5. In Chapter 4, we used logistic regression to predict the probability of default using income and balance on the Default data set. We will now estimate the test error of this logistic regression model using the validation set approach. Do not forget to set a random seed before beginning your analysis.

Importing python libraries

In [1]:

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.utils.validation import column_or_1d
```

In [2]:

```
df = pd.read_csv('/Volumes/work/sem_1/MTH522/data/Default.csv')
df.head()
```

Out[2]:

	default	student	balance	income
0	No	No	729.526495	44361.625074
1	No	Yes	817.180407	12106.134700
2	No	No	1073.549164	31767.138947
3	No	No	529.250605	35704.493935
4	No	No	785.655883	38463.495879

(a) Fit a logistic regression model that uses income and balance to predict default.

In [3]:

```
Y = df['default']
X = df.drop(['default', 'student'], axis=1)

X.head()
```

Out[3]:

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

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

Out[4]:
```
0    No
1    No
2    No
3    No
4    No
Name: default, dtype: object
```

In [5]:
```
model = LogisticRegression(random_state=5).fit(X, Y)
model.predict(X)
model.score(X, Y)
```

Out[5]:
```
0.9737
```

(b) Using the validation set approach, estimate the test error of this model. In order to do this, you must perform the following steps:

i. Split the sample set into a training set and a validation set.

In [6]:
```
train, valid_test = train_test_split(df.drop(['student'], axis=1), test_s
valid, test = train_test_split(valid_test, test_size=0.5, random_state=42
```

In [7]:
```
print(test.shape, test.shape)
```

```
(1500, 3) (1500, 3)
```

ii. Fit a multiple logistic regression model using only the training observations.

In [8]:
```
model_2 = LogisticRegression(random_state=5).fit(train.iloc[:, 1:], colum
model_2
```

Out[8]:
```
▾           LogisticRegression
LogisticRegression(random_state=5)
```

iii. Obtain a prediction of default status for each individual in the validation set by computing the posterior probability of default for that individual, and classifying the individual to the default category if the posterior probability is greater than 0.5.

In [9]:

```
probs = model.predict_proba(valid.iloc[:, 1:])
probs[:5]
```

Out[9]:

```
array([[9.74301203e-01, 2.56987967e-02],
       [9.99950985e-01, 4.90149459e-05],
       [9.99815785e-01, 1.84214590e-04],
       [9.76006252e-01, 2.39937481e-02],
       [9.99773158e-01, 2.26841521e-04]])
```

iv. Compute the validation set error, which is the fraction of the observations in the validation set that are misclassified.

In [10]:

```
valid_error = 1 - model_2.score(valid.iloc[:, 1:], valid.iloc[:, :1])
print('Validation error is ', valid_error * 100)
test_error = 1 - model_2.score(test.iloc[:, 1:], test.iloc[:, :1])
print('Test error is ', test_error * 100)
```

```
Validation error is  3.3333333333333326
Test error is  3.4666666666666623
```

(c) Repeat the process in (b) three times, using three different splits of the observations into a training set and a validation set. Comment on the results obtained.

First

In [11]:

```
def f(df, random_state_1, random_state_2, random_state_3):
    train, valid_test = train_test_split(df.drop(['student'], axis=1), te
    valid, test = train_test_split(valid_test, test_size=0.5, random_stat
                                   stratify=valid_test['default'])

    model = LogisticRegression(random_state=random_state_3).fit(train.ilo

    valid_err = 1 - model.score(valid.iloc[:, 1:], valid.iloc[:, :1])
    test_err = 1 - model.score(test.iloc[:, 1:], test.iloc[:, :1])

    return valid_err, test_err
```

In [12]:

```
valid_err, test_err = f(df=df, random_state_1=100, random_state_2=200, ra
print('Valid error is ', valid_err * 100)
print('Test error is ', test_err * 100)
```

```
Valid error is  3.200000000000003
Test error is  3.0000000000000027
```

Second

In [13]:

```
valid_err, test_err = f(df=df, random_state_1=1000, random_state_2=2000,
print('Valid error is ', valid_err * 100)
print('Test error is ', test_err * 100)
```

```
Valid error is  3.1333333333333324
Test error is  3.1333333333333324
```

Third

In [14]:

```
valid_err, test_err = f(df=df, random_state_1=10, random_state_2=20, rand
print('Valid error is ', valid_err * 100)
print('Test error is ', test_err * 100)
```

```
Valid error is  3.8000000000000034
Test error is  3.7333333333333333
```

(d) Now consider a logistic regression model that predicts the probability of default using income, balance, and a dummy variable for student. Estimate the test error for this model using the validation set approach. Comment on whether or not including a dummy variable for student leads to a reduction in the test error rate.

In [15]:

```
df_new = pd.get_dummies(df['student'])
df_new = pd.concat([df, df_new], axis=1).drop(['student'], axis=1)
df_new.head()
```

Out[15]:

	default	balance	income	No	Yes
0	No	729.526495	44361.625074	1	0
1	No	817.180407	12106.134700	0	1
2	No	1073.549164	31767.138947	1	0
3	No	529.250605	35704.493935	1	0
4	No	785.655883	38463.495879	1	0

In [16]:

```
def f_new(df, random_state_1, random_state_2, random_state_3):
    train, valid_test = train_test_split(df, test_size=0.3, random_state=
    valid, test = train_test_split(valid_test, test_size=0.5, random_stat
                                   stratify=valid_test['default'])

    model = LogisticRegression(random_state=random_state_3).fit(train.ilo

    valid_err = 1 - model.score(valid.iloc[:, 1:], valid.iloc[:, :1])
    test_err = 1 - model.score(test.iloc[:, 1:], test.iloc[:, :1])

    return valid_err, test_err
```

In [17]:

```
valid_err, test_err = f_new(df=df_new, random_state_1=400, random_state_2
print('Valid error is ', valid_err * 100)
print('Test error is ', test_err * 100)
```

```
Valid error is  2.8000000000000025
Test error is  2.6000000000000023
```

Observation:

1. With the addition of an extra feature the error has been increased.

In [17]: