Predicting Internet Access Among Low-Income Seattle Residents

By Brittney Nitta-Lee

Date: April 21, 2023

Business Understanding

This project aims to address the issue of technology access among residents in Seattle and is intended for government officials, policymakers, and organizations working to bridge the digital divide in the community. By ensuring that low-income households have access to technology, this project can have a positive impact by providing opportunities to those who may otherwise be left behind.

Seattle is a technology-driven city with existing research on digital equity and technology access, including data on the digital divide among low-income households. As an employee of a Public Housing Authority that serves low-income residents, I have witnessed firsthand the negative impact of a lack of technology access on individuals and their ability to participate in society. The motivation for this project is to contribute to ongoing efforts to address the digital divide and promote digital equity among low-income residents in Seattle.

Data Understanding

The data used in this project was obtained from the publicly available City of Seattle's Data Portal from 2018. The dataset comprises responses from a random survey of 4,315 Seattle residents. To focus on low-income residents, the analysis will concentrate on survey respondents whose household income is below $90,000, using the 2018 King County Income Limits from the U.S. Department of Housing and Development as a reference.

The dataset has certain drawbacks, such as a poor representation of Seattle's population, as the survey was administered only in English and Spanish, omitting speakers of other languages. This exclusion is unfortunate since data on technology accessibility among low-income residents who speak other languages could be highly informative.

The dataset comprises 479 columns, each containing binary data, and was gathered via a survey that involved responding to 38 questions. To see the full survey, click here. A codebook was provided that you can find here.

Data Collection

A total of 4,315 surveys were collected from May 23rd through June 25th, 2018, representing 4,315 Seattle households and 10,358 Seattle residents.

- Conducted as a multi-mode survey: across mail, online, telephone, and in-person.
- Completed in both English (4,312 surveys) and Spanish (3 surveys).
- The overall average length of the online surveys was 34.0 minutes.
- The overall survey response was 18% (e.g. 18% of those invited to respond returned a survey).

All eligible respondents are:

- Individuals living within Seattle city limits; Able to conduct the survey in either English or Spanish;
- Has the ability to complete the survey via mail with paper and pencil or pen, online via computer, tablet, or smartphone, or in-person via paper and pencil or pen;
- Able to answer on behalf of the whole household on their use of technology and the internet (though if they needed help completing the survey, they could ask another household member or the survey helpline to assist them).

Data Prepration

The dataset is stored in a CSV file and contains a combination of numeric and categorical variables. As the dataset surveys a random sample of 4,315 residents, I filtered the data to focus only on those with an income below $95,000.

To visualize the significant aspects of the data, I used a combination of bar charts to visualize the distribution of categorical and numerical variables.

The data is publicly available, and can be accessed by clicking on the links below:

- Technology Access and Adoption Survey 2018 dataset
- Technology Access and Adoption Survey Codebook
- Technolgy Access and Adoption Survey 2018

Notebooks

I utilized three notebooks to complete this study. Click on the links below to access the notebooks.

- Exploratory Data Analysis
- Models
- Evaluation

Importing Libraries

In [1]:
```
# Import necessary libraries
import numpy as np # For mathematical operations
import pandas as pd # For data manipulation and analysis
%matplotlib inline
import statistics # For statistics operations
import matplotlib.pyplot as plt # For data visualization
import seaborn as sns # For data visualization

# Libraries for data preprocessing
from sklearn.preprocessing import FunctionTransformer, MinMaxScaler, OneHotEncod
from sklearn.impute import SimpleImputer
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer

# Libraries for model building and evaluation
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.metrics import plot_confusion_matrix, classification_report, accura

# Library for handling warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# Create new dataframe import csv file
technology_df = pd.read_csv("Technology_Access_and_Adoption_Survey_2018.csv")
```

In [3]:
```
technology_df
```

Out[3]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
0	2858	1	98125	5	9.0	1	1	2	3	4	...	
1	2859	4	98101	7	9.0	1	2	3	4	0	...	
2	2860	1	98144	3	NaN	1	0	2	3	0	...	
3	2861	4	98199	7	9.0	1	2	3	4	6	...	
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4310	90	1	98122	3	9.0	1	1	2	3	4	...	
4311	23	1	98103	4	9.0	1	2	3	0	0	...	
4312	24	1	98122	3	9.0	1	1	2	3	4	...	
4313	2910	3	98199	7	NaN	2	0	2	3	0	...	
4314	2911	1	98103	4	9.0	1	2	3	4	5	...	

4315 rows × 479 columns

In [4]:
```
technology_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4315 entries, 0 to 4314
Columns: 479 entries, ID to no_response
dtypes: float64(251), int64(219), object(9)
memory usage: 15.8+ MB
```

There's a total of 4,315 survey respondents and 479 rows. Each row contains binary data and is encoded according to the respondents answer.

Data Filtering

Sample Groups

I want to focus on low-income residents in Seattle and this dataset contains information about their household income. Luckily, the dataset is cleaned and simple to use. According to the codebook, there are five `samplegroups`. Here are the sample groups and their values in the dataset.

1 = General Population

2 = Target Population

3 = Seattle Housing Authority

4 = Seattle Public Schools

5 = Homeless Encampments

The `samplegroup` contains data from either the household or individual. Again, the target is low-income indivduals and `samplegroup` 3 and 5, contains my target variable. I will explore further into the dataset and filter respondents who are low-income.

Poverty Guidelines

The dataset contains information about the respondent's poverty guidelines. The value is either 0 = No and 1 = Yes.

Poverty guidelines are a set of income thresholds used to determine eligibility for various federal assistance programs in the United States. The guidelines are issued by the Department of Health and Human Services (HHS) and are updated annually to account for inflation.

The poverty guidelines are based on the federal poverty level (FPL), which is the minimum amount of income a household needs to meet basic needs. The FPL is calculated based on household size and income, and it varies depending on the state and the year.

Federal assistance programs such as Medicaid, the Supplemental Nutrition Assistance Program (SNAP), and the Low Income Home Energy Assistance Program (LIHEAP) use the poverty guidelines to determine eligibility. Individuals and families whose income falls below the poverty guidelines may qualify for these programs and other forms of assistance.

The poverty guidelines in the dataset are as follows:

`POV` = 2018 Poverty Guidelines 100% level

`POV135` = 2018 Poverty Guidelines 135% level

`POV400` = 2018 Poverty Guidelines 140% level

Dataframes

Based on `samplegroup` 3 and 5, it appears that the respondents belong to the low-income group. However, in order to ensure accurate data, I will filter the sample groups based on poverty levels and create separate dataframes for each. It is possible that some individuals may belong to multiple categories, therefore, after creating the dataframes, I will join them using their ID to obtain a clean and complete dataset.

In [5]:
```
# Create new dataframe for samplegroup 3
samplegroup3 = technology_df[technology_df['samplegroup'] == 3]
```

In [6]:
```
samplegroup3
```

Out[6]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
33	3119	3	98119	7	NaN	1	0	2	3	0	...	
59	3184	3	98104	3	9.0	1	1	3	0	0	...	
71	1249	3	98107	6	NaN	2	7	0	0	0	...	
77	3191	3	98108	1	NaN	2	7	0	0	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4237	161	3	98104	3	NaN	2	0	0	3	0	...	
4249	198	3	98121	7	NaN	1	0	0	3	4	...	
4293	144	3	98107	6	NaN	1	1	0	0	0	...	
4297	150	3	98105	4	NaN	1	0	0	3	0	...	
4313	2910	3	98199	7	NaN	2	0	2	3	0	...	

274 rows × 479 columns

In [7]:
```
# create new dataframe for samplegroup 5
samplegroup5 = technology_df[technology_df['samplegroup'] == 5]
```

In [8]:
```
samplegroup5
```

Out[8]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell_I
136	793	5	0	8	NaN	2	0	0	3	0	...	
150	3232	5	0	8	NaN	2	1	2	3	0	...	
290	331	5	0	8	NaN	1	1	2	3	4	...	

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell_I
393	949	5	0	8	NaN	1	0	0	3	0	...	
421	1102	5	0	8	NaN	2	0	0	3	0	...	
425	1122	5	0	8	NaN	2	0	2	3	0	...	
533	1318	5	0	8	NaN	2	7	0	0	0	...	
599	1517	5	0	8	NaN	1	0	0	3	0	...	
725	2414	5	0	8	NaN	1	1	2	3	4	...	
789	2798	5	0	8	NaN	2	0	2	3	0	...	
862	2796	5	0	8	NaN	1	0	2	3	4	...	
943	3449	5	0	8	NaN	2	0	0	3	0	...	
986	3507	5	0	8	NaN	2	0	2	3	4	...	
1006	3509	5	0	8	NaN	1	0	0	3	0	...	
1036	3541	5	0	8	NaN	1	0	2	3	4	...	
1043	3548	5	0	8	NaN	2	0	0	3	0	...	
1120	3622	5	0	8	NaN	2	0	0	3	0	...	
1135	3638	5	0	8	NaN	1	0	0	3	4	...	
1253	1726	5	0	8	NaN	1	0	0	3	0	...	
1263	2610	5	0	8	NaN	1	0	2	3	4	...	
1322	2547	5	0	8	NaN	1	1	2	3	4	...	
1559	2237	5	0	8	NaN	1	0	2	3	0	...	
1581	2202	5	0	8	NaN	1	0	0	0	4	...	
1608	2176	5	0	8	NaN	1	0	2	3	4	...	
1703	2136	5	0	8	NaN	1	0	2	3	0	...	
1819	3741	5	0	8	NaN	2	1	2	0	0	...	
2027	1924	5	0	8	NaN	2	0	2	3	4	...	
2067	1945	5	0	8	NaN	1	1	2	3	0	...	
2119	3849	5	0	8	NaN	1	1	0	0	0	...	
2179	3909	5	0	8	NaN	1	0	0	3	0	...	
2233	3962	5	0	8	NaN	2	1	2	3	4	...	
2447	3985	5	0	8	NaN	1	0	0	3	0	...	
2491	4029	5	0	8	NaN	1	0	2	3	0	...	
2505	4040	5	0	8	NaN	2	7	0	0	0	...	
2586	4120	5	0	8	NaN	1	0	0	3	0	...	
2877	1349	5	0	8	NaN	1	0	2	3	0	...	
2984	4300	5	0	8	NaN	2	1	2	3	4	...	
3120	1160	5	0	8	NaN	1	0	2	0	0	...	

	ID	samplegroup	qzip	CD	l	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell_l
3233	920	5	0	8	NaN	1	0	2	3	0	...	
3283	813	5	0	8	NaN	1	0	0	3	0	...	
3296	734	5	0	8	NaN	1	0	0	3	0	...	
3412	422	5	0	8	NaN	1	1	2	3	4	...	
3487	255	5	0	8	NaN	1	0	2	3	4	...	
3558	147	5	0	8	NaN	1	0	2	3	4	...	
3639	2968	5	0	8	NaN	1	0	2	3	0	...	
3681	3012	5	0	8	NaN	1	0	0	3	0	...	
3776	196	5	0	8	NaN	1	0	2	0	4	...	
3799	184	5	0	8	NaN	2	0	0	3	0	...	
3820	484	5	0	8	NaN	2	1	2	3	4	...	
3963	990	5	0	8	NaN	1	0	0	3	0	...	

50 rows × 479 columns

In [9]:
```
# Create new dataframe for POV who answered yes
POV_df = technology_df[technology_df['POV'] == 1]
```

In [10]:
```
POV_df
```

Out[10]:

	ID	samplegroup	qzip	CD	l	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
15	2873	2	98125	5	NaN	1	1	2	3	4	...	
33	3119	3	98119	7	NaN	1	0	2	3	0	...	
71	1249	3	98107	6	NaN	2	7	0	0	0	...	
100	3183	1	98118	2	NaN	1	1	0	3	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4227	246	2	98133	5	9.0	2	2	3	0	0	...	
4229	336	3	98107	6	9.0	1	1	2	3	0	...	
4231	257	3	98119	7	9.0	1	1	2	3	0	...	
4249	198	3	98121	7	NaN	1	0	0	3	4	...	
4261	107	1	98121	7	NaN	2	7	0	0	0	...	

361 rows × 479 columns

In [11]:
```
POV_135 = technology_df[technology_df['POV135'] == 1]
```

In [12]:
```
POV_135
```

Out[12]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
15	2873	2	98125	5	NaN	1	1	2	3	4	...	
33	3119	3	98119	7	NaN	1	0	2	3	0	...	
71	1249	3	98107	6	NaN	2	7	0	0	0	...	
100	3183	1	98118	2	NaN	1	1	0	3	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4227	246	2	98133	5	9.0	2	2	3	0	0	...	
4229	336	3	98107	6	9.0	1	1	2	3	0	...	
4231	257	3	98119	7	9.0	1	1	2	3	0	...	
4249	198	3	98121	7	NaN	1	0	0	3	4	...	
4261	107	1	98121	7	NaN	2	7	0	0	0	...	

412 rows × 479 columns

In [13]:
```
POV_400 = technology_df[technology_df['POV400'] == 1]
```

In [14]:
```
POV_400
```

Out[14]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
0	2858	1	98125	5	9.0	1	1	2	3	4	...	
1	2859	4	98101	7	9.0	1	2	3	4	0	...	
2	2860	1	98144	3	NaN	1	0	2	3	0	...	
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
7	2865	2	98104	2	NaN	1	1	2	3	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4293	144	3	98107	6	NaN	1	1	0	0	0	...	
4295	66	1	98109	7	NaN	1	1	2	3	4	...	
4297	150	3	98105	4	NaN	1	0	0	3	0	...	
4301	5	2	98108	2	NaN	1	1	2	3	0	...	
4305	11	1	98121	7	NaN	1	1	0	3	0	...	

1332 rows × 479 columns

In [15]:
```
# Join POV_400 and POV_135 on 'ID'
pov_combined = pd.merge(POV_400, POV_135, on='ID')
```

```
# Join POV_combined and POV_df on 'ID'
pov_all = pd.merge(pov_combined, POV_df, on='ID')
```

In [16]:
```
pov_all
```

Out[16]:

	ID	samplegroup_x	qzip_x	CD_x	l_x	q1_x	q2a_1_x	q2a_2_x	q2a_3_x	q2a_4_x	...
0	2862	3	98112	3	NaN	1	0	2	3	0	...
1	2873	2	98125	5	NaN	1	1	2	3	4	...
2	3119	3	98119	7	NaN	1	0	2	3	0	...
3	1249	3	98107	6	NaN	2	7	0	0	0	...
4	3183	1	98118	2	NaN	1	1	0	3	0	...
...	...	...	...	...	...	...	...	...	...	...	...
356	246	2	98133	5	9.0	2	2	3	0	0	...
357	336	3	98107	6	9.0	1	1	2	3	0	...
358	257	3	98119	7	9.0	1	1	2	3	0	...
359	198	3	98121	7	NaN	1	0	0	3	4	...
360	107	1	98121	7	NaN	2	7	0	0	0	...

361 rows × 1435 columns

In [17]:
```
# Concatenate samplegroup3 and samplegroup5 vertically
samplegroup_low = pd.concat([samplegroup3, samplegroup5], axis=0)
```

In [18]:
```
samplegroup_low
```

Out[18]:

	ID	samplegroup	qzip	CD	l	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
4	2862	3	98112	3	NaN	1	0	2	3	0	...	
33	3119	3	98119	7	NaN	1	0	2	3	0	...	
59	3184	3	98104	3	9.0	1	1	3	0	0	...	
71	1249	3	98107	6	NaN	2	7	0	0	0	...	
77	3191	3	98108	1	NaN	2	7	0	0	0	...	
...	...	...	...	...	...	...	...	...	...	...	...	
3681	3012	5	0	8	NaN	1	0	0	3	0	...	
3776	196	5	0	8	NaN	1	0	2	0	4	...	
3799	184	5	0	8	NaN	2	0	0	3	0	...	
3820	484	5	0	8	NaN	2	1	2	3	4	...	
3963	990	5	0	8	NaN	1	0	0	3	0	...	

324 rows × 479 columns

In [19]:
```
if set(pov_all['samplegroup']) == set(samplegroup_low['samplegroup']):
    print("The respondent IDs are the same in both dataframes.")
else:
    print("The respondent IDs are not the same in both dataframes.")
```

```
The respondent IDs are not the same in both dataframes.
```

Since the respondent IDS are not the same I will keep the dataframe separate.

General population

Now that I have my dataframes for individuals who are low-income in Seattle, I'll create new dataframes for the rest of the general population.

In [20]:
```
# Create new dataframe for samplegroup 1
samplegroup1 = technology_df[technology_df['samplegroup'] == 1]
```

In [21]:
```
# Create new dataframe for samplegroup 2
samplegroup2 = technology_df[technology_df['samplegroup'] == 2]
```

In [22]:
```
# Create new dataframe for samplegroup 4
samplegroup4 = technology_df[technology_df['samplegroup'] == 4]
```

In [23]:
```
samplegroup1
```

Out[23]:

	ID	samplegroup	qzip	CD	l	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
0	2858	1	98125	5	9.0	1	1	2	3	4	...	
2	2860	1	98144	3	NaN	1	0	2	3	0	...	
6	2864	1	98103	4	9.0	1	2	3	0	0	...	
8	2866	1	98105	4	NaN	1	1	2	3	4	...	
9	2867	1	98115	4	NaN	1	0	2	3	4	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4309	89	1	98115	4	NaN	1	1	2	3	0	...	
4310	90	1	98122	3	9.0	1	1	2	3	4	...	
4311	23	1	98103	4	9.0	1	2	3	0	0	...	
4312	24	1	98122	3	9.0	1	1	2	3	4	...	
4314	2911	1	98103	4	9.0	1	2	3	4	5	...	

2937 rows × 479 columns

In [24]:
```
samplegroup2
```

Out[24]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
7	2865	2	98104	2	NaN	1	1	2	3	0	...	
10	2868	2	98125	5	NaN	1	1	2	3	0	...	
14	2872	2	98106	1	NaN	2	0	0	3	0	...	
15	2873	2	98125	5	NaN	1	1	2	3	4	...	
28	3115	2	98133	5	NaN	1	0	2	3	4	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4252	408	2	98104	2	NaN	2	0	2	0	0	...	
4256	96	2	98133	5	9.0	1	2	3	5	0	...	
4267	119	2	98105	4	NaN	1	0	2	3	0	...	
4301	5	2	98108	2	NaN	1	1	2	3	0	...	
4304	84	2	98118	2	NaN	1	1	2	3	4	...	

385 rows × 479 columns

In [25]:
```
samplegroup4
```

Out[25]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell_
1	2859	4	98101	7	9.0	1	2	3	4	0	...	
3	2861	4	98199	7	9.0	1	2	3	4	6	...	
5	2863	4	98115	4	9.0	1	1	2	3	4	...	
12	2870	4	98117	6	9.0	1	2	3	4	5	...	
18	3105	4	98118	2	9.0	1	1	3	4	5	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4255	197	4	98115	5	9.0	1	2	3	4	0	...	
4285	282	4	98136	1	9.0	1	1	3	4	6	...	
4286	47	4	98125	5	9.0	1	1	2	3	4	...	
4288	48	4	98122	3	9.0	1	1	3	4	0	...	
4308	18	4	98177	5	9.0	1	2	3	4	0	...	

669 rows × 479 columns

In [26]:
```
# Concatenate samplegroup1, samplegroup2, and samplegroup4 vertically
samplegroup_all = pd.concat([samplegroup1, samplegroup2, samplegroup4], axis=0)
```

In [27]:
```
samplegroup_all
```

Out[27]:

	ID	samplegroup	qzip	CD	I	q1	q2a_1	q2a_2	q2a_3	q2a_4	...	have_data_cell
0	2858	1	98125	5	9.0	1	1	2	3	4	...	
2	2860	1	98144	3	NaN	1	0	2	3	0	...	
6	2864	1	98103	4	9.0	1	2	3	0	0	...	
8	2866	1	98105	4	NaN	1	1	2	3	4	...	
9	2867	1	98115	4	NaN	1	0	2	3	4	...	
...	...	...	...	...	...	...	...	...	...	...	...	
4255	197	4	98115	5	9.0	1	2	3	4	0	...	
4285	282	4	98136	1	9.0	1	1	3	4	6	...	
4286	47	4	98125	5	9.0	1	1	2	3	4	...	
4288	48	4	98122	3	9.0	1	1	3	4	0	...	
4308	18	4	98177	5	9.0	1	2	3	4	0	...	

3991 rows × 479 columns

The dataframes that were created are pov_all, samplegroup_low and samplegroup_all.