

深圳大学考试答题纸

(以论文、报告等形式考核专用)

二〇二二 ~ 二〇二四 学年度第 一 学期

课程编号	课程名称	GIS 网络开发	主讲教师	夏吉喆	评分
学 号	2021104037	姓 名	姜雨轩	专业年级	21 级地理空间信息工程一班

教师评语：

题目： spacElite——地理可视化数据分析平台

GISWeb开发 SpacELite项目

1. 项目概述:

本项目名称为SpacELite,由Space(空间),Elite(杰出),Lite(轻量)三个单词组合而成。项目成员由我(娄雨轩 2021104037)组成。本项目基于`vue`框架意在构建一个轻量级的优秀的空间统计分析平台(尽管就最后的实现来看并不优秀),目前已经实现大部分前端界面内容和后端算法内容,遗憾的是由于对`vue`框架仍不够熟悉以及单人作业的时间精力问题,在最后的算法分析结果渲染上出现了一些问题,导致无法渲染出最终图标,至今仍未解决该问题。本项目实现了包括: *Kmeans, DBSCAN, MeanShift, GaussianMixture*等聚类算法, *KNN, SVC, DecisionTree, 逻辑回归*等分类算法, *线性回归, SVR, 随机森林*等回归算法, 以及*地理加权回归, MoransI计算, 局部MoransI计算, 局部G指数计算, 局部JointCount和OD矩阵的计算*等空间分析算法。本项目前端界面负责数据的接收和算法参数的选取等, 将算法执行所需要的数据以json格式发送到后端(由python编写), 由python执行计算后将计算结果包装成json格式发回web端, web端根据发回的不同信息进行渲染。除了部分关于代码框架组织和css样式设计是经过咨询github copilot ai助手以外, 其余代码(包括具体的模板, 前后端信息交互, 大部分样式表, python算法代码)均为自己编写和自己在往日的积累中记录下的算法的代码, 因此在项目源代码中不特别注明哪一块为ai辅助完成。

2. 项目文件结构:

SPACELITE ---- node_modules

```
|---- public ---- planet.svg
|
|          | ---- vite.svg
| ---- src ---- assets ---- 图标资源...
|          | ---- common ---- particles.js
|          |          |---- style.css
|          | ---- components ---- Core ---- Css ---- start.css
|          |          |          | ---- DropZoneX ---- DropZoneX.css
|          |          |          |          | ---- DropZoneX.vue
|          |          |          |          | ---- DropZoneY ---- DropZoneY.css
|          |          |          |          | -----
DropZoneY.vue
|          |          |          | ---- MethodsList ---- assets ---
-.
|          |          |          | ---- methodsList.css
|          |          |          | ----
```

```

methodsList.vue
|           |           |           | ---- MiddleBlock ---- ~.css
|           |           |           |           | ----
middleBlock.vue
|           |           |           | ---- paramsBlock ---- ~.css
|           |           |           |           | ----
paramsBlock.vue
|           |           |           |           | ---- selections
|           |           |           |           |           |
---- .....
|           |           |           |
..... 各算法参数区域组件
|           |           |           | ---- ResultBlock ---- ~.css
|           |           |           |           | -----
ResultBlock.vue
|           |           |           | ---- route
|           |           |           | ---- script
|           |           |           | ---- core.vue
|           |           |           | ---- Logo ---- Css ---- CenterLogo.css
|           |           |           |           | ---- Logo.css
|           |           |           |           | ---- Center.vue
|           |           |           |           | ---- logo.vue
|           |           |           | ---- Navigation ---- assets
|           |           |           |           | ---- Css ---- navigatorCSS.css
|           |           |           |           | ---- navigator.vue
|           |           |           | ---- route ---- CSS ---- contact.css
|           |           |           |           | ---- uploadfile.css
|           |           |           |           | ---- contact.vue
|           |           |           |           | ---- help.vue
|           |           |           |           | ---- upload.vue
|           |           | ---- CSS ---- mainAPP.css
|           | ---- App.vue
|           | ---- i18n.js
|           | ---- main.js
|           | ---- algorithm.py
| ---- index.html
| ---- app.py

```

3. 前端页面功能介绍



主页主要由以下部分构成：

- 导航栏 红色框内（src-components-Navigation.vue）
- 大Logo 黄色框内（src-components-Logo-Center.vue）
- 小Logo 蓝色框内（src-components-Logo-logo.vue）
- 以及背景颜色（src-CSS-mainAPP.css），背景粒子特效(src-common-particles.js)

在导航栏中分几个界面：首页，联系，帮助手册，上传文件，进入分析，以及中英文切换按键。我是通过切换路由选项来实现页面切换的，这样做的好处是页面初次加载时就会将所有路由加载进来，切换时不会出现卡顿，并且可以设置路由守卫(route guard)，此处来确保在点击进入分析前需要先上传文件。首先在main.js中注册组件路由，i18n(i18n.js中设置了中英文切换的对照表)并且初始化某些全局变量（fileData, hasUploadedFile, responseData）

main.js

```
import { createApp } from 'vue'
import './common/style.css'
import { createRouter, createWebHistory } from 'vue-router'
import App from './App.vue'
import Contact from './components/route/contact.vue'
import Help from './components/route/help.vue'
import Center from './components/Logo/center.vue'
import Core from './components/Core/core.vue'
import Uploadfile from './components/route/upload.vue'
import ParamsZone from './components/Core/paramsZone/paramsZone.vue'
import i18n from './i18n' // 引入 i18n
import Vuex from 'vuex'
```

```

const router = createRouter({
  history: createWebHistory(),
  routes: [
    { path: '/', redirect: '/center' },
    { path: '/center', name: 'center', component: Center },
    { path: '/contact', component: Contact },
    { path: '/help', component: Help },
    { path: '/core', component: Core, beforeEnter: (to, from, next) => {
      // 检查用户是否已经上传了文件
      if (!store.state.hasUploadedFile) {
        // 如果用户没有上传文件，先提示未上传文件，然后重定向到 'uploadfile'
        alert("请先上传文件 Please Upload File before Start !")
        next('/uploadfile');
      } else {
        // 如果用户已经上传了文件，则允许导航
        next();
      }
    }
  ]
})

const store = new Vuex.Store({
  state: {
    fileData: null,
    hasUploadedFile: false,
    responseData: null,
  },
  mutations: {
    setfileData(state, data) {
      state.fileData = data;
    },
    setHasUploadedFile(state, hasUploadedFile) {
      state.hasUploadedFile = hasUploadedFile;
    },
    setResponseData(state, data) {
      state.responseData = data;
    }
  }
});

createApp(App)
  .use(i18n)
  .use(store)
  .use(router)
  .mount('#app')

```

在app.vue中设置路由路径，背景样式，初始化背景粒子。

app.vue

```
<script>
import Navigator from './components/Navigation/navigator.vue'
import Center from './components/Logo/center.vue'
import Core from './components/Core/core.vue'
import createParticles from './common/particles.js'
import Uploadfile from './components/route/upload.vue'

export default {
  components: {
    Navigator,
    Center,
    Core,
    Uploadfile,
  },
  mounted() {
    createParticles('backgroundcanvas')
  },
  computed: {
    className() {
      return {
        'home': this.$route.path === '/',
        'contact': this.$route.path === '/contact',
        'help': this.$route.path === '/help',
        'core': this.$route.path === '/core',
        'uploadfile': this.$route.path === '/uploadfile',
      };
    }
  },
}
</script>
<style scoped>
@import './CSS/mainApp.css';
</style>
<template>
  <div class="app" :class="className">
    <Navigator />
    <canvas id = 'backgroundcanvas'></canvas>
    <RouterView/>
  </div>
</template>
```



在联系界面中 可以点击头像右侧三个图标进入不同的网页，依次是（微信二维码，github个人主页，csdn个人主页）



在帮助界面中，写了一些帮助使用的信息



在上传文件的界面中，给出了上传表格文件内数据排列的示例，支持点击上传或拖拽上传(不超过10MB，超过10MB会导致网页瘫痪)

导入后在下方会出现预览



如果未上传文件直接进入分析则会强制跳转到上传文件页面



分析界面分为四大块，从左往右依次为：算法选择栏，字段选择栏，参数选择栏和结果栏。在算法选择栏中可以通过搜索算法选取某个算法，选取的算法会在搜索栏下方显示，然后在参数选择栏中会出现对应的参数选择框，将需要分析的变量拖入对应的框，点击apply即可进行分析



前后端交互原理

一般来说，由前端发出的json包含以下信息。

```
{  
  request: ..  
  Xs: ..  
  Ys: ..  
}
```

```

        param1:...
        param2:...
        ...
    }

```

`request` 字段规定了此次请求所需要调用的算法, `Xs`和`Ys`是需要用到的数据, 以及算法可以选择的参数`params`, 在前端包装好信息后, 发送给服务端(python端)。

```

@app.route('/data', methods=['POST'])
def receive_data():
    data = request.get_json()
    requestFunc = data['request']
    res = eval(requestFunc)(data)
    return jsonify(res)

```

python端直接解析`request`字段内容, 并把数据`data`转发给`request`的同名函数, 我将所有函数的命名都命名为`request`的所有可能的请求名称, 这样设计避免了判断不同`request`而需要调用不同的对应的函数, 减少了代码量, 然后我们将结果以`json`的格式返回出去。返回的`json`内容包括需要可视化的图标的类型: `type`字段, 横纵轴数据`Xs`, `Ys`, 执行算法的类型`event`, 横纵坐标标签`Xname`, `Yname`, 颜色`Color`, 以及算法可能有的假设检验或其他检验的总结`Summery`, 算法的具体实现在 `algorithm.py` 文件中。其中某一个算法实现如下:

app.py

```

def RandomForest(data: dict):
    """
    data = {
        Xs:
        Ys:
        criterion: str
    }
    """
    Xs = data['Xs']
    Ys = data['Ys']
    criterion = data['criterion']
    return jsonify({
        'type': 'scatter',
        'event': 'regression',
        'Xs': Xs,
        'Ys': algo.RandomForestRegression(Xs, Ys, Xs, criterion),
        'Xname': data['Xname'],
        'Yname': data['Yname'],
    })

```

```

    'Colors': 0,
    'Summery': 0,
  })

```

发回到前端后进行解析并渲染（未充分完成这部分内容）

```
<template>
  <div class="resultsZone">
    <div v-if="responseData">
      <div ref="chart"></div>
    </div>
    <div v-if="Summery">{{ Summery }}</div>
  </div>
</template>

<script>
import * as echarts from "echarts";
import { mapState } from 'vuex';
export default {
  data() {
    return {
      chartInstance: null,
    };
  },
  computed: {
    ...mapState(['responseData']),
  },
  watch: {
    responseData: {
      handler(newData) {
        this.updateChart();
      },
      deep: true,
    },
  },
  mounted() {
    this.$nextTick(() => {
      this.chartInstance = echarts.init(this.$refs.chart);
      this.updateChart();
    });
  },
  methods: {
    updateChart() {
      if (this.responseData) {
        const { type, Xs, Ys, Yname } = this.responseData;
```


```
const chartTypes = {
  bar: {
    title: {
      text: this.type,
    },
    xAxis: {
      data: this.Xs
    },
    yAxis: {},
    series: [
      {
        name: this.Yname,
        type: 'bar',
        data: this.Ys,
      }
    ]
  },
  scatter: {
    // 配置项...
    title: {
      text: this.type,
    },
    xAxis: {
      type: 'value',
      scale: true,
      data: this.Xs,
    },
    yAxis: {
      type: 'value',
      scale: true,
    },
    series: [
      {
        name: this.Yname,
        type: 'scatter',
        data: this.Ys,
      }
    ]
  },
  polyline: {
    title: {
      text: this.type,
    },
    xAxis: {
      type: 'value',
```

```

        scale: true,
        data: this.Xs,
    },
    yAxis: {
        type: 'value',
        scale: true,
    },
    series: [
        {
            name: this.Yname,
            type: 'line',
            data: this.Ys,
        }
    ]
},
};
const option = chartTypes[this.responseData.type];
if (option) {
    this.chartInstance.setOption(option);
}
}
},
},
};
</script>

```

4. 其他说明

在dist文件夹中有index.html，但是需要在vscode中用`Open with Live Server`打开  其余文件夹均为项目源码

5. 总结

选择独自一个人完成GISWeb开发任务是源于希望掌握全套web开发技术的热情，虽然最后还差一块拼图没有完成，但是对于web开发的主要流程和当下比较热门的web框架我已经稍微入门了。虽然一个人完成很累，期间面临着无人交流，无人可以寻求帮助，无人可以分担任务的困难，但是我还是克服了，凭借着现在强大的AI以及搜索引擎我解决了一个又一个的问题，虽然没有完完全全完成这个项目，但是我仍然因为我所坚持下来学习到的而感到很满足。并且除此之外，我还学习到了很多其他的知识，例如使用html写ppt (slidev)，使用html写笔记（包括此篇报告）等等，我充分领略到html这门语言的魅力。