方案说明书

**BSN福建省区块链主干网开发者大赛**

**赛题：**基于区块链的创新应用解决方案

**方案说明书**

参赛团队名称：区块链了个二手车

参赛团队成员：兰政文、唐圣宇、林鹏基、杨隆森、林贤龙、苏梓滨

编制日期： 2021.4.28

目录

[方案说明书](#_Toc1531)

[1引言 1](#_Toc18474)

[1.1背景 1](#_Toc24204)

[1.2术语定义及说明 6](#_Toc5511)

[2设计概述 7](#_Toc3937)

[2.1任务和目标 7](#_Toc24033)

[2.2需求概述 8](#_Toc13543)

[2.3运行环境概述 12](#_Toc17458)

[2.4条件与限制 12](#_Toc1385)

[3系统详细设计 15](#_Toc12270)

[3.1 项目总体规划 15](#_Toc29591)

[3.2系统结构设计及子系统划分 21](#_Toc30176)

[3.3系统功能模块详细设计 22](#_Toc616)

[3.4界面详细设计 38](#_Toc28957)

[4数据库系统设计 39](#_Toc15556)

[4.1设计依据 39](#_Toc21607)

[4.2数据库逻辑结构 39](#_Toc6116)

[5智能合约编码设计 44](#_Toc32684)

[5.1智能合约代码结构设计 44](#_Toc32072)

[5.2智能合约代码编制 45](#_Toc17035)

[5.3智能合约使用 45](#_Toc25002)

[6神经网络价格模型编码设计 48](#_Toc14436)

[6.1需求分析 48](#_Toc18650)

[6.2数据集优化 48](#_Toc20251)

[6.3建模调参 50](#_Toc15582)

[6.4神经网络详细设计 51](#_Toc15281)

## 

## 1引言

### 1.1背景

**1.1.1开发团队简介**

团队名称：区块链了个二手车

指导老师：黄晞

团队成员：兰政文（队长）、唐圣宇、林贤龙、林鹏基、杨隆森、苏梓滨

我们是来自福建师范大学光电与信息工程学院信息工程专业的六位大三学生。

团队队长兰政文，曾带领本团队中的部分成员（林贤龙、林鹏基、杨隆森、苏梓滨）参加2020年第四届中国高校计算机大赛——网络竞赛挑战赛，设计制作的《基于SDN的自动化运维配置系统》作品获华东赛区二等奖，设计制作的作品《星图运维系统》获得全国赛二等奖。在本开发团队中参与制定产品的战略规划，收集并调查相关的市场需求、用户需求、产品需求，分析市场、行业、竞争对手发展状况及趋势，提供决策参考，协调项目实施，包括沟通、会议、项目管理，领导和管理项目团队，参与项目整体开发。

团队成员唐圣宇曾参与火币大学创业训练营，对于区块链领域有深入认知，获欧科云链区块链产业青年人才认证，曾参与波卡生态卡槽项目的研发，熟悉DeFi,NFT,IPFS等区块链主流赛道的技术底层架构，拥有充分的研发经验，在本项目中负责区块链框架的搭建，以及相应区块链部分逻辑功能的实现，在校期间参与国家级大学生创新创业项目，省级大学生创业实践项目，积极报名互联网+，挑战杯，三创赛等大学生创新创业赛事，累计获得国家级比赛奖项1项，省级比赛奖项5项，获校二等奖学金，校学习优秀奖学金，发表CN论文2篇，其中以第一作者发表与区块链相关的密码学论文1篇。

团队成员林贤龙，在2020年robomaster机甲大师赛中参与步兵机器人的电控部分和算法部分的研发与测试，项目获得了对抗赛全国二等奖，步兵机器人全国二等奖，算法全国三等奖，嵌入式全国三等奖。此外还获得了2020年全国计算机高校计算机能力挑战赛程序设计赛（C语言）华东赛区三等奖等荣誉。在开发团队中参与制定产品的战略规划，收集并调查相关的市场需求、用户需求、产品需求，分析市场、行业、竞争对手发展状况及趋势，提供决策参考。

团队成员林鹏基，在2020年大学生创新创业训练计划中参与AI刷脸支付系统的开发，参加卓跃教育嵌入式AI培训课程，参与完成仓储盘点机系统、智能行车记录仪、360度智能行车辅助系统以及校园防御系统等项目，熟悉go语言、C语言、C++、Python等多种编程语言，有良好的编程开发基础。在开发团队中负责区块链部分智能合约的编写和迭代以及系统整体测试。

团队成员杨隆森，在2019年参加于晨教育网络构架大赛获得二等奖。在两个校级的大学生创新创业训练计划立项中，分别担任了负责人和组员。熟系路由器及交换机的实机配置，对go语言，C++，Python，Java，Html5，MySQL，Linux编程开发，安卓开发等方向有深入研究，具有扎实的编程基础。在开发团队中负责神经网络机器学习价格预测模块的部分代码和迭代以及整体测试。

团队成员苏梓滨，熟悉C，C++，Python编程开发和网页开发。在开发团队中，负责项目的需求分析、神经网络机器学习价格预测模块的数据及优化、整体结构优化和预测精度提升，并担任产品的路演、宣传工作，为项目展示制作动效视频和PPT。

本项目团队在黄晞老师的指导下，重新出发，把目光汇聚于二手车市场，利用区块链+机器学习的模式，针对二手车信息不透明、估价不可靠的行业痛点，探索区块链技术产业落地的可行性。

**1.1.2待开发软件系统名称**

**基于区块链溯源与AI估价模型的二手车信息查询系统**

**1.1.3系统基本概念**

近十年来，我国二手车行业发展十分迅速。一方面是由于汽车市场的规模较大，给二手车行业带来丰富的资源。另一方面是由于政府政策的倾斜，国家陆续修订出台包括《二手车流通管理办法》在内的立法规章，规范行业发展。然而，行业虽然已经取得了一定程度的发展，但还存在着许多问题亟须解决。没有建立标准规范的价格体系，买卖双方信息不对称等问题在制约着行业发展。隐瞒质量缺陷、虚报价格的现象普遍存在,消费者对二手车市场难以产生信任。我们的团队从中看到了发展的机会，针对行业发展的痛点。我们提供了相应的解决方案。

1. **建立二手车诚信档案，实现一车一档。**

覆盖机动车生产、销售、登记、检验、保养、维修、保险等全过程信息并以车辆状况报告的形式提供给消费者。打消消费者对二手车信息的真假和质量的好坏的疑虑，不让消费者为车辆事故，性能和里程数的真实性而担忧。

1. **基于市场成交情况，相关评估机构和成交车具体车况，通过机器学习训练评估模型为用户提供可靠的价格参考。**

我们利用自己制作的车辆报告提供的相关信息帮助用户作出购买的决定。利用二手车的诚信档案的绝对真实，增强品牌公信力，领跑整个行业，进而可为二手车市场制定明确和透明的价格参考体系，规范行业秩序。

我们利用区块链溯源技术可以实现从多方位提供真实数据，为每一部车辆打造一份独一无二的历史档案；提取区块链中大量的真实数据作为数据集辅以人工智能训练估价模型最后给出科学可靠的车辆真实报价。用户只需在终端输入目标车辆的车架号就能获取对应车辆的相应数据，包括过往的维修，保养，过户，出险等记录。将二手车信息整合给用户，辅助用户作出决定，以解决二手车市场价格体系混乱，挂羊头卖狗肉现象丛生的痛点问题。从而将二手车市场透明化，将二手车交易诚信化。利用区块链公开透明可查的独特优势，开发新的二手车信息服务方式，力求成本的最小化和利益的最大化，拓展市场空间，扩大市场占有率，成为二手车信息服务市场的领跑者。

**1.1.4项目建设目标及效益**

**1.1.4.1项目建设目标**

开放车辆登记信息数据，构建完善的二手车信息系统,从而能够统一车辆检测及销售方式, 查询车辆维修保养和出险记录,排查法院查封车辆和营运车辆;通过历史成交记录保证车辆充分询价，信息公开透明，打通售前、售后, 共享信息资源,加快形成健康诚信的统一大市场。

**（一）具体战略**

**（1）初期（1-2年）**

主要是让区块链＋二手车这种新型二手车信息服务方式走进市场，建立自己的品牌，积累无形资产，并且收回初期投资，准备扩大服务范围，开始准备研究提供新的服务方式。

**第一年：**

●服务进入市场，与交管所、维修店、二手车售卖方，保险方进行合作；

●为区块链二手车项目做广告宣传，提高项目的知名度，建立自己的品牌；

●逐渐与更多的二手车交易方进行合作，每天区块链二手车上链数开始增加；

**第二年：**

●扩大区块链在服务群体的影响，加深与交警等政府方面合作；

●根据上年的收入情况将二手车上链的价格适当提高，逐步完善整个服务体制；

●市场逐渐扩大，开始扩大服务范围；

1. **中期（3-5年）**

●整个区块链服务已经逐渐成熟，上链加配套服务不断完善；

●进一步完善服务质量，同时不断开拓市场，向中小城市发展，保持每年开拓一定的市场数；

**（3）远期（6-9年）**

利用区块链公开透明可查的独特优势，开发新的领域的二手车信息服务方式，力求成本的最小化和利益的最大化，再度拓展市场空间，扩大市场占有率，成为二手车信息服务市场的领跑者。

**（二）发展延伸方向**

**（1）横向延伸：**依靠对区块链技术的不断研究，研究更多保障二手车公平公正交易，为二手车售卖方健全诚信经营制度的规则，以促进二手车信息服务商的市场影响力，减少未来竞争者的加入，抢占市场资源。

**（2）纵向延伸：**立足于二手车交易市场领域，增加除现有二手车上链服务以外其他的保障二手车交易过程中各种信息公开与透明的手段，并拥护一定粉丝群体，让大众对区块链的信任度显著提升，占据国内二手车信息服务提供商的主导地位。

**1.1.4.2效益**

基于区块链溯源与AI估价模型的二手车信息查询系统是以信息供应为基础，提供信息供应与评估一体的多元化服务。系统以提供准确快速的服务为目标。力求为客户提供最真实可靠的二手车诚信档案，加强顾客对相关车辆的具体车况信息的了解。

**系统的核心竞争力**：以客户服务为中心，以真实准确，合理可靠为理念，通过区块链溯源和机器学习技术的结合运用，解决了二手车市场价格体系混乱，挂羊头卖狗肉现象丛生的痛点问题，使得用户可以放心购买心仪的二手车而不用担心受到欺骗，或被隐瞒信息。同时，应用大数据进行机器学习能够随着成交数量的上升，更加准确的对车辆进行价格评估，提供精准的价格评估服务。

### 1.2术语定义及说明

（1）本地：运营本软件的公司。

（2）VIN：车辆识别码。

（3）车辆历史档案信息：该档案包括车辆各项参数，历史记录的里程数，是否泡水火烧，是否遭遇重大事故，历史维修保养记录，历史出险记录等信息。

（4）机器学习：机器学习就是通过算法，使得机器能从大量历史数据中学习规律，从而对新的样本做智能识别或对未来做预测。

（5）神经网络:一组粗略模仿人类大脑，用于模式识别的算法。神经网络这个术语来源于这些系统架构设计背后的灵感，这些系统是用于模拟生物大脑自身神经网络的基本结构，以便计算机能够执行特定的任务。

（6）输入层:接收输入数据,输入层会将输入数据传递给第一个隐藏层。

（7）隐藏层:对输入数据进行数学计算。创建神经网络的挑战之一是决定隐藏层的数量，以及每一层中的神经元的数量。

（8）输出层：人工神经网络的输出层是神经元的最后一层，主要作用是为此程序产生给定的输出，在本例中输出结果是预测的价格值。

（9）数据集：是一个单独地或组合地或作为一个整体被访问的数据集合。

（10）梯度下降：梯度下降法是一种求函数最小值的方法。在这种情况下，目标是取得成本函数的最小值。它通过每次数据集迭代之后优化模型的权重来训练模型。通过计算某一权重集下代价函数的梯度，可以看出最小值的梯度方向。

（11）归一化处理：归一化方法有两种形式，一种是把数变为（0，1）之间的小数，一种是把有量纲表达式变为无量纲表达式。主要是为了数据处理方便提出来的，把数据映射到0～1范围之内处理，更加便捷快速。

（12）损失函数：损失函数或[代价函数](https://baike.baidu.com/item/%E4%BB%A3%E4%BB%B7%E5%87%BD%E6%95%B0/7048599)是将随机事件或其有关随机变量的取值映射为非负实数以表示该随机事件的“风险”或“损失”的函数。

## 2设计概述

### 2.1任务和目标

**2.1.1目标**

针对整个二手车市场行业乱相，提出基于区块链技术的二手车数据查询、评估方案，以解决在二手车交易过程中出现信息错误，售卖方隐藏车辆真实信息等问题，通过对二手车链上数据的溯源，实现二手车辆可靠，精确评估。同时在区块链链网络中引入交警方和保险方，针对不同的业务逻辑加以实现。从二手车的售卖，监管，保险三方面一条龙服务，提高二手车市场运作效率与参与方的相互信任度，为我国繁荣的区二手车市场铺出一条光辉之路。

**2.1.2任务**

采用区块链技术记录车辆信息提供方产生的相关数据信息以确保车辆信息的可靠性，通过机器学习技术创建及训练估价模型向用户提供车辆估值，并将用户实际成交价反馈给估价模型，以闭环的形式再训练模型，提升估价模型的精度。

### 2.2需求概述

**2.2.1概要描述**

从车辆信息产生方（交管所，保险公司，维修店）获取二手车的里程数，车龄，车架号，维修保养记录等相关数据。自主设计创世块，并将相关数据上链。通过智能合约的形式，从区块链中调取数据作为机器学习估价模型训练的数据集。消费者只需在前端界面输入目标车辆车架号即能得到一个包含车辆真实历史信息的档案以及目前车况下基于机器学习的可靠报价。

**2.2.2业务需求**

1970年美国经济学家乔治·阿克尔洛夫在其论文《“柠檬”市场：质量不确定性与市场机制》中为读者深入地分析了当时的市场上二手车市场交易的背景和实际的状态，这也是信息不对称理论以及逆向选择理论第一次出现在学术圈子中，各种因素的作用下导致二手车市场暴露出了信息的不对称性的弊端，就使逆向选择现象在市场中上演，买二手车的一方并不能准确地判断二手车的使用情况以及质量，由于大多数的购买者抱着低价购买的心理，这就使得很多拥有车辆状况较好的车主参与二手车交易的意愿不强。

目前的二手车行业龙蛇混杂，改里程，瞒车况，事故车翻新，的现象时有发生。基于区块链溯源与AI估价模型的二手车信息查询系统为车辆提供上链技术，以可靠的数据来源保证车辆情况的真实性，使二手车交易行业的“水”越来越浅，越来越清，从而减少消费者对购买二手车辆产生的疑虑，增强购买欲，刺激消费。

**2.2.3主要功能**

（1）查询指定车辆的大到里程数、车龄、车况，小到篡改痕迹、车内饰等一系列历史信息，提供二手车辆最真实报告。  
（2）针对不同车辆信息产生方制定对应数据上链接口，实现上链数据的高效无误，打破数据孤岛。  
（3）利用车辆车架号与区块链进行绑定，实现一车一码，落实区块链溯源防伪。  
（4）利用区块链中数据作为数据集，进行机器学习，得出真实可靠估价模型。  
（5）根据用户反馈的真实价格加入区块链中更新数据集，形成数据璧煌，进一步提高训练精度。  
（6）结合用户目标车辆VIN、目标车辆位于区块链中历史数据以及基于机器学习的估价模型，生成带有参考价格的车辆专属档案。

**2.2.4输入**

所要查询车辆的车架号

**2.2.5输出**

包含车辆所有历史信息的档案报告及基于真实车况的可靠估价

**2.2.6需求报告书**

**（1）区块链相关：**

* 实现XuperChain百度超级链的部署。
* 实现超级链部署在区块链服务网络BSN中。
* 区块链可以实现节点数据上传，数据查询，数据统计，记录时间功能。
* 在区块链中分别设立维修店，保险公司，车管所以及本地的节点。
* 实现可以通过VIN号调取对应车辆位于区块链内所有信息的功能。
* 实现对区块链中所记录数据的整合，包括在库车辆总数等信息的功能。
* 实现通过遍历，整合，筛选区块链内数据，进而生成可用于机器学习的数据集的功能。

**（2）机器学习相关：**

* 实现通过将生成的数据集加入神经网络训练集中的功能。
* 通过神经网络训练获得二手车估价模型功能。
* 用户提交的真实成交价格可以加入数据集中进行神经网络训练。

**（3）用户相关：**

* 实现通过VIN查询车辆在区块链中的历史档案信息以及该车辆通过估价模型计算后产生的价格。
* 用户可以通过本软件反馈真实成交价格。

**（4）交管所相关：**

交管所上传数据接口可以实现通过VIN对车辆基本信息，是否盗抢，发生何种交通事故等信息的上链操作。基本信息包括VIN、发动机号、上牌时间、轴距、汽车厂商、变速箱、汽车类型、排放标准、汽车排量、最大马力、过户次数、重大事故、火烧泡水。

**（5）保险公司相关：**

* 保险公司上传数据接口可以实现通过VIN对车辆出险记录等信息的上链操作。出险记录包括VIN、车辆原价、出险内容、更换零件。

1. **维修店相关：**

维修店数据上传接口可以实现通过VIN对车辆信息进行上链操作。车辆信息包括VIN、里程数、电动门窗、全景天窗、GPS导航、倒车雷达、倒车影像、真皮座椅、轻微碰撞、漆面修复次数、钣金修复次数、外观件更换次数、维修保养内容、更换零件。

**（7）运营本软件公司相关：**

本地上传数据接口可以实现通过VIN对车辆真实成交价格的上链操作。

**（8）需求分析验证：**

1. 所有的需求是一致的，任何一条需求都没有和其他需求相互矛盾，故符合一致性。
2. 需求完整，规格说明书包括用户需要的每一个功能性能，故符合完整性。
3. 指定的需求是现有硬件技术和软件技术基本上可以实现的，故符合现实性。
4. 需求是正确有效的，确定能解决用户面对的问题，故符合有效性。

### 2.3运行环境概述

**2.3.1区块链运行环境**

* 基于Ubuntu Linux16.04操作系统，
* GCC版本需在4.9以上，
* Golang版本为1.13.x系列
* 使用的框架为xuperchain，
* 需预先部署好xuperchain的运行环境

**2.3.2机器学习运行环境**

* 基于百度AIstudio平台进行训练
* Python版本：Python3.7
* 框架版本：PaddlePaddle 1.5.1
* CPU：Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz
* RAM ：32GB
* GPU：NVIDIA Tesla V100 16GB

### 2.4条件与限制

**2.4.1条件**

近十年来，我国二手车行业发展十分迅速。且在未来很长的一个时期内，我国二手车市场仍会不断扩大。但是二手车市场却存在着许多亟需解决的问题，在国家陆续修订出台包括《二手车流通管理办法》在内的立法规章，规范行业发展的环境下，我们的项目应运而生。

**2.4.2限制**

**2.4.2.1外部限制**

（1）国内二手车行业没有一个公开透明的评估鉴定方式。

解决方案：由本软件基于市场状况以及真实车辆数据训练的评估模型给定评估体系。

（2）二手车相关 行业法规制定不完善。

解决方案：国家正在大力扶持二手车行业，会陆续建立健全相关法规。

（3）没有公开透明的价格体系。

解决方案：由本软件基于市场状况及真实车辆数据训练的评估模型给定价格体系。

（4）区块链实地应用在市场上还不普遍，群众接受度不高。

解决方案：在初期投入精力宣传，通过优秀的产品力培养产品粉丝。

（5）从维修店，经销商，车管所等机构获取数据不易。

解决方案：展开相应的商业合作，互惠互利。

**2.4.2.2进度限制**

（1）项目还处于起步阶段，用于训练和测试的数据较少，训练出来的模型还不够准确.

解决方案：前期用于训练模型的数据主要来源于经销商，可能导致模型不够精准，而随着用户群体的增多，数据样本逐渐丰富与清晰，模型的精准度会相应提高。

（2）还未能与相关机构展开合作。

（3）附加功能还有待开发。

**2.4.2.3内部限制**

项目团队是一个开放活泼的大学生创业团体。对于市场大环境不够敏感，在业务层面更多的是基于理性的思考，没有实际落地的经验。

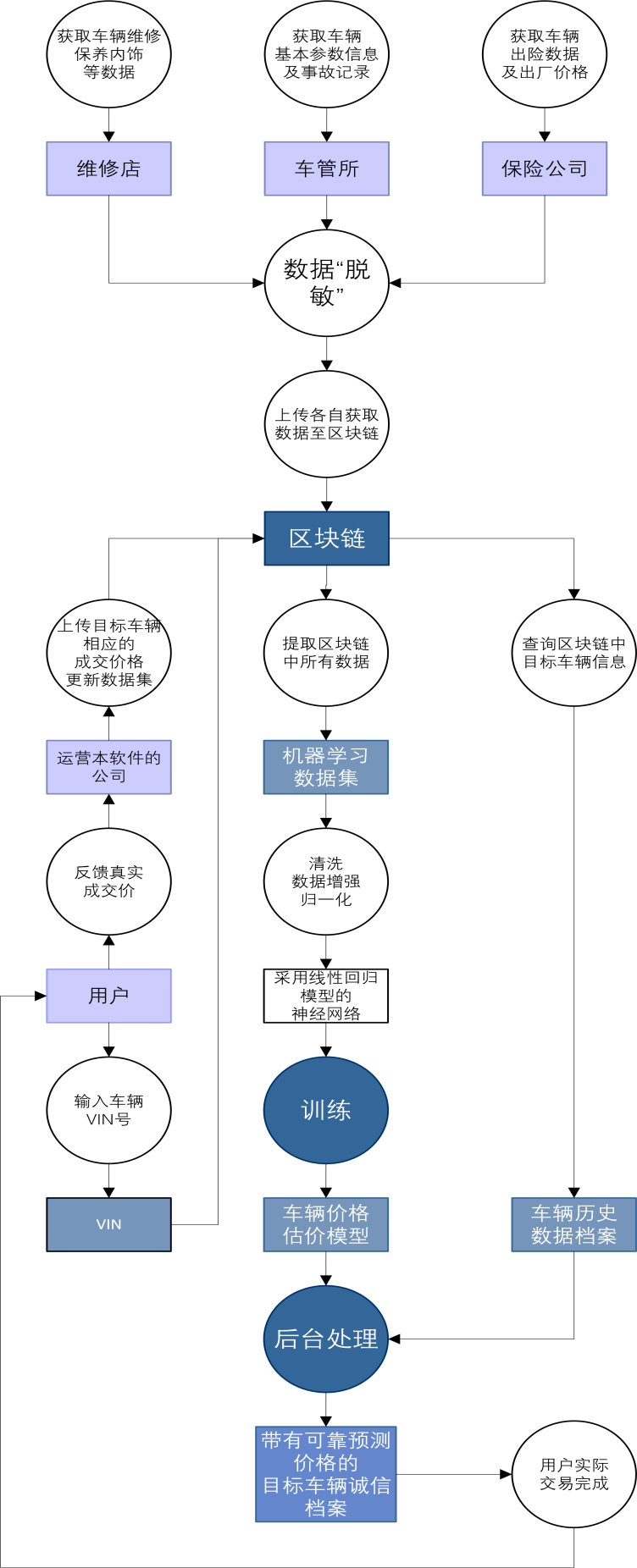
由于计算机算力有限，神经网络机器学习过程较为缓慢。

**2.4.2.4技术限制**

随着上链数据的不断上升，对区块链性能的要求越来越高，原有的区块链底层架构可能不能满足不断增加的业务需求，后期需要对架构进行完善和改造。

## 3系统详细设计

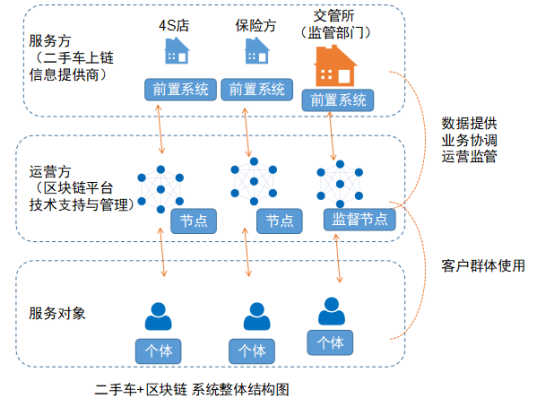
### 3.1 项目总体规划

****

**图3-1 项目总体规划图**

**3.1.1整体架构**

项目把二手车信息处理业务与区块链技术相结合，以去中心化的方式按照一定的时间顺序，集体记录，维护二手车售卖，上保，处罚等二手车相关的信息，实现跨区域的业务协调，为个人提供便利的二手车历史溯源信息“一检查询”服务。后续还可以根据具体的业务实现模块，将更多的车辆处理信息加入到基于BSN网络环境的区块链平台中，发挥区块链的技术优势。整体的结构图如图3-2所示：



**图3-2 系统整体结构图**

本架构是由交管所及监管部门作为区块链的发起方，并由各处二手车贩卖的维修店，为二手车上保险的保险机构参与并建设二手车联盟链。第一批次有维修店，保险方，交管所三类节点，其中交管所的权限最大，除了可以上链违规二手车处罚信息外，还起到监管的作用。后续平台运营成熟后，可扩展至全国各地交通部门。

**（1）服务方：**也是区块链联盟，包括发起方兼监督方交管所，参与方维修店和保险方。联盟成员也是链上数据保存节点，主要承担数据提供、数据共享、监管等职能。服务方的业务系统通过其前置系统实现二手车属性信息数据上链、链上业务协同和业务监管。

**（2）第三方企业：**区块链平台建设方、技术支持与运营管理方，承担区块链平台的建设、日常维护、节点管理、性能测试等职能。

服务对象：链上数据使用者，本方案主要为个人以及需要查询二手车信息的机构。个人可以登录平台，查询各车俩的属性信息，保险上保信息，交警处罚信息，并能够从一键查询操作中得到一个由人工智能大数据算法根据车辆链上信息计算的一个购置参考价格。

**3.1.2技术选型**

**（1）底层联盟链网络选择xuperchain**

区块链按照开放性及去中心化程度可以分为公有链、私有链、联盟链。本项目主要是为交管所，二手车售卖方，车链保险销售方提供服务的，因此更适合采用联盟链模式。技术选型上，选择由百度超级链团队研发的较为成熟的开源项目XuperChain。

XuperChain是超级链体系下的第一个开源项目，是构建超级联盟网络的底层方案。其主要特点是高性能，通过原创的XuperModel模型，真正实现了智能合约的并行执行和验证，通过自研的WASM虚拟机，做到了指令集级别的极致优化。在架构方面，其可插拔、插件化的设计使得用户可以方便选择适合自己业务场景的解决方案，通过独有的XuperBridge技术，可插拔多语言虚拟机，从而支持丰富的合约开发语言。在网络能力方面，XuperChain具备全球化部署能力，节点通信基于加密的P2P网络，支持广域网超大规模节点，且底层账本支持分叉管理，自动收敛一致性，TDPOS算法确保了大规模节点下的快速共识。在账号安全方面，XuperChain内置了多私钥保护的账号体系，支持权重累计、集合运算等灵活的策略。

**（2）共识机制选择XPOA**

XPoA是超级链对PoA的一种实现，其基本思想是在节点中动态设定一组验证节点，验证节点组在预设的时间段内进行组内轮流出块(称之为轮值)，即其余节点在某特定验证节点V出块的时间段内统一将交易发送给V，交易由该验证节点V打包成区块。XPoA支持动态变更验证节点，可以通过指令修改现有的验证节点组，包括对当前验证节点组进行删除和添加操作。在该算法中，预设时间段包括确定单个区块的出块时间，以及验证节点单次轮值出块数量。 同样，XPoA通过Chained-BFT算法来保证轮值期间的安全性。

在XPoA中，网络中的节点有两种角色，分别是“普通节点”和“验证节点”：

**A、普通节点：**普通节点仅对验证节点进行验证，计算当前时间点下验证节点地址是否于计算结果吻合。

**B、验证节点：**进行区块打包工作；在更改验证节点组过程中，多数验证节点需确定更改结果添加和删除操作方能生效。

验证组信息通过合约调用进行修改，流程主要有以下几点：

A、在收到信息后，验证节点通过签名信息确认交易真实性。

B、验证节点在UtxoVM中进行系统调用并更新当前验证人集合读写集。

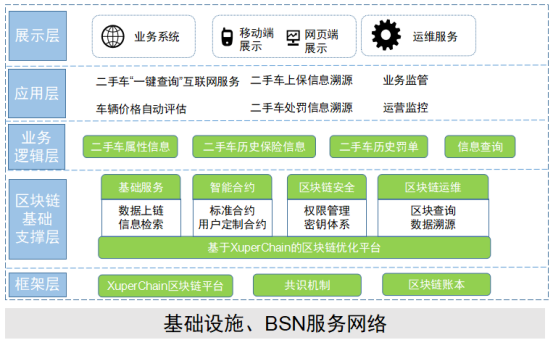
C、验证人集合并不会立即影响当前共识，在三个区块后集合才能生效。

这样每一轮的时间由配置xuper.json指定，在单轮时间段内，区块打包由目前验证节点组中的节点按顺序轮流完成。在通过合约发起验证节点变更后，变更会在三个区块后才触发，然后验证节点按照新的验证组继续进行轮值。这种模式更适用于大型网络的部署应用，故选择XPOA共识。

**（3）数据存储**

由于二手车上链数据需求多样，分为全量数据上链、轻量数据上链、大数据上链等，同时需要兼顾数据隐私和存储空间的要求，本方案采用区块链账本、传统数据库（如 MySQL）和 IPFS（InterPlanetary File System，星际文件系统）相结合的方式。其中IPFS是一个旨在创建持久且分布式存储和共享文件的网络传输协议，它是一种内容可寻址的对等超媒体分发协议，在IPFS 网络中的节点将构成一个分布式文件系统，支持私有化部署。

**3.1.3系统架构**



**图3-3 系统架构图**

**（1）基础设施、BSN服务网络**

本项目在基础设施层面将依托现有的BSN服务网络、网络安全设备，二手车售卖方节点，保险方节点，交管所节点等在网络层面是连通的、安全的，网络带宽至少在达在200M以上。

在硬件设施层面，作为区块链存储节点，可以使用云平台环境或单独部署生产服务器来部署区块链网络。区块链的存储节点服务器系统配置要求大致如表3-1所示：

**表3-1 存储节点服务器系统配置要求**

|  |  |  |  |
| --- | --- | --- | --- |
| CPU | 内存 | 硬盘存储 | 备注 |
| 4G | 8G | 80G | 车辆信息迭代前期相对缓慢，可采用数据全量上链 |

**（2）框架层**

框架层提供平台了所需的基础软件支撑，包括区块链账本、共识机制、区块链平台等。

区块链账本：按照时间顺序将数据区块以首尾顺序相连的方式组合成的一种链式数据结构，并以密码学和可信时间戳机制保证不可篡改、不可伪造。

区块链平台：通过 P2P 协议实现区块链节点之间的数据通讯，包括分布式组网机制、数据传播机制和数据验证机制等。

共识机制：主要封装网络节点的各类共识算法，如xuperchain所用到的XPOA、XPOS等共识机制。

另外，在链上数据量较大的场景，采取本地数据库保存主体内容、链上保存内容摘要的方式存储数据，此时需要分布式的数据库予以支撑。

**（3）基础支撑层**

提供基础服务、智能合约、安全体系和区块链运维的支撑。基础服务：基础服务包括数据上链、内容发布、信息检索等功能。

智能合约：提供合约部署、测试、接口和实例管理的服务，并开发相关职能合约进行链上数据的操作。

区块链安全体系：包括区块链平台用户的秘钥管理、权限管理等。

区块链运维：提供平台上各节点的运行监控功能，以及节点上区块信息查询、链上数据溯源等功能。

**（4）业务逻辑层**

实现平台的业务逻辑，为应用层提供操作链上数据的中间层，包括二手车属性信息，各个节点权限操作以及信息查询服务。

**（5）应用层**

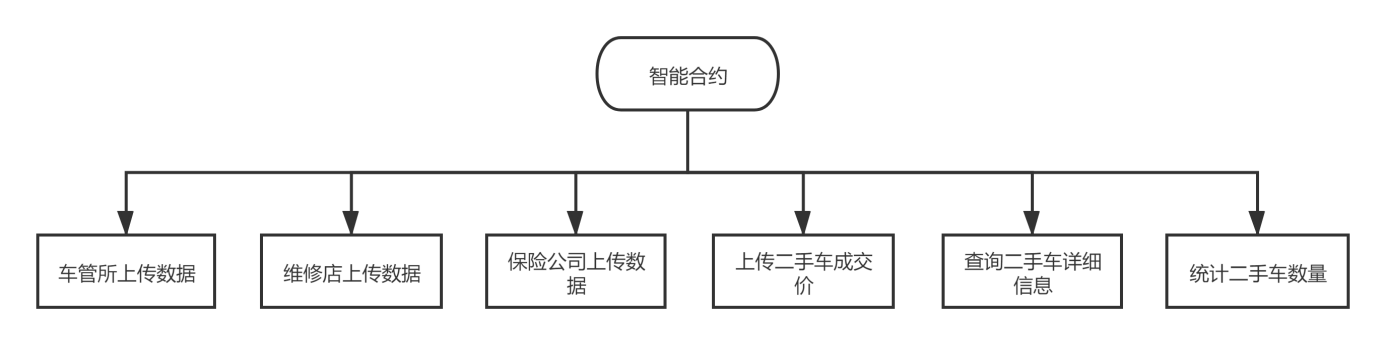
本方案是为二手车领域的用户及商家提供基础的区块链应用。各个联盟链协同作用下，实现二手车链上数据的写入，查询，通过链上治理的逻辑，实现业务监管，运营监控的逻辑。

**（6）展示层**

展示层将区块链数据可视化，为个人提供移动应用、服务网站、自助服务等应用，需要在原相关系统中增加区块链业务的使用功能。

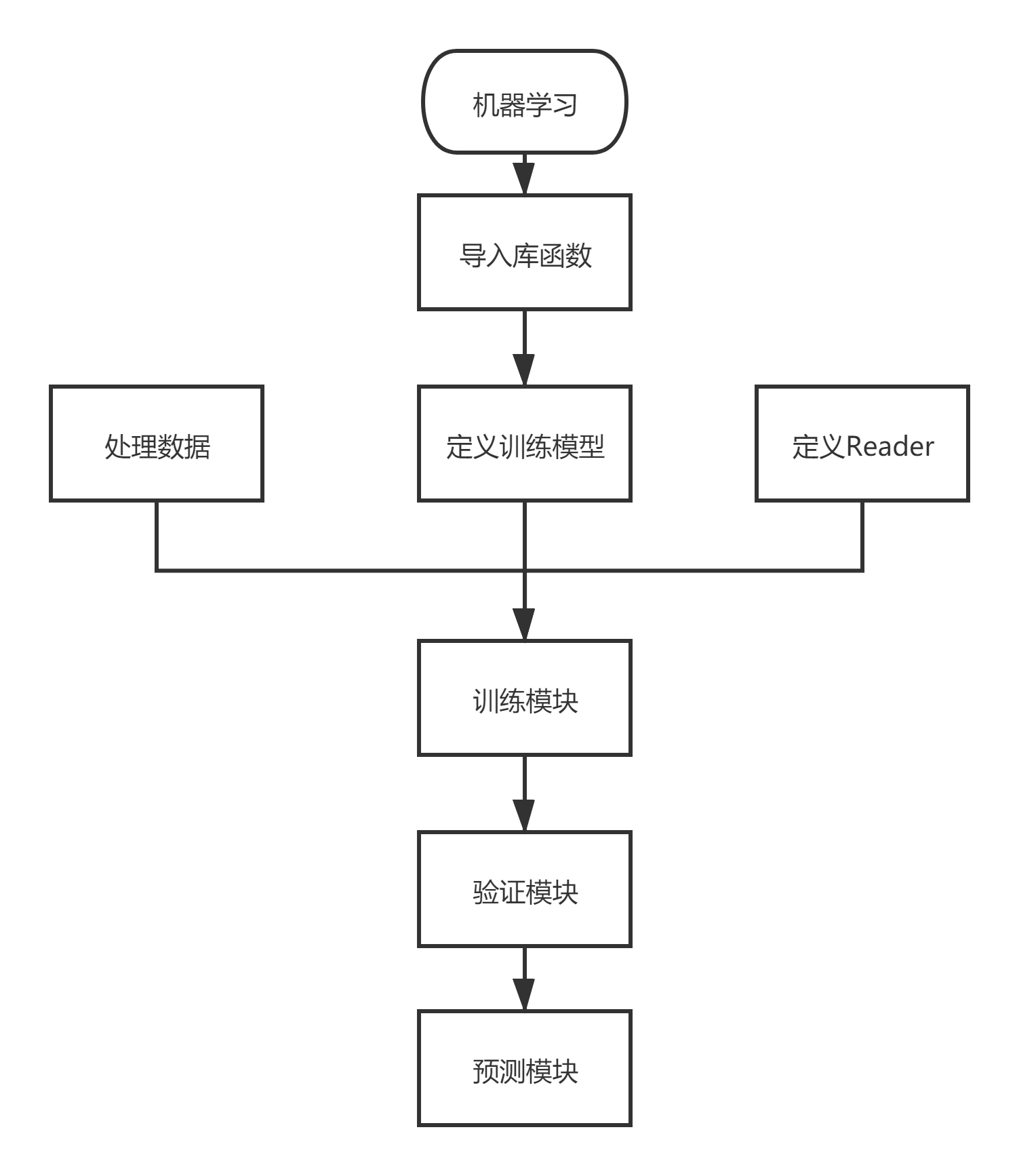
### 3.2系统结构设计及子系统划分

**3.2.1智能合约的组成**



**图3-4 智能合约模块组成图**

**3.2.2机器学习的组成**

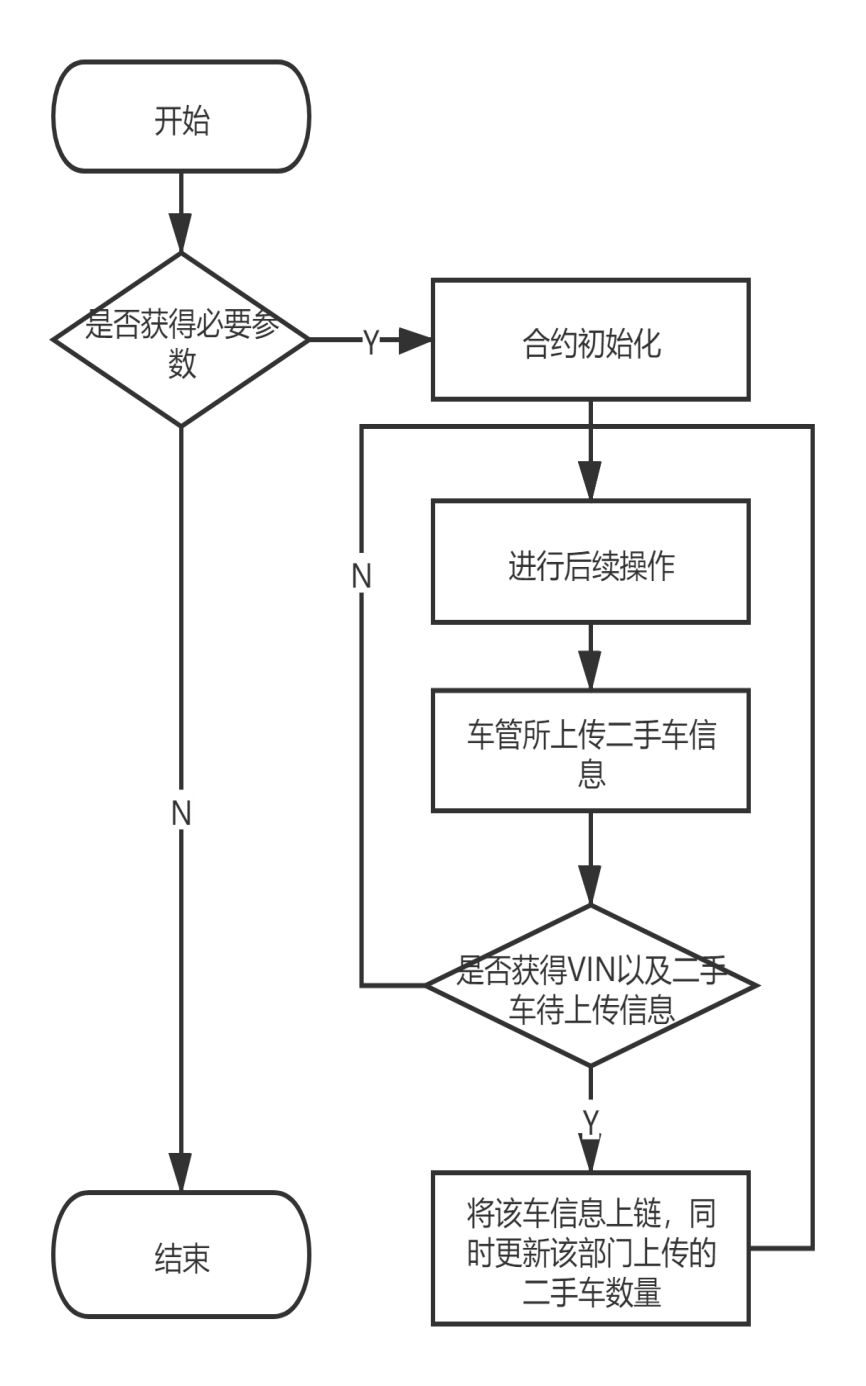


**图3-5 机器学习模块组成图**

### 3.3系统功能模块详细设计

**3.3.1.1车管所上传数据的智能合约**

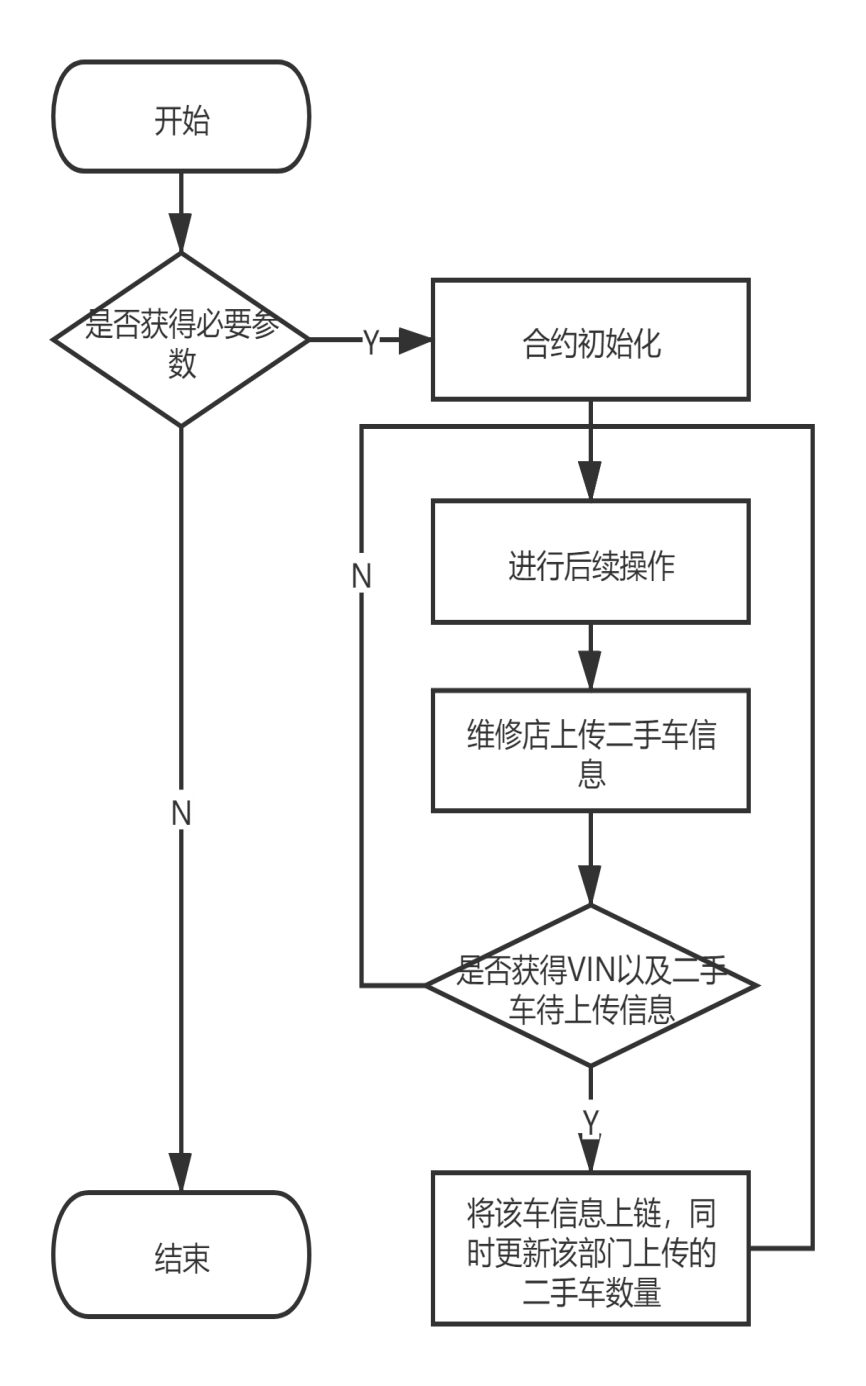
合约在调用其方法前需要进行初始化，在成功进行初始化后，等待车管所将二手车信息进行上链。上链操作需要提供车辆的VIN以及该车辆的具体信息。



**图3-6 车管所上传数据的流程图**

**3.3.1.2维修店上传数据的智能合约**

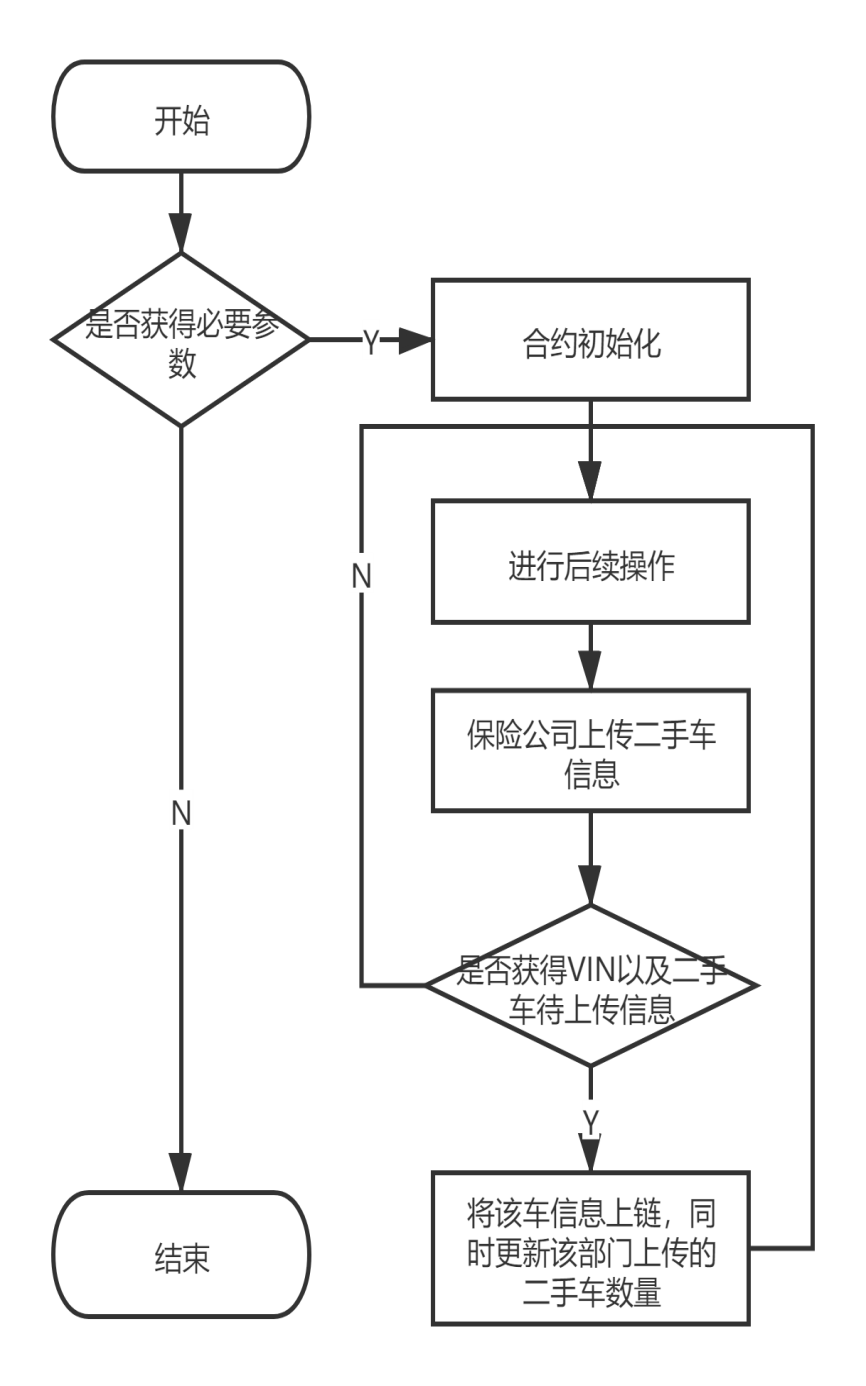
合约在调用其方法前需要进行初始化，在成功进行初始化后，等待维修店将二手车信息进行上链。上链操作需要提供车辆的VIN以及该车辆的具体信息。



**图3-7 维修店上传数据流程图**

**3.3.1.3保险公司上传数据的智能合约**

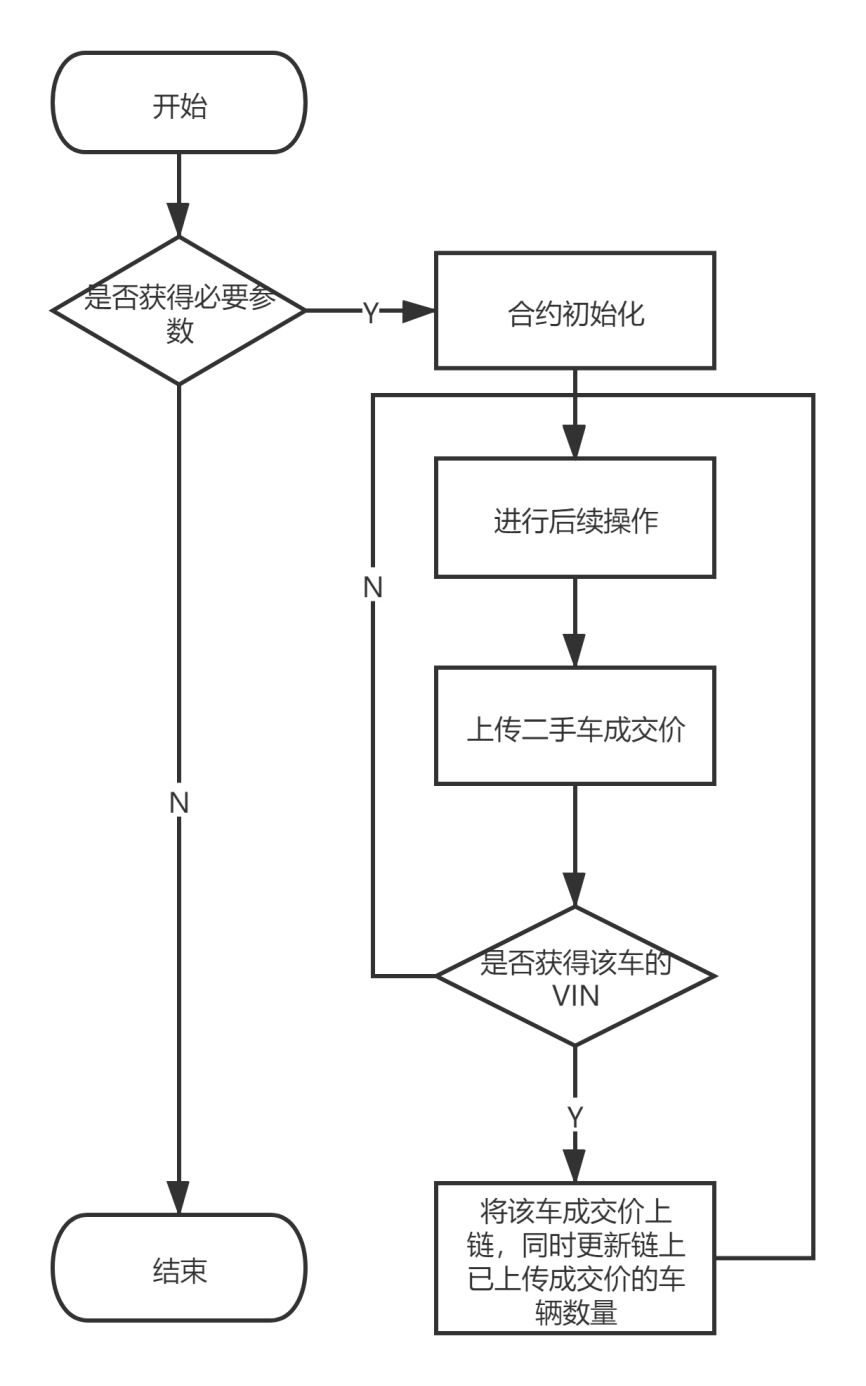
合约在调用其方法前需要进行初始化，在成功进行初始化后，等待保险公司将二手车信息进行上链。上链操作需要提供车辆的VIN以及该车辆的具体信息，如图3-8所示。



**图3-8 保险公司上传数据流程图**

**3.3.1.4上传成交价数据的智能合约**

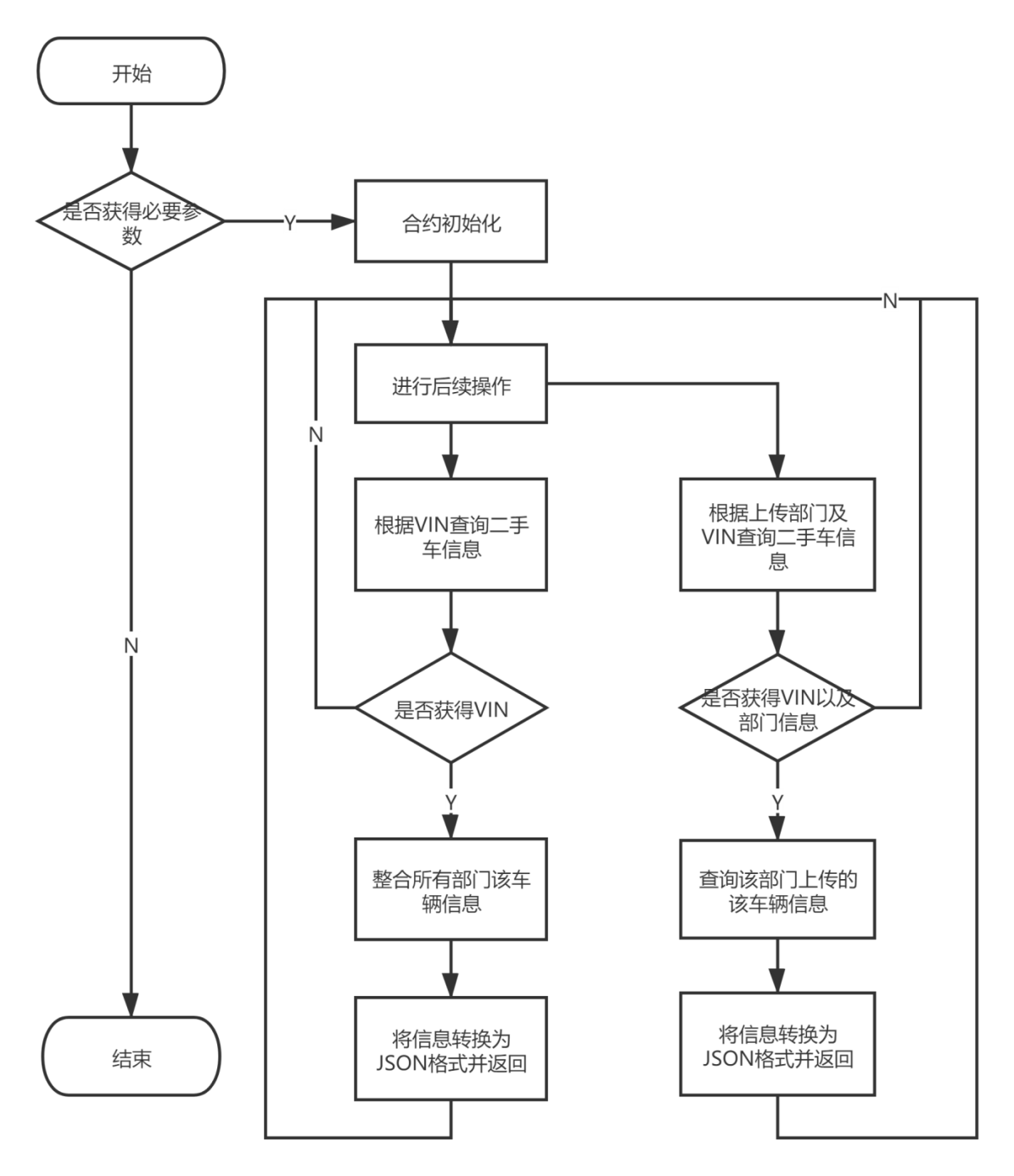
合约在调用其方法前需要进行初始化，在成功进行初始化后，等待本公司将客户反馈的成交价进行上链。上链操作需要提供车辆的VIN以及该车辆的成交价，如图3-9所示。



**图3-9上传成交价数据的流程图**

**3.3.1.5查询二手车详细信息的智能合约**

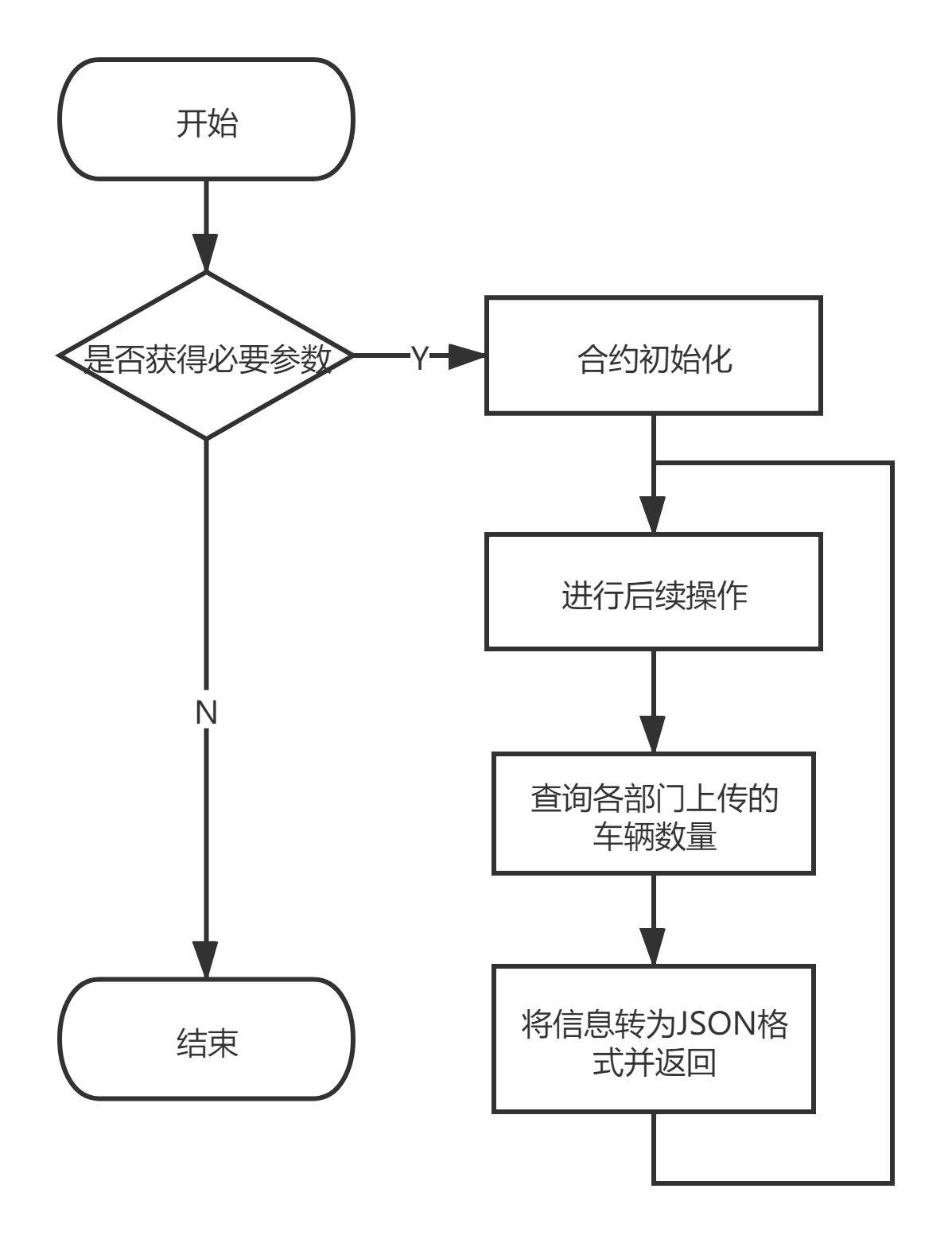
合约在调用其方法前需要进行初始化，在成功进行初始化后，等待客户进行查询操作。合约提供了两种方式进行查询，第一种方式根据车辆的VIN进行查询，第二种方式根据车辆的VIN以及上传该车辆的部门角色进行查询，如图3-10所示。



**图3-10 查询二手车信息的流程图**

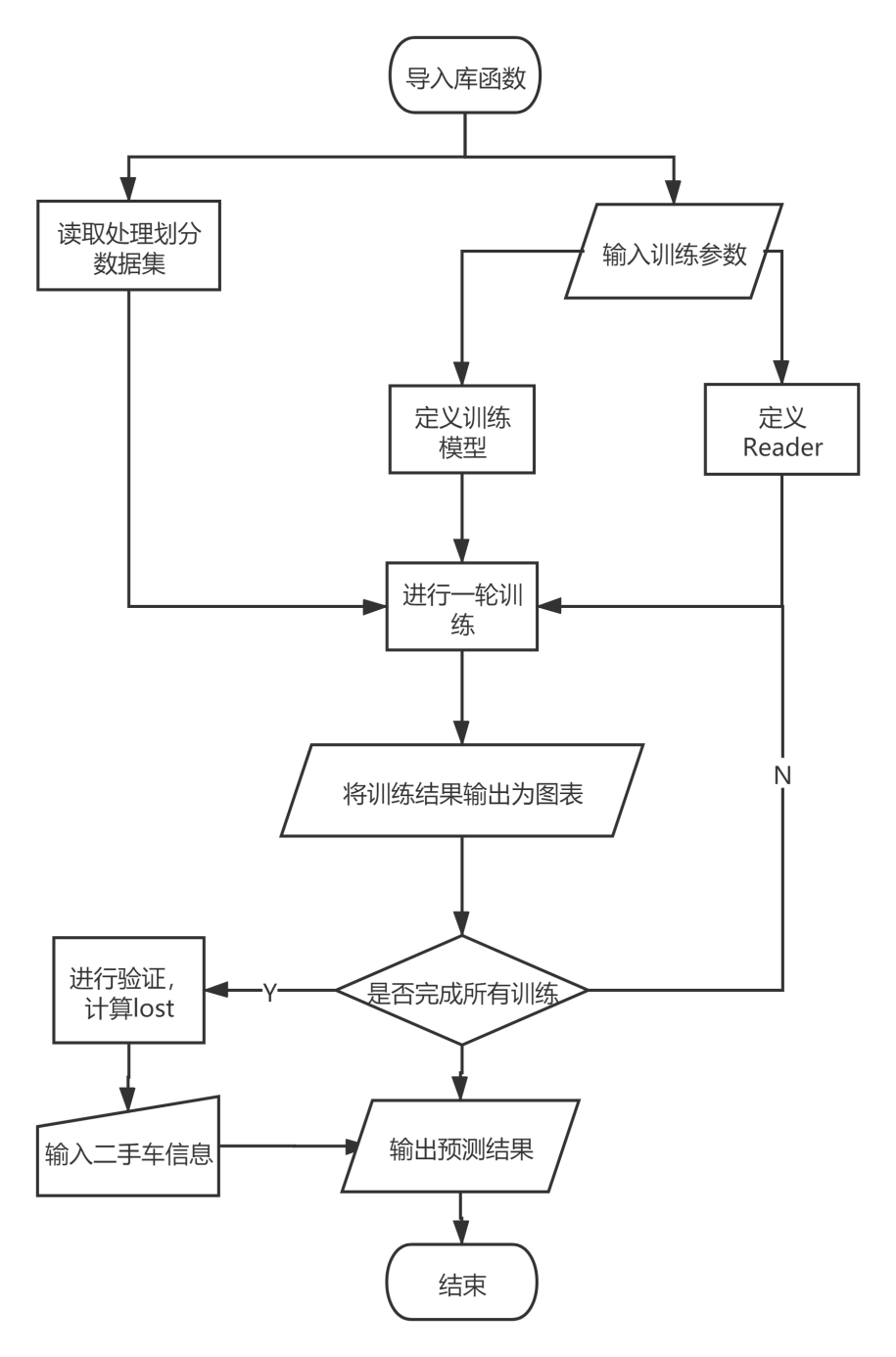
**3.3.1.6统计二手车数量的智能合约**

合约在调用其方法前需要进行初始化，在成功进行初始化后，等待客户进行查询操作。查询操作无需提供额外信息，查询结果将以JSON格式返回，如图3-11所示。

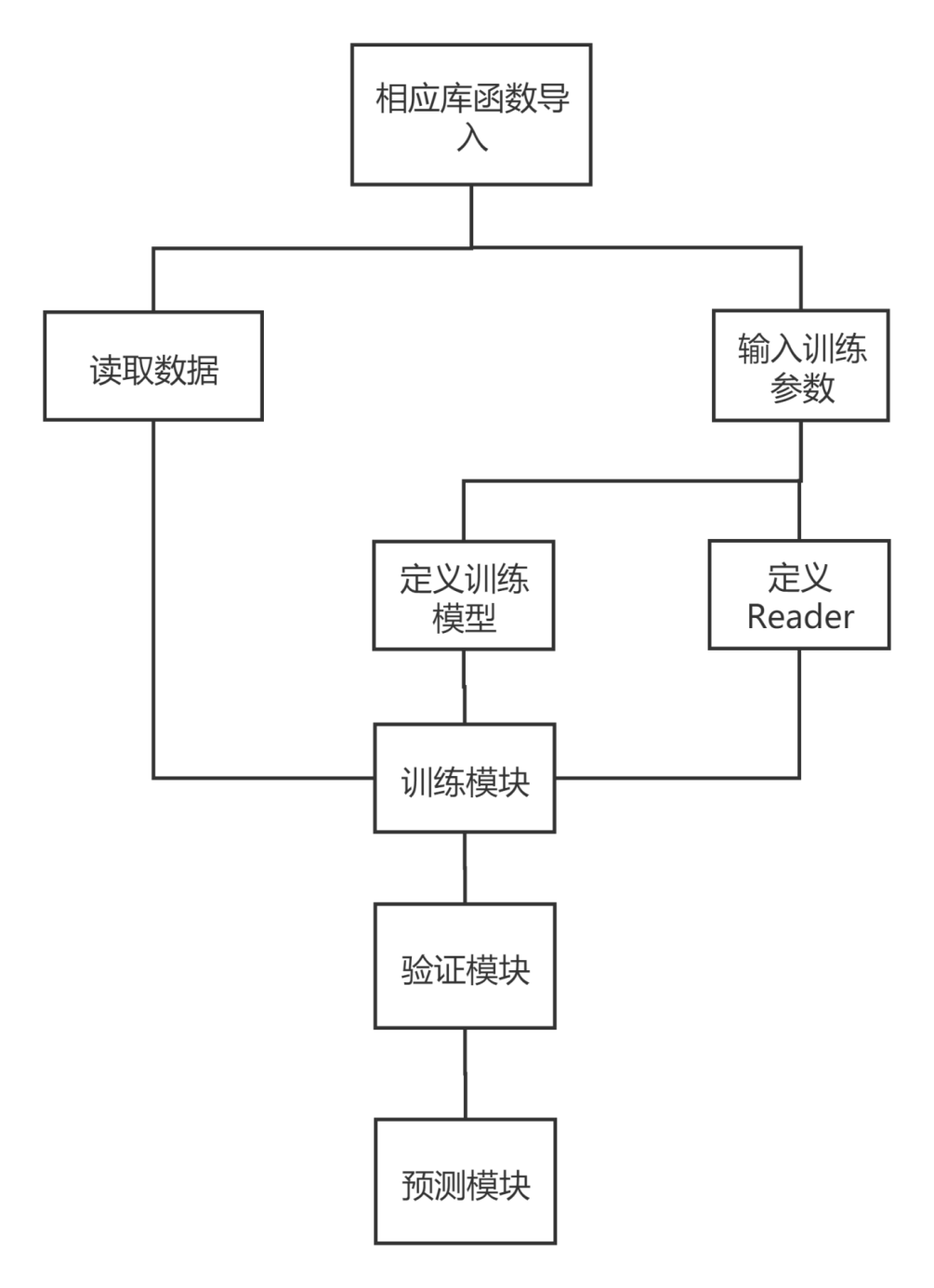


**图3-11 统计二手车数量的流程图**

**3.3.2 神经网络模块详细设计**



**图3-12 神经网络机器学习流程图**

****

**图3-13 HIPO图**

导入相关的库函数，并给予简短的别名，以便下层模块调用，如表3-2所示。

**表3-2 相应库函数导入模块表**

|  |  |
| --- | --- |
| 模块名：相应库函数导入 | IPO图编号：301 |
| 上层模块：无 | 下层模块：数据读取;训练参数输入 |
| 编制者：YLSenQAQ | |
| 输入：无 | 输出：无 |
| 算法说明：  import 库函数 as 别名 | |

将存放在CSV文件的数据集进行读取，进行归一化处理，再根据划分比例，将数据集划分为训练集和测试集，如表3-3所示。

**表3-3 数据读取模块表**

|  |  |
| --- | --- |
| 模块名：数据读取 | IPO图编号：302 |
| 上层模块：相应库函数导入 | 下层模块：训练模块 |
| 编制者：YLSenQAQ | |
| 输入：数据集地址 | 输出：处理划分好的数据集 |
| 算法说明：  #读取数据  data = pd.read\_csv(datapath)  #归一化处理  data.iloc[:, [i]] = (data.iloc[:, [i]] - minimums[i])/ (maximums[i] - minimums[i])  ratio = 0.8 #设定训练集和测试集比例  offset = int(data\_3.shape[0] \* ratio) #划分训练集和测试集 | |

对模型进行调参，如表3-4所示。

**表3-4 训练参数输入表**

|  |  |
| --- | --- |
| 模块名：训练参数输入 | IPO图编号：303 |
| 上层模块：相应库函数导入 | 下层模块：训练模型定义，Reader定义 |
| 编制者：YLSenQAQ | |
| 输入：相应参数 | 输出：为训练模型通过训练参数 |
| 算法说明：  params\_dirname = "mycar.model" #定义模型保存路径  BATCH\_SIZE = 100 #定义每次读取数据量大小  num\_epochs = 40 #训练轮次  learning\_rate = 0.001 #学习率  use\_cuda = True  #是否使用显卡进行训练 | |

对数据读取器进行定义，以备训练模块调用，如表3-5所示。

**表3-5 Reader定义模块表**

|  |  |
| --- | --- |
| 模块名：Reader定义 | IPO图编号：304 |
| 上层模块：训练参数输入 | 下层模块：训练模块 |
| 编制者：YLSenQAQ | |
| 输入：相应训练参数 | 输出：数据读取器 |
| 算法说明：  def read\_data(data\_set) #定义读取器  def train\_test(executor, program, reader, feeder, fetch\_list)  #定义训练数据读取器 | |

对神经网络进行定义，以备训练模块调用，如表3-6所示。

**表3-6 定义训练模型模块表**

|  |  |
| --- | --- |
| 模块名：定义训练模型 | IPO图编号：305 |
| 上层模块：输入训练参数 | 下层模块：训练模块 |
| 编制者：YLSenQAQ | |
| 输入：无 | 输出：训练模型 |
| **算法说明：**  # fluid.layers.data表示数据层x = fluid.layers.data(name='x', shape=[13], dtype='float32') y = fluid.layers.data(name='y', shape=[1], dtype='float32') #隐藏层 fluid.layers.fc表示全连接层 y\_1 = fluid.layers.fc(input=x, size=4, act=None, bias\_attr=True) #输出层 y\_predict = fluid.layers.fc(input=y\_1, size=1, act=None, bias\_attr=True) # 定义损失函数为均方差损失函数,并且求平均损失 loss = fluid.layers.square\_error\_cost(input=y\_predict, label=y)  # 定义损失函数为均方差损失函数,并且求平均损失  loss = fluid.layers.square\_error\_cost(input=y\_predict, label=y)  # 定义执行器  exe = fluid.Executor(place)  # 定义优化器  sgd\_optimizer = fluid.optimizer.SGDOptimizer(learning\_rate)  #随机梯度下降算法 | |

根据定义好的reader和神经网络模型，读取数据集进行训练，并打印结果，如表3-7所示。

**表3-7 训练模块表**

|  |  |
| --- | --- |
| 模块名：训练模块 | IPO图编号：306 |
| 上层模块：训练模型定义 | 下层模块：验证模块 |
| 编制者：YLSenQAQ | |
| 输入：训练集 测试集 | 输出：机器学习生成的model |
| 算法说明：  feeder = fluid.DataFeeder(place=place, feed\_list=[x, y] #数据投食器  exe.run(startup\_program) #创建训练用的executor  plot\_prompt.plot() #画出阶段训练结果  #保存训练模型 fluid.io.save\_inference\_model(params\_dirname, ['x'], [y\_predict], exe) | |

读取训练好的模型，根据校验集进行模型泛用性进行测试，得到损失率，如表3-8所示。

**表3-8 验证模块表**

|  |  |
| --- | --- |
| 模块名：验证模块 | IPO图编号：307 |
| 上层模块：训练模块 | 下层模块：预测模块 |
| 编制者：YLSenQAQ | |
| 输入：验证集 | 输出：训练模型的偏差率 |
| 算法说明：  #创建推测用的executor  infer\_exe = fluid.Executor(place)  #读取校验集  datatest = pd.read\_csv(testdatapath)  #从指定目录中加载 预测用的model(inference model)  [inference\_program,                             #预测模型  feed\_target\_names,                             #提供数据的变量名称  fetch\_targets] = fluid.io.load\_inference\_model(   #fetch\_targets: 推断结果                                      params\_dirname,    #模型路径                                      infer\_exe)         #预测用executor  #得到预测结果 results = infer\_exe.run(inference\_program,   feed={feed\_target\_names[0]: np.array(tensor\_x)}, #喂入要预测的x值  fetch\_list=fetch\_targets) #预测值  #输出平均偏差  print("±△=",mean\_lost) | |

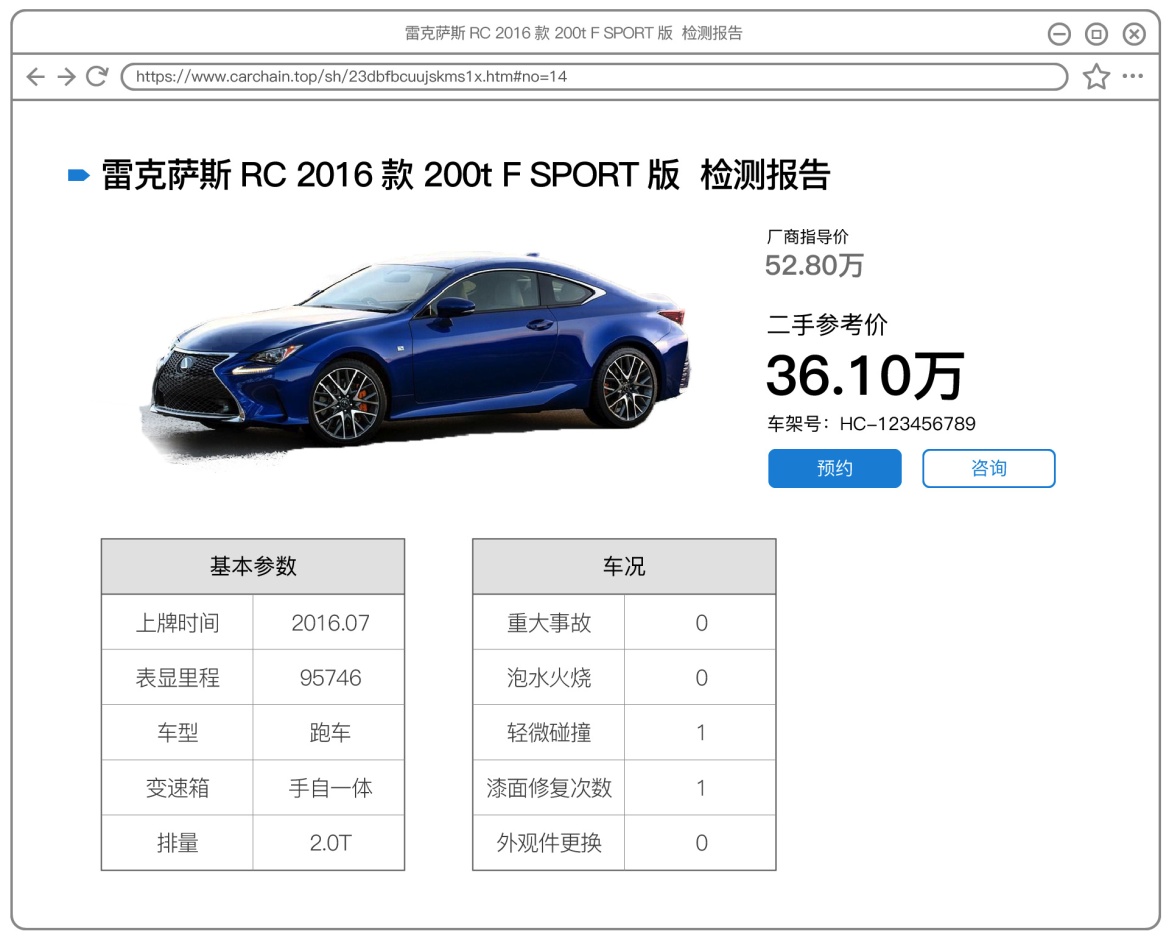
根据训练得到的模型，输入要预测价格的二手车数据，得到预测值，

如表3-9所示。

**表3-9 预测模块表**

|  |  |
| --- | --- |
| 模块名：预测模块 | IPO图编号：308 |
| 上层模块：验证模块 | 下层模块：无 |
| 编制者：YLSenQAQ | |
| 输入：需要预测价格的二手车信息 | 输出：估价 |
| 算法说明：  #填入要预测的二手车的数据 raw\_x = [76,89908,2,1,2,216,2,0,0,2,2,0,0]  #对数据进行归一化处理 for i in range(len(raw\_x)):    tensor\_x.append((raw\_x[i] - minimums[i])/(maximums[i]-minimums[i])) #进行预测 results = infer\_exe.run(inference\_program,             #喂入要预测的x值                 feed={feed\_target\_names[0]: np.array(tensor\_x)},          fetch\_list=fetch\_targets)  #得到推测结果    #输出预测结果 print("预测：现价=（%.4f±%.4f）\*原价"%(val2,mean\_lost)) | |

### 3.4界面详细设计



**图3-14 用户界面图（车辆信息详情）**



**图3-13 用户界面图（车辆信息概览）**

## 4数据库系统设计

### 4.1设计依据

平台数据存储有两种形式，一是全量数据存于区块链上；二是分步存储方式，即链上仅存储数据的摘要，数据内容本身则选择存于数据库或IPFS中。本方案采用链上存全量数据的方式，原因如下：初始数据量较小。链上数据主要包括潜在售卖二手车的属性信息，以及保险方、交警方信息。

按2019年一季度的数据，福建省交易二手车12万辆，按每每季度10K 数据量算，需 12.5G 存储量，全年则为50G。

随着业务的开展，如果以后不适合继续将全量数据在链上存储时，可通过修改业务逻辑和智能合约，将数据分别存储。同时也可以将原先链上的历史数据也存储于分布式数据库中。

### 4.2数据库逻辑结构

详细列出所使用的数据结构中每个数据项、记录和文件的标识、定义、长度及它们之间的相互关系。此节内容为数据库设计的主要部分。

本项目采用结构体存储二手车的相关信息，在每个合约中定义有各自的结构体，每个结构体的具体项及相关说明展示如下：

在查询二手车详细信息的智能合约中，定义了一个名为CarStruct的结构体，该结构体中包含车管所、维修店以及保险公司上传的二手车所有信息和二手车的最终成交价，如表4-1所示。

**表4-1创世块变量表**

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 备注 |
| OriginalPrice | float64 | 车辆原价 |
| SellingPrice | float64 | 车辆报价 |
| SalePrice | float64 | 车辆成交价 |
| EngineNumber | string | 发动机号 |
| Time | int | 上牌时间 |
| Mileage | float64 | 里程数 |
| WheelBase | string | 轴距 |
| Manufacture | int | 车辆厂商 |
| GearboxType | int | 变速类型 |
| CarType | int | 汽车类型 |
| EmissionStandard | int | 排放标准 |
| MaxPs | int | 最大马力 |
| PowerSunroof | bool | 电动天窗，true代表有电动天窗，false代表没有 |
| PanoramicSunroof | bool | 全景天窗，true代表有全景天窗，false代表没有 |
| GPS | bool | GPS导航，true代表有GPS导航，false代表没有 |
| ReversingRadar | bool | 倒车雷达，true代表有倒车雷达，false代表没有 |
| ReversingImageSystem | bool | 倒车影像系统，true代表有倒车影像系统，false代表没有 |
| LeatherSeat | bool | 真皮座椅，true代表有真皮座椅，false代表没有 |
| MajorMalfunction | bool | 重大事故，true代表有重大事故，false代表没有 |
| BubbleAndBurn | bool | 泡水火烧，true代表有泡水火烧，false代表没有 |
| SlightCollision | bool | 轻微碰撞，true代表有轻微碰撞，false代表没有 |
| PaintRepairTime | int | 漆面修复次数 |
| SheetmetalRepairTime | int | 钣金修复次数 |
| AppearancePartsReplaceTime | int | 外观件更换次数 |
| TransferTime | int | 过户次数 |
| DMVCarDescribe | interface{} | 车管所详细描述 |
| RepairShopDescribe | interface{} | 维修店详细描述 |
| InsuranceCompanyDescribe | interface{} | 保险公司详细描述 |

在车管所上传数据的智能合约中，定义了一个名为CarStruct的结构体，该结构体中包含车管所上传的二手车信息，如表4-2所示。

**表4-2 车管所上传数据变量表**

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 备注 |
| Manufacture | int | 上牌时间 |
| WheelBase | string | 轴距 |
| Manufacture | int | 车辆厂商 |
| GearboxType | int | 变速类型 |
| CarType | int | 汽车类型 |
| EmissionStr | int | 排放标准 |
| MaxPs | int | 最大马力 |
| TransferTime | int | 过户次数 |
| MajorMalfunction | bool | 重大事故，true代表有重大事故，false代表没有 |
| BubbleAndBurn | bool | 泡水火烧，true代表有泡水火烧，false代表没有 |
| EngineNumber | string | 发动机号 |
| DMVDescribe | interface{} | 详细描述 |

在维修店上传数据的智能合约中，定义了一个名为CarStruct的结构体，该结构体中包含维修店上传的二手车信息，如表4-3所示。

**表4-3 维修店上传数据变量表**

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 备注 |
| Mileage | float64 | 里程数 |
| PowerSunroof | bool | 电动天窗，true代表有电动天窗，false代表没有 |
| PanoramicSunroof | bool | 全景天窗，true代表有全景天窗，false代表没有 |
| GPS | bool | GPS导航，true代表有GPS导航，false代表没有 |
| ReversingRadar | bool | 倒车雷达，true代表有倒车雷达，false代表没有 |
| ReversingImageSystem | bool | 倒车影像系统，true代表有倒车影像系统，false代表没有 |
| LeatherSeat | bool | 真皮座椅，true代表有真皮座椅，false代表没有 |
| SlightCollision | bool | 轻微碰撞，true代表有轻微碰撞，false代表没有 |
| PaintRepairTime | int | 漆面修复次数 |
| SheetmetalRepairTime | int | 钣金修复次数 |
| AppearancePartsReplacementTime | int | 外观件更换次数 |
| RepairShopDescribe | interface{} |  |

在保险公司上传数据的智能合约中，定义了一个名为CarStruct的结构体，该结构体中包含保险公司上传的二手车信息，如表4-4所示。

**表4-4保险公司上传数据变量表**

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 备注 |
| SellingPrice | float64 | 报价 |
| InsuranceCompanyDescribe | interface{} | 详细描述 |

在上传成交价数据的智能合约中，定义了一个名为CarStruct的结构体，该结构体中包含车辆的成交价，如表4-5所示。

**表4-5本公司上传成交价变量表**

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 备注 |
| SalePrice | float64 | 成交价 |

## 5智能合约编码设计

### 5.1智能合约代码结构设计

源码文件均存放在src文件夹下，具体目录的设计结构如下：

（1）DMVUpload文件夹下存放车管所上传数据的智能合约，DMV\_upload.go即为该智能合约的源码文件。

（2）repairShopUpload文件夹下存放维修店上传数据的智能合约，repairShop\_upload.go即为该智能合约的源码文件。

（3）insuranceCompanyUpload文件夹下存放保险公司上传数据的智能合约，insuranceCompany\_upload.go即为该智能合约的源码文件。

（4）salePriceUpload文件夹下存放上传成交价数据的智能合约，salePrice\_upload.go即为该智能合约的源码文件。

（5）enquire文件夹下存放查询二手车信息的智能合约，enquire.go即为该智能合约的源码文件。

（6）statistics文件夹下存放统计二手车数量的智能合约，statistics.go即为该智能合约的源码文件。

### 5.2智能合约代码编制

本次项目依据功能需求总共设计了六个智能合约，智能合约采用的系统架构为百度超级链xuperchain，每个合约均采用go语言进行合约代码的编写。

### 5.3智能合约使用

首先依据2.3内容配置项目运行的环境，依照XuperChain官方部署文档下载代码并进行编译，编译完成后进入到output目录，代码样例如下：

cd output

在output目录下创建百度超级链并在后台启动xuperchain节点，代码样例如下：

./xchain-cli createChain

nohup ./xchain &

创建合约账号并往账号里转入充足的测试资源,代码样例如下：

./xchain-cli account new --account 0123456789123456 --fee 2000

./xchain-cli transfer --to XC0123456789123456@xuper --amount 100000000

本项目已写好Makefile文件，进行智能合约的编译时只需进入智能合约源码文件夹，在src文件夹下执行make命令即可，若编译成功，将会在该目录下生成对应的wasm文件，代码样例如下：

make

进入output目录下进行合约的部署，代码样例如下：

./xchain-cli wasm deploy --account XC0123456789123456@xuper --cname contract\_name -H 127.0.1.1:37105 -m ./contract\_name.wasm --arg '{"creator":"xchain"}'--output tx\_contract\_name.out --runtime go --fee 10000

依据功能需求进行合约相应方法的调用。

车管所上传二手车信息代码样例如下：

./xchain-cli wasm invoke -a '{"VIN" :"LSGGA53WXBH292533","DMV\_INFO" :{"Time" :1619616395,

"WheelBase" :"1.765","Manufacture" :1,"GearboxType" :1,"CarType" :1 ,"EmissionStandard" :1,"MaxPs" :585,"TransferTime" :3,"MajorMalfunction" :true,"BubbleAndBurn" :false,"EngineNumber" :"35100-23701","DMVDescribe" :""}' --method UploadByDMV --fee 110000 contract\_DMV

保险公司上传二手车信息代码样例如下：

./xchain-cli wasm invoke -a '{"VIN" :"LSGGA53WXBH292533","InsuranceCompany\_INFO" :{ "SellingPrice" :110000,"InsuranceCompanyDescribe":""}' --method UploadByInsuranceCompany --fee 110085 contract\_InsuranceCompany

维修店上传二手车信息代码样例如下：

./xchain-cli wasm invoke -a '{"VIN" :"LSGGA53WXBH292533","RepairShop\_INFO" :{"Mileage" :25.3,"PowerSunroof" :true,"PanoramicSunroof" :false,"GPS" :true,"ReversingRadar" :true,"ReversingImageSystem" :true,"LeatherSeat" :true,"SlightCollision" :false,"PaintRepairTime" :true,"SheetmetalRepairTime" :1,"AppearancePartsReplacementTime" :1,"RepairShopDescribe":""}' --method UploadByRepairShop --fee 110000 contract\_RepairShop

二手车成交价数据上传代码样例如下：

根据VIN查询二手车信息代码样例如下：

./xchain-cli wasm invoke -a '{"VIN" :"LSGGA53WXBH292533","LocalCompany\_INFO" :{"SalePrice":11333333}' --method UploadByLocalCompany --fee 110000 contract\_LocalCompany

查询各部门上传二手车数量代码样例如下：

./xchain-cli wasm invoke -a '{"VIN" :"LSGGA53WXBH292533"}' --method QueryCarByVIN --fee 110000 contract\_enquery

./xchain-cli wasm invoke --method Statistics --fee 110000 contract\_statistics

## 

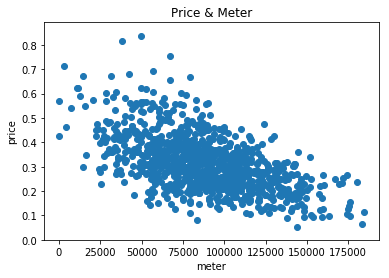
## 6神经网络价格模型编码设计

### 6.1需求分析

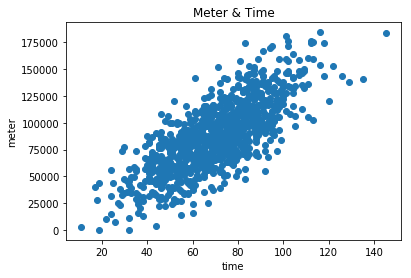
利用区块链其可追溯和难以篡改性，保证二手车数据的真实可信，因此可利用神经网络机器学习获得一个泛用性与准确度皆备的价格评估模型。借此我们可为客户提供可靠的价格评估服务，而我们对用户关于每次二手车的交易进行信息悬赏，形成公司和客户的互惠互利关系。根据收集的交易信息，我们将每次的交易价上链，并提取归化为可用于神经网络机器学习的数据集，由此区块链技术与神经网络学习技术可形成正反馈的闭环。

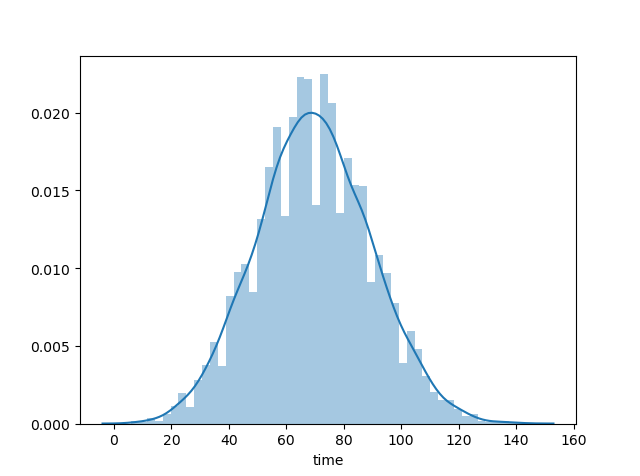
### 6.2数据集优化

根据现有的数据集，使用相关软件，市场调研以及图表分析，从多达二十多个的数据项中筛选出十三个特征项。

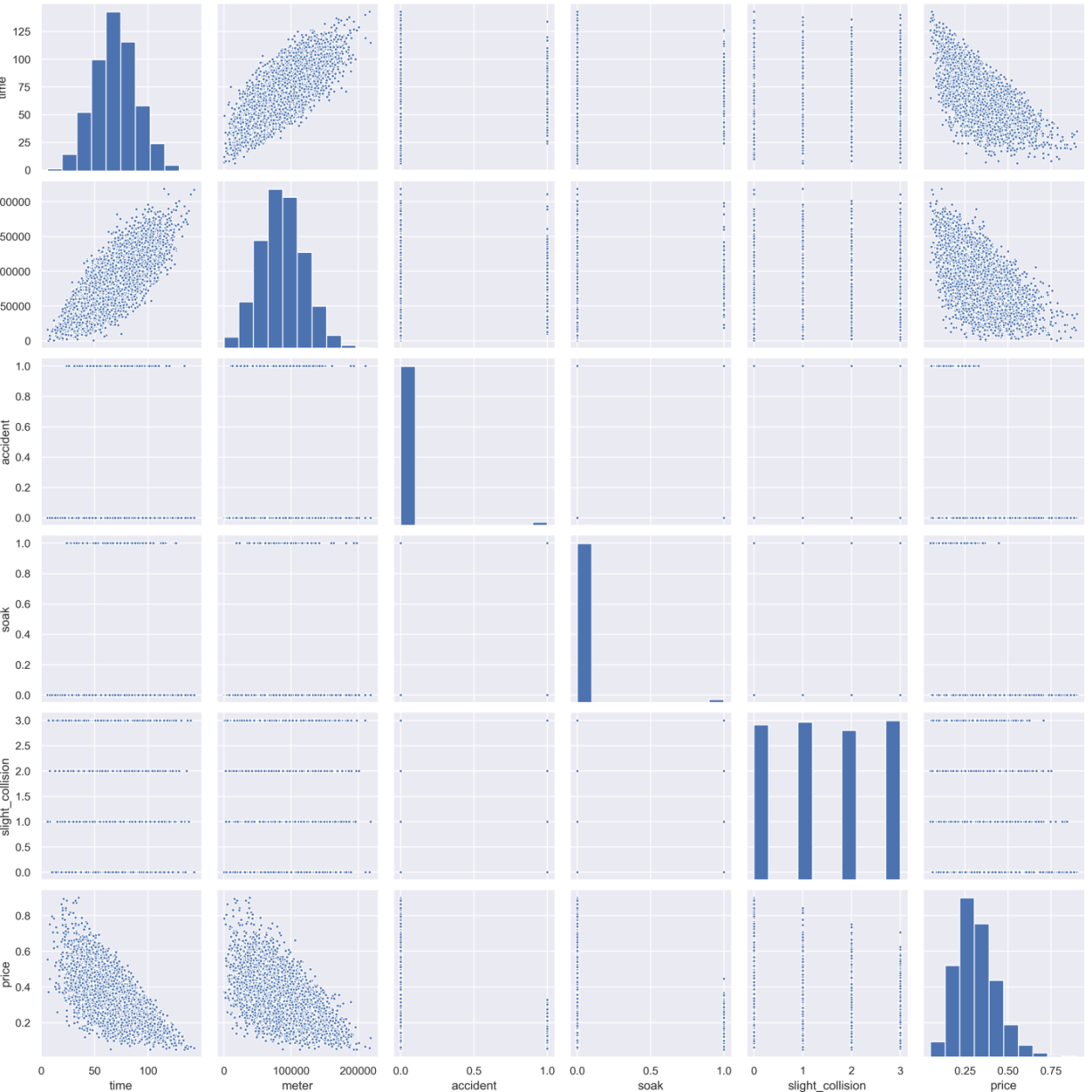


**图6-1 里程数与价格的关系分布图**



**图6-2 里程数与时间的关系分布图**

**图6-3 时间分布直方图**

****

**图6-4 各项数据相关性分析图**

### 6.3建模调参

根据相关资料以及查阅相关论文可知：二手车价格与其车辆相关信息近似为线性关系。因此本项目采用线性回归模型。经典的线性回归模型主要用来预测一些存在着线性关系的数据集。回归模型可以理解为：存在一个点集，用一条曲线去拟合它分布的过程。如果拟合曲线是一条直线，则称为线性回归。如果是一条二次曲线，则被称为二次回归。

该模型根据二手车的各种属性进行计算，如车辆品牌，车辆公里数，车辆购买时长，车辆是否发生过事故等，将这些参数导入框架后，得出各参数对于二手车价格影响的权重，进而构建出价格计算的线性模型。在构建模型的同时，将计算值与真实值进行比较得到均方误差，即损失函数，不断减小差值，从而让价格的预测值不断逼近真实值。最终目标是让价格的损失函数的最小值达到5%。

### 6.4神经网络详细设计

对于二手车价格预估的线性模型，假设属性和二手车价格之间的关系可以被属性间的线性组合描述：

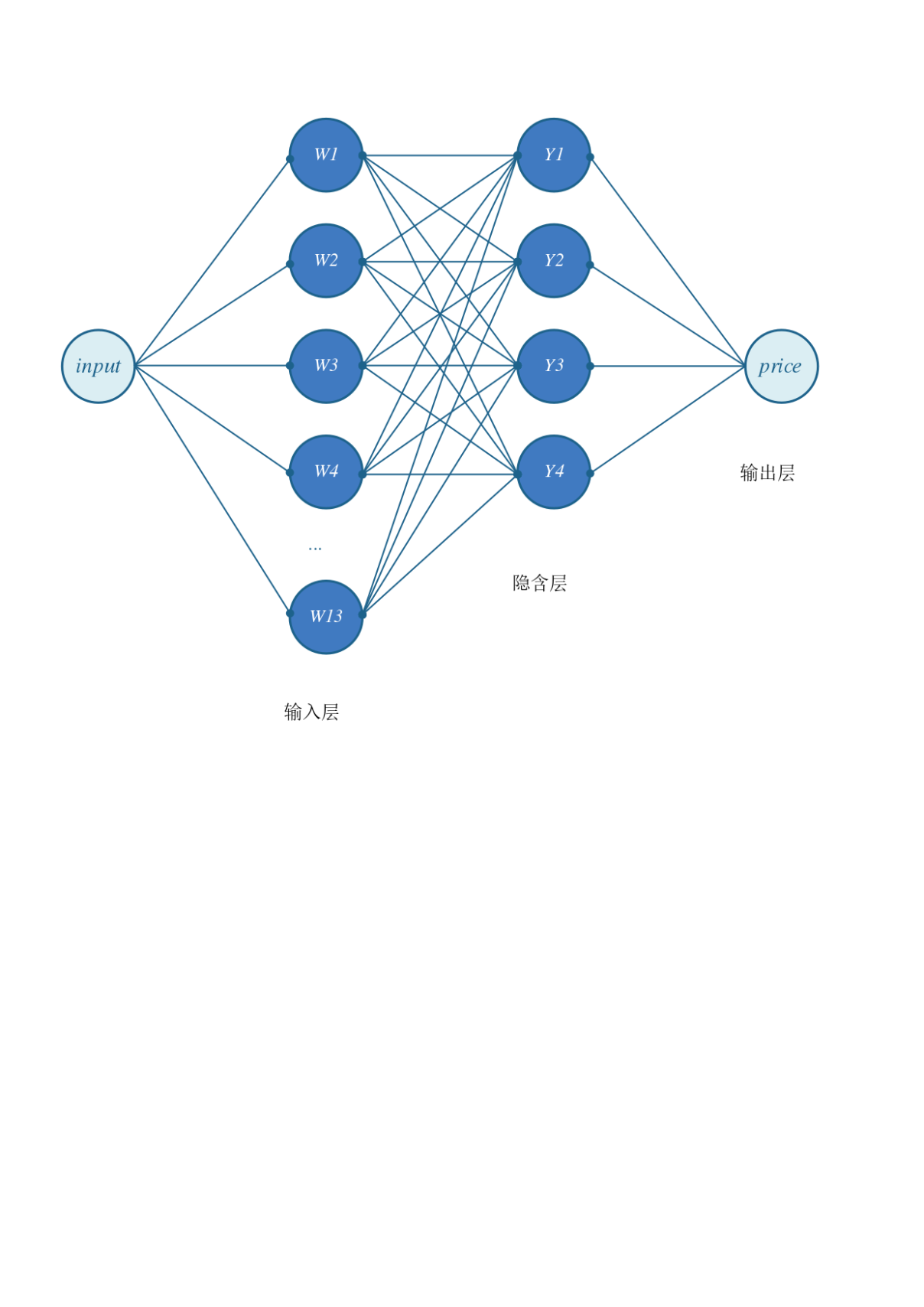
  
其中：

[]

[]

[]

该模型是一个多层感知器，包括4层：一个输入层、两个隐含层、一个输出层。如图6-5所示。



**图6-5 神经网络模型图**

**输入层：**将二手车各项数据输入给神经网络。

**隐藏层：**增加网络深度和复杂度，隐含层的节点数是可以调整的，节点数越多，神经网络表示能力越强，参数量也会增加。在该模型中，中间的隐含层为4的结构，通常隐含层会比输入层的尺寸小，以便对关键信息做抽象。

**输出层：**输出网络计算结果，输出层的节点数是固定的。在该模型中，模型的输出是回归一个数字，输出层的尺寸为1。

神经网络模型具体代码：

# 标签数据，fluid.layers.data表示数据层,输出类型为tensor

x = fluid.layers.data(name='x', shape=[13], dtype='float32')# 输入层

y = fluid.layers.data(name='y', shape=[1], dtype='float32')

#隐含层

y\_first = fluid.layers.fc(input=x, size=4, act=None, bias\_attr=True)

#输出层

y\_predict = fluid.layers.fc(input=y\_first, size=1, act=None, bias\_attr=True)

#采用梯度下降优化器

sgd\_optimizer = fluid.optimizer.SGDOptimizer(learning\_rate)