In [174]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
plt.rc('font', size = 12)


health = pd.read_csv('Healthcare_data_updated.csv', header = 0)  #this is the updated file with the more u


#One column has a comma in the middle of its name so I altered that col name (it was causing problems when
health=health.rename(columns = {'Comorb_Encntr_For_General_Exam_W_O_Complaint,_Susp_Or_Reprtd_Dx':'Comorb_



#correct the entry with the odd version of OB GYN name on specialities field
#print(health.index[health['Ntm_Speciality'] == 'OBSTETRICS & OBSTETRICS & GYNECOLOGY & OBSTETRICS & GYNEC
health.set_value(2847,'Ntm_Speciality','OBSTETRICS & GYNECOLOGY')
#print(health['Ntm_Speciality'][2847])



#create some new columns that flag properties that by the bar charts seem associated with drug persistency

#create a column that flags when the value of Dexa_Freq_During_Rx is <=2  - these are disproportionately n
Dexa_Freq_During_Rx_Bucket_Flag = list()    #initializing this list with the list() function
for i in range(0,3424):
    result = 0
    if health['Dexa_Freq_During_Rx'][i] <= 2:
        result = 1
    Dexa_Freq_During_Rx_Bucket_Flag.append(result)

#insert the new column created above
health.insert(15, 'Dexa_Freq_During_Rx_Bucket_Flag', Dexa_Freq_During_Rx_Bucket_Flag, True)

###
#create two new columns that flags Ntm Specialities
#first group
Ntm_Speciality_MyBuckets1 = list()  #initializing this list with the list() function
```

```
for i in range(0,3424):
    result = 0
    if health['Ntm_Speciality'][i] in ['ONCOLOGY', 'PEDIATRICS', 'PATHOLOGY', 'ENDOCRINOLOGY','VASCULAR SU
        result = 1
    Ntm_Speciality_MyBuckets1.append(result)

#new column added to health df
health.insert(11, 'Ntm_Speciality_MyBuckets1', Ntm_Speciality_MyBuckets1, True)
###


#second group
Ntm_Speciality_MyBuckets2 = list()   #initializing this list with the list() function
for i in range(0,3424):
    result = 0
    if health['Ntm_Speciality'][i] in ['PAIN MEDICINE', 'ORTHOPEDIC SURGERY', 'PULMONARY MEDICINE', 'CARDI
        result = 1
    Ntm_Speciality_MyBuckets2.append(result)

#new column added to health df
health.insert(11, 'Ntm_Speciality_MyBuckets2', Ntm_Speciality_MyBuckets2, True)
###

#midwest region is associated with persistence
Midwest_Flag = list()    #initializing this list with the list() function
for i in range(0,3424):
    result = 0
    if health['Region'][i] == 'Midwest':
        result = 1
    if health['Region'][i] != 'Midwest':
        result = 0
    Midwest_Flag.append(result)
 #new column added to health df
health.insert(7,'Midwest_Flag', Midwest_Flag, True)
####


#asian race is associated with persistence
AsianRace_Flag = list()    #initializing this list with the list() function
for i in range(0,3424):
    result = 0
```

```
    if health['Race'][i] == 'Asian':
        result = 1
    if health['Race'][i] != 'Asian':
        result = 0
    AsianRace_Flag.append(result)
 #new column added to health df
health.insert(5,'AsianRace_Flag', AsianRace_Flag, True)


###

#create 3 flags for Change in T Score
#the reference No Change is the default value (no dummy variable created for that one)
ChangedTScore_Worsened = list()  #initializing this list with the list() function
ChangedTScore_Improved = list()   #initializing this list with the list() function
ChangedTScore_Unk = list()

for i in range(0,3424):
    resultw = 0
    resulti = 0
    resultu = 0
    if health['Change_T_Score'][i] == 'Worsened':
        resultw = 1
    if health['Change_T_Score'][i] == 'Improved':
        resulti = 1
    if health['Change_T_Score'][i] == 'Unknown':
        resultu = 1
    ChangedTScore_Worsened.append(resultw)
    ChangedTScore_Improved.append(resulti)
    ChangedTScore_Unk.append(resultu)

#these three new variables are inserted into the df health
health.insert(15,'ChangedTScore_Worsened', ChangedTScore_Worsened, True)
health.insert(15,'ChangedTScore_Improved', ChangedTScore_Improved, True)
health.insert(15,'ChangedTScore_Unk', ChangedTScore_Unk, True)
#print('orig T score')
#print(health['Change_T_Score'][0:10])
#print(ChangedTScore_Worsened[0:10])
#print(ChangedTScore_Improved[0:10])
#print(ChangedTScore_Unk[0:10])
```

```
#####


#Change in Risk segment, three dummy flag variables created, the default is No Change and it has no variab
Change_RiskSeg_Worsened = list()
Change_RiskSeg_Improved = list()
Change_RiskSeg_Unk = list()


for i in range(0,3424):
    resultw = 0
    resulti = 0
    resultu = 0
    if health['Change_Risk_Segment'][i] == 'Worsened':
        resultw = 1
    if health['Change_Risk_Segment'][i] == 'Improved':
        resulti = 1
    if health['Change_Risk_Segment'][i] == 'Unknown':
        resultu = 1
    Change_RiskSeg_Worsened.append(resultw)
    Change_RiskSeg_Improved.append(resulti)
    Change_RiskSeg_Unk.append(resultu)

 #these three new variables are inserted
health.insert(17,'Change_RiskSeg_Worsened', Change_RiskSeg_Worsened, True)
health.insert(17,'Change_RiskSeg_Improved', Change_RiskSeg_Improved, True)
health.insert(17,'Change_RiskSeg_Unk', Change_RiskSeg_Unk, True)

#print('health change risk seg, follwed by worsened, improved, unknown')
#print(health['Change_Risk_Segment'])
#print('Worsened', Change_RiskSeg_Worsened[0:10])
#print('improved risk seg', Change_RiskSeg_Improved[0:10])
#print('unknown risk seg', Change_RiskSeg_Unk[0:10])



#print(list(health.columns))   #to see updated column names
#print(health.shape)    #to see updated df size
```

```
/Users/jen/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:18: FutureWarning: set_value i
```

s deprecated and will be removed in a future release. Please use .at[] or .iat[] accessors instead

In [175]:
```
#####    Next, the Y/N variables get converted to 1/0


health['Persistency_Flag'] = health['Persistency_Flag'].replace('Non-Persistent', 0)
health['Persistency_Flag'] = health['Persistency_Flag'].replace('Persistent', 1)


health['Adherent_Flag'] = health['Adherent_Flag'].replace("Adherent",1)
health['Adherent_Flag'] = health['Adherent_Flag'].replace("Non-Adherent",0)


#since unknown risk segment during rx is associated with different persistency than the other options, set
health['Risk_Segment_During_Rx'] = health['Risk_Segment_During_Rx'].replace("Unknown",1)
health['Risk_Segment_During_Rx'] = health['Risk_Segment_During_Rx'].replace("HR_VHR",0)
health['Risk_Segment_During_Rx'] = health['Risk_Segment_During_Rx'].replace("VLR_LR",0)


#since unknown risk segment during rx is associated with different persistency than the other options, set
health['Tscore_Bucket_During_Rx'] = health['Tscore_Bucket_During_Rx'].replace("Unknown",1)
health['Tscore_Bucket_During_Rx'] = health['Tscore_Bucket_During_Rx'].replace("<=2.5",0)
health['Tscore_Bucket_During_Rx'] = health['Tscore_Bucket_During_Rx'].replace(">2.5",0)



#Now to convert a large batch at once:
list1 = ['Idn_Indicator',
            'Injectable_Experience_During_Rx', 'Comorb_Encounter_For_Screening_For_Malignant_Neoplasms', '
            'Comorb_Encntr_For_General_Exam_W_O_Complaint_Susp_Or_Reprtd_Dx', 'Comorb_Vitamin_D_Deficiency
            'Comorb_Encntr_For_Oth_Sp_Exam_W_O_Complaint_Suspected_Or_Reprtd_Dx', 'Comorb_Long_Term_Curren
            'Comorb_Dorsalgia', 'Comorb_Personal_History_Of_Other_Diseases_And_Conditions', 'Comorb_Other_
            'Comorb_Disorders_of_lipoprotein_metabolism_and_other_lipidemias', 'Comorb_Osteoporosis_withou
            'Comorb_Personal_history_of_malignant_neoplasm', 'Comorb_Gastro_esophageal_reflux_disease', 'C
            'Concom_Narcotics', 'Concom_Systemic_Corticosteroids_Plain', 'Concom_Anti_Depressants_And_Mood
            'Concom_Cephalosporins', 'Concom_Macrolides_And_Similar_Types', 'Concom_Broad_Spectrum_Penicil
            'Risk_Rheumatoid_Arthritis', 'Risk_Untreated_Chronic_Hyperthyroidism', 'Risk_Untreated_Chronic
            'Risk_Chronic_Liver_Disease',  'Risk_Low_Calcium_Intake', 'Risk_Vitamin_D_Insufficiency', 'Ris
            'Risk_Immobilization', 'Gluco_Record_During_Rx', 'Dexa_During_Rx', 'Frag_Frac_During_Rx']
health[list1] = health[list1].replace("Y",1)
health[list1] = health[list1].replace("N", 0)
```

```
#print(health[list1].head())
```

In [ ]:

In [176]:
```
#define the list of features (columns) to be used as predictors
#note this subset is constructed to create greater independence between columns and to eliminate unhelpful

features = ['Gluco_Record_During_Rx', 'Dexa_During_Rx', 'Frag_Frac_During_Rx', 'Risk_Segment_During_Rx', '
            'Injectable_Experience_During_Rx', 'Comorb_Encounter_For_Screening_For_Malignant_Neoplasms', '
            'Comorb_Encntr_For_General_Exam_W_O_Complaint_Susp_Or_Reprtd_Dx', 'Comorb_Vitamin_D_Deficiency
            'Comorb_Encntr_For_Oth_Sp_Exam_W_O_Complaint_Suspected_Or_Reprtd_Dx', 'Comorb_Long_Term_Curren
            'Comorb_Dorsalgia', 'Comorb_Personal_History_Of_Other_Diseases_And_Conditions', 'Comorb_Other_
            'Comorb_Disorders_of_lipoprotein_metabolism_and_other_lipidemias', 'Comorb_Osteoporosis_withou
            'Comorb_Personal_history_of_malignant_neoplasm', 'Comorb_Gastro_esophageal_reflux_disease', 'C
            'Concom_Narcotics', 'Concom_Systemic_Corticosteroids_Plain', 'Concom_Anti_Depressants_And_Mood
            'Concom_Cephalosporins', 'Concom_Macrolides_And_Similar_Types', 'Concom_Broad_Spectrum_Penicil
            'Risk_Rheumatoid_Arthritis', 'Risk_Untreated_Chronic_Hyperthyroidism', 'Risk_Untreated_Chronic
            'Risk_Chronic_Liver_Disease',  'Risk_Low_Calcium_Intake', 'Risk_Vitamin_D_Insufficiency', 'Ris
            'Risk_Immobilization', 'Dexa_Freq_During_Rx_Bucket_Flag','Change_RiskSeg_Worsened','Change_Ris
            'ChangedTScore_Improved', 'ChangedTScore_Unk', 'AsianRace_Flag',  'Midwest_Flag', 'Ntm_Special

health[features] = health[features].replace('N', 0)
health[features] = health[features].replace('Y', 1)


print(health['Persistency_Flag'].head(5))
print(health[features].head())
```

```
0    1
1    0
2    0
3    0
4    0
Name: Persistency_Flag, dtype: int64
   Gluco_Record_During_Rx  Dexa_During_Rx  Frag_Frac_During_Rx  \
0                       0               0                    0
1                       0               0                    0
2                       0               0                    0
3                       1               0                    0
4                       1               0                    0

   Risk_Segment_During_Rx  Adherent_Flag  Idn_Indicator  \
0                       0              1              0
1                       1              1              0
2                       0              1              0
3                       0              1              0
4                       1              1              0

   Injectable_Experience_During_Rx  \
0                                1
1                                1
2                                1
3                                1
4                                1

   Comorb_Encounter_For_Screening_For_Malignant_Neoplasms  \
0                                                  0
1                                                  0
2                                                  1
3                                                  0
4                                                  1

   Comorb_Encounter_For_Immunization  \
0                                  1
1                                  0
2                                  0
3                                  1
4                                  1
```

```
   Comorb_Encntr_For_General_Exam_W_O_Complaint_Susp_Or_Reprtd_Dx  ...  \
0                                                  1               ...
1                                                  1               ...
2                                                  1               ...
3                                                  1               ...
4                                                  1               ...

   Change_RiskSeg_Worsened  Change_RiskSeg_Improved  Change_RiskSeg_Unk  \
0                        0                        0                   1
1                        0                        0                   1
2                        0                        0                   0
3                        0                        0                   0
4                        0                        0                   1

   ChangedTScore_Worsened  ChangedTScore_Improved  ChangedTScore_Unk  \
0                       0                       0                  0
1                       0                       0                  1
2                       0                       0                  0
3                       0                       0                  0
4                       0                       0                  1

   AsianRace_Flag  Midwest_Flag  Ntm_Speciality_MyBuckets1  \
0               0             0                          0
1               1             0                          0
2               0             1                          0
3               0             1                          0
4               0             1                          0

   Ntm_Speciality_MyBuckets2
0                          0
1                          0
```

In [177]:
```
# import the package as needed
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

# instantiate the model
logreg = LogisticRegression()
```

In [178]:
```
X = health[features]
```

```
y = health.Persistency_Flag


# split X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=0)
```

In [179]:
```
# fit the model with data - cannot run it yet, until variables are transformed
logreg.fit(X_train,y_train)

#
y_pred=logreg.predict(X_test)
```

```
/Users/jen/opt/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:432: FutureWarning:
Default solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

In [180]:
```
confusion_matrix = confusion_matrix(y_test, y_pred)
print(confusion_matrix)
```

```
[[487  56]
 [ 98 215]]
```

In [183]:
```
print(round(100*(98+56)/(487+56+98+215),1), '%')

print("This is the percentage incorrectly classified within the test set")
print("Further refinements will be applied in an effort to improve this error rate")
```

```
18.0 %
This is the percentage incorrectly classified within the test set
Further refinements will be applied in an effort to improve this error rate
```

In [184]:
```
#statsmodels provides more information by way of summary()
#we can use this to refine the columns further (eliminating some)
#we are looking to keep those columns (predictor variables) with low values in the 'P>[t]'' column of the

import statsmodels.api as sm
X_train = sm.add_constant(X_train)
lm_2 = sm.OLS(y_train, X_train).fit()
lm_2.summary()
```

```
/Users/jen/opt/anaconda3/lib/python3.7/site-packages/numpy/core/fromnumeric.py:2495: FutureWarning: Metho
d .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Out[184]:

OLS Regression Results

Dep. Variable:	Persistency_Flag	**R-squared:**	0.457
Model:	OLS	**Adj. R-squared:**	0.445
Method:	Least Squares	**F-statistic:**	39.85
Date:	Mon, 06 Dec 2021	**Prob (F-statistic):**	4.47e-289
Time:	01:53:59	**Log-Likelihood:**	-1004.7
No. Observations:	2568	**AIC:**	2117.
Df Residuals:	2514	**BIC:**	2433.
Df Model:	53		
Covariance Type:	nonrobust		

In [173]:

```
#create and export a new csv with all predictors converted to non-categorical, new csv will be health_NoCa

health_NonCat = health[features]
health_NonCat.insert(0,'Persistency_Flag', health['Persistency_Flag'], True )
health_NonCat.to_csv('health_NoCats')
```