**교육일지**

**교육 제목 : 머신러닝(로지스틱, 확률적 경사하강법)**

**교육 장소 : YGL C6 강의실**

**교육 일시 : 2021/10/19**

**로지스틱 회귀**

**fish = pd.read\_csv('https://bit.ly/fish\_csv\_data')**

**fish.head()**

**#print(fish.shape)**

**fish['Species'].value\_counts()**

**fish\_input=fish[['Weight','Length','Diagonal','Height','Width']].to\_numpy()**

**#fish\_input.head()**

**fish\_input[:5]**

**fish\_target = fish['Species'].to\_numpy()**

**fish\_target[:10]**

**from sklearn.model\_selection import train\_test\_split**

**train\_input, test\_input, train\_target, test\_target = train\_test\_split(**

**fish\_input, fish\_target, stratify = fish\_target)**

**print(np.shape(train\_input))**

**print(np.shape(test\_input))**

**from sklearn.preprocessing import StandardScaler**

**ss = StandardScaler()**

**ss.fit(train\_input)**

**train\_scaled = ss.transform(train\_input)**

**test\_scaled = ss.transform(test\_input)**

**z = np.arange(-5, 5, 0.1)**

**prob\_y= 1/(1 + np.exp(-z))**

**plt.plot(z, prob\_y)**

**plt.axhline(1, linestyle = "--", color = "r")**

**plt.axhline(0, linestyle = "--", color = "r")**

**plt.axhline(0.5, linestyle = "--", color = "b")**

**plt.show()**

**#Boolean index**

**bream\_smelt\_index = (train\_target =="Bream") | (train\_target == "Smelt")**

**#bream\_smelt\_index**

**#train\_target**

**train\_bream\_smelt = train\_scaled[bream\_smelt\_index]**

**target\_bream\_smelt = train\_target[bream\_smelt\_index]**

**print(np.shape(train\_scaled))**

**print(np.shape(train\_bream\_smelt))**

**from sklearn.linear\_model import LogisticRegression**

**lr = LogisticRegression(C=20, max\_iter=1000)**

**lr.fit(train\_bream\_smelt, target\_bream\_smelt)**

**lr.score(train\_bream\_smelt, target\_bream\_smelt)**

**print(train\_bream\_smelt[:5])**

**print(lr.predict\_proba(train\_bream\_smelt[:5]))**

**print(lr.predict(train\_bream\_smelt[:5]))**

**print(lr.coef\_, lr.intercept\_)**

**#C = 1/lambda**

**lr = LogisticRegression(C = 20, max\_iter = 1000)**

**lr.fit(train\_scaled, train\_target)**

**print(lr.score(train\_scaled, train\_target))**

**print(lr.score(test\_scaled, test\_target))**

**print(lr.predict(test\_scaled[:10]))**

**print(test\_target[:10])**

**lr.predict\_proba(test\_scaled[:5]).round(3)**

**print(lr.classes\_)**

**print(lr.coef\_)**

**print(lr.intercept\_)**

**확률적 경사 하강법**

**from sklearn.linear\_model import SGDClassifier**

**sc = SGDClassifier(loss = "log", max\_iter = 100)**

**sc.fit(train\_scaled, train\_target)**

**print(sc.score(train\_scaled, train\_target))**

**print(sc.score(test\_scaled, test\_target))**

**## Partial fit**

**sc.partial\_fit(train\_scaled, train\_target)**

**print(sc.score(train\_scaled, train\_target))**

**print(sc.score(test\_scaled, test\_target))**

**classes = np.unique(train\_target)**

**sc = SGDClassifier(loss='log')**

**train\_score = []**

**test\_score = []**

**for \_ in range(0, 300):**

**sc.partial\_fit(train\_scaled, train\_target, classes=classes)**

**train\_score.append(sc.score(train\_scaled, train\_target))**

**test\_score.append(sc.score(test\_scaled, test\_target))**

**plt.plot(train\_score, label = "Train")**

**plt.plot(test\_score, label = "Test")**

**plt.ylim(0.5, 1.04)**

**plt.legend()**

**plt.show()**