实验1:回归模型

曾令泽 1120162062

## 线性回归模型 Linear Regression

代码：

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Sat Mar 30 19:31:39 2019

@author: e1ixir

"""

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression,Ridge,Lasso,\

LogisticRegression,ElasticNet

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

from sklearn.datasets import fetch\_california\_housing,\

fetch\_20newsgroups\_vectorized

from sklearn.datasets import make\_regression

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

def evalue(model,y\_test,x\_test):

y\_predict=model.predict(x\_test)

mse=mean\_squared\_error(y\_test,y\_predict)

mae=mean\_absolute\_error(y\_test,y\_predict)

score=model.score(x\_test,y\_test)

r2score=r2\_score(y\_test,y\_predict)

print("mse:{} \t mae:{}\t score:{}\t r2\_score:{}\n"\

.format(mse,mae,score,r2score))

# 导入房价数据

data=fetch\_california\_housing()

data\_x=data.data

data\_y=data.target

feature=data.feature\_names

#分割训练数据集

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data\_x,data\_y,test\_size=0.2,random\_state=1)

#定义模型

lr=LinearRegression()

#l\_ridge=Ridge(alpha=1.0)

#l\_lasso=Lasso(alpha=0.1)

#l\_elas=ElasticNet(random\_state=0)

#训练模型（只进行了一次迭代）

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

#l\_ridge.fit(x\_train,y\_train)

#l\_lasso.fit(x\_train,y\_train)

#l\_elas.fit(x\_train,y\_train)

#打印训练结果

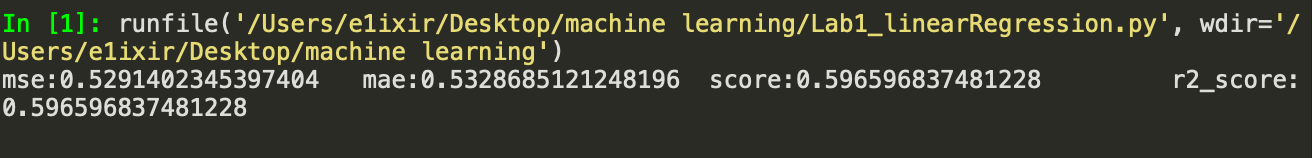
evalue(lr,y\_test,x\_test)

#evalue(l\_ridge,y\_test,x\_test)

#evalue(l\_lasso,y\_test,x\_test)

#evalue(l\_elas,y\_test,x\_test)

实验结果：



## 岭回归模型 Ridge Regression

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Sat Mar 30 19:39:08 2019

@author: e1ixir

"""

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression,Ridge,Lasso,\

LogisticRegression,ElasticNet

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

from sklearn.datasets import fetch\_california\_housing,\

fetch\_20newsgroups\_vectorized

from sklearn.datasets import make\_regression

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

def evalue(model,y\_test,x\_test):

y\_predict=model.predict(x\_test)

mse=mean\_squared\_error(y\_test,y\_predict)

mae=mean\_absolute\_error(y\_test,y\_predict)

score=model.score(x\_test,y\_test)

r2score=r2\_score(y\_test,y\_predict)

print("mse:{} \t mae:{}\t score:{}\t r2\_score:{}\n"\

.format(mse,mae,score,r2score))

# =============================================================================

# 导入房价数据

# =============================================================================

data=fetch\_california\_housing()

data\_x=data.data

data\_y=data.target

feature=data.feature\_names

# =============================================================================

# 分割训练数据集

# =============================================================================

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data\_x,data\_y,test\_size=0.2,random\_state=1)

# =============================================================================

# 定义模型

# =============================================================================

#lr=LinearRegression()

l\_ridge=Ridge(alpha=1.0)

#l\_lasso=Lasso(alpha=0.1)

#l\_elas=ElasticNet(random\_state=0)

# =============================================================================

# 训练模型（只进行了一次迭代）

# =============================================================================

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

l\_ridge.fit(x\_train,y\_train)

#l\_lasso.fit(x\_train,y\_train)

#l\_elas.fit(x\_train,y\_train)

# =============================================================================

# 打印训练测试结果

# =============================================================================

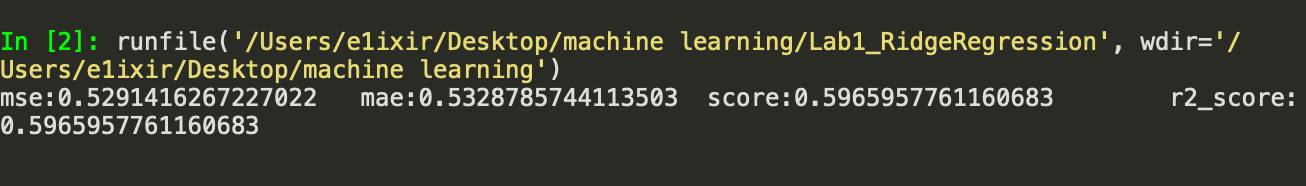
#evalue(lr,y\_test,x\_test)

evalue(l\_ridge,y\_test,x\_test)

#evalue(l\_lasso,y\_test,x\_test)

#evalue(l\_elas,y\_test,x\_test)

实验结果：



## 套索回归模型 Lasso Regression+GridSearch

代码：

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Sat Mar 30 19:42:32 2019

@author: e1ixir

"""

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression,Ridge,Lasso,\

LogisticRegression,ElasticNet

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

from sklearn.datasets import fetch\_california\_housing,\

fetch\_20newsgroups\_vectorized

from sklearn.datasets import make\_regression

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

def evalue(model,y\_test,x\_test):

y\_predict=model.predict(x\_test)

mse=mean\_squared\_error(y\_test,y\_predict)

mae=mean\_absolute\_error(y\_test,y\_predict)

score=model.score(x\_test,y\_test)

r2score=r2\_score(y\_test,y\_predict)

print("mse:{} \t mae:{}\t score:{}\t r2\_score:{}\n"\

.format(mse,mae,score,r2score))

# =============================================================================

# 导入房价数据

# =============================================================================

data=fetch\_california\_housing()

data\_x=data.data

data\_y=data.target

feature=data.feature\_names

# =============================================================================

# 分割训练数据集

# =============================================================================

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data\_x,data\_y,test\_size=0.2,random\_state=1)

# =============================================================================

# 参数调优

# =============================================================================

from sklearn.model\_selection import GridSearchCV

parameter={"alpha":[0.1,0.5,1.0,0.05],"max\_iter":[100,500,1000,50]}

#损失函数中的alpha 选择和最大迭代次数

clf=GridSearchCV(Lasso(),parameter,cv=5)

#5折交叉验证

# =============================================================================

# 训练模型（

# =============================================================================

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

# =============================================================================

# 打印训练测试结果

# =============================================================================

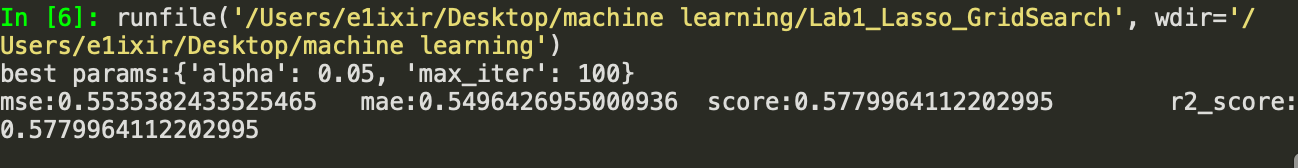
# 自动参数调优结果

print("best params:{}".format(clf.best\_params\_))

#测试结果

evalue(clf,y\_test,x\_test)

实验结果：



## 弹性网回归模型 ElasticNet Regression

代码：

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Sat Mar 30 19:41:30 2019

@author: e1ixir

"""

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression,Ridge,Lasso,\

LogisticRegression,ElasticNet

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

from sklearn.datasets import fetch\_california\_housing,\

fetch\_20newsgroups\_vectorized

from sklearn.datasets import make\_regression

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

def evalue(model,y\_test,x\_test):

y\_predict=model.predict(x\_test)

mse=mean\_squared\_error(y\_test,y\_predict)

mae=mean\_absolute\_error(y\_test,y\_predict)

score=model.score(x\_test,y\_test)

r2score=r2\_score(y\_test,y\_predict)

print("mse:{} \t mae:{}\t score:{}\t r2\_score:{}\n"\

.format(mse,mae,score,r2score))

# =============================================================================

# 导入房价数据

# =============================================================================

data=fetch\_california\_housing()

data\_x=data.data

data\_y=data.target

feature=data.feature\_names

# =============================================================================

# 分割训练数据集

# =============================================================================

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data\_x,data\_y,test\_size=0.2,random\_state=1)

# =============================================================================

# 定义模型

# =============================================================================

#lr=LinearRegression()

#l\_ridge=Ridge(alpha=1.0)

#l\_lasso=Lasso(alpha=0.1)

l\_elas=ElasticNet(random\_state=0)

# =============================================================================

# 训练模型（只进行了一次迭代）

# =============================================================================

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

#l\_ridge.fit(x\_train,y\_train)

#l\_lasso.fit(x\_train,y\_train)

l\_elas.fit(x\_train,y\_train)

# =============================================================================

# 打印训练测试结果

# =============================================================================

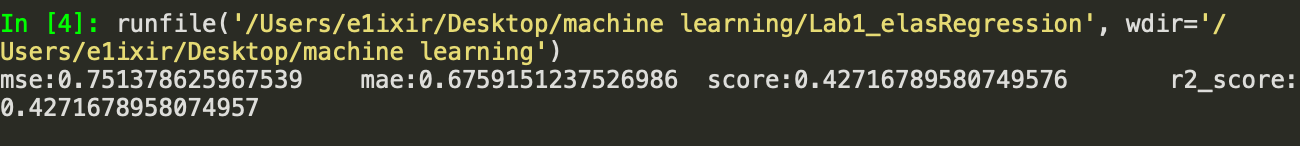
#evalue(lr,y\_test,x\_test)

#evalue(l\_ridge,y\_test,x\_test)

#evalue(l\_lasso,y\_test,x\_test)

evalue(l\_elas,y\_test,x\_test)

实验结果：



## 逻辑回归模型 Logistic Regression

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

"""

Created on Sat Mar 30 19:48:58 2019

@author: e1ixir

"""

import numpy as np

import pandas as pd

from sklearn.linear\_model import LinearRegression,Ridge,Lasso,\

LogisticRegression,ElasticNet

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

from sklearn.datasets import fetch\_california\_housing,\

fetch\_20newsgroups\_vectorized

from sklearn.datasets import make\_regression

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

def evalue(model,y\_test,x\_test):

y\_predict=model.predict(x\_test)

mse=mean\_squared\_error(y\_test,y\_predict)

mae=mean\_absolute\_error(y\_test,y\_predict)

score=model.score(x\_test,y\_test)

r2score=r2\_score(y\_test,y\_predict)

print("mse:{} \t mae:{}\t score:{}\t r2\_score:{}\n"\

.format(mse,mae,score,r2score))

# =============================================================================

# 导入新闻数据

# =============================================================================

data\_logistic=fetch\_20newsgroups\_vectorized()

data\_log\_x=data\_logistic.data

data\_log\_y=data\_logistic.target

# =============================================================================

# 分割训练数据集

# =============================================================================

x\_train,x\_test,y\_train,y\_test=train\_test\_split(data\_log\_x,data\_log\_y,test\_size=0.2,random\_state=1)

# =============================================================================

# 定义模型

# =============================================================================

logist=LogisticRegression()

# =============================================================================

# 训练模型（只进行了一次迭代）

# =============================================================================

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

# =============================================================================

# 打印训练测试结果

# =============================================================================

evalue(logist,y\_test,x\_test)

实验结果：

