Xueru Xie, Eric Wang, Alex Levine

Data Hack 2020

Beginner Track

08 February 2020

An Overview of San Diego in the 1970s

Looking back at the 1970s, San Diego was very different than today. Given both "SD1970_housing " and "SD1970_population" csv files, we utilized some data analysis and visualization to show what San Diego was like in the 1970s. We wanted to explore the population distribution, gender distribution, demographics, and relationship between San Diego house units and total aggregate values for house owners.

Data Cleaning

To start off, we opened both "SD1970_housing" and "SD1970_population" csv and did a thorough cleaning. We were able to strip "$, -, …" signs inside every value and convert each number from string to float/integers so that we could perform operations on them. As we cleaned the data, we realized that there were some rows that had missing values of the city name, so we decided to remove all of these rows, as they introduced potentially mislabelled data. If we included these nameless rows, our result potentially loses accuracy since we can't confirm the data's correctness.

Clean population csv

```
In [3]: population = pd.read_csv("SD1970_population.csv")

        population = population.dropna()
        population = population.drop(population[population["White persons"] == "..."].index)

        TractName = population['Census Tract Name']
        BlockGroup = population['Block Group']
        Place_Name = population['Place Name']

        population = population.drop('Census Tract Name',axis= 1)
        population = population.drop('Block Group',axis= 1)
        population = population.drop('Place Name',axis= 1)

        for i in population.columns:
            #population[i] = population[i].str.replace(',',"").astype(int)
            population[i] = population[i].astype(str).str.replace(',',"")
            population[i] = population[i].replace(' ... ',"0")
            population[i] = population[i].replace('...',"0")
            population[i] = population[i].astype(float)


        population.insert(0, 'Place Name', Place_Name)
        population.insert(0, 'Block Group', BlockGroup)
        population.insert(0, 'Census Tract Name', TractName)

        population.head()
```

Out[3]:

	Census Tract Name	Block Group	Place Name	Total persons	Male persons age under 5 years	Male persons age 5 years	Male persons age 6 years	Male persons age 7-9 years	Male persons age 10-13 years	Male persons age 14 years	...	Husband-wife family members 65 and over, none under 18	Husband-wife family members under 18 and 65 and over	Other family with male head no members under 18 or 65 and over	Other family with male head members under 18, none 65 and over	Other family with male head members 65 and over, none under 18	Othe famil with mal hea member under 1 and 6 and ove
0	Census Tract 1	1	San Diego	901.0	31.0	6.0	9.0	29.0	32.0	9.0	...	2.0	3.0	2.0	0.0	3.0	0.
1	Census Tract 1	2	San Diego	683.0	28.0	2.0	9.0	14.0	21.0	6.0	...	3.0	5.0	0.0	1.0	1.0	1.
2	Census Tract 1	3	San Diego	532.0	18.0	4.0	4.0	15.0	26.0	6.0	...	2.0	2.0	1.0	1.0	0.0	1.
3	Census Tract 1	4	San Diego	421.0	9.0	3.0	5.0	5.0	7.0	3.0	...	0.0	1.0	0.0	1.0	0.0	0.
4	Census Tract 1	5	San Diego	489.0	12.0	2.0	4.0	6.0	13.0	4.0	...	4.0	1.0	1.0	1.0	0.0	0.

5 rows × 155 columns

Clean housing csv

```
In [4]: housing = pd.read_csv("SD1970_housing.csv")


        housing = housing.dropna()
        housing = housing.drop(housing[housing["Total owner occupied real $ aggregate (total) value of housing units"] == "..."].index)
        tract_name = housing['Tract name']
        housing = housing.drop("Tract name", axis= 1)
        Block_group = housing['Block group']
        housing = housing.drop("Block group", axis= 1)
        place_name = housing['Place Name']
        housing = housing.drop('Place Name',axis= 1)


        for i in housing.columns:
            housing[i] = housing[i].astype(str).str.replace(',',"")
            housing[i] = housing[i].str.replace('$',"")
            housing[i] = housing[i].replace(' ... ',"0")
            housing[i] = housing[i].replace('...',"0")
            housing[i] = housing[i].replace('  -  ',"0")
            housing[i] = housing[i].astype(float)

        housing.insert(0, 'Place Name', place_name)
        housing.insert(0, 'Block group', Block_group)
        housing.insert(0, 'Tract name', tract_name)
        housing.head()
```

Out[4]:

	Tract name	Block group	Place Name	Total housing units	Total owner occupied real $ aggregate (total) value of housing units	Black owner occupied real $ aggregate (total) value of housing units	Vacant for sale only real $ aggregate (total) value of housing units	Total owner occupied average $ value of housing units	Black owner occupied average $ value of housing units	Vacant for sale only average $ value of housing units	...	Renter occupied housing units with 1.01 or more persons per room	Total Black occupied housing units with 1.01 or more persons per room	Black owner occupied housing units with 1.01 or more persons per room	Black renter occupied housing units with 1.01 or more persons per room	Persons in occupied housing units with all plumbing facilities	p
0	Census Tract 1	1	San Diego	302.0	7783750.0	0.0	0.0	32298.0	0.0	0.0	...	10.0	0.0	0.0	0.0	897.0	
1	Census Tract 1	2	San Diego	234.0	5720000.0	0.0	0.0	29485.0	0.0	0.0	...	11.0	0.0	0.0	0.0	683.0	
2	Census Tract 1	3	San Diego	176.0	5721250.0	0.0	0.0	37151.0	0.0	0.0	...	13.0	0.0	0.0	0.0	532.0	
3	Census Tract 1	4	San Diego	159.0	5633750.0	0.0	0.0	44712.0	0.0	0.0	...	9.0	0.0	0.0	0.0	421.0	
4	Census Tract 1	5	San Diego	209.0	5815000.0	0.0	0.0	33229.0	0.0	0.0	...	0.0	0.0	0.0	0.0	486.0	

5 rows × 205 columns

Population Distribution

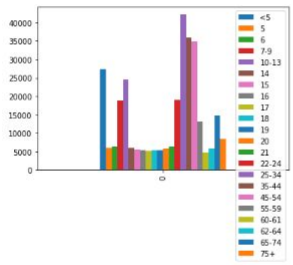
To find out the number of San Diego residents, we figured that filtering out the population of other cities and grouping the table afterward would be the optimal solution. The csv file contained information of other cities besides San Diego, we filtered that out to avoid distraction. The csv file also divided a city into multiple blocks, so we grouped them using the

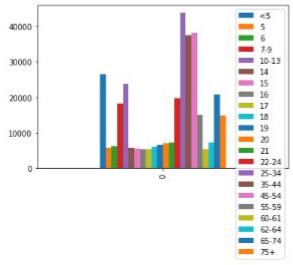
pandas groupby and sum function to find the population of every block in San Diego. The columns in the table divided genders and separated ages into many categories. We constructed a basic bar graph to visualize the population of males and females in San Diego. From our findings, we saw that males and females were about the same for each age interval during the 1970s. The population of San Diego contained people mostly ages from 25 to 55.

What type of people does San Diego contain?

```
In [7]: San_diego = population.loc[population['Place Name'] == 'San Diego']
group_sd = San_diego.groupby(['Place Name']).sum()
list(group_sd.columns)
male_population_sd = group_sd[group_sd.columns[2:24]]
female_population_sd = group_sd[group_sd.columns[24:46]]

male_row = male_population_sd.iloc[0].values.tolist()
female_row = female_population_sd.iloc[0].values.tolist()
#Create a new table

indexs = ['<5', '5', '6', '7-9', '10-13', '14', '15', '16', '17', '18', '19', '20', '21', '22-24', '25-34', '35-44', '45-54', '5
male_population = pd.DataFrame(np.array([male_row]), columns = indexs)
female_population = pd.DataFrame(np.array([female_row]), columns = indexs)

print(male_population.plot.bar())
print(female_population.plot.bar())
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
AxesSubplot(0.125,0.125;0.775x0.755)
```





Demographics

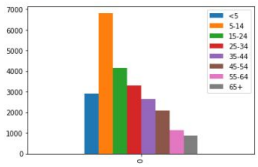
From our given data, we knew that the city of San Diego contained XXX people. Our data also provided the numbers of black people and nonwhite people in San Diego as well. We wanted to see how many of them lived in San Diego in 1970. From the bar graph below, we

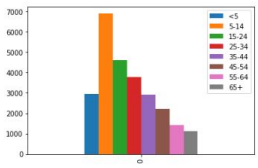
could see that there were many 5-14 kids living in San Diego. We figured that this was caused by the Baby Boom during the 1950s, which was represented by the peaks in our graphs below. Compared with the black population and the nonwhites, we could see that black population outnumbered the nonwhites by more than a half.

Black people in San Diego?

In [8]:
```
bmale_population_sd = group_sd[group_sd.columns[46:54]]
bfemale_population_sd = group_sd[group_sd.columns[54:62]]
bmale_row = bmale_population_sd.iloc[0].values.tolist()
bfemale_row = bfemale_population_sd.iloc[0].values.tolist()
combined_blacks_row = bmale_row + bfemale_row


demo_indexs = ['<5', '5-14', '15-24', '25-34', '35-44', '45-54', '55-64', '65+']
bmale_population = pd.DataFrame(np.array([bmale_row]), columns = demo_indexs)
bfemale_population = pd.DataFrame(np.array([bfemale_row]), columns = demo_indexs)

print(bmale_population.plot.bar())
print(bfemale_population.plot.bar())
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
AxesSubplot(0.125,0.125;0.775x0.755)
```

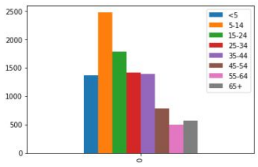


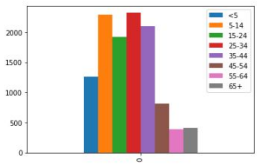


Nonwhites in San Diego?

In [9]:
```
nwmale_population_sd = group_sd[group_sd.columns[62:70]]
nwfemale_population_sd = group_sd[group_sd.columns[70:78]]
nwmale_row = nwmale_population_sd.iloc[0].values.tolist()
nwfemale_row = nwfemale_population_sd.iloc[0].values.tolist()
combined_nw_row = nwmale_row + nwfemale_row

demo2_indexs = ['<5', '5-14', '15-24', '25-34', '35-44', '45-54', '55-64', '65+']
nwmale_population = pd.DataFrame(np.array([nwmale_row]), columns = demo2_indexs)
nwfemale_population = pd.DataFrame(np.array([nwfemale_row]), columns = demo2_indexs)

print(nwmale_population.plot.bar())
print(nwfemale_population.plot.bar())
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
AxesSubplot(0.125,0.125;0.775x0.755)
```



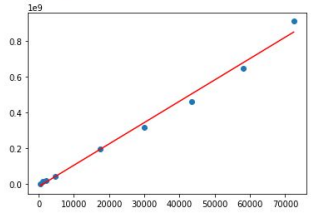


Linear Regression

After looking at the people in San Diego, we wanted to see the housing relationship in it as well. We began to plot graphs that showed the number of housing units in each block of a city and the total aggregates of the units for all of the cities. Then we realized that there was a linear relationship between the two. So, we imported a linear regression model from sklearn to prove our thinking. The model did apply almost perfectly with a positive correlation.

Linear regression between Block group and Total aggregate?

```
In [46]: sd_housing_units_total_aggr = sd_condensed.drop('Black owner aggregate / Total owner',
               axis=1).drop('Black owner occupied real $ aggregate (total) value of housing units', axis=1) # Clean the table
         X = sd_housing_units_total_aggr.iloc[:, 0].values.reshape(-1, 1)  # Make each row into an array
         Y = sd_housing_units_total_aggr.iloc[:, 1].values.reshape(-1, 1)
         linear_regressor = LinearRegression()  # Create linear regression
         linear_regressor.fit(X, Y)  # Perform linear regression
         Y_pred = linear_regressor.predict(X)  # predictions


         plt.scatter(X, Y)
         plt.plot(X, Y_pred, color='red')
         plt.show()
```



Correlation?

```
In [48]: sd_housing_units_total_aggr.corr(method ='pearson')
```

Out[48]:

	Total housing units	Total owner occupied real $ aggregate (total) value of housing units
Total housing units	1.000000	0.995291
Total owner occupied real $ aggregate (total) value of housing units	0.995291	1.000000

Conclusion

We were able to use the given datasets and produced data visualization from them. We also utilized a basic machine learning model linear regression to test our hypothesis. This project was very interesting because I was able to utilize data science skills to find solutions.