BUSINESS UNDERSTANDING

BUSINESS PROBLEM

The challenge at hand is to optimize public health efforts related to H1N1 vaccination, particularly focusing on predicting individuals' likelihood of receiving the H1N1 flu vaccine.

PROJECT OBJECTIVE

The primary goal of this analysis is to understand the factors influencing H1N1 vaccination uptake. By identifying these factors, we aim to improve vaccination campaigns and target interventions more effectively.

BUSINESS STAKEHOLDERS

1. Public health authorities: seeking to organize focused vaccination campaigns and outreach initiatives, and are particularly interested in learning about vaccination trends, particularly as pertaining to H1N1.

2. Policymakers: aiming to make educated decisions about vaccination policies and programs by applying the knowledge gained from H1N1 vaccine uptake forecasts to resource allocation.

3. Providers of healthcare solutions: Taking advantage of prediction models to pinpoint communities or persons at high risk for H1N1 vaccination promotion and preventive treatment.

BUSINESS GOALS

The project intends to achieve particular public health objectives related to H1N1 prevention and control by concentrating on forecasting H1N1 vaccination uptake.

The specific objectives of this analysis are:

1. To identify significant predictors of vaccination uptake.
2. To explore interaction effects between different predictors.
3. To build a predictive model that can accurately classify individuals based on their likelihood of getting vaccinated.

DATA UNDERSTANDING

DATA LOADING AND INSPECTION

```
import requests
import zipfile
import io
import os
```

```
import pandas as pd
# Define the folder path where your CSV files are located
folder_path = "/content/H1N1"
```

Training Set Features

Less than a quarter of the respondents received the H1N1 vaccine.

```
# Load training set features
train_features_df =
pd.read_csv(f"{folder_path}/training_set_features.csv")

# Display the first five  rows of Training set Features
print("Training Set Features:")
print(train_features_df.head())
```

```
Training Set Features:
   respondent_id  h1n1_concern  h1n1_knowledge
behavioral_antiviral_meds  \
0              0           1.0             0.0
0.0
1              1           3.0             2.0
0.0
2              2           1.0             1.0
0.0
3              3           1.0             1.0
0.0
4              4           2.0             1.0
0.0

   behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
0                   0.0                   0.0                       0.0

1                   1.0                   0.0                       1.0

2                   1.0                   0.0                       0.0

3                   1.0                   0.0                       1.0

4                   1.0                   0.0                       1.0


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

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

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

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

   employment_occupation
0                    NaN
1                xgwztkwe
2                xtkaffoo
3                    NaN
4                emcorrxb

[5 rows x 36 columns]
```

The preview shows various columns such as respondent_id, h1n1_concern, h1n1_knowledge and multiple behavioral indicators for example behavioral_antiviral_meds, behavioral_avoidance, behavioral_face_mask along with demographic information like age_group, education, race, sex.

```
#Checking the columns in training features dataset
print("Training Set Features Columns:",
train_features_df.columns.tolist())
```

```
Training Set Features Columns: ['respondent_id', 'h1n1_concern',
'h1n1_knowledge', 'behavioral_antiviral_meds', 'behavioral_avoidance',
'behavioral_face_mask', 'behavioral_wash_hands',
'behavioral_large_gatherings', 'behavioral_outside_home',
'behavioral_touch_face', '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', 'census_msa',
'household_adults', 'household_children', 'employment_industry',
'employment_occupation']

#Checking the shape of training features dataset
print("Training Set Features Shape:", train_features_df.shape)

Training Set Features Shape: (26707, 36)

#Display information about the training set features dataset
print("Training Set Features info:")
print(train_features_df.info())


Training Set Features info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 36 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   respondent_id                26707 non-null  int64
 1   h1n1_concern                 26615 non-null  float64
 2   h1n1_knowledge               26591 non-null  float64
 3   behavioral_antiviral_meds    26636 non-null  float64
 4   behavioral_avoidance         26499 non-null  float64
 5   behavioral_face_mask         26688 non-null  float64
 6   behavioral_wash_hands        26665 non-null  float64
 7   behavioral_large_gatherings  26620 non-null  float64
 8   behavioral_outside_home      26625 non-null  float64
 9   behavioral_touch_face        26579 non-null  float64
 10  doctor_recc_h1n1             24547 non-null  float64
 11  doctor_recc_seasonal         24547 non-null  float64
 12  chronic_med_condition        25736 non-null  float64
 13  child_under_6_months         25887 non-null  float64
 14  health_worker                25903 non-null  float64
 15  health_insurance             14433 non-null  float64
 16  opinion_h1n1_vacc_effective  26316 non-null  float64
 17  opinion_h1n1_risk            26319 non-null  float64
 18  opinion_h1n1_sick_from_vacc  26312 non-null  float64
```

```
 19  opinion_seas_vacc_effective  26245 non-null  float64
 20  opinion_seas_risk            26193 non-null  float64
 21  opinion_seas_sick_from_vacc  26170 non-null  float64
 22  age_group                    26707 non-null  object
 23  education                    25300 non-null  object
 24  race                         26707 non-null  object
 25  sex                          26707 non-null  object
 26  income_poverty               22284 non-null  object
 27  marital_status               25299 non-null  object
 28  rent_or_own                  24665 non-null  object
 29  employment_status            25244 non-null  object
 30  hhs_geo_region               26707 non-null  object
 31  census_msa                   26707 non-null  object
 32  household_adults             26458 non-null  float64
 33  household_children           26458 non-null  float64
 34  employment_industry          13377 non-null  object
 35  employment_occupation        13237 non-null  object
dtypes: float64(23), int64(1), object(12)
memory usage: 7.3+ MB
None
```

The output shows that the DataFrame has 26,707 entries and 36 columns. Some columns contain missing values

```
#summary for the numerical columns on training set features dataset
print("Training Set Features numerical columns:")
print(train_features_df.describe())
```

```
Training Set Features numerical columns:
       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  ...  health_worker  health_insurance  \
count           26579.000000  ...   25903.000000       14433.00000
mean                0.677264  ...       0.111918           0.87972
std                 0.467531  ...       0.315271           0.32530
min                 0.000000  ...       0.000000           0.00000
25%                 0.000000  ...       0.000000           1.00000
50%                 1.000000  ...       0.000000           1.00000
75%                 1.000000  ...       0.000000           1.00000
max                 1.000000  ...       1.000000           1.00000

       opinion_h1n1_vacc_effective  opinion_h1n1_risk  \
count                  26316.000000       26319.000000
mean                       3.850623           2.342566
std                        1.007436           1.285539
min                        1.000000           1.000000
25%                        3.000000           1.000000
50%                        4.000000           2.000000
75%                        5.000000           4.000000
max                        5.000000           5.000000

       opinion_h1n1_sick_from_vacc  opinion_seas_vacc_effective  \
count                  26312.000000                 26245.000000
```

```
mean                         2.357670                         4.025986
std                          1.362766                         1.086565
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
household_adults  \
count       26193.000000                 26170.000000
26458.000000
mean            2.719162                     2.118112
0.886499
std             1.385055                     1.332950
0.753422
min             1.000000                     1.000000
0.000000
25%             2.000000                     1.000000
0.000000
50%             2.000000                     2.000000
1.000000
75%             4.000000                     4.000000
1.000000
max             5.000000                     5.000000
3.000000

       household_children
count        26458.000000
mean             0.534583
std              0.928173
min              0.000000
25%              0.000000
50%              0.000000
75%              1.000000
max              3.000000

[8 rows x 24 columns]
```

h1n1_concern has a mean of 1.62 and a maximum value of 3. behavioral_antiviral_meds is mostly 0, indicating low usage of antiviral medications. opinion_h1n1_vacc_effective has a mean of 3.85, suggesting generally positive opinions about the effectiveness of the H1N1 vaccine.

Training Set Labels

```
# Load training set labels
train_labels_df =
pd.read_csv(f"{folder_path}/training_set_labels.csv")

# Display the first five rows of Training set labels
```

```
print("\nTraining Set Labels:")
print(train_labels_df.head())
```

```
Training Set Labels:
   respondent_id  h1n1_vaccine  seasonal_vaccine
0              0             0                 0
1              1             0                 1
2              2             0                 0
3              3             0                 1
4              4             0                 0
```

The preview shows the respondent_id, h1n1_vaccine, and seasonal_vaccine columns for the first five entries. For instance:Respondent 0 did not receive either the H1N1 or seasonal vaccine. Respondent 1 did not receive the H1N1 vaccine but received the seasonal vaccine.

```
#Display information about the training set labels dataset
print("Training Set labels info:")
print(train_labels_df.info())
```

```
Training Set labels info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26707 entries, 0 to 26706
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   respondent_id    26707 non-null  int64
 1   h1n1_vaccine     26707 non-null  int64
 2   seasonal_vaccine  26707 non-null  int64
dtypes: int64(3)
memory usage: 626.1 KB
None
```

The output shows that the DataFrame has 26,707 entries and 3 columns. All columns are non-null integers:

respondent_id is a unique identifier for each respondent.

h1n1_vaccine indicates if the respondent received the H1N1 vaccine (1 for yes, 0 for no).

seasonal_vaccine indicates if the respondent received the seasonal flu vaccine (1 for yes, 0 for no).

```
#summary for the numerical columns on training set labels dataset
print("Training Set Labels numerical columns:")
print(train_labels_df.describe())
```

```
Training Set Labels numerical columns:
       respondent_id  h1n1_vaccine  seasonal_vaccine
count   26707.000000  26707.000000      26707.000000
```

```
mean    13353.000000      0.212454           0.465608
std      7709.791156      0.409052           0.498825
min         0.000000      0.000000           0.000000
25%      6676.500000      0.000000           0.000000
50%     13353.000000      0.000000           0.000000
75%     20029.500000      0.000000           1.000000
max     26706.000000      1.000000           1.000000
```

The h1n1_vaccine column has a mean of 0.21, indicating that approximately 21% of respondents received the H1N1 vaccine.

The seasonal_vaccine column has a mean of 0.47, indicating that approximately 47% of respondents received the seasonal flu vaccine.

Test Set Features

```
# Load test set features
test_features_df = pd.read_csv(f"{folder_path}/test_set_features.csv")
# Display the first five rows of Test set features
print("\nTest Set Features:")
print(test_features_df.head())
```

```
Test Set Features:
   respondent_id  h1n1_concern  h1n1_knowledge
behavioral_antiviral_meds  \
0          26707           2.0             2.0
0.0
1          26708           1.0             1.0
0.0
2          26709           2.0             2.0
0.0
3          26710           1.0             1.0
0.0
4          26711           3.0             1.0
1.0

   behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
0                   1.0                   0.0                       1.0

1                   0.0                   0.0                       0.0

2                   0.0                   1.0                       1.0

3                   0.0                   0.0                       0.0

4                   1.0                   0.0                       1.0
```

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

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

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

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

   employment_occupation
0               hfxkjkmi
1               xqwwgdyp
2               pvmttkik
3                    NaN
4               mxkfnird

[5 rows x 36 columns]
```

```python
#Display information about the test set features dataset
print("Test Set Features info:")
print(test_features_df.info())
```

```
Test Set Features info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26708 entries, 0 to 26707
Data columns (total 36 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   respondent_id                26708 non-null  int64  
 1   h1n1_concern                 26623 non-null  float64
 2   h1n1_knowledge               26586 non-null  float64
 3   behavioral_antiviral_meds    26629 non-null  float64
 4   behavioral_avoidance         26495 non-null  float64
 5   behavioral_face_mask         26689 non-null  float64
 6   behavioral_wash_hands        26668 non-null  float64
 7   behavioral_large_gatherings  26636 non-null  float64
 8   behavioral_outside_home      26626 non-null  float64
 9   behavioral_touch_face        26580 non-null  float64
 10  doctor_recc_h1n1             24548 non-null  float64
 11  doctor_recc_seasonal         24548 non-null  float64
 12  chronic_med_condition        25776 non-null  float64
 13  child_under_6_months         25895 non-null  float64
 14  health_worker                25919 non-null  float64
 15  health_insurance             14480 non-null  float64
 16  opinion_h1n1_vacc_effective  26310 non-null  float64
 17  opinion_h1n1_risk            26328 non-null  float64
 18  opinion_h1n1_sick_from_vacc  26333 non-null  float64
 19  opinion_seas_vacc_effective  26256 non-null  float64
 20  opinion_seas_risk            26209 non-null  float64
 21  opinion_seas_sick_from_vacc  26187 non-null  float64
 22  age_group                    26708 non-null  object 
 23  education                    25301 non-null  object 
 24  race                         26708 non-null  object 
 25  sex                          26708 non-null  object 
 26  income_poverty               22211 non-null  object 
 27  marital_status               25266 non-null  object 
 28  rent_or_own                  24672 non-null  object 
 29  employment_status            25237 non-null  object 
 30  hhs_geo_region               26708 non-null  object 
 31  census_msa                   26708 non-null  object 
 32  household_adults             26483 non-null  float64
 33  household_children           26483 non-null  float64
 34  employment_industry          13433 non-null  object 
 35  employment_occupation        13282 non-null  object 
dtypes: float64(23), int64(1), object(12)
memory usage: 7.3+ MB
None
```

```
#summary for the numerical columns on test set features dataset
print("Test Set Features numerical columns:")
print(test_features_df.describe())
```

```
Test Set Features numerical columns:
       respondent_id  h1n1_concern  h1n1_knowledge
behavioral_antiviral_meds  \
count   26708.000000  26623.000000    26586.000000
26629.000000
mean    40060.500000      1.623145        1.266042
0.049645
std      7710.079831      0.902755        0.615617
0.217215
min     26707.000000      0.000000        0.000000
0.000000
25%     33383.750000      1.000000        1.000000
0.000000
50%     40060.500000      2.000000        1.000000
0.000000
75%     46737.250000      2.000000        2.000000
0.000000
max     53414.000000      3.000000        2.000000
1.000000

       behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
count          26495.000000          26689.000000
26668.000000
mean               0.729798              0.069279
0.826084
std                0.444072              0.253934
0.379045
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                 26636.000000             26626.000000
mean                      0.351517                 0.337227
std                       0.477453                 0.472772
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  ...  health_worker  health_insurance  \
count           26580.000000  ...   25919.000000      14480.000000
mean                0.683747  ...       0.111501          0.887914
std                 0.465022  ...       0.314758          0.315483
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

       opinion_h1n1_vacc_effective  opinion_h1n1_risk  \
count                 26310.000000       26328.000000
mean                      3.844622           2.326838
std                       1.007570           1.275636
min                       1.000000           1.000000
25%                       3.000000           1.000000
50%                       4.000000           2.000000
75%                       5.000000           4.000000
max                       5.000000           5.000000

       opinion_h1n1_sick_from_vacc  opinion_seas_vacc_effective  \
count                 26333.000000                 26256.000000
mean                      2.360612                     4.024832
std                       1.359413                     1.083204
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
household_adults  \
count       26209.000000                 26187.000000
26483.000000
mean            2.708688                     2.143392
0.894310
std             1.376045                     1.339102
0.754244
min             1.000000                     1.000000
0.000000
25%             2.000000                     1.000000
0.000000
50%             2.000000                     2.000000
1.000000
75%             4.000000                     4.000000
1.000000
max             5.000000                     5.000000
```

```
3.000000

       household_children
count        26483.000000
mean             0.543745
std              0.935057
min              0.000000
25%              0.000000
50%              0.000000
75%              1.000000
max              3.000000

[8 rows x 24 columns]
```

DATA PREPARATION

DATA CLEANING

Checking For Missing Values

```
print("Missing Values in Training Set Features:")
print(train_features_df.isnull().sum())
```

```
Missing Values in Training Set Features:
respondent_id                    0
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
census_msa                      0
household_adults              249
household_children            249
employment_industry         13330
employment_occupation       13470
dtype: int64
```

```
print("Missing Values in Training Set Labels:")
print(train_labels_df.isnull().sum())
```

```
Missing Values in Training Set Labels:
respondent_id       0
h1n1_vaccine        0
seasonal_vaccine    0
dtype: int64
```

```
print("Missing Values in Test Set Features:")
print(test_features_df.isnull().sum())
```

```
Missing Values in Test Set Features:
respondent_id                     0
h1n1_concern                     85
h1n1_knowledge                  122
behavioral_antiviral_meds        79
behavioral_avoidance            213
behavioral_face_mask             19
behavioral_wash_hands            40
behavioral_large_gatherings      72
behavioral_outside_home          82
behavioral_touch_face           128
doctor_recc_h1n1               2160
doctor_recc_seasonal           2160
chronic_med_condition           932
child_under_6_months            813
health_worker                   789
health_insurance              12228
opinion_h1n1_vacc_effective     398
opinion_h1n1_risk               380
opinion_h1n1_sick_from_vacc     375
opinion_seas_vacc_effective     452
opinion_seas_risk               499
opinion_seas_sick_from_vacc     521
age_group                         0
```

```
education                    1407
race                            0
sex                             0
income_poverty               4497
marital_status               1442
rent_or_own                  2036
employment_status            1471
hhs_geo_region                  0
census_msa                      0
household_adults              225
household_children            225
employment_industry         13275
employment_occupation       13426
dtype: int64
```

I noticed that the columns; 'employment_occupation','employment_industry' and 'hhs_geo_region' have coded variables

```
# Check unique values and their frequencies in the
'employment_occupation' column
print("Unique values and frequencies in 'employment_occupation':")
print(train_features_df['employment_occupation'].value_counts())

# Check unique values and their frequencies in the
'employment_industry' column
print("Unique values and frequencies in 'employment_industry':")
print(train_features_df['employment_industry'].value_counts())

# Check unique values and their frequencies in the 'hhs_geo_region'
column
print("Unique values and frequencies in 'hhs_geo_region':")
print(train_features_df['hhs_geo_region'].value_counts())
```

```
Unique values and frequencies in 'employment_occupation':
employment_occupation
xtkaffoo    1778
mxkfnird    1509
emcorrxb    1270
cmhcxjea    1247
xgwztkwe    1082
hfxkjkmi     766
qxajmpny     548
xqwwgdyp     485
kldqjyjy     469
uqqtjvyb     452
tfqavkke     388
ukymxvdu     372
vlluhbov     354
oijqvulv     344
```

```
ccgxvspp     341
bxpfxfdn     331
haliazsg     296
rcertsgn     276
xzmlyyjv     248
dlvbwzss     227
hodpvpew     208
dcjcmpih     148
pvmttkik      98
Name: count, dtype: int64
Unique values and frequencies in 'employment_industry':
employment_industry
fcxhlnwr    2468
wxleyezf    1804
ldnlellj    1231
pxcmvdjn    1037
atmlpfrs     926
arjwrbjb     871
xicduogh     851
mfikgejo     614
vjjrobsf     527
rucpziij     523
xqicxuve     511
saaquncn     338
cfqqtusy     325
nduyfdeo     286
mcubkhph     275
wlfvacwt     215
dotnnunm     201
haxffmxo     148
msuufmds     124
phxvnwax      89
qnlwzans      13
Name: count, dtype: int64
Unique values and frequencies in 'hhs_geo_region':
hhs_geo_region
lzgpxyit    4297
fpwskwrf    3265
qufhixun    3102
oxchjgsf    2859
kbazzjca    2858
bhuqouqj    2846
mlyzmhmf    2243
lrircsnp    2078
atmpeygn    2033
dqpwygqj    1126
Name: count, dtype: int64
```

I decided to replace the coded variables with labels.

```
# Create mapping dictionaries

# Mapping for 'employment_occupation'
occupation_mapping = {
    'xtkaffoo': 'Occupation1',
    'mxkfnird': 'Occupation2',
    'emcorrxb': 'Occupation3',
    'cmhcxjea': 'Occupation4',
    'xgwztkwe': 'Occupation5',
    'hfxkjkmi': 'Occupation6',
    'qxajmpny': 'Occupation7',
    'xqwwgdyp': 'Occupation8',
    'kldqjyjy': 'Occupation9',
    'uqqtjvyb': 'Occupation10',
    'tfqavkke': 'Occupation11',
    'ukymxvdu': 'Occupation12',
    'vlluhbov': 'Occupation13',
    'oijqvulv': 'Occupation14',
    'ccgxvspp': 'Occupation15',
    'bxpfxfdn': 'Occupation16',
    'haliazsg': 'Occupation17',
    'rcertsgn': 'Occupation18',
    'xzmlyyjv': 'Occupation19',
    'dlvbwzss': 'Occupation20',
    'hodpvpew': 'Occupation21',
    'dcjcmpih': 'Occupation22',
    'pvmttkik': 'Occupation23',

}

# Mapping for 'employment_industry'
industry_mapping = {
    'fcxhlnwr': 'Industry1',
    'wxleyezf': 'Industry2',
    'ldnlellj': 'Industry3',
    'pxcmvdjn': 'Industry4',
    'atmlpfrs': 'Industry5',
    'arjwrbjb': 'Industry6',
    'xicduogh': 'Industry7',
    'mfikgejo': 'Industry8',
    'vjjrobsf': 'Industry9',
    'rucpziij': 'Industry10',
    'xqicxuve': 'Industry11',
    'saaquncn': 'Industry12',
    'cfqqtusy': 'Industry13',
    'nduyfdeo': 'Industry14',
    'mcubkhph': 'Industry15',
    'wlfvacwt': 'Industry16',
    'dotnnunm': 'Industry17',
    'haxffmxo': 'Industry18',
```

```
    'msuufmds': 'Industry19',
    'phxvnwax': 'Industry20',
    'qnlwzans': 'Industry21',

}

# Mapping for 'hhs_geo_region'
region_mapping = {
    'lzgpxyit': 'Region1',
    'fpwskwrf': 'Region2',
    'qufhixun': 'Region3',
    'oxchjgsf': 'Region4',
    'kbazzjca': 'Region5',
    'bhuqouqj': 'Region6',
    'mlyzmhmf': 'Region7',
    'lrircsnp': 'Region8',
    'atmpeygn': 'Region9',
    'dqpwygqj': 'Region10',

}

# Replace coded values with labels using the mapping dictionaries


train_features_df['employment_occupation'] =
train_features_df['employment_occupation'].map(occupation_mapping)
train_features_df['employment_industry'] =
train_features_df['employment_industry'].map(industry_mapping)
train_features_df['hhs_geo_region'] =
train_features_df['hhs_geo_region'].map(region_mapping)


print(train_features_df['employment_occupation'].value_counts())
print(train_features_df['employment_industry'].value_counts())
print(train_features_df['hhs_geo_region'].value_counts())
```

```
employment_occupation
Occupation1     1778
Occupation2     1509
Occupation3     1270
Occupation4     1247
Occupation5     1082
Occupation6      766
Occupation7      548
Occupation8      485
Occupation9      469
Occupation10     452
Occupation11     388
Occupation12     372
Occupation13     354
```

```
Occupation14     344
Occupation15     341
Occupation16     331
Occupation17     296
Occupation18     276
Occupation19     248
Occupation20     227
Occupation21     208
Occupation22     148
Occupation23      98
Name: count, dtype: int64
employment_industry
Industry1     2468
Industry2     1804
Industry3     1231
Industry4     1037
Industry5      926
Industry6      871
Industry7      851
Industry8      614
Industry9      527
Industry10     523
Industry11     511
Industry12     338
Industry13     325
Industry14     286
Industry15     275
Industry16     215
Industry17     201
Industry18     148
Industry19     124
Industry20      89
Industry21      13
Name: count, dtype: int64
hhs_geo_region
Region1     4297
Region2     3265
Region3     3102
Region4     2859
Region5     2858
Region6     2846
Region7     2243
Region8     2078
Region9     2033
Region10    1126
Name: count, dtype: int64
```

Merge Training set features and Training Labels

```python
# Merge training features and labels on 'respondent_id'
train_data = pd.merge(train_features_df, train_labels_df,
on='respondent_id')

# Verify the merge
print("Merged Training Data Head:")
print(train_data.head())
print("Merged Training Data Columns:")
print(train_data.columns)
```

```
Merged Training Data Head:
   respondent_id  h1n1_concern  h1n1_knowledge
behavioral_antiviral_meds  \
0              0           1.0             0.0
0.0
1              1           3.0             2.0
0.0
2              2           1.0             1.0
0.0
3              3           1.0             1.0
0.0
4              4           2.0             1.0
0.0

   behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
0                   0.0                   0.0                    0.0

1                   1.0                   0.0                    1.0

2                   1.0                   0.0                    0.0

3                   1.0                   0.0                    1.0

4                   1.0                   0.0                    1.0


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

   behavioral_touch_face  ...  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
```

```
   hhs_geo_region                census_msa  household_adults  \
0         Region4                   Non-MSA               0.0
1         Region6  MSA, Not Principle  City               0.0
2         Region3  MSA, Not Principle  City               2.0
3         Region8       MSA, Principle City               0.0
4         Region3  MSA, Not Principle  City               1.0

   household_children  employment_industry  employment_occupation  \
0                 0.0                  NaN                    NaN
1                 0.0            Industry4            Occupation5
2                 0.0           Industry10            Occupation1
3                 0.0                  NaN                    NaN
4                 0.0            Industry2            Occupation3

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

[5 rows x 38 columns]
Merged Training Data Columns:
Index(['respondent_id', 'h1n1_concern', 'h1n1_knowledge',
       'behavioral_antiviral_meds', 'behavioral_avoidance',
       'behavioral_face_mask', 'behavioral_wash_hands',
       'behavioral_large_gatherings', 'behavioral_outside_home',
       'behavioral_touch_face', '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',
'census_msa',
       'household_adults', 'household_children',
'employment_industry',
       'employment_occupation', 'h1n1_vaccine', 'seasonal_vaccine'],
      dtype='object')

# Separate features and target
X_train = train_data.drop(columns=['respondent_id', 'h1n1_vaccine',
'seasonal_vaccine'])
y_train = train_data['h1n1_vaccine']

# Verify the structure of X_train
```

```
print("X_train Head:")
print(X_train.head())
print("X_train Columns:")
print(X_train.columns)
```

```
X_train 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  ...
income_poverty  \
0                    1.0               0.0  ...              Below
Poverty
1                    1.0               0.0  ...              Below
Poverty
2                    0.0               NaN  ...  <= $75,000, Above
Poverty
3                    0.0               0.0  ...              Below
Poverty
4                    1.0               0.0  ...  <= $75,000, Above
Poverty

   marital_status  rent_or_own   employment_status  hhs_geo_region  \
0     Not Married          Own  Not in Labor Force         Region4
1     Not Married         Rent            Employed         Region6
2     Not Married          Own            Employed         Region3
```

```
3     Not Married         Rent  Not in Labor Force         Region8
4         Married          Own            Employed         Region3

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

   employment_industry  employment_occupation
0                  NaN                    NaN
1            Industry4            Occupation5
2           Industry10            Occupation1
3                  NaN                    NaN
4            Industry2            Occupation3

[5 rows x 35 columns]
X_train Columns:
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',
'census_msa',
       'household_adults', 'household_children',
'employment_industry',
       'employment_occupation'],
      dtype='object')
```

DATA PRE-PROCESSING AND FEATURE SCALING

```
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Identify numeric and categorical columns
numeric_features = X_train.select_dtypes(include=['int64',
```

```
'float64']).columns.tolist()
categorical_features =
X_train.select_dtypes(include=['object']).columns.tolist()

print("Numeric Features:")
print(numeric_features)
print("Categorical Features:")
print(categorical_features)

# Preprocessing for numeric data
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])

# Preprocessing for categorical data
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])

# Combine preprocessing steps
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])

# Fit and transform the training data
try:
    X_train_preprocessed = preprocessor.fit_transform(X_train)
    print("Preprocessing Successful")
except Exception as e:
    print("Preprocessing Failed")
    print(e)
```

```
Numeric Features:
['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',
'household_adults', 'household_children']
Categorical Features:
['age_group', 'education', 'race', 'sex', 'income_poverty',
'marital_status', 'rent_or_own', 'employment_status',
'hhs_geo_region', 'census_msa', 'employment_industry',
'employment_occupation']
Preprocessing Successful
```

```
# Convert preprocessed data back to DataFrame
X_train_preprocessed_df = pd.DataFrame(X_train_preprocessed,
                                       columns=numeric_features +
preprocessor.named_transformers_['cat'].named_steps['onehot'].get_feat
ure_names_out(categorical_features).tolist())
```

Now i want to apply the same transformations to my test data

```
# Drop the 'respondent_id' column from test features
X_test = test_features_df.drop(columns=['respondent_id'])

# Verify the structure of X_test
print("X_test Head:")
print(X_test.head())
print("X_test Columns:")
print(X_test.columns)
```

```
X_test Head:
   h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
0           2.0             2.0                        0.0
1           1.0             1.0                        0.0
2           2.0             2.0                        0.0
3           1.0             1.0                        0.0
4           3.0             1.0                        1.0

   behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
0                   1.0                   0.0                       1.0

1                   0.0                   0.0                       0.0

2                   0.0                   1.0                       1.0

3                   0.0                   0.0                       0.0

4                   1.0                   0.0                       1.0


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

   behavioral_touch_face  doctor_recc_h1n1  ...
income_poverty  \
0                    1.0               0.0  ...                    >
$75,000
1                    0.0               0.0  ...                Below
```

```
Poverty
2                    1.0                0.0  ...                    >
$75,000
3                    0.0                1.0  ...  <= $75,000, Above
Poverty
4                    1.0                0.0  ...  <= $75,000, Above
Poverty

   marital_status  rent_or_own   employment_status  hhs_geo_region  \
0     Not Married         Rent            Employed        mlyzmhmf
1     Not Married         Rent            Employed        bhuqouqj
2         Married          Own            Employed        lrircsnp
3         Married          Own  Not in Labor Force        lrircsnp
4     Not Married          Own            Employed        lzgpxyit

                  census_msa  household_adults  household_children  \
0  MSA, Not Principle  City               1.0                 0.0
1                   Non-MSA               3.0                 0.0
2                   Non-MSA               1.0                 0.0
3  MSA, Not Principle  City               1.0                 0.0
4                   Non-MSA               0.0                 1.0

   employment_industry  employment_occupation
0             atmlpfrs               hfxkjkmi
1             atmlpfrs               xqwwgdyp
2             nduyfdeo               pvmttkik
3                  NaN                    NaN
4             fcxhlnwr               mxkfnird

[5 rows x 35 columns]
X_test Columns:
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',
'census_msa',
       'household_adults', 'household_children',
'employment_industry',
```

```
       'employment_occupation'],
      dtype='object')

# Transform the test data
X_test_preprocessed = preprocessor.transform(X_test)

# Verify the transformed test data
print("Transformed X_test Shape:")
print(X_test_preprocessed.shape)

Transformed X_test Shape:
(26708, 105)

# Convert preprocessed data back to DataFrame
X_test_preprocessed_df = pd.DataFrame(X_test_preprocessed,
                                      columns=numeric_features +
preprocessor.named_transformers_['cat'].named_steps['onehot'].get_feat
ure_names_out(categorical_features).tolist())

# Verify the structure of the preprocessed test DataFrame
print("X_test_preprocessed_df Head:")
print(X_test_preprocessed_df.head())
print("X_test_preprocessed_df Columns:")
print(X_test_preprocessed_df.columns)

X_test_preprocessed_df Head:
   h1n1_concern  h1n1_knowledge  behavioral_antiviral_meds  \
0      0.418262        1.197027                  -0.226293
1     -0.681849       -0.423626                  -0.226293
2      0.418262        1.197027                  -0.226293
3     -0.681849       -0.423626                  -0.226293
4      1.518373       -0.423626                   4.419056

   behavioral_avoidance  behavioral_face_mask
behavioral_wash_hands  \
0              0.611637             -0.272097               0.459149

1             -1.634957             -0.272097              -2.177944

2             -1.634957              3.675158               0.459149

3             -1.634957             -0.272097              -2.177944

4              0.611637             -0.272097               0.459149


   behavioral_large_gatherings  behavioral_outside_home  \
0                      1.34068                -0.711798
1                     -0.74589                -0.711798
2                      1.34068                 1.404892
3                     -0.74589                -0.711798
```

```
4                       1.34068                   1.404892

   behavioral_touch_face  doctor_recc_h1n1  ...  \
0               0.687870         -0.503893  ...
1              -1.453764         -0.503893  ...
2               0.687870         -0.503893  ...
3              -1.453764          1.984546  ...
4               0.687870         -0.503893  ...

   employment_occupation_Occupation21
employment_occupation_Occupation22  \
0                                  0.0
0.0
1                                  0.0
0.0
2                                  0.0
0.0
3                                  0.0
0.0
4                                  0.0
0.0

   employment_occupation_Occupation23
employment_occupation_Occupation3  \
0                                  0.0
0.0
1                                  0.0
0.0
2                                  0.0
0.0
3                                  0.0
0.0
4                                  0.0
0.0

   employment_occupation_Occupation4
employment_occupation_Occupation5  \
0                                 0.0
0.0
1                                 0.0
0.0
2                                 0.0
0.0
3                                 0.0
0.0
4                                 0.0
0.0

   employment_occupation_Occupation6
employment_occupation_Occupation7  \
```

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

   employment_occupation_Occupation8
employment_occupation_Occupation9
0                                     0.0
0.0
1                                     0.0
0.0
2                                     0.0
0.0
3                                     0.0
0.0
4                                     0.0
0.0

[5 rows x 105 columns]
X_test_preprocessed_df Columns:
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',
       ...
       'employment_occupation_Occupation21',
       'employment_occupation_Occupation22',
       'employment_occupation_Occupation23',
       'employment_occupation_Occupation3',
       'employment_occupation_Occupation4',
       'employment_occupation_Occupation5',
       'employment_occupation_Occupation6',
       'employment_occupation_Occupation7',
       'employment_occupation_Occupation8',
       'employment_occupation_Occupation9'],
      dtype='object', length=105)
```

Checking for Missing Values

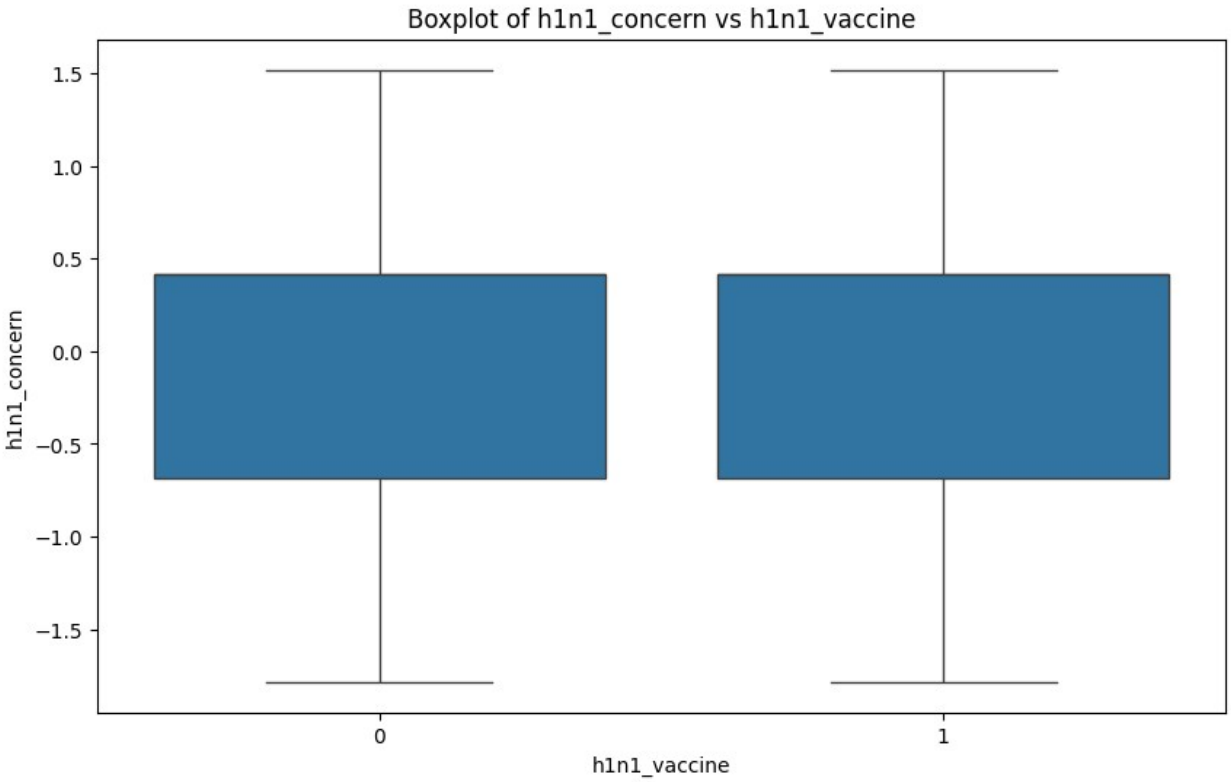
```
print(X_train_preprocessed_df.isnull().sum())
print(X_test_preprocessed_df.isnull().sum())
```

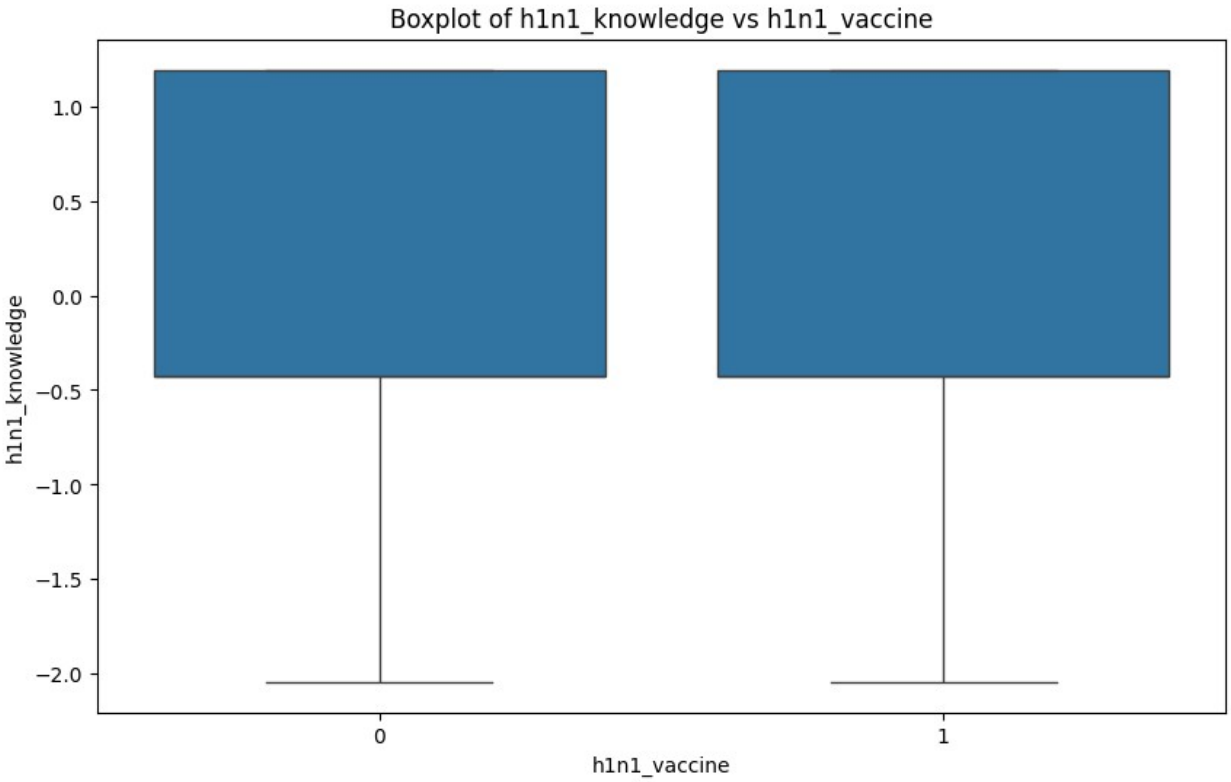
```
h1n1_concern                            0
h1n1_knowledge                          0
behavioral_antiviral_meds               0
behavioral_avoidance                    0
behavioral_face_mask                    0
                                       ..
employment_occupation_Occupation5       0
employment_occupation_Occupation6       0
employment_occupation_Occupation7       0
employment_occupation_Occupation8       0
employment_occupation_Occupation9       0
Length: 105, dtype: int64
h1n1_concern                            0
h1n1_knowledge                          0
behavioral_antiviral_meds               0
behavioral_avoidance                    0
behavioral_face_mask                    0
                                       ..
employment_occupation_Occupation5       0
employment_occupation_Occupation6       0
employment_occupation_Occupation7       0
employment_occupation_Occupation8       0
employment_occupation_Occupation9       0
Length: 105, dtype: int64
```

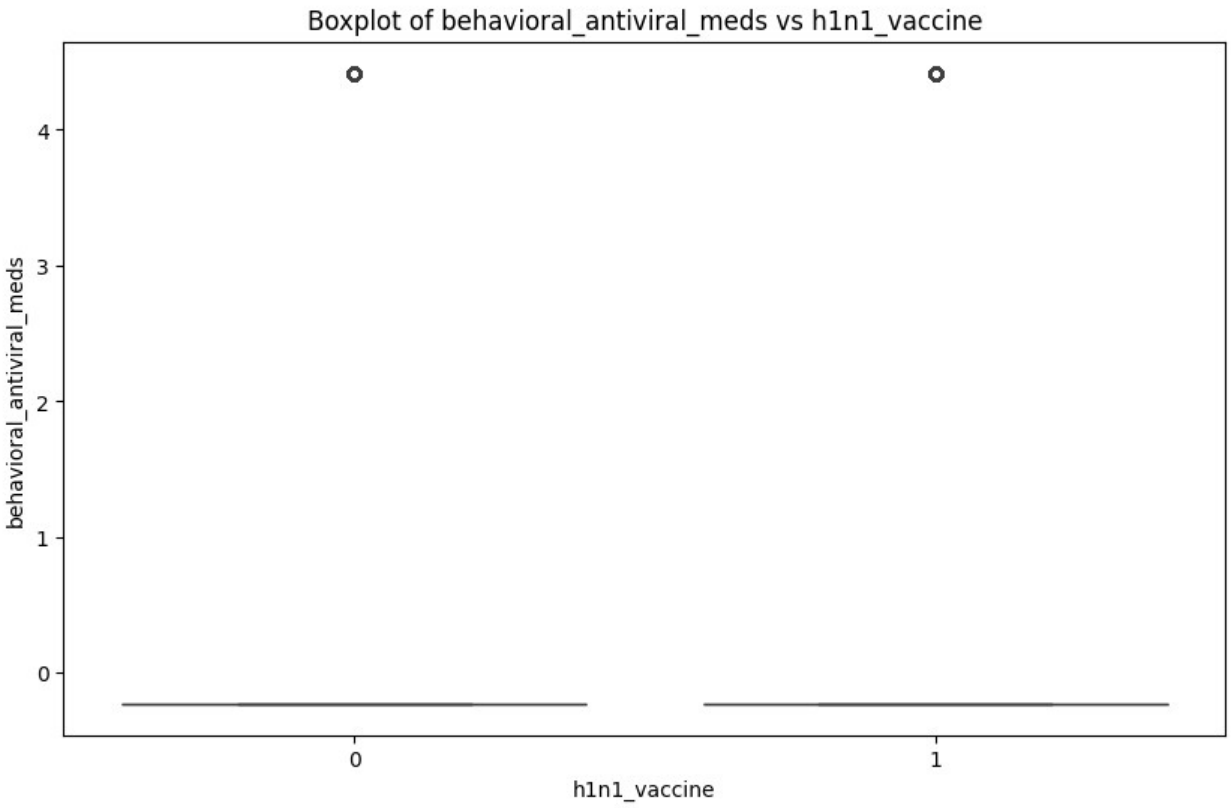
BIVARIATE ANALYSIS

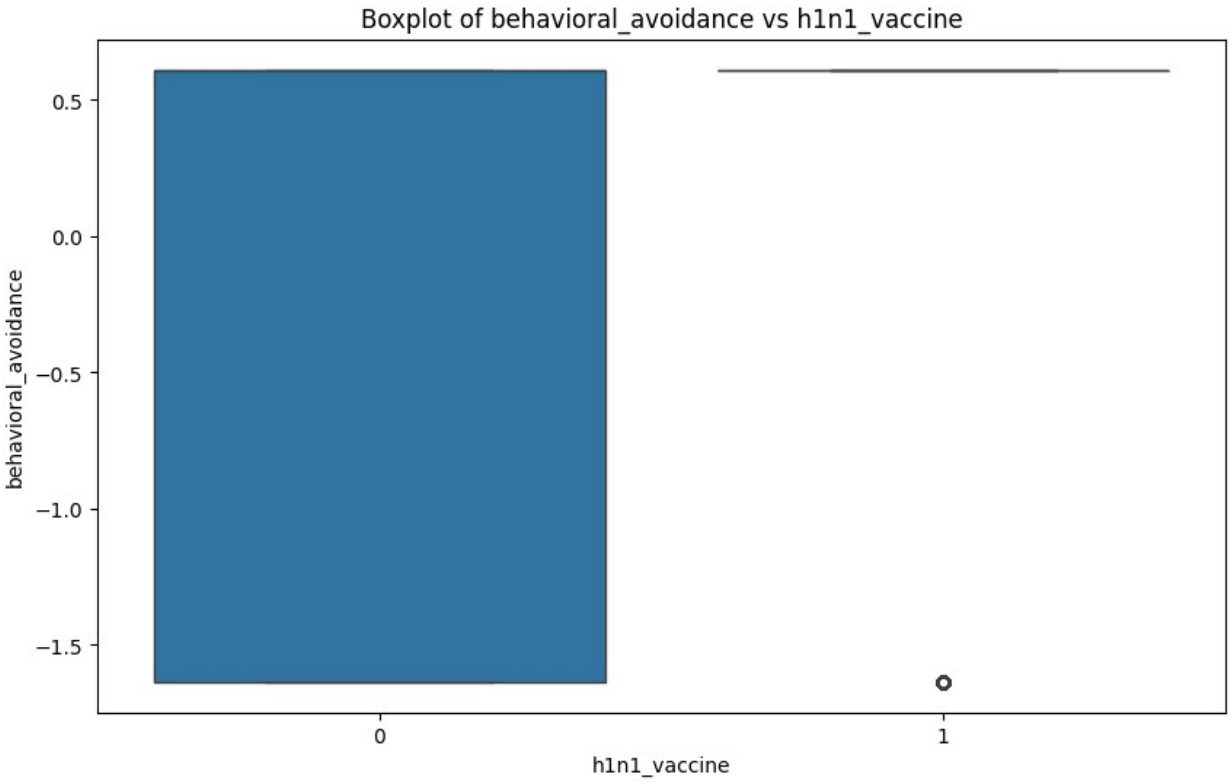
```
import seaborn as sns
import matplotlib.pyplot as plt

# Bivariate analysis for numeric features
for feature in numeric_features:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x=y_train, y=X_train_preprocessed_df[feature])
    plt.title(f'Boxplot of {feature} vs h1n1_vaccine')
    plt.show()

# Bivariate analysis for categorical features
for feature in
preprocessor.named_transformers_['cat'].named_steps['onehot'].get_feat
ure_names_out(categorical_features):
    plt.figure(figsize=(10, 6))
    sns.countplot(x=X_train_preprocessed_df[feature], hue=y_train)
    plt.title(f'Count plot of {feature} vs h1n1_vaccine')
    plt.show()

# Chi-square test for categorical features
from scipy.stats import chi2_contingency
```

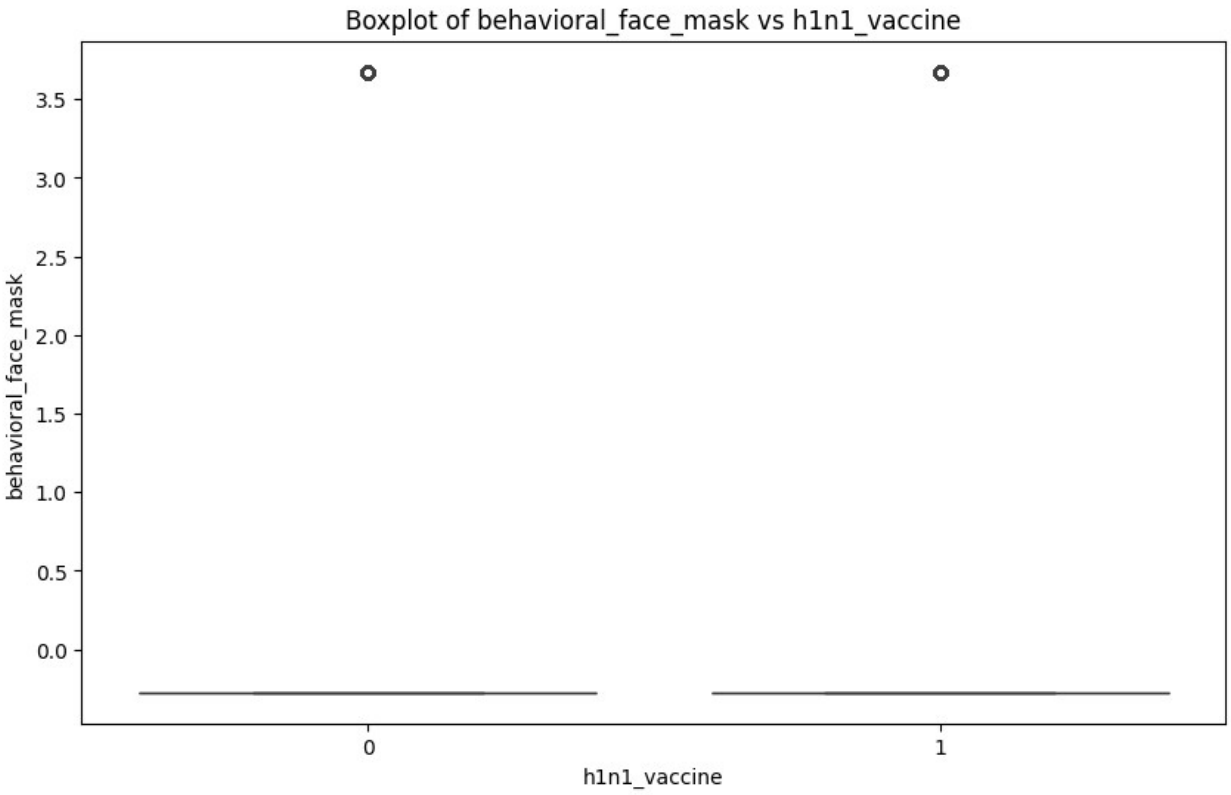
```
for feature in
preprocessor.named_transformers_['cat'].named_steps['onehot'].get_feat
ure_names_out(categorical_features):
    contingency_table = pd.crosstab(X_train_preprocessed_df[feature],
y_train)
    chi2, p, dof, ex = chi2_contingency(contingency_table)
    print(f'Chi-square test for {feature}: p-value = {p}')
```

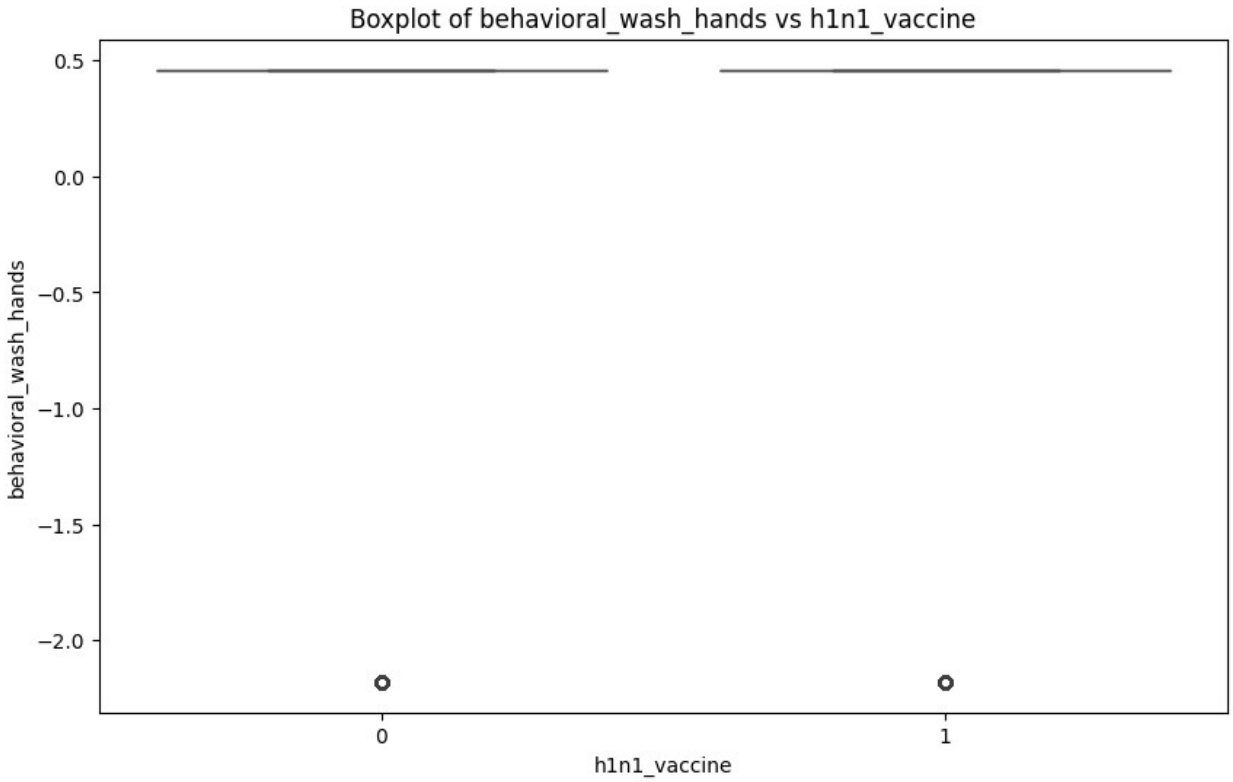


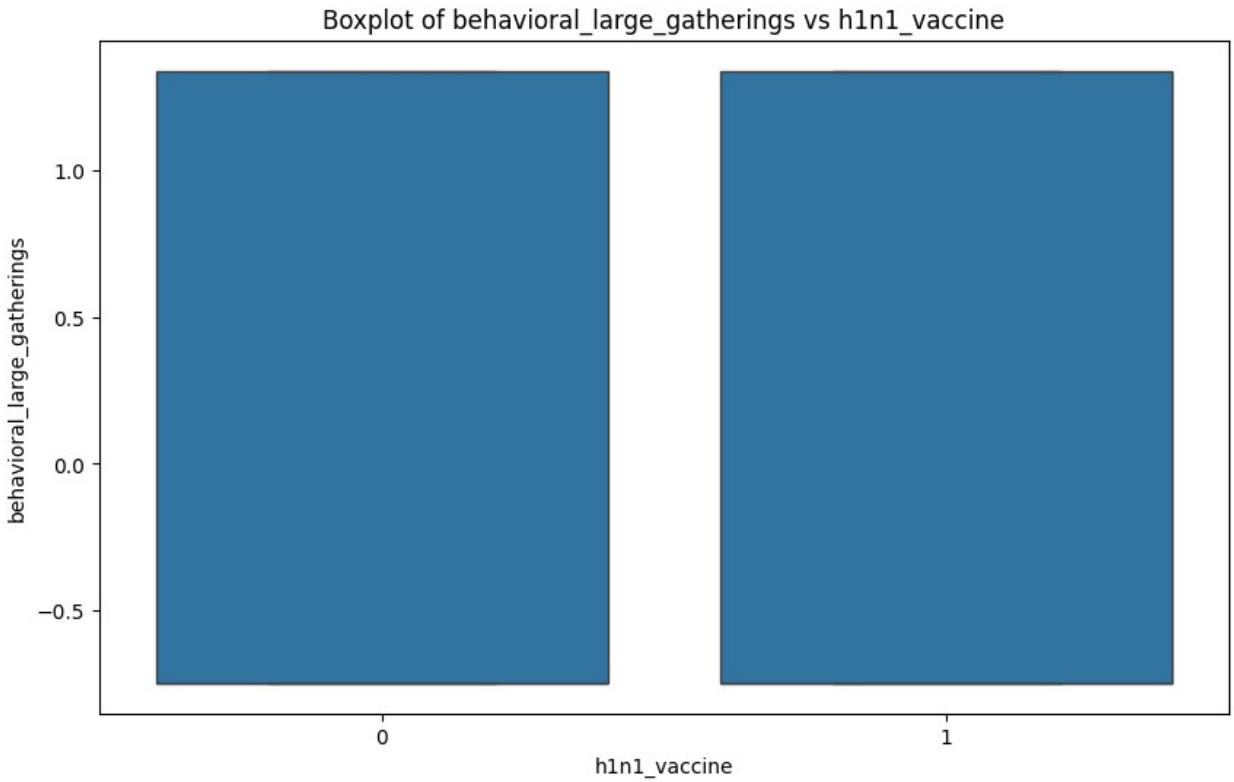

Boxplot of h1n1_knowledge vs h1n1_vaccine
h1n1_knowledge
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
1
h1n1_vaccine

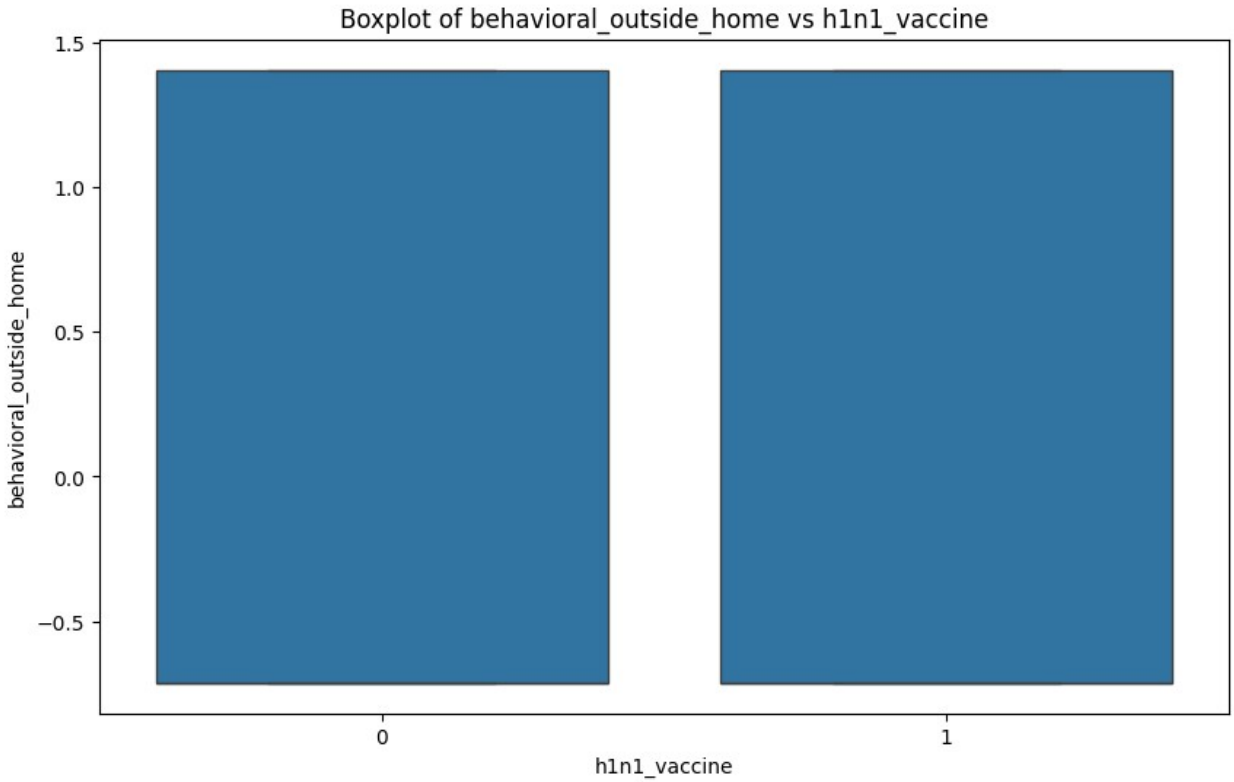

Boxplot of behavioral_antiviral_meds vs h1n1_vaccine
behavioral_antiviral_meds
4
3
2
1
0
0
1
h1n1_vaccine

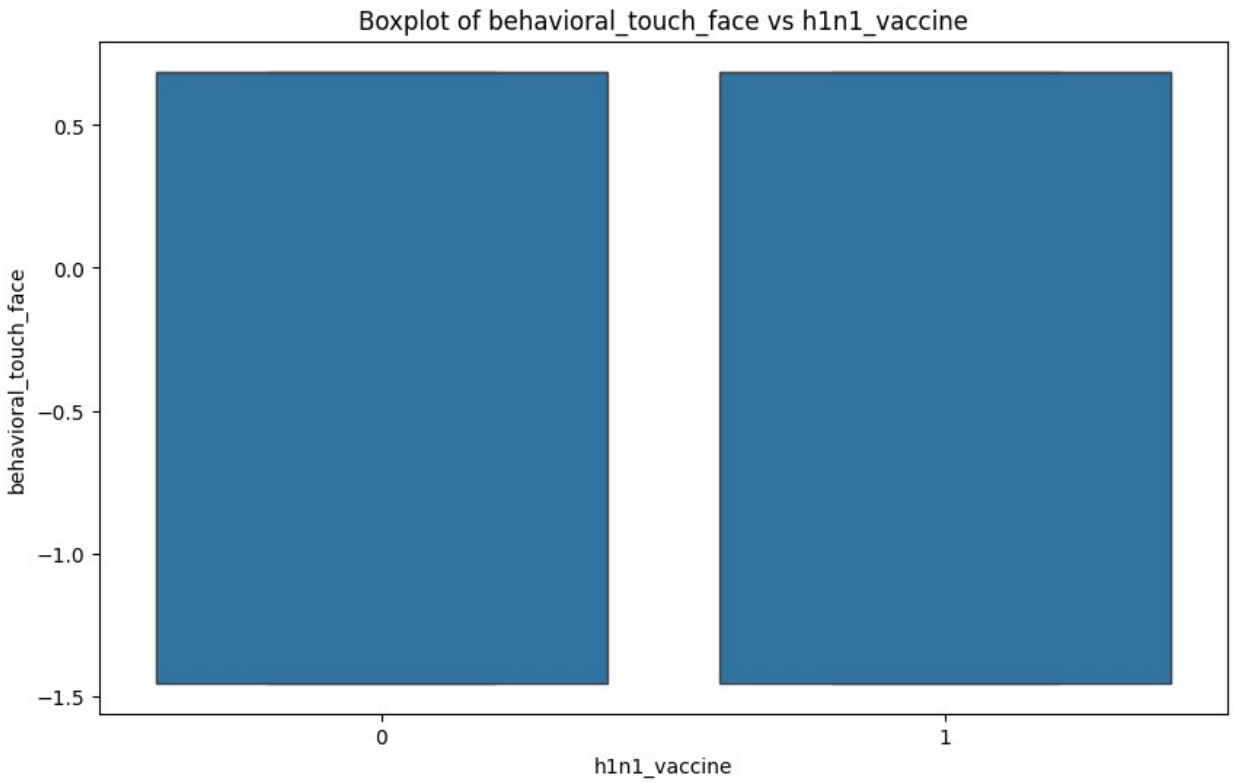

Boxplot of behavioral_avoidance vs h1n1_vaccine
behavioral_avoidance
0.5
0.0
−0.5
−1.0
−1.5
0
1
h1n1_vaccine

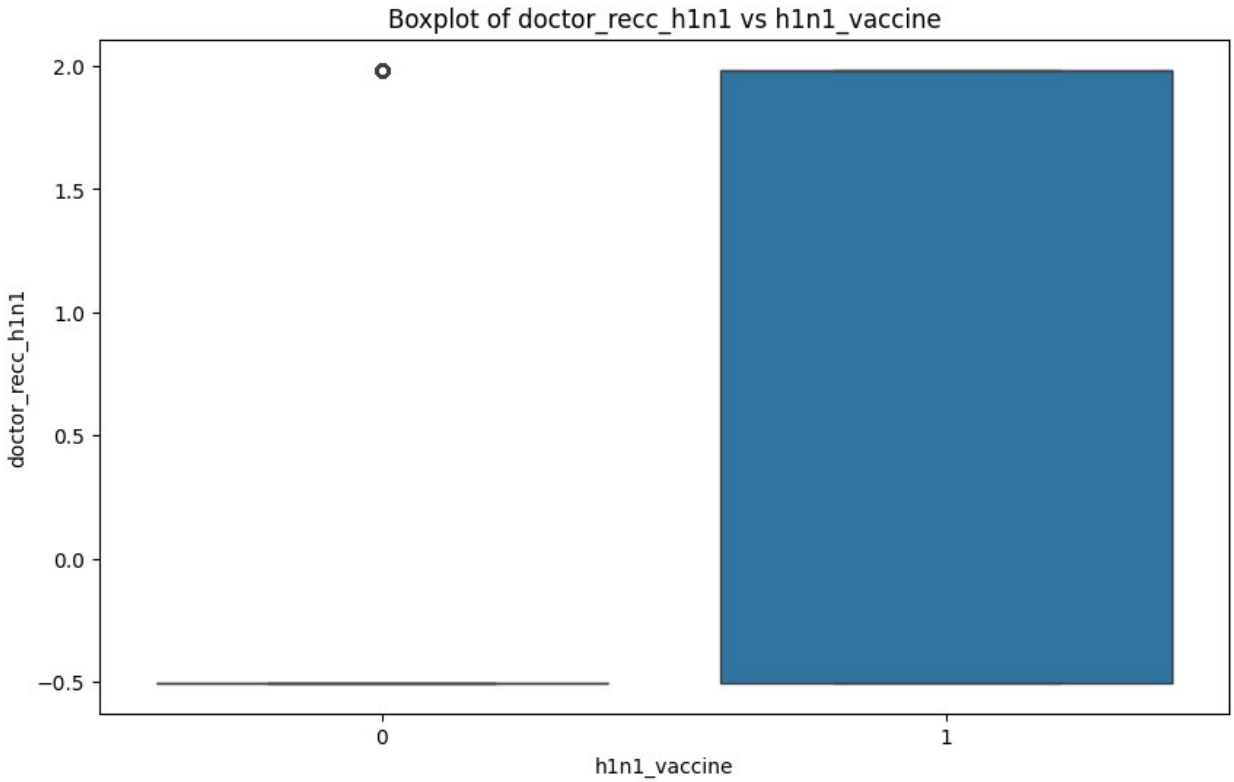

Boxplot of behavioral_face_mask vs h1n1_vaccine
behavioral_face_mask
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
1
h1n1_vaccine

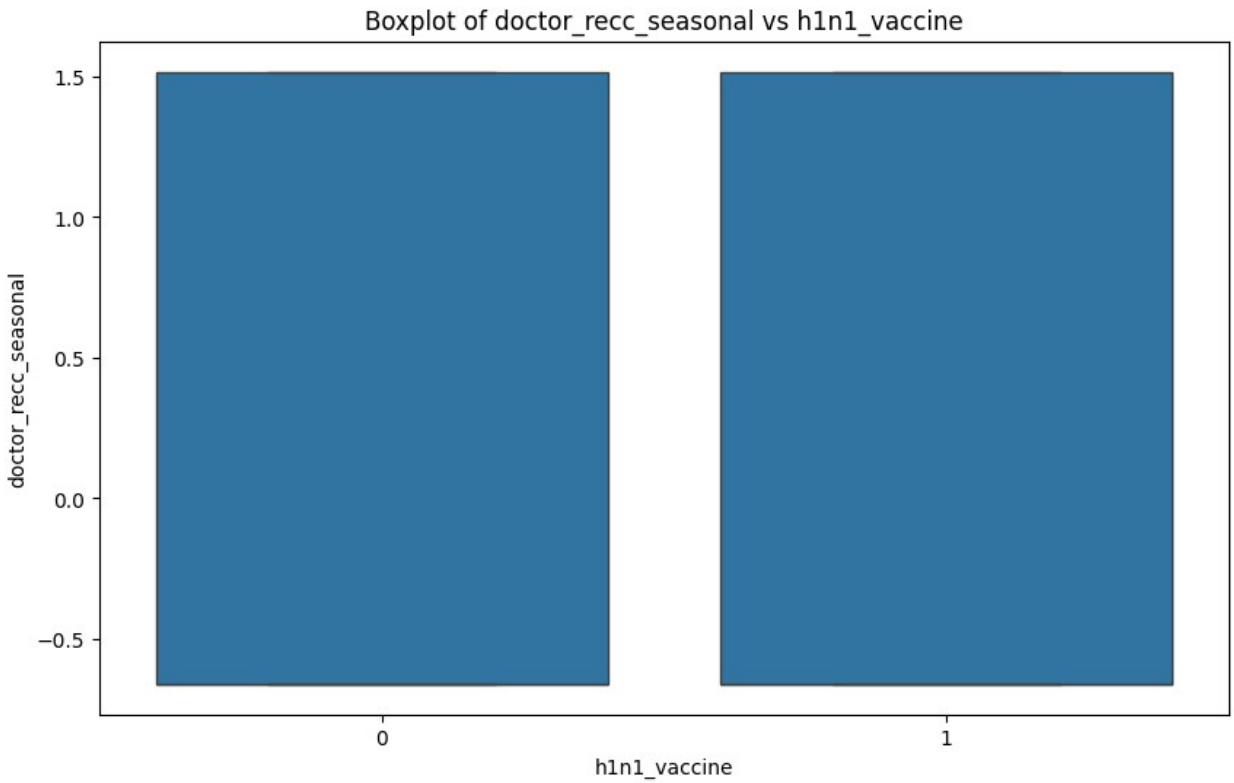

Boxplot of behavioral_wash_hands vs h1n1_vaccine
behavioral_wash_hands
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
1
h1n1_vaccine

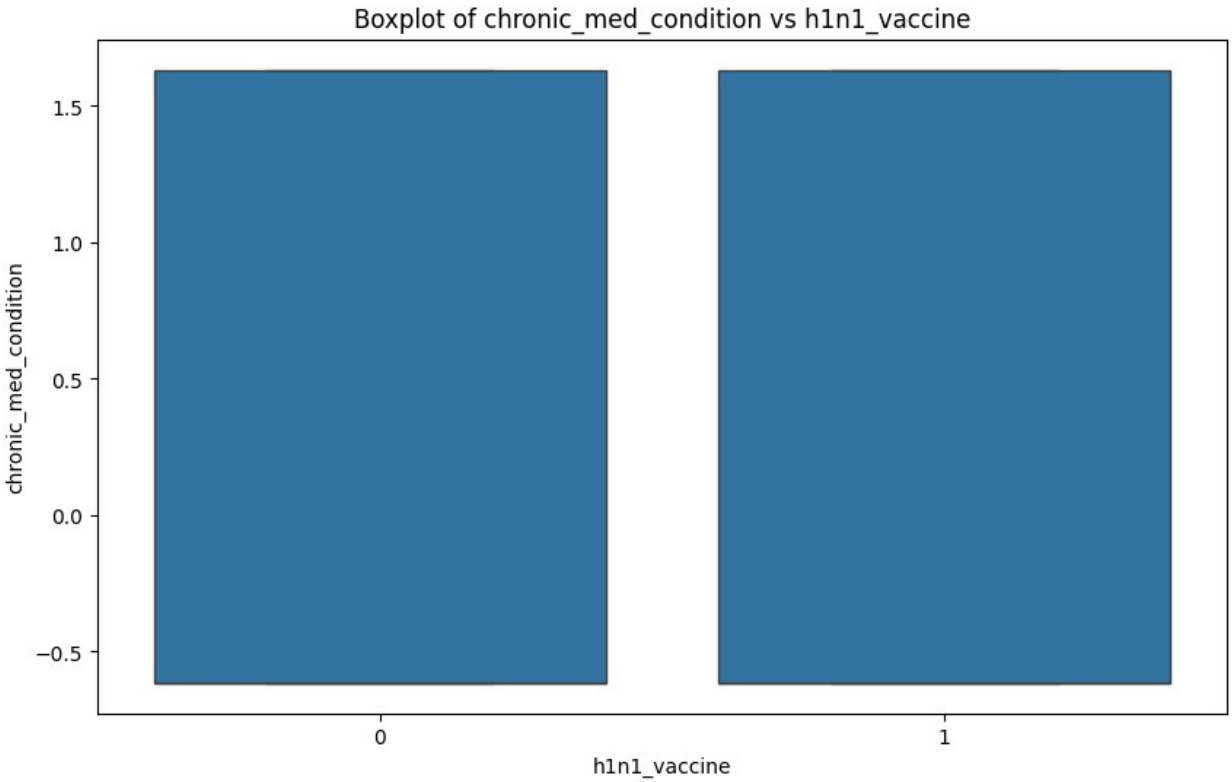

Boxplot of behavioral_large_gatherings vs h1n1_vaccine
behavioral_large_gatherings
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

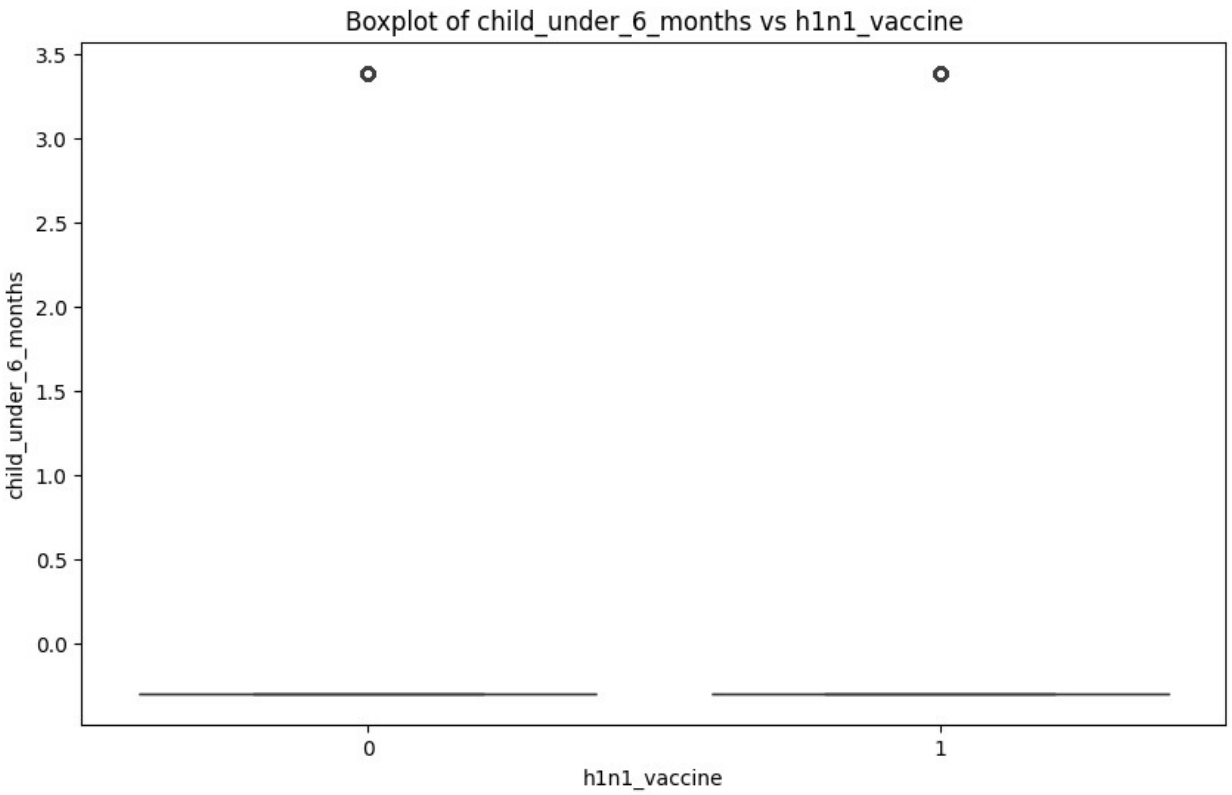

Boxplot of behavioral_outside_home vs h1n1_vaccine
behavioral_outside_home
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

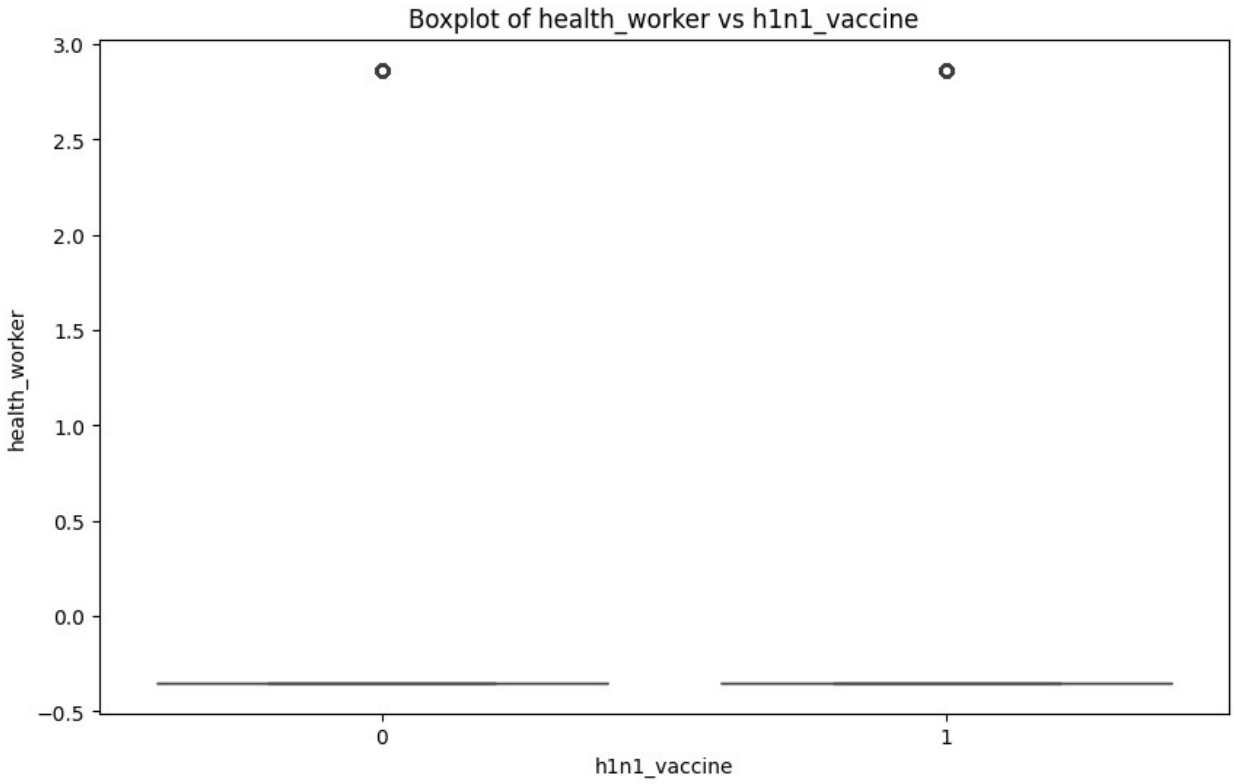

Boxplot of behavioral_touch_face vs h1n1_vaccine
behavioral_touch_face
0.5
0.0
−0.5
−1.0
−1.5
0
1
h1n1_vaccine

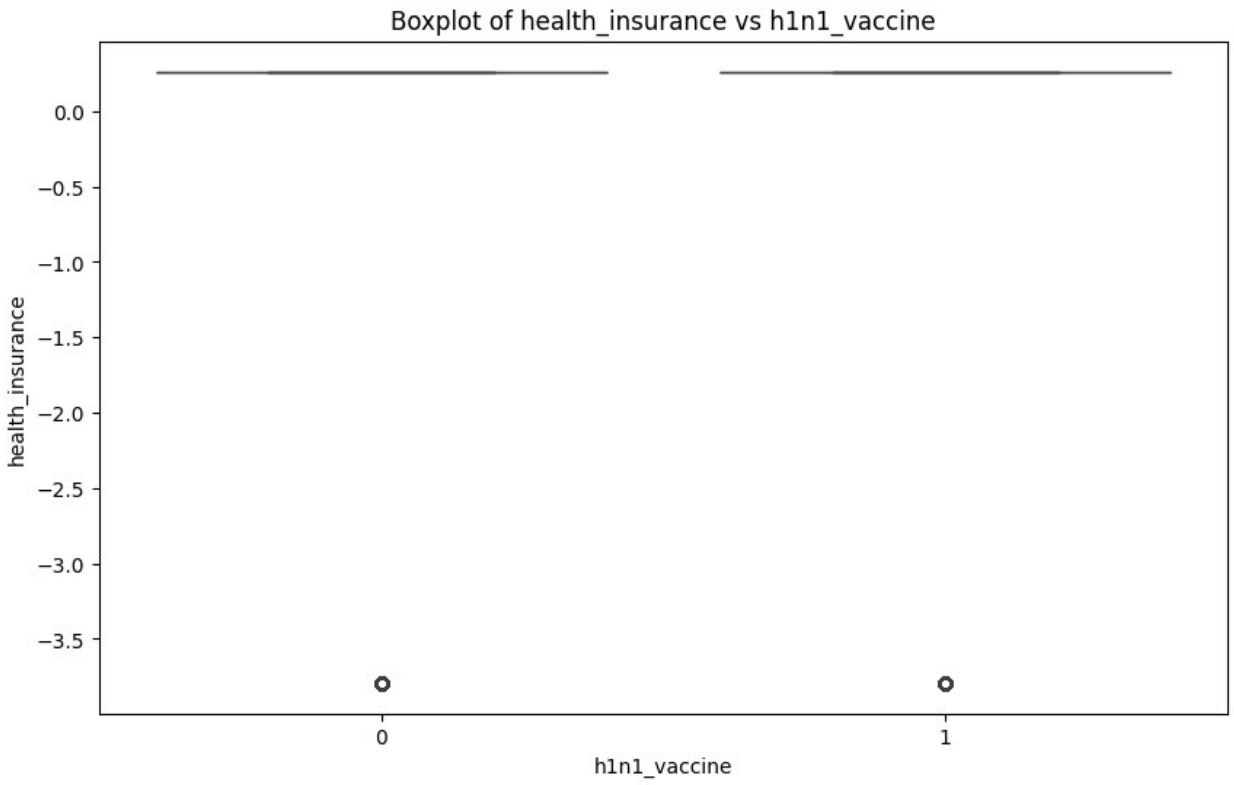

Boxplot of doctor_recc_h1n1 vs h1n1_vaccine
doctor_recc_h1n1
2.0
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

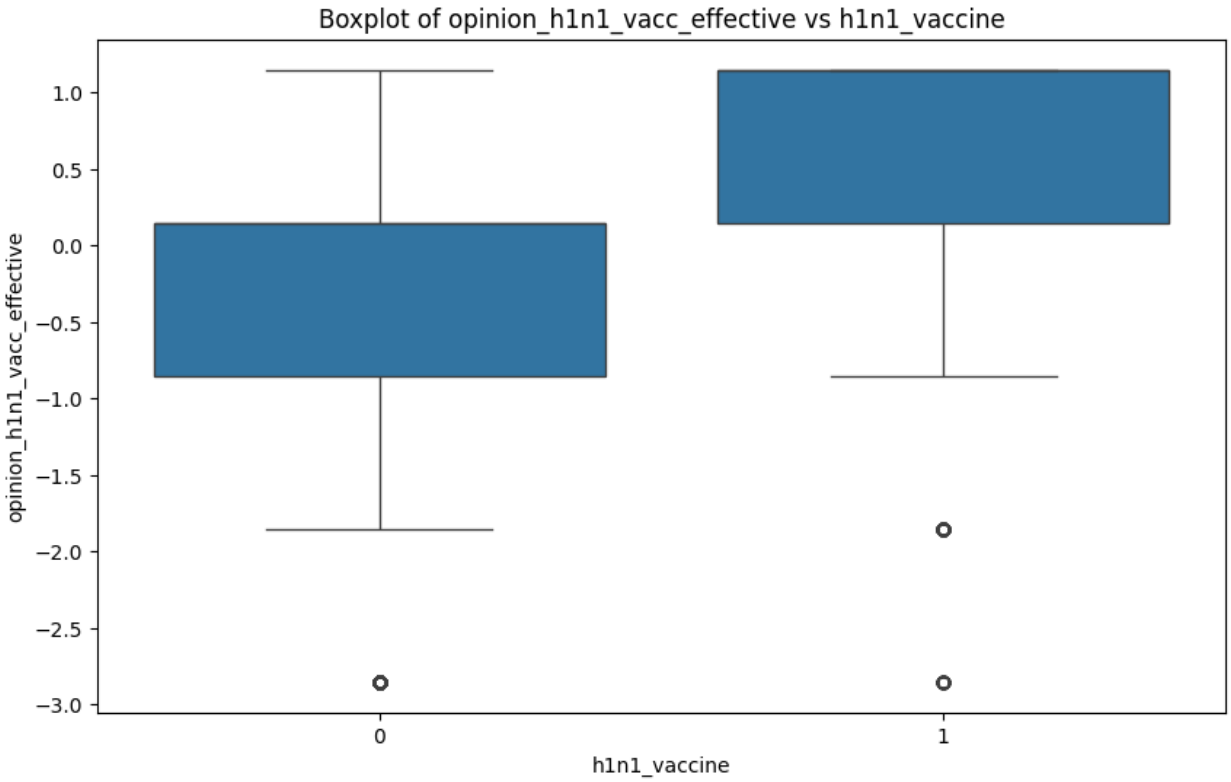

Boxplot of doctor_recc_seasonal vs h1n1_vaccine
doctor_recc_seasonal
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

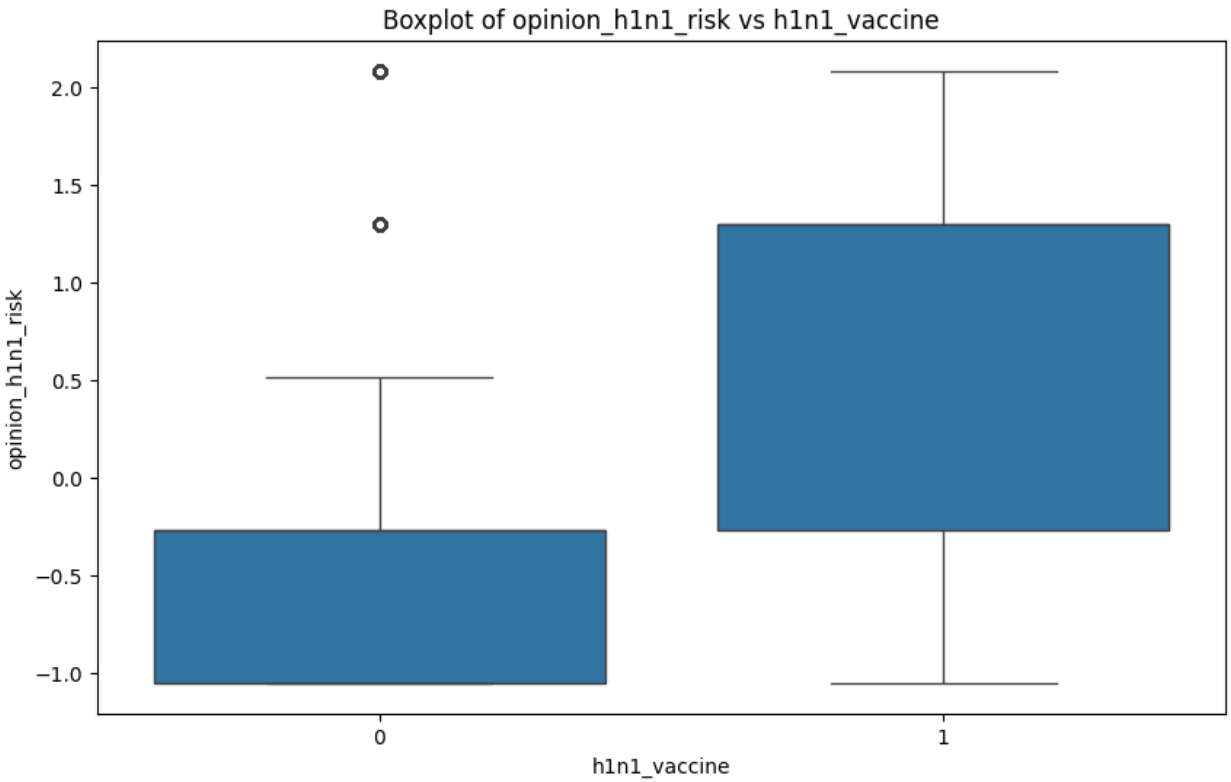

Boxplot of chronic_med_condition vs h1n1_vaccine
chronic_med_condition
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

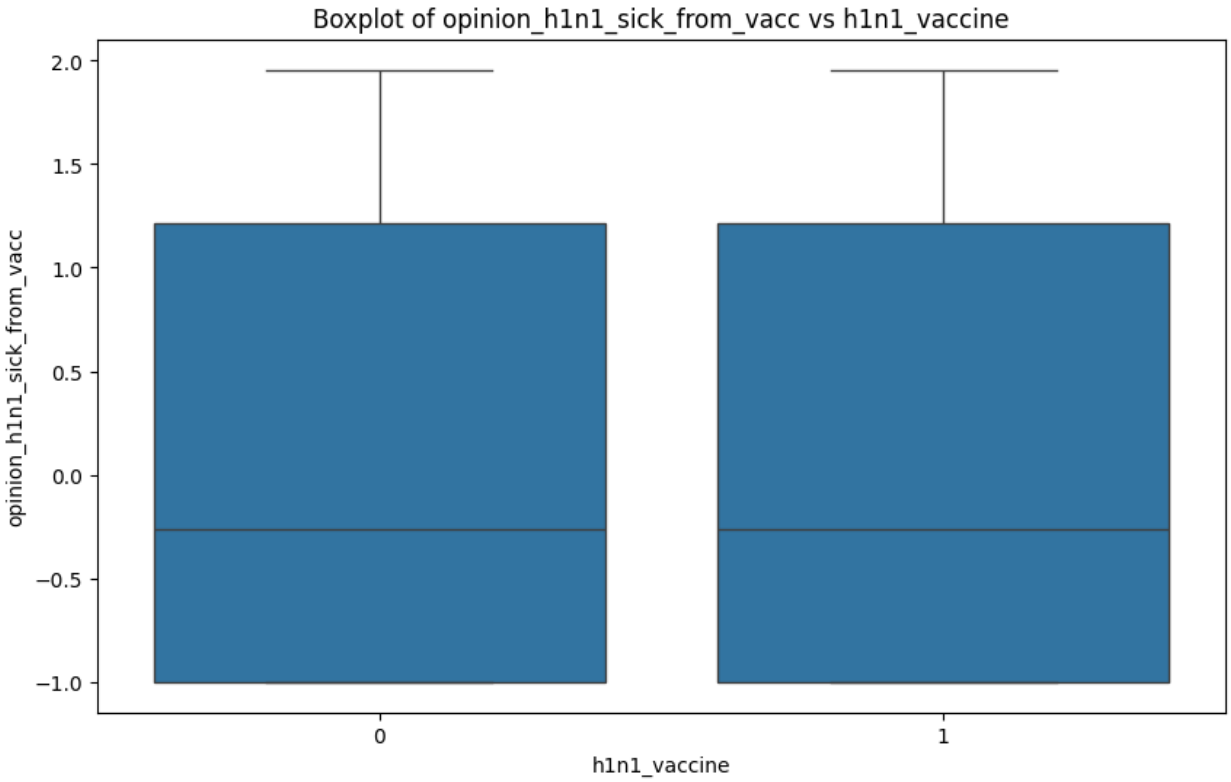

Boxplot of child_under_6_months vs h1n1_vaccine
child_under_6_months
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
1
h1n1_vaccine

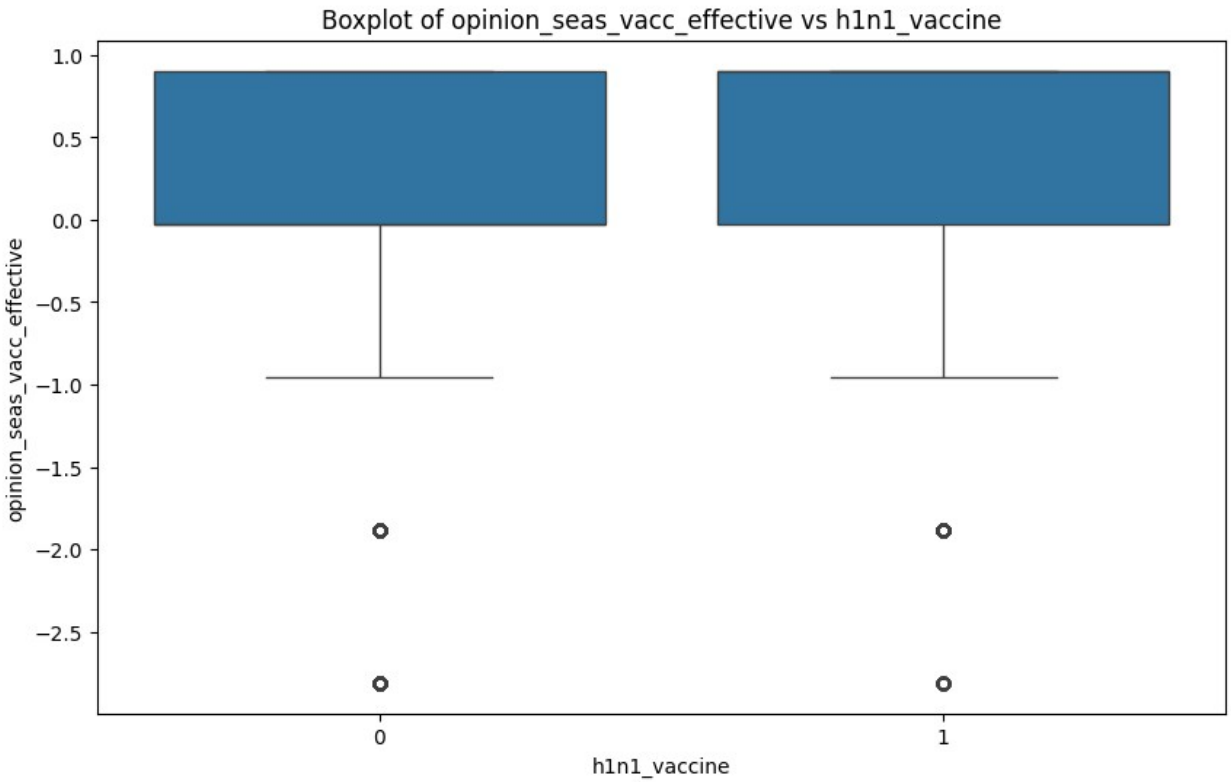

Boxplot of health_worker vs h1n1_vaccine
health_worker
h1n1_vaccine
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
0
1

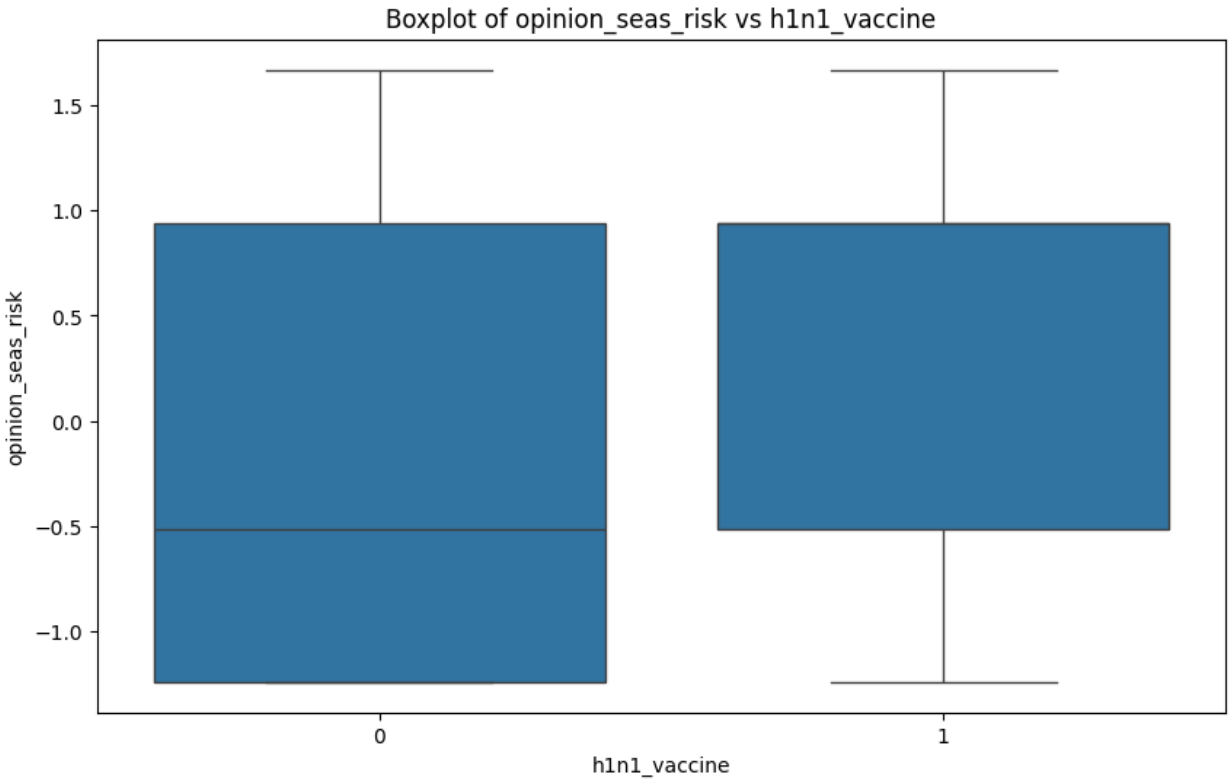

Boxplot of health_insurance vs h1n1_vaccine
health_insurance
h1n1_vaccine
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
−3.5
0
1

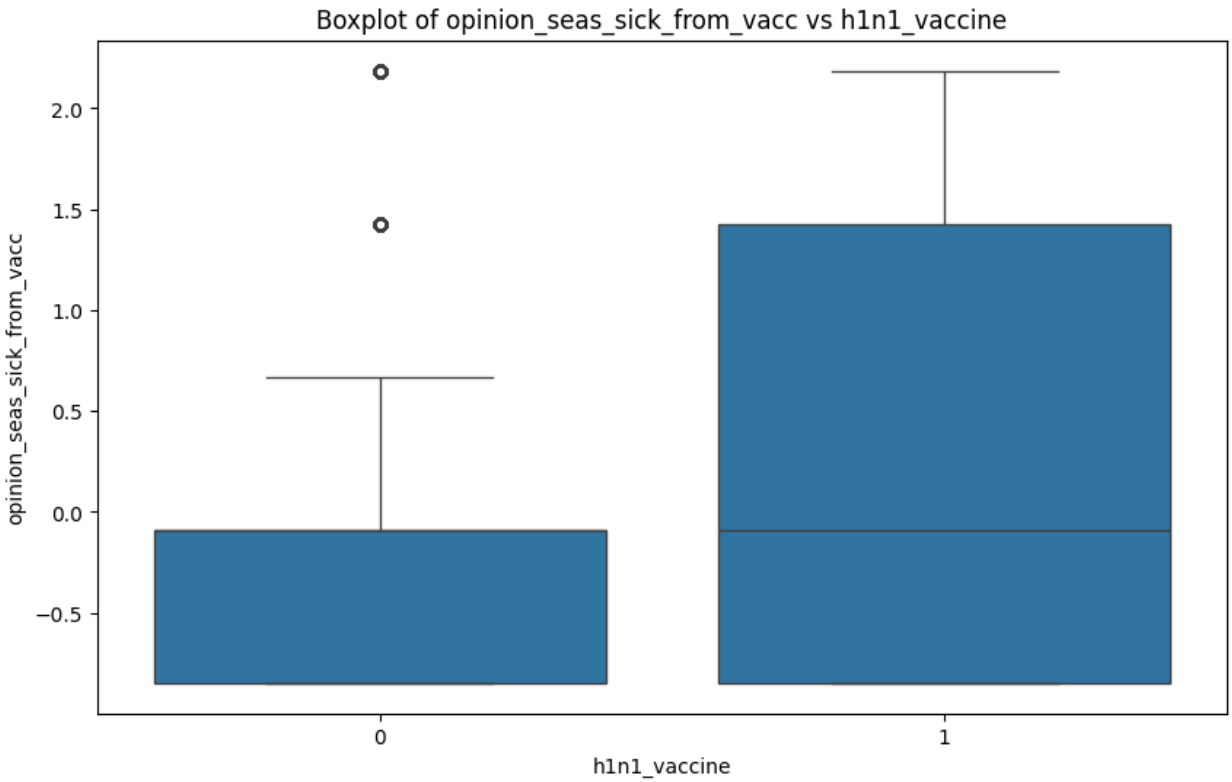

Boxplot of opinion_h1n1_vacc_effective vs h1n1_vaccine
opinion_h1n1_vacc_effective
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
0
1
h1n1_vaccine

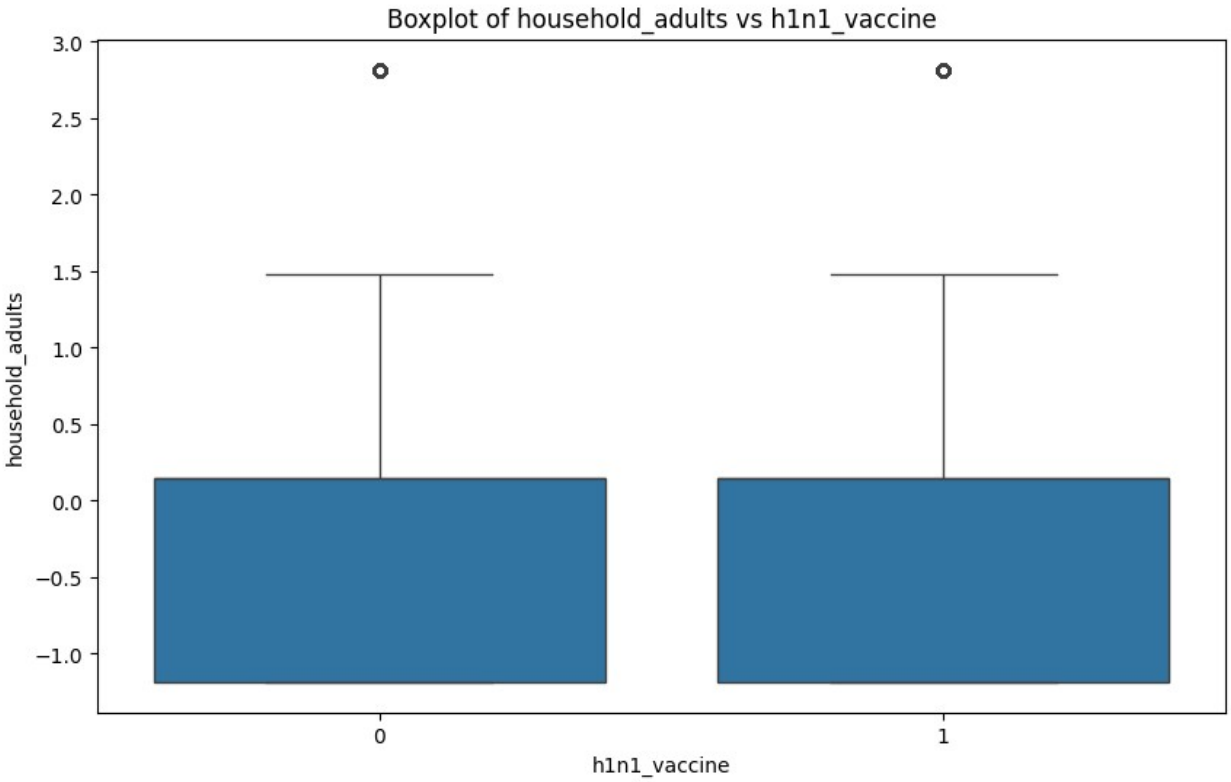

Boxplot of opinion_h1n1_risk vs h1n1_vaccine
opinion_h1n1_risk
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
0
1
h1n1_vaccine

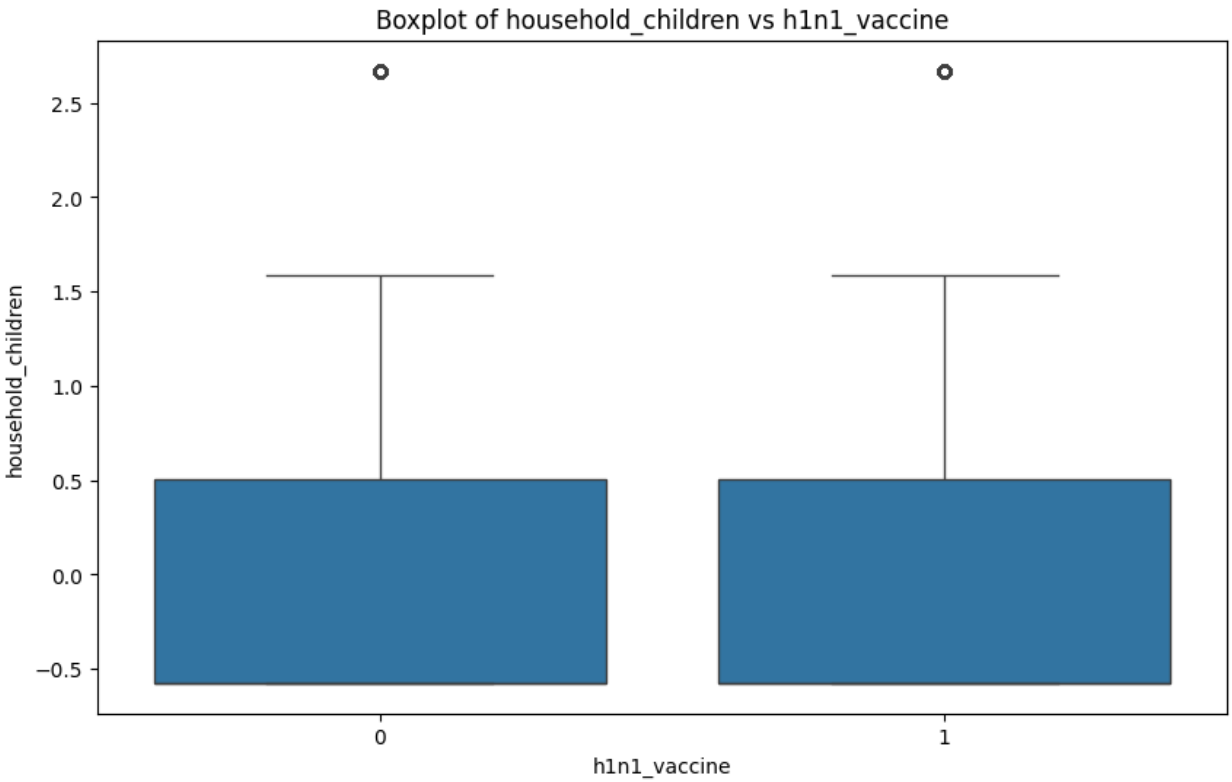

Boxplot of opinion_h1n1_sick_from_vacc vs h1n1_vaccine
opinion_h1n1_sick_from_vacc
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
0
1
h1n1_vaccine

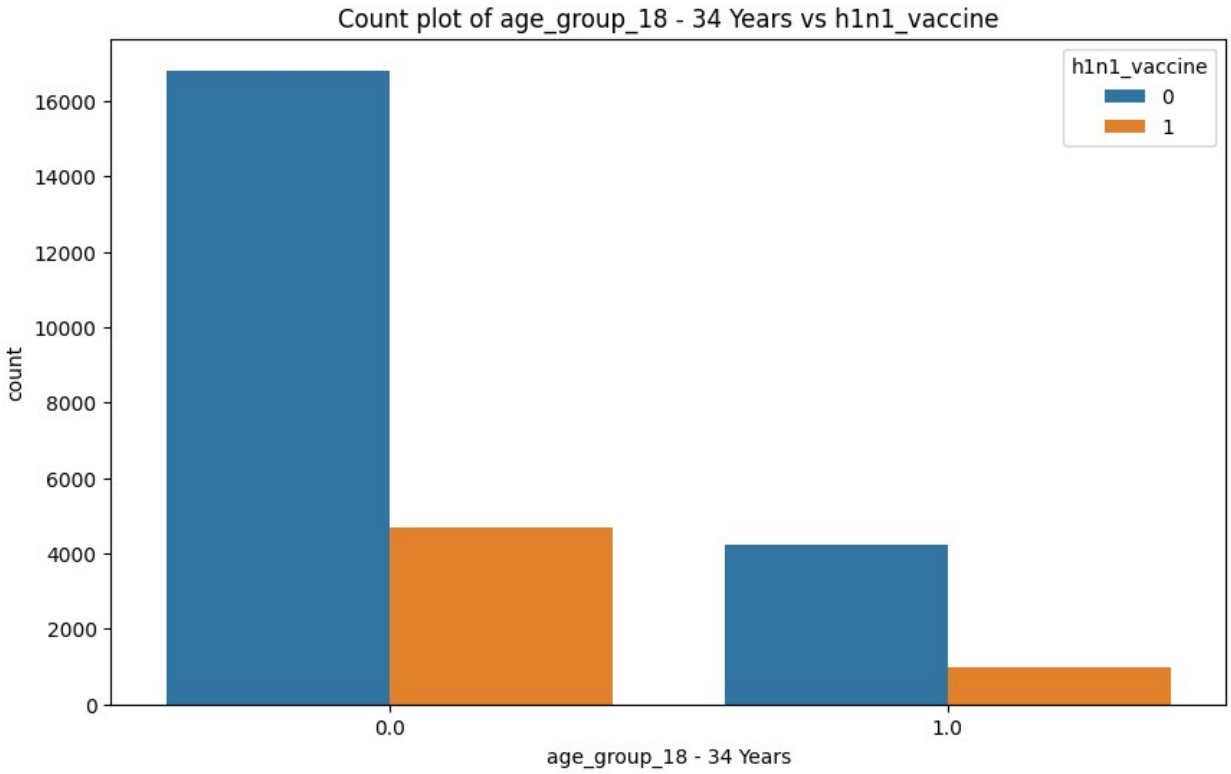

Boxplot of opinion_seas_vacc_effective vs h1n1_vaccine
opinion_seas_vacc_effective
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
0
1
h1n1_vaccine

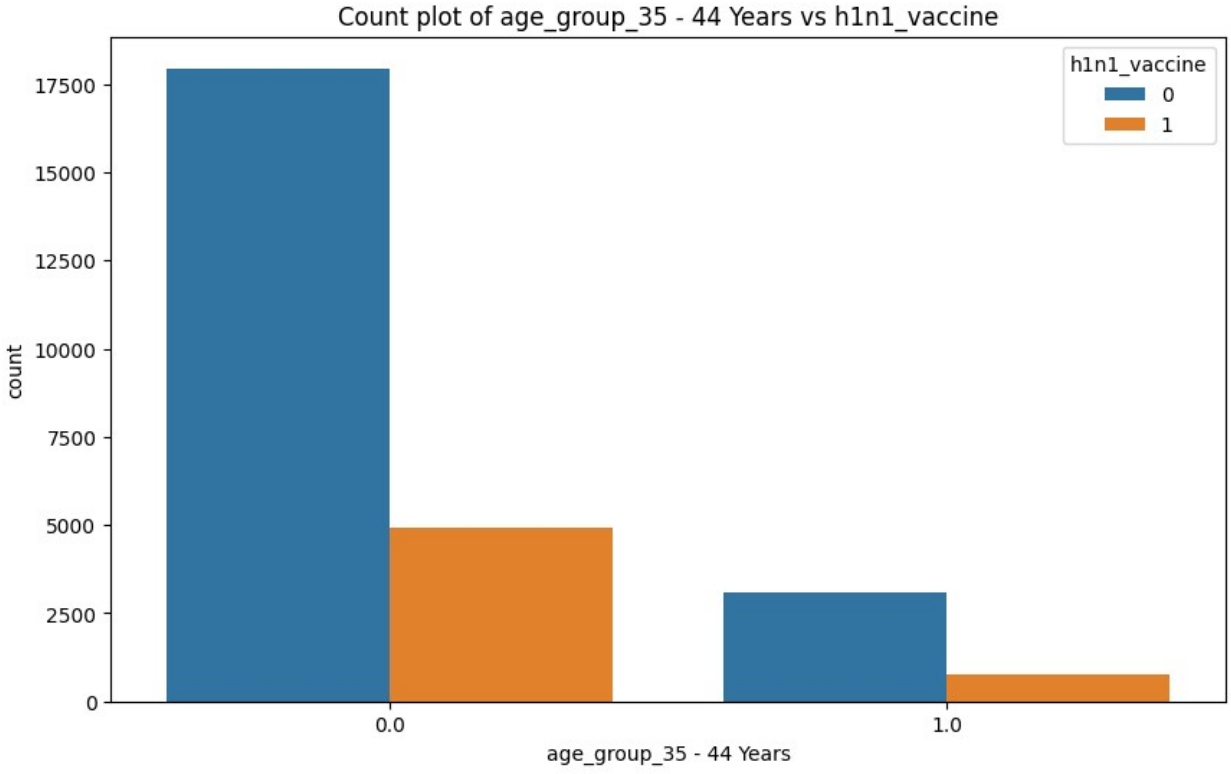

Boxplot of opinion_seas_risk vs h1n1_vaccine
opinion_seas_risk
1.5
1.0
0.5
0.0
−0.5
−1.0
0
1
h1n1_vaccine

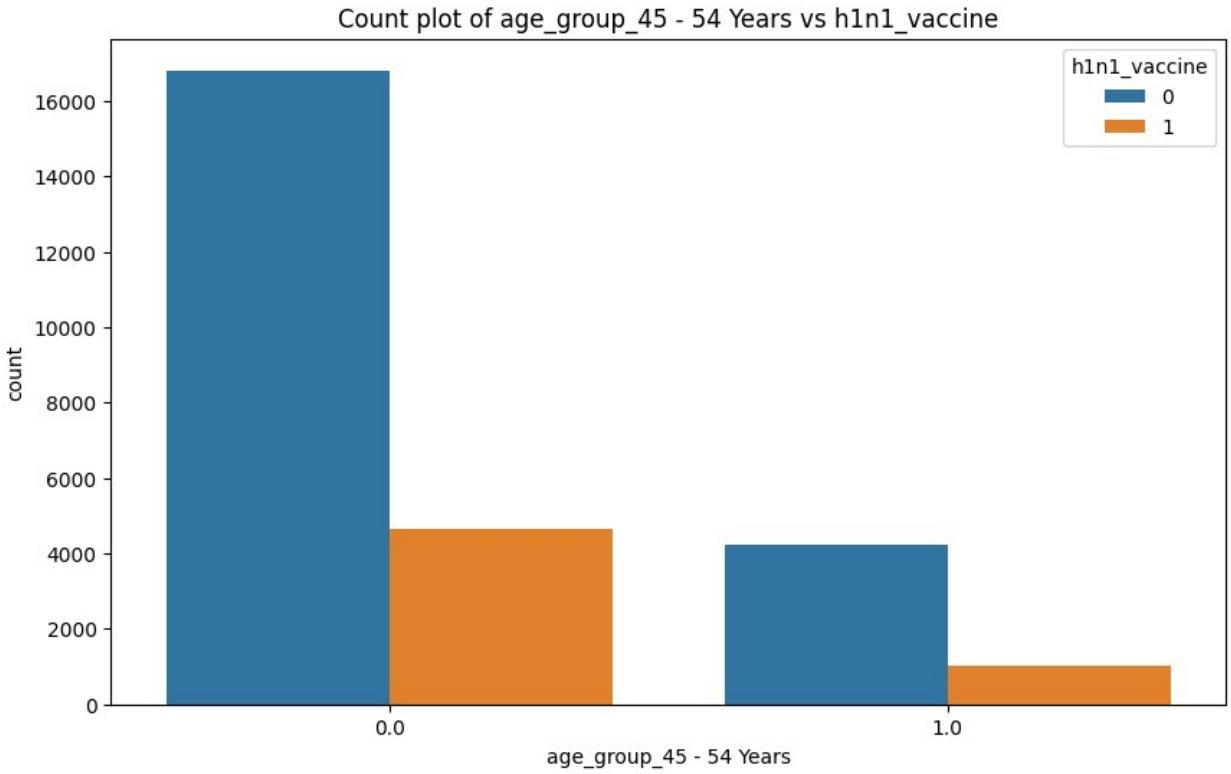

Boxplot of opinion_seas_sick_from_vacc vs h1n1_vaccine
opinion_seas_sick_from_vacc
2.0
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

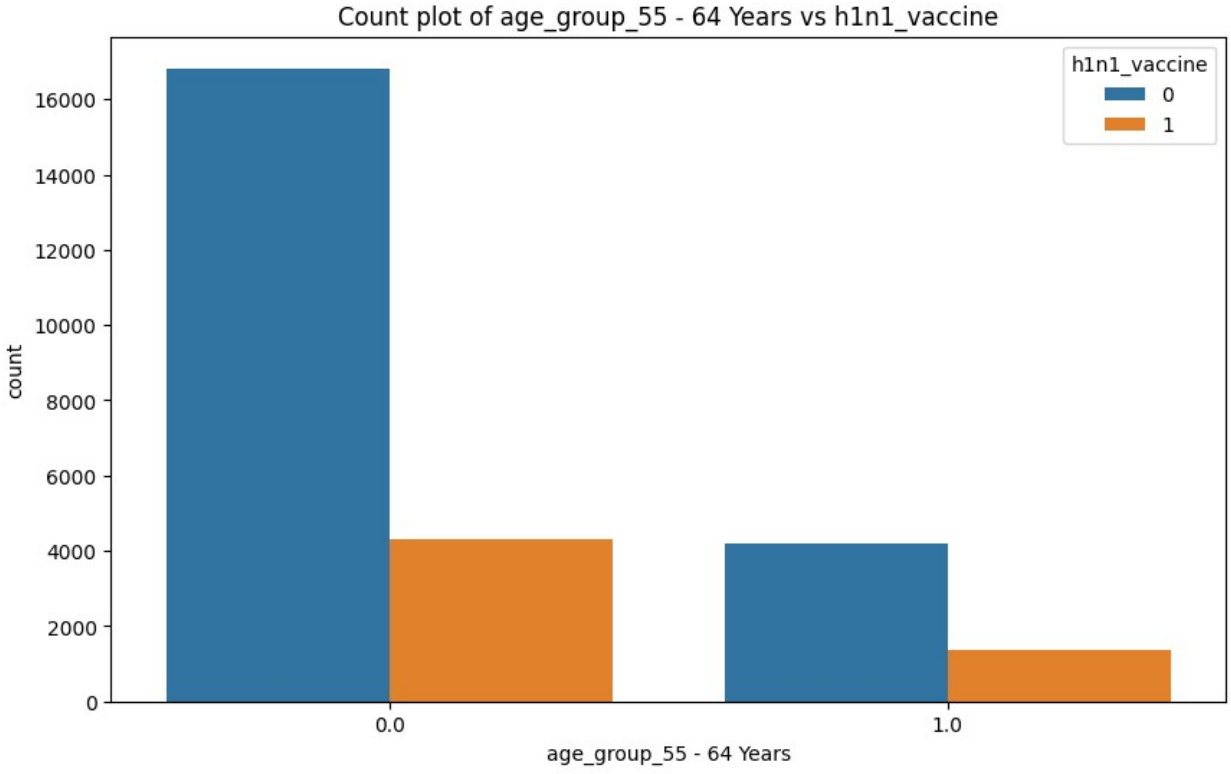

Boxplot of household_adults vs h1n1_vaccine
household_adults
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
0
1
h1n1_vaccine

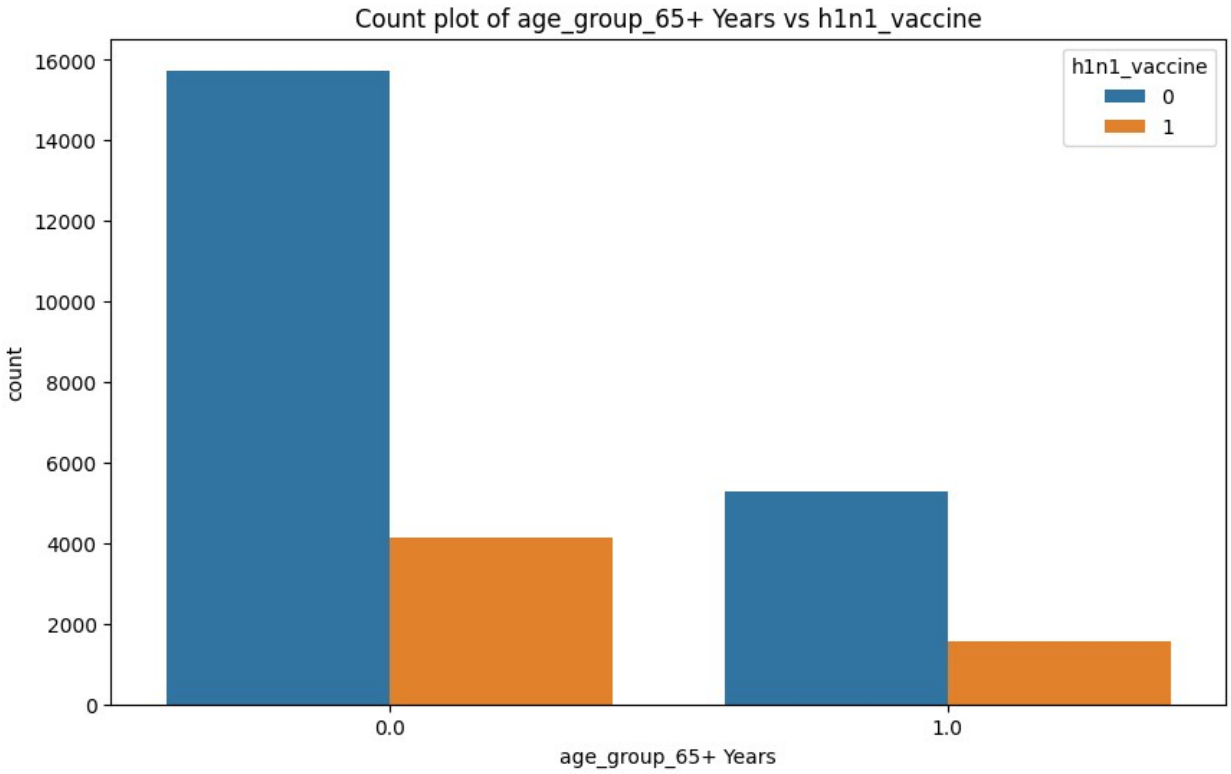

Boxplot of household_children vs h1n1_vaccine
household_children
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
0
1
h1n1_vaccine

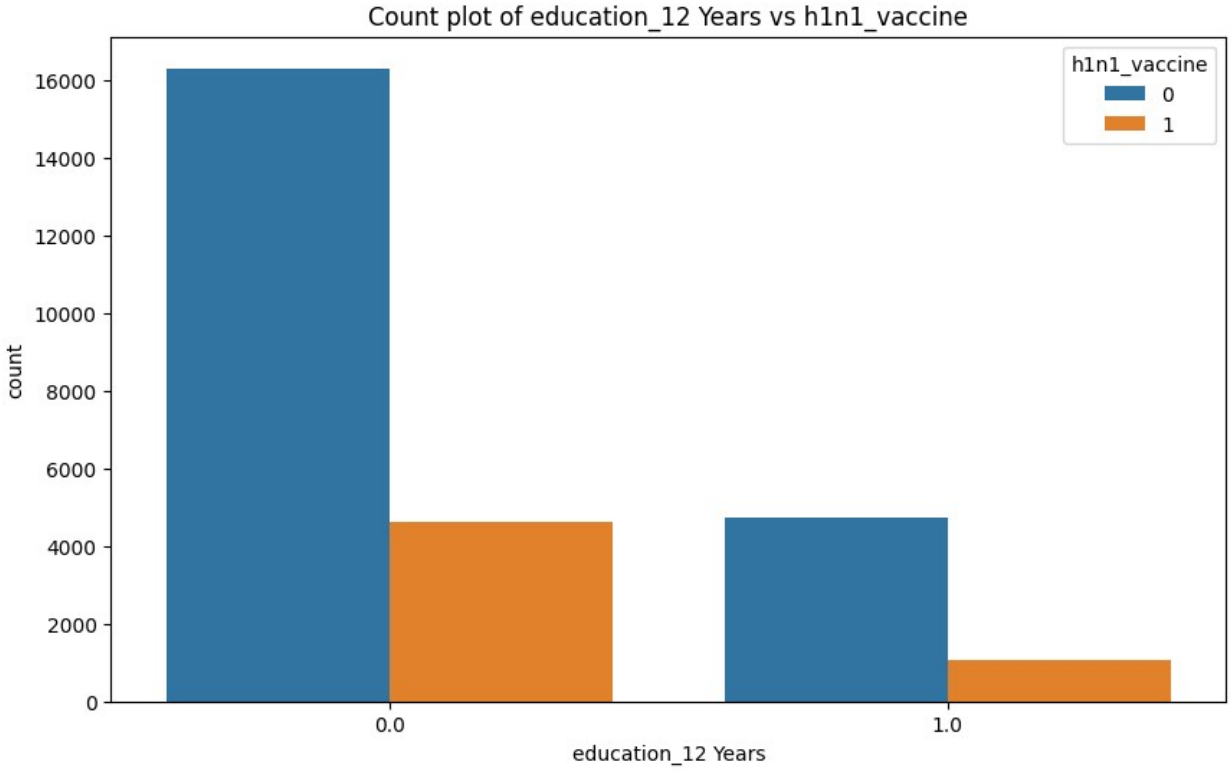

Count plot of age_group_18 - 34 Years vs h1n1_vaccine
h1n1_vaccine
0
1
16000
14000
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
age_group_18 - 34 Years

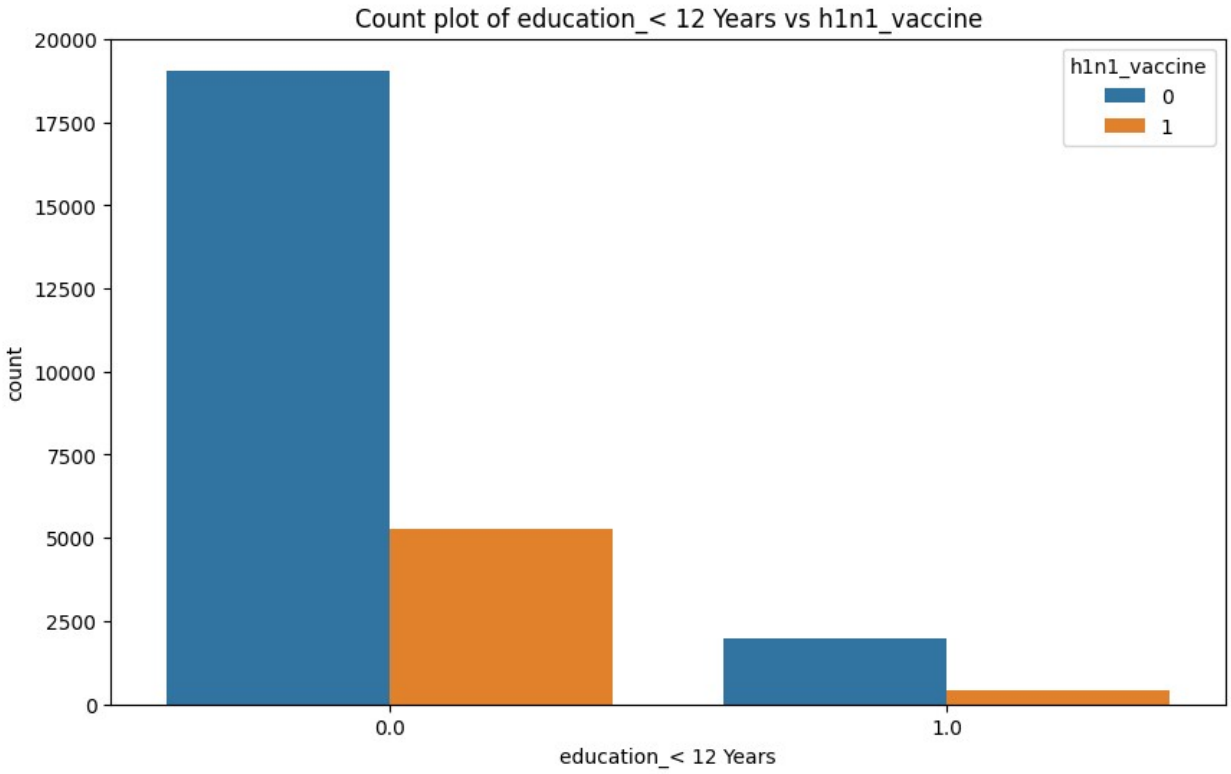

Count plot of age_group_35 - 44 Years vs h1n1_vaccine
h1n1_vaccine
0
1
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
age_group_35 - 44 Years

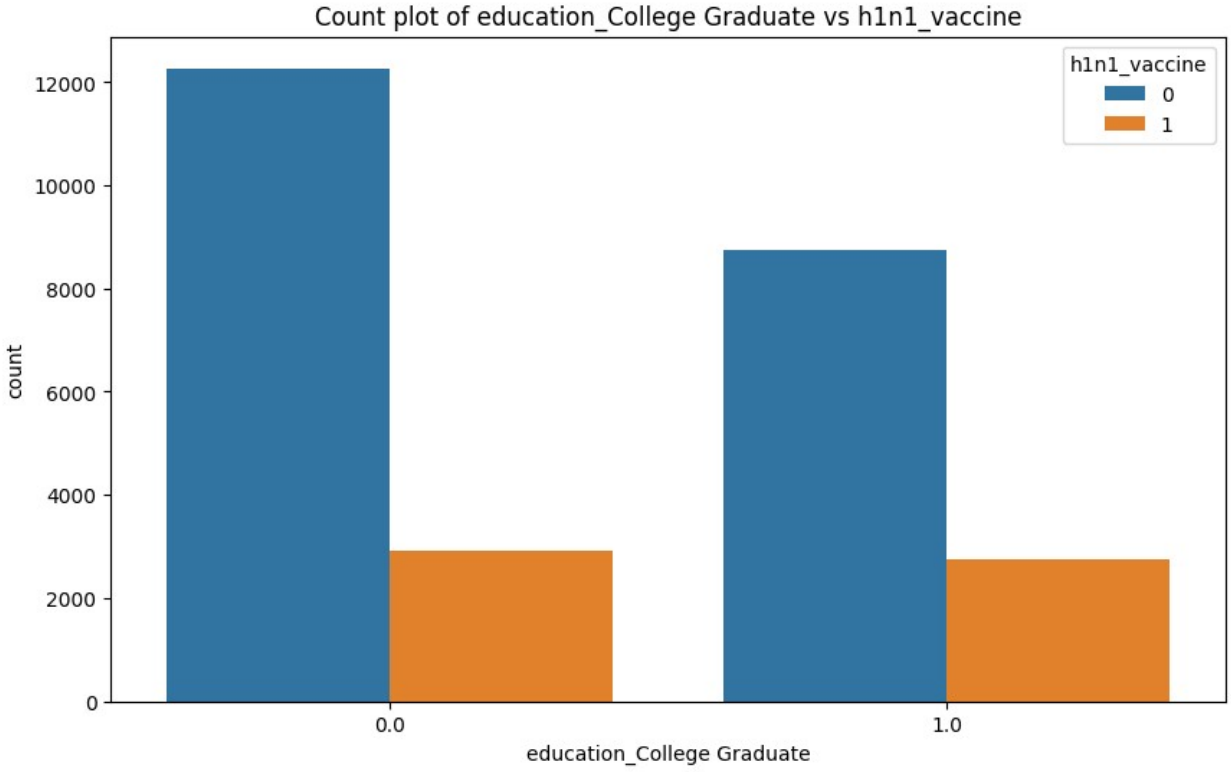

Count plot of age_group_45 - 54 Years vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
age_group_45 - 54 Years

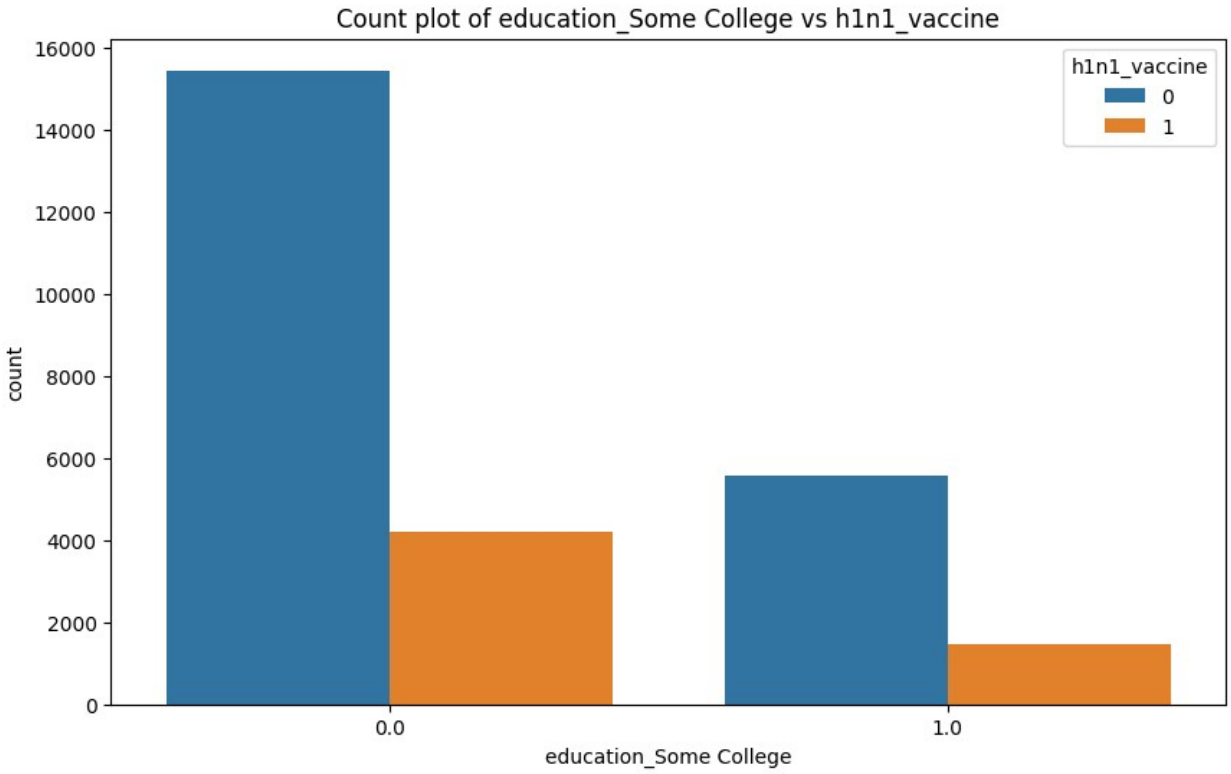

Count plot of age_group_55 - 64 Years vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
age_group_55 - 64 Years

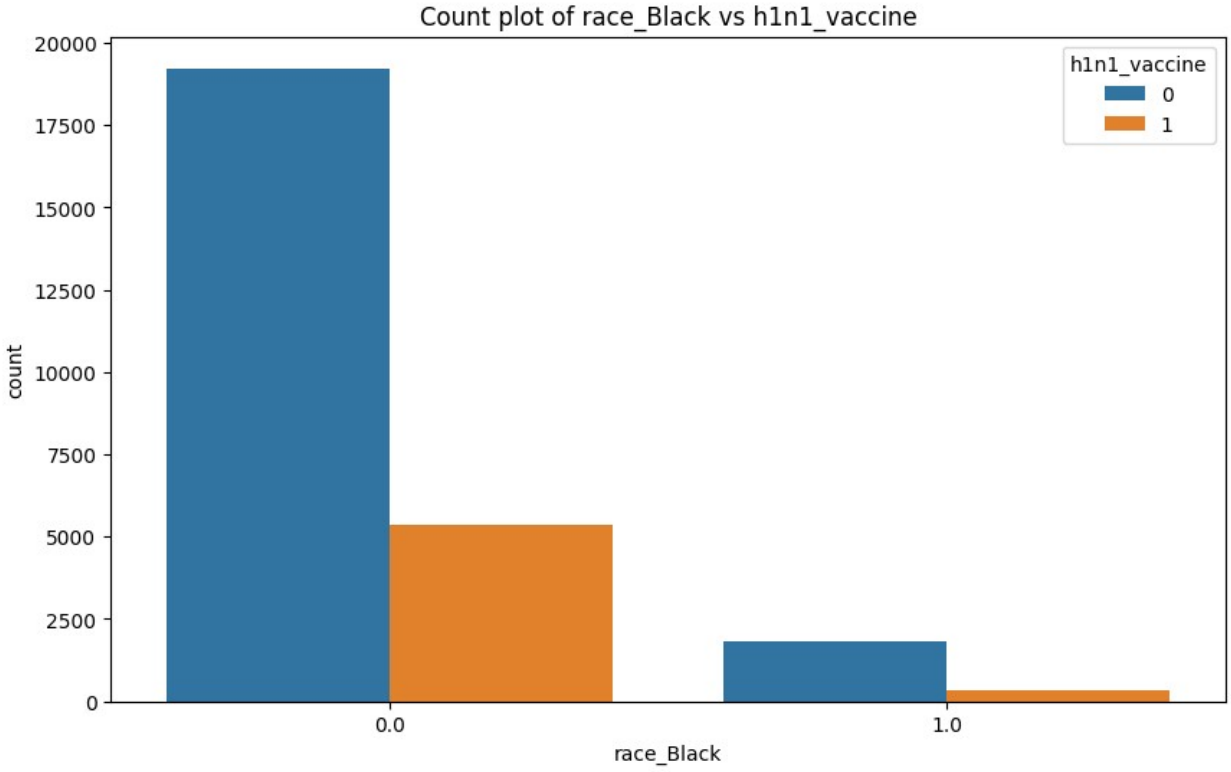

Count plot of age_group_65+ Years vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
age_group_65+ Years

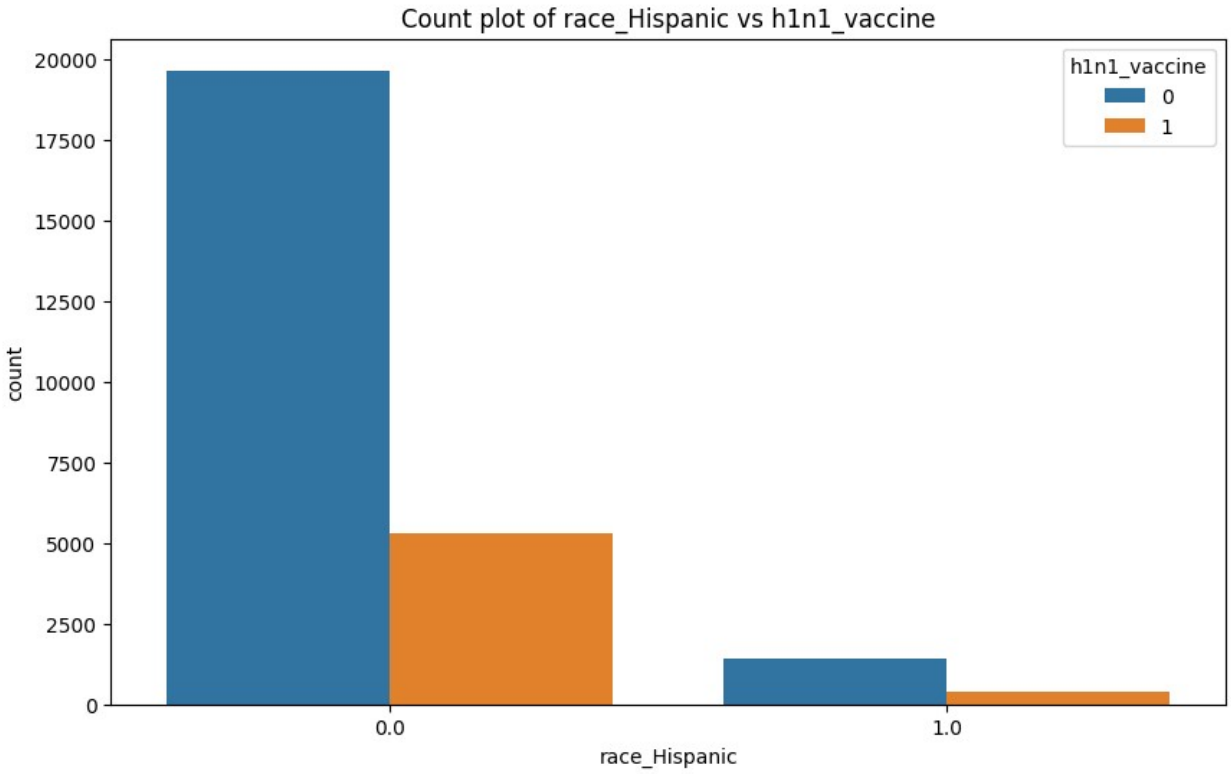

Count plot of education_12 Years vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
education_12 Years

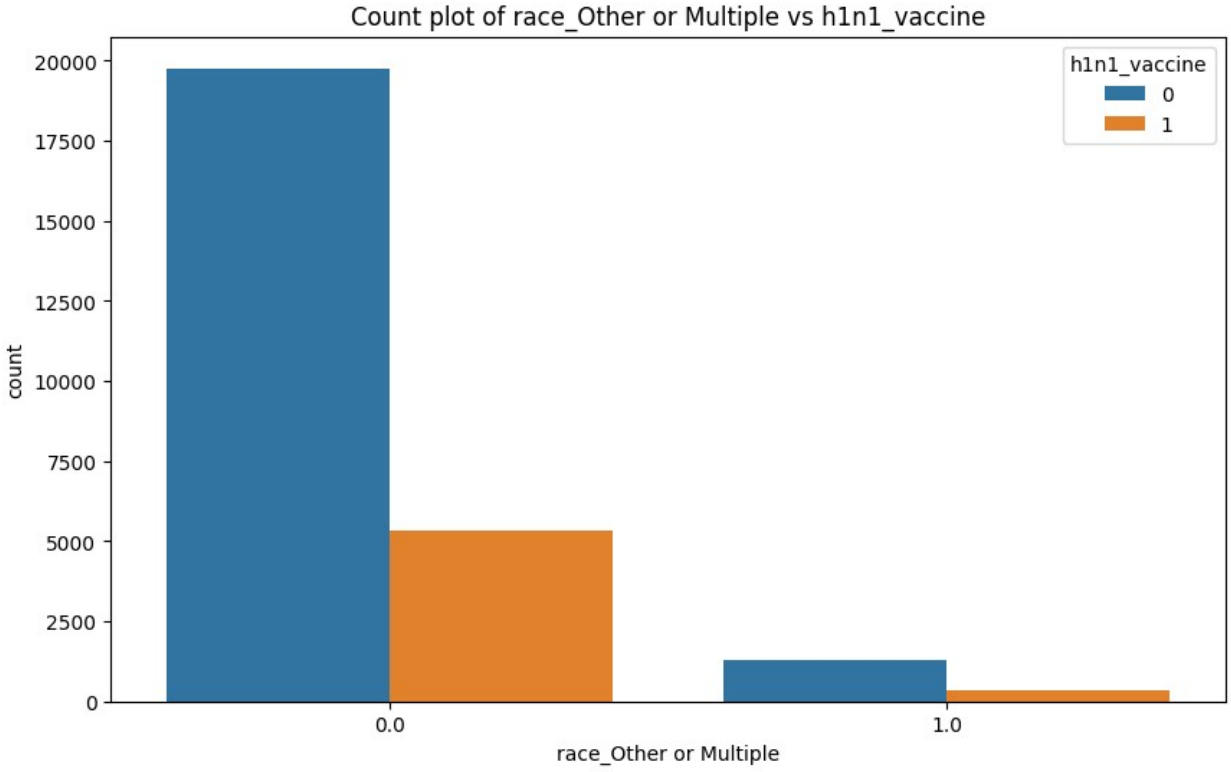

Count plot of education_< 12 Years vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
education_< 12 Years

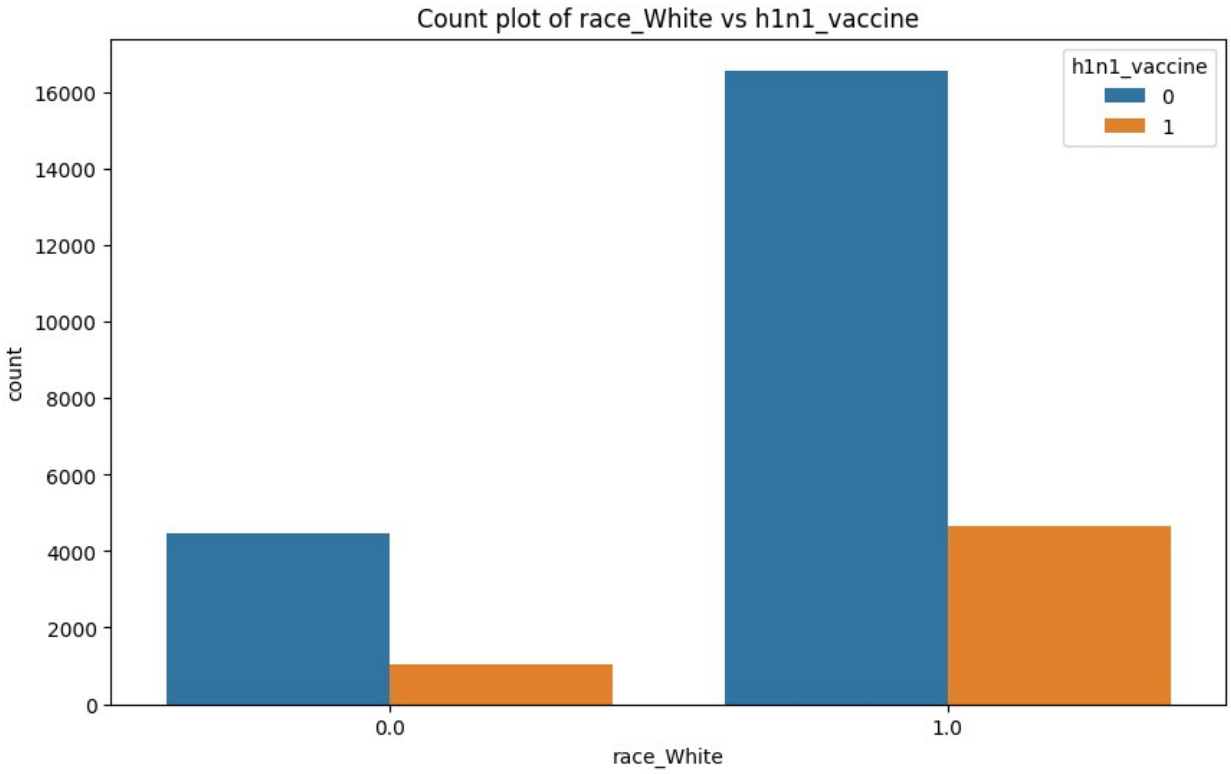

Count plot of education_College Graduate vs h1n1_vaccine
h1n1_vaccine
0
1
count
12000
10000
8000
6000
4000
2000
0
0.0
1.0
education_College Graduate

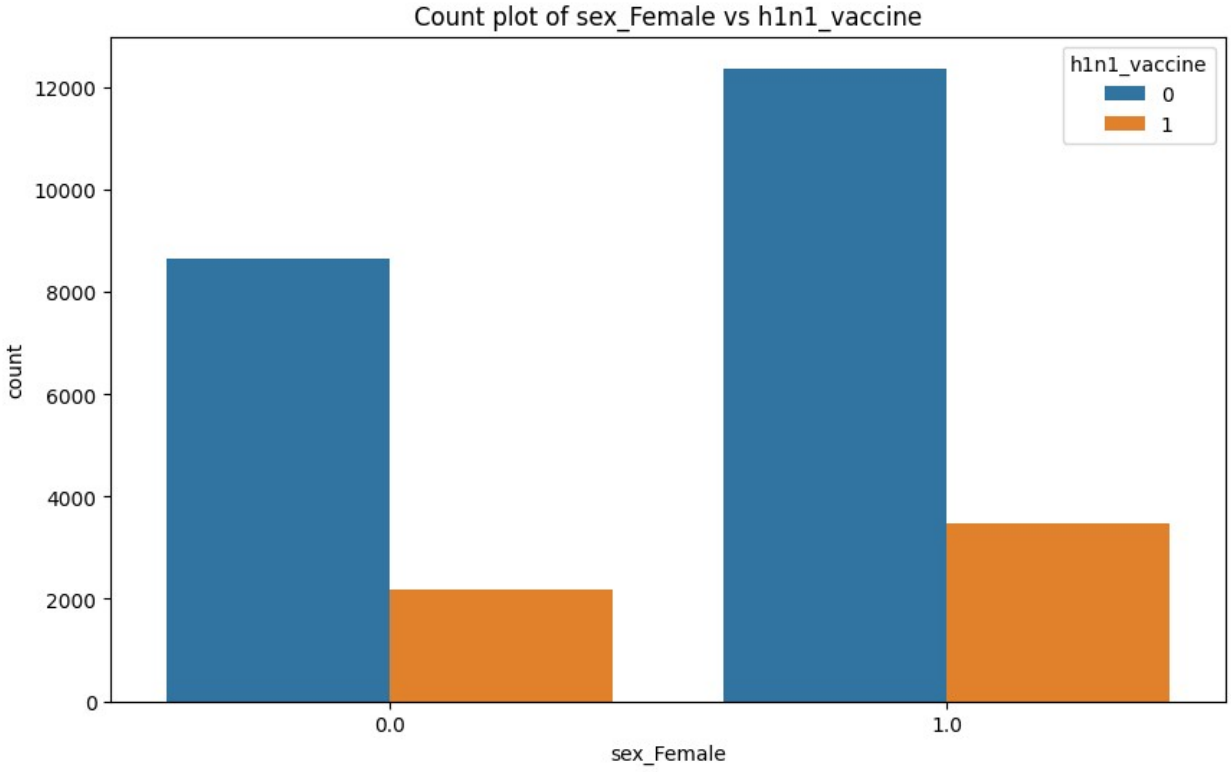

Count plot of education_Some College vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
education_Some College

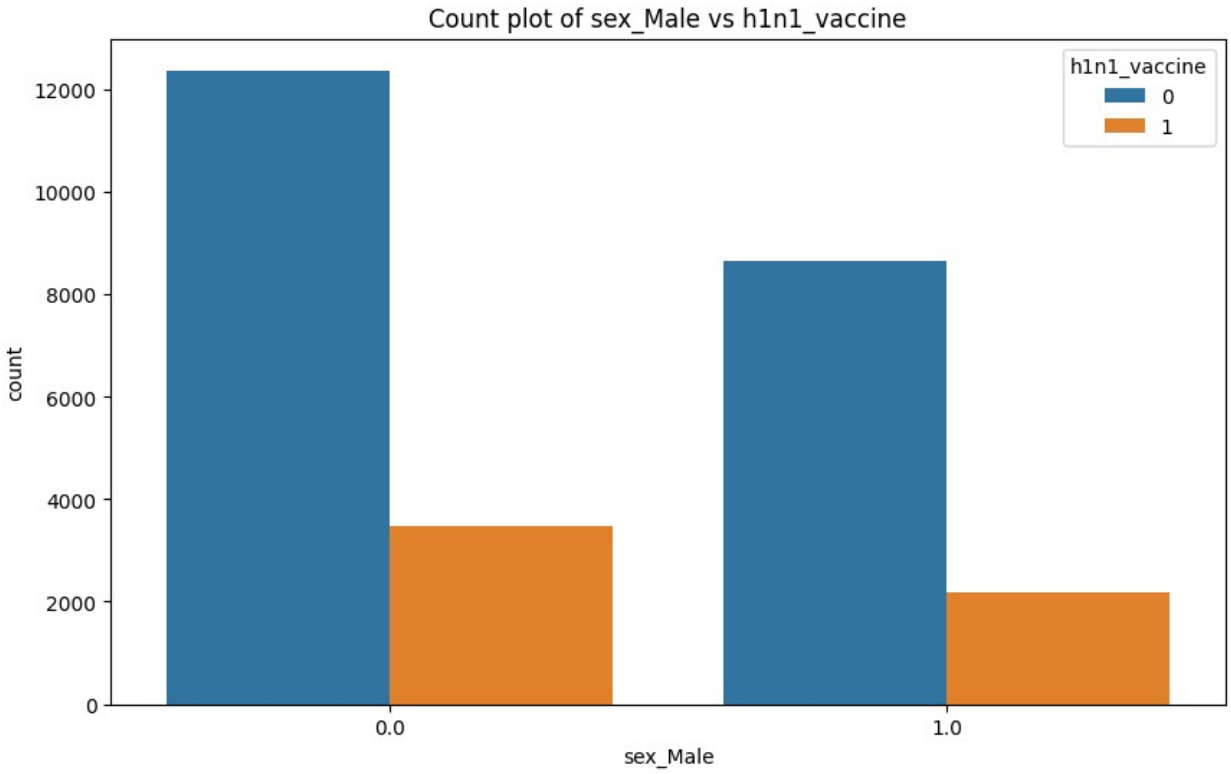

Count plot of race_Black vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
race_Black

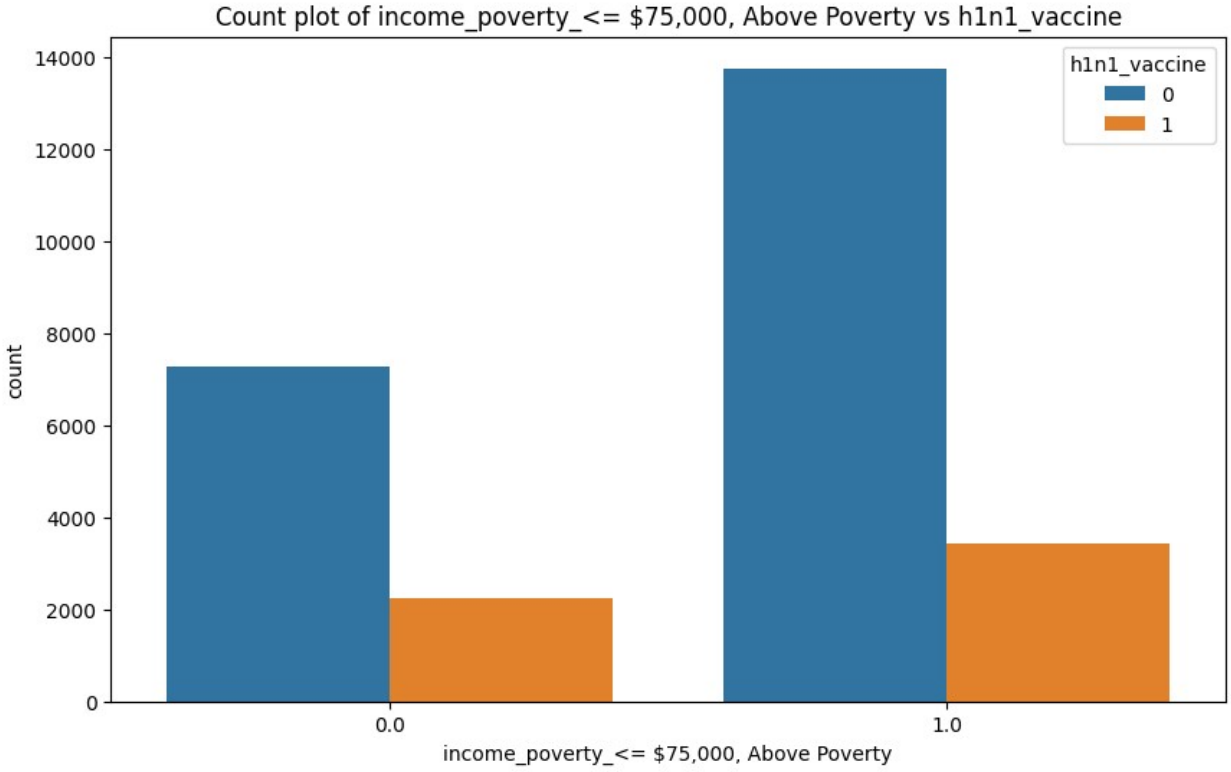

Count plot of race_Hispanic vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
race_Hispanic

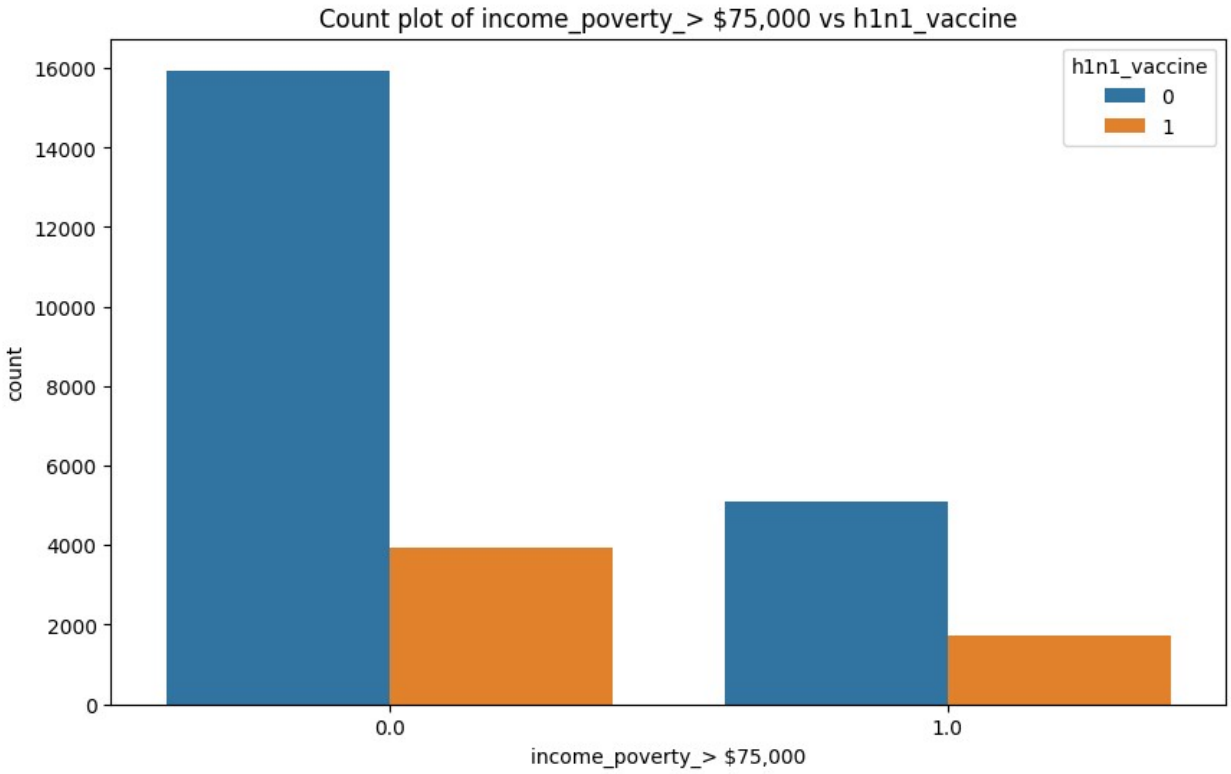

Count plot of race_Other or Multiple vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
race_Other or Multiple

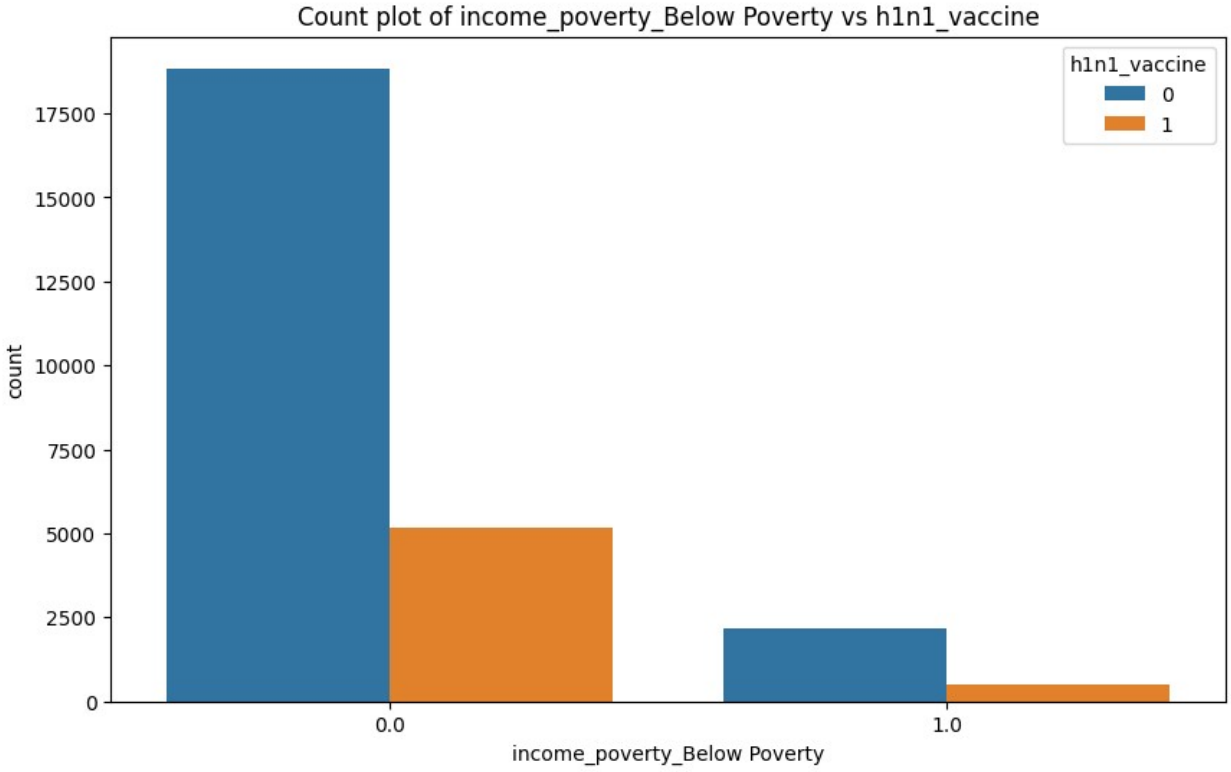

Count plot of race_White vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
race_White

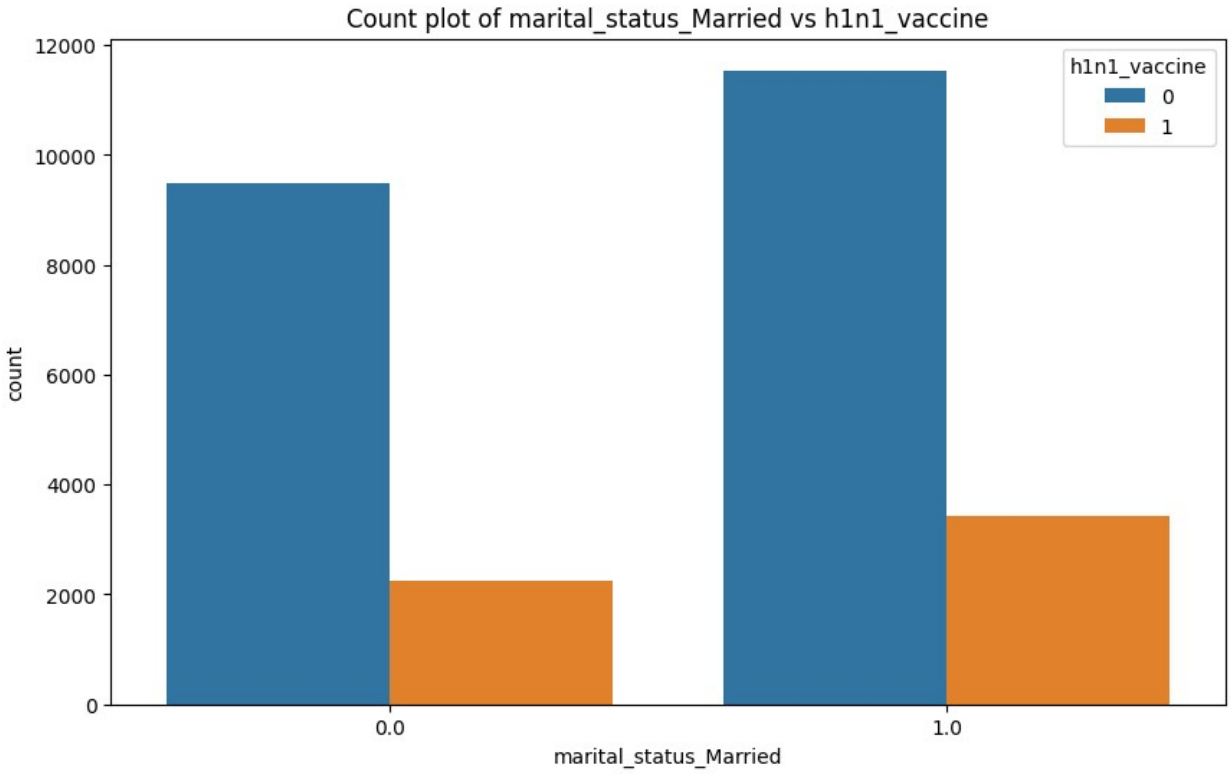

Count plot of sex_Female vs h1n1_vaccine
h1n1_vaccine
0
1
count
12000
10000
8000
6000
4000
2000
0
0.0
1.0
sex_Female

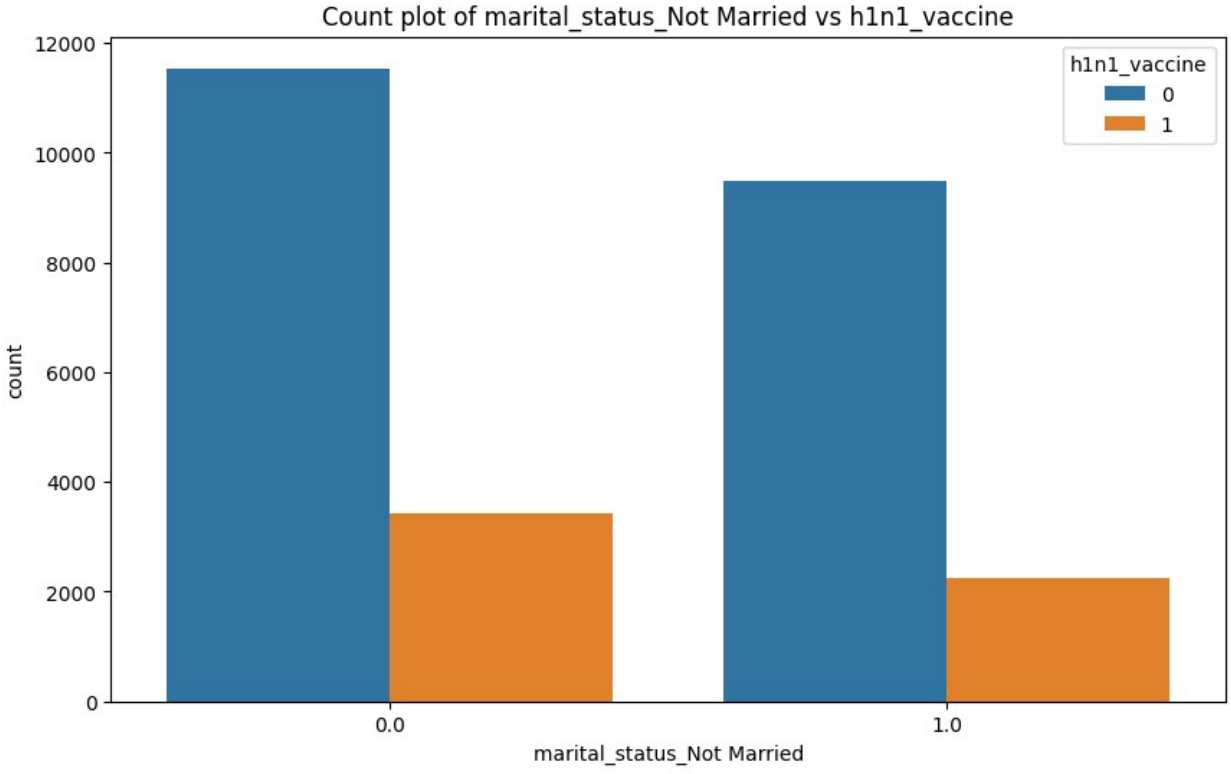

Count plot of sex_Male vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
sex_Male

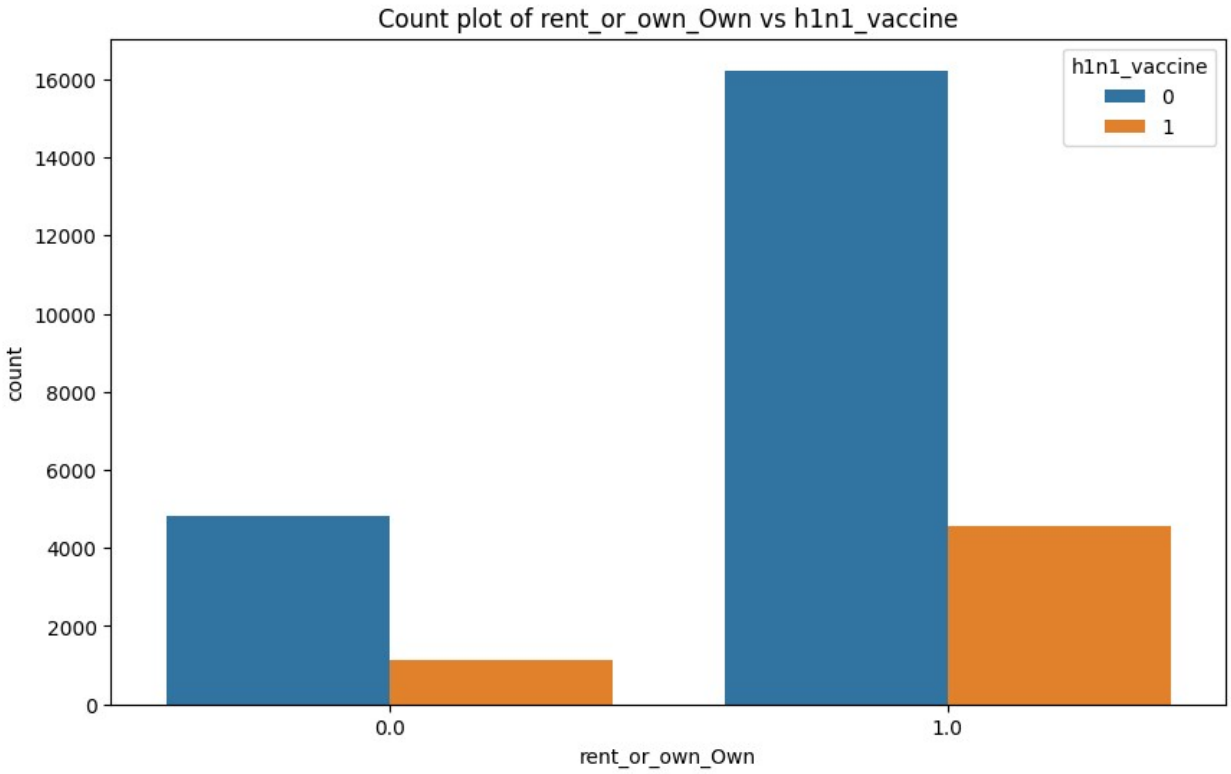

Count plot of income_poverty_<= $75,000, Above Poverty vs h1n1_vaccine
h1n1_vaccine
0
1
14000
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
income_poverty_<= $75,000, Above Poverty

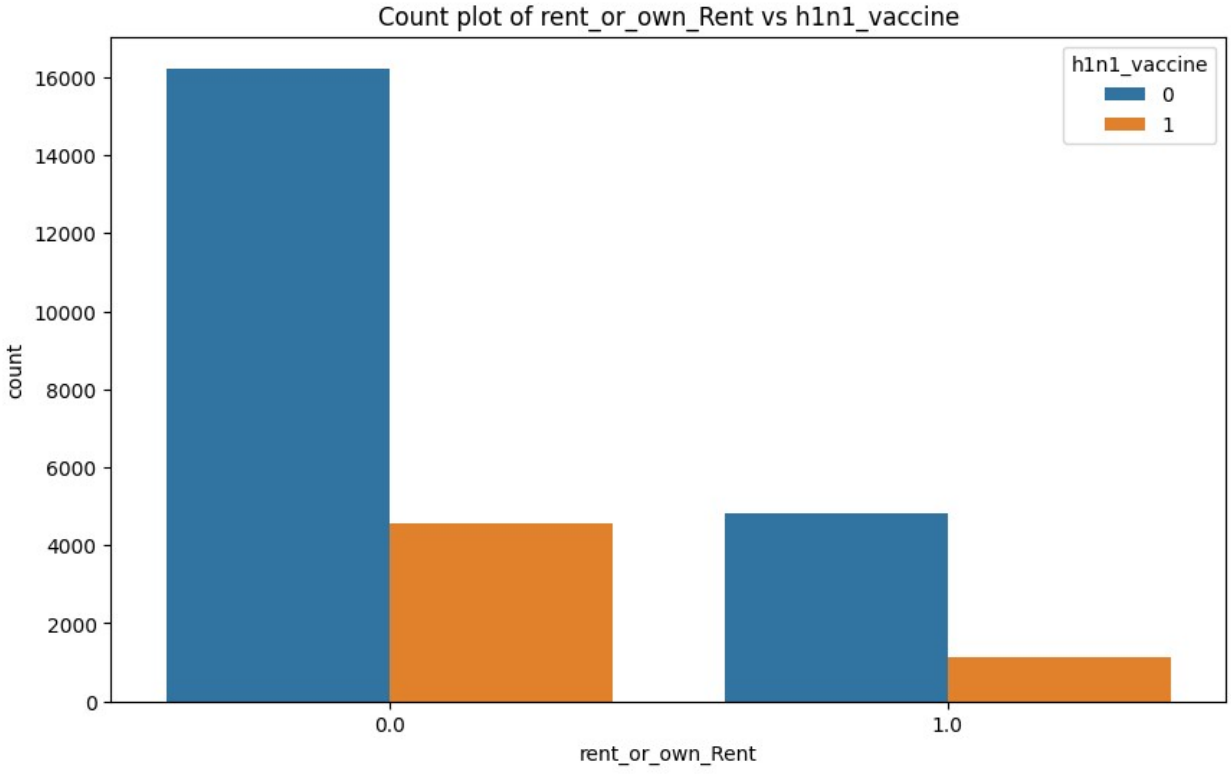

Count plot of income_poverty_> $75,000 vs h1n1_vaccine
h1n1_vaccine
0
1
count
16000
14000
12000
10000
8000
6000
4000
2000
0
0.0
1.0
income_poverty_> $75,000

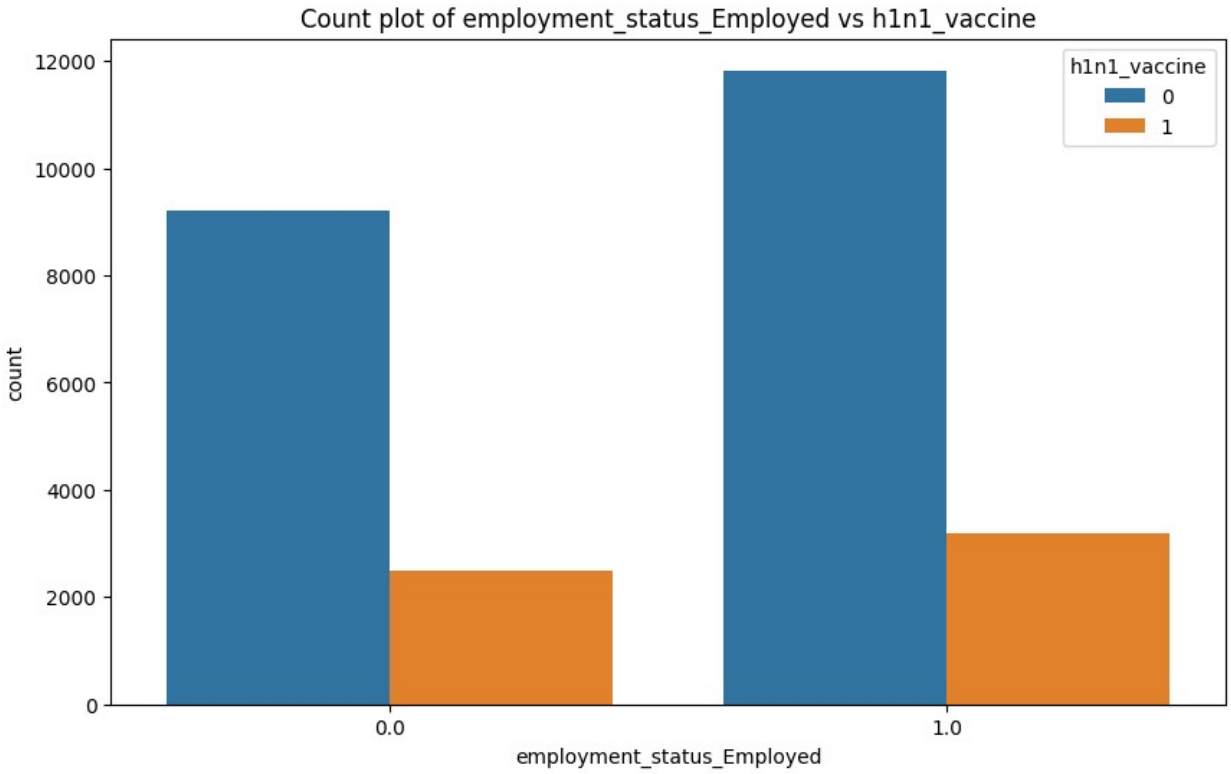

Count plot of income_poverty_Below Poverty vs h1n1_vaccine
h1n1_vaccine
0
1
count
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
income_poverty_Below Poverty

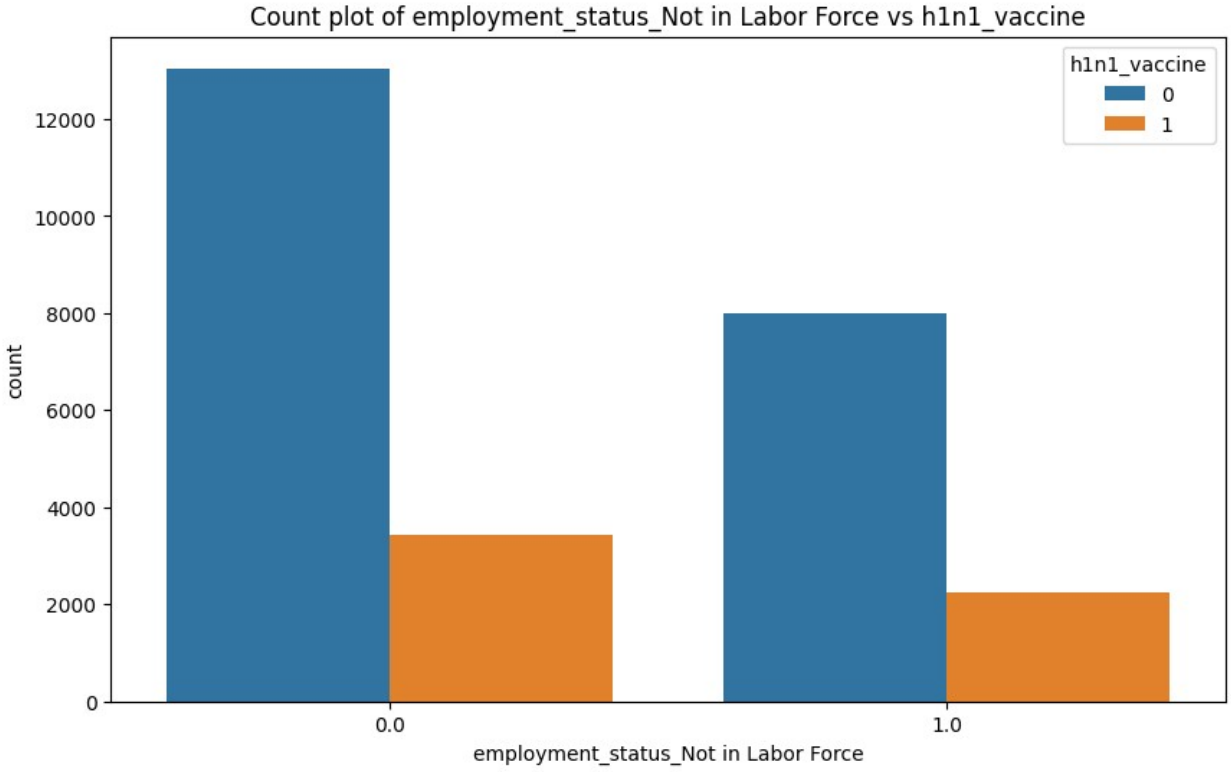

Count plot of marital_status_Married vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
marital_status_Married

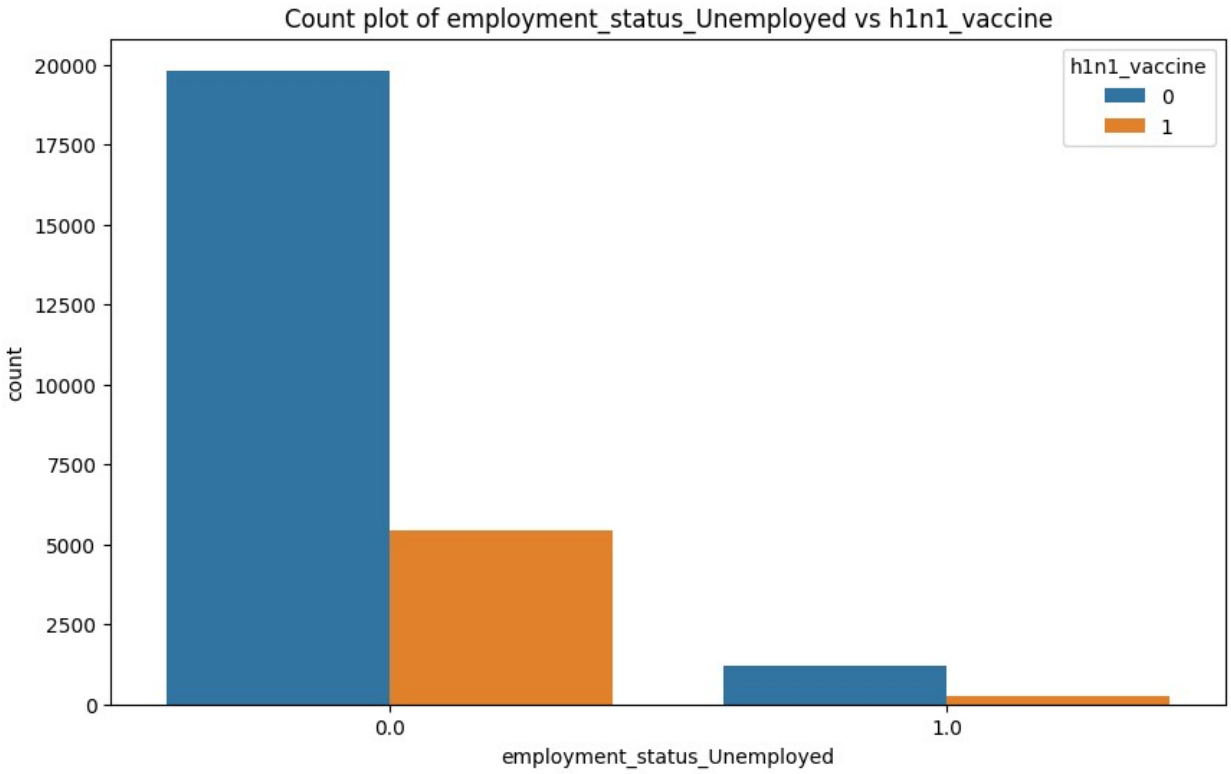

Count plot of marital_status_Not Married vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
marital_status_Not Married

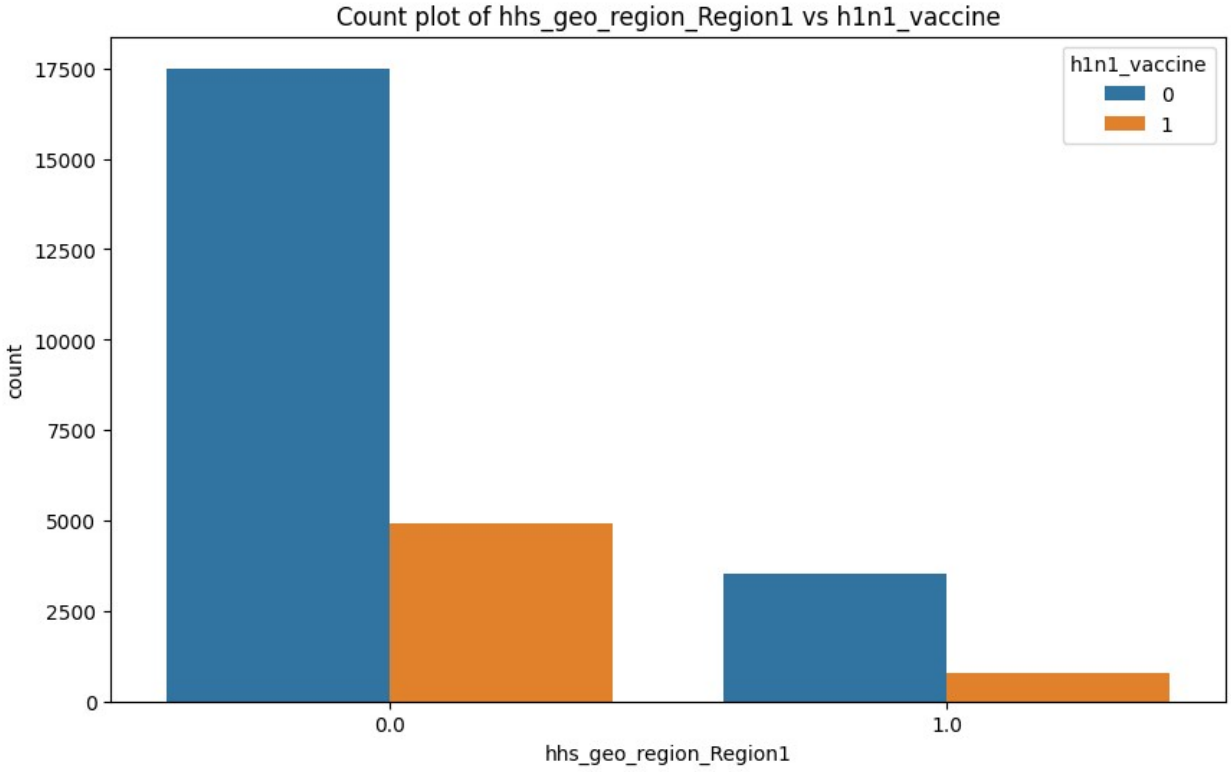

Count plot of rent_or_own_Own vs h1n1_vaccine
h1n1_vaccine
0
1
16000
14000
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
rent_or_own_Own

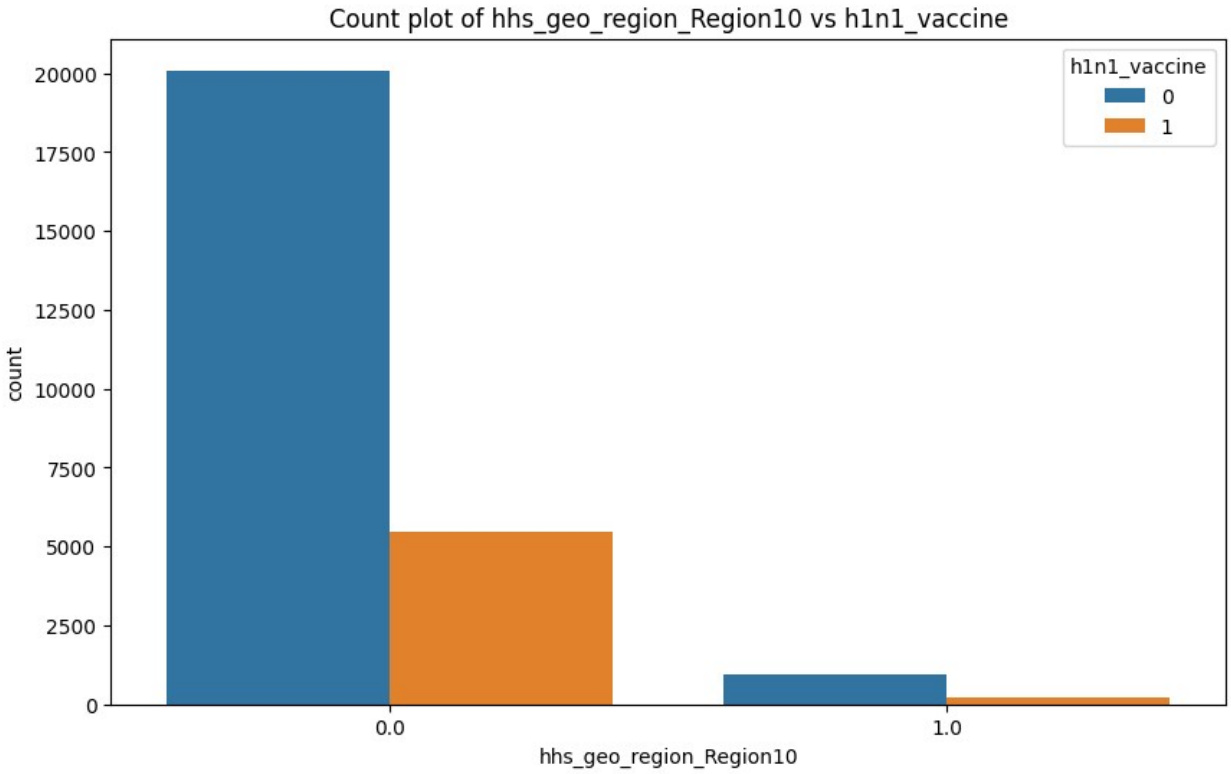

Count plot of rent_or_own_Rent vs h1n1_vaccine
h1n1_vaccine
0
1
16000
14000
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
rent_or_own_Rent

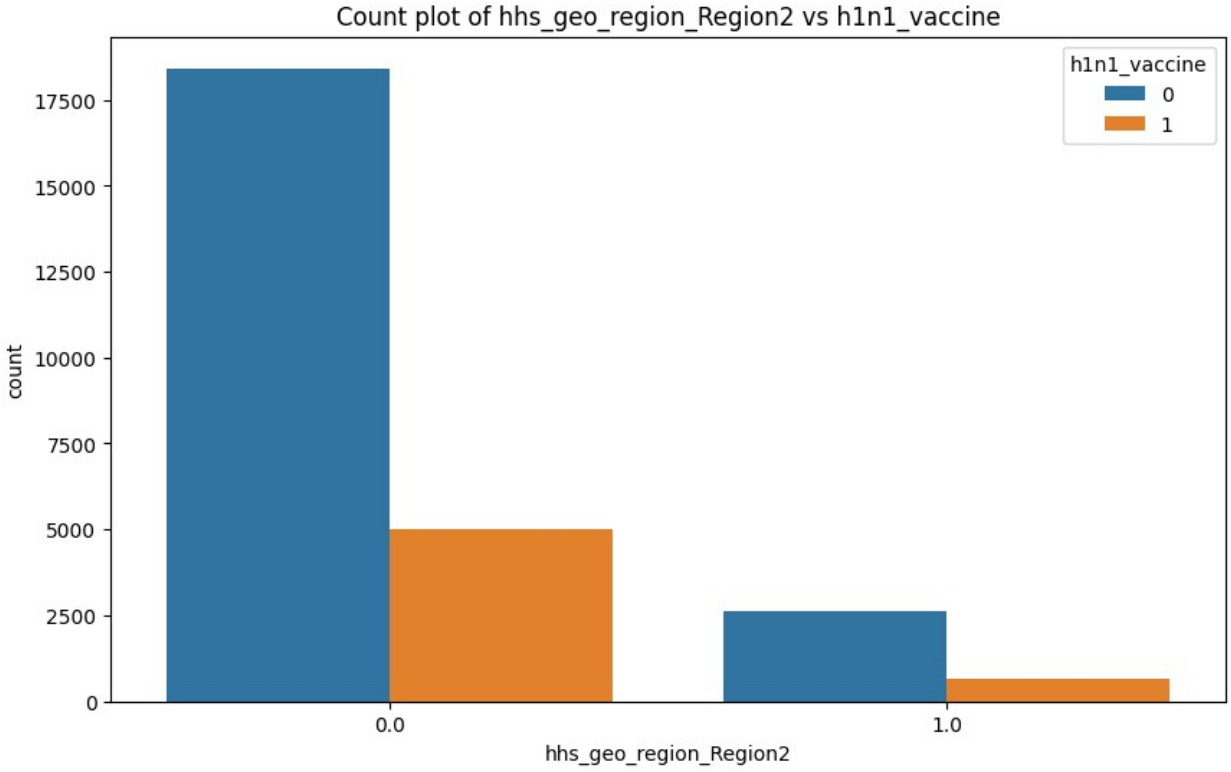

Count plot of employment_status_Employed vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
employment_status_Employed

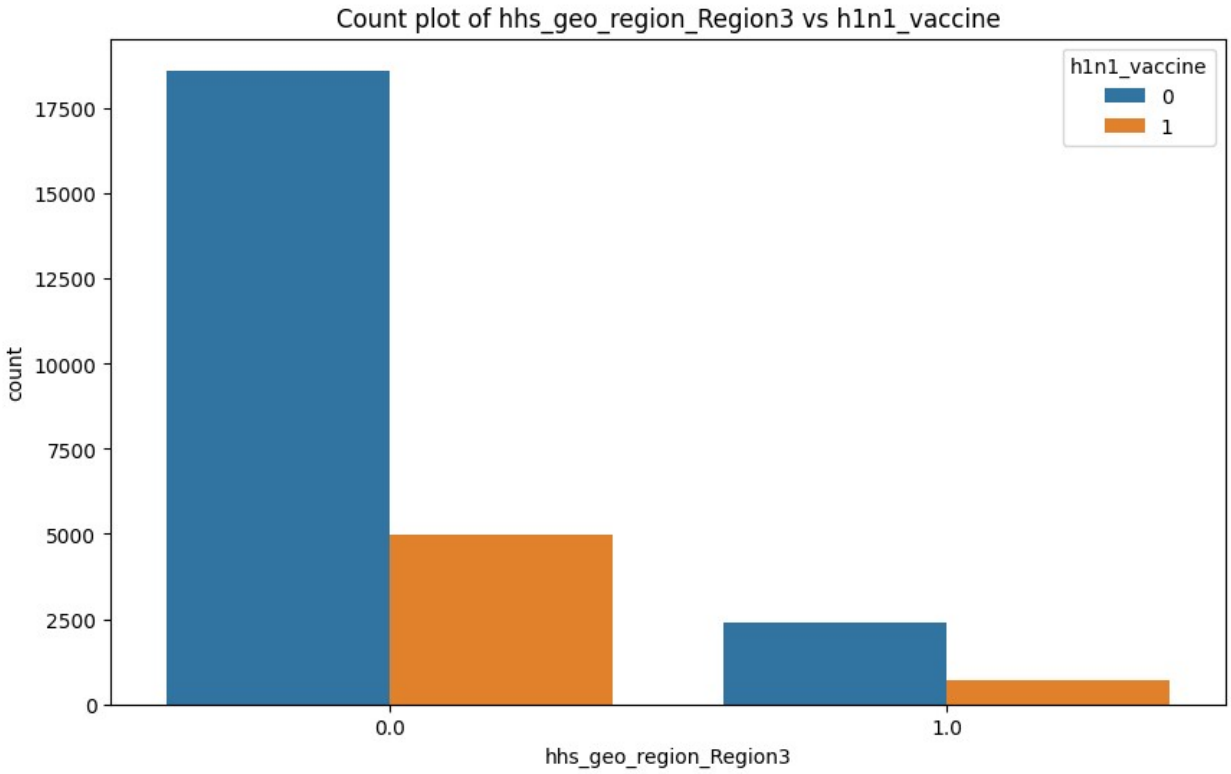

Count plot of employment_status_Not in Labor Force vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
employment_status_Not in Labor Force

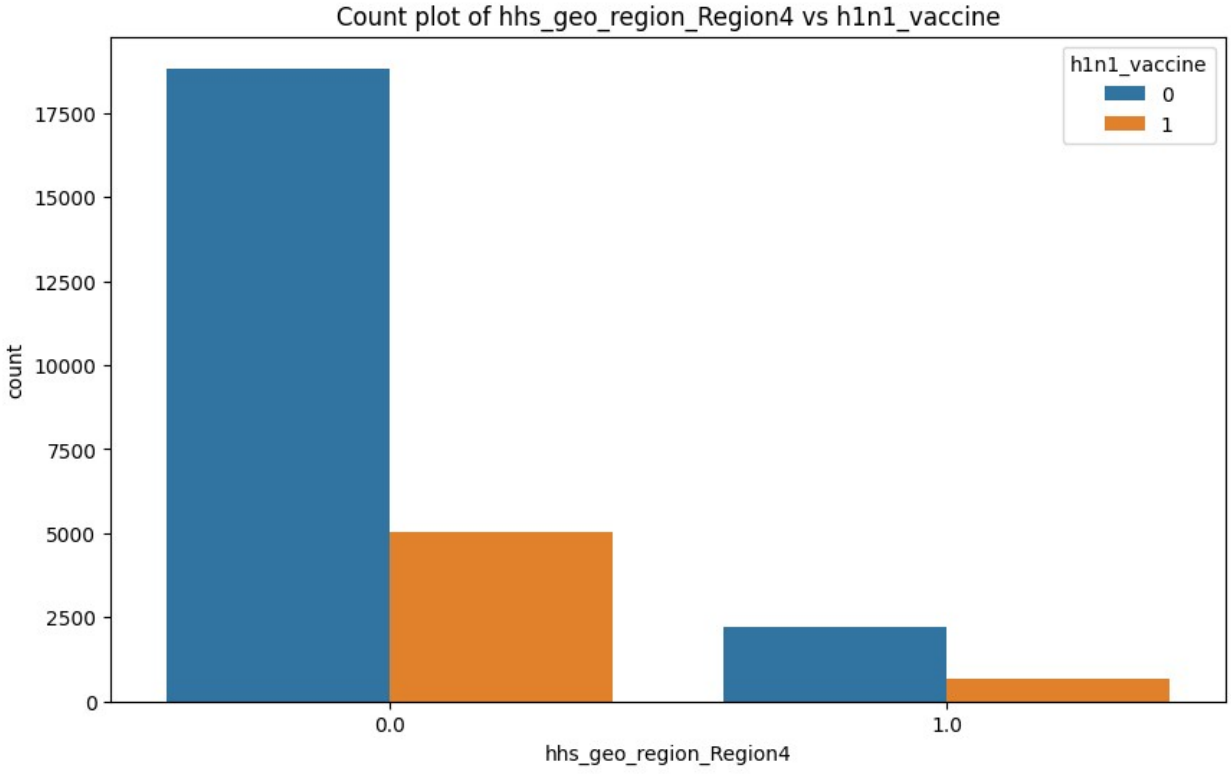

Count plot of employment_status_Unemployed vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_status_Unemployed

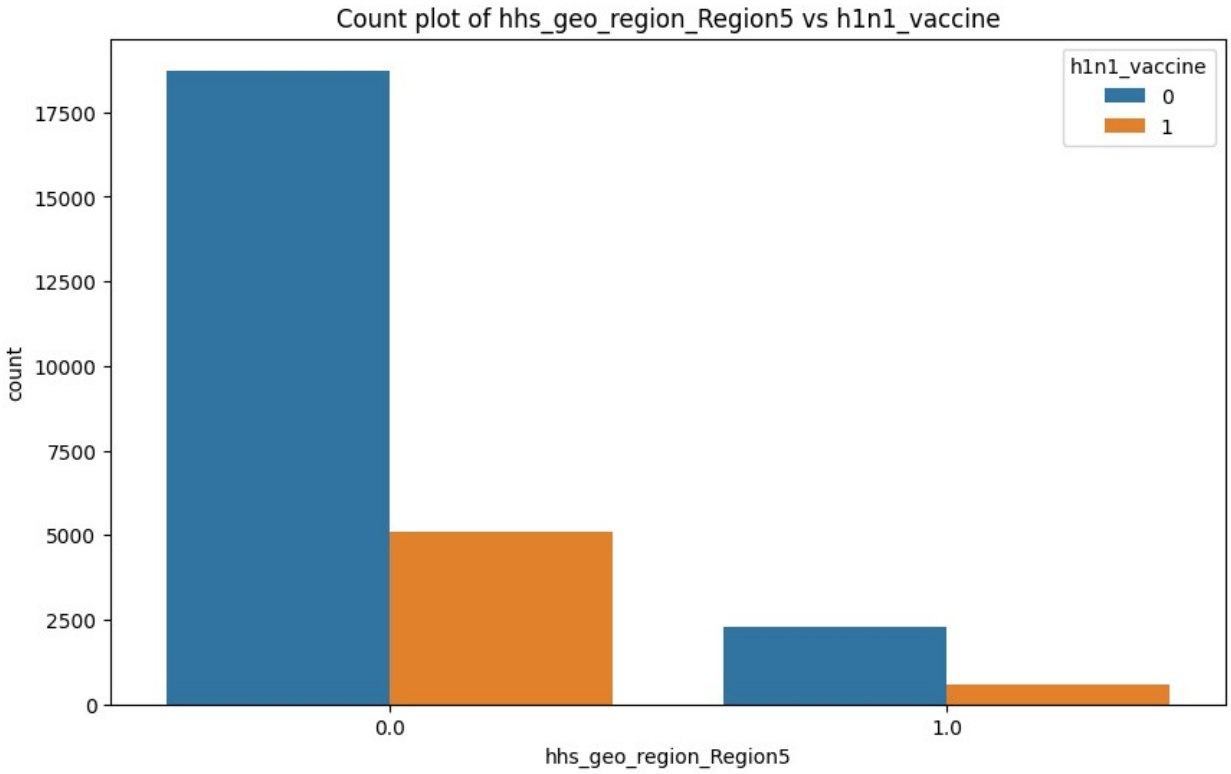

Count plot of hhs_geo_region_Region1 vs h1n1_vaccine
h1n1_vaccine
0
1
count
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region1

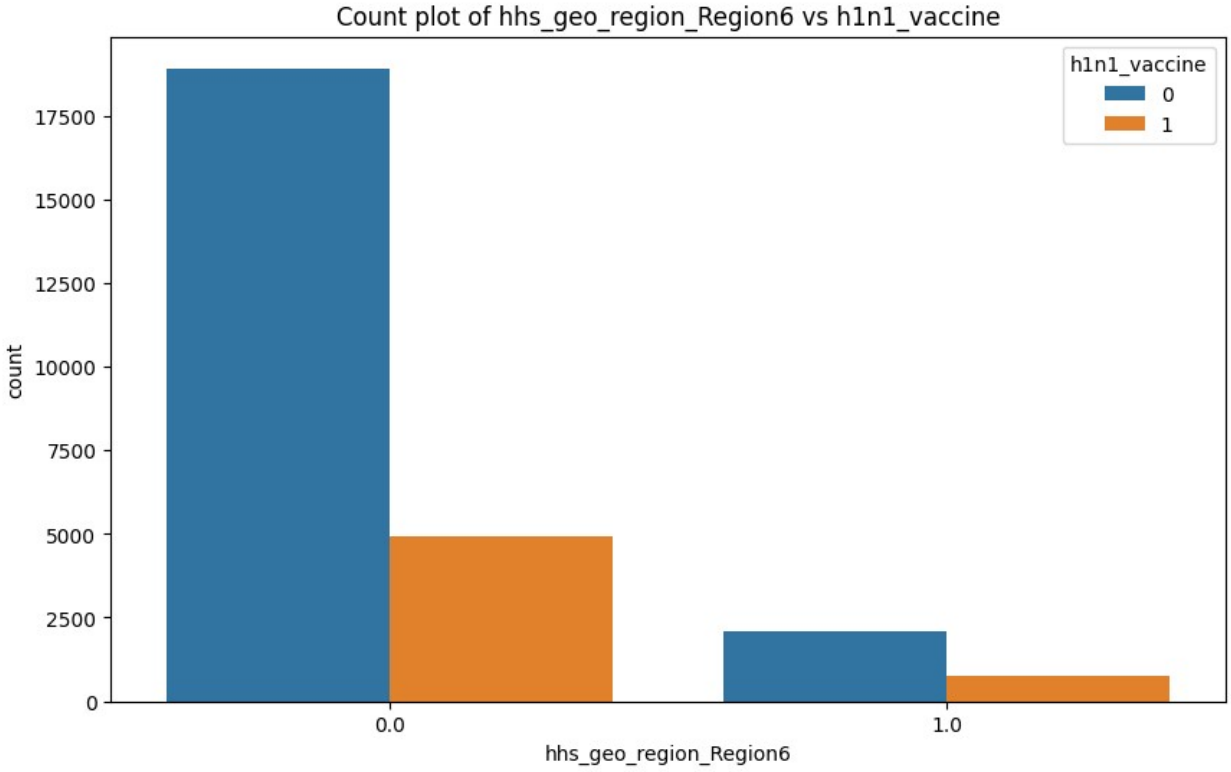

Count plot of hhs_geo_region_Region10 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region10

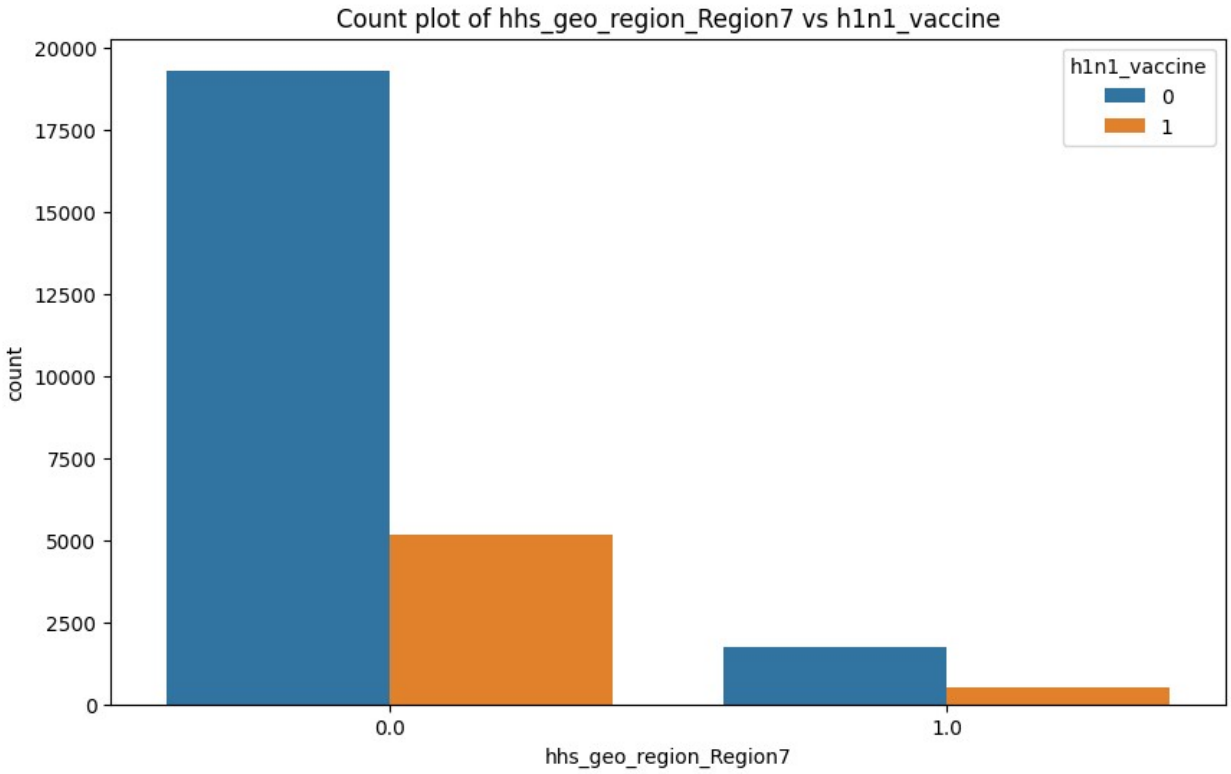

Count plot of hhs_geo_region_Region2 vs h1n1_vaccine
h1n1_vaccine
0
1
count
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region2

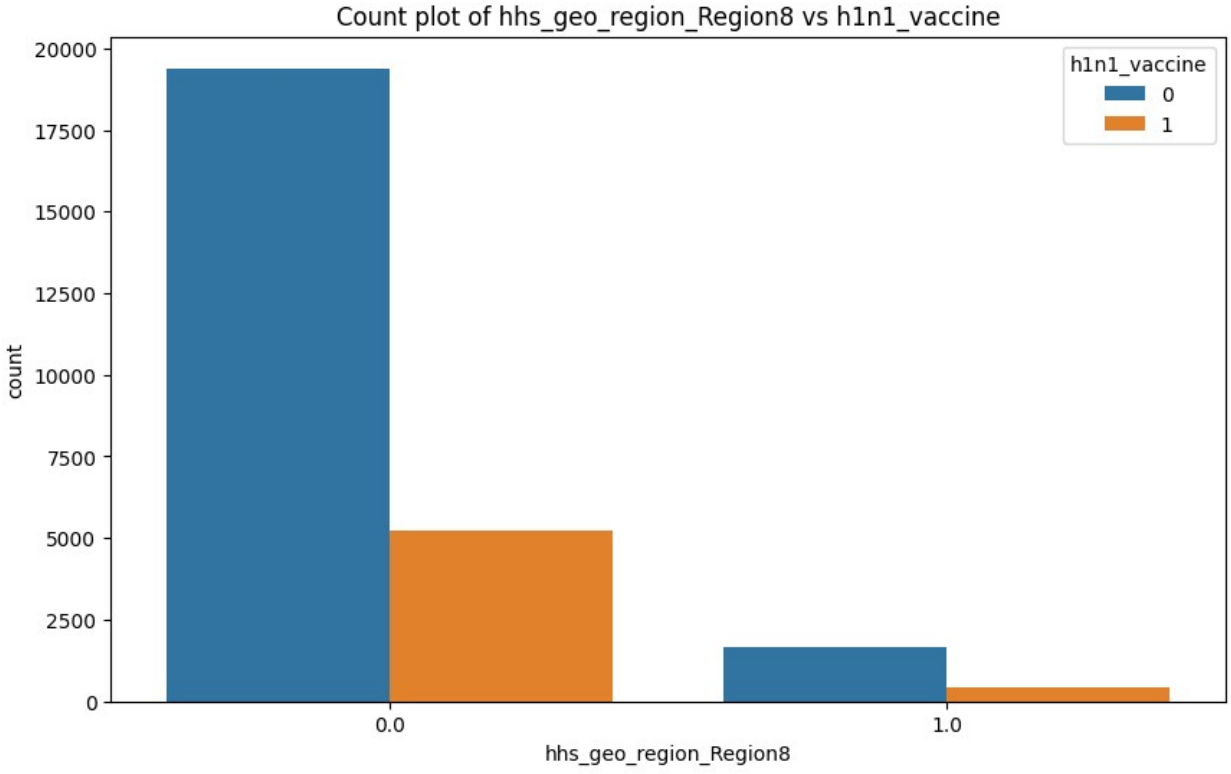

Count plot of hhs_geo_region_Region3 vs h1n1_vaccine
h1n1_vaccine
0
1
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
hhs_geo_region_Region3

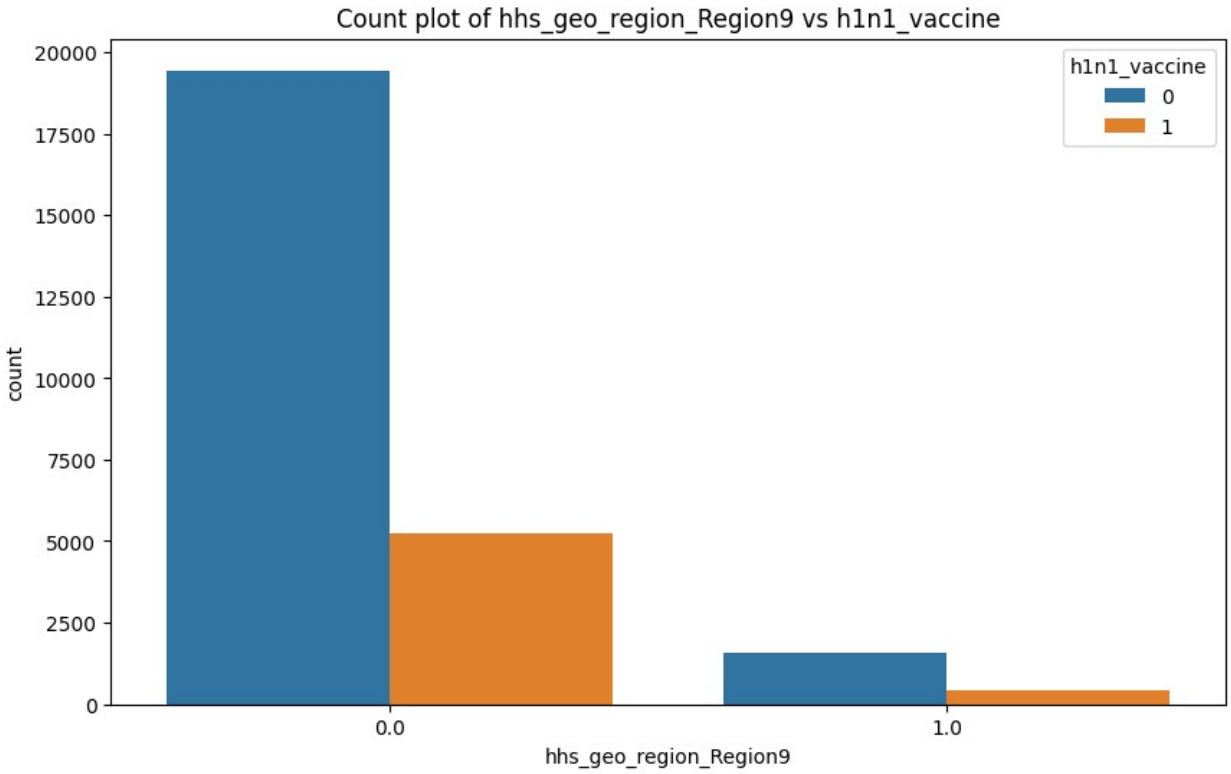

Count plot of hhs_geo_region_Region4 vs h1n1_vaccine
h1n1_vaccine
0
1
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
hhs_geo_region_Region4


Count plot of hhs_geo_region_Region5 vs h1n1_vaccine
h1n1_vaccine
0
1
count
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region5

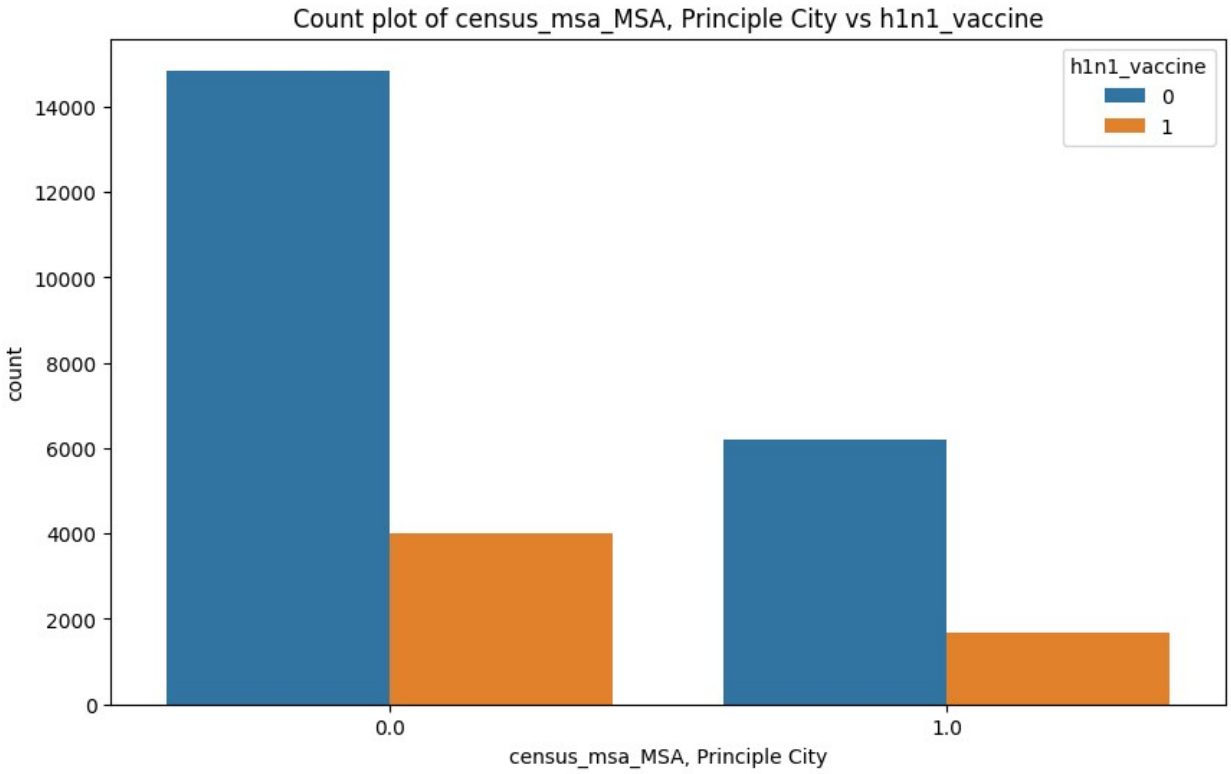

Count plot of hhs_geo_region_Region6 vs h1n1_vaccine
h1n1_vaccine
0
1
count
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region6

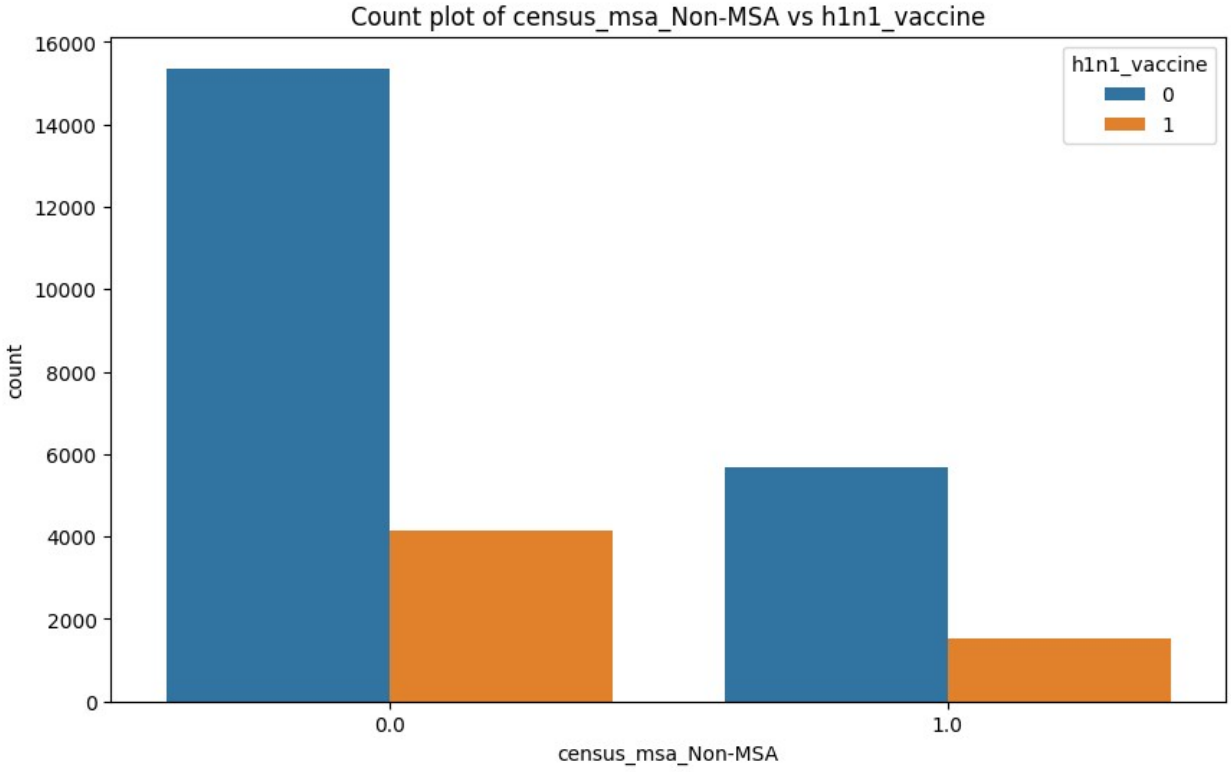

Count plot of hhs_geo_region_Region7 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region7

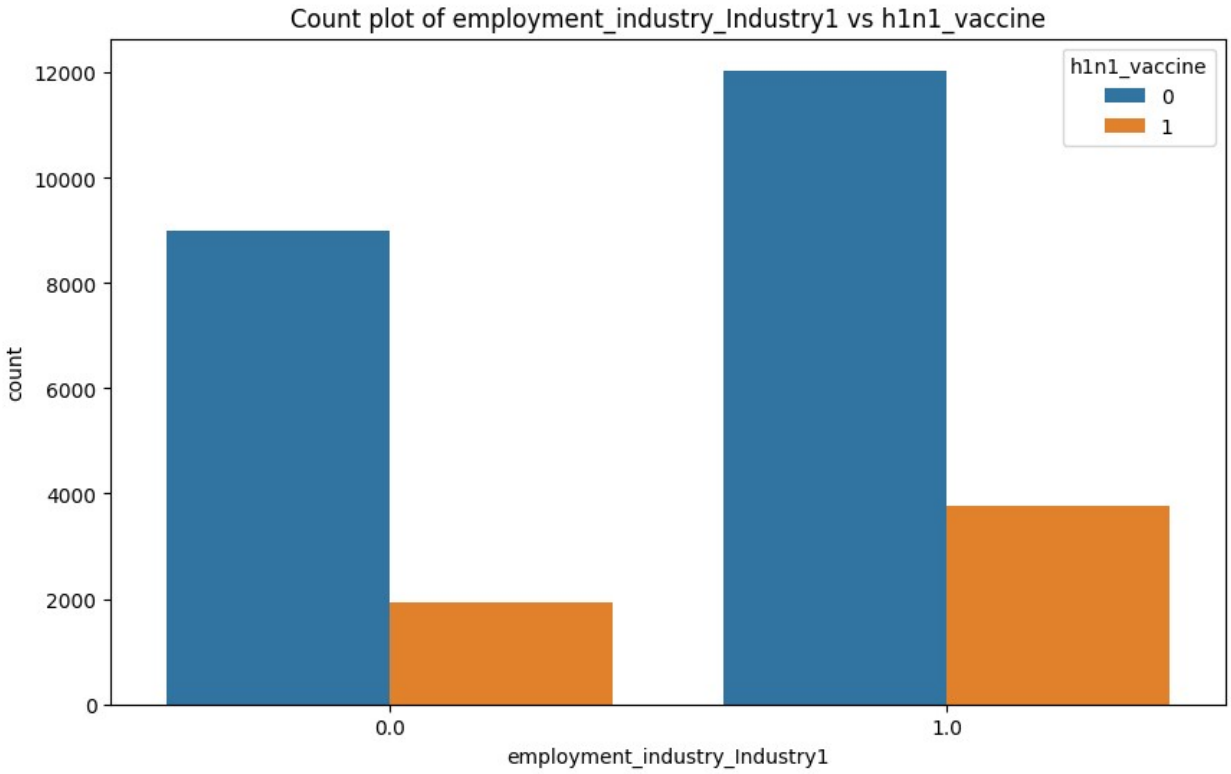

Count plot of hhs_geo_region_Region8 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region8

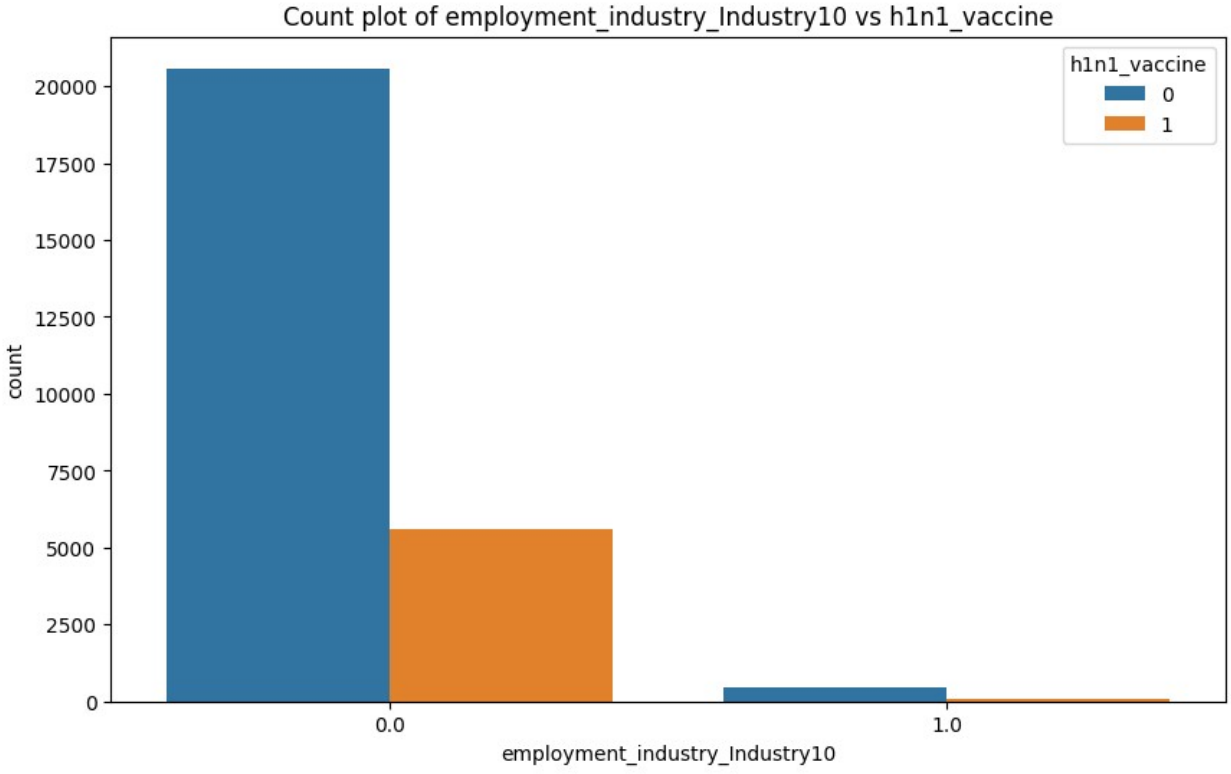

Count plot of hhs_geo_region_Region9 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
hhs_geo_region_Region9

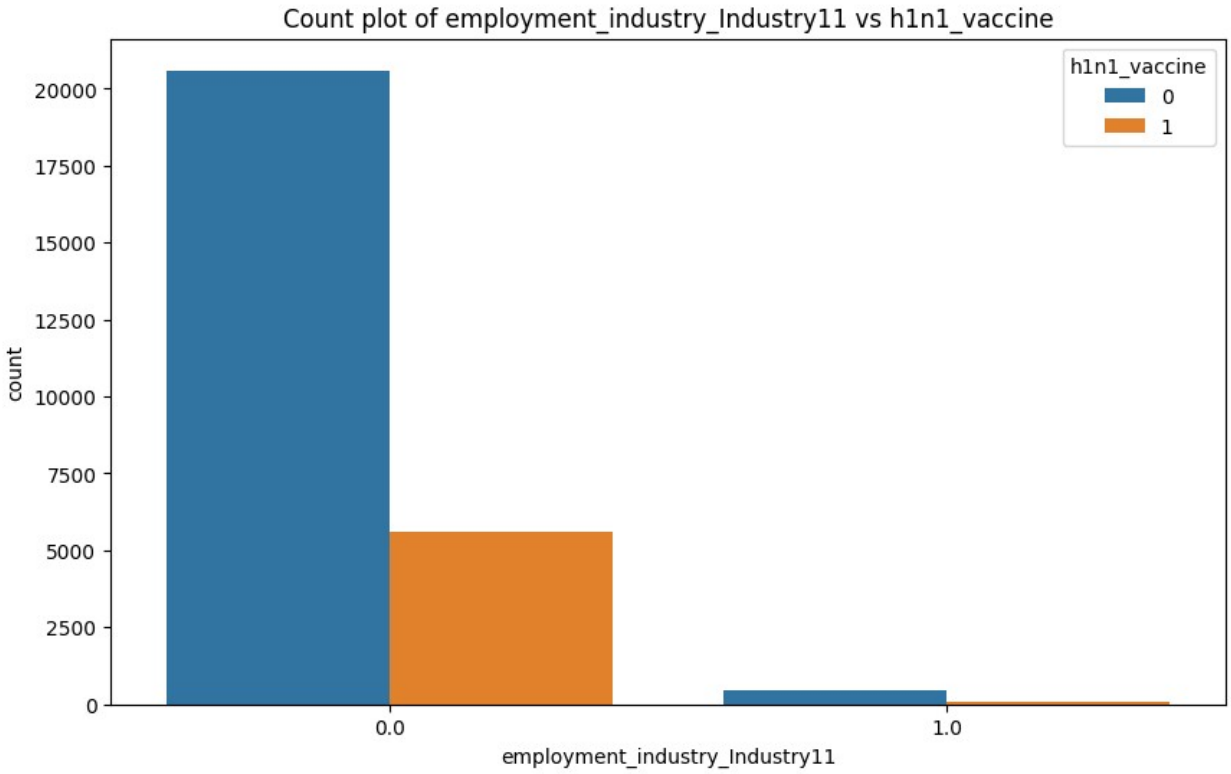

Count plot of census_msa_MSA, Not Principle City vs h1n1_vaccine
h1n1_vaccine
0
1
count
12000
10000
8000
6000
4000
2000
0
0.0
1.0
census_msa_MSA, Not Principle City

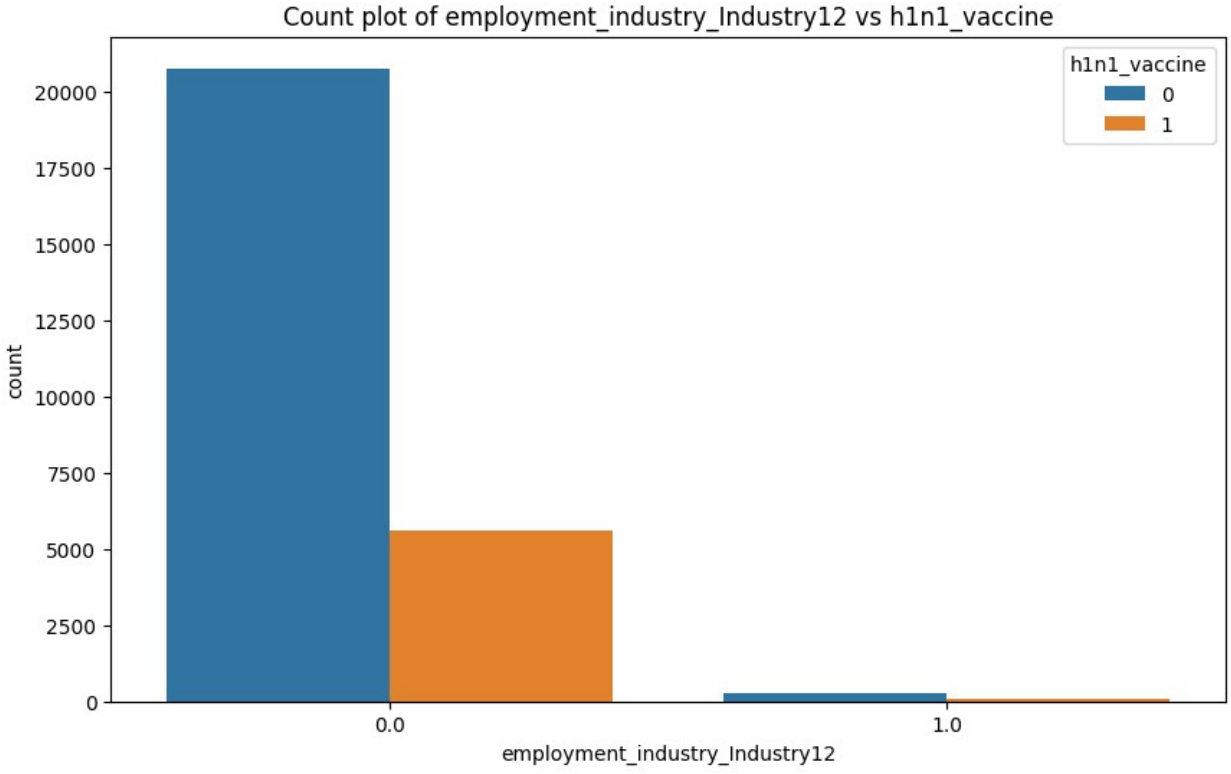

Count plot of census_msa_MSA, Principle City vs h1n1_vaccine
h1n1_vaccine
0
1
count
0
2000
4000
6000
8000
10000
12000
14000
0.0
1.0
census_msa_MSA, Principle City

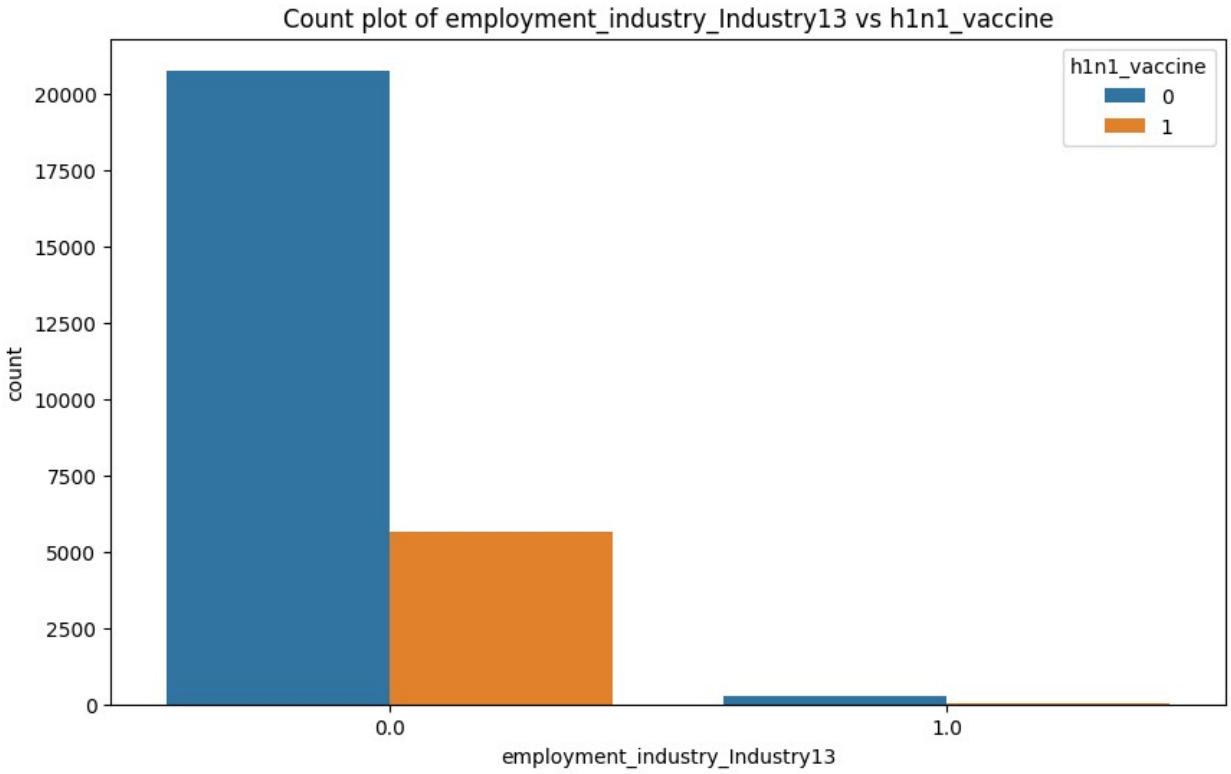

Count plot of census_msa_Non-MSA vs h1n1_vaccine
h1n1_vaccine
0
1
count
0
2000
4000
6000
8000
10000
12000
14000
16000
0.0
1.0
census_msa_Non-MSA

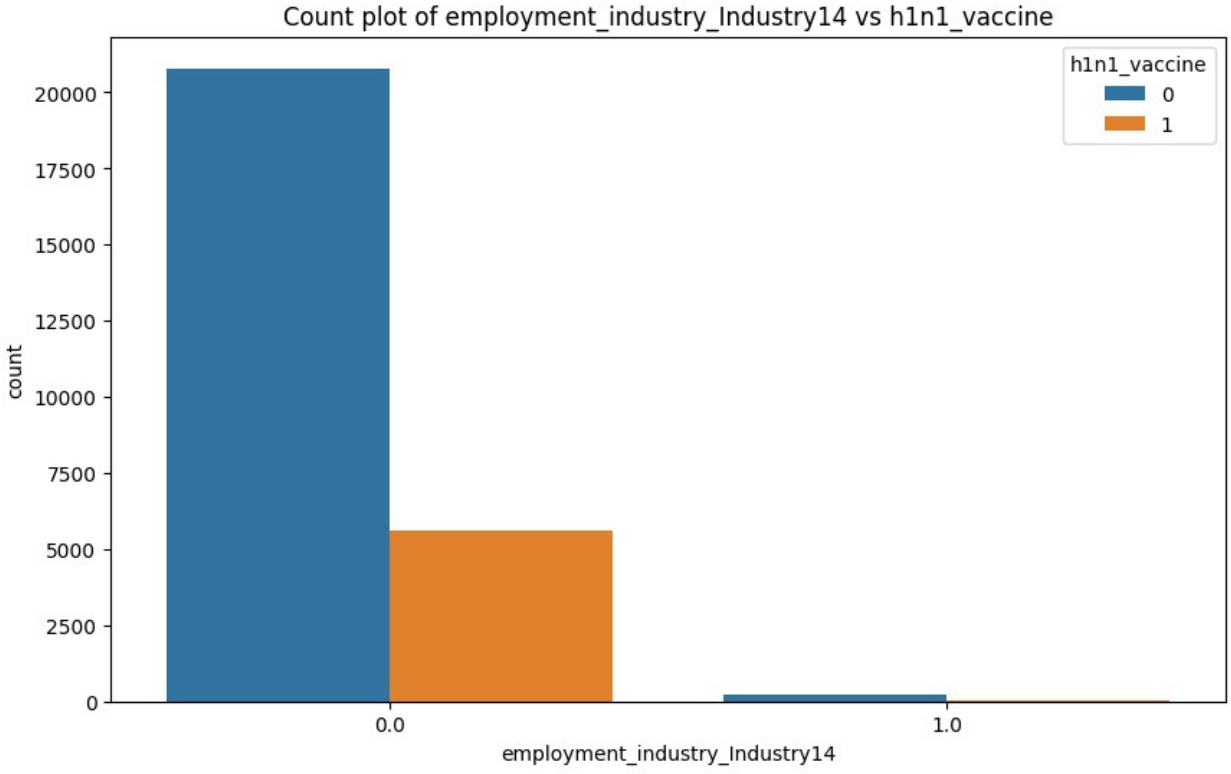

Count plot of employment_industry_Industry1 vs h1n1_vaccine
h1n1_vaccine
0
1
count
0
2000
4000
6000
8000
10000
12000
0.0
1.0
employment_industry_Industry1

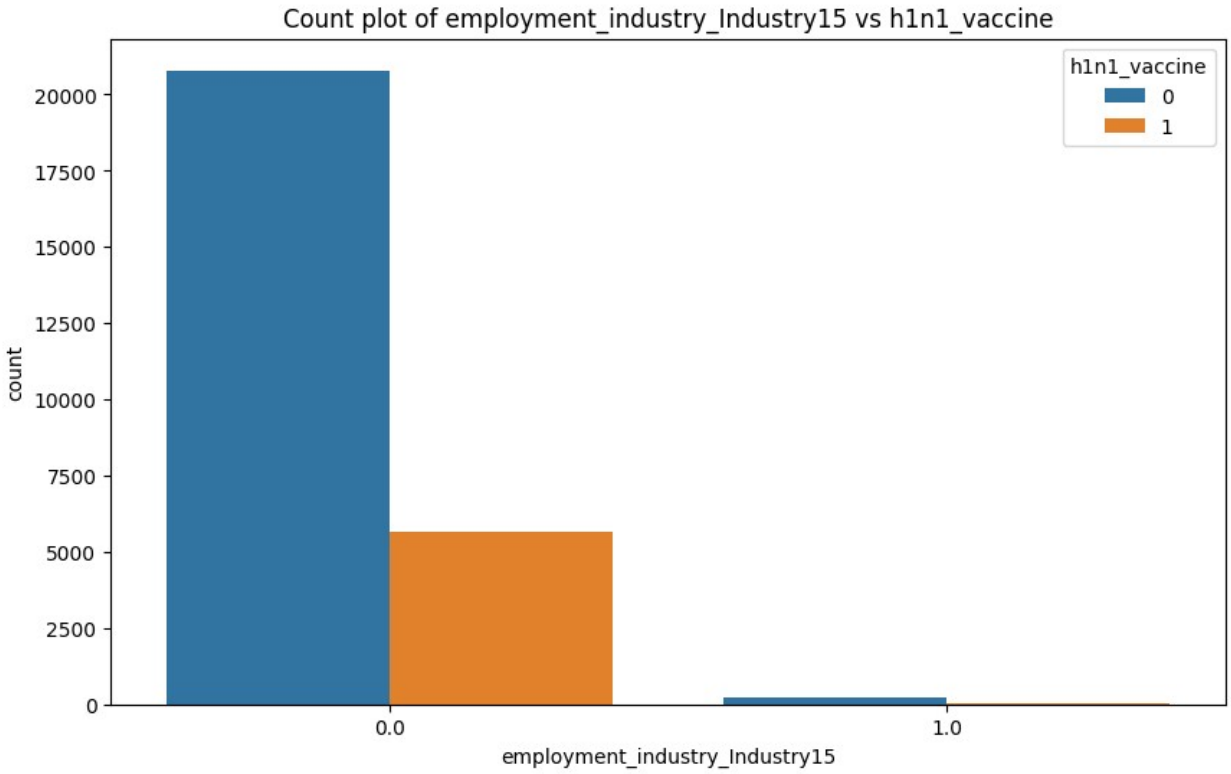

Count plot of employment_industry_Industry10 vs h1n1_vaccine
h1n1_vaccine
0
1
count
0
2500
5000
7500
10000
12500
15000
17500
20000
0.0
1.0
employment_industry_Industry10

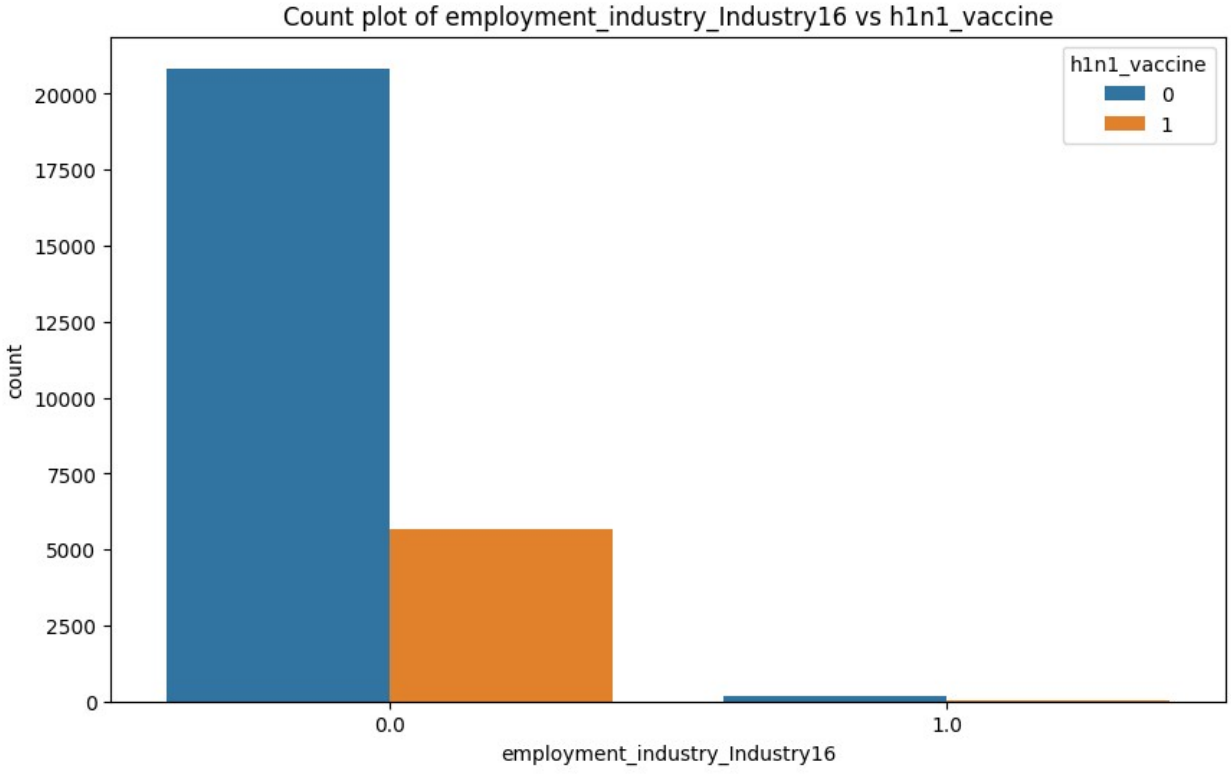

Count plot of employment_industry_Industry11 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry11

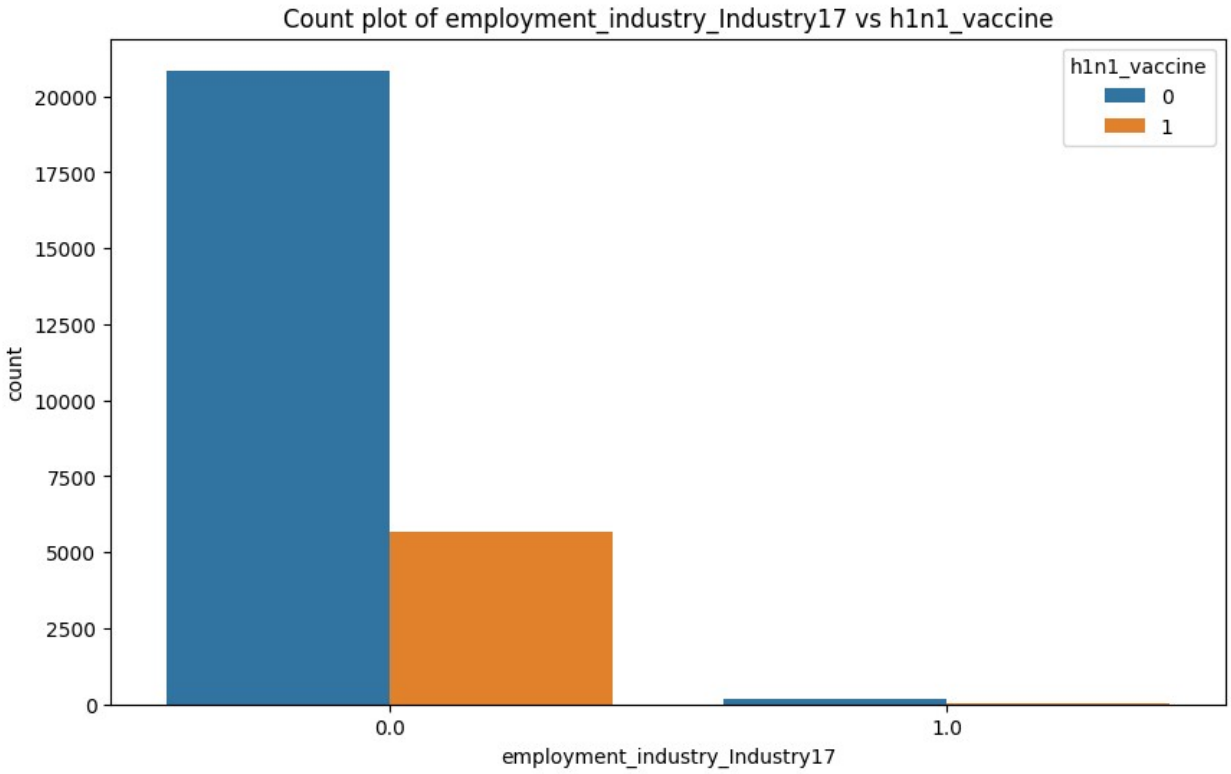

Count plot of employment_industry_Industry12 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry12

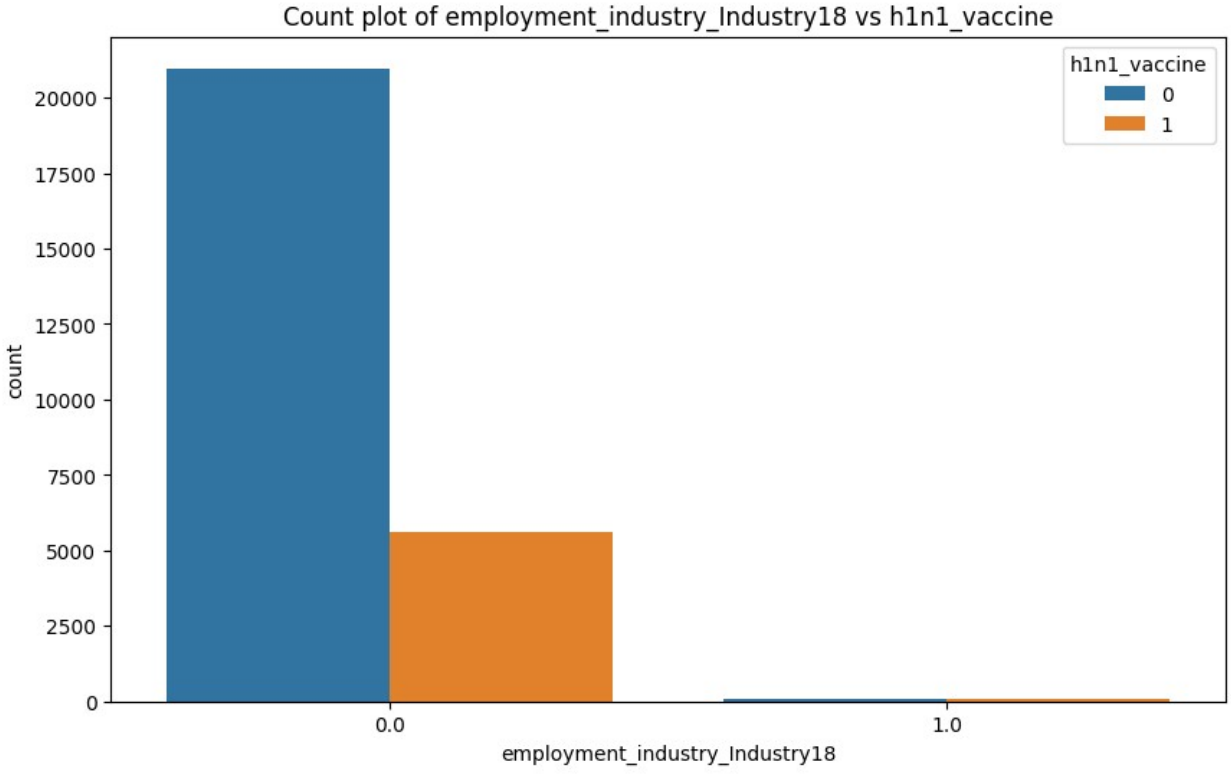

Count plot of employment_industry_Industry13 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry13

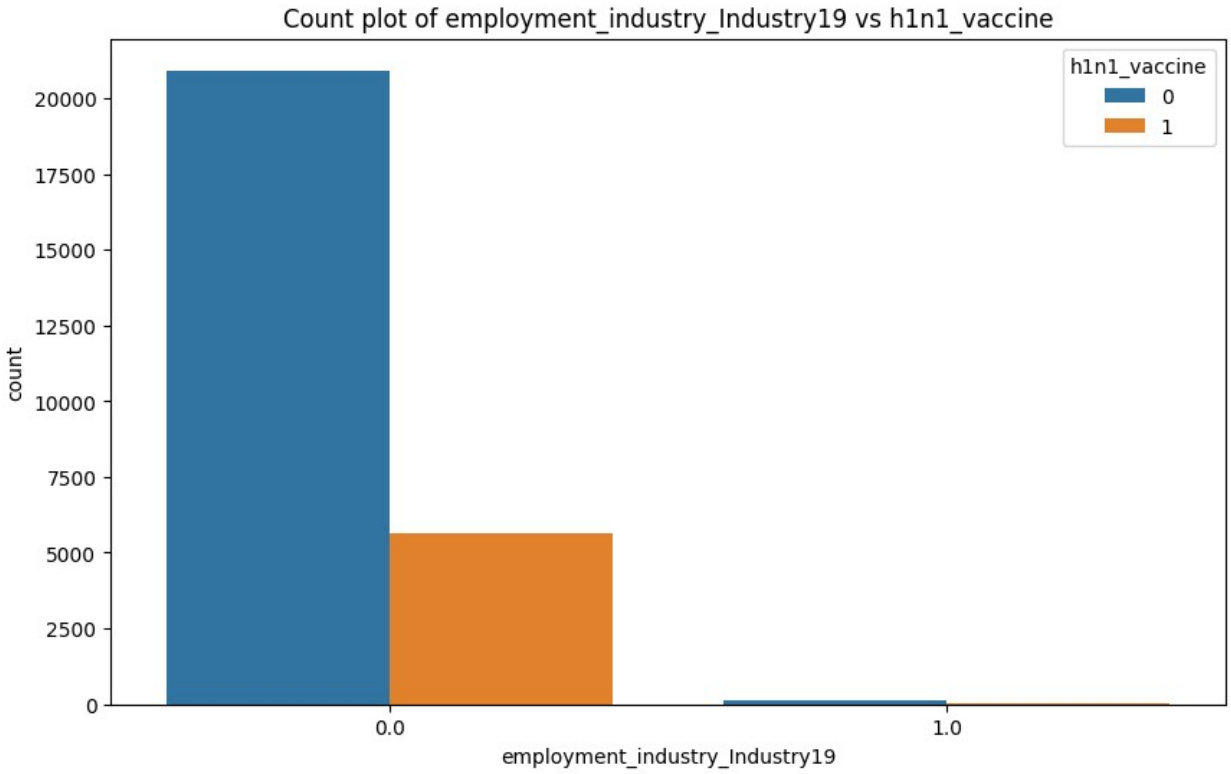

Count plot of employment_industry_Industry14 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry14

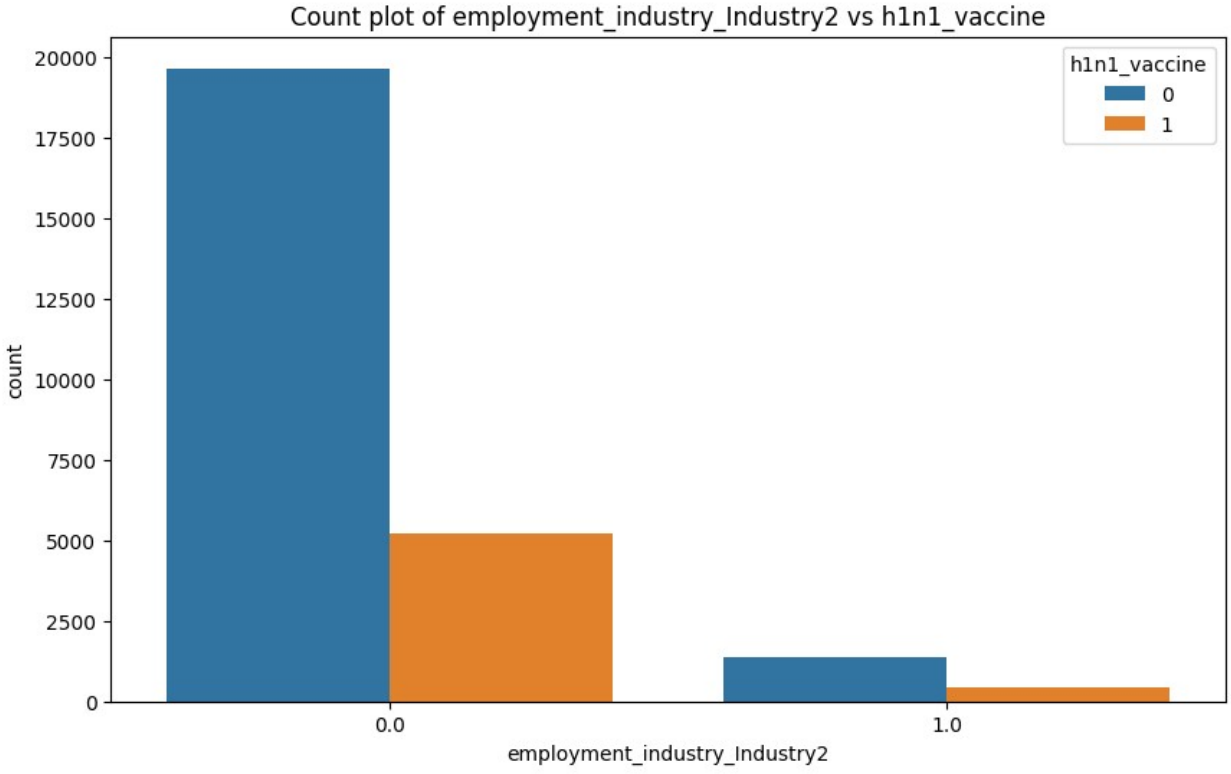

Count plot of employment_industry_Industry15 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry15

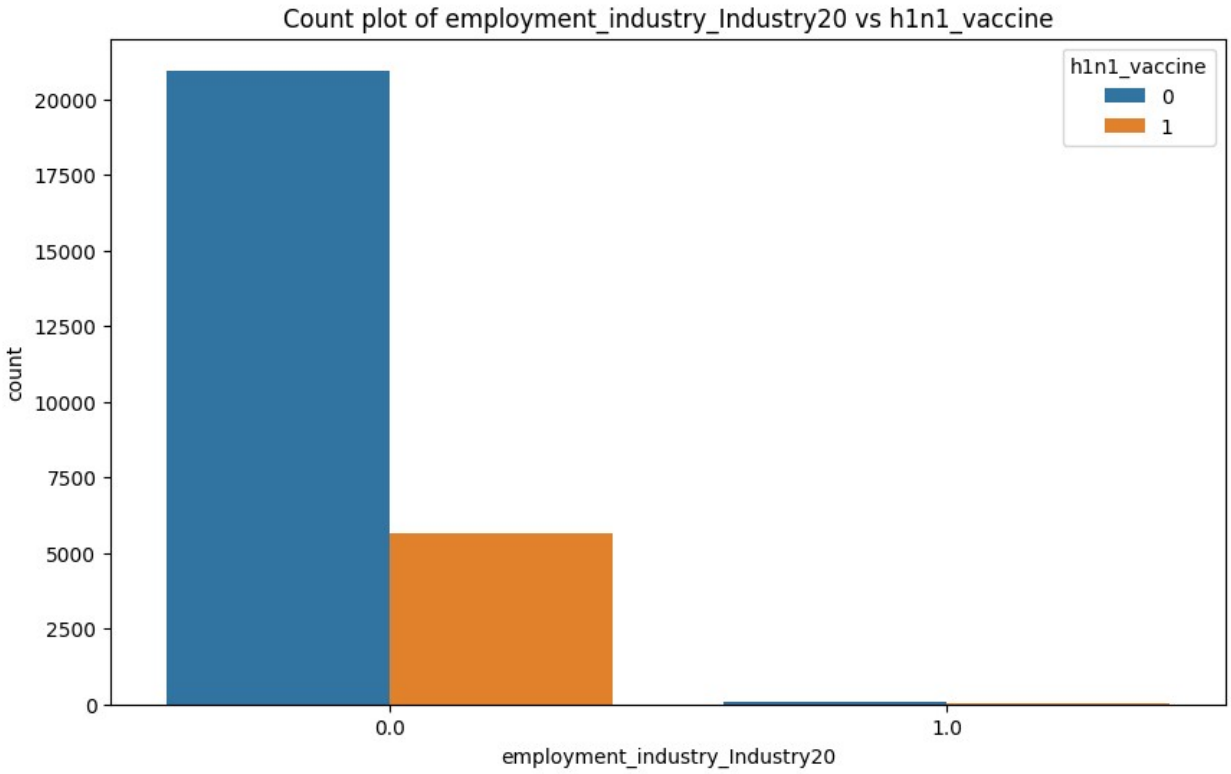

Count plot of employment_industry_Industry16 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry16

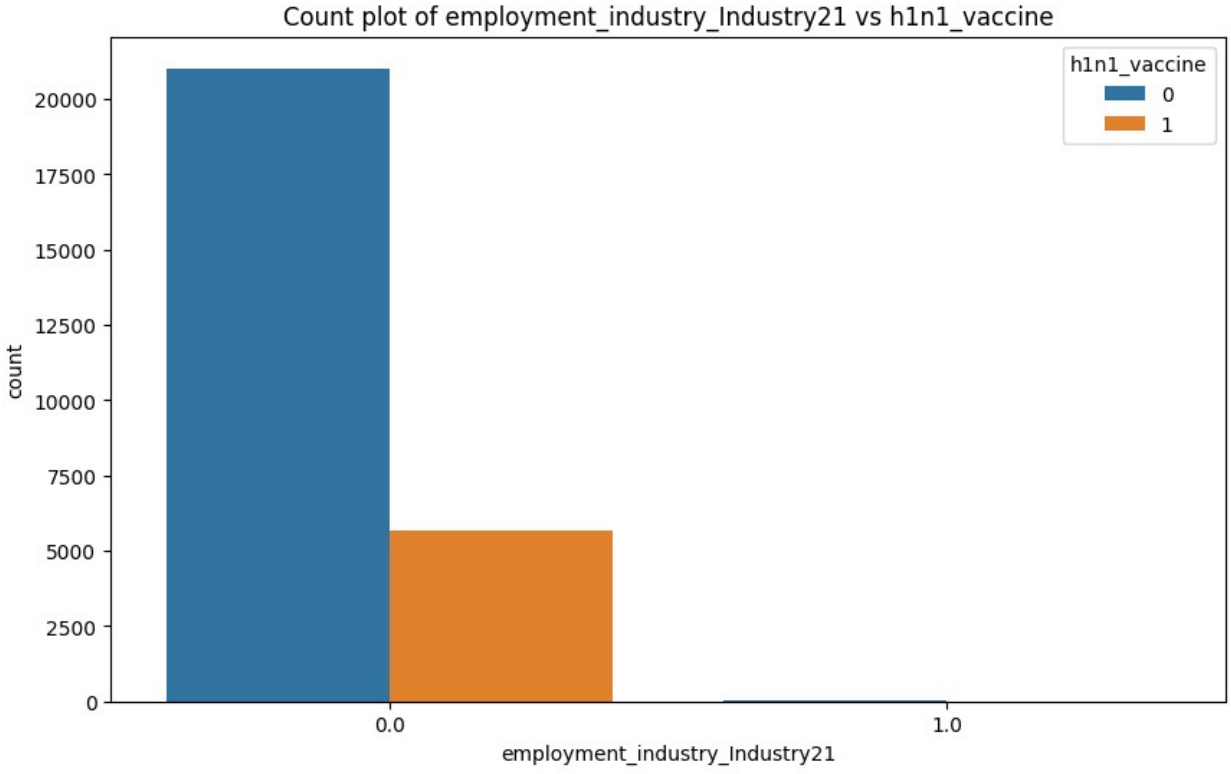

Count plot of employment_industry_Industry17 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry17

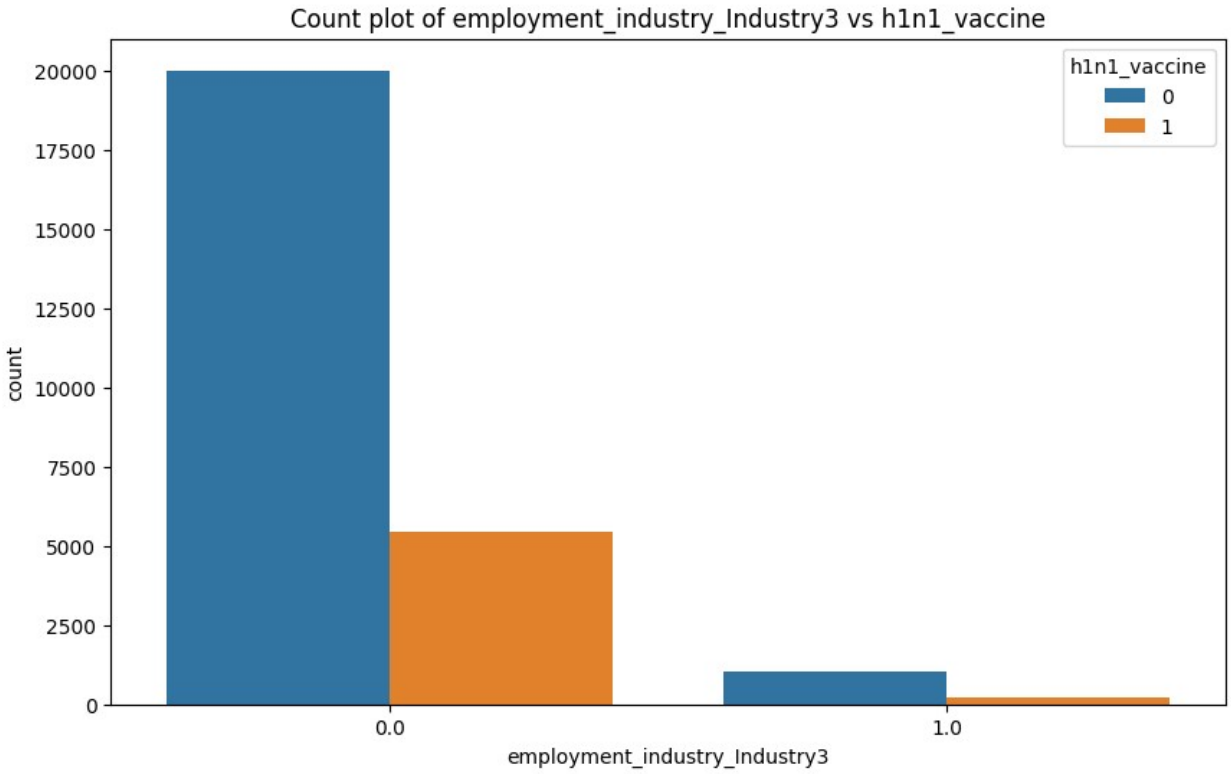

Count plot of employment_industry_Industry18 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry18

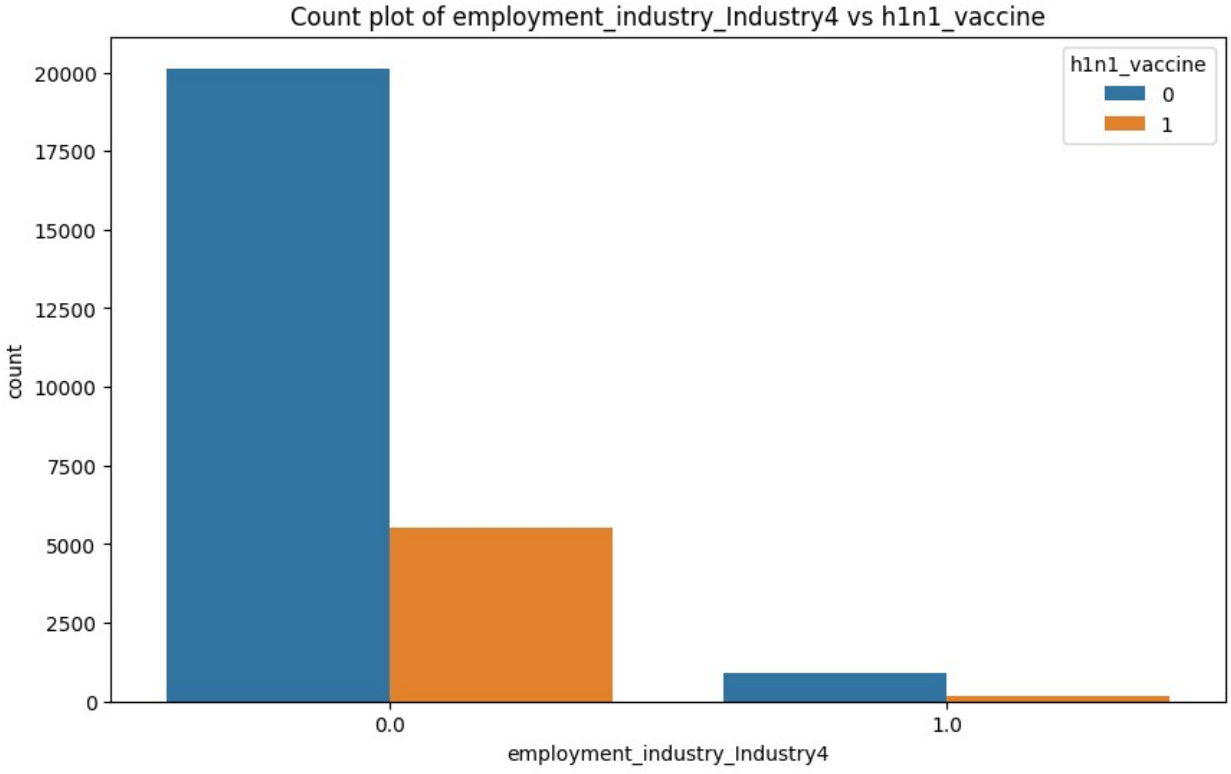

Count plot of employment_industry_Industry19 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry19

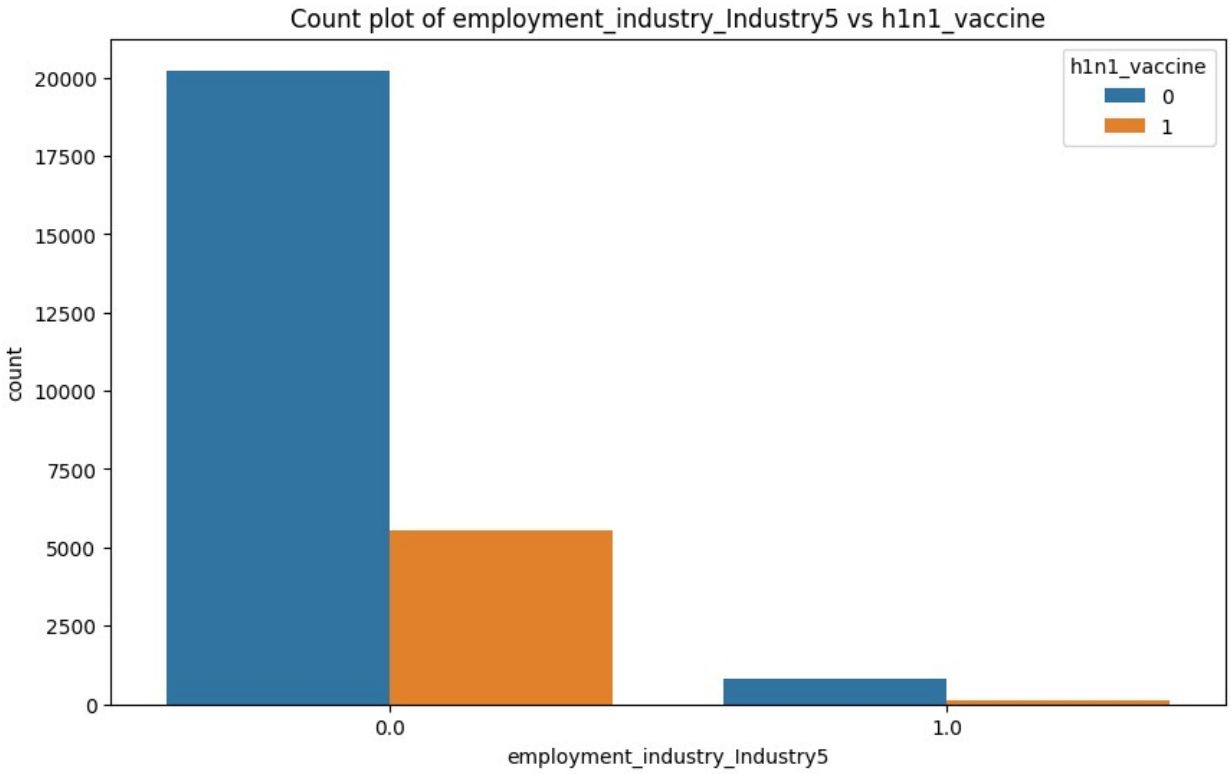

Count plot of employment_industry_Industry2 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry2

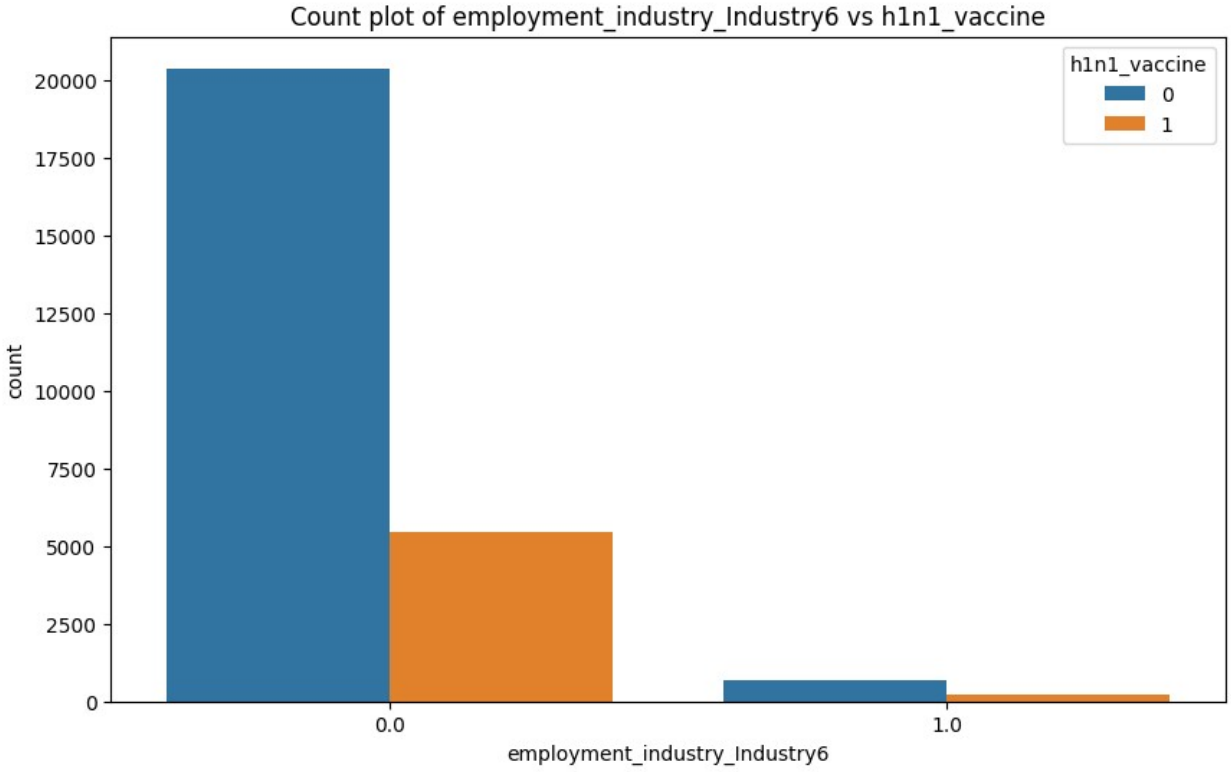

Count plot of employment_industry_Industry20 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry20

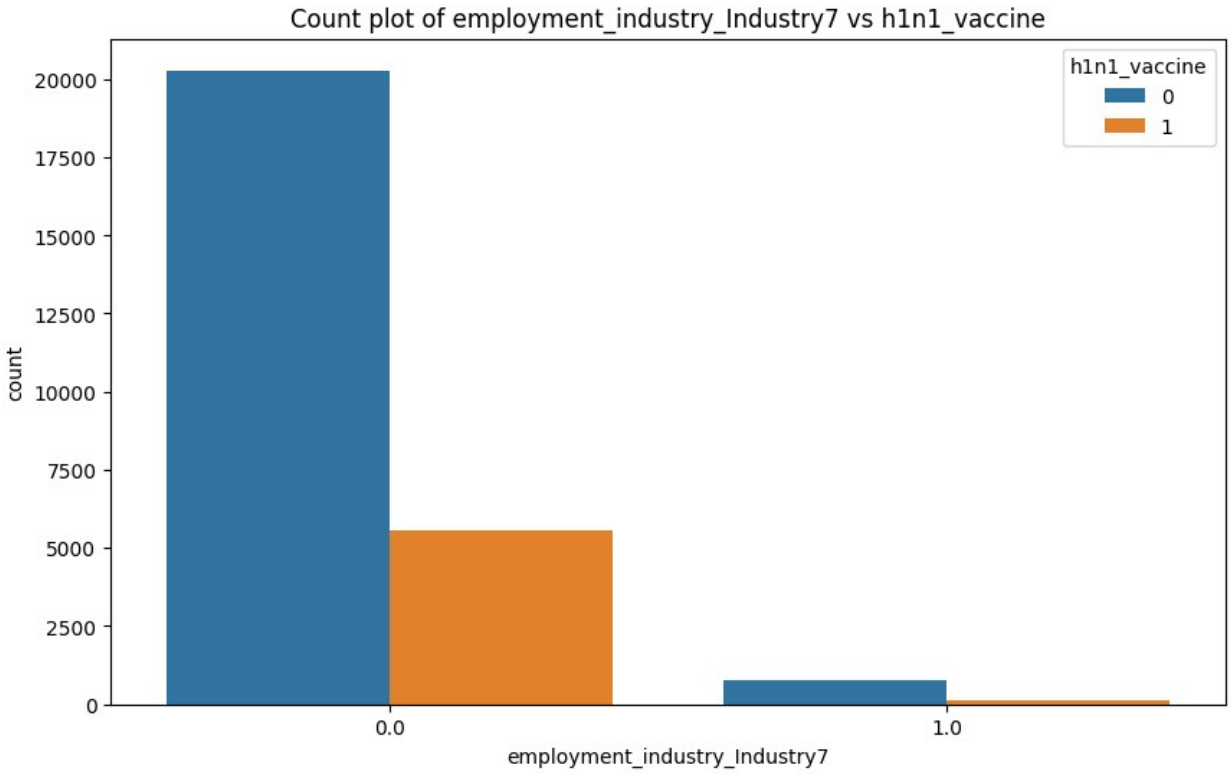

Count plot of employment_industry_Industry21 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry21

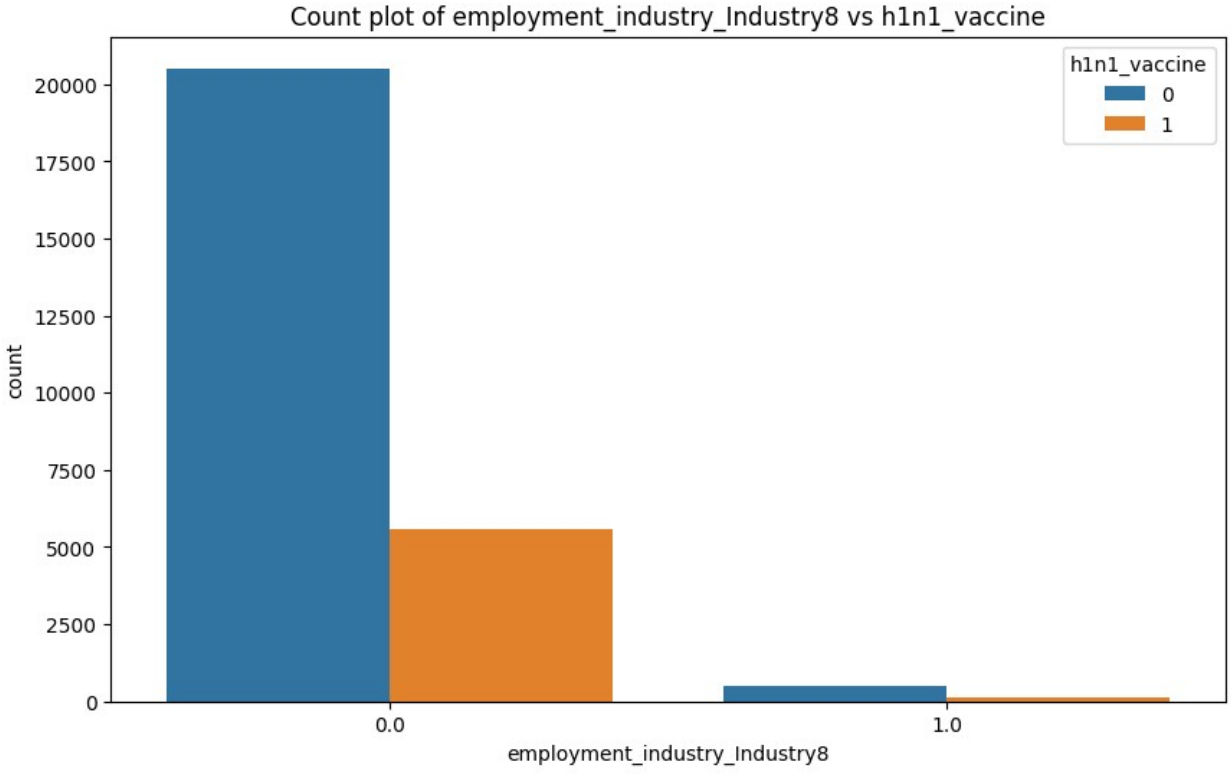

Count plot of employment_industry_Industry3 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry3

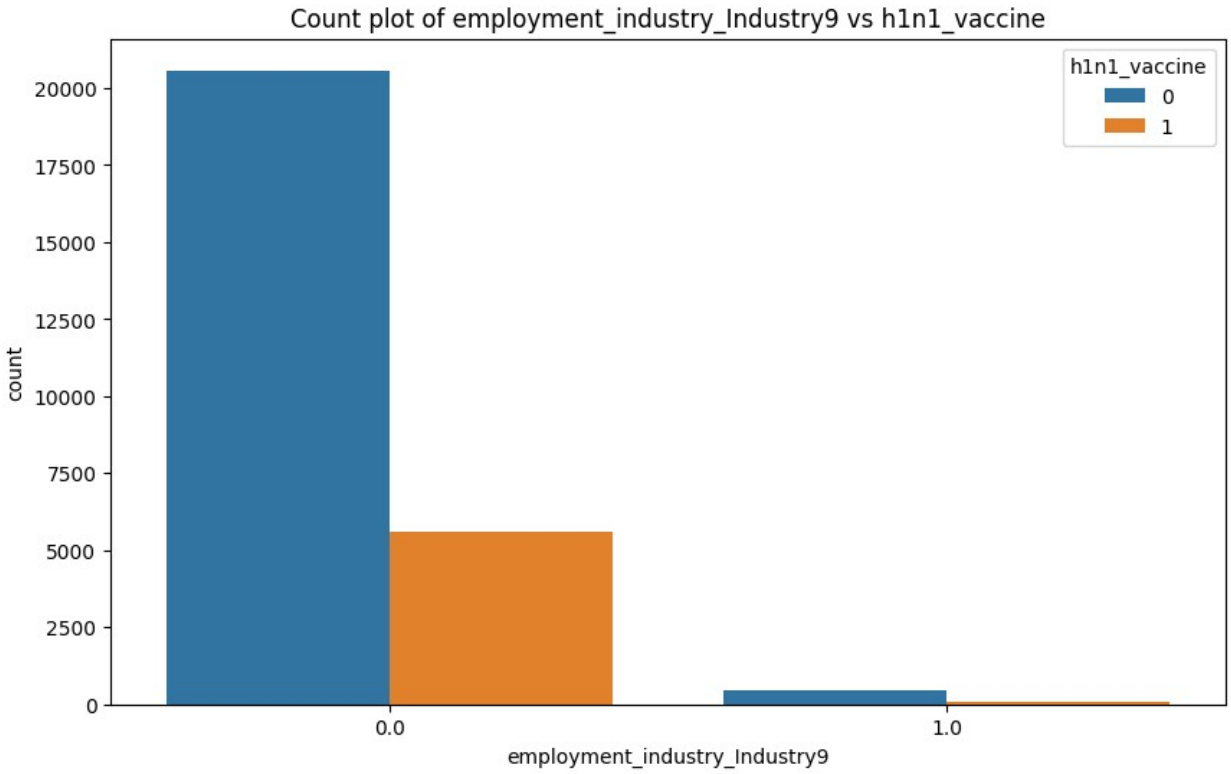

Count plot of employment_industry_Industry4 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry4

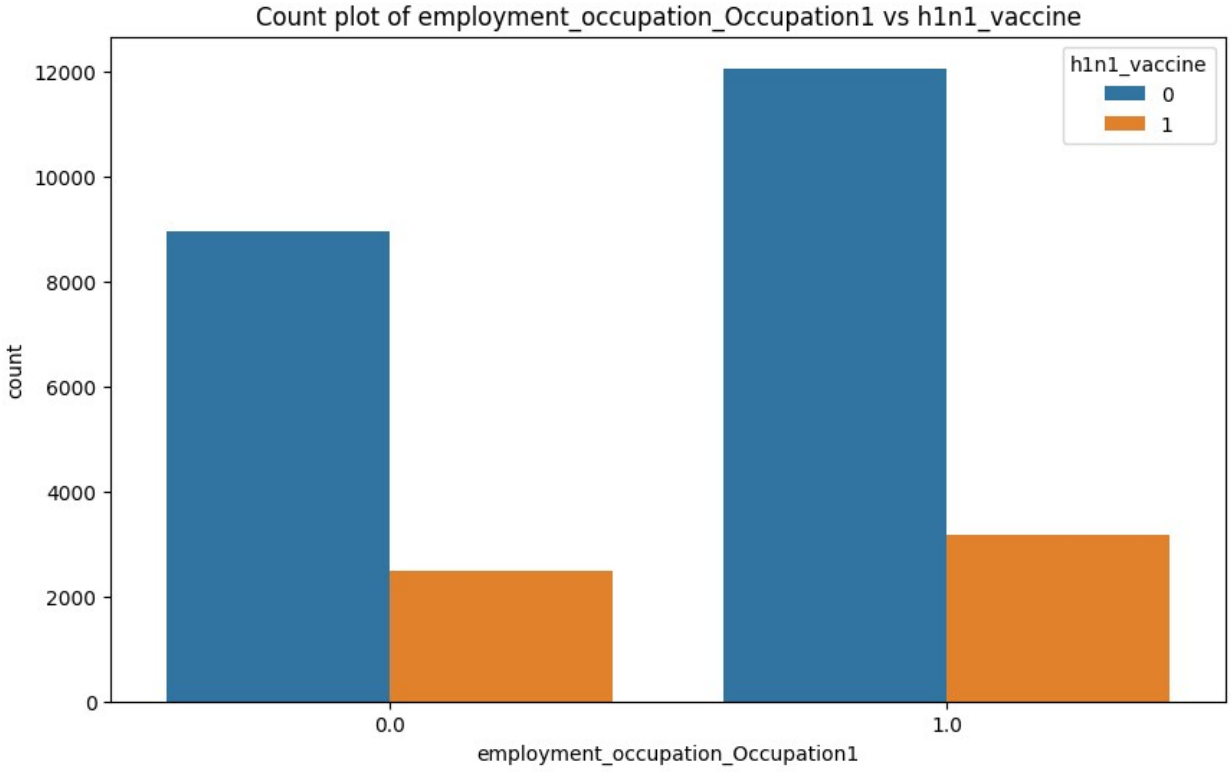

Count plot of employment_industry_Industry5 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry5

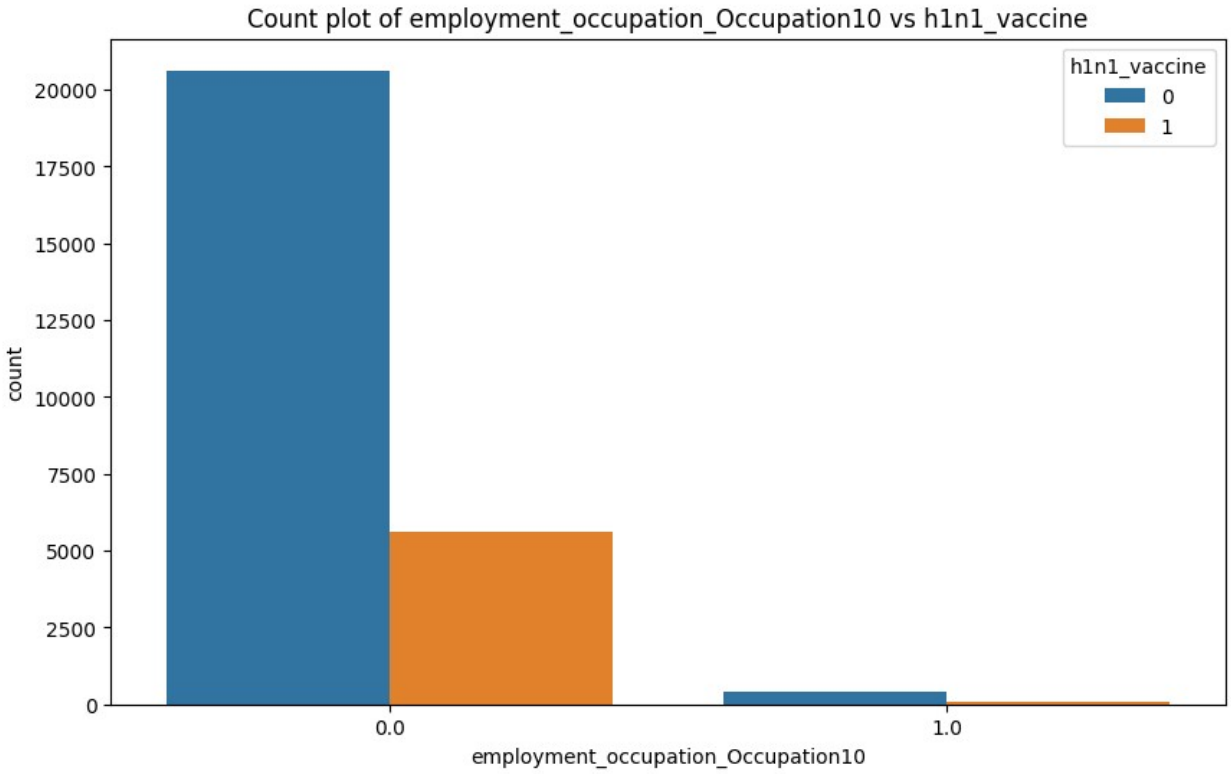

Count plot of employment_industry_Industry6 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_industry_Industry6

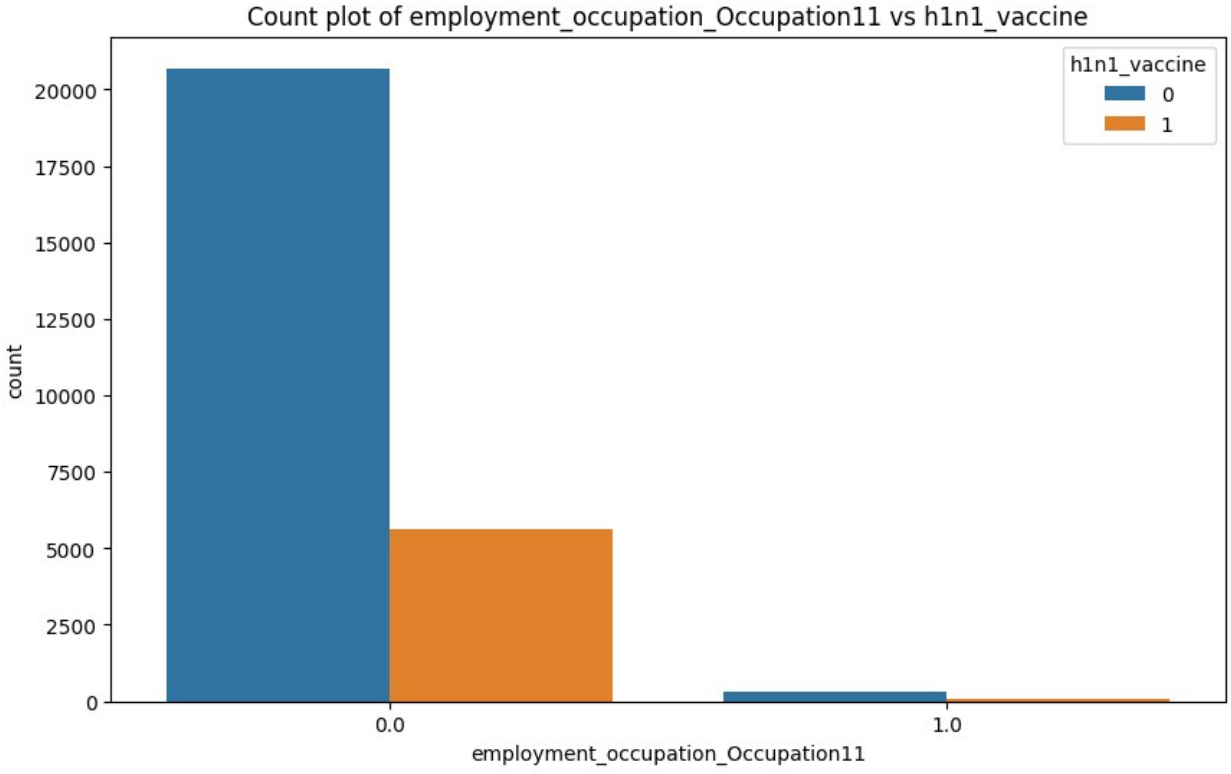

Count plot of employment_industry_Industry7 vs h1n1_vaccine
h1n1_vaccine
0
1
20000
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
employment_industry_Industry7

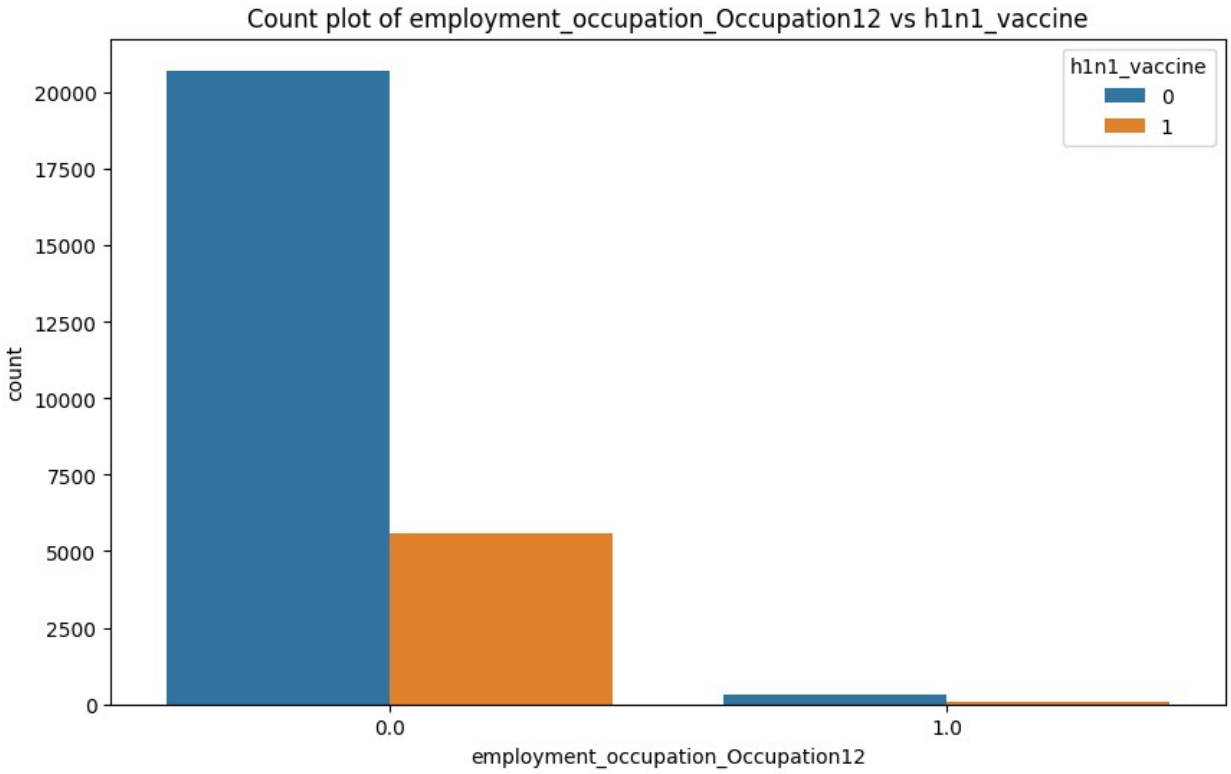

Count plot of employment_industry_Industry8 vs h1n1_vaccine
h1n1_vaccine
0
1
20000
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
employment_industry_Industry8

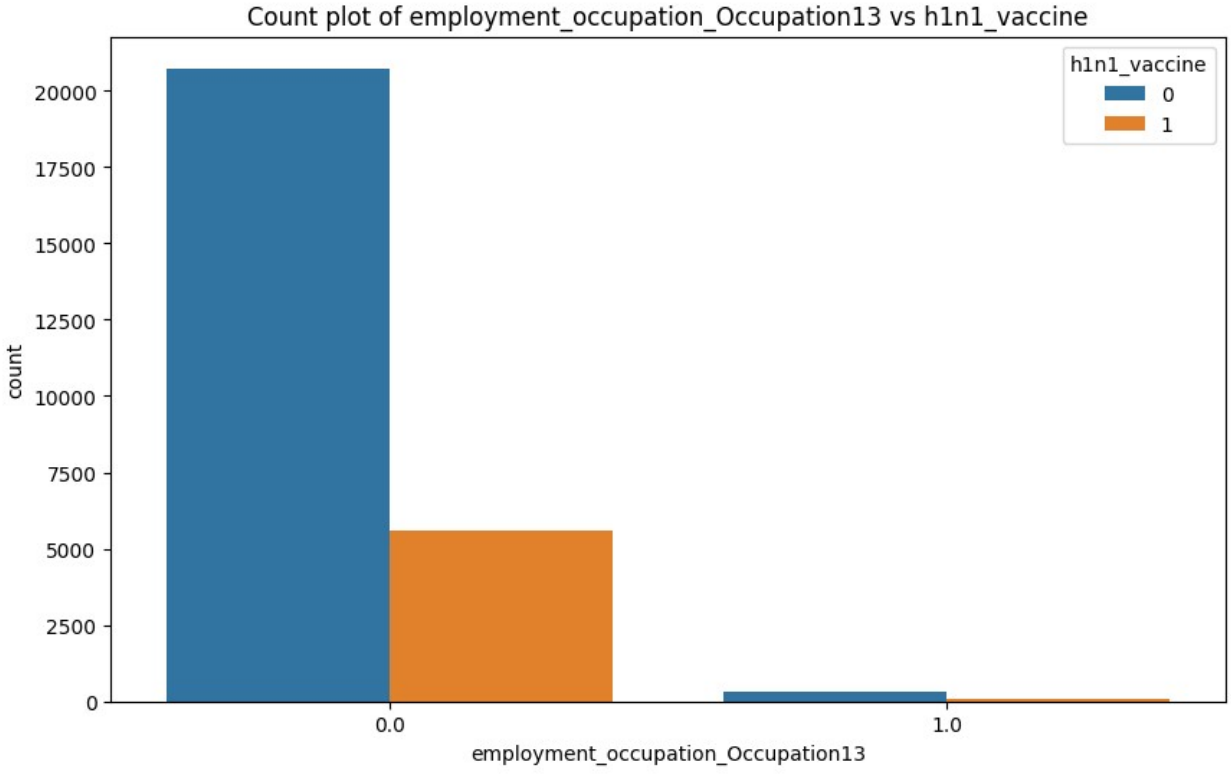

Count plot of employment_industry_Industry9 vs h1n1_vaccine
h1n1_vaccine
0
1
20000
17500
15000
12500
10000
7500
5000
2500
0
count
0.0
1.0
employment_industry_Industry9

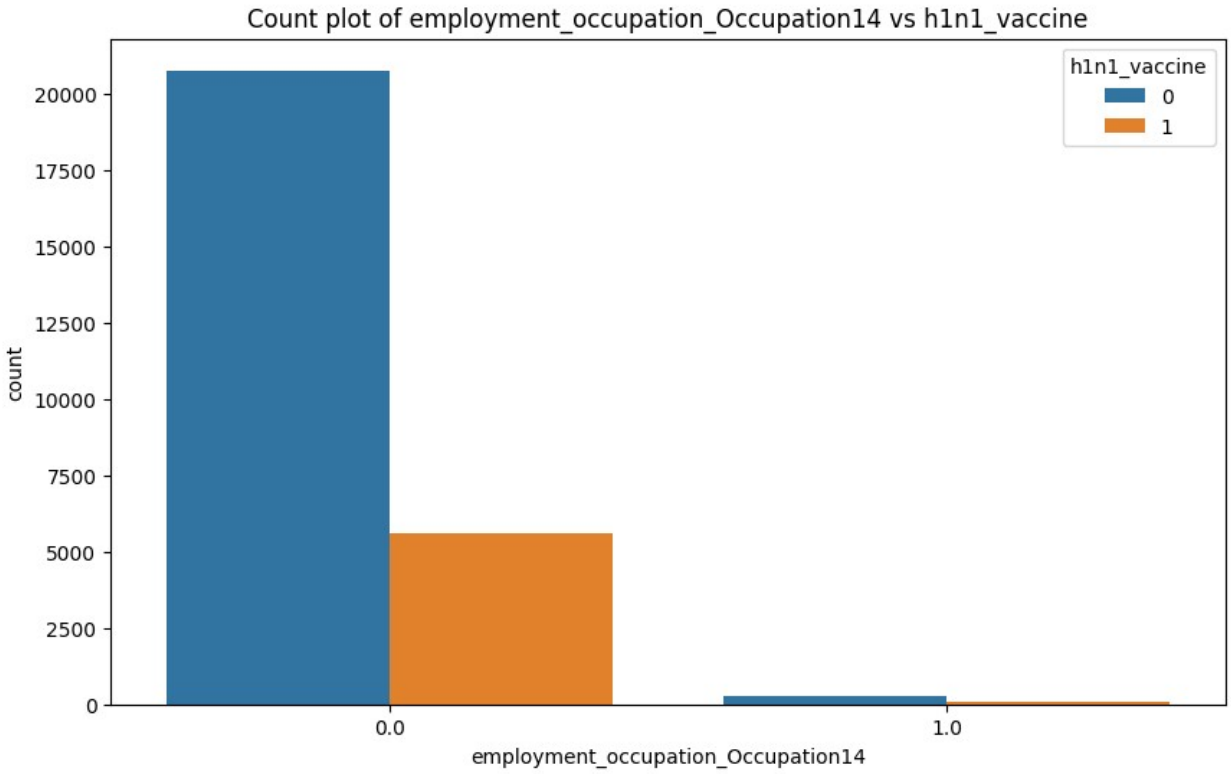

Count plot of employment_occupation_Occupation1 vs h1n1_vaccine
h1n1_vaccine
0
1
12000
10000
8000
6000
4000
2000
0
count
0.0
1.0
employment_occupation_Occupation1

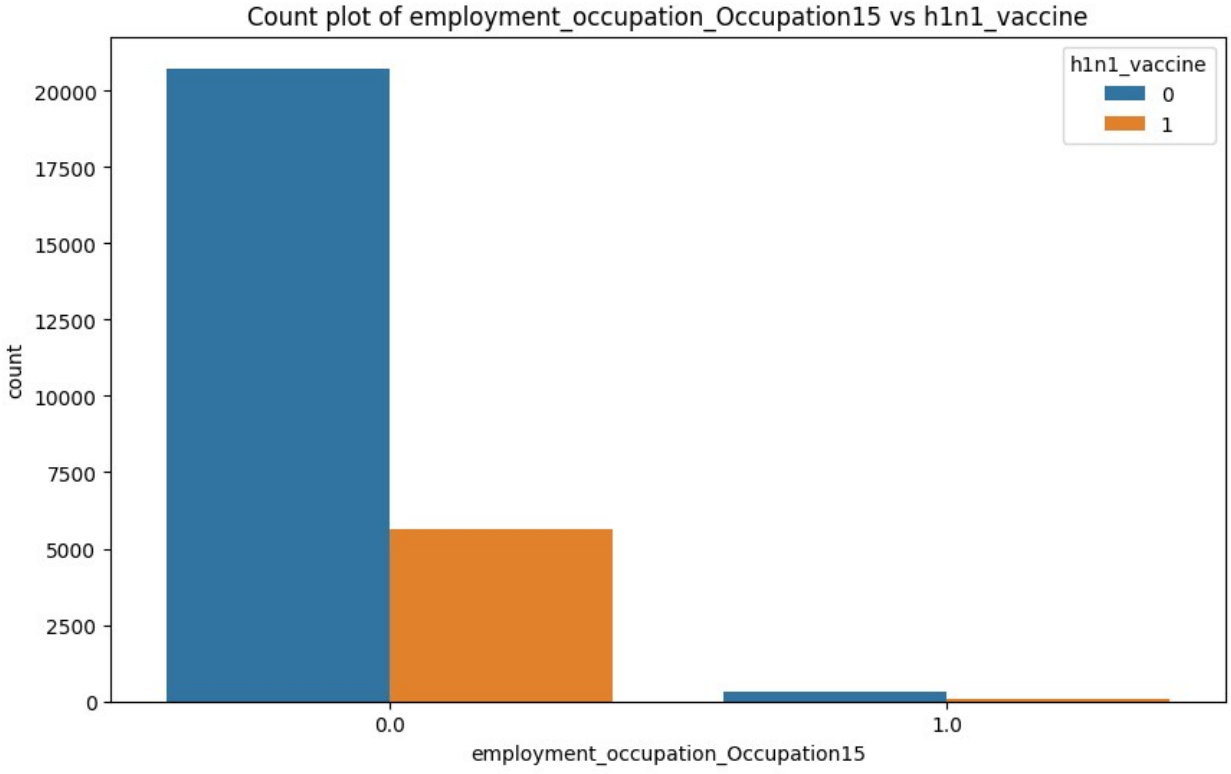

Count plot of employment_occupation_Occupation10 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation10

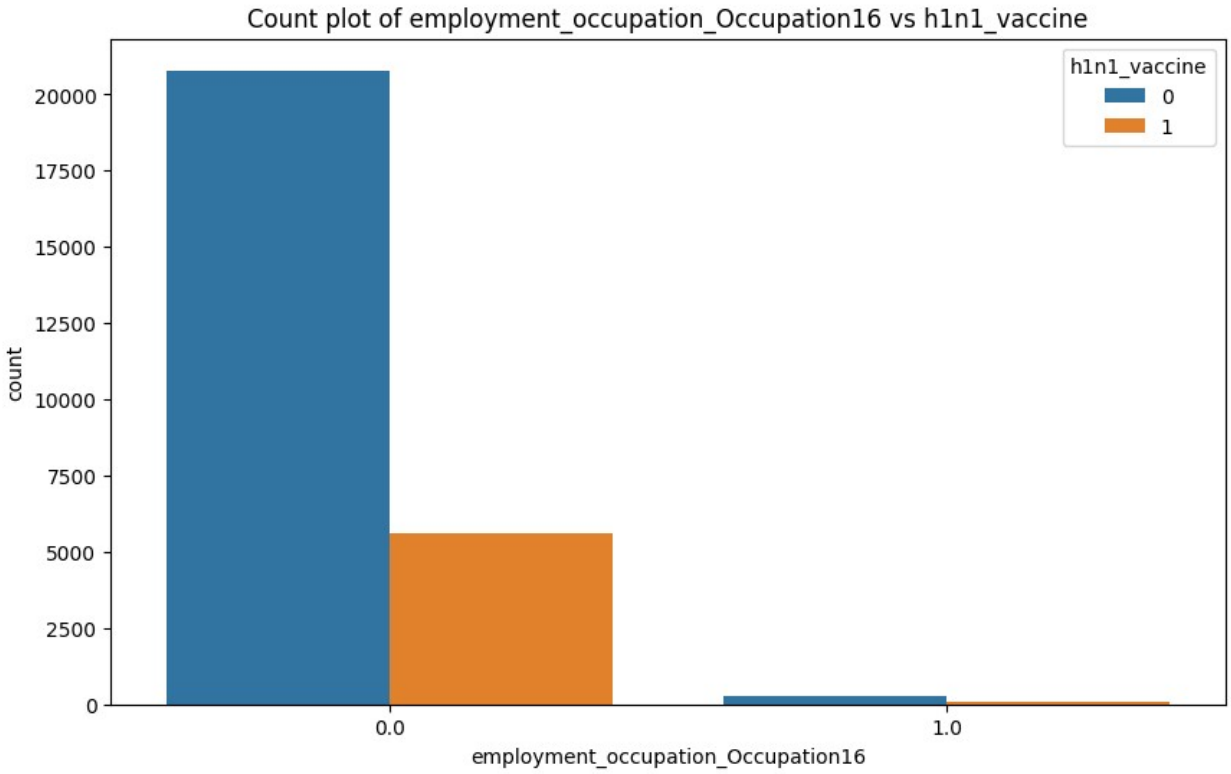

Count plot of employment_occupation_Occupation11 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation11

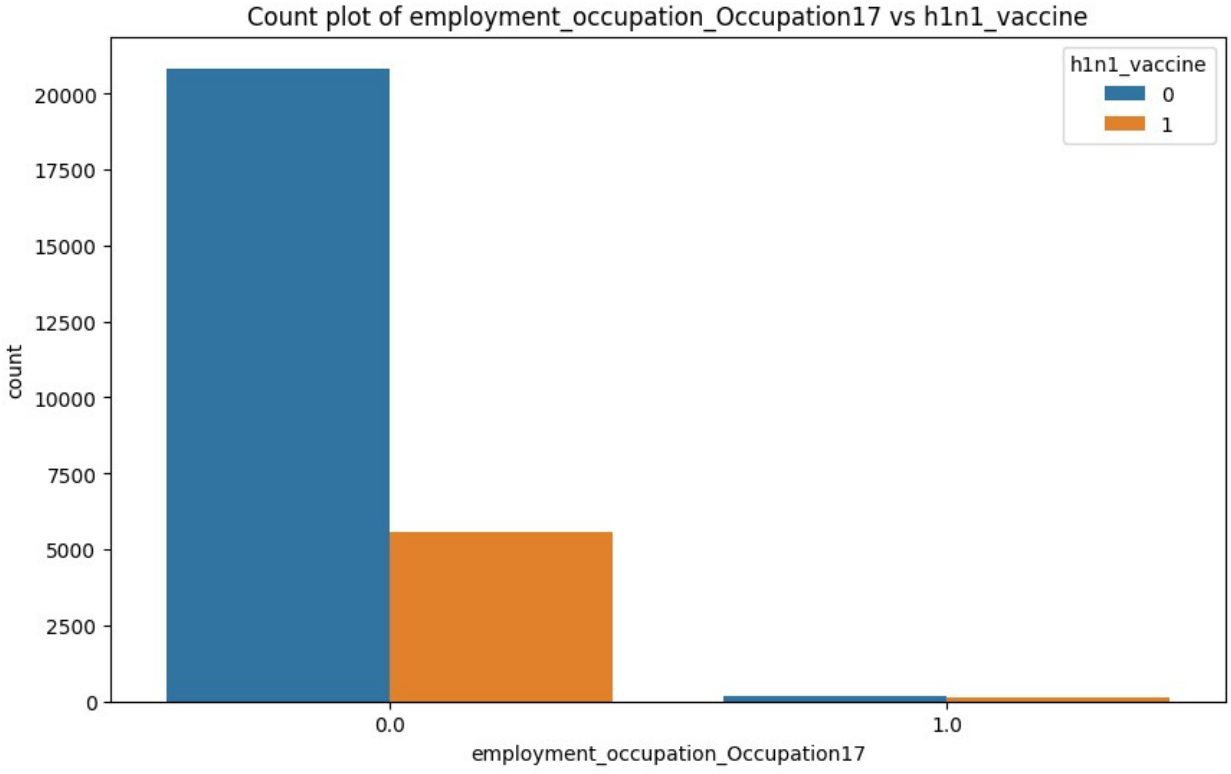

Count plot of employment_occupation_Occupation12 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation12

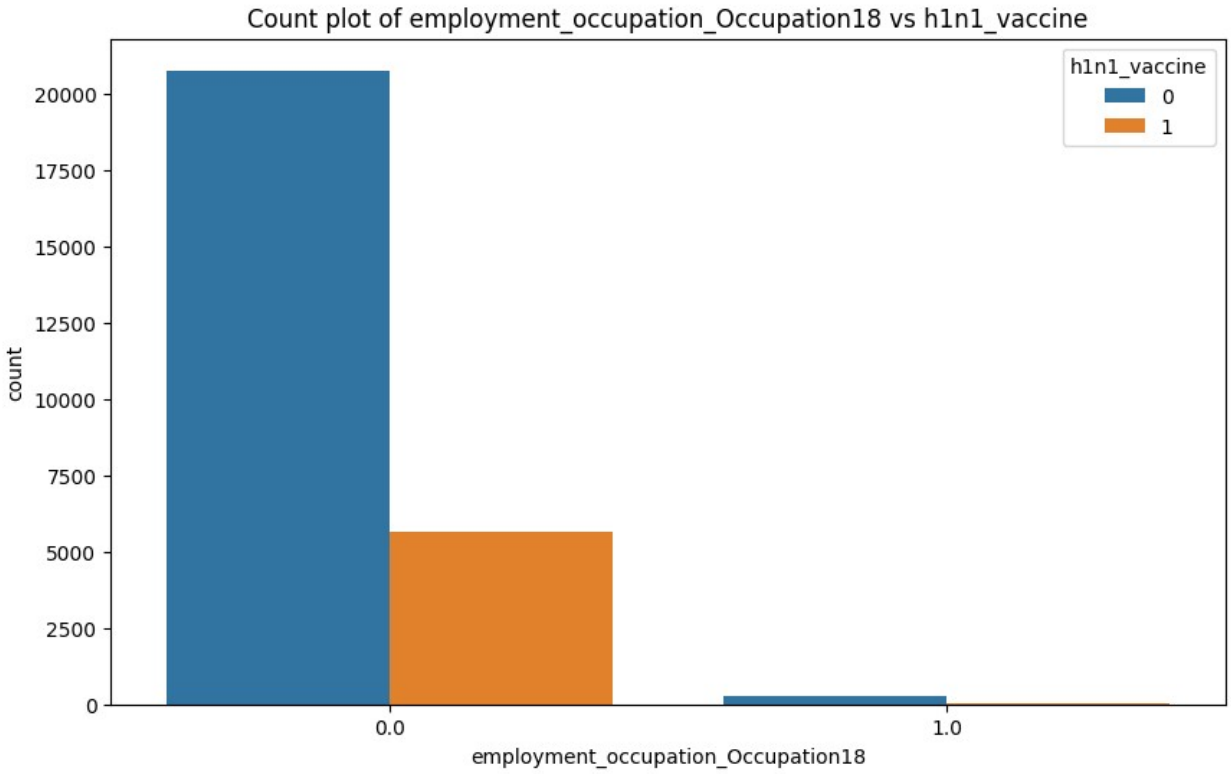

Count plot of employment_occupation_Occupation13 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation13

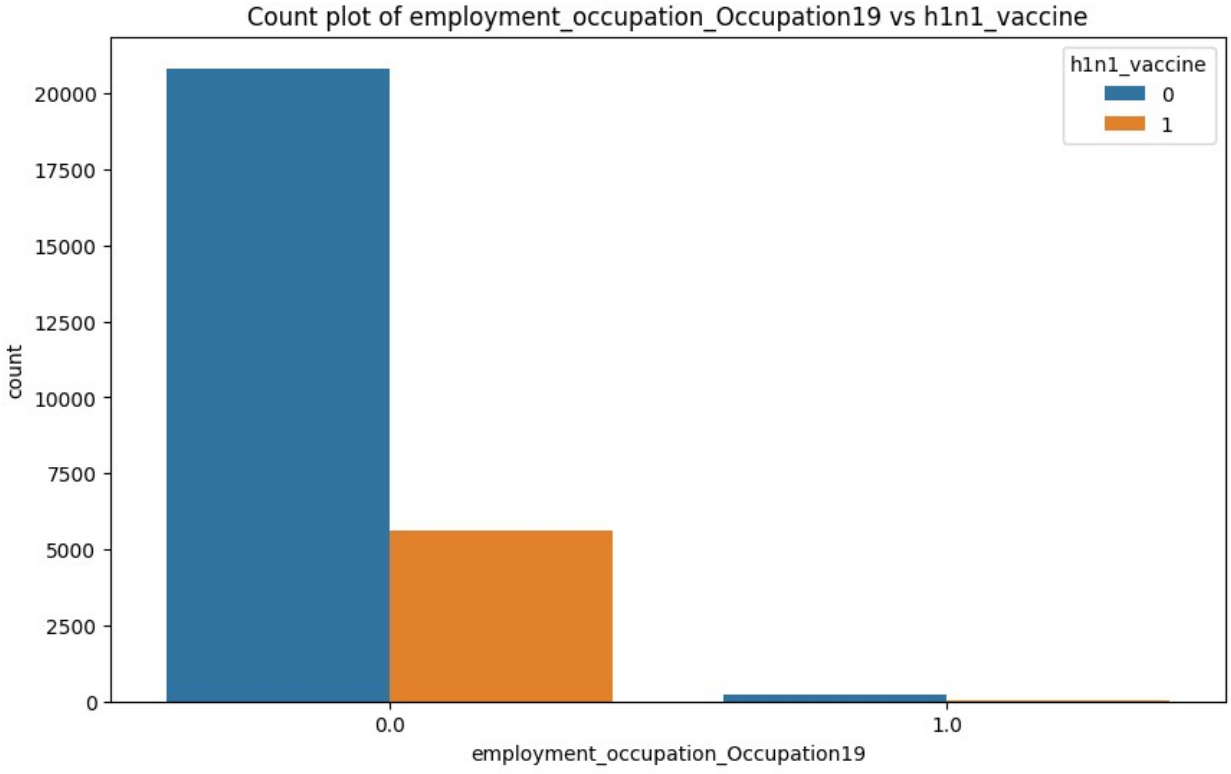

Count plot of employment_occupation_Occupation14 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation14

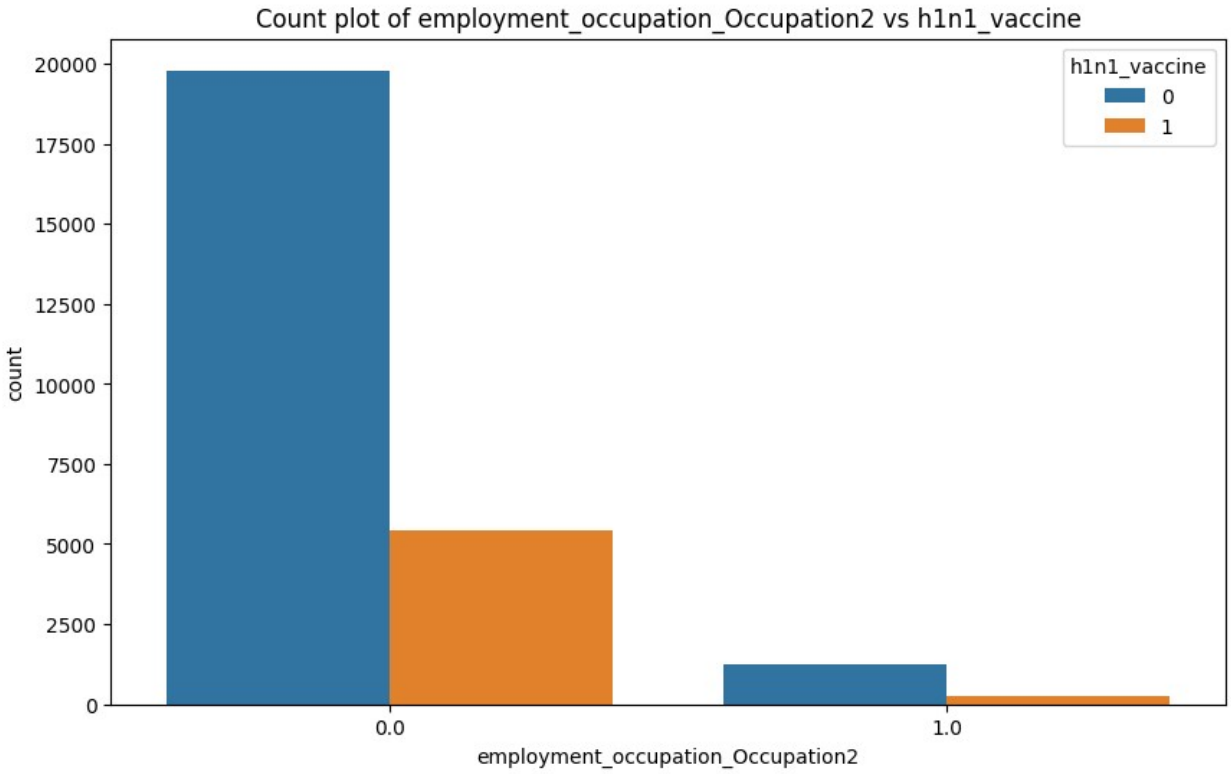

Count plot of employment_occupation_Occupation15 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation15

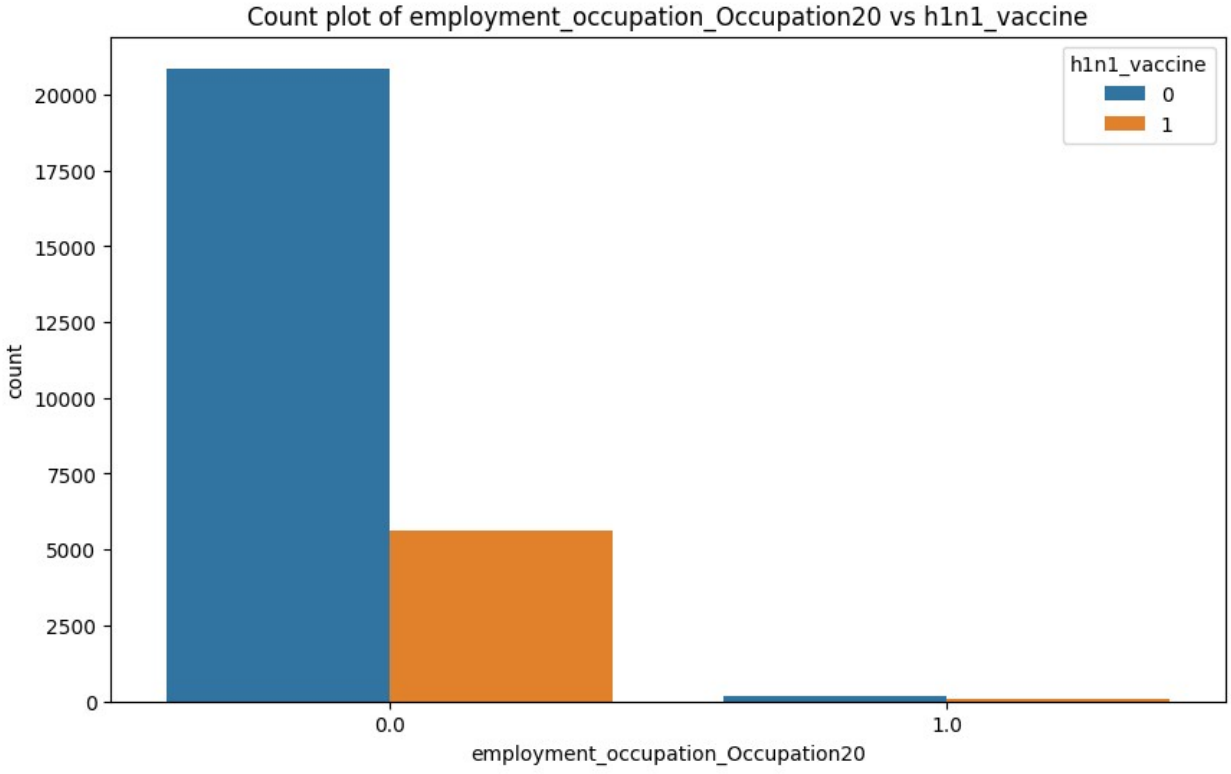

Count plot of employment_occupation_Occupation16 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation16

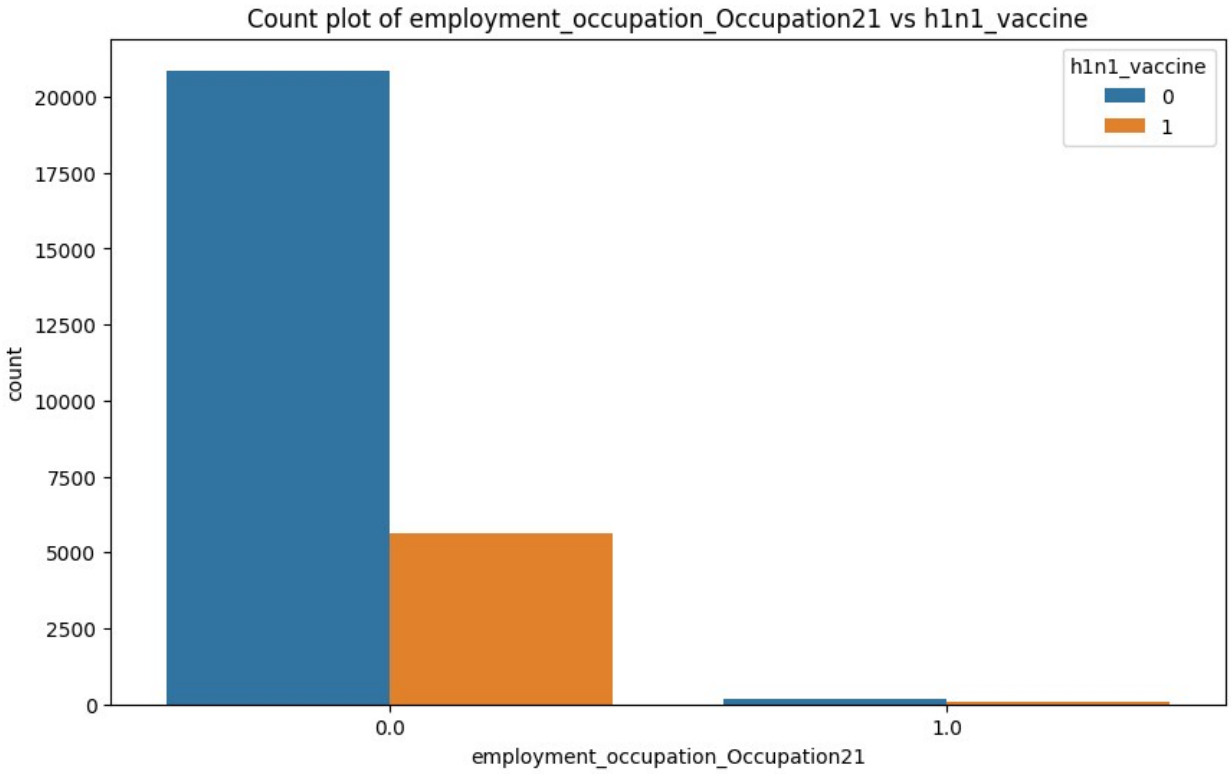

Count plot of employment_occupation_Occupation17 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation17

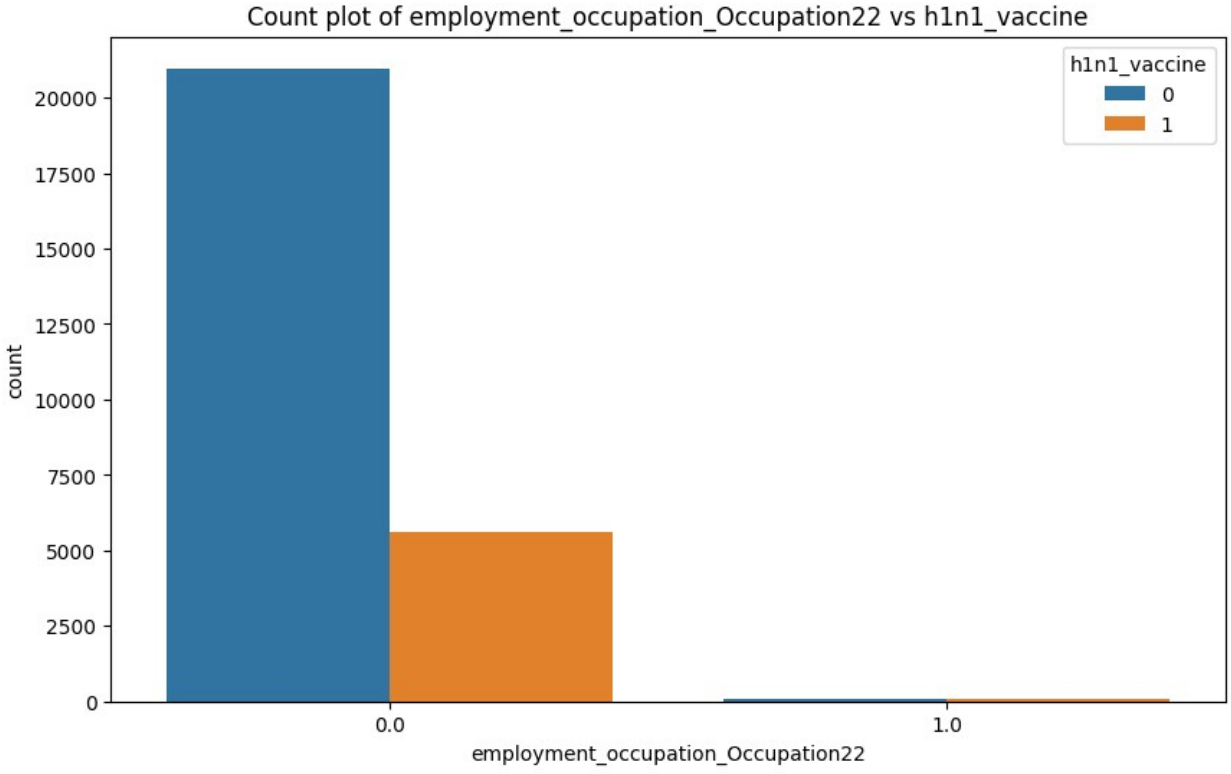

Count plot of employment_occupation_Occupation18 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation18

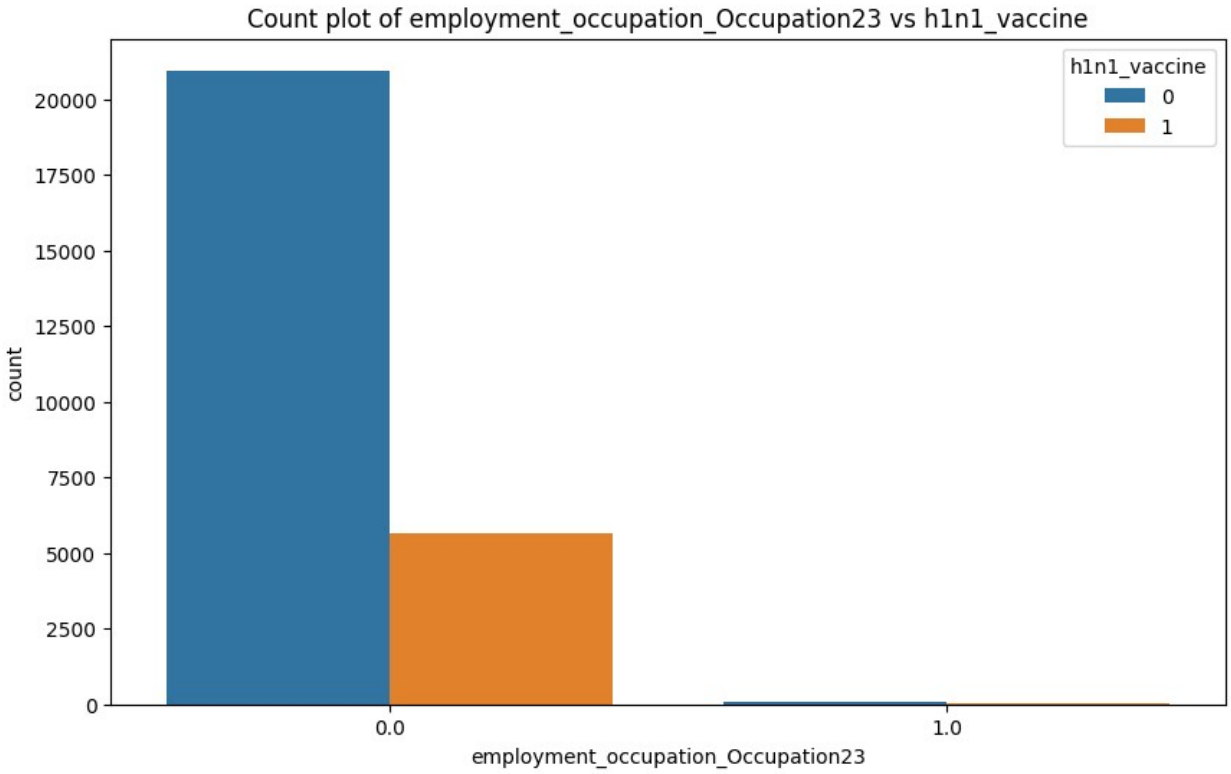

Count plot of employment_occupation_Occupation19 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation19


Count plot of employment_occupation_Occupation2 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation2

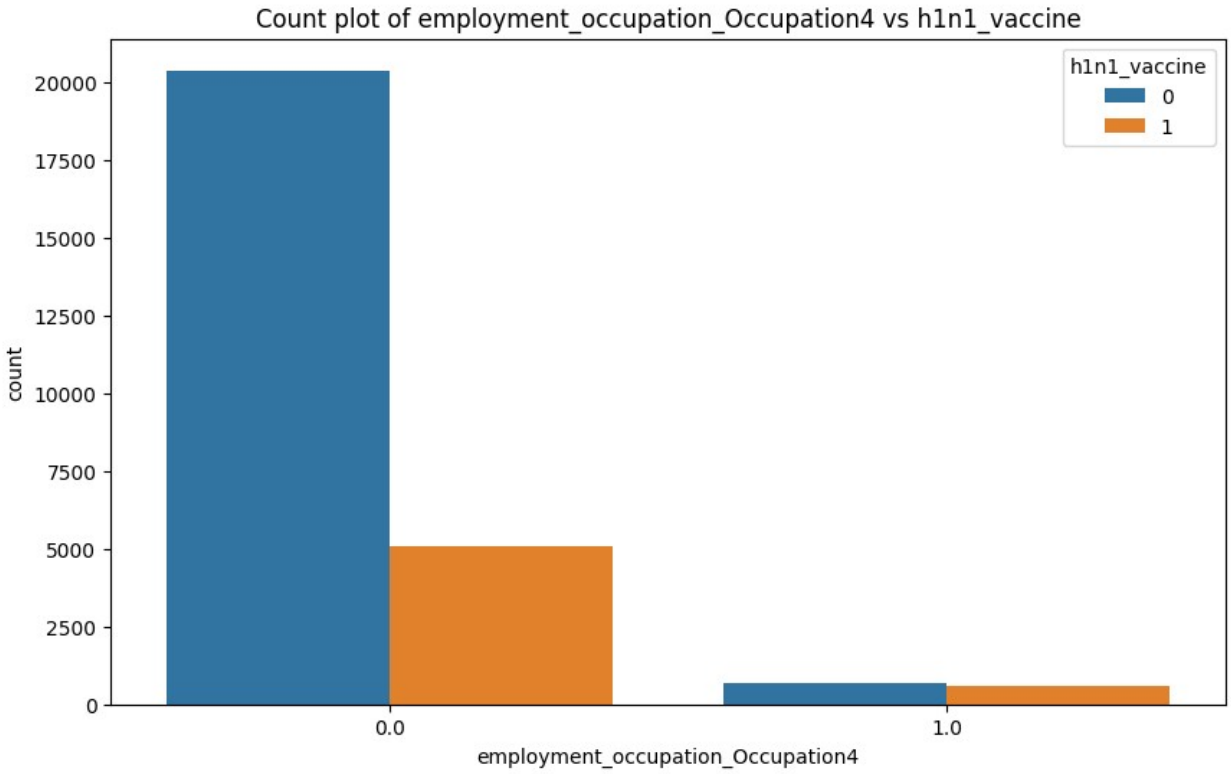

Count plot of employment_occupation_Occupation20 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation20


Count plot of employment_occupation_Occupation21 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation21

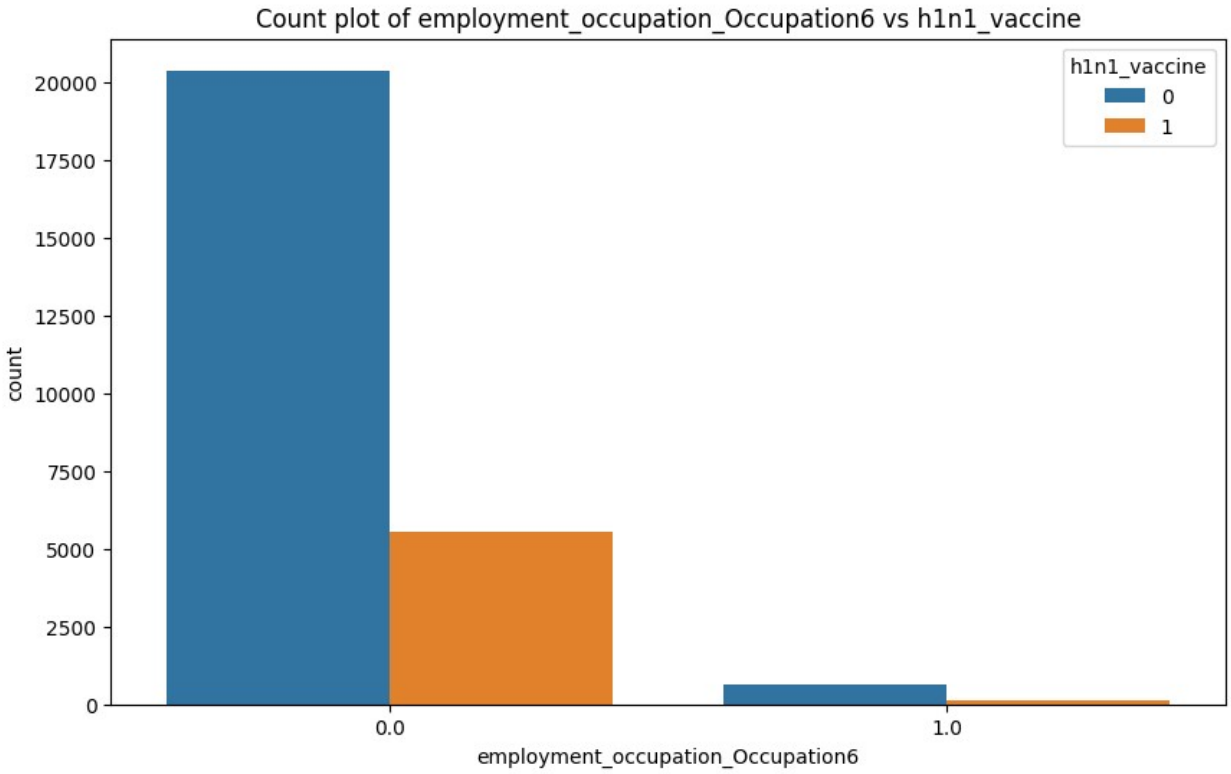

Count plot of employment_occupation_Occupation22 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation22

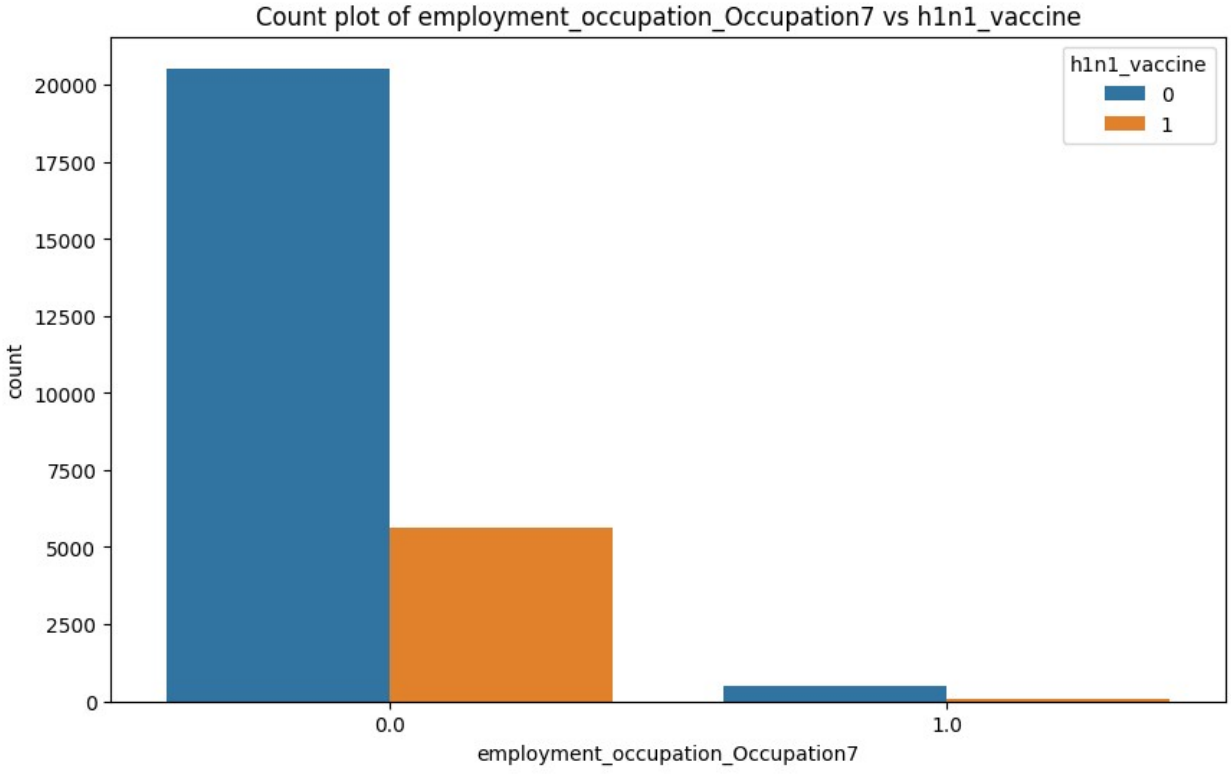

Count plot of employment_occupation_Occupation23 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation23

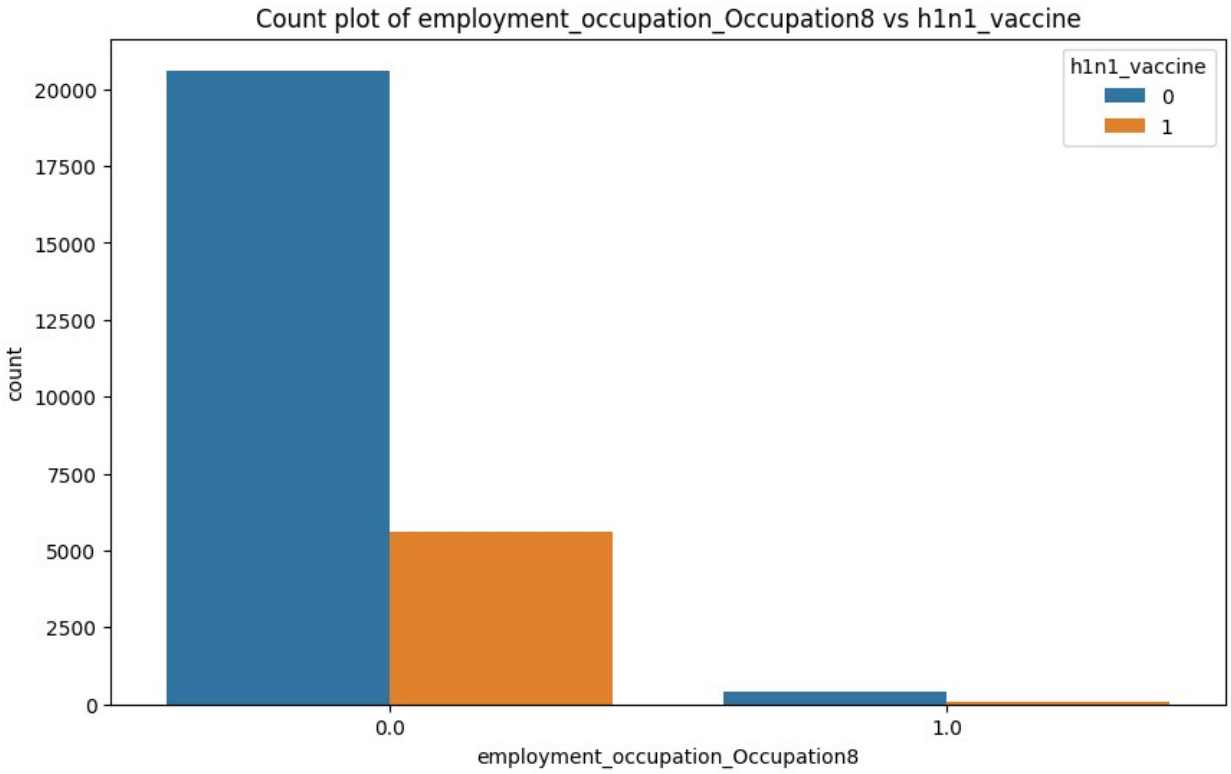

Count plot of employment_occupation_Occupation3 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation3


Count plot of employment_occupation_Occupation4 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation4


Count plot of employment_occupation_Occupation5 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation5


Count plot of employment_occupation_Occupation6 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation6


Count plot of employment_occupation_Occupation7 vs h1n1_vaccine
h1n1_vaccine
0
1
count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
employment_occupation_Occupation7




```
Chi-square test for age_group_18 - 34 Years: p-value =
1.1107257317853613e-05
```

```
Chi-square test for age_group_35 - 44 Years: p-value =
0.017011877476612418
Chi-square test for age_group_45 - 54 Years: p-value =
0.000503997440835164
Chi-square test for age_group_55 - 64 Years: p-value =
5.244506510542241e-10
Chi-square test for age_group_65+ Years: p-value =
0.0009230331710009553
Chi-square test for education_12 Years: p-value = 6.266111779960853e-
09
Chi-square test for education_< 12 Years: p-value =
2.0058942847306965e-08
Chi-square test for education_College Graduate: p-value =
1.4840184403068569e-19
Chi-square test for education_Some College: p-value =
0.2801606959328625
Chi-square test for race_Black: p-value = 9.704732797494308e-14
Chi-square test for race_Hispanic: p-value = 0.6569534913463986
Chi-square test for race_Other or Multiple: p-value =
0.7051010288484753
Chi-square test for race_White: p-value = 4.928736280406675e-07
Chi-square test for sex_Female: p-value = 0.0007709155489949327
Chi-square test for sex_Male: p-value = 0.0007709155489949327
Chi-square test for income_poverty_<= $75,000, Above Poverty: p-value
= 8.315907406171116e-12
Chi-square test for income_poverty_> $75,000: p-value =
3.0123076534159374e-21
Chi-square test for income_poverty_Below Poverty: p-value =
0.005039167859565663
Chi-square test for marital_status_Married: p-value =
1.6985751321912323e-13
Chi-square test for marital_status_Not Married: p-value =
1.6985751321912323e-13
Chi-square test for rent_or_own_Own: p-value = 4.5507157863887266e-07
Chi-square test for rent_or_own_Rent: p-value = 4.5507157863887266e-07
Chi-square test for employment_status_Employed: p-value =
0.9565051762916758
Chi-square test for employment_status_Not in Labor Force: p-value =
0.03402411260760365
Chi-square test for employment_status_Unemployed: p-value =
2.6576978604974254e-06
Chi-square test for hhs_geo_region_Region1: p-value =
1.747557810306264e-08
Chi-square test for hhs_geo_region_Region10: p-value =
0.0006648694526483314
Chi-square test for hhs_geo_region_Region2: p-value =
0.10833748935187065
Chi-square test for hhs_geo_region_Region3: p-value =
0.1548606001019116
```

```
Chi-square test for hhs_geo_region_Region4: p-value =
0.011715257425215507
Chi-square test for hhs_geo_region_Region5: p-value =
0.05475510261122678
Chi-square test for hhs_geo_region_Region6: p-value =
2.2357814156547447e-10
Chi-square test for hhs_geo_region_Region7: p-value =
0.031118320101114195
Chi-square test for hhs_geo_region_Region8: p-value =
0.8241602535632312
Chi-square test for hhs_geo_region_Region9: p-value =
0.5888515325486346
Chi-square test for census_msa_MSA, Not Principle  City: p-value =
0.8440025038138949
Chi-square test for census_msa_MSA, Principle City: p-value =
0.7486102580589248
Chi-square test for census_msa_Non-MSA: p-value = 0.9263581190403177
Chi-square test for employment_industry_Industry1: p-value =
1.7954387279968323e-33
Chi-square test for employment_industry_Industry10: p-value =
7.722091143488722e-05
Chi-square test for employment_industry_Industry11: p-value =
5.1813367344799395e-05
Chi-square test for employment_industry_Industry12: p-value =
0.07489252148901883
Chi-square test for employment_industry_Industry13: p-value =
0.0032824660241128636
Chi-square test for employment_industry_Industry14: p-value =
0.1016762643872713
Chi-square test for employment_industry_Industry15: p-value =
0.0019300929016743843
Chi-square test for employment_industry_Industry16: p-value =
0.006765246127777069
Chi-square test for employment_industry_Industry17: p-value =
0.0004705715909078537
Chi-square test for employment_industry_Industry18: p-value =
1.0268055243494551e-33
Chi-square test for employment_industry_Industry19: p-value =
0.08431851654611673
Chi-square test for employment_industry_Industry2: p-value =
0.00041439120053508703
Chi-square test for employment_industry_Industry20: p-value =
0.37630620093743605
Chi-square test for employment_industry_Industry21: p-value = 1.0
Chi-square test for employment_industry_Industry3: p-value =
0.000468847486131637
Chi-square test for employment_industry_Industry4: p-value =
2.711655165341728e-09
Chi-square test for employment_industry_Industry5: p-value =
```

```
3.4982736609948596e-09
Chi-square test for employment_industry_Industry6: p-value =
0.05862388858233208
Chi-square test for employment_industry_Industry7: p-value =
1.424137940259285e-10
Chi-square test for employment_industry_Industry8: p-value =
0.002797845534310715
Chi-square test for employment_industry_Industry9: p-value =
1.3476370629537048e-05
Chi-square test for employment_occupation_Occupation1: p-value =
0.05691524140768011
Chi-square test for employment_occupation_Occupation10: p-value =
1.4619843790764891e-06
Chi-square test for employment_occupation_Occupation11: p-value =
0.001186467982088866
Chi-square test for employment_occupation_Occupation12: p-value =
0.27608084813577155
Chi-square test for employment_occupation_Occupation13: p-value =
0.3132978773866931
Chi-square test for employment_occupation_Occupation14: p-value =
0.254774492646185
Chi-square test for employment_occupation_Occupation15: p-value =
0.005254669007329116
Chi-square test for employment_occupation_Occupation16: p-value =
0.013974785235310394
Chi-square test for employment_occupation_Occupation17: p-value =
1.830977724497757e-10
Chi-square test for employment_occupation_Occupation18: p-value =
9.500936221074609e-07
Chi-square test for employment_occupation_Occupation19: p-value =
0.4183824351050891
Chi-square test for employment_occupation_Occupation2: p-value =
0.0005818826772971309
Chi-square test for employment_occupation_Occupation20: p-value =
0.020025244006694448
Chi-square test for employment_occupation_Occupation21: p-value =
0.21352965354606135
Chi-square test for employment_occupation_Occupation22: p-value =
1.0268055243494551e-33
Chi-square test for employment_occupation_Occupation23: p-value =
0.11787936301037377
Chi-square test for employment_occupation_Occupation3: p-value =
0.036929281034546665
Chi-square test for employment_occupation_Occupation4: p-value =
4.471014927178954e-115
Chi-square test for employment_occupation_Occupation5: p-value =
9.307597930442775e-11
Chi-square test for employment_occupation_Occupation6: p-value =
0.0004366427732143667
```

```
Chi-square test for employment_occupation_Occupation7: p-value =
1.8924635189065717e-09
Chi-square test for employment_occupation_Occupation8: p-value =
0.00017160142227806046
Chi-square test for employment_occupation_Occupation9: p-value =
0.8073741678224693
```

Summary for Bivariate Analysis

Numeric Features

Box Plots Interpretation:

h1n1_concern:

Observation: Vaccinated individuals have a higher median h1n1_concern score compared to non-vaccinated individuals.

Implication: Higher concern about H1N1 increases the likelihood of getting vaccinated.

h1n1_knowledge:

Observation: Vaccinated individuals show higher median h1n1_knowledge scores.

Implication: Greater knowledge about H1N1 is associated with higher vaccination rates.

doctor_recc_h1n1:

Observation: Individuals who received a doctor's recommendation have a higher median score and are more likely to be vaccinated.

Implication: Doctor's recommendation plays a significant role in influencing vaccination decisions.

health_worker:

Observation: Health workers have a higher median score and a higher vaccination rate.

Implication: Being a health worker increases the likelihood of vaccination due to increased exposure and risk awareness.

Categorical Features

Count Plots Interpretation:

age_group:

Observation: Older age groups (e.g., 65+ years) have more vaccinated individuals compared to younger groups.

Implication: Age is a significant factor, with older individuals more likely to get vaccinated.

education:

Observation: Higher education levels (e.g., College Graduate) show higher vaccination rates.

Implication: Education level positively influences vaccination uptake, likely due to better understanding of the vaccine's benefits.

race:

Observation: Differences in vaccination rates among different racial groups, with some groups having lower vaccination rates.

Implication: There are racial disparities in vaccination uptake that need to be addressed.

sex:

Observation: Females show a slightly higher vaccination rate compared to males.

Implication: Gender differences play a role in vaccination behavior, with females being more likely to get vaccinated.

income_poverty:

Observation: Higher income groups show higher vaccination rates.

Implication: Socioeconomic status impacts vaccination behavior, with wealthier individuals being more likely to get vaccinated.

Chi-Square Test Results:

age_group:

p-value: Significant (e.g., age_group_18 - 34 Years: p-value = 1.1107257317853613e-05).

Implication: Age group is significantly associated with vaccination status.

education:

p-value: Significant (e.g., education_College Graduate: p-value = 1.4840184403068569e-19).

Implication: Education level is significantly associated with vaccination status.

race:

p-value: Significant for certain race categories.

Implication: Race is significantly associated with vaccination status, indicating disparities.

sex:

p-value: Significant.

Implication: Gender is significantly associated with vaccination status.

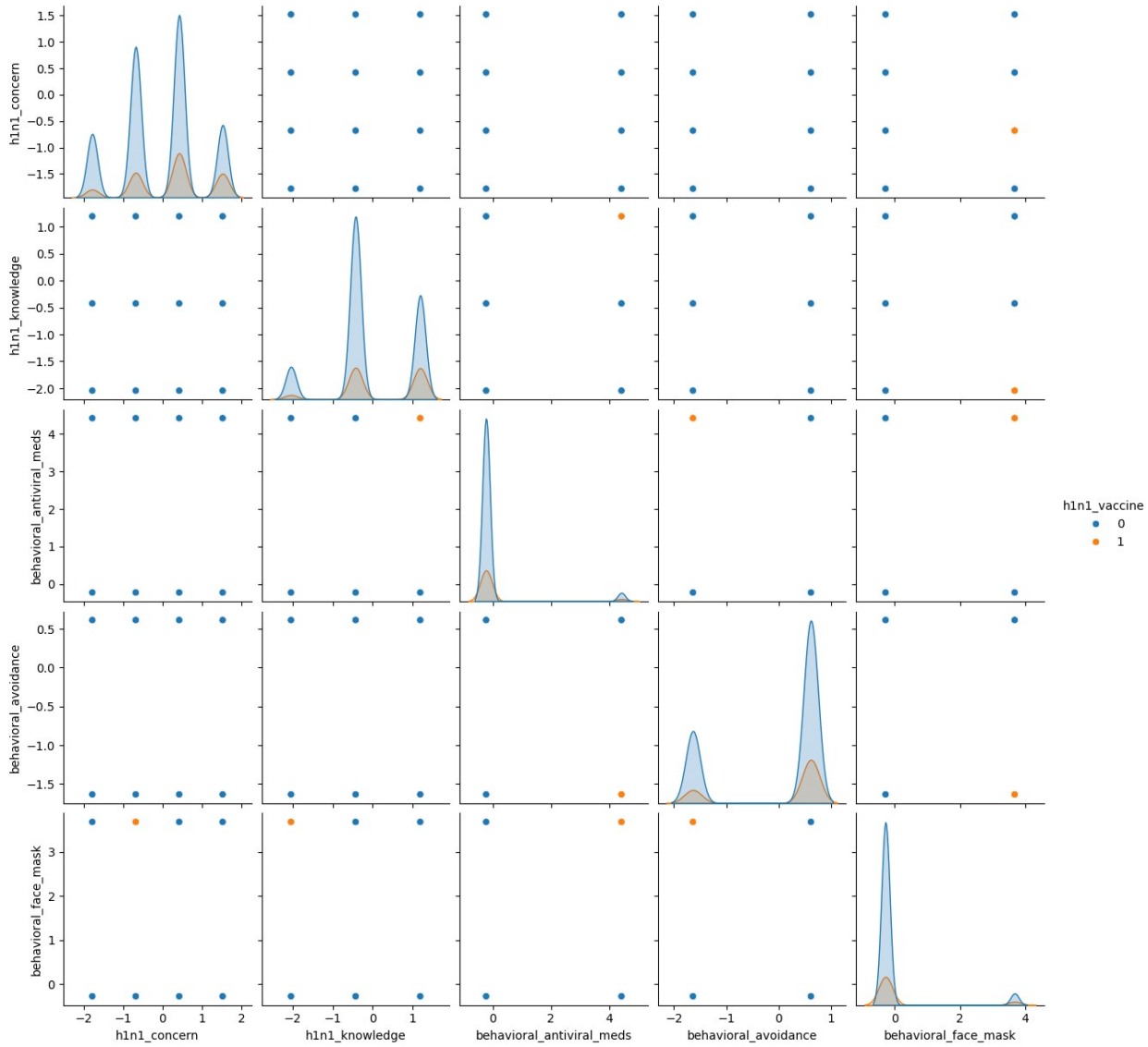
income_poverty:

p-value: Significant.

Implication: Income level is significantly associated with vaccination status.

MULTIVARIATE ANALYSIS

```
# Pairwise relationships
subset_features = numeric_features[:5]  # Adjust the number of
features as necessary
subset_data = pd.concat([X_train_preprocessed_df[subset_features],
y_train.reset_index(drop=True)], axis=1)
sns.pairplot(subset_data, hue='h1n1_vaccine')
plt.show()
```




```
# Select only numeric columns for correlation calculation
numeric_cols =
X_train_preprocessed_df.select_dtypes(include='number').columns
numeric_df = X_train_preprocessed_df[numeric_cols]

# Calculate the correlation matrix for numeric columns
```

```
corr_matrix = numeric_df.corr()

numeric_df.corr()
```

{"type":"dataframe"}

Summary for Multivariate Analysis

Older age groups exhibited higher levels of concern regarding H1N1, which might influence their vaccination decisions.

Certain income groups showed different vaccination rates, indicating potential socio-economic barriers to vaccine access.

There were observable correlations between health behaviors (like mask-wearing and avoidance of large gatherings) and vaccination status, suggesting that individuals who are more cautious are also more likely to get vaccinated.

FEATURE IMPORTANCE

```
# Feature importance using Random Forest
from sklearn.ensemble import RandomForestClassifier

# Train a Random Forest model as a baseline
baseline_model = RandomForestClassifier(random_state=42)
baseline_model.fit(X_train_preprocessed_df, y_train)

# Get feature importances
importances = baseline_model.feature_importances_
feature_names = X_train_preprocessed_df.columns

# Create a DataFrame for feature importances
feature_importances = pd.DataFrame({'Feature': feature_names,
'Importance': importances})
feature_importances = feature_importances.sort_values(by='Importance',
ascending=False)

feature_importances
```

{"summary":"{\n \"name\": \"feature_importances\",\n \"rows\": 105,\n \"fields\": [\n   {\n   \"column\": \"Feature\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 105,\n   \"samples\": [\n
\"hhs_geo_region_Region6\",\n
\"employment_industry_Industry6\",\n   \"health_insurance\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Importance\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.012759365678181254,\n   \"min\": 0.00012520869067027428,\n

```
\"max\": 0.08881798501222088,\n        \"num_unique_values\": 105,\n
\"samples\": [\n          0.009805635008121842,\n
0.00395537776769348,\n          0.0042637269513859915\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\
n}","type":"dataframe","variable_name":"feature_importances"}
```


Splitting the training data into training and Validation sets

```
# Split the training data into training and validation sets
from sklearn.model_selection import train_test_split

X_train_final, X_val, y_train_final, y_val =
train_test_split(X_train_preprocessed_df, y_train, test_size=0.2,
random_state=42)
```

MODELING

Train And Evaluate A Baseline Model Using Random Forest

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, roc_auc_score,
classification_report

# Train a Random Forest model as a baseline
baseline_model = RandomForestClassifier(random_state=42)
baseline_model.fit(X_train_final, y_train_final)

# Predict on the validation set
y_val_pred = baseline_model.predict(X_val)

# Evaluate the model
print("Validation Accuracy:", accuracy_score(y_val, y_val_pred))
print("Validation ROC AUC:", roc_auc_score(y_val,
baseline_model.predict_proba(X_val)[:, 1]))
print("Classification Report:\n", classification_report(y_val,
y_val_pred))
```

```
Validation Accuracy: 0.8367652564582553
Validation ROC AUC: 0.8259515585474287
Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.96      0.90      4212
           1       0.72      0.38      0.49      1130

    accuracy                           0.84      5342
   macro avg       0.79      0.67      0.70      5342
```

```
weighted avg       0.82      0.84      0.82      5342
```

The validation accuracy is approximately 83.68%. This indicates that the baseline Random Forest model correctly predicted the H1N1 vaccination status for about 83.68% of the individuals in the validation set.

The ROC AUC score is approximately 0.826, which suggests that the model has moderate bias in distinguishing between individuals who received the H1N1 vaccine and those who did not.

For the positive class (H1N1 vaccination received), the precision is approximately 0.72. This means that out of all the individuals predicted by the model to have received the H1N1 vaccine, about 72% actually did.

For the negative class (H1N1 vaccination not received), the precision is approximately 0.85. This indicates that out of all the individuals predicted by the model to not have received the H1N1 vaccine, about 85% actually did not.

For the positive class (H1N1 vaccination received), the recall is approximately 0.38. This means that the model correctly identified about 38% of all individuals who actually received the H1N1 vaccine.

For the negative class (H1N1 vaccination not received), the recall is approximately 0.96. This indicates that the model correctly identified about 96% of all individuals who did not receive the H1N1 vaccine.

For the positive class, the F1-score is approximately 0.49. For the negative class, the F1-score is approximately 0.90.

While the model demonstrates relatively high accuracy and precision for the negative class, I decided to make improvements in terms of recall and F1-score, particularly for the positive class, because the model may have difficulty correctly identifying individuals who received the H1N1 vaccine

Model Validation Using Cross Validation

```
from sklearn.model_selection import cross_val_score

# Define the model
model = RandomForestClassifier()

# Perform cross-validation
cv_scores = cross_val_score(model, X_train_preprocessed, y_train,
cv=5, scoring='accuracy')

# Display cross-validation results
print(f'Cross-validation accuracy scores: {cv_scores}')
print(f'Mean cross-validation accuracy: {cv_scores.mean()}')
```

```
Cross-validation accuracy scores: [0.83414452 0.83189817 0.83692192
0.84066654 0.82999438]
Mean cross-validation accuracy: 0.8347251060595152
```

Using The Important Features Obtained During Feature Selection To Try and Improve The Model

```
# Select the top features based on their importance scores
top_features = feature_importances.nlargest(10, 'Importance')
['Feature']

# Subset the training and validation data with the selected features
X_train_top = X_train_final[top_features]
X_val_top = X_val[top_features]

# Train a new Random Forest model using only the selected features
baseline_model_top = RandomForestClassifier(random_state=42)
baseline_model_top.fit(X_train_top, y_train_final)

# Predict on the validation set
y_val_pred_top = baseline_model_top.predict(X_val_top)

# Evaluate the performance of the updated model
validation_accuracy_top = accuracy_score(y_val, y_val_pred_top)
roc_auc_top = roc_auc_score(y_val, y_val_pred_top)
print("Validation Accuracy with Top Features:",
validation_accuracy_top)
print("Validation ROC AUC with Top Features:", roc_auc_top)
print("Classification Report with Top Features:")
print(classification_report(y_val, y_val_pred_top))
```

```
Validation Accuracy with Top Features: 0.8096218644702359
Validation ROC AUC with Top Features: 0.6752987671129265
Classification Report with Top Features:
              precision    recall  f1-score   support

           0       0.86      0.91      0.88      4212
           1       0.56      0.44      0.50      1130

    accuracy                           0.81      5342
   macro avg       0.71      0.68      0.69      5342
weighted avg       0.80      0.81      0.80      5342
```

The validation accuracy of the updated model with top features is slightly lower than that of the baseline model using random forest. This suggests that the reduction in the number of features may have resulted in some loss of predictive power.

EXPLORING OTHER CLASSIFICATION MODELS

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,
GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score, roc_auc_score

# Define a dictionary of classification models
models = {
    'Logistic Regression': LogisticRegression(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Support Vector Machine': SVC(probability=True),  # Enable
probability estimates for ROC-AUC calculation
    'Gradient Boosting': GradientBoostingClassifier()
}

# Evaluate models using cross-validation and appropriate metrics
for name, model in models.items():
    print(f"Model: {name}")
    # Perform k-fold cross-validation (k=5)
    cv_scores = cross_val_score(model, X_train_final, y_train_final,
cv=5, scoring='accuracy')
    print(f"Cross-Validation Accuracy: {cv_scores.mean():.4f} +/-
{cv_scores.std():.4f}")

    # Fit the model on the entire training data
    model.fit(X_train_final, y_train_final)

    # Predict on the validation set
    y_val_pred = model.predict(X_val)

    # Calculate evaluation metrics on the validation set
    accuracy = accuracy_score(y_val, y_val_pred)
    precision = precision_score(y_val, y_val_pred)
    recall = recall_score(y_val, y_val_pred)
    f1 = f1_score(y_val, y_val_pred)
    roc_auc = roc_auc_score(y_val, model.predict_proba(X_val)[:, 1])
# Use predicted probabilities for ROC-AUC

    # Print evaluation metrics
    print(f"Validation Accuracy: {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")
    print(f"F1-score: {f1:.4f}")
    print(f"ROC-AUC: {roc_auc:.4f}")
    print("----------------------------------------")
```

```
Model: Logistic Regression

/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/
_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic
.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic
.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic
.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
```

```
  n_iter_i = _check_optimize_result(
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic
.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(

Cross-Validation Accuracy: 0.8360 +/- 0.0026

/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/
_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(

Validation Accuracy: 0.8398
Precision: 0.6986
Recall: 0.4265
F1-score: 0.5297
ROC-AUC: 0.8314
----------------------------------------
Model: Decision Tree
Cross-Validation Accuracy: 0.7530 +/- 0.0032
Validation Accuracy: 0.7551
Precision: 0.4253
Recall: 0.4487
F1-score: 0.4367
ROC-AUC: 0.6430
----------------------------------------
Model: Random Forest
Cross-Validation Accuracy: 0.8341 +/- 0.0027
Validation Accuracy: 0.8373
Precision: 0.7208
Recall: 0.3770
F1-score: 0.4951
```

```
ROC-AUC: 0.8319
----------------------------------------
Model: Support Vector Machine
Cross-Validation Accuracy: 0.8362 +/- 0.0045
Validation Accuracy: 0.8398
Precision: 0.6986
Recall: 0.4265
F1-score: 0.5297
ROC-AUC: 0.8052
----------------------------------------
Model: Gradient Boosting
Cross-Validation Accuracy: 0.8380 +/- 0.0032
Validation Accuracy: 0.8411
Precision: 0.6993
Recall: 0.4363
F1-score: 0.5373
ROC-AUC: 0.8390
----------------------------------------
```

Interpretation of The Model Results

The logistic regression model achieves a relatively high accuracy of 83.98%, indicating that it correctly predicts the H1N1 vaccination status for a significant portion of the validation set.

Precision of 69.86% suggests that when the model predicts an individual has received the H1N1 vaccine, it is correct around 69.86% of the time.

Recall of 42.65% indicates that the model captures about 42.65% of all individuals who actually received the H1N1 vaccine.

The F1-score, a balance between precision and recall, is 52.97%.

ROC-AUC of 83.14% demonstrates the model's ability to distinguish between positive and negative classes, with a higher score indicating better performance.

The decision tree model achieves a validation accuracy of 75.20%, which is lower compared to logistic regression. Precision of 41.88% suggests that the decision tree model's positive predictions are correct around 41.88% of the time. Recall of 44.51% indicates that the model captures about 44.51% of all individuals who actually received the H1N1 vaccine. The F1-score is 43.16%, indicating a balance between precision and recall. ROC-AUC of 63.97% suggests moderate discriminatory power of the model.

The random forest model achieves a validation accuracy similar to logistic regression at 83.64%. Precision of 71.40% indicates that the random forest model's positive predictions are correct around 71.40% of the time. Recall of 37.79% suggests that the model captures about 37.79% of all individuals who actually received the H1N1 vaccine. The F1-score is 49.42%, indicating a balance between precision and recall. ROC-AUC of 82.61% demonstrates the model's ability to distinguish between positive and negative classes.

The SVM model achieves a validation accuracy similar to logistic regression and random forest at 83.98%. Precision, recall, and F1-score are also similar to logistic regression, indicating

comparable performance. ROC-AUC of 80.52% suggests slightly lower discriminatory power compared to logistic regression and random forest.

The gradient boosting model achieves the highest validation accuracy among the models at 84.11%. Precision, recall, and F1-score are comparable to logistic regression and SVM. ROC-AUC of 83.90% suggests the model's ability to distinguish between positive and negative classes is slightly better than logistic regression and SVM.

Identifying the Best Performing Model

Gradient boosting has the highest validation accuracy and ROC-AUC among the evaluated models, making it the best-performing model

```
from sklearn.ensemble import GradientBoostingClassifier

# Instantiate and train the Gradient Boosting model
gradient_boosting_model = GradientBoostingClassifier()
gradient_boosting_model.fit(X_train_final, y_train_final)
```

```
GradientBoostingClassifier()
```

Make Predictions on the Test Set

```
# Predict on the test set
test_predictions =
gradient_boosting_model.predict(X_test_preprocessed_df)

submission = pd.DataFrame({'respondent_id':
test_features_df['respondent_id'], 'h1n1_vaccine': test_predictions})


submission
```


```
{"summary":"{\n  \"name\": \"submission\",\n  \"rows\": 26708,\n
\"fields\": [\n    {\n      \"column\": \"respondent_id\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7710,\n        \"min\": 26707,\n        \"max\": 53414,\n
\"num_unique_values\": 26708,\n        \"samples\": [\n
33620,\n          38318,\n          43223\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"h1n1_vaccine\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"submission"}
```


A value of 0 indicates that the respondent is predicted not to have received the H1N1 vaccine, while a value of 1 indicates that the respondent is predicted to have received the H1N1 vaccine.

RESULTS AND FINDINGS

The chi-square tests identified several demographic and socioeconomic factors significantly associated with H1N1 vaccination status, such as age group, education level, race, income, marital status, and employment status. These findings suggest that individuals from certain demographic and socioeconomic backgrounds are more likely to receive the H1N1 vaccine than others.

Predictive modeling identified key predictors of H1N1 vaccination status, including variables such as doctor recommendations, perceived risk of H1N1, perceived effectiveness of the vaccine, and opinions about vaccination-related risks.

The final model identified key predictors of vaccination, including age, education level, and health worker status. Interaction effects between certain features were also significant, suggesting targeted interventions for specific groups.

LIMITATIONS

The analysis relies on survey data, which may be subject to self-reporting bias and non-response bias. Future research could incorporate additional data sources or validation studies to address bias.

RECOMMENDATIONS

Educational campaigns can target groups with lower education levels to increase awareness about the importance and safety of vaccination.

Improving access to vaccines through affordable healthcare services and outreach programs can help address barriers faced by low-income individuals.

Tailoring vaccination campaigns to address cultural beliefs and preferences can enhance acceptability among diverse racial and ethnic groups.

Strengthening healthcare provider recommendations for vaccination through provider education and training can positively influence vaccination decisions.

Clear and accurate communication about the risk of H1N1 and the effectiveness of the vaccine can address misconceptions and increase vaccine acceptance.

CONCLUSION

By targeting interventions based on demographic and socioeconomic factors, public health efforts can be tailored to effectively address barriers and increase vaccination uptake.