Project 4 Solutions

Kevin Choe

Collaborators: (Collaborators listed here. Include names, which part of the project you gave or sought help with, and how you helped or were helped.)

TA help: Summeth Guda

Online resources used: (List of links/resources (if any) here. Include web addresses, which part of the project the resource helped with, and how you were helped.)

Question 1

```
import pandas as pd
import csv
myDF = pd.read_csv("/class/datamine/data/craigslist/vehicles.csv")
myDF.head()
```

```
           id  ...      long
0  7119256118  ... -114.2690
1  7120880186  ... -123.8240
2  7115048251  ...  -81.9654
3  7119250502  ... -114.2710
4  7120433904  ...  -68.8963

[5 rows x 25 columns]
```

```
myDF.shape

#myDF = pd.DataFrame(columns = ['LAT', 'LONG', 'STATE'])

#pair = list(myDF.loc[:'LAT', 'LONG']].dropna().to_records(index=False))
```

```
(435849, 25)
```

```
stateDict = {}
for my_index, my_row in myDF.iterrows():
    stateDict[my_row["state"]] = []
for my_index, my_row in myDF.iterrows():
    stateDict[my_row["state"]].append( (my_row['lat'],my_row['long']))
print(stateDict.get("in")[0:2])
```

```
[(39.0295, -86.8675), (38.8585, -86.4806)]
```

Question 2

```
for mystate, value in stateDict.items():
  print(f'{mystate}:')
  for myindex, mytriple in enumerate(value):
    if (myindex % 5000 == 0):
      print(f'Lat: {mytriple[0]} Long: {mytriple[1]}')
```

```
az:
Lat: 34.4554 Long: -114.26899999999999
Lat: 33.5094 Long: -112.117
or:
Lat: 46.1837 Long: -123.824
Lat: 45.4072 Long: -122.625
Lat: 45.4931 Long: -122.781
Lat: 43.6671 Long: -121.484
sc:
Lat: 34.9352 Long: -81.9654
Lat: 32.78 Long: -79.99
me:
Lat: 44.4699 Long: -68.8963
fl:
Lat: 25.813000000000002 Long: -80.232
Lat: 27.4474 Long: -80.3512
Lat: 30.4564 Long: -87.2104
Lat: 29.0112 Long: -82.0325
Lat: 26.7492 Long: -80.0725
Lat: 28.7924 Long: -81.2313
Lat: 25.7665 Long: -80.2351
mt:
Lat: 45.6546 Long: -110.561
Lat: 47.7989 Long: -116.742
wi:
Lat: 46.4638 Long: -90.1588
Lat: 44.6105 Long: -88.0491
Lat: 44.482 Long: -88.0205
ia:
Lat: 41.6002 Long: -93.5701
Lat: 42.7407 Long: -93.2025
al:
Lat: 32.592 Long: -85.5189
Lat: 34.7257 Long: -87.6633
sd:
Lat: 44.0805 Long: -103.23100000000001
tx:
Lat: 32.9317 Long: -96.7465
Lat: 31.6725 Long: -97.1138
Lat: 32.9317 Long: -96.7465
Lat: 29.6183 Long: -95.19200000000001
Lat: 29.5189 Long: -98.6006
va:
Lat: nan Long: nan
Lat: 37.1387 Long: -76.5462
Lat: 38.0985 Long: -78.4646
nc:
Lat: 35.0552 Long: -80.8195
Lat: 33.7792 Long: -84.4118
Lat: 36.0345 Long: -79.7907
Lat: 35.5018 Long: -82.9913
```

```
ca:
Lat: 38.5443 Long: -122.807
Lat: 33.7586 Long: -116.96600000000001
Lat: 34.2271 Long: -118.46799999999999
Lat: 38.6088 Long: -121.40100000000001
Lat: 39.443000000000005 Long: -119.772
Lat: 38.6465 Long: -120.964
Lat: 29.8824 Long: -90.1227
Lat: 38.5892 Long: -121.406
Lat: 39.1234 Long: -121.611
Lat: 36.9774 Long: -122.03200000000001
ny:
Lat: 42.0854 Long: -76.0561
Lat: 41.5596 Long: -74.1764
Lat: 40.9166 Long: -73.7877
Lat: 43.2204 Long: -75.48100000000001
md:
Lat: 38.9263 Long: -76.7098
tn:
Lat: 35.8619 Long: -83.6706
Lat: 36.5356 Long: -82.3656
Lat: 33.7792 Long: -84.4118
ut:
Lat: 40.7466 Long: -111.939
ma:
Lat: 42.0681 Long: -71.3283
Lat: 41.7182 Long: -71.14
vt:
Lat: 44.5716 Long: -72.0288
ga:
Lat: 31.9566 Long: -83.7835
Lat: 36.8804 Long: -108.181
ak:
Lat: 64.8155 Long: -147.713
oh:
Lat: 39.5235 Long: -84.3931
Lat: 39.955999999999996 Long: -82.7714
Lat: 41.5749 Long: -83.9513
Lat: 40.0287 Long: -84.2077
wa:
Lat: 47.8865 Long: -122.26
Lat: 47.1991 Long: -122.315
Lat: 47.2047 Long: -122.484
mi:
Lat: 42.57 Long: -83.7484
Lat: 42.7851 Long: -83.7294
Lat: 42.4723 Long: -85.7035
ok:
Lat: 34.1971 Long: -97.1606
Lat: 35.5109 Long: -97.5796
pa:
Lat: 40.1305 Long: -75.0699
Lat: 40.285 Long: -78.8654
Lat: 40.3588 Long: -80.1029
```

```
id:
Lat: 47.6584 Long: -117.15700000000001
Lat: 47.6294 Long: -117.404
wy:
Lat: 42.8261 Long: -106.39
mn:
Lat: 48.6006 Long: -94.57
Lat: 45.3328 Long: -93.7369
ar:
Lat: 36.459 Long: -94.12100000000001
Lat: 37.1328 Long: -95.7856
wv:
Lat: 39.906 Long: -83.0829
ms:
Lat: 34.2098 Long: -86.75200000000001
mo:
Lat: 38.8023 Long: -90.6737
Lat: 36.669000000000004 Long: -93.2481
nj:
Lat: 40.068000000000005 Long: -74.8454
Lat: 40.6965 Long: -75.2266
ks:
Lat: 37.6421 Long: -97.3361
Lat: 38.7922 Long: -95.5616
hi:
Lat: 21.2939 Long: -157.83700000000002
il:
Lat: 40.5124 Long: -88.9883
Lat: 41.4885 Long: -90.321
Lat: 39.6753 Long: -89.7017
ri:
Lat: 41.7004 Long: -71.5183
ne:
Lat: 37.1328 Long: -95.7856
nv:
Lat: 40.7461 Long: -111.939
nd:
Lat: 47.9202 Long: -97.045
la:
Lat: 30.0005 Long: -90.1331
ct:
Lat: 41.6541 Long: -72.7758
nm:
Lat: 35.0712 Long: -106.525
co:
Lat: 40.1779 Long: -105.101
Lat: 39.676 Long: -105.009
Lat: 33.7792 Long: -84.4118
ky:
Lat: 36.9375 Long: -86.4481
de:
Lat: 39.4815 Long: -75.6832
in:
Lat: 39.0295 Long: -86.8675
```

```
Lat: 41.6741 Long: -86.2104
nh:
Lat: 42.769 Long: -71.4121
dc:
Lat: 38.9263 Long: -76.7098
```

Question 3

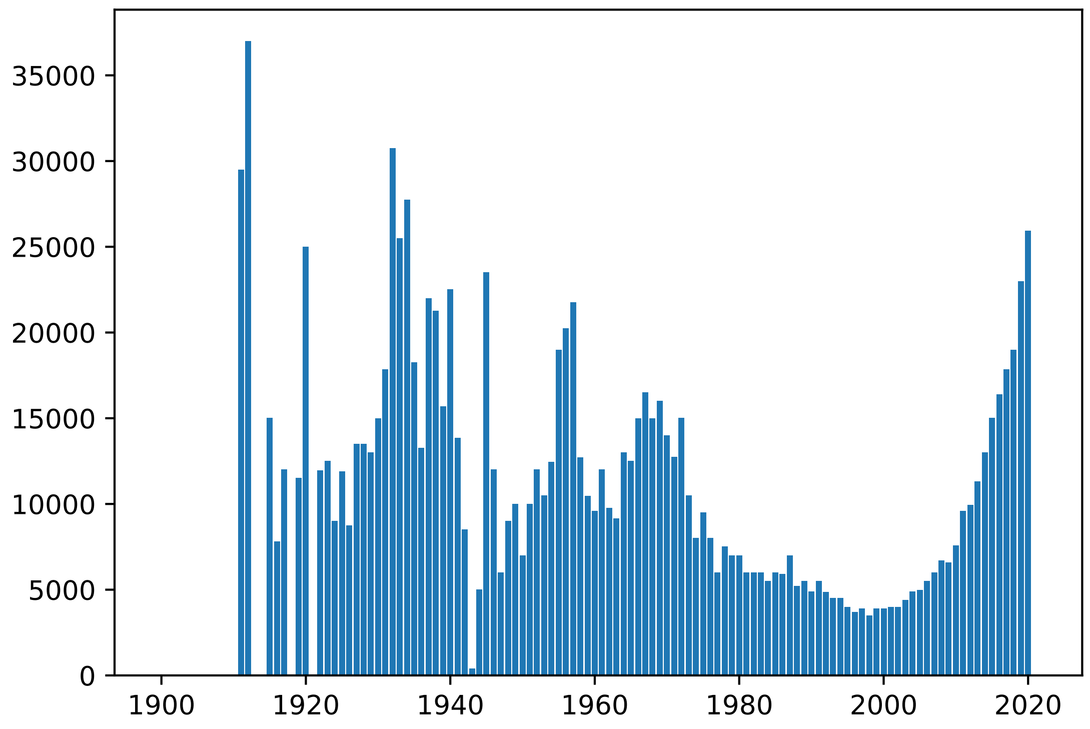
```
from collections import defaultdict
import matplotlib.pyplot as plt

my_list = list(myDF.loc[:, ["year", "price",]].dropna().to_records(index=False))

dictionary = defaultdict(list)

for year, price in my_list:
    dictionary[year].append(float(price))

listyear = []
listprice = []

import statistics
for year, price in sorted(dictionary.items()):
  listyear.append(year)
  bob = statistics.median(dictionary[year])
  listprice.append(bob)
  print(f'Year {year} Price {bob}')
```

```
Year 1900.0 Price 0.0
Year 1911.0 Price 29500.0
Year 1912.0 Price 36980.0
Year 1913.0 Price 0.0
Year 1915.0 Price 15000.0
Year 1916.0 Price 7800.0
Year 1917.0 Price 12000.0
Year 1919.0 Price 11500.0
Year 1920.0 Price 25000.0
Year 1922.0 Price 11950.0
Year 1923.0 Price 12500.0
Year 1924.0 Price 9000.0
Year 1925.0 Price 11900.0
Year 1926.0 Price 8750.0
Year 1927.0 Price 13500.0
Year 1928.0 Price 13500.0
Year 1929.0 Price 12999.0
Year 1930.0 Price 14980.0
Year 1931.0 Price 17850.0
Year 1932.0 Price 30750.0
Year 1933.0 Price 25500.0
Year 1934.0 Price 27750.0
Year 1935.0 Price 18250.0
Year 1936.0 Price 13250.0
Year 1937.0 Price 22000.0
Year 1938.0 Price 21250.0
Year 1939.0 Price 15700.0
Year 1940.0 Price 22500.0
Year 1941.0 Price 13850.0
Year 1942.0 Price 8500.0
Year 1943.0 Price 400.0
Year 1944.0 Price 5000.0
Year 1945.0 Price 23500.0
Year 1946.0 Price 12000.0
Year 1947.0 Price 6000.0
Year 1948.0 Price 9000.0
Year 1949.0 Price 10000.0
Year 1950.0 Price 7000.0
Year 1951.0 Price 10000.0
Year 1952.0 Price 12000.0
Year 1953.0 Price 10500.0
Year 1954.0 Price 12450.0
Year 1955.0 Price 18995.0
Year 1956.0 Price 20250.0
Year 1957.0 Price 21750.0
Year 1958.0 Price 12700.0
Year 1959.0 Price 10450.0
Year 1960.0 Price 9600.0
Year 1961.0 Price 12000.0
Year 1962.0 Price 9750.0
Year 1963.0 Price 9150.0
Year 1964.0 Price 13000.0
Year 1965.0 Price 12500.0
Year 1966.0 Price 14990.0
Year 1967.0 Price 16500.0
Year 1968.0 Price 14990.0
Year 1969.0 Price 16000.0
Year 1970.0 Price 14000.0
Year 1971.0 Price 12750.0
Year 1972.0 Price 15000.0
Year 1973.0 Price 10500.0
Year 1974.0 Price 8000.0
Year 1975.0 Price 9500.0
Year 1976.0 Price 8000.0
Year 1977.0 Price 6000.0
Year 1978.0 Price 7500.0
Year 1979.0 Price 7000.0
Year 1980.0 Price 7000.0
Year 1981.0 Price 6000.0
Year 1982.0 Price 6000.0
Year 1983.0 Price 6000.0
Year 1984.0 Price 5500.0
Year 1985.0 Price 6000.0
Year 1986.0 Price 5900.0
Year 1987.0 Price 7000.0
Year 1988.0 Price 5200.0
Year 1989.0 Price 5500.0
Year 1990.0 Price 4895.0
Year 1991.0 Price 5500.0
Year 1992.0 Price 4862.5
Year 1993.0 Price 4500.0
Year 1994.0 Price 4498.0
Year 1995.0 Price 3995.0
Year 1996.0 Price 3700.0
Year 1997.0 Price 3895.0
Year 1998.0 Price 3495.0
Year 1999.0 Price 3900.0
Year 2000.0 Price 3900.0
Year 2001.0 Price 3988.0
Year 2002.0 Price 3990.0
Year 2003.0 Price 4390.0
Year 2004.0 Price 4879.0
Year 2005.0 Price 4990.0
Year 2006.0 Price 5499.0
Year 2007.0 Price 5999.0
Year 2008.0 Price 6695.0
Year 2009.0 Price 6595.0
Year 2010.0 Price 7586.0
Year 2011.0 Price 9600.0
Year 2012.0 Price 9950.0
Year 2013.0 Price 11295.0
Year 2014.0 Price 12998.0
Year 2015.0 Price 15000.0
Year 2016.0 Price 16380.5
Year 2017.0 Price 17833.0
Year 2018.0 Price 18995.0
Year 2019.0 Price 22975.0
Year 2020.0 Price 25920.0
Year 2021.0 Price 37.0
```

```
plt.bar(listyear, listprice)
```

```
<BarContainer object of 109 artists>
```

```
plt.show()
```



```
plt.close()
```

Question 4

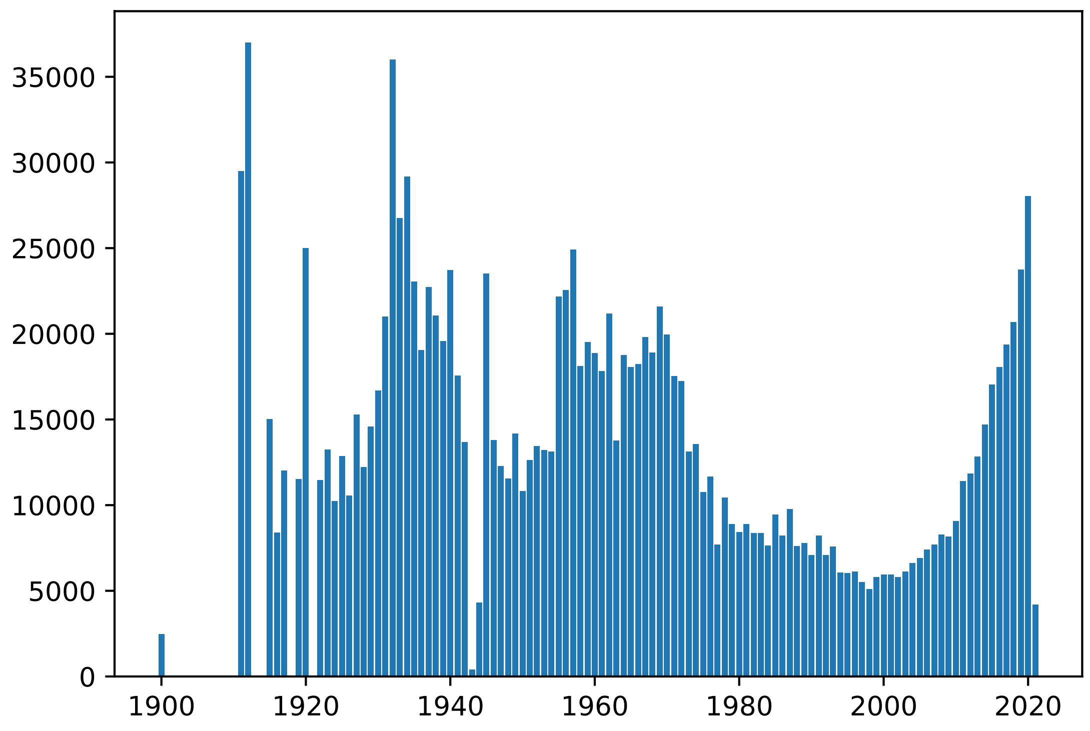
```
dictionary = defaultdict(list)

for year, price in my_list:
  if (price) < 200000:
    dictionary[year].append(price)

listyear = []
listprice = []

import statistics
for year, price in sorted(dictionary.items()):
  listyear.append(year)
  bob = statistics.mean(dictionary[year])
  listprice.append(bob)
  print(f'Year {year} Price {bob}')
```

```
Year 1900.0 Price 2473
Year 1911.0 Price 29500
Year 1912.0 Price 36980
Year 1913.0 Price 0
Year 1915.0 Price 15000
Year 1916.0 Price 8400
Year 1917.0 Price 12000
Year 1919.0 Price 11500
Year 1920.0 Price 25000
Year 1922.0 Price 11450
Year 1923.0 Price 13222
Year 1924.0 Price 10222
Year 1925.0 Price 12849
Year 1926.0 Price 10545
Year 1927.0 Price 15276
Year 1928.0 Price 12199
Year 1929.0 Price 14581
Year 1930.0 Price 16677
Year 1931.0 Price 21008
Year 1932.0 Price 35999
Year 1933.0 Price 26752
Year 1934.0 Price 29159
Year 1935.0 Price 23052
Year 1936.0 Price 19043
Year 1937.0 Price 22732
Year 1938.0 Price 21064
Year 1939.0 Price 19566
Year 1940.0 Price 23698
Year 1941.0 Price 17551
Year 1942.0 Price 13684
Year 1943.0 Price 400
Year 1944.0 Price 4308
Year 1945.0 Price 23500
Year 1946.0 Price 13791
Year 1947.0 Price 12261
Year 1948.0 Price 11535
Year 1949.0 Price 14164
Year 1950.0 Price 10814
Year 1951.0 Price 12614
Year 1952.0 Price 13449
Year 1953.0 Price 13194
Year 1954.0 Price 13114
Year 1955.0 Price 22155
Year 1956.0 Price 22542
Year 1957.0 Price 24921
Year 1958.0 Price 18113
Year 1959.0 Price 19523
Year 1960.0 Price 18873
Year 1961.0 Price 17821
Year 1962.0 Price 21167
Year 1963.0 Price 13747
Year 1964.0 Price 18746
Year 1965.0 Price 18039
Year 1966.0 Price 18236
Year 1967.0 Price 19812
Year 1968.0 Price 18893
Year 1969.0 Price 21577
Year 1970.0 Price 19938
Year 1971.0 Price 17534
Year 1972.0 Price 17242
Year 1973.0 Price 13106
Year 1974.0 Price 13558
Year 1975.0 Price 10762
Year 1976.0 Price 11652
Year 1977.0 Price 7690
Year 1978.0 Price 10436
Year 1979.0 Price 8873
Year 1980.0 Price 8415
Year 1981.0 Price 8900
Year 1982.0 Price 8367
Year 1983.0 Price 8375
Year 1984.0 Price 7645
Year 1985.0 Price 9426
Year 1986.0 Price 8210
Year 1987.0 Price 9773
Year 1988.0 Price 7598
Year 1989.0 Price 7788
Year 1990.0 Price 7080
Year 1991.0 Price 8213
Year 1992.0 Price 7067
Year 1993.0 Price 7568
Year 1994.0 Price 6051
Year 1995.0 Price 6012
Year 1996.0 Price 6121
Year 1997.0 Price 5504
Year 1998.0 Price 5097
Year 1999.0 Price 5796
Year 2000.0 Price 5941
Year 2001.0 Price 5928
Year 2002.0 Price 5797
Year 2003.0 Price 6123
Year 2004.0 Price 6603
Year 2005.0 Price 6907
Year 2006.0 Price 7393
Year 2007.0 Price 7695
Year 2008.0 Price 8283
Year 2009.0 Price 8150
Year 2010.0 Price 9054
Year 2011.0 Price 11398
Year 2012.0 Price 11819
Year 2013.0 Price 12814
Year 2014.0 Price 14688
Year 2015.0 Price 17042
Year 2016.0 Price 18058
Year 2017.0 Price 19372
Year 2018.0 Price 20665
Year 2019.0 Price 23729
Year 2020.0 Price 28025
Year 2021.0 Price 4176
```

```
plt.bar(listyear, listprice)
```

```
<BarContainer object of 109 artists>
```

```
plt.show()
```



```
plt.close()
```

Question 5

```
import statistics

my_list = list(myDF.loc[:, ["state", "price"]].dropna().to_records(index=False))
myIndianaprices = [price for (state, price) in my_list if state.upper() == "IN"]
statistics.mean(myIndianaprices)
```

```
34428
```

```
myMidwestprices = [price for (state, price) in my_list if state.upper() == "IN" or state.upper() == "MI
statistics.mean(myMidwestprices)
```

```
22719
```

```
my_list = list(myDF.loc[:, ["manufacturer", "year", "price",]].dropna().to_records(index=False))
mycarprices = [price for (manufacturer, year, price) in my_list if manufacturer == "honda" and year >= 
statistics.mean(mycarprices)
```

```
12242
```

Question 6

```python
import spacy

# get list of descriptions
my_list = list(myDF.loc[:, ["description",]].dropna().to_records(index=False))
my_list = [m[0] for m in my_list]

# load the pre-built spacy model
nlp = spacy.load("en_core_web_lg")

# apply the model to a description
doc = nlp(my_list[0])

for doc in my_list[:10]:
    d = nlp(doc)
    for entity in d.ents:
        print(entity.text.encode('ascii', errors='ignore'), entity.label_)
```

```
b'2012' DATE
b'JEEP' ORG
b'4CYL' ORDINAL
b'AC' ORG
b'STEREO' ORG
b'132' CARDINAL
b'3495' MONEY
b'LHC' ORG
b'TED' PERSON
b'951' CARDINAL
b'712-9436' CARDINAL
b'US' GPE
b'2014' DATE
b'BMW' ORG
b'328i' PRODUCT
b'about seven months ago' DATE
b'a few months ago' DATE
b'the past year' DATE
b'Graf' PERSON
b'TODAY' DATE
b'243' MONEY
b'13,750.00' MONEY
b'01' CARDINAL
b'DODGE' ORG
b'AUT' ORG
b'04' CARDINAL
b'2004' DATE
b'Chevy' ORG
b'Colorado' GPE
b'ONLY 54000 ORIGINAL MILES' QUANTITY
b'3.5L' CARDINAL
b'5cyl' ORG
b'3.73' CARDINAL
b'4L60E' PRODUCT
b'AM/FM/CD' ORG
```

```
b'4600 lb' QUANTITY
b'GVW' ORG
b'Dark Pewter' ORG
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'SEARSPORT MOTOR COMPANY' ORG
b'100%' PERCENT
b'FOUR' CARDINAL
b'MAINE' GPE
b'500.00' MONEY
b'2' CARDINAL
b'1 MONTH' DATE
b'14 DAY' DATE
b'3 MONTH/ 3000 MILE' QUANTITY
b'50-80%' PERCENT
b'DODGE' ORG
b'FORD' ORG
b'CHEVY' ORG
b'GMC' ORG
b'SUBARU' ORG
b'BMW' ORG
b'SATURN' ORG
b'CHRYSLER' ORG
b'5' CARDINAL
b'US' GPE
b'1-800-979' QUANTITY
b'US' GPE
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'' ORG
b'100' CARDINAL
b'today' DATE
b'today' DATE
b'603' CARDINAL
b'Carfax' ORG
b'100%' PERCENT
b'603-382-1400' CARDINAL
b'www.jtsautoandtruck.com' CARDINAL
b'as little as $500' MONEY
b'today' DATE
b'minutes' TIME
b'603' CARDINAL
b'just minutes' TIME
b'2014' DATE
b'Ford' ORG
b'F-150' PRODUCT
b'_ F150' PRODUCT
b'f150' PRODUCT
b'_ F' PRODUCT
b'CREW CAB Pickup' PRODUCT
b'603' CARDINAL
b'605' CARDINAL
b'Salem' GPE
b'NH 03079Copy & Paste' ORG
```

```
b'Year' DATE
b'2014' DATE
b'Ford' ORG
b'F-150' CARDINAL
b'4X4' QUANTITY
b'4X4' CARDINAL
b'4WD' CARDINAL
b'V8\t   \tDescription' PRODUCT
b'Ford' ORG
b'F-150' PRODUCT
b'_ F150_ f150' PRODUCT
b'_ F150' PRODUCT
b'US' GPE
b'SUV' ORG
b'van' PERSON
b'WWW.HUSSONMOTORS.COM' PERSON
b'24 hours' TIME
b'CREDIT!!!So' GPE
b'first' ORDINAL
b'HUSSON MOTORS' FAC
b'100%' PERCENT
b'FOUR' CARDINAL
b'500.00' MONEY
b'2' CARDINAL
b'20 DAY' DATE
b'30 DAY' DATE
b'1000 MILE' QUANTITY
b'50-80%' PERCENT
b'DODGE' ORG
b'FORD' ORG
b'CHEVY' ORG
b'GMC' ORG
b'SUBARU' ORG
b'BMW' ORG
b'CHRYSLER' ORG
b'HONDA' ORG
b'NISSAN' ORG
b'TOYOTA' ORG
b'US' GPE
b'1' CARDINAL
b'US' GPE
b'US' GPE
b'8' CARDINAL
b'603' CARDINAL
b'605' CARDINAL
b'Ford' ORG
b'F-150 4X4' PRODUCT
b'Husson Motors' ORG
b'VEHICLE?Of' PRODUCT
b'Car' PRODUCT
b'Ford' ORG
b'F-150 4X4' PRODUCT
b'CREW CAB' PRODUCT
b'2014, 2015,' DATE
```

```
b'2016' DATE
b'2013' DATE
b'2012' DATE
b'2011' DATE
b'2010' DATE
b'Ford' ORG
b'F-150' PRODUCT
b'Crown Victoria, Edge' ORG
b'E150' PRODUCT
b'Escape Hybrid' ORG
b'Escape' PRODUCT
b'F150' PRODUCT
b'F350' PRODUCT
b'F450' PRODUCT
b'Expedition' PRODUCT
b'Explorer' PRODUCT
b'Explorer Sport Trac' ORG
b'F250' PRODUCT
b'Flex' ORG
b'Focus, Fusion' ORG
b'Mustang' ORG
b'Ranger' PRODUCT
b'Taurus' PRODUCT
b'E250' PRODUCT
b'E350 Super Duty, Fusion Hybrid, Transit Connect' PRODUCT
b'Fiesta' PRODUCT
b'Focus Electric' ORG
b'Focus ST' ORG
b'C-Max Hybrid' ORG
b'C-Max Energi' ORG
b'Fusion Energi' ORG
b'Sedan Police Interceptor' ORG
b'Today' DATE
b'603' CARDINAL
b'605' CARDINAL
b'Salem' GPE
b'NH 03079Copy & Paste' ORG
b'2014 14' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'4X4 CREW' PRODUCT
b'4X4' PRODUCT
b'CREW CAB Pickup' PRODUCT
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'2014' CARDINAL
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'f-150 F-150 4X4' PRODUCT
b'Ford' ORG
```

```
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4 CREW' FAC
b'Carfax' ORG
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'CREW CAB' PRODUCT
b'2014' CARDINAL
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'2014' CARDINAL
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'2014' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'Husson Motors' ORG
b'today' DATE
b'603' CARDINAL
b'605-1616' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'2016' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'4X4' PRODUCT
b'CREW CAB Pickup' PRODUCT
b'2017' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'150' CARDINAL
b'F150 f150 F150 4X4' PRODUCT
b'Sale F-150 F150 f150' PRODUCT
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'Carfax 4X4' ORG
b'Ford' ORG
```

```
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'4X4 CREW CAB' PRODUCT
b'Pickup 4X4' PRODUCT
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'4X4' PRODUCT
b'603' CARDINAL
b'605' CARDINAL
b'Husson Motors' ORG
b'2013 2012' DATE
b'2011' DATE
b'Ford' ORG
b'F-150 F150 f150' PRODUCT
b'F 150' PRODUCT
b'New 4X4' ORG
b'2010 2009 2008 2007 2006 2005' DATE
b'2005' DATE
b'Ford' ORG
b'Automatic' PRODUCT
b'V6' LAW
b'62,800' CARDINAL
b'PS' ORG
b'Power Disc Brakes' ORG
b'Cruise' PRODUCT
b'500 6' CARDINAL
b'Pony' ORG
b'Premium' ORG
b'3' CARDINAL
b'between mid April and mid November' DATE
b'the remaining months' DATE
b'summer' DATE
b'8500' MONEY
b'Sunrise Auto SalesTry our Delivery Service' ORG
b'2004' DATE
b'Sunrise' PERSON
b'The Centers for Disease Control & Prevention' ORG
b'CDC' ORG
b'one' CARDINAL
b'724985' CARDINAL
b'2000' DATE
b'Jeep' ORG
b'TODAY' DATE
b'503' CARDINAL
b'461' CARDINAL
b'3500' CARDINAL
b'VIN' PRODUCT
b'CUMMINS' ORG
b'116,515    ' QUANTITY
b'TRANS' ORG
b'Diesel Turbo I6' ORG
b'800ft' QUANTITY
b'6' CARDINAL
```

```
b'100%' PERCENT
b'3 MinuteOnline Credit Application' TIME
b'' ORG
b'' ORG
b'This Vehicle was Hand Selected, Safety Inspected' WORK_OF_ART
b'2012' DATE
b'3500' CARDINAL
b'Oregon' GPE
b'2004' DATE
b'Sunrise Auto Sales' FAC
b'Credit Union Direct Lending' ORG
b'as low as 3.89%' PERCENT
b'760' CARDINAL
b'Credit union' GPE
b'that!Please' PERSON
b'Antenna' PERSON
b'USB' ORG
b'30' CARDINAL
b'Infinity, Radio' ORG
b'AM/FM' ORG
b'SiriusXM' ORG
b'1' CARDINAL
b'10' CARDINAL
b'14.2' CARDINAL
b'14.1' CARDINAL
b'Dash' PRODUCT
b'Delivery Service' ORG
b'SUV' ORG
b'metro' LOC
b'Southwest Washington' LOC
b'Idaho' GPE
b'Montana California' GPE
b'16001' DATE
b'SE McLoughlin Blvd Milwaukie' ORG
b'97267' CARDINAL
b'DA1146  ' PERSON
b'today' DATE
b'971' CARDINAL
b'220' CARDINAL
b'Optional Credit or Liability Insurance' ORG
b'State Documentary Service Fee' ORG
b'Dealership' ORG
b'Dealership' ORG
b'over 80' CARDINAL
b'Portland' GPE
b'F-350' FAC
b'dually drw' PRODUCT
b'f150' PRODUCT
b'f-350' PRODUCT
b'2500 3500' CARDINAL
b'7.3 6.7' CARDINAL
b'6.7l' CARDINAL
b'6.0' CARDINAL
b'7.3l' CARDINAL
```

```
b'5.9' CARDINAL
b'5.9l' CARDINAL
b'cummins' ORG
b'Duramax' ORG
b'6.6 6.7' PRODUCT
b'disel' PRODUCT
b'laramie' FAC
b'portland metro' FAC
b'50 miles' QUANTITY
b'miles' QUANTITY
b'four' CARDINAL
b'4' CARDINAL
b'Long Horn' LOC
b'Powerstroke' PRODUCT
b'Cummings, Cumins' ORG
b'2008' DATE
b'08, 2009' DATE
b'09, 2010' DATE
b'10, 2011' DATE
b'11, 2012' DATE
b'12' DATE
b'2013' DATE
b'13' DATE
b'2014' DATE
b'14, 2015' DATE
b'15, 2016' DATE
b'16' CARDINAL
b'D&M Motors LLC' ORG
b'971' CARDINAL
b'803' CARDINAL
b'2004' DATE
b'Honda' ORG
b'Odyssey EX-L' PRODUCT
b'102k!!!Our' CARDINAL
b'D&M Motors' FAC
b'503' CARDINAL
b'549' CARDINAL
b'the following:- History-' ORG
b'off)- Warranty' PERSON
b'503' CARDINAL
b'D&M Motors' ORG
b'2004' DATE
b'Honda' PRODUCT
b'Odyssey ' PRODUCT
b'R012679' ORG
b'VIN' ORG
b'102,415' MONEY
b'TRANS' ORG
b'Automatic    ' PRODUCT
b'3L' CARDINAL
b'NA' ORG
b'V6' PRODUCT
b'SOHC' ORG
b'24V' ORG
```

```
b'6' CARDINAL
b'FWD      ' PRODUCT
b'16' CARDINAL
b'2004' DATE
b'HondaOdyssey' PRODUCT
b'971' CARDINAL
b'803' CARDINAL
b'ADDRESSD&M Motors LLC' ORG
b'842' CARDINAL
b'N Lombard St Portland' ORG
b'97217' CARDINAL
b'Optional Credit or Liability Insurance' ORG
b'State Documentary Service Fee' ORG
b'Dealership' ORG
b'Dealership' ORG
b'Automatic' PRODUCT
b'FWD' ORG
b'Van' GPE
b'2000' DATE
b'00, 2001' DATE
b'01, 2002' DATE
b'02, 2003' DATE
b'03, 2004' DATE
b'04, 2005' DATE
b'05, 2006' DATE
b'06, 2007' DATE
b'07, 2008' DATE
b'08' CARDINAL
```

```python
for doc in my_list[:10]:
    d = nlp(doc)
    for entity in d.ents:
        if entity.label_ == "CARDINAL":
            print(entity.text.encode('ascii', errors='ignore'), entity.label_)
```

```
b'132' CARDINAL
b'951' CARDINAL
b'712-9436' CARDINAL
b'01' CARDINAL
b'04' CARDINAL
b'3.5L' CARDINAL
b'3.73' CARDINAL
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'FOUR' CARDINAL
b'2' CARDINAL
b'5' CARDINAL
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'100' CARDINAL
b'603' CARDINAL
b'603-382-1400' CARDINAL
b'www.jtsautoandtruck.com' CARDINAL
b'603' CARDINAL
b'603' CARDINAL
```

```
b'605' CARDINAL
b'F-150' CARDINAL
b'4X4' CARDINAL
b'4WD' CARDINAL
b'FOUR' CARDINAL
b'2' CARDINAL
b'1' CARDINAL
b'8' CARDINAL
b'603' CARDINAL
b'605' CARDINAL
b'603' CARDINAL
b'605' CARDINAL
b'150' CARDINAL
b'2014' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'2014' CARDINAL
b'150' CARDINAL
b'2014' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'603' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'150' CARDINAL
b'603' CARDINAL
b'605' CARDINAL
b'62,800' CARDINAL
b'500 6' CARDINAL
b'3' CARDINAL
b'one' CARDINAL
b'724985' CARDINAL
b'503' CARDINAL
b'461' CARDINAL
b'3500' CARDINAL
b'6' CARDINAL
b'3500' CARDINAL
b'760' CARDINAL
b'30' CARDINAL
b'1' CARDINAL
b'10' CARDINAL
b'14.2' CARDINAL
b'14.1' CARDINAL
b'97267' CARDINAL
b'971' CARDINAL
b'220' CARDINAL
b'over 80' CARDINAL
b'2500 3500' CARDINAL
b'7.3 6.7' CARDINAL
b'6.7l' CARDINAL
b'6.0' CARDINAL
b'7.3l' CARDINAL
```

```
b'5.9' CARDINAL
b'5.91' CARDINAL
b'four' CARDINAL
b'4' CARDINAL
b'16' CARDINAL
b'971' CARDINAL
b'803' CARDINAL
b'102k!!!Our' CARDINAL
b'503' CARDINAL
b'549' CARDINAL
b'503' CARDINAL
b'3L' CARDINAL
b'6' CARDINAL
b'16' CARDINAL
b'971' CARDINAL
b'803' CARDINAL
b'842' CARDINAL
b'97217' CARDINAL
b'08' CARDINAL
```

```
for doc in my_list[:10]:
    d = nlp(doc)
    for entity in d.ents:
        if entity.label_ == "CARDINAL" and len(entity.text) in [7, 8, 10, 11, 12, 14]:
            print(entity.text.encode('ascii', errors='ignore'), entity.label_)
```

```
b'712-9436' CARDINAL
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'207.548.6500' CARDINAL
b'207.407.5598' CARDINAL
b'603-382-1400' CARDINAL
b'over 80' CARDINAL
b'7.3 6.7' CARDINAL
b'102k!!!Our' CARDINAL
```

```
import re
pattern = '\(?([0-9]{3})?\)?[-.]?([0-9]{3})[-.]?([0-9]{4})'
for doc in my_list[:10]:
    if matches := re.finditer(pattern, doc):
        for match in matches:
            print(match.group())
#It filters through all the data and keeping instances that match the pattern in line 119. The output i
```

```
712-9436
207.548.6500
207.407.5598
800-979-4222
207.548.6500
207.407.5598
603-382-1400
603-382-1400
603-382-1400
603-382-1400
603-382-1400
```

```
603-382-1400
603-382-1400
603-382-1400
605-1616
1112796
603-965-2721
605-1616
605-1616
1112796
605-1616
605-1616
(503)855-6226
461-6101
220-7571
803-4990
2572176
549-4776
503-549-4776
803-4990
803-4990
```

Pledge

By submitting this work I hereby pledge that this is my own, personal work. I've acknowledged in the designated place at the top of this file all sources that I used to complete said work, including but not limited to: online resources, books, and electronic communications. I've noted all collaboration with fellow students and/or TA's. I did not copy or plagiarize another's work.

> As a Boilermaker pursuing academic excellence, I pledge to be honest and true in all that I do. Accountable together – We are Purdue.