大数据人工智能大作业

林灿源

**一、作业要求：**

1、在Kaggle网站上下载一份数据到本地，网址<https://www.kaggle.com/datasets>；

2、针对下载的数据，选择一个sklearn提供的回归算法进行回归模型训练和评估

3、使用模型对数据进行回归预测

4、将预测结果保存成csv文件，结果文件要同时提交到github上。

1. 在Kaggle网站上下载一份数据到本地，

名称：(Bike Sharing Demand)

源文件：test.csv与train.csv

结果生成文件：submission.csv

相关代码：select\_model.py与result.py

1. 针对下载的数据，选择一个sklearn提供的回归算法进行回归模型训练和评估；

代码：select\_model.py

导入包：

*import* numpy *as* np  
*import* pandas *as* pd  
*import* matplotlib.pyplot *as* plt  
*import* seaborn *as* sns  
*from* sklearn *import* model\_selection  
*from* sklearn *import* svm  
*from* sklearn.ensemble *import* RandomForestRegressor

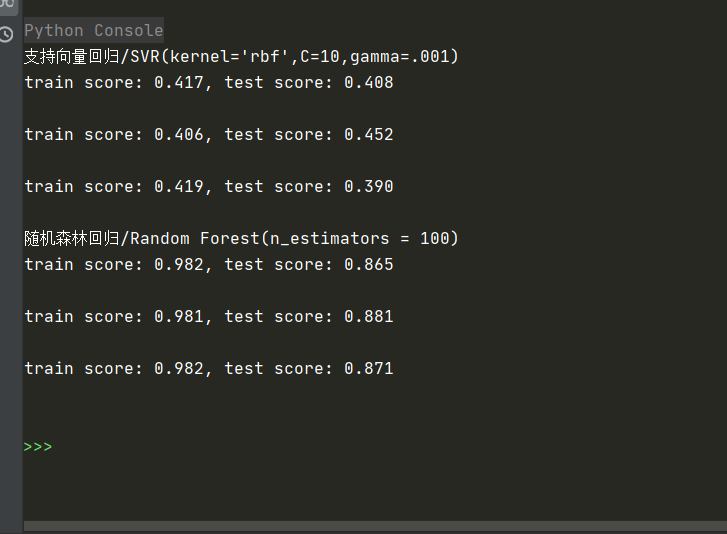
导入数据，划分数据集，处理数据：

f\_train = pd.read\_csv('C:/Users/lincanyuan/Downloads/Big-Data-Competition-Project/02\_共享单车(Bike Sharing Demand)/input/train.csv',header = 0)  
  
df\_train['month'] = pd.DatetimeIndex(df\_train.datetime).month  
df\_train['day'] = pd.DatetimeIndex(df\_train.datetime).dayofweek  
df\_train['hour'] = pd.DatetimeIndex(df\_train.datetime).hour  
  
df\_train = df\_train.drop(['datetime','casual','registered'], axis = 1)  
df\_train\_target = df\_train['count'].values  
df\_train\_data = df\_train.drop(['count'],axis = 1).values

模型比较："支持向量回归与随机森林回归，最后选择随机森林回归

spliter = model\_selection.ShuffleSplit(n\_splits=3, test\_size=0.2, random\_state=0)  
  
  
print("支持向量回归/SVR(kernel='rbf',C=10,gamma=.001)")  
cv = spliter.split(df\_train\_data)  
*for* train, test *in* cv:  
 svc = svm.SVR(kernel='rbf', C=10, gamma=.001).fit(df\_train\_data[train], df\_train\_target[train])  
 print("train score: {0:.3f}, test score: {1:.3f}\n".format(  
 svc.score(df\_train\_data[train], df\_train\_target[train]), svc.score(df\_train\_data[test], df\_train\_target[test])))  
  
print("随机森林回归/Random Forest(n\_estimators = 100)")  
cv = spliter.split(df\_train\_data)  
*for* train, test *in* cv:  
 svc = RandomForestRegressor(n\_estimators=100).fit(df\_train\_data[train], df\_train\_target[train])  
 print("train score: {0:.3f}, test score: {1:.3f}\n".format(  
 svc.score(df\_train\_data[train], df\_train\_target[train]), svc.score(df\_train\_data[test], df\_train\_target[test])))

比较结果：最后选择随机森林回归



1. 使用模型对数据进行回归预测

代码：result.py

# 特征值处理

X\_train\_df['month'] = pd.DatetimeIndex(X\_train\_df.datetime).month

X\_train\_df['day'] = pd.DatetimeIndex(X\_train\_df.datetime).dayofweek

X\_train\_df['hour'] = pd.DatetimeIndex(X\_train\_df.datetime).hour

X\_train\_df = X\_train\_df.drop(['datetime'], axis=1) # axis=1意思为每行执行，axis=0表示每列执行

X\_test\_df['month'] = pd.DatetimeIndex(X\_test\_df.datetime).month

X\_test\_df['day'] = pd.DatetimeIndex(X\_test\_df.datetime).dayofweek

X\_test\_df['hour'] = pd.DatetimeIndex(X\_test\_df.datetime).hour

X\_test\_df = X\_test\_df.drop(['datetime'], axis=1)

# 采用DictVectorizer进行特征向量化

from sklearn.feature\_extraction import DictVectorizer

dict\_vec = DictVectorizer(sparse=False)

X\_train\_df = dict\_vec.fit\_transform(X\_train\_df.to\_dict(orient='record'))

X\_test\_df = dict\_vec.transform(X\_test\_df.to\_dict(orient='record'))

# 使用RandomForestRegressor进行回归预测

# from sklearn.ensemble import GradientBoostingRegressor

from sklearn.ensemble import RandomForestRegressor

gbr = RandomForestRegressor()

gbr.fit(X\_train\_df, y\_train\_df)

gbr\_y\_predict = gbr.predict(X\_test\_df)

# 输出结果

gbr\_submission = pd.DataFrame({'datetime': test\_df['datetime'], 'count': gbr\_y\_predict})

gbr\_submission.to\_csv('submission.csv', index=False)

4、将预测结果保存成csv文件，结果文件要同时提交到github上。

生成结果：

