h1n1-flu-vaccine

March 10, 2024

1 Overview

Vaccination is a key technique used to fight diseases. It is specifically helpful in fighting infectious diseases. Through the use of immunization the spread of diseases can be reduced because it provides herd immunity. The aim of this project is to predict whether individuals got the H1N! flu vaccine based on data obtained from the United States National 2009 H1N1 flu survey. The respondents were asked whether they received the H1N1 flu vaccine in conjuction with questions about themselves that provided data about various features such as whether they had health insurance or if they were health workers.

2 Business Understanding

A vaccine for the H1N1 flu virus became publicly available in October 2009. In late 2009 and early 2010, the United States conducted the National 2009 H1N1 Flu Survey. The aim of the survey was to find out if individuals took the vaccine or not. The classification model in this project will help know whether one took the vaccine or not based on certain traits. In the future this study can then be used by the stakeholders in the public health sector to know which groups of the population to target for vaccination.

3 Data understanding

The data for this project was obtained from .It contains 35 features investigated from 26707 observations which were the responses provided to the survey questions.

4 Metrics of success

The final model will be considered a success if it has f1 score of not less 75%. The goal is to make as accurate as possible predictions, that is why the choice of success metrics is the accuracy score and f1 score.

Importing the necessary libraries

[122]:
```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

```

```
import seaborn as sns

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.linear_model import Lasso, Ridge, LinearRegression

from sklearn.metrics import mean_squared_error

from sklearn.linear_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.model_selection import train_test_split, cross_val_score,␣
↪GridSearchCV

from sklearn.metrics import accuracy_score, recall_score, precision_score,␣
↪f1_score
```

Loading the dataset and have a preview

[123]:
```
training_set_features_df = pd.read_csv("training_set_features.csv")

training_set_features_df.head()
```

[123]:

	respondent_id	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
0	0	1.0	0.0	0.0	
1	1	3.0	2.0	0.0	
2	2	1.0	1.0	0.0	
3	3	1.0	1.0	0.0	
4	4	2.0	1.0	0.0	

	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	\
0	0.0	0.0	0.0	
1	1.0	0.0	1.0	
2	1.0	0.0	0.0	
3	1.0	0.0	1.0	
4	1.0	0.0	1.0	

	behavioral_large_gatherings	behavioral_outside_home	\
0	0.0	1.0	
1	0.0	1.0	

```
2                         0.0                     0.0
3                         1.0                     0.0
4                         1.0                     0.0

   behavioral_touch_face  …             income_poverty  marital_status  \
0                    1.0  …              Below Poverty     Not Married
1                    1.0  …              Below Poverty     Not Married
2                    0.0  …  <= $75,000, Above Poverty     Not Married
3                    0.0  …              Below Poverty     Not Married
4                    1.0  …  <= $75,000, Above Poverty         Married

   rent_or_own   employment_status  hhs_geo_region               census_msa  \
0          Own  Not in Labor Force        oxchjgsf                   Non-MSA
1         Rent            Employed        bhuqouqj  MSA, Not Principle  City
2          Own            Employed        qufhixun  MSA, Not Principle  City
3         Rent  Not in Labor Force        lrircsnp       MSA, Principle City
4          Own            Employed        qufhixun  MSA, Not Principle  City

   household_adults  household_children  employment_industry  \
0               0.0                 0.0                  NaN
1               0.0                 0.0             pxcmvdjn
2               2.0                 0.0             rucpziij
3               0.0                 0.0                  NaN
4               1.0                 0.0             wxleyezf

   employment_occupation
0                    NaN
1                xgwztkwe
2                xtkaffoo
3                    NaN
4                emcorrxb

[5 rows x 36 columns]
```

Exploring the dataset to check the number of rows and columns

[124]:
```
num_rows, num_columns = training_set_features_df.shape
print("Number of rows:", num_rows)
print("Number of columns:", num_columns)
```

```
Number of rows: 26707
Number of columns: 36
```

We check for the data types in our dataset

[125]:
```
training_set_features_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 36 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   respondent_id                26707 non-null  int64
 1   h1n1_concern                 26615 non-null  float64
 2   h1n1_knowledge               26591 non-null  float64
 3   behavioral_antiviral_meds    26636 non-null  float64
 4   behavioral_avoidance         26499 non-null  float64
 5   behavioral_face_mask         26688 non-null  float64
 6   behavioral_wash_hands        26665 non-null  float64
 7   behavioral_large_gatherings  26620 non-null  float64
 8   behavioral_outside_home      26625 non-null  float64
 9   behavioral_touch_face        26579 non-null  float64
 10  doctor_recc_h1n1             24547 non-null  float64
 11  doctor_recc_seasonal         24547 non-null  float64
 12  chronic_med_condition        25736 non-null  float64
 13  child_under_6_months         25887 non-null  float64
 14  health_worker                25903 non-null  float64
 15  health_insurance             14433 non-null  float64
 16  opinion_h1n1_vacc_effective  26316 non-null  float64
 17  opinion_h1n1_risk            26319 non-null  float64
 18  opinion_h1n1_sick_from_vacc  26312 non-null  float64
 19  opinion_seas_vacc_effective  26245 non-null  float64
 20  opinion_seas_risk            26193 non-null  float64
 21  opinion_seas_sick_from_vacc  26170 non-null  float64
 22  age_group                    26707 non-null  object
 23  education                    25300 non-null  object
 24  race                         26707 non-null  object
 25  sex                          26707 non-null  object
 26  income_poverty               22284 non-null  object
 27  marital_status               25299 non-null  object
 28  rent_or_own                  24665 non-null  object
 29  employment_status            25244 non-null  object
 30  hhs_geo_region               26707 non-null  object
 31  census_msa                   26707 non-null  object
 32  household_adults             26458 non-null  float64
 33  household_children           26458 non-null  float64
 34  employment_industry          13377 non-null  object
 35  employment_occupation        13237 non-null  object
dtypes: float64(23), int64(1), object(12)
memory usage: 7.3+ MB
```

Loading the label dataset and exploring the dataset.

[126]:
```
training_set_labels_df = pd.read_csv("training_set_labels.csv")
training_set_labels_df.head()
```

[126]:

	respondent_id	h1n1_vaccine	seasonal_vaccine
0	0	0	0
1	1	0	1
2	2	0	0
3	3	0	1
4	4	0	0

Checking the number of observations and features in our dataset

[127]:
```
num_rows, num_columns = training_set_labels_df.shape
print("Number of rows:", num_rows)
print("Number of columns:", num_columns)
```

```
Number of rows: 26707
Number of columns: 3
```

Determining the data types of our labels dataset

[128]:
```
training_set_labels_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 3 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   respondent_id     26707 non-null  int64
 1   h1n1_vaccine      26707 non-null  int64
 2   seasonal_vaccine  26707 non-null  int64
dtypes: int64(3)
memory usage: 626.1 KB
```

Data Preparation

Our dataset contains responses to whether our respondent recieved the H1N1 vaccine or seasonal flu vaccine. Our target for this project only requires data related to H1N1. Thefore the steps that follow will involve the removal of any data that relates the seasonal flu. It will also involve handling of inconsistencies in our dataset

[129]:
```
#We begin with the features dataframe. Here we drop all columns related to the␣
 ↪seasonal flu vaccine

training_set_features_df= training_set_features_df.
 ↪drop(['doctor_recc_seasonal', 'opinion_seas_vacc_effective',␣
 ↪'opinion_seas_risk', 'opinion_seas_sick_from_vacc'], axis=1)
training_set_features_df.head()
```

[129]:

	respondent_id	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
0	0	1.0	0.0	0.0	
1	1	3.0	2.0	0.0	
2	2	1.0	1.0	0.0	

```
3              3           1.0             1.0                         0.0
4              4           2.0             1.0                         0.0

   behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
0                   0.0                   0.0                    0.0
1                   1.0                   0.0                    1.0
2                   1.0                   0.0                    0.0
3                   1.0                   0.0                    1.0
4                   1.0                   0.0                    1.0

   behavioral_large_gatherings  behavioral_outside_home  \
0                          0.0                      1.0
1                          0.0                      1.0
2                          0.0                      0.0
3                          1.0                      0.0
4                          1.0                      0.0

   behavioral_touch_face  …             income_poverty  marital_status  \
0                    1.0  …               Below Poverty     Not Married
1                    1.0  …               Below Poverty     Not Married
2                    0.0  …  <= $75,000, Above Poverty     Not Married
3                    0.0  …               Below Poverty     Not Married
4                    1.0  …  <= $75,000, Above Poverty         Married

   rent_or_own   employment_status  hhs_geo_region               census_msa  \
0          Own  Not in Labor Force        oxchjgsf                   Non-MSA
1         Rent            Employed        bhuqouqj  MSA, Not Principle  City
2          Own            Employed        qufhixun  MSA, Not Principle  City
3         Rent  Not in Labor Force        lrircsnp       MSA, Principle City
4          Own            Employed        qufhixun  MSA, Not Principle  City

   household_adults  household_children employment_industry  \
0               0.0                0.0                  NaN
1               0.0                0.0            pxcmvdjn
2               2.0                0.0            rucpziij
3               0.0                0.0                  NaN
4               1.0                0.0            wxleyezf

  employment_occupation
0                   NaN
1              xgwztkwe
2              xtkaffoo
3                   NaN
4              emcorrxb

[5 rows x 32 columns]
```

[130]:
```
#Dropping the seasonal vaccine column in our labels dataset
training_set_labels_df = training_set_labels_df.drop(['seasonal_vaccine'],␣
 ↪axis=1)
training_set_labels_df.head()
```

[130]:

	respondent_id	h1n1_vaccine
0	0	0
1	1	0
2	2	0
3	3	0
4	4	0

Creating a dataframe combined_df that contains both the features and the labels dataframes joined into one DataFrame.

[131]:
```
combined_df = training_set_features_df.merge(training_set_labels_df,␣
 ↪on='respondent_id', how='left')
print(combined_df.shape)
combined_df.head()
```

```
(26707, 33)
```

[131]:

	respondent_id	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
0	0	1.0	0.0	0.0	
1	1	3.0	2.0	0.0	
2	2	1.0	1.0	0.0	
3	3	1.0	1.0	0.0	
4	4	2.0	1.0	0.0	

	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	\
0	0.0	0.0	0.0	
1	1.0	0.0	1.0	
2	1.0	0.0	0.0	
3	1.0	0.0	1.0	
4	1.0	0.0	1.0	

	behavioral_large_gatherings	behavioral_outside_home	\
0	0.0	1.0	
1	0.0	1.0	
2	0.0	0.0	
3	1.0	0.0	
4	1.0	0.0	

	behavioral_touch_face	…	marital_status	rent_or_own	\
0	1.0	…	Not Married	Own	
1	1.0	…	Not Married	Rent	
2	0.0	…	Not Married	Own	
3	0.0	…	Not Married	Rent	

```
4                   1.0  …         Married          Own

    employment_status  hhs_geo_region               census_msa  \
0  Not in Labor Force        oxchjgsf                   Non-MSA
1            Employed        bhuqouqj  MSA, Not Principle  City
2            Employed        qufhixun  MSA, Not Principle  City
3  Not in Labor Force        lrircsnp       MSA, Principle City
4            Employed        qufhixun  MSA, Not Principle  City

   household_adults  household_children  employment_industry  \
0               0.0                 0.0                  NaN
1               0.0                 0.0             pxcmvdjn
2               2.0                 0.0             rucpziij
3               0.0                 0.0                  NaN
4               1.0                 0.0             wxleyezf

  employment_occupation h1n1_vaccine
0                   NaN            0
1              xgwztkwe            0
2              xtkaffoo            0
3                   NaN            0
4              emcorrxb            0

[5 rows x 33 columns]
```

[132]:
```
#Loading data from our test dataframe and getting a preview of it
test_set_features_df = pd.read_csv("test_set_features.csv")
test_set_features_df.head()
```

[132]:
```
   respondent_id  h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
0          26707           2.0             2.0                        0.0
1          26708           1.0             1.0                        0.0
2          26709           2.0             2.0                        0.0
3          26710           1.0             1.0                        0.0
4          26711           3.0             1.0                        1.0

   behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
0                   1.0                   0.0                    1.0
1                   0.0                   0.0                    0.0
2                   0.0                   1.0                    1.0
3                   0.0                   0.0                    0.0
4                   1.0                   0.0                    1.0

   behavioral_large_gatherings  behavioral_outside_home  \
0                          1.0                      0.0
1                          0.0                      0.0
2                          1.0                      1.0
```

```
3                         0.0                       0.0
4                         1.0                       1.0

   behavioral_touch_face  …             income_poverty  marital_status  \
0                    1.0  …                  > $75,000     Not Married
1                    0.0  …               Below Poverty     Not Married
2                    1.0  …                  > $75,000         Married
3                    0.0  …  <= $75,000, Above Poverty         Married
4                    1.0  …  <= $75,000, Above Poverty     Not Married

   rent_or_own   employment_status  hhs_geo_region               census_msa  \
0         Rent            Employed       mlyzmhmf  MSA, Not Principle  City
1         Rent            Employed       bhuqouqj                   Non-MSA
2          Own            Employed       lrircsnp                   Non-MSA
3          Own  Not in Labor Force       lrircsnp  MSA, Not Principle  City
4          Own            Employed       lzgpxyit                   Non-MSA

   household_adults  household_children  employment_industry  \
0               1.0                 0.0             atmlpfrs
1               3.0                 0.0             atmlpfrs
2               1.0                 0.0             nduyfdeo
3               1.0                 0.0                  NaN
4               0.0                 1.0             fcxhlnwr

   employment_occupation
0               hfxkjkmi
1               xqwwgdyp
2               pvmttkik
3                    NaN
4               mxkfnird

[5 rows x 36 columns]
```

[133]:
```
#Columns related to the seasonal flu are also dropped in this dataframe
test_set_features_df  =test_set_features_df.drop(['doctor_recc_seasonal',␣
 ↪'opinion_seas_vacc_effective', 'opinion_seas_risk',␣
 ↪'opinion_seas_sick_from_vacc'], axis=1)
test_set_features_df.head()
```

[133]:
```
   respondent_id  h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
0          26707           2.0             2.0                        0.0
1          26708           1.0             1.0                        0.0
2          26709           2.0             2.0                        0.0
3          26710           1.0             1.0                        0.0
4          26711           3.0             1.0                        1.0

   behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
```

```
0                   1.0                   0.0                   1.0
1                   0.0                   0.0                   0.0
2                   0.0                   1.0                   1.0
3                   0.0                   0.0                   0.0
4                   1.0                   0.0                   1.0

   behavioral_large_gatherings  behavioral_outside_home  \
0                          1.0                      0.0
1                          0.0                      0.0
2                          1.0                      1.0
3                          0.0                      0.0
4                          1.0                      1.0

   behavioral_touch_face  …             income_poverty  marital_status  \
0                    1.0  …                  > $75,000     Not Married
1                    0.0  …               Below Poverty     Not Married
2                    1.0  …                  > $75,000         Married
3                    0.0  …  <= $75,000, Above Poverty         Married
4                    1.0  …  <= $75,000, Above Poverty     Not Married

   rent_or_own   employment_status  hhs_geo_region                census_msa  \
0         Rent            Employed        mlyzmhmf  MSA, Not Principle  City
1         Rent            Employed        bhuqouqj                   Non-MSA
2          Own            Employed        lrircsnp                   Non-MSA
3          Own  Not in Labor Force        lrircsnp  MSA, Not Principle  City
4          Own            Employed        lzgpxyit                   Non-MSA

   household_adults  household_children employment_industry  \
0               1.0                 0.0            atmlpfrs
1               3.0                 0.0            atmlpfrs
2               1.0                 0.0            nduyfdeo
3               1.0                 0.0                 NaN
4               0.0                 1.0            fcxhlnwr

  employment_occupation
0             hfxkjkmi
1             xqwwgdyp
2             pvmttkik
3                  NaN
4             mxkfnird

[5 rows x 32 columns]
```

Checking combined_df for missing values

[134]:
```
combined_df.isna().sum().sum()
```

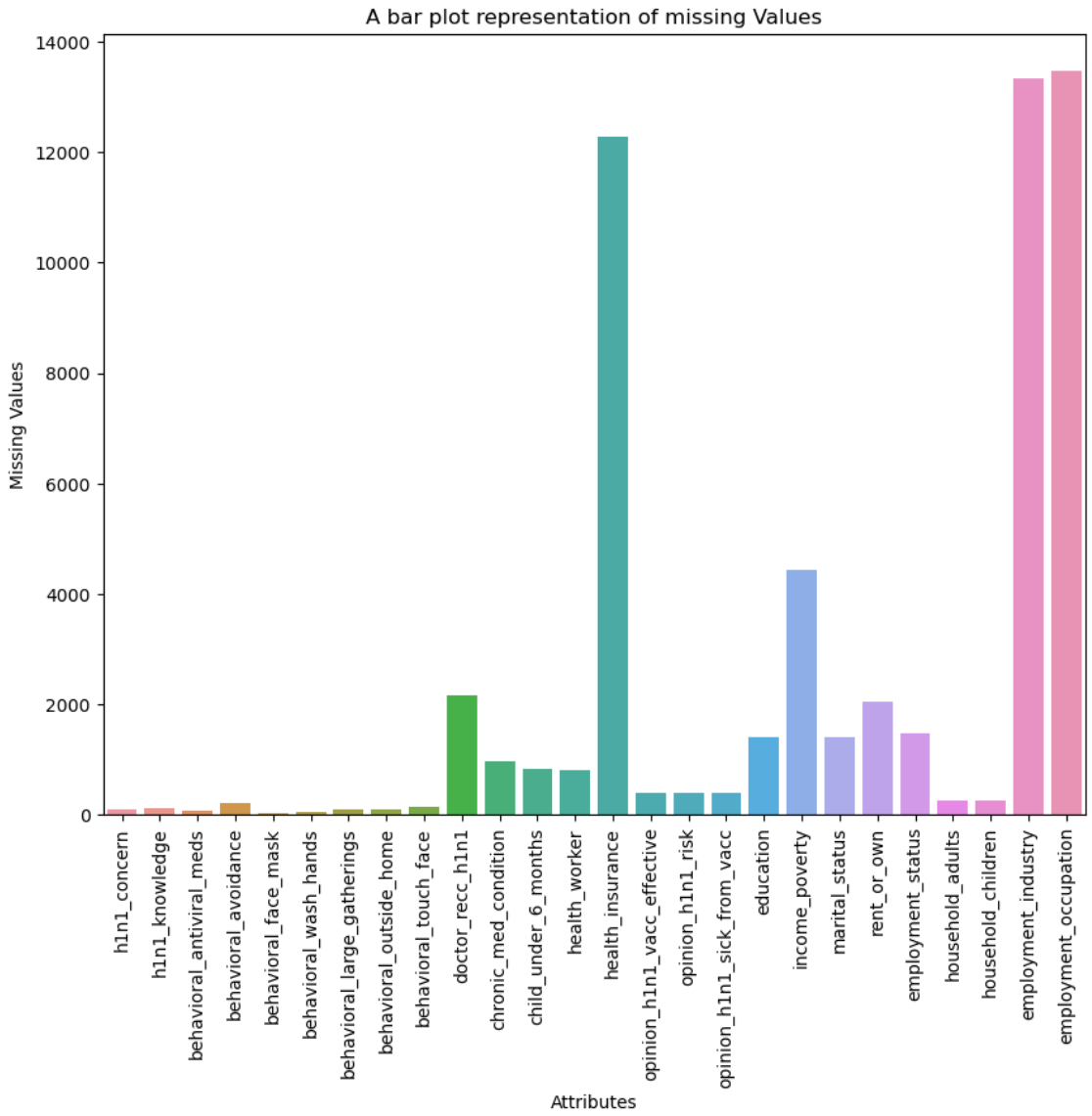
[134]: 57089

[135]:
```
import pandas as pd

def missing_values(df):
    missing = df.isnull().sum()
    percentage = (df.isnull().sum() / len(df))
    missing_df = pd.DataFrame({"No of missing values": missing, "Percentage":␣
 ↪percentage})
    missing_df.drop(missing_df[missing_df["Percentage"] == 0].index,␣
 ↪inplace=True)
    return missing_df  # Return missing_df instead of missing

# Assuming combined_df is your DataFrame containing both features and labels
missing_df = missing_values(combined_df)
print(missing_df)
```

	No of missing values	Percentage
h1n1_concern	92	0.003445
h1n1_knowledge	116	0.004343
behavioral_antiviral_meds	71	0.002658
behavioral_avoidance	208	0.007788
behavioral_face_mask	19	0.000711
behavioral_wash_hands	42	0.001573
behavioral_large_gatherings	87	0.003258
behavioral_outside_home	82	0.003070
behavioral_touch_face	128	0.004793
doctor_recc_h1n1	2160	0.080878
chronic_med_condition	971	0.036358
child_under_6_months	820	0.030704
health_worker	804	0.030104
health_insurance	12274	0.459580
opinion_h1n1_vacc_effective	391	0.014640
opinion_h1n1_risk	388	0.014528
opinion_h1n1_sick_from_vacc	395	0.014790
education	1407	0.052683
income_poverty	4423	0.165612
marital_status	1408	0.052720
rent_or_own	2042	0.076459
employment_status	1463	0.054780
household_adults	249	0.009323
household_children	249	0.009323
employment_industry	13330	0.499120
employment_occupation	13470	0.504362

[136]:
```
#  graphical representation to get a better understanding of the distribution␣
 ↪of missing values
```

```
fig, axes = plt.subplots(figsize = (10, 8))
sns.barplot(x = missing_df.index , y = missing_df["No of missing values"])
plt.xlabel("Attributes",)
plt.xticks(rotation = 'vertical')
plt.ylabel("Missing Values")
plt.title("A bar plot representation of missing Values")
plt.show()
```



The columns **health insurance, employment industry and employment occupation** seem to have a very high percentage of missing values and will therefore need to be dropped

[137]:
```
combined_df = combined_df.drop(['employment_occupation', 'employment_industry',␣
↪'health_insurance'],axis=1)
combined_df.head()
```

[137]:

	respondent_id	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	behavioral_large_gatherings	behavioral_outside_home	behavioral_touch_face	…	sex	income_poverty	marital_status	rent_or_own	employment_status	hhs_geo_region	census_msa	household_adults	household_children	h1n1_vaccine
0	0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	…	Female	Below Poverty	Not Married	Own	Not in Labor Force	oxchjgsf	Non-MSA	0.0	0.0	0
1	1	3.0	2.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0	…	Male	Below Poverty	Not Married	Rent	Employed	bhuqouqj	MSA, Not Principle City	0.0	0.0	0
2	2	1.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	…	Male	<= $75,000, Above Poverty	Not Married	Own	Employed	qufhixun	MSA, Not Principle City	2.0	0.0	0
3	3	1.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	0.0	…	Female	Below Poverty	Not Married	Rent	Not in Labor Force	lrircsnp	MSA, Principle City	0.0	0.0	0
4	4	2.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	1.0	…	Female	<= $75,000, Above Poverty	Married	Own	Employed	qufhixun	MSA, Not Principle City	1.0	0.0	0

[5 rows x 30 columns]

To handle the missing value in our dataset we use a method called backward filling this technique uses the next valid observation along the column and uses to fill our missing value.

[138]:
```
combined_df.fillna(method="bfill", inplace=True)
#The method below checks if there are any null values left
combined_df.isna().sum()
```

[138]:
```
respondent_id                  0
h1n1_concern                   0
h1n1_knowledge                 0
behavioral_antiviral_meds      0
behavioral_avoidance           0
behavioral_face_mask           0
behavioral_wash_hands          0
behavioral_large_gatherings    0
behavioral_outside_home        0
behavioral_touch_face          0
doctor_recc_h1n1               0
chronic_med_condition          0
child_under_6_months           0
health_worker                  0
opinion_h1n1_vacc_effective    0
opinion_h1n1_risk              0
opinion_h1n1_sick_from_vacc    0
age_group                      0
education                      0
race                           0
sex                            0
income_poverty                 0
marital_status                 0
rent_or_own                    0
employment_status              0
hhs_geo_region                 0
census_msa                     0
household_adults               0
household_children             0
h1n1_vaccine                   0
dtype: int64
```

The columns **respondent_id, hhs_geo_region** are irrelevent to our study and therefore we will need to drop them.

[139]:
```
combined_df = combined_df.drop(['respondent_id', 'hhs_geo_region'], axis=1)
combined_df.head()
```

[139]:

	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
0	1.0	0.0	0.0	
1	3.0	2.0	0.0	

```
2           1.0             1.0                         0.0
3           1.0             1.0                         0.0
4           2.0             1.0                         0.0

   behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
0                   0.0                   0.0                    0.0
1                   1.0                   0.0                    1.0
2                   1.0                   0.0                    0.0
3                   1.0                   0.0                    1.0
4                   1.0                   0.0                    1.0

   behavioral_large_gatherings  behavioral_outside_home  \
0                          0.0                      1.0
1                          0.0                      1.0
2                          0.0                      0.0
3                          1.0                      0.0
4                          1.0                      0.0

   behavioral_touch_face  doctor_recc_h1n1  …   race     sex  \
0                    1.0               0.0  …  White  Female
1                    1.0               0.0  …  White    Male
2                    0.0               0.0  …  White    Male
3                    0.0               0.0  …  White  Female
4                    1.0               0.0  …  White  Female

              income_poverty  marital_status  rent_or_own   employment_status  \
0               Below Poverty     Not Married          Own  Not in Labor Force
1               Below Poverty     Not Married         Rent            Employed
2  <= $75,000, Above Poverty     Not Married          Own            Employed
3               Below Poverty     Not Married         Rent  Not in Labor Force
4  <= $75,000, Above Poverty         Married          Own            Employed

                 census_msa household_adults household_children h1n1_vaccine
0                   Non-MSA              0.0                0.0            0
1  MSA, Not Principle  City              0.0                0.0            0
2  MSA, Not Principle  City              2.0                0.0            0
3       MSA, Principle City              0.0                0.0            0
4  MSA, Not Principle  City              1.0                0.0            0

[5 rows x 28 columns]
```

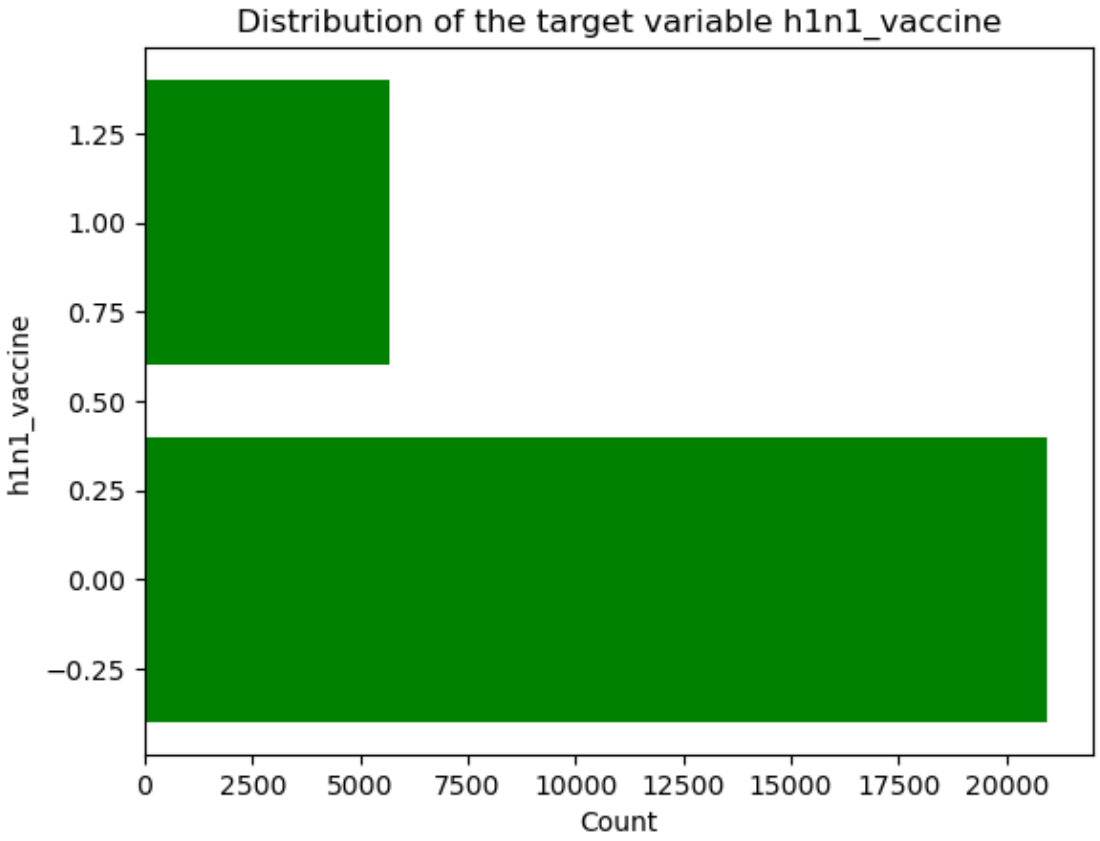
[140]:
```
#We then remove any duplicate values from our dataset

combined_df = combined_df.drop_duplicates()
```

EDA

Univariate analysis

This section involves the exploration of our data to then determine which combination of varibales from our dataset will provide us with information on whether one got vaccinated or not. Our aim is to find which features correlate best with our target variable.

[141]:
```
#our target variable in this case is the column h1n1_vaccine

h1n1_vaccine = combined_df["h1n1_vaccine"]
unique_values, counts = np.unique(h1n1_vaccine, return_counts=True)
fig, ax = plt.subplots()
ax.barh(unique_values, counts, color='green')
ax.set_xlabel('Count')
ax.set_ylabel('h1n1_vaccine')
ax.set_title('Distribution of the target variable h1n1_vaccine')
plt.show()
```



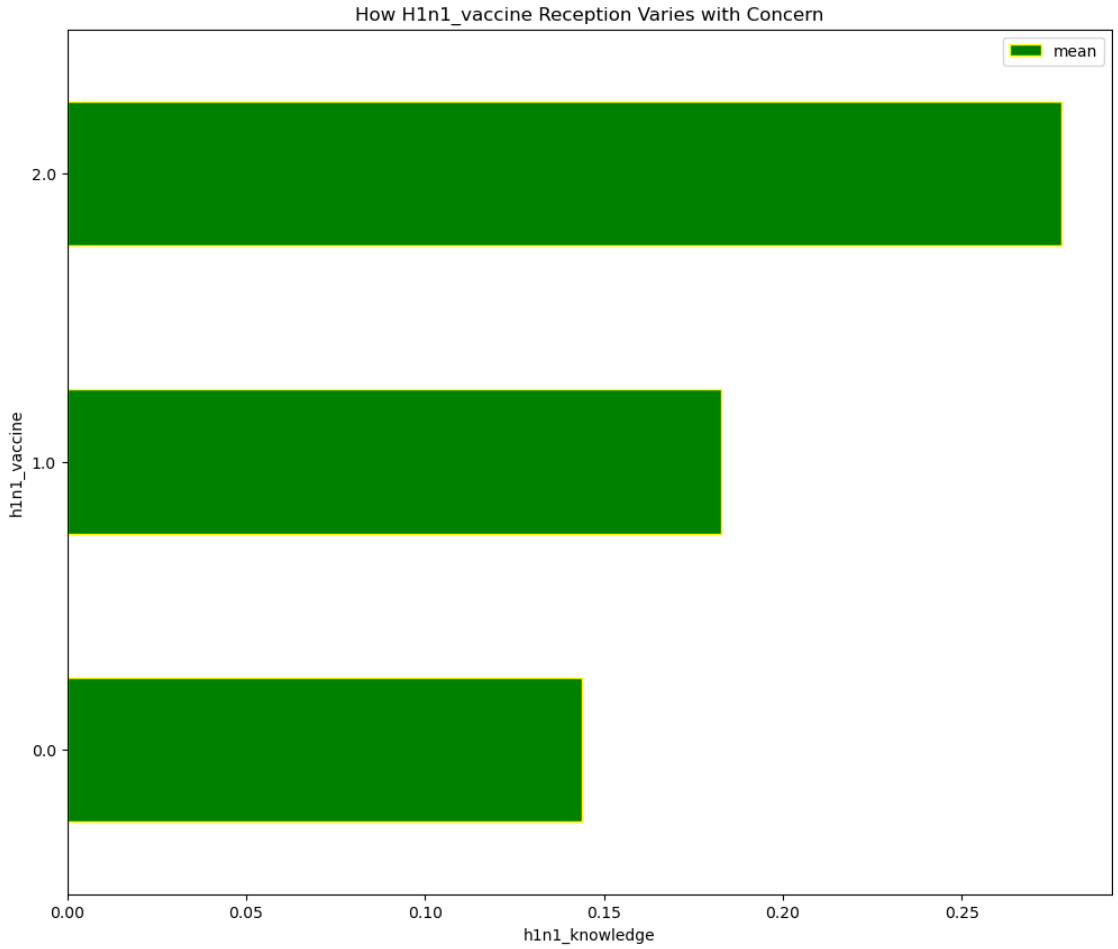
Our analysis from the above data shows that around 20% of the whole population received the H1N1 flu vaccine.

Bivariate analysis

In this stage we look at how various features of our dataset are correlated with the target variable. To begin we will use the h1n1_knowledge, the knowledge one has about h1n1 and the target

variable h1n1_vaccine.

[142]:
```
combined_df.groupby('h1n1_knowledge')['h1n1_vaccine'].agg(['mean']).
 ↪plot(kind='barh',
                                                                    ␣
 ↪figsize=(12,10),
                                                                    ␣
 ↪color='green',
   edgecolor="yellow")
plt.xlabel('h1n1_knowledge')
plt.ylabel('h1n1_vaccine')
plt.title('How H1n1_vaccine Reception Varies with Concern')
plt.show()
```



From the above it is evident that the more the knowledge one has about H1N1 vaccine the more they are likely to receive the vaccine.

Multi-variate analysis

In this section we will check for the correlation of between other variables in our dataset and the the target variables to determine the level of influence it has on one's decision to get the vaccine.

[143]:
```
combined_df
```

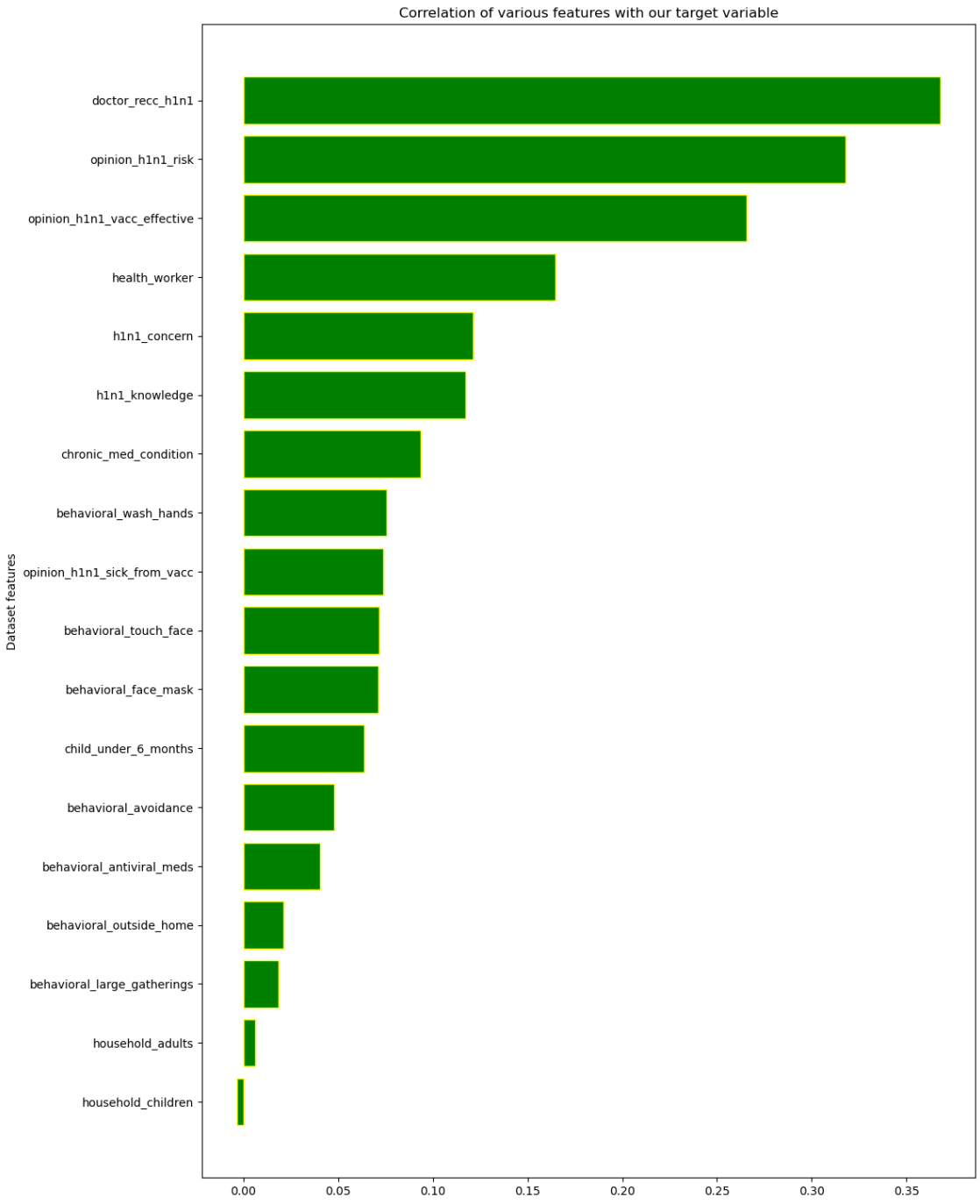
[143]:

```
       h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
0               1.0             0.0                        0.0
1               3.0             2.0                        0.0
2               1.0             1.0                        0.0
3               1.0             1.0                        0.0
4               2.0             1.0                        0.0
…               …               …                          …
26702           2.0             0.0                        0.0
26703           1.0             2.0                        0.0
26704           2.0             2.0                        0.0
26705           1.0             1.0                        0.0
26706           0.0             0.0                        0.0

       behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
0                       0.0                   0.0                    0.0
1                       1.0                   0.0                    1.0
2                       1.0                   0.0                    0.0
3                       1.0                   0.0                    1.0
4                       1.0                   0.0                    1.0
…                       …                     …                      …
26702                   1.0                   0.0                    0.0
26703                   1.0                   0.0                    1.0
26704                   1.0                   1.0                    1.0
26705                   0.0                   0.0                    0.0
26706                   1.0                   0.0                    0.0

       behavioral_large_gatherings  behavioral_outside_home  \
0                              0.0                      1.0
1                              0.0                      1.0
2                              0.0                      0.0
3                              1.0                      0.0
4                              1.0                      0.0
…                              …                        …
26702                          0.0                      1.0
26703                          0.0                      0.0
26704                          1.0                      0.0
26705                          0.0                      0.0
26706                          0.0                      0.0

       behavioral_touch_face  doctor_recc_h1n1  …      race     sex  \
0                        1.0               0.0  …     White  Female
1                        1.0               0.0  …     White    Male
```

```
2                      0.0              0.0  …     White    Male
3                      0.0              0.0  …     White  Female
4                      1.0              0.0  …     White  Female
…                      …                …  …       …      …
26702                  0.0              0.0  …     White  Female
26703                  0.0              1.0  …     White    Male
26704                  1.0              0.0  …     White  Female
26705                  0.0              0.0  …  Hispanic  Female
26706                  0.0              0.0  …     White    Male

                 income_poverty  marital_status  rent_or_own  \
0                   Below Poverty     Not Married          Own
1                   Below Poverty     Not Married         Rent
2      <= $75,000, Above Poverty     Not Married          Own
3                   Below Poverty     Not Married         Rent
4      <= $75,000, Above Poverty         Married          Own
…                          …              …           …
26702  <= $75,000, Above Poverty     Not Married          Own
26703  <= $75,000, Above Poverty     Not Married         Rent
26704  <= $75,000, Above Poverty     Not Married          Own
26705  <= $75,000, Above Poverty         Married         Rent
26706  <= $75,000, Above Poverty         Married          Own

        employment_status                census_msa household_adults  \
0      Not in Labor Force                   Non-MSA              0.0
1                Employed  MSA, Not Principle  City              0.0
2                Employed  MSA, Not Principle  City              2.0
3      Not in Labor Force       MSA, Principle City              0.0
4                Employed  MSA, Not Principle  City              1.0
…                    …                        …               …
26702  Not in Labor Force                   Non-MSA              0.0
26703            Employed       MSA, Principle City              1.0
26704            Employed  MSA, Not Principle  City              0.0
26705            Employed                   Non-MSA              1.0
26706  Not in Labor Force       MSA, Principle City              1.0

      household_children h1n1_vaccine
0                    0.0            0
1                    0.0            0
2                    0.0            0
3                    0.0            0
4                    0.0            0
…                    …              …
26702                0.0            0
26703                0.0            0
26704                0.0            0
26705                0.0            0
```

```
26706                 0.0             0

[26632 rows x 28 columns]
```

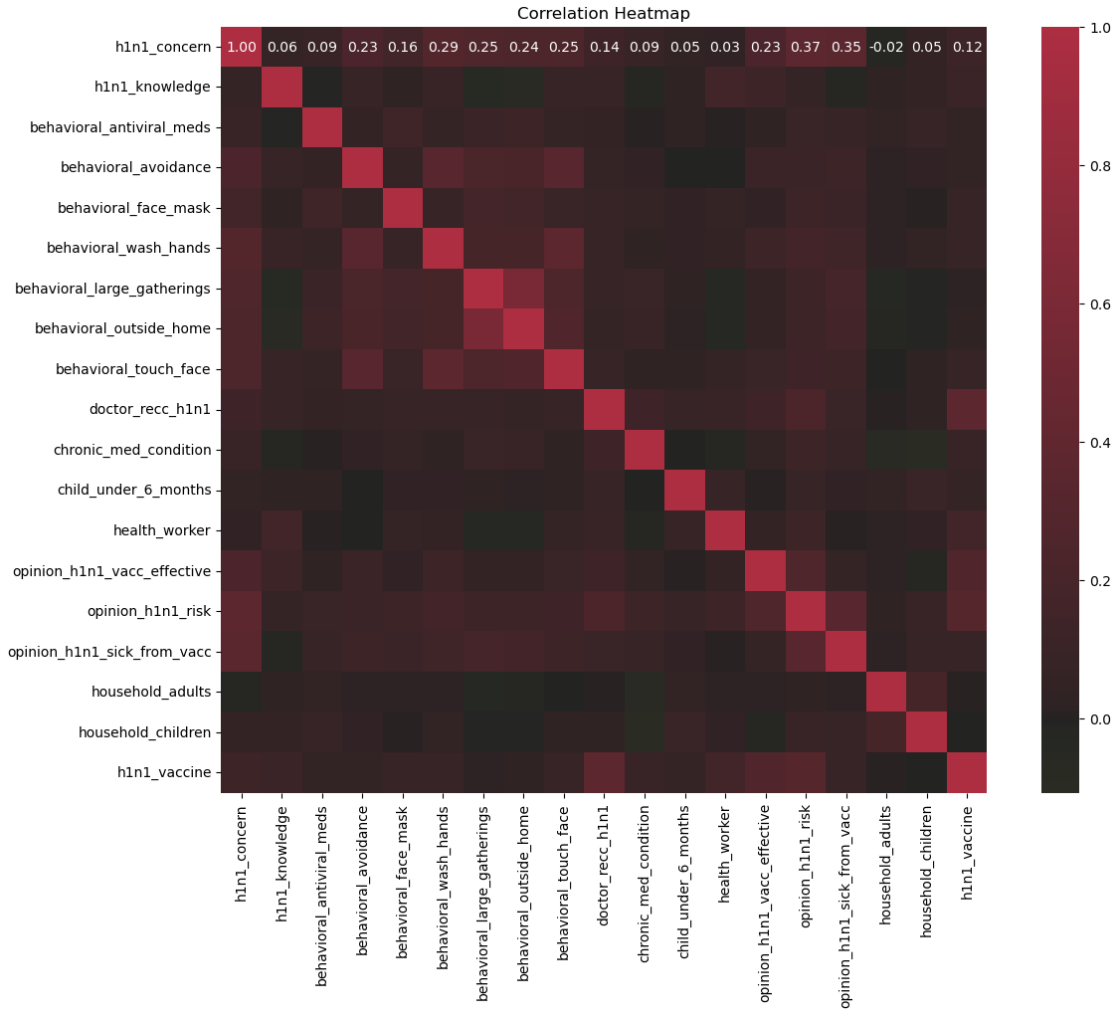
[144]:
```
#Computing the correlation between the features and the target column.
# Drop non-numeric columns from combined_df
numeric_combined_df = combined_df.select_dtypes(include=['number'])

# Calculate correlation with h1n1_vaccine
corr = numeric_combined_df.drop('h1n1_vaccine', axis=1).
 ↪corrwith(combined_df['h1n1_vaccine'])
#Sorting the values in ascending order
corr = corr.sort_values(ascending=True)
#Creating a bar graph to visualize the different values
fig, ax = plt.subplots(figsize=(12, len(corr)*1.0))
ax.barh(corr.index, corr.values, color='green', edgecolor='yellow')
ax.set_ylabel('Dataset features')
ax.set_title('Correlation of various features with our target variable')
plt.show()
```



The above visaulization shows us that the values with a high correlation with the target variable are **doctor_recc_h1n1, opinion_h1n1_risk, and opinion_h1n1_vacc_effective**.

[145]:
```
# Compute the correlation matrix
corr = numeric_combined_df.corr()
```

```
# Create a custom diverging palette
cmap = sns.diverging_palette(100, 7, s=75, l=40,
                             n=5, center="dark", as_cmap=True)

plt.figure(figsize=(14, 10))
sns.heatmap(corr, annot=True,center=0,
            fmt='.2f',square=True, cmap=cmap)

plt.title('Correlation Heatmap')
plt.show()
```



The features behavioral_outside_home and behavioral_large_gatherings will be dropped because of the high level of correlation between them since this is against the classification assumption that the features are independent.

Feature Engineering

We will split our data into training and test dataset

[146]:
```
X = combined_df.drop('h1n1_vaccine',axis = 1)
y = combined_df['h1n1_vaccine']

X_train,X_test,y_train,y_test = train_test_split(X,y,random_state = 42,␣
 ↪test_size = 0.3)
```

[147]:
```
#We identify the categorical columns in our trining dataset
categorical_columns = X_train.select_dtypes(include='object').columns

print("Categorical columns:")
print(categorical_columns)
```

```
Categorical columns:
Index(['age_group', 'education', 'race', 'sex', 'income_poverty',
       'marital_status', 'rent_or_own', 'employment_status', 'census_msa'],
      dtype='object')
```

[148]:
```
#We then convert this categorical non-numemrical values into numerical values

label_encoder = LabelEncoder()

X_train['age_group'] = label_encoder.fit_transform(X_train['age_group'])
X_train['education'] = label_encoder.fit_transform(X_train['education'])
X_train['race'] = label_encoder.fit_transform(X_train['race'])
X_train['sex'] = label_encoder.fit_transform(X_train['sex'])
X_train['marital_status'] = label_encoder.
 ↪fit_transform(X_train['marital_status'])
X_train['rent_or_own'] = label_encoder.fit_transform(X_train['rent_or_own'])
X_train['census_msa'] = label_encoder.fit_transform(X_train['census_msa'])
X_train['employment_status'] = label_encoder.
 ↪fit_transform(X_train['employment_status'])
X_train['income_poverty'] = label_encoder.
 ↪fit_transform(X_train['income_poverty'])

X_train.head()
```

[148]:

	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
9573	1.0	1.0	0.0	
19859	1.0	2.0	0.0	
6996	1.0	2.0	0.0	
15355	2.0	2.0	0.0	
26674	1.0	1.0	0.0	

	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	\
9573	1.0	0.0	1.0	

```
19859                    1.0                    0.0                    1.0
6996                     1.0                    0.0                    1.0
15355                    1.0                    0.0                    1.0
26674                    1.0                    0.0                    1.0

       behavioral_large_gatherings  behavioral_outside_home  \
9573                           0.0                      1.0
19859                          1.0                      1.0
6996                           0.0                      0.0
15355                          0.0                      0.0
26674                          0.0                      0.0

       behavioral_touch_face  doctor_recc_h1n1  …  education  race  sex  \
9573                     0.0               1.0  …          2     3    1
19859                    1.0               0.0  …          2     3    0
6996                     1.0               0.0  …          3     3    0
15355                    1.0               0.0  …          3     3    0
26674                    0.0               0.0  …          0     3    0

       income_poverty  marital_status  rent_or_own  employment_status  \
9573                1               0            0                  1
19859               0               1            1                  2
6996                0               1            0                  1
15355               0               1            0                  0
26674               1               0            0                  0

       census_msa  household_adults  household_children
9573           1               1.0                 0.0
19859          0               0.0                 0.0
6996           1               0.0                 0.0
15355          2               0.0                 0.0
26674          0               2.0                 1.0

[5 rows x 27 columns]
```

[149]:
```
#We repeat the same process for our test data

label_encoder = LabelEncoder()

X_test['age_group'] = label_encoder.fit_transform(X_test['age_group'])
X_test['education'] = label_encoder.fit_transform(X_test['education'])
X_test['race'] = label_encoder.fit_transform(X_test['race'])
X_test['sex'] = label_encoder.fit_transform(X_test['sex'])
X_test['marital_status'] = label_encoder.fit_transform(X_test['marital_status'])
X_test['rent_or_own'] = label_encoder.fit_transform(X_test['rent_or_own'])
X_test['census_msa'] = label_encoder.fit_transform(X_test['census_msa'])
```

```
X_test['employment_status'] = label_encoder.
  ↪fit_transform(X_test['employment_status'])
X_test['income_poverty'] = label_encoder.fit_transform(X_test['income_poverty'])


X_test.head()
```

[149]:

	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	behavioral_large_gatherings	behavioral_outside_home	behavioral_touch_face	doctor_recc_h1n1	…	education	race	sex	income_poverty	marital_status	rent_or_own	employment_status	census_msa	household_adults	household_children
10445	2.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	…	3	3	0	2	1	1	1	2	0.0	0.0
11409	2.0	2.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	…	0	3	0	0	0	0	1	0	1.0	0.0
17504	2.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	…	2	3	1	1	1	0	0	1	0.0	0.0
19391	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	…	3	2	1	0	1	1	1	0	1.0	1.0
7968	2.0	2.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	…	3	0	0	0	1	1	0	0	1.0	0.0

```
7968            1               1.0                 2.0

[5 rows x 27 columns]
```

[150]:
```
#Our next step will be to normalize the train and the test data
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train_transformed = scaler.fit_transform(X_train)
X_test_transformed = scaler.transform(X_test)
```

Modelling

We begin with our baseline model which is a simple logistic regression

[151]:
```
from sklearn.linear_model import LogisticRegression

# Instantiate a Logistic Regression model with specified parameters
logreg = LogisticRegression(fit_intercept=False, C=1e12, solver='sag')

# Fit the Logistic Regression model to the training data
model_log = logreg.fit(X_train_transformed, y_train)

# The trained model is stored in the variable model_log
```

Evaluating the model

[152]:
```
# Predicting the target variable using the trained logistic regression model on␣
 ↪the training data
y_hat_train = logreg.predict(X_train_transformed)

# Calculating the absolute residuals
train_residuals = np.abs(y_train - y_hat_train)

# Printing counts of residuals
print(pd.Series(train_residuals, name="Residuals (counts)").value_counts())
print()

# Printing proportions of residuals
print(pd.Series(train_residuals, name="Residuals (proportions)").
 ↪value_counts(normalize=True))
```

```
Residuals (counts)
0    15192
1     3450
Name: count, dtype: int64

Residuals (proportions)
0    0.814934
```

```
1    0.185066
Name: proportion, dtype: float64
```

[153]:
```python
# Predicting the target variable using the trained logistic regression model on the test data
y_hat_test = logreg.predict(X_test_transformed)

# Calculating the absolute residuals for the test data
test_residuals = np.abs(y_test - y_hat_test)

# Printing counts of residuals for the test data
print(pd.Series(test_residuals, name="Residuals (counts)").value_counts())
print()

# Printing proportions of residuals for the test data
print(pd.Series(test_residuals, name="Residuals (proportions)").value_counts(normalize=True))
```

```
Residuals (counts)
0    6494
1    1496
Name: count, dtype: int64

Residuals (proportions)
0    0.812766
1    0.187234
Name: proportion, dtype: float64
```

[154]:
```python
def display_metrics(true, preds):

    print(f'Accuracy: {accuracy_score(true, preds)}')
    print(f'F1-Score: {f1_score(true, preds)}')
    print(f'Recall-Score: {recall_score(true, preds)}')
    print(f'Precision-Score: {precision_score(true, preds)}')

print('Logistic Regression\n')
display_metrics(y_train, y_hat_train)
print('----\n')
display_metrics(y_test, y_hat_test)
```

```
Logistic Regression

Accuracy: 0.8149340199549404
F1-Score: 0.45290199809705045
Recall-Score: 0.35870384325546345
Precision-Score: 0.6141935483870967
----
```

```
Accuracy: 0.8127659574468085
F1-Score: 0.4367469879518072
Recall-Score: 0.3431952662721893
Precision-Score: 0.6004140786749482
```

Although our accuracy levels are at a realistic level of 81%, showing that an individual got the vaccine or not the f1 score is too low. Using feature selection before building a K-nearest model in the next step will make the data we use more controlled and improve the accuracy of the model.

Feature selection

It is important to select feature based on correlation to build appropiate models. The use of appropiate and fewer features may reduce the occurence of problems such as overfitting.

[155]:
```
#We begin by an analysis of our dataset
X_train_transformed
X_train_transformed.shape
```

[155]:
```
(18642, 27)
```

[156]:
```
#We then convert the data into a pandas Dataframe
columns = ['h1n1_concern', 'h1n1_knowledge', 'behavioral_antiviral_meds',␣
 ↪'behavioral_avoidance', 'behavioral_face_mask', 'behavioral_wash_hands',␣
 ↪'behavioral_large_gatherings', 'behavioral_outside_home',␣
 ↪'behavioral_touch_face', 'doctor_recc_h1n1', 'chronic_med_condition',␣
 ↪'child_under_6_months', 'health_worker', 'opinion_h1n1_vacc_effective',␣
 ↪'opinion_h1n1_risk', 'opinion_h1n1_sick_from_vacc', 'age_group',␣
 ↪'education', 'race', 'sex', 'income_poverty', 'marital_status',␣
 ↪'rent_or_own', 'employment_status', 'census_msa', 'household_adults',␣
 ↪'household_children']

X_train_transformed = pd.DataFrame(X_train_transformed, columns=columns)
X_train_transformed
```

[156]:

	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
0	0.333333	0.5	0.0	
1	0.333333	1.0	0.0	
2	0.333333	1.0	0.0	
3	0.666667	1.0	0.0	
4	0.333333	0.5	0.0	
…	…	…	…	
18637	0.000000	0.5	0.0	
18638	0.666667	0.5	1.0	
18639	0.666667	0.5	0.0	
18640	0.666667	1.0	0.0	
18641	0.000000	0.5	0.0	

	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	\
0	1.0	0.0	1.0	

```
1                        1.0                    0.0                    1.0
2                        1.0                    0.0                    1.0
3                        1.0                    0.0                    1.0
4                        1.0                    0.0                    1.0
…                        …                      …                      …
18637                    0.0                    0.0                    0.0
18638                    1.0                    1.0                    1.0
18639                    1.0                    0.0                    0.0
18640                    1.0                    0.0                    1.0
18641                    0.0                    0.0                    0.0

       behavioral_large_gatherings  behavioral_outside_home  \
0                              0.0                      1.0
1                              1.0                      1.0
2                              0.0                      0.0
3                              0.0                      0.0
4                              0.0                      0.0
…                              …                        …
18637                          1.0                      1.0
18638                          1.0                      1.0
18639                          0.0                      0.0
18640                          0.0                      0.0
18641                          0.0                      0.0

       behavioral_touch_face  doctor_recc_h1n1  …  education  race  sex  \
0                        0.0               1.0  …   0.666667   1.0  1.0
1                        1.0               0.0  …   0.666667   1.0  0.0
2                        1.0               0.0  …   1.000000   1.0  0.0
3                        1.0               0.0  …   1.000000   1.0  0.0
4                        0.0               0.0  …   0.000000   1.0  0.0
…                        …                 …  …   …          …    …
18637                    1.0               0.0  …   0.666667   1.0  1.0
18638                    1.0               0.0  …   0.666667   1.0  1.0
18639                    1.0               0.0  …   0.000000   1.0  0.0
18640                    1.0               0.0  …   0.666667   1.0  0.0
18641                    0.0               0.0  …   1.000000   1.0  0.0

       income_poverty  marital_status  rent_or_own  employment_status  \
0                 0.5             0.0          0.0                0.5
1                 0.0             1.0          1.0                1.0
2                 0.0             1.0          0.0                0.5
3                 0.0             1.0          0.0                0.0
4                 0.5             0.0          0.0                0.0
…                 …               …            …                  …
18637             0.0             0.0          0.0                0.5
18638             0.5             1.0          0.0                0.0
18639             0.0             0.0          0.0                0.0
```

```
18640              0.5              0.0          0.0                 0.5
18641              0.0              1.0          0.0                 0.0

       census_msa  household_adults  household_children
0             0.5          0.333333            0.000000
1             0.0          0.000000            0.000000
2             0.5          0.000000            0.000000
3             1.0          0.000000            0.000000
4             0.0          0.666667            0.333333
…             …             …                  …
18637         1.0          0.333333            0.000000
18638         0.0          0.000000            0.000000
18639         1.0          0.333333            0.000000
18640         1.0          0.333333            0.000000
18641         1.0          0.000000            0.000000

[18642 rows x 27 columns]
```

We will use the scikit-learn SelectKBest method to perform feature selection on our dataset. This we help us select a subset of relevant features from the original set of features.

[157]:
```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2, SelectPercentile

selector = SelectKBest(score_func=chi2, k=10)
X_train_transformed_new = selector.fit_transform(X_train_transformed, y_train)


print("Original number of features:", X_train_transformed.shape[1])
print("Reduced number of features:", X_train_transformed_new.shape[1])
```

```
Original number of features: 27
Reduced number of features: 10
```

[158]:
```
#Our next step is to check the selected features
df = pd.DataFrame({'columns': X_train.columns,
                   'scores': selector.scores_,
                  'selected': selector.get_support()})
df
```

[158]:

	columns	scores	selected
0	h1n1_concern	46.014720	True
1	h1n1_knowledge	47.548372	True
2	behavioral_antiviral_meds	23.036727	False
3	behavioral_avoidance	13.994965	False
4	behavioral_face_mask	96.811001	True
5	behavioral_wash_hands	21.816694	False

```
6   behavioral_large_gatherings     4.075825     False
7       behavioral_outside_home     4.176994     False
8         behavioral_touch_face    31.063768     False
9              doctor_recc_h1n1  2022.913655      True
10        chronic_med_condition   120.804871      True
11         child_under_6_months    54.173526      True
12                health_worker   499.360767      True
13  opinion_h1n1_vacc_effective   120.480308      True
14           opinion_h1n1_risk   580.886916      True
15  opinion_h1n1_sick_from_vacc    34.120955      True
16                    age_group     6.063482     False
17                    education     6.612211     False
18                         race     3.934393     False
19                          sex     5.932438     False
20              income_poverty     2.681979     False
21              marital_status    28.124907     False
22                 rent_or_own    23.210897     False
23           employment_status     2.216011     False
24                  census_msa     0.190748     False
25            household_adults     0.551008     False
26           household_children     0.000312     False
```

[159]:
```
#We then coveert our dataset back to a pandas dataframe using the selected␣
 ↪columns
selected_columns =['h1n1_concern','h1n1_knowledge', 'behavioral_face_mask',␣
 ↪'doctor_recc_h1n1', 'chronic_med_condition', 'child_under_6_months',␣
 ↪'health_worker', 'opinion_h1n1_vacc_effective', 'opinion_h1n1_risk',␣
 ↪'opinion_h1n1_sick_from_vacc']
# X_train_transformed_new = pd.DataFrame(X_train_transformed_new,␣
 ↪columns=['h1n1_concern','h1n1_knowledge', 'behavioral_face_mask',␣
 ↪'doctor_recc_h1n1', 'chronic_med_condition', 'child_under_6_months',␣
 ↪'health_worker', 'opinion_h1n1_vacc_effective', 'opinion_h1n1_risk',␣
 ↪'opinion_h1n1_sick_from_vacc'])
X_train_transformed_new = pd.DataFrame(X_train_transformed_new, columns =␣
 ↪selected_columns)
X_train_transformed_new.head()
```

[159]:
```
   h1n1_concern  h1n1_knowledge  behavioral_face_mask  doctor_recc_h1n1  \
0      0.333333             0.5                   0.0               1.0
1      0.333333             1.0                   0.0               0.0
2      0.333333             1.0                   0.0               0.0
3      0.666667             1.0                   0.0               0.0
4      0.333333             0.5                   0.0               0.0

   chronic_med_condition  child_under_6_months  health_worker  \
0                    0.0                   0.0            0.0
1                    0.0                   0.0            0.0
```

```
2                    0.0                    0.0           0.0
3                    0.0                    0.0           1.0
4                    1.0                    0.0           0.0

   opinion_h1n1_vacc_effective  opinion_h1n1_risk  opinion_h1n1_sick_from_vacc
0                         1.00               0.25                         0.25
1                         1.00               0.25                         0.25
2                         0.50               0.25                         0.25
3                         0.75               0.25                         0.75
4                         0.25               0.00                         0.00
```

[160]:
```
type(X_train_transformed_new)
```

[160]:
```
pandas.core.frame.DataFrame
```

[161]:
```
# we create a SelectKBest object with chi-squared scoring function and select␣
 ↪top 10 features
selector = SelectKBest(score_func=chi2, k=10)

# Next we fit the selector to the test data and transform it
X_test_transformed_new = selector.fit_transform(X_test_transformed, y_test)

# Printing the original number of features in the test data
print("Original number of features:", X_test_transformed.shape[1])

# Printing the reduced number of features after feature selection
print("Reduced number of features:", X_test_transformed_new.shape[1])
```

```
Original number of features: 27
Reduced number of features: 10
```

[162]:
```
# We create a new DataFrame from the transformed test data with selected columns
X_test_transformed_new = pd.DataFrame(X_test_transformed_new,␣
 ↪columns=selected_columns)


X_test_transformed_new.head()
```

[162]:
```
   h1n1_concern  h1n1_knowledge  behavioral_face_mask  doctor_recc_h1n1  \
0      0.666667             0.0                   0.0               0.0
1      0.666667             0.0                   0.0               0.0
2      0.666667             0.0                   0.0               0.0
3      0.000000             0.0                   0.0               0.0
4      0.666667             0.0                   0.0               0.0

   chronic_med_condition  child_under_6_months  health_worker  \
0                    1.0                   0.0            0.0
```

```
1                    0.0                    0.0            0.0
2                    0.0                    0.0            0.0
3                    0.0                    0.0            0.0
4                    1.0                    0.0            1.0

   opinion_h1n1_vacc_effective  opinion_h1n1_risk  opinion_h1n1_sick_from_vacc
0                         0.75               1.00                         0.75
1                         0.75               0.25                         0.25
2                         0.75               0.75                         0.00
3                         0.75               0.00                         0.00
4                         0.75               0.75                         0.25
```

[163]:
```
#Lets attempt the data into the K-Nearest Neighbors model
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train_transformed_new, y_train)
test_preds = knn.predict(X_test_transformed_new)
```

[164]:
```
def print_metrics(labels, preds):
    print("Accuracy Score: {}".format(accuracy_score(labels, preds)))
    print("F1 Score: {}".format(f1_score(labels, preds)))
print_metrics(y_test, test_preds)
```

```
Accuracy Score: 0.793241551939925
F1 Score: 0.3655913978494624
```

This model still has a very low f1 Score of 36%. The model is therefore cannot be relied upon to make accurate predictions.

Decision Tree

[165]:
```
def print_metrics(labels, preds):
    print("Accuracy Score: {}".format(accuracy_score(labels, preds)))
    print("F1 Score: {}".format(f1_score(labels, preds)))

print_metrics(y_test, test_preds)
```

```
Accuracy Score: 0.793241551939925
F1 Score: 0.3655913978494624
```

[166]:
```
#We begin by fitting the decision tree classifier to our training model.
# We initialize a Decision Tree classifier with entropy as the criterion for␣
 ↪splitting
clf = DecisionTreeClassifier(criterion='entropy')

# Our next step is to fit the Decision Tree classifier on the transformed␣
 ↪training data
clf.fit(X_train_transformed_new, y_train)
```

```
clf
```

[166]:
```
DecisionTreeClassifier(criterion='entropy')
```

[167]:
```
#We then begin to make predictions using our test data
y_preds = clf.predict(X_test_transformed_new)

print('Accuracy: ', accuracy_score(y_test, y_preds))
print('F1-Score: ', f1_score(y_test, y_preds))
```

```
Accuracy:  0.7889862327909888
F1-Score:  0.37230081906180196
```

Our f1-score is 37% here. For us to consider our model to be showing a good performance this score should improve. Therefore this calls for us to apply hyperparameter tuning.

Hyperparameter Tuning

[168]:
```
X_train_transformed_new.shape
```

[168]:
```
(18642, 10)
```

[169]:
```
#With this current number of observations and feature
#We could set our:
#max_depth: A value between 5 and 10.
#min_samples_split: A value between 10 and 50.
#min_samples_leaf: A value between 1 and 5.
# Define the hyperparameters grid for tuning
param_grid = {'max_depth': [5, 7, 9, 10],
              'min_samples_split': [12, 25, 36],
              'min_samples_leaf': [2, 3, 5]}

# Initialize the DecisionTreeClassifier
dt = DecisionTreeClassifier()

# Create a GridSearchCV object for hyperparameter tuning
grid_search = GridSearchCV(dt, param_grid, cv=3, scoring='accuracy')

# Fit the GridSearchCV object to find the best hyperparameters
grid_search.fit(X_train_transformed_new, y_train)

# Get the best hyperparameters from the GridSearchCV results
best_params = grid_search.best_params_

# Initialize a new DecisionTreeClassifier with the best hyperparameters
dt_best = DecisionTreeClassifier(max_depth=best_params['max_depth'],
```

```
    min_samples_split=best_params['min_samples_split'],
    min_samples_leaf=best_params['min_samples_leaf'])

# Fit the new DecisionTreeClassifier with the best hyperparameters
dt_best.fit(X_train_transformed_new, y_train)

# Predict the target on the test data using the tuned model
y_pred = dt_best.predict(X_test_transformed_new)

# Calculate the accuracy score
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: {:.2f}%".format(accuracy * 100))

# Calculate the F1 score
f1 = f1_score(y_test, y_pred, average="weighted")

# Print the F1 score and accuracy as percentages
print("F1 Score: {:.2f}%".format(f1 * 100))
```

```
Accuracy: 80.38%
F1 Score: 78.58%
```

Evaluation of the final model

Thhe process of hyperparameter tuning proved to improve our model's f1 score from 37% to 78%. The F1 score of 78.58% represents the harmonic mean of precision and recall. It provides a balanced measure of the model's performance, taking into account both false positives and false negatives. This better performance. This model meets our success metrics and therefore there is no need to repeat the process.

Conclusion

The decision tree model, which has undergone hyperparameter tuning is the perfect choice in this case. This is beacuse of the f1-score of 78%. The model is complex enough to be able to handle the problems of unerfitting and overfitting