**SenseStudy課程平臺“人工智能入門（上）”**

**實驗5 – 2 多項式回歸模型**

train\_x = [27,29,34,40,42,47,48,49,50,52,52,52,54]

train\_y = [6,7.5,9,10.7,12.8,15.1,16,18.5,19.4,18.4,19.7,21.8,21.7]

fig() + scatter(train\_x, train\_y)

model = linear\_regressor()

model.train(train\_x, train\_y)

model.show()

x=40

pred\_y = model.predict(x)

print(pred\_y)

train\_x = [27,29,34,40,42,47,48,49,50,52,52,52,54]

train\_y = [6,7.5,9,10.7,12.8,15.1,16,18.5,19.4,18.4,19.7,21.8,21.7]

model = linear\_regressor()

model.train(train\_x, train\_y)

model.show()

x=40; pred\_y = model.predict(x); print(pred\_y)

model = poly\_regressor(2)

model.train(train\_x, train\_y)

model.show()

x=40; pred\_y = model.predict(x); print(pred\_y)

model = poly\_regressor(30)

model.train(train\_x, train\_y)

model.show()

x=40; pred\_y = model.predict(x); print(pred\_y)

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗5 – 3 線性回歸模型評估與測試集**

train\_x = [27,29,34,40,42,47,48,49,50,52,52,52,54]

train\_y = [6,7.5,9,10.7,12.8,15.1,16,18.5,19.4,18.4,19.7,21.8,21.7]

model = linear\_regressor()

model.train(train\_x, train\_y)

def compute\_error(model, x, y):

pred = model.predict(x)

error = 0

for i in range(len(pred)):

error = error + (y[i]-pred[i])\*\*2

error = error / len(pred)

return error

print(compute\_error(model, train\_x, train\_y))

model2 = poly\_regressor(3)

model2.train(train\_x, train\_y)

print(compute\_error(model2, train\_x, train\_y))

model3 = poly\_regressor(30)

model3.train(train\_x, train\_y)

print(compute\_error(model3, train\_x, train\_y))

test\_x = [23,31,32,38,40,45,49,50,50,51,51,53,55]

test\_y = [6.3,7.2,9.1,10.5,12.9,15.5,15.9,18.3,19.7,18.9,19.3,21.3,22.1]

print(compute\_error(model, test\_x, test\_y))

print(compute\_error(model2, test\_x, test\_y))

print(compute\_error(model3, test\_x, test\_y))

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗5 – 5 線性分類模型**

train\_x = [60,56,60,55,60,57,65,60,62,59,43,52,41,45,43,50,46,52,56,56]

train\_y = [1,1,1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1]

model = linear\_classifier()

model.train(train\_x,train\_y)

model.show()

x=60.5

pred\_y = model.predict(x)

print(pred\_y)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗7 – 1 線性分類器-商鋪類型分類**

coor\_x, coor\_y, label = load('eshop.train')

fig() + scatter(coor\_x, coor\_y, c=label)

model = LinearClassifier()

feat = model.merge\_features(coor\_x, coor\_y)

model.train(feat,label)

coor\_tx, coor\_ty, t\_label = load('eshop.test')

t\_feat = model.merge\_features(coor\_tx, coor\_ty)

pred = model.predict(t\_feat)

accuracy = model.get\_accuracy(t\_feat, t\_label)

print("the predicted accuracy is %f"%accuracy)

coor\_x, coor\_y, latent\_v, label = load('tr\_eshop.train')

feat = model.merge\_features(coor\_x, coor\_y, latent\_v)

model.train(feat, label)

coor\_tx, coor\_ty, tlatent\_v, t\_label = load('tr\_eshop.test')

t\_feat = model.merge\_features(coor\_tx, coor\_ty, tlatent\_v)

pred = model.predict(t\_feat)

accuracy = model.get\_accuracy(t\_feat, t\_label)

print("the predicted accuracy is %f"%accuracy)

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗4 – 1 圖像資料基礎**

fig() + image(img\_gray, cmap='gray')

print(img\_gray.shape)

printMat2D(img\_gray)

img\_gray\_copy = image\_copy(img\_gray)

img\_gray\_copy[18,16] = 255

fig() + image(img\_gray\_copy, cmap='gray')

img\_gray\_copy = image\_copy(img\_gray)

img\_gray\_copy[18:25,16] = 255

fig() + image(img\_gray\_copy, cmap='gray')

img\_gray\_copy = image\_copy(img\_gray)

img\_gray\_copy[18:25,16:25] = 255

fig() + image(img\_gray\_copy, cmap='gray')

img\_gray\_copy = image\_copy(img\_gray)

img\_gray\_copy[10:,:] = 255

fig() + image(img\_gray\_copy, cmap='gray')

fig() + image(img\_shibe)

print(img\_shibe.shape)

r=img\_shibe[:,:,0]

g=img\_shibe[:,:,1]

b=img\_shibe[:,:,2]

fig(1,3)+[image(r, cmap='Reds'), image(g, cmap='Greens'), image(b, cmap='Blues')]

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗4 – 2 影像處理技術-色彩變換**

fig() + image(img\_raw)

img\_bgr = image\_copy(img\_raw)

tmp = image\_copy(img\_bgr[:,:,0])

img\_bgr[:,:,0] = img\_bgr[:,:,2]

img\_bgr[:,:,2] = tmp

fig() + image(img\_bgr)

url = 'https://raw.githubusercontent.com/yzhang0301/codes/master/chow.jpg'

img\_raw = imread(url)

fig() + image(img\_raw)

mat = [

[0.393, 0.769, 0.189],

[0.349, 0.686, 0.168],

[0.272, 0.534, 0.131]

]

img\_retro = image\_copy(img\_raw)

img\_retro = image\_int2float(img\_retro)

img\_retro = map\_color\_space(img\_retro, mat)

img\_retro = bound(img\_retro, 0, 255)

img\_retro = image\_float2int(img\_retro)

fig() + image(img\_retro)

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗4 – 4 表情包1 – 圖像扭曲**

print(img\_boy.shape)

warp\_params = [(100,100), (0,20), 50]

img\_warp = local\_warp\_image(img\_boy, warp\_params)

fig(1,2) + [image(img\_boy),image(img\_warp)]

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗4 – 5 表情包2-人臉關鍵點檢測**

url = 'https://raw.githubusercontent.com/yzhang0301/codes/master/chow.jpg'

img\_wzj = imread(url)

points=detect\_keypoints(img\_wzj)

img\_girl\_with\_points=draw\_points(img\_wzj,points)

fig()+image(img\_wzj)

fig()+image(img\_girl\_with\_points)

mouth\_left = points[48]

mouth\_right = points[54]

left\_eyebrow = points[19]

right\_eyebrow = points[24]

warp\_params = [

[left\_eyebrow, (0, -5), 35],

[right\_eyebrow, (0, -5), 35],

[mouth\_left, (-5, -5), 40],

[mouth\_right, (5, -5), 40]

]

sticker = make\_sticker(img\_wzj, warp\_params)

fig() + gif(sticker)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗7 – 2 深度學習體驗: 深度學習-人工神經網路**

x\_train, y\_train, l\_train = load\_dataset("data/train.npz")  
fig() + scatter(x\_train, y\_train, c=l\_train)

model = DeepLearning([2,4,4,1])

fig() + structure(model)  
f\_train = merge\_features([x\_train, y\_train])  
model.demo\_train('business')  
model.show\_learning\_curve()

x\_test, y\_test, l\_test = load\_dataset("data/test.npz")  
f\_test = merge\_features([x\_test, y\_test])  
pred = model.predict(f\_test)  
print(accuracy(pred > 0.5, l\_test))

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗7 – 3 神經網路回歸-基於身高預測體重**

train\_x, train\_y = load('hw.train')

test\_x, test\_y = load('hw.test')

fig() + scatter(train\_x, train\_y)

net=MLP([1,4,4,1])

fig()+structure(net)

net.train(train\_x,train\_y)

pred=net.predict(data=test\_x)

fig()+scatter(test\_x, test\_y)+scatter(test\_x, pred)

error=net.compute\_error(pred,test\_y)

print("test error = %f" % error)

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗5 – 8 圖像分類**

train\_x, train\_y, test\_x, test\_y = load\_CelebA\_sample\_dataset()

for i in range(5):

sample = train\_x[i\*20]

img\_sample = list2numpy(sample)

fig() + image(img\_sample)

print(train\_y[i\*20])

train\_x\_flat = flatten(train\_x)

test\_x\_flat = flatten(test\_x)

model = linear\_classifier()

model.train(train\_x\_flat, train\_y)

pred = model.predict(test\_x\_flat, fake=False)

acc = accuracy(pred, test\_y)

print('The accuracy is: ',acc)

for i in range(len(test\_x)):

if pred[i] != test\_y[i]:

sample = test\_x[i]

img\_sample = list2numpy(sample)

fig() + image(img\_sample)

print('Prediction is: ',pred[i], ' The truth is: ',test\_y[i])

train\_feat, test\_feat = load\_CelebA\_features()

model = linear\_classifier()

model.train(train\_feat, train\_y)

pred\_y = model.predict(test\_feat, fake=False)

test\_acc = accuracy(pred\_y, test\_y)

print('The accuracy is: ',test\_acc)

for i in range(len(test\_feat)):

if pred\_y[i] != test\_y[i]:

sample = test\_x[i]

img\_sample = list2numpy(sample)

fig() + image(img\_sample)

print('Prediction is: ',pred\_y[i], ' The truth is: ',test\_y[i])

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗7 – 4 線性分類器技術-圖像分類**

train\_data, train\_lable, test\_data = load\_data()

show\_data(train\_data)

print(train\_lable)

train\_features = load\_hog('train\_features')

test\_features = load\_hog('test\_features')

model = LinearClassifier()

model.train(train\_features, train\_lable)

acc, Y\_predict = model.pred(test\_features)

show\_data(test\_data)

print("Prediction: ", Y\_predict, "accuracy is: ",acc)

**SenseStudy課程平臺“人工智能入門（上）”**

**實驗4 – 7 認識卷積核**

fig() + image(img\_furn)

img\_gray = rgb2gray(img\_furn)

v\_kernel =[

[-1,0,1],

[-1,0,1],

[-1,0,1]

]

img\_v = apply\_kernel(img\_gray, v\_kernel)

img\_v = abs(img\_v)

fig() + image(img\_v, cmap='gray')

**SenseStudy課程平臺“人工智能基礎（體驗版）”**

**實驗3 – 2 利用深度神經網路完成圖像分類**

cifar\_train = data.get('cifar10-small', subset='train')

fig() + plot(cifar\_train)

label\_names = cifar\_train.meta['label\_names']

print(label\_names)

net = CNNClassifier(in\_shape=(32,32,3),backbone=ResNet(10),num\_classes=10)

net.demo\_train(cifar\_train)

cifar\_test = data.get('cifar10-small',subset='test')

img, label = cifar\_test[300]

fig() + plot(img)

label\_name = label\_names[label]

pred = net.predict(img)

pred\_name = label\_names[pred]

print("Ground truth is {}, Prediction is {}.".format(label\_name,pred\_name))

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 2 分詞和列表**

text = readtext("./nlp/data/textbooks/grade0/text0.txt")

words = splitwords(text)

print(words)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 3 字典和詞頻統計**

word\_freq\_dict = {'you': 0.098, 'the': 0.020, 'friend': 0.059}

print(word\_freq\_dict)

print(word\_freq\_dict['you'])

word\_freq\_dict['you'] = 0.088

print(word\_freq\_dict)

print('you' in word\_freq\_dict)

print('he' in word\_freq\_dict)

for key, value in word\_freq\_dict.items():

print(key + ':' + str(value))

def word\_freq(words):

freq\_dict = {}

for word in words:

if word in freq\_dict:

freq\_dict[word] += 1

else:

freq\_dict[word] = 1

for word, freq in freq\_dict.items():

freq\_dict[word] = freq / len(words)

return freq\_dict

text = readtext("./nlp/data/textbooks/grade0/text0.txt")

words = splitwords(text)

freq\_dict = word\_freq(words)

print(freq\_dict)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 4 智能課文分析1**

textbooks\_data = load\_textbooks\_data()

print(textbooks\_data)

print(len(textbooks\_data))

def get\_diff\_level(path\_grade):

diff\_level = {}

for path, grade in path\_grade:

text = readtext(path)

words = splitwords(text)

#grade = int(grade)

for word in words:

if word in diff\_level and diff\_level[word] <= grade:

continue

else:

diff\_level[word] = grade

return diff\_level

diff\_level = get\_diff\_level(textbooks\_data)

print(diff\_level)

save\_private(diff\_level, "./data/tmp/diff\_level")

diff\_level = load\_private("./data/tmp/diff\_level")

print(diff\_level)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 5 智能課文分析2**

diff\_level = load\_private("./data/tmp/diff\_level")

def extract\_features(path, diff\_level):

text = readtext(path)

words = splitwords(text)

grade\_freq = [0]\*12

for word in words:

if word in diff\_level:

grade = diff\_level[word]

grade\_freq[grade] += 1

else:

continue

num = sum(grade\_freq)

for i in range(12):

grade\_freq[i] /= num

return grade\_freq

grade\_freq = extract\_features("./nlp/data/reading/train/text0.txt", diff\_level)

print(grade\_freq)

grade\_freq = extract\_features("./nlp/data/reading/train/text1.txt", diff\_level)

print(grade\_freq)

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 6 智能課文分析3**

diff\_level = load\_private("./data/tmp/diff\_level")

train\_data = load\_train\_data()

print(len(train\_data))

print(train\_data)

test\_data = load\_test\_data()

print(len(test\_data))

print(test\_data)

features = []

labels = []

def get\_feats\_labels(data, diff\_level):

features = []

labels = []

for path, label in data:

features.append(extract\_features(path, diff\_level))

labels.append(int(label))

return features, labels

train\_feats, train\_labels = get\_feats\_labels(train\_data, diff\_level)

print("train\_feats:", train\_feats)

print(train\_labels)

save\_private([train\_feats, train\_labels], "./data/tmp/train\_features")

test\_feats, test\_labels = get\_feats\_labels(test\_data, diff\_level)

print(test\_feats)

print(test\_labels)

save\_private([test\_feats, test\_labels], "./data/tmp/tests\_features")

**SenseStudy課程平臺“人工智能入門（下）”**

**實驗6 – 7 智能課文分析4**

diff\_level = load\_private("./data/tmp/diff\_level")

train\_data = load\_binary\_train\_data("primary", "junior")

print(len(train\_data))

print(train\_data)

train\_feats, train\_labels = get\_feats\_labels(train\_data, diff\_level)

print(train\_labels)

test\_data = load\_binary\_test\_data("primary", "junior")

test\_feats, test\_labels = get\_feats\_labels(test\_data, diff\_level)

print(test\_labels)

save\_private([train\_feats, train\_labels], "./data/tmp/pri\_jun\_train\_features")

save\_private([test\_feats, test\_labels], "./data/tmp/pri\_jun\_test\_features")

model = linear\_classifier()

model.train(train\_feats, train\_labels)

pred\_y = model.pred(test\_feats)

print(test\_labels)

print(pred\_y)

acc = accuracy(pred\_y, test\_labels)

print(acc)

**SenseStudy課程平臺“人工智能基礎（體驗版）”**

**實驗4 – 1 觀察聲音的波形，理解聲音的數位化**

gtzan = data.get('gtzan')

print(len(gtzan))

music, label = gtzan[10]

fig() + plot(music, type='audio')

print(label)

segment = music.cut(0.76, 0.78)

fig() + plot(segment, type='waveform')

segment\_small = music.cut(0.76, 0.761)

fig() + plot(segment\_small, type='waveform')

print(music.sample\_rate)

**SenseStudy課程平臺“人工智能基礎（體驗版）”**

**實驗4 – 2 用頻譜圖分析樂音的特點**

piano = data.get('piano')

guitar = data.get('guitar')

piano\_seg = piano.cut(t\_start=1.6, t\_end=2.4)

guitar\_seg = guitar.cut(t\_start=1.6, t\_end=2.4)

fig(2,1) + [plot(piano\_seg, type='waveform'), plot(guitar\_seg, type = 'waveform')]

fig(2,1) + [plot(piano\_seg, type='spectrum'), plot(guitar\_seg, type = 'spectrum')]

**SenseStudy課程平臺“人工智能基礎（體驗版）”**

**實驗4 – 3 觀察並理解MFCC特徵**

gtzan = data.get('gtzan')

index = 10

music, label = gtzan[index]

music\_seg1 = music.cut(1.5, 1.53)

music\_seg2 = music.cut(5.5, 5.53)

mfcc = MFCCExtractor()

mfcc\_feature1 = mfcc(music\_seg1)

mfcc\_feature2 = mfcc(music\_seg2)

fig(2,2) + [

plot(music\_seg1, type='spectrum'),

plot(mfcc\_feature1, type='mfcc'),

plot(music\_seg2, type='spectrum'),

plot(mfcc\_feature2, type='mfcc'),

]

**SenseStudy課程平臺“人工智能基礎（體驗版）”**

**實驗4 – 4 使用MFCC特徵和神經網路完成音樂風格分類**

dataset = data.get('gtzan-mfcc')

label\_names = dataset.meta['label\_names']

print(label\_names)

trainset, testset = dataset.split(8,2)

net = CNNClassifier(backbone = AudioNet(), in\_shape=(1, 624, 13), num\_classes=10)

net.train(trainset, alg=SGD(lr=0.001, epoch=5, bs=16))

acc = net.accuracy(testset)

print('Accuracy: ', acc)

index = 20

mfcc\_feature, label = testset[index]

label\_name = label\_names[label]

print('Ground Truth: ', label\_name)

pred = net.predict(mfcc\_feature)

pred\_name = label\_names[pred]

print('Prediction: ', pred\_name)