EER_replication_code

October 15, 2024

1 Tables in main text

1.1 Create dataframe - Stage Races

[129]:
```python
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy
import re
from scipy import stats
import statsmodels.formula.api as sm
from stargazer.stargazer import Stargazer, LineLocation
from scipy.stats import chi2_contingency

# Load and clean the 'stage_races_nf15.xlsx' dataset
stage = pd.read_excel('stage_races_nf15.xlsx')
stage = stage.drop(stage.columns[0], axis=1)  # Drop the first unnamed column
stage = stage.drop(columns=['Team_Score'])  # Drop the 'Team_Score' column
stage = stage.drop_duplicates()  # Remove duplicate rows

# Load and clean the 'itts.xlsx' dataset (individual time trials)
itt = pd.read_excel('itts.xlsx')
itt = itt.drop(itt.columns[0], axis=1)  # Drop the first unnamed column
itt = itt.drop_duplicates()  # Remove duplicate rows

# Find common rows between 'stage' and 'itt' based on 'Race' and 'Stage' columns
common_rows = stage.merge(itt, on=['Race', 'Stage'])

# Remove the common rows from the 'stage' dataset
stage_noitt = stage[~stage.set_index(['Race', 'Stage']).index.isin(itt.
 ↪set_index(['Race', 'Stage']).index)]

# Reset the index for the resulting 'stage_noitt' dataframe
stage_noitt.reset_index(drop=True, inplace=True)

# Update 'stage' to reflect the dataset without individual time trial (ITT)␣
 ↪stages
```

```
stage = stage_noitt
```

[130]:
```python
# Initialize columns related to teammates, clusters, and rider roles
stage['Teammates'] = 0
stage['Cluster_size_teams'] = 1
stage['Cluster_size_teams_hyp'] = 1
stage['Star'] = 0
stage['not_a_Star'] = 0
stage['Star_other'] = 0
stage['Star_other_team'] = 0
stage['Star_my_team'] = 0
stage['Star_of_Cluster'] = 0
stage['Star_other_in_Cluster'] = 0
stage['Star_other_team_in_Cluster'] = 0
stage['Star_my_team_in_Cluster'] = 0
stage['Helper_in_Cluster'] = 0
stage['Captain_in_Cluster'] = 0
stage['Helper_hyp_in_Cluster'] = 0
stage['Captain_hyp_in_Cluster'] = 0
stage['Star_in_Cluster1'] = 0
stage['Star_in_Cluster2'] = 0
stage['Winner_is_Star'] = 0
stage['Teammates_behind'] = 0
stage['eliminate'] = 0
stage['Teammates_behind_hyp'] = 0
stage['Star_Teammate_behind'] = 0
stage['Teammates_front'] = 0
stage['Teammates_front_hyp'] = 0
stage['Win'] = 0
stage['Cluster_size'] = 1
stage['Cluster'] = 1

# Mark winners and extract year from stage info
stage.loc[stage['Place'] == 1, 'Win'] = 1
stage['Year'] = stage['Stage'].str.split(':', expand=True)[0].astype(int)
```

1.1.1 Stars

[131]:
```python
# Loop through years to create dummy variables for each year and identify stars
for i in range(1981, 2024):
    stage[f'Dummy_{i}'] = 0
    stage.loc[stage['Year'] == i, f'Dummy_{i}'] = 1
    threshold_up = stage.loc[stage['Year'] == i, 'Score'].quantile(0.80)
    threshold_down = stage.loc[stage['Year'] == i, 'Score'].quantile(0.20)

    stage.loc[stage['Year'] == i, 'Star'] = (stage.loc[stage['Year'] == i,␣
↪'Score'] >= threshold_up).astype(int)
```

```
    stage.loc[stage['Year'] == i, 'not_a_Star'] = (stage.loc[stage['Year'] ==␣
↪i, 'Score'] < threshold_up).astype(int)

# Clean up race names and create dummy variables for each race
for race in stage['Race'].unique():
    cleaned_race = race.replace('/', '').replace('-', '_')
    stage.loc[stage['Race'] == race, 'Race'] = cleaned_race
    stage[f'Dummy_{cleaned_race}'] = (stage['Race'] == cleaned_race).astype(int)

# Create dummy variables for each stage type
for stagetype in stage['Stagetype'].unique():
    stage[f'Dummy_stagetype_{stagetype}'] = (stage['Stagetype'] == stagetype).
↪astype(int)

# Create additional composite identifiers
stage['Race_Stage'] = stage['Race'] + stage['Stage'].astype(str)  # E.g., "2003:
↪ 2"
stage['Race_Year'] = stage['Race'] + stage['Year'].astype(str)  # E.g., "2003:␣
↪2"

# Assign hypothetical teams (randomly assigning teams 1 to 22)
stage['hyp_team'] = np.random.randint(1, 23, size=len(stage))
```

1.1.2 Groups

[132]:
```
# Iterate through unique race stages to define clusters and teammate roles
for group_name in stage['Race_Stage'].unique():
    group_data = stage[stage['Race_Stage'] == group_name]

    # Define clusters based on time gaps
    for i in range(1, len(group_data)):
        gap_difference = group_data.iloc[i]['Gap'] - group_data.iloc[i -␣
↪1]['Gap']
        if gap_difference > 4:  # 5+ seconds gap creates a new cluster
            stage.loc[group_data.index[i], 'Cluster'] = stage.loc[group_data.
↪index[i - 1], 'Cluster'] + 1
            stage.loc[group_data.index[i], 'Gap_front'] = gap_difference
        else:
            stage.loc[group_data.index[i], 'Gap_front'] = 0
            stage.loc[group_data.index[i], 'Cluster'] = stage.loc[group_data.
↪index[i - 1], 'Cluster']

    # Update cluster sizes and teammate information
    for i in range(len(group_data)):
        for j in range(i + 1, len(group_data)):
            same_cluster = stage.loc[group_data.index[i], 'Cluster'] == stage.
↪loc[group_data.index[j], 'Cluster']
            same_team = stage.loc[group_data.index[i], 'Team'] == stage.
↪loc[group_data.index[j], 'Team']
            if same_cluster:
                stage.loc[group_data.index[i], 'Cluster_size'] += 1
                stage.loc[group_data.index[j], 'Cluster_size'] += 1
                if same_team:
                    # Mark teammates in same cluster
                    stage.loc[group_data.index[i], 'Helper_in_Cluster'] = 1
                    stage.loc[group_data.index[j], 'Captain_in_Cluster'] = 1
                    stage.loc[group_data.index[i], 'Teammates'] = 1
                    stage.loc[group_data.index[j], 'Teammates'] = 1
                else:
                    # Non-teammates in the same cluster
                    pass

            # Mark teammates in neighboring clusters
            if same_team and stage.loc[group_data.index[i], 'Cluster'] + 1 ==␣
↪stage.loc[group_data.index[j], 'Cluster']:
                stage.loc[group_data.index[i], 'Teammates_behind'] = 1
                stage.loc[group_data.index[j], 'Teammates_front'] = 1

            if same_team and stage.loc[group_data.index[i], 'Cluster'] + 2 ==␣
↪stage.loc[group_data.index[j], 'Cluster']:
                stage.loc[group_data.index[j], 'Teammates_front'] = 1

            # Hypothetical teammates in same cluster
            if same_cluster and stage.loc[group_data.index[i], 'hyp_team'] ==␣
↪stage.loc[group_data.index[j], 'hyp_team']:
                stage.loc[group_data.index[i], 'Helper_hyp_in_Cluster'] = 1
                stage.loc[group_data.index[j], 'Captain_hyp_in_Cluster'] = 1

            # Hypothetical teammates in neighboring clusters
            if stage.loc[group_data.index[i], 'Cluster'] + 1 == stage.
↪loc[group_data.index[j], 'Cluster'] and stage.loc[group_data.index[i],␣
↪'hyp_team'] == stage.loc[group_data.index[j], 'hyp_team']:
                stage.loc[group_data.index[i], 'Teammates_behind_hyp'] = 1
                stage.loc[group_data.index[j], 'Teammates_front_hyp'] = 1
```

1.1.3 Remaining code

[133]:
```
# Calculate the number of unique teams per cluster
stage['Cluster_size_teams'] = stage.groupby(['Race_Stage', 'Cluster'])['Team'].
↪transform('nunique')
```

```
# Create dummy variables for clusters
for s in stage['Cluster'].unique():
    stage.loc[stage['Cluster'] == 1, 'Dummy_Cluster_1'] = 1
    stage.loc[stage['Cluster'] != 1, 'Dummy_Cluster_1'] = 0

# Identify the winners of each race stage and merge their cluster size␣
 ↪information
winners = stage[stage['Place'] == 1][['Race_Stage', 'Cluster_size',␣
 ↪'Cluster_size_teams']]
stage = pd.merge(stage, winners, on='Race_Stage', suffixes=('', '_winner'),␣
 ↪how='left')

# Identify the cluster size for the second cluster and merge with the main␣
 ↪dataset
second = stage[stage['Cluster'] == 2][['Race_Stage', 'Cluster_size',␣
 ↪'Cluster_size_teams']]
stage = pd.merge(stage, second, on='Race_Stage', suffixes=('', '_second'),␣
 ↪how='left')

# Identify the cluster size for the third cluster and merge with the main␣
 ↪dataset
third = stage[stage['Cluster'] == 3][['Race_Stage', 'Cluster_size',␣
 ↪'Cluster_size_teams']]
stage = pd.merge(stage, third, on='Race_Stage', suffixes=('', '_third'),␣
 ↪how='left')

# Filter and mark races for elimination based on conditions
for s in stage['Race_Stage'].unique():
    group_data = stage.loc[stage['Race_Stage'] == s]

    # Mark races for elimination if a rider in cluster 1, 2, or 3 places 15th
    for i in range(len(group_data)):
        if (stage.loc[group_data.index[i], 'Place'] == 15) and (stage.
 ↪loc[group_data.index[i], 'Cluster'] in [1, 2, 3]):
            stage.loc[group_data.index[i], 'eliminate'] = 1

        # Eliminate races where both the first and second clusters have only␣
 ↪one team each
        if (stage.loc[group_data.index[i], 'Cluster_size_teams_winner'] == 1)␣
 ↪and (stage.loc[group_data.index[i], 'Cluster_size_teams_second'] == 1):
            stage.loc[group_data.index[i], 'eliminate'] = 1

# Identify 'Race_Stage' values that should be eliminated
eliminate_race_stages = stage.loc[stage['eliminate'] == 1, 'Race_Stage'].
 ↪unique()
```

```python
# Apply elimination to all rows with the identified 'Race_Stage' values
stage.loc[stage['Race_Stage'].isin(eliminate_race_stages), 'eliminate'] = 1

# Remove rows marked for elimination
stage = stage[stage['eliminate'] != 1].copy()

# Drop duplicates and lay focus on first three clusters
stage = stage.drop_duplicates()
stage_filtered = stage[stage['Cluster'] <= 3].copy()

# Reset the index of the filtered DataFrame
stage_filtered.reset_index(drop=True, inplace=True)

# Update the main 'stage' DataFrame with the filtered data
stage = stage_filtered

# Print the number of stages used after filtering
print('We use a total of', len(stage['Race_Stage'].unique()), 'stages of stage␣
 ↪races.')

# Step 1: Identify if there are other 'Stars' in the same cluster
# Create a unique 'Cluster_id' for each combination of Race_Stage and Cluster
stage['Cluster_id'] = stage['Race_Stage'] + stage['Cluster'].astype(str)

# Group by 'Cluster_id' and count the number of 'Stars' in each cluster
grouped_data = stage.groupby('Cluster_id')['Star']
sum_star = grouped_data.transform('sum')

# Mark if there is another 'Star' in the cluster (either from the same or␣
 ↪different team)
stage['Star_other_in_Cluster'] = (((sum_star >= 2) & (stage['Star'] == 1)) |
                                   ((sum_star >= 1) & (stage['Star'] != 1))).
 ↪astype(int)

# Step 2: Identify if there is another 'Star' from a different team or the same␣
 ↪team in the cluster
for cluster_id in stage['Cluster_id'].unique():
    group_data = stage.loc[stage['Cluster_id'] == cluster_id]

    # Loop through each rider in the cluster and check for 'Star' teammates or␣
 ↪'Stars' from other teams
    for i in range(len(group_data)):
        for j in range(len(group_data)):
            # Other 'Star' from a different team
            if (stage.loc[group_data.index[i], 'Team'] != stage.loc[group_data.
 ↪index[j], 'Team']) and (stage.loc[group_data.index[j], 'Star'] == 1):
```

```
                stage.loc[group_data.index[i], 'Star_other_team_in_Cluster'] = 1

            # Other 'Star' from the same team (not the current rider)
            if (stage.loc[group_data.index[i], 'Team'] == stage.loc[group_data.
↪index[j], 'Team']) and (stage.loc[group_data.index[j], 'Star'] == 1) and (i !
↪= j):
                stage.loc[group_data.index[i], 'Star_my_team_in_Cluster'] = 1

# Drop the 'Cluster_id' column as it is no longer needed
stage.drop('Cluster_id', axis=1, inplace=True)

# Loop through each unique 'Race_Stage'
for race_stage in stage['Race_Stage'].unique():
    # Filter the data for the current race stage
    group_data = stage.loc[stage['Race_Stage'] == race_stage]

    # Loop through each rider in the current race stage
    for i in range(len(group_data)):
        # Check if there is another 'Star' in clusters 1 or 2
        stage.loc[group_data.index[i], 'Star_other'] = (np.
↪sum(group_data[(group_data['Cluster'] == 1) | (group_data['Cluster'] ==␣
↪2)]['Star']) > stage.loc[group_data.index[i], 'Star']).astype(int)

        # Check if there is a 'Star' in Cluster 1
        stage.loc[group_data.index[i], 'Star_in_Cluster1'] = (np.
↪sum(group_data[group_data['Cluster'] == 1]['Star']) > 0).astype(int)

        # Check if there is a 'Star' in Cluster 2
        stage.loc[group_data.index[i], 'Star_in_Cluster2'] = (np.
↪sum(group_data[group_data['Cluster'] == 2]['Star']) > 0).astype(int)

        # Check if the winner is a 'Star'
        stage.loc[group_data.index[i], 'Winner_is_Star'] = (np.
↪sum(group_data[group_data['Win'] == 1]['Star']) > 0).astype(int)

        # Calculate the maximum gap between Cluster 1 and Cluster 2
        stage.loc[group_data.index[i], 'Gap_Cluster12'] =␣
↪group_data[group_data['Cluster'] == 2]['Gap_front'].max()

        # Calculate the maximum gap between Cluster 2 and Cluster 3
        stage.loc[group_data.index[i], 'Gap_Cluster23'] =␣
↪group_data[group_data['Cluster'] == 3]['Gap_front'].max()

        # Check if there is a 'Helper' in Cluster 2
        stage.loc[group_data.index[i], 'Helper_in_Cluster2'] = (np.
↪sum(group_data[group_data['Cluster'] == 2]['Helper_in_Cluster']) > 0).
↪astype(int)

        # Check if the winner is part of a 'Satellite' group (teammates behind)
        stage.loc[group_data.index[i], 'Winner_is_Satellite'] = (np.
↪sum(group_data[group_data['Win'] == 1]['Teammates_behind']) > 0).astype(int)

        # Calculate the standard deviation of scores in Cluster 2
        stage.loc[group_data.index[i], 'Cluster2_std'] =␣
↪group_data[group_data['Cluster'] == 2]['Score'].std()

        # Check if there is a 'Star' from another team in clusters 1 or 2
        if (stage.loc[group_data.index[i], 'Team'] != stage.loc[group_data.
↪index[j], 'Team']) and ((stage.loc[group_data.index[j], 'Cluster'] == 1) |␣
↪(stage.loc[group_data.index[j], 'Cluster'] == 2)) and (stage.loc[group_data.
↪index[j], 'Star'] == 1) and (i != j):
            stage.loc[group_data.index[i], 'Star_other_team'] = 1

            # Check if there is a 'Star' from the same team in clusters 1 or 2
        if (stage.loc[group_data.index[i], 'Team'] == stage.loc[group_data.
↪index[j], 'Team']) and ((stage.loc[group_data.index[j], 'Cluster'] == 1) |␣
↪(stage.loc[group_data.index[j], 'Cluster'] == 2)) and (stage.loc[group_data.
↪index[j], 'Star'] == 1) and (i != j):
            stage.loc[group_data.index[i], 'Star_my_team'] = 1

# Create variables indicating the absence of stars within the rider's team,␣
↪other teams, and the cluster
stage['no_Star_my_team_in_Cluster'] = 1 - stage['Star_my_team_in_Cluster']
stage['no_Star_other_team_in_Cluster'] = 1 - stage['Star_other_team_in_Cluster']
stage['no_Star_other_team'] = 1 - stage['Star_other_team']
stage['no_Star'] = 1 - stage['Star']

# Create a variable indicating if there is a better rider in the cluster
# (i.e., dummy equal to 1 if the rider is not a Star but a Star exists in the␣
↪cluster)
stage['better_rider_in_Cluster'] = stage.apply(lambda row: 1 if␣
↪row['Star_other_team_in_Cluster'] == 1 and row['Star'] == 0 else 0, axis=1)

# Create a variable indicating if there is a better rider nearby (in the entire␣
↪group)
stage['better_rider_around'] = stage.apply(lambda row: 1 if␣
↪row['Star_other_team'] == 1 and row['Star'] == 0 else 0, axis=1)

# Identify solo wins (i.e., Cluster size for the winner equals 1)
stage['Solo_Win'] = (stage['Cluster_size_winner'] == 1).astype(int)
```

```
# Create dummy variables for the existence of helpers and gap sizes
stage['Helper_in_Cluster_exists'] = (stage['Cluster_size'] >␣
↪stage['Cluster_size_teams']).astype(int)
stage['Gap_12_larger1'] = (stage['Gap_Cluster12'] >= 60).astype(int)  # Gap␣
↪between Cluster 1 and Cluster 2
stage['Gap_23_larger1'] = (stage['Gap_Cluster23'] >= 60).astype(int)  # Gap␣
↪between Cluster 2 and Cluster 3

# Identify if the standard deviation in Cluster 2 is larger than the mean␣
↪standard deviation
stage['Cluster2_std_large'] = (stage['Cluster2_std'] >= stage['Cluster2_std'].
↪mean()).astype(int)

# Remove duplicate rows
stage = stage.drop_duplicates()

# Filter for captains only (no teammates in front and no captains in the␣
↪cluster)
stage_c = stage[(stage['Teammates_front'] == 0) & (stage['Captain_in_Cluster']␣
↪== 0)]

# Further filter captains to only include years after 1980 (since we don't have␣
↪scores before 1981)
stage_c = stage_c[stage_c['Year'].astype(int) > 1980]
```

```
We use a total of 729 stages of stage races.
```

1.2 Main Tables 4-7

[134]:
```
# Table 4: Asymmetry in losing versus winning groups

# Step 1: Filter out races where the Solo winner has a helper in Cluster 1
df2 = stage_c[~stage_c.Race_Stage.isin(
    stage_c[(stage_c['Cluster_size'] >= 2) &
            (stage_c['Cluster'] == 1) &
            (stage_c['Cluster_size_teams'] == 1)].Race_Stage)]

# Step 2: Exclude races where the second place has a gap less than 10 seconds
df1 = df2[~df2.Race_Stage.isin(
    df2[(df2['Place'] == 2) &
        (df2['Gap'] < 10)].Race_Stage)]

# Step 3: Select Stage 1 for Cluster 2 where cluster size is between 3 and 6
stage1 = df1[(df1['Cluster_size_teams_winner'] == 1) &
             (df1['Cluster'] == 2) &
             (df1['Cluster_size'] >= 3) &
             (df1['Cluster_size'] <= 6)].copy()  # Use .copy() to avoid␣
↪SettingWithCopyWarning

# Solo wins in Cluster 2
stage1['Cluster2_Solo'] = 1  # Solo win in Cluster 2
stage1['Cluster1_noSolo'] = 0  # Not a solo win in Cluster 1
stage1['Star_in_Cluster_exists'] = stage1['Star_in_Cluster2']  # Existence of␣
↪Star in Cluster 2
stage1['Helper_in_Cluster_exists'] = (stage1['Cluster_size'] >␣
↪stage1['Cluster_size_teams']).astype(int)  # Helper exists in Cluster 2
stage1['Solo_is_Satellite'] = 0  # No satellite win for Cluster 2
stage1['Star_has_Helper'] = 0  # Initialize Star_has_Helper as 0
stage1.loc[(stage1['Star'] + stage1['Helper_in_Cluster']) > 1,␣
↪'Star_has_Helper'] = 1
stage1['Star_w_Helper_exists'] = stage1.
↪groupby('Race_Stage')['Star_has_Helper'].transform('max')  # Check if any␣
↪Star has helper in the race

# Step 4: Select Stage 2 for Cluster 1 where cluster size is between 3 and 6
stage2 = df2[(df2['Cluster'] == 1) &
             (df2['Cluster_size'] >= 3) &
             (df2['Cluster_size'] <= 6)].copy()  # Use .copy() to avoid␣
↪SettingWithCopyWarning

# Add new variables for stage2
stage2['Cluster2_Solo'] = 0  # Not a solo win in Cluster 2
stage2['Cluster1_noSolo'] = 1  # Solo win in Cluster 1
stage2['Star_in_Cluster_exists'] = stage2['Star_in_Cluster1']  # Existence of␣
↪Star in Cluster 1
stage2['Helper_in_Cluster_exists'] = (stage2['Cluster_size'] >␣
↪stage2['Cluster_size_teams']).astype(int)  # Helper exists in Cluster 1
stage2['Solo_is_Satellite'] = stage2['Winner_is_Satellite']  # Satellite win in␣
↪Cluster 1
stage2['Star_has_Helper'] = 0  # Initialize Star_has_Helper as 0
stage2.loc[(stage2['Star'] + stage2['Helper_in_Cluster']) > 1,␣
↪'Star_has_Helper'] = 1
stage2['Star_w_Helper_exists'] = stage2.
↪groupby('Race_Stage')['Star_has_Helper'].transform('max')  # Check if any␣
↪Star has helper in the race

# Concatenate the two dataframes and remove duplicates
df_stage = pd.concat([stage1, stage2]).drop_duplicates('Race_Stage')

# Set heterogeneity indicator
df_stage['heterog'] = 0
```

```
df_stage.loc[(df_stage['Star_in_Cluster_exists'] +␣
 ↪df_stage['Helper_in_Cluster_exists']) >= 1, 'heterog'] = 1

# Step 5: Combine stage1 and stage2
df_stage = pd.concat([stage1, stage2])

# Remove duplicate 'Race_Stage' entries
df_stage = df_stage.drop_duplicates('Race_Stage')

# Step 6: Create heterogeneity variable 'heterog' and drop mountain finishes
df_stage['heterog'] = 0
df_stage.loc[(df_stage['Star_in_Cluster_exists'] +␣
 ↪df_stage['Helper_in_Cluster_exists']) >= 1, 'heterog'] = 1
df_stage = df_stage[df_stage['Dummy_stagetype_5'] == 0]  # Drop mountain␣
 ↪finishes

# Print the cells of Table 4:
# Calculate and print the mean of Helper_in_Cluster_exists for Cluster2_Solo␣
 ↪and Cluster1_noSolo
print(df_stage[df_stage['Cluster2_Solo'] == 1]['Helper_in_Cluster_exists'].
 ↪mean())
print(df_stage[df_stage['Cluster1_noSolo'] == 1]['Helper_in_Cluster_exists'].
 ↪mean())

# Calculate and print the mean of Star_in_Cluster_exists for Cluster2_Solo and␣
 ↪Cluster1_noSolo
print(df_stage[df_stage['Cluster2_Solo'] == 1]['Star_in_Cluster_exists'].mean())
print(df_stage[df_stage['Cluster1_noSolo'] == 1]['Star_in_Cluster_exists'].
 ↪mean())
```

```
0.14285714285714285
0.232
0.5238095238095238
0.616
```

[135]:
```
# Table 5: Linear Probability Model: Being part of a winning Group (with 3 to 6␣
 ↪riders)

# LHS: versus Group behind Solo winner
resultNoSolo = sm.ols('Cluster1_noSolo ~ Star_in_Cluster_exists +␣
 ↪Helper_in_Cluster_exists + Cluster_size_teams + Dummy_stagetype_1 +␣
 ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4',
                                   data=df_stage).fit()

print(resultNoSolo.summary())

# RHS: versus riders not finishing as Group
```

```python
# Step 1: Filter races where cluster size 1 and 2 are not too large
df = stage_c[(stage_c['Cluster_size_teams_winner'] < 3) &␣
 ↪(stage_c['Cluster_size_teams_second'] <= 4)]

# Step 2: Only keep riders that are not too far away (Gap < 30)
df2 = df[df['Gap'] < 30]

# Step 3: Exclude races where Cluster 2 is too far away (Gap >= 40)
stage1 = df2[~df2.Race_Stage.isin(df2[(df2['Cluster'] == 2) & (df2['Gap'] >=␣
 ↪40)].Race_Stage)].copy()

# Step 4: Create group size variables for stage1
stage1['Group_size_teams'] = stage1.groupby('Race_Stage')['Team'].
 ↪transform('nunique')
stage1['Group_size'] = stage1.groupby('Race_Stage')['Rider'].
 ↪transform('nunique')

# Step 5: Add group and helper/star variables for stage1
stage1['Group_together'] = 0
stage1['Star_in_Cluster_exists'] = stage1.groupby('Race_Stage')['Star'].
 ↪transform('max').astype(int)
stage1['Helper_in_Cluster_exists'] = (stage1['Group_size'] >␣
 ↪stage1['Group_size_teams']).astype(int)

# Step 6: Prepare stage2 with no solo riders (modifying .loc to avoid␣
 ↪SettingWithCopyWarning)
stage2 = stage_c[(stage_c['Cluster'] == 1) & (stage_c['Cluster_size'] >= 3) &␣
 ↪(stage_c['Cluster_size'] <= 6)].copy()
stage2.loc[:, 'Group_together'] = 1
stage2.loc[:, 'Star_in_Cluster_exists'] = stage2['Star_in_Cluster1']
stage2.loc[:, 'Helper_in_Cluster_exists'] = (stage2['Cluster_size'] >␣
 ↪stage2['Cluster_size_teams']).astype(int)
stage2.loc[:, 'Group_size_teams'] = stage2['Cluster_size_teams']
stage2.loc[:, 'Group_size'] = stage2['Cluster_size']

# Step 7: Combine stage1 and stage2 into a single DataFrame
df_stage = pd.concat([stage1, stage2]).drop_duplicates('Race_Stage')

# Step 8: Create a heterogeneity variable (heterog) indicating presence of star/
 ↪helper in cluster
df_stage['heterog'] = 0
df_stage.loc[(df_stage['Star_in_Cluster_exists'] +␣
 ↪df_stage['Helper_in_Cluster_exists']) >= 1, 'heterog'] = 1

# Step 9: Exclude mountain finishes (Dummy_stagetype_5 == 0)
```

```python
df_stage = df_stage[df_stage['Dummy_stagetype_5'] == 0]

# Step 10: Fit an OLS model to predict 'Group_together' based on various␣
 ↪variables
resultTog = sm.ols('Group_together ~ Star_in_Cluster_exists +␣
 ↪Helper_in_Cluster_exists + Group_size_teams + Dummy_stagetype_1 +␣
 ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4',
                                data=df_stage).fit()

# Print the summary of the OLS regression results
print(resultTog.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        Cluster1_noSolo   R-squared:                       0.019
Model:                            OLS   Adj. R-squared:                 -0.015
Method:                 Least Squares   F-statistic:                    0.5703
Date:                Tue, 15 Oct 2024   Prob (F-statistic):              0.780
Time:                        17:46:05   Log-Likelihood:                -145.53
No. Observations:                 209   AIC:                             307.1
Df Residuals:                     201   BIC:                             333.8
Df Model:                           7
Covariance Type:            nonrobust
================================================================================
============
                               coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
------------
Intercept                    0.5673      0.145      3.918      0.000       0.282
0.853
Star_in_Cluster_exists       0.0546      0.079      0.690      0.491      -0.101
0.211
Helper_in_Cluster_exists     0.1027      0.095      1.085      0.279      -0.084
0.289
Cluster_size_teams          -0.0050      0.036     -0.136      0.892      -0.077
0.067
Dummy_stagetype_1            0.0516      0.154      0.335      0.738      -0.252
0.355
Dummy_stagetype_2           -0.0403      0.092     -0.436      0.664      -0.222
0.142
Dummy_stagetype_3           -0.0413      0.129     -0.320      0.749      -0.296
0.213
Dummy_stagetype_4            0.0331      0.093      0.356      0.722      -0.150
0.216
==============================================================================
Omnibus:                     1405.312   Durbin-Watson:                   0.048
```

```
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               32.528
Skew:                          -0.388   Prob(JB):                     8.64e-08
Kurtosis:                       1.230   Cond. No.                         20.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:         Group_together   R-squared:                       0.164
Model:                            OLS   Adj. R-squared:                  0.148
Method:                 Least Squares   F-statistic:                     9.973
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           2.19e-11
Time:                        17:46:05   Log-Likelihood:                -207.57
No. Observations:                 364   AIC:                             431.1
Df Residuals:                     356   BIC:                             462.3
Df Model:                           7
Covariance Type:            nonrobust
================================================================================
============
                             coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
------------
Intercept                  0.1419      0.055      2.572      0.011       0.033
0.250
Star_in_Cluster_exists     0.0487      0.051      0.954      0.341      -0.052
0.149
Helper_in_Cluster_exists   0.6900      0.098      7.017      0.000       0.497
0.883
Group_size_teams           0.0328      0.013      2.445      0.015       0.006
0.059
Dummy_stagetype_1          0.1502      0.112      1.342      0.180      -0.070
0.370
Dummy_stagetype_2          0.0550      0.063      0.873      0.383      -0.069
0.179
Dummy_stagetype_3         -0.0831      0.087     -0.953      0.341      -0.255
0.088
Dummy_stagetype_4         -0.0109      0.059     -0.186      0.852      -0.126
0.104
==============================================================================
Omnibus:                       83.839   Durbin-Watson:                   0.328
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               63.693
Skew:                           0.920   Prob(JB):                     1.48e-14
Kurtosis:                       2.098   Cond. No.                         20.4
==============================================================================
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[136]:
```
# Table 6: Linear Probability Model: Finishing in Group 1

# LHS: G1 if in G1/G2
resultS12 = sm.ols(formula='Dummy_Cluster_1 ~ better_rider_around +␣
 ↪Teammates_behind + Gap_12_larger1 + Gap_23_larger1 +␣
 ↪Cluster_size_teams_winner + Cluster_size_teams_second +␣
 ↪Cluster_size_teams_third + Dummy_1982 + Dummy_1983 + Dummy_1985 + Dummy_1986␣
 ↪+ Dummy_1987 + Dummy_1988 + Dummy_1989 + Dummy_1990 + Dummy_1991 +␣
 ↪Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 + Dummy_1996 + Dummy_1997␣
 ↪+ Dummy_1998 + Dummy_1999 + Dummy_2000 + Dummy_2001 + Dummy_2002 +␣
 ↪Dummy_2003 + Dummy_2004 + Dummy_2005 + Dummy_2006 + Dummy_2007 + Dummy_2008␣
 ↪+ Dummy_2009 + Dummy_2010 + Dummy_2011 + Dummy_2012 + Dummy_2013 +␣
 ↪Dummy_2014 + Dummy_2015 + Dummy_2016 + Dummy_2017 + Dummy_2018 + Dummy_2019␣
 ↪+ Dummy_2020 + Dummy_2021 + Dummy_2022 + Dummy_2023 + Dummy_giro_d_italia +␣
 ↪Dummy_vuelta_a_espana + Dummy_dauphine + Dummy_tour_de_romandie +␣
 ↪Dummy_volta_a_catalunya + Dummy_itzulia_basque_country +␣
 ↪Dummy_tour_de_suisse + Dummy_tour_de_pologne + Dummy_paris_nice +␣
 ↪Dummy_tirreno_adriatico + Dummy_stagetype_1 + Dummy_stagetype_2 +␣
 ↪Dummy_stagetype_3 + Dummy_stagetype_4 + Dummy_stagetype_5',
                      data=stage_c[(stage_c['Cluster'] == 1) |␣
 ↪(stage_c['Cluster'] == 2)]).fit()
print(resultS12.summary())

# RHS: G1 if in G1/G2/G3
resultS123 = sm.ols(formula='Dummy_Cluster_1 ~ better_rider_around +␣
 ↪Gap_12_larger1 + Gap_23_larger1 + Cluster_size_teams_winner +␣
 ↪Cluster_size_teams_second + Cluster_size_teams_third + Dummy_1982 +␣
 ↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989␣
 ↪+ Dummy_1990 + Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 +␣
 ↪Dummy_1995 + Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000␣
 ↪+ Dummy_2001 + Dummy_2002 + Dummy_2003 + Dummy_2004 + Dummy_2005 +␣
 ↪Dummy_2006 + Dummy_2007 + Dummy_2008 + Dummy_2009 + Dummy_2010 + Dummy_2011␣
 ↪+ Dummy_2012 + Dummy_2013 + Dummy_2014 + Dummy_2015 + Dummy_2016 +␣
 ↪Dummy_2017 + Dummy_2018 + Dummy_2019 + Dummy_2020 + Dummy_2021 + Dummy_2022␣
 ↪+ Dummy_2023 + Dummy_giro_d_italia + Dummy_vuelta_a_espana + Dummy_dauphine␣
 ↪+ Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice + Dummy_tirreno_adriatico + Dummy_stagetype_1 +␣
 ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4 +␣
 ↪Dummy_stagetype_5',
                      data=stage_c[(stage_c['Cluster'] == 1) |␣
 ↪(stage_c['Cluster'] == 2) | (stage_c['Cluster'] == 3)]).fit()
print(resultS123.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   R-squared:                       0.270
Model:                            OLS   Adj. R-squared:                  0.257
Method:                 Least Squares   F-statistic:                     20.35
Date:                Tue, 15 Oct 2024   Prob (F-statistic):          3.58e-189
Time:                        17:46:05   Log-Likelihood:                -1979.7
No. Observations:                3523   AIC:                             4087.
Df Residuals:                    3459   BIC:                             4482.
Df Model:                          63
Covariance Type:            nonrobust
================================================================================
================
                                   coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                        0.4672      0.072      6.470      0.000
0.326       0.609
better_rider_around             -0.1721      0.021     -8.142      0.000
-0.214      -0.131
Teammates_behind                 0.1198      0.024      5.002      0.000
0.073       0.167
Gap_12_larger1                  -0.0334      0.020     -1.639      0.101
-0.073       0.007
Gap_23_larger1                   0.0065      0.025      0.260      0.795
-0.043       0.055
Cluster_size_teams_winner        0.0771      0.004     18.666      0.000
0.069       0.085
Cluster_size_teams_second       -0.0605      0.004    -15.943      0.000
-0.068      -0.053
Cluster_size_teams_third        -0.0026      0.005     -0.556      0.578
-0.012       0.006
Dummy_1982                       0.0966      0.094      1.031      0.303
-0.087       0.280
Dummy_1983                       0.0842      0.095      0.883      0.377
-0.103       0.271
Dummy_1985                       0.0206      0.115      0.179      0.858
-0.205       0.246
Dummy_1986                       0.0064      0.093      0.069      0.945
-0.176       0.189
Dummy_1987                       0.0984      0.095      1.040      0.298
-0.087       0.284
Dummy_1988                      -0.0331      0.117     -0.283      0.777
-0.262       0.196
Dummy_1989                      -0.0153      0.097     -0.158      0.875
-0.205       0.175
Dummy_1990                       0.0419      0.094      0.445      0.656
```

```
-0.142      0.226
Dummy_1991                      0.0311     0.089     0.347     0.728
-0.144      0.206
Dummy_1992                     -0.0130     0.114    -0.114     0.909
-0.236      0.210
Dummy_1993                      0.0739     0.092     0.806     0.420
-0.106      0.253
Dummy_1994                     -0.0079     0.110    -0.072     0.943
-0.224      0.208
Dummy_1995                      0.0384     0.087     0.440     0.660
-0.133      0.209
Dummy_1996                      0.0485     0.086     0.564     0.573
-0.120      0.217
Dummy_1997                      0.0005     0.082     0.006     0.995
-0.160      0.161
Dummy_1998                      0.0362     0.083     0.438     0.662
-0.126      0.198
Dummy_1999                      0.0217     0.077     0.282     0.778
-0.129      0.172
Dummy_2000                      0.0210     0.079     0.265     0.791
-0.134      0.176
Dummy_2001                      0.0506     0.081     0.623     0.534
-0.109      0.210
Dummy_2002                      0.0161     0.079     0.203     0.839
-0.139      0.171
Dummy_2003                      0.0299     0.081     0.372     0.710
-0.128      0.188
Dummy_2004                      0.0757     0.085     0.893     0.372
-0.090      0.242
Dummy_2005                      0.0289     0.080     0.362     0.718
-0.128      0.185
Dummy_2006                     -0.0116     0.080    -0.145     0.885
-0.168      0.145
Dummy_2007                      0.0221     0.079     0.281     0.778
-0.132      0.176
Dummy_2008                      0.0172     0.079     0.217     0.828
-0.138      0.172
Dummy_2009                     -0.0119     0.079    -0.151     0.880
-0.167      0.143
Dummy_2010                      0.0440     0.077     0.572     0.567
-0.107      0.195
Dummy_2011                      0.0165     0.084     0.195     0.845
-0.149      0.182
Dummy_2012                     -0.0058     0.079    -0.073     0.942
-0.161      0.150
Dummy_2013                      0.0481     0.082     0.585     0.559
-0.113      0.209
Dummy_2014                      0.0010     0.085     0.011     0.991
```

```
-0.165       0.167
Dummy_2015                      -0.0187      0.083     -0.225      0.822
-0.182       0.144
Dummy_2016                       0.0454      0.080      0.564      0.573
-0.112       0.203
Dummy_2017                      -0.0095      0.077     -0.123      0.902
-0.161       0.142
Dummy_2018                       0.0250      0.082      0.306      0.759
-0.135       0.185
Dummy_2019                       0.0161      0.079      0.205      0.838
-0.138       0.171
Dummy_2020                       0.0279      0.081      0.343      0.731
-0.132       0.188
Dummy_2021                      -0.0205      0.078     -0.262      0.793
-0.174       0.133
Dummy_2022                       0.0391      0.078      0.500      0.617
-0.114       0.193
Dummy_2023                       0.0573      0.080      0.719      0.472
-0.099       0.213
Dummy_giro_d_italia             -0.0042      0.025     -0.166      0.868
-0.054       0.045
Dummy_vuelta_a_espana           -0.0184      0.025     -0.748      0.454
-0.067       0.030
Dummy_dauphine                   0.0120      0.039      0.307      0.759
-0.065       0.089
Dummy_tour_de_romandie           0.0153      0.049      0.309      0.757
-0.082       0.112
Dummy_volta_a_catalunya          0.0008      0.046      0.018      0.986
-0.089       0.090
Dummy_itzulia_basque_country     0.0169      0.049      0.344      0.731
-0.080       0.113
Dummy_tour_de_suisse            -0.0178      0.038     -0.466      0.641
-0.093       0.057
Dummy_tour_de_pologne            0.0047      0.058      0.081      0.935
-0.109       0.119
Dummy_paris_nice                -0.0138      0.040     -0.346      0.729
-0.092       0.064
Dummy_tirreno_adriatico         -0.0183      0.050     -0.362      0.717
-0.117       0.081
Dummy_stagetype_1               -0.0072      0.047     -0.155      0.877
-0.099       0.084
Dummy_stagetype_2               -0.0146      0.033     -0.443      0.658
-0.079       0.050
Dummy_stagetype_3               -0.0355      0.040     -0.884      0.377
-0.114       0.043
Dummy_stagetype_4                0.0209      0.033      0.631      0.528
-0.044       0.086
Dummy_stagetype_5                0.0120      0.028      0.423      0.672
```

```
-0.044      0.068
==============================================================================
Omnibus:                     1015.551   Durbin-Watson:                   1.074
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              195.196
Skew:                           0.250   Prob(JB):                     4.11e-43
Kurtosis:                       1.961   Cond. No.                         314.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   R-squared:                       0.219
Model:                            OLS   Adj. R-squared:                  0.209
Method:                 Least Squares   F-statistic:                     21.47
Date:                Tue, 15 Oct 2024   Prob (F-statistic):         2.28e-206
Time:                        17:46:05   Log-Likelihood:                -2582.5
No. Observations:                4805   AIC:                             5291.
Df Residuals:                    4742   BIC:                             5699.
Df Model:                          62
Covariance Type:            nonrobust
================================================================================
================
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       0.3952      0.066      5.995      0.000
0.266       0.524
better_rider_around            -0.1578      0.017     -9.333      0.000
-0.191      -0.125
Gap_12_larger1                 -0.0178      0.017     -1.052      0.293
-0.051       0.015
Gap_23_larger1                  0.0027      0.021      0.131      0.896
-0.038       0.043
Cluster_size_teams_winner       0.0818      0.004     22.530      0.000
0.075       0.089
Cluster_size_teams_second      -0.0344      0.003    -10.507      0.000
-0.041      -0.028
Cluster_size_teams_third       -0.0289      0.003     -8.821      0.000
-0.035      -0.023
Dummy_1982                      0.0646      0.086      0.751      0.453
-0.104       0.233
Dummy_1983                      0.0045      0.082      0.055      0.956
-0.156       0.165
Dummy_1985                     -0.0125      0.107     -0.117      0.907
-0.222       0.197
```

```
Dummy_1986                     -0.0315      0.084     -0.376      0.707
-0.196      0.133
Dummy_1987                      0.0206      0.084      0.246      0.806
-0.143      0.185
Dummy_1988                     -0.0838      0.095     -0.879      0.380
-0.271      0.103
Dummy_1989                     -0.0737      0.086     -0.862      0.389
-0.241      0.094
Dummy_1990                     -0.0070      0.081     -0.087      0.930
-0.165      0.151
Dummy_1991                     -0.0212      0.079     -0.270      0.787
-0.176      0.133
Dummy_1992                     -0.0138      0.102     -0.135      0.893
-0.214      0.187
Dummy_1993                     -0.0076      0.081     -0.094      0.925
-0.166      0.150
Dummy_1994                     -0.0602      0.090     -0.672      0.502
-0.236      0.116
Dummy_1995                      0.0052      0.077      0.067      0.947
-0.146      0.157
Dummy_1996                     -0.0320      0.076     -0.421      0.674
-0.181      0.117
Dummy_1997                     -0.0445      0.074     -0.603      0.546
-0.189      0.100
Dummy_1998                     -0.0005      0.075     -0.006      0.995
-0.147      0.146
Dummy_1999                     -0.0295      0.070     -0.420      0.674
-0.167      0.108
Dummy_2000                     -0.0322      0.071     -0.453      0.650
-0.172      0.107
Dummy_2001                     -0.0273      0.073     -0.374      0.709
-0.171      0.116
Dummy_2002                     -0.0348      0.072     -0.484      0.629
-0.176      0.106
Dummy_2003                     -0.0479      0.071     -0.670      0.503
-0.188      0.092
Dummy_2004                      0.0174      0.076      0.229      0.819
-0.131      0.166
Dummy_2005                     -0.0341      0.072     -0.477      0.633
-0.174      0.106
Dummy_2006                     -0.0610      0.071     -0.862      0.389
-0.200      0.078
Dummy_2007                     -0.0338      0.071     -0.477      0.633
-0.173      0.105
Dummy_2008                     -0.0248      0.071     -0.350      0.726
-0.164      0.114
Dummy_2009                     -0.0437      0.071     -0.617      0.537
-0.182      0.095
```

```
Dummy_2010                        -0.0304      0.069     -0.441      0.659
-0.166        0.105
Dummy_2011                        -0.0202      0.076     -0.267      0.789
-0.169        0.128
Dummy_2012                        -0.0467      0.072     -0.650      0.516
-0.187        0.094
Dummy_2013                         0.0047      0.073      0.064      0.949
-0.139        0.148
Dummy_2014                        -0.0390      0.076     -0.516      0.606
-0.187        0.109
Dummy_2015                        -0.0689      0.074     -0.936      0.349
-0.213        0.075
Dummy_2016                        -0.0082      0.072     -0.113      0.910
-0.149        0.133
Dummy_2017                        -0.0374      0.070     -0.534      0.593
-0.175        0.100
Dummy_2018                        -0.0263      0.073     -0.357      0.721
-0.170        0.118
Dummy_2019                        -0.0445      0.071     -0.624      0.532
-0.184        0.095
Dummy_2020                        -0.0290      0.073     -0.396      0.692
-0.173        0.115
Dummy_2021                        -0.0509      0.070     -0.724      0.469
-0.189        0.087
Dummy_2022                        -0.0277      0.071     -0.393      0.694
-0.166        0.111
Dummy_2023                        -0.0070      0.072     -0.098      0.922
-0.148        0.134
Dummy_giro_d_italia                0.0013      0.021      0.064      0.949
-0.039        0.042
Dummy_vuelta_a_espana             -0.0199      0.020     -0.995      0.320
-0.059        0.019
Dummy_dauphine                    -0.0021      0.032     -0.064      0.949
-0.065        0.061
Dummy_tour_de_romandie             0.0135      0.043      0.317      0.752
-0.070        0.097
Dummy_volta_a_catalunya           -0.0012      0.038     -0.031      0.975
-0.075        0.073
Dummy_itzulia_basque_country      -0.0159      0.040     -0.395      0.693
-0.095        0.063
Dummy_tour_de_suisse              -0.0385      0.032     -1.221      0.222
-0.100        0.023
Dummy_tour_de_pologne             -0.0138      0.051     -0.272      0.786
-0.113        0.086
Dummy_paris_nice                  -0.0052      0.033     -0.160      0.873
-0.069        0.059
Dummy_tirreno_adriatico           -0.0161      0.043     -0.375      0.708
-0.100        0.068
```

```
Dummy_stagetype_1                -0.0365      0.038     -0.953      0.341
-0.112       0.039
Dummy_stagetype_2                -0.0300      0.027     -1.095      0.274
-0.084       0.024
Dummy_stagetype_3                -0.0135      0.034     -0.401      0.689
-0.080       0.053
Dummy_stagetype_4                -0.0046      0.027     -0.170      0.865
-0.058       0.049
Dummy_stagetype_5                -0.0018      0.024     -0.077      0.939
-0.048       0.044
==============================================================================
Omnibus:                      508.951   Durbin-Watson:                   0.903
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              437.329
Skew:                           0.659   Prob(JB):                     1.08e-95
Kurtosis:                       2.331   Cond. No.                         347.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[137]:
```
# Table 7: Linear Probability Model: Winning the Race from Group 1
stage_c_nsw= stage_c[stage_c['Cluster_size_teams_winner']!=1]

#LHS
resultS1x = sm.ols(formula='Win ~ better_rider_in_Cluster * Helper_in_Cluster +␣
 ↪Teammates_behind + Gap_12_larger1 + Cluster_size_teams_winner +␣
 ↪Cluster_size_teams_second + Dummy_1982 + Dummy_1983 + Dummy_1985 +␣
 ↪Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989 + Dummy_1990 + Dummy_1991␣
 ↪+ Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 + Dummy_1996 +␣
 ↪Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000 + Dummy_2001 + Dummy_2002␣
 ↪+ Dummy_2003 + Dummy_2004 + Dummy_2005 + Dummy_2006 + Dummy_2007 +␣
 ↪Dummy_2008 + Dummy_2009 + Dummy_2010 + Dummy_2011 + Dummy_2012 + Dummy_2013␣
 ↪+ Dummy_2014 + Dummy_2015 + Dummy_2016 + Dummy_2017 + Dummy_2018 +␣
 ↪Dummy_2019 + Dummy_2020 + Dummy_2021 + Dummy_2022 + Dummy_2023 +␣
 ↪Dummy_giro_d_italia + Dummy_vuelta_a_espana + Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice + Dummy_tirreno_adriatico + Dummy_stagetype_1 +␣
 ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4 +␣
 ↪Dummy_stagetype_5',
                      data=stage_c_nsw[stage_c_nsw['Cluster'] == 1]).fit()
print(resultS1x.summary())

#Middle column
```

```
resultS1 = sm.ols(formula='Win ~ better_rider_in_Cluster + Helper_in_Cluster +␣
  ↪Teammates_behind + Gap_12_larger1 + Cluster_size_teams_winner +␣
  ↪Cluster_size_teams_second + Dummy_1982 + Dummy_1983 + Dummy_1985 +␣
  ↪Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989 + Dummy_1990 + Dummy_1991␣
  ↪+ Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 + Dummy_1996 +␣
  ↪Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000 + Dummy_2001 + Dummy_2002␣
  ↪+ Dummy_2003 + Dummy_2004 + Dummy_2005 + Dummy_2006 + Dummy_2007 +␣
  ↪Dummy_2008 + Dummy_2009 + Dummy_2010 + Dummy_2011 + Dummy_2012 + Dummy_2013␣
  ↪+ Dummy_2014 + Dummy_2015 + Dummy_2016 + Dummy_2017 + Dummy_2018 +␣
  ↪Dummy_2019 + Dummy_2020 + Dummy_2021 + Dummy_2022 + Dummy_2023 +␣
  ↪Dummy_giro_d_italia + Dummy_vuelta_a_espana + Dummy_dauphine +␣
  ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
  ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
  ↪+ Dummy_paris_nice + Dummy_tirreno_adriatico + Dummy_stagetype_1 +␣
  ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4 +␣
  ↪Dummy_stagetype_5',
                   data=stage_c_nsw[stage_c_nsw['Cluster'] == 1]).fit()
print(resultS1.summary())

#RHS
#Hypothetical teams
# Note that there is randomness in how we define hypothetical teams.
# Thus, the results presented in our paper cannot be replicated perfectly.

# Filter for captains only, no teammates in front or captain in cluster
stage_c_hyp = stage[(stage["Teammates_front_hyp"] == 0) &␣
  ↪(stage["Captain_hyp_in_Cluster"] == 0)]
stage_c_hyp = stage_c_hyp[stage_c_hyp['Year'].astype(int) > 1980]  # Exclude␣
  ↪data before 1980
stage_c_hyp.loc[:, 'Cluster_size_teams_hyp'] = stage_c_hyp.
  ↪groupby(['Race_Stage', 'Cluster'])['Rider'].transform('nunique')

# Find winners and merge with stage data
winners = stage_c_hyp[stage_c_hyp['Place'] == 1][['Race_Stage',␣
  ↪'Cluster_size_teams']]
stage_c_hyp = pd.merge(stage_c_hyp, winners, on='Race_Stage', suffixes=('',␣
  ↪'_winner'), how='left')

# Find second and third cluster data and merge with stage data
second = stage_c_hyp[stage_c_hyp['Cluster'] == 2][['Race_Stage',␣
  ↪'Cluster_size_teams']]
stage_c_hyp = pd.merge(stage_c_hyp, second, on='Race_Stage', suffixes=('',␣
  ↪'_second'), how='left')

third = stage_c_hyp[stage_c_hyp['Cluster'] == 3][['Race_Stage',␣
  ↪'Cluster_size_teams']]
```

```
stage_c_hyp = pd.merge(stage_c_hyp, third, on='Race_Stage', suffixes=('',␣
 ↪'_third'), how='left')

# Create dummies for gap size and standard deviation, drop duplicates
stage_c_hyp['Gap_12_larger1'] = (stage_c_hyp['Gap_Cluster12'] >= 60).astype(int)
stage_c_hyp['Gap_23_larger1'] = (stage_c_hyp['Gap_Cluster23'] >= 60).astype(int)
stage_c_hyp = stage_c_hyp.drop_duplicates()

# Repeat the process for hypothetical data: Find winners, second, and third␣
 ↪cluster size
winners = stage_c_hyp[stage_c_hyp['Place'] == 1][['Race_Stage',␣
 ↪'Cluster_size_teams_hyp']]
stage_c_hyp = pd.merge(stage_c_hyp, winners, on='Race_Stage', suffixes=('',␣
 ↪'_winner'), how='left')

second = stage_c_hyp[stage_c_hyp['Cluster'] == 2][['Race_Stage',␣
 ↪'Cluster_size_teams_hyp']]
stage_c_hyp = pd.merge(stage_c_hyp, second, on='Race_Stage', suffixes=('',␣
 ↪'_second'), how='left')

third = stage_c_hyp[stage_c_hyp['Cluster'] == 3][['Race_Stage',␣
 ↪'Cluster_size_teams_hyp']]
stage_c_hyp = pd.merge(stage_c_hyp, third, on='Race_Stage', suffixes=('',␣
 ↪'_third'), how='left')
stage_c_hyp = stage_c_hyp.drop_duplicates()

#No solo wins
stage_c_hyp_nsw= stage_c_hyp[stage_c_hyp['Cluster_size_winner']!=1]

# Winning the race for hypothetical teams
resultHyp = sm.ols(formula='Win ~ better_rider_in_Cluster +␣
 ↪Helper_hyp_in_Cluster + Teammates_behind_hyp + Gap_12_larger1 +␣
 ↪Cluster_size_teams_hyp_winner + Cluster_size_teams_hyp_second + Dummy_1982 +␣
 ↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987 + Dummy_1989 + Dummy_1990␣
 ↪+ Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 +␣
 ↪Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000 + Dummy_2001␣
 ↪+ Dummy_2002 + Dummy_2003 + Dummy_2004 + Dummy_2005 + Dummy_2006 +␣
 ↪Dummy_2007 + Dummy_2008 + Dummy_2009 + Dummy_2010 + Dummy_2011 + Dummy_2012␣
 ↪+ Dummy_2013 + Dummy_2014 + Dummy_2015 + Dummy_2016 + Dummy_2017 +␣
 ↪Dummy_2018 + Dummy_2019 + Dummy_2020 + Dummy_2021 + Dummy_2022 + Dummy_2023␣
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana + Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice + Dummy_tirreno_adriatico + Dummy_stagetype_1 +␣
 ↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4 +␣
 ↪Dummy_stagetype_5',
                        data=stage_c_hyp_nsw[stage_c_hyp_nsw['Cluster'] == 1]).
 ↪fit()
print(resultHyp.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.097
Model:                            OLS   Adj. R-squared:                  0.048
Method:                 Least Squares   F-statistic:                     1.967
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           1.67e-05
Time:                        17:46:05   Log-Likelihood:                -722.41
No. Observations:                1212   AIC:                             1573.
Df Residuals:                    1148   BIC:                             1899.
Df Model:                          63
Covariance Type:            nonrobust
================================================================================
=============================
                                                coef    std err          t
P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
-----------------------------
Intercept                                     0.4804      0.168      2.857
0.004       0.150       0.810
better_rider_in_Cluster                      -0.0408      0.031     -1.309
0.191      -0.102       0.020
Helper_in_Cluster                             0.1222      0.074      1.658
0.098      -0.022       0.267
better_rider_in_Cluster:Helper_in_Cluster     0.1260      0.115      1.097
0.273      -0.099       0.351
Teammates_behind                              0.1385      0.044      3.147
0.002       0.052       0.225
Gap_12_larger1                                0.0144      0.040      0.355
0.722      -0.065       0.094
Cluster_size_teams_winner                    -0.0562      0.007     -8.372
0.000      -0.069      -0.043
Cluster_size_teams_second                    -0.0080      0.009     -0.868
0.386      -0.026       0.010
Dummy_1982                                   -0.0506      0.183     -0.277
0.782      -0.409       0.308
Dummy_1983                                   -0.0325      0.194     -0.168
0.867      -0.412       0.347
Dummy_1985                                    0.0064      0.222      0.029
0.977      -0.428       0.441
Dummy_1986                                    0.1726      0.208      0.829
0.407      -0.236       0.581
Dummy_1987                                    0.0421      0.191      0.221
0.825      -0.332       0.416
Dummy_1988                                    0.1129      0.362      0.312
```

```
0.755      -0.597       0.823
Dummy_1989                                           0.0681      0.224      0.304
0.761      -0.371       0.507
Dummy_1990                                           0.0344      0.193      0.178
0.858      -0.344       0.413
Dummy_1991                                           0.0277      0.192      0.144
0.885      -0.349       0.404
Dummy_1992                                           0.0095      0.216      0.044
0.965      -0.415       0.434
Dummy_1993                                           0.0183      0.200      0.092
0.927      -0.375       0.411
Dummy_1994                                           0.1128      0.249      0.453
0.650      -0.375       0.601
Dummy_1995                                           0.0607      0.198      0.306
0.759      -0.328       0.449
Dummy_1996                                           0.0228      0.194      0.118
0.906      -0.358       0.403
Dummy_1997                                           0.0223      0.190      0.117
0.907      -0.350       0.395
Dummy_1998                                           0.0280      0.183      0.153
0.879      -0.332       0.388
Dummy_1999                                           0.0755      0.174      0.435
0.664      -0.265       0.416
Dummy_2000                                           0.0151      0.184      0.082
0.934      -0.345       0.375
Dummy_2001                                           0.0170      0.189      0.090
0.928      -0.354       0.388
Dummy_2002                                           0.0410      0.187      0.220
0.826      -0.326       0.408
Dummy_2003                                           0.0008      0.196      0.004
0.997      -0.384       0.386
Dummy_2004                                          -0.0098      0.188     -0.052
0.958      -0.378       0.358
Dummy_2005                                           0.0159      0.187      0.085
0.932      -0.350       0.382
Dummy_2006                                           0.0884      0.182      0.486
0.627      -0.268       0.445
Dummy_2007                                           0.0122      0.181      0.067
0.946      -0.343       0.367
Dummy_2008                                           0.0054      0.183      0.030
0.976      -0.353       0.364
Dummy_2009                                           0.0185      0.184      0.101
0.920      -0.342       0.380
Dummy_2010                                           0.0170      0.185      0.092
0.927      -0.346       0.380
Dummy_2011                                           0.0756      0.192      0.393
0.694      -0.301       0.452
Dummy_2012                                           0.0250      0.190      0.131
```

```
0.896      -0.348       0.398
Dummy_2013                                         -0.0301      0.194     -0.155
0.877      -0.411       0.351
Dummy_2014                                         -0.0349      0.195     -0.179
0.858      -0.418       0.348
Dummy_2015                                          0.0125      0.203      0.062
0.951      -0.386       0.411
Dummy_2016                                          0.0123      0.189      0.065
0.948      -0.358       0.383
Dummy_2017                                          0.0046      0.182      0.025
0.980      -0.353       0.362
Dummy_2018                                         -0.0157      0.187     -0.084
0.933      -0.383       0.352
Dummy_2019                                         -0.0104      0.186     -0.056
0.956      -0.376       0.355
Dummy_2020                                          0.0235      0.186      0.126
0.899      -0.341       0.388
Dummy_2021                                          0.0421      0.192      0.219
0.826      -0.335       0.419
Dummy_2022                                         -0.0047      0.180     -0.026
0.979      -0.358       0.349
Dummy_2023                                          0.0101      0.182      0.056
0.956      -0.346       0.366
Dummy_giro_d_italia                                 0.0129      0.048      0.268
0.788      -0.081       0.107
Dummy_vuelta_a_espana                               0.0264      0.047      0.557
0.577      -0.066       0.119
Dummy_dauphine                                      0.0472      0.080      0.588
0.557      -0.110       0.205
Dummy_tour_de_romandie                              0.0250      0.090      0.278
0.781      -0.151       0.201
Dummy_volta_a_catalunya                             0.0107      0.086      0.125
0.901      -0.158       0.180
Dummy_itzulia_basque_country                        0.0098      0.086      0.114
0.909      -0.159       0.178
Dummy_tour_de_suisse                                0.0327      0.078      0.418
0.676      -0.121       0.186
Dummy_tour_de_pologne                               0.0383      0.120      0.319
0.749      -0.197       0.273
Dummy_paris_nice                                    0.0205      0.081      0.255
0.799      -0.137       0.179
Dummy_tirreno_adriatico                             0.0072      0.099      0.073
0.942      -0.187       0.201
Dummy_stagetype_1                                   0.0288      0.085      0.339
0.735      -0.138       0.196
Dummy_stagetype_2                                   0.0430      0.063      0.685
0.494      -0.080       0.166
Dummy_stagetype_3                                   0.1013      0.074      1.361
```

```
0.174      -0.045       0.247
Dummy_stagetype_4                            0.0244      0.062      0.393
0.695      -0.098       0.146
Dummy_stagetype_5                            0.0474      0.057      0.831
0.406      -0.064       0.159
==============================================================================
Omnibus:                      719.811   Durbin-Watson:                   0.133
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              156.326
Skew:                           0.665   Prob(JB):                     1.13e-34
Kurtosis:                       1.847   Cond. No.                         445.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.096
Model:                            OLS   Adj. R-squared:                  0.048
Method:                 Least Squares   F-statistic:                     1.979
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           1.59e-05
Time:                        17:46:05   Log-Likelihood:                -723.04
No. Observations:                1212   AIC:                             1572.
Df Residuals:                    1149   BIC:                             1893.
Df Model:                          62
Covariance Type:            nonrobust
================================================================================
================
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       0.4758      0.168      2.830      0.005
0.146       0.806
better_rider_in_Cluster        -0.0323      0.030     -1.069      0.285
-0.091       0.027
Helper_in_Cluster               0.1724      0.058      2.979      0.003
0.059       0.286
Teammates_behind                0.1379      0.044      3.134      0.002
0.052       0.224
Gap_12_larger1                  0.0136      0.040      0.336      0.737
-0.066       0.093
Cluster_size_teams_winner      -0.0563      0.007     -8.387      0.000
-0.069      -0.043
Cluster_size_teams_second      -0.0075      0.009     -0.814      0.416
-0.026       0.011
Dummy_1982                     -0.0461      0.183     -0.253      0.800
-0.404       0.312
```

```
Dummy_1983                      -0.0261      0.194     -0.135      0.893
-0.406       0.354
Dummy_1985                       0.0082      0.222      0.037      0.971
-0.427       0.443
Dummy_1986                       0.1695      0.208      0.814      0.416
-0.239       0.578
Dummy_1987                       0.0459      0.191      0.241      0.810
-0.328       0.420
Dummy_1988                       0.1134      0.362      0.313      0.754
-0.597       0.823
Dummy_1989                       0.0715      0.224      0.319      0.749
-0.368       0.510
Dummy_1990                       0.0350      0.193      0.181      0.856
-0.344       0.414
Dummy_1991                       0.0278      0.192      0.145      0.885
-0.349       0.404
Dummy_1992                       0.0091      0.216      0.042      0.966
-0.415       0.434
Dummy_1993                       0.0163      0.200      0.081      0.935
-0.377       0.409
Dummy_1994                       0.1145      0.249      0.460      0.646
-0.374       0.603
Dummy_1995                       0.0619      0.198      0.312      0.755
-0.327       0.451
Dummy_1996                       0.0208      0.194      0.107      0.914
-0.360       0.401
Dummy_1997                       0.0233      0.190      0.123      0.902
-0.349       0.396
Dummy_1998                       0.0316      0.183      0.172      0.863
-0.328       0.391
Dummy_1999                       0.0740      0.174      0.426      0.670
-0.267       0.415
Dummy_2000                       0.0131      0.184      0.071      0.943
-0.347       0.373
Dummy_2001                       0.0142      0.189      0.075      0.940
-0.357       0.386
Dummy_2002                       0.0428      0.187      0.229      0.819
-0.324       0.409
Dummy_2003                       0.0025      0.196      0.013      0.990
-0.383       0.388
Dummy_2004                      -0.0094      0.188     -0.050      0.960
-0.378       0.359
Dummy_2005                       0.0179      0.187      0.096      0.924
-0.348       0.384
Dummy_2006                       0.0912      0.182      0.502      0.616
-0.265       0.448
Dummy_2007                       0.0126      0.181      0.070      0.944
-0.342       0.368
```

```
Dummy_2008                       0.0048      0.183      0.026      0.979
-0.353       0.363
Dummy_2009                       0.0193      0.184      0.105      0.916
-0.342       0.380
Dummy_2010                       0.0151      0.185      0.082      0.935
-0.348       0.378
Dummy_2011                       0.0767      0.192      0.399      0.690
-0.300       0.454
Dummy_2012                       0.0243      0.190      0.128      0.898
-0.349       0.398
Dummy_2013                      -0.0261      0.194     -0.134      0.893
-0.407       0.355
Dummy_2014                      -0.0306      0.195     -0.157      0.875
-0.414       0.352
Dummy_2015                       0.0204      0.203      0.101      0.920
-0.377       0.418
Dummy_2016                       0.0110      0.189      0.058      0.953
-0.359       0.382
Dummy_2017                       0.0091      0.182      0.050      0.960
-0.348       0.366
Dummy_2018                      -0.0115      0.187     -0.062      0.951
-0.379       0.356
Dummy_2019                      -0.0100      0.186     -0.054      0.957
-0.376       0.356
Dummy_2020                       0.0208      0.186      0.112      0.911
-0.344       0.385
Dummy_2021                       0.0444      0.192      0.231      0.817
-0.333       0.421
Dummy_2022                      -0.0024      0.180     -0.014      0.989
-0.356       0.351
Dummy_2023                       0.0122      0.182      0.067      0.946
-0.344       0.368
Dummy_giro_d_italia              0.0138      0.048      0.287      0.774
-0.080       0.108
Dummy_vuelta_a_espana            0.0273      0.047      0.577      0.564
-0.066       0.120
Dummy_dauphine                   0.0490      0.080      0.609      0.543
-0.109       0.207
Dummy_tour_de_romandie           0.0239      0.090      0.266      0.790
-0.152       0.200
Dummy_volta_a_catalunya          0.0088      0.086      0.102      0.919
-0.160       0.178
Dummy_itzulia_basque_country     0.0109      0.086      0.127      0.899
-0.158       0.179
Dummy_tour_de_suisse             0.0342      0.078      0.438      0.662
-0.119       0.188
Dummy_tour_de_pologne            0.0386      0.120      0.322      0.748
-0.197       0.274
```

```
Dummy_paris_nice                0.0193      0.081      0.240      0.810
-0.139       0.177
Dummy_tirreno_adriatico         0.0136      0.099      0.138      0.890
-0.180       0.208
Dummy_stagetype_1               0.0309      0.085      0.364      0.716
-0.136       0.198
Dummy_stagetype_2               0.0417      0.063      0.664      0.507
-0.082       0.165
Dummy_stagetype_3               0.0957      0.074      1.289      0.197
-0.050       0.241
Dummy_stagetype_4               0.0248      0.062      0.399      0.690
-0.097       0.147
Dummy_stagetype_5               0.0469      0.057      0.822      0.411
-0.065       0.159
==============================================================================
Omnibus:                      716.487   Durbin-Watson:                   0.131
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              157.484
Skew:                           0.669   Prob(JB):                     6.35e-35
Kurtosis:                       1.848   Cond. No.                         445.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.103
Model:                            OLS   Adj. R-squared:                  0.049
Method:                 Least Squares   F-statistic:                     1.904
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           5.73e-05
Time:                        17:46:06   Log-Likelihood:                -652.86
No. Observations:                1071   AIC:                             1430.
Df Residuals:                    1009   BIC:                             1738.
Df Model:                          61
Covariance Type:            nonrobust
================================================================================
=================
                                    coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-----------------
Intercept                         0.5931      0.155      3.831      0.000
0.289       0.897
better_rider_in_Cluster          -0.0203      0.033     -0.607      0.544
-0.086       0.045
Helper_hyp_in_Cluster             0.2479      0.059      4.187      0.000
0.132       0.364
Teammates_behind_hyp              0.1032      0.055      1.889      0.059
```

```
-0.004       0.210
Gap_12_larger1                  0.0128      0.044       0.291       0.771
-0.073       0.099
Cluster_size_teams_hyp_winner  -0.0755      0.009      -8.173       0.000
-0.094      -0.057
Cluster_size_teams_hyp_second  -0.0026      0.011      -0.248       0.804
-0.023       0.018
Dummy_1982                      0.0416      0.173       0.241       0.810
-0.298       0.381
Dummy_1983                     -0.0486      0.182      -0.266       0.790
-0.406       0.309
Dummy_1985                     -0.0349      0.212      -0.165       0.869
-0.451       0.381
Dummy_1986                      0.0239      0.254       0.094       0.925
-0.475       0.523
Dummy_1987                      0.0373      0.178       0.210       0.834
-0.312       0.386
Dummy_1989                      0.0164      0.230       0.071       0.943
-0.435       0.468
Dummy_1990                     -0.0013      0.180      -0.007       0.994
-0.355       0.352
Dummy_1991                     -0.0226      0.186      -0.122       0.903
-0.387       0.342
Dummy_1992                     -0.0186      0.246      -0.076       0.940
-0.502       0.465
Dummy_1993                     -0.0204      0.194      -0.105       0.916
-0.401       0.360
Dummy_1994                      0.0559      0.275       0.204       0.839
-0.483       0.595
Dummy_1995                      0.0476      0.243       0.196       0.845
-0.430       0.525
Dummy_1996                     -0.0068      0.187      -0.036       0.971
-0.373       0.359
Dummy_1997                     -0.0264      0.179      -0.148       0.883
-0.378       0.325
Dummy_1998                      0.0011      0.173       0.007       0.995
-0.337       0.340
Dummy_1999                      0.0524      0.162       0.323       0.746
-0.266       0.370
Dummy_2000                     -0.0209      0.172      -0.122       0.903
-0.358       0.316
Dummy_2001                     -0.0083      0.178      -0.047       0.963
-0.357       0.340
Dummy_2002                      0.0056      0.175       0.032       0.974
-0.338       0.349
Dummy_2003                     -0.0125      0.190      -0.066       0.948
-0.386       0.361
Dummy_2004                      0.0114      0.177       0.064       0.949
```

```
-0.336       0.359
Dummy_2005                        0.0043      0.172      0.025      0.980
-0.333       0.342
Dummy_2006                        0.0372      0.171      0.218      0.828
-0.298       0.372
Dummy_2007                        0.0313      0.168      0.187      0.852
-0.297       0.360
Dummy_2008                       -0.0398      0.170     -0.234      0.815
-0.373       0.294
Dummy_2009                       -0.0209      0.179     -0.116      0.907
-0.373       0.331
Dummy_2010                       -0.0277      0.171     -0.162      0.871
-0.363       0.307
Dummy_2011                        0.0071      0.195      0.036      0.971
-0.375       0.389
Dummy_2012                        0.0419      0.178      0.236      0.814
-0.307       0.391
Dummy_2013                       -0.0111      0.182     -0.061      0.951
-0.367       0.345
Dummy_2014                       -0.0087      0.184     -0.047      0.962
-0.370       0.353
Dummy_2015                       -0.0248      0.196     -0.127      0.899
-0.409       0.359
Dummy_2016                        0.0183      0.175      0.105      0.917
-0.325       0.362
Dummy_2017                       -0.0104      0.169     -0.062      0.951
-0.341       0.321
Dummy_2018                        0.0187      0.176      0.106      0.916
-0.327       0.364
Dummy_2019                       -0.0031      0.177     -0.018      0.986
-0.351       0.344
Dummy_2020                        0.0308      0.185      0.166      0.868
-0.332       0.394
Dummy_2021                        0.0129      0.180      0.072      0.943
-0.340       0.366
Dummy_2022                       -0.0185      0.168     -0.110      0.912
-0.348       0.311
Dummy_2023                        0.0325      0.169      0.192      0.848
-0.300       0.365
Dummy_giro_d_italia              -0.0074      0.051     -0.144      0.885
-0.108       0.093
Dummy_vuelta_a_espana             0.0158      0.050      0.318      0.751
-0.082       0.113
Dummy_dauphine                   -0.0014      0.083     -0.017      0.987
-0.163       0.161
Dummy_tour_de_romandie           -0.0220      0.098     -0.223      0.823
-0.215       0.171
Dummy_volta_a_catalunya          -0.0118      0.088     -0.134      0.894
```

```
-0.185       0.161
Dummy_itzulia_basque_country      0.0224      0.096      0.234      0.815
-0.166       0.210
Dummy_tour_de_suisse             -0.0092      0.083     -0.111      0.912
-0.171       0.153
Dummy_tour_de_pologne             0.0051      0.148      0.035      0.972
-0.285       0.295
Dummy_paris_nice                  0.0267      0.083      0.323      0.746
-0.135       0.189
Dummy_tirreno_adriatico           0.0246      0.101      0.243      0.808
-0.174       0.224
Dummy_stagetype_1                 0.0054      0.091      0.059      0.953
-0.174       0.185
Dummy_stagetype_2                 0.0006      0.069      0.009      0.993
-0.134       0.135
Dummy_stagetype_3                 0.0334      0.082      0.405      0.686
-0.128       0.195
Dummy_stagetype_4                -0.0242      0.067     -0.363      0.717
-0.155       0.107
Dummy_stagetype_5                 0.0138      0.060      0.230      0.818
-0.104       0.132
==============================================================================
Omnibus:                     1086.678   Durbin-Watson:                   0.155
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              132.929
Skew:                           0.605   Prob(JB):                     1.36e-29
Kurtosis:                       1.770   Cond. No.                         331.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

2 Appendix

2.1 Summary Statistics

[138]:
```
# Table C.9: Summary Statistics Non-Dummies

# Create DataFrame 'races' for the calculations
races = pd.DataFrame()

# Calculate the mean for each race and each metric
races['Gap_Cluster12'] = stage.groupby(['Race_Stage'])['Gap_Cluster12'].mean()
races['Gap_Cluster23'] = stage.groupby(['Race_Stage'])['Gap_Cluster23'].mean()
races['Cluster_size_winner'] = stage.
  ↪groupby(['Race_Stage'])['Cluster_size_winner'].mean()
```

```
races['Cluster_size_second'] = stage.
  ↪groupby(['Race_Stage'])['Cluster_size_second'].mean()
races['Cluster_size_third'] = stage.
  ↪groupby(['Race_Stage'])['Cluster_size_third'].mean()

# Combine the statistics into one DataFrame
summary_stats = pd.DataFrame({
    'Group 1 size': races['Cluster_size_winner'],
    'Group 2 size': races['Cluster_size_second'],
    'Group 3 size': races['Cluster_size_third'],
    'Gap between Groups 1 and 2': races['Gap_Cluster12'],
    'Gap between Groups 2 and 3': races['Gap_Cluster23']
})

# Use the .describe() function and filter for the relevant stats (mean, std,␣
  ↪min, 50%, max)
summary_stats = summary_stats.describe().loc[['mean', 'std', 'min', '50%',␣
  ↪'max']]

# Rename index values to match your desired output
summary_stats.index = ['mean', 'std', 'min', '50%', 'max']

# Convert to LaTeX format and print
print(summary_stats.to_latex(index=True, float_format="%.2f"))
```

```
\begin{tabular}{lrrrrr}
\toprule
 & Group 1 size & Group 2 size & Group 3 size & Gap between Groups 1 and 2 & Gap
between Groups 2 and 3 \\
\midrule
mean & 2.26 & 3.16 & 2.36 & 40.66 & 33.35 \\
std & 1.91 & 2.27 & 1.97 & 60.58 & 69.64 \\
min & 1.00 & 1.00 & 1.00 & 5.00 & 5.00 \\
50% & 2.00 & 2.00 & 1.00 & 22.00 & 14.00 \\
max & 12.00 & 12.00 & 11.00 & 853.00 & 1155.00 \\
\bottomrule
\end{tabular}
```

[139]:
```
# Table C.10: Mean occurrence of Dummies

cl_one = stage[stage['Cluster'] == 1]
cl_two = stage[stage['Cluster'] == 2]
cl_three = stage[stage['Cluster'] == 3]

# Calculate mean values for each group
```

```
group1_mean = cl_one[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()
group2_mean = cl_two[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()
group3_mean = cl_three[['Star', 'better_rider_in_Cluster',␣
 ↪'Helper_in_Cluster']].mean()

# Calculate overall mean (across all groups)
overall_mean = stage[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()

# Create DataFrame for the table
mean_occurrence = pd.DataFrame({
    'Group 1': group1_mean,
    'Group 2': group2_mean,
    'Group 3': group3_mean,
    'Overall': overall_mean
}).T

# Replace NaN values with 0.0 instead of an empty string (for compatibility)
mean_occurrence.fillna(0.0, inplace=True)

# Convert to LaTeX format
print(mean_occurrence[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].to_latex(index=True, float_format="%.3f"))
```

```
\begin{tabular}{lrrrr}
\toprule
 & Star & better_rider_in_Cluster & Helper_in_Cluster & Teammates_behind \\
\midrule
Group 1 & 0.284 & 0.262 & 0.051 & 0.122 \\
Group 2 & 0.269 & 0.351 & 0.066 & 0.093 \\
Group 3 & 0.193 & 0.271 & 0.053 & 0.000 \\
Overall & 0.251 & 0.301 & 0.058 & 0.108 \\
\bottomrule
\end{tabular}
```

2.2 In-race data

[140]:
```
# Load and clean data
km = pd.read_excel('1km_stats.xlsx')
km = km.drop(km.columns[0], axis=1).drop_duplicates()

# Calculate and print race summary
same = (km['same'] == 'y').sum()
total_races = len(km['same'])
```

```
share_same = same / total_races

# Convert rider and team columns to lists
for col in ['km_Riders_in_Cluster1', 'km_Riders_in_Cluster2',␣
 ↪'km_Riders_in_Cluster3', 'km_Teams_in_Cluster1', 'km_Teams_in_Cluster2',␣
 ↪'km_Teams_in_Cluster3']:
    km[col] = km[col].str.split(", ")

# Melt DataFrame for Riders and Teams
riders_df = km.melt(id_vars=['Race_Stage'],␣
 ↪value_vars=['km_Riders_in_Cluster1', 'km_Riders_in_Cluster2',␣
 ↪'km_Riders_in_Cluster3'], var_name='km_Cluster', value_name='Rider')
teams_df = km.melt(id_vars=['Race_Stage'], value_vars=['km_Teams_in_Cluster1',␣
 ↪'km_Teams_in_Cluster2', 'km_Teams_in_Cluster3'], var_name='km_Cluster',␣
 ↪value_name='Team')

# Function to extract the cluster number
def extract_last_number(s):
    numbers = [int(num) for num in re.findall(r'\d+', s)]
    return numbers[-1] if numbers else None

# Apply cluster extraction
riders_df['km_Cluster'] = riders_df['km_Cluster'].apply(extract_last_number).
 ↪astype(int)
teams_df['km_Cluster'] = teams_df['km_Cluster'].apply(extract_last_number).
 ↪astype(int)

# Merge Riders and Teams
merged_df = pd.merge(riders_df, teams_df, on=['Race_Stage', 'km_Cluster'])
df_1km = merged_df.explode(['Rider', 'Team'])

# Merge additional columns and sort
df_1km = pd.merge(df_1km, km[['Race_Stage', 'same', 'km_Cluster_size_winner',␣
 ↪'km_Cluster_size_second', 'km_Cluster_size_third']], on='Race_Stage',␣
 ↪how='left')
df_1km = df_1km.sort_values(by=['Race_Stage', 'km_Cluster']).
 ↪reset_index(drop=True)

# Merge with stage data
df = pd.merge(df_1km, stage[['Race_Stage', 'Rider', 'Cluster', 'Win', 'Place',␣
 ↪'Teammates_behind', 'Teammates_front',
                              'Helper_in_Cluster', 'Captain_in_Cluster',␣
 ↪'Star', 'Cluster_size_teams_winner',
                              'Cluster_size_teams_second',␣
 ↪'Cluster_size_teams_third', 'Dummy_Cluster_1',
                                'Gap_12_larger1', 'Gap_23_larger1',␣
↪'better_rider_around',
                                'better_rider_in_Cluster', 'Dummy_2020',␣
↪'Dummy_2021', 'Dummy_2022', 'Dummy_2023']],
                    on=['Race_Stage', 'Rider'], how='left')

# Initialize new columns
df['km_Helper_in_Cluster'] = 0
df['km_Captain_in_Cluster'] = 0
df['km_Teammates_front'] = 0
df['km_Teammates_behind'] = 0

# Update teammate/helper information based on clusters
for group_name in df['Race_Stage'].unique():
    group_data = df[df['Race_Stage'] == group_name]
    for i in range(len(group_data)):
        for j in range(i + 1, len(group_data)):
            if (df.loc[group_data.index[i], 'km_Cluster'] == df.loc[group_data.
↪index[j], 'km_Cluster']) and \
                (df.loc[group_data.index[i], 'Team'] == df.loc[group_data.
↪index[j], 'Team']):
                df.loc[group_data.index[i], 'km_Helper_in_Cluster'] = 1
                df.loc[group_data.index[j], 'km_Captain_in_Cluster'] = 1
            if (df.loc[group_data.index[i], 'km_Cluster'] + 1 == df.
↪loc[group_data.index[j], 'km_Cluster']) and \
                ((df.loc[group_data.index[i], 'Team'] == df.loc[group_data.
↪index[j], 'Team']) or (df.loc[group_data.index[j], 'Team'] == 'peloton')):
                df.loc[group_data.index[i], 'km_Teammates_behind'] = 1
                df.loc[group_data.index[j], 'km_Teammates_front'] = 1

# Count various scenarios for winners and non-winners
winner_tb = len(df[(df['Place'] == 1) & (df['Teammates_behind'] == 1)])
nonwinner_tb = len(df[(df['Cluster'] == 1) & (df['Place'] != 1) &␣
↪(df['Teammates_behind'] == 1)])
winner_tb_1km = len(df[(df['Place'] == 1) & (df['km_Teammates_behind'] == 1)])
nonwinner_tb_1km = len(df[(df['Cluster'] == 1) & (df['Place'] != 1) &␣
↪(df['km_Teammates_behind'] == 1)])
helper_turns_tb = len(df[(df['Cluster'] == 1) & (df['km_Cluster'] == 1) &␣
↪(df['Teammates_behind'] == 1) & (df['km_Helper_in_Cluster'] == 1)])
tb_turns_helper = len(df[(df['Cluster'] == 1) & (df['km_Cluster'] == 1) &␣
↪(df['km_Teammates_behind'] == 1) & (df['Helper_in_Cluster'] == 1)])
winner_gets_tb = len(df[(df['Place'] == 1) & (df['Teammates_behind'] == 1) &␣
↪(df['km_Teammates_behind'] != 1)])
nonwinner_gets_tb = len(df[(df['Place'] != 1) & (df['Cluster'] == 1) &␣
↪(df['Teammates_behind'] == 1) & (df['km_Teammates_behind'] != 1)])
```

```
# Print summary statistics
print(f"The share of races where there is no change between the finish and 1km␣
  ↪before is: {share_same:.2%} ({same} out of {total_races}).")
print(f"In our original dataset, the winner had a teammate behind in␣
  ↪{winner_tb} cases, while non-winners in Cluster 1 had a teammate behind in␣
  ↪{nonwinner_tb} cases.")
print(f"At 1km before the finish, the winner had a teammate behind in␣
  ↪{winner_tb_1km} cases, and non-winners in Cluster 1 had a teammate behind in␣
  ↪{nonwinner_tb_1km} cases.")
print(f"There are {helper_turns_tb} cases where a Cluster 1 rider had a␣
  ↪teammate behind, who was a helper 1km before. In {tb_turns_helper} cases,␣
  ↪the reverse occurred.")
print(f"Additionally, {winner_gets_tb} winners had a teammate behind at the␣
  ↪finish but not 1km before, while {nonwinner_gets_tb} non-winners in Cluster␣
  ↪1 experienced the same.")

# Filter non-solo wins and run regressions
df_c = df[(df['Teammates_front'] == 0) & (df['Captain_in_Cluster'] == 0)].
  ↪drop_duplicates()
df_c_nsw = df_c[df_c['Cluster_size_teams_winner'] != 1]

# Regression analysis

# Table C.11: Linear Probability Model: Finishing in Group 1 - In-Race Data
resultS12 = sm.ols('Dummy_Cluster_1 ~ better_rider_around + km_Teammates_behind␣
  ↪+ Gap_12_larger1 + Gap_23_larger1 + Cluster_size_teams_winner +␣
  ↪Cluster_size_teams_second + Cluster_size_teams_third + Dummy_2021 +␣
  ↪Dummy_2022 + Dummy_2023', data=df_c[(df_c['Cluster'] == 1) |␣
  ↪(df_c['Cluster'] == 2)]).fit()
print(resultS12.summary())

# Table C.12: Linear Probability Model: Winning the Race from Group 1 - In-Race␣
  ↪Data
resultS1 = sm.ols('Win ~ better_rider_in_Cluster + km_Helper_in_Cluster +␣
  ↪km_Teammates_behind + Gap_12_larger1 + Cluster_size_teams_winner +␣
  ↪Cluster_size_teams_second + Dummy_2021 + Dummy_2022 + Dummy_2023',␣
  ↪data=df_c_nsw[df_c_nsw['Cluster'] == 1]).fit()
print(resultS1.summary())
```

```
The share of races where there is no change between the finish and 1km before
is: 53.47% (54 out of 101).
In our original dataset, the winner had a teammate behind in 16 cases, while
non-winners in Cluster 1 had a teammate behind in 9 cases.
At 1km before the finish, the winner had a teammate behind in 17 cases, and non-
winners in Cluster 1 had a teammate behind in 10 cases.
There are 5 cases where a Cluster 1 rider had a teammate behind, who was a
helper 1km before. In 1 cases, the reverse occurred.
```

Additionally, 2 winners had a teammate behind at the finish but not 1km before, while 3 non-winners in Cluster 1 experienced the same.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   R-squared:                       0.293
Model:                            OLS   Adj. R-squared:                  0.278
Method:                 Least Squares   F-statistic:                     19.43
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           5.09e-30
Time:                        17:46:06   Log-Likelihood:                -263.99
No. Observations:                 480   AIC:                             550.0
Df Residuals:                     469   BIC:                             595.9
Df Model:                          10
Covariance Type:            nonrobust
================================================================================
=============
                               coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------
Intercept                    0.5279      0.078      6.779      0.000
0.375       0.681
better_rider_around         -0.1507      0.058     -2.577      0.010
-0.266      -0.036
km_Teammates_behind          0.0231      0.059      0.393      0.694
-0.092       0.139
Gap_12_larger1              -0.1014      0.054     -1.878      0.061
-0.207       0.005
Gap_23_larger1               0.0771      0.078      0.989      0.323
-0.076       0.230
Cluster_size_teams_winner    0.0744      0.010      7.279      0.000
0.054       0.094
Cluster_size_teams_second   -0.0595      0.010     -6.036      0.000
-0.079      -0.040
Cluster_size_teams_third    -0.0090      0.014     -0.654      0.513
-0.036       0.018
Dummy_2021                  -0.0527      0.062     -0.853      0.394
-0.174       0.069
Dummy_2022                  -0.0077      0.063     -0.123      0.902
-0.131       0.115
Dummy_2023                   0.0235      0.064      0.367      0.714
-0.102       0.149
==============================================================================
Omnibus:                       94.216   Durbin-Watson:                   2.029
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               26.147
Skew:                           0.293   Prob(JB):                     2.10e-06
Kurtosis:                       2.018   Cond. No.                         28.9
==============================================================================
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.126
Model:                            OLS   Adj. R-squared:                  0.079
Method:                 Least Squares   F-statistic:                     2.686
Date:                Tue, 15 Oct 2024   Prob (F-statistic):            0.00609
Time:                        17:46:06   Log-Likelihood:                -101.32
No. Observations:                 178   AIC:                             222.6
Df Residuals:                     168   BIC:                             254.5
Df Model:                           9
Covariance Type:            nonrobust
================================================================================
=============
                               coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------
Intercept                    0.5426      0.120      4.516      0.000
0.305       0.780
better_rider_in_Cluster     -0.1412      0.081     -1.736      0.084
-0.302       0.019
km_Helper_in_Cluster         0.0717      0.135      0.531      0.596
-0.195       0.338
km_Teammates_behind          0.2621      0.115      2.281      0.024
0.035       0.489
Gap_12_larger1               0.0263      0.121      0.218      0.828
-0.212       0.264
Cluster_size_teams_winner   -0.0458      0.016     -2.879      0.005
-0.077      -0.014
Cluster_size_teams_second   -0.0109      0.020     -0.535      0.593
-0.051       0.029
Dummy_2021                  -0.0009      0.119     -0.008      0.994
-0.235       0.234
Dummy_2022                  -0.0536      0.108     -0.496      0.621
-0.267       0.160
Dummy_2023                   0.0020      0.109      0.018      0.986
-0.213       0.217
==============================================================================
Omnibus:                       40.453   Durbin-Watson:                   3.107
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               20.498
Skew:                           0.667   Prob(JB):                     3.54e-05
Kurtosis:                       2.008   Cond. No.                         32.4
==============================================================================

Notes:
```

```
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

2.3 Additional analyses

2.3.1 Fewer Stars

[141]:
```
# Table C.13: Linear Probability Model: Finishing in Group 1 - Fewer Stars
# Table C.14: Linear Probability Model: Winning the Race from Group 1 - Fewer␣
 ↪Stars

#Change Section "Stars"
# Here, do the same analysis as for Tables 6 and 7 after adjusting the␣
 ↪definition of STARS in the following way:

# Loop through years to create dummy variables for each year and identify stars
for i in range(1981, 2024):
    stage[f'Dummy_{i}'] = 0
    stage.loc[stage['Year'] == i, f'Dummy_{i}'] = 1
    threshold_up = stage.loc[stage['Year'] == i, 'Score'].quantile(0.90)

    stage.loc[stage['Year'] == i, 'Star'] = (stage.loc[stage['Year'] == i,␣
 ↪'Score'] >= threshold_up).astype(int)
    stage.loc[stage['Year'] == i, 'not_a_Star'] = (stage.loc[stage['Year'] ==␣
 ↪i, 'Score'] < threshold_up).astype(int)

# other than that, use the same code as above
```

2.3.2 Smaller Groups

[142]:
```
# Table C.15: Linear Probability Model: Finishing in Group 1 - Smaller Groups
# Table C.16: Linear Probability Model: Winning the Race from Group 1 - Smaller␣
 ↪Groups

#Change Section "Groups"
# Here, do the same analysis as for Tables 6 and 7 after adjusting the␣
 ↪definition of GROUPS in the following way:

# Iterate through unique race stages to define clusters and teammate roles
for group_name in stage['Race_Stage'].unique():
    group_data = stage[stage['Race_Stage'] == group_name]

    # Define clusters based on time gaps
    for i in range(1, len(group_data)):
        gap_difference = group_data.iloc[i]['Gap'] - group_data.iloc[i -␣
 ↪1]['Gap']
        if gap_difference > 0:  # 1+ seconds gap creates a new cluster
```

```
            stage.loc[group_data.index[i], 'Cluster'] = stage.loc[group_data.
↪index[i - 1], 'Cluster'] + 1
            stage.loc[group_data.index[i], 'Gap_front'] = gap_difference
        else:
            stage.loc[group_data.index[i], 'Gap_front'] = 0
            stage.loc[group_data.index[i], 'Cluster'] = stage.loc[group_data.
↪index[i - 1], 'Cluster']

    # Update cluster sizes and teammate information
    for i in range(len(group_data)):
        for j in range(i + 1, len(group_data)):
            same_cluster = stage.loc[group_data.index[i], 'Cluster'] == stage.
↪loc[group_data.index[j], 'Cluster']
            same_team = stage.loc[group_data.index[i], 'Team'] == stage.
↪loc[group_data.index[j], 'Team']
            if same_cluster:
                stage.loc[group_data.index[i], 'Cluster_size'] += 1
                stage.loc[group_data.index[j], 'Cluster_size'] += 1
                if same_team:
                    # Mark teammates in same cluster
                    stage.loc[group_data.index[i], 'Helper_in_Cluster'] = 1
                    stage.loc[group_data.index[j], 'Captain_in_Cluster'] = 1
                    stage.loc[group_data.index[i], 'Teammates'] = 1
                    stage.loc[group_data.index[j], 'Teammates'] = 1
                else:
                    # Non-teammates in the same cluster
                    pass

            # Mark teammates in neighboring clusters
            if same_team and stage.loc[group_data.index[i], 'Cluster'] + 1 ==␣
↪stage.loc[group_data.index[j], 'Cluster']:
                stage.loc[group_data.index[i], 'Teammates_behind'] = 1
                stage.loc[group_data.index[j], 'Teammates_front'] = 1

            if same_team and stage.loc[group_data.index[i], 'Cluster'] + 2 ==␣
↪stage.loc[group_data.index[j], 'Cluster']:
                stage.loc[group_data.index[j], 'Teammates_front'] = 1

            # Hypothetical teammates in same cluster
            if same_cluster and stage.loc[group_data.index[i], 'hyp_team'] ==␣
↪stage.loc[group_data.index[j], 'hyp_team']:
                stage.loc[group_data.index[i], 'Helper_hyp_in_Cluster'] = 1
                stage.loc[group_data.index[j], 'Captain_hyp_in_Cluster'] = 1

            # Hypothetical teammates in neighboring clusters
            if stage.loc[group_data.index[i], 'Cluster'] + 1 == stage.
↪loc[group_data.index[j], 'Cluster'] and stage.loc[group_data.index[i],␣
↪'hyp_team'] == stage.loc[group_data.index[j], 'hyp_team']:
                stage.loc[group_data.index[i], 'Teammates_behind_hyp'] = 1
                stage.loc[group_data.index[j], 'Teammates_front_hyp'] = 1

# other than that, use the same code as above
```

2.3.3 Hypothetical teammates

[143]:
```
# Table C.17: Linear Probability Model: Winning the Race from Group 1

# Only the RHS is new (for LHS, see Table 7)
resultHypx = sm.ols(formula='Win ~ better_rider_in_Cluster *␣
↪Helper_hyp_in_Cluster + Teammates_behind_hyp + Gap_12_larger1 +␣
↪Cluster_size_teams_hyp_winner + Cluster_size_teams_hyp_second + Dummy_1982 +␣
↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987  + Dummy_1989 + Dummy_1990␣
↪+ Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 +␣
↪Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000 + Dummy_2001␣
↪+ Dummy_2002 + Dummy_2003 + Dummy_2004 + Dummy_2005 + Dummy_2006 +␣
↪Dummy_2007 + Dummy_2008 + Dummy_2009 + Dummy_2010 + Dummy_2011 + Dummy_2012␣
↪+ Dummy_2013 + Dummy_2014 + Dummy_2015 + Dummy_2016 + Dummy_2017 +␣
↪Dummy_2018 + Dummy_2019 + Dummy_2020 + Dummy_2021 + Dummy_2022 + Dummy_2023␣
↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana + Dummy_dauphine +␣
↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
↪+ Dummy_paris_nice + Dummy_tirreno_adriatico + Dummy_stagetype_1 +␣
↪Dummy_stagetype_2 + Dummy_stagetype_3 + Dummy_stagetype_4 +␣
↪Dummy_stagetype_5',
                      data=stage_c_hyp_nsw[stage_c_hyp_nsw['Cluster'] == 1]).
↪fit()
print(resultHypx.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.103
Model:                            OLS   Adj. R-squared:                  0.048
Method:                 Least Squares   F-statistic:                     1.872
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           8.07e-05
Time:                        17:46:08   Log-Likelihood:                -652.86
No. Observations:                1071   AIC:                             1432.
Df Residuals:                    1008   BIC:                             1745.
Df Model:                          62
Covariance Type:            nonrobust
================================================================================
=================================
                                                   coef    std err          t
```

```
P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
---------------------------------
Intercept                                      0.5931      0.155      3.828
0.000       0.289       0.897
better_rider_in_Cluster                       -0.0204      0.035     -0.590
0.556      -0.088       0.047
Helper_hyp_in_Cluster                          0.2475      0.076      3.253
0.001       0.098       0.397
better_rider_in_Cluster:Helper_hyp_in_Cluster  0.0009      0.120      0.007
0.994      -0.235       0.236
Teammates_behind_hyp                           0.1032      0.055      1.887
0.059      -0.004       0.211
Gap_12_larger1                                 0.0128      0.044      0.291
0.771      -0.073       0.099
Cluster_size_teams_hyp_winner                 -0.0755      0.009     -8.164
0.000      -0.094      -0.057
Cluster_size_teams_hyp_second                 -0.0026      0.011     -0.248
0.804      -0.023       0.018
Dummy_1982                                     0.0416      0.173      0.240
0.810      -0.298       0.381
Dummy_1983                                    -0.0485      0.182     -0.266
0.790      -0.406       0.309
Dummy_1985                                    -0.0350      0.212     -0.165
0.869      -0.451       0.381
Dummy_1986                                     0.0239      0.254      0.094
0.925      -0.475       0.523
Dummy_1987                                     0.0374      0.178      0.210
0.834      -0.312       0.387
Dummy_1989                                     0.0164      0.230      0.071
0.943      -0.435       0.468
Dummy_1990                                    -0.0013      0.180     -0.007
0.994      -0.355       0.353
Dummy_1991                                    -0.0226      0.186     -0.121
0.903      -0.388       0.343
Dummy_1992                                    -0.0187      0.246     -0.076
0.940      -0.502       0.465
Dummy_1993                                    -0.0203      0.194     -0.105
0.916      -0.401       0.360
Dummy_1994                                     0.0559      0.275      0.203
0.839      -0.483       0.595
Dummy_1995                                     0.0476      0.243      0.196
0.845      -0.430       0.525
Dummy_1996                                    -0.0068      0.187     -0.036
0.971      -0.373       0.359
Dummy_1997                                    -0.0264      0.179     -0.148
0.883      -0.378       0.325
Dummy_1998                                     0.0011      0.173      0.007
```

```
0.995      -0.338        0.340
Dummy_1999                                          0.0524      0.162      0.323
0.746      -0.266        0.371
Dummy_2000                                         -0.0208      0.172     -0.121
0.904      -0.358        0.316
Dummy_2001                                         -0.0083      0.178     -0.046
0.963      -0.357        0.340
Dummy_2002                                          0.0056      0.175      0.032
0.974      -0.338        0.349
Dummy_2003                                         -0.0125      0.190     -0.066
0.948      -0.386        0.361
Dummy_2004                                          0.0114      0.177      0.064
0.949      -0.336        0.359
Dummy_2005                                          0.0043      0.172      0.025
0.980      -0.333        0.342
Dummy_2006                                          0.0372      0.171      0.218
0.828      -0.298        0.372
Dummy_2007                                          0.0313      0.168      0.187
0.852      -0.298        0.360
Dummy_2008                                         -0.0398      0.170     -0.234
0.815      -0.373        0.294
Dummy_2009                                         -0.0209      0.179     -0.116
0.907      -0.373        0.331
Dummy_2010                                         -0.0278      0.171     -0.162
0.871      -0.363        0.307
Dummy_2011                                          0.0071      0.195      0.036
0.971      -0.376        0.390
Dummy_2012                                          0.0419      0.178      0.236
0.814      -0.307        0.391
Dummy_2013                                         -0.0110      0.182     -0.061
0.952      -0.367        0.345
Dummy_2014                                         -0.0087      0.184     -0.047
0.963      -0.371        0.353
Dummy_2015                                         -0.0248      0.196     -0.127
0.899      -0.409        0.360
Dummy_2016                                          0.0183      0.175      0.105
0.917      -0.325        0.362
Dummy_2017                                         -0.0104      0.169     -0.062
0.951      -0.342        0.321
Dummy_2018                                          0.0187      0.176      0.106
0.916      -0.327        0.364
Dummy_2019                                         -0.0032      0.177     -0.018
0.986      -0.351        0.345
Dummy_2020                                          0.0308      0.185      0.166
0.868      -0.333        0.394
Dummy_2021                                          0.0129      0.180      0.072
0.943      -0.340        0.366
Dummy_2022                                         -0.0185      0.168     -0.110
```

```
0.912      -0.349      0.312
Dummy_2023                                       0.0325      0.169      0.192
0.848      -0.300      0.365
Dummy_giro_d_italia                             -0.0074      0.051     -0.144
0.885      -0.108      0.094
Dummy_vuelta_a_espana                            0.0158      0.050      0.318
0.751      -0.082      0.113
Dummy_dauphine                                  -0.0014      0.083     -0.017
0.987      -0.164      0.161
Dummy_tour_de_romandie                          -0.0220      0.098     -0.223
0.823      -0.215      0.171
Dummy_volta_a_catalunya                         -0.0118      0.088     -0.134
0.894      -0.185      0.161
Dummy_itzulia_basque_country                     0.0224      0.096      0.234
0.815      -0.166      0.211
Dummy_tour_de_suisse                            -0.0092      0.083     -0.111
0.912      -0.171      0.153
Dummy_tour_de_pologne                            0.0051      0.148      0.035
0.972      -0.285      0.295
Dummy_paris_nice                                 0.0267      0.083      0.322
0.747      -0.136      0.189
Dummy_tirreno_adriatico                          0.0247      0.102      0.243
0.808      -0.175      0.224
Dummy_stagetype_1                                0.0054      0.091      0.059
0.953      -0.174      0.185
Dummy_stagetype_2                                0.0006      0.069      0.009
0.993      -0.134      0.135
Dummy_stagetype_3                                0.0334      0.083      0.405
0.686      -0.129      0.195
Dummy_stagetype_4                               -0.0242      0.067     -0.363
0.717      -0.155      0.107
Dummy_stagetype_5                                0.0138      0.060      0.230
0.818      -0.104      0.132
==============================================================================
Omnibus:                     1086.784   Durbin-Watson:                   0.155
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              132.930
Skew:                           0.605   Prob(JB):                     1.36e-29
Kurtosis:                       1.770   Cond. No.                         331.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

2.4 Logistic regressions

```
[144]: # Table C.18: Logistic Probability Model: Finishing in Group 1

# LHS: G1 if in G1/G2
resultS12L = sm.logit(formula = 'Dummy_Cluster_1 ~ better_rider_around +␣
 ↪Teammates_behind  + Gap_12_larger1 + Gap_23_larger1 +␣
 ↪Cluster_size_teams_winner + Cluster_size_teams_second␣
 ↪+Cluster_size_teams_third+ Dummy_1982 + Dummy_1983 + Dummy_1985 + Dummy_1986␣
 ↪+ Dummy_1987 + Dummy_1988 + Dummy_1989 + Dummy_1990 + Dummy_1991 +␣
 ↪Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 + Dummy_1996 + Dummy_1997␣
 ↪+ Dummy_1998 + Dummy_1999 + Dummy_2000 +␣
 ↪Dummy_2001+Dummy_2002+Dummy_2003+Dummy_2004+Dummy_2005+Dummy_2006+Dummy_2007+Dummy_2008+Dumm
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana +Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice +Dummy_tirreno_adriatico␣
 ↪+Dummy_stagetype_1+Dummy_stagetype_2+Dummy_stagetype_2+Dummy_stagetype_3+Dummy_stagetype_4+
 ↪data=stage_c[(stage_c['Cluster']==1) |(stage_c['Cluster']==2)]).
 ↪fit()#cov_type='cluster', cov_kwds={'groups':␣
 ↪stage_c[(stage_c['Cluster']==1) |(stage_c['Cluster']==2)]['Race']})
print(resultS12L.summary())

# RHS: G1 if in G1/G2/G3
resultS123L = sm.logit(formula = 'Dummy_Cluster_1 ~ better_rider_around+␣
 ↪Gap_12_larger1+ Gap_23_larger1 + Cluster_size_teams_winner +␣
 ↪Cluster_size_teams_second +Cluster_size_teams_third + Dummy_1982 +␣
 ↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989␣
 ↪+ Dummy_1990 + Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 +␣
 ↪Dummy_1995 + Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000␣
 ↪+␣
 ↪Dummy_2001+Dummy_2002+Dummy_2003+Dummy_2004+Dummy_2005+Dummy_2006+Dummy_2007+Dummy_2008+Dumm
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana +Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice +Dummy_tirreno_adriatico␣
 ↪+Dummy_stagetype_1+Dummy_stagetype_2+Dummy_stagetype_2+Dummy_stagetype_3+Dummy_stagetype_4+
 ↪data=stage_c[(stage_c['Cluster']==1) |(stage_c['Cluster']==2)␣
 ↪|(stage_c['Cluster']==3)]).fit()#cov_type='cluster', cov_kwds={'groups':␣
 ↪stage_c[(stage_c['Cluster']==1) |(stage_c['Cluster']==2)␣
 ↪|(stage_c['Cluster']==3)]['Race']})
print(resultS123L.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.526260
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   No. Observations:                 3523
Model:                          Logit   Df Residuals:                     3459
Method:                           MLE   Df Model:                           63
Date:                Tue, 15 Oct 2024   Pseudo R-squ.:                  0.2339
Time:                        17:46:09   Log-Likelihood:                -1854.0
converged:                       True   LL-Null:                       -2420.2
Covariance Type:            nonrobust   LLR p-value:                8.404e-196
================================================================================
================
                                   coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       -0.2329      0.394     -0.591      0.554
-1.005       0.539
better_rider_around             -1.0893      0.129     -8.446      0.000
-1.342      -0.836
Teammates_behind                 0.6421      0.129      4.959      0.000
0.388       0.896
Gap_12_larger1                  -0.0769      0.111     -0.694      0.487
-0.294       0.140
Gap_23_larger1                  -0.0104      0.139     -0.075      0.940
-0.283       0.262
Cluster_size_teams_winner        0.5041      0.031     16.106      0.000
0.443       0.565
Cluster_size_teams_second       -0.3234      0.023    -14.113      0.000
-0.368      -0.278
Cluster_size_teams_third        -0.0121      0.024     -0.498      0.618
-0.060       0.035
Dummy_1982                       0.3071      0.561      0.547      0.584
-0.793       1.407
Dummy_1983                       0.2495      0.504      0.495      0.621
-0.739       1.238
Dummy_1985                      -0.0984      0.616     -0.160      0.873
-1.306       1.110
Dummy_1986                       0.2178      0.526      0.414      0.679
-0.813       1.248
Dummy_1987                       0.3771      0.506      0.745      0.456
-0.615       1.369
Dummy_1988                      -0.1170      0.594     -0.197      0.844
-1.281       1.047
Dummy_1989                      -0.0824      0.508     -0.162      0.871
-1.078       0.913
Dummy_1990                      -0.0430      0.499     -0.086      0.931
-1.022       0.936
Dummy_1991                       0.0122      0.472      0.026      0.979
-0.914       0.938
Dummy_1992                      -0.3257      0.647     -0.504      0.615
```

```
-1.594       0.942
Dummy_1993                      0.2590      0.485      0.534      0.594
-0.692       1.210
Dummy_1994                     -0.0761      0.570     -0.133      0.894
-1.194       1.041
Dummy_1995                      0.2235      0.486      0.460      0.646
-0.729       1.176
Dummy_1996                      0.1458      0.468      0.312      0.755
-0.772       1.063
Dummy_1997                     -0.1027      0.443     -0.232      0.817
-0.972       0.766
Dummy_1998                     -0.0044      0.442     -0.010      0.992
-0.871       0.863
Dummy_1999                      0.1768      0.436      0.405      0.685
-0.678       1.032
Dummy_2000                     -0.0138      0.433     -0.032      0.975
-0.862       0.835
Dummy_2001                      0.1306      0.439      0.298      0.766
-0.729       0.991
Dummy_2002                     -0.0086      0.434     -0.020      0.984
-0.860       0.843
Dummy_2003                      0.0309      0.447      0.069      0.945
-0.846       0.907
Dummy_2004                      0.2617      0.460      0.569      0.569
-0.640       1.164
Dummy_2005                      0.0916      0.431      0.213      0.832
-0.753       0.936
Dummy_2006                      0.0660      0.432      0.153      0.879
-0.781       0.913
Dummy_2007                     -0.0309      0.436     -0.071      0.944
-0.886       0.824
Dummy_2008                     -0.0820      0.424     -0.193      0.847
-0.912       0.748
Dummy_2009                     -0.1131      0.431     -0.262      0.793
-0.958       0.732
Dummy_2010                      0.1131      0.419      0.270      0.787
-0.709       0.935
Dummy_2011                      0.0389      0.480      0.081      0.935
-0.902       0.980
Dummy_2012                     -0.0669      0.439     -0.152      0.879
-0.928       0.794
Dummy_2013                      0.2105      0.454      0.464      0.643
-0.679       1.100
Dummy_2014                     -0.0731      0.461     -0.159      0.874
-0.977       0.831
Dummy_2015                     -0.0659      0.457     -0.144      0.885
-0.962       0.830
Dummy_2016                      0.1812      0.435      0.417      0.677
```

```
-0.671       1.033
Dummy_2017                    -0.2052      0.420     -0.489      0.625
-1.028       0.617
Dummy_2018                    -0.0145      0.452     -0.032      0.974
-0.901       0.872
Dummy_2019                    -0.0025      0.431     -0.006      0.995
-0.847       0.842
Dummy_2020                     0.0996      0.449      0.222      0.824
-0.780       0.979
Dummy_2021                    -0.0806      0.428     -0.188      0.851
-0.920       0.759
Dummy_2022                     0.1533      0.427      0.359      0.720
-0.684       0.991
Dummy_2023                     0.1117      0.436      0.256      0.798
-0.743       0.967
Dummy_giro_d_italia           -0.0639      0.136     -0.469      0.639
-0.331       0.203
Dummy_vuelta_a_espana         -0.0470      0.133     -0.353      0.724
-0.308       0.214
Dummy_dauphine                -0.0022      0.215     -0.010      0.992
-0.424       0.420
Dummy_tour_de_romandie         0.0586      0.277      0.211      0.833
-0.485       0.603
Dummy_volta_a_catalunya       -0.0198      0.252     -0.079      0.937
-0.514       0.475
Dummy_itzulia_basque_country  -0.0778      0.280     -0.277      0.782
-0.628       0.472
Dummy_tour_de_suisse          -0.0672      0.213     -0.315      0.752
-0.485       0.350
Dummy_tour_de_pologne          0.2302      0.369      0.624      0.533
-0.493       0.953
Dummy_paris_nice              -0.0282      0.214     -0.132      0.895
-0.447       0.391
Dummy_tirreno_adriatico       -0.1656      0.300     -0.552      0.581
-0.753       0.422
Dummy_stagetype_1              0.0354      0.271      0.131      0.896
-0.496       0.567
Dummy_stagetype_2             -0.0670      0.182     -0.367      0.714
-0.424       0.291
Dummy_stagetype_3             -0.0678      0.224     -0.302      0.762
-0.507       0.372
Dummy_stagetype_4              0.0873      0.181      0.482      0.630
-0.268       0.443
Dummy_stagetype_5              0.0566      0.156      0.362      0.717
-0.250       0.363
==============================================================================
================
Optimization terminated successfully.
```

```
         Current function value: 0.514178
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   No. Observations:                 4805
Model:                          Logit   Df Residuals:                     4742
Method:                           MLE   Df Model:                           62
Date:                Tue, 15 Oct 2024   Pseudo R-squ.:                  0.1855
Time:                        17:46:09   Log-Likelihood:                -2470.6
converged:                       True   LL-Null:                       -3033.1
Covariance Type:            nonrobust   LLR p-value:                 6.470e-195
================================================================================
================
                                   coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       -0.3211      0.363     -0.884      0.377
-1.033       0.391
better_rider_around             -1.0216      0.111     -9.164      0.000
-1.240      -0.803
Gap_12_larger1                  -0.0343      0.098     -0.350      0.726
-0.227       0.158
Gap_23_larger1                  -0.0262      0.119     -0.220      0.825
-0.259       0.207
Cluster_size_teams_winner        0.4440      0.025     17.953      0.000
0.396       0.493
Cluster_size_teams_second       -0.2058      0.020    -10.089      0.000
-0.246      -0.166
Cluster_size_teams_third        -0.1719      0.020     -8.435      0.000
-0.212      -0.132
Dummy_1982                       0.1487      0.494      0.301      0.763
-0.819       1.116
Dummy_1983                      -0.0467      0.440     -0.106      0.915
-0.910       0.816
Dummy_1985                      -0.2428      0.569     -0.427      0.669
-1.358       0.872
Dummy_1986                      -0.0674      0.479     -0.141      0.888
-1.006       0.871
Dummy_1987                       0.0137      0.453      0.030      0.976
-0.874       0.902
Dummy_1988                      -0.4307      0.536     -0.803      0.422
-1.481       0.620
Dummy_1989                      -0.3944      0.466     -0.846      0.398
-1.308       0.520
Dummy_1990                      -0.1483      0.445     -0.333      0.739
-1.021       0.724
Dummy_1991                      -0.1549      0.427     -0.363      0.717
```

```
-0.992      0.683
Dummy_1992                      -0.2293      0.564     -0.406      0.685
-1.335      0.877
Dummy_1993                      -0.0692      0.434     -0.159      0.873
-0.921      0.782
Dummy_1994                      -0.3661      0.509     -0.719      0.472
-1.365      0.632
Dummy_1995                       0.0137      0.441      0.031      0.975
-0.850      0.878
Dummy_1996                      -0.2121      0.425     -0.500      0.617
-1.044      0.620
Dummy_1997                      -0.2765      0.408     -0.677      0.498
-1.077      0.524
Dummy_1998                      -0.1151      0.407     -0.283      0.777
-0.914      0.683
Dummy_1999                      -0.1787      0.393     -0.454      0.650
-0.950      0.592
Dummy_2000                      -0.2413      0.397     -0.608      0.543
-1.019      0.536
Dummy_2001                      -0.1636      0.402     -0.407      0.684
-0.951      0.624
Dummy_2002                      -0.2083      0.399     -0.522      0.601
-0.990      0.573
Dummy_2003                      -0.4058      0.405     -1.002      0.316
-1.200      0.388
Dummy_2004                      -0.0403      0.415     -0.097      0.923
-0.853      0.772
Dummy_2005                      -0.2217      0.394     -0.563      0.574
-0.994      0.551
Dummy_2006                      -0.2244      0.393     -0.570      0.568
-0.996      0.547
Dummy_2007                      -0.3279      0.398     -0.825      0.410
-1.107      0.452
Dummy_2008                      -0.2113      0.389     -0.543      0.587
-0.974      0.552
Dummy_2009                      -0.3081      0.395     -0.780      0.435
-1.082      0.466
Dummy_2010                      -0.2316      0.384     -0.603      0.546
-0.984      0.521
Dummy_2011                      -0.1312      0.437     -0.300      0.764
-0.987      0.725
Dummy_2012                      -0.2911      0.403     -0.722      0.470
-1.081      0.499
Dummy_2013                      -0.0447      0.411     -0.109      0.913
-0.851      0.762
Dummy_2014                      -0.3342      0.422     -0.791      0.429
-1.162      0.494
Dummy_2015                      -0.4211      0.419     -1.006      0.314
```

```
-1.241       0.399
Dummy_2016                        -0.0950      0.398     -0.239      0.811
-0.875       0.685
Dummy_2017                        -0.2728      0.386     -0.706      0.480
-1.030       0.484
Dummy_2018                        -0.3118      0.411     -0.759      0.448
-1.117       0.494
Dummy_2019                        -0.2902      0.396     -0.734      0.463
-1.065       0.485
Dummy_2020                        -0.2453      0.412     -0.596      0.551
-1.052       0.561
Dummy_2021                        -0.3295      0.395     -0.833      0.405
-1.104       0.445
Dummy_2022                        -0.2041      0.389     -0.525      0.599
-0.966       0.558
Dummy_2023                        -0.1703      0.397     -0.429      0.668
-0.948       0.607
Dummy_giro_d_italia               -0.0315      0.119     -0.265      0.791
-0.265       0.202
Dummy_vuelta_a_espana             -0.0609      0.115     -0.529      0.597
-0.286       0.165
Dummy_dauphine                    -0.0396      0.191     -0.208      0.835
-0.413       0.334
Dummy_tour_de_romandie             0.1085      0.243      0.446      0.655
-0.368       0.585
Dummy_volta_a_catalunya           -0.0190      0.217     -0.087      0.930
-0.445       0.407
Dummy_itzulia_basque_country      -0.1563      0.237     -0.658      0.510
-0.622       0.309
Dummy_tour_de_suisse              -0.1543      0.186     -0.830      0.406
-0.518       0.210
Dummy_tour_de_pologne              0.0651      0.333      0.196      0.845
-0.587       0.717
Dummy_paris_nice                  -0.0135      0.190     -0.071      0.943
-0.386       0.359
Dummy_tirreno_adriatico           -0.1493      0.273     -0.548      0.584
-0.684       0.385
Dummy_stagetype_1                 -0.1538      0.228     -0.675      0.500
-0.601       0.293
Dummy_stagetype_2                 -0.1126      0.160     -0.704      0.481
-0.426       0.201
Dummy_stagetype_3                  0.0822      0.199      0.414      0.679
-0.307       0.472
Dummy_stagetype_4                 -0.0153      0.158     -0.097      0.923
-0.325       0.294
Dummy_stagetype_5                  0.0182      0.138      0.132      0.895
-0.252       0.288
================================================================================
================
```

[145]:
```
# Table C.19: Logistic Probability Model: Winning the Race from Group 1

#LHS
resultS1Lx = sm.logit(formula = 'Win ~ better_rider_in_Cluster␣
 ↪*Helper_in_Cluster + Teammates_behind + Gap_12_larger1 +␣
 ↪Cluster_size_teams_winner + Cluster_size_teams_second+Dummy_1982 +␣
 ↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989␣
 ↪+ Dummy_1990 + Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 +␣
 ↪Dummy_1995 + Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000␣
 ↪+␣
 ↪Dummy_2001+Dummy_2002+Dummy_2003+Dummy_2004+Dummy_2005+Dummy_2006+Dummy_2007+Dummy_2008+Dumm
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana +Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice +Dummy_tirreno_adriatico␣
 ↪+Dummy_stagetype_1+Dummy_stagetype_2+Dummy_stagetype_2+Dummy_stagetype_3+Dummy_stagetype_4+
 ↪data=stage_c_nsw[stage_c_nsw['Cluster']==1]).fit()#cov_type='cluster',␣
 ↪cov_kwds={'groups': stage_c[stage_c['Cluster']==1]['Race']})
print(resultS1Lx.summary())

#Middle column
resultS1L = sm.logit(formula = 'Win ~ better_rider_in_Cluster␣
 ↪+Helper_in_Cluster + Teammates_behind + Gap_12_larger1 +␣
 ↪Cluster_size_teams_winner + Cluster_size_teams_second+Dummy_1982 +␣
 ↪Dummy_1983 + Dummy_1985 + Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989␣
 ↪+ Dummy_1990 + Dummy_1991 + Dummy_1992 + Dummy_1993 + Dummy_1994 +␣
 ↪Dummy_1995 + Dummy_1996 + Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000␣
 ↪+␣
 ↪Dummy_2001+Dummy_2002+Dummy_2003+Dummy_2004+Dummy_2005+Dummy_2006+Dummy_2007+Dummy_2008+Dumm
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana +Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice +Dummy_tirreno_adriatico␣
 ↪+Dummy_stagetype_1+Dummy_stagetype_2+Dummy_stagetype_2+Dummy_stagetype_3+Dummy_stagetype_4+
 ↪data=stage_c_nsw[stage_c_nsw['Cluster']==1]).fit()#cov_type='cluster',␣
 ↪cov_kwds={'groups': stage_c[stage_c['Cluster']==1]['Race']})
print(resultS1L.summary())

#RHS
# Winning the race for hypothetical teams
```

```
resultHypL = sm.logit(formula = 'Win ~␣
 ↪better_rider_in_Cluster+Helper_hyp_in_Cluster + Teammates_behind_hyp +␣
 ↪Gap_12_larger1 + Cluster_size_teams_hyp_winner +␣
 ↪Cluster_size_teams_hyp_second+Dummy_1982 + Dummy_1983 + Dummy_1985 +␣
 ↪Dummy_1986 + Dummy_1987 + Dummy_1988 + Dummy_1989 + Dummy_1990 + Dummy_1991␣
 ↪+ Dummy_1992 + Dummy_1993 + Dummy_1994 + Dummy_1995 + Dummy_1996 +␣
 ↪Dummy_1997 + Dummy_1998 + Dummy_1999 + Dummy_2000 +␣
 ↪Dummy_2001+Dummy_2002+Dummy_2003+Dummy_2004+Dummy_2005+Dummy_2006+Dummy_2007+Dummy_2008+Dumm
 ↪+ Dummy_giro_d_italia + Dummy_vuelta_a_espana +Dummy_dauphine +␣
 ↪Dummy_tour_de_romandie + Dummy_volta_a_catalunya +␣
 ↪Dummy_itzulia_basque_country + Dummy_tour_de_suisse + Dummy_tour_de_pologne␣
 ↪+ Dummy_paris_nice +Dummy_tirreno_adriatico␣
 ↪+Dummy_stagetype_1+Dummy_stagetype_2+Dummy_stagetype_2+Dummy_stagetype_3+Dummy_stagetype_4+
 ↪data=stage_c_hyp_nsw[stage_c_hyp_nsw['Cluster']==1]).
 ↪fit()#cov_type='cluster', cov_kwds={'groups':␣
 ↪stage_c[stage_c['Cluster']==1]['Race']})
print(resultHypL.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.563631
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                    Win   No. Observations:                 1212
Model:                          Logit   Df Residuals:                     1148
Method:                           MLE   Df Model:                           63
Date:                Tue, 15 Oct 2024   Pseudo R-squ.:                 0.08890
Time:                        17:46:10   Log-Likelihood:                -683.12
converged:                       True   LL-Null:                       -749.77
Covariance Type:            nonrobust   LLR p-value:                 5.868e-07
================================================================================
=============================
                                                coef    std err          z
P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
-----------------------------
Intercept                                     0.2043      0.797      0.256
0.798      -1.357       1.766
better_rider_in_Cluster                      -0.1746      0.160     -1.088
0.277      -0.489       0.140
Helper_in_Cluster                             0.6425      0.359      1.789
0.074      -0.062       1.347
better_rider_in_Cluster:Helper_in_Cluster     0.6308      0.547      1.154
0.248      -0.440       1.702
Teammates_behind                              0.6676      0.211      3.167
0.002       0.254       1.081
Gap_12_larger1                                0.0255      0.200      0.127
0.899      -0.367       0.418
```

```
Cluster_size_teams_winner                              -0.3400      0.041     -8.229
0.000      -0.421      -0.259
Cluster_size_teams_second                              -0.0393      0.044     -0.889
0.374      -0.126       0.047
Dummy_1982                                             -0.1865      0.893     -0.209
0.835      -1.937       1.564
Dummy_1983                                             -0.1153      0.922     -0.125
0.900      -1.923       1.692
Dummy_1985                                              0.1026      1.044      0.098
0.922      -1.944       2.149
Dummy_1986                                              0.5693      1.104      0.516
0.606      -1.595       2.733
Dummy_1987                                              0.2222      0.900      0.247
0.805      -1.541       1.985
Dummy_1988                                              0.4051      1.629      0.249
0.804      -2.788       3.598
Dummy_1989                                              0.2571      1.033      0.249
0.803      -1.767       2.281
Dummy_1990                                              0.2453      0.916      0.268
0.789      -1.551       2.042
Dummy_1991                                              0.1527      0.910      0.168
0.867      -1.632       1.937
Dummy_1992                                              0.1548      1.091      0.142
0.887      -1.983       2.293
Dummy_1993                                              0.0952      0.937      0.102
0.919      -1.742       1.932
Dummy_1994                                              0.4104      1.136      0.361
0.718      -1.816       2.637
Dummy_1995                                              0.3075      0.958      0.321
0.748      -1.570       2.185
Dummy_1996                                              0.1564      0.909      0.172
0.863      -1.625       1.937
Dummy_1997                                              0.1531      0.902      0.170
0.865      -1.615       1.921
Dummy_1998                                              0.1446      0.863      0.168
0.867      -1.547       1.836
Dummy_1999                                              0.3321      0.836      0.397
0.691      -1.307       1.972
Dummy_2000                                              0.1231      0.873      0.141
0.888      -1.588       1.834
Dummy_2001                                              0.1111      0.893      0.124
0.901      -1.639       1.861
Dummy_2002                                              0.2194      0.889      0.247
0.805      -1.524       1.962
Dummy_2003                                              0.0429      0.925      0.046
0.963      -1.770       1.855
Dummy_2004                                             -0.0706      0.881     -0.080
0.936      -1.797       1.656
```

```
Dummy_2005                                       0.1229      0.879      0.140
0.889     -1.601       1.846
Dummy_2006                                       0.2911      0.868      0.335
0.737     -1.410       1.992
Dummy_2007                                       0.1196      0.868      0.138
0.890     -1.581       1.821
Dummy_2008                                       0.0681      0.868      0.078
0.937     -1.633       1.769
Dummy_2009                                       0.0935      0.876      0.107
0.915     -1.624       1.811
Dummy_2010                                       0.1255      0.879      0.143
0.886     -1.597       1.848
Dummy_2011                                       0.3306      0.932      0.355
0.723     -1.495       2.157
Dummy_2012                                       0.1746      0.912      0.191
0.848     -1.613       1.962
Dummy_2013                                      -0.1056      0.929     -0.114
0.909     -1.926       1.715
Dummy_2014                                      -0.0691      0.937     -0.074
0.941     -1.906       1.767
Dummy_2015                                       0.0578      0.969      0.060
0.952     -1.841       1.956
Dummy_2016                                       0.0385      0.890      0.043
0.965     -1.705       1.782
Dummy_2017                                       0.0641      0.872      0.074
0.941     -1.645       1.773
Dummy_2018                                      -0.0205      0.895     -0.023
0.982     -1.775       1.734
Dummy_2019                                      -0.0102      0.887     -0.012
0.991     -1.748       1.728
Dummy_2020                                       0.1207      0.897      0.135
0.893     -1.637       1.878
Dummy_2021                                       0.2314      0.913      0.254
0.800     -1.558       2.021
Dummy_2022                                      -0.0110      0.863     -0.013
0.990     -1.702       1.679
Dummy_2023                                       0.0913      0.867      0.105
0.916     -1.608       1.790
Dummy_giro_d_italia                              0.0688      0.231      0.298
0.766     -0.384       0.522
Dummy_vuelta_a_espana                            0.0812      0.230      0.353
0.724     -0.370       0.532
Dummy_dauphine                                   0.1936      0.385      0.504
0.615     -0.560       0.947
Dummy_tour_de_romandie                           0.0710      0.448      0.158
0.874     -0.808       0.950
Dummy_volta_a_catalunya                          0.0283      0.420      0.067
0.946     -0.794       0.851
```

```
Dummy_itzulia_basque_country                     0.0386      0.438      0.088
0.930      -0.821       0.898
Dummy_tour_de_suisse                             0.1338      0.379      0.353
0.724      -0.609       0.877
Dummy_tour_de_pologne                            0.1302      0.640      0.203
0.839      -1.124       1.384
Dummy_paris_nice                                 0.0890      0.400      0.222
0.824      -0.695       0.873
Dummy_tirreno_adriatico                         -0.0081      0.551     -0.015
0.988      -1.088       1.072
Dummy_stagetype_1                                0.0745      0.437      0.170
0.865      -0.783       0.932
Dummy_stagetype_2                                0.1637      0.311      0.526
0.599      -0.447       0.774
Dummy_stagetype_3                                0.3549      0.378      0.939
0.348      -0.386       1.096
Dummy_stagetype_4                                0.0959      0.305      0.314
0.754      -0.503       0.694
Dummy_stagetype_5                                0.1973      0.278      0.708
0.479      -0.348       0.743
================================================================================
============================
Optimization terminated successfully.
         Current function value: 0.564179
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                    Win   No. Observations:                 1212
Model:                          Logit   Df Residuals:                     1149
Method:                           MLE   Df Model:                           62
Date:                Tue, 15 Oct 2024   Pseudo R-squ.:                 0.08801
Time:                        17:46:10   Log-Likelihood:                -683.78
converged:                       True   LL-Null:                       -749.77
Covariance Type:            nonrobust   LLR p-value:                 5.705e-07
================================================================================
================
                                   coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                        0.1810      0.795      0.228      0.820
-1.378       1.740
better_rider_in_Cluster         -0.1249      0.154     -0.811      0.417
-0.427       0.177
Helper_in_Cluster                0.8983      0.281      3.200      0.001
0.348       1.449
Teammates_behind                 0.6627      0.211      3.144      0.002
0.250       1.076
```

```
Gap_12_larger1                    0.0207      0.200       0.103       0.918
-0.372      0.413
Cluster_size_teams_winner        -0.3402      0.041      -8.241       0.000
-0.421     -0.259
Cluster_size_teams_second        -0.0364      0.044      -0.825       0.410
-0.123      0.050
Dummy_1982                       -0.1590      0.891      -0.178       0.858
-1.906      1.588
Dummy_1983                       -0.0783      0.920      -0.085       0.932
-1.881      1.724
Dummy_1985                        0.1105      1.044       0.106       0.916
-1.935      2.156
Dummy_1986                        0.5643      1.101       0.512       0.608
-1.594      2.723
Dummy_1987                        0.2431      0.898       0.271       0.787
-1.517      2.004
Dummy_1988                        0.4070      1.628       0.250       0.803
-2.784      3.598
Dummy_1989                        0.2767      1.031       0.268       0.788
-1.745      2.298
Dummy_1990                        0.2505      0.915       0.274       0.784
-1.543      2.044
Dummy_1991                        0.1554      0.909       0.171       0.864
-1.626      1.937
Dummy_1992                        0.1573      1.089       0.144       0.885
-1.977      2.292
Dummy_1993                        0.0856      0.937       0.091       0.927
-1.750      1.922
Dummy_1994                        0.4194      1.135       0.370       0.712
-1.805      2.644
Dummy_1995                        0.3193      0.956       0.334       0.738
-1.554      2.193
Dummy_1996                        0.1447      0.908       0.159       0.873
-1.636      1.925
Dummy_1997                        0.1603      0.901       0.178       0.859
-1.605      1.926
Dummy_1998                        0.1636      0.861       0.190       0.849
-1.524      1.851
Dummy_1999                        0.3207      0.835       0.384       0.701
-1.316      1.957
Dummy_2000                        0.1121      0.872       0.129       0.898
-1.597      1.822
Dummy_2001                        0.0998      0.892       0.112       0.911
-1.649      1.848
Dummy_2002                        0.2298      0.888       0.259       0.796
-1.511      1.970
Dummy_2003                        0.0538      0.924       0.058       0.954
-1.757      1.864
```

```
Dummy_2004                    -0.0662      0.880     -0.075      0.940
-1.792      1.659
Dummy_2005                     0.1366      0.878      0.156      0.876
-1.584      1.858
Dummy_2006                     0.3091      0.866      0.357      0.721
-1.389      2.007
Dummy_2007                     0.1264      0.866      0.146      0.884
-1.572      1.825
Dummy_2008                     0.0651      0.867      0.075      0.940
-1.635      1.765
Dummy_2009                     0.1047      0.875      0.120      0.905
-1.610      1.820
Dummy_2010                     0.1183      0.878      0.135      0.893
-1.602      1.838
Dummy_2011                     0.3327      0.932      0.357      0.721
-1.493      2.159
Dummy_2012                     0.1684      0.912      0.185      0.853
-1.618      1.955
Dummy_2013                    -0.0810      0.927     -0.087      0.930
-1.898      1.736
Dummy_2014                    -0.0433      0.936     -0.046      0.963
-1.877      1.791
Dummy_2015                     0.1006      0.966      0.104      0.917
-1.792      1.993
Dummy_2016                     0.0355      0.889      0.040      0.968
-1.707      1.778
Dummy_2017                     0.0982      0.870      0.113      0.910
-1.606      1.803
Dummy_2018                     0.0065      0.893      0.007      0.994
-1.744      1.757
Dummy_2019                    -0.0049      0.886     -0.006      0.996
-1.741      1.731
Dummy_2020                     0.1132      0.896      0.126      0.899
-1.642      1.869
Dummy_2021                     0.2462      0.912      0.270      0.787
-1.541      2.033
Dummy_2022                  3.034e-05      0.861   3.52e-05      1.000
-1.688      1.688
Dummy_2023                     0.1091      0.865      0.126      0.900
-1.587      1.805
Dummy_giro_d_italia            0.0708      0.231      0.306      0.759
-0.382      0.524
Dummy_vuelta_a_espana          0.0848      0.230      0.368      0.713
-0.366      0.536
Dummy_dauphine                 0.1983      0.384      0.516      0.606
-0.554      0.951
Dummy_tour_de_romandie         0.0598      0.449      0.133      0.894
-0.820      0.939
```

```
Dummy_volta_a_catalunya          0.0128      0.420      0.030      0.976
-0.810       0.835
Dummy_itzulia_basque_country     0.0398      0.438      0.091      0.928
-0.819       0.899
Dummy_tour_de_suisse             0.1414      0.379      0.373      0.709
-0.601       0.884
Dummy_tour_de_pologne            0.1286      0.639      0.201      0.841
-1.124       1.381
Dummy_paris_nice                 0.0812      0.400      0.203      0.839
-0.702       0.865
Dummy_tirreno_adriatico          0.0368      0.549      0.067      0.947
-1.039       1.112
Dummy_stagetype_1                0.0788      0.437      0.180      0.857
-0.778       0.936
Dummy_stagetype_2                0.1504      0.311      0.483      0.629
-0.459       0.760
Dummy_stagetype_3                0.3251      0.377      0.862      0.389
-0.414       1.064
Dummy_stagetype_4                0.0929      0.305      0.304      0.761
-0.505       0.691
Dummy_stagetype_5                0.1914      0.278      0.687      0.492
-0.354       0.737
================================================================================
================
Optimization terminated successfully.
         Current function value: 0.576950
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:                    Win   No. Observations:                 1071
Model:                          Logit   Df Residuals:                     1008
Method:                           MLE   Df Model:                           62
Date:                Tue, 15 Oct 2024   Pseudo R-squ.:                 0.08983
Time:                        17:46:11   Log-Likelihood:                -617.91
converged:                       True   LL-Null:                       -678.90
Covariance Type:            nonrobust   LLR p-value:                 8.498e-06
================================================================================
=================
                                   coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
-----------------
Intercept                         0.6466      0.790      0.819      0.413
-0.901       2.195
better_rider_in_Cluster          -0.0825      0.167     -0.494      0.621
-0.410       0.245
Helper_hyp_in_Cluster             1.2661      0.297      4.260      0.000
0.684       1.849
```

```
Teammates_behind_hyp              0.4854      0.253       1.915       0.056
-0.011       0.982
Gap_12_larger1                    0.0262      0.212       0.124       0.901
-0.389       0.442
Cluster_size_teams_hyp_winner    -0.4186      0.052      -8.012       0.000
-0.521      -0.316
Cluster_size_teams_hyp_second    -0.0139      0.049      -0.282       0.778
-0.111       0.083
Dummy_1982                        0.1606      0.916       0.175       0.861
-1.635       1.956
Dummy_1983                       -0.1316      0.906      -0.145       0.884
-1.907       1.644
Dummy_1985                       -0.0670      1.040      -0.064       0.949
-2.106       1.972
Dummy_1986                        0.1508      1.195       0.126       0.900
-2.191       2.493
Dummy_1987                        0.1705      0.893       0.191       0.849
-1.580       1.921
Dummy_1988                        0.2000      1.623       0.123       0.902
-2.981       3.381
Dummy_1989                        0.1167      1.089       0.107       0.915
-2.018       2.251
Dummy_1990                        0.0934      0.908       0.103       0.918
-1.687       1.874
Dummy_1991                       -0.0554      0.924      -0.060       0.952
-1.866       1.755
Dummy_1992                        0.0214      1.209       0.018       0.986
-2.348       2.391
Dummy_1993                       -0.0894      0.957      -0.093       0.926
-1.964       1.785
Dummy_1994                        0.2083      1.265       0.165       0.869
-2.270       2.687
Dummy_1995                        0.1956      1.141       0.171       0.864
-2.040       2.431
Dummy_1996                        0.0509      0.914       0.056       0.956
-1.740       1.842
Dummy_1997                       -0.0436      0.889      -0.049       0.961
-1.786       1.699
Dummy_1998                        0.0710      0.856       0.083       0.934
-1.606       1.748
Dummy_1999                        0.2627      0.833       0.315       0.752
-1.370       1.895
Dummy_2000                       -0.0580      0.864      -0.067       0.947
-1.751       1.635
Dummy_2001                        0.0460      0.886       0.052       0.959
-1.691       1.783
Dummy_2002                        0.1105      0.880       0.126       0.900
-1.614       1.835
```

```
Dummy_2003                           0.0013      0.945      0.001      0.999
-1.851       1.854
Dummy_2004                           0.1264      0.889      0.142      0.887
-1.616       1.868
Dummy_2005                           0.0983      0.862      0.114      0.909
-1.592       1.788
Dummy_2006                           0.1749      0.867      0.202      0.840
-1.525       1.875
Dummy_2007                           0.1653      0.859      0.192      0.847
-1.519       1.850
Dummy_2008                          -0.0818      0.860     -0.095      0.924
-1.768       1.605
Dummy_2009                          -0.0147      0.892     -0.016      0.987
-1.763       1.734
Dummy_2010                          -0.0301      0.866     -0.035      0.972
-1.727       1.667
Dummy_2011                           0.0365      0.978      0.037      0.970
-1.881       1.954
Dummy_2012                           0.1818      0.905      0.201      0.841
-1.592       1.956
Dummy_2013                        -6.625e-05      0.930  -7.13e-05      1.000
-1.823       1.822
Dummy_2014                           0.0091      0.940      0.010      0.992
-1.834       1.852
Dummy_2015                          -0.0897      0.976     -0.092      0.927
-2.002       1.822
Dummy_2016                           0.1133      0.883      0.128      0.898
-1.617       1.844
Dummy_2017                           0.0689      0.863      0.080      0.936
-1.622       1.760
Dummy_2018                           0.1752      0.900      0.195      0.846
-1.589       1.940
Dummy_2019                           0.0785      0.895      0.088      0.930
-1.676       1.833
Dummy_2020                           0.1849      0.917      0.202      0.840
-1.613       1.983
Dummy_2021                           0.1298      0.907      0.143      0.886
-1.649       1.908
Dummy_2022                          -0.0671      0.859     -0.078      0.938
-1.751       1.617
Dummy_2023                           0.1800      0.867      0.208      0.836
-1.520       1.880
Dummy_giro_d_italia                 -0.0383      0.245     -0.156      0.876
-0.519       0.442
Dummy_vuelta_a_espana                0.0452      0.241      0.188      0.851
-0.426       0.517
Dummy_dauphine                      -0.0461      0.392     -0.118      0.906
-0.815       0.723
```

```
Dummy_tour_de_romandie           -0.1589      0.488     -0.326      0.744
-1.114       0.797
Dummy_volta_a_catalunya          -0.0842      0.430     -0.196      0.845
-0.927       0.758
Dummy_itzulia_basque_country      0.0394      0.486      0.081      0.935
-0.914       0.992
Dummy_tour_de_suisse             -0.0300      0.393     -0.076      0.939
-0.800       0.740
Dummy_tour_de_pologne             0.0040      0.709      0.006      0.995
-1.386       1.394
Dummy_paris_nice                  0.0472      0.410      0.115      0.908
-0.756       0.850
Dummy_tirreno_adriatico           0.0438      0.546      0.080      0.936
-1.026       1.113
Dummy_stagetype_1                -0.0019      0.449     -0.004      0.997
-0.883       0.879
Dummy_stagetype_2                -0.0428      0.336     -0.127      0.899
-0.700       0.615
Dummy_stagetype_3                 0.0877      0.415      0.211      0.833
-0.726       0.901
Dummy_stagetype_4                -0.1330      0.321     -0.414      0.679
-0.763       0.497
Dummy_stagetype_5                 0.0457      0.293      0.156      0.876
-0.529       0.621
================================================================================
=================
```

2.5 One-day races

2.5.1 Create dataframe - One-day races

[146]:
```
#ONEDAY RACES
oneday = pd.read_excel('one_day_races_nf15.xlsx')
oneday = oneday.drop(oneday.columns[0], axis=1)
oneday = oneday.drop(columns=['Team_Score'])
oneday = oneday.drop_duplicates()

oneday['Helper_in_Cluster'] = 0
oneday['Helper_hyp_in_Cluster'] = 0
oneday['Captain_hyp_in_Cluster'] = 0
oneday['Captain_in_Cluster'] = 0
oneday['Teammates_behind'] = 0
oneday['Teammates_behind_hyp'] = 0
oneday['Teammates_front'] = 0
oneday['Teammates_front_hyp'] = 0
oneday['Teammates'] = 0
oneday['Cluster'] = 1
oneday['Cluster_size'] = 1
```

```
oneday['Cluster_size_teams'] = 1
oneday['Cluster_size_teams_hyp'] = 1
oneday['Win'] = 0
oneday['Star'] = 0
oneday['Star_other_in_Cluster'] = 0
oneday['not_a_Star'] = 0
oneday['Star_other'] = 0
oneday['Star_other_team'] = 0
oneday['Star_my_team'] = 0
oneday['Star_of_Cluster'] = 0
oneday['Star_other_team_in_Cluster'] = 0
oneday['Star_my_team_in_Cluster'] = 0
oneday['eliminate'] = 0
oneday.loc[oneday['Race'] == 'Flèche Wallone', 'Race']='FW'
oneday.loc[oneday['Race'] == 'San Sebastian', 'Race']='SS'
oneday['Race_Year'] = oneday['Race'] + oneday['Year'].astype(str)
oneday['hyp_team'] = np.random.randint(1, 23, size=len(oneday))

print('We downloaded a total of', len(oneday['Race_Year'].unique()), 'one-day␣
 ↪races.')

oneday.loc[oneday['Place'] == 1, 'Win'] = 1
oneday['Year'] = oneday['Year'].astype(int)
for i in range(1981, 2024):
    oneday.loc[oneday['Year'] == i, 'Dummy_'+str(i)] = 1
    oneday.loc[oneday['Year'] != i, 'Dummy_'+str(i)] = 0
    threshold_up = oneday.loc[oneday['Year'] == i, "Score"].quantile(80/100)
    threshold_down = oneday.loc[oneday['Year'] == i, "Score"].quantile(20/100)
    oneday.loc[oneday['Year'] == i, 'Star'] = (oneday.loc[oneday['Year'] == i,␣
 ↪"Score"] >= threshold_up).astype(int)
for s in oneday['Race'].unique():
    oneday.loc[oneday['Race'] == s, 'Dummy_'+s] = 1
    oneday.loc[oneday['Race'] != s, 'Dummy_'+s] = 0

# Iterate through unique Races
group_names = oneday['Race_Year'].unique()
for group_name in group_names:
    group_data = oneday[oneday['Race_Year'] == group_name]
    for i in range(1, len(group_data)):
        gap_difference = group_data.iloc[i]['Gap'] - group_data.iloc[i -␣
 ↪1]['Gap']
        #If difference to rider in front is at least 5sec then next group
        if gap_difference > 4:
            oneday.loc[group_data.index[i], 'Cluster'] = oneday.loc[group_data.
 ↪index[i - 1], 'Cluster'] + 1
            oneday.loc[group_data.index[i], 'Gap_front'] = gap_difference
        else:
            oneday.loc[group_data.index[i], 'Gap_front'] = 0
            oneday.loc[group_data.index[i], 'Cluster'] = oneday.loc[group_data.
↪index[i - 1], 'Cluster']
    for i in range(0, len(group_data)):
        for j in range(i+1, len(group_data)):
            if (oneday.loc[group_data.index[i], 'Cluster'] == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'Team'] != oneday.loc[group_data.index[j], 'Team']):
                oneday.loc[group_data.index[i], 'Cluster_size'] +=1
                oneday.loc[group_data.index[j], 'Cluster_size'] +=1
            if (oneday.loc[group_data.index[i], 'Cluster'] == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'Team'] == oneday.loc[group_data.index[j], 'Team']):
                oneday.loc[group_data.index[i], 'Cluster_size'] +=1
                oneday.loc[group_data.index[j], 'Cluster_size'] +=1
                oneday.loc[group_data.index[i], 'Helper_in_Cluster'] = 1
                oneday.loc[group_data.index[j], 'Captain_in_Cluster'] = 1
                oneday.loc[group_data.index[i], 'Teammates'] = 1
                oneday.loc[group_data.index[j], 'Teammates'] = 1
            if (oneday.loc[group_data.index[i], 'Cluster']+1 == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'Team'] == oneday.loc[group_data.index[j], 'Team']):
                oneday.loc[group_data.index[i], 'Teammates_behind'] = 1
                oneday.loc[group_data.index[j], 'Teammates_front'] = 1
                oneday.loc[group_data.index[i], 'Teammates'] = 1
                oneday.loc[group_data.index[j], 'Teammates'] = 1
            if (oneday.loc[group_data.index[i], 'Cluster']+2 == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'Team'] == oneday.loc[group_data.index[j], 'Team']):
                oneday.loc[group_data.index[j], 'Teammates_front'] = 1
            if (oneday.loc[group_data.index[i], 'Cluster'] == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'hyp_team'] == oneday.loc[group_data.index[j], 'hyp_team']):
                oneday.loc[group_data.index[i], 'Helper_hyp_in_Cluster'] = 1
                oneday.loc[group_data.index[j], 'Captain_hyp_in_Cluster'] = 1
            if (oneday.loc[group_data.index[i], 'Cluster']+1 == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'hyp_team'] == oneday.loc[group_data.index[j], 'hyp_team']):
                oneday.loc[group_data.index[i], 'Teammates_behind_hyp'] = 1
                oneday.loc[group_data.index[j], 'Teammates_front_hyp'] = 1
            if (oneday.loc[group_data.index[i], 'Cluster']+2 == oneday.
↪loc[group_data.index[j], 'Cluster']) and (oneday.loc[group_data.index[i],␣
↪'hyp_team'] == oneday.loc[group_data.index[j], 'hyp_team']):
                oneday.loc[group_data.index[j], 'Teammates_front_hyp'] = 1
oneday.loc[:, 'Cluster_size_teams'] = oneday.groupby(['Race_Year',␣
↪'Cluster'])['Team'].transform('nunique')
oneday = oneday.copy()
oneday = oneday.drop_duplicates()

for s in oneday['Cluster'].unique():
    oneday.loc[oneday['Cluster'] == 1, 'Dummy_Cluster_1'] = 1
    oneday.loc[oneday['Cluster'] != 1, 'Dummy_Cluster_1'] = 0

# Group by 'Race_Year' and count the unique teams in each group
teams_per_year= oneday.groupby('Race_Year')['Team'].nunique()
# Calculate the mean number of distinct teams per Race_Year
mean_teams_per_year = teams_per_year.mean()
# Group by 'Race_Year' and count the unique hyp_teams in each group
teams_per_year = oneday.groupby('Race_Year')['hyp_team'].nunique()
# Calculate the mean number of distinct teams per Race_Year
hyp_mean_teams_per_year = teams_per_year.mean()

# Find the winners for each group and include 'Cluster_size_winner' and same␣
↪for second
winners = oneday[oneday['Place'] == 1][['Race_Year', 'Cluster_size',␣
↪'Cluster_size_teams']]
oneday = pd.merge(oneday, winners, on='Race_Year', suffixes=('', '_winner'),␣
↪how='left')
second = oneday[oneday['Cluster'] == 2][['Race_Year', 'Cluster_size',␣
↪'Cluster_size_teams']]
oneday = pd.merge(oneday, second, on='Race_Year', suffixes=('', '_second'),␣
↪how='left')
third = oneday[oneday['Cluster'] == 3][['Race_Year', 'Cluster_size',␣
↪'Cluster_size_teams']]
oneday = pd.merge(oneday, third, on='Race_Year', suffixes=('', '_third'),␣
↪how='left')

#we need 3 complete clusters and we need to drop all races where the first two␣
↪groups consist of only one team each
for s in oneday['Race_Year'].unique():
    group_data = oneday.loc[oneday['Race_Year'] == s]
    for i in range(len(group_data)):
        if (oneday.loc[group_data.index[i], 'Place'] == 15) & ((oneday.
↪loc[group_data.index[i], 'Cluster'] == 1) | (oneday.loc[group_data.index[i],␣
↪'Cluster'] == 2) | (oneday.loc[group_data.index[i], 'Cluster'] == 3)):
        #if (oneday.loc[group_data.index[i], 'Place'] == 15) & ((oneday.
↪loc[group_data.index[i], 'Cluster'] == 1) | (oneday.loc[group_data.index[i],␣
↪'Cluster'] == 2)):
            oneday.loc[group_data.index[i], 'eliminate'] = 1
        if  (oneday.loc[group_data.index[i], 'Cluster_size_teams_winner'] == 1)␣
↪& (oneday.loc[group_data.index[i], 'Cluster_size_teams_second'] == 1):
            oneday.loc[group_data.index[i], 'eliminate'] = 1
# Identify unique values of 'Race_Year' where 'eliminate' is already 1
eliminate_race_years = oneday.loc[oneday['eliminate'] == 1, 'Race_Year'].
↪unique()

# Update 'eliminate' column for all rows with a unique value of 'Race_Year'
oneday.loc[oneday['Race_Year'].isin(eliminate_race_years), 'eliminate'] = 1
oneday = oneday[oneday['eliminate'] != 1]
oneday = oneday.copy()
# Filter out riders where Cluster <= 3
oneday = oneday.drop_duplicates()
stage_filtered = oneday[oneday['Cluster'] <= 3].copy()
# Reset the index of the filtered DataFrame
stage_filtered.reset_index(drop=True, inplace=True)
oneday=stage_filtered

oneday = oneday.drop_duplicates()
#other Star in Cluster
oneday['Cluster_id'] = oneday['Race_Year']+oneday['Cluster'].astype(str)
grouped_data = oneday.groupby('Cluster_id')['Star']
sum_star = grouped_data.transform('sum')
oneday['Star_other_in_Cluster'] = (((sum_star >= 2) & (oneday['Star'] == 1))␣
↪|((sum_star >= 1) & (oneday['Star'] != 1))).astype(int)
#other Star in Cluster from another team
for c in oneday['Cluster_id'].unique():
    group_data = oneday.loc[oneday['Cluster_id'] == c]
    for i in range(len(group_data)):
        for j in range(len(group_data)):
            if (oneday.loc[group_data.index[i], 'Team'] != oneday.
↪loc[group_data.index[j], 'Team']) and (oneday.loc[group_data.index[j],␣
↪'Star'] == 1):
                oneday.loc[group_data.index[i], 'Star_other_team_in_Cluster'] =␣
↪1
            if (oneday.loc[group_data.index[i], 'Team'] == oneday.
↪loc[group_data.index[j], 'Team']) and (oneday.loc[group_data.index[j],␣
↪'Star'] == 1) and (i!=j):
                oneday.loc[group_data.index[i], 'Star_my_team_in_Cluster'] = 1
oneday.drop('Cluster_id', axis=1, inplace=True)
oneday = oneday.drop_duplicates()

for s in oneday['Race_Year'].unique():
    group_data = oneday.loc[oneday['Race_Year'] == s]
    for i in range(len(group_data)):
        #let us define Star_other as a dummy that indicates whether one of the␣
↪other riders in the first two clusters has 'Star score'
        oneday.loc[group_data.index[i], 'Star_other'] = (np.
↪sum(group_data[(group_data['Cluster']==1) |␣
↪(group_data['Cluster']==2)]['Star']) > oneday.loc[group_data.
↪index[i]]["Star"]).astype(int)
        oneday.loc[group_data.index[i], 'Star_in_Cluster1'] = (np.
↪sum(group_data[group_data['Cluster']==1]['Star']) >0).astype(int)
        oneday.loc[group_data.index[i], 'Star_in_Cluster2'] = (np.
↪sum(group_data[group_data['Cluster']==2]['Star']) >0).astype(int)
        oneday.loc[group_data.index[i], 'Winner_is_Star'] = (np.
↪sum(group_data[group_data['Win']==1]['Star']) >0).astype(int)
        oneday.loc[group_data.index[i], 'Gap_Cluster12'] =␣
↪(group_data[group_data['Cluster']==2]['Gap_front']).max()
        oneday.loc[group_data.index[i], 'Gap_Cluster23'] =␣
↪(group_data[group_data['Cluster']==3]['Gap_front']).max()
        oneday.loc[group_data.index[i], 'Helper_in_Cluster2'] = (np.
↪sum(group_data[group_data['Cluster']==2]['Helper_in_Cluster']) >0).
↪astype(int)
        oneday.loc[group_data.index[i], 'Winner_is_Satellite'] = (np.
↪sum(group_data[group_data['Win']==1]['Teammates_behind']) >0).astype(int)
        oneday.loc[group_data.index[i], 'Cluster2_std'] =␣
↪group_data[group_data['Cluster']==2]['Score'].std()
        if (oneday.loc[group_data.index[i], 'Team'] != oneday.loc[group_data.
↪index[j], 'Team']) and ((oneday.loc[group_data.index[j], 'Cluster'] == 1) |␣
↪(oneday.loc[group_data.index[j], 'Cluster'] == 2)) and (oneday.
↪loc[group_data.index[j], 'Star'] == 1) and (i!=j):
            oneday.loc[group_data.index[i], 'Star_other_team'] = 1
        if (oneday.loc[group_data.index[i], 'Team'] == oneday.loc[group_data.
↪index[j], 'Team']) and ((oneday.loc[group_data.index[j], 'Cluster'] == 1) |␣
↪(oneday.loc[group_data.index[j], 'Cluster'] == 2)) and (oneday.
↪loc[group_data.index[j], 'Star'] == 1) and (i!=j):
            oneday.loc[group_data.index[i], 'Star_my_team'] = 1
oneday = oneday.drop_duplicates()

oneday['no_Star_my_team_in_Cluster'] = 1 - oneday['Star_my_team_in_Cluster']
oneday['no_Star_other_team_in_Cluster'] = 1 -␣
↪oneday['Star_other_team_in_Cluster']
oneday['no_Star_other_team'] = 1 - oneday['Star_other_team']
oneday['no_Star'] = 1 - oneday['Star']
#we want a variable indicating whether there is a better rider in the group
#(i.e., dummy equal to 1 if rider is not a Star but Star in group exists)
oneday['better_rider_in_Cluster'] = oneday.apply(lambda row: 1 if␣
↪row['Star_other_team_in_Cluster'] == 1 and row['Star'] == 0 else 0, axis=1)
oneday['better_rider_around'] = oneday.apply(lambda row: 1 if␣
↪row['Star_other_team'] == 1 and row['Star'] == 0 else 0, axis=1)
```

```
#find solo wins
oneday['Solo_Win'] = (oneday['Cluster_size_winner']==1).astype(int)
#Dummy for Gap Size and std
oneday['Helper_in_Cluster_exists'] =␣
  ↪(oneday['Cluster_size']>oneday['Cluster_size_teams']).astype(int)
oneday['Gap_12_larger1'] = (oneday['Gap_Cluster12']>=60).astype(int)
oneday['Gap_23_larger1'] = (oneday['Gap_Cluster23']>=60).astype(int)
oneday['Cluster2_std_large'] = (oneday['Cluster2_std']>=oneday.Cluster2_std.
  ↪mean()).astype(int)
oneday = oneday.drop_duplicates()

#captains only
oneday_c = oneday[(oneday["Teammates_front"]==0) &␣
  ↪(oneday["Captain_in_Cluster"]==0)]
oneday_c = oneday_c[oneday_c['Year'].astype(int) > 1980] #in 1980 we do not␣
  ↪have any Scores

#captains only hyp
oneday_c_hyp = oneday[(oneday["Teammates_front_hyp"]==0) &␣
  ↪(oneday["Captain_hyp_in_Cluster"]==0)]
oneday_c_hyp = oneday_c_hyp[oneday_c_hyp['Year'].astype(int) > 1980] #in 1980␣
  ↪we do not have any Scores
oneday_c_hyp.loc[:, 'Cluster_size_teams_hyp'] = oneday_c_hyp.
  ↪groupby(['Race_Year', 'Cluster'])['Rider'].transform('nunique')

#Find the winners for each group and include 'Cluster_size_winner' ... and same␣
  ↪for cluster 2
winners = oneday_c_hyp[oneday_c_hyp['Place'] == 1][['Race_Year',␣
  ↪'Cluster_size_teams']]
oneday_c_hyp = pd.merge(oneday_c_hyp, winners, on='Race_Year', suffixes=('',␣
  ↪'_winner'), how='left')
second = oneday_c_hyp[oneday_c_hyp['Cluster'] == 2][['Race_Year',␣
  ↪'Cluster_size_teams']]
oneday_c_hyp = pd.merge(oneday_c_hyp, second, on='Race_Year', suffixes=('',␣
  ↪'_second'), how='left')
third = oneday_c_hyp[oneday_c_hyp['Cluster'] == 3][['Race_Year',␣
  ↪'Cluster_size_teams']]
oneday_c_hyp = pd.merge(oneday_c_hyp, third, on='Race_Year', suffixes=('',␣
  ↪'_third'), how='left')

#Dummy for Gap Size and std
oneday_c_hyp['Gap_12_larger1'] = (oneday_c_hyp['Gap_Cluster12']>=60).astype(int)
oneday_c_hyp['Gap_23_larger1'] = (oneday_c_hyp['Gap_Cluster23']>=60).astype(int)
oneday_c_hyp = oneday_c_hyp.drop_duplicates()
```

```
#include 'Cluster_size_hyp_winner' ... and same for cluster 2
winners = oneday_c_hyp[oneday_c_hyp['Place'] == 1][['Race_Year',␣
 ↪'Cluster_size_teams_hyp']]
oneday_c_hyp = pd.merge(oneday_c_hyp, winners, on='Race_Year', suffixes=('',␣
 ↪'_winner'), how='left')
second = oneday_c_hyp[oneday_c_hyp['Cluster'] == 2][['Race_Year',␣
 ↪'Cluster_size_teams_hyp']]
oneday_c_hyp = pd.merge(oneday_c_hyp, second, on='Race_Year', suffixes=('',␣
 ↪'_second'), how='left')
third = oneday_c_hyp[oneday_c_hyp['Cluster'] == 3][['Race_Year',␣
 ↪'Cluster_size_teams_hyp']]
oneday_c_hyp = pd.merge(oneday_c_hyp, third, on='Race_Year', suffixes=('',␣
 ↪'_third'), how='left')
oneday_c_hyp = oneday_c_hyp.drop_duplicates()
```

```
We downloaded a total of 298 one-day races.
```

2.5.2 Summary statistics

[147]:
```
# Table C.20: Summary Statistics Non-Dummies

# Create DataFrame 'races' for the calculations
races = pd.DataFrame()

# Calculate the mean for each race and each metric
races['Gap_Cluster12'] = oneday.groupby(['Race_Year'])['Gap_Cluster12'].mean()
races['Gap_Cluster23'] = oneday.groupby(['Race_Year'])['Gap_Cluster23'].mean()
races['Cluster_size_winner'] = oneday.
 ↪groupby(['Race_Year'])['Cluster_size_winner'].mean()
races['Cluster_size_second'] = oneday.
 ↪groupby(['Race_Year'])['Cluster_size_second'].mean()
races['Cluster_size_third'] = oneday.
 ↪groupby(['Race_Year'])['Cluster_size_third'].mean()

# Combine the statistics into one DataFrame
summary_stats = pd.DataFrame({
    'Group 1 size': races['Cluster_size_winner'],
    'Group 2 size': races['Cluster_size_second'],
    'Group 3 size': races['Cluster_size_third'],
    'Gap between Groups 1 and 2': races['Gap_Cluster12'],
    'Gap between Groups 2 and 3': races['Gap_Cluster23']
})

# Use the .describe() function and filter for the relevant stats (mean, std,␣
 ↪min, 50%, max)
summary_stats = summary_stats.describe().loc[['mean', 'std', 'min', '50%',␣
 ↪'max']]
```

```
# Rename index values to match your desired output
summary_stats.index = ['mean', 'std', 'min', '50%', 'max']

# Convert to LaTeX format and print
print(summary_stats.to_latex(index=True, float_format="%.2f"))
```

```
\begin{tabular}{lrrrrr}
\toprule
 & Group 1 size & Group 2 size & Group 3 size & Gap between Groups 1 and 2 & Gap
between Groups 2 and 3 \\
\midrule
mean & 2.18 & 3.48 & 2.92 & 50.77 & 46.52 \\
std & 1.58 & 2.54 & 2.29 & 49.38 & 64.72 \\
min & 1.00 & 1.00 & 1.00 & 5.00 & 5.00 \\
50% & 2.00 & 3.00 & 2.00 & 28.00 & 23.00 \\
max & 10.00 & 12.00 & 11.00 & 219.00 & 408.00 \\
\bottomrule
\end{tabular}
```

[148]:
```
# Table C.21: Mean occurrence of Dummies

cl_one = oneday[oneday['Cluster'] == 1]
cl_two = oneday[oneday['Cluster'] == 2]
cl_three = oneday[oneday['Cluster'] == 3]

# Calculate mean values for each group
group1_mean = cl_one[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()
group2_mean = cl_two[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()
group3_mean = cl_three[['Star', 'better_rider_in_Cluster',␣
 ↪'Helper_in_Cluster']].mean()

# Calculate overall mean (across all groups)
overall_mean = oneday[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
 ↪'Teammates_behind']].mean()

# Create DataFrame for the table
mean_occurrence = pd.DataFrame({
    'Group 1': group1_mean,
    'Group 2': group2_mean,
    'Group 3': group3_mean,
    'Overall': overall_mean
}).T
```

```
# Replace NaN values with 0.0 instead of an empty string (for compatibility)
mean_occurrence.fillna(0.0, inplace=True)

# Convert to LaTeX format
print(mean_occurrence[['Star', 'better_rider_in_Cluster', 'Helper_in_Cluster',␣
  ↪'Teammates_behind']].to_latex(index=True, float_format="%.3f"))
```

```
\begin{tabular}{lrrrr}
\toprule
 & Star & better_rider_in_Cluster & Helper_in_Cluster & Teammates_behind \\
\midrule
Group 1 & 0.323 & 0.288 & 0.038 & 0.165 \\
Group 2 & 0.227 & 0.454 & 0.072 & 0.126 \\
Group 3 & 0.138 & 0.325 & 0.057 & 0.000 \\
Overall & 0.221 & 0.368 & 0.059 & 0.149 \\
\bottomrule
\end{tabular}
```

2.5.3 Regressions

```
[149]: # Table C.22: Linear Probability Model: Finishing in Group 1 - One-Day Races

# LHS
result012 = sm.ols(formula = 'Dummy_Cluster_1 ~   better_rider_around␣
  ↪+Teammates_behind + Gap_12_larger1 + Gap_23_larger1 +␣
  ↪Cluster_size_teams_winner + Cluster_size_teams_second␣
  ↪+Cluster_size_teams_third+ Dummy_MSR + Dummy_LBL + Dummy_FW +Dummy_RVV +␣
  ↪Dummy_PR + Dummy_SS ', data=oneday_c[(oneday_c['Cluster']==1)␣
  ↪|(oneday_c['Cluster']==2)]).fit()
print(result012.summary())

# RHS
result0123 = sm.ols(formula = 'Dummy_Cluster_1 ~   better_rider_around +␣
  ↪Gap_12_larger1+ Gap_23_larger1 + Cluster_size_teams_winner +␣
  ↪Cluster_size_teams_second +Cluster_size_teams_third + Dummy_MSR + Dummy_LBL␣
  ↪+ Dummy_FW +Dummy_RVV + Dummy_PR + Dummy_SS ',␣
  ↪data=oneday_c[(oneday_c['Cluster']==1) |(oneday_c['Cluster']==2)␣
  ↪|(oneday_c['Cluster']==3)]).fit()
print(result012.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   R-squared:                       0.265
Model:                            OLS   Adj. R-squared:                  0.249
Method:                 Least Squares   F-statistic:                     16.09
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           1.52e-31
Time:                        17:46:21   Log-Likelihood:                -331.56
```

```
No. Observations:                 593   AIC:                             691.1
Df Residuals:                     579   BIC:                             752.5
Df Model:                          13
Covariance Type:            nonrobust
================================================================================
=============
                                coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------
Intercept                     0.4634      0.077      6.012      0.000
0.312       0.615
better_rider_around          -0.1679      0.044     -3.843      0.000
-0.254      -0.082
Teammates_behind              0.1589      0.052      3.044      0.002
0.056       0.261
Gap_12_larger1                0.0133      0.041      0.325      0.745
-0.067       0.093
Gap_23_larger1               -0.0252      0.045     -0.562      0.574
-0.113       0.063
Cluster_size_teams_winner     0.0929      0.013      7.262      0.000
0.068       0.118
Cluster_size_teams_second    -0.0554      0.009     -6.235      0.000
-0.073      -0.038
Cluster_size_teams_third      0.0003      0.010      0.027      0.978
-0.019       0.019
Dummy_MSR                    -0.0501      0.171     -0.292      0.770
-0.386       0.286
Dummy_LBL                     0.0319      0.069      0.460      0.646
-0.104       0.168
Dummy_FW                     -0.0686      0.080     -0.853      0.394
-0.226       0.089
Dummy_RVV                    -0.0128      0.061     -0.210      0.834
-0.133       0.107
Dummy_PR                     -0.0406      0.055     -0.739      0.460
-0.149       0.067
Dummy_SS                     -0.0666      0.058     -1.158      0.248
-0.180       0.046
==============================================================================
Omnibus:                      154.914   Durbin-Watson:                   1.111
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               30.892
Skew:                           0.201   Prob(JB):                     1.96e-07
Kurtosis:                       1.957   Cond. No.                         54.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        Dummy_Cluster_1   R-squared:                       0.265
Model:                            OLS   Adj. R-squared:                  0.249
Method:                 Least Squares   F-statistic:                     16.09
Date:                Tue, 15 Oct 2024   Prob (F-statistic):           1.52e-31
Time:                        17:46:21   Log-Likelihood:                -331.56
No. Observations:                 593   AIC:                             691.1
Df Residuals:                     579   BIC:                             752.5
Df Model:                          13
Covariance Type:            nonrobust
================================================================================
=============
                                coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------
Intercept                     0.4634      0.077      6.012      0.000
0.312       0.615
better_rider_around          -0.1679      0.044     -3.843      0.000
-0.254      -0.082
Teammates_behind              0.1589      0.052      3.044      0.002
0.056       0.261
Gap_12_larger1                0.0133      0.041      0.325      0.745
-0.067       0.093
Gap_23_larger1               -0.0252      0.045     -0.562      0.574
-0.113       0.063
Cluster_size_teams_winner     0.0929      0.013      7.262      0.000
0.068       0.118
Cluster_size_teams_second    -0.0554      0.009     -6.235      0.000
-0.073      -0.038
Cluster_size_teams_third      0.0003      0.010      0.027      0.978
-0.019       0.019
Dummy_MSR                    -0.0501      0.171     -0.292      0.770
-0.386       0.286
Dummy_LBL                     0.0319      0.069      0.460      0.646
-0.104       0.168
Dummy_FW                     -0.0686      0.080     -0.853      0.394
-0.226       0.089
Dummy_RVV                    -0.0128      0.061     -0.210      0.834
-0.133       0.107
Dummy_PR                     -0.0406      0.055     -0.739      0.460
-0.149       0.067
Dummy_SS                     -0.0666      0.058     -1.158      0.248
-0.180       0.046
==============================================================================
Omnibus:                      154.914   Durbin-Watson:                   1.111
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               30.892
```

```
Skew:                           0.201   Prob(JB):                     1.96e-07
Kurtosis:                       1.957   Cond. No.                         54.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[150]:
```
# Table C.23: Linear Probability Model: Winning the Race from Group 1 - One-Day␣
↪Races

oneday_c_nsw= oneday_c[oneday_c['Cluster_size_teams_winner']!=1]
oneday_c_hyp_nsw= oneday_c_hyp[oneday_c_hyp['Cluster_size_winner']!=1]

#LHS
result1x_OD = sm.ols(formula = 'Win ~ ␣
↪better_rider_in_Cluster*Helper_in_Cluster + Teammates_behind +␣
↪Gap_12_larger1 + Cluster_size_teams_winner + Cluster_size_teams_second+␣
↪Dummy_LBL + Dummy_FW +Dummy_RVV + Dummy_PR + Dummy_SS',␣
↪data=oneday_c_nsw[oneday_c_nsw['Cluster']==1]).fit()#cov_type='cluster',␣
↪cov_kwds={'groups': stage_c[stage_c['Cluster']==1]['Race']})
print(result1x_OD.summary())

#Middle column
result1_OD = sm.ols(formula = 'Win ~  better_rider_in_Cluster+Helper_in_Cluster␣
↪+ Teammates_behind + Gap_12_larger1 + Cluster_size_teams_winner +␣
↪Cluster_size_teams_second+ Dummy_LBL + Dummy_FW +Dummy_RVV + Dummy_PR +␣
↪Dummy_SS', data=oneday_c_nsw[oneday_c_nsw['Cluster']==1]).
↪fit()#cov_type='cluster', cov_kwds={'groups':␣
↪stage_c[stage_c['Cluster']==1]['Race']})
print(result1_OD.summary())

#RHS
# Winning the race for hypothetical teams
resultHyp_OD = sm.ols(formula = 'Win ~␣
↪better_rider_in_Cluster+Helper_hyp_in_Cluster + Teammates_behind_hyp +␣
↪Gap_12_larger1 + Cluster_size_teams_hyp_winner +␣
↪Cluster_size_teams_hyp_second+Dummy_LBL + Dummy_FW +Dummy_RVV + Dummy_PR +␣
↪Dummy_SS', data=oneday_c_hyp_nsw[oneday_c_hyp_nsw['Cluster']==1]).fit()
print(resultHyp_OD.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.138
Model:                            OLS   Adj. R-squared:                  0.081
Method:                 Least Squares   F-statistic:                     2.424
Date:                Tue, 15 Oct 2024   Prob (F-statistic):            0.00611
Time:                        17:46:22   Log-Likelihood:                -113.63
```

```
No. Observations:                 194   AIC:                             253.3
Df Residuals:                     181   BIC:                             295.7
Df Model:                          12
Covariance Type:            nonrobust
==================================================================================
=============================
                                                   coef    std err          t
P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
-----------------------------
Intercept                                        0.6962      0.135      5.149
0.000       0.429       0.963
better_rider_in_Cluster                         -0.2268      0.072     -3.145
0.002      -0.369      -0.084
Helper_in_Cluster                               -0.1051      0.468     -0.225
0.823      -1.028       0.818
better_rider_in_Cluster:Helper_in_Cluster        0.3174      0.506      0.627
0.532      -0.682       1.317
Teammates_behind                                -0.2405      0.102     -2.355
0.020      -0.442      -0.039
Gap_12_larger1                                  -0.0038      0.081     -0.047
0.963      -0.163       0.155
Cluster_size_teams_winner                       -0.0697      0.022     -3.209
0.002      -0.113      -0.027
Cluster_size_teams_second                       -0.0018      0.024     -0.076
0.940      -0.048       0.045
Dummy_LBL                                        0.0023      0.114      0.020
0.984      -0.222       0.227
Dummy_FW                                        -0.0587      0.140     -0.419
0.676      -0.335       0.218
Dummy_RVV                                        0.0264      0.121      0.218
0.828      -0.213       0.266
Dummy_PR                                        -0.0361      0.100     -0.363
0.717      -0.233       0.160
Dummy_SS                                         0.0383      0.117      0.328
0.743      -0.192       0.269
==============================================================================
Omnibus:                       78.277   Durbin-Watson:                   0.269
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               20.677
Skew:                           0.554   Prob(JB):                     3.24e-05
Kurtosis:                       1.846   Cond. No.                         98.3
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.137
Model:                            OLS   Adj. R-squared:                  0.084
Method:                 Least Squares   F-statistic:                     2.617
Date:                Tue, 15 Oct 2024   Prob (F-statistic):            0.00403
Time:                        17:46:22   Log-Likelihood:                -113.84
No. Observations:                 194   AIC:                             251.7
Df Residuals:                     182   BIC:                             290.9
Df Model:                          11
Covariance Type:            nonrobust
================================================================================
=============
                               coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------
Intercept                    0.7001      0.135      5.191      0.000
0.434       0.966
better_rider_in_Cluster     -0.2200      0.071     -3.091      0.002
-0.360      -0.080
Helper_in_Cluster            0.1652      0.181      0.911      0.363
-0.192       0.523
Teammates_behind            -0.2416      0.102     -2.370      0.019
-0.443      -0.040
Gap_12_larger1               0.0001      0.080      0.002      0.999
-0.158       0.158
Cluster_size_teams_winner   -0.0718      0.021     -3.347      0.001
-0.114      -0.029
Cluster_size_teams_second   -0.0014      0.024     -0.061      0.952
-0.048       0.045
Dummy_LBL                    0.0013      0.113      0.012      0.991
-0.222       0.225
Dummy_FW                    -0.0530      0.140     -0.380      0.705
-0.329       0.222
Dummy_RVV                    0.0268      0.121      0.221      0.825
-0.212       0.266
Dummy_PR                    -0.0359      0.099     -0.361      0.718
-0.232       0.160
Dummy_SS                     0.0329      0.116      0.283      0.778
-0.197       0.262
==============================================================================
Omnibus:                       77.083   Durbin-Watson:                   0.271
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               20.916
Skew:                           0.563   Prob(JB):                     2.87e-05
Kurtosis:                       1.850   Cond. No.                         31.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
```

```
specified.
                            OLS Regression Results
==============================================================================
Dep. Variable:                    Win   R-squared:                       0.133
Model:                            OLS   Adj. R-squared:                  0.078
Method:                 Least Squares   F-statistic:                     2.420
Date:                Tue, 15 Oct 2024   Prob (F-statistic):            0.00797
Time:                        17:46:22   Log-Likelihood:                -110.80
No. Observations:                 186   AIC:                             245.6
Df Residuals:                     174   BIC:                             284.3
Df Model:                          11
Covariance Type:            nonrobust
================================================================================
=================
                                    coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-----------------
Intercept                         0.6571      0.127      5.177      0.000
0.407       0.908
better_rider_in_Cluster          -0.2023      0.074     -2.748      0.007
-0.348      -0.057
Helper_hyp_in_Cluster             0.3294      0.170      1.932      0.055
-0.007       0.666
Teammates_behind_hyp              0.2044      0.119      1.719      0.087
-0.030       0.439
Gap_12_larger1                   -0.0275      0.082     -0.334      0.738
-0.190       0.135
Cluster_size_teams_hyp_winner    -0.0635      0.022     -2.905      0.004
-0.107      -0.020
Cluster_size_teams_hyp_second    -0.0187      0.024     -0.792      0.430
-0.065       0.028
Dummy_LBL                        -0.0285      0.114     -0.250      0.803
-0.253       0.196
Dummy_FW                         -0.1154      0.142     -0.812      0.418
-0.396       0.165
Dummy_RVV                         0.0418      0.128      0.325      0.745
-0.212       0.295
Dummy_PR                         -0.0253      0.102     -0.248      0.804
-0.226       0.176
Dummy_SS                          0.0260      0.118      0.221      0.826
-0.207       0.259
==============================================================================
Omnibus:                       97.886   Durbin-Watson:                   0.241
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               20.102
Skew:                           0.530   Prob(JB):                     4.31e-05
Kurtosis:                       1.787   Cond. No.                         29.4
==============================================================================
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```