Boston Bike Share Data

This project uses data from [Blue Bikes (https://www.bluebikes.com/system-data)](https://www.bluebikes.com/system-data), the bike sharing program in Boston. The website contains all trip histories for customers over all time. Generally, interesting questions to consider looking into relate to what the customer-base looks like:

- Who uses the bikes for commuting to work vs fun? Can you determine where each of these happens most? What are the attributes of trips taken during the week vs weekend? What does usage look like across the seasons?
- Profile the users by trip duration and trip distance (between stations). How much can we understand about what the person did on the trip just from know the beginning and end.
- "Is bike share use preventing drunk driving?". That is, can you tell if people use bikes to going out to bars and restaurants? When might this happen? Who might be using the bikes for these activities? While this is at heart a casual question, investigating associations is the first step toward answering the question.
- For all of the above questions, what is the gender/age of the bike share users doing these activity? Can you infer anything about who is doing what?

Getting the data

The data is available on [this website (https://s3.amazonaws.com/hubway-data/index.html)](https://s3.amazonaws.com/hubway-data/index.html). You should use trip data between (and including) 2018-05 to 2019-03.

- Use the fact that `pd.read_csv` can read in both `zip` files, as well as urls.
- Once the data is downloaded, write it to a local file so you don't have to download the data repeatedly.

Cleaning the data and descriptive statistics

- Clean the data.
- Understand the data in ways relevant to your question, using univariate and bivariate analysis of the data, as well as aggregations.

Tips:

- To measure distances between two lat/long pairs, use [the haversine distance formula (https://gist.github.com/rochacbruno/2883505)](https://gist.github.com/rochacbruno/2883505) -- in the comments, someone also provides a Numpy-vectorized formula.
- If you'd like to try plotting your statistics on a map, try [Folium (https://blog.dominodatalab.com/creating-interactive-crime-maps-with-folium/)](https://blog.dominodatalab.com/creating-interactive-crime-maps-with-folium/).

Missingness

While the dataset has no empty entries, both the `age` and `birth year` columns certainly contain missing data, as they are self-reported. You must:

1. Figure out which rows likely contain "missing" data for `age` and `birth year` and blank them out.
2. Assess the missingness of `age` and/or `birth year`.

Hypothesis Test

Find a hypothesis test to perform. You can use the questions at the top of the notebook for inspiration.

Summary of Findings

Introduction

For the dataset, we have combined 11 similar datasets which are the datas of blue bikes from May 2018 to March 2019. In each of this dataset, there are several columns which describe numbers of status and details of a single trip when users used the bike. From here, we conduct an interesting question: "What is the relationship between the user's age (or gender) and their distance traveled when using the blue bikes?"

Results of Cleaning and EDA:

We first combined all 11 datasets into one large dataset. Then we get the "age" column for deducting the relevant data ('birth year'). From here, we found out some 1800s birth years which are not valid datas. After this, we calculated the distance for a single trip and add these distances to a column called 'distance'. Then we investigated the column 'gender' and interpret the representation of data. We found out that '1' has the biggest population in the combined dataset and assumed it represents for male users, and '2' for the female users.

Results of Missingness:

we compared Null vs. Non-Null (age) distributions: gender, and we find that the distributions for these two groups are not similar. Therefore, the missingness of age and gender is not MCAR, but could be MAR or NMAR. If the age is not missing at random, the cause might be some teenagers (under 16) which cannot legally ride the bike, had gave false informations on their age in order to fulfill the age limitation. Therefore the missingness of age is not relevant other columns but the value itself. If the age is missing at random, the cause might be that some female users are not willing to share their age information. We found out that the proportion of female user in age missingness is larger. Therefore the missingness of age is dependent on the gender of user.

Results of Hypothesis Test

For users who commute to work by using blue bikes, there is a significant difference between gender and distance traveled. And the difference in distance travelled for commuting to work is about 0.174 that the bike-riding distance for male is 0.17km more than it for female. Therefore, we conclude that male may probably live further from their working places than females.

Your Code Starts Here

In [104]:

```
%matplotlib inline
import os

import pandas as pd
import numpy as np

import matplotlib.pyplot as plot
import seaborn as sns
```

Cleaning and EDA:

Get data from 201805 to 201903

In [105]:

```
b1805_fp = os.path.join('data', '1805.csv')
b1805= pd.read_csv(b1805_fp)
b1805.head()
b1806_fp = os.path.join('data', '1806.csv')
b1806= pd.read_csv(b1806_fp)
b1806.head()
b1807_fp = os.path.join('data', '1807.csv')
b1807= pd.read_csv(b1807_fp)
b1807.head()
b1808_fp = os.path.join('data', '1808.csv')
b1808= pd.read_csv(b1808_fp)
b1808.head()
b1809_fp = os.path.join('data', '1809.csv')
b1809= pd.read_csv(b1809_fp)
b1809.head()
b1810_fp = os.path.join('data', '1810.csv')
b1810= pd.read_csv(b1810_fp)
b1810.head()
b1811_fp = os.path.join('data', '1811.csv')
b1811= pd.read_csv(b1811_fp)
b1811.head()
b1812_fp = os.path.join('data', '1812.csv')
b1812= pd.read_csv(b1812_fp)
b1812.head()
b1901_fp = os.path.join('data', '1901.csv')
b1901= pd.read_csv(b1901_fp)
b1901.head()
b1902_fp = os.path.join('data', '1902.csv')
b1902= pd.read_csv(b1902_fp)
b1902.head()
b1903_fp = os.path.join('data', '1903.csv')
b1903= pd.read_csv(b1903_fp)
b1903.head()
```

Out[105]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id
0	513	2019-03-01 00:00:50.9430	2019-03-01 00:09:24.6550	27	Roxbury Crossing T Stop - Columbus Ave at Trem...	42.331184	-71.095171	282
1	322	2019-03-01 00:04:49.2220	2019-03-01 00:10:11.5170	39	Washington St at Rutland St	42.338515	-71.074041	46
2	425	2019-03-01 00:05:56.0230	2019-03-01 00:13:01.7960	12	Ruggles T Stop - Columbus Ave at Melnea Cass Blvd	42.336244	-71.087986	21

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id
3	159	2019-03-01 00:08:18.7730	2019-03-01 00:10:57.8110	279	Williams St at Washington St	42.306539	-71.107669	133
4	1229	2019-03-01 00:09:13.0300	2019-03-01 00:29:42.5170	16	Back Bay T Stop - Dartmouth St at Stuart St	42.348074	-71.076570	78

combined all the dataframe:

get the ages by seperately substracting 2018 and 2019 and create a column named age

In [106]:

```
b18 = pd.concat([b1805,b1806,b1807,b1808,b1809,b1810,b1811,b1812], ignore_index = Tr
b18.head()
b18['age']=2018-b18['birth year']

b19 = pd.concat([b1901, b1902, b1903], ignore_index = True)
b19.head()
b19['age']=2019-b19['birth year']
# combined_year = pd.concat([b18, b19], keys = ['18', '19'])
# combined_year.head()
combined = pd.concat([b18, b19], ignore_index=True)
combined.head()
```

Out[106]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	
0	1177	2018-05-01 00:01:32.4590	2018-05-01 00:21:10.0260	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	
1	733	2018-05-01 00:05:19.4970	2018-05-01 00:17:32.7190	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	C
2	437	2018-05-01 00:05:37.7590	2018-05-01 00:12:54.8300	54	Tremont St at West St	42.354979	-71.063348	6	
3	730	2018-05-01 00:05:39.6780	2018-05-01 00:17:50.5880	54	Tremont St at West St	42.354979	-71.063348	46	S M
4	411	2018-05-01 00:06:10.1590	2018-05-01 00:13:02.0490	54	Tremont St at West St	42.354979	-71.063348	6	

In [ ]:

In [ ]:

get the distance and create a column 'distance'

In [108]:

```
def distance(start_long, start_lat, end_long, end_lat):
    radius = 6371 #km
    dlat = np.radians(end_lat-start_lat)
    dlon = np.radians(end_long-start_long)
    a = np.sin(dlat/2) * np.sin(dlat/2) + np.cos(np.radians(start_lat)) \
        * np.cos(np.radians(end_lat)) * np.sin(dlon/2) * np.sin(dlon/2)
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1-a))
    d = radius * c
    return d
```

add the column 'distance' to the dataframe

In [109]:

```
ned['distance']=distance(combined['start station longitude'],combined['start statior
ned.head()
```

Out[109]:

ipduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	end s
1177	2018-05-01 00:01:32.4590	2018-05-01 00:21:10.0260	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	Ke
733	2018-05-01 00:05:19.4970	2018-05-01 00:17:32.7190	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	Pac C Common Bri
437	2018-05-01 00:05:37.7590	2018-05-01 00:12:54.8300	54	Tremont St at West St	42.354979	-71.063348	6	Cambri at
730	2018-05-01 00:05:39.6780	2018-05-01 00:17:50.5880	54	Tremont St at West St	42.354979	-71.063348	46	Ch Science I Massach A
411	2018-05-01 00:06:10.1590	2018-05-01 00:13:02.0490	54	Tremont St at West St	42.354979	-71.063348	6	Cambri at

convert the starttime and stoptime to readable datetime

In [110]:

```
combined['starttime']=pd.to_datetime(combined['starttime'])
combined['stoptime']=pd.to_datetime(combined['stoptime'])
combined.head()
```

Out[110]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	
0	1177	2018-05-01 00:01:32.459	2018-05-01 00:21:10.026	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	
1	733	2018-05-01 00:05:19.497	2018-05-01 00:17:32.719	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	Con
2	437	2018-05-01 00:05:37.759	2018-05-01 00:12:54.830	54	Tremont St at West St	42.354979	-71.063348	6	Ca
3	730	2018-05-01 00:05:39.678	2018-05-01 00:17:50.588	54	Tremont St at West St	42.354979	-71.063348	46	Scie Ma
4	411	2018-05-01 00:06:10.159	2018-05-01 00:13:02.049	54	Tremont St at West St	42.354979	-71.063348	6	Ca

clean the column of age and birth year

In [113]:

```
age_bool=combined['age']>100
combined.loc[age_bool,'birth year']=np.NaN
combined.loc[age_bool, 'age']=np.NaN
```

In [114]:

```
combined.head()
combined['gender'].value_counts()
```

Out[114]:

```
1    1138973
2     394623
0     221989
Name: gender, dtype: int64
```

investigate information about gender

Because according to the survey male are more likely to use bike sharing, 1 is male. Moreover, because other would be the lowest portion of the distribution of gender values, 0 is other. Therefore, 2 represents female

In [115]:

```

count_1=(combined['gender']==1).sum()#1138973
count_2=(combined['gender']==2).sum()#394623
count_0=(combined['gender']==0).sum()#221989


```

Missingness

In [125]:

```
combined['is_null']=combined['age'].isnull()
combined['speed']=combined['distance']/(combined['tripduration']/60)#km/min
combined.head()
```

Out[125]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	
0	1177	2018-05-01 00:01:32.459	2018-05-01 00:21:10.026	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	
1	733	2018-05-01 00:05:19.497	2018-05-01 00:17:32.719	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	Con
2	437	2018-05-01 00:05:37.759	2018-05-01 00:12:54.830	54	Tremont St at West St	42.354979	-71.063348	6	Ca
3	730	2018-05-01 00:05:39.678	2018-05-01 00:17:50.588	54	Tremont St at West St	42.354979	-71.063348	46	Scie Ma
4	411	2018-05-01 00:06:10.159	2018-05-01 00:13:02.049	54	Tremont St at West St	42.354979	-71.063348	6	Ca

In [126]:

```
gender_bool=(combined['gender']==1)^(combined['gender']==2)
sort_df=combined.loc[gender_bool]
sort_df.head()
```

Out[126]:

starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	end station name	s la
2018-05-01 00:01:32.459	2018-05-01 00:21:10.026	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	Kendall T	42.3
2018-05-01 00:05:19.497	2018-05-01 00:17:32.719	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	Packard's Corner - Commonwealth Ave at Brighto...	42.3
2018-05-01 00:05:37.759	2018-05-01 00:12:54.830	54	Tremont St at West St	42.354979	-71.063348	6	Cambridge St at Joy St	42.3
2018-05-01 00:05:39.678	2018-05-01 00:17:50.588	54	Tremont St at West St	42.354979	-71.063348	46	Christian Science Plaza - Massachusetts Ave at...	42.3
2018-05-01 00:06:25.245	2018-05-01 00:15:33.797	88	Inman Square at Vellucci Plaza / Hampshire St	42.374035	-71.101427	87	Harvard University Housing - 115 Putnam Ave at...	42.3

In [127]:

```
sort_df.loc[sort_df['age'].isnull()==True].mean()
```

Out[127]:

```
tripduration               1203.773663
start station id             90.362140
start station latitude       42.353945
start station longitude     -71.087095
end station id               82.522634
end station latitude         42.352985
end station longitude       -71.087377
bikeid                     3042.596708
birth year                         NaN
gender                        1.135802
age                                NaN
distance                      1.945956
is_null                       1.000000
speed                         0.124225
dtype: float64
```

In [128]:

```
sort_df[['tripduration','distance', 'gender','is_null','speed']].mean()
```

Out[128]:

```
tripduration    1197.166992
distance           1.925244
gender             1.257319
is_null            0.000158
speed              0.159145
dtype: float64
```

according to the data, the mean of gender is apparently different

Therefore, we want to investigate the missingness of age on gender

In [129]:

```
observed=sort_df.pivot_table(index='is_null', columns='gender', aggfunc='size').appl
observed_diff=observed.diff().iloc[-1].abs().sum() / 2
observed_diff
```

Out[129]:

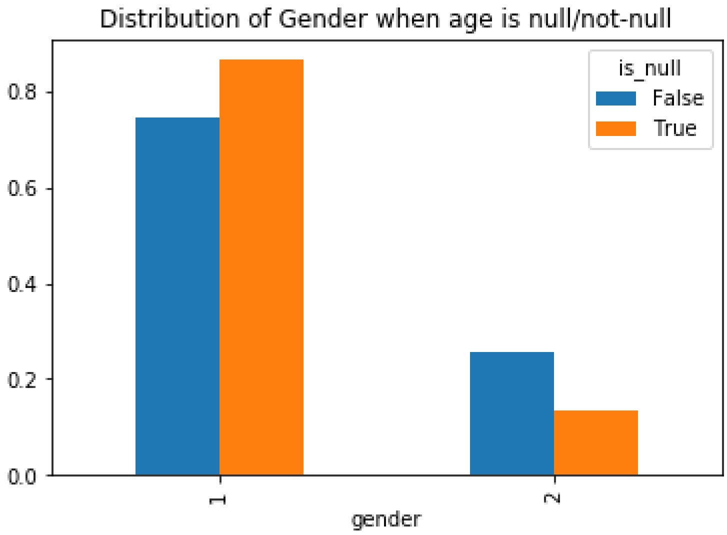
```
0.12153553457241084
```

In [130]:

```
observed.T.plot(kind='bar', title='Distribution of Gender when age is null/not-null
```

Out[130]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1b3d1dd940>
```



Comparing Null vs. Non-Null (age) distributions: gender Are the distributions 'similar enough'? If yes, then missingness of age is not dependent on gender Use a permutation test to assess the two distributions are similar. For categorical columns, use TVD as the test-statistic.

In [96]:

```
n_repetitions=500
tvds=[]
for _ in range(n_repetitions):
    # shuffle the gender column
    shuffled_col = (
        combined['gender']
        .sample(replace=False, frac=1)
        .reset_index(drop=True)
    )

    # put them in a table
    shuffled = (
        combined
        .assign(**{
            'gender': shuffled_col
        })
    )

    # compute the tvd
    shuffled = (
        shuffled
        .pivot_table(index='is_null', columns='gender', aggfunc='size')
        .apply(lambda x:x / x.sum(), axis=1)
    )

    tvd = shuffled.diff().iloc[-1].abs().sum() / 2
    # add it to the list of results

    tvds.append(tvd)
```

In [97]:

```
tvds
```

```
 0.030086531321056258,
 0.014548031327141914,
 0.010684597346546865,
 0.06654029299902822,
 0.03893458270971273,
 0.014417316566447985,
 0.04938072587945164,
 0.01964038815131742,
 0.02475572032007277,
 0.0118057807740133,
 0.029848077144248288,
 0.011805780774013286,
 0.029978791904942237,
 0.008955790804770603,
 0.022144184527638028,
 0.024755720320072797,
 0.028488439539973875,
 0.025876903747539155,
 0.035440317666619675,
 0.011567326597205281,
```

In [101]:

```
pval = np.mean(tvds > observed_diff)
pval
```

Out[101]:

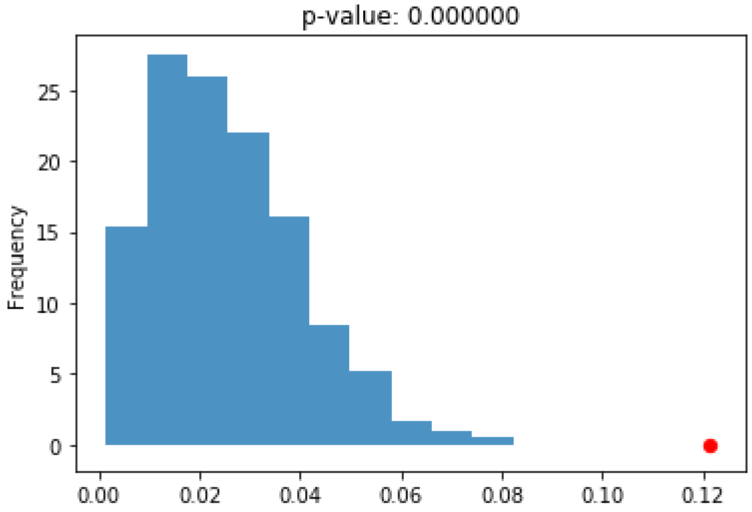
```
0.0
```

because pval is less than significant level 0.05, we reject the null hypo that the distribution is similar

Therefore, age and gender is not MCAR, but could be MAR or NMAR

In [103]:

```
pd.Series(tvds).plot(kind='hist', density=True, alpha=0.8, title='p-value: %f' % pva
plot.scatter(observed_diff, 0, color='red', s=40);
```



Hypothesis Test

hypothesis test: we want to figure out for users who commuting to work, if there is a significant difference between gender and distance travel

first filter our the user who commute to work by using bike sharing because people mostly work in weekdays, we create a column 'weekday' to figure out the day of the week and then we extract the rows with values from 1 to 5 in column 'weekday'

In [225]:

```
combined['weekday'] = combined['starttime'].dt.dayofweek
weekday_bool=combined['weekday']<=5
combined_weekday=combined.loc[weekday_bool]
combined_weekday.head()
```

Out[225]:

e	stoptime	start station id	start station name	start station latitude	start station longitude	end station id	end station name	end station latitude	s long
1 9	2018-05-01 00:21:10.026	184	Sidney Research Campus/ Erie Street at Waverly	42.357753	-71.103934	189	Kendall T	42.362428	-71.0
1 7	2018-05-01 00:17:32.719	67	MIT at Mass Ave / Amherst St	42.358100	-71.093198	41	Packard's Corner - Commonwealth Ave at Brighto...	42.352261	-71.1
1 9	2018-05-01 00:12:54.830	54	Tremont St at West St	42.354979	-71.063348	6	Cambridge St at Joy St	42.361291	-71.0
1 8	2018-05-01 00:17:50.588	54	Tremont St at West St	42.354979	-71.063348	46	Christian Science Plaza - Massachusetts Ave at...	42.343666	-71.0
1 9	2018-05-01 00:13:02.049	54	Tremont St at West St	42.354979	-71.063348	6	Cambridge St at Joy St	42.361291	-71.0

From the website https://fivethirtyeight.com/features/which-cities-sleep-in-and-which-get-to-work-early/ (https://fivethirtyeight.com/features/which-cities-sleep-in-and-which-get-to-work-early/), the graph "what time do you get to work" indicates that people in Boston usually get to work at around 8:11am. We will set the time of getting to work from 8:00 am to 8:30am Therefore, we will filter our the users whose endtime is around 8:00am to 8:30am

In [228]:

```
ol=(combined_weekday['stoptime'].dt.strftime('%H:%M')<'08:30')&(combined_weekday['sto
ekday=combined_weekday.loc[stoptime_bool]
ekday.head()
```

Out[228]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id
216	3987	2018-05-01 07:02:22.395	2018-05-01 08:08:49.635	134	Boylston St at Dartmouth St	42.350413	-71.076550	134
380	1665	2018-05-01 07:34:46.640	2018-05-01 08:02:32.015	208	Oak Square - 615 Washington St	42.350570	-71.166491	134
392	1860	2018-05-01 07:36:14.873	2018-05-01 08:07:15.066	11	Longwood Ave at Binney St	42.338629	-71.106500	73
403	1640	2018-05-01 07:37:43.236	2018-05-01 08:05:03.585	47	Cross St at Hanover St	42.362811	-71.056067	10
410	2209	2018-05-01 07:39:00.687	2018-05-01 08:15:49.991	81	Chinatown T Stop	42.352409	-71.062679	185

from the website [https://www.wbur.org/bostonomix/2016/09/01/bra-report-commute (https://www.wbur.org/bostonomix/2016/09/01/bra-report-commute)](https://www.wbur.org/bostonomix/2016/09/01/bra-report-commute), the graph shows that almost 80% of people in Boston can commute to work within 60min, which is 3600 seconds. Therefore, filter out users whose tripdutation is less than 3600

In [229]:

```
tripduration_bool=combined_weekday['tripduration']<3600
combined_weekday=combined_weekday.loc[tripduration_bool]
combined_weekday.head()
```

Out[229]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id
380	1665	2018-05-01 07:34:46.640	2018-05-01 08:02:32.015	208	Oak Square - 615 Washington St	42.350570	-71.166491	134
392	1860	2018-05-01 07:36:14.873	2018-05-01 08:07:15.066	11	Longwood Ave at Binney St	42.338629	-71.106500	73
403	1640	2018-05-01 07:37:43.236	2018-05-01 08:05:03.585	47	Cross St at Hanover St	42.362811	-71.056067	10
410	2209	2018-05-01 07:39:00.687	2018-05-01 08:15:49.991	81	Chinatown T Stop	42.352409	-71.062679	185
413	1330	2018-05-01 07:39:22.219	2018-05-01 08:01:32.658	109	TD Garden - West End Park	42.365908	-71.064467	46

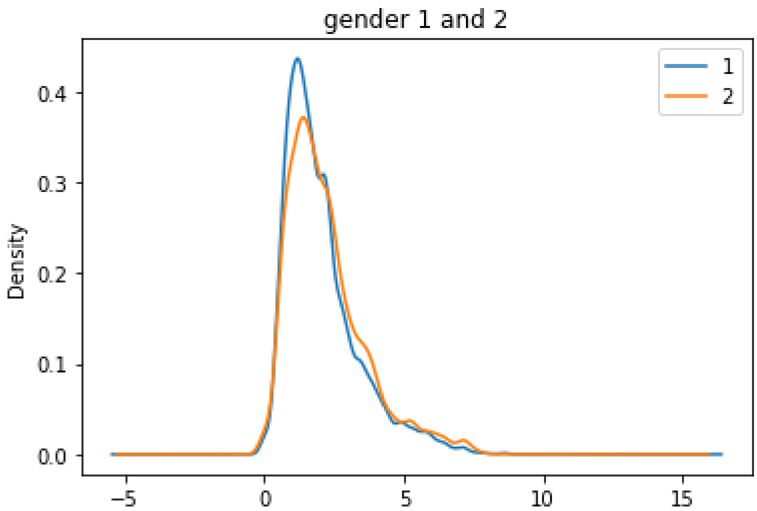
In [230]:

```
```

In [231]:

```
gender_bool=(combined_weekday['gender']==1)^(combined_weekday['gender']==2)
graph_data=combined_weekday.loc[gender_bool]
title='gender 1 and 2'

(
    graph_data
    .groupby('gender')['distance']
    .plot(kind='kde', legend=True, subplots=False, title=title)
);
```



From the graph, we can see that the distribution is almost same, but this does not give much information about the differeces between gender on distance travel using Blue Bikes. I'm going to conduct a T-Test to test the hypothesis.

In [232]:

```
observed_diff=graph_data.groupby('gender')['distance'].agg('mean').diff().iloc[-1]#(
observed_diff
```

Out[232]:

```
0.17396769943705115
```

In [233]:

```
shuffled_distance=(graph_data['distance'].sample(replace=False, frac=1).reset_index(
original_and_shuffled_df=graph_data.assign(**{'shuffled distance': shuffled_distance
original_and_shuffled_df.head()
```

Out[233]:

	tripduration	starttime	stoptime	start station id	start station name	start station latitude	start station longitude	end station id
380	1665	2018-05-01 07:34:46.640	2018-05-01 08:02:32.015	208	Oak Square - 615 Washington St	42.350570	-71.166491	134
392	1860	2018-05-01 07:36:14.873	2018-05-01 08:07:15.066	11	Longwood Ave at Binney St	42.338629	-71.106500	73
403	1640	2018-05-01 07:37:43.236	2018-05-01 08:05:03.585	47	Cross St at Hanover St	42.362811	-71.056067	10
410	2209	2018-05-01 07:39:00.687	2018-05-01 08:15:49.991	81	Chinatown T Stop	42.352409	-71.062679	185
413	1330	2018-05-01 07:39:22.219	2018-05-01 08:01:32.658	109	TD Garden - West End Park	42.365908	-71.064467	46

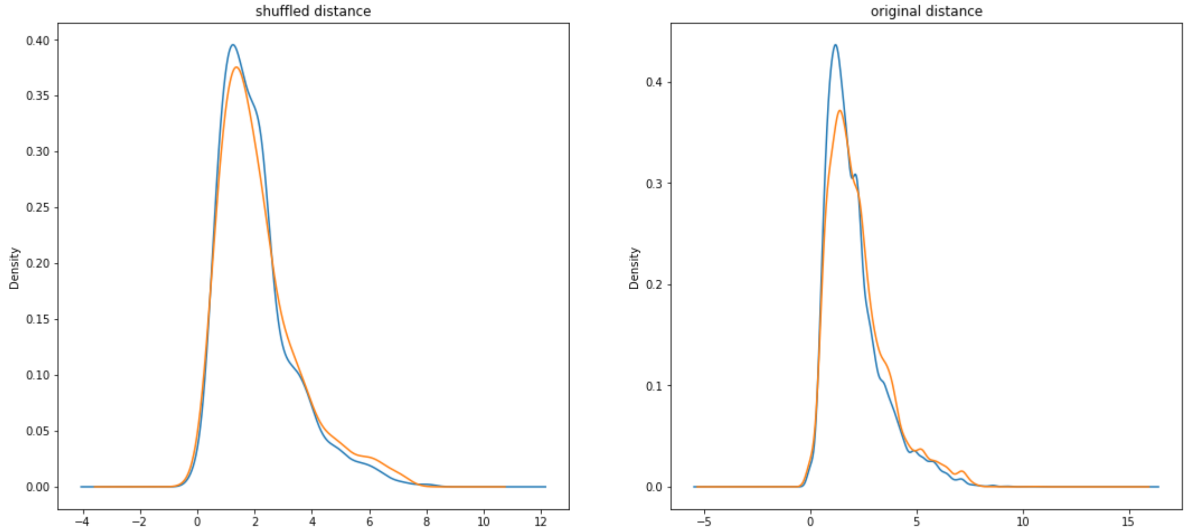
In [234]:

```
difference=original_and_shuffled_df.groupby('gender')[['shuffled distance', 'distanc
```

the distribution of the shuffled groups

In [235]:

```
fig, axes = plot.subplots(1,2, figsize=(18,8))
title = 'shuffled distance'
original_and_shuffled_df.groupby('gender')['shuffled distance'].plot(kind='kde', tit
title = 'original distance'
original_and_shuffled_df.groupby('gender')['distance'].plot(kind='kde', title=title,
```



#do permutation test to get the p_val

In [237]:

```
n_repetitions = 500

differences = []
for _ in range(n_repetitions):

    shuffled_dis=(graph_data['distance'].sample(replace=False, frac=1).reset_index(c
    original_and_shuffled=graph_data.assign(**{'shuffled distance': shuffled_dis})

    # compute the group differences (test statistic!)
    #####TODO: difference abs()????
    difference = abs(original_and_shuffled.groupby('gender')['shuffled distance'].ag


    # add it to the list of results
    differences.append(difference)
```

In [238]:

```
differences
```

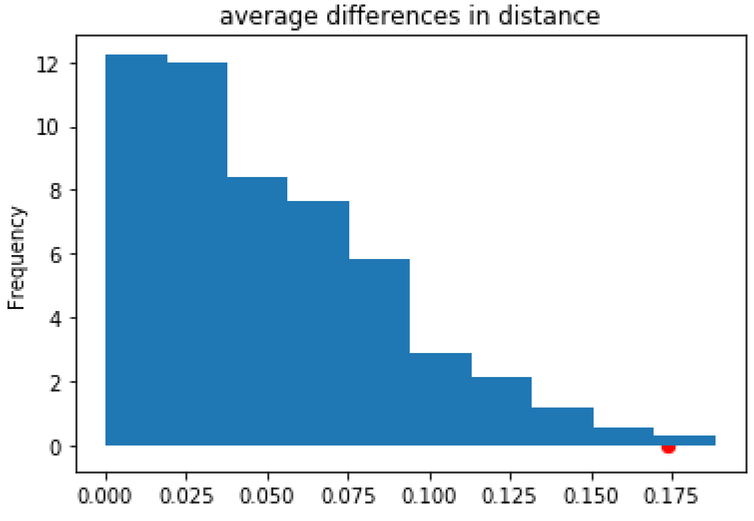
```
 0.03345370261318159,
 0.0936695267492409,
 0.05033497230786965,
 0.07811962238488768,
 0.030224820632367422,
 0.03651879140304004,
 0.04481776206811183,

 0.042352314275097225,
 0.0012121909974198708,
 0.1140453627014901З,
 0.05605381095795137,
 0.03236280867101504,
 0.07797385652973254,
 0.056244063157248725,
 0.012569373617589363,
 0.08594330826030849,
 0.07444890773255342,
 0.10896627152634464,
 0.08419272775199182,
```

Under the null hypothesis, we rarely see differences as large as this. Therefore, we reject the null hypothesis: the two groups do not come from the same distribution.

In [239]:

```
title = 'average differences in distance'
pd.Series(differences).plot(kind='hist',density=True, title=title)
plot.scatter(observed_diff, 0, color='red', s=40);
```



get the p_val

In [241]:

```
p_val=np.count_nonzero(differences >= observed_diff) / n_repetitions
p_val
```

Out[241]:

```
0.004
```

because pval is less than significant level 0.05, we reject the null hypo

In [ ]:

```

```