Introduction to Big Data

- Developed by Dr. Keungoui KIM
- https://awekim.github.io/portfolio/

Lecture 11. Classification

```
from google.colab import drive
drive.mount('/content/drive')
```

```
import numpy as np
import pandas as pd
import seaborn as sns
```

Classification with Personal Loan Data

- Experience
- Income
- Famliy
- CCAvg: Average monthly card spent
- Education: Education level (1: undergrad; 2, Graduate; 3; Advance)
- Mortgage
- Securities account: Securities (1:Yes, 0:No)
- CD account: CD account (1:Yes, 0:No)
- Online: Online account (1:Yes, 0:No)
- CreidtCard: Credit Card (1:Yes, 0:No)

```
PerLoan = pd.read_csv("/content/drive/MyDrive/[Lecture]/IntBigData/BigData_Python/11_Classification/pe
PerLoan.head()
```

```
PerLoan.shape
```

```
PerLoan.columns
```

```
PerLoan.rename(columns={'Personal Loan':'PersonalLoan'},
               inplace=True)
```

```
PerLoan.columns
```

```
PerLoan.describe()
```

```
# check missing values
PerLoan.isnull().any()
```

```
PerLoan.count()
```

```
PL_X = PerLoan[['Age','CCAvg','Income','Education']]
PL_Y = PerLoan['PersonalLoan']
```

Logit Regression with statsmodels

```
import statsmodels.api as sm

statsLogitModel = sm.Logit(PL_Y, PL_X)
statsLogitModel

from statsmodels.formula.api import logit

statsLogitModel = (
    logit('PersonalLoan ~ Age + CCAvg + Income + Education',
          data=PerLoan))
statsLogitModel

statsLogitModel_res = statsLogitModel.fit()

print(statsLogitModel_res.summary())

statsLogitModel_res.params

np.exp(statsLogitModel_res.params)
```

Logit Regression with sklearn

```
# from sklearn import metrics
from sklearn.linear_model import LogisticRegression

LogitModel0 = LogisticRegression()

LogitModel0_res = LogitModel0.fit(PL_X, PL_Y)
LogitModel0_res

LogitModel0_res.coef_

LogitModel0_res.intercept_

from sklearn.model_selection import train_test_split

PL_X_train, PL_X_test, PL_Y_train, PL_Y_test = train_test_split(PL_X, PL_Y, test_size=0.3, random_stat

# Practice of Random Sampling
PL_X_train1, PL_X_test1, PL_Y_train1, PL_Y_test1 = train_test_split(PL_X, PL_Y,
                                                                    test_size=0.3)
PL_X_train1.shape

PL_X_train1, PL_X_test1, PL_Y_train1, PL_Y_test1 = train_test_split(PL_X, PL_Y,
                                                                    test_size=0.3)
PL_X_train1.head()

PL_X_train1, PL_X_test1, PL_Y_train1, PL_Y_test1 = train_test_split(PL_X, PL_Y,
                                                                    test_size=0.3, random_state=1)
PL_X_train1.head()

PL_X_train1, PL_X_test1, PL_Y_train1, PL_Y_test1 = train_test_split(PL_X, PL_Y,
                                                                    test_size=0.3, random_state=1)
PL_X_train1.head()

PL_X_train
```

```
PL_X_test
```

```
PL_Y_train
```

```
PL_Y_test
```

```
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

PL_X_train, PL_X_test, PL_Y_train, PL_Y_test = (
    train_test_split(PL_X, PL_Y,
                     test_size=0.3,
                     random_state=0) )

LogitModel = LogisticRegression()
LogitModel.fit(PL_X_train, PL_Y_train)

PL_Y_train_pred = LogitModel.predict(PL_X_train)
PL_Y_test_pred = LogitModel.predict(PL_X_test)
```

Performance Check

```
from sklearn.metrics import accuracy_score, recall_score
from sklearn.metrics import precision_score, f1_score, confusion_matrix

# train set
print(confusion_matrix(PL_Y_train, PL_Y_train_pred))
print(accuracy_score(PL_Y_train, PL_Y_train_pred))
print(recall_score(PL_Y_train, PL_Y_train_pred))
print(precision_score(PL_Y_train, PL_Y_train_pred))
print(f1_score(PL_Y_train, PL_Y_train_pred))

# test set
print(confusion_matrix(PL_Y_test, PL_Y_test_pred))
print(accuracy_score(PL_Y_test, PL_Y_test_pred))
print(recall_score(PL_Y_test, PL_Y_test_pred))
print(precision_score(PL_Y_test, PL_Y_test_pred))
print(f1_score(PL_Y_test, PL_Y_test_pred))
```

```
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score, confusion_matrix
```

Accuracy Score

```
accuracy_score(PL_Y_test, PL_Y_test_pred)
```

Recall Score

```
recall_score(PL_Y_test, PL_Y_test_pred)
```

Precision Score

```
precision_score(PL_Y_test, PL_Y_test_pred)
```

F1 Score

```
f1_score(PL_Y_test, PL_Y_test_pred)
```

Confusion Matrix

```
confusion_matrix(PL_Y_test, PL_Y_test_pred)
```

Specificity

```
tn, fp, fn, tp = confusion_matrix(PL_Y_test, PL_Y_test_pred).ravel()
specificity = tn / (tn+fp)
specificity
```

ROC Curve

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
import matplotlib.pyplot as plt

logit_roc_auc = roc_auc_score(PL_Y_test, LogitModel.predict(PL_X_test))
fpr, tpr, thresholds = roc_curve(PL_Y_test, LogitModel.predict_proba(PL_X_test)[:,1])

plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
```