

Imports

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from scipy import stats
import collections
from sklearn.preprocessing import StandardScaler, RobustScaler, MinMaxScaler
from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score, acc
from sklearn.model_selection import train_test_split
from sklearn.utils import resample
import warnings

#importing packages for modeling
from sklearn.linear_model import LogisticRegression, RidgeClassifier
from sklearn.svm import SVC, LinearSVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import make_pipeline

%matplotlib inline
warnings.filterwarnings('ignore')
```

```
In [2]: target_names=['Non-Persistent', 'Persistent']
```

Dataset

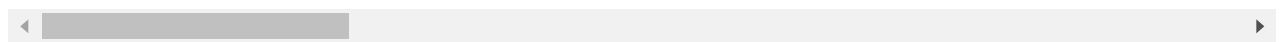
```
In [3]: xls = pd.ExcelFile('Healthcare_dataset.xlsx')
df = pd.read_excel(xls, 'Dataset')
```

```
In [4]: df.head()
```

```
Out[4]:
```

	Ptid	Persistency_Flag	Gender	Race	Ethnicity	Region	Age_Bucket	Ntm_Speciality	Ntm
0	P1	Persistent	Male	Caucasian	Not Hispanic	West	>75	GENERAL PRACTITIONER	
1	P2	Non-Persistent	Male	Asian	Not Hispanic	West	55-65	GENERAL PRACTITIONER	
2	P3	Non-Persistent	Female	Other/Unknown	Hispanic	Midwest	65-75	GENERAL PRACTITIONER	
3	P4	Non-Persistent	Female	Caucasian	Not Hispanic	Midwest	>75	GENERAL PRACTITIONER	
4	P5	Non-Persistent	Female	Caucasian	Not Hispanic	Midwest	>75	GENERAL PRACTITIONER	

5 rows × 69 columns



Data Understanding

In [5]: `df.shape`

Out[5]: (3424, 69)

In [6]: `df.columns`

Out[6]: Index(['Ptid', 'Persistency_Flag', 'Gender', 'Race', 'Ethnicity', 'Region', 'Age_Bucket', 'Ntm_Speciality', 'Ntm_Specialist_Flag', 'Ntm_Speciality_Bucket', 'Gluko_Record_Prior_Ntm', 'Gluko_Record_During_Rx', 'Dexa_Freq_During_Rx', 'Dexa_During_Rx', 'Frag_Frac_Prior_Ntm', 'Frag_Frac_During_Rx', 'Risk_Segment_Prior_Ntm', 'Tscore_Bucket_Prior_Ntm', 'Risk_Segment_During_Rx', 'Tscore_Bucket_During_Rx', 'Change_T_Score', 'Change_Risk_Segment', 'Adherent_Flag', 'Idn_Indicator', 'Injectable_Experience_During_Rx', 'Comorb_Encounter_For_Screening_For_Malignant_Neoplasms', 'Comorb_Encounter_For_Immunization', 'Comorb_Encntr_For_General_Exam_W_O_Complaint,_Susp_Or_Reprtd_Dx', 'Comorb_Vitamin_D_Deficiency', 'Comorb_Other_Joint_Disorder_Not_Elsewhere_Classified', 'Comorb_Encntr_For_Oth_Sp_Exam_W_O_Complaint_Suspected_Or_Reprtd_Dx', 'Comorb_Long_Term_Current_Drug_Therapy', 'Comorb_Dorsalgia', 'Comorb_Personal_History_Of_Other_Diseases_And_Conditions', 'Comorb_Other_Disorders_Of_Bone_Density_And_Structure', 'Comorb_Disorders_of_lipoprotein_metabolism_and_other_lipidemias', 'Comorb_Osteoporosis_without_current_pathological_fracture', 'Comorb_Personal_history_of_malignant_neoplasm', 'Comorb_Gastro_esophageal_reflux_disease', 'Concom_Cholesterol_And_Triglyceride_Regulating_Preparations', 'Concom_Narcotics', 'Concom_Systemic_Corticosteroids_Plain', 'Concom_Anti_Depressants_And_Mood_Stabilisers', 'Concom_Fluoroquinolones', 'Concom_Cephalosporins', 'Concom_Macrolides_And_Similar_Types', 'Concom_Broad_Spectrum_Penicillins', 'Concom_Anaesthetics_General', 'Concom_Viral_Vaccines', 'Risk_Type_1_Insulin_Dependent_Diabetes', 'Risk_Osteogenesis_Imperfecta', 'Risk_Rheumatoid_Arthritis', 'Risk_Untreated_Chronic_Hyperthyroidism', 'Risk_Untreated_Chronic_Hypogonadism', 'Risk_Untreated_Early_Menopause', 'Risk_Patient_Parent_Fractured_Their_Hip', 'Risk_Smoking_Tobacco', 'Risk_Chronic_Malnutrition_Or_Malabsorption', 'Risk_Chronic_Liver_Disease', 'Risk_Family_History_Of_Osteoporosis', 'Risk_Low_Calcium_Intake', 'Risk_Vitamin_D_Insufficiency', 'Risk_Poor_Health_Frailty', 'Risk_Excessive_Thinness', 'Risk_Hysterectomy_Oophorectomy', 'Risk_Estrogen_Deficiency', 'Risk_Immobilization', 'Risk_Recurring_Falls', 'Count_Of_Risks'], dtype='object')

In [7]: `df.values`

Out[7]: array([['P1', 'Persistent', 'Male', ..., 'N', 'N', 0],

```

['P2', 'Non-Persistent', 'Male', ..., 'N', 'N', 0],
['P3', 'Non-Persistent', 'Female', ..., 'N', 'N', 2],
...,
['P3422', 'Persistent', 'Female', ..., 'N', 'N', 1],
['P3423', 'Non-Persistent', 'Female', ..., 'N', 'N', 0],
['P3424', 'Non-Persistent', 'Female', ..., 'N', 'N', 1]],
dtype=object)

```

In [8]: `df.dtypes`

```

Out[8]: Ptid                object
Persistency_Flag         object
Gender                   object
Race                     object
Ethnicity                 object
...
Risk_Hysterectomy_Oophorectomy  object
Risk_Estrogen_Deficiency         object
Risk_Immobilization              object
Risk_Recurring_Falls             object
Count_Of_Risks                   int64
Length: 69, dtype: object

```

In [9]: `df.info()`

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3424 entries, 0 to 3423
Data columns (total 69 columns):
 #   Column                                Non-Null Count
Dtype
---  -
0    Ptid                                3424 non-null
object
1    Persistency_Flag                  3424 non-null
object
2    Gender                           3424 non-null
object
3    Race                             3424 non-null
object
4    Ethnicity                        3424 non-null
object
5    Region                           3424 non-null
object
6    Age_Bucket                       3424 non-null
object
7    Ntm_Speciality                   3424 non-null
object
8    Ntm_Specialist_Flag              3424 non-null
object
9    Ntm_Speciality_Bucket            3424 non-null
object
10   Gluco_Record_Prior_Ntm           3424 non-null
object
11   Gluco_Record_During_Rx           3424 non-null
object
12   Dexa_Freq_During_Rx              3424 non-null
int64
13   Dexa_During_Rx                   3424 non-null

```

object		
14	Frag_Frac_Prior_Ntm	3424 non-null
object		
15	Frag_Frac_During_Rx	3424 non-null
object		
16	Risk_Segment_Prior_Ntm	3424 non-null
object		
17	Tscore_Bucket_Prior_Ntm	3424 non-null
object		
18	Risk_Segment_During_Rx	3424 non-null
object		
19	Tscore_Bucket_During_Rx	3424 non-null
object		
20	Change_T_Score	3424 non-null
object		
21	Change_Risk_Segment	3424 non-null
object		
22	Adherent_Flag	3424 non-null
object		
23	Idn_Indicator	3424 non-null
object		
24	Injectable_Experience_During_Rx	3424 non-null
object		
25	Comorb_Encounter_For_Screening_For_Malignant_Neoplasms	3424 non-null
object		
26	Comorb_Encounter_For_Immunization	3424 non-null
object		
27	Comorb_Encntr_For_General_Exam_W_O_Complaint,_Susp_Or_Reprtd_Dx	3424 non-null
object		
28	Comorb_Vitamin_D_Deficiency	3424 non-null
object		
29	Comorb_Other_Joint_Disorder_Not_Elsewhere_Classified	3424 non-null
object		
30	Comorb_Encntr_For_Oth_Sp_Exam_W_O_Complaint_Suspected_Or_Reprtd_Dx	3424 non-null
object		
31	Comorb_Long_Term_Current_Drug_Therapy	3424 non-null
object		
32	Comorb_Dorsalgia	3424 non-null
object		
33	Comorb_Personal_History_Of_Other_Diseases_And_Conditions	3424 non-null
object		
34	Comorb_Other_Disorders_Of_Bone_Density_And_Structure	3424 non-null
object		
35	Comorb_Disorders_of_lipoprotein_metabolism_and_other_lipidemias	3424 non-null
object		
36	Comorb_Osteoporosis_without_current_pathological_fracture	3424 non-null
object		
37	Comorb_Personal_history_of_malignant_neoplasm	3424 non-null
object		
38	Comorb_Gastro_esophageal_reflux_disease	3424 non-null
object		
39	Concom_Cholesterol_And_Triglyceride_Regulating_Preparations	3424 non-null
object		
40	Concom_Narcotics	3424 non-null
object		
41	Concom_Systemic_Corticosteroids_Plain	3424 non-null
object		
42	Concom_Anti_Depressants_And_Mood_Stabilisers	3424 non-null
object		
43	Concom_Fluoroquinolones	3424 non-null

```

object
  44  Concom_Cephalosporins                                3424 non-null
object
  45  Concom_Macrolides_And_Similar_Types                  3424 non-null
object
  46  Concom_Broad_Spectrum_Penicillins                    3424 non-null
object
  47  Concom_Anaesthetics_General                           3424 non-null
object
  48  Concom_Viral_Vaccines                                3424 non-null
object
  49  Risk_Type_1_Insulin_Dependent_Diabetes                3424 non-null
object
  50  Risk_Osteogenesis_Imperfecta                         3424 non-null
object
  51  Risk_Rheumatoid_Arthritis                             3424 non-null
object
  52  Risk_Untreated_Chronic_Hyperthyroidism               3424 non-null
object
  53  Risk_Untreated_Chronic_Hypogonadism                  3424 non-null
object
  54  Risk_Untreated_Early_Menopause                       3424 non-null
object
  55  Risk_Patient_Parent_Fractured_Their_Hip              3424 non-null
object
  56  Risk_Smoking_Tobacco                                  3424 non-null
object
  57  Risk_Chronic_Malnutrition_Or_Malabsorption            3424 non-null
object
  58  Risk_Chronic_Liver_Disease                           3424 non-null
object
  59  Risk_Family_History_Of_Osteoporosis                  3424 non-null
object
  60  Risk_Low_Calcium_Intake                               3424 non-null
object
  61  Risk_Vitamin_D_Insufficiency                         3424 non-null
object
  62  Risk_Poor_Health_Frailty                             3424 non-null
object
  63  Risk_Excessive_Thinness                              3424 non-null
object
  64  Risk_Hysterectomy_Oophorectomy                       3424 non-null
object
  65  Risk_Estrogen_Deficiency                             3424 non-null
object
  66  Risk_Immobilization                                   3424 non-null
object
  67  Risk_Recurring_Falls                                  3424 non-null
object
  68  Count_Of_Risks                                        3424 non-null
int64
dtypes: int64(2), object(67)
memory usage: 1.8+ MB

```

Data Exploratory

```
In [10]: df.columns=[x.lower() for x in df.columns]
```

```
In [11]: cat_corr = df.apply(lambda x : pd.factorize(x)[0]).corr(method='pearson', min_periods=1
np.abs(cat_corr).sort_values(by=['persistency_flag'], ascending=False)
```

```
Out[11]:
```

	persistency_flag
persistency_flag	1.000000
dexa_during_rx	0.491823
dexa_freq_during_rx	0.395247
comorb_long_term_current_drug_therapy	0.352760
comorb_encounter_for_screening_for_malignant_neoplasms	0.322320
...	...
risk_untreated_early_menopause	0.005279
risk_family_history_of_osteoporosis	0.003492
risk_osteogenesis_imperfecta	0.002636
risk_segment_during_rx	0.000700
frag_frac_prior_ntm	0.000318

69 rows × 1 columns

```
In [12]: df.isnull().sum()
```

```
Out[12]: ptid          0
persistency_flag    0
gender              0
race               0
ethnicity          0
..
risk_hysterectomy_oophorectomy  0
risk_estrogen_deficiency        0
risk_immobilization             0
risk_recurring_falls            0
count_of_risks                  0
Length: 69, dtype: int64
```

```
In [13]: def uniquevals(col):
print(f'Details of the particular col {col} is : {df[col].unique()}')

def valuecounts(col):
print(f'Valuecounts of the particular col {col} is : {df[col].value_counts()}')

for col in df.columns:
uniquevals(col)
print("-"*75)
```

```
Details of the particular col ptid is : ['P1' 'P2' 'P3' ... 'P3422' 'P3423' 'P3424']
-----
```

```

Details of the particular col persistency_flag is : ['Persistent' 'Non-Persistent']
-----
Details of the particular col gender is : ['Male' 'Female']
-----
Details of the particular col race is : ['Caucasian' 'Asian' 'Other/Unknown' 'African Am
erican']
-----
Details of the particular col ethnicity is : ['Not Hispanic' 'Hispanic' 'Unknown']
-----
Details of the particular col region is : ['West' 'Midwest' 'South' 'Other/Unknown' 'Nor
theast']
-----
Details of the particular col age_bucket is : ['>75' '55-65' '65-75' '<55']
-----
Details of the particular col ntm_speciality is : ['GENERAL PRACTITIONER' 'Unknown' 'END
OCRINOLOGY' 'RHEUMATOLOGY'
'ONCOLOGY' 'PATHOLOGY' 'OBSTETRICS AND GYNECOLOGY'
'PSYCHIATRY AND NEUROLOGY' 'ORTHOPEDIC SURGERY'
'PHYSICAL MEDICINE AND REHABILITATION' 'SURGERY AND SURGICAL SPECIALTIES'
'PEDIATRICS' 'PULMONARY MEDICINE' 'HEMATOLOGY & ONCOLOGY' 'UROLOGY'
'PAIN MEDICINE' 'NEUROLOGY' 'RADIOLOGY' 'GASTROENTEROLOGY'
'EMERGENCY MEDICINE' 'PODIATRY' 'OPHTHALMOLOGY' 'OCCUPATIONAL MEDICINE'
'TRANSPLANT SURGERY' 'PLASTIC SURGERY' 'CLINICAL NURSE SPECIALIST'
'OTOLARYNGOLOGY' 'HOSPITAL MEDICINE' 'ORTHOPEDICS' 'NEPHROLOGY'
'GERIATRIC MEDICINE' 'HOSPICE AND PALLIATIVE MEDICINE'
'OBSTETRICS & OBSTETRICS & GYNECOLOGY & OBSTETRICS & GYNECOLOGY'
'VASCULAR SURGERY' 'CARDIOLOGY' 'NUCLEAR MEDICINE']
-----
Details of the particular col ntm_specialist_flag is : ['Others' 'Specialist']
-----
Details of the particular col ntm_speciality_bucket is : ['OB/GYN/Others/PCP/Unknown' 'E
ndo/Onc/Uro' 'Rheum']
-----
Details of the particular col gluco_record_prior_ntm is : ['N' 'Y']
-----
Details of the particular col gluco_record_during_rx is : ['N' 'Y']
-----
Details of the particular col dexa_freq_during_rx is : [ 0 2 7 3 5 20 13 1
6 12 4 10 25 11 18 21 15 28
22 37 14 8 9 17 81 42 16 30 19 45 27 24 58 26 23 33
110 36 34 88 66 32 118 48 69 38 40 68 52 50 146 44 35 39
108 54 72 29]
-----
Details of the particular col dexa_during_rx is : ['N' 'Y']
-----
Details of the particular col frag_frac_prior_ntm is : ['N' 'Y']
-----
Details of the particular col frag_frac_during_rx is : ['N' 'Y']
-----
Details of the particular col risk_segment_prior_ntm is : ['VLR_LR' 'HR_VHR']
-----
Details of the particular col tscore_bucket_prior_ntm is : ['>-2.5' '<=-2.5']
-----
Details of the particular col risk_segment_during_rx is : ['VLR_LR' 'Unknown' 'HR_VHR']
-----
Details of the particular col tscore_bucket_during_rx is : ['<=-2.5' 'Unknown' '>-2.5']
-----
Details of the particular col change_t_score is : ['No change' 'Unknown' 'Worsened' 'Imp
roved']
-----

```

Details of the particular col change_risk_segment is : ['Unknown' 'No change' 'Worsened' 'Improved']

Details of the particular col adherent_flag is : ['Adherent' 'Non-Adherent']

Details of the particular col idn_indicator is : ['N' 'Y']

Details of the particular col injectable_experience_during_rx is : ['Y' 'N']

Details of the particular col comorb_encounter_for_screening_for_malignant_neoplasms is : ['N' 'Y']

Details of the particular col comorb_encounter_for_immunization is : ['Y' 'N']

Details of the particular col comorb_encntr_for_general_exam_w_o_complaint_susp_or_reprtd_dx is : ['Y' 'N']

Details of the particular col comorb_vitamin_d_deficiency is : ['N' 'Y']

Details of the particular col comorb_other_joint_disorder_not_elsewhere_classified is : ['N' 'Y']

Details of the particular col comorb_encntr_for_oth_sp_exam_w_o_complaint_suspected_or_reprtd_dx is : ['Y' 'N']

Details of the particular col comorb_long_term_current_drug_therapy is : ['N' 'Y']

Details of the particular col comorb_dorsalgia is : ['Y' 'N']

Details of the particular col comorb_personal_history_of_other_diseases_and_conditions is : ['Y' 'N']

Details of the particular col comorb_other_disorders_of_bone_density_and_structure is : ['N' 'Y']

Details of the particular col comorb_disorders_of_lipoprotein_metabolism_and_other_lipidemias is : ['N' 'Y']

Details of the particular col comorb_osteoporosis_without_current_pathological_fracture is : ['N' 'Y']

Details of the particular col comorb_personal_history_of_malignant_neoplasm is : ['N' 'Y']

Details of the particular col comorb_gastro_esophageal_reflux_disease is : ['N' 'Y']

Details of the particular col concom_cholesterol_and_triglyceride_regulating_preparations is : ['N' 'Y']

Details of the particular col concom_narcotics is : ['N' 'Y']

Details of the particular col concom_systemic_corticosteroids_plain is : ['N' 'Y']

Details of the particular col concom_anti_depressants_and_mood_stabilisers is : ['N' 'Y']

Details of the particular col concom_fluoroquinolones is : ['N' 'Y']

Details of the particular col concom_cephalosporins is : ['N' 'Y']


```

Details of the particular col concom_macrolides_and_similar_types is : ['N' 'Y']
-----
Details of the particular col concom_broad_spectrum_penicillins is : ['N' 'Y']
-----
Details of the particular col concom_anaesthetics_general is : ['N' 'Y']
-----
Details of the particular col concom_viral_vaccines is : ['N' 'Y']
-----
Details of the particular col risk_type_1_insulin_dependent_diabetes is : ['N' 'Y']
-----
Details of the particular col risk_osteogenesis_imperfecta is : ['N' 'Y']
-----
Details of the particular col risk_rheumatoid_arthritis is : ['N' 'Y']
-----
Details of the particular col risk_untreated_chronic_hyperthyroidism is : ['N' 'Y']
-----
Details of the particular col risk_untreated_chronic_hypogonadism is : ['N' 'Y']
-----
Details of the particular col risk_untreated_early_menopause is : ['N' 'Y']
-----
Details of the particular col risk_patient_parent_fractured_their_hip is : ['N' 'Y']
-----
Details of the particular col risk_smoking_tobacco is : ['N' 'Y']
-----
Details of the particular col risk_chronic_malnutrition_or_malabsorption is : ['N' 'Y']
-----
Details of the particular col risk_chronic_liver_disease is : ['N' 'Y']
-----
Details of the particular col risk_family_history_of_osteoporosis is : ['N' 'Y']
-----
Details of the particular col risk_low_calcium_intake is : ['N' 'Y']
-----
Details of the particular col risk_vitamin_d_insufficiency is : ['N' 'Y']
-----
Details of the particular col risk_poor_health_frailty is : ['N' 'Y']
-----
Details of the particular col risk_excessive_thinness is : ['N' 'Y']
-----
Details of the particular col risk_hysterectomy_oophorectomy is : ['N' 'Y']
-----
Details of the particular col risk_estrogen_deficiency is : ['N' 'Y']
-----
Details of the particular col risk_immobilization is : ['N' 'Y']
-----
Details of the particular col risk_recurring_falls is : ['N' 'Y']
-----
Details of the particular col count_of_risks is : [0 2 1 3 4 5 6 7]
-----

```

```

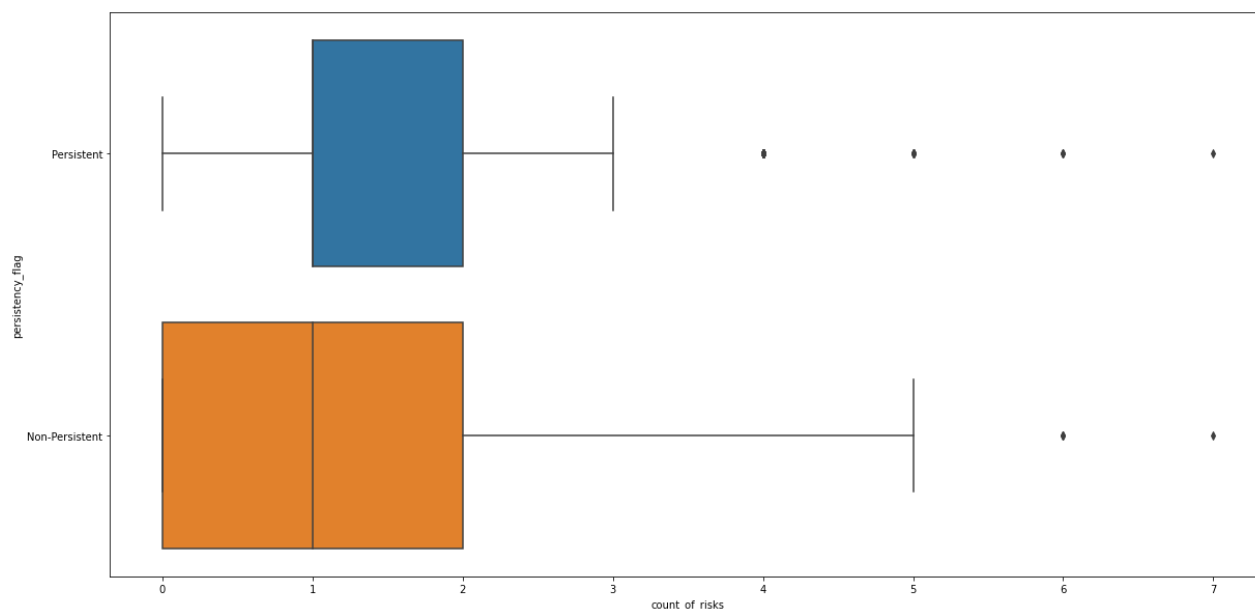
In [14]: plt.figure(figsize=(20,10))
          var ="count_of_risks"
          sns.boxplot(x=var,y ="persistency_flag",data=df)

```

```

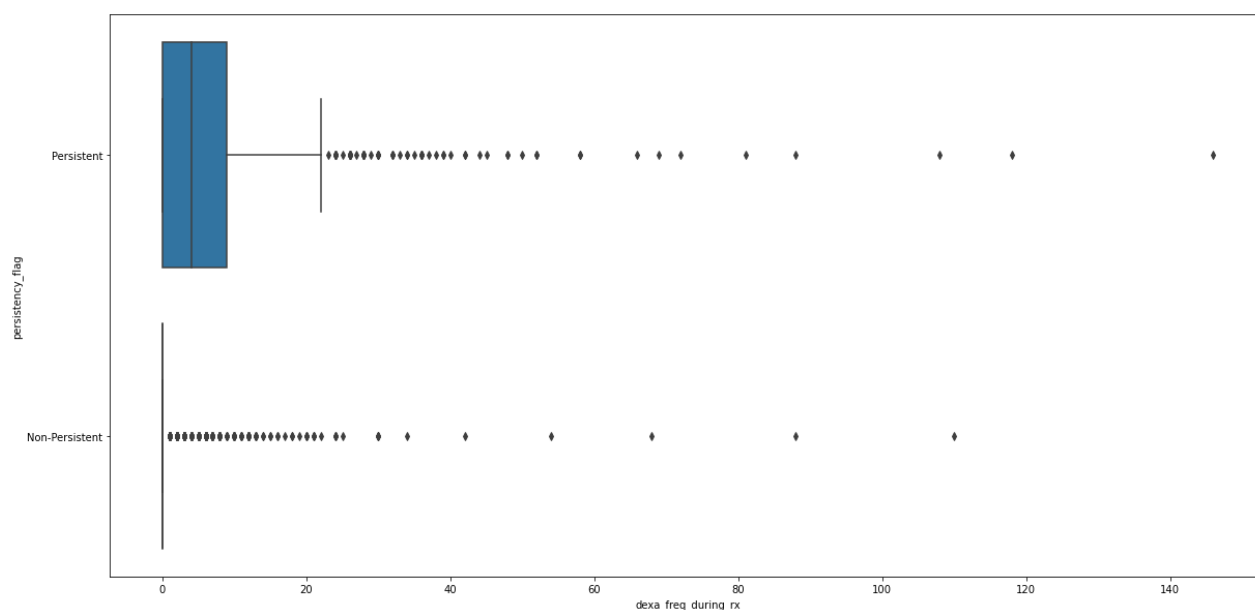
Out[14]: <AxesSubplot:xlabel='count_of_risks', ylabel='persistency_flag'>

```



```
In [15]: plt.figure(figsize=(20,10))
var = "dexa_freq_during_rx"
sns.boxplot(x=var, y = "persistence_flag", data=df)
```

```
Out[15]: <AxesSubplot:xlabel='dexa_freq_during_rx', ylabel='persistence_flag'>
```



```
In [16]: print("Count of risks skweness: ",df["count_of_risks"].skew())
```

Count of risks skweness: 0.8797905232898707

```
In [17]: print("dexa_freq_during_rx skweness: ",df["dexa_freq_during_rx"].skew())
```

dexa_freq_during_rx skweness: 6.8087302112992285

```
In [18]: #standardizing dexa_freq_during_rx df
dexa_scaled = StandardScaler().fit_transform(df['dexa_freq_during_rx'][:,np.newaxis]);
low_range = dexa_scaled[dexa_scaled[:,0].argsort()][:10]
high_range= dexa_scaled[dexa_scaled[:,0].argsort()][-10:]
```

```
print('outer range (low) of the distribution:')
print(low_range)
print('\nouter range (high) of the distribution:')
print(high_range)
```

outer range (low) of the distribution:

```
[[-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]
 [-0.3707352]]
```

outer range (high) of the distribution:

```
[[ 7.98784109]
 [ 8.11076133]
 [ 8.47952205]
 [ 9.58580421]
 [10.44624589]
 [10.44624589]
 [12.90465068]
 [13.15049116]
 [14.13385307]
 [17.57561978]]
```

```
In [19]: scaler = RobustScaler()
df['dexa_freq_during_rx'] = scaler.fit_transform(df['dexa_freq_during_rx'].values.reshape(-1,1))
```

```
In [20]: scaler = RobustScaler()
df['count_of_risks'] = scaler.fit_transform(df['count_of_risks'].values.reshape(-1,1))
```

```
In [21]: ''' Detection '''
# IQR
Q1 = np.percentile(df['dexa_freq_during_rx'], 25,
                    interpolation = 'midpoint')

Q3 = np.percentile(df['dexa_freq_during_rx'], 75,
                    interpolation = 'midpoint')
IQR = Q3 - Q1

print("Old Shape: ", df.shape)

# Upper bound
upper = np.where(df['dexa_freq_during_rx'] >= (Q3+1.5*IQR))
# Lower bound
lower = np.where(df['dexa_freq_during_rx'] <= (Q1-1.5*IQR))

print("lower",lower[0])
print("Upper",upper[0])

''' Removing the Outliers '''
df.drop(upper[0], inplace = True)
```

```
df.drop(lower[0], inplace = True)

print("New Shape: ", df.shape)

df = df.reset_index(drop=True)
```

Old Shape: (3424, 69)

lower []

```
Upper [ 32  33  62  65  89 101 110 116 164 180 186 194 198 201
 217 241 246 256 264 282 292 303 327 340 349 358 368 369
 373 378 382 390 415 417 426 433 448 457 462 464 480 495
 496 497 505 514 517 541 545 549 563 575 588 589 592 599
 603 605 613 640 646 651 653 656 657 678 684 688 700 705
 710 711 726 728 729 730 759 760 764 765 785 786 804 814
 823 834 847 849 864 870 873 885 909 915 925 926 930 937
 946 978 982 991 994 1006 1008 1016 1042 1061 1073 1074 1076 1113
1118 1119 1128 1134 1141 1148 1151 1196 1240 1265 1267 1270 1272 1273
1280 1283 1286 1291 1315 1359 1360 1363 1365 1370 1372 1396 1398 1404
1448 1474 1513 1524 1533 1539 1546 1550 1554 1555 1564 1566 1570 1576
1599 1628 1641 1642 1647 1654 1662 1671 1691 1703 1724 1732 1734 1746
1752 1773 1782 1783 1788 1793 1803 1815 1826 1833 1834 1836 1838 1848
1852 1854 1870 1876 1895 1901 1904 1909 1910 1914 1915 1919 1920 1928
1936 1943 1948 1949 1952 1956 1959 1963 1964 1965 1968 1970 1971 1975
1982 1983 1988 1993 1996 1997 2000 2002 2005 2006 2009 2010 2011 2013
2015 2016 2020 2024 2028 2029 2030 2031 2033 2034 2038 2041 2042 2043
2044 2046 2049 2054 2057 2058 2059 2060 2062 2065 2066 2069 2075 2081
2083 2086 2087 2132 2134 2139 2142 2151 2161 2163 2168 2169 2170 2171
2172 2175 2176 2178 2181 2186 2187 2189 2192 2196 2197 2205 2207 2213
2214 2215 2217 2220 2227 2231 2233 2235 2236 2237 2238 2239 2240 2247
2255 2262 2263 2264 2271 2272 2275 2278 2279 2280 2285 2289 2294 2298
2307 2308 2310 2314 2320 2321 2323 2324 2325 2327 2329 2330 2331 2332
2333 2335 2336 2337 2338 2350 2356 2359 2364 2376 2379 2381 2390 2393
2413 2416 2422 2425 2428 2429 2430 2432 2447 2459 2469 2478 2503 2528
2529 2537 2556 2557 2558 2560 2562 2567 2569 2570 2575 2578 2580 2582
2591 2595 2598 2603 2606 2608 2609 2611 2612 2617 2625 2631 2635 2640
2644 2654 2660 2675 2676 2681 2686 2696 2698 2710 2713 2715 2727 2733
2751 2753 2757 2789 2790 2791 2794 2799 2802 2804 2806 2809 2814 2819
2821 2822 2828 2833 2880 2882 2894 2895 2906 2910 2920 2927 2940 2944
2945 2952 2957 2959 2960 2984 2996 2998 3013 3021 3022 3023 3042 3044
3046 3048 3052 3058 3066 3068 3080 3100 3131 3138 3159 3172 3177 3236
3281 3309 3311 3325 3363 3378 3382 3384 3396 3400 3411 3414]
```

New Shape: (2964, 69)

In [22]:

```
''' Detection '''
# IQR
Q1 = np.percentile(df['count_of_risks'], 25,
                    interpolation = 'midpoint')

Q3 = np.percentile(df['count_of_risks'], 75,
                    interpolation = 'midpoint')
IQR = Q3 - Q1

print("Old Shape: ", df.shape)

# Upper bound
upper = np.where(df['count_of_risks'] >= (Q3+1.5*IQR))
# Lower bound
lower = np.where(df['count_of_risks'] <= (Q1-1.5*IQR))
```

```
print("lower",lower[0])
print("Upper",upper[0])

''' Removing the Outliers '''
df.drop(upper[0], inplace = True)
df.drop(lower[0], inplace = True)

print("New Shape: ", df.shape)

df = df.reset_index(drop=True)
```

Old Shape: (2964, 69)

lower []

Upper [281 318 327 507 655 665 678 705 733 952 1001 1126 1590 1624
1836 2227 2234 2450 2611 2702 2755 2888]

New Shape: (2942, 69)

In [23]: df.describe

Out[23]:

	ethnicity	region \	ptid	persistence_flag	gender	race
0	P1	Persistent	Male	Caucasian	Not Hispanic	West
1	P2	Non-Persistent	Male	Asian	Not Hispanic	West
2	P3	Non-Persistent	Female	Other/Unknown	Hispanic	Midwest
3	P4	Non-Persistent	Female	Caucasian	Not Hispanic	Midwest
4	P5	Non-Persistent	Female	Caucasian	Not Hispanic	Midwest
...
2937	P3420	Persistent	Female	Caucasian	Not Hispanic	South
2938	P3421	Persistent	Female	Caucasian	Not Hispanic	South
2939	P3422	Persistent	Female	Caucasian	Not Hispanic	South
2940	P3423	Non-Persistent	Female	Caucasian	Not Hispanic	South
2941	P3424	Non-Persistent	Female	Caucasian	Not Hispanic	South

	age_bucket	ntm_speciality	ntm_specialist_flag \
0	>75	GENERAL PRACTITIONER	Others
1	55-65	GENERAL PRACTITIONER	Others
2	65-75	GENERAL PRACTITIONER	Others
3	>75	GENERAL PRACTITIONER	Others
4	>75	GENERAL PRACTITIONER	Others
...
2937	>75	GENERAL PRACTITIONER	Others
2938	>75	Unknown	Others
2939	>75	ENDOCRINOLOGY	Specialist
2940	55-65	Unknown	Others
2941	65-75	Unknown	Others

	ntm_speciality_bucket	... risk_family_history_of_osteoporosis \
0	OB/GYN/Others/PCP/Unknown	...
1	OB/GYN/Others/PCP/Unknown	...
2	OB/GYN/Others/PCP/Unknown	...
3	OB/GYN/Others/PCP/Unknown	...
4	OB/GYN/Others/PCP/Unknown	...
...
2937	OB/GYN/Others/PCP/Unknown	...
2938	OB/GYN/Others/PCP/Unknown	...
2939	Endo/Onc/Uro	...
2940	OB/GYN/Others/PCP/Unknown	...
2941	OB/GYN/Others/PCP/Unknown	...

	risk_low_calcium_intake	risk_vitamin_d_insufficiency	\
0	N	N	
1	N	N	
2	Y	N	
3	N	N	
4	N	N	
...	
2937	N	Y	
2938	N	N	
2939	N	Y	
2940	N	N	
2941	N	Y	

	risk_poor_health_frailty	risk_excessive_thinness	\
0	N	N	
1	N	N	
2	N	N	
3	N	N	
4	N	N	
...	
2937	N	N	
2938	N	N	
2939	N	N	
2940	N	N	
2941	N	N	

	risk_hysterectomy_oophorectomy	risk_estrogen_deficiency	\
0	N	N	
1	N	N	
2	N	N	
3	N	N	
4	N	N	
...	
2937	N	N	
2938	N	N	
2939	N	N	
2940	N	N	
2941	N	N	

	risk_immobilization	risk_recurring_falls	count_of_risks
0	N	N	-0.5
1	N	N	-0.5
2	N	N	0.5
3	N	N	0.0
4	N	N	0.0
...
2937	N	N	0.0
2938	N	N	-0.5
2939	N	N	0.0
2940	N	N	-0.5
2941	N	N	0.0

[2942 rows x 69 columns]>

In [24]: `df.groupby(['persistency_flag']).mean().T`

Out[24]:

	persistency_flag	Non-Persistent	Persistent
dexa_freq_during_rx		0.085491	0.662570

persistence_flag	Non-Persistent	Persistent
count_of_risks	0.074744	0.155866

In [25]: `df.groupby(['gender']).mean().T`

Out[25]:

	gender	Female	Male
dexa_freq_during_rx		0.263874	0.215800
count_of_risks		0.099494	0.098266

In [26]: `df.groupby(['race']).mean()`

Out[26]:

	dexa_freq_during_rx	count_of_risks
race		
African American	0.246377	0.168478
Asian	0.135266	0.021739
Caucasian	0.266445	0.098297
Other/Unknown	0.204167	0.125000

In [27]: `df.groupby(['ethnicity']).mean().T`

Out[27]:

	ethnicity	Hispanic	Not Hispanic	Unknown
dexa_freq_during_rx		0.279835	0.260417	0.264069
count_of_risks		0.265432	0.097342	0.000000

In [28]: `df.groupby(['age_bucket']).mean().T`

Out[28]:

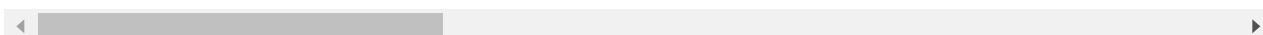
	age_bucket	55-65	65-75	<55	>75
dexa_freq_during_rx		0.242229	0.297880	0.273973	0.242208
count_of_risks		0.118167	0.097039	0.089041	0.093106

In [29]: `df.groupby(['ntm_speciality']).mean().T`

Out[29]:

	ntm_speciality	CARDIOLOGY	CLINICAL NURSE SPECIALIST	EMERGENCY MEDICINE	ENDOCRINOLOGY	GASTROENTEROLOGY
dexa_freq_during_rx		0.285714	0.0	0.0	0.392265	0.0
count_of_risks		0.380952	-0.5	0.0	0.279006	0.0

2 rows × 35 columns



```
In [30]: df.groupby(['ntm_specialist_flag']).mean().T
```

```
Out[30]:
```

	ntm_specialist_flag	Others	Specialist
dexa_freq_during_rx	0.215145	0.330765	
count_of_risks	0.056370	0.164812	

```
In [31]: df.groupby(['ntm_speciality_bucket']).mean().T
```

```
Out[31]:
```

	ntm_speciality_bucket	Endo/Onc/Uro	OB/GYN/Others/PCP/Unknown	Rheum
dexa_freq_during_rx		0.442907	0.215274	0.221349
count_of_risks		0.170415	0.053639	0.185658

```
In [32]: df.groupby(['risk_chronic_liver_disease']).mean().T
```

```
Out[32]:
```

	risk_chronic_liver_disease	N	Y
dexa_freq_during_rx		0.260132	0.452381
count_of_risks		0.096482	0.714286

```
In [33]: df.groupby(['risk_family_history_of_osteoporosis']).mean().T
```

```
Out[33]:
```

	risk_family_history_of_osteoporosis	N	Y
dexa_freq_during_rx		0.258113	0.287671
count_of_risks		0.045283	0.590753

```
In [34]: df.groupby(['risk_low_calcium_intake']).mean().T
```

```
Out[34]:
```

	risk_low_calcium_intake	N	Y
dexa_freq_during_rx		0.261069	0.259259
count_of_risks		0.090502	0.819444

```
In [35]: df.groupby(['risk_vitamin_d_insufficiency']).mean().T
```

```
Out[35]:
```

	risk_vitamin_d_insufficiency	N	Y
dexa_freq_during_rx		0.223363	0.303468
count_of_risks		-0.175866	0.409321

In [36]: `df.groupby(['risk_excessive_thinness']).mean().T`

Out[36]:

risk_excessive_thinness	N	Y
dexa_freq_during_rx	0.261946	0.218579
count_of_risks	0.085908	0.737705

In [37]: `df.groupby(['risk_hysterectomy_oophorectomy']).mean().T`

Out[37]:

risk_hysterectomy_oophorectomy	N	Y
dexa_freq_during_rx	0.261650	0.222222
count_of_risks	0.089748	0.722222

In [38]: `df.groupby(['risk_estrogen_deficiency']).mean().T`

Out[38]:

risk_estrogen_deficiency	N	Y
dexa_freq_during_rx	0.261052	0.259259
count_of_risks	0.097682	0.666667

In [39]: `df.groupby(['risk_immobilization']).mean().T`

Out[39]:

risk_immobilization	N	Y
dexa_freq_during_rx	0.262002	0.027778
count_of_risks	0.096416	0.833333

In [40]: `df.groupby(['risk_recurring_falls']).mean().T`

Out[40]:

risk_recurring_falls	N	Y
dexa_freq_during_rx	0.259901	0.321212
count_of_risks	0.087634	0.718182

In []:

In []:

Data Wrangling Transformation

```
In [41]: df = df.drop(['ptid'], axis=1)

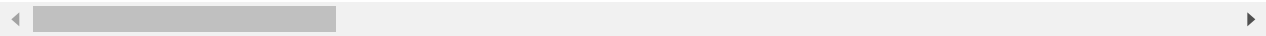
In [42]: mapper = {'N': 0, 'Y':1}
df = df.replace(mapper)

In [43]: df['persistency_flag'] = df['persistency_flag'].replace(['Non-Persistent', 'Persistent']
df.head()
```

Out[43]:

	persistency_flag	gender	race	ethnicity	region	age_bucket	ntm_speciality	ntm_speciali
0	1	Male	Caucasian	Not Hispanic	West	>75	GENERAL PRACTITIONER	
1	0	Male	Asian	Not Hispanic	West	55-65	GENERAL PRACTITIONER	
2	0	Female	Other/Unknown	Hispanic	Midwest	65-75	GENERAL PRACTITIONER	
3	0	Female	Caucasian	Not Hispanic	Midwest	>75	GENERAL PRACTITIONER	
4	0	Female	Caucasian	Not Hispanic	Midwest	>75	GENERAL PRACTITIONER	

5 rows × 68 columns



Data Exploratory (After Transformation)**

```
In [44]: np.abs(df.corr()).sort_values(by=['persistency_flag'], ascending=False)

Out[44]:
```

	persistency_flag	gluco_record_prio
persistency_flag	1.000000	0.0
dexa_freq_during_rx	0.414876	0.0
dexa_during_rx	0.374966	0.0
comorb_long_term_current_drug_therapy	0.342776	0.0
comorb_encounter_for_screening_for_malignant_neoplasms	0.268339	0.0
comorb_encounter_for_immunization	0.268305	0.0
comorb_encntr_for_general_exam_w_o_complaint_susp_or_reprtd_dx	0.257895	0.0
concom_systemic_corticosteroids_plain	0.249048	0.3
concom_viral_vaccines	0.227004	0.0
comorb_other_disorders_of_bone_density_and_structure	0.227003	0.0
concom_anaesthetics_general	0.220619	0.1
concom_cephalosporins	0.217821	0.1

	persistence_flag	gluco_record_prio
comorb_other_joint_disorder_not_elsewhere_classified	0.215937	0.0
gluco_record_during_rx	0.212778	0.3
comorb_gastro_esophageal_reflux_disease	0.207985	0.0
concom_macrolides_and_similar_types	0.192350	0.0
comorb_personal_history_of_other_diseases_and_conditions	0.189565	0.0
concom_narcotics	0.188391	0.1
concom_broad_spectrum_penicillins	0.186610	0.0
concom_fluoroquinolones	0.181213	0.1
comorb_dorsalgia	0.179925	0.0
comorb_encntr_for_oth_sp_exam_w_o_complaint_suspected_or_reprtd_dx	0.164096	0.0
comorb_personal_history_of_malignant_neoplasm	0.157273	0.0
comorb_vitamin_d_deficiency	0.151592	0.0
comorb_disorders_of_lipoprotein_metabolism_and_other_lipidemias	0.147411	0.0
comorb_osteoporosis_without_current_pathological_fracture	0.132641	0.0
idn_indicator	0.125887	0.0
concom_cholesterol_and_triglyceride_regulating_preparations	0.125322	0.0
risk_smoking_tobacco	0.115573	0.0
concom_anti_depressants_and_mood_stabilisers	0.111728	0.1
frag_frac_during_rx	0.102944	0.0
injectable_experience_during_rx	0.097495	0.0
count_of_risks	0.071565	0.1
risk_vitamin_d_insufficiency	0.069520	0.0
risk_rheumatoid_arthritis	0.059501	0.0
risk_poor_health_frailty	0.055891	0.0
risk_untreated_chronic_hypogonadism	0.045216	0.0
risk_immobilization	0.042316	0.0
risk_chronic_malnutrition_or_malabsorption	0.031632	0.0
risk_chronic_liver_disease	0.029426	0.0
risk_excessive_thinness	0.023628	0.0
risk_estrogen_deficiency	0.023250	0.0
risk_recurring_falls	0.020356	0.0
risk_untreated_chronic_hyperthyroidism	0.017246	0.0
risk_family_history_of_osteoporosis	0.016878	0.0

	persistency_flag	gluco_record_prio
risk_hysterectomy_oophorectomy	0.016192	0.0
risk_patient_parent_fractured_their_hip	0.015073	0.0
risk_low_calcium_intake	0.013116	0.0
risk_type_1_insulin_dependent_diabetes	0.007144	0.0
frag_frac_prior_ntm	0.005521	0.0
risk_untreated_early_menopause	0.004193	0.0
gluco_record_prior_ntm	0.003027	1.0
risk_osteogenesis_imperfecta	0.002022	0.0

53 rows × 53 columns

Creating Dummy values

```
In [45]: X=df.drop(['persistency_flag'],axis=1)
y=df['persistency_flag']

X = pd.get_dummies(X)
X.columns=[x.lower() for x in X.columns]
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=42,test_size=0.3, strat
```

```
In [46]: df_train = X_train.copy()
df_train['persistency_flag'] = y_train
df_train.head()
```

```
Out[46]:
```

	gluco_record_prior_ntm	gluco_record_during_rx	dexa_freq_during_rx	dexa_during_rx	frag_frac_prio
1493	1	1	0.0	0	
1375	0	0	0.0	0	
1217	1	1	0.0	0	
1157	1	1	0.0	0	
2766	0	0	0.0	0	

5 rows × 131 columns

```
In [47]: classes=df_train['persistency_flag'].value_counts()
normal_share=round(classes[0]/df_train['persistency_flag'].count()*100,2)
fraud_share=round(classes[1]/df_train['persistency_flag'].count()*100, 2)
print("Non-Persistent : {} %".format(normal_share))
print("Persistent : {} %".format(fraud_share))
```

Non-Persistent : 69.6 %

Persistent : 30.4 %

```
In [48]: fig = px.histogram(df_train, x="persistency_flag", color="persistency_flag", title="Per  
fig.show()
```

```
In [49]: # Upsampling  
df_minority_upsampled = resample(df_train[df_train['persistency_flag'] == 1],  
                                replace=True,      # sample with replacement  
                                n_samples=len(df_train[df_train['persistency_flag'] ==  
                                random_state=123) # reproducible results  
  
# Combine majority class with upsampled minority class  
df_train = pd.concat([df_train[df_train['persistency_flag'] == 0], df_minority_upsample  
  
# Display new class counts  
df_train.persistency_flag.value_counts()
```

```
Out[49]: 0    1433  
        1    1433  
        Name: persistency_flag, dtype: int64
```

```
In [50]: X_train=df_train.drop(['persistency_flag'],axis=1)  
y_train=df_train['persistency_flag']
```

```
In [51]: fig = px.histogram(df_train, x="persistency_flag", color="persistency_flag", title="Per  
fig.show()
```

Models Building

```
In [58]: def evaluation_metrics(y_test, y_pre, target_names):  
#scores  
print("Accuracy :", accuracy_score(y_test, y_pre))  
print("Precision :", precision_score(y_test, y_pre))  
print("Recall :", recall_score(y_test, y_pre))  
print("F1 Score :", f1_score(y_test, y_pre))  
  
print(classification_report(y_test, y_pre, target_names=target_names))  
  
#AUC  
fpr, tpr, _ = roc_curve(y_test, y_pre)  
auc = roc_auc_score(y_test, y_pre)  
print("AUC :", auc)  
  
#ROC  
plt.plot(fpr, tpr, label="uc={:.3f}".format(auc))  
plt.plot([0, 1], [0, 1], 'k--')  
plt.xlabel('False positive rate')
```

```

plt.ylabel('True positive rate')
plt.title('ROC curve')
plt.legend(loc=4)
plt.show()

#CM matrix
matrix = confusion_matrix(y_test, y_pre)
cm = pd.DataFrame(matrix, index=target_names, columns=target_names)

sns.heatmap(cm, annot=True, cbar=None, cmap="Blues", fmt = 'g')
plt.title("Confusion Matrix"), plt.tight_layout()
plt.ylabel("True Class"), plt.xlabel("Predicted Class")
plt.show()

```

Logistic Regression

In [59]:

```

def log(X_train,X_test,y_train,y_test):
    model=LogisticRegression()
    model.fit(X_train,y_train)
    y_pre=model.predict(X_test)
    evaluation_metrics(y_test, y_pre, target_names)

log(X_train,X_test,y_train,y_test)

```

Accuracy : 0.8301630434782609

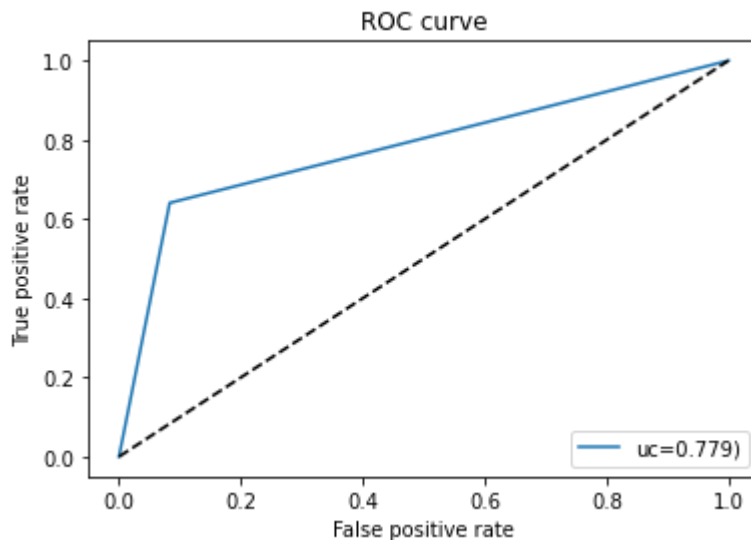
Precision : 0.7789473684210526

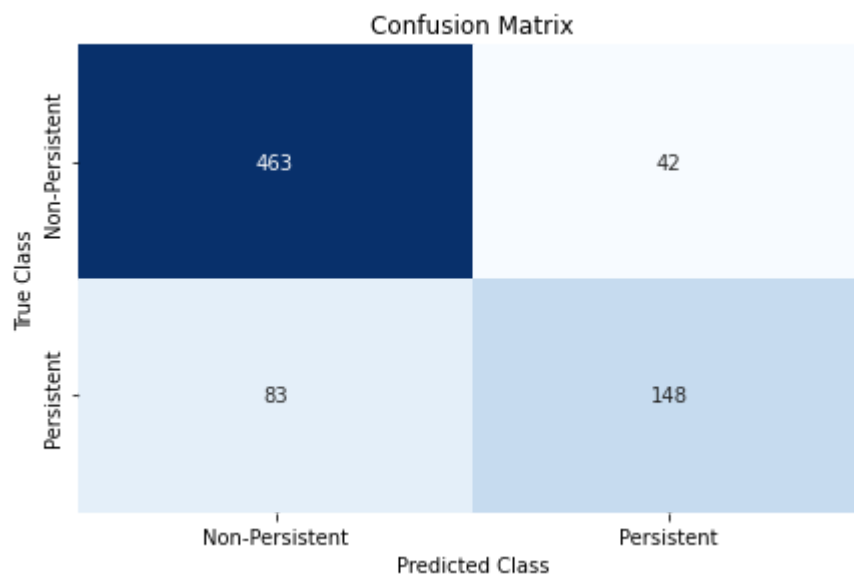
Recall : 0.6406926406926406

F1 Score : 0.7030878859857482

	precision	recall	f1-score	support
Non-Persistent	0.85	0.92	0.88	505
Persistent	0.78	0.64	0.70	231
accuracy			0.83	736
macro avg	0.81	0.78	0.79	736
weighted avg	0.83	0.83	0.83	736

AUC : 0.7787621619304788





RidgeClassifier

```
In [60]: def Ridge(X_train,X_test,y_train,y_test):
#train the model
model = RidgeClassifier(random_state=2)
model.fit(X_train, y_train)
#predictions
y_pre = model.predict(X_test)
evaluation_metrics(y_test, y_pre, target_names)
```

```
In [61]: Ridge(X_train,X_test,y_train,y_test)
```

Accuracy : 0.8206521739130435

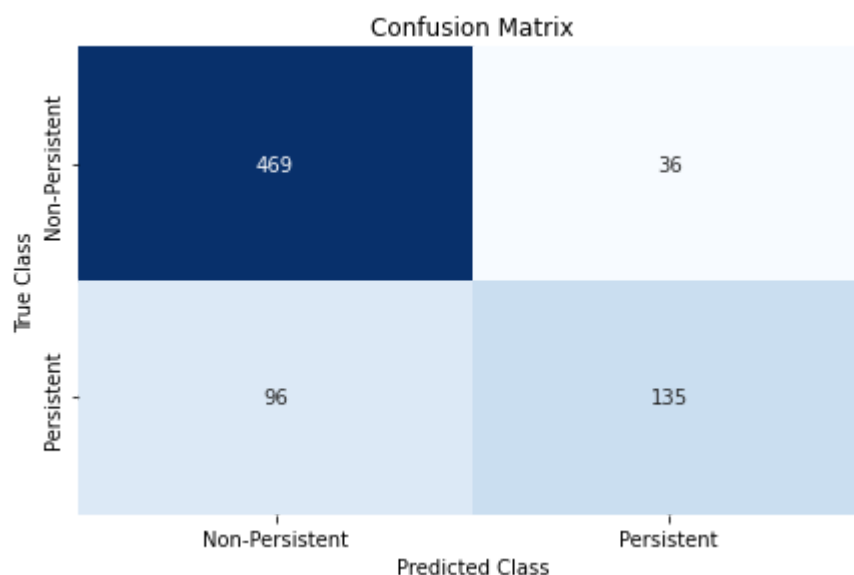
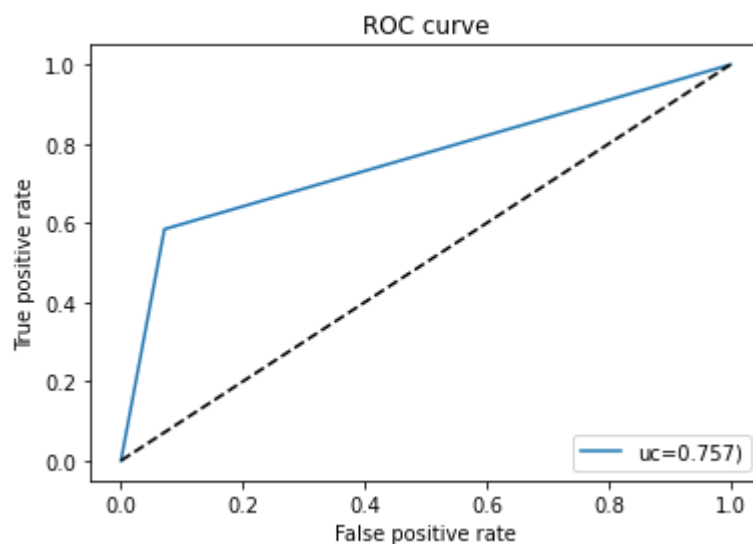
Precision : 0.7894736842105263

Recall : 0.5844155844155844

F1 Score : 0.6716417910447761

	precision	recall	f1-score	support
Non-Persistent	0.83	0.93	0.88	505
Persistent	0.79	0.58	0.67	231
accuracy			0.82	736
macro avg	0.81	0.76	0.77	736
weighted avg	0.82	0.82	0.81	736

AUC : 0.7565642278513565



RandomForestClassifier

```
In [63]: def RF(X_train,X_test,y_train,y_test):
#train the model
model = RandomForestClassifier(random_state=2)
model.fit(X_train, y_train)
#predictions
y_pre = model.predict(X_test)
evaluation_metrics(y_test, y_pre, target_names)
```

```
In [64]: RF(X_train,X_test,y_train,y_test)
```

Accuracy : 0.8247282608695652

Precision : 0.8227848101265823

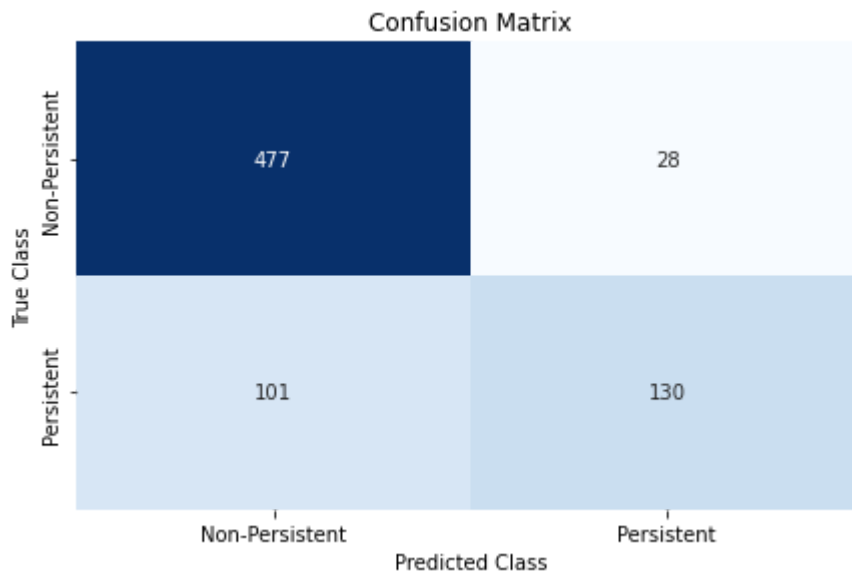
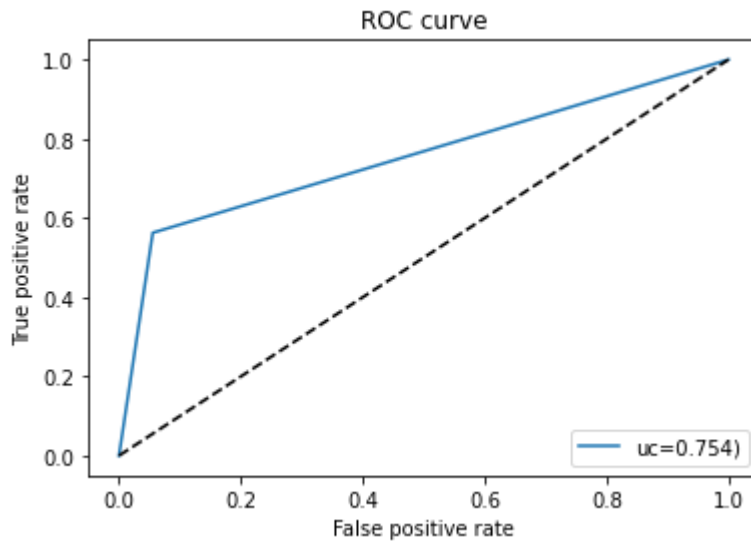
Recall : 0.5627705627705628

F1 Score : 0.6683804627249357

	precision	recall	f1-score	support
Non-Persistent	0.83	0.94	0.88	505
Persistent	0.82	0.56	0.67	231

accuracy			0.82	736
macro avg	0.82	0.75	0.77	736
weighted avg	0.82	0.82	0.81	736

AUC : 0.7536625091080535



Conclusion

- Approximately all the classifiers have same result, but the Ridge Classifier and the Random Forest were the best one.
- These two models have around 82% Accuracy.
- Ridge Classifier has 78% Precision, 58% Recall, & 67% F1 Score.
- Random Forest has 82% Precision, 56% Recall, & 66% F1 Score.
- We can also see the results for each classifier as well.

Model Deployment

```
In [73]: from sklearn.ensemble import StackingClassifier
```

```
In [74]: def Stacking(X_train,X_test,y_train,y_test):  
    #train the model  
    estimators = [('rf', RandomForestClassifier(n_estimators=10, random_state=42)), ('svr',  
    model = StackingClassifier(estimators=estimators, final_estimator=LogisticRegression(  
    model.fit(X_train, y_train)  
    #predictions  
    y_pre = model.predict(X_test)  
    evaluation_metrics(y_test, y_pre, target_names)
```

```
In [75]: ###Stacking classifier  
import pickle  
estimators = [('rf', RandomForestClassifier(n_estimators=10, random_state=42)), ('svr',  
final_model = StackingClassifier(estimators=estimators, final_estimator=LogisticRegress  
final_model.fit(X, y)  
filename = 'final_model.sav'  
pickle.dump(final_model, open(filename, 'wb'))
```