In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import RandomUnderSampler
from imblearn.over_sampling import SMOTE
import sklearn
from sklearn.preprocessing import scale
from sklearn import model_selection
from sklearn.linear_model import LinearRegression, Ridge, RidgeCV, Lasso, LassoCV, ElasticNet, ElasticNetCV
from sklearn.decomposition import PCA
from sklearn.cross_decomposition import PLSRegression
from sklearn.model_selection import KFold, cross_val_score
from sklearn.metrics import mean_squared_error
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns #visualization library

from sklearn.linear_model import LogisticRegression #problem will be solved with scikit
from sklearn.metrics import accuracy_score
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis # LDA
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis # QDA
from sklearn.neighbors import KNeighborsClassifier #(KNN)
from sklearn.metrics import confusion_matrix, classification_report, precision_score

import statsmodels.api as sm #to compute p-values
from patsy import dmatrices
%matplotlib inline
plt.style.use('seaborn-white')
```

In [2]:
```
df = pd.read_csv("desktop/fraudTest.csv")
```

In [3]:
```
df1 = df.iloc[:,1:]
```

In [4]:
```
df1
```

Out[4]:

	trans_date_trans_time	cc_num	merchant	category	amt	first	last	gender	street	city	...	lat	long	city_pop	job	dob	tr
0	2020-06-21 12:14:25	2291163933867244	fraud_Kirlin and Sons	personal_care	2.86	Jeff	Elliott	M	351 Darlene Green	Columbia	...	33.9659	-80.9355	333497	Mechanical engineer	1968-03-19	2da90c7d74bd46a0caf37774
1	2020-06-21 12:14:33	3573030041201292	fraud_Sporer-Keebler	personal_care	29.84	Joanne	Williams	F	3638 Marsh Union	Altonah	...	40.3207	-110.4360	302	Sales professional, IT	1990-01-17	324cc204407e99f51b0d6ca0
2	2020-06-21 12:14:53	3598215285024754	fraud_Swaniawski, Nitzsche and Welch	health_fitness	41.28	Ashley	Lopez	F	9333 Valentine Point	Bellmore	...	40.6729	-73.5365	34496	Librarian, public	1970-10-21	c81755dbbbea9d5c77f09434
3	2020-06-21 12:15:15	3591919803438423	fraud_Haley Group	misc_pos	60.05	Brian	Williams	M	32941 Krystal Mill Apt. 552	Titusville	...	28.5697	-80.8191	54767	Set designer	1987-07-25	2159175b9efe66dc301f149d
4	2020-06-21 12:15:17	3526826139003047	fraud_Johnston-Casper	travel	3.19	Nathan	Massey	M	5783 Evan Roads Apt. 465	Falmouth	...	44.2529	-85.0170	1126	Furniture designer	1955-07-06	57ff021bd3f328f8738bb535
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
555714	2020-12-31 23:59:07	30560609640617	fraud_Reilly and Sons	health_fitness	43.77	Michael	Olson	M	558 Michael Estates	Luray	...	40.4931	-91.8912	519	Town planner	1966-02-13	9b1f753c79894c9f4b71f045
555715	2020-12-31 23:59:09	3556613125071656	fraud_Hoppe-Parisian	kids_pets	111.84	Jose	Vasquez	M	572 Davis Mountains	Lake Jackson	...	29.0393	-95.4401	28739	Futures trader	1999-12-27	2090647dac2c89a1d86c514c
555716	2020-12-31 23:59:15	6011724471098086	fraud_Rau-Robel	kids_pets	86.88	Ann	Lawson	F	144 Evans Islands Apt. 683	Burbank	...	46.1966	-118.9017	3684	Musician	1981-11-29	6c5b7c8add471975aa0fec02
555717	2020-12-31 23:59:24	4079773899158	fraud_Breitenberg LLC	travel	7.99	Eric	Preston	M	7020 Doyle Stream Apt. 951	Mesa	...	44.6255	-116.4493	129	Cartographer	1965-12-15	14392d723bb7737606b2700a
555718	2020-12-31 23:59:34	4170689372027579	fraud_Dare-Marvin	entertainment	38.13	Samuel	Frey	M	830 Myers Plaza Apt. 384	Edmond	...	35.6665	-97.4798	116001	Media buyer	1993-05-10	1765bb45b3aa3224b4cdcb6e

555719 rows × 22 columns

In [5]:
```
df1['gender'] = df1['gender'].map({'F':0, 'M':1})
```

In [6]:
```
df1
```

Out[6]:

	trans_date_trans_time	cc_num	merchant	category	amt	first	last	gender	street	city	...	lat	long	city_pop	job	dob	tr
0	2020-06-21 12:14:25	2291163933867244	fraud_Kirlin and Sons	personal_care	2.86	Jeff	Elliott	1	351 Darlene Green	Columbia	...	33.9659	-80.9355	333497	Mechanical engineer	1968-03-19	2da90c7d74bd46a0caf37774
1	2020-06-21 12:14:33	3573030041201292	fraud_Sporer-Keebler	personal_care	29.84	Joanne	Williams	0	3638 Marsh Union	Altonah	...	40.3207	-110.4360	302	Sales professional, IT	1990-01-17	324cc204407e99f51b0d6ca0
2	2020-06-21 12:14:53	3598215285024754	fraud_Swaniawski, Nitzsche and Welch	health_fitness	41.28	Ashley	Lopez	0	9333 Valentine Point	Bellmore	...	40.6729	-73.5365	34496	Librarian, public	1970-10-21	c81755dbbbea9d5c77f09434
3	2020-06-21 12:15:15	3591919803438423	fraud_Haley Group	misc_pos	60.05	Brian	Williams	1	32941 Krystal Mill Apt. 552	Titusville	...	28.5697	-80.8191	54767	Set designer	1987-07-25	2159175b9efe66dc301f149d
4	2020-06-21 12:15:17	3526826139003047	fraud_Johnston-Casper	travel	3.19	Nathan	Massey	1	5783 Evan Roads Apt. 465	Falmouth	...	44.2529	-85.0170	1126	Furniture designer	1955-07-06	57ff021bd3f328f8738bb535
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
555714	2020-12-31 23:59:07	30560609640617	fraud_Reilly and Sons	health_fitness	43.77	Michael	Olson	1	558 Michael Estates	Luray	...	40.4931	-91.8912	519	Town planner	1966-02-13	9b1f753c79894c9f4b71f045
555715	2020-12-31 23:59:09	3556613125071656	fraud_Hoppe-Parisian	kids_pets	111.84	Jose	Vasquez	1	572 Davis Mountains	Lake Jackson	...	29.0393	-95.4401	28739	Futures trader	1999-12-27	2090647dac2c89a1d86c514c
555716	2020-12-31 23:59:15	6011724471098086	fraud_Rau-Robel	kids_pets	86.88	Ann	Lawson	0	144 Evans Islands Apt. 683	Burbank	...	46.1966	-118.9017	3684	Musician	1981-11-29	6c5b7c8add471975aa0fec02
555717	2020-12-31 23:59:24	4079773899158	fraud_Breitenberg LLC	travel	7.99	Eric	Preston	1	7020 Doyle Stream Apt. 951	Mesa	...	44.6255	-116.4493	129	Cartographer	1965-12-15	14392d723bb7737606b2700a
555718	2020-12-31 23:59:34	4170689372027579	fraud_Dare-Marvin	entertainment	38.13	Samuel	Frey	1	830 Myers Plaza Apt. 384	Edmond	...	35.6665	-97.4798	116001	Media buyer	1993-05-10	1765bb45b3aa3224b4cdcb6e

555719 rows × 22 columns

In [7]:
```
X = df1[["gender","city_pop","amt","lat"]]
Y = df1["is_fraud"]
```

In [8]:
```
X_train,X_test,Y_train,Y_test = sklearn.model_selection.train_test_split( X, Y, test_size=0.3, random_state = 10)
```

In [9]:
```
ros = RandomOverSampler(random_state=10)
X_oversampled, Y_oversampled = ros.fit_resample(X_train,Y_train)
```

In [10]:
```
rus = RandomUnderSampler(random_state=10)
X_undersampled, Y_undersampled = rus.fit_resample(X_train,Y_train)
```

In [11]:
```
oversample = SMOTE(random_state=10)
X_SMOTE,Y_SMOTE = oversample.fit_resample(X_train,Y_train)
```

In [12]:
```
lr = LogisticRegression()
```

In [13]:
```
mod1 = lr.fit(X_oversampled,Y_oversampled)
mod2 = lr.fit(X_undersampled,Y_undersampled)
mod3 = lr.fit(X_SMOTE,Y_SMOTE)
```

In [14]:
```
conf_mat = confusion_matrix(Y_test, mod1.predict(X_test))
print(conf_mat)

mod1.score(X_test, Y_test)
print('Accuracy =', mod1.score(X_test, Y_test))
```

```
[[157579   8453]
 [   168    516]]
Accuracy = 0.9482893063653158
```

In [20]:
```
conf_mat1 = confusion_matrix(Y_test, mod2.predict(X_test))
print(conf_mat1)

mod2.score(X_test, Y_test)
print('Accuracy =', mod2.score(X_test, Y_test))
```

```
[[157579   8453]
 [   168    516]]
Accuracy = 0.9482893063653158
```

In [21]:
```
conf_mat2 = confusion_matrix(Y_test, mod3.predict(X_test))
print(conf_mat2)

mod3.score(X_test, Y_test)
print('Accuracy =', mod3.score(X_test, Y_test))
```

```
[[157579   8453]
 [   168    516]]
Accuracy = 0.9482893063653158
```

In [15]:
```
mod1.score(X_test, Y_test)
print('Accuracy =', mod1.score(X_oversampled,Y_oversampled))
```

```
Accuracy = 0.8493840667592157
```

In [16]:
```
mod2.score(X_test, Y_test)
print('Accuracy =', mod2.score(X_undersampled,Y_undersampled))
```

```
Accuracy = 0.8463381245722108
```

In [17]:
```
mod3.score(X_test, Y_test)
print('Accuracy =', mod3.score(X_SMOTE,Y_SMOTE))
```

```
Accuracy = 0.8519579813284753
```

The logistic regression with X_SMOTE and Y_SMOTE performed the best.

In [18]:
```
raw_temp = pd.concat([X_train,Y_train],axis = 1)
```

In [19]:
```
plt.scatter(raw_temp[raw_temp['is_fraud'] == 0]['amt'], raw_temp[raw_temp['is_fraud'] == 0]['city_pop'])
plt.scatter(raw_temp[raw_temp['is_fraud'] == 1]['amt'], raw_temp[raw_temp['is_fraud'] == 1]['city_pop'])
plt.legend(['Fraud', 'Not Fraud'])
plt.xlabel('Amount')
plt.ylabel('Population')
plt.show()
```