give-directly-assessment

June 8, 2022

1 GiveDirectly Take Home Assessment

1.1 Question 1

1. Please evaluate the data in `recipients.csv` and `survey_attempts.csv` to answer the following questions:
    1. How many recipients are in each of the four stages? Please provide the calculation(s) in the spreadsheet or code that you submit.
    2. How many surveys were successfully completed in December, 2020? Please provide the calculation(s) in the spreadsheet or code that you submit.
    3. Did you find any abnormalities in the source data? If so, how did you account for them in your analysis?

First install the required packages.

[ ]:
```
!pip install -r requirements.txt
```

```
Requirement already satisfied: pandas==1.4.2 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 1)) (1.4.2)
Requirement already satisfied: jupyterlab==3.4.2 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 2)) (3.4.2)
Requirement already satisfied: matplotlib==3.5.2 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 3)) (3.5.2)
Requirement already satisfied: ipython==8.4.0 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 4)) (8.4.0)
Requirement already satisfied: flake8==4.0.1 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 5)) (4.0.1)
Requirement already satisfied: PyQt5==5.15.6 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 6)) (5.15.6)
Requirement already satisfied: statsmodels==0.13.2 in
./.venv/lib/python3.8/site-packages (from -r requirements.txt (line 7)) (0.13.2)
Requirement already satisfied: ipykernel==6.13.0 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 8)) (6.13.0)
Requirement already satisfied: missingno==0.5.1 in ./.venv/lib/python3.8/site-
packages (from -r requirements.txt (line 9)) (0.5.1)
Requirement already satisfied: scikit-learn==1.1.1 in
./.venv/lib/python3.8/site-packages (from -r requirements.txt (line 10)) (1.1.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
./.venv/lib/python3.8/site-packages (from pandas==1.4.2->-r requirements.txt
```

```
(line 1)) (2.8.2)
Requirement already satisfied: numpy>=1.18.5; platform_machine != "aarch64" and
platform_machine != "arm64" and python_version < "3.10" in
./.venv/lib/python3.8/site-packages (from pandas==1.4.2->-r requirements.txt
(line 1)) (1.22.4)
Requirement already satisfied: pytz>=2020.1 in ./.venv/lib/python3.8/site-
packages (from pandas==1.4.2->-r requirements.txt (line 1)) (2022.1)
Requirement already satisfied: packaging in ./.venv/lib/python3.8/site-packages
(from jupyterlab==3.4.2->-r requirements.txt (line 2)) (21.3)
Requirement already satisfied: jupyter-server~=1.16 in
./.venv/lib/python3.8/site-packages (from jupyterlab==3.4.2->-r requirements.txt
(line 2)) (1.17.0)
Requirement already satisfied: tornado>=6.1.0 in ./.venv/lib/python3.8/site-
packages (from jupyterlab==3.4.2->-r requirements.txt (line 2)) (6.1)
Requirement already satisfied: jupyter-core in ./.venv/lib/python3.8/site-
packages (from jupyterlab==3.4.2->-r requirements.txt (line 2)) (4.10.0)
Requirement already satisfied: jinja2>=2.1 in ./.venv/lib/python3.8/site-
packages (from jupyterlab==3.4.2->-r requirements.txt (line 2)) (3.1.2)
Requirement already satisfied: nbclassic~=0.2 in ./.venv/lib/python3.8/site-
packages (from jupyterlab==3.4.2->-r requirements.txt (line 2)) (0.3.7)
Requirement already satisfied: jupyterlab-server~=2.10 in
./.venv/lib/python3.8/site-packages (from jupyterlab==3.4.2->-r requirements.txt
(line 2)) (2.14.0)
Requirement already satisfied: pillow>=6.2.0 in ./.venv/lib/python3.8/site-
packages (from matplotlib==3.5.2->-r requirements.txt (line 3)) (9.1.1)
Requirement already satisfied: fonttools>=4.22.0 in ./.venv/lib/python3.8/site-
packages (from matplotlib==3.5.2->-r requirements.txt (line 3)) (4.33.3)
Requirement already satisfied: pyparsing>=2.2.1 in ./.venv/lib/python3.8/site-
packages (from matplotlib==3.5.2->-r requirements.txt (line 3)) (3.0.9)
Requirement already satisfied: cycler>=0.10 in ./.venv/lib/python3.8/site-
packages (from matplotlib==3.5.2->-r requirements.txt (line 3)) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in ./.venv/lib/python3.8/site-
packages (from matplotlib==3.5.2->-r requirements.txt (line 3)) (1.4.2)
Requirement already satisfied: pickleshare in ./.venv/lib/python3.8/site-
packages (from ipython==8.4.0->-r requirements.txt (line 4)) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
./.venv/lib/python3.8/site-packages (from ipython==8.4.0->-r requirements.txt
(line 4)) (3.0.29)
Requirement already satisfied: traitlets>=5 in ./.venv/lib/python3.8/site-
packages (from ipython==8.4.0->-r requirements.txt (line 4)) (5.2.2.post1)
Requirement already satisfied: backcall in ./.venv/lib/python3.8/site-packages
(from ipython==8.4.0->-r requirements.txt (line 4)) (0.2.0)
Requirement already satisfied: pygments>=2.4.0 in ./.venv/lib/python3.8/site-
packages (from ipython==8.4.0->-r requirements.txt (line 4)) (2.12.0)
Requirement already satisfied: jedi>=0.16 in ./.venv/lib/python3.8/site-packages
(from ipython==8.4.0->-r requirements.txt (line 4)) (0.18.1)
Requirement already satisfied: pexpect>4.3; sys_platform != "win32" in
./.venv/lib/python3.8/site-packages (from ipython==8.4.0->-r requirements.txt
```

```
(line 4)) (4.8.0)
Requirement already satisfied: setuptools>=18.5 in ./.venv/lib/python3.8/site-
packages (from ipython==8.4.0->-r requirements.txt (line 4)) (44.0.0)
Requirement already satisfied: decorator in ./.venv/lib/python3.8/site-packages
(from ipython==8.4.0->-r requirements.txt (line 4)) (5.1.1)
Requirement already satisfied: matplotlib-inline in ./.venv/lib/python3.8/site-
packages (from ipython==8.4.0->-r requirements.txt (line 4)) (0.1.3)
Requirement already satisfied: stack-data in ./.venv/lib/python3.8/site-packages
(from ipython==8.4.0->-r requirements.txt (line 4)) (0.2.0)
Requirement already satisfied: pyflakes<2.5.0,>=2.4.0 in
./.venv/lib/python3.8/site-packages (from flake8==4.0.1->-r requirements.txt
(line 5)) (2.4.0)
Requirement already satisfied: mccabe<0.7.0,>=0.6.0 in
./.venv/lib/python3.8/site-packages (from flake8==4.0.1->-r requirements.txt
(line 5)) (0.6.1)
Requirement already satisfied: pycodestyle<2.9.0,>=2.8.0 in
./.venv/lib/python3.8/site-packages (from flake8==4.0.1->-r requirements.txt
(line 5)) (2.8.0)
Requirement already satisfied: PyQt5-Qt5>=5.15.2 in ./.venv/lib/python3.8/site-
packages (from PyQt5==5.15.6->-r requirements.txt (line 6)) (5.15.2)
Requirement already satisfied: PyQt5-sip<13,>=12.8 in
./.venv/lib/python3.8/site-packages (from PyQt5==5.15.6->-r requirements.txt
(line 6)) (12.10.1)
Requirement already satisfied: patsy>=0.5.2 in ./.venv/lib/python3.8/site-
packages (from statsmodels==0.13.2->-r requirements.txt (line 7)) (0.5.2)
Requirement already satisfied: scipy>=1.3 in ./.venv/lib/python3.8/site-packages
(from statsmodels==0.13.2->-r requirements.txt (line 7)) (1.8.1)
Requirement already satisfied: debugpy>=1.0 in ./.venv/lib/python3.8/site-
packages (from ipykernel==6.13.0->-r requirements.txt (line 8)) (1.6.0)
Requirement already satisfied: jupyter-client>=6.1.12 in
./.venv/lib/python3.8/site-packages (from ipykernel==6.13.0->-r requirements.txt
(line 8)) (7.3.1)
Requirement already satisfied: nest-asyncio in ./.venv/lib/python3.8/site-
packages (from ipykernel==6.13.0->-r requirements.txt (line 8)) (1.5.5)
Requirement already satisfied: psutil in ./.venv/lib/python3.8/site-packages
(from ipykernel==6.13.0->-r requirements.txt (line 8)) (5.9.1)
Requirement already satisfied: seaborn in ./.venv/lib/python3.8/site-packages
(from missingno==0.5.1->-r requirements.txt (line 9)) (0.11.2)
Requirement already satisfied: joblib>=1.0.0 in ./.venv/lib/python3.8/site-
packages (from scikit-learn==1.1.1->-r requirements.txt (line 10)) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in
./.venv/lib/python3.8/site-packages (from scikit-learn==1.1.1->-r
requirements.txt (line 10)) (3.1.0)
Requirement already satisfied: six>=1.5 in ./.venv/lib/python3.8/site-packages
(from python-dateutil>=2.8.1->pandas==1.4.2->-r requirements.txt (line 1))
(1.16.0)
Requirement already satisfied: terminado>=0.8.3 in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
```

```
(line 2)) (0.15.0)
Requirement already satisfied: anyio<4,>=3.1.0 in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (3.6.1)
Requirement already satisfied: nbconvert>=6.4.4 in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (6.5.0)
Requirement already satisfied: Send2Trash in ./.venv/lib/python3.8/site-packages
(from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2))
(1.8.0)
Requirement already satisfied: websocket-client in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (1.3.2)
Requirement already satisfied: pyzmq>=17 in ./.venv/lib/python3.8/site-packages
(from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2))
(23.1.0)
Requirement already satisfied: argon2-cffi in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (21.3.0)
Requirement already satisfied: prometheus-client in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (0.14.1)
Requirement already satisfied: nbformat>=5.2.0 in ./.venv/lib/python3.8/site-
packages (from jupyter-server~=1.16->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (5.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in ./.venv/lib/python3.8/site-
packages (from jinja2>=2.1->jupyterlab==3.4.2->-r requirements.txt (line 2))
(2.1.1)
Requirement already satisfied: notebook-shim>=0.1.0 in
./.venv/lib/python3.8/site-packages (from nbclassic~=0.2->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (0.1.0)
Requirement already satisfied: notebook<7 in ./.venv/lib/python3.8/site-packages
(from nbclassic~=0.2->jupyterlab==3.4.2->-r requirements.txt (line 2)) (6.4.11)
Requirement already satisfied: babel in ./.venv/lib/python3.8/site-packages
(from jupyterlab-server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2))
(2.10.1)
Requirement already satisfied: json5 in ./.venv/lib/python3.8/site-packages
(from jupyterlab-server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2))
(0.9.8)
Requirement already satisfied: jsonschema>=3.0.1 in ./.venv/lib/python3.8/site-
packages (from jupyterlab-server~=2.10->jupyterlab==3.4.2->-r requirements.txt
(line 2)) (4.6.0)
Requirement already satisfied: requests in ./.venv/lib/python3.8/site-packages
(from jupyterlab-server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2))
(2.27.1)
Requirement already satisfied: importlib-metadata>=3.6; python_version < "3.10"
in ./.venv/lib/python3.8/site-packages (from jupyterlab-
server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2)) (4.11.4)
```

```
Requirement already satisfied: wcwidth in ./.venv/lib/python3.8/site-packages
(from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython==8.4.0->-r
requirements.txt (line 4)) (0.2.5)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in
./.venv/lib/python3.8/site-packages (from jedi>=0.16->ipython==8.4.0->-r
requirements.txt (line 4)) (0.8.3)
Requirement already satisfied: ptyprocess>=0.5 in ./.venv/lib/python3.8/site-
packages (from pexpect>4.3; sys_platform != "win32"->ipython==8.4.0->-r
requirements.txt (line 4)) (0.7.0)
Requirement already satisfied: pure-eval in ./.venv/lib/python3.8/site-packages
(from stack-data->ipython==8.4.0->-r requirements.txt (line 4)) (0.2.2)
Requirement already satisfied: asttokens in ./.venv/lib/python3.8/site-packages
(from stack-data->ipython==8.4.0->-r requirements.txt (line 4)) (2.0.5)
Requirement already satisfied: executing in ./.venv/lib/python3.8/site-packages
(from stack-data->ipython==8.4.0->-r requirements.txt (line 4)) (0.8.3)
Requirement already satisfied: entrypoints in ./.venv/lib/python3.8/site-
packages (from jupyter-client>=6.1.12->ipykernel==6.13.0->-r requirements.txt
(line 8)) (0.4)
Requirement already satisfied: sniffio>=1.1 in ./.venv/lib/python3.8/site-
packages (from anyio<4,>=3.1.0->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (1.2.0)
Requirement already satisfied: idna>=2.8 in ./.venv/lib/python3.8/site-packages
(from anyio<4,>=3.1.0->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (3.3)
Requirement already satisfied: nbclient>=0.5.0 in ./.venv/lib/python3.8/site-
packages (from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (0.6.4)
Requirement already satisfied: jupyterlab-pygments in
./.venv/lib/python3.8/site-packages (from nbconvert>=6.4.4->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (0.2.2)
Requirement already satisfied: tinycss2 in ./.venv/lib/python3.8/site-packages
(from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (1.1.1)
Requirement already satisfied: pandocfilters>=1.4.1 in
./.venv/lib/python3.8/site-packages (from nbconvert>=6.4.4->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (1.5.0)
Requirement already satisfied: mistune<2,>=0.8.1 in ./.venv/lib/python3.8/site-
packages (from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (0.8.4)
Requirement already satisfied: defusedxml in ./.venv/lib/python3.8/site-packages
(from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (0.7.1)
Requirement already satisfied: beautifulsoup4 in ./.venv/lib/python3.8/site-
packages (from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (4.11.1)
Requirement already satisfied: bleach in ./.venv/lib/python3.8/site-packages
(from nbconvert>=6.4.4->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (5.0.0)
```

```
Requirement already satisfied: argon2-cffi-bindings in
./.venv/lib/python3.8/site-packages (from argon2-cffi->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (21.2.0)
Requirement already satisfied: fastjsonschema in ./.venv/lib/python3.8/site-
packages (from nbformat>=5.2.0->jupyter-server~=1.16->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (2.15.3)
Requirement already satisfied: ipython-genutils in ./.venv/lib/python3.8/site-
packages (from notebook<7->nbclassic~=0.2->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (0.2.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in
./.venv/lib/python3.8/site-packages (from jsonschema>=3.0.1->jupyterlab-
server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2)) (0.18.1)
Requirement already satisfied: importlib-resources>=1.4.0; python_version <
"3.9" in ./.venv/lib/python3.8/site-packages (from
jsonschema>=3.0.1->jupyterlab-server~=2.10->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (5.7.1)
Requirement already satisfied: attrs>=17.4.0 in ./.venv/lib/python3.8/site-
packages (from jsonschema>=3.0.1->jupyterlab-server~=2.10->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (21.4.0)
Requirement already satisfied: certifi>=2017.4.17 in ./.venv/lib/python3.8/site-
packages (from requests->jupyterlab-server~=2.10->jupyterlab==3.4.2->-r
requirements.txt (line 2)) (2022.5.18.1)
Requirement already satisfied: charset-normalizer~=2.0.0; python_version >= "3"
in ./.venv/lib/python3.8/site-packages (from requests->jupyterlab-
server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2)) (2.0.12)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
./.venv/lib/python3.8/site-packages (from requests->jupyterlab-
server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2)) (1.26.9)
Requirement already satisfied: zipp>=0.5 in ./.venv/lib/python3.8/site-packages
(from importlib-metadata>=3.6; python_version < "3.10"->jupyterlab-
server~=2.10->jupyterlab==3.4.2->-r requirements.txt (line 2)) (3.8.0)
Requirement already satisfied: webencodings>=0.4 in ./.venv/lib/python3.8/site-
packages (from tinycss2->nbconvert>=6.4.4->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (0.5.1)
Requirement already satisfied: soupsieve>1.2 in ./.venv/lib/python3.8/site-
packages (from beautifulsoup4->nbconvert>=6.4.4->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (2.3.2.post1)
Requirement already satisfied: cffi>=1.0.1 in ./.venv/lib/python3.8/site-
packages (from argon2-cffi-bindings->argon2-cffi->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (1.15.0)
Requirement already satisfied: pycparser in ./.venv/lib/python3.8/site-packages
(from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->jupyter-
server~=1.16->jupyterlab==3.4.2->-r requirements.txt (line 2)) (2.21)
```

Import the required libraries.

[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

```
import matplotlib
from sklearn.preprocessing import LabelEncoder, StandardScaler
matplotlib.use('QtAgg')

from statsmodels.stats.weightstats import CompareMeans
import statsmodels.formula.api as smf
import missingno as msno
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
from sklearn.linear_model import LogisticRegressionCV
from sklearn.ensemble import ExtraTreesClassifier, ExtraTreesRegressor
from sklearn.model_selection import train_test_split
from collections import defaultdict
```

Let's now import the two datasets and merge them.

[ ]:
```
recipients = pd.read_csv('recipients.csv')
attempts = pd.read_csv('survey_attempts.csv')
merged = pd.merge(recipients, attempts, on='recipient_id', how='left',␣
 ↪indicator=True)
merged._merge.value_counts(dropna=False)
```

[ ]:
```
both          520
left_only       0
right_only      0
Name: _merge, dtype: int64
```

It looks like all the recipients are matched with survey attempts. We can drop the `_merge` variable.

[ ]:
```
merged.drop(columns='_merge', inplace=True)
```

A quick glance at our data.

[ ]:
```
merged.head(10)
```

[ ]:
```
  recipient_id    county  time_county   age  account_number account_status  \
0      r-00085  County B         22.0  22.0     100000022.0            NaN
1      r-00085  County B         22.0  22.0     100000022.0            NaN
2      r-00082  County C         29.0  31.0     100000023.0     Not Active
3      r-00082  County C         29.0  31.0     100000023.0     Not Active
4      r-00048  County C         22.0  24.0     100000035.0         Active
5      r-00048  County C         22.0  24.0     100000035.0         Active
6      r-00048  County C         22.0  24.0     100000035.0         Active
7      r-00096  County B         22.0  25.0     100000166.0         Active
8      r-00096  County B         22.0  25.0     100000166.0         Active
9      r-00064  County D         24.0  26.0     100000076.0         Active

  survey_id      date  success
```

```
0  s-000342  12/23/20     True
1  s-000448  11/25/20    False
2  s-000151  12/20/20     True
3  s-000305  11/22/20    False
4  s-000108  11/28/20    False
5  s-000116  01/17/21     True
6  s-000439  11/18/20    False
7  s-000077  11/06/20    False
8  s-000130  01/03/21     True
9  s-000352  11/04/20    False
```

Check for duplicates. We expect that `recipient_id` and `survey_id` together form a unique id.

[ ]:
```
merged[merged.duplicated(subset=['recipient_id', 'survey_id'], keep=False)]
```

[ ]:
```
    recipient_id    county  time_county   age  account_number account_status  \
15       r-00100  County C         30.0  32.0     100000107.0         Active
17       r-00100  County C         30.0  32.0     100000107.0         Active
19       r-00030  County C         23.0  26.0     100000179.0         Active
22       r-00030  County C         23.0  26.0     100000179.0         Active
28       r-00193  County B         10.0  62.0     100000977.0         Active
..           …       …           …   …              …              …
506      r-00191  County B         10.0  62.0     100000126.0         Active
510      r-00036  County C         32.0  32.0     100000183.0         Active
513      r-00036  County C         32.0  32.0     100000183.0         Active
518      r-00122  County A          8.0  49.0     100000013.0     Not Active
519      r-00122  County A          8.0  49.0     100000013.0     Not Active

    survey_id      date  success
15   s-000045  11/10/20    False
17   s-000045  11/10/20    False
19   s-000036  11/10/20    False
22   s-000036  11/10/20    False
28   s-000018  01/22/21    False
..        …       …       …
506  s-000026  12/11/20    False
510  s-000003  11/16/20    False
513  s-000003  11/16/20    False
518  s-000005  01/29/21     True
519  s-000005  01/29/21     True

[100 rows x 9 columns]
```

There are 50 pairs of duplicates that need to be dropped.

[ ]:
```
merged = merged.drop_duplicates(subset=['recipient_id', 'survey_id'],␣
↪keep='first').reset_index(drop=True)
```

Let's now calculate the stage variable, starting with 'Start'.

[ ]:
```
mask = merged.groupby('recipient_id')['success'].any()
one_success = [mask.index[i] for i, m in enumerate(mask) if m]
# Set the value to start for those with no successful surveys. '~' negates the␣
 ↪value
# of the mask. In this case, '~' means find those without any successful survey
merged.loc[~merged.recipient_id.isin(one_success), 'stage'] = 'Start'
```

Next, the 'Ineligible' stage

[ ]:
```
# Remove the text 'County' from the column
merged.county = merged.county.str.replace('County ', '', regex=False)

inABC = merged.county.isin(['A', 'B', 'C'])
recipient_noABC = merged.recipient_id[~inABC]

merged.loc[merged.recipient_id.isin(one_success) &
    merged.recipient_id.isin(recipient_noABC), 'stage'] = 'Ineligible'
```

The 'Review' stage

[ ]:
```
recipient_yesABC = merged.recipient_id[inABC]
notActive = merged.account_status == 'Not Active'
recipient_notActive = merged.recipient_id[notActive]

merged.loc[merged.recipient_id.isin(one_success) &
           merged.recipient_id.isin(recipient_yesABC) &
           merged.recipient_id.isin(recipient_notActive),
           'stage'] = 'Review'
```

And finally the 'Pay' stage

[ ]:
```
active = merged.account_status == 'Active'
recipient_active = merged.recipient_id[active]
merged.loc[merged.recipient_id.isin(one_success) &
           merged.recipient_id.isin(recipient_yesABC) &
           merged.recipient_id.isin(recipient_active), 'stage'] = 'Pay'
```

1.1.1 1.1 How many recipients in each stage?

[ ]:
```
merged.stage.value_counts(dropna=False)
```

[ ]:
```
Pay           209
Start         150
Ineligible     70
Review         35
NaN             6
```

```
Name: stage, dtype: int64
```

There are 209 in the 'Pay' stage, 150 in the 'Start', 70 'Ineligible', 35 in 'Review', and 6 not in any stage. This is because there are 6 recipients with a missing `account_status`.

[ ]:
```
merged.loc[merged.stage.isna()]
```

[ ]:
```
   recipient_id county  time_county   age  account_number account_status  \
0       r-00085      B         22.0  22.0     100000022.0             NaN
1       r-00085      B         22.0  22.0     100000022.0             NaN
14      r-00145      A         22.0  24.0     100000089.0             NaN
50      r-00045      C         26.0  30.0     100000199.0             NaN
51      r-00045      C         26.0  30.0     100000199.0             NaN
52      r-00045      C         26.0  30.0     100000199.0             NaN

   survey_id      date  success stage
0   s-000342  12/23/20     True   NaN
1   s-000448  11/25/20    False   NaN
14  s-000137  12/21/20     True   NaN
50  s-000161  11/25/20    False   NaN
51  s-000236  11/15/20    False   NaN
52  s-000337  01/14/21     True   NaN
```

We will impute a value for recipients with a missing `account_status`, but it could be worthwhile to follow up with someone for the appropriate status.

1.1.2 1.2 How many successful surveys in December?

[ ]:
```
merged['month'] = merged['date'].astype(str).str[:2]
merged.groupby('month')['success'].sum()
```

[ ]:
```
month
01    77
11     0
12    93
Name: success, dtype: int64
```

There were 93 successful surveys in December.

1.1.3 1.3 Abnormalities in the source data

There were duplicates in the data that had to be dropped. Some values also did not make much sense and were assgined missing value. The missing data was later imputed.

[ ]:
```
merged.describe()
```

[ ]:
```
       time_county          age  account_number
count   455.000000   460.000000    4.630000e+02
```

```
mean      19.138462   124.126087    1.000001e+08
std       10.501550   925.986058    1.074233e+02
min      -45.000000    20.000000    1.000000e+08
25%       10.000000    25.000000    1.000001e+08
50%       21.000000    32.000000    1.000001e+08
75%       28.000000    51.000000    1.000002e+08
max       35.000000  9999.000000    1.000010e+08
```

`time_county` has a minimum of -45 and `age` has a max of 9999. To deal with these problematic cases, I replace them with missing values to be imputed later. It could also be good to follow up with other teams on the correct value.

[ ]:
```
merged.age[merged.age == 9999] = pd.np.nan
merged.time_county[merged.time_county < 0] = pd.np.nan
```

```
/tmp/ipykernel_14671/1017566221.py:1: FutureWarning: The pandas.np module is
deprecated and will be removed from pandas in a future version. Import numpy
directly instead.
  merged.age[merged.age == 9999] = pd.np.nan
/tmp/ipykernel_14671/1017566221.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged.age[merged.age == 9999] = pd.np.nan
/tmp/ipykernel_14671/1017566221.py:2: FutureWarning: The pandas.np module is
deprecated and will be removed from pandas in a future version. Import numpy
directly instead.
  merged.time_county[merged.time_county < 0] = pd.np.nan
/tmp/ipykernel_14671/1017566221.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  merged.time_county[merged.time_county < 0] = pd.np.nan
```

There are missing values in the data as well. Six columns have missing data.

[ ]:
```
merged.isna().sum()
```

[ ]:
```
recipient_id       0
county             8
time_county       20
age               14
account_number     7
account_status    13
survey_id          0
date               0
```

```
success             0
stage               6
month               0
dtype: int64
```

Missing data is usually deleted or imputed. Deleting missing data is easiest, but it can lead to biases if the data is not missing at random. If `age` is more likely to be missing in certain counties, for example, deleting missing data isn't the best approach. If the data is missing completely at random or missing at random, it can be deleted.

If there are many missing values in a row, the row can be deleted. Likewise, if there are many missing values in a column, the column can be deleted.

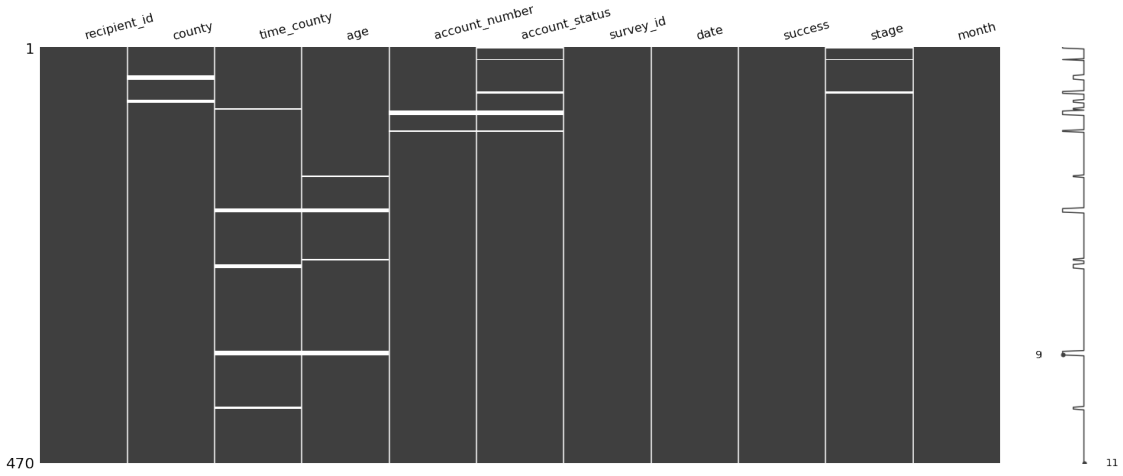
Imputation can be as simple as replacing the missing value in a column with an arbitrary value such as '0' or the column mean or mode. Other techniques take into account the values in other columns. Suppose a recipient has a missing `age`, but lives in a certain county and has also responded to the survey. Then the `age` might be imputed to be slightly lower because of a correlation between `success` and `age` or `county` and `age`. See more here.

We can investigate the missing data with the `missingno` package.

[ ]:
```
%matplotlib inline

msno.matrix(merged)
plt.xticks(rotation=15)
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_14671/2359591310.py:5: UserWarning: This figure includes Axes
that are not compatible with tight_layout, so results might be incorrect.
  plt.tight_layout()
```

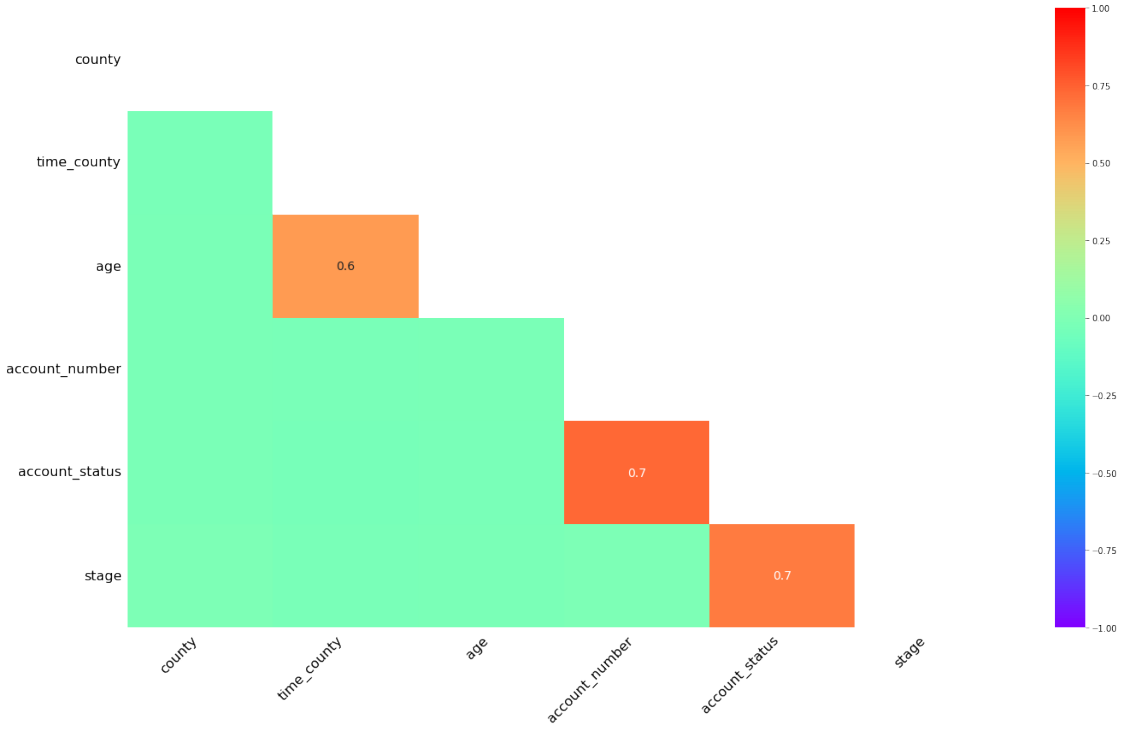


The white lines represent missing values. The `account_number` and `account_status` appear to

have coinciding missing values, and the same for `age` and `time_county`. We can visualize these relationships with a heatmap.

[ ]:
```
%matplotlib inline

msno.heatmap(merged, cmap='rainbow')
plt.tight_layout()
plt.show()
```



There is a strong correlation between `account_number` and `account_status` being missing, which would make sense considering that someone without an `account_number` would not have an `account_status` and vice versa. There is also a strong association between `age` and `time_county` missingness.

To investigate this, let's look at recipients with a missing `time_county`.

[ ]:
```
time_county_missing = merged.loc[merged.time_county.isna()]
print(time_county_missing.head())
```

```
    recipient_id county  time_county   age  account_number account_status  \
69       r-00070      D          NaN  23.0     100000080.0         Active
70       r-00070      D          NaN  23.0     100000080.0         Active
182      r-00198      B          NaN   NaN     100000006.0         Active
183      r-00198      B          NaN   NaN     100000006.0         Active
184      r-00198      B          NaN   NaN     100000006.0         Active
```

```
    survey_id      date  success       stage month
69   s-000252  12/08/20     True  Ineligible    12
70   s-000280  11/10/20    False  Ineligible    11
182  s-000284  01/07/21    False       Start    01
183  s-000285  12/08/20    False       Start    12
184  s-000311  12/18/20    False       Start    12
```

And a missing `age`

[ ]:
```
age_missing = merged.loc[merged.age.isna()]
print(age_missing.head())
```

```
    recipient_id county  time_county  age  account_number account_status  \
145      r-00060      B         28.0  NaN     100000018.0     Not Active
146      r-00060      B         28.0  NaN     100000018.0     Not Active
182      r-00198      B          NaN  NaN     100000006.0         Active
183      r-00198      B          NaN  NaN     100000006.0         Active
184      r-00198      B          NaN  NaN     100000006.0         Active

    survey_id      date  success   stage month
145  s-000273  11/30/20    False  Review    11
146  s-000455  01/29/21     True  Review    01
182  s-000284  01/07/21    False   Start    01
183  s-000285  12/08/20    False   Start    12
184  s-000311  12/18/20    False   Start    12
```

[ ]:
```
print(time_county_missing.recipient_id.nunique())
```

```
5
```

[ ]:
```
print(age_missing.recipient_id.nunique())
```

```
4
```

So `age` is usually missing when `time_county` is missing. There are five recipients that did not fill in their `age` and four that did not fill in `time_county` for multiple survey attempts. They may have been uncomfortable sharing that information.

Dropping these rows could lead to bias because they are not missing at random. Let's impute the `age` and `time_county` for them. We'll also impute the `county` and `account_status`, which are categorical variables. The imputation strategy will regress the column with missing data, say `age`, on other columns in the data. It will then use the regression coefficients to predict the missing values for `age`. More details can be found here, here, and here

Starting with the categorical columns, we'll prepare the dataset for imputation by converting `account_status` and `county` to categorical codes.

[ ]:
```
cat_cols_na = ['account_status', 'county']
merged[cat_cols_na] = merged[cat_cols_na].astype('category')
```

```
d_na = {col: {n: cat for n, cat in enumerate(merged[col].cat.categories)}
            for col in cat_cols_na}
merged[cat_cols_na] = pd.DataFrame(
    {col: merged[col].cat.codes for col in cat_cols_na},
    index=merged.index
)
print(merged[cat_cols_na].head())
```

```
   account_status  county
0              -1       1
1              -1       1
2               1       2
3               1       2
4               0       2
```

Now we can impute the missing values.

[ ]:
```
imp_cat = IterativeImputer(estimator=ExtraTreesClassifier(),
                           initial_strategy='most_frequent',
                           max_iter=10, random_state=0, missing_values=-1)


merged[cat_cols_na] = imp_cat.fit_transform(merged[cat_cols_na])
```

Finally, we convert the numerical codes back to their original labels

[ ]:
```
for col in cat_cols_na:
        merged[col].replace(d_na[col], inplace=True)
print(merged[cat_cols_na].head())
```

```
  account_status county
0         Active      B
1         Active      B
2     Not Active      C
3     Not Active      C
4         Active      C
```

`age` and `time_county` are numeric variables, so they do not need a conversion.

[ ]:
```
num_cols_na = ['age', 'time_county']
imp_num = IterativeImputer(estimator=ExtraTreesRegressor(),
                           initial_strategy='median',
                           max_iter=10, random_state=0)
merged[num_cols_na] = imp_num.fit_transform(merged[num_cols_na])
```

Now that `account_status` has been imputed, the `stage` variable can be recalculated because the six missing cases were missing because `account_status` was missing at the time of calculation.

```
[ ]: notActive = merged.account_status == 'Not Active'
     recipient_notActive = merged.recipient_id[notActive]

     merged.loc[merged.recipient_id.isin(one_success) &
                merged.recipient_id.isin(recipient_yesABC) &
                merged.recipient_id.isin(recipient_notActive),
                'stage'] = 'Review'
```

```
[ ]: active = merged.account_status == 'Active'
     recipient_active = merged.recipient_id[active]
     merged.loc[merged.recipient_id.isin(one_success) &
                merged.recipient_id.isin(recipient_yesABC) &
                merged.recipient_id.isin(recipient_active), 'stage'] = 'Pay'
```

Another look at which columns contain missing data shows that there is only the `account_number` column, which can be ignored for analysis. It also would not be appropriate to impute an identifier.

```
[ ]: merged.isna().sum()
```

```
[ ]: recipient_id      0
     county            0
     time_county       0
     age               0
     account_number    7
     account_status    0
     survey_id         0
     date              0
     success           0
     stage             0
     month             0
     dtype: int64
```

1.2 Question 2

2. The program manager has asked for data to help determine whether the field team should focus more effort on calling those in stage Start or following up to resolve issues with those in stage Review. Please write a response to the program manager, including data that may help inform the decision, and some additional factors that you would take into consideration to make the decision. Assume that the program manager's expertise does not include interpreting data and complex analytics. Please limit your written response to 300 words or less.

It would not make sense to calculate the chance of someone having a successful survey based on the `stage` variable because the 'Start' group would have no successful surveys, and the other three groups would have all recipients with at least one successful survey. On the other hand, it would be good to know the chances of someone moving from the 'Review' stage to a 'Pay' stage at a later date, for example.

[ ]:
```
# Check how many recipients have more than one stage
(merged.groupby('recipient_id')['stage'].nunique() > 1).sum()
```

[ ]:
```
0
```

There do not appear to be recipients that have changed their `stage`. Let's check for recipients that have changed from 'Not Active' to 'Active' `account_status`.

[ ]:
```
(merged.groupby('recipient_id')['account_status'].nunique() > 1).sum()
```

[ ]:
```
0
```

No recipients have changed from 'Not Active' to 'Active' `account_status` either.

One thing to consider is that the 'Start' group has 150 recipients compared to only 35 in the 'Review' group. The 'Start' group success rate only needs to be 35/150 ~ 23% to match a 100% success rate in the 'Review' group. It would be good to look up historical data on the conversion rate from 'Start' to 'Pay' versus 'Review' to 'Pay'. Data on the cost of converting the 'Start' group versus the 'Review' group would also be helpful for the decision.

[ ]:
```
merged.stage.value_counts(dropna=False)
```

[ ]:
```
Pay           215
Start         150
Ineligible     70
Review         35
Name: stage, dtype: int64
```

1.3 Question 3

3. The country director is considering investing resources into proactively conducting in-person surveys with recipients *in the highest age group* across projects to increase overall survey success rate. They believe that this additional cost might outweigh the current costs of repeated failed phone survey attempts, if we can accurately target those recipients *least likely* to respond to a phone survey.
    1. What analysis would you provide from the provided project data to help make this decision? Please provide the calculation(s) in the spreadsheet or code that you submit.
    2. Are there other factors that might explain the observed survey success rate from this project? Please use your judgment to determine these factors and limit your written response to 400 words or less.

1.3.1 3.1

Let's graph the relationship between `age` and at least one successful survey.

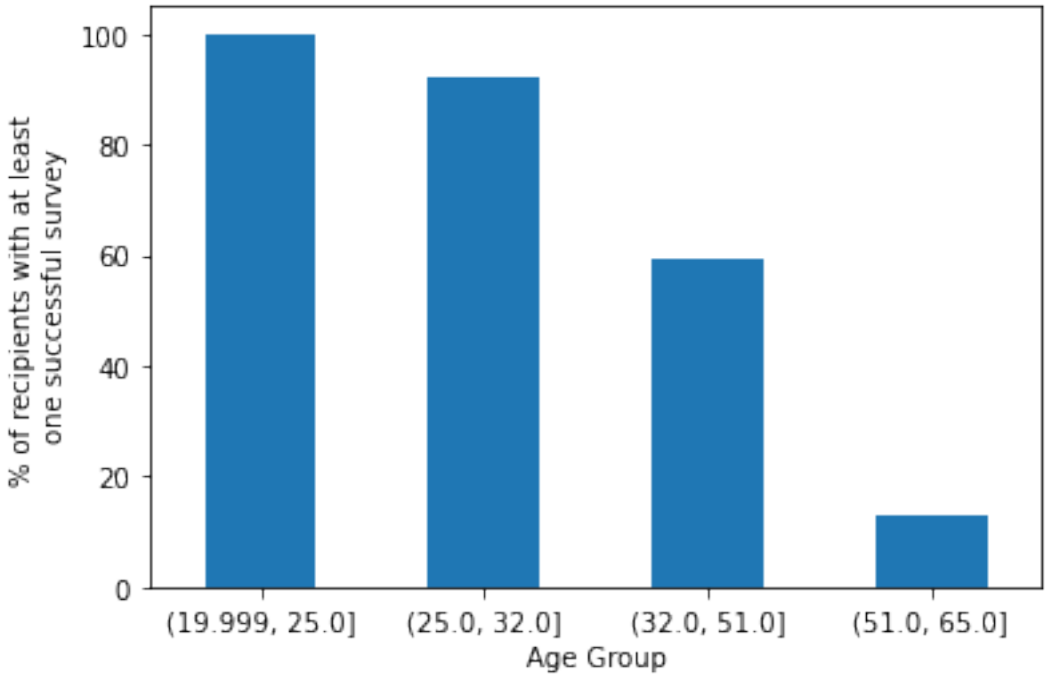
[ ]:
```
%matplotlib inline

merged['one_success'] = 0
merged.loc[merged.recipient_id.isin(one_success), 'one_success'] = 1
```

```
# Create bins for age broken into quartiles
merged['age_bin'] = pd.qcut(merged.age, q=4)
age_success = merged.groupby('age_bin')['one_success'].sum()
age = merged.groupby('age_bin')['one_success'].count()
results = age_success.div(age, level='age_bin') * 100

results.plot(kind='bar')

plt.xticks(rotation=0)
plt.xlabel('Age Group')
plt.ylabel('% of recipients with at least\n one successful survey')
plt.show()
```



People in the highest age group are the least likely to have at least one successful survey.

Running logistic regression gives

[ ]:
```
logit = smf.logit('one_success ~ age', data=merged).fit()
print(logit.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.278114
         Iterations 7
                           Logit Regression Results
==============================================================================
```

```
Dep. Variable:            one_success   No. Observations:                  470
Model:                          Logit   Df Residuals:                      468
Method:                           MLE   Df Model:                            1
Date:                Wed, 08 Jun 2022   Pseudo R-squ.:                  0.5559
Time:                        13:10:08   Log-Likelihood:                -130.71
converged:                       True   LL-Null:                       -294.33
Covariance Type:            nonrobust   LLR p-value:                 3.865e-73
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      7.8313      0.615     12.744      0.000       6.627       9.036
age           -0.1740      0.014    -12.308      0.000      -0.202      -0.146
==============================================================================
```

There is a negative relationship between age and the chance of at least one successful survey. The coefficient is also statistically significant. To interpret it, we calculate the odds ratio.

[ ]:
```
odds_ratios = pd.DataFrame({
        'OR': logit.params,
        'Lower CI': logit.conf_int()[0],
        'Upper CI': logit.conf_int()[1]
    })

odds_ratios = pd.np.exp(odds_ratios)
print(odds_ratios)
print(round((odds_ratios['OR'][1] - 1) * 100, 2))
```

```
                    OR    Lower CI     Upper CI
Intercept  2518.096539  755.093254  8397.386877
age           0.840278    0.817312     0.863888
-15.97
```

```
/tmp/ipykernel_14671/4211037893.py:7: FutureWarning: The pandas.np module is
deprecated and will be removed from pandas in a future version. Import numpy
directly instead.
  odds_ratios = pd.np.exp(odds_ratios)
```

Each additional increase of one year in age is associated with a roughly 16 percent decrease in odds of having at least one successful survey.

So far, it appears that older recipients are less likely to respond to surveys. Focusing on the older recipients would help target those with lower response rates.

1.3.2 3.2

We include other variables in our logistic regression to check for confounders.

[ ]:
```
big_logit = smf.logit('one_success ~ time_county + age + C(month) +␣
 ↪C(account_status)', data=merged).fit()
print(big_logit.summary())
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 0.074409
         Iterations: 35
                           Logit Regression Results
==============================================================================
Dep. Variable:            one_success   No. Observations:                  470
Model:                          Logit   Df Residuals:                      464
Method:                           MLE   Df Model:                            5
Date:                Wed, 08 Jun 2022   Pseudo R-squ.:                  0.8812
Time:                        13:10:08   Log-Likelihood:                -34.972
converged:                      False   LL-Null:                       -294.33
Covariance Type:            nonrobust   LLR p-value:                7.306e-110
==============================================================================
                                    coef    std err          z      P>|z|
[0.025      0.975]
------------------------------------------------------------------------------
Intercept                          10.2072      2.959      3.449      0.001
4.407      16.007
C(month)[T.11]                     24.6815   8.53e+04      0.000      1.000
-1.67e+05    1.67e+05
C(month)[T.12]                     -0.3861      0.670     -0.576      0.564
-1.699       0.927
C(account_status)[T.Not Active]    30.8016   4900.380      0.006      0.995
-9573.767    9635.370
time_county                         0.1367      0.069      1.967      0.049
0.001       0.273
age                                -0.3718      0.093     -4.019      0.000
-0.553      -0.190
==============================================================================

Possibly complete quasi-separation: A fraction 0.54 of observations can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identified.

/home/dylan/give-directly-exercise/.venv/lib/python3.8/site-
packages/statsmodels/base/model.py:604: ConvergenceWarning: Maximum Likelihood
optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

When `time_county`, `month`, and `account_status` are included along with `age`, `age` and `time_county` are still significant predictors. We can also look at other feature selection methods.

Logistic regression can also be used to select features based on their importance in predicting at least one successful survey.

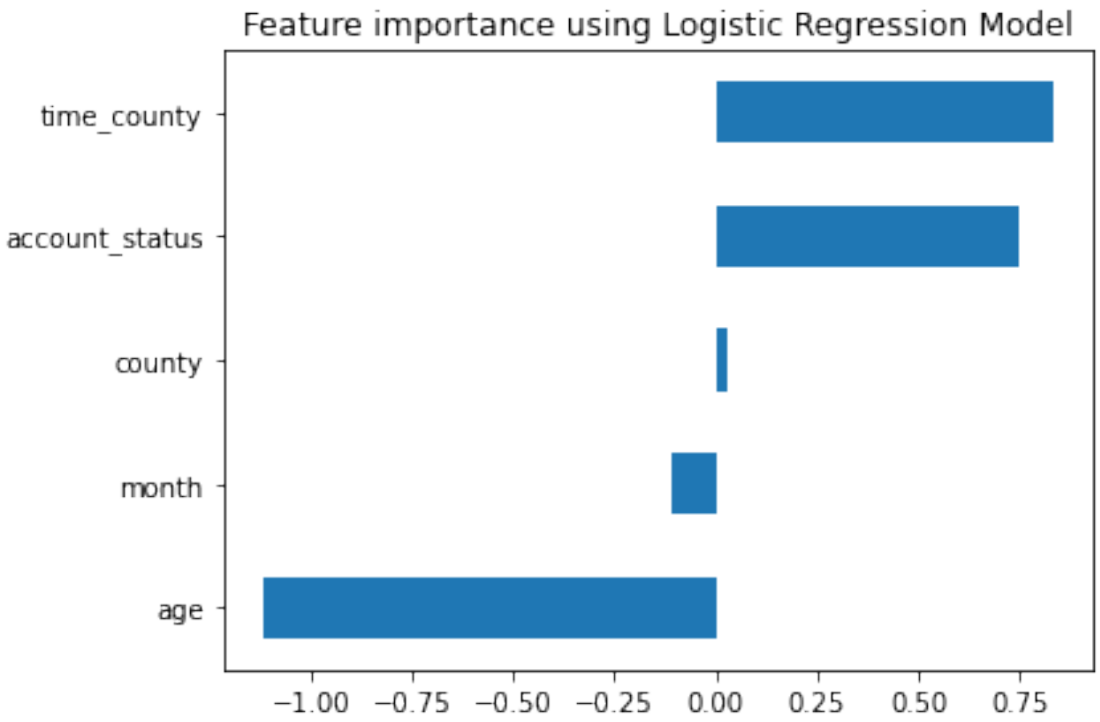
[ ]:
```
reg = LogisticRegressionCV()
cat_cols = ['month', 'account_status', 'county']
num_cols = ['age', 'time_county']
d = defaultdict(LabelEncoder)
le_fit = merged[cat_cols].apply(lambda x: d[x.name].fit_transform(x))
X = pd.concat((merged[num_cols], le_fit), axis=1)
y = merged['one_success']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.3, random_state=42
)

reg.fit(X_train, y_train)
print("Best accuracy score using built-in LogisticRegCV: %f" % reg.
 ↪score(X_test, y_test))
coef = pd.Series(reg.coef_.flatten(), index=X.columns)
imp_coef = coef.sort_values()

imp_coef.plot(kind="barh")
plt.title("Feature importance using Logistic Regression Model")
plt.tight_layout()
plt.show()
```

```
Best accuracy score using built-in LogisticRegCV: 0.992908
```



It appears that the account_status and age have the largest effect on whether a survey is successful. `age` has a negative effect however. Recall that `account_status` and `age` are correlated.

[ ]:
```
merged.groupby('account_status')['age'].mean()
```

[ ]:
```
account_status
Active        36.535321
Not Active    46.908571
Name: age, dtype: float64
```

The 'Not Active' group is slightly older. A test of the equality of group means can confirm this. The null hypothesis is that the means are equal.

[ ]:
```
active_age = merged.age[merged.account_status == 'Active']
notactive_age = merged.age[merged.account_status == 'Not Active']
print(CompareMeans.from_data(active_age, notactive_age).ttest_ind())
```

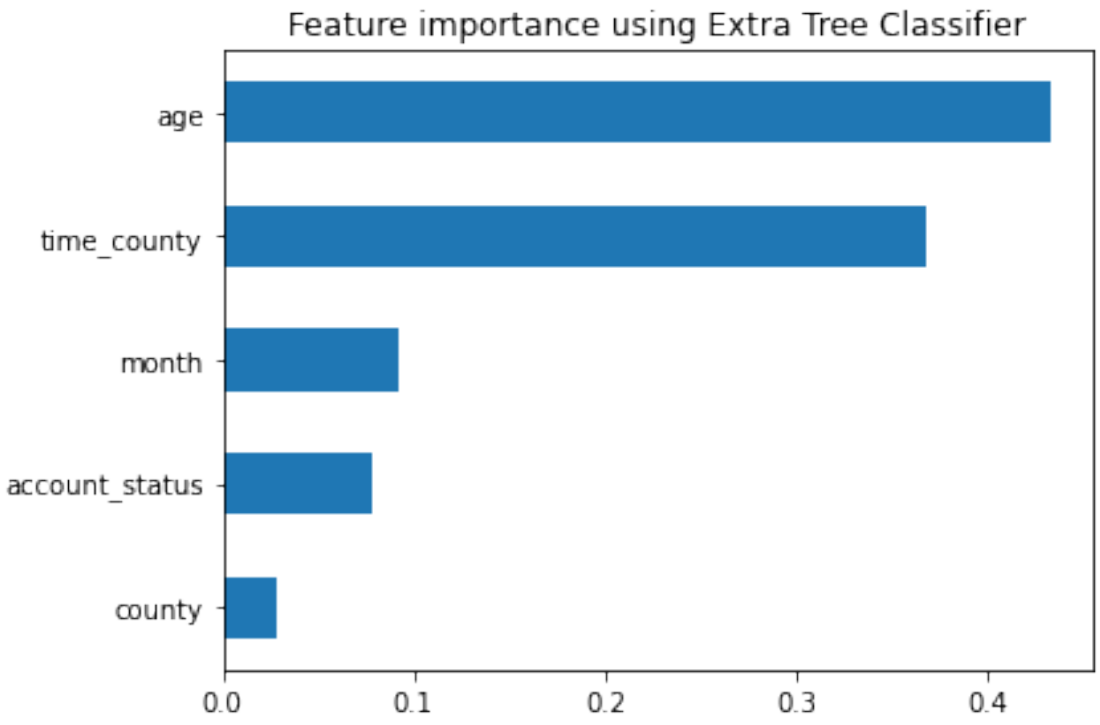
```
(-4.194989773330013, 3.265347532278871e-05, 468.0)
```

The p-value is close to zero, rejecting the null hypothesis that the means of the two groups are equal. So there is a statistically significant difference between the ages of 'Active' and 'Not Active' recipients.

Let's now use a tree-based classifier for comparison.

[ ]:
```
reg_extra_tree = ExtraTreesClassifier(n_estimators=10)
reg_extra_tree.fit(X_train, y_train)
feat_imp = pd.Series(
    reg_extra_tree.feature_importances_,
    index=X.columns
).sort_values()
print(f"Mean accuracy on test data is {reg_extra_tree.score(X_test, y_test)}")
feat_imp.plot(kind="barh")
plt.title("Feature importance using Extra Tree Classifier")
plt.tight_layout()
plt.show()
```

```
Mean accuracy on test data is 1.0
```



This time the model has `age` as the most important effect, with `time_county` as a close second. All of the features are chosen by the model, however. Note that there is randomness in the model, so sometimes the features will be ranked differently across different runs. But the model will still choose features important for prediction.