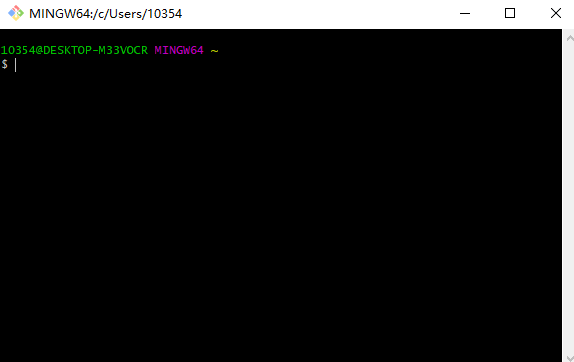
**友情提示：**

1. **本文中涉及的所有操作，请在shadowsock（以下简称ss）处于PAC模式的状态下进行。**
2. 由于安装sourcetree与版本更新等操作较为复杂，中间会遇到各种各样的问题，并且大家的电脑环境差别较大，本教程并不能给出所有情况的解决办法，请谅解。有问题及时沟通。

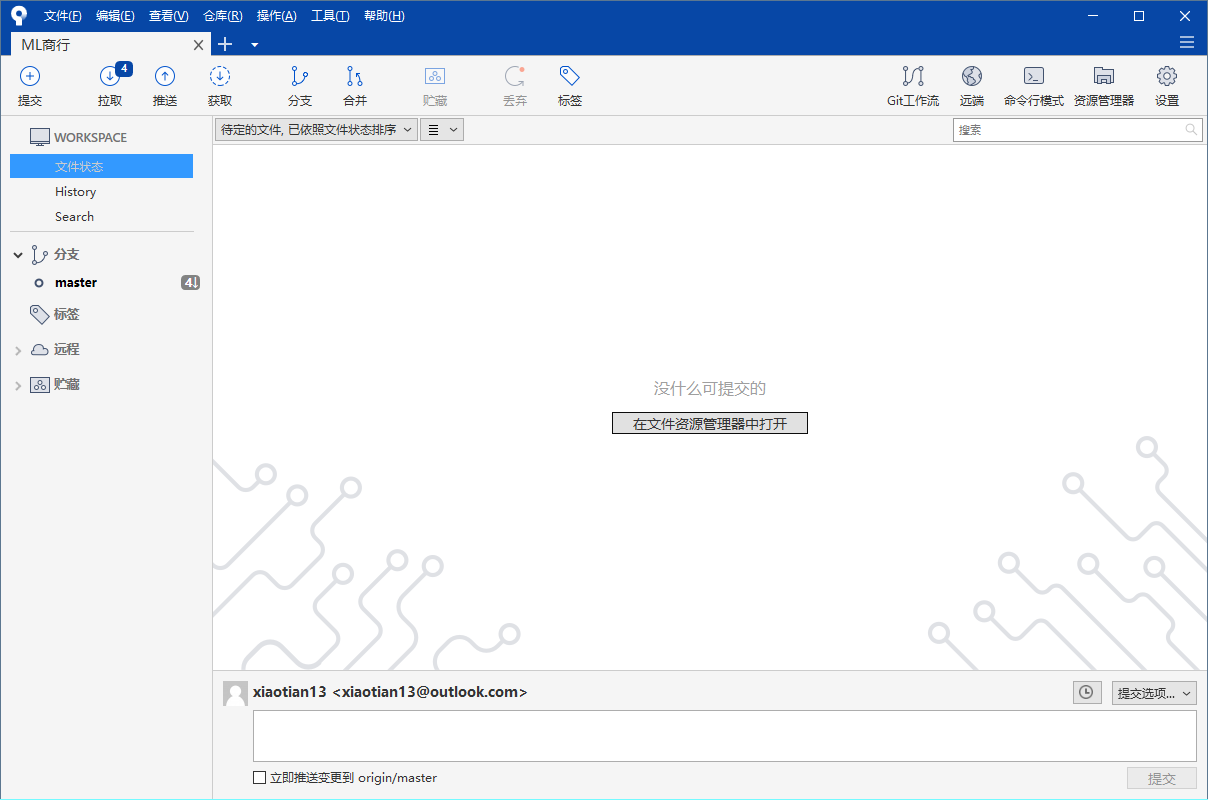
**一、Sourcetree、Git简介**

Git是一个版本控制的工具。所谓的版本控制，实质上是把每一个版本都存下来，只不过我们自己在操作时容易手忙脚乱，记不清修改了哪里或者每次修改文件后并不会做出记号来记录这种改动，因此将版本控制这件事情交给专业的软件去做。

Git bash的操作界面：



就是上面这个黑乎乎的东西，它与命令行(cmd)长得一模一样。显然，这样的界面对初学者/用户而言并不友好。sourcetree就是做了这样一件事情，即将所有需要在命令行中进行的操作，全部打包至一个软件中，方便使用。下面是sourcetree的界面。



**二、Sourcetree、Git的安装**

上面已经提到，Sourcetree是对git的一个包装，本质上的操作工具依然是git，因此若要安装sourcetree，必然也要安装git。

**首先安装git，git的安装包已经给出。**Git的安装较为简单，请自行参考以下两篇文章。安装过程中会遇到几个要打勾的地方，具体勾哪个对我们后续的操作影响不大，请自行参考链接判断。

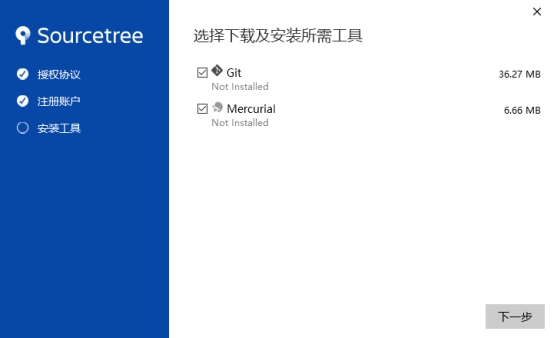
<https://www.cnblogs.com/yelanggu/p/11004724.html>

<https://www.cnblogs.com/wj-1314/p/7993819.html>

注意，**安装路径如果没做修改，即安装到了默认路径，一定要记得装到了哪里**。

**接下来安装sourcetree，安装包同样已经给出。**由于我已经安装过了sourcetree，这里无法给出详细的图片。安装分为三步，分别需要注意的地方如下：

1. 第一步中，软件要求注册bitbucket账号，给出了两个可选的按钮。由于大家都没有注册，因此**点的是右边那个选项**，之后会弹出页面，自行注册一个账号即可。**再次提醒，一定要在ss处于PAC的情况下进行操作**，不然会遇到网络问题。注册过账号、验证邮箱后，安装界面处会自动跳转至第二步，无需其他操作。
2. 第二步中，会提示安装git、Mercurial，类似这样的界面：



不过我们前面已经安装过了Git，这里可能不会提示再安装git，可能显示已检测到哪里哪里安装的git；或者显示



所以这里要点第二项，在你的系统里找到你刚刚安装的git的路径，把git添加进去。至于另一个Mercurial，可以选择不安装，因为它与我们要用到的git无关；也可以选择安装，但安装过程中可能会出现卡住/安装失败的情况，关闭安装界面再重新安装即可，重复几次后会安装成功。

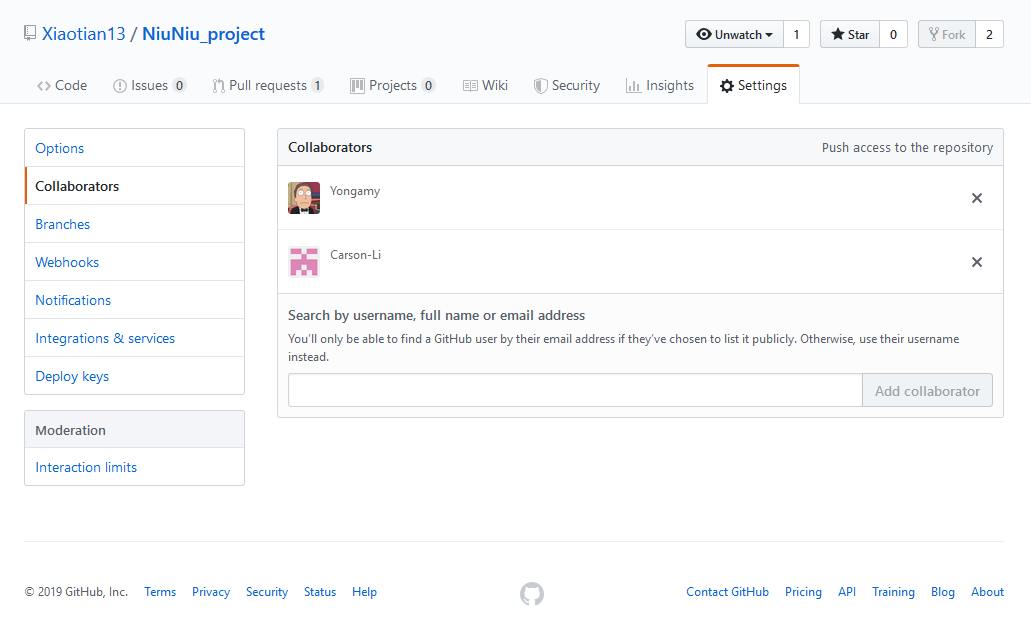
**三、Sourcetree、Git的使用**

至此，我们已经完成了环境的配置。下面介绍如何使用软件进行项目多人协作。

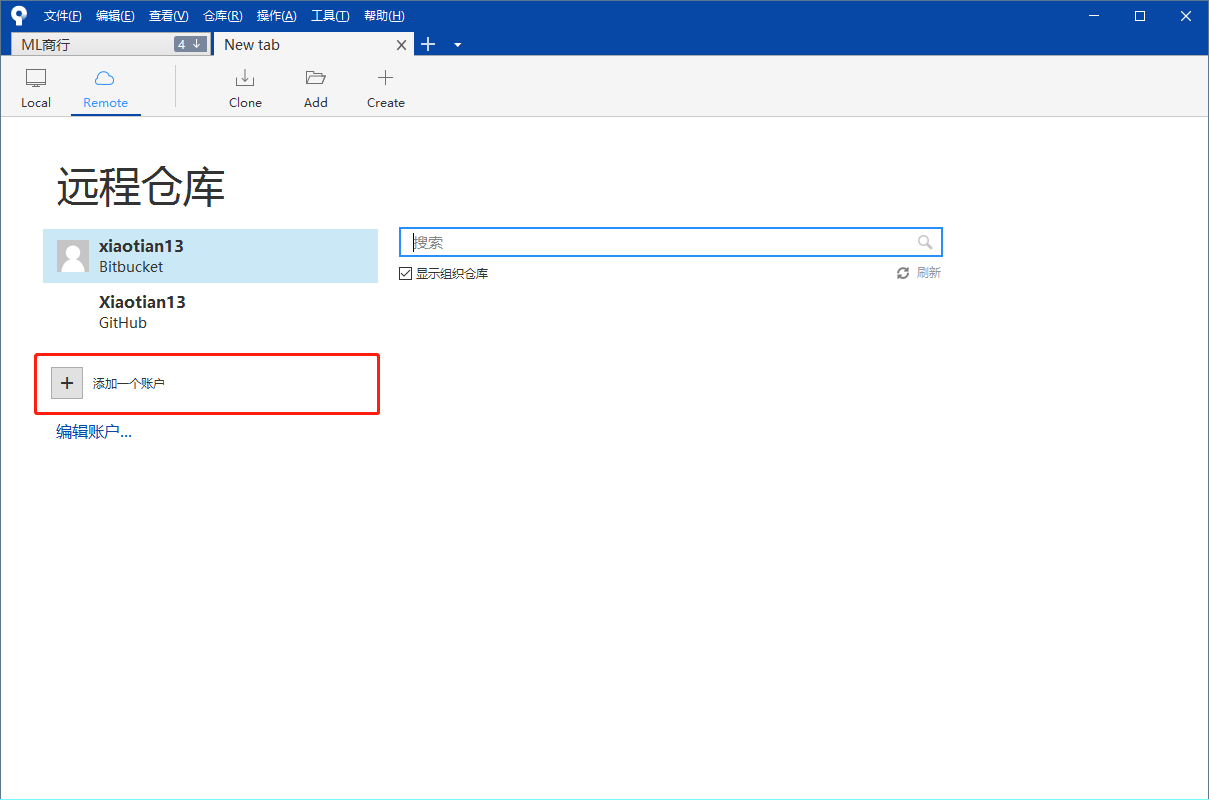
**首先，去注册一个github账号**，网站如下。注册过程较简单，此处略去。

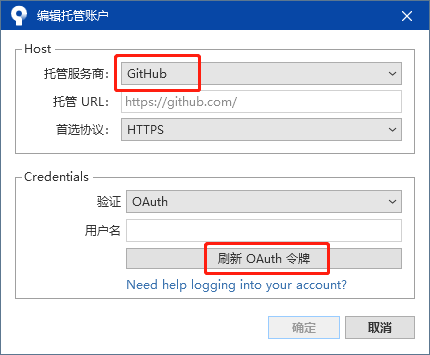
<https://github.com/>

注册完毕后，**将你的用户名微信私发给李啸天，**如xiaotian13（李啸天）、Yongamy（仲兆戎）、Carson-Li（李金苗）。我会把你添加为项目的协作者，如下图。

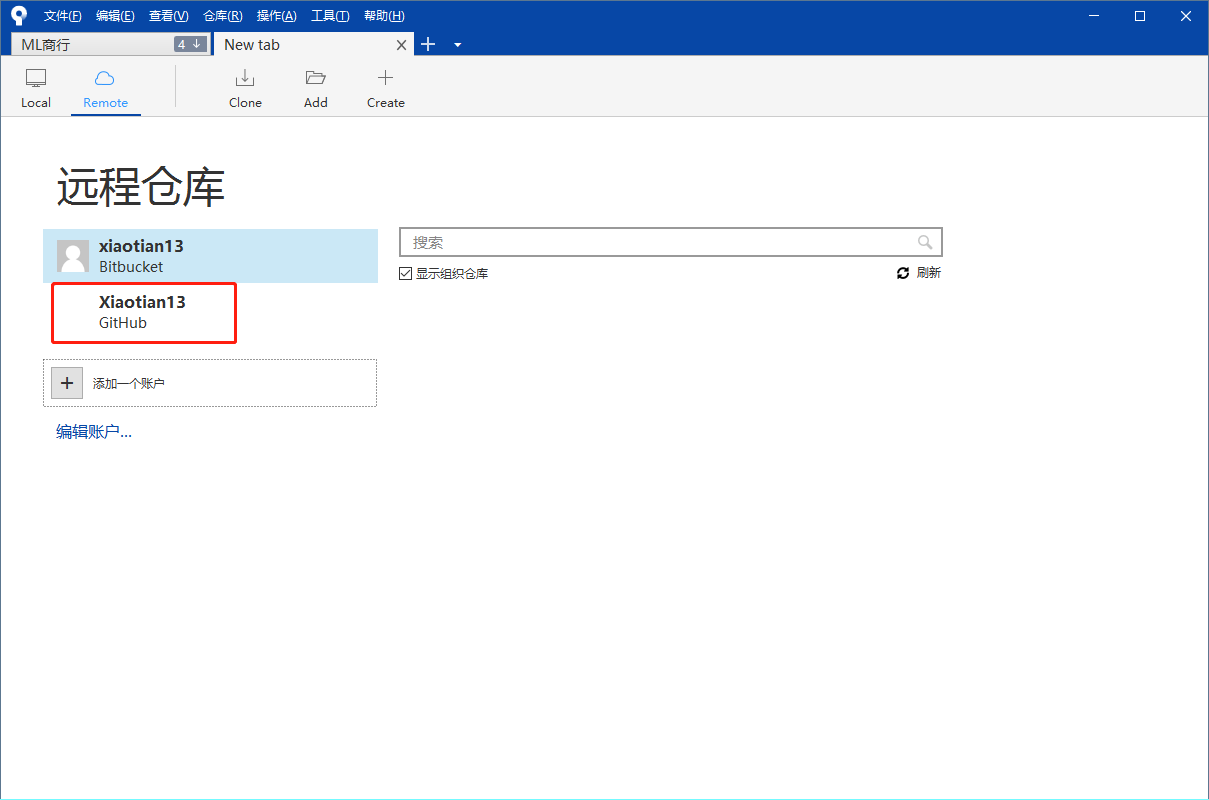


其次，**双击打开sourcetree软件**。下图应当是你们首先看到的界面。**点击“添加一个账户”。**

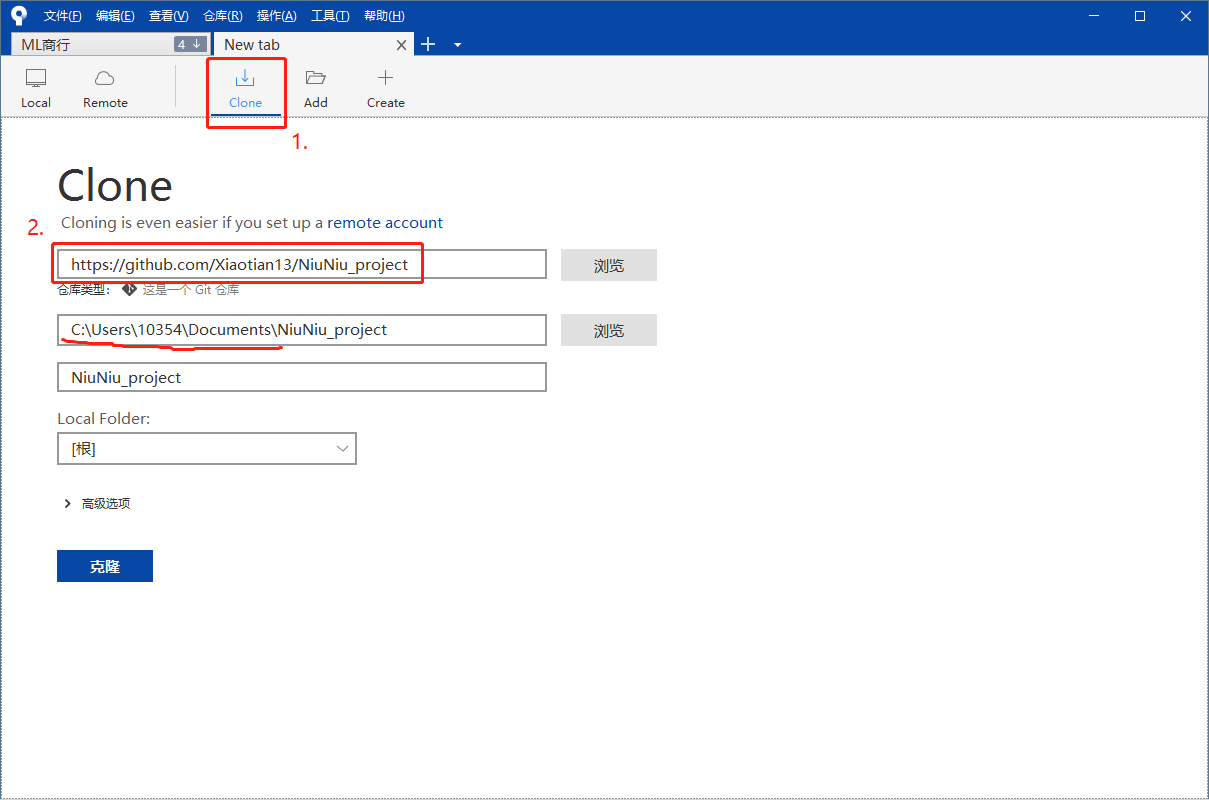




**将托管服务商改为github，点击“刷新OAuth令牌”按钮。**之后会弹出一个网页，在网页上点确认即可。之后回到这个界面，点击“确定”。至此，你已经把你的github界面添加到了sourcetree软件中，如图。



接下来**点击“Clone”**。



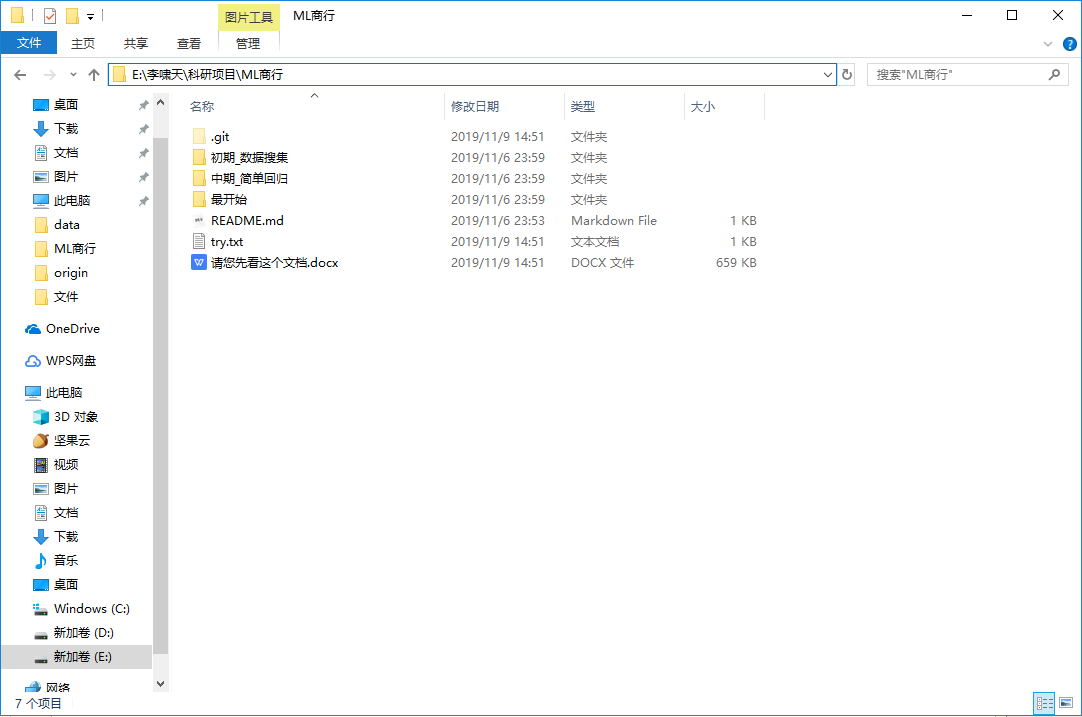
**将项目地址输入第一栏中**，项目地址为：

<https://github.com/Xiaotian13/NiuNiu_project>

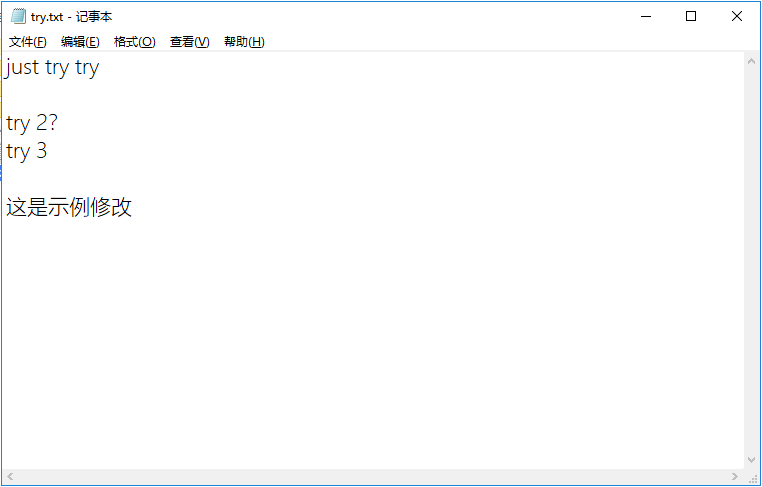
之后的第二栏会自动填充一个默认路径，可以选择更改路径；同样可以选择更改项目名（即文件夹名），但不建议更改。

原理：Clone是指，将一个项目拉取到本地，可以类比为将百度云盘中的资源下载到本地。

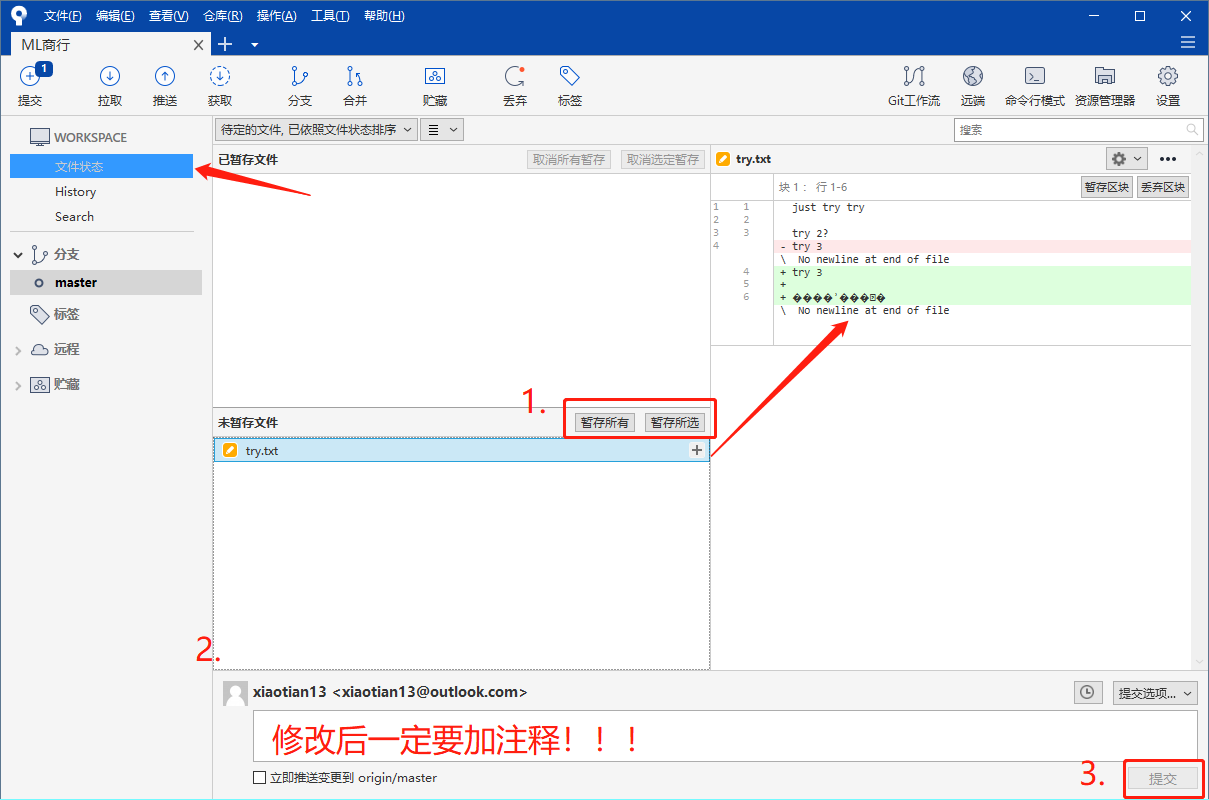
Clone成功后，在本地相应的路径处，应当已有了相应的文件。如：



下面，我们尝试对项目内的文件做出一定的修改。打开try.txt，在其中做出一定的修改，并保存文件。此处，我添加了一行“这是示例修改”，在自己实践的时候请添加其他的内容。



回到sourcetree界面，如果项目Clone成功，则会看到以下界面：

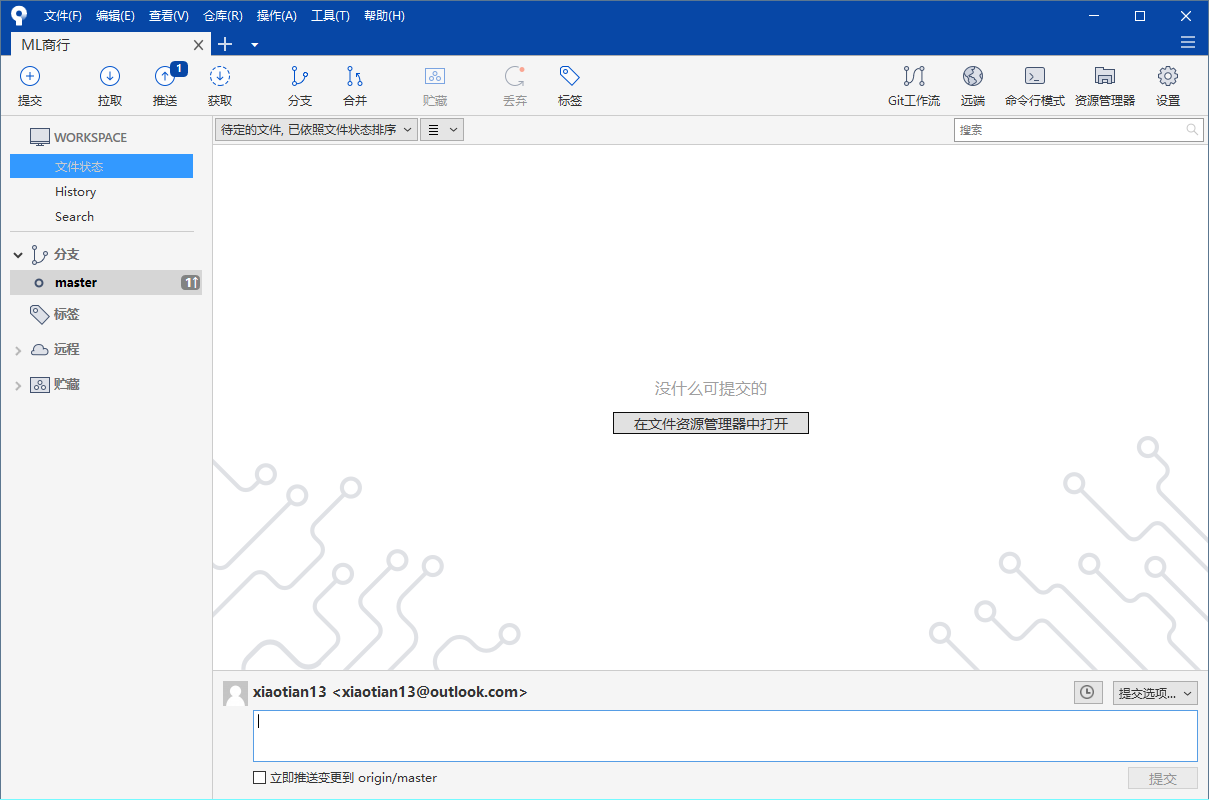


如果已经对项目内的try文件做出了修改，可以看到，sourcetree软件已经自动识别到了更改。**点击“未暂存文件”列表下的“try.txt”**，可以在右上角看到做出了什么样的修改（由于加入的是中文，所以显示了乱码）。

**接下来点击“暂存所有”或“暂存所选”**（举例：如果修改了两个文件，但暂时只想上传一个修改后的文件，就可以点击暂存所选，但不建议这样的操作，后面会解释）；

再在注释处填上对所做修改的描述，（**一定要填注释！**），最后**点击“提交”。**

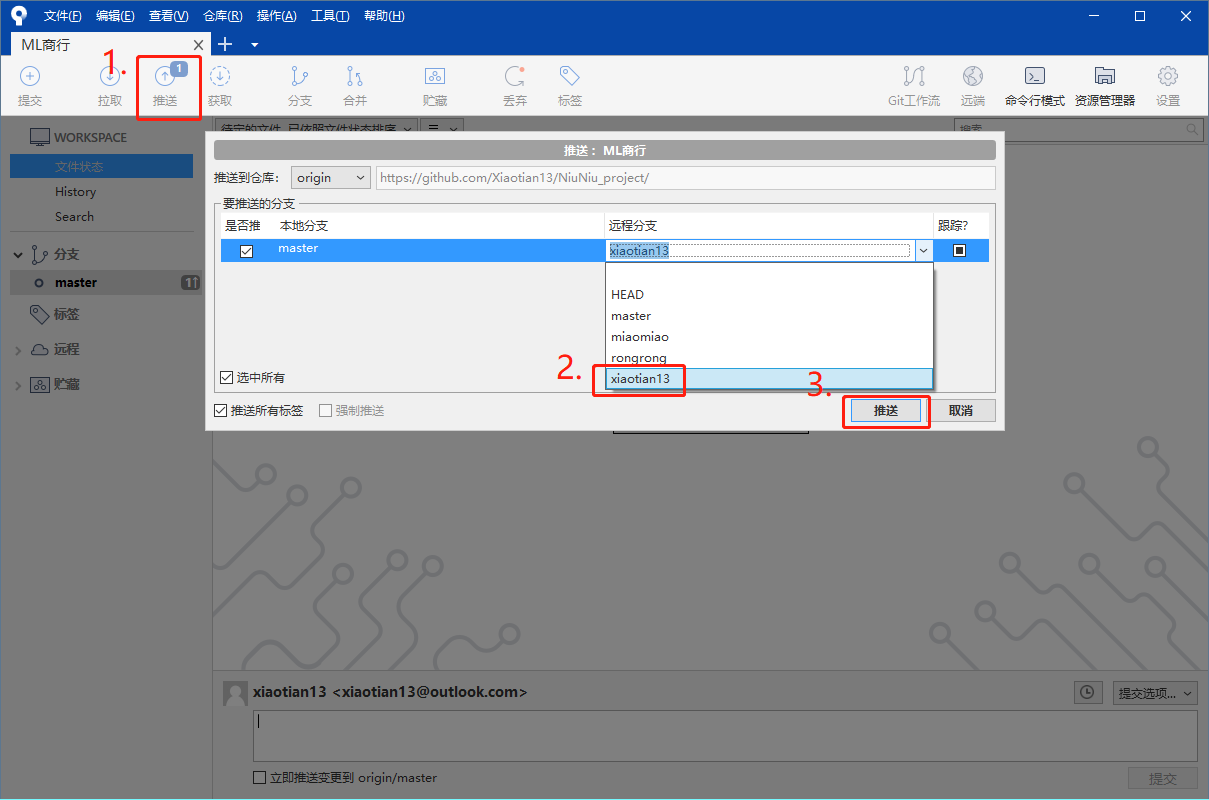
提交过后，应有下面界面：



可以看到，上面的“推送”按钮有一个小小的1上标。

原理：点击“提交”，就意味着将本地做出的更改上传至本地服务器。但也仅仅是上传到了本地的服务器，并不是直接对github上的项目做出了修改，GitHub上的项目是最终版本，显然本地做出的修改并不应该直接对最终版本做出修改，避免出现问题。

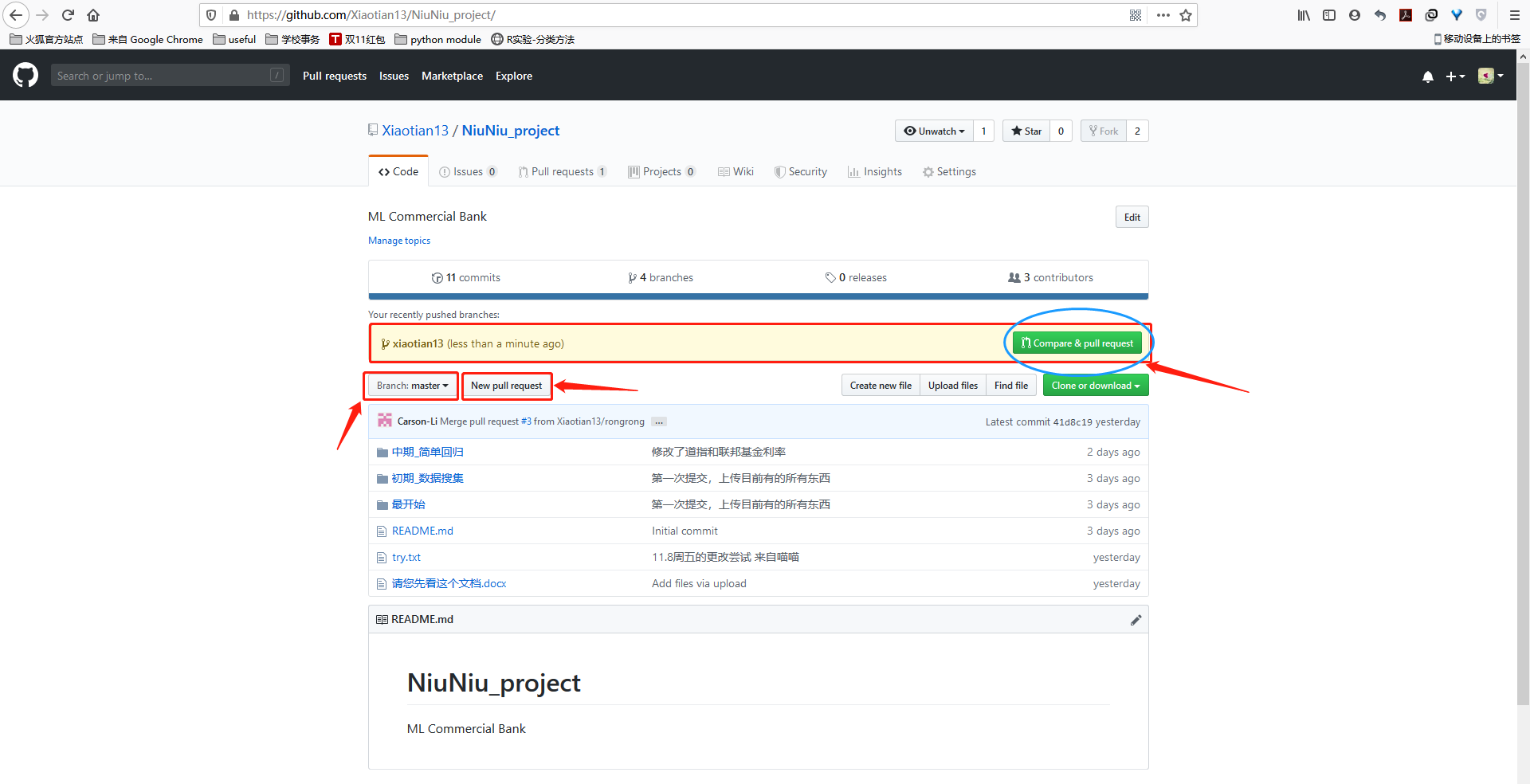
下一步，**点击“推送”**，将对项目的更改请求发送至远端。



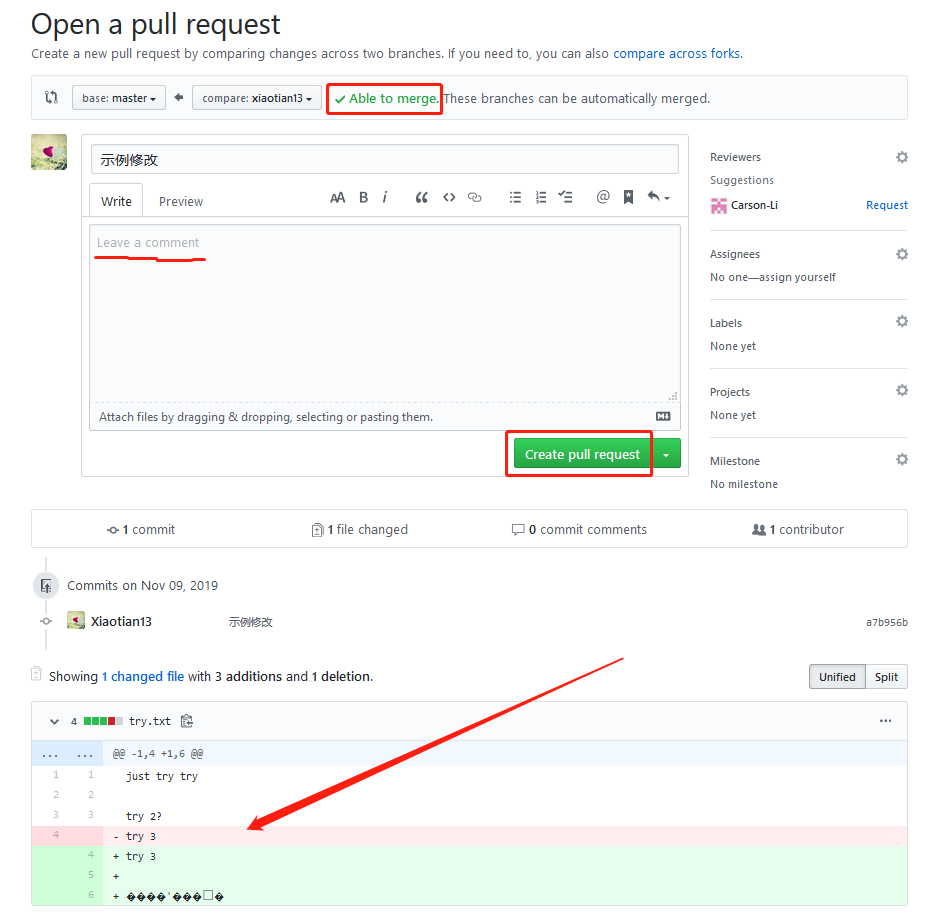
可以看到，这里要求选择“远程分支”，**请选择自己对应的分支**。如，李啸天应选择分支xiaotian13。阿仙和潘秋云将GitHub用户名发给李啸天后，李啸天也会创建额外分别属于你们两个人的分支。

原理：在一个项目的工作过程中，通常我们可以分工，如张三写函数A来实现功能1，李四写函数B来实现功能2，这两项工作可以同时进行、且无交叉。因此，不同的人将不同的工作提交至不同的分支，项目负责人审核无误后，再将各自的修改与最终版本master进行合并。

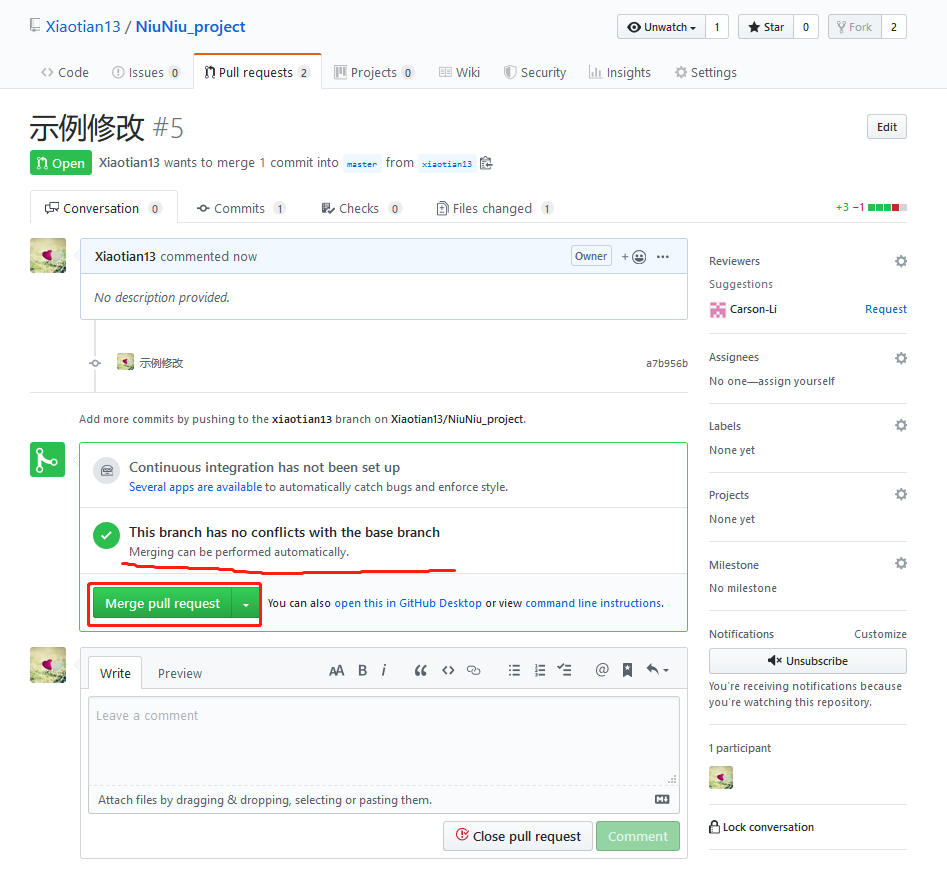
推送成功后，我们进入项目界面。



1. 可以看到，“Branch: master”，master是最终的版本，点开后分别查看不同人的分支，可以看到不同人的项目版本。
2. 可以看到，在最上面，已经显示出了最近的更改，在最右边有按钮“Compare & pull request”。顾名思义，意思是指对比新的提交与原版的差别以及创造一个pull request，及与原版本合并的请求。我们**点击它“Compare & pull request”**。

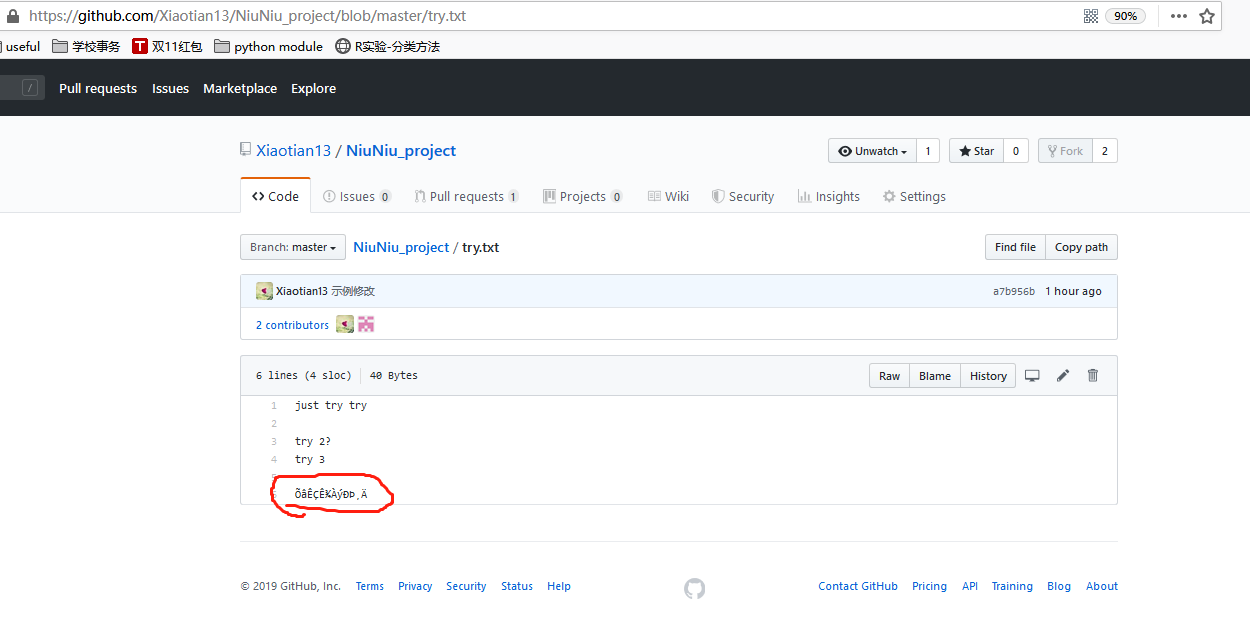


可以看到，网页中显示“Able to merge”，即你的修改与版本master之间并无冲突。最下面可以看到做出修改的内容（还是乱码）。中间部分“Leave a comment”，我们依然还是填上注释；最后**点击“Create pull request”**。



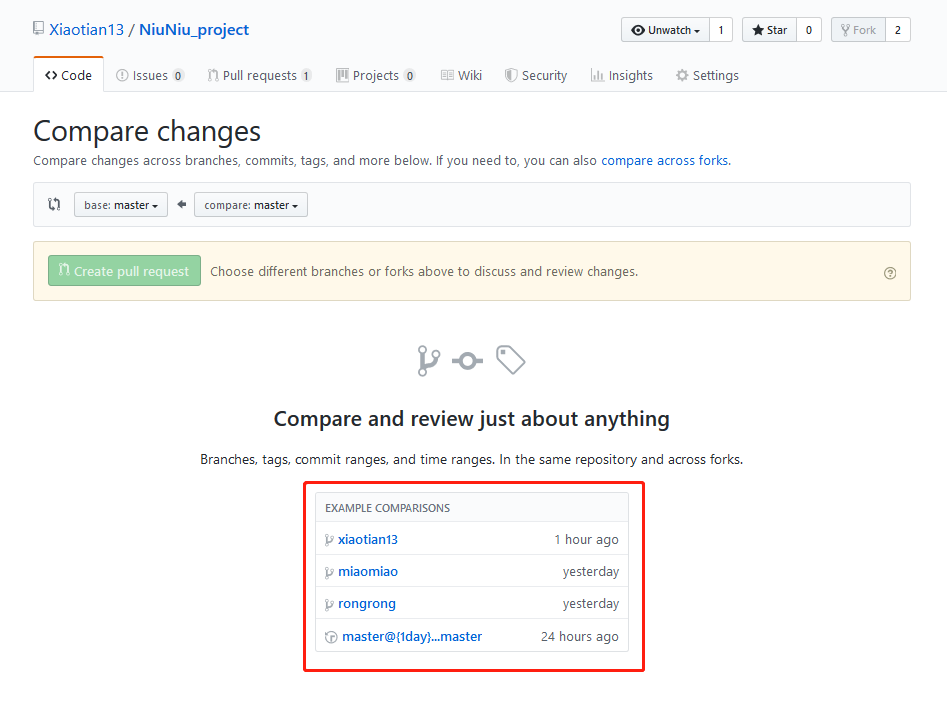
跳转界面到这里，**点击“Merge pull request”**。

我们再回到项目界面，点开try.txt文件，可以看到，我们的修改已经加入到了master版本中。

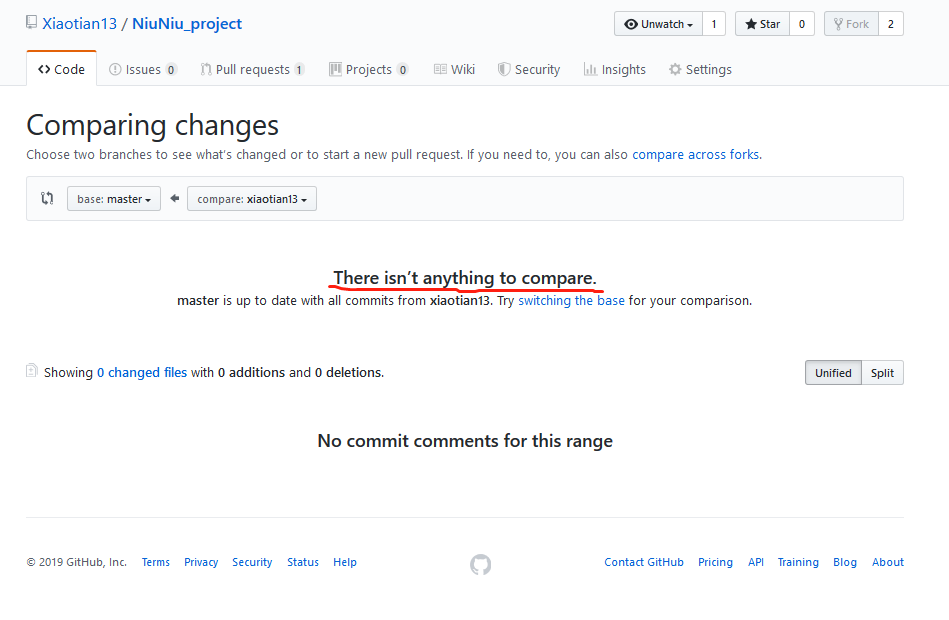


这里应该是由于编码的问题，显示还是乱码。

1. 如果sourcetree推送后，在github项目界面处找不到第2步中我们直接点的那个“Compare & pull request”，就**点击图中的“New pull request”**。



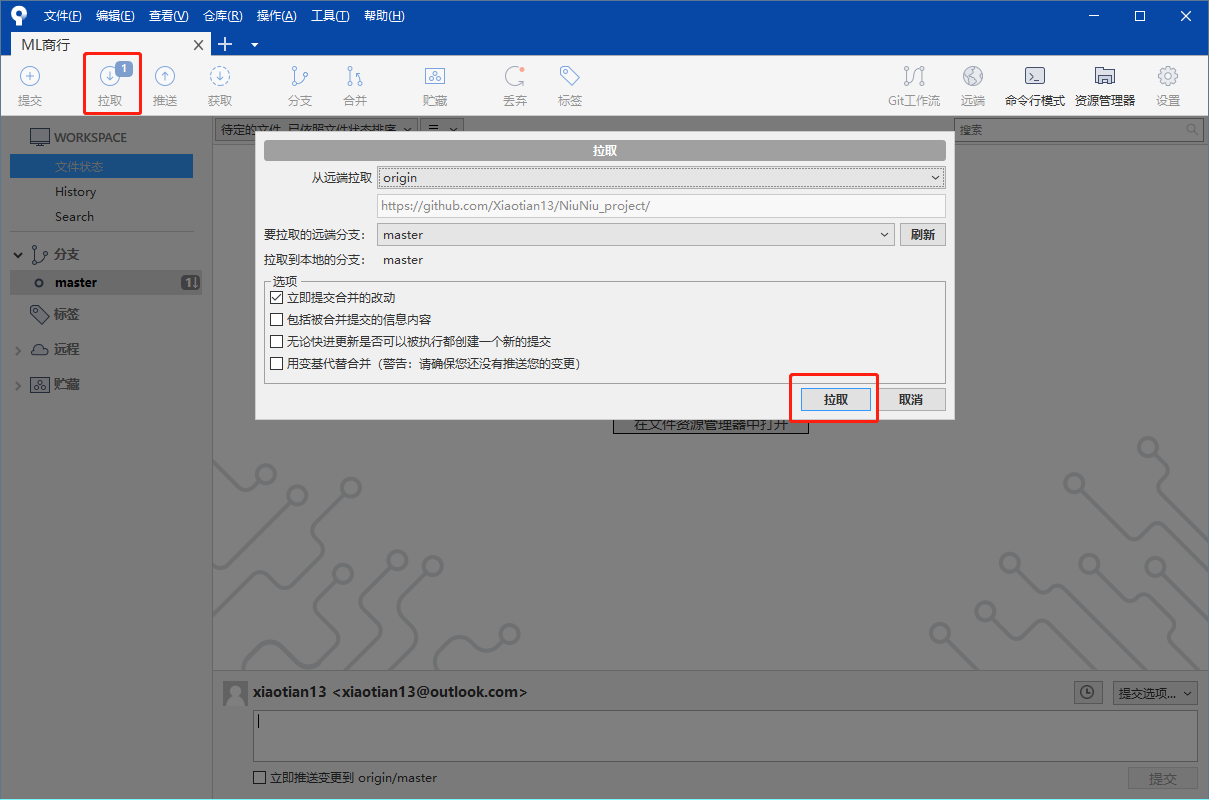
这里依然能找到不同分支对应的版本信息，以及最近一次修改的提交时间。如这里我点开分支xiaotian13：



跳转至这里。由于我已经将修改合并至最终版本中，所以这里显示的就是没有新的命令提交。

最后一步：当他人修改了master版本后，我们可以选择将他人的修改与本地同步。**注意！**如果你在项目路径下工作，同时又拉取了master的最新版本，可能会存在将你自己的工作文件覆盖的情况。所以，建议日常工作在你们原本的文件夹中，需要大家共享的文件再放到项目路径中。

回到sourcetree界面，点击“拉取。”



至此，所有的操作已经全部讲述完毕。