参赛人员：程铭、张舒琳、刘喆、孙钰曦

疫情对北京社会经济影响分析

1. **赛题分析及问题需求**

1.1大赛简介

新型冠状病毒（COVID-19）感染的肺炎疫情牵动着全国人民的心，全国同舟共济、众志成城，打响了一场没有硝烟的疫情阻击战。习近平指出：要鼓励运用大数据、人工智能、云计算等数字技术，在疫情监测分析、病毒溯源、防控救治、资源调配等方面更好发挥支撑作用。

新型冠状病毒感染肺炎疫情迅速向全国蔓延，举国上下共同抗击疫情，为避免人口大规模流动和聚集，采取了居家隔离、延长春节假期等防控措施。北京作为全国中心城市，社会经济等多方面的影响极具代表性。通过对疫情期间各方面数据分析，研判对中小企业、商业服务业、文化旅游业等影响，为社会经济恢复政策提供决策参考。

1.2 问题需求

针对本次的比赛，我们组从各类商品的销售总值出发，根据疫情的情况并结合历史上各个月份的销售总值进行分析，通过收集的数据进行分析、计算，预测出后面各类商品可能的销售总值，以此为基础为后面的商品销售的调整提供一定的参照，以达到可以更好的补齐缺漏产品并减少部分产品的浪费情况。

1. **数据使用**

2.1 数据使用

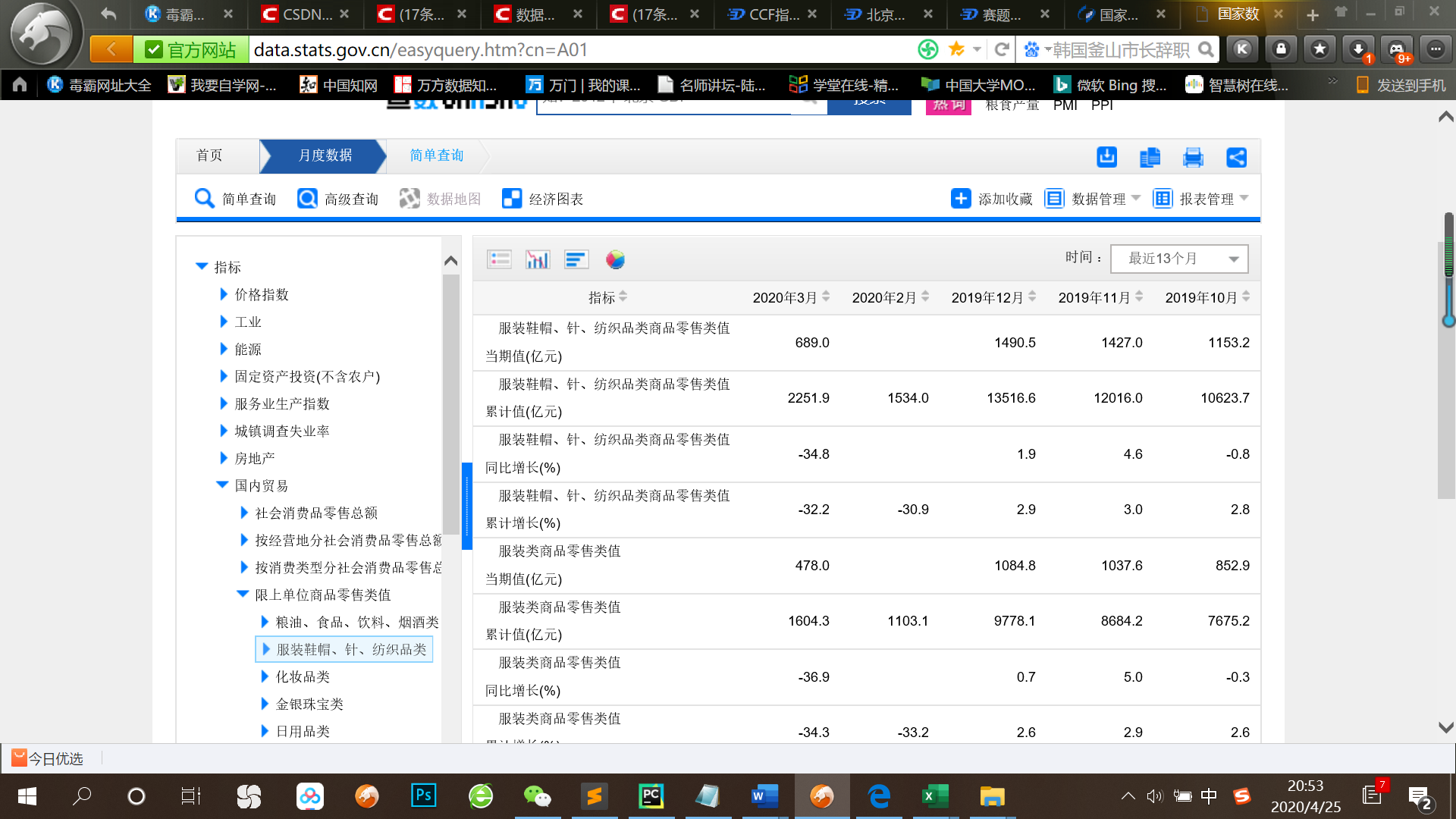
数据使用主要包括以下两个部分：

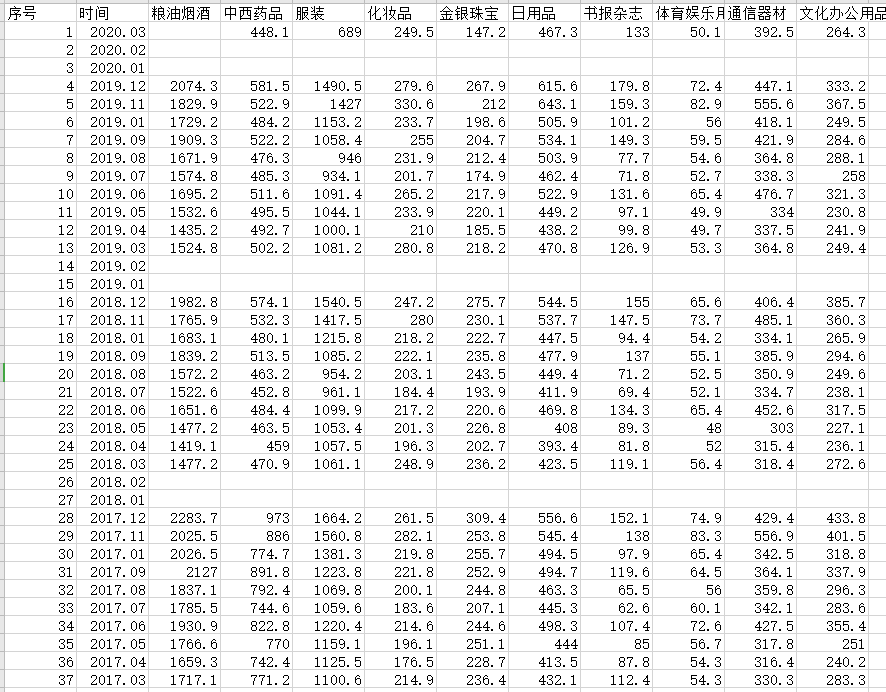
1. 北京疫情情况



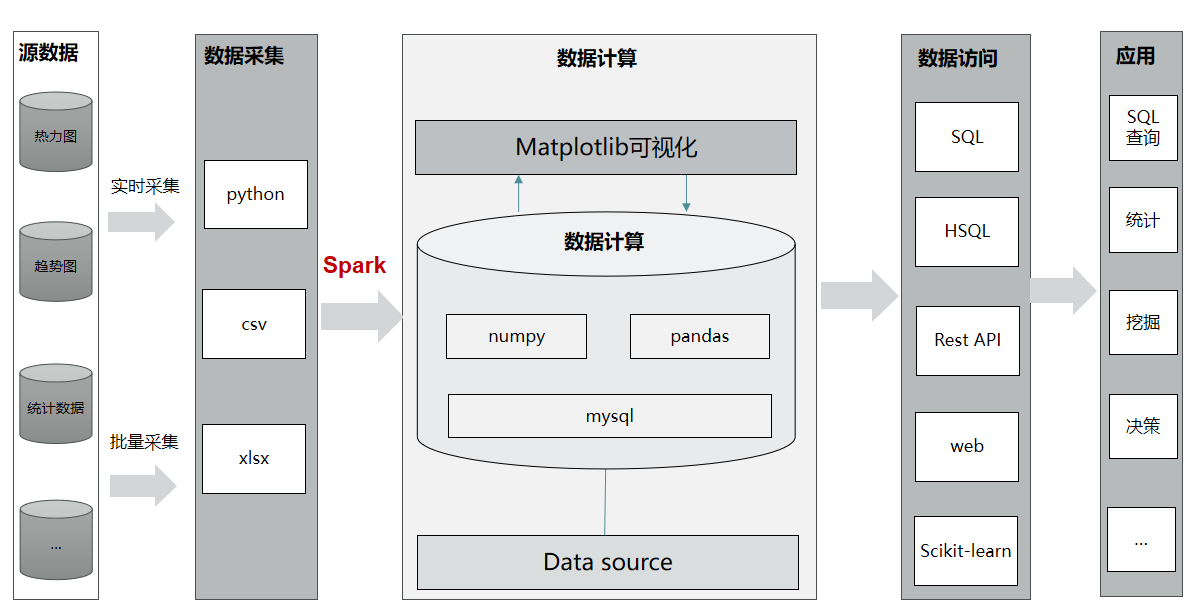
1. 各类商品零售总值

本文主要选取了国家统计局网站（http://data.stats.gov.cn/easyquery.htm?cn=A01）的零售商品当期值作为模型的训练对象





1. **技术架构**



1. **技术细节**

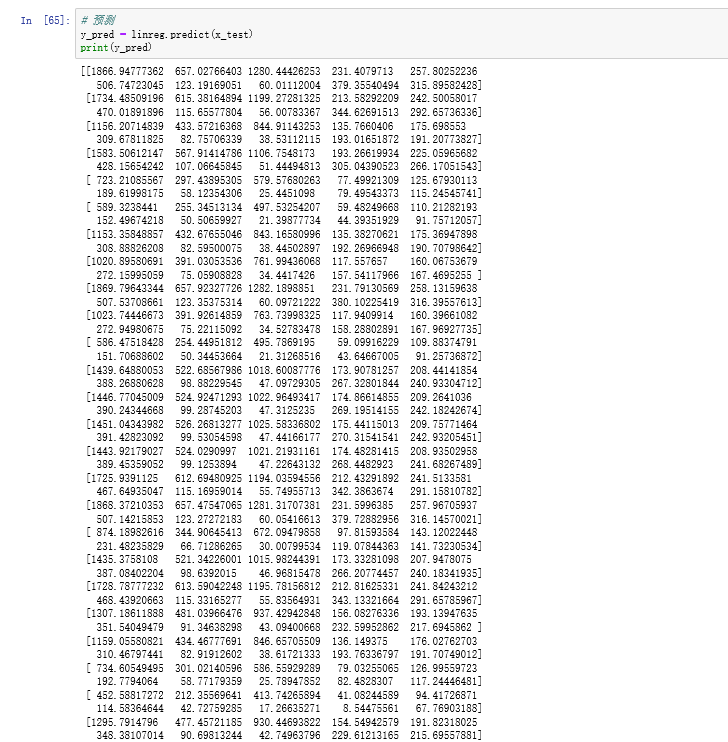
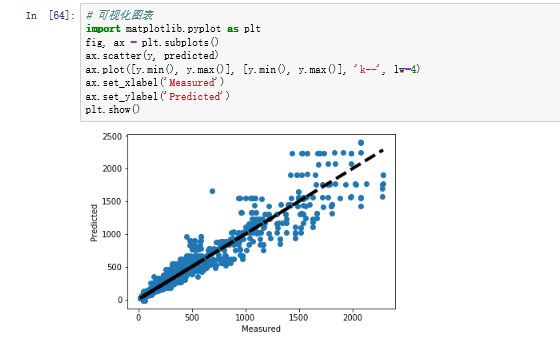
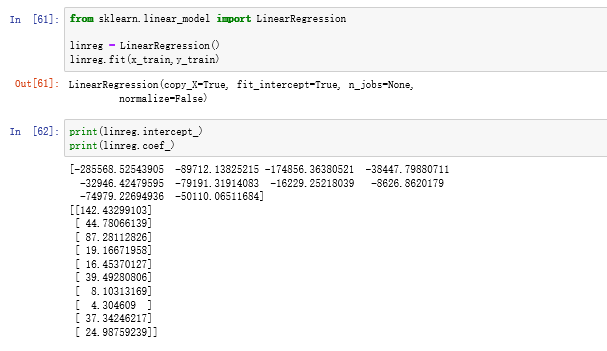
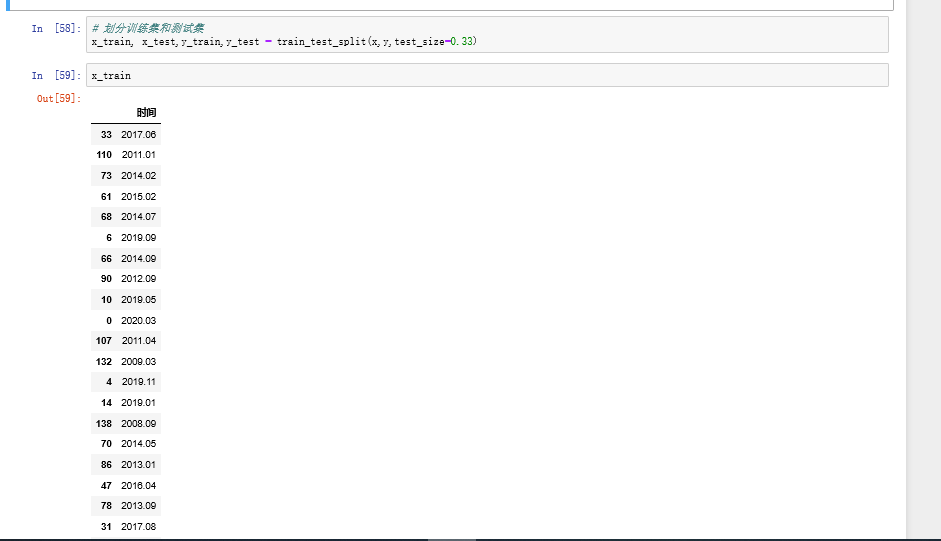
4.1爬虫

import requests  
import re  
import json  
from pyquery import PyQuery  
import pymysql  
  
# 数据库连接  
def connect():  
 conn = pymysql.connect(host='localhost',  
 port=3306,  
 user='root',  
 password='password',  
 database='test',  
 charset='utf8mb4')  
  
 # 获取操作游标  
 cursor = conn.cursor()  
 return {"conn": conn, "cursor": cursor}  
  
connection = connect()  
conn, cursor = connection['conn'], connection['cursor']  
  
sql\_insert = "insert into stock(code, name, jinkai, chengjiaoliang, zhenfu, zuigao, chengjiaoe, huanshou, zuidi, zuoshou, liutongshizhi, create\_date) values (%(code)s, %(name)s, %(jinkai)s, %(chengjiaoliang)s, %(zhenfu)s, %(zuigao)s, %(chengjiaoe)s, %(huanshou)s, %(zuidi)s, %(zuoshou)s, %(liutongshizhi)s, now())"  
  
headers = {  
 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36'  
}  
  
def get\_stock\_list(stockListURL):  
 r =requests.get(stockListURL, headers = headers)  
 doc = PyQuery(r.text)  
 list = []  
 # 获取所有 section 中 a 节点，并进行迭代  
 for i in doc('.stockTable a').items():  
 try:  
 href = i.attr.href  
 list.append(re.findall(r"\d{6}", href)[0])  
 except:  
 continue  
 list = [item.lower() for item in list] # 将爬取信息转换小写  
 return list  
  
  
def getStockInfo(list, stockInfoURL):  
 count = 0  
 for stock in list:  
 try:  
 url = stockInfoURL + stock  
 r = requests.get(url, headers=headers)  
 # 将获取到的数据封装进字典  
 dict1 = json.loads(r.text[14: int(len(r.text)) - 1])  
 print(dict1)  
  
 # 获取字典中的数据构建写入数据模版  
 insert\_data = {  
 "code": stock,  
 "name": dict1['info'][stock]['name'],  
 "jinkai": dict1['data'][stock]['7'],  
 "chengjiaoliang": dict1['data'][stock]['13'],  
 "zhenfu": dict1['data'][stock]['526792'],  
 "zuigao": dict1['data'][stock]['8'],  
 "chengjiaoe": dict1['data'][stock]['19'],  
 "huanshou": dict1['data'][stock]['1968584'],  
 "zuidi": dict1['data'][stock]['9'],  
 "zuoshou": dict1['data'][stock]['6'],  
 "liutongshizhi": dict1['data'][stock]['3475914']  
 }  
 cursor.execute(sql\_insert, insert\_data)  
 conn.commit()  
 print(stock, '：写入完成')  
 except:  
 print('写入异常')  
 # 遇到错误继续循环  
 continue  
def main():  
 stock\_list\_url = 'https://hq.gucheng.com/gpdmylb.html'  
 stock\_info\_url = 'http://qd.10jqka.com.cn/quote.php?cate=real&type=stock&callback=showStockDate&return=json&code='  
 list = get\_stock\_list(stock\_list\_url)  
 # list = ['601766']  
 getStockInfo(list, stock\_info\_url)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

4.2xgboost

from sklearn.neighbors import KNeighborsClassifier  
#%%  
import numpy as np  
  
import pandas as pd  
  
from sklearn.model\_selection import train\_test\_split  
  
xls = pd.ExcelFile(r'统计.xlsx')  
data=pd.read\_excel(xls,1)  
data  
# print(data)  
  
data.drop\_duplicates()  
data = data.fillna(method='bfill',axis=0).fillna(method='ffill',axis=0)  
x=data[["时间"]]  
print(x.shape)  
y=data[["粮油烟酒","中西药品","服装","化妆品","金银珠宝","日用品","书报杂志","体育娱乐用品","通信器材","文化办公用品"]]  
print(y.shape)  
#%%  
# 划分训练集和测试集  
x\_train, x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.33)  
#%%  
x\_train  
#%%  
x\_test  
#%%  
from sklearn.linear\_model import LinearRegression  
  
linreg = LinearRegression()  
linreg.fit(x\_train,y\_train)  
#%%  
print(linreg.intercept\_)  
print(linreg.coef\_)  
#%%  
y\_pred = linreg.predict(x\_test)  
from sklearn import metrics  
from sklearn.model\_selection import cross\_val\_predict  
predicted = cross\_val\_predict(linreg,x,y,cv=8) # cv可以调  
print("MSE:",metrics.mean\_squared\_error(y, predicted))  
print("RMSE:",np.sqrt(metrics.mean\_squared\_error(y, predicted)))  
#%%  
# 可视化图表  
import matplotlib.pyplot as plt  
fig, ax = plt.subplots()  
ax.scatter(y, predicted)  
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)  
ax.set\_xlabel('Measured')  
ax.set\_ylabel('Predicted')  
plt.show()  
#%%  
# 预测  
y\_pred = linreg.predict(x\_test)  
print(y\_pred)   
#%%  
x\_test  
#%%

1. **原型测试**



1. **价值导向**

本次作品旨在通过计算，较为合理的预测出各类商品的总值，以此来给北京的商品销售调整提供一定的数据支撑，