Feature Challenge - Soren Laney

Feature Table

tract	total_starbucks_by_tract	Average_Years_of_Edcuation_by_tract
04019002506	2	14.473276655508052
06037701801	2	15.57204377455066
02020000400	3	16.060375443937087
17097863701	1	17.340452865639847
42011011101	1	16.121747734765187

Feature Dictionary

Feature	Description	Data Type
total_starbucks_by_tract	The Total Number of Starbucks per Tract	Integer
Average_Years_of_Edcuation_by_tract	Average Number of Years of Education per Tract	Double

Feature 1

Describing the feature

I realized it would be useful to know what census tracts have very popular brands that indicate there is a lot of traffic or people there. I decided to located how many starbucks there are per census tract becuase it is not only the second most popular fast food establishment behind subway, it is also a location that a typical customer frequents daily. I started by getting data from the hive_metastore.safegraph2.places, specifically the naics codes and the points of intereset. I then gathered data from hive_metastore.safegraph2.censusmapping to get tract code, census block group and placekey information. I then merged the files and began filtering. After I filtered done to only the starbucks locations I joined the tables on placekey. From there I grouped the data by tract code and counting by placekey, or every individual starbucks. This gave me the total number of starbucks per census tract. I will say however, that some of the starbucks did not register with a census tract leaving us with aproximately two thrid's of the total starbucks located in the US.

Pseudocode

```
# Select columns from df_places dataset
df_places_select = df_places.select("placekey", "poi_cbg", "naics_code", "location_name")

# Selects starbucks loctions from dataset
df_starbucks = df_places_select.filter(df_places_select.location_name == "Starbucks")

#join on placekey
df_starbucks_tract = df_starbucks.join(df_censusmapping,df_starbucks.placekey ==  df_censusmapping.plac

#Select Columns from dataset after join
df_starbucks_tract_1 = df_starbucks_tract.select("tract", "location_name")

df_starbucks_tract_total = df_starbucks_tract_1.groupBy("tract").count() \
.withColumnRenamed("count", "total_starbucks_by_tract")
```

Data Wrangling

```
#join on placekey
df_starbucks_tract = df_starbucks.join(df_censusmapping,df_starbucks.placekey ==  df_censusmapping.plac

#Select Columns from dataset after join
df_starbucks_tract_1 = df_starbucks_tract.select("tract", "location_name")

df_starbucks_tract_total = df_starbucks_tract_1.groupBy("tract").count() \
.withColumnRenamed("count", "total_starbucks_by_tract")
```

Visualization

Will update when DataBricks is working

```
import pandas as pd
import bamboolib as bam
import seaborn as sns

feature_1 = soren.select("tract", "total_starbucks_by_tract")

pandasDF = feature_1.toPandas()

sns.violinplot(x=pandasDF["total_starbucks_by_tract"])
```

Table

Table Styling in Markdown

tract	total_starbucks_by_tract
04019002506	2
06037701801	2
02020000400	3
17097863701	1
42011011101	1

Feature 2

Describing the feature

After studying the data, I detirmined that the average education level, or years of schooling, in a census tract could be an excelent indicator for whether or not they have an ev charging and if they will need on in the future. This assumption is made on the principle that typically urban areas have a higher education rate per capita and drive more eletric vehicles compared to that of rural areas where these wehicles are not as practicle and there are less jobs requiring higher education. I gathered the data that I needed for this directly from the census databse and specifically table B15. Within b15 I selected the rows of B15003e2 through B15003e25 which could the population of the that census tract over the age of 25 that has x years of education. I then wrangled this data by creating a seperate dataframe for every level of education provided from zero years of education to a doctorates level of education. After creating these datasets I added a new column assigning an integer variable

to represent the number of years of education. Then I merged the dataframes together to create one dataframe that I could then derive the feature from. I did this by creating a new column that was the population of the census block group multiplied by the years of education to get the total years of education by census block group. I then divided this column by the total population by census block to get the average years of education per census block. After getting the average education of each census block group I then simply grouped by tract to arrive at the target variable.`

Pseudocode

```
# Select all of the education columns from the dataset
df_education_1 = df_education.select("census_block_group", "B15003e2", "B15003e3", "B15003e4", "B15003e
df_education_1.display()

# Dataset for yeo_0
df_education_yoe_0 = df_education_1.select("census_block_group", "B15003e2") \
.withColumn("yoe", lit(0)) \
.withColumnRenamed('B15003e2', 'population') \
#.show()

#...

df_weighted_education_yoe_0 = df_education_yoe_0.withColumn("yoe_weighted", df_education_yoe_0.populati

#...

df_weighted_yoe = [df_weighted_education_yoe_0,
                   df_weighted_education_yoe_1,

#...

total_population = df_merged_2.groupBy("census_block_group") \
.sum("population") \
.withColumnRenamed("sum(population)", "sum_population")#.display(truncate=False)

# Select the columns
total_population_reorderd = total_population.select("sum_population", "census_block_group")

total_yoe = df_merged_2.groupBy("census_block_group") \
.sum("yoe_weighted") \
.withColumnRenamed("sum(yoe_weighted)", "sum_yoe_weighted")#.display(truncate=False)

df_joined = total_population_reorderd.join(total_yoe, ["census_block_group"])

from pyspark.sql.functions import lit

df_avg_yoe_cbg = df_joined.withColumn("avg_years_of_education_cgb", df_joined.sum_yoe_weighted / df_joi

df_final_cbg = df_avg_yoe_cbg.select("census_block_group", "avg_years_of_education_cgb")

from pyspark.sql.functions import substring, length, col, expr
df_tract = df_final_cbg.withColumn("tract",expr("substring(census_block_group, 1, length(census_block_g

df_final_yoe = df_tract.groupBy("tract").mean("avg_years_of_education_cgb") \
.withColumnRenamed("avg(avg_years_of_education_cgb)","Average_Years_of_Edcuation_by_tract")
```

Data Wrangling

```
# Dataset for yeo_0
df_education_yoe_0 = df_education_1.select("census_block_group", "B15003e2") \
.withColumn("yoe", lit(0)) \
.withColumnRenamed('B15003e2', 'population') \
#.show()

#...

df_weighted_education_yoe_0 = df_education_yoe_0.withColumn("yoe_weighted", df_education_yoe_0.populati

#...

df_weighted_yoe = [df_weighted_education_yoe_0,
                   df_weighted_education_yoe_1,

#...

total_population = df_merged_2.groupBy("census_block_group") \
.sum("population") \
.withColumnRenamed("sum(population)", "sum_population")#.display(truncate=False)
```

Visualization

Will Update when databricks is working

```
import pandas as pd
import bamboolib as bam
import seaborn as sns

feature_1 = soren.select("tract", "total_starbucks_by_tract")

pandasDF = feature_1.toPandas()

sns.boxplot(x=pandasDF["total_starbucks_by_tract"])
```

Table

Table Styling in Markdown

tract	Average_Years_of_Edcuation_by_tract
04019002506	14.473276655508052
06037701801	15.57204377455066
02020000400	16.060375443937087

tract	Average_Years_of_Edcuation_by_tract
17097863701	17.340452865639847
42011011101	16.121747734765187