final-draft

May 21, 2024

0.1 Student Name: Magdalene Ondimu

0.2 Student Pace : Part time

0.3 Instructor Name: Noah Kandie, William Okwomba

0.4 Phase 3 project

1 Predicting Seasonal Flu Vaccine Uptake

1.1 1. Introduction

Influenza, commonly known as the flu, is a respiratory illness that can lead to severe complications, hospitalization, and even death, particularly in high-risk groups such as older adults, young children, and individuals with pre-existing conditions. An annual flu vaccine can significantly reduce the severity of the flu by prompting the development of protective antibodies within two weeks of vaccination. The Centers for Disease Control and Prevention (CDC) recommends that everyone aged 6 months and older receive the flu vaccine each year .

The H1N1 pandemic began in 2009 when the influenza A virus emerged in the United States, causing an estimated 151,700 to 575,400 deaths worldwide. The pandemic predominantly affected young children and middle-aged adults, while people over 60 were less impacted, likely due to previous exposure to a similar H1N1 virus. A vaccine for H1N1 became publicly available in October 2009, leading the World Health Organization (WHO) to declare an end to the pandemic in August 2010. Nevertheless, the H1N1 virus continues to circulate as a seasonal flu strain

1.1.1 1.1 Problem Statement

In 2019, a novel coronavirus (COVID-19) emerged in Wuhan, China, causing severe respiratory issues similar to pneumonia. The rapid development of the COVID-19 vaccine led to widespread anxiety and reluctance, fueled by various conspiracy theories. This hesitancy mirrored the public's reaction during the H1N1 pandemic, as noted by Relias Media and Seale, H., Heywood, A.E., McLaws, ML. et al..

Vaccine hesitancy, defined as the reluctance or refusal to vaccinate despite the availability of vaccines, is a significant global health threat. It is influenced by personal and social factors and can vary over time. During pandemics, vaccine hesitancy poses heightened risks to public health, as highlighted by Wiysonge CS, Ndwandwe D. et al..

Research indicates a strong correlation between individuals who declined the H1N1 vaccine and those unwilling to receive the COVID-19 vaccine. A study by Nair P, Wales DP. found that 92.9% of those who did not receive the H1N1 vaccine also opposed getting the COVID-19 vaccinTo address

this issue, it is crucial for scientists and public health professionals to understand the underlying reasons for vaccine hesitancy. By analyzing socioeconomic and demographic data, beliefs, and opinions on vaccine effectiveness and risks, targeted public awareness campaigns and new public health policies can be developed to reduce vaccine hesitancy and its associated risks in future pandemics.

1.1.2 1.2 Main Objective

The primary aim of this project is to develop a model that predicts seasonal flu vaccine uptake based on individuals' backgrounds and behavioral patterns.

1.1.3 1.3 Metric for Success

The project will be considered successful if a model is developed that achieves an accuracy of 85% or higher.

1.1.4 1.4 Methodology

The following steps outline the work flow for this project:

2. Data Preparation
3. Data Cleaning
4. Exploratory Data Analysis
5. Data Preprocessing
6. Modelling
7. Evaluation
8. Conculusionemics. .

1.1.5 1.5. Data Description

The datasets used for this project were downloaded from Kaggle. The original data source is the National 2009 H1N1 Flu Survey (NHFS) and it contains information on the social, economic and demographic backgrounds of the respondents as well as their opinions on the H1N1 and seasonal flu vaccines. The data has 26707 rows and 38 columns. The information contained with the columns is as follows as described by the data dictionary:

No.	Column	Description
1	`respondent_id`	Unique and random identifier for the respondents
2	`h1n1_concern`	Level of concern about H1N1 flu with 0 being not concerned at all and 3 being very concerned
3	`h1n1_knowledge`	Level of knowledge about H1N1 with 0 being no knowledge and 2 being a lot of knowledge
4	`behavioral_antiviral_meds`	Has taken any antiviral medication (0-no,1-yes)

No.	Column	Description
5	behavioral_avoidance	Has avoided close contact with anyone with flu-like symptoms (0-no,1-yes)
6	behavioral_face_mask	Has bought a face mask (0-no,1-yes)
7	behavioral_wash_hands	Has frequently washed hands or used hand sanitizer (0-no,1-yes)
8	behavioral_large_gatherings	Has reduced time at large gatherings (0-no,1-yes)
9	behavioral_outside_home	Has reduced contact with people outside of own household (0-no,1-yes)
10	behavioral_touch_face	Has avoided touching eyes, nose or mouth (0-no,1-yes)
11	doctor_recc_h1n1	H1N1 flu vaccine was recommended by doctor (0-no,1-yes)
12	doctor_recc_seasonal	H1N1 flu vaccine was recommended by doctor (0-no,1-yes)
13	chronic_med_condition	Has any of the following chronic conditions: asthma or any lung condition, a heart condition, a kidney condition, sickle cell anaemia or any other anaemia, a neurological or neouromuscular condition, a liver condition, or a weakened immune system as a result of a chronic illness or medicines taken for a chronic illness (0-no,1-yes)
14	child_under_6_months	Has regular close contact with a child under the age of six months (0-no,1-yes)
15	health_worker	Is a healthcare worker (0-no,1-yes)
16	health_insurance	Has health insurance (0-no,1-yes)
17	opinion_h1n1_vacc_effective	Respondent's opinion on the efficacy of the vaccine with 1 being not at all effective and 5 being very effective

No.	Column	Description
18	opinion_h1n1_risk	Respondent's opinion about risk of getting sick with H1N1 flu without vaccine with 1 being very low and 5 being very high
19	opinion_h1n1_sick_from_vacc	Respondent's worry of getting sick from H1N1 vaccine with 1 being not worried at all and 5 being very worried
20	opinion_seas_vacc_effective	Respondent's opinion about seasonal flu vaccine effectiveness with 1 being not effective at all and 5 being very effective
21	opinion_seas_risk	Respondent's opinion about risk of getting sick with seasonal flu without vaccine with 1 being very low and 5 being very high
22	opinion_seas_sick_from_vacc	Respondent's worry of getting sick from taking seasonal flu vaccine with 1 being not worried at all and 5 being very worried
23	age_group	Age group of respondents
24	education	Self-reported educational level
25	race	Race of respondent
26	sex	Sex of respondent
27	income_poverty	Household annual income of respondent with respect to 2008 Census poverty thresholds
28	marital status	Marital status of respondent
29	rent_or_own	Housing situation of respondent
30	employment_status	Employment status of respondent
31	hhs_geo_region	Respondent's residence using a 10-region geographic classification defined by the U.S. Dept. of Health and Human Services. Values are represented as short random character strings

No.	Column	Description
32	census_msa	Respondent's residence within metropolitan statistical areas (MSA) as defined by the U.S. Census
33	household_adults	Number of *other* adults in the household, top-coded to 3
34	household_children	Number of children in the household, top-coded to 3
35	employment_industry	Type of industry respondent is employed in. Values are represented as short random character strings
36	employment_occupation	Type of occupation of respondent.Values are represented as short random character strings
37	h1n1_vaccination	Whether respondent received H1N1 flu vaccine. (0-no, 1-yes)
38	seasonal _vaccination	Whether respondent received seasonal flu vaccine. (0-no, 1-yes)

1.2 2. Data Preparation

1.2.1 2.1 Data Overview

We'll begin by loading the dataset and examining its structure, including the number of rows, columns, and types of data present. This helps us get a high-level understanding of what we're working with.

[1]:
```
# Data manipulation and analysis
import pandas as pd
import numpy as np

# Data visualization
import matplotlib.pyplot as plt
import seaborn as sns


# Machine learning and model evaluation
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler, LabelEncoder
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score, roc_auc_score
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.feature_selection import RFECV
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import ConfusionMatrixDisplay

# Handling class imbalance
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler
from imblearn.combine import SMOTEENN

# Advanced boosting methods
import xgboost as xgb
from xgboost import XGBClassifier
```

[2]:
```
# Load the dataset and preview the first 5 rows and last 5 rows.
df_1 = pd.read_csv(r"C:\Users\Magda\OneDrive\Documents\Flatiron\Phase 3␣
 ↪project\archive (1)\H1N1_Flu_Vaccines.csv")
df_1
```

[2]:

	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	
…	…	…	…	…	
26702	26702	2.0	0.0	0.0	
26703	26703	1.0	2.0	0.0	
26704	26704	2.0	2.0	0.0	
26705	26705	1.0	1.0	0.0	
26706	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  …  rent_or_own   employment_status  \
0                        1.0  …          Own  Not in Labor Force
1                        1.0  …         Rent            Employed
2                        0.0  …          Own            Employed
3                        0.0  …         Rent  Not in Labor Force
4                        1.0  …          Own            Employed
…                        …  …           …                  …
26702                    0.0  …          Own  Not in Labor Force
26703                    0.0  …         Rent            Employed
26704                    1.0  …          Own                 NaN
26705                    NaN  …         Rent            Employed
26706                    0.0  …          Own  Not in Labor Force

       hhs_geo_region                census_msa  household_adults  \
0            oxchjgsf                   Non-MSA               0.0
1           bhuqouqj  MSA, Not Principle  City               0.0
2           qufhixun  MSA, Not Principle  City               2.0
3           lrircsnp       MSA, Principle City               0.0
4           qufhixun  MSA, Not Principle  City               1.0
…              …                        …                 …
26702       qufhixun                   Non-MSA               0.0
26703       lzgpxyit       MSA, Principle City               1.0
26704       lzgpxyit  MSA, Not Principle  City               0.0
26705       lrircsnp                   Non-MSA               1.0
26706       mlyzmhmf       MSA, Principle City               1.0

       household_children  employment_industry  employment_occupation  \
0                     0.0                  NaN                    NaN
1                     0.0            pxcmvdjn               xgwztkwe
2                     0.0            rucpziij               xtkaffoo
```

```
3                     0.0                  NaN                     NaN
4                     0.0              wxleyezf                emcorrxb
…                     …                    …                       …
26702                 0.0                  NaN                     NaN
26703                 0.0              fcxhlnwr                cmhcxjea
26704                 0.0                  NaN                     NaN
26705                 0.0              fcxhlnwr                haliazsg
26706                 0.0                  NaN                     NaN

       h1n1_vaccine  seasonal_vaccine
0                 0                 0
1                 0                 1
2                 0                 0
3                 0                 1
4                 0                 0
…             …                 …
26702             0                 0
26703             0                 0
26704             0                 1
26705             0                 0
26706             0                 0

[26707 rows x 38 columns]
```

The dataset description is given above 1.5 Data description.

[3]:
```
# Lets see the summary of the dataframe
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 38 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
 36  h1n1_vaccine                 26707 non-null  int64
 37  seasonal_vaccine             26707 non-null  int64
dtypes: float64(23), int64(3), object(12)
memory usage: 7.7+ MB
```

This summary gives us detailed information about the dataset's columns. The dataset consists of a mix of numerical and categorical features. Numerical features include 'float64' and int64, while categorical features are of type 'object'. Some features have missing values and this varies across the features. 'health_insurance' has a large number of missing values.

[4]:
```
# Check summary statistics
df_1.describe()
```

[4]:

	respondent_id	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	\
count	26707.000000	26615.000000	26591.000000	26636.000000	
mean	13353.000000	1.618486	1.262532	0.048844	
std	7709.791156	0.910311	0.618149	0.215545	
min	0.000000	0.000000	0.000000	0.000000	
25%	6676.500000	1.000000	1.000000	0.000000	
50%	13353.000000	2.000000	1.000000	0.000000	
75%	20029.500000	2.000000	2.000000	0.000000	
max	26706.000000	3.000000	2.000000	1.000000	

	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	\
count	26499.000000	26688.000000	26665.000000	
mean	0.725612	0.068982	0.825614	

```
std                0.446214             0.253429              0.379448
min                0.000000             0.000000              0.000000
25%                0.000000             0.000000              1.000000
50%                1.000000             0.000000              1.000000
75%                1.000000             0.000000              1.000000
max                1.000000             1.000000              1.000000

       behavioral_large_gatherings  behavioral_outside_home  \
count                  26620.00000             26625.000000
mean                       0.35864                 0.337315
std                        0.47961                 0.472802
min                        0.00000                 0.000000
25%                        0.00000                 0.000000
50%                        0.00000                 0.000000
75%                        1.00000                 1.000000
max                        1.00000                 1.000000

       behavioral_touch_face  …  opinion_h1n1_vacc_effective  \
count           26579.000000  …                 26316.000000
mean                0.677264  …                     3.850623
std                 0.467531  …                     1.007436
min                 0.000000  …                     1.000000
25%                 0.000000  …                     3.000000
50%                 1.000000  …                     4.000000
75%                 1.000000  …                     5.000000
max                 1.000000  …                     5.000000

       opinion_h1n1_risk  opinion_h1n1_sick_from_vacc  \
count       26319.000000                 26312.000000
mean            2.342566                     2.357670
std             1.285539                     1.362766
min             1.000000                     1.000000
25%             1.000000                     1.000000
50%             2.000000                     2.000000
75%             4.000000                     4.000000
max             5.000000                     5.000000

       opinion_seas_vacc_effective  opinion_seas_risk  \
count                 26245.000000       26193.000000
mean                      4.025986           2.719162
std                       1.086565           1.385055
min                       1.000000           1.000000
25%                       4.000000           2.000000
50%                       4.000000           2.000000
75%                       5.000000           4.000000
max                       5.000000           5.000000
```

```
       opinion_seas_sick_from_vacc  household_adults  household_children  \
count                 26170.000000      26458.000000        26458.000000
mean                      2.118112          0.886499            0.534583
std                       1.332950          0.753422            0.928173
min                       1.000000          0.000000            0.000000
25%                       1.000000          0.000000            0.000000
50%                       2.000000          1.000000            0.000000
75%                       4.000000          1.000000            1.000000
max                       5.000000          3.000000            3.000000

       h1n1_vaccine  seasonal_vaccine
count  26707.000000      26707.000000
mean       0.212454          0.465608
std        0.409052          0.498825
min        0.000000          0.000000
25%        0.000000          0.000000
50%        0.000000          0.000000
75%        0.000000          1.000000
max        1.000000          1.000000

[8 rows x 26 columns]
```

For the target variables, "h1n1_vaccine" and "seasonal_vaccine," there are 26 predictor variables. The count of observations varies slightly across columns, ranging from 26170 to 26707. The mean values for "h1n1_vaccine" and "seasonal_vaccine" are 0.212454 and 0.465608, respectivel . Standard deviations for the two target variables are 0.409052 and 0.49885. The minimum and maximum values for both target variables are 0 and 1, indicating they are binary variables. The mean values of the predictor variables range from approximately 0 to 4, indicating varying levels of responses across different questions. The standard deviations for the predictor variables vary, suggesting differing levels of dispersion around the mean. The quartile ranges give insights into the distribution of responses for both target and predictor variables, with 50% of respondents falling within certain ranges for each var iable. These statistics provide a basic understanding of the distribution and central tendencies of the variables in the dataset, aiding further analysis and interpretation.

1.3 3. Data Cleaning

1.3.1 3.1 Drop Columns

[5]:
```
# Lets first drop the columns that do not add value to our analysis and may␣
 ↪introduce noise
columns_to_drop = ['respondent_id', 'employment_industry',
                   'census_msa', 'household_adults', 'household_children']
df_1.drop(columns=columns_to_drop, inplace=True)
df_1.columns
```

[5]:
```
Index(['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', 'doctor_recc_seasonal',
       'chronic_med_condition', 'child_under_6_months', 'health_worker',
       'health_insurance', 'opinion_h1n1_vacc_effective', 'opinion_h1n1_risk',
       'opinion_h1n1_sick_from_vacc', 'opinion_seas_vacc_effective',
       'opinion_seas_risk', 'opinion_seas_sick_from_vacc', 'age_group',
       'education', 'race', 'sex', 'income_poverty', 'marital_status',
       'rent_or_own', 'employment_status', 'hhs_geo_region',
       'employment_occupation', 'h1n1_vaccine', 'seasonal_vaccine'],
      dtype='object')
```

The above columns were dropped so as to focus on the most relevant features by imporoving the efficiency or our model and reduce the risk of overfitting. This approach ensures our analysis is streamlined and focused on the key predictors of H1N1 and sesonal vaccination status.

1.4 3.2 Check for missing values

[6]:
```
# lets identify the missing values
missing_values = df_1.isna().sum()
missing_values
```

[6]:
```
h1n1_concern                    92
h1n1_knowledge                 116
behavioral_antiviral_meds       71
behavioral_avoidance           208
behavioral_face_mask            19
behavioral_wash_hands           42
behavioral_large_gatherings     87
behavioral_outside_home         82
behavioral_touch_face          128
doctor_recc_h1n1              2160
doctor_recc_seasonal          2160
chronic_med_condition          971
child_under_6_months           820
health_worker                  804
health_insurance             12274
opinion_h1n1_vacc_effective    391
opinion_h1n1_risk              388
opinion_h1n1_sick_from_vacc    395
opinion_seas_vacc_effective    462
opinion_seas_risk              514
opinion_seas_sick_from_vacc    537
age_group                        0
education                     1407
race                             0
sex                              0
income_poverty                4423
marital_status                1408
```

```
rent_or_own                  2042
employment_status            1463
hhs_geo_region                  0
employment_occupation       13470
h1n1_vaccine                    0
seasonal_vaccine                0
dtype: int64
```

[7]:
```
# let's check the percentage of missing values
# Calculate the percentage of missing values
missing_percentage = (df_1.isna().sum() / len(df_1)) * 100

# Create a DataFrame to display both count and percentage of missing values
missing_data = pd.DataFrame({'Missing Values': missing_values, 'Percentage':␣
 ↪missing_percentage})

# Print the missing data DataFrame
print(missing_data)
```

```
                           Missing Values  Percentage
h1n1_concern                           92    0.344479
h1n1_knowledge                        116    0.434343
behavioral_antiviral_meds              71    0.265848
behavioral_avoidance                  208    0.778822
behavioral_face_mask                   19    0.071142
behavioral_wash_hands                  42    0.157262
behavioral_large_gatherings            87    0.325757
behavioral_outside_home                82    0.307036
behavioral_touch_face                 128    0.479275
doctor_recc_h1n1                     2160    8.087767
doctor_recc_seasonal                 2160    8.087767
chronic_med_condition                 971    3.635751
child_under_6_months                  820    3.070356
health_worker                         804    3.010447
health_insurance                    12274   45.957989
opinion_h1n1_vacc_effective           391    1.464036
opinion_h1n1_risk                     388    1.452803
opinion_h1n1_sick_from_vacc           395    1.479013
opinion_seas_vacc_effective           462    1.729884
opinion_seas_risk                     514    1.924589
opinion_seas_sick_from_vacc           537    2.010709
age_group                               0    0.000000
education                            1407    5.268282
race                                    0    0.000000
sex                                     0    0.000000
income_poverty                       4423   16.561201
marital_status                       1408    5.272026
```

```
rent_or_own                    2042    7.645936
employment_status              1463    5.477965
hhs_geo_region                    0    0.000000
employment_occupation         13470   50.436215
h1n1_vaccine                      0    0.000000
seasonal_vaccine                  0    0.000000
```

The 'income_poverty' has a high % of missing data at 16%, followed by 'doctor_recc_h1n1' and 'doctor_recc_seasonal' where both have 8% of the same. This means we can fill in the missing values as I dont want to drop any columns.

[8]:
```
# Identify numerical columns
numerical_columns = df_1.select_dtypes(include=['float64', 'int64']).columns

# Fill missing values in numerical columns with the mode
for col in numerical_columns:
    mode_val = df_1[col].mode()[0]
    df_1[col] = df_1[col].fillna(mode_val)

# Identify categorical columns
categorical_columns = df_1.select_dtypes(include=['object']).columns

# Fill missing values in categorical columns with the most frequent value
for col in categorical_columns:
    mode_val = df_1[col].mode()[0]
    df_1[col] = df_1[col].fillna(mode_val)

# Display the DataFrame
df_1
```

[8]:
```
       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                    1.0               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  hhs_geo_region  employment_occupation  \
0      Not in Labor Force        oxchjgsf               xtkaffoo
1                Employed        bhuqouqj               xgwztkwe
2                Employed        qufhixun               xtkaffoo
3      Not in Labor Force        lrircsnp               xtkaffoo
4                Employed        qufhixun               emcorrxb
…                     …               …                      …
26702  Not in Labor Force        qufhixun               xtkaffoo
26703            Employed        lzgpxyit               cmhcxjea
26704            Employed        lzgpxyit               xtkaffoo
26705            Employed        lrircsnp               haliazsg
26706  Not in Labor Force        mlyzmhmf               xtkaffoo

       h1n1_vaccine  seasonal_vaccine
0                 0                 0
1                 0                 1
2                 0                 0
3                 0                 1
4                 0                 0
…               …                 …
26702             0                 0
26703             0                 0
26704             0                 1
26705             0                 0
26706             0                 0

[26707 rows x 33 columns]
```

Filling missing data with the mode, which represents the most frequent value in each column, was chosen to address the missing values in both numerical and categorical columns. This approach ensures that the filled values closely align with the existing data distribution.

When considering numerical columns, using the mean might not be appropriate because integer columns may have a limited range of unique values, making the mean potentially misrepresentative. In such cases, filling with the mode helps maintain the integrity of the data's distribution.

Similarly, for categorical columns, using placeholders like 'unknown' or 'missing' could disrupt the data's inherent structure, especially if these placeholders become prevalent. By filling with the mode, the most common category is used, preserving the categorical distribution and minimizing disruption to downstream analyses.

In summary, employing the mode for filling missing values ensures that the data remains consistent with its original distribution, avoiding potential distortions that could arise from using other methods like mean for numerical data or generic placeholders for categorical data.

[9]:
```
# Check if there are still any missing data
df_1.isna().sum()
```

[9]:
```
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
doctor_recc_seasonal         0
chronic_med_condition        0
child_under_6_months         0
health_worker                0
health_insurance             0
opinion_h1n1_vacc_effective  0
opinion_h1n1_risk            0
opinion_h1n1_sick_from_vacc  0
opinion_seas_vacc_effective  0
opinion_seas_risk            0
opinion_seas_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
employment_occupation        0
h1n1_vaccine                 0
seasonal_vaccine             0
dtype: int64
```

All the missing data has been successfully handled.

1.4.1 3.3 Check for duplicates

[10]:
```
# Check for duplicates

duplicates = df_1.duplicated()

# Count the number of duplicates
num_duplicates = duplicates.sum()
print(f"Number of duplicate rows: {num_duplicates}")
```

```
# Display duplicate rows
duplicate_rows = df_1[duplicates]

print("Duplicate rows:")
print(duplicate_rows)
```

```
Number of duplicate rows: 3
Duplicate rows:
       h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
18054           0.0             1.0                        0.0
22215           2.0             1.0                        0.0
25056           2.0             1.0                        0.0

       behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
18054                   0.0                   0.0                    0.0
22215                   1.0                   0.0                    1.0
25056                   1.0                   0.0                    1.0

       behavioral_large_gatherings  behavioral_outside_home  \
18054                          0.0                      0.0
22215                          0.0                      0.0
25056                          0.0                      0.0

       behavioral_touch_face  doctor_recc_h1n1  …   race   sex  \
18054                    0.0               0.0  …  White  Male
22215                    1.0               0.0  …  White  Male
25056                    1.0               0.0  …  White  Male

                 income_poverty  marital_status  rent_or_own  \
18054  <= $75,000, Above Poverty         Married          Own
22215  <= $75,000, Above Poverty         Married          Own
25056  <= $75,000, Above Poverty         Married          Own

       employment_status  hhs_geo_region  employment_occupation  h1n1_vaccine  \
18054           Employed        lzgpxyit                xgwztkwe             0
22215           Employed        lzgpxyit                xtkaffoo             0
25056           Employed        lzgpxyit                xtkaffoo             0

       seasonal_vaccine
18054                 0
22215                 0
25056                 0

[3 rows x 33 columns]
```

In datasets where each row corresponds to a unique individual, finding identical rows would typically suggest duplicates. However, if the dataset contains multiple entries for different respondents, it's natural to have similar or even identical rows due to respondents sharing similar characteristics or

responses.

In this case, rather than duplicates, these entries likely indicate that multiple respondents provided identical responses across all variables measured in the dataset. This occurrence is common, especially in survey data, where respondents may share similar demographics or attitudes.

Therefore, there's no need for investigation or action to address these rows as they represent valid entries from distinct individuals. It's important to recognize the distinction between true duplicates and repeated patterns among different respondents to ensure appropriate handling of the data

1.4.2 3.4 Unique Values

[11]:
```
# Lets check for unique values
df_1.nunique()
```

[11]:
```
h1n1_concern                 4
h1n1_knowledge               3
behavioral_antiviral_meds    2
behavioral_avoidance         2
behavioral_face_mask         2
behavioral_wash_hands        2
behavioral_large_gatherings  2
behavioral_outside_home      2
behavioral_touch_face        2
doctor_recc_h1n1             2
doctor_recc_seasonal         2
chronic_med_condition        2
child_under_6_months         2
health_worker                2
health_insurance             2
opinion_h1n1_vacc_effective  5
opinion_h1n1_risk            5
opinion_h1n1_sick_from_vacc  5
opinion_seas_vacc_effective  5
opinion_seas_risk            5
opinion_seas_sick_from_vacc  5
age_group                    5
education                    4
race                         4
sex                          2
income_poverty               3
marital_status               2
rent_or_own                  2
employment_status            3
hhs_geo_region              10
employment_occupation       23
h1n1_vaccine                 2
seasonal_vaccine             2
```

```
dtype: int64
```

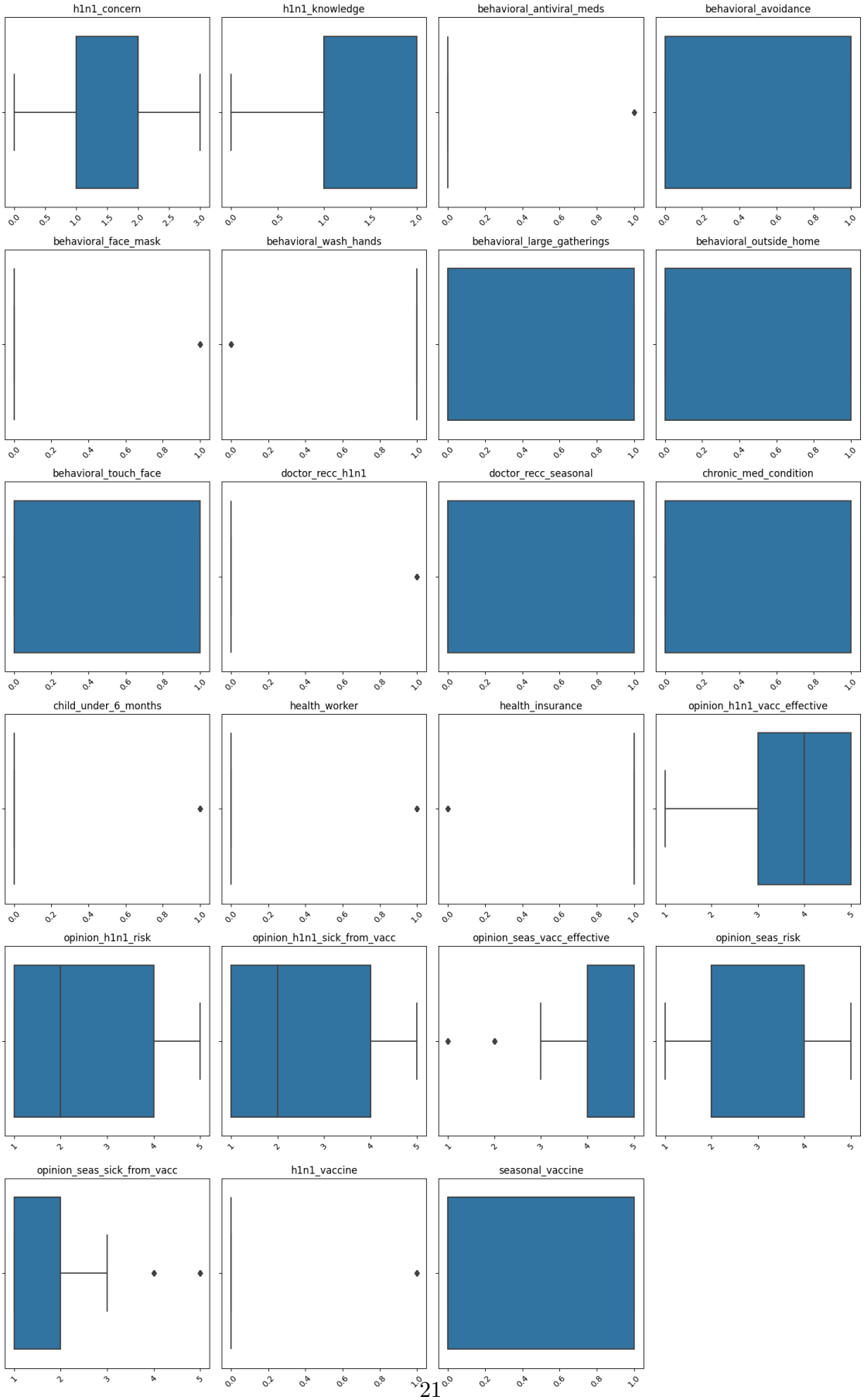
The unique values in the dataset appear to be well within manageable limits, aligning with our analytical objectives. Given the manageable nature of these unique values, preserving the dataset in its current form is a prudent decision. This approach maintains the original granularity and integrity of the data, facilitating a straightforward analysis process without the need for additional data manipulation.

By acknowledging the adequacy of the existing unique values, we ensure that our analytical endeavors remain focused and efficient. This decision underscores our confidence in the dataset's suitability for addressing our research questions effectively. Should any complexities arise during the analysis, we can address them with clarity and precision, leveraging the inherent structure of the data.

In essence, retaining the dataset in its current state affords us the opportunity to capitalize on its inherent richness and coherence, thereby enhancing the robustness and reliability of our analytical insights.

1.4.3 3.5 Checking for outliers

[12]:
```
# Set up the layout for the plots
num_plots = len(numerical_columns)
numb_cols = 4  # Number of columns in the grid
num_rows = (num_plots - 1) // numb_cols + 1  # Calculate the number of rows␣
 ↪needed

# Create a grid of subplots
fig, axes = plt.subplots(num_rows, numb_cols, figsize=(15, 4 * num_rows))

# Flatten the axes array for easier indexing
axes = axes.flatten()

# Plot each numerical column as a box plot
for i, col in enumerate(numerical_columns):
    sns.boxplot(x=df_1[col], ax=axes[i])
    axes[i].set_title(col)
    axes[i].set_xlabel('')
    axes[i].set_ylabel('')
    axes[i].tick_params(axis='x', rotation=45)

# Hide empty subplots
for i in range(num_plots, num_rows * numb_cols):
    axes[i].axis('off')

plt.tight_layout()
plt.show()
```


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
doctor_recc_seasonal
chronic_med_condition
child_under_6_months
health_worker
health_insurance
opinion_h1n1_vacc_effective
opinion_h1n1_risk
opinion_h1n1_sick_from_vacc
opinion_seas_vacc_effective
opinion_seas_risk
opinion_seas_sick_from_vacc
h1n1_vaccine
seasonal_vaccine

In that case, the values identified are not outliers in the traditional sense. Rather, they represent individual responses given by respondents, with each value being one of many responses that share the same characteristic or answer.

These responses are integral to the dataset and provide valuable insights into the distribution of opinions, behaviors, and demographics among the surveyed population. Each response contributes to the overall understanding of the phenomenon under study and helps capture the diversity of perspectives within the dataset.

Therefore, there's no need to treat these values as outliers or take any corrective action. Instead, they should be retained and analyzed in conjunction with the rest of the data to gain a comprehensive understanding of the underlying trends and patterns.

By recognizing the uniqueness and diversity of responses within the dataset, we can ensure that our analysis remains inclusive and representative of the full range of perspectives expressed by respondents.

[13]:
```
# Save the cleaned DataFrame to a CSV file)
import os

# Original file path
original_file_path = r"C:\Users\Magda\OneDrive\Documents\Flatiron\Phase 3␣
 ↪project\archive (1)\H1N1_Flu_Vaccines.csv"

# Extract directory path from the original file path
directory = os.path.dirname(original_file_path)

# Save the cleaned DataFrame to a CSV file in the same directory
df_1.to_csv(os.path.join(directory, 'cleaned_H1N1_Flu_Vaccine.csv'),␣
 ↪index=False)
```

1.5 4. Exploratory Data Analysis

1.5.1 4.1. Univariate Analysis

This section mainly explores the distribution of some features

[14]:
```
# Lets first load our cleaned data
df_1_cleaned = pd.read_csv(r"C:\Users\Magda\OneDrive\Documents\Flatiron\Phase 3␣
 ↪project\archive (1)\cleaned_H1N1_Flu_Vaccine.csv")
df_1_cleaned
```

[14]:

	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                    1.0               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  hhs_geo_region  employment_occupation  \
0      Not in Labor Force        oxchjgsf               xtkaffoo
1                Employed        bhuqouqj               xgwztkwe
2                Employed        qufhixun               xtkaffoo
3      Not in Labor Force        lrircsnp               xtkaffoo
4                Employed        qufhixun               emcorrxb
…                   …             …                     …
26702  Not in Labor Force        qufhixun               xtkaffoo
26703            Employed        lzgpxyit               cmhcxjea
26704            Employed        lzgpxyit               xtkaffoo
26705            Employed        lrircsnp               haliazsg
26706  Not in Labor Force        mlyzmhmf               xtkaffoo

       h1n1_vaccine  seasonal_vaccine
0                 0                 0
1                 0                 1
2                 0                 0
3                 0                 1
4                 0                 0
…               …                 …
26702             0                 0
26703             0                 0
26704             0                 1
26705             0                 0
26706             0                 0

[26707 rows x 33 columns]
```

[15]:
```
# Univariate analysis on numerical columns
numerical_summary = df_1_cleaned.describe()

# Univariate analysis on categorical columns
categorical_summary = df_1_cleaned.describe(include='object')
```

```r
# Display the summary statistics
print("Summary statistics for numerical columns:")
print(numerical_summary)
print("\nSummary statistics for categorical columns:")
print(categorical_summary)
```

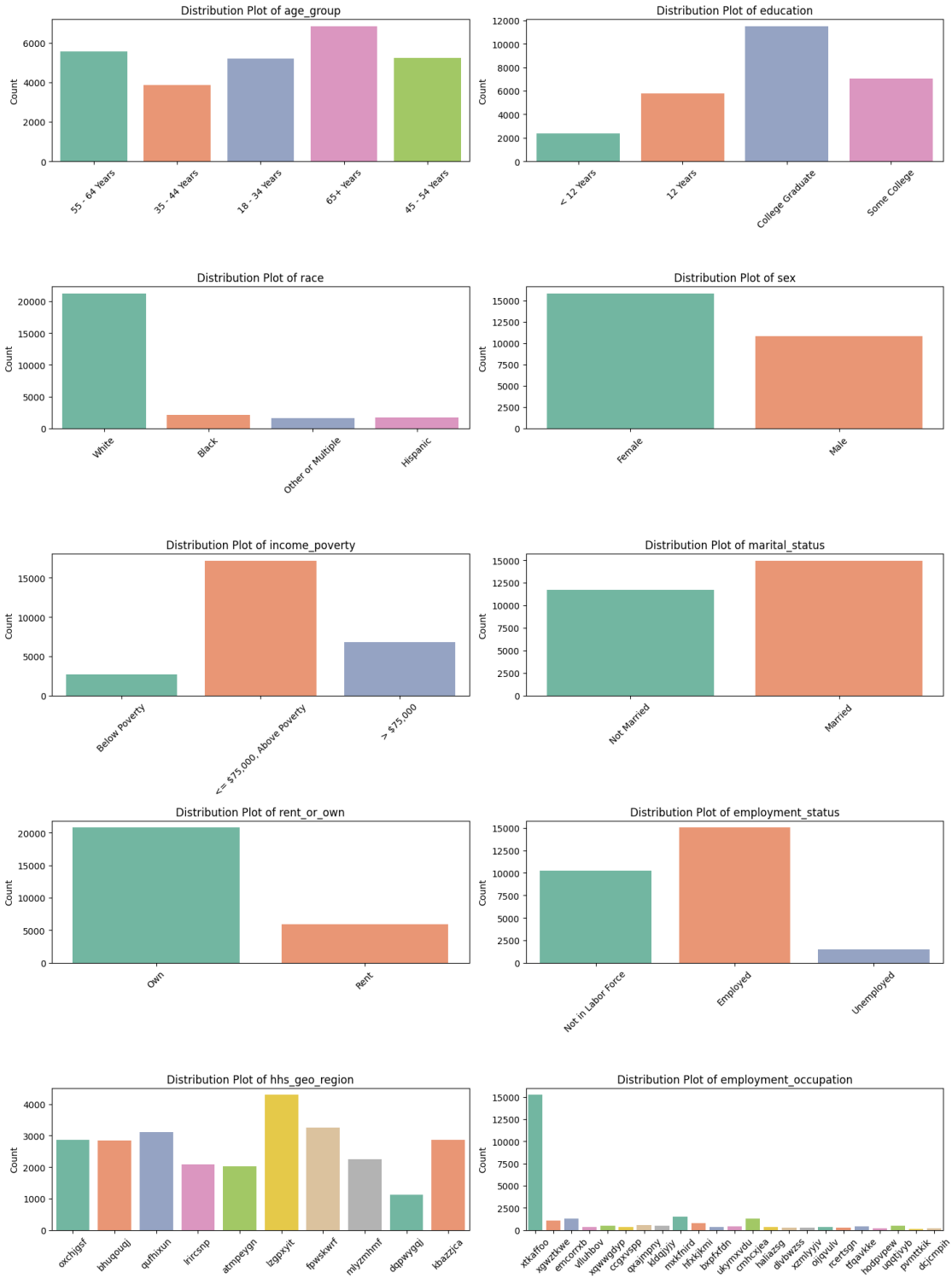
```
Summary statistics for numerical columns:
       h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
count  26707.000000    26707.000000               26707.000000
mean       1.619800        1.261392                   0.048714
std        0.909016        0.617047                   0.215273
min        0.000000        0.000000                   0.000000
25%        1.000000        1.000000                   0.000000
50%        2.000000        1.000000                   0.000000
75%        2.000000        2.000000                   0.000000
max        3.000000        2.000000                   1.000000

       behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
count          26707.000000          26707.000000           26707.000000
mean               0.727749              0.068933               0.825888
std                0.445127              0.253345               0.379213
min                0.000000              0.000000               0.000000
25%                0.000000              0.000000               1.000000
50%                1.000000              0.000000               1.000000
75%                1.000000              0.000000               1.000000
max                1.000000              1.000000               1.000000

       behavioral_large_gatherings  behavioral_outside_home  \
count                 26707.000000             26707.000000
mean                      0.357472                 0.336279
std                       0.479264                 0.472444
min                       0.000000                 0.000000
25%                       0.000000                 0.000000
50%                       0.000000                 0.000000
75%                       1.000000                 1.000000
max                       1.000000                 1.000000

       behavioral_touch_face  doctor_recc_h1n1  …  health_worker  \
count           26707.000000      26707.000000  …   26707.000000
mean                0.678811          0.202494  …       0.108548
std                 0.466942          0.401866  …       0.311077
min                 0.000000          0.000000  …       0.000000
25%                 0.000000          0.000000  …       0.000000
50%                 1.000000          0.000000  …       0.000000
75%                 1.000000          0.000000  …       0.000000
max                 1.000000          1.000000  …       1.000000
```

```
       health_insurance  opinion_h1n1_vacc_effective  opinion_h1n1_risk  \
count      26707.000000                 26707.000000       26707.000000
mean           0.934998                     3.852810           2.337589
std            0.246533                     1.000195           1.276825
min            0.000000                     1.000000           1.000000
25%            1.000000                     3.000000           1.000000
50%            1.000000                     4.000000           2.000000
75%            1.000000                     5.000000           4.000000
max            1.000000                     5.000000           5.000000

       opinion_h1n1_sick_from_vacc  opinion_seas_vacc_effective  \
count                 26707.000000                 26707.000000
mean                      2.352380                     4.025536
std                       1.353339                     1.077131
min                       1.000000                     1.000000
25%                       1.000000                     4.000000
50%                       2.000000                     4.000000
75%                       4.000000                     5.000000
max                       5.000000                     5.000000

       opinion_seas_risk  opinion_seas_sick_from_vacc  h1n1_vaccine  \
count       26707.000000                 26707.000000  26707.000000
mean            2.705321                     2.095630      0.212454
std             1.375216                     1.328782      0.409052
min             1.000000                     1.000000      0.000000
25%             2.000000                     1.000000      0.000000
50%             2.000000                     2.000000      0.000000
75%             4.000000                     2.000000      0.000000
max             5.000000                     5.000000      1.000000

       seasonal_vaccine
count      26707.000000
mean           0.465608
std            0.498825
min            0.000000
25%            0.000000
50%            0.000000
75%            1.000000
max            1.000000

[8 rows x 23 columns]

Summary statistics for categorical columns:
        age_group         education   race     sex             income_poverty  \
count       26707             26707  26707   26707                      26707
unique          5                 4      4       2                          3
top     65+ Years  College Graduate  White  Female  <= $75,000, Above Poverty
freq         6843             11504  21222   15858                      17200
```

```
       marital_status rent_or_own employment_status hhs_geo_region  \
count           26707       26707             26707          26707
unique              2           2                 3             10
top           Married         Own          Employed       lzgpxyit
freq            14963       20778             15023           4297

       employment_occupation
count                  26707
unique                    23
top                 xtkaffoo
freq                   15248
```

This is a summary of the summary statistics for both numerical and categorical columns in the dataframe. These summary statistics provide a comprehensive overview of the distribution and characteristics of the data across both numeric and categorical features, allowing one to better understand the dataset and make informed decisions in subsequent analysis steps.

[16]:
```python
# Set up the layout for the plots
num_plots = len(categorical_columns)
numb_cols = 2  # Number of columns in the grid
num_rows = (num_plots - 1) // numb_cols + 1  # Calculate the number of rows␣
 ↪needed

# Create a grid of subplots
fig, axes = plt.subplots(num_rows, numb_cols, figsize=(15, 4 * num_rows))

# Flatten the axes array for easier indexing
axes = axes.flatten()

# Define a colorful palette
palette = sns.color_palette('Set2')

# Plot each categorical column
for i, col in enumerate(categorical_columns):
    sns.countplot(x=col, data=df_1_cleaned, ax=axes[i], palette=palette)
    axes[i].set_title(f'Distribution Plot of {col}')
    axes[i].set_xlabel('')
    axes[i].set_ylabel('Count')
    axes[i].tick_params(axis='x', rotation=45)

# Hide empty subplots
for i in range(num_plots, num_rows * numb_cols):
    axes[i].axis('off')

plt.tight_layout()
plt.show()
```



Here are the key observations on the respondents based on the provided categorical data:

Age Group: The majority of respondents are 65 years and above, indicating a significant representation of older individuals in the dataset.

Education: Over 11,000 respondents are graduates, suggesting a substantial portion of the surveyed population has attained higher education qualifications.

Race and Gender: A considerable number of respondents identify as white, with over 16,000 identifying as female. This distribution highlights the gender and racial demographics of the surveyed popul

Income and Marital Status: Over 15,000 respondents have an income above the poverty threshold, indicating a relatively affluent sample. Furthermore, more than 13,000 respondents are married, reflecting the marital status distribution among the surveyed population.ation.

Home Ownership and Employment: More than 20,000 respondents own their own homes, indicating a significant proportion of homeowners among the surveyed individuals. Additionally, approximately 15,000 respondents are employed, reflecting the employment status of the sampled population.

Geographic and Occupational Distribution: Around 5,000 respondents come from the 'lzgpxyit' region, while over 15,000 individuals report their employment occupation as 'xtkaffoo'. These findings provide insights into the geographic distribution and employment sectors represented within the dataset.

Overall, these observations provide valuable insights into the demographic and socioeconomic characteristics of the respondents, shedding light on their backgrounds and potential factors influencing their responses.

1.5.2 4.2 Bivariate Analysis

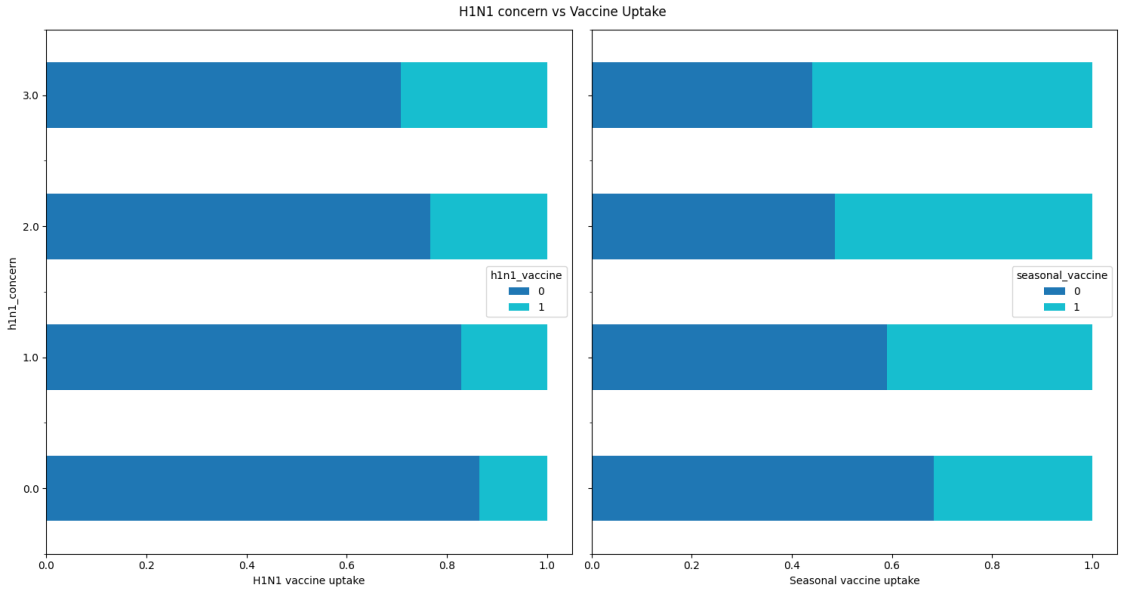
This is a statistical method used to examine the relationship between two variables. This will allow us to identify and quantify associations, dependencies, and correlations between variables.

1.5.3 4.2.1 What is the relationship between concern and H1N1 and vaccines uptake(both h1n1_vaccine and seasonal_vaccine)?

The relationship between concern about H1N1 and vaccine uptake suggests that higher levels of concern regarding contracting H1N1 are likely to correlate with increased uptake of both the seasonal flu vaccine and the H1N1 vaccine. To visualize this relationship, we can create a bar graph depicting the proportion of vaccine uptake across different levels of concern about H1N1.

[17]:
```
# Define custom color map
my_cmap = sns.color_palette("tab10", as_cmap=True)

# Create subplots
fig, (ax_1, ax_2) = plt.subplots(figsize=(15, 8), ncols=2, sharey=True)

# Calculate normalized crosstab for H1N1 vaccine uptake
crosstab_concern1 = pd.crosstab(df_1_cleaned["h1n1_concern"],␣
 ↪df_1_cleaned['h1n1_vaccine'], normalize="index")
```

```
# Plot bar chart for H1N1 vaccine uptake
crosstab_concern1.plot(kind="barh", stacked=True, colormap=my_cmap, ax=ax_1)
ax_1.set_xlabel("H1N1 vaccine uptake")

# Calculate normalized crosstab for seasonal vaccine uptake
crosstab_concern2 = pd.crosstab(df_1_cleaned["h1n1_concern"],␣
 ↪df_1_cleaned['seasonal_vaccine'], normalize="index")

# Plot bar chart for seasonal vaccine uptake
crosstab_concern2.plot(kind="barh", stacked=True, colormap=my_cmap, ax=ax_2)
ax_2.set_xlabel("Seasonal vaccine uptake")

# Set title and adjust layout
fig.suptitle("H1N1 concern vs Vaccine Uptake")
fig.tight_layout()
plt.show()
```



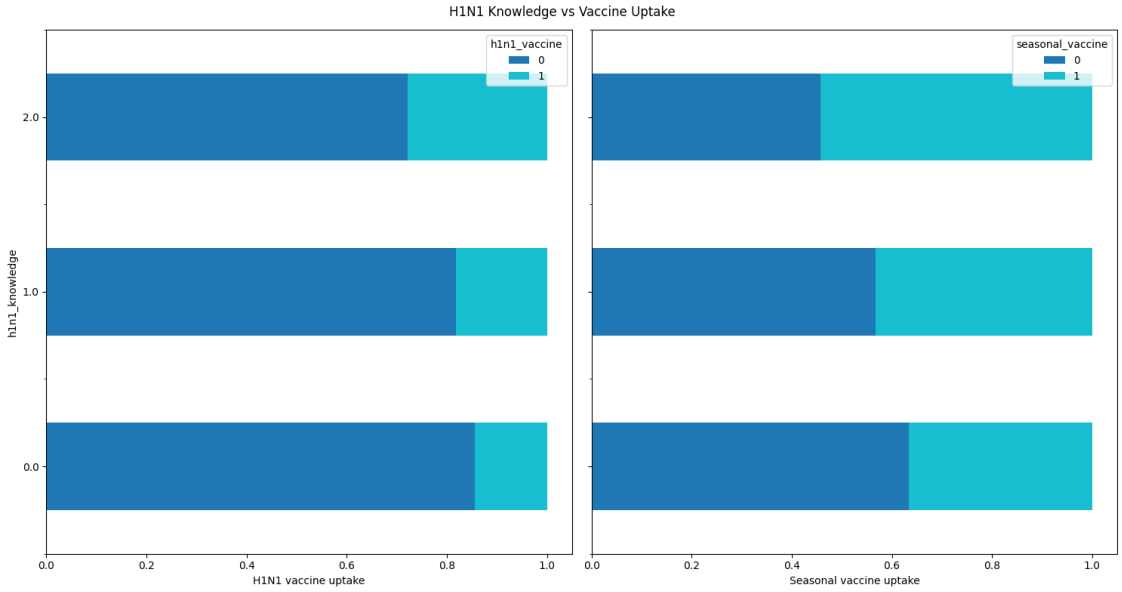
The graph indicates a notable contrast in the levels of concern regarding vaccine uptake between H1N1 and seasonal flu. While a significant portion of respondents exhibited low concern about taking the H1N1 vaccine, there was a considerable increase in concern regarding the uptake of the seasonal flu vaccine.

1.5.4 4.2.2 What is the relationship between knowledge about H1N1 and vaccine uptake?

The relationship between knowledge about H1N1 and vaccine uptake suggests that individuals who possess a deeper understanding of the cause, effects, and prevention methods related to H1N1 are more inclined to seek vaccinations. We can visualize this relationship by plotting a graph that

illustrates the proportion of vaccine uptake across different levels of H1N1 knowledge.

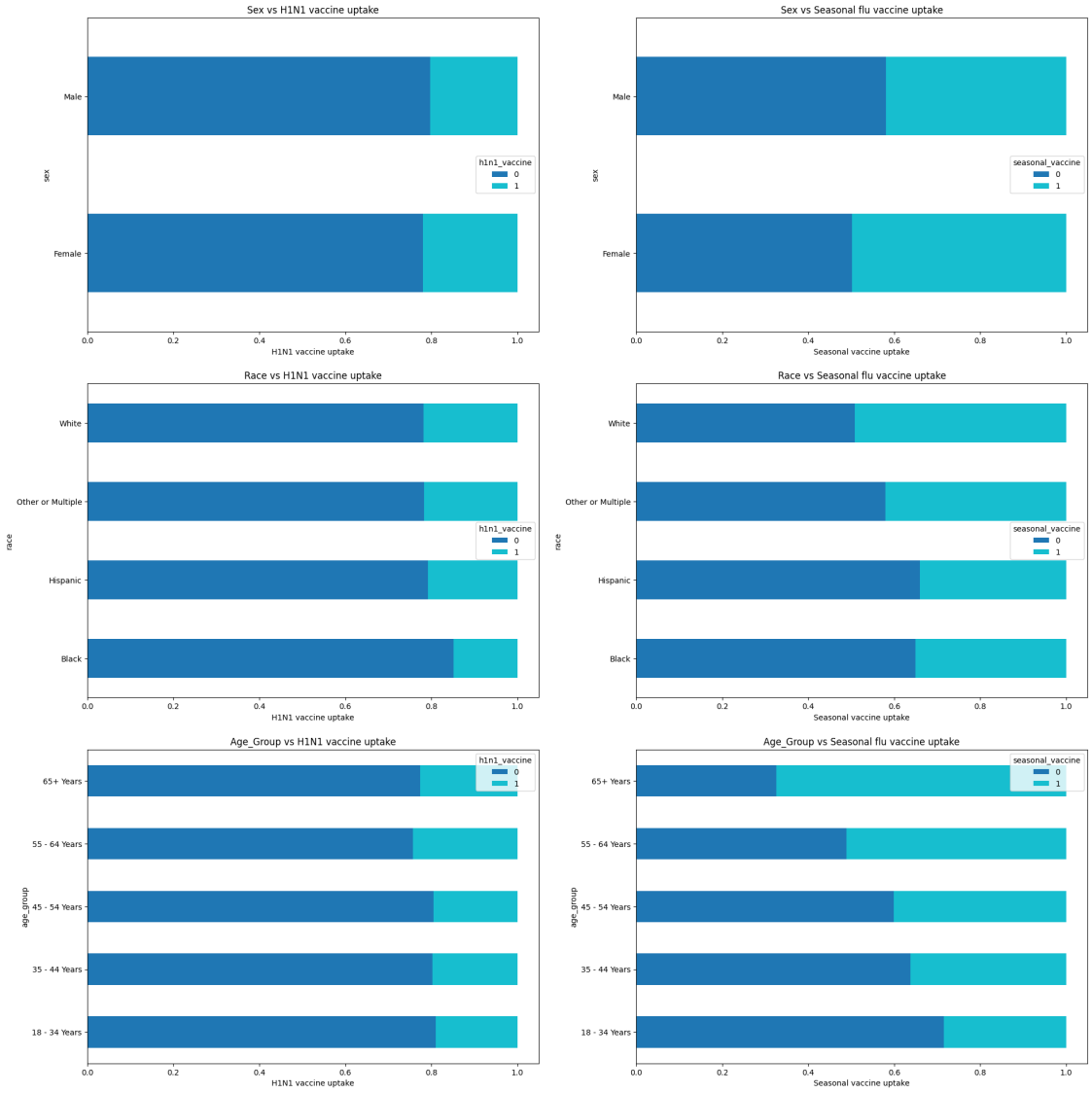
[18]:
```
# Define custom color map
my_cmap = sns.color_palette("tab10", as_cmap=True)

# Create subplots
fig, (ax_1, ax_2) = plt.subplots(figsize=(15, 8), ncols=2, sharey=True)

# Calculate normalized crosstab for H1N1 vaccine uptake
crosstab_know1 = pd.crosstab(df_1_cleaned["h1n1_knowledge"],␣
 ↪df_1_cleaned['h1n1_vaccine'], normalize="index")

# Plot bar chart for H1N1 vaccine uptake
crosstab_know1.plot(kind="barh", stacked=True, colormap=my_cmap, ax=ax_1)
ax_1.set_xlabel("H1N1 vaccine uptake")

# Calculate normalized crosstab for seasonal vaccine uptake
crosstab_know2 = pd.crosstab(df_1_cleaned["h1n1_knowledge"],␣
 ↪df_1_cleaned['seasonal_vaccine'], normalize="index")

# Plot bar chart for seasonal vaccine uptake
crosstab_know2.plot(kind="barh", stacked=True, colormap=my_cmap, ax=ax_2)
ax_2.set_xlabel("Seasonal vaccine uptake")

# Set title and adjust layout
fig.suptitle("H1N1 Knowledge vs Vaccine Uptake")
fig.tight_layout()
plt.show()
```



Despite a considerable portion of respondents exhibiting knowledge about H1N1, there was a notable reluctance among many to uptake the H1N1 vaccine. However, there was a contrasting trend observed with a significant number of respondents demonstrating a willingness to take the seasonal flu vaccine.

1.5.5 4.2.3. What is the relationship between gender, race, age group and vaccine uptake?

[19]:
```
# Define custom color map
my_cmap = sns.color_palette("tab10", as_cmap=True)

# Create subplots
fig, axes = plt.subplots(figsize=(20, 20), ncols=2, nrows=3)

# Define columns to plot
to_plot = ["sex", "race", "age_group"]

# Iterate over columns
for idx, col in enumerate(to_plot):
    left_ax = axes[idx, 0]
    right_ax = axes[idx, 1]

    # Calculate normalized crosstab for H1N1 vaccine uptake
    crosstab1 = pd.crosstab(df_1_cleaned[col], df_1_cleaned['h1n1_vaccine'],␣
 ↪normalize="index")
    crosstab1.plot(kind="barh", stacked=True, colormap=my_cmap, ax=left_ax)
    left_ax.set_xlabel("H1N1 vaccine uptake")
    left_ax.set_title(f"{col.title()} vs H1N1 vaccine uptake")

    # Calculate normalized crosstab for seasonal vaccine uptake
    crosstab2 = pd.crosstab(df_1_cleaned[col],␣
 ↪df_1_cleaned['seasonal_vaccine'], normalize="index")
    crosstab2.plot(kind="barh", stacked=True, colormap=my_cmap, ax=right_ax)
    right_ax.set_xlabel("Seasonal vaccine uptake")
    right_ax.set_title(f"{col.title()} vs Seasonal flu vaccine uptake")

# Adjust layout
fig.tight_layout(pad=2)
plt.show()
```

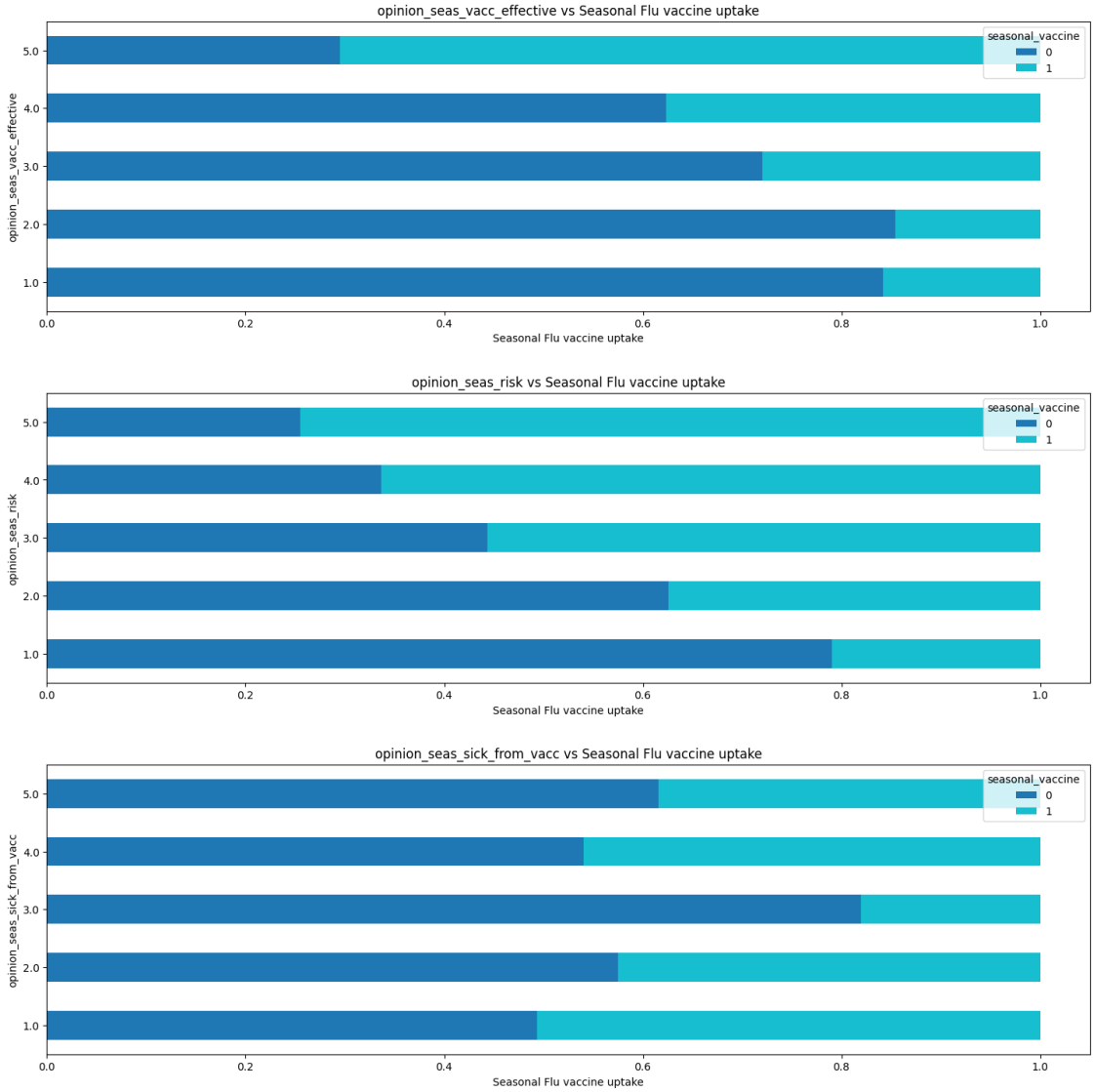


Observations from the graphs reveal the following trends:

Vaccine uptake does not significantly differ between genders, although a slightly higher proportion of women tend to take the seasonal flu vaccine. Across different racial groups, H1N1 vaccine uptake shows a relatively uniform distribution, albeit at low levels. Notably, individuals identifying as Black constitute a minority in vaccine uptake. Conversely, seasonal flu vaccine uptake is substantially higher across all racial groups. Analysis of vaccine uptake across age groups indicates that younger respondents are less inclined to take vaccines for both H1N1 and seasonal flu. Interestingly, there appears to be a linear correlation between age and vaccine uptake, aligning with the understanding that older individuals are at a higher risk of experiencing complications from seasonal flu, as highlighted by the CDC.

1.5.6 4.2.4 Do people's opinions influence seasonal flu vaccine uptake?

[20]:
```
# Define custom color map
my_cmap = sns.color_palette("tab10", as_cmap=True)

# Create subplots
fig, axes = plt.subplots(figsize=(15, 15), nrows=3)

# Define features to plot
features_to_plot = ['opinion_seas_vacc_effective', 'opinion_seas_risk',␣
 ↪'opinion_seas_sick_from_vacc']

# Iterate over features
for idx, col in enumerate(features_to_plot):
    ax = axes[idx]

    # Calculate normalized crosstab for seasonal flu vaccine uptake
    crosstab1 = pd.crosstab(df_1_cleaned[col],␣
 ↪df_1_cleaned['seasonal_vaccine'], normalize="index")
    crosstab1.plot(kind="barh", stacked=True, colormap=my_cmap, ax=ax)
    ax.set_xlabel("Seasonal Flu vaccine uptake")
    ax.set_title(f"{col} vs Seasonal Flu vaccine uptake")

# Adjust layout
fig.tight_layout(pad=3)
plt.show()
```



The graphs depict notable trends in respondents' opinions about the seasonal flu vaccine and their uptake of it:

Respondents who perceived the vaccine as more effective tended to have a higher uptake of the seasonal flu vaccine. Similarly, individuals who viewed the vaccine as posing a risk were also inclined to take it. Furthermore, respondents who believed they would become sick from the vaccine also demonstrated a higher uptake of the seasonal flu vaccine.

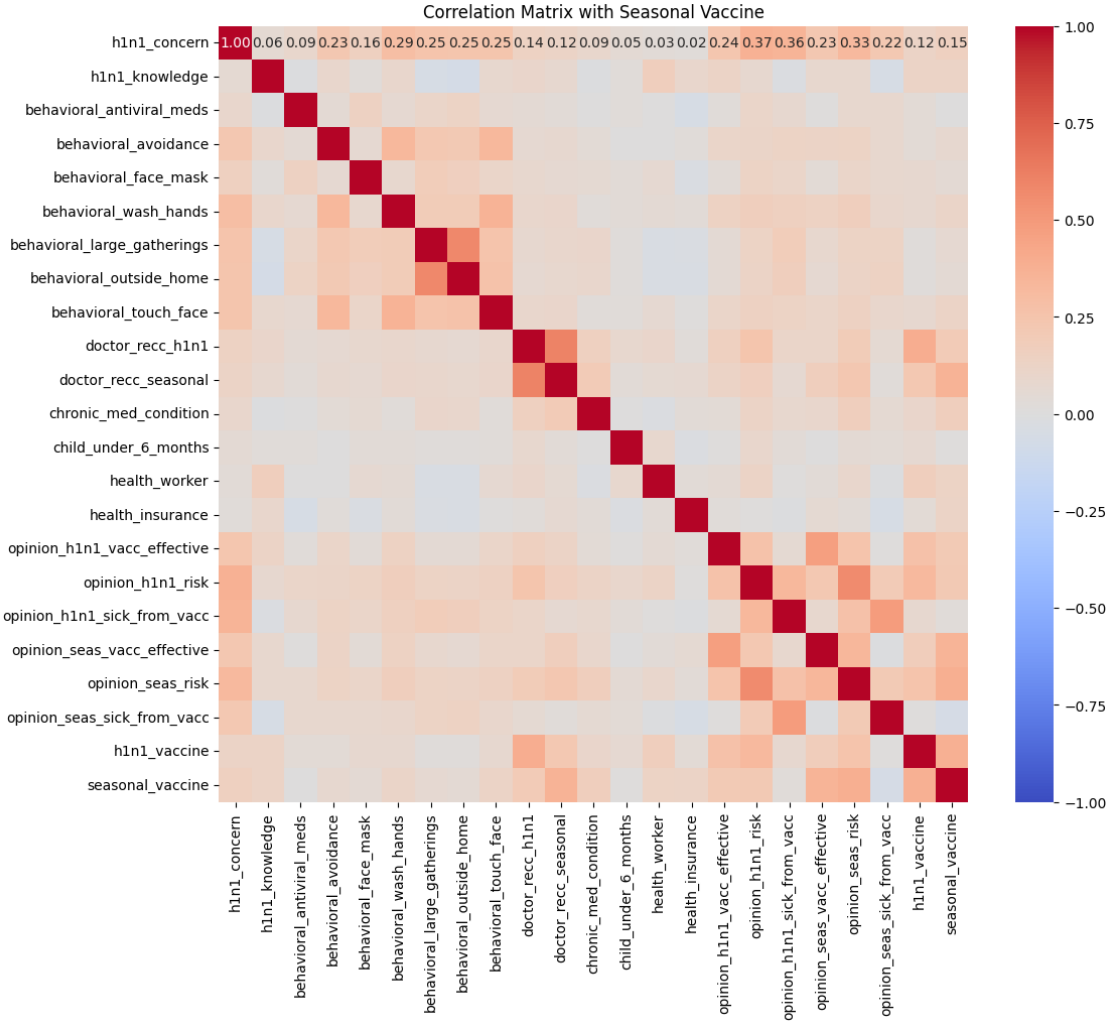
These observations suggest that individuals' perceptions regarding the effectiveness, risk, and potential side effects of the vaccine influence their decision to take it.

1.6 4.3 Multivariate Analysis

Lets consider the joint variation of multiple variables

1.6.1 4.3.1 Correlation Analysis

[21]:
```
# Ensure numerical_columns is a DataFrame (convert it if it's not already)
numerical_columns = df_1_cleaned.select_dtypes(include=['number']).copy()

# Add the target variable 'seasonal_vaccine' to the numerical columns DataFrame
numerical_columns['seasonal_vaccine'] = df_1_cleaned['seasonal_vaccine']

# Calculate the correlation matrix
correlation_matrix = numerical_columns.corr()

# Visualize the correlation matrix with the target variable using a heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f",␣
 ↪vmin=-1, vmax=1)
plt.title('Correlation Matrix with Seasonal Vaccine')
plt.show()
```



The correlation matrix provided shows the correlations between various variables and seasonal vaccine. The values in the matrix range from -1 to 1, where: A value of 1 indicates a perfect positive correlation. A value of -1 indicates a perfect negative correlation. A value of 0 indicates no correlation.

Based on the given data, it appears that there is a moderate positive correlation (0.36) between "doctor recc seasonal" and "seasonal vaccine," indicating that individuals who receive a recommendation for the seasonal vaccine from their doctor are more likely to get vaccinated. Additionally, there is also a weak positive correlation (0.24) between "opinion seas vacc effective" and "seasonal vaccine," suggesting that individuals with a favorable opinion about the effectiveness of the seasonal vaccine are somewhat more likely to get vaccinated. It's important to note that these correlations provide insight into potential relationships but do not imply causation.

1.6.2 4.3.2 Cluster Analysis

Let's identify natural groupings within the data based on the values of multiple variables.

[22]:
```
from sklearn.cluster import KMeans

# Drop non-numeric columns if any
numerical_df = df_1_cleaned.select_dtypes(include=['float64', 'int64'])

# Fill missing values if any
#numerical_df.fillna(numerical_df.mean(), inplace=True)  # You can use␣
 ↪different methods for imputation

# Perform standardization if needed
# from sklearn.preprocessing import StandardScaler
# scaler = StandardScaler()
# numerical_df_scaled = scaler.fit_transform(numerical_df)

# Specify the number of clusters (K)
k = 3  # Adjust as needed

# Initialize KMeans model
kmeans = KMeans(n_clusters=k, random_state=42)

# Fit the model to your data
kmeans.fit(numerical_df)

# Get the cluster labels for each data point
cluster_labels = kmeans.labels_

# Add cluster labels to the DataFrame
numerical_df['Cluster'] = cluster_labels
```

```python
# Analyze the clusters
cluster_summary = numerical_df.groupby('Cluster').mean()
print(cluster_summary)
```

```
         h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
Cluster
0            2.172683        1.302713                   0.082088
1            1.453193        1.282734                   0.032675
2            1.112472        1.142615                   0.032404

         behavioral_avoidance  behavioral_face_mask  behavioral_wash_hands  \
Cluster
0                    0.818119              0.118878               0.926075
1                    0.710889              0.045902               0.806629
2                    0.620008              0.042200               0.707800

         behavioral_large_gatherings  behavioral_outside_home  \
Cluster
0                           0.492987                 0.466199
1                           0.296355                 0.276514
2                           0.281650                 0.266390

         behavioral_touch_face  doctor_recc_h1n1  …  health_worker  \
Cluster                                           …
0                     0.793631          0.328926  …       0.147505
1                     0.643965          0.153767  …       0.093221
2                     0.574039          0.111907  …       0.081387

         health_insurance  opinion_h1n1_vacc_effective  opinion_h1n1_risk  \
Cluster
0                0.934583                     4.192228           3.562773
1                0.946146                     4.063066           1.841115
2                0.909005                     2.793519           1.517898

         opinion_h1n1_sick_from_vacc  opinion_seas_vacc_effective  \
Cluster
0                           3.422281                     4.427685
1                           1.680261                     4.380206
2                           2.207423                     2.517898

         opinion_seas_risk  opinion_seas_sick_from_vacc  h1n1_vaccine  \
Cluster
0                 3.977006                     2.911589      0.357324
1                 2.314385                     1.452012      0.178647
2                 1.556895                     2.298606      0.055953

         seasonal_vaccine
```

```
Cluster
0                0.628075
1                0.491457
2                0.137528

[3 rows x 23 columns]
```

This cluster summary provides insights into the characteristics of each cluster based on the numerical variables.

Cluster 0: This cluster has the highest average values for h1n1_concern, h1n1_knowledge, and behavioral_wash_hands among all clusters, indicating that individuals in this cluster are more concerned about the H1N1 flu, have higher knowledge about it, and are more likely to wash their hands frequently. They also have relatively high values for other behavioral measures such as behavioral_avoidance and behavioral_touch_face. In terms of opinions about H1N1 and seasonal flu vaccines, they have relatively positive opinions and are more likely to have health insurance. This cluster has the highest vaccination rates for both H1N1 and seasonal flu.

Cluster 1: Individuals in this cluster have lower average values for h1n1_concern, h1n1_knowledge, and behavioral_wash_hands compared to Cluster 0, indicating lower levels of concern, knowledge, and hygiene practices regarding the H1N1 flu. They also have lower average values for other behavioral measures and opinions about H1N1 and seasonal flu vaccines. However, they still exhibit moderate vaccination rates, with a slightly lower rate compared to Cluster 0.

Cluster 2: This cluster has the lowest average values for most variables, indicating lower levels of concern, knowledge, and preventive behaviors related to the H1N1 flu compared to the other clusters. They also have less positive opinions about H1N1 and seasonal flu vaccines and lower vaccination rates for both H1N1 and seasonal flu.

Overall, this analysis helps identify distinct groups within the dataset based on their characteristics related to H1N1 flu concerns, knowledge, behaviors, and vaccination rates.

[23]:
```
# Save the cleaned DataFrame after EDA to a new CSV file
df_1_cleaned.to_csv(r'C:\Users\Magda\OneDrive\Documents\Flatiron\Phase 3␣
 ↪project\archive (1)\cleaned_H1N1_Flu_Vaccine_after_EDA.csv', index=False)
```

1.7 5. Data Preprocessing

1.7.1 5.1 Data Transformation

[24]:
```
# Lets first load our cleaned data after EDA
df_cleaned = pd.read_csv (r'C:\Users\Magda\OneDrive\Documents\Flatiron\Phase 3␣
 ↪project\archive (1)\cleaned_H1N1_Flu_Vaccine_after_EDA.csv')
df_cleaned
```

[24]:
```
       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                    1.0               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  hhs_geo_region  employment_occupation  \
0      Not in Labor Force        oxchjgsf               xtkaffoo
1                Employed        bhuqouqj               xgwztkwe
2                Employed        qufhixun               xtkaffoo
3      Not in Labor Force        lrircsnp               xtkaffoo
4                Employed        qufhixun               emcorrxb
…                   …             …                     …
26702  Not in Labor Force        qufhixun               xtkaffoo
26703            Employed        lzgpxyit               cmhcxjea
26704            Employed        lzgpxyit               xtkaffoo
26705            Employed        lrircsnp               haliazsg
26706  Not in Labor Force        mlyzmhmf               xtkaffoo

       h1n1_vaccine  seasonal_vaccine
0                 0                 0
1                 0                 1
2                 0                 0
3                 0                 1
4                 0                 0
…               …                 …
26702             0                 0
26703             0                 0
26704             0                 1
26705             0                 0
26706             0                 0

[26707 rows x 33 columns]
```

[25]:
```
# Lets check if there are any missing values
df_cleaned.isna().sum()
```

[25]:
```
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
doctor_recc_seasonal           0
chronic_med_condition          0
child_under_6_months           0
health_worker                  0
health_insurance               0
opinion_h1n1_vacc_effective    0
opinion_h1n1_risk              0
opinion_h1n1_sick_from_vacc    0
opinion_seas_vacc_effective    0
opinion_seas_risk              0
opinion_seas_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
employment_occupation          0
h1n1_vaccine                   0
seasonal_vaccine               0
dtype: int64
```

[26]:
```
# I  ran my onehotencode code twice coz when am running it again its giving me␣
 ↪the one hot encoded values.
# This is the code but ill comment it.
#cat_columns = ['age_group', 'education', 'race', 'sex', 'income_poverty',
               #'marital_status', 'rent_or_own', 'employment_status',
               #'hhs_geo_region', 'employment_occupation']

# Use pandas' get_dummies function to one-hot encode categorical columns
#df_cleaned = pd.get_dummies(df_cleaned, columns=cat_columns)

# Display the encoded DataFrame
print(df_cleaned)
```

```
      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                    1.0               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  hhs_geo_region  employment_occupation  \
0      Not in Labor Force        oxchjgsf               xtkaffoo
1                Employed        bhuqouqj               xgwztkwe
2                Employed        qufhixun               xtkaffoo
3      Not in Labor Force        lrircsnp               xtkaffoo
4                Employed        qufhixun               emcorrxb
…                   …             …                     …
26702  Not in Labor Force        qufhixun               xtkaffoo
26703            Employed        lzgpxyit               cmhcxjea
26704            Employed        lzgpxyit               xtkaffoo
26705            Employed        lrircsnp               haliazsg
26706  Not in Labor Force        mlyzmhmf               xtkaffoo

       h1n1_vaccine  seasonal_vaccine
0                 0                 0
1                 0                 1
2                 0                 0
3                 0                 1
4                 0                 0
…               …                 …
26702             0                 0
26703             0                 0
26704             0                 1
26705             0                 0
26706             0                 0

[26707 rows x 33 columns]
```

Transformation has been applied successfully to the categorical columns in the dataset. Each categorical column has been converted into multiple boolean columns using one-hot encoding. Now, your dataset contains these additional columns representing the various categories within each original categorical column.

1.7.2 5.1.1 Transform the boolean values to integers(1 and 0)

[27]:
```
# Transform boolean values to integers (1 and 0)
# Identify boolean columns
boolean_columns = df_cleaned.select_dtypes(include=['bool']).columns

# Convert boolean columns to integers
df_cleaned[boolean_columns] = df_cleaned[boolean_columns].astype(int)

# Display the first few rows to verify the transformation
df_cleaned.head()
```

[27]:

	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	…	race	sex	income_poverty	marital_status	rent_or_own	employment_status
0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	…	White	Female	Below Poverty	Not Married	Own	Not in Labor Force
1	3.0	2.0	0.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	…	White	Male	Below Poverty	Not Married	Rent	Employed
2	1.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	…	White	Male	<= $75,000, Above Poverty	Not Married	Own	Employed
3	1.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	0.0	0.0	…	White	Female	Below Poverty	Not Married	Rent	Not in Labor Force
4	2.0	1.0	0.0	1.0	0.0	1.0	1.0	0.0	1.0	0.0	…	White	Female	<= $75,000, Above Poverty	Married	Own	Employed

```
   hhs_geo_region  employment_occupation  h1n1_vaccine  seasonal_vaccine
0        oxchjgsf               xtkaffoo             0                 0
1        bhuqouqj               xgwztkwe             0                 1
2        qufhixun               xtkaffoo             0                 0
3        lrircsnp               xtkaffoo             0                 1
4        qufhixun               emcorrxb             0                 0

[5 rows x 33 columns]
```

[28]:
```
# Check the datatypes of the dataframe
df_cleaned.dtypes
```

[28]:
```
h1n1_concern                 float64
h1n1_knowledge               float64
behavioral_antiviral_meds    float64
behavioral_avoidance         float64
behavioral_face_mask         float64
behavioral_wash_hands        float64
behavioral_large_gatherings  float64
behavioral_outside_home      float64
behavioral_touch_face        float64
doctor_recc_h1n1             float64
doctor_recc_seasonal         float64
chronic_med_condition        float64
child_under_6_months         float64
health_worker                float64
health_insurance             float64
opinion_h1n1_vacc_effective  float64
opinion_h1n1_risk            float64
opinion_h1n1_sick_from_vacc  float64
opinion_seas_vacc_effective  float64
opinion_seas_risk            float64
opinion_seas_sick_from_vacc  float64
age_group                     object
education                     object
race                          object
sex                           object
income_poverty                object
marital_status                object
rent_or_own                   object
employment_status             object
hhs_geo_region                object
employment_occupation         object
h1n1_vaccine                   int64
seasonal_vaccine               int64
dtype: object
```

[29]:
```
# Check the dataframe summary
df_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 33 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   h1n1_concern                 26707 non-null  float64
 1   h1n1_knowledge               26707 non-null  float64
 2   behavioral_antiviral_meds    26707 non-null  float64
 3   behavioral_avoidance         26707 non-null  float64
 4   behavioral_face_mask         26707 non-null  float64
 5   behavioral_wash_hands        26707 non-null  float64
 6   behavioral_large_gatherings  26707 non-null  float64
 7   behavioral_outside_home      26707 non-null  float64
 8   behavioral_touch_face        26707 non-null  float64
 9   doctor_recc_h1n1             26707 non-null  float64
 10  doctor_recc_seasonal         26707 non-null  float64
 11  chronic_med_condition        26707 non-null  float64
 12  child_under_6_months         26707 non-null  float64
 13  health_worker                26707 non-null  float64
 14  health_insurance             26707 non-null  float64
 15  opinion_h1n1_vacc_effective  26707 non-null  float64
 16  opinion_h1n1_risk            26707 non-null  float64
 17  opinion_h1n1_sick_from_vacc  26707 non-null  float64
 18  opinion_seas_vacc_effective  26707 non-null  float64
 19  opinion_seas_risk            26707 non-null  float64
 20  opinion_seas_sick_from_vacc  26707 non-null  float64
 21  age_group                    26707 non-null  object
 22  education                    26707 non-null  object
 23  race                         26707 non-null  object
 24  sex                          26707 non-null  object
 25  income_poverty               26707 non-null  object
 26  marital_status               26707 non-null  object
 27  rent_or_own                  26707 non-null  object
 28  employment_status            26707 non-null  object
 29  hhs_geo_region               26707 non-null  object
 30  employment_occupation        26707 non-null  object
 31  h1n1_vaccine                 26707 non-null  int64
 32  seasonal_vaccine             26707 non-null  int64
dtypes: float64(21), int64(2), object(10)
memory usage: 6.7+ MB
```

The datatypes i have so far are float, integer and objects. Let me handle the object.

[30]:
```
# Select categorical columns (with data types object or category)
```

```
categorical_columns = df_cleaned.select_dtypes(include=['object', 'category']).
 ↪columns

# Display the list of categorical columns
print(categorical_columns)
```

```
Index(['age_group', 'education', 'race', 'sex', 'income_poverty',
       'marital_status', 'rent_or_own', 'employment_status', 'hhs_geo_region',
       'employment_occupation'],
      dtype='object')
```

[31]:
```
# Transform the two categorical columns in my dataframe
# Display the categorical columns identified
print(categorical_columns)

# Perform one-hot encoding for the categorical columns
df_cleaned = pd.get_dummies(df_cleaned, columns=categorical_columns)

# Display the first few rows to verify the transformation
print(df_cleaned.head())
```

```
Index(['age_group', 'education', 'race', 'sex', 'income_poverty',
       'marital_status', 'rent_or_own', 'employment_status', 'hhs_geo_region',
       'employment_occupation'],
      dtype='object')
   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  …  \
```

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

   employment_occupation_qxajmpny  employment_occupation_rcertsgn  \
0                           False                           False
1                           False                           False
2                           False                           False
3                           False                           False
4                           False                           False

   employment_occupation_tfqavkke  employment_occupation_ukymxvdu  \
0                           False                           False
1                           False                           False
2                           False                           False
3                           False                           False
4                           False                           False

   employment_occupation_uqqtjvyb  employment_occupation_vlluhbov  \
0                           False                           False
1                           False                           False
2                           False                           False
3                           False                           False
4                           False                           False

   employment_occupation_xgwztkwe  employment_occupation_xqwwgdyp  \
0                           False                           False
1                            True                           False
2                           False                           False
3                           False                           False
4                           False                           False

   employment_occupation_xtkaffoo  employment_occupation_xzmlyyjv
0                            True                           False
1                           False                           False
2                            True                           False
3                            True                           False
4                           False                           False

[5 rows x 81 columns]
```

[32]:
```
# Convert boolean columns to integers (if there are any left)
df_cleaned = df_cleaned.astype({col: int for col in df_cleaned.
  ↪select_dtypes(include=['bool']).columns})
```

```python
# Display the first few rows to verify the transformation
print(df_cleaned.head())
```

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

   employment_occupation_qxajmpny  employment_occupation_rcertsgn  \
0                               0                               0
1                               0                               0
2                               0                               0
3                               0                               0
4                               0                               0

   employment_occupation_tfqavkke  employment_occupation_ukymxvdu  \
0                               0                               0
1                               0                               0
2                               0                               0
3                               0                               0
4                               0                               0

   employment_occupation_uqqtjvyb  employment_occupation_vlluhbov  \
0                               0                               0
```

```
1                                 0                                 0
2                                 0                                 0
3                                 0                                 0
4                                 0                                 0

   employment_occupation_xgwztkwe  employment_occupation_xqwwgdyp  \
0                               0                               0
1                               1                               0
2                               0                               0
3                               0                               0
4                               0                               0

   employment_occupation_xtkaffoo  employment_occupation_xzmlyyjv
0                               1                               0
1                               0                               0
2                               1                               0
3                               1                               0
4                               0                               0

[5 rows x 81 columns]
```

[33]:
```
df_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 81 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   h1n1_concern                          26707 non-null  float64
 1   h1n1_knowledge                        26707 non-null  float64
 2   behavioral_antiviral_meds             26707 non-null  float64
 3   behavioral_avoidance                  26707 non-null  float64
 4   behavioral_face_mask                  26707 non-null  float64
 5   behavioral_wash_hands                 26707 non-null  float64
 6   behavioral_large_gatherings           26707 non-null  float64
 7   behavioral_outside_home               26707 non-null  float64
 8   behavioral_touch_face                 26707 non-null  float64
 9   doctor_recc_h1n1                      26707 non-null  float64
 10  doctor_recc_seasonal                  26707 non-null  float64
 11  chronic_med_condition                 26707 non-null  float64
 12  child_under_6_months                  26707 non-null  float64
 13  health_worker                         26707 non-null  float64
 14  health_insurance                      26707 non-null  float64
 15  opinion_h1n1_vacc_effective           26707 non-null  float64
 16  opinion_h1n1_risk                     26707 non-null  float64
 17  opinion_h1n1_sick_from_vacc           26707 non-null  float64
 18  opinion_seas_vacc_effective           26707 non-null  float64
 19  opinion_seas_risk                     26707 non-null  float64
```

```
20  opinion_seas_sick_from_vacc                26707 non-null  float64
21  h1n1_vaccine                               26707 non-null  int64
22  seasonal_vaccine                           26707 non-null  int64
23  age_group_18 - 34 Years                    26707 non-null  int32
24  age_group_35 - 44 Years                    26707 non-null  int32
25  age_group_45 - 54 Years                    26707 non-null  int32
26  age_group_55 - 64 Years                    26707 non-null  int32
27  age_group_65+ Years                        26707 non-null  int32
28  education_12 Years                         26707 non-null  int32
29  education_< 12 Years                       26707 non-null  int32
30  education_College Graduate                 26707 non-null  int32
31  education_Some College                     26707 non-null  int32
32  race_Black                                 26707 non-null  int32
33  race_Hispanic                              26707 non-null  int32
34  race_Other or Multiple                     26707 non-null  int32
35  race_White                                 26707 non-null  int32
36  sex_Female                                 26707 non-null  int32
37  sex_Male                                   26707 non-null  int32
38  income_poverty_<= $75,000, Above Poverty   26707 non-null  int32
39  income_poverty_> $75,000                   26707 non-null  int32
40  income_poverty_Below Poverty               26707 non-null  int32
41  marital_status_Married                     26707 non-null  int32
42  marital_status_Not Married                 26707 non-null  int32
43  rent_or_own_Own                            26707 non-null  int32
44  rent_or_own_Rent                           26707 non-null  int32
45  employment_status_Employed                 26707 non-null  int32
46  employment_status_Not in Labor Force       26707 non-null  int32
47  employment_status_Unemployed               26707 non-null  int32
48  hhs_geo_region_atmpeygn                    26707 non-null  int32
49  hhs_geo_region_bhuqouqj                    26707 non-null  int32
50  hhs_geo_region_dqpwygqj                    26707 non-null  int32
51  hhs_geo_region_fpwskwrf                    26707 non-null  int32
52  hhs_geo_region_kbazzjca                    26707 non-null  int32
53  hhs_geo_region_lrircsnp                    26707 non-null  int32
54  hhs_geo_region_lzgpxyit                    26707 non-null  int32
55  hhs_geo_region_mlyzmhmf                    26707 non-null  int32
56  hhs_geo_region_oxchjgsf                    26707 non-null  int32
57  hhs_geo_region_qufhixun                    26707 non-null  int32
58  employment_occupation_bxpfxfdn             26707 non-null  int32
59  employment_occupation_ccgxvspp             26707 non-null  int32
60  employment_occupation_cmhcxjea             26707 non-null  int32
61  employment_occupation_dcjcmpih             26707 non-null  int32
62  employment_occupation_dlvbwzss             26707 non-null  int32
63  employment_occupation_emcorrxb             26707 non-null  int32
64  employment_occupation_haliazsg             26707 non-null  int32
65  employment_occupation_hfxkjkmi             26707 non-null  int32
66  employment_occupation_hodpvpew             26707 non-null  int32
67  employment_occupation_kldqjyjy             26707 non-null  int32
```

```
 68  employment_occupation_mxkfnird            26707 non-null  int32
 69  employment_occupation_oijqvulv            26707 non-null  int32
 70  employment_occupation_pvmttkik            26707 non-null  int32
 71  employment_occupation_qxajmpny            26707 non-null  int32
 72  employment_occupation_rcertsgn            26707 non-null  int32
 73  employment_occupation_tfqavkke            26707 non-null  int32
 74  employment_occupation_ukymxvdu            26707 non-null  int32
 75  employment_occupation_uqqtjvyb            26707 non-null  int32
 76  employment_occupation_vlluhbov            26707 non-null  int32
 77  employment_occupation_xgwztkwe            26707 non-null  int32
 78  employment_occupation_xqwwgdyp            26707 non-null  int32
 79  employment_occupation_xtkaffoo            26707 non-null  int32
 80  employment_occupation_xzmlyyjv            26707 non-null  int32
dtypes: float64(21), int32(58), int64(2)
memory usage: 10.6 MB
```

All the variables including those that were previously categorical or boolean, are now represented as numerical data types suitable for analysis.

[34]:
```
# Lets check for missing values
# Calculate percentage of missing values in each column
missing_percentage = df_cleaned.isnull().mean() * 100

# Display columns with missing values and their corresponding percentages
print("Columns with missing values:")
print(missing_percentage[missing_percentage > 0])
```

```
Columns with missing values:
Series([], dtype: float64)
```

1.7.3 5.2 Drop Columns

[35]:
```
# Lets drop columns not needed for modelling
# List of columns to drop
columns_to_drop = [
    'hhs_geo_region_atmpeygn', 'hhs_geo_region_bhuqouqj',␣
 ↪'hhs_geo_region_dqpwygqj',
    'hhs_geo_region_fpwskwrf', 'hhs_geo_region_kbazzjca',␣
 ↪'hhs_geo_region_lrircsnp',
    'hhs_geo_region_lzgpxyit', 'hhs_geo_region_mlyzmhmf',␣
 ↪'hhs_geo_region_oxchjgsf',
    'hhs_geo_region_qufhixun', 'employment_occupation_bxpfxfdn',␣
 ↪'employment_occupation_ccgxvspp',
    'employment_occupation_cmhcxjea', 'employment_occupation_dcjcmpih',␣
 ↪'employment_occupation_dlvbwzss',
    'employment_occupation_emcorrxb', 'employment_occupation_haliazsg',␣
 ↪'employment_occupation_hfxkjkmi',
```

```
    'employment_occupation_hodpvpew', 'employment_occupation_kldqjyjy',␣
 ↪'employment_occupation_mxkfnird',
    'employment_occupation_oijqvulv', 'employment_occupation_pvmttkik',␣
 ↪'employment_occupation_qxajmpny',
    'employment_occupation_rcertsgn', 'employment_occupation_tfqavkke',␣
 ↪'employment_occupation_ukymxvdu',
    'employment_occupation_uqqtjvyb', 'employment_occupation_vlluhbov',␣
 ↪'employment_occupation_xgwztkwe',
    'employment_occupation_xqwwgdyp', 'employment_occupation_xtkaffoo',␣
 ↪'employment_occupation_xzmlyyjv'
]

# Drop the specified columns
df_cleaned = df_cleaned.drop(columns=columns_to_drop)

# Display the first few rows to verify the transformation
print(df_cleaned.head())

# Display the new shape of the DataFrame to ensure columns are dropped
print(df_cleaned.shape)

# Display the new column names to ensure columns are dropped
print(df_cleaned.columns)
```

```
   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  …  \
0                    1.0               0.0  …
```

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

   income_poverty_<= $75,000, Above Poverty  income_poverty_> $75,000  \
0                                         0                         0
1                                         0                         0
2                                         1                         0
3                                         0                         0
4                                         1                         0

   income_poverty_Below Poverty  marital_status_Married  \
0                             1                       0
1                             1                       0
2                             0                       0
3                             1                       0
4                             0                       1

   marital_status_Not Married  rent_or_own_Own  rent_or_own_Rent  \
0                           1                1                 0
1                           1                0                 1
2                           1                1                 0
3                           1                0                 1
4                           0                1                 0

   employment_status_Employed  employment_status_Not in Labor Force  \
0                           0                                     1
1                           1                                     0
2                           1                                     0
3                           0                                     1
4                           1                                     0

   employment_status_Unemployed
0                             0
1                             0
2                             0
3                             0
4                             0

[5 rows x 48 columns]
(26707, 48)
Index(['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', 'doctor_recc_seasonal',
       'chronic_med_condition', 'child_under_6_months', 'health_worker',
       'health_insurance', 'opinion_h1n1_vacc_effective', 'opinion_h1n1_risk',
```

```
       'opinion_h1n1_sick_from_vacc', 'opinion_seas_vacc_effective',
       'opinion_seas_risk', 'opinion_seas_sick_from_vacc', 'h1n1_vaccine',
       'seasonal_vaccine', 'age_group_18 - 34 Years',
       'age_group_35 - 44 Years', 'age_group_45 - 54 Years',
       'age_group_55 - 64 Years', 'age_group_65+ Years', 'education_12 Years',
       'education_< 12 Years', 'education_College Graduate',
       'education_Some College', 'race_Black', 'race_Hispanic',
       'race_Other or Multiple', 'race_White', 'sex_Female', 'sex_Male',
       'income_poverty_<= $75,000, Above Poverty', 'income_poverty_> $75,000',
       'income_poverty_Below Poverty', 'marital_status_Married',
       'marital_status_Not Married', 'rent_or_own_Own', 'rent_or_own_Rent',
       'employment_status_Employed', 'employment_status_Not in Labor Force',
       'employment_status_Unemployed'],
      dtype='object')
```

We have successfully dropped the columns not needed for further analysis.

1.7.4 5.3 Data Splitting

We shall split the data in training and test sets.

[36]:
```
# Define the target variable
target = 'seasonal_vaccine'

# Define the features (all other columns except the target variable)
X = df_cleaned.drop(columns=[target])

# Define the target variable
y = df_cleaned[target]

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
 ↪random_state=42)

# Print the shapes of the resulting splits to verify
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
X_train shape: (20030, 47)
X_test shape: (6677, 47)
y_train shape: (20030,)
y_test shape: (6677,)
```

After the split the data is ready for modelling.

1.8 6. Modelling

This section look at the iterative modelling process in a bid to arrive at one with the highest accuracy on the test data. A baseline model is built first and its accuracy evaluated.

1.8.1 6.1 Baseline Modelling

A logistic regression model is used as the baseline model as we are dealing with a classification problem. A random state is assigned for reproducibility.

[37]:
```
# Initialize the logistic regression model
baseline_model = LogisticRegression(max_iter=1000)

# Train the model on the training data
baseline_model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = baseline_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
roc_auc = roc_auc_score(y_test, baseline_model.predict_proba(X_test)[:, 1])

# Print the evaluation metrics
print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
print("ROC AUC Score:", roc_auc)
```

```
Accuracy: 0.8081473715740602
Confusion Matrix:
 [[3059  575]
 [ 706 2337]]
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.84      0.83      3634
           1       0.80      0.77      0.78      3043

    accuracy                           0.81      6677
   macro avg       0.81      0.80      0.81      6677
weighted avg       0.81      0.81      0.81      6677

ROC AUC Score: 0.8775378083825469
```

The evaluation results indicate that the logistic regression model performs reasonably well in predicting seasonal flu vaccine uptake.Accuracy: 0.8081 The interpretation of results shows that:

Precision and Recall: The model has balanced precision and recall for both classes, indicating that it performs well in identifying both vaccinated and non-vaccinated individuals. The slightly higher recall for class 0 suggests the model is better at identifying those who won't take the vaccine. F1-Score: The F1-scores are reasonably high, indicating a good balance between precision and recall. ROC AUC Score: The high ROC AUC score indicates the model is effective at distinguishing between those who will and will not take the vaccine.

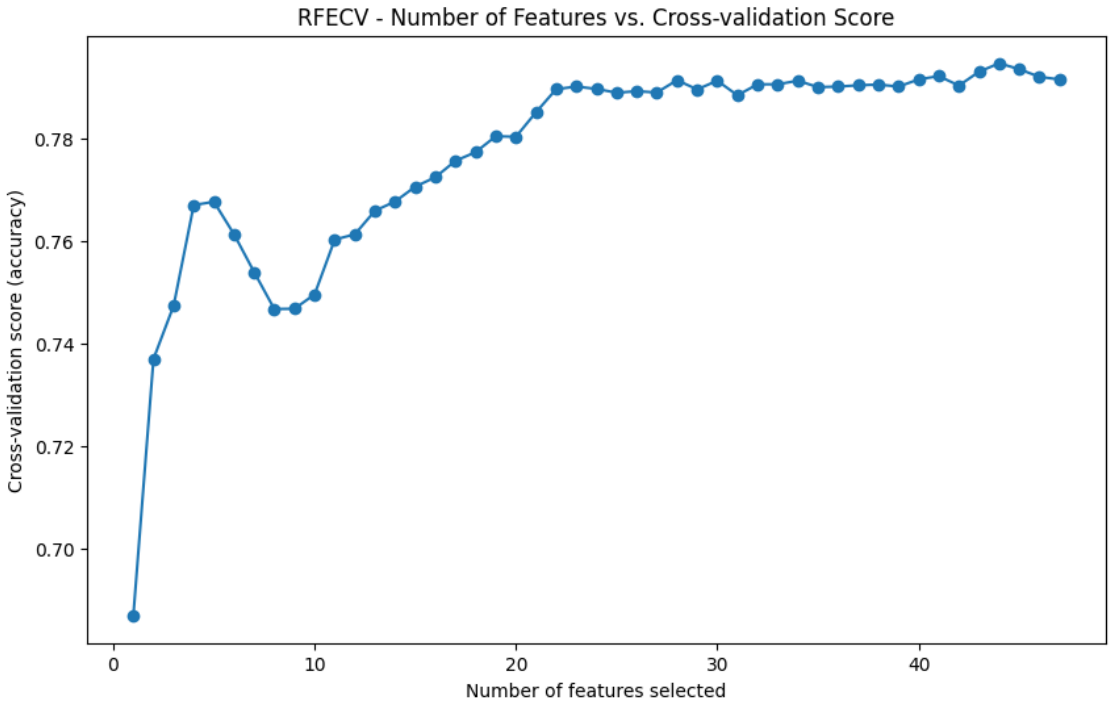
1.9 6.2 Feature Importance

Lets analyze which features are most influential in the model's predictions. This can provide insights into factors that drive vaccine uptake.

1.9.1 6.2.1 Recursive Feature Elimination with Cross-Validation (RFECV)

This is a technique used to select the optimal subset of features for a machine learning model while performing cross-validation to evaluate feature subsets. It recursively removes features, ranks them by importance, and selects the best subset based on a specified evaluation metric.

[38]:
```
# Initialize the Random Forest classifier
rf = RandomForestClassifier(random_state=42)

# Initialize RFECV with Random Forest classifier and 5-fold cross-validation
rfecv = RFECV(estimator=rf, cv=5, scoring='accuracy')

# Fit RFECV to the data
rfecv.fit(X_train, y_train)

# Get the optimal number of features
optimal_num_features = rfecv.n_features_

# Get the selected features
selected_features = X_train.columns[rfecv.support_]

# Plot number of features VS. cross-validation scores
plt.figure(figsize=(10, 6))
plt.xlabel("Number of features selected")
plt.ylabel("Cross-validation score (accuracy)")
plt.plot(range(1, len(rfecv.cv_results_["mean_test_score"]) + 1), rfecv.
 ↪cv_results_["mean_test_score"], marker='o', linestyle='-')
plt.title('RFECV - Number of Features vs. Cross-validation Score')
plt.show()

# Print the optimal number of features and the selected features
print("Optimal number of features:", optimal_num_features)
print("Selected features:", selected_features)
```




```
Optimal number of features: 44
Selected features: Index(['h1n1_concern', 'h1n1_knowledge',
'behavioral_avoidance',
       'behavioral_face_mask', 'behavioral_wash_hands',
       'behavioral_large_gatherings', 'behavioral_outside_home',
       'behavioral_touch_face', 'doctor_recc_h1n1', 'doctor_recc_seasonal',
       'chronic_med_condition', 'child_under_6_months', 'health_worker',
       'health_insurance', 'opinion_h1n1_vacc_effective', 'opinion_h1n1_risk',
       'opinion_h1n1_sick_from_vacc', 'opinion_seas_vacc_effective',
       'opinion_seas_risk', 'opinion_seas_sick_from_vacc', 'h1n1_vaccine',
       'age_group_18 - 34 Years', 'age_group_35 - 44 Years',
       'age_group_45 - 54 Years', 'age_group_55 - 64 Years',
       'age_group_65+ Years', 'education_12 Years', 'education_< 12 Years',
       'education_College Graduate', 'education_Some College', 'race_Black',
       'race_Hispanic', 'race_White', 'sex_Female', 'sex_Male',
       'income_poverty_<= $75,000, Above Poverty', 'income_poverty_> $75,000',
       'income_poverty_Below Poverty', 'marital_status_Married',
       'marital_status_Not Married', 'rent_or_own_Own', 'rent_or_own_Rent',
       'employment_status_Employed', 'employment_status_Not in Labor Force'],
      dtype='object')
```

[39]:
```
# Check the number of features before feature selection
num_features_before = X_train.shape[1]
```

```
# Check the number of features after feature selection
num_features_after = len(selected_features)

# Print the results
print("Number of features before feature selection:", num_features_before)
print("Number of features after feature selection:", num_features_after)
```

```
Number of features before feature selection: 47
Number of features after feature selection: 44
```

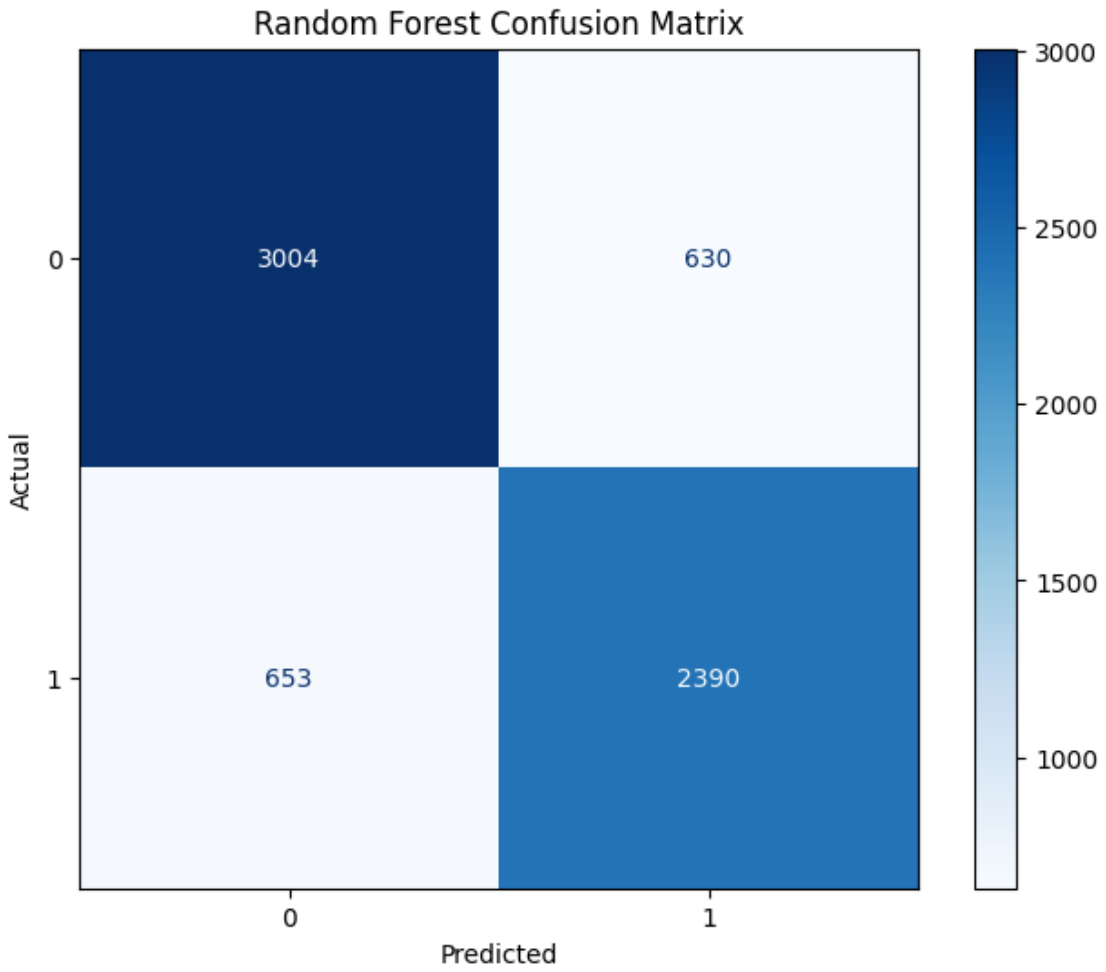
After assessing the number of features I had before and comparing it to the count of features remaining after applying RFECV, I concluded that the difference was negligible. Given the relatively small number of features involved, I determined that there was no need to discard any features. As such, I opted to proceed with utilizing all features in my analysis. Each feature was deemed potentially valuable for predicting the target variable, and retaining them all ensures that the model can capture a comprehensive range of information from the dataset.

1.9.2 6.3 Random Forest Classifier

[40]:
```
# Define the parameter grid
param_grid_rf = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_split': [2, 5, 10]
}

# Initialize the Random Forest model
rf = RandomForestClassifier(random_state=42)

# Perform Grid Search with Cross Validation
grid_search_rf = GridSearchCV(estimator=rf, param_grid=param_grid_rf, cv=5,␣
 ↪n_jobs=-1, verbose=2)
grid_search_rf.fit(X_train, y_train)

# Best parameters and model
best_rf = grid_search_rf.best_estimator_
y_pred_rf = best_rf.predict(X_test)

# Evaluation
accuracy_rf = accuracy_score(y_test, y_pred_rf)
roc_auc_rf = roc_auc_score(y_test, best_rf.predict_proba(X_test)[:, 1])
class_report_rf = classification_report(y_test, y_pred_rf)

# Print results
print("Random Forest Best Parameters:", grid_search_rf.best_params_)
print("Random Forest Accuracy:", accuracy_rf)
print("Random Forest ROC AUC Score:", roc_auc_rf)
print("Random Forest Classification Report:\n", class_report_rf)
```

```python
# Confusion Matrix Visualization using ConfusionMatrixDisplay
conf_matrix_rf = confusion_matrix(y_test, y_pred_rf)
disp_rf = ConfusionMatrixDisplay(confusion_matrix=conf_matrix_rf)
plt.figure(figsize=(8, 6))
disp_rf.plot(ax=plt.gca(), cmap='Blues')
plt.title('Random Forest Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Random Forest Best Parameters: {'max_depth': 30, 'min_samples_split': 10,
'n_estimators': 300}
Random Forest Accuracy: 0.8078478358544257
Random Forest ROC AUC Score: 0.8782474135628185
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.83      0.82      3634
           1       0.79      0.79      0.79      3043

    accuracy                           0.81      6677
   macro avg       0.81      0.81      0.81      6677
weighted avg       0.81      0.81      0.81      6677
```



The Random Forest model was trained using 5-fold cross-validation with grid search over 27 parameter combinations to identify the best hyperparameters. The optimal parameters determined by the grid search were {'max_depth': 30, 'min_samples_split': 10, 'n_estimators': 300}.

The model performance was: Accuracy of 80.78% and ROC AUC of 0.8782. These metrics suggest that the model performs well overall, providing a good balance between true positive and true negative rates.

The classification report indicates that the model is slightly better at predicting class 0 (not receiving the vaccine) compared to class 1 (receiving the vaccine), but it still maintains a reasonable balance in performance across both classes.

The confusion matrix provides a detailed breakdown of the model's predictions: The model correctly identified 3004 individuals who did not receive the vaccine (True Negatives). It incorrectly identified 630 individuals as having received the vaccine when they did not (False Positives). It correctly identified 2390 individuals who did receive the vaccine (True Positives). It incorrectly identified 653 individuals as not having received the vaccine when they did (False Negatives).

Overall Performance: The model has an accuracy of approximately 80.78%, indicating it is generally

effective at predicting vaccine uptake. Class Imbalance Handling: The precision and recall for both classes (around 79% for class 1 and slightly higher for class 0) suggest that the model is reasonably good at distinguishing between those who do and do not take the vaccine, though there is still room for improvement. Error Analysis: The number of false positives (630) and false negatives (653) highlights areas where the model can be refined to reduce misclassificatisn

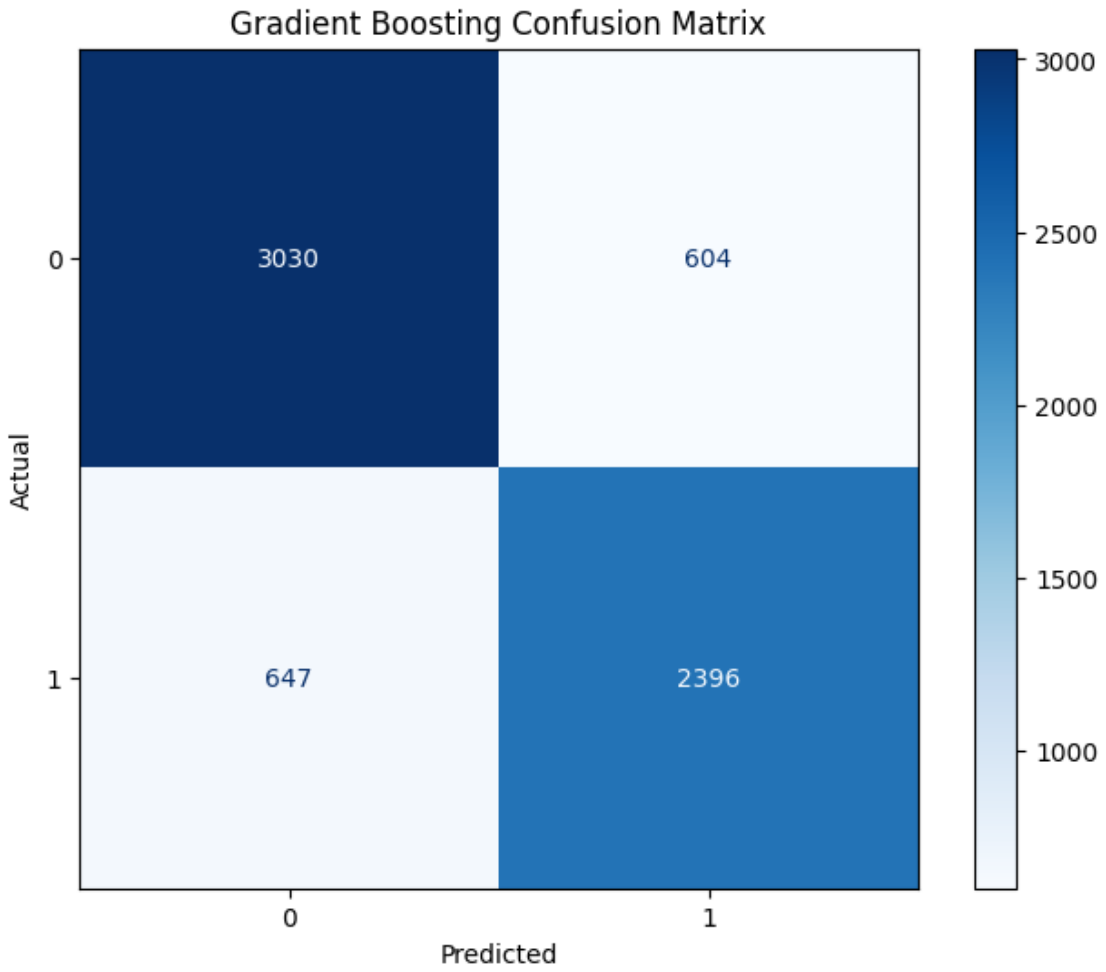
The Random Forest model demonstrates robust performance in predicting seasonal flu vaccine uptake based on a person's background and behavioral patterns. While the model performs well overall, there is potential to enhance its precision and recall, particularly for class 1 (receiving the vaccine). This analysis provides valuable insights for further model refinements and adjustments to improve prediction accuracy

1.9.3 6.4 Gradient Boosting Classifierss.

[41]:
```
# Define the parameter grid
param_grid_gb = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7]
}

# Initialize the Gradient Boosting model
gb = GradientBoostingClassifier(random_state=42)

# Perform Grid Search with Cross Validation
grid_search_gb = GridSearchCV(estimator=gb, param_grid=param_grid_gb, cv=5,␣
 ↪n_jobs=-1, verbose=2)
grid_search_gb.fit(X_train, y_train)

# Best parameters and model
best_gb = grid_search_gb.best_estimator_
y_pred_gb = best_gb.predict(X_test)

# Evaluation
accuracy_gb = accuracy_score(y_test, y_pred_gb)
roc_auc_gb = roc_auc_score(y_test, best_gb.predict_proba(X_test)[:, 1])
class_report_gb = classification_report(y_test, y_pred_gb)

# Print results
print("Gradient Boosting Best Parameters:", grid_search_gb.best_params_)
print("Gradient Boosting Accuracy:", accuracy_gb)
print("Gradient Boosting ROC AUC Score:", roc_auc_gb)
print("Gradient Boosting Classification Report:\n", class_report_gb)

# Confusion Matrix Visualization using ConfusionMatrixDisplay
conf_matrix_gb = confusion_matrix(y_test, y_pred_gb)
disp_gb = ConfusionMatrixDisplay(confusion_matrix=conf_matrix_gb)
plt.figure(figsize=(8, 6))
```

```
disp_gb.plot(ax=plt.gca(), cmap='Blues')
plt.title('Gradient Boosting Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Gradient Boosting Best Parameters: {'learning_rate': 0.1, 'max_depth': 3,
'n_estimators': 200}
Gradient Boosting Accuracy: 0.8126404073685787
Gradient Boosting ROC AUC Score: 0.8846940414325506
Gradient Boosting Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.83      0.83      3634
           1       0.80      0.79      0.79      3043

    accuracy                           0.81      6677
   macro avg       0.81      0.81      0.81      6677
weighted avg       0.81      0.81      0.81      6677
```



The Gradient Boosting model was trained using 5-fold cross-validation with grid search over 27 parameter combinations to identify the best hyperparameters. The optimal parameters determined by the grid search were {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200}.

Model Performance has the Accuracy of 81.26% and ROC AUC Score of 0.8847. These metrics suggest that the Gradient Boosting model performs well overall, with a slightly higher ROC AUC score compared to the Random Forest model, indicating better performance in distinguishing between the classes.

The classification report indicates that the model has balanced performance for both classes, with slightly better recall and F1-score for class 0 (not receiving the vaccine).

The confusion matrix provides a detailed breakdown of the model's predictions:

The model correctly identified 3030 individuals who did not receive the vaccine (True Negatives). It incorrectly identified 604 individuals as having received the vaccine when they did not (False Positives). It correctly identified 2396 individuals who did receive the vaccine (True Positives). It incorrectly identified 647 individuals as not having received the vaccine when they did (False Negati

Overall Performance: The model has an accuracy of approximately 81.26%, indicating it is generally effective at predicting vaccine uptake. Class Imbalance Handling: The precision and recall for both classes (around 79% for class 1 and slightly higher for class 0) suggest that the model is reasonably good at distinguishing between those who do and do not take the vaccine. Error Analysis: The number of false positives (604) and false negatives (647) is slightly lower than in the Random Forest model, indicating slightly better performanc

True Negatives (TN): 3030 - These are the correctly predicted cases where individuals did not receive the vaccine. False Positives (FP): 604 - These are the cases where individuals were incorrectly predicted to have received the vaccine. False Negatives (FN): 647 - These are the cases where individuals who received the vaccine were incorrectly predicted not to have received it. True Positives (TP): 2396 - These are the correctly predicted cases where individuals received the vacci

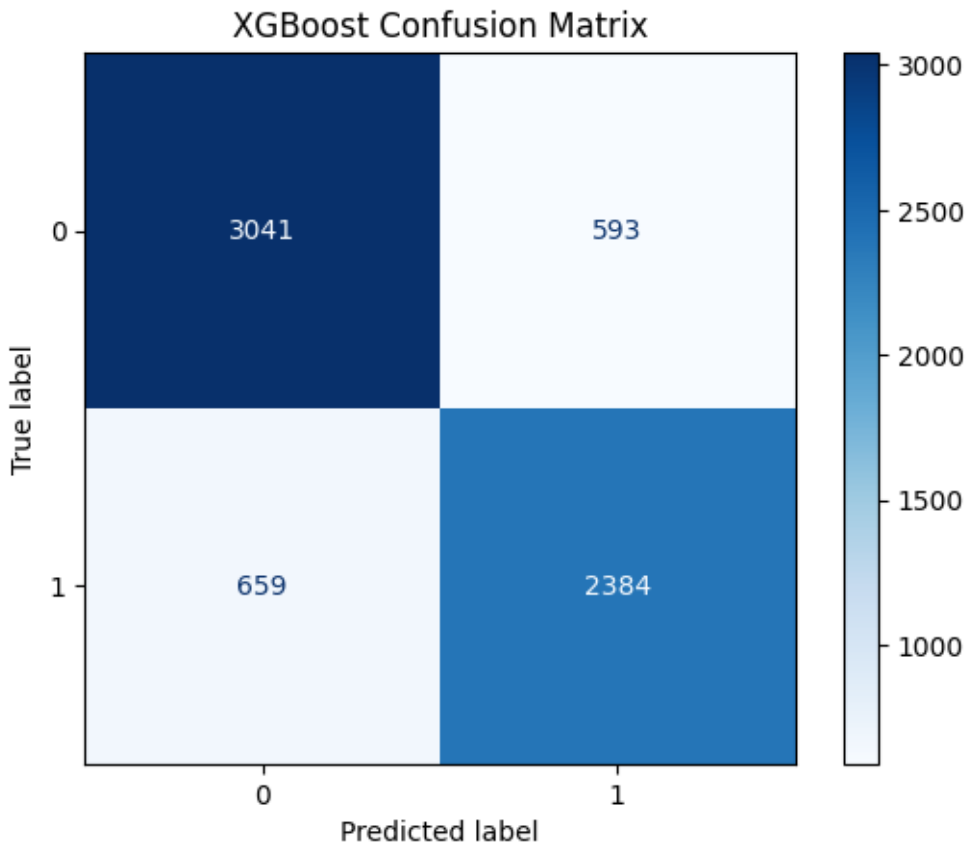
The Gradient Boosting model demonstrates robust performance in predicting seasonal flu vaccine uptake based on a person's background and behavioral patterns. The model achieves a higher accuracy and ROC AUC score compared to the Random Forest model, indicating improved performance. The classification report and confusion matrix analysis reveal balanced precision and recall for both classes, with slightly better error rates. These results provide valuable insights into the model's strengths and areas for potential improvement.ne.e. ves).

1.10 6.5 XGBoost Classifier

[42]:
```
# Ensure all feature names are strings and remove any problematic characters
X_train.columns = [str(col).replace('[', '').replace(']', '').replace('<', '').
 ↪replace('>', '') for col in X_train.columns]
X_test.columns = [str(col).replace('[', '').replace(']', '').replace('<', '').
 ↪replace('>', '') for col in X_test.columns]

# Define the parameter grid for XGBoost
param_grid_xgb = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}

# Initialize the XGBoost model
xgb = XGBClassifier(random_state=42)

# Perform Grid Search with Cross Validation
grid_search_xgb = GridSearchCV(estimator=xgb, param_grid=param_grid_xgb, cv=5,␣
 ↪n_jobs=-1, verbose=2)
grid_search_xgb.fit(X_train, y_train)

# Best parameters and model
best_xgb = grid_search_xgb.best_estimator_
y_pred_xgb = best_xgb.predict(X_test)
```

```
# Evaluation
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
roc_auc_xgb = roc_auc_score(y_test, best_xgb.predict_proba(X_test)[:, 1])
conf_matrix_xgb = confusion_matrix(y_test, y_pred_xgb)
class_report_xgb = classification_report(y_test, y_pred_xgb)

# Print results
print("XGBoost Best Parameters:", grid_search_xgb.best_params_)
print("XGBoost Accuracy:", accuracy_xgb)
print("XGBoost ROC AUC Score:", roc_auc_xgb)
print("XGBoost Classification Report:\n", class_report_xgb)

# Confusion Matrix Visualization
disp = ConfusionMatrixDisplay(confusion_matrix=conf_matrix_xgb,␣
 ↪display_labels=best_xgb.classes_)
disp.plot(cmap='Blues')
plt.title('XGBoost Confusion Matrix')
plt.show()
```

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
XGBoost Best Parameters: {'colsample_bytree': 0.8, 'learning_rate': 0.2,
'max_depth': 3, 'n_estimators': 100, 'subsample': 1.0}
XGBoost Accuracy: 0.8124906395087614
XGBoost ROC AUC Score: 0.8849819709462482
XGBoost Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.84      0.83      3634
           1       0.80      0.78      0.79      3043

    accuracy                           0.81      6677
   macro avg       0.81      0.81      0.81      6677
weighted avg       0.81      0.81      0.81      6677
```



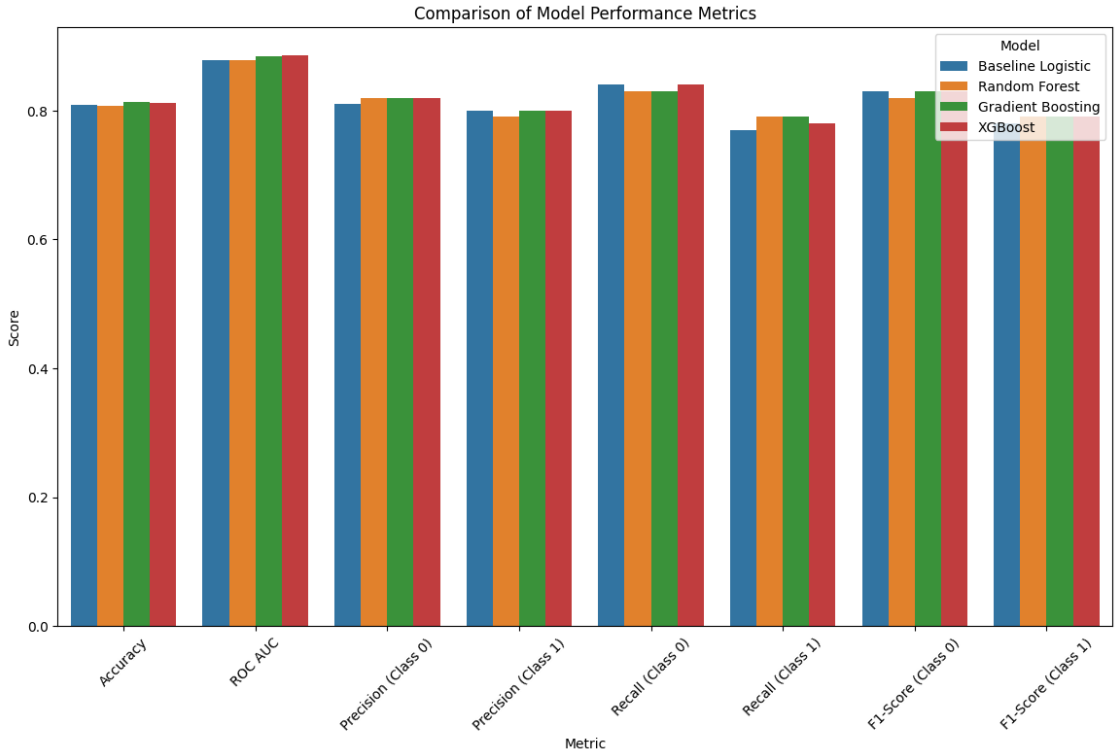
The XGBoost model was trained with 5-fold cross-validation using grid search over 108 parameter combinations. The best parameters found by the grid search were {'colsample_bytree': 0.8, 'learning_rate': 0.2, 'max_depth': 3, 'n_estimators': 100, 'subsample': 1.0}.

Accuracy (81.25%): This indicates that approximately 81.25% of the predictions made by the model are correct. ROC AUC Score (0.8850): This suggests that the model has a good ability to distinguish between the two classes (those who received the vaccine and those who did not).

Precision and Recall: The precision for class 0 is slightly higher than for class 1, indicating the model is slightly better at identifying those who did not receive the vaccine. The recall is also higher for class 0, suggesting the model is more effective at identifying actual negatives. Balanced Performance: Both precision and recall for class 1 are reasonably high, indicating the model performs well in predicting those who receive the vaccine, though there is a slight tendency towards false negatives (missed predictions of those who received the vaccine). Overall, the XGBoost model demonstrates strong performance in predicting seasonal flu vaccine uptake, with good accuracy and ROC AUC scores, and balanced precision and recall metrics across both classes.

1.11 7. Evaluation

[43]:
```
# Create a dataframe with the metrics
metrics_df = pd.DataFrame({
    'Model': ['Baseline Logistic', 'Random Forest', 'Gradient Boosting',␣
 ↪'XGBoost'],
    'Accuracy': [0.8081, 0.8078, 0.8126, 0.8125],
    'ROC AUC': [0.8775, 0.8782, 0.8847, 0.8850],
    'Precision (Class 0)': [0.81, 0.82, 0.82, 0.82],
    'Precision (Class 1)': [0.80, 0.79, 0.80, 0.80],
    'Recall (Class 0)': [0.84, 0.83, 0.83, 0.84],
    'Recall (Class 1)': [0.77, 0.79, 0.79, 0.78],
    'F1-Score (Class 0)': [0.83, 0.82, 0.83, 0.83],
    'F1-Score (Class 1)': [0.78, 0.79, 0.79, 0.79]
})

# Melt the dataframe for easier plotting
metrics_melted_df = metrics_df.melt(id_vars='Model', var_name='Metric',␣
 ↪value_name='Value')

# Create a barplot
plt.figure(figsize=(14, 8))
sns.barplot(x='Metric', y='Value', hue='Model', data=metrics_melted_df)
plt.title('Comparison of Model Performance Metrics')
plt.xticks(rotation=45)
plt.ylabel('Score')
plt.show()
```



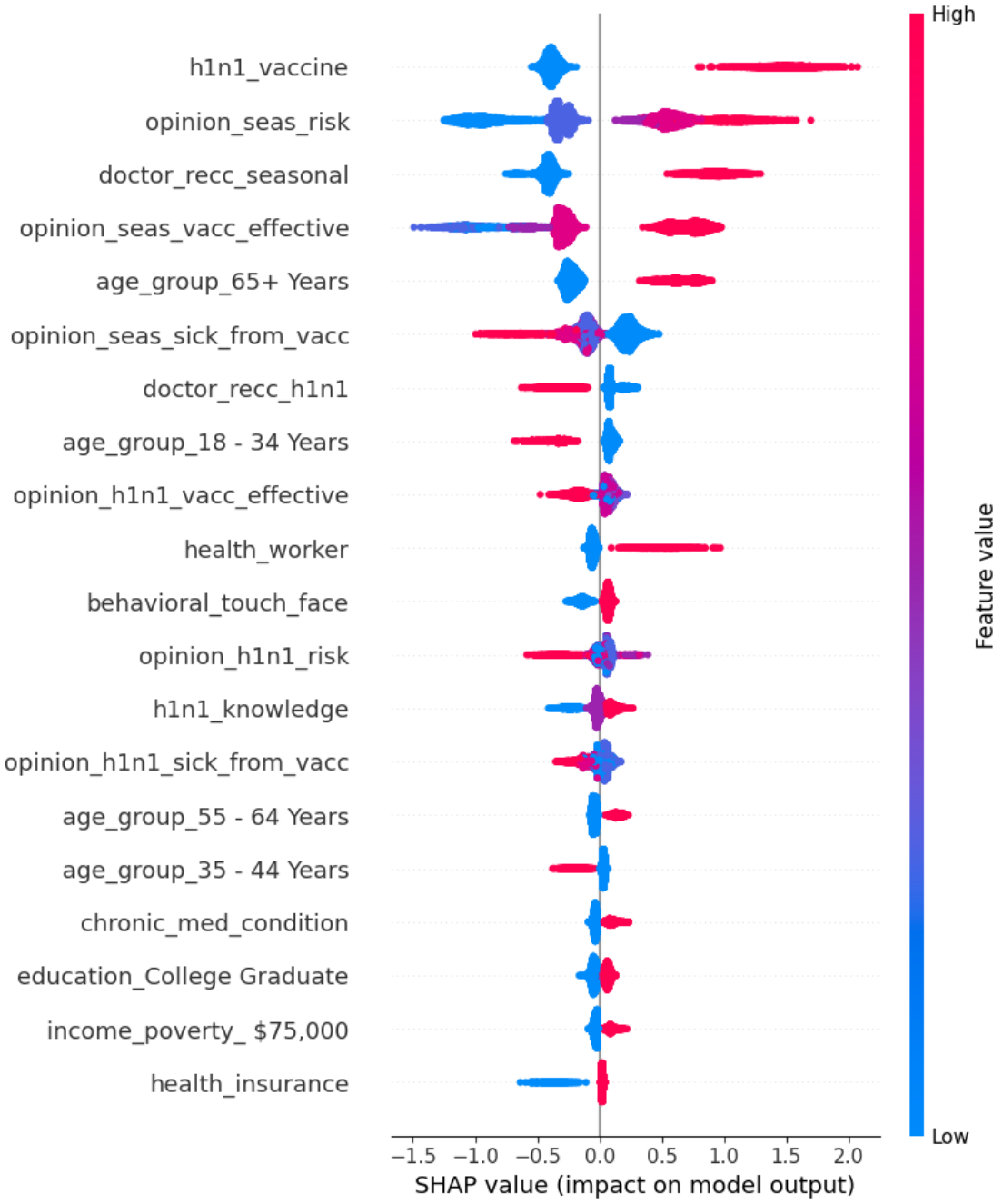
Accuracy Gradient Boosting and XGBoost have the highest accuracy (0.8126 and 0.8125, respectively). The Baseline Logistic model and Random Forest have slightly lower accuracy (0.8081 and 0.8078, respectively). ROC AUC Score XGBoost has the highest ROC AUC score (0.8850), indicating it has the best overall ability to distinguish between the classes. Gradient Boosting is close behind (0.8847). The Baseline Logistic model (0.8775) and Random Forest (0.8782) have slightly lower ROC AUC scores. Precision, Recall, and F1-Score Precision: Both Gradient Boosting and XGBoost models have slightly higher precision for Class 1 (0.80) compared to the baseline and Random Forest models. Recall: XGBoost has the highest recall for Class 0 (0.84) and a good recall for Class 1 (0.78). F1-Score: Both Gradient Boosting and XGBoost have balanced F1-scores for both classes, with XGBoost slightly edging out in performance. Confusion Matrix XGBoost and Gradient Boosting have fewer false positives and false negatives compared to the Baseline Logistic and Random Forest models, indicating a better balance between precision and recall.

XGBoost performs the best overall with the highest ROC AUC score and balanced performance across all metrics. Gradient Boosting is also a strong performer with similar accuracy and slightly lower ROC AUC score. The Baseline Logistic model and Random Forest are good, but not as strong as the other two.

1.12 7.1 Impact of each feature on the model's prediction

[44]:
```
import shap

# Create SHAP explainer object
```

```
explainer = shap.TreeExplainer(best_xgb)  # Assuming best_xgb is your trained␣
 ↪XGBoost model
shap_values = explainer.shap_values(X_test)

# Visualize SHAP summary plot
shap.summary_plot(shap_values, X_test)
```



[45]:
```
# Extract feature contributions for a single instance (e.g., the first instance␣
 ↪in the test set)
instance_index = 0
shap_values_instance = shap_values[instance_index]

# Extract feature contributions for all instances in the test set
shap_values_all = shap_values

# Analyze feature contributions for a single instance
feature_names = X_test.columns  # Assuming X_test is your feature matrix
for i, shap_value in enumerate(shap_values_instance):
    feature_name = feature_names[i]
    print(f"Feature: {feature_name}, SHAP Value: {shap_value}")

# Analyze feature contributions for all instances (aggregate or summarize as␣
 ↪needed)
```

```
Feature: h1n1_concern, SHAP Value: 0.04767724499106407
Feature: h1n1_knowledge, SHAP Value: -0.017361100763082504
Feature: behavioral_antiviral_meds, SHAP Value: 0.0001643907744437456
Feature: behavioral_avoidance, SHAP Value: -0.001978125888854265
Feature: behavioral_face_mask, SHAP Value: 0.004836446605622768
Feature: behavioral_wash_hands, SHAP Value: 0.009754137136042118
Feature: behavioral_large_gatherings, SHAP Value: 0.009427014738321304
Feature: behavioral_outside_home, SHAP Value: -0.002131927292793989
Feature: behavioral_touch_face, SHAP Value: 0.06577429175376892
Feature: doctor_recc_h1n1, SHAP Value: 0.08538646996021271
Feature: doctor_recc_seasonal, SHAP Value: -0.41614165902137756
Feature: chronic_med_condition, SHAP Value: -0.0340990275144577
Feature: child_under_6_months, SHAP Value: -0.0005908544408157468
Feature: health_worker, SHAP Value: -0.08163565397262573
Feature: health_insurance, SHAP Value: 0.020958883687853813
Feature: opinion_h1n1_vacc_effective, SHAP Value: 0.043822139501571655
Feature: opinion_h1n1_risk, SHAP Value: 0.06132790446281433
Feature: opinion_h1n1_sick_from_vacc, SHAP Value: 0.058376505970954895
Feature: opinion_seas_vacc_effective, SHAP Value: -0.3025767207145691
Feature: opinion_seas_risk, SHAP Value: -0.2586129903793335
Feature: opinion_seas_sick_from_vacc, SHAP Value: 0.24643325805664062
Feature: h1n1_vaccine, SHAP Value: -0.38181719183921814
Feature: age_group_18 - 34 Years, SHAP Value: -0.31630465388298035
Feature: age_group_35 - 44 Years, SHAP Value: 0.0240556001663208
Feature: age_group_45 - 54 Years, SHAP Value: 0.001037635374814272
Feature: age_group_55 - 64 Years, SHAP Value: -0.060009829699993134
Feature: age_group_65+ Years, SHAP Value: -0.26624447107315063
Feature: education_12 Years, SHAP Value: 0.0037686373107135296
Feature: education_ 12 Years, SHAP Value: 0.01096810307353735
Feature: education_College Graduate, SHAP Value: 0.06896541267633438
```

```
Feature: education_Some College, SHAP Value: 0.0003223167732357979
Feature: race_Black, SHAP Value: 0.013288723304867744
Feature: race_Hispanic, SHAP Value: 0.0023284712806344032
Feature: race_Other or Multiple, SHAP Value: -0.005447184666991234
Feature: race_White, SHAP Value: 0.008386798202991486
Feature: sex_Female, SHAP Value: 0.002958218101412058
Feature: sex_Male, SHAP Value: 0.005599070340394974
Feature: income_poverty_= $75,000, Above Poverty, SHAP Value:
-0.003708350472152233
Feature: income_poverty_ $75,000, SHAP Value: -0.03751743584871292
Feature: income_poverty_Below Poverty, SHAP Value: 0.012048929929733276
Feature: marital_status_Married, SHAP Value: 0.0034778432454913855
Feature: marital_status_Not Married, SHAP Value: 0.0024580059107393026
Feature: rent_or_own_Own, SHAP Value: 0.030099909752607346
Feature: rent_or_own_Rent, SHAP Value: 0.0029669818468391895
Feature: employment_status_Employed, SHAP Value: -0.009090590290725231
Feature: employment_status_Not in Labor Force, SHAP Value: -0.014593102969229221
Feature: employment_status_Unemployed, SHAP Value: 0.009050801396369934
```

These SHAP values represent the impact of each feature on the model's predictions for a single instance. Positive SHAP values indicate that the feature pushed the model's output higher (towards predicting a positive outcome, i.e., receiving the vaccine), while negative SHAP values indicate the opposite.

Here's an interpretation of some of the features based on their SHAP values:

doctor_recc_h1n1: Individuals recommended by a doctor to take the H1N1 vaccine have a higher likelihood of receiving the seasonal flu vaccine. behavioral_touch_face: People who exhibit more hand-to-face behavior are more likely to take the seasonal flu vaccine. opinion_h1n1_vacc_effective and opinion_h1n1_sick_from_vacc: Positive opinions about the effectiveness and perceived risk of the H1N1 vaccine contribute to a higher likelihood of taking the seasonal flu vaccine. doctor_recc_seasonal: Surprisingly, individuals recommended by a doctor to take the seasonal flu vaccine have a negative impact on their likelihood of actually receiving it. This might indicate that they are less trusting of medical advice or have concerns about the vaccine's safety or effectiveness. opinion_seas_vacc_effective and opinion_seas_risk: Positive opinions about the effectiveness and perceived risk of the seasonal flu vaccine influence the decision to receive it. h1n1_vaccine: Those who did not take the H1N1 vaccine are less likely to take the seasonal flu vaccine. age_group: Older age groups (55+ years) and younger age groups (18-34 years) are less likely to take the seasonal flu vaccine compared to middle-aged adults (35-54 years). education: College graduates are more likely to take the seasonal flu vaccine compared to those with only a high school education. income_poverty: Individuals below the poverty line are more likely to take the seasonal flu vaccine. marital_status: Married individuals are slightly more likely to take the seasonal flu vaccine compared to unmarried individuals. rent_or_own: Those who own their homes are more likely to take the seasonal flu vaccine compared to renters. employment_status: Unemployed individuals are more likely to take the seasonal flu vaccine compared to those who are employed or not in the labor force. These interpretations provide insights into the factors influencing seasonal flu vaccine uptake based on the SHAP values derived from thet model.

1.13 8. Conclusion

1.13.1 8.1 Recommendations

Based on the analysis conducted using various machine learning models and SHAP values, here are some recommendations for stakeholders:

Targeted Public Health Campaigns: Develop targeted public health campaigns aimed at specific demographic groups identified as less likely to take the seasonal flu vaccine. For example, focus on educating younger and older age groups, individuals with lower education levels, and those living below the poverty line about the importance of vaccination.

Doctor Recommendations: Encourage healthcare providers to emphasize the importance of vaccination during routine medical visits, especially for individuals who did not receive the H1N1 vaccine. Address concerns and misconceptions about vaccine safety and effectiveness raised by healthcare providers.

Community Engagement: Foster community engagement initiatives to address vaccine hesitancy and increase trust in vaccines. Collaborate with community leaders, organizations, and influencers to disseminate accurate information and combat misinformation about seasonal flu vaccination.

Education and Awareness: Implement educational programs in schools, workplaces, and community centers to raise awareness about the benefits of vaccination and dispel myths surrounding flu vaccines. Provide resources in multiple languages and formats to ensure accessibility to diverse populations.

Access to Vaccination Services: Improve access to vaccination services, particularly for underserved populations such as those with lower socioeconomic status. This may involve offering free or low-cost vaccination clinics in convenient locations and providing transportation assistance if needed.

Policy Support: Advocate for policies that support vaccination efforts, such as mandating vaccination for certain occupations or providing incentives for vaccination uptake. Collaborate with policymakers to address structural barriers to vaccination and promote equitable access to healthcare services.

Continuous Monitoring and Evaluation: Continuously monitor vaccination rates and assess the effectiveness of interventions using real-time data. Adjust strategies as needed based on feedback and emerging trends to maximize vaccination coverage and public health impact.

By implementing these recommendations, stakeholders can work towards increasing seasonal flu vaccine uptake rates and mitigating the risks associated with vaccine hesitancy, thereby enhancing community immunity and reducing the burden of seasonal flu outbreaks.

1.13.2 8.2 Next Steps

Based on the analysis using various machine learning models, XGBoost emerged as the best performer in predicting seasonal flu vaccine uptake, with the highest ROC AUC score. This indicates that the model is highly effective at distinguishing between individuals who are likely to receive the vaccine and those who are not. Key features influencing vaccine uptake include recommendations from doctors, opinions about vaccine effectiveness and risks, and demographic factors such as age, education, and socioeconomic status. To address vaccine hesitancy, targeted public health campaigns, enhanced doctor-patient communication, and improved access to vaccination services are recommended.

For stakeholders, the next steps involve developing and executing a comprehensive action plan based on these insights. This includes engaging with community leaders, securing funding, and setting up educational programs and vaccination clinics. Continuous monitoring and evaluation will be crucial to assess the effectiveness of these interventions and make necessary adjustments. By implementing these strategies, stakeholders can work towards increasing vaccine uptake, thereby reducing the risks associated with vaccine hesitancy and improving public health outcomes.

1.14 9. Save the Model

[46]:
```
import joblib

# Save the trained XGBoost model
joblib.dump(best_xgb, 'xgboost_model.pkl')
```

[46]:
```
['xgboost_model.pkl']
```

Saving the model makes it possible to load the model later without retraining.