from IPython.core.interactiveshell import InteractiveShell InteractiveShell.ast node interactivity = "all" # import libraries import numpy as np import pandas as pd # import the CSV df = pd.read csv("Traffic Crashes - Crashes.csv") # split the date and time into two columns df[['CRASH\_DATE', 'CRASH\_TIME']] = df.CRASH\_DATE.str.split(expand=True) df[['DATE POLICE NOTIFIED', 'TIME POLICE NOTIFIED']] = df.DATE POLICE NOTIFIED.str.sp # Remove the ID because it provides no info df = df.drop(columns=['CRASH RECORD ID']) df.head(5)CRASH\_DATE POSTED\_SPEED\_LIMIT TRAFFIC\_CONTROL\_DEVICE DEVICE\_CONDITION WEATHER\_CONDITION 0 7/10/19 35 NO CONTROLS NO CONTROLS **CLEAR FUNCTIONING** 6/30/17 STOP SIGN/FLASHER CLEAR 35 **PROPERLY FUNCTIONING** 2 7/10/20 30 TRAFFIC SIGNAL **CLEAR PROPERLY** 3 7/11/20 30 NO CONTROLS **NO CONTROLS** CLEAR 7/8/20 20 NO CONTROLS **NO CONTROLS CLEAR** 5 rows × 28 columns In [4]: # replace the white spaces and '' with Nan df['INTERSECTION RELATED I'].replace(r'^\s\*\$', np.nan, reqex=True) #df['INTERSECTION RELATED I'].head(5) df['NOT RIGHT OF WAY'].replace(r'^\s\*\$', np.nan, regex=True) #df['NOT RIGHT OF WAY'].head(5) df['HIT AND RUN I'].replace(r'^\s\*\$', np.nan, regex=True) #df['HIT AND RUN I'].head(5) Out[4]: 0 NaN 1 Y 2 NaN 3 NaN NaN . . . NaN 481618 NaN 481619 481620 NaN 481621 Y 481622 NaN Name: INTERSECTION RELATED\_I, Length: 481623, dtype: object Out[4]: 0 NaN NaN 2 NaN 3 NaN 4 NaN 481618 481619 NaN 481620 NaN 481621 NaN NaN Name: NOT RIGHT OF WAY, Length: 481623, dtype: object NaN Out[4]: NaN 1 2 NaN 3 Υ 4 NaN 481618 481619 Υ 481620 NaN 481621 NaN 481622 Name: HIT AND RUN I, Length: 481623, dtype: object # what are the reported primary top 10 causes of accidents reasons = df.groupby(['PRIM CONTRIBUTORY CAUSE']).size().reset index(name='Count') reasons.sort values(by='Count', ascending=False).head(10) reasons.plot(x='PRIM CONTRIBUTORY CAUSE', y='Count', kind='barh', figsize = (9,9)) # seems like the vast majority of the accidents' reasons are "unable to determine" # we will explore that deeper using other variables PRIM\_CONTRIBUTORY\_CAUSE Count UNABLE TO DETERMINE 178010 36 FAILING TO YIELD RIGHT-OF-WAY 52982 18 19 FOLLOWING TOO CLOSELY 51238 26 25849 **NOT APPLICABLE** IMPROPER OVERTAKING/PASSING 23 22858 21 **IMPROPER BACKING** 21096 FAILING TO REDUCE SPEED TO AVOID CRASH 20751 IMPROPER LANE USAGE 22 18637 IMPROPER TURNING/NO SIGNAL 24 15986 12 DRIVING SKILLS/KNOWLEDGE/EXPERIENCE 14968 Out[5]: <AxesSubplot:ylabel='PRIM CONTRIBUTORY CAUSE'> Count VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED) UNABLE TO DETERMINE TURNING RIGHT ON RED ROAD ENGINEERING/SURFACE/MARKING DEFECTS
ROAD CONSTRUCTION/MAINTENANCE
RELATED TO BUS STOP
PHYSICAL CONDITION OF DRIVEN
PASSING STOPPED SCHOOL BUS
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER
OBSTRUCTED CROSSWALKS
NOT APPLICABLE
MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT
IMPROPER TURNING/NO SIGNAL
IMPROPER TURNING/NO SIGNAL
IMPROPER TURNING/NO SIGNAL
IMPROPER LANE USAGE
IMPROPER BACKING
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)
FOLLOWING TOO LOSELY
FAILING TO YIELD RIGHT-OF-WAY
FAILING TO REDUCE SPEED TO AVOID CRASH
EXCEEDING SAFE SPEED FOR CONDITIONS ROAD ENGINEERING/SURFACE/MARKING DEFECTS CONTRIBUTORY CAUSE FAILING TO REDUCE SPEED TO AVOID CRASH

EXCEEDING SAFE SPEED FOR CONDITIONS

EXCEEDING SAFE SPEED FOR CONDITIONS

EXCEEDING AUTHORIZED SPEED LIMIT

EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST

EQUIPMENT - VEHICLE CONDITION

DRIVING SKILLS/KNOWLEDGE/EXPERIENCE

DRIVING ON WRONG SIDE/WRONG WAY)

DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ET)

DISTRACTION - FROM OUTSIDE VEHICLE

DISTRACTION - FROM OUTSIDE VEHICLE

DISREGARDING YIELD SIGN

DISREGARDING TRAFFIC SIGNALS

DISREGARDING STOP SIGN

DISREGARDING STOP SIGN

DISREGARDING OTHER TRAFFIC SIGNS

CELL PHONE USE OTHER THAN TEXTING

BICYCLE ADVANCING LEGALLY ON RED LIGHT

ANIMAL ANIMAL 25000 50000 100000 125000 150000 175000 # let us explore some more general statistics about the accidents # what times do the accidents happen time\_hour = df.groupby(['CRASH\_HOUR']).size().reset\_index(name='Count') time\_hour['Count'].plot(kind='bar', figsize = (9,9)) Out[6]: <AxesSubplot:> 35000 30000 25000 20000 15000 10000 5000 6  $\infty$ 10 11 12 13 14 15 17 18 5 2 21 # see what day of the week day week = df.groupby(['CRASH DAY OF WEEK']).size().reset index(name='Count') day week.plot(x = 'CRASH DAY OF WEEK', y = 'Count', kind='bar', figsize = (9,9)) Out[7]: <AxesSubplot:xlabel='CRASH DAY OF WEEK'> 80000 Count 70000 60000 50000 40000 30000 20000 10000 CRASH DAY OF WEEK # see what what weather weather = df.groupby(['WEATHER CONDITION']).size().reset index(name='Count') weather.plot(x = 'WEATHER CONDITION', y = 'Count', figsize=(15,5)) # first crash type crash type = df.groupby(['FIRST CRASH TYPE']).size().reset index(name='Count') crash type.plot(x = 'FIRST CRASH TYPE', y = 'Count', kind='barh', figsize=(15,5)) # lightning condition light = df.groupby(['LIGHTING CONDITION']).size().reset index(name='Count') light.plot(x = 'LIGHTING CONDITION', y = 'Count', kind='barh', figsize=(15,5)) Out[8]: <AxesSubplot:xlabel='WEATHER CONDITION'> Out[8]: <AxesSubplot:ylabel='FIRST CRASH TYPE'> Out[8]: <AxesSubplot:ylabel='LIGHTING CONDITION'> Count 350000 300000 250000 200000 150000 100000 50000 BLOWING SAND, SOIL, DIRT CLÉAR FOG/SMOKE/HAZE OTHER SEVERE CROSS WIND GATE SNOW WEATHER CONDITION TURNING Count TRAIN SIDESWIPE SAME DIRECTION SIDESWIPE OPPOSITE DIRECTION REAR TO SIDE REAR TO REAR CRASH TYPE REAR TO FRONT REAR END PEDESTRIAN PEDALCYCLIST PARKED MOTOR VEHICLE OVERTURNED OTHER OBJECT OTHER NONCOLLISION HEAD ON FIXED OBJECT ΑΝΙΜΔΙ ANGLE 100000 20000 40000 60000 80000 120000 Count UNKNOWN DUSK IGHTING CONDITION DAYLIGHT DAWN DARKNESS, LIGHTED ROAD DARKNESS 50000 100000 150000 200000 250000 300000 import matplotlib.pyplot as plt # see injuries total and fatal injuries per type of accident inj = df.groupby(['PRIM CONTRIBUTORY CAUSE'])['INJURIES TOTAL'].sum() #inj inj.plot(x = 'INJURIES\_TOTAL', y = 'PRIM\_CONTRIBUTORY\_CAUSE', kind='barh', figsize=(1) Out[9]: <AxesSubplot:ylabel='PRIM\_CONTRIBUTORY CAUSE'> LCOHOL/DRUGS (USE WHEN ARREST IS EFFECTE)
UNABLE TO DETERMIN
TURNING RIGHT ON RE
TEXTIN
ROAD ENGINEERING/SURFACE/MARKING DEFECT
ROAD CONSTRUCTION/MAINTENANC ROAD CONSTRUCT TOWNSHIPS IN THE PRINTS OF TH EXCEEDING SAFE S EXCEEDING AUT EVASIVE ACTION DUE TO ANIMAL, O EQUIPMENT DRIVING SKILLS/KNC DRIVING ON WRG ECTRONIC DEVICE (MAVIGATION DEVI DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DI DISTRACTION DI DI DISTRACTION DI DISTRACTION DI 12500 15000 20000 # fatal accidents by type inj f = df.groupby(['PRIM CONTRIBUTORY CAUSE'])['INJURIES FATAL'].sum() inj f.plot(x = 'INJURIES FATAL', y = 'PRIM CONTRIBUTORY CAUSE', kind='barh', figsize= Out[10]: <AxesSubplot:ylabel='PRIM CONTRIBUTORY CAUSE'> VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC. UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED UNABLE TO DETERMINE MOTORCYCLE ADVANCING LEGALLY ON RET
MOTORCYCLE ADVANCING LEGALLY ON RET
MORPOPER OVERTAKING/PI
MPROPER OVERTAKING/PI
MPROPER STEEL MORPOPER
HAD BEEN DRINKING (USE WHEN ARREST IS NOT
FOLLOWING TOOL
FALING TO YELD NIGHT DISTRACTION - OTHER ELECTRONIC DEVICE ( # change to die based on type of accident death rate = (inj f / inj) d r = (death rate\*100).plot(kind='barh', figsize=(15,8))  $\#frames_inj = [inj, inj_f]$ #r = pd.concat(frames inj, axis=1) #r.plot(kind='barh', figsize=(15,8), xticks = range(0,25000,1000)) print('The graph shows that the death per injury is incredibly small.') The graph shows that the death per injury is incredibly small. VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, FIC. UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS FFFECTED UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS FFFECTED UNDER TO DETERMINE TURNING RIGHT ON RFF ROAD ENGINEERING/SURFACE/MARKING TIDE

ROAD CONSTRUCTION/MAINTENAM
RELATED TO BUS ST
PHYSICAL CONDITION OF DRIVE
PASSING STOPPED SCHOOL IN
OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT TO MESTAL
OSFRUCTED CROSSWAL HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)
FOLLOWING TOO CLOSELY EXCEEDING A EVASIVE ACTION DUE TO ANIMAI DRIVING U
DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION
DISTRACTION
DISTRACTION DISTRACTION - FROM DUTSIDE VEHICLE
DISTRACTION - FROM INSIDE VEHICLE
DISTRACTION - STOM INSIDE VEHICLE
DISTREGARDING THE SIGN STOP SIGN
DISTREGARDING STOP SIGN
DISTREGARDING STOP SIGN
DISTREGARDING STOP SIGN
DISTREGARDING STOP SIGN
CELL PHONE USE OTHER THAN TEXTING
BICYCLE ADVANCING LEGALLY ON RED LIGHT
ANIMAL # convert date string to date format df['CRASH DATE'] = pd.to datetime(df['CRASH DATE'], errors='coerce') df['CRASH DATE'].head(2) # create a column with years only df['YEAR'] = df['CRASH DATE'].dt.year df.YEAR.head(2) Out[12]: 0 2019-07-10 2017-06-30 Name: CRASH DATE, dtype: datetime64[ns] 0 2019 2017 Name: YEAR, dtype: int64 # deadly accidents by year fatal year = df.groupby('YEAR')['INJURIES FATAL'].sum() fatal year.plot(figsize=(9,9)) print('As the graph is showing, the deadly injuries have been growing up to 2020, then Out[13]: <AxesSubplot:xlabel='YEAR'> As the graph is showing, the deadly injuries have been growing up to 2020, then the ra te decreased significantly in 2021. 160 140 120 100 80 60 40 20 0 2013 2014 2015 2018 2019 2020 2016 2017 2021 In [14]: # let's see the number and type of injuries based on the speed limit # create subsets and join them tot inj speed = df.groupby('POSTED SPEED LIMIT')['INJURIES TOTAL'].sum().reset index() n\_inj\_speed = df.groupby('POSTED\_SPEED\_LIMIT')['INJURIES\_FATAL'].sum().reset\_index(nar n\_inj2\_speed = df.groupby('POSTED\_SPEED\_LIMIT')['INJURIES\_INCAPACITATING'].sum().reset n\_inj3\_speed = df.groupby('POSTED\_SPEED\_LIMIT')['INJURIES\_NON\_INCAPACITATING'].sum().: n inj4 speed = df.groupby('POSTED SPEED LIMIT')['INJURIES REPORTED NOT EVIDENT'].sum() frames\_inj = [n\_inj\_speed, n\_inj2\_speed, n\_inj3\_speed, n\_inj4\_speed] r = pd.concat(frames inj) # plot is very big with very small numbers r.plot(x = 'POSTED SPEED LIMIT', figsize=(15,10), xticks = range(0,100,5))Out[14]: POSTED\_SPEED\_LIMIT FATAL INCAPACITATING NON\_INCAPACITATING NO\_INJURY 0 NaN NaN NaN 0.0 NaN NaN NaN 2 2 0.0 NaN NaN NaN 0.0 NaN NaN NaN 0.0 NaN NaN NaN **37** NaN NaN NaN 5.0 60 NaN NaN 0.0 63 NaN NaN NaN 2.0 39 65 NaN 40 NaN NaN NaN 0.0 41 99 NaN NaN NaN 5.0 168 rows × 5 columns Out[14]: <AxesSubplot:xlabel='POSTED SPEED LIMIT'> FATAL INCAPACITATING NON\_INCAPACITATING 35000 NO\_INJURY 30000 25000 20000 15000 10000 5000 0 15 10 20 50 75 90 95 25 30 35 40 45 60 65 70 80 POSTED SPEED LIMIT import math F Is there a relationship between hit and run crashes and number of fatal injuries? hit run = df.groupby('POSTED SPEED LIMIT')['HIT AND RUN I'].count().reset index(name ratio = (sum(n inj speed['FATAL']) / sum(hit run['COUNT']))\*100 print('It appears that hit and run accidents are not strongly correlated with fatal ac print("There's an overall ratio of {:.2f}% of fatal incidents to hit-and-run accidents hit run tot = df.groupby(['HIT AND RUN I']).size().reset index(name='COUNT') hit run fatal = df.groupby('HIT AND RUN I')['INJURIES FATAL'].sum().reset index(name ratio2 = (sum(hit run fatal['SUM']) / sum(hit run tot['COUNT']))\*100 print("There's an overall chance of  $\{:.2f\}$ % to die if involved in a hit-and-run accide # lets calculate the correlation coefficient N = len(df) # total number of cases# x is count of all hit and run x = hit run tot.iloc[int(hit run tot[['COUNT']].idxmax())]['COUNT'] # y is count of all fatal injuries tot fat = df.groupby(['INJURIES FATAL']).size().reset index(name='COUNT') y = tot\_fat.iloc[int(hit\_run\_tot[['COUNT']].idxmax())]['COUNT'] # sum(xy) = sum(1\*1) = sum(y). x and y are 1 because we count them as 1 if Y, 0 if N numerator = (N\*sum(hit run fatal['SUM'])) - (x\*y) # sigma x\*y -> x\*y will be 1 only denominator = math.sqrt(((N\*x) - (x\*\*2))\*((N\*y) - (y\*\*2))) r = numerator / denominator print("The correlation coefficient p is {:.4f} which is a weak correlation.".format(r) It appears that hit and run accidents are not strongly correlated with fatal accident There's an overall ratio of 0.36% of fatal incidents to hit-and-run accidents There's an overall chance of 0.06% to die if involved in a hit-and-run accident The correlation coefficient p is -0.0052 which is a weak correlation. # Do intersection-related crashes result in more fatal injuries? # compare intersection deaths to non-intersection deaths intersection deaths = df[df.INTERSECTION RELATED I == 'Y'] sum deaths = intersection deaths['INJURIES FATAL'].sum() #sum deaths sum deaths tot = df['INJURIES FATAL'].sum() #sum deaths tot ratio = sum deaths / sum deaths tot #ratio print("There's a {:.2f} ratio intersection-deaths to total-deaths, which is quite sign print("{:.2f}% of deadly injuries are related to intersections, we can conclude that There's a 0.33 ratio intersection-deaths to total-deaths, which is quite significant. 32.75% of deadly injuries are related to intersections, we can conclude that intersect ions result in more deaths. # let's see how the accidents graph looks like for reasons we actually know # remove the unable to determine and non applicable clear reasons = reasons[reasons.PRIM CONTRIBUTORY CAUSE != 'UNABLE TO DETERMINE'] clear reasons = clear reasons[clear reasons.PRIM CONTRIBUTORY CAUSE != 'NOT APPLICABLE #plot the data clear reasons.plot(x='PRIM CONTRIBUTORY CAUSE', y='Count', kind='barh', figsize = (9,5) clear reasons.sort values(by='Count', ascending=False).head(10) Out[17]: <AxesSubplot:ylabel='PRIM CONTRIBUTORY CAUSE'> PRIM\_CONTRIBUTORY\_CAUSE Count 18 FAILING TO YIELD RIGHT-OF-WAY 52982 19 FOLLOWING TOO CLOSELY 51238 23 IMPROPER OVERTAKING/PASSING 22858 21 IMPROPER BACKING 21096 17 FAILING TO REDUCE SPEED TO AVOID CRASH 20751 22 IMPROPER LANE USAGE 18637 24 IMPROPER TURNING/NO SIGNAL 15986 12 DRIVING SKILLS/KNOWLEDGE/EXPERIENCE 14968 DISREGARDING TRAFFIC SIGNALS 6 8713 39 WEATHER 8439 WEATHER
VISION OBSCURED (SIGNS, TREE LIMBS, BUILDINGS, ETC.)
UNDER THE INFLUENCE OF ALCOHOL/DRUGS (USE WHEN ARREST IS EFFECTED)
TURNING RIGHT ON RED Count ROAD ENGINEERING/SURFACE/MARKING DEFECTS ROAD ENGINEERING/SOURALLE/MARKING DEFECTS

ROAD CONSTRUCTION/MAINTENANCE

RELATED TO BUS STOP

PHYSICAL CONDITION OF DRIVER

PASSING STOPPED SCHOOL BUS

OPERATING VEHICLE IN ERRATIC, RECKLESS, CARELESS, NEGLIGENT OR AGGRESSIVE MANNER

OPERATION OF THE PROPERTY OF THE PR OBSTRUCTED CROSSWALKS MOTORCYCLE ADVANCING LEGALLY ON RED LIGHT
IMPROPER TURNING/NO SIGNAL
IMPROPER OVERTAKING/PASSING
IMPROPER LANE USAGE CONTRIBUTORY CAUSE IMPROPER BACKING IMPROPER BACKING
HAD BEEN DRINKING (USE WHEN ARREST IS NOT MADE)
FOLLOWING TOO CLOSELY
FAILING TO YIELD RIGHT-OF-WAY
FAILING TO REDUCE SPEED TO AVOID CRASH
EXCEEDING SAFE SPEED FOR CONDITIONS
EXCEEDING AUTHORIZED SPEED LIMIT EXCEEDING AUTHORIZED SPEED LIMIT
EVASIVE ACTION DUE TO ANIMAL, OBJECT, NONMOTORIST
EQUIPMENT - VEHICLE CONDITION
DRIVING SKILLS/KNOWLEDGE/EXPERIENCE
DRIVING ON WRONG SIDE/WRONG WAY

DISTRACTION - OTHER ELECTRONIC DEVICE (NAVIGATION DEVICE, DVD PLAYER, ETC.)
DISTRACTION - FROM OUTSIDE VEHICLE
DISTRACTION - FROM INSIDE VEHICLE
DISREGARDING YIELD SIGN
DISREGARDING STOP SIGN
DISREGARDING TRAFFIC SIGNALS
DISREGARDING TOPS SIGNALS
DISREGARDING TOPS SIGNALS
DISREGARDING TOP SIGNALS
CELL PHONE USE OTHER THAN TEXTING
CELL PHONE USE OTHER THAN TEXTING
BICYCLE ADVANCING LEGALLY ON RED LIGHT BICYCLE ADVANCING LEGALLY ON RED LIGHT 10000 20000 30000 40000 50000 # let us explore what other details the top 2 most popular accidents reasons have # failing to yeld right-of-way right of way = df[df.PRIM CONTRIBUTORY CAUSE == 'FAILING TO YIELD RIGHT-OF-WAY'] #right of way.head(5) # let us see how many of these are actually reported to be intersaction related, no r. intersection = right of way.groupby(['INTERSECTION RELATED I']).size().reset index(nar print(intersection) not r o w = right of way.groupby(['NOT RIGHT OF WAY']).size().reset index(name='Count print(not r o w) INTERSECTION RELATED I Count 0 618 Ν 21090 NOT RIGHT OF WAY Count Ν 108 1 1077 # so far we know that most of the accidents reported are caused by a fail of yelding percent = intersection.iloc[int(intersection[['Count']].idxmax())]['Count']/clear reas print('{:.2f}% of accidents that fail to yeld right-of-way are reported on an intersec print('It appears that the places of the accidents underreported because the yelding # let's see how many deaths are caused by the fail of yelding right-of-way sum deaths = right of way['INJURIES FATAL'].sum() #sum deaths sum deaths tot = df['INJURIES FATAL'].sum() #sum deaths tot ratio2 = sum deaths / sum deaths tot print ("The ratio of fatal injuries due failing to yelding right-of-way to total fatal print ("Although deaths due intersection amount to {:.2f}%, deaths on intersections due 39.81% of accidents that fail to yeld right-of-way are reported on an intersection. It appears that the places of the accidents underreported because the yelding right-of -way happens mostly on an intersection. The ratio of fatal injuries due failing to yelding right-of-way to total fatal injurie s is 0.08 Although deaths due intersection amount to 32.75%, deaths on intersections due to fail ing to yeld right-of-way is 8.43% # let's look now at the second most popular type of accident # following too closely, this might happen for multiple reasons. Let's explore some of follow too close = df[df.PRIM CONTRIBUTORY CAUSE == 'FOLLOWING TOO CLOSELY'] # let's see what type of trafficway, time of the day, weather condition and roadway co traffic way = follow too close.groupby(['TRAFFICWAY TYPE']).size().reset index(name='( #traffic way.sort values(by='Count', ascending=False).head(10) time hour = follow too close.groupby(['CRASH HOUR']).size().reset index(name='Count') #time hour.sort values(by='Count', ascending=False).head(10) weather cond = follow too close.groupby(['WEATHER CONDITION']).size().reset index(name #weather cond.sort values(by='Count', ascending=False).head(10) road cond = follow too close.groupby(['ROADWAY SURFACE COND']).size().reset index(name #road cond.sort values(by='Count', ascending=False).head(10) print('It appears that most accidents of type "Follow too close" happen under these co t = traffic way.sort values(by='Count', ascending=False) tt = t.iloc[0]['TRAFFICWAY TYPE'] tn = t.iloc[0]['Count'] tpercent = tn/sum(traffic way['Count'])\*100 print('Trafficway: {}, with {} accidents, amounting to {:.2f}% of total accidents of w = weather cond.sort values(by='Count', ascending=False) ww = w.iloc[0]['WEATHER CONDITION'] wn = w.iloc[0]['Count'] wpercent = wn/sum(weather cond['Count'])\*100 print('Weather: {}, with {} accidents, amounting to {:.2f}% of total accidents of this r = road cond.sort values(by='Count', ascending=False) rr = r.iloc[0]['ROADWAY SURFACE COND'] rn = r.iloc[0]['Count'] rpercent = rn/sum(weather cond['Count'])\*100 print('Road Condition: {}, with {} accidents, amounting to {:.2f}% of total accidents print('Time of accidents:') time hour['Count'].plot() print('Contrary to popular belief, most accidents appear not to be because of the envi It appears that most accidents of type "Follow too close" happen under these condition s: Trafficway: NOT DIVIDED, with 26991 accidents, amounting to 52.68% of total accidents of this type. Weather: CLEAR, with 41507 accidents, amounting to 81.01% of total accidents of this t Road Condition: DRY, with 39133 accidents, amounting to 76.37% of total accidents of t his type. Time of accidents: Out[21]: <AxesSubplot:> Contrary to popular belief, most accidents appear not to be because of the environmen t, but rather because of human error. 4000 3000 2000 1000 10 # let's now explore the 'unable to determine' and 'non applicable' types of accident non det df = df[df.PRIM CONTRIBUTORY CAUSE == 'UNABLE TO DETERMINE'] #non det df.head(5) # lets compare the non determined to the determined accident types traffic way nd = non det df.groupby(['TRAFFICWAY TYPE']).size().reset index(name='Cour #traffic way nd.sort values(by='Count', ascending=False).head(20) time hour nd = non det df.groupby(['CRASH HOUR']).size().reset index(name='Count') #time hour.sort values(by='Count', ascending=False).head(10) weather cond nd = non det df.groupby(['WEATHER CONDITION']).size().reset index(name='( #weather cond.sort values(by='Count', ascending=False).head(10) road cond nd = non det df.groupby(['ROADWAY SURFACE COND']).size().reset index(name='( #road cond.sort values(by='Count', ascending=False).head(10) print('It appears that most accidents of type "UNABLE TO DETERMINE" happen under these values(by=  $tna = traffic_way_na.sort_$ ttnd = tnd.iloc[0]['TRAFFICWAY TYPE'] tnnd = tnd.iloc[0]['Count'] tndpercent = tnnd/sum(traffic way nd['Count'])\*100 print('Trafficway: {}, with {} accidents, amounting to {:.2f}% of total accidents of wnd = weather cond nd.sort values(by='Count', ascending=False) wwnd = wnd.iloc[0]['WEATHER CONDITION'] wnnd = wnd.iloc[0]['Count'] wndpercent = wnnd/sum(weather cond nd['Count'])\*100 print('Weather: {}, with {} accidents, amounting to {:.2f}% of total accidents of this rnd = road cond nd.sort values(by='Count', ascending=False) rrnd = rnd.iloc[0]['ROADWAY SURFACE COND'] rnnd = rnd.iloc[0]['Count'] rndpercent = rnnd/sum(road cond nd['Count'])\*100 print('Road Condition: {}, with {} accidents, amounting to {:.2f}% of total accidents print('Time of accidents:') time hour nd['Count'].plot() print('The "UNABLE TO DETERMINE" subset of accidents is relatively similtar to the de-It appears that most accidents of type "UNABLE TO DETERMINE" happen under these condit ions: Trafficway: NOT DIVIDED, with 79462 accidents, amounting to 44.64% of total accidents of this type. Weather: CLEAR, with 138237 accidents, amounting to 77.66% of total accidents of this Road Condition: DRY, with 129209 accidents, amounting to 72.59% of total accidents of this type. Time of accidents: Out[22]: <AxesSubplot:> The "UNABLE TO DETERMINE" subset of accidents is relatively similtar to the defined ty pe of accidents subset. 12000 10000 8000 6000 4000 2000 15 20 # as a final thought, let us see how death rate of this specific type of accident is # device condition and damage tot\_d\_nd = non\_det\_df['INJURIES FATAL'].sum() tot d = df['INJURIES FATAL'].sum() percent d nd = (tot d nd / tot d)\*100traffic dev death nd = non det df.groupby('TRAFFIC CONTROL DEVICE')['INJURIES FATAL'] #traffic dev death nd tdd nd = traffic dev death nd.sort values(by='DEATHS', ascending=False) a = tdd nd.iloc[0]['TRAFFIC CONTROL DEVICE'] b = tdd nd.iloc[0]['DEATHS'] dev death nd = non det df.groupby('DEVICE CONDITION')['INJURIES FATAL'].sum().reset in #dev death nd tdd\_nd2 = dev\_death\_nd.sort\_values(by='DEATHS', ascending=False) a2 = tdd nd2.iloc[1]['DEVICE CONDITION'] b2 = tdd nd2.iloc[1]['DEATHS'] dmg death nd = non det df.groupby('DAMAGE')['INJURIES FATAL'].sum().reset index(name= #dmg death nd tdd nd3 = dmg death nd.sort values(by='DEATHS', ascending=False) a3 = tdd nd3.iloc[0]['DAMAGE'] b3 = tdd nd3.iloc[0]['DEATHS'] print("There's been {} deaths by unspecified accident type out of {} total deaths. A print("Most of the accidents use traffic control: {}, total deaths: {}.".format(a, b) print("Second type of most accidents' devices: {}, total deaths: {}.".format(a2, b2)) print("Most of the accidents causing death also do damages: {}, {:.2f}% of the time." There's been 156.0 deaths by unspecified accident type out of 510.0 total deaths. A 3 0.59% of the total. Most of the accidents use traffic control: NO CONTROLS, total deaths: 106.0. Second type of most accidents' devices: FUNCTIONING PROPERLY, total deaths: 45.0. Most of the accidents causing death also do damages: OVER \$1,500, 85.90% of the time. In [24]: # let us see what are the times most deaths happen in this subset of accidents fatal nd = non det df[non det df['INJURIES FATAL'] > 0] fatal\_nd\_time = fatal\_nd.groupby(['CRASH\_HOUR']).size().reset\_index(name='Count') fatal nd time['Count'].plot() print ("As the graph shows, most of the deaths for accidents of unspecified reasons has Out[24]: <AxesSubplot:> As the graph shows, most of the deaths for accidents of unspecified reasons happen at night between 20 and 03 14 12 10 8 6 4 2 10 15