In [494]:
```
import pandas as pd
from collections import Counter
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn import metrics

from array import *
```

Data Cleaning

In [541]:
```
data = pd.read_csv('C:/Users/Frank Shi/Desktop/ADS Project 4/compas-scores-two-ye
```

In [542]:
```
data.head()
```

Out[542]:

	id	name	first	last	compas_screening_date	sex	dob	age	age_cat	r
0	1	miguel hernandez	miguel	hernandez	14/08/2013	Male	18/04/1947	69	Greater than 45	O
1	3	kevon dixon	kevon	dixon	27/01/2013	Male	22/01/1982	34	25 - 45	Afri Amer
2	4	ed philo	ed	philo	14/04/2013	Male	14/05/1991	24	Less than 25	Afri Amer
3	5	marcu brown	marcu	brown	13/01/2013	Male	21/01/1993	23	Less than 25	Afri Amer
4	6	bouthy pierrelouis	bouthy	pierrelouis	26/03/2013	Male	22/01/1973	43	25 - 45	O

5 rows × 53 columns

In [543]:
```
### remove rows contains other races, update AA to be 1 and Cau to be 0
data = data[data["race"].str.contains("Other")==False]
data['race'] = data['race'].replace(['African-American', 'Caucasian'], [1, 0])
```

In [544]:
```
#### update vr_charge_degree to be dummy
data['vr_charge_degree']
data['vr_charge_degree'] = data['vr_charge_degree'].fillna('0')
data['vr_charge_degree'] = data['vr_charge_degree'].str.contains(pat = '0')
```

In [545]:
```
Counter(data['vr_charge_degree'])
```

Out[545]:
```
Counter({False: 781, True: 6056})
```

In [546]:
```
### DROP the following columns
df = data.drop(['type_of_assessment','id', 'name', 'first', 'last','compas_screen
df.head()
```

Out[546]:

	sex	age	age_cat	race	juv_fel_count	decile_score	juv_misd_count	juv_other_count	priors
1	Male	34	25 - 45	1	0	3	0	0	
2	Male	24	Less than 25	1	0	4	0	1	
3	Male	23	Less than 25	1	0	8	1	0	
6	Male	41	25 - 45	0	0	6	0	0	
8	Female	39	25 - 45	0	0	1	0	0	

5 rows × 24 columns

In [547]:
```
##fill na with 0
df['days_b_screening_arrest'].fillna(0, inplace=True)
df['c_days_from_compas'].fillna(0, inplace=True)
```

In [548]:
```
###Dummie transformation
to_dummy = ['sex', 'age_cat' ,'c_charge_degree', 'vr_charge_degree', 'v_score_tex
dummies = pd.get_dummies(df[to_dummy])
df = pd.concat([df, dummies], axis=1)
df = df.drop(to_dummy, axis=1)
```

In [549]:
```
df.isna().sum()
```

Out[549]:
```
age                          0
race                         0
juv_fel_count                0
decile_score                 0
juv_misd_count               0
juv_other_count              0
priors_count                 0
days_b_screening_arrest      0
c_days_from_compas           0
is_recid                     0
is_violent_recid             0
decile_score.1               0
v_decile_score               0
priors_count.1               0
start                        0
end                          0
event                        0
two_year_recid               0
sex_Female                   0
sex_Male                     0
age_cat_25 - 45              0
age_cat_Greater than 45      0
age_cat_Less than 25         0
c_charge_degree_F            0
c_charge_degree_M            0
v_score_text_High            0
v_score_text_Low             0
v_score_text_Medium          0
score_text_High              0
score_text_Low               0
score_text_Medium            0
dtype: int64
```

In [550]:
```
df_cau = df[df["race"] == 0]
df_aa = df[df["race"] == 1]
print(df_cau.shape[0])
print(df_aa.shape[0])
print('Number of Cau race Commit a Crime in 2 years',   df_cau[df_cau["two_year_r
print( 'Number of AA race Commit a Crime in 2 years', df_aa[df_aa["two_year_recid
print('Percentage of Cau race Commit a Crime in 2 years' ,df_cau[df_cau["two_year
print('Percentage of AA race Commit a Crime in 2 years', df_aa[df_aa["two_year_re


```

```
2454
3696
Number of Cau race Commit a Crime in 2 years 966
Number of AA race Commit a Crime in 2 years 1901
Percentage of Cau race Commit a Crime in 2 years 0.39364303178484106
Percentage of AA race Commit a Crime in 2 years 0.5143398268398268
```

In [551]:
```
## drop race
df_cau = df_cau.drop('race', axis=1)
df_aa = df_aa.drop('race', axis=1)
```

In [552]:
```
df
```

Out[552]:

	age	race	juv_fel_count	decile_score	juv_misd_count	juv_other_count	priors_count	da
1	34	1	0	3	0	0	0	
2	24	1	0	4	0	1	4	
3	23	1	0	8	1	0	1	
6	41	0	0	6	0	0	14	
8	39	0	0	1	0	0	0	
...	...	...	...	...	...	...	...	
7208	20	1	0	9	0	0	0	
7209	23	1	0	7	0	0	0	
7210	23	1	0	3	0	0	0	
7212	33	1	0	2	0	0	3	
7213	23	Hispanic	0	4	0	0	2	

6837 rows × 31 columns

In [553]:
```
###base model combined every thing
df = df.drop('race', axis=1)
X = df.drop("two_year_recid", axis=1)
y = df["two_year_recid"]
```

In [554]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

log = LogisticRegression()
log.fit(X_train, y_train)

y_pred = log.predict(X_test)
accuracy = metrics.accuracy_score(y_test, y_pred)
accuracy
```

```
C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.p
y:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[554]:
```
0.97953216374269
```

Local Sampling method (Logitic regression)

Race: African American. Local Sampling

In [555]:
```
###data split for 2 races
X_cau = df_cau.drop("two_year_recid", axis=1)
y_cau = df_cau["two_year_recid"]
X_train_cau, X_test_cau, y_train_cau, y_test_cau = train_test_split(X_cau, y_cau,


X_aa = df_aa.drop("two_year_recid", axis=1)
y_aa = df_aa["two_year_recid"]
X_train_aa, X_test_aa, y_train_aa, y_test_aa = train_test_split(X_aa, y_aa, test_
```

In [556]:
```
### Initial logistic regression on training data for African American
log_aa = LogisticRegression()
log_aa.fit(X_train_aa, y_train_aa)

y_pred_aa = log_aa.predict(X_test_aa)
accuracy = metrics.accuracy_score(y_test_aa, y_pred_aa)
print('ACC for AA without resampling: ', accuracy)
```

```
ACC for AA without resampling:  0.9648648648648649

C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.p
y:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

In [557]:
```
## probability table construction
dd = log_aa.predict_proba(X_train_aa)
(abs(dd[1,0] - dd[1,1]))
```

Out[557]:
```
0.9987336295627811
```

In [558]:
```
## calculate the logit differences
logit_diff = []
for i in range(len(dd)):
    logit_diff.append(abs(dd[i,0] - dd[i,1]))
```

In [559]:
```
np.array(logit_diff)[np.array(logit_diff) <= 0.4]   ###max logit = 0.65
print(len(np.array(logit_diff)[np.array(logit_diff) <= 0.4])) ### number of logit

### contains True False, length equals to number of rows in African American trai
position = np.array(logit_diff) <= 0.4
```

```
136
```

In [560]:
```
##label resample for df_aa
selected_rows = []  #### index with True
not_selected_rows = [] ### index with False

for i in range(len(position)):
    if position[i] == True :
        selected_rows.append(i)
    else:
        not_selected_rows.append(i)
```

In [561]:
```
### X and y with distance below threshold
selected_X = X_train_aa.iloc[selected_rows, ] ### with true in positions
selected_y = y_train_aa.iloc[selected_rows, ]

### X and y with distance above threshold
unselected_X = X_train_aa.iloc[not_selected_rows, ] ### with true in positions
unselected_y = y_train_aa.iloc[not_selected_rows, ]

### merge X and y
selected = pd.concat([selected_X, selected_y], axis=1)
unselected = pd.concat([unselected_X, unselected_y], axis=1)

### Only keep rows from selected that has two_year_recid == 0
### Duplicate kept rows by c
selected = selected[selected.two_year_recid == 0] #######remain the labels with 0
repeated = pd.concat([selected]*4, ignore_index=True)


### merge duplicated rows and unselected rows vertically
df_aa_train_new = pd.concat([unselected, repeated], axis=0)
(df_aa_train_new)
```

Out[561]:

	age	juv_fel_count	decile_score	juv_misd_count	juv_other_count	priors_count	days_b_scre
2793	60	0	8	0	0	9	
6376	20	0	6	0	0	2	
3255	23	0	9	0	0	6	
6947	23	2	6	0	0	3	
4772	39	0	6	0	0	19	
...	...	...	...	...	...	...	
275	22	0	5	0	0	1	
276	36	0	3	0	0	5	
277	31	0	8	0	0	14	
278	49	0	2	0	0	0	
279	61	0	2	0	0	14	

3100 rows × 30 columns

In [562]:
```
print(df_aa_train_new.shape[0])
print( 'Number of AA race Commit a Crime in 2 years', df_aa_train_new[df_aa_train
print('Percentage of AA race Commit a Crime in 2 years', df_aa_train_new[df_aa_tr
```

```
3100
Number of AA race Commit a Crime in 2 years 1440
Percentage of AA race Commit a Crime in 2 years 0.4645161290322581
```

In [563]:
```
print(Counter(y_train_aa))
Counter(y_train_cau)
770/(1184 +770)
```

```
Counter({1: 1506, 0: 1450})
```

Out[563]:
```
0.3940634595701126
```

In [564]:
```
print(Counter(y_train_aa))
1519/(1519+1437)
```

```
Counter({1: 1506, 0: 1450})
```

Out[564]:
```
0.5138700947225981
```

Local resampling on Causian

In [565]:
```
### Method 2 on Cau
log_cau = LogisticRegression()
log_cau.fit(X_train_cau, y_train_cau)

y_pred_cau = log_cau.predict(X_test_cau)
accuracy = metrics.accuracy_score(y_test_cau, y_pred_cau)
print('Acc for Cau wihtout resampling',accuracy)

dd = log_cau.predict_proba(X_train_cau)

## calculate the logit differences
logit_diff = []
for i in range(len(dd)):
    logit_diff.append(abs(dd[i,0] - dd[i,1]))

np.array(logit_diff)[np.array(logit_diff) <= 0.3]   ###max logit = 0.65
print(len(np.array(logit_diff)[np.array(logit_diff) <= 0.3])) ### number of logit
position = np.array(logit_diff) <= 0.3


##label resample for df_aa
selected_rows = []
not_selected_rows = []
for i in range(len(position)):
    if position[i] == True :
        selected_rows.append(i)
    else:
        not_selected_rows.append(i)

selected_X = X_train_cau.iloc[selected_rows, ] ### with true in positions
selected_y = y_train_cau.iloc[selected_rows, ]

unselected_X = X_train_cau.iloc[not_selected_rows, ] ### with true in positions
unselected_y = y_train_cau.iloc[not_selected_rows, ]

selected = pd.concat([selected_X, selected_y], axis=1)
unselected = pd.concat([unselected_X, unselected_y], axis=1)

selected = selected[selected.two_year_recid == 1] #######remain the labels with 1
repeated = pd.concat([selected]*5, ignore_index=True)


### merge duplicated rows and unselected rows vertically
df_cau_train_new = pd.concat([unselected, repeated], axis=0)
(df_cau_train_new)
```

```
Acc for Cau wihtout resampling 0.9389002036659878
82

C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logisti
c.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
```

```
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[565]:

	age	juv_fel_count	decile_score	juv_misd_count	juv_other_count	priors_count	days_b_scre
5926	69	0	5	0	0	0	
4820	25	0	3	0	0	0	
6235	57	0	1	0	0	2	
1772	54	0	1	0	0	0	
3569	32	0	3	0	0	1	
...	...	...	...	...	...	...	
150	31	0	5	0	2	2	
151	32	0	10	0	0	2	
152	47	0	10	0	0	5	
153	29	0	7	0	0	1	
154	22	0	10	0	0	3	

2036 rows × 30 columns

In [566]:
```
print(df_cau_train_new.shape[0])
print( 'Number of CAU race Commit a Crime in 2 years', df_cau_train_new[df_cau_tr
print('Percentage of CAU race Commit a Crime in 2 years', df_cau_train_new[df_cau
```

```
2036
Number of CAU race Commit a Crime in 2 years 896
Percentage of CAU race Commit a Crime in 2 years 0.4400785854616896
```

In [567]:
```
#### Overall ACC and Calibration
```

In [568]:
```
### merge the new training sets
df_train_new_total = pd.concat([df_aa_train_new, df_cau_train_new], axis=0)
df_train_new_x =  df_train_new_total.drop("two_year_recid", axis=1)
df_train_new_y = df_train_new_total["two_year_recid"]


model2 = LogisticRegression()
model2.fit(df_train_new_x, df_train_new_y)
```

```
C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.p
y:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[568]:
```
LogisticRegression()
```

In [569]:
```
### calibration for AA
y_pred_aa = model2.predict(X_test_aa)
accuracy_aa = metrics.accuracy_score(y_test_aa, y_pred_aa)
print('Accuracy for African American:' + str(accuracy_aa))

### calibration for Cau
y_pred_cau = model2.predict(X_test_cau)
accuracy_cau = metrics.accuracy_score(y_test_cau, y_pred_cau)
print('Accuracy for Cauasin:' + str(accuracy_cau))

### overall acc
print('Accuracy total:' + str((accuracy_cau+accuracy_aa)/2))
```

```
Accuracy for African American:0.972972972972973
Accuracy for Cauasin:0.9775967413441955
Accuracy total:0.9752848571585843
```

Local Massaging (logistic)

In [570]:
```
####Scratch
X = df_aa.drop("two_year_recid", axis=1)
y = df_aa["two_year_recid"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

log = LogisticRegression()
log.fit(X_train, y_train)

y_pred = log.predict(X_test)
accuracy = metrics.accuracy_score(y_test, y_pred)
accuracy


dd = log.predict_proba(X_train)
(abs(dd[1,0] - dd[1,1]))

## calculate the logit differences
logit_diff = []
for i in range(len(dd)):
    logit_diff.append(abs(dd[i,0] - dd[i,1]))

np.array(logit_diff)[np.array(logit_diff) <= 0.3]   ###max logit = 0.65
print(len(np.array(logit_diff)[np.array(logit_diff) <= 0.3])) ### number of logit
position = np.array(logit_diff) <= 0.3

##label update for df_aa
for i in range(len(position)):
    if position[i] == True :
        y_train.iloc[i] =0

log2 = LogisticRegression()
log2.fit(X_train, y_train)

y_pred = log2.predict(X_test)
accuracy = metrics.accuracy_score(y_test, y_pred)
accuracy
```

```
127
```

```
C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logisti
c.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
  n_iter_i = _check_optimize_result(
C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logisti
c.py:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[570]: 0.9702702702702702

local massage for AA

In [571]:
```
Counter(y_train_aa)
```

Out[571]: Counter({0: 1450, 1: 1506})

In [572]:
```
Counter(y_train_aa)[0]
```

Out[572]: 1450

In [573]:
```
print('Number of AA race Commit a Crime in 2 years', Counter(y_train_aa)[1]/(Cour
```

```
Number of AA race Commit a Crime in 2 years 0.5094722598105548
```

In [574]:
```
### Method 1AAogit differences
logit_diff_aa = []
for i in range(len(table_aa)):
    logit_diff_aa.append(abs(table_aa[i,0] - table_aa[i,1]))

print('Number of obervations below threshold',len(np.array(logit_diff_aa)[np.arra
position_aa = np.array(logit_diff_aa) <= 0.6


##label update
for i in range(len(position_aa)):
    if position_aa[i] == True :
        y_train_aa.iloc[i] =0
print(X_train_aa.shape)
print(len(y_train_aa))
```

```
Number of obervations below threshold 244
(2956, 29)
2956
```

In [575]:
```
Counter(y_train_aa) ##2956
```

Out[575]: Counter({0: 1563, 1: 1393})

In [576]:
```
print('Number of AA race Commit a Crime in 2 years', Counter(y_train_aa)[1]/(Cour
```

```
Number of AA race Commit a Crime in 2 years 0.4712449255751015
```

local massage for Cau

In [577]:
```
### Method 2 on Cau
y_pred_cau = log_cau.predict(X_test_cau)
table_cau = log_cau.predict_proba(X_train_cau)

## calculate the logit differences
logit_diff_cau = []
for i in range(len(table_cau)):
    logit_diff_cau.append(abs(table_cau[i,0] - table_cau[i,1]))

print('Number of obervations below threshold', len(np.array(logit_diff_cau)[np.ar
position_cau = np.array(logit_diff_cau) <= 0.5


##label update
for i in range(len(position_cau)):
    if position_cau[i] == True :
        y_train_cau.iloc[i] =1

print(X_train_cau.shape)
print(len(y_train_cau))
```

```
Number of obervations below threshold 140
(1963, 29)
1963
```

In [578]:
```
Counter(y_train_cau) ##1960
```

Out[578]:
```
Counter({0: 1106, 1: 857})
```

In [579]:
```
print('Number of Cau race Commit a Crime in 2 years', Counter(y_train_cau)[1]/(Cc
```

```
Number of Cau race Commit a Crime in 2 years 0.43657666836474784
```

In [580]:
```
pd.concat([y_train_aa, y_train_cau], axis=0)
```

Out[580]:
```
2793    0
6376    1
3255    1
6947    0
4772    1
       ..
6053    1
4412    0
5241    1
4305    0
3388    0
Name: two_year_recid, Length: 4919, dtype: int64
```

Overall acc and calibration

In [581]:
```
### merge the new training sets
X_total_new = pd.concat([X_train_aa, X_train_cau], axis=0)
y_total_new = pd.concat([y_train_aa, y_train_cau], axis=0)

model3 = LogisticRegression()
model3.fit(X_total_new, y_total_new)
```

```
C:\Users\Frank Shi\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.p
y:814: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[581]:
```
LogisticRegression()
```

In [582]:
```
### calibration for AA
y_pred_aa = model3.predict(X_test_aa)
accuracy_aa = metrics.accuracy_score(y_test_aa, y_pred_aa)
print('Accuracy for African American:' + str(accuracy_aa))

### calibration for Cau
y_pred_cau = model3.predict(X_test_cau)
accuracy_cau = metrics.accuracy_score(y_test_cau, y_pred_cau)
print('Accuracy for Cauasin:' + str(accuracy_cau))

### overall acc
print('Accuracy total:' + str((accuracy_cau+accuracy_aa)/2))
```

```
Accuracy for African American:0.9662162162162162
Accuracy for Cauasin:0.9592668024439919
Accuracy total:0.962741509330104
```