In [15]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statistics
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
```

In [36]:

```python
df = pd.read_csv('Default.csv')
df_copy = df.copy()
df_copy = df_copy.drop('Unnamed: 0', 1)
mapping = {'Yes': 1.0, 'No': 0.0}
df_copy['student'] = df_copy['student'].replace(mapping)
df_copy['default'] = df_copy['default'].replace(mapping)
df_copy.head()
```

Out[36]:

	default	student	balance	income
0	0.0	0.0	729.526495	44361.625074
1	0.0	1.0	817.180407	12106.134700
2	0.0	0.0	1073.549164	31767.138947
3	0.0	0.0	529.250605	35704.493935
4	0.0	0.0	785.655883	38463.495879

Problem a

In [37]:

```
y_true = df_copy['default']


# Make dataset of desired predictors
data = {
        'balance': df_copy['balance'],
        'income': df_copy['income'],
       }
X_train = pd.DataFrame(data)

# Create Logistic Regression model and fit to data
Logit_classifier = LogisticRegression()
Logit_classifier.fit(X_train, y_true)

# Print out the intercepts and coefficients of the Logistic Regression Model
print("The Intercept coefficient: " + repr(Logit_classifier.intercept_[0]))
print("The Balance coefficient: " + repr(Logit_classifier.coef_[0][0]))
print("The Income coefficient: " + repr(Logit_classifier.coef_[0][1]))
```

```
The Intercept coefficient: -1.9416412485359134e-06
The Balance coefficient: 0.00040756463034230067
The Income coefficient: -0.00012588074258791348
```

Problem b

In [46]:

```
y_true = df_copy['default']

# Make dataset of desired predictors
data = {
        'balance': df_copy['balance'],
        'income': df_copy['income'],
       }
desired_data = pd.DataFrame(data)

# VALIDATION SET APPROACH
# Splits dataset into training set and validation set
X_train, X_valid, y_train, y_valid = train_test_split(desired_data, y_true, te
st_size=0.20)

# Create Logistic Regression model and fit to TRAINING data
Logit_classifier = LogisticRegression()
Logit_classifier.fit(X_train, y_train)

# Make prediction for validation set
y_pred = Logit_classifier.predict(X_valid)

# Compute validation set error
num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_valid, y_pred).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
valid_set_accuracy = (num_TP + num_TN)/(num_TP + num_TN + num_FP + num_FN)
valid_set_error = 1 - valid_set_accuracy
valid_set_error
```

```
Number of True Positives: 0
Number of False Negatives: 75
Number of False Positives: 0
Number of True Negatives: 1925
```

Out[46]:

```
0.03749999999999998
```

Problem c

In [45]:

```
y_true = df_copy['default']

# Make dataset of desired predictors
data = {
        'balance': df_copy['balance'],
        'income': df_copy['income'],
       }
desired_data = pd.DataFrame(data)

numExperiments = 3
errors_list = []


# VALIDATION SET APPROACH
for i in range(numExperiments):
    # Splits dataset into training set and validation set
    X_train, X_valid, y_train, y_valid = train_test_split(desired_data, y_true
, test_size=0.20)

    # Create Logistic Regression model and fit to TRAINING data
    Logit_classifier = LogisticRegression()
    Logit_classifier.fit(X_train, y_train)

    # Make prediction for validation set
    y_pred = Logit_classifier.predict(X_valid)

    # Compute validation set error
    num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_valid, y_pred).ravel()
    valid_set_accuracy = (num_TP + num_TN)/(num_TP + num_TN + num_FP + num_FN)
    valid_set_error = 1 - valid_set_accuracy
    errors_list.append(valid_set_error)

errors_list
```

Out[45]:

```
[0.035499999999999976, 0.03400000000000003, 0.031000000000000028]
```

Problem d

In [52]:

```
y_true = df_copy['default']

# Make dataset of desired predictors
data = {
        'balance': df_copy['balance'],
        'income': df_copy['income'],
        'student': df_copy['student']
       }
desired_data = pd.DataFrame(data)

numExperiments = 3
errors_list = []


# VALIDATION SET APPROACH
for i in range(numExperiments):
    # Splits dataset into training set and validation set
    X_train, X_valid, y_train, y_valid = train_test_split(desired_data, y_true
, test_size=0.20)

    # Create Logistic Regression model and fit to TRAINING data
    Logit_classifier = LogisticRegression()
    Logit_classifier.fit(X_train, y_train)

    # Make prediction for validation set
    y_pred = Logit_classifier.predict(X_valid)

    # Compute validation set error
    num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_valid, y_pred).ravel()
    valid_set_accuracy = (num_TP + num_TN)/(num_TP + num_TN + num_FP + num_FN)
    valid_set_error = 1 - valid_set_accuracy
    errors_list.append(valid_set_error)

errors_list
```

Out[52]:

```
[0.03149999999999997, 0.034499999999999975, 0.033499999999999974]
```

In [ ]: