使用线性回归和人工神经网络拓扑分析基金增长情况

前言:

有的人认为基金的浮动是不固定的，无法用固定的套路去分析某一个基金的浮动趋势。

但是，我个人认为， 也许我们的确无法单独分析一个基金的浮动，但是如果我们可以将其他各种基金的浮动作为参考，来预测一种基金的趋势是可行的。 毕竟不同的市场之间肯定是存在着一定的关联的。 由于基金数据是按天为单位变化的， 我们甚至可以使用其他的同样是以天为单位变化的数据作为参考，分析基金变化。 这里， 我仅仅使用其他基金的数据作为参考， 初步尝试分析一个特定基金变化。 我获取数据的来源是 京东金融 :https://jr.jd.com/ , 使用python爬虫技术， 爬取数据。 打算分析的基金是 ‘华安动态灵活配置混合型证券投资基金(040015)’。

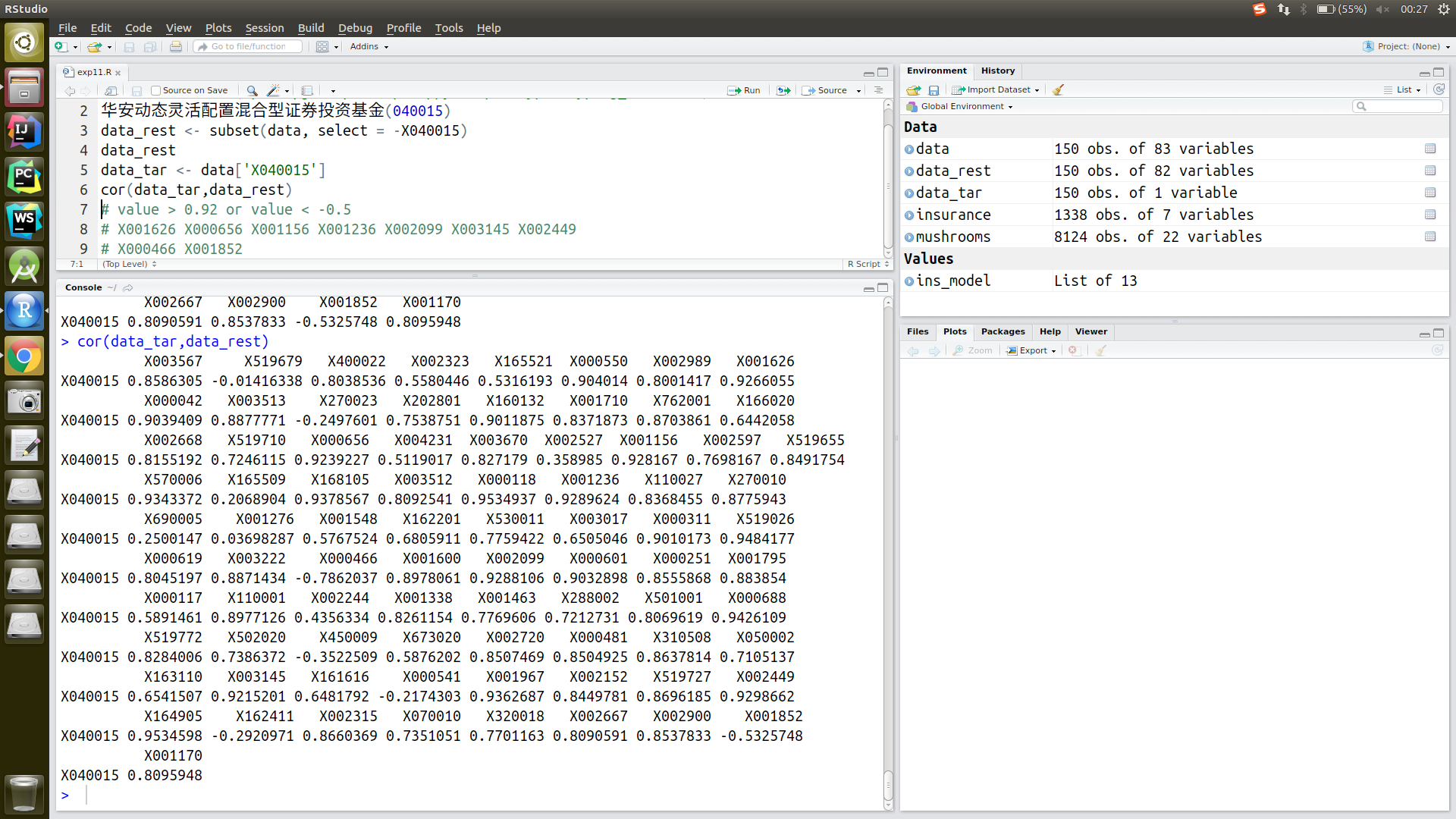
1. 收集数据:根据python爬虫，在京东金融上得到‘华安动态灵活配置混合型证券投资基金(040015)’的历史基金份额， 然后随机得到112份基金的历史份额。
2. 探索和准备数据: 将不同文件的数据合并， 每一列代表一个属性（即一种基金）的历史份额。但是在处理过程中发现，有些基金的某天的基金分为为NA, 可能是爬虫的过程中出现了失误或者是京东金融的网站服务器出了问题而导致的。 另外， 我对于数据的选取要求必须要有500条， 但是有的基金可能是最近才上市，所以爬到的条数低于500 条。 基于以上两点， 我最终从随机得到的112份基金中筛选出了48 份基金作为’040015号’基金的参考变量。 筛选的代码我是用python实现的（个人感觉python对于数据的操作还是比较方便的）， 我用R语言主要实现了数据分析（这才是R语言的长处）。
3. 基于数据训练模型:首先想到的是使用线性回归的方法训练数据得到一个合适的线性回归方程。 但是后来考虑到基金的涨幅在一定的情况下不是我们可以用理论来解释的， 所以在优化的时候使用人工网络拓扑进行建模分析。
4. .开始时， 对于040015 号基金 分析和其他48份基金的相关性

data<-read.csv('/home/feilong/Rstudy/analyse/base\_money.csv')

data\_rest <- subset(data, select = -X040015)

data\_tar <- data['X040015']

cor(data\_tar,data\_rest)



当且仅当他们的相关性的绝对值大于）0.8的时候将它们筛选出来作为最终的参考。

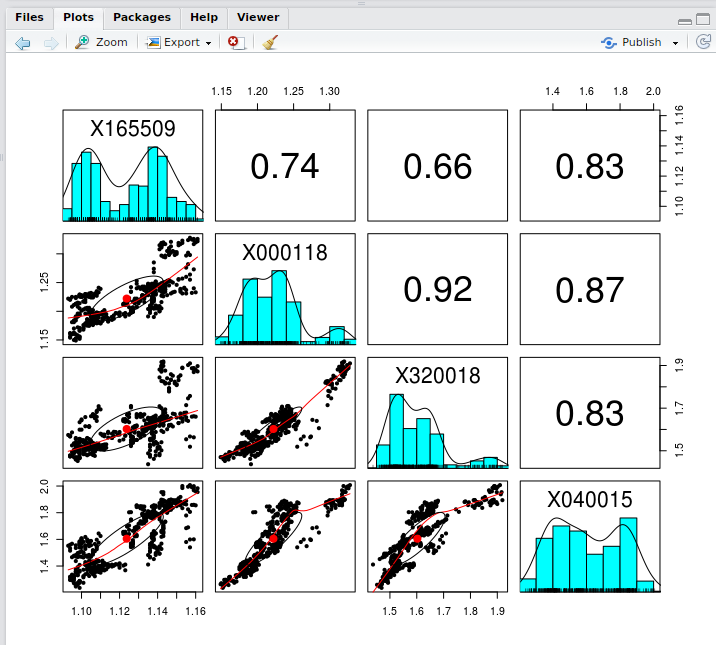
得到符合条件的参考基金有:X165509、X000118、X320018

b).考虑到基金的份额分布本来就在-5到+5之间， 比较集中， 就不考虑0~1化了。

c).初步建模时， 需要考虑因变量和自变量之间的关系，来保证建模的准确性和有效性(不能认为它们之间都是一次关系，可能存在有二次、三次、甚至log2关系)。 所以使用图形表示， 粗略判断它们之间呈现怎样的一种关系。

library(psych)

pairs.panels(data[c("X165509","X000118","X320018","X040015")])



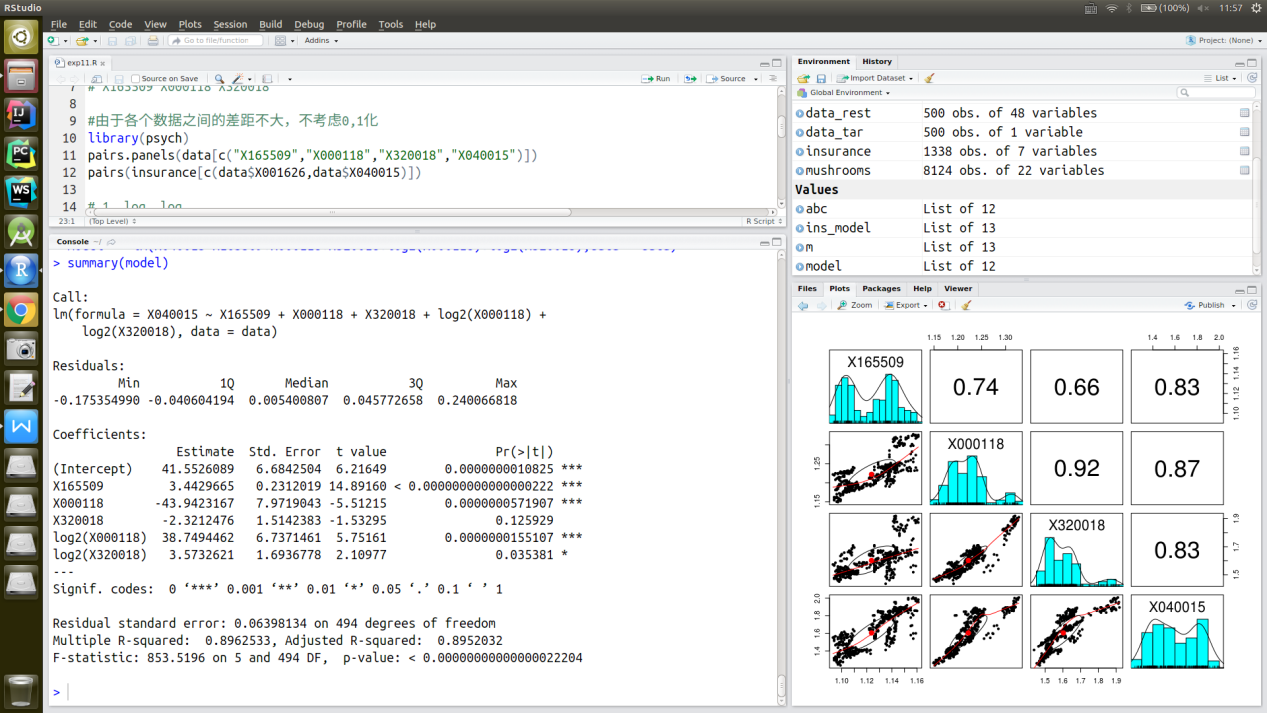
根据图形， 我初步判断出 X040015 和 X165509呈现一次关系， 和 X000118呈现log2对数关系, 和 X320018呈现log2对数关系。 （毕竟X040015和后两种呈现出的线性关系不是很强， 如果采用一次关系的话， 反而不妥。）

根据这个， 得到建立模型的公式:

ins\_model<-lm(X040015~X165509+X000118+X320018+log2(X000118)+log2(X320018) , data = data)

1. 评估模型的性能:

summary(ins\_model)



根据得到的多元R方值， 该模型可以适用于89.6%的情况。 说明预测还是满成功的，但是， 正如我前面所说的， 从单个方面来看， 基金的变化趋势， 的确是不可预测的， 所以， 对于无法从理论解释的数据分析， 我决定尝试着用人工网络拓扑的方法，看看能不能提高模型的性能，提高准确度。

1. 提高模型的性能:

尝试着使用人工网络拓扑分析如下:

a).先分出 训练数据 和 测试数据

data\_train <- data[131:500,]

data\_test <- data[1:130,]

b).建立数据关系公式

由于在人工网络拓扑中， 我们基于的是大量的数据， 并不要求数据之间的相关性的强弱（这也就是为什么我们通过网络拓扑得到的结果无法用理论分析的原因了）， 所以， 在这里， 我将剩下的所有48列数据全用来作为参考量

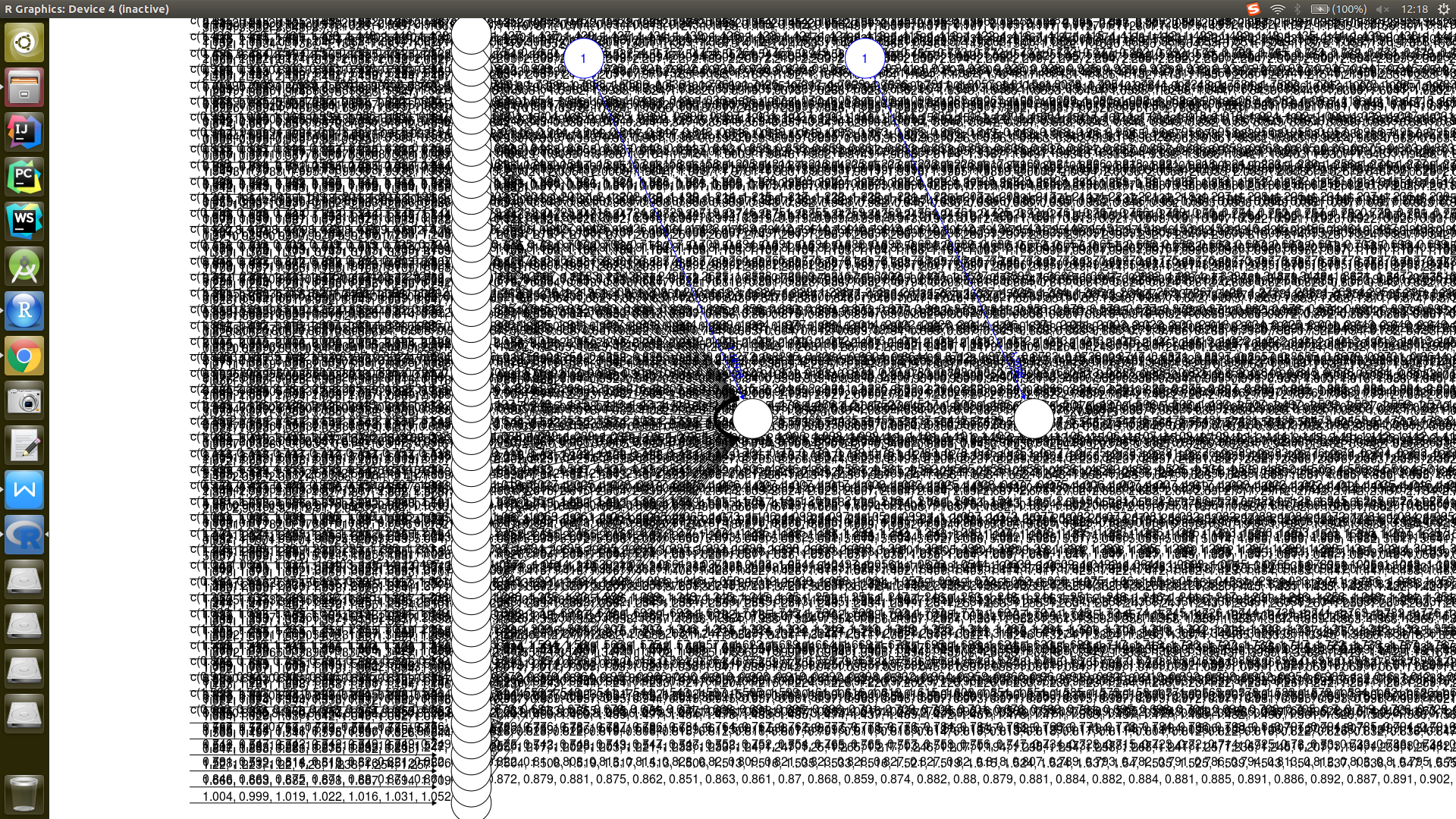
f <- as.formula(paste("X040015 ~", paste(data[!data %in% "X040015"], collapse ="+")))

这样就构造除了一个预测量和观测量之间的关系,这个公式是从http://dataunion.org/24729.html 这上面学到的， 很有用， 尤其是当在一个数据框中， 你有很多的观测量， 你不用手动一个一个的写（就像 A~a1+a2+a3.....+a1000）。

c).数据训练

m <- neuralnet(f, hidden = 1)

plot(m)

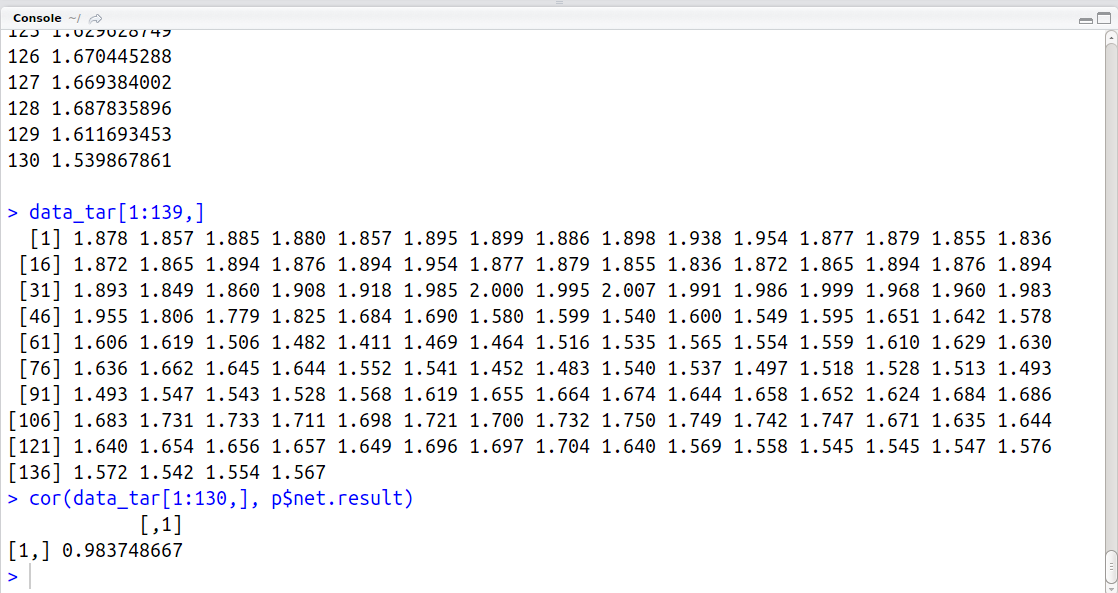


由于测试的数据条目太大， 这个拓扑图结构无法清晰的展示， 但是我们也可以看出这里的隐藏层只有一个。

d).性能测试

p <- compute(m, data\_dep[1:139,])

cor(data\_tar[1:130,], p$net.result)

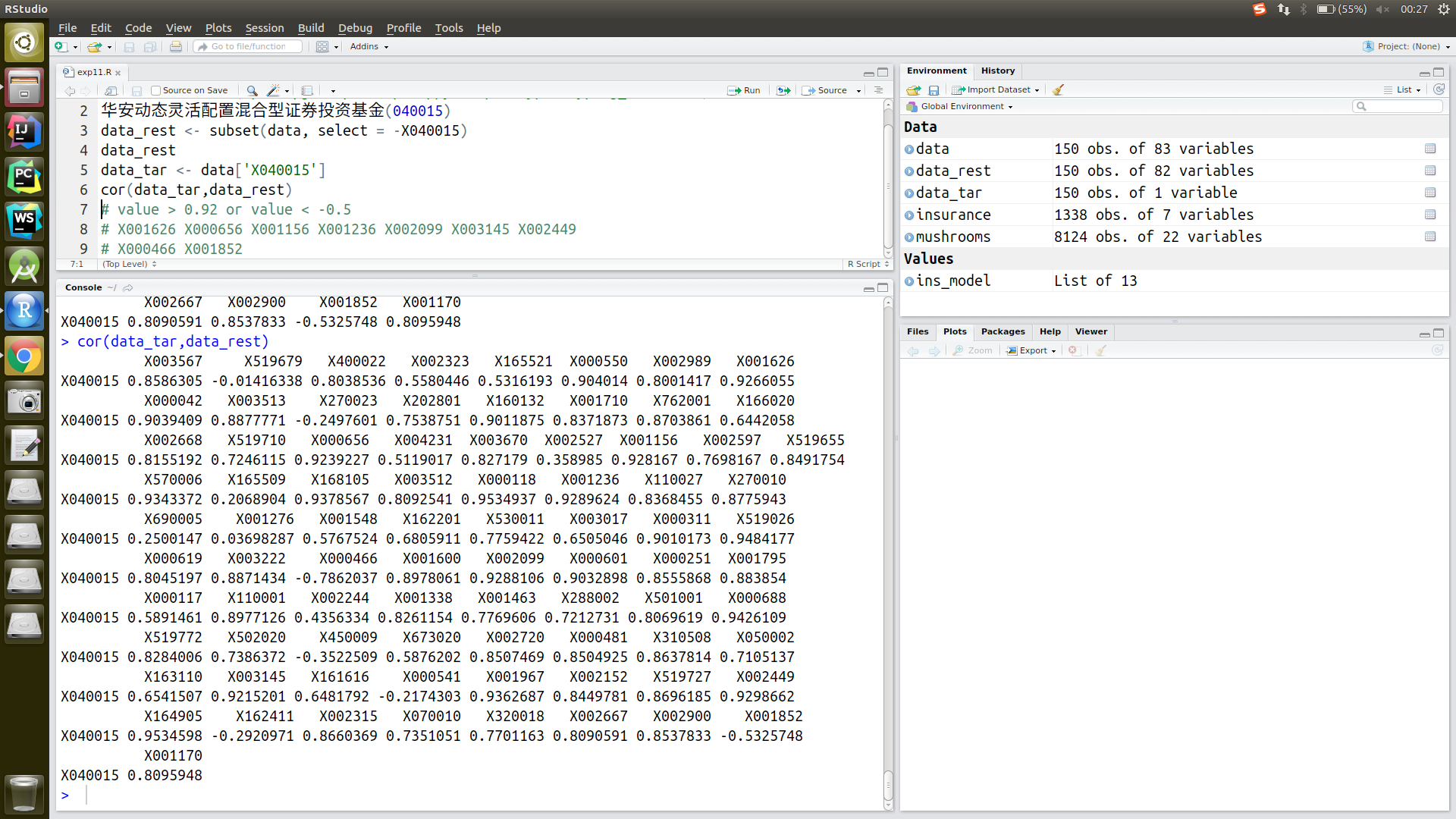


对比预测值和真实值之间的相关性，我惊讶的发现， 相关性居然达到了98.37%, 趋近于一直， 这个结果简直出乎我的意料(我还打算增加隐藏节点，看来是不必了)。

1. 进一步思考

我们最后通过人工网络拓扑结构的方法，成功构造出了一个相关性达到98.37%的一种模型，但是，这真的是成功的吗? 我后来仔细的考虑了一下， 我们在建模的时候， 是存在一个很大的问题的,那就是 过度拟合。 查阅资料 <https://www.zhihu.com/question/26898675> , 我发现在使用人工网络拓扑结构的时候， 不应该将所有的因素都无差别的考虑在内， 尤其是在数据量比较小的时候（每列只有500条, 测试数据每一列只有370条）, 这是造成过度拟合的主要因素。 于是， 我又重新使用人工网络拓扑， 再一次进行了分析。

这一次， 我同时考虑了数据间的相关性大小以及相关性的正负情况。



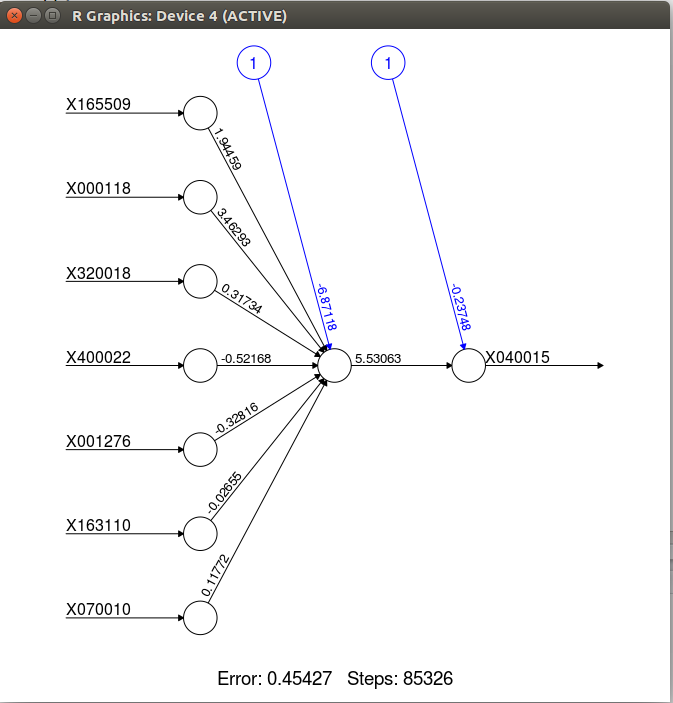
a).我分别筛选出:X165509, X000118, X320018 (它们的相关性>0.8)

以及 X400022, X001276, X163110, X070010 (它们的相关性<-0.5)

b).训练数据:

f<-X040015~X165509+X000118+X320018+X400022+X001276+X163110+X070010

m <- neuralnet(f, data= data\_train, hidden = 1)

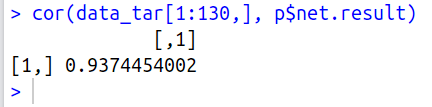


这一次构造出的图形就好看多了。 我在这里设置的隐藏节点还是只有一个， 根据得到的性能测试结果决定是否要增加隐藏层

c). 性能测试

p <- compute(m, data\_test[c("X165509", "X000118", "X320018","X400022", "X001276", "X163110", "X070010")])

cor(data\_tar[1:130,], p$net.result)

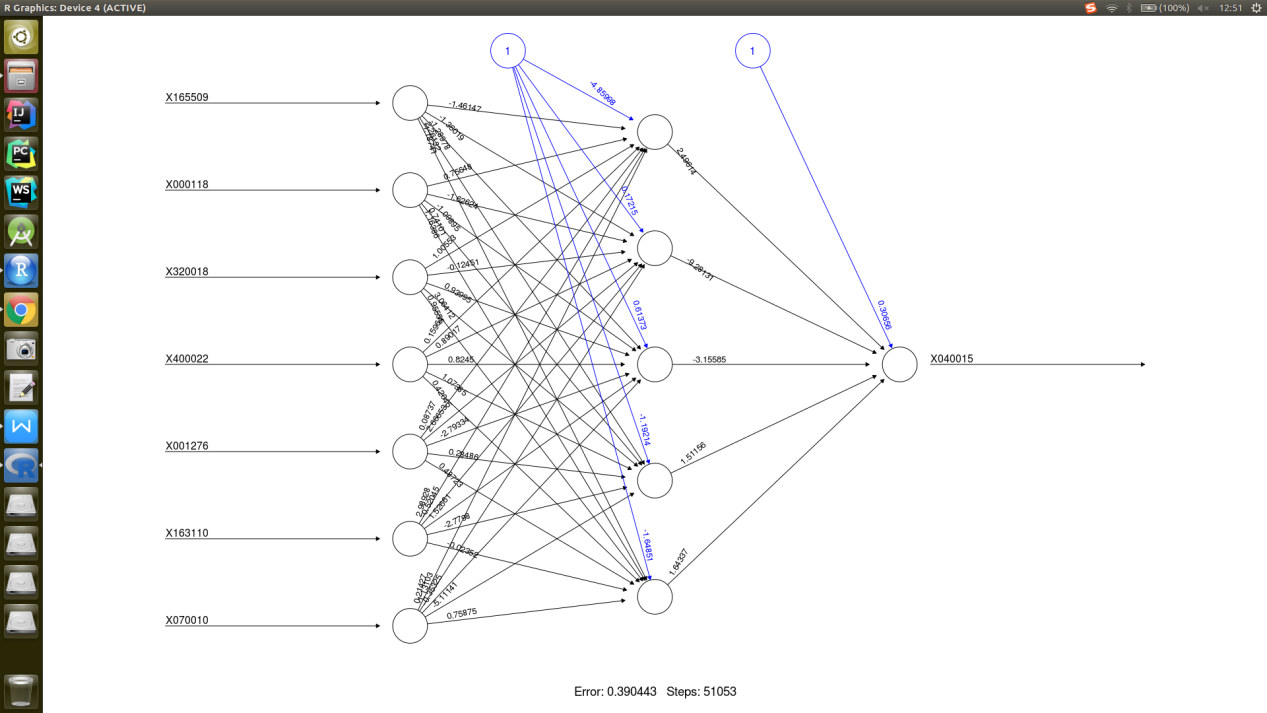


可以看出测试值和真实值之间的相关性还是到达93.7%的， 感觉比较满意。 如果我们增加了隐藏层， 性能可以提高多少呢？ 测试一下

d). 性能优化

m <- neuralnet(f, data= data\_train, hidden = 5)

plot(m)

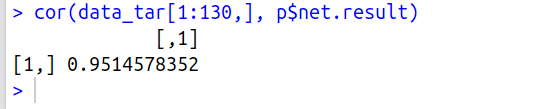


这里， 我们将隐藏节点增加到5个， 得到的拓扑结构如上.

性能分析:

p <- compute(m, data\_test[c("X165509", "X000118", "X320018","X400022", "X001276", "X163110", "X070010")])

cor(data\_tar[1:130,], p$net.result)



得到的预测值和真实值之间的相关性提升了3%左右， 还是很客观的， 而且还在一定程度上避免的过度拟合问题。

1. 感悟：

随着计算机处理速度越来越来快， 我感觉人工智能的时代即将到来。 大数据的分析可以帮我们完成很多以前感觉不可能完成的任务。 比如说 通过相关性研究， 判断某类事件之间是否存在一定的关联； 通过建立模型， 我们甚至可以预测某一类事件的发展趋势。 也许正如老师所说， 中国人民在古代就可能已经掌握基于数据的预测能力(周易算经、天演八卦)。

我们生活在一个充满大量数据的时代， 懂得如何利用这些数据， 也许就可以站的更高、看得更远。