Mental Health Hack Fest 2021

Team 13

By Sam Yao

In [180…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [160…
```
#Import overall data
df = pd.read_csv('./HackFest21-Data.csv')
df.shape
```

Out[160… (702, 187)

Assumptions and Givens:

- Data was fielded in March of 2021 in a period of about two weeks
- The dataset is a representative sample of college students under 30 years of age in the US

Study of Mental Health Related Questions

Prompt 1:

In [23]:
```
mental_health_df = df.copy()
mental_health_df=mental_health_df[['Q32','Q33','Q34','Q35','Q36_1','Q36_2','Q36_3',
                    'Q36_4','Q36_5','Q36_6','Q36_7','Q36_8','Q37',
                    'Q38','Q38a','Q38b','Q38c']]
```

In [24]:
```
mental_health_df.columns = ['anxiety',
                            'stop_worrying',
                            'anhedonia',
                            'depression',
                            'insurance_employer',
                            'insurance_company',
                            'insurance_medicare',
                            'insurance_medicaid',
                            'insurance_tricare',
                            'insurance_va',
                            'insurance_ihs',
                            'insurance_other',
                            'delay_medical_care',
                            'failed_medical_care',
                            'mental_health_prescription',
                            'got_counseling',
                            'no_counseling']
mental_health_df.sample(5)
```

Out[24]:

	anxiety	stop_worrying	anhedonia	depression	insurance_employer	insurance_company	insurance
200	2	2	2	3	1	2	
12	2	1	1	1	1	2	
346	1	1	1	1	1	1	
307	2	2	4	4	2	2	
268	4	4	4	4	2	2	

In [37]:
```
# For the following, Yes is coded as '1' and No is coded as '2'
# Did not Answer is coded as '-99'
print(mental_health_df['got_counseling'].value_counts(sort=False))
print(mental_health_df['no_counseling'].value_counts(sort=False))
print("\n\n")
print("{:.2f}".format((153/702)*100), "% of respondents got counseling")
print("{:.2f}".format((514/702)*100), "% of respondents did not get counseling")
```

```
 1     153
 2     548
-99      1
Name: got_counseling, dtype: int64
1    188
2    514
Name: no_counseling, dtype: int64



21.79 % of respondents got counseling
73.22 % of respondents did not get counseling
```

Percentage of people who reported mental health problems

In [33]:
```
print(mental_health_df['anxiety'].value_counts(sort=False))
print(mental_health_df['stop_worrying'].value_counts(sort=False))
print(mental_health_df['anhedonia'].value_counts(sort=False))
print(mental_health_df['depression'].value_counts(sort=False))
print("\n\n")
print("{:.2f}".format(((702-127)/702)*100), "% of respondents reported several days of
print("{:.2f}".format(((702-195)/702)*100), "% of respondents reported several days of
print("{:.2f}".format(((702-173)/702)*100), "% of respondents reported several days of
print("{:.2f}".format(((702-188)/702)*100), "% of respondents reported several days of
```

```
1    127
2    219
3    114
4    242
Name: anxiety, dtype: int64
1    195
2    215
3    110
4    182
Name: stop_worrying, dtype: int64
1    173
2    233
3    123
4    173
Name: anhedonia, dtype: int64
1    188
```

```
2    235
3    108
4    171
Name: depression, dtype: int64
```

```
81.91 % of respondents reported several days of anxiety
72.22 % of respondents reported several days of non-stop worrying
75.36 % of respondents reported several days of general disintrest
73.22 % of respondents reported several days of depression
```

In [75]:
```
# Query to ask how many students had inconsistent access to therapy
mental_health_df.query('got_counseling==1 and no_counseling==1').shape[0]
```

Out[75]: 33

Despite more than 70% of respondents experiencing at least several days or more of one of the following:

- Anxiety
- The inability to stop worrying
- Having little interest in doing things
- Feeling depressed or hopeless

only 21.79% of respondents received counseling or therapy in the last four weeks.

Additionally, 33 respondents reported they both received counseling and needed it but failed to get it, suggesting that therapy was denied at least once.

Logistic Regression Model on Mental Health Resources

There was a respondent who didn't fill out the question if they had counseling or not, so their entry was deleted

In [168…
```
mental_health_df= mental_health_df.query('got_counseling > 0')
mental_health_df.shape
```

Out[168… (701, 17)

In [169…
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X=mental_health_df[['anxiety',
                    'stop_worrying',
                    'anhedonia',
                    'depression']]
y=mental_health_df['got_counseling']
```

In [171…
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3)
X.info()
```

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

```
Int64Index: 701 entries, 0 to 701
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   anxiety        701 non-null    int64
 1   stop_worrying  701 non-null    int64
 2   anhedonia      701 non-null    int64
 3   depression     701 non-null    int64
dtypes: int64(4)
memory usage: 27.4 KB
```

Cross validation through K-folds (6 Splits)

In [172…
```
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()

kf = KFold(n_splits=6, shuffle=True)
acc_train=[]
acc_test=[]
for fold, (train_i, test_i) in enumerate(kf.split(X)):
    X_train, X_test = X.iloc[train_i], X.iloc[test_i]
    y_train, y_test = y.iloc[train_i], y.iloc[test_i]
    lr.fit(X_train, y_train)
    y_hat_train = lr.predict(X_train)
    train_acc = accuracy_score(y_train, y_hat_train)
    acc_train.append(train_acc)

    y_hat_test = lr.predict(X_test)
    test_acc = accuracy_score(y_test, y_hat_test)
    acc_test.append(test_acc)

    print('Fold {}: Train Accuracy={:0.2f}, Test Accuracy={:0.2f}'.format(fold+1, train
avg_tr=0
avg_te=0
for tr_rec, te_rec in zip(acc_train, acc_test):
    avg_tr += np.mean(tr_rec)
    avg_te += np.mean(te_rec)
avg_tr =avg_tr/len(acc_train)
avg_te = avg_te/len(acc_test)
print('Average Accuracy Train:{:0.2f} Accuracy Test:{:0.2f}'.format(avg_tr,avg_te))
```

```
Fold 1: Train Accuracy=0.78, Test Accuracy=0.79
Fold 2: Train Accuracy=0.79, Test Accuracy=0.73
Fold 3: Train Accuracy=0.77, Test Accuracy=0.85
Fold 4: Train Accuracy=0.78, Test Accuracy=0.79
Fold 5: Train Accuracy=0.78, Test Accuracy=0.79
Fold 6: Train Accuracy=0.79, Test Accuracy=0.74
Average Accuracy Train:0.78 Accuracy Test:0.78
```

In [173…
```
print(lr.coef_)
print(lr.intercept_)
```

```
[[-0.19648408 -0.29056985  0.00909317  0.00705883]]
[2.58245882]
```

Intercept: 2.58245882

Attribute	Beta-Score
anxiety/Q32	-0.19648408
stop_worrying/Q33	-0.29056985

Attribute	Beta-Score
anhedonia/Q34	0.00909317
depression/Q35	0.00705883

The Logistic Regression model trained on the dataset showed, on average, a 78% accuracy while training and testing, which isn't great, but it isn't terrible either. How would you use this?

Say you had a respondent who experienced several days of anxiety (2), +1/2 days unable to stop worrying (3), Not anhedonic (1), and Spent nearly every day Depressed (4). One would insert this as a list into the model to try to predict if they had received counseling.

In [195…
```
example1 = [[2,3,1,4]]
# Array Definition: [Class'1'(received counseling), Class '2' (Did not receive therapy)
lr.predict_proba(example1)
```

Out[195…
```
array([[0.20503627, 0.79496373]])
```

Clearly, student 'example1' is not having a good time, yet they still did not receive counseling. So what attributes would it take to predict someone as having received therapy?

In [198…
```
optimal_score=[0,0,0,0]
optimal_percent=[-1,-1]
for i in range(1,5):
    for j in range(1,5):
        for k in range(1,5):
            for l in range(1,5):
                temp=[[i,j,k,l]]
                temp_prob = lr.predict_proba(temp)
                if(temp_prob[0][0] > optimal_percent[0]):
                    optimal_percent[0] = temp_prob[0][0]
                    optimal_percent[1] = temp_prob[0][1]
                    optimal_score = temp
print(optimal_score)
print(optimal_percent)
```

```
[[4, 4, 1, 1]]
[0.34290050334873823, 0.6570994966512618]
```

The respondent that is most likely to go to therapy is someone who experiences anxiety nearly everyday, cannot stop themselves from worrying nearly every day, while not experiencing any anhedonia or depression, and even then, by the calculations of the prediction model, they are only 34% likely to respond that they have received therapy in the last four weeks. This is reflective of the overall low rates at which US college students under the age of 30 receive counseling or therapy according to the study.

-Percentage of students seeking counseling is low -students high in anxiety and worry but not necessarily in depression or anhedonia -Which students are most likely to seek counseling?