线性回归是通过历史数据拟合出一条直线，因变量与自变量是线性关系，对新的数据用这条直线进行预测。 线性回归的公式如下：

y=w0+w1x1+...+wnxn=wTx+b

对数几率回归是一种广义的线性回归分析模型，是一种预测分析。它不是仅预测出“类别”， 而是可以得到近似概率预测。对数几率函数是一种“Sigmoid”函数，呈现*S*型曲线，它将z值转化为一个接近0或1的 y值。 对数几率回归公式如下：

y=g(z)=11+e−z, z=wTx+b，

其中，y=11+e−x 被称作**Sigmoid**函数。

Logistic Regression算法是将线性函数的结果映射到了Sigmoid函数中，即y=11+e(wTx+b)。

import csv

csv\_reader = csv.reader(open(r'D:\Tensorflow\bezdekIris.CSV',encoding='utf-8'))

for row in csv\_reader:

print(row)

from sklearn.datasets import load\_iris

import matplotlib.pyplot as plt

iris = load\_iris()

# data对应了样本的4个特征，共150个样本，即150行x4列的矩阵

print("Iris Dataset contains %s samples in total，%s features."%(iris.data.shape[0], iris.data.shape[1]))

iris.data[:5]

print("Labels' shape %s." %(iris.target.shape))

iris.target#target代表150个样本对应的类别label,即150行x1列的矩阵

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

X = iris.data[:, :2] # 取前两列数据

Y = iris.target

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X,Y, test\_size = 0.3, random\_state = 0)

# 画出训练集数据点

trace = go.Scatter(x = X[:,0], y = X[:,1], mode = 'markers',

marker = dict(color = np.random.randn(150),size = 10, colorscale='Viridis',showscale=False))

layout = go.Layout(title = '训练点', xaxis=dict(title='花萼长度 Sepal length', showgrid=False),

yaxis=dict(title='花萼宽度 Sepal width',showgrid=False),

width = 700, height = 380)

fig = go.Figure(data=[trace], layout=layout)

iplot(fig)from sklearn.linear\_model import LogisticRegression

# lr = LogisticRegression(C = 1e5) # C: Inverse of regularization strength

lr = LogisticRegression(penalty='l2',solver='newton-cg',multi\_class='multinomial')

lr.fit(x\_train,y\_train)

#模型评估

print("Logistic Regression模型训练集的准确率：%.3f" %lr.score(x\_train, y\_train))

print("Logistic Regression模型测试集的准确率：%.3f" %lr.score(x\_test, y\_test))

from sklearn import metrics

y\_hat = lr.predict(x\_test)

accuracy = metrics.accuracy\_score(y\_test, y\_hat) #错误率，也就是np.average(y\_test==y\_pred)

print("Logistic Regression模型正确率：%.3f" %accuracy)

target\_names = ['setosa', 'versicolor', 'virginica']

print(metrics.classification\_report(y\_test, y\_hat, target\_names = target\_names))

import pandas as pd

df = pd.read\_csv(r'http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data')

df.tail()y = df.iloc[0:99, 4].values

y

y = df.iloc[0:99, 4].values

y

import numpy as np

import matplotlib.pyplot as plt

y = np.where(y == 'Iris-setosa', -1, 1)

x = df.iloc[0: 99, [0, 2]].values

plt.scatter(x[:49, 0], x[:49, 1], color='red', marker='o', label='setosa')

plt.scatter(x[49:99, 0], x[49: 99, 1], color='blue', marker='x', label='versicolor')

plt.xlabel('petal length')

plt.ylabel('sepal length')

plt.legend(loc='upper left')

plt.show()

我们根据实验结果建立一个简单的分类模型，同时机器学习需要有数据集来进行训练，而训练数据的多少也影响着判别精度，本次实验让我了解到神经网络的大体构成，从而对这门课产生很大的兴趣。而由于本身能力不足，借鉴了部分网络上的代码，此次实验让我认识到自己能力上的差距，今后会更加努力，学好这门课程。