ANALYZING VACCINATION COVERAGE TRENDS DURING THE 2009 H1N1 PANDEMIC

Submitted to: CSE 587: Data Intensive Computing

Fall'2023

Manali Ramchandani, MS
Syed Razauddin Shahlal, MS
Tajammul Shuja Sayyad, MS

University at Buffalo
School of Engineering and Applied Sciences

Deliverables

1. **Problem Statement:**

   The 2009 H1N1 pandemic posed significant challenges for public health authorities, necessitating rapid and effective vaccination campaigns. This study aims to analyze the vaccination coverage trends during the 2009-2010 flu season, with a primary focus on the distribution and uptake of monovalent pH1N1 and trivalent seasonal influenza vaccines. Additionally, it investigates the impact of socioeconomic and demographic factors on vaccination rates.

   a. The 2009 H1N1 pandemic presented a unique challenge in terms of public health response, including the distribution and uptake of monovalent pH1N1 and trivalent seasonal influenza vaccines. This study aims to analyze vaccination coverage trends during the 2009-2010 flu season and assess the influence of socioeconomic factors on vaccination rates. This problem is significant because it addresses the need to assess the impact of socioeconomic factors on vaccination coverage. Such an assessment can inform future vaccination campaigns, help identify vulnerable populations, and guide resource allocation in response to pandemics and other public health crises.

   b. This project's potential contribution is in shedding light on the relationship between socioeconomic factors and vaccination coverage. By identifying the barriers and disparities in vaccine uptake, public health authorities can develop more effective strategies. Understanding this link is crucial for enhancing overall vaccination coverage and promoting health equity.

2. **Data Sources:**

   The primary data source for this study is the National 2009 H1N1 Flu Survey (NHFS), conducted by the Centers for Disease Control and Prevention (CDC). The NHFS dataset, comprising over 2000 records, provides comprehensive information on vaccination coverage, socioeconomic variables, and other relevant demographics. This dataset offers a substantial sample size for robust analysis, and it aligns with ethical data use guidelines, ensuring the privacy and confidentiality of individuals.

Citing the NHFS dataset is essential to maintain data integrity and transparency. The dataset is well-suited to address the research questions and objectives of this project, allowing for a thorough examination of vaccination coverage trends and their relationship with socioeconomic factors.

[1] https://www.drivendata.org/competitions/66/flu-shot-learning/page/211/

Data Link

Data Description:

The data for this comes from the National 2009 H1N1 Flu Survey (NHFS).

Labels:

For this competition, there are two target variables:

- h1n1_vaccine - Whether respondent received H1N1 flu vaccine.
- seasonal vaccine - Whether respondent received seasonal flu vaccine.

Both are binary variables: 0 = No; 1 = Yes. Some respondents didn't get either vaccine, others got only one, and some got both. This is formulated as a multilabel (and *not* multiclass) problem.

The features in this dataset:

We are provided with a dataset with 36 columns. The first column respondent_id is a unique and random identifier. The remaining 35 features are described below.

For all binary variables: 0 = No; 1 = Yes.

- h1n1_concern - Level of concern about the H1N1 flu.
  0 = Not at all concerned; 1 = Not very concerned; 2 = Somewhat concerned; 3 = Very concerned.
- h1n1_knowledge - Level of knowledge about H1N1 flu. 0 = No knowledge; 1 = A little knowledge; 2 = A lot of knowledge.
- behavioral_antiviral_meds - Has taken antiviral medications. (binary)

- behavioral_avoidance - Has avoided close contact with others with flu-like symptoms. (binary)

- behavioral_face_mask - Has bought a face mask. (binary)
- behavioral_wash_hands - Has frequently washed hands or used hand sanitizer. (binary)
- behavioral_large_gatherings - Has reduced time at large gatherings. (binary)
- behavioral_outside_home - Has reduced contact with people outside of own household. (binary)
- behavioral_touch_face - Has avoided touching eyes, nose, or mouth. (binary)
- doctor_recc_h1n1 - H1N1 flu vaccine was recommended by doctor. (binary)
- doctor_recc_seasonal - Seasonal flu vaccine was recommended by doctor. (binary)
- chronic_med_condition - Has any of the following chronic medical conditions: asthma or another lung condition, diabetes, a heart condition, a kidney condition, sickle cell anemia or other anemia, a neurological or neuromuscular condition, a liver condition, or a weakened immune system caused by a chronic illness or by medicines taken for a chronic illness. (binary)
- child_under_6_months - Has regular close contact with a child under the age of six months. (binary)
- health_worker - Is a healthcare worker. (binary)
- health_insurance - Has health insurance. (binary)
- opinion_h1n1_vacc_effective - Respondent's opinion about H1N1 vaccine effectiveness.

  1 = Not at all effective; 2 = Not very effective; 3 = Don't know; 4 = Somewhat effective; 5 = Very effective.
- opinion_h1n1_risk - Respondent's opinion about risk of getting sick with H1N1 flu without vaccine.

  1 = Very Low; 2 = Somewhat low; 3 = Don't know; 4 = Somewhat high; 5 = Very high.
- opinion_h1n1_sick_from_vacc - Respondent's worry of getting sick from taking H1N1 vaccine.

  1 = Not at all worried; 2 = Not very worried; 3 = Don't know; 4 = Somewhat worried; 5 = Very worried.
- opinion_seas_vacc_effective - Respondent's opinion about seasonal flu vaccine effectiveness.

  1 = Not at all effective; 2 = Not very effective; 3 = Don't know; 4 = Somewhat effective; 5 = Very effective.

- opinion_seas_risk - Respondent's opinion about risk of getting sick with seasonal flu without vaccine.

  1 = Very Low; 2 = Somewhat low; 3 = Don't know; 4 = Somewhat high; 5 = Very high.
- opinion_seas_sick_from_vacc - Respondent's worry of getting sick from taking seasonal flu vaccine.

  1 = Not at all worried; 2 = Not very worried; 3 = Don't know; 4 = Somewhat worried; 5 = Very worried.
- age_group - Age group of respondents.
- education - Self-reported education level.
- race - Race of respondent.
- sex - Sex of respondent.
- income_poverty - Household annual income of respondent with respect to 2008 Census poverty thresholds.
- marital_status - Marital status of respondent.
- rent_or_own - Housing situation of respondent.
- employment_status - Employment status of respondent.
- hhs_geo_region - Respondent's residence using a 10-region geographic classification defined by the U.S. Dept. of Health and Human Services. Values are represented as short random character strings.
- census_msa - Respondent's residence within metropolitan statistical areas (MSA) as defined by the U.S. Census.
- household_adults - Number of *other* adults in household, top-coded to 3.
- household_children - Number of children in household, top-coded to 3.
- employment_industry - Type of industry respondent is employed in. Values are represented as short random character strings.
- employment_occupation - Type of occupation of respondent. Values are represented as short random character strings.

3. Data Cleaning/Processing:

Introduction:

The objective of this project is to predict the likelihood of individuals receiving their H1N1 and seasonal flu vaccines. The dataset consists of various features related to individual demographics, behaviors, and opinions. This report chronicles the data cleaning and exploratory data analysis (EDA) steps undertaken to prepare the dataset for modeling.

1. Merging and Initial Exploration:

The feature set and label set were read from two separate CSV files and merged based on the 'respondent_id' to form a single dataset. An initial inspection revealed a total of 38 columns with varied datatypes and missing values.

```
df_features=pd.read_csv('training_set_features.csv')
df_labels=pd.read_csv('training_set_labels.csv')
```

```
df = df_features.merge(df_labels, on='respondent_id')
```

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 26707 entries, 0 to 26706
Data columns (total 38 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   respondent_id                26707 non-null  int64  
 1   h1n1_concern                 26615 non-null  float64
 2   h1n1_knowledge               26591 non-null  float64
 3   behavioral_antiviral_meds    26636 non-null  float64
 4   behavioral_avoidance         26499 non-null  float64
 5   behavioral_face_mask         26688 non-null  float64
 6   behavioral_wash_hands        26665 non-null  float64
 7   behavioral_large_gatherings  26620 non-null  float64
 8   behavioral_outside_home      26625 non-null  float64
 9   behavioral_touch_face        26579 non-null  float64
 10  doctor_recc_h1n1             24547 non-null  float64
 11  doctor_recc_seasonal         24547 non-null  float64
 12  chronic_med_condition        25736 non-null  float64
 13  child_under_6_months         25887 non-null  float64
 14  health_worker                25903 non-null  float64
 15  health_insurance             14433 non-null  float64
 16  opinion_h1n1_vacc_effective  26316 non-null  float64
 17  opinion_h1n1_risk            26319 non-null  float64
 18  opinion_h1n1_sick_from_vacc  26312 non-null  float64
 19  opinion_seas_vacc_effective  26245 non-null  float64
 20  opinion_seas_risk            26193 non-null  float64
 21  opinion_seas_sick_from_vacc  26170 non-null  float64
 22  age_group                    26707 non-null  object 
 23  education                    25300 non-null  object 
 24  race                         26707 non-null  object 
 25  sex                          26707 non-null  object 
 26  income_poverty               22284 non-null  object 
 27  marital_status               25299 non-null  object 
 28  rent_or_own                  24665 non-null  object 
 29  employment_status            25244 non-null  object 
 30  hhs_geo_region               26707 non-null  object 
 31  census_msa                   26707 non-null  object 
 32  household_adults             26458 non-null  float64
 33  household_children           26458 non-null  float64
 34  employment_industry          13377 non-null  object 
 35  employment_occupation        13237 non-null  object 
 36  h1n1_vaccine                 26707 non-null  int64  
 37  seasonal_vaccine             26707 non-null  int64  
dtypes: float64(23), int64(3), object(12)
memory usage: 7.9+ MB
```

2. Duplicate Values:

On inspection, it was determined that the dataset did not contain any duplicate values.

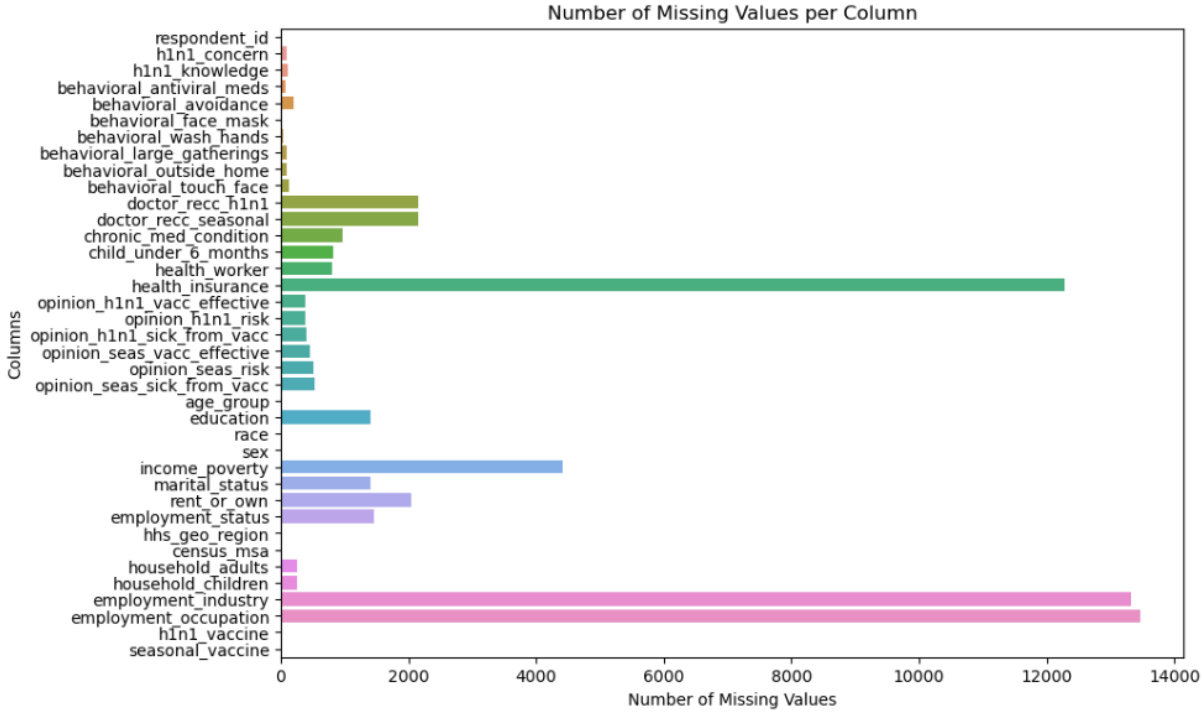
```
1 df.duplicated().sum()
```

```
0
```

3. Handling Missing Values:

a. Bar plot for Number of Missing Values per Column:

A bar plot was generated to visualize the number of missing values per column. The columns **health_insurance**, **employment_industry**, and **employment_occupation** had significant missing values.



b. Numerical Columns:

All missing values in numerical columns were imputed with the mode of their respective columns.

```
for column in numerical_columns:

    if df[column].isnull().any():

        mode_val = df[column].mode()[0]
        df[column].fillna(mode_val, inplace=True)
```

c. Categorical Columns:

Similarly, missing values in categorical columns were replaced with the mode of their respective columns.

```
for column in categ_columns:

    if df[column].isnull().any():

        mode_val = df[column].mode()[0]
        df[column].fillna(mode_val, inplace=True)
```

We have removed all the missing values for now.

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 26707 entries, 0 to 26706
Data columns (total 38 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   respondent_id                26707 non-null  int64  
 1   h1n1_concern                 26707 non-null  float64
 2   h1n1_knowledge               26707 non-null  float64
 3   behavioral_antiviral_meds    26707 non-null  float64
 4   behavioral_avoidance         26707 non-null  float64
 5   behavioral_face_mask         26707 non-null  float64
 6   behavioral_wash_hands        26707 non-null  float64
 7   behavioral_large_gatherings  26707 non-null  float64
 8   behavioral_outside_home      26707 non-null  float64
 9   behavioral_touch_face        26707 non-null  float64
 10  doctor_recc_h1n1             26707 non-null  float64
 11  doctor_recc_seasonal         26707 non-null  float64
 12  chronic_med_condition        26707 non-null  float64
 13  child_under_6_months         26707 non-null  float64
 14  health_worker                26707 non-null  float64
 15  health_insurance             26707 non-null  float64
 16  opinion_h1n1_vacc_effective  26707 non-null  float64
 17  opinion_h1n1_risk            26707 non-null  float64
 18  opinion_h1n1_sick_from_vacc  26707 non-null  float64
 19  opinion_seas_vacc_effective  26707 non-null  float64
 20  opinion_seas_risk            26707 non-null  float64
 21  opinion_seas_sick_from_vacc  26707 non-null  float64
 22  age_group                    26707 non-null  object 
 23  education                    26707 non-null  object 
 24  race                         26707 non-null  object 
 25  sex                          26707 non-null  object 
 26  income_poverty               26707 non-null  object 
 27  marital_status               26707 non-null  object 
 28  rent_or_own                  26707 non-null  object 
 29  employment_status            26707 non-null  object 
 30  hhs_geo_region               26707 non-null  object 
 31  census_msa                   26707 non-null  object 
 32  household_adults             26707 non-null  float64
 33  household_children           26707 non-null  float64
 34  employment_industry          26707 non-null  object 
 35  employment_occupation        26707 non-null  object 
 36  h1n1_vaccine                 26707 non-null  int64  
 37  seasonal_vaccine             26707 non-null  int64  
dtypes: float64(23), int64(3), object(12)
```

4. Outlier Detection (using frequency analysis):
By observing the frequency distributions of the columns, potential outliers were identified. However, given the nature of the dataset, these "outliers" were retained for further analysis.

5. Text Data Cleaning:
Columns such as **hhs_geo_region**, **employment_industry**, and **employment_occupation** contained cryptic string values. These were mapped to more interpretable labels like 'region 1', 'industry 1', etc., for better clarity and visualization.

5) Text data cleaning for region

```
regions=['lzgpxyit', 'fpwskwrf', 'qufhixun', 'oxchjgsf', 'kbazzjca', 'bhuqouqj',
       'mlyzmhmf', 'lrircsnp', 'atmpeygn', 'dqpwygqj']
region_dict={}
for i in range(10):
    region_dict[regions[i]]=f'region {i+1}'

def region_name(r):
    return region_dict[r]
```

```
df['hhs_geo_region_modified']=df['hhs_geo_region'].apply(region_name)
```

6) Text data cleaning for industry

```
df['employment_industry'].unique()
```

```
array(['fcxhlnwr', 'pxcmvdjn', 'rucpziij', 'wxleyezf', 'saaquncn',
       'xicduogh', 'ldnlellj', 'wlfvacwt', 'nduyfdeo', 'vjjrobsf',
       'arjwrbjb', 'atmlpfrs', 'msuufmds', 'xqicxuve', 'phxvnwax',
       'dotnnunm', 'mfikgejo', 'cfqqtusy', 'mcubkhph', 'haxffmxo',
       'qnlwzans'], dtype=object)
```

```
industry=['fcxhlnwr', 'pxcmvdjn', 'rucpziij', 'wxleyezf', 'saaquncn',
       'xicduogh', 'ldnlellj', 'wlfvacwt', 'nduyfdeo', 'vjjrobsf',
       'arjwrbjb', 'atmlpfrs', 'msuufmds', 'xqicxuve', 'phxvnwax',
       'dotnnunm', 'mfikgejo', 'cfqqtusy', 'mcubkhph', 'haxffmxo',
       'qnlwzans']
```

```
industry_dict={}
for i in range(len(industry)):
    industry_dict[industry[i]]=f'industry {i+1}'

def industry_name(r):
    return industry_dict[r]
```

```
df['employment_industry_modified']=df['employment_industry'].apply(industry_name)
```

6. Encoding Categorical Columns:

Categorical columns were one-hot encoded to convert them into a format suitable for machine learning models. The original columns were then dropped to avoid redundancy.

8) Encoding Categorical columns

But first we will have to delete the columns industry and occupation and they have many missing values. For now we will proceed without them.

```
categ_cols=['age_group',
 'education',
 'race',
 'sex',
 'income_poverty',
 'marital_status',
 'rent_or_own',
 'employment_status',
 'hhs_geo_region_modified',
 'census_msa']
```

```
df.drop(columns=['employment_industry_modified','employment_occupation_modified'],inplace=True)
```

```
df=pd.get_dummies(df,drop_first=True)
```

7. Encoding Numerical Columns:

Considering that the numerical columns contained binary or discrete positive integer values, they too were one-hot encoded.

9) Encoding numerical columns

As our numerical columns also have binary values and values those are discrete positive integer. We can convert them to dummy varaibles as well and later check whether nominal or ordinal encoding works.

```
1 df=pd.get_dummies(df,columns=numerical_columns[1:],drop_first=True)
```

8. Data Splitting:

The dataset was split into features (X) and target variables (**h1n1_vaccine** and **seasonal_vaccine**) (y).

```
1 y=df[['h1n1_vaccine','seasonal_vaccine']]
```

```
1 X=df.drop(columns=['h1n1_vaccine','seasonal_vaccine'])
```

9. Feature Engineering Using MCA:

Multiple Correspondence Analysis (MCA) was applied for dimensionality reduction on the one-hot encoded categorical columns.

```
1 import prince
```

```
1 mca = prince.MCA()
2 mca_fit = mca.fit_transform(X)
```

```
1 mca_fit
```

	0	1
0	-0.066717	0.429501
1	0.206415	-0.019460
2	-0.273362	0.228814
3	0.282959	0.435534
4	-0.063625	0.180143
...	...	...
26702	-0.044062	0.233159
26703	-0.113557	-0.158918
26704	0.128216	-0.174698
26705	-0.166956	0.441623
26706	-0.018449	0.089590

26707 rows × 2 columns

10. Feature Engineering Using Truncated SVD:

Truncated Singular Value Decomposition (SVD) was applied to further reduce the dimensionality of the dataset.

```
1 from sklearn.decomposition import TruncatedSVD
2 svd = TruncatedSVD(n_components=10)
3 reduced_data = svd.fit_transform(X)
```

```
1 reduced_data
```

```
array([[ 2.01523393,  0.11682187, -1.01582408, ...,  0.61896094,
        -0.41725369,  0.11800925],
       [ 2.90581034,  0.52702323,  0.06288554, ...,  0.62982194,
        -0.09898879,  1.00408243],
       [ 2.16787878, -0.8192168 , -0.66410511, ...,  0.26901638,
         0.26813073,  0.3829792 ],
       ...,
       [ 2.96372998,  0.60163938,  0.26353493, ..., -0.87968879,
        -1.24550125, -0.42184107],
       [ 1.40570436, -0.47082602, -0.79571095, ...,  0.60115755,
        -1.00553259, -0.35139146],
       [ 1.94624351,  0.53913893,  0.12950637, ...,  0.16925163,
         0.07593415,  0.46502966]])
```

Points to Remember:

- Outliers were identified but not removed.
- Certain columns with significant missing values were dropped.
- Both original and one-hot encoded versions of numerical columns will be used for modeling in the subsequent stages.

4. Exploratory Data Analysis:

The Exploratory Data Analysis (EDA) was conducted on a dataset aimed at predicting the likelihood of individuals getting their H1N1 and seasonal flu vaccines. The objective was to gain insights, identify patterns, and prepare the data for modeling.

1. General Characteristics of Data

1) General Characteristic of data- Descriptive Statistics

```
1 df.describe()
```

	h1n1_concern	h1n1_knowledge	behavioral_antiviral_meds	behavioral_avoidance	behavioral_face_mask	behavioral_wash_hands	behavioral_large_gatherir
count	26707.000000	26707.000000	26707.000000	26707.000000	26707.000000	26707.000000	26707.000(
mean	1.619800	1.261392	0.048714	0.727749	0.068933	0.825888	0.357
std	0.909016	0.617047	0.215273	0.445127	0.253345	0.379213	0.479
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000(
25%	1.000000	1.000000	0.000000	0.000000	0.000000	1.000000	0.000(
50%	2.000000	1.000000	0.000000	1.000000	0.000000	1.000000	0.000(
75%	2.000000	2.000000	0.000000	1.000000	0.000000	1.000000	1.000(
max	3.000000	2.000000	1.000000	1.000000	1.000000	1.000000	1.000(

8 rows × 53 columns

Descriptive Statistics: A general overview of the data was obtained using descriptive statistics. This helps in understanding the central tendency, spread, and shape of the dataset's distribution.

2. Target Distribution Analysis

We have a multi label classification problem so lets check the distribution of each.

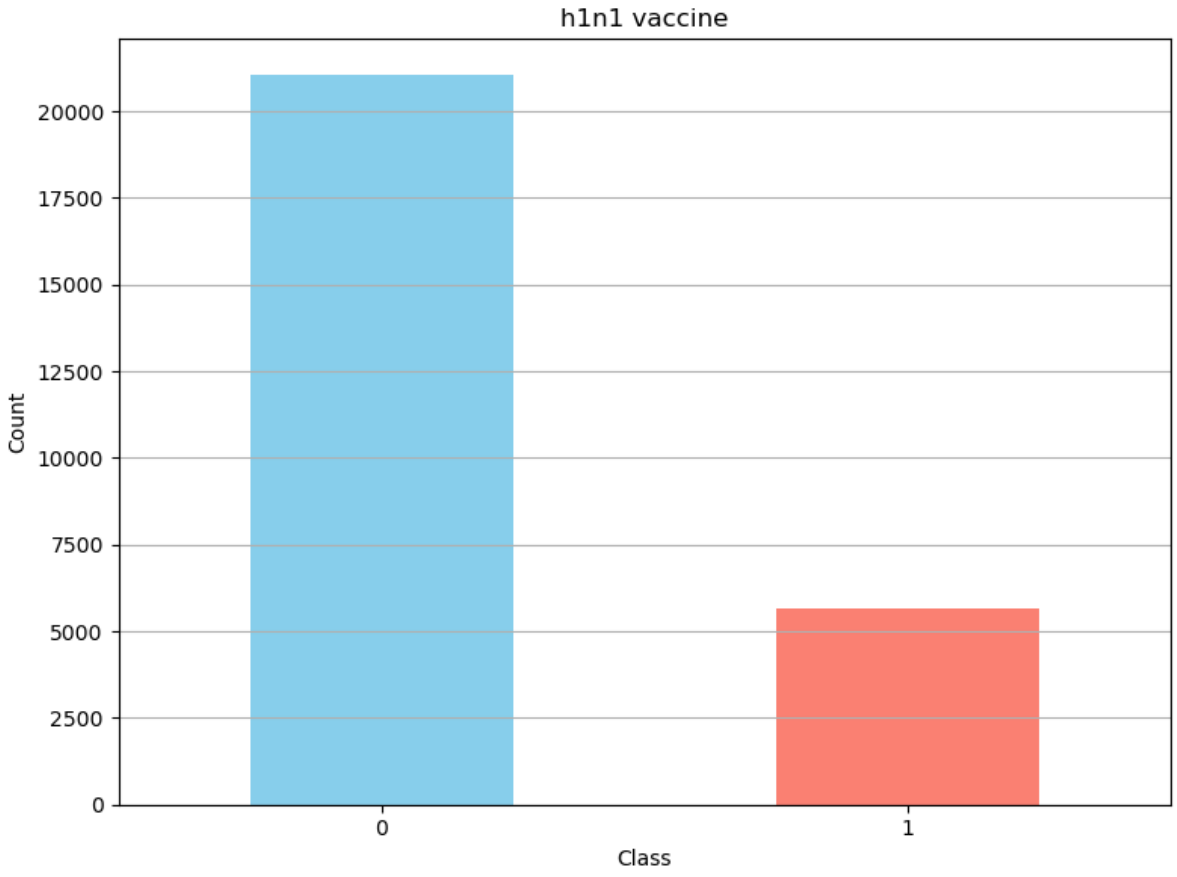
```
1 y['h1n1_vaccine'].value_counts()
```

```
0    21033
1     5674
Name: h1n1_vaccine, dtype: int64
```

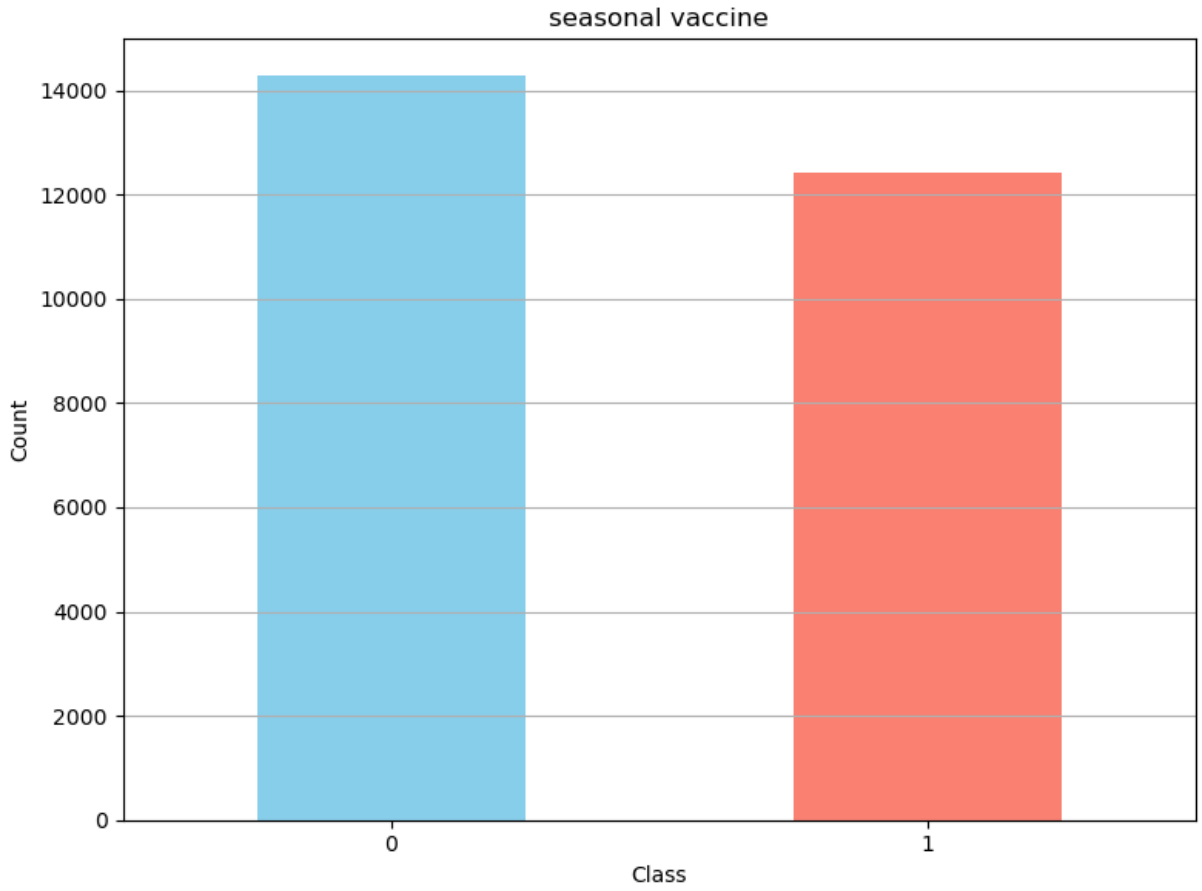
```
1 y['seasonal_vaccine'].value_counts()
```

```
0    14272
1    12435
Name: seasonal_vaccine, dtype: int64
```

```python
class_counts = y['h1n1_vaccine'].value_counts()

# Plot
plt.figure(figsize=(8, 6))
class_counts.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('h1n1 vaccine')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```

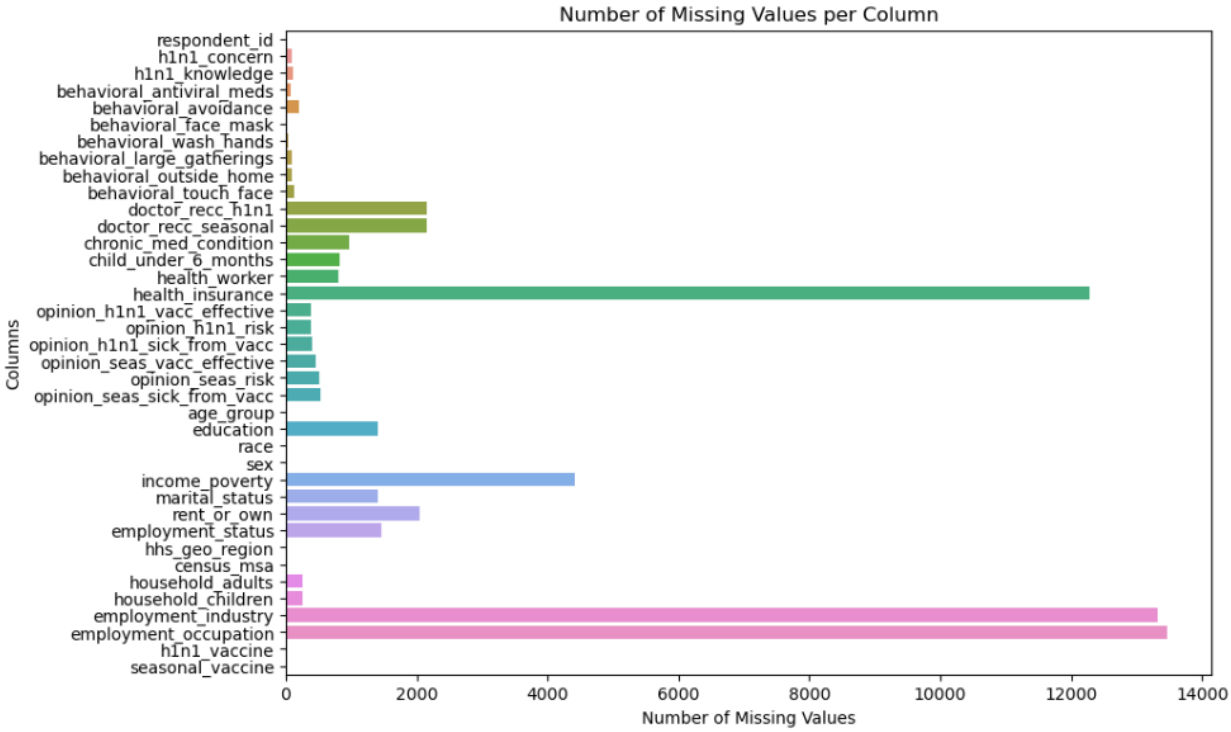


```python
class_counts = y['seasonal_vaccine'].value_counts()

# Plot
plt.figure(figsize=(8, 6))
class_counts.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('seasonal vaccine')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```



- Observations:
  - The distribution for h1n1_vaccine is skewed, suggesting possible class imbalance issues.
  - The distribution for seasonal_vaccine is more balanced.
- Implication: Techniques like SMOTE might be required to deal with potential sampling issues for the h1n1_vaccine target.

3. Handling Missing Data



Visual examination revealed the distribution of missing data across columns. This step is critical as missing data can lead to biased models if not handled appropriately. As we can see that employment_industry and employment_occupation have nearly 50% missing values we can drop these from our feature set.

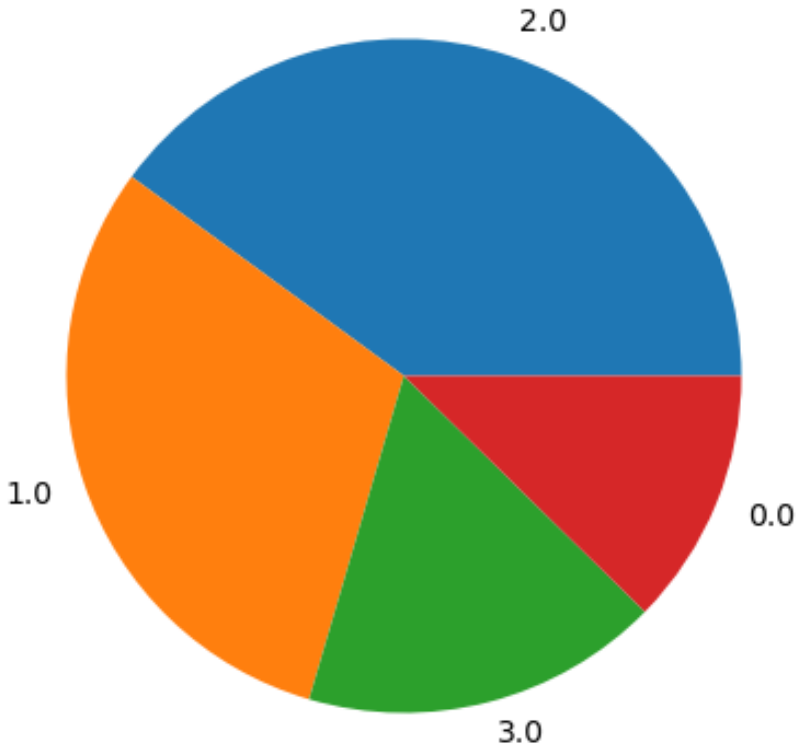
4. Univariate Analysis

h1n1 concern

From the below plot we can see that the h1n1 values are equally distributed.

```
1 plot_and_display_valuecounts(df,'h1n1_concern',False)
```

	h1n1_concern	Value Count	Percentage
0	2.0	10667	39.940839
1	1.0	8153	30.527577
2	3.0	4591	17.190250
3	0.0	3296	12.341334

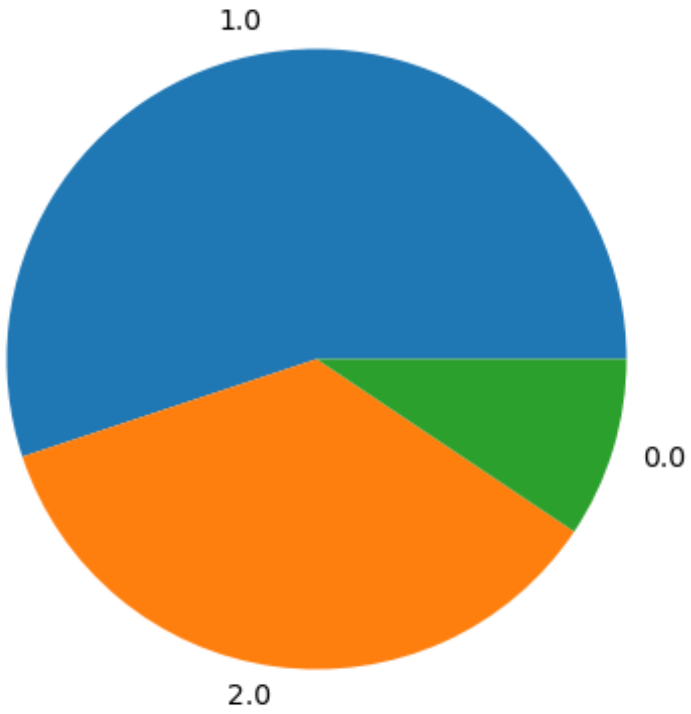


h1n1 knowledge

Distribution of 0 (no knowledge is very less) which means most people knew about h1n1

```
1 plot_and_display_valuecounts(df,'h1n1_knowledge',False)
```

	h1n1_knowledge	Value Count	Percentage
0	1.0	14714	55.094170
1	2.0	9487	35.522522
2	0.0	2506	9.383308



Insights were derived from individual features using bar plots. For instance, the majority had some knowledge of H1N1, indicating general awareness among the respondents.

5. Multivariate Analysis

Mosaic Plot:

```
1 mosaic(df, ['health_worker', 'health_insurance'])
2 plt.title('Mosaic Plot of health_worker vs. health_insurance')
3 plt.show()
```

The relationship between being a health worker and having health insurance was explored. A clear correlation was observed, indicating potential multicollinearity.

Implication: One of the correlated features might need to be dropped during modeling to prevent multicollinearity issues.

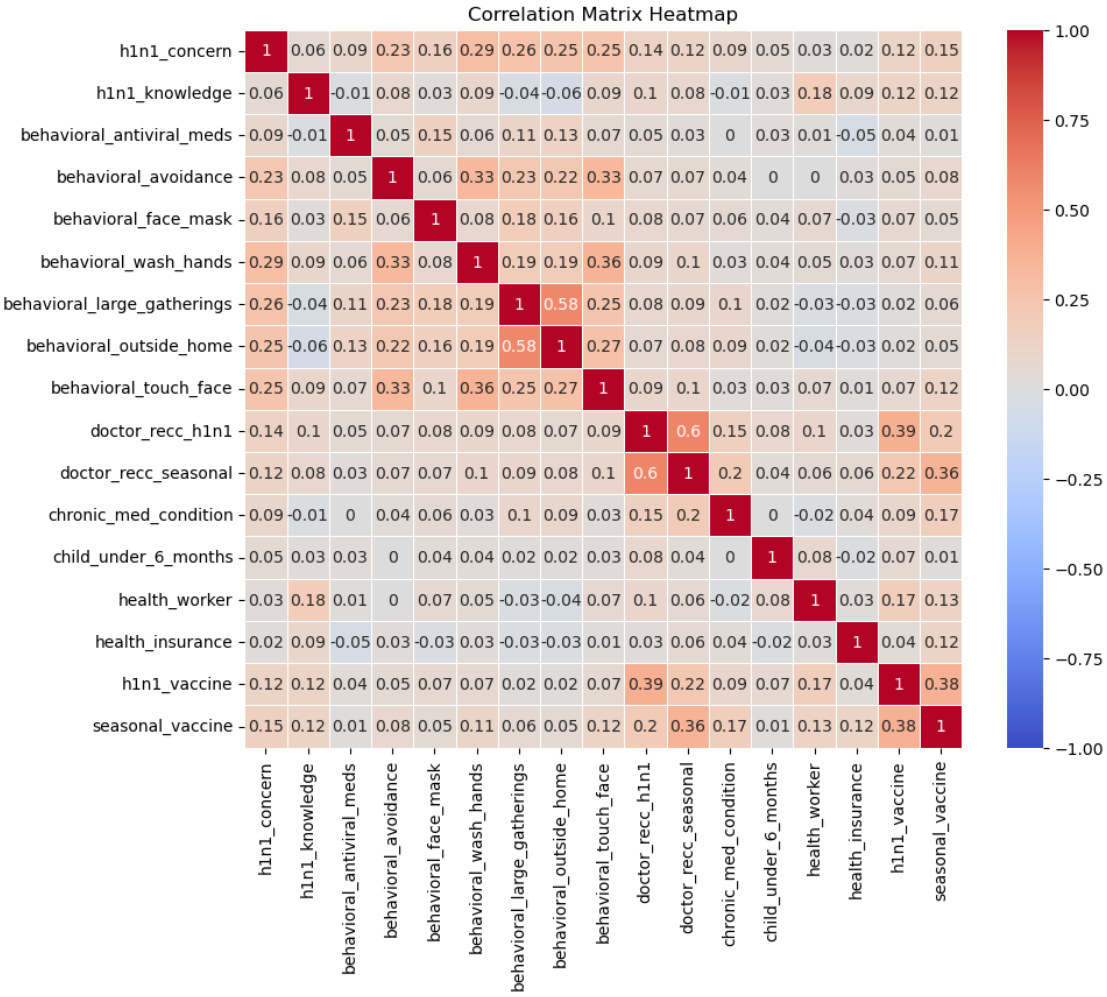
Observations

- Almost all health workers (health_worker = 1.0) have health insurance (health_insurance = 1.0).
- Among those who are not health workers, a vast majority do not have health insurance.

So, from this we can conclude that if remove one of the features it won't affect the classification as these are correlated and thus this can be used in feature reduction.

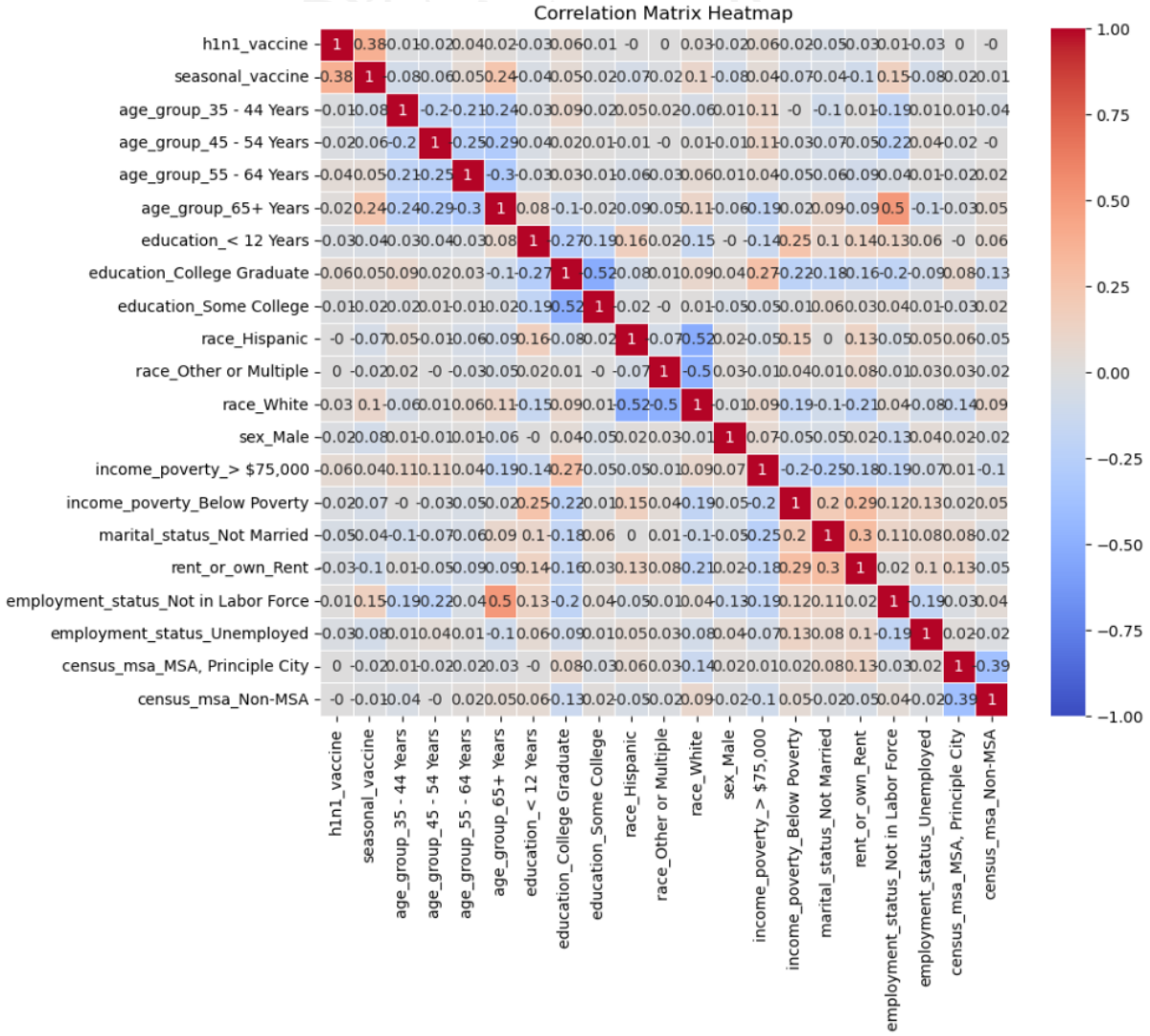
6. Correlation Analysis

Heatmaps were used to visualize correlations for binary features.



Observations: As the feature set is huge, we will check the numerical features first and check its correlation with the target and among themselves. Normally we drop those features that are highly correlated among themselves as they have redundant information. From the plot above only doctor_res_Seasonal and doctor_Rec_h1n1 have a considerable correlation of 0.6. A correlation of >0.8 is considered high.

Heatmaps were used to visualize correlations for categorical features.



Observations : No significant correlations were observed among the features, which is good for modeling. The only notable correlation was between doctor recommendations for H1N1 and seasonal vaccines.

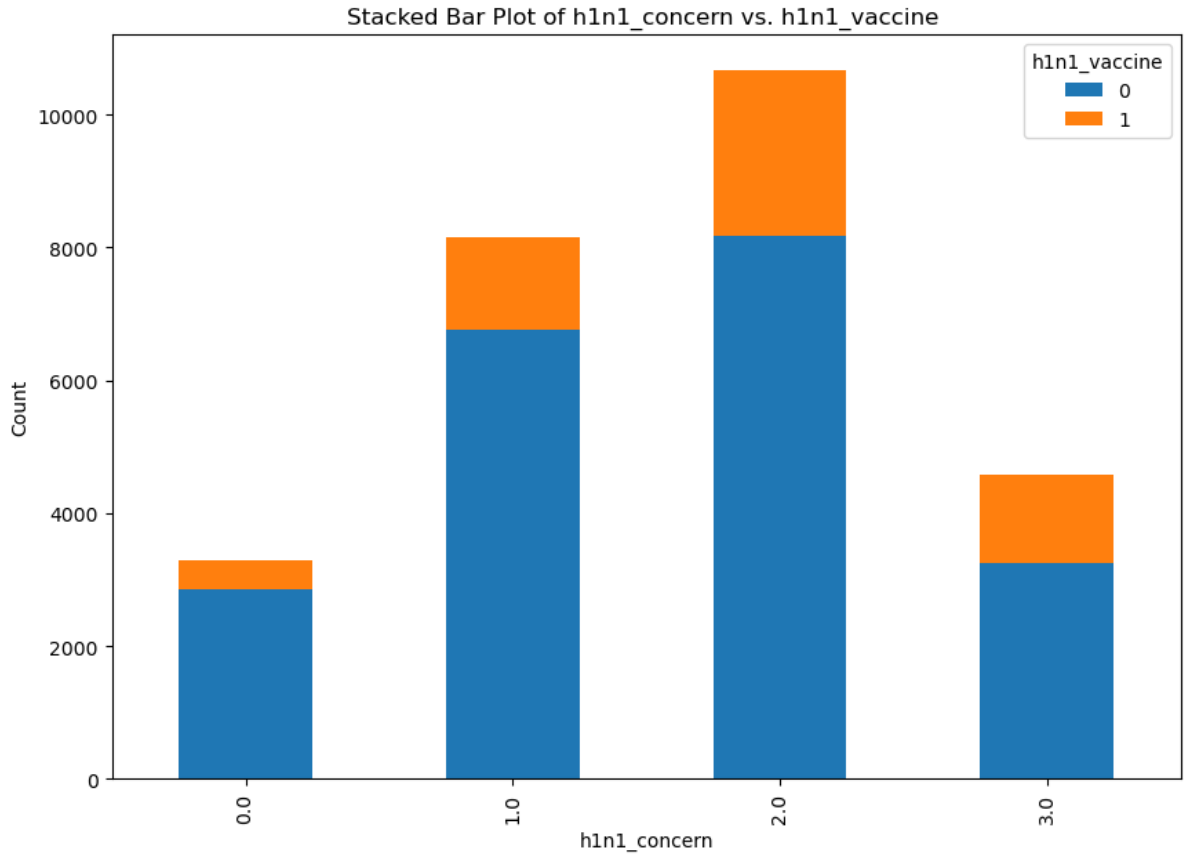
7. Distribution Analysis with Target

h1n1 vaccine and h1n1 concern

Observations: For the group with the lowest concern (0.0), a higher number of people did not get vaccinated compared to those who did. As the level of concern increases to 1.0 and 2.0, the number of people who got vaccinated exceeds those who did not. For the highest concern level (3.0), while more people got vaccinated, there's still a significant portion who chose not to get the vaccine.

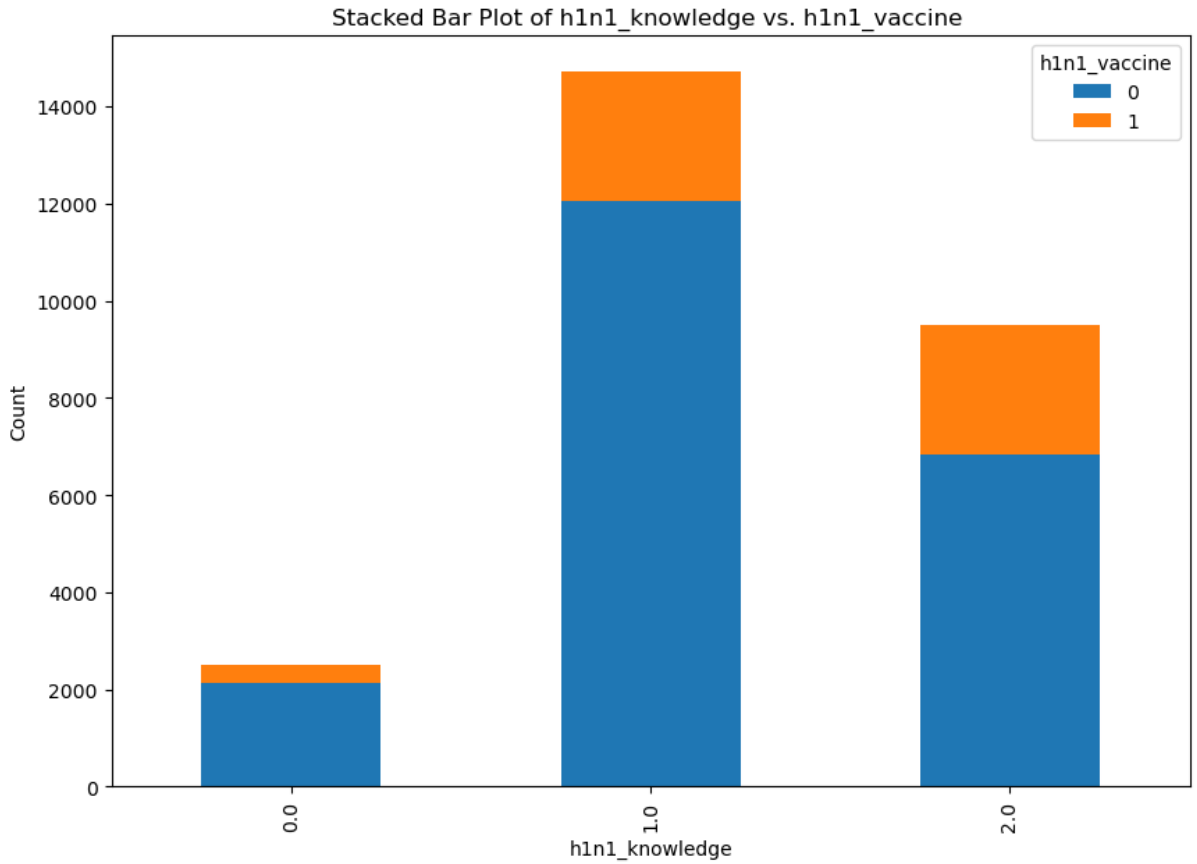
In summary, as the level of concern regarding h1n1 increases, the number of people getting vaccinated also seems to increase, although not everyone with high concern opted for vaccination.

```
ct = pd.crosstab(df['h1n1_concern'], y['h1n1_vaccine'])

# Plotting
ct.plot(kind='bar', stacked=True, figsize=(10,7))
plt.title('Stacked Bar Plot of h1n1_concern vs. h1n1_vaccine ')
plt.ylabel('Count')
plt.xlabel('h1n1_concern')
plt.show()
```



h1n1 knowledge and h1n1 vaccine

```
ct = pd.crosstab(df['h1n1_knowledge'], y['h1n1_vaccine'])

# Plotting
ct.plot(kind='bar', stacked=True, figsize=(10,7))
plt.title('Stacked Bar Plot of h1n1_knowledge vs. h1n1_vaccine ')
plt.ylabel('Count')
plt.xlabel('h1n1_knowledge')
plt.show()
```

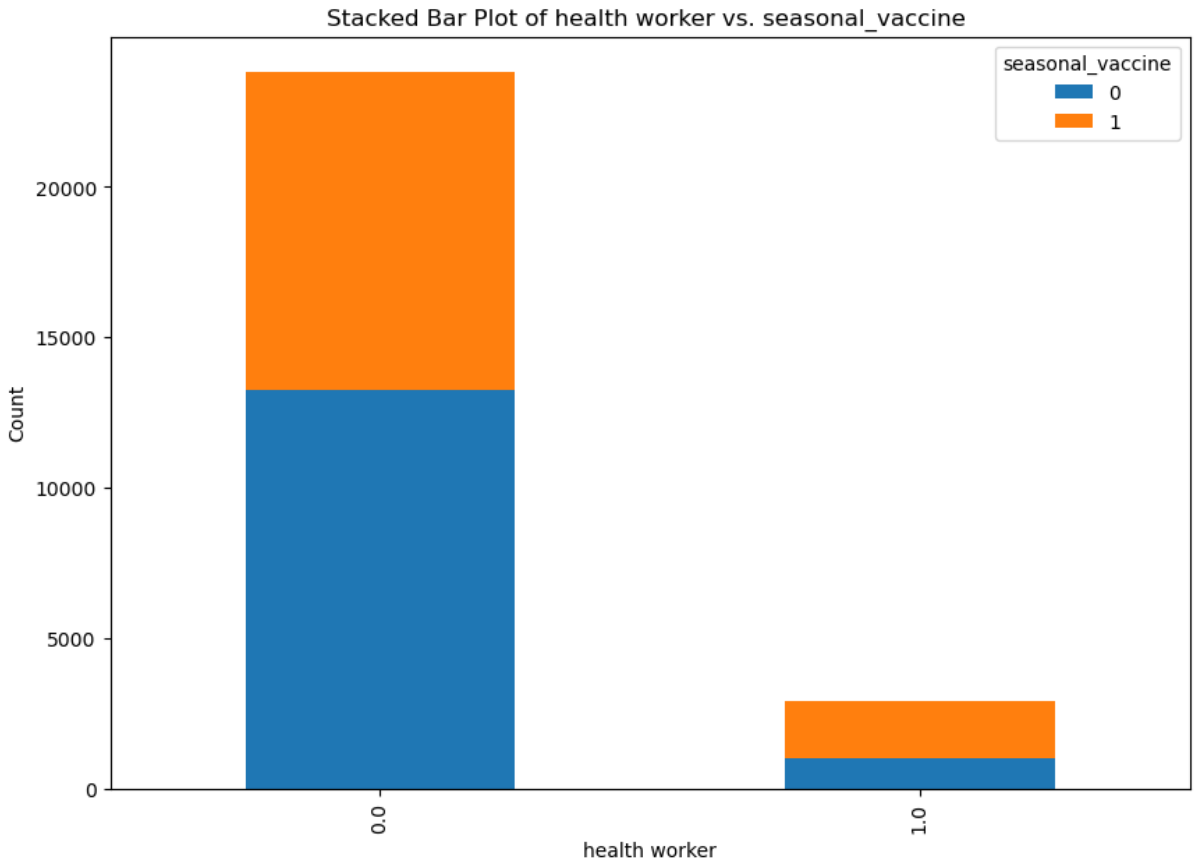


Observations:

The plot suggests a potential relationship between knowledge about H1N1 and the likelihood of getting vaccinated. As knowledge increases, a larger proportion of people seem to opt for the vaccine.

Health worker and seasonal vaccine

```
ct = pd.crosstab(df['health_worker'], y['seasonal_vaccine'])

# Plotting
ct.plot(kind='bar', stacked=True, figsize=(10,7))
plt.title('Stacked Bar Plot of health worker vs. seasonal_vaccine ')
plt.ylabel('Count')
plt.xlabel('health worker')
plt.show()
```

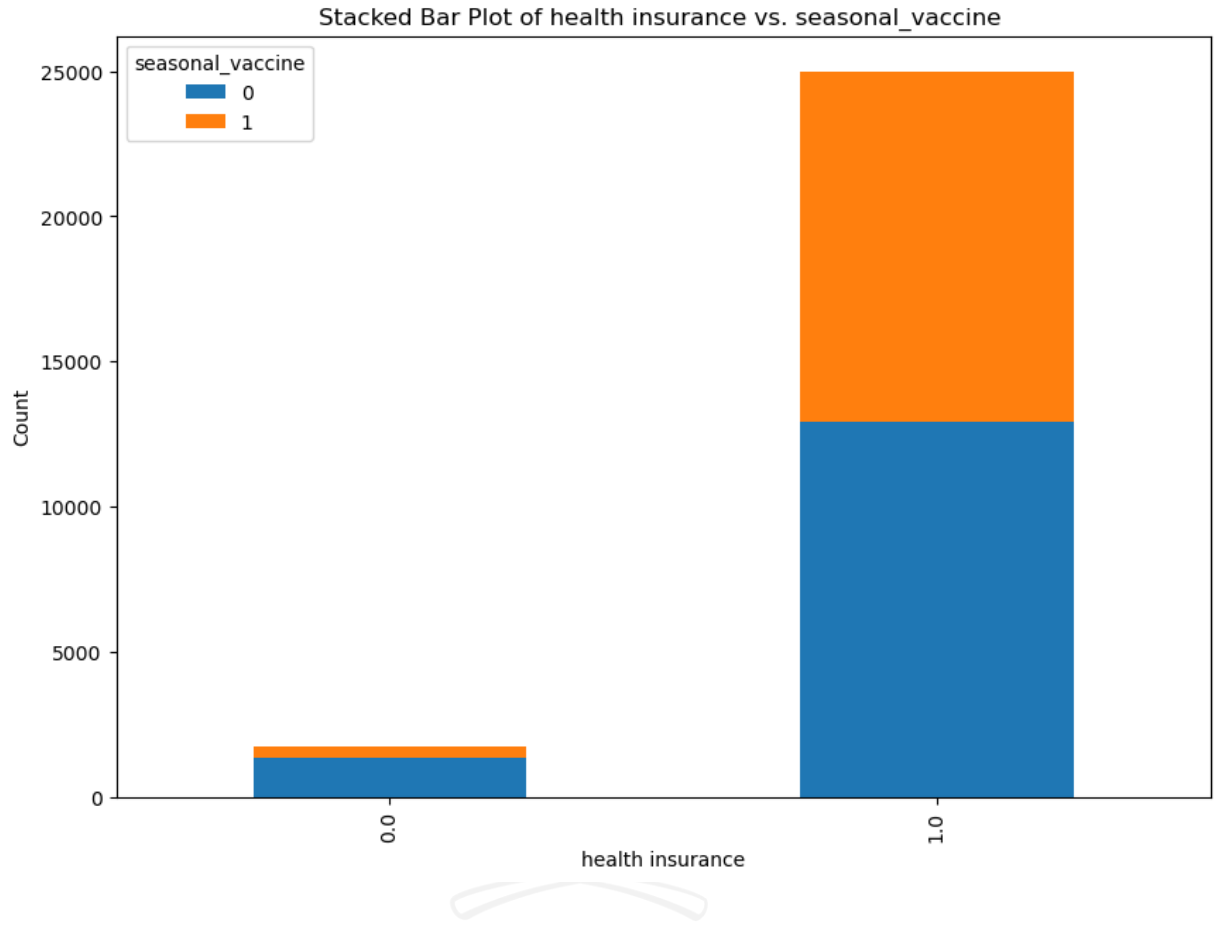


Observations:

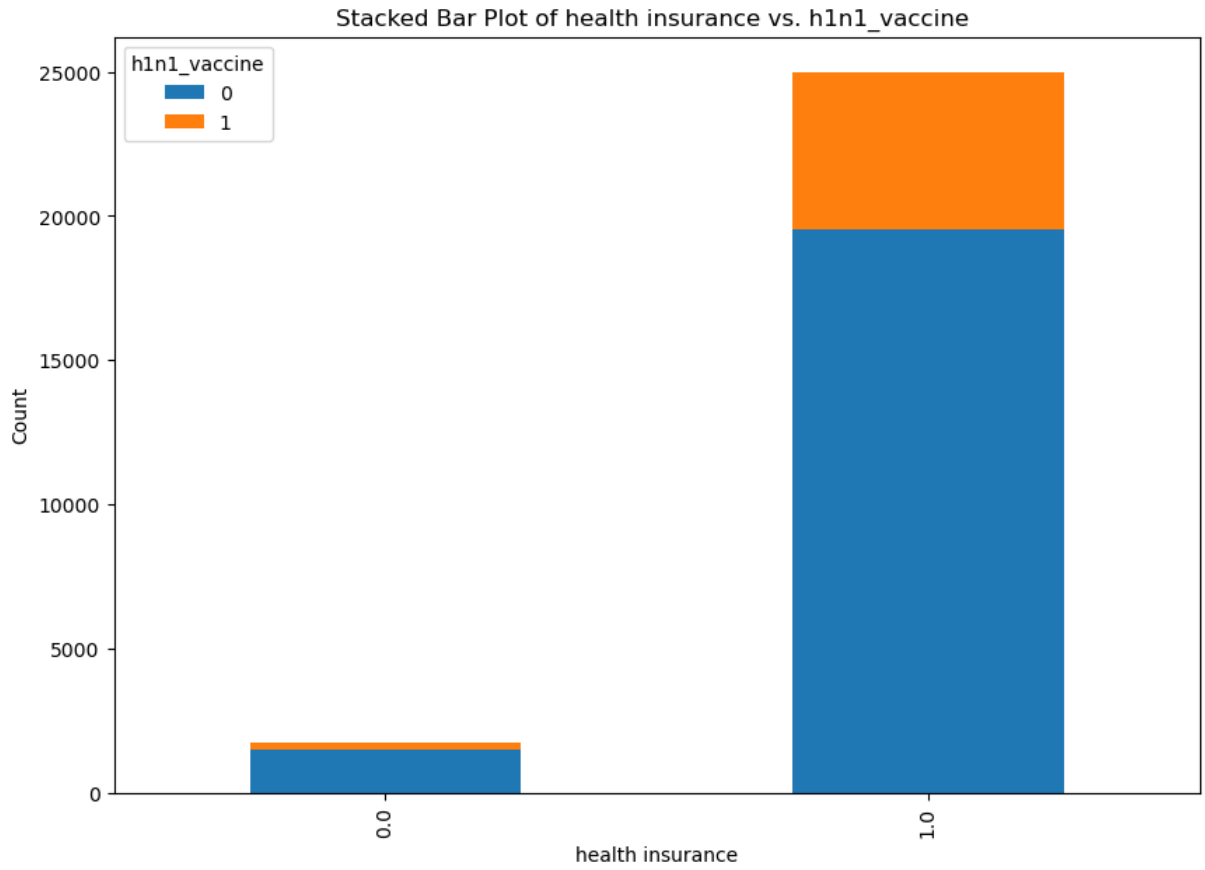
From the above plot we can see that health workers are vaccinated more as compared to non-health workers.

health insurance vs seasonal and h1n1 vaccine

```
ct = pd.crosstab(df['health_insurance'], y['seasonal_vaccine'])

# Plotting
ct.plot(kind='bar', stacked=True, figsize=(10,7))
plt.title('Stacked Bar Plot of health insurance vs. seasonal_vaccine ')
plt.ylabel('Count')
plt.xlabel('health insurance')
plt.show()
```



```
ct = pd.crosstab(df['health_insurance'], y['h1n1_vaccine'])

# Plotting
ct.plot(kind='bar', stacked=True, figsize=(10,7))
plt.title('Stacked Bar Plot of health insurance vs. h1n1_vaccine ')
plt.ylabel('Count')
plt.xlabel('health insurance')
plt.show()
```



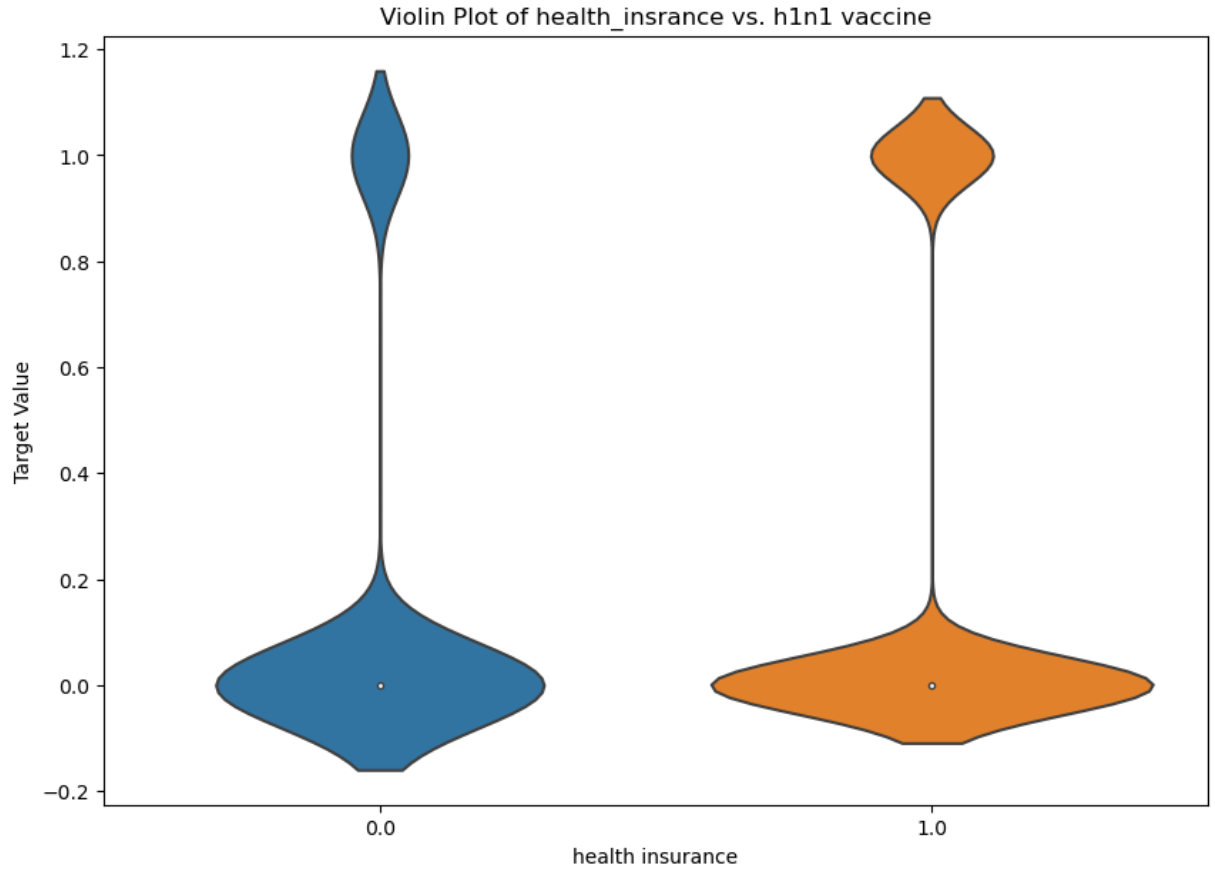
Observations:

For seasonal vaccine: people having health insurance are more vaccinated than those who don't have insurance, but we can see almost 50-50 distribution for people with health insurance. For h1n1 vaccine Number of people with h1n1 vaccine is less as compared to seasonal vaccine for people having health insurance.

Stacked Bar Plots revealed relationships between features and the target variable. For instance, as the level of concern about H1N1 increased, the likelihood of vaccination also increased.

violin plot

```
plt.figure(figsize=(10,7))
sns.violinplot(x='health_insurance', y='h1n1_vaccine', data=df)
plt.title('Violin Plot of health_insrance vs. h1n1 vaccine')
plt.ylabel('Target Value')
plt.xlabel('health insurance')
plt.show()
```



Observation

1. The wide portion at the bottom (around 0) suggests that a significant portion of individuals without health insurance (represented by 0.0 on the x-axis) did not take the h1n1 vaccine.
2. Individuals with health insurance (1.0) are more likely to have taken the h1n1 vaccine than those without health insurance (0.0).
3. The vaccination rate is higher among those with health insurance compared to those without.

This plot provides a clear visual representation of the relationship between having health insurance and the likelihood of getting the h1n1 vaccine. It's an effective way to communicate that health insurance status appears to be associated with vaccination behavior, at least in this dataset.

8. Feature Engineering

Top 10 features

```
from sklearn.feature_selection import SelectKBest, chi2
selector = SelectKBest(chi2, k=10)  # Select top 10 features
reduced_data = selector.fit_transform(X, y)

```

```
reduced_data
```

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 1, 1, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [1, 0, 0, ..., 1, 0, 0]], dtype=uint8)
```

```
feature_names = list(X.columns)
```

```
selected_mask = selector.get_support()

# Use the mask to get the selected feature names
selected_features = [feature for (feature, selected) in zip(feature_names, selected_mask) if selected]
```

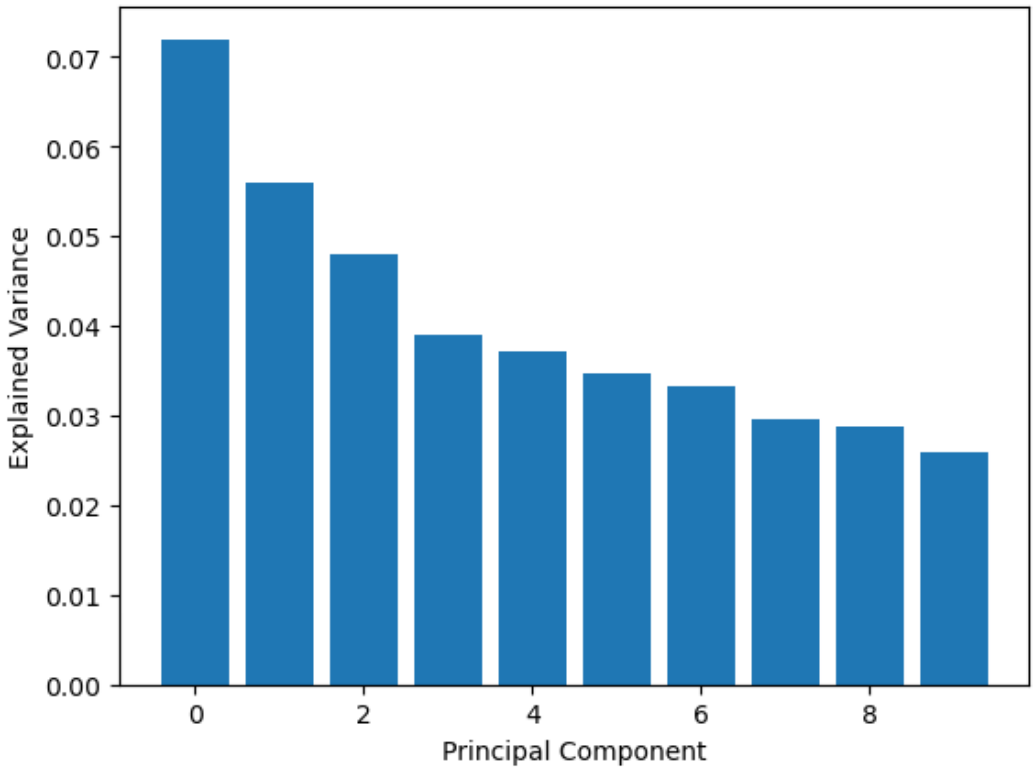
```
selected_features
```

```
['age_group_65+ Years',
 'doctor_recc_h1n1_1.0',
 'doctor_recc_seasonal_1.0',
 'health_worker_1.0',
 'opinion_h1n1_vacc_effective_5.0',
 'opinion_h1n1_risk_4.0',
 'opinion_h1n1_risk_5.0',
 'opinion_seas_vacc_effective_5.0',
 'opinion_seas_risk_4.0',
 'opinion_seas_risk_5.0']
```

- Top K Features: Using SelectKBest, the top 10 influential features were identified.
- Implication: These features can be prioritized in modeling for potentially better results.

9. Dimensionality Reduction using PCA

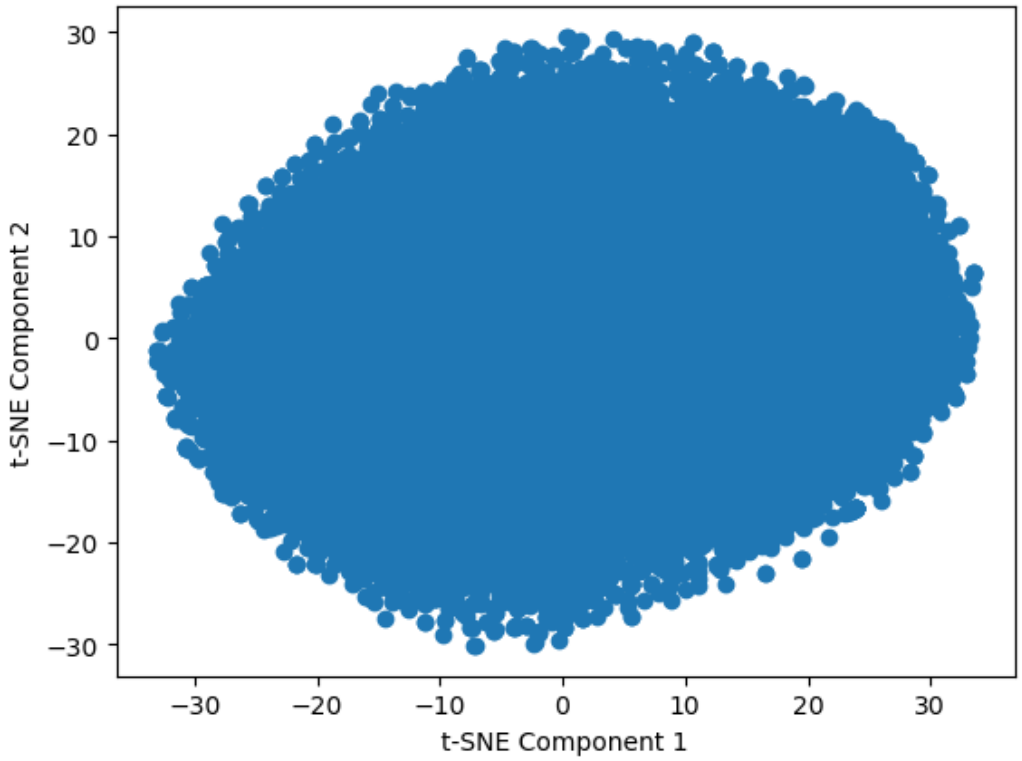
```python
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Applying PCA
pca = PCA(n_components=10) # for illustration, change n_components as needed
principalComponents = pca.fit_transform(X)

# Visualizing the variance explained by each component
explained_variance = pca.explained_variance_ratio_
plt.bar(range(len(explained_variance)), explained_variance)
plt.xlabel('Principal Component')
plt.ylabel('Explained Variance')
plt.show()
```



- PCA was used to reduce dimensionality.
- Observation: The PCA plot shows a decay in explained variance as we keep on increasing the number of principal components. The explained variance by each principal component decreased progressively, implying that a few components might capture most of the variance.

10. Visualization using t-SNE

```python
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt

# Applying t-SNE
tsne = TSNE(n_components=2, perplexity=30, n_iter=300)
tsne_results = tsne.fit_transform(X)

# Visualizing the results
plt.scatter(tsne_results[:, 0], tsne_results[:, 1])
plt.xlabel('t-SNE Component 1')
plt.ylabel('t-SNE Component 2')
plt.show()
```



t-SNE was used for visualization, showing the data points densely packed without clear separation. This could indicate that the data might not have strong inherent clusters.

Observation

From the tsne plot it can be inferred that since the data points are densely packed and there's no clear separation between them, it's indicative that the dataset might not have strong inherent clusters (at least in the chosen perplexity and iteration settings of t-SNE).

Conclusion:

The EDA provided valuable insights into the data, laying a solid foundation for the next steps in the data analysis pipeline. These include dropping columns such as employment_industry and employment_occupataion as half the values are missing. Dropping column health_insurance as we can see from the mosaic plot it is correlated to health_worker. Additionally in the modelling stage we can choose top-k features with random forest as well as use feature components we derived from PCA in the modelling pipeline.