14362	1579

In [7]:
```python
c, p, dof, expected = chi2_contingency(m_htw_tab)
print("Chi-squared males HTW by age group, p: ", p)
```

```
Chi-squared males HTW by age group, p:  6.021907258238046e-130
```

In [8]:
```python
# Overall runners hitting the wall per age group
htw_tab
```

Out[8]:

HTW	False	True
AgeGroup		
17-29	79833	9198
30-39	114930	10554
40-49	114989	9286
50-59	57104	5157
60+	17514	1758

In [9]:
```python
c, p, dof, expected = chi2_contingency(htw_tab)
print("Chi-squared (all) by age group, p: ", p)
```

```
Chi-squared (all) by age group, p:  1.955975541098763e-120
```

In [10]:
```python
# Effect sizes between men and women, within age group
df1 = df.loc[df["AgeGroup"] == "17-29"]
df2 = df.loc[df["AgeGroup"] == "30-39"]
df3 = df.loc[df["AgeGroup"] == "40-49"]
df4 = df.loc[df["AgeGroup"] == "50-59"]
df5 = df.loc[df["AgeGroup"] == "60+"]
oddsr1, p1 = fisher_exact(pd.crosstab(df1['Gender'], df1['HTW']))
oddsr2, p2 = fisher_exact(pd.crosstab(df2['Gender'], df2['HTW']))
oddsr3, p3 = fisher_exact(pd.crosstab(df3['Gender'], df3['HTW']))
oddsr4, p4 = fisher_exact(pd.crosstab(df4['Gender'], df4['HTW']))
oddsr5, p5 = fisher_exact(pd.crosstab(df5['Gender'], df5['HTW']))
#print("Difference between Male/Female within age group: ")
print("Effect size for HTW between M/F within each age group")
print ("Age Group 17-29 M vs. F:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 M vs. F:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 M vs. F:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 M vs. F:\n p: ",p4, " OR: ", oddsr4)
print ("Age Group 60+ M vs. F:\n p: ",p5, " OR: ", oddsr5)
```

```
Effect size for HTW between M/F within each age group
Age Group 17-29 M vs. F:
 p:  1.6474952537552732e-191  OR:  1.978860548932631
Age Group 30-39 M vs. F:
 p:  2.7750442515647507e-259  OR:  2.2882955190897842
Age Group 40-49 M vs. F:
 p:  5.189780685309522e-208  OR:  2.2243309287758075
Age Group 50-59 M vs. F:
 p:  2.5021991745757493e-54  OR:  1.7458132243052247
Age Group 60+ M vs. F:
 p:  2.9065017297764597e-18  OR:  1.9359778559031087
```

In [11]:
```python
#Effect sizes between successive age groups
g1 = df.loc[(df["AgeGroup"] == "17-29") | (df["AgeGroup"] == "30-39") ]
g2 = df.loc[(df["AgeGroup"] == "30-39") | (df["AgeGroup"] == "40-49") ]
g3 = df.loc[(df["AgeGroup"] == "40-49") | (df["AgeGroup"] == "50-59") ]
g4 = df.loc[(df["AgeGroup"] == "50-59") | (df["AgeGroup"] == "60+") ]

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1["HTW"]))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2["HTW"]))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3["HTW"]))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4["HTW"]))
print("Effect size for HTW between consequtive age groups (F+M).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for HTW between consequtive age groups (F+M).
Age Group 17-29 vs. 30-39:
 p:  1.8442832507110868e-51  OR:  0.797026438112674
Age Group 30-39 vs. 40-49:
 p:  4.157398679773083e-18  OR:  0.8794045312221599
Age Group 40-49 vs. 50-59:
 p:  7.933325469560573e-10  OR:  1.1182997263359846
Age Group 50-59 vs. 60+:
 p:  0.00028888051472587795  OR:  1.1114832558452532
```

In [12]:
```python
#Effect sizes between successive age groups, female only
f_df = (df.loc[df["Gender"] == "F"])
g1 = f_df.loc[(f_df["AgeGroup"] == "17-29") | (f_df["AgeGroup"] == "30-39") ]
g2 = f_df.loc[(f_df["AgeGroup"] == "30-39") | (f_df["AgeGroup"] == "40-49") ]
g3 = f_df.loc[(f_df["AgeGroup"] == "40-49") | (f_df["AgeGroup"] == "50-59") ]
g4 = f_df.loc[(f_df["AgeGroup"] == "50-59") | (f_df["AgeGroup"] == "60+") ]

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1["HTW"]))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2["HTW"]))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3["HTW"]))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4["HTW"]))
print("Effect size for HTW between consequtive age groups (F only).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for HTW between consequtive age groups (F only).
Age Group 17-29 vs. 30-39:
 p:  8.604064034396752e-43  OR:  0.6605223904972399
Age Group 30-39 vs. 40-49:
 p:  0.0005722777015625179  OR:  0.8889189990261253
Age Group 40-49 vs. 50-59:
 p:  6.18639820700976e-11  OR:  1.3248724699687346
Age Group 50-59 vs. 60+:
 p:  0.7391330695303394  OR:  0.9699021506812717
```

In [13]:
```python
# Effect size between 17-29 and 40-49 groups for females, where we have largest differences.
g = f_df.loc[(f_df["AgeGroup"] == "17-29") | (f_df["AgeGroup"] == "40-49") ]
oddsr1, p1 = fisher_exact(pd.crosstab(g["AgeGroup"],g["HTW"]))
print("Female 17-29 vs 40-49:\n")
print("p: ",p1)
print("OR: ",oddsr1)
```

```
Female 17-29 vs 40-49:

p:  5.5167562182057243e-64
OR:  0.58715090219515
```

In [14]:
```python
#Effect sizes between successive age groups, male only
m_df = (df.loc[df["Gender"] == "M"])
g1 = m_df.loc[(m_df["AgeGroup"] == "17-29") | (m_df["AgeGroup"] == "30-39") ]
g2 = m_df.loc[(m_df["AgeGroup"] == "30-39") | (m_df["AgeGroup"] == "40-49") ]
g3 = m_df.loc[(m_df["AgeGroup"] == "40-49") | (m_df["AgeGroup"] == "50-59") ]
g4 = m_df.loc[(m_df["AgeGroup"] == "50-59") | (m_df["AgeGroup"] == "60+") ]

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1["HTW"]))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2["HTW"]))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3["HTW"]))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4["HTW"]))

print("Effect size for HTW between consequtive age groups (M only).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for HTW between consequtive age groups (M only).
Age Group 17-29 vs. 30-39:
 p:  1.27518532304456e-52  OR:  0.7638084589884682
Age Group 30-39 vs. 40-49:
 p:  9.370911033589651e-19  OR:  0.8640710984290758
Age Group 40-49 vs. 50-59:
 p:  0.05270217948188681  OR:  1.0398542090417837
Age Group 50-59 vs. 60+:
 p:  0.01990636204899734  OR:  1.075549812528776
```

In [15]:
```python
# Effect size between 17-29 and 40-49 for males, where we have largest differences.
g = m_df.loc[(m_df["AgeGroup"] == "17-29") | (m_df["AgeGroup"] == "40-49") ]
oddsr1, p1 = fisher_exact(pd.crosstab(g["AgeGroup"],g["HTW"]))
print("Male 17-29 vs 40-49:\n")
print("p: ",p1)
print("OR: ",oddsr1)
```

```
Male 17-29 vs 40-49:

p:  4.94315278868589e-117
OR:  0.6599848141475854
```

Analysis of runner pacing well: running a negative or equal split.

In [16]:
```python
splits_tab = pd.crosstab(df['AgeGroup'], df['SplitRatio'] <= 1)
f_splits_tab = pd.crosstab((df.loc[df["Gender"] == "F"])["AgeGroup"], df['SplitRatio'] <= 1)
m_splits_tab = pd.crosstab((df.loc[df["Gender"] == "M"])["AgeGroup"], df['SplitRatio'] <= 1)
```

In [17]:
```python
# Female negative splits per age group
f_splits_tab
```

Out[17]:

SplitRatio	False	True
AgeGroup		
17-29	34451	4849
30-39	37147	4173
40-49	35924	3014
50-59	15718	770
60+	3232	99

In [18]:
```python
c, p, dof, expected = chi2_contingency(f_splits_tab)
print("Chi-squared female negatice splits per age group, p: ", p)
```

```
Chi-squared female negatice splits per age group, p:  8.558705866172767e-249
```

In [19]:
```python
# Male negative splits per age group
m_splits_tab
```

Out[19]:

SplitRatio	False	True
AgeGroup		
17-29	42409	7322
30-39	74372	9792
40-49	77684	7653
50-59	42892	2881
60+	15241	700

In [20]:
```python
c, p, dof, expected = chi2_contingency(m_splits_tab)
print("Chi-squared male negative splits per age group, p: ",p)
```

```
Chi-squared male negative splits per age group, p:  0.0
```

In [21]:
```python
# Overall negative splits per age group
splits_tab
```

Out[21]:

SplitRatio	False	True
AgeGroup		
17-29	76860	12171
30-39	111519	13965
40-49	113608	10667
50-59	58610	3651
60+	18473	799

In [22]:
```python
c, p, dof, expected = chi2_contingency(splits_tab)
print("Chi-squared all runners negative splits per age group, p: ",p)
```

```
Chi-squared all runners negative splits per age group, p:  0.0
```

In [23]:
```python
# Effect sizes between men and women, within age group

df1 = df.loc[df["AgeGroup"] == "17-29"]
df2 = df.loc[df["AgeGroup"] == "30-39"]
df3 = df.loc[df["AgeGroup"] == "40-49"]
df4 = df.loc[df["AgeGroup"] == "50-59"]
df5 = df.loc[df["AgeGroup"] == "60+"]
oddsr1, p1 = fisher_exact(pd.crosstab(df1['Gender'], df1['SplitRatio'] <= 1))
oddsr2, p2 = fisher_exact(pd.crosstab(df2['Gender'], df2['SplitRatio'] <= 1))
oddsr3, p3 = fisher_exact(pd.crosstab(df3['Gender'], df3['SplitRatio'] <= 1))
oddsr4, p4 = fisher_exact(pd.crosstab(df4['Gender'], df4['SplitRatio'] <= 1))
oddsr5, p5 = fisher_exact(pd.crosstab(df5['Gender'], df5['SplitRatio'] <= 1))
print("Effect size for Negative Split between M/F within each age group")
print ("Age Group 17-29 M vs. F:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 M vs. F:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 M vs. F:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 M vs. F:\n p: ",p4, " OR: ", oddsr4)
print ("Age Group 60+ M vs. F:\n p: ",p5, " OR: ", oddsr5)
```

```
Effect size for Negative Split between M/F within each age group
Age Group 17-29 M vs. F:
 p:  6.327592315403746e-25  OR:  1.2266519146322405
Age Group 30-39 M vs. F:
 p:  2.8885770635229154e-16  OR:  1.172026159671495
Age Group 40-49 M vs. F:
 p:  5.267760069681919e-13  OR:  1.1741986412412397
Age Group 50-59 M vs. F:
 p:  8.311889480682368e-15  OR:  1.3711138576987687
Age Group 60+ M vs. F:
 p:  0.0001275621518623354  OR:  1.4994111444475593
```

In [24]:
```python
# Effect sizes between age groups
g1 = df.loc[(df["AgeGroup"] == "17-29") | (df["AgeGroup"] == "30-39") ]
g2 = df.loc[(df["AgeGroup"] == "30-39") | (df["AgeGroup"] == "40-49") ]
g3 = df.loc[(df["AgeGroup"] == "40-49") | (df["AgeGroup"] == "50-59") ]
g4 = df.loc[(df["AgeGroup"] == "50-59") | (df["AgeGroup"] == "60+") ]
htw_tab1 = pd.crosstab(g4["AgeGroup"], df['HTW'])

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1['SplitRatio'] <= 1))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2['SplitRatio'] <= 1))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3['SplitRatio'] <= 1))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4['SplitRatio'] <= 1))
print("Effect size for Negative Split between consequtive age groups (F+M).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for Negative Split between consequtive age groups (F+M).
Age Group 17-29 vs. 30-39:
 p:  9.04047385903589e-70  OR:  0.7907991454275962
Age Group 30-39 vs. 40-49:
 p:  3.0403979217925753e-101  OR:  0.7497928713511726
Age Group 40-49 vs. 50-59:
 p:  3.6240509276245927e-100  OR:  0.6634477581294234
Age Group 50-59 vs. 60+:
 p:  3.916675395737675e-21  OR:  0.6943352874759751
```

In [25]:
```python
# Effect sizes between age groups female only.
f_df = (df.loc[df["Gender"] == "F"])
g1 = f_df.loc[(f_df["AgeGroup"] == "17-29") | (f_df["AgeGroup"] == "30-39") ]
g2 = f_df.loc[(f_df["AgeGroup"] == "30-39") | (f_df["AgeGroup"] == "40-49") ]
g3 = f_df.loc[(f_df["AgeGroup"] == "40-49") | (f_df["AgeGroup"] == "50-59") ]
g4 = f_df.loc[(f_df["AgeGroup"] == "50-59") | (f_df["AgeGroup"] == "60+") ]

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1['SplitRatio'] <= 1))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2['SplitRatio'] <= 1))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3['SplitRatio'] <= 1))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4['SplitRatio'] <= 1))

print("Effect size for Negative Split between consequtive age groups (F only).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for Negative Split between consequtive age groups (F only).
Age Group 17-29 vs. 30-39:
 p:  7.049468683382514e-24  OR:  0.798131198338086
Age Group 30-39 vs. 40-49:
 p:  1.0651855290240096e-31  OR:  0.7468509217731616
Age Group 40-49 vs. 50-59:
 p:  9.146643315814892e-42  OR:  0.5838951669154245
Age Group 50-59 vs. 60+:
 p:  6.3866926537615255e-06  OR:  0.6252740452616691
```

In [26]:
```python
# Effect sizes between age groups male only
m_df = (df.loc[df["Gender"] == "M"])
g1 = m_df.loc[(m_df["AgeGroup"] == "17-29") | (m_df["AgeGroup"] == "30-39") ]
g2 = m_df.loc[(m_df["AgeGroup"] == "30-39") | (m_df["AgeGroup"] == "40-49") ]
g3 = m_df.loc[(m_df["AgeGroup"] == "40-49") | (m_df["AgeGroup"] == "50-59") ]
g4 = m_df.loc[(m_df["AgeGroup"] == "50-59") | (m_df["AgeGroup"] == "60+") ]

oddsr1, p1 = fisher_exact(pd.crosstab(g1["AgeGroup"],g1['SplitRatio'] <= 1))
oddsr2, p2 = fisher_exact(pd.crosstab(g2["AgeGroup"],g2['SplitRatio'] <= 1))
oddsr3, p3 = fisher_exact(pd.crosstab(g3["AgeGroup"],g3['SplitRatio'] <= 1))
oddsr4, p4 = fisher_exact(pd.crosstab(g4["AgeGroup"],g4['SplitRatio'] <= 1))

print("Effect size for Negative Split between consequtive age groups (M only).")
print ("Age Group 17-29 vs. 30-39:\n p: ",p1, " OR: ", oddsr1)
print ("Age Group 30-39 vs. 40-49:\n p: ",p2, " OR: ", oddsr2)
print ("Age Group 40-49 vs. 50-59:\n p: ",p3, " OR: ", oddsr3)
print ("Age Group 50-59 vs. 60+:\n p: ",p4, " OR: ", oddsr4)
```

```
Effect size for Negative Split between consequtive age groups (M only).
Age Group 17-29 vs. 30-39:
 p:  3.173761322558781e-59  OR:  0.7625884997559754
Age Group 30-39 vs. 40-49:
 p:  4.5356937185736234e-73  OR:  0.7482352934866341
Age Group 40-49 vs. 50-59:
 p:  6.641758363870426e-67  OR:  0.6818154328255547
Age Group 50-59 vs. 60+:
 p:  1.0258060794659224e-19  OR:  0.6837819241158386
```

In [ ]:

Finish Time Groups

In [1]:

```
import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency
from scipy.stats import fisher_exact
data = pd.read_csv(filepath_or_buffer='../../../Archive/HTWTempRatios.csv')
```

15 Min time groups

In [2]:

```
#Compute a contingency table for age groups hitting the wall.
data["HTW"] = (data['DoS15km'] >= 0.25) | (data['DoS20km'] >= 0.25)
data["FTGroup"] = 0

data.loc[(data['Time'] >= 45*60) & (data['Time'] < 75*60), 'FTGroup'] = 1 #remove any with missing/unrealistic time
data.loc[(data['Time'] >= 75*60) & (data['Time'] < 90*60), 'FTGroup'] = 2
data.loc[(data['Time'] >= 90*60) & (data['Time'] < 105*60), 'FTGroup'] = 3
data.loc[(data['Time'] >= 105*60) & (data['Time'] < 120*60), 'FTGroup'] = 4
data.loc[(data['Time'] >= 120*60) & (data['Time'] < 135*60), 'FTGroup'] = 5
data.loc[(data['Time'] >= 135*60) & (data['Time'] < 150*60), 'FTGroup'] = 6
data.loc[(data['Time'] >= 150*60) & (data['Time'] < 165*60), 'FTGroup'] = 7
data.loc[(data['Time'] >= 165*60) & (data['Time'] < 180*60), 'FTGroup'] = 8
data.loc[data["Time"] >= 180*60, 'FTGroup'] = 9


ctab = pd.crosstab(data["FTGroup"], data['HTW'])
f_ctab = pd.crosstab((data.loc[data["Gender"] == "F"])["FTGroup"], data['HTW'])
m_ctab = pd.crosstab((data.loc[data["Gender"] == "M"])["FTGroup"], data['HTW'])
ctab
```

Out[2]:

HTW	False	True
FTGroup		
1	594	1
2	11321	37
3	62201	831
4	123745	4701
5	99970	9649
6	53992	10185
7	22463	6052
8	8424	3105
9	4445	1780

In [3]:

```
c, p, dof, expected = chi2_contingency(ctab)
print("Chi-square HTW (all runners) per 15-min finish time group p: ", p)
```

```
Chi-square HTW (all runners) per 15-min finish time group p:  0.0
```

In [4]:

```
f_ctab
```

Out[4]:

HTW	False	True
FTGroup		
1	62	0
2	519	0
3	7039	24
4	32408	217
5	43086	1078
6	29039	2155
7	13169	1977
8	5079	1263
9	2473	821

In [5]:

```
m_ctab
```

Out[5]:

HTW	False	True
FTGroup		
1	532	1
2	10802	37
3	55162	807
4	91337	4484
5	56884	8571
6	24953	8030
7	9294	4075
8	3345	1842
9	1972	959

In [6]:

```
c, p, dof, expected = chi2_contingency(f_ctab)
print("Chi-square HTW (female) per 15-min finish time group p: ", p)
c, p, dof, expected = chi2_contingency(m_ctab)
print("Chi-square HTW (male) per 15-min finish time group p: ", p)
```

```
Chi-square HTW (female) per 15-min finish time group p:  0.0
Chi-square HTW (male) per 15-min finish time group p:  0.0
```

In [7]:

```
f_ctab2 = pd.crosstab((data.loc[data["Gender"] == "F"])["FTGroup"], (data['SplitRatio'] <= 1))
m_ctab2 = pd.crosstab((data.loc[data["Gender"] == "M"])["FTGroup"], (data['SplitRatio'] <= 1))
ctab2 = pd.crosstab(data["FTGroup"], (data['SplitRatio'] <= 1))
ctab2
```

Out[7]:

SplitRatio	False	True
FTGroup		
1	565	30
2	9668	1690
3	53137	9895
4	112058	16388
5	100379	9240
6	61133	3044
7	27797	718
8	11256	273
9	6017	208

In [8]:

```
c, p, dof, expected = chi2_contingency(ctab2)
print("Chi-square (neg split) for all runners per 15-min finish time group p: ", p)
```

```
Chi-square (neg split) for all runners per 15-min finish time group p:  0.0
```

In [9]:

```
f_ctab2
```

Out[9]:

SplitRatio	False	True
FTGroup		
1	62	0
2	484	35
3	6054	1009
4	27905	4720
5	39549	4615
6	29304	1890
7	14713	433
8	6187	155
9	3180	114

In [10]:

```
m_ctab2
```

Out[10]:

SplitRatio	False	True
FTGroup		
1	503	30
2	9184	1655
3	47083	8886
4	84153	11668
5	60830	4625
6	31829	1154
7	13084	285
8	5069	118
9	2837	94

In [11]:

```
c, p, dof, expected = chi2_contingency(f_ctab2)
print("Female (neg split) per 15-min finish time group p: ", p)

c, p, dof, expected = chi2_contingency(m_ctab2)
print("Male (neg split) per 15-min finish time group p: ", p)
```

```
Female (neg split) per 15-min finish time group p:  0.0
Male (neg split) per 15-min finish time group p:  0.0
```

In [ ]:

Temperature

In [1]:

```
import pandas as pd
import numpy as np
from scipy import stats
from scipy.stats import fisher_exact

data = pd.read_csv(filepath_or_buffer='../../../Archive/HTWTempRatios.csv')
data["HTW"] = (data['DoS15km'] >= 0.25) | (data['DoS20km'] >= 0.25)
temp = [21.7, 16.6, 13.6, 25, 18.9, 14.7, 15.1, 13.9, 20, 19.4]
years = [2010,2011,2012,2013,2014,2015,2016,2017,2018,2019]
runners = list(map (lambda x: len(data.loc[data['Year'] == x].index), years))
female = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['Gender'] == 'F')].index), years))
male = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['Gender'] == 'M')].index), years))
avg_time = list(map (lambda x: ((data.loc[(data['Year'] == x)])['Time']).mean(), years))
avg_time_f = list(map (lambda x: ((data.loc[(data['Year'] == x) & (data['Gender'] == 'F')])['Time']).mean(), years))
avg_time_m = list(map (lambda x: ((data.loc[(data['Year'] == x) & (data['Gender'] == 'M')])['Time']).mean(), years))
htw = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['HTW'] == True)].index), years))
htw_f = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['HTW'] == True) & (data['Gender'] == 'F')].index), years))
htw_m = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['HTW'] == True) & (data['Gender'] == 'M')].index), years))
neg_split = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['SplitRatio'] <= 1)].index), years))
neg_split_f = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['SplitRatio'] <= 1) & (data['Gender'] == 'F')].index), years))
neg_split_m = list(map (lambda x: len(data.loc[(data['Year'] == x) & (data['SplitRatio'] <= 1) & (data['Gender'] == 'M')].index), years))

d =  {'Year': years, 'Temp': temp, 'Runners': runners,'Female': female, 'Male': male,
      'Avg Time' : avg_time, 'Avg Time F' : avg_time_f, 'Avg Time M' : avg_time_m,
      'HTW': htw, 'HTW F': htw_f, 'HTW M' : htw_m, 'Neg Split' : neg_split,
     'Neg Split F' : neg_split_f, 'Neg Split M' : neg_split_m}
df = pd.DataFrame(data=d)
df['HTW%'] = df['HTW'] / df['Runners']
df['F HTW%'] = df['HTW F'] / df['Female']
df['M HTW%'] = df['HTW M'] / df['Male']
df['Neg Split%'] = df['Neg Split'] / df['Runners']
df['F Neg Split%'] = df['Neg Split F'] / df['Female']
df['M Neg Split%'] = df['Neg Split M'] / df['Male']
```

In [2]:

```
df
```

Out[2]:

	Year	Temp	Runners	Female	Male	Avg Time	Avg Time F	Avg Time M	HTW	HTW F	HTW M	Neg Split	Neg Split F	Neg Split M	HTW%	F HTW%	M HTW%	Neg Split%	F Neg Split%	M Neg Split%
0	2010	21.7	37982	10996	26986	7643.674056	8145.278192	7439.285148	5141	622	4519	2171	740	1431	0.135354	0.056566	0.167457	0.057159	0.067297	0.053027
1	2011	16.6	42838	13179	29659	7263.707736	7798.754989	7025.959068	2358	507	1851	5940	1597	4343	0.055045	0.038470	0.062409	0.138662	0.121178	0.146431
2	2012	13.6	44094	13750	30344	7208.821881	7749.826618	6963.672423	2768	607	2161	6297	1725	4572	0.062775	0.044145	0.071217	0.142809	0.125455	0.150672
3	2013	25.0	44919	14814	30105	7747.183441	8205.571756	7521.620761	6189	1123	5066	2997	1178	1819	0.137781	0.075807	0.168278	0.066720	0.079519	0.060422
4	2014	18.9	47187	16323	30864	7458.688389	7989.805060	7177.797466	5007	1179	3828	3165	1058	2107	0.106110	0.072229	0.124028	0.067074	0.064817	0.068267
5	2015	14.7	46207	16086	30121	7335.203281	7845.001865	7062.947379	3339	794	2545	5139	1592	3547	0.072262	0.049360	0.084493	0.111217	0.098968	0.117758
6	2016	15.1	44972	15662	29310	7343.250111	7876.157323	7058.487479	2522	573	1949	4442	1601	2841	0.056079	0.036585	0.066496	0.098773	0.102222	0.096929
7	2017	13.9	42252	14557	27695	7323.356078	7848.933915	7047.102726	2220	559	1661	5803	1834	3969	0.052542	0.038401	0.059975	0.137343	0.125987	0.143311
8	2018	20.0	39911	13775	26136	7519.654431	8079.968494	7224.340450	3614	814	2800	3133	1035	2098	0.090551	0.059093	0.107132	0.078500	0.075136	0.080272
9	2019	19.4	33134	11267	21867	7492.973200	8066.226502	7197.603695	3183	757	2426	2399	611	1788	0.096064	0.067187	0.110943	0.072403	0.054229	0.081767

Average times by temperature

In [3]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Avg Time'])
print ("Linear regression all runners finish time")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("intercept: ", intercept)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Linear regression all runners finish time
r-squared: 0.9111089869658573
p:  1.7715249036464247e-05
intercept:  6655.2428230633195
slope:  43.510812593271396
std err:  4.805028447513403
```

In [4]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Avg Time F'])
print ("Linear regression female finish time")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("intercept: ", intercept)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Linear regression female finish time
r-squared: 0.8979538035167087
p:  3.094503499458341e-05
intercept:  7248.00461670852
slope:  39.82939378516619
std err:  4.74711897897793
```

In [5]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Avg Time M'])
print ("Linear regression male finish time")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("intercept: ", intercept)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Linear regression male finish time
r-squared: 0.9098993795605811
p:  1.8709167068037667e-05
intercept:  6338.6711042585885
slope:  46.57409475643498
std err:  5.181632825176719
```

HTW rates by temperature

In [6]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['HTW'])
print("Number of runners HTW by temperature")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Number of runners HTW by temperature
r-squared: 0.7463542610048088
p:  0.0012689337299441673
slope:  312.5197011887763
std err:  64.4130233512836
```

In [7]:

```
#linear regression for men HTW by temperature.
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['HTW M'])
print("Male runners HTW by temperature")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Male runners HTW by temperature
r-squared: 0.7585656976834076
p:  0.0010352165980106564
slope:  274.76796377663305
std err:  54.80548035366965
```

In [8]:

```
#linear regression for women HTW by temperature.
#It's not a good fit and women seem to not increase their risk of HTW as much as men.
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['HTW F'])
print("Female runners HTW by temperature")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
Female runners HTW by temperature
r-squared: 0.3666048334437487
p:  0.06359434553891391
slope:  37.751737412143335
std err:  17.544065061839188
```

In [9]:

```
#linear regression for % of men HTW by temperature.
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['M HTW%'])
print("% Male runners HTW by temperature")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
% Male runners HTW by temperature
r-squared: 0.8455525035440344
p:  0.0001662461930069671
slope:  0.010041298056111033
std err:  0.0015172774786787909
```

In [10]:

```
#linear regression for % of women HTW by temperature.
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['F HTW%'])
print("% Female runners HTW by temperature")
print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
% Female runners HTW by temperature
r-squared: 0.66332037137115
p:  0.0041185241140052785
slope:  0.003169793192571831
std err:  0.000798422352730542
```

Negative Split Rates by Temperature

For the negative splits, the relationship isn't really linear (see plots), which will be shown below, but rather there are two clusters, below and above 18 degrees, when more runners manage a negative split in the five cooler years.

In [11]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Neg Split'])

print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
r-squared: 0.6841474625576467
p:  0.003153405979409332
slope:  -343.9905655168061
std err:  82.63591930164884
```

In [12]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Neg Split M'])

print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
r-squared: 0.6863935996805802
p:  0.0030608845197343205
slope:  -257.58702837444486
std err:  61.55804823171513
```

In [13]:

```
slope, intercept, r_value, p_value, std_err = stats.linregress(df['Temp'],df['Neg Split F'])

print ("r-squared:", r_value**2)
print ("p: ", p_value)
print ("slope: ", slope)
print ("std err: ", std_err)
```

```
r-squared: 0.569516782636373
p:  0.011643943985669154
slope:  -86.40353714236127
std err:  26.55897911788878
```

As the number of negative splits isn't linear, it appear to form two clusers, one for the cold years < 18 degrees, and one for the warm years > 18 degrees. Let's compute the differences between these conditions.

In [14]:

```
ct = pd.crosstab(data['temperature'] < 18, data['SplitRatio'] <= 1)
#temperature = False means the temperature is above 18 degrees C, temperatur = True that it is below
f_ct = pd.crosstab((data.loc[data["Gender"] == "F"])['temperature'] < 18, data['SplitRatio'] <= 1)
m_ct = pd.crosstab((data.loc[data["Gender"] == "M"])['temperature'] < 18, data['SplitRatio'] <= 1)
ct
```

Out[14]:

SplitRatio	False	True
temperature		
False	189268	13865
True	192742	27621

In [15]:

```
f_ct
```

Out[15]:

SplitRatio	False	True
temperature		
False	62553	4622
True	64885	8349

In [16]:

```
m_ct
```

Out[16]:

SplitRatio	False	True
temperature		
False	126715	9243
True	127857	19272

In [17]:

```
oddsr, p = fisher_exact(ct)
oddsrF, pF = fisher_exact(f_ct)
oddsrM, pM = fisher_exact(m_ct)
print("Neagtive Splits warm vs cold years: \n p: ", p, "OR: ", oddsr)
print("Neagtive Splits warm vs cold years (Female): \n p: ", pF, "OR: ", oddsrF)
print("Neagtive Splits warm vs cold years (Men): \n p: ", pM, "OR: ", oddsrM)
```

```
Neagtive Splits warm vs cold years:
 p:  0.0 OR:  1.9562319862232438
Neagtive Splits warm vs cold years (Female):
 p:  1.50459121297107e-190 OR:  1.7414393511243988
Neagtive Splits warm vs cold years (Men):
 p:  0.0 OR:  2.0664140774948905
```

In [ ]: