171240510
马宇恒
2019年7月6日

人工智能行业薪资分析

一、选题背景

众所周知，发家致富就要做一名程序员，虽然头发掉的多，但是相应也拥有不错的薪资。所以，我马宇恒，家里穷，不怕苦，不怕累，不在乎工作环境，不在乎工作时长，就是要赚钱多，要成为一个程序员！那么程序员的工作具体情况是怎样的呢？什么样的工作条件工资较高？为此，我爬取了拉勾网上有关机器学习的所有招聘信息进行探究。

二、项目概述

项目爬取了机器学习相关的包括深度学习等八种岗位共3600个招聘岗位，包含岗位名称、工作地点、薪资范围、岗位要求、公司情况等信息。本文中所有的代码为蓝色字体，所有的输出结果为绿色字体。爬虫原始数据和清洗后的数据见附录。

三、实现及代码

1. 爬虫

```
library(stringr)
library(xml2)
library(RSelenium)
library(rvest)
###############################################设置浏览器
remDr = remoteDriver('localhost',4444L,browserName='chrome')
#url1 <- 'https://www.lagou.com/'
###############################################找到爬取的页面
url0 <- 'https://www.lagou.com/zhaopin/jiqixuexi/'
#remDr$open()
remDr$navigate(url0)
```

```
tpage <- remDr$getPageSource()
pageSource <- tpage[[1]]
web <- read_html(pageSource)
############################################找到总页数
#奇怪的一点是直接浏览网页时，使用safari打开网页有总共300个结果，但是chrome
就有450....
pgcttxt <- web %>% html_nodes('div.item_con_pager') %>%
html_nodes('div')%>%html_nodes('a:nth-child(5)')%>% html_text()
pgct = as.numeric(pgcttxt)
setwd('/Users/mayuheng/Desktop')
############################################初始化向量和数据框
name=salary=require=location=time=company=companysituation=companyintro
=tags=NULL
data=data.frame(name,location,salary,require,time,company,companysituation,co
mpanyintro,tags)
############################################开始循环
for(i in 1:pgct)
{
  url <- paste(url0,i,"/?filterOption=2",sep = '')
  web <- read_html(url)
  ############################################
  name <- c(name,web %>% html_nodes('div.position') %>%
            html_nodes('div.p_top') %>% html_nodes('a') %>%
            html_nodes('h3') %>%html_text())
  ############################################
  location <- c(location,web %>% html_nodes('div.position') %>%
                html_nodes('div.p_top') %>% html_nodes('a') %>%
                html_nodes('span') %>%html_nodes('em') %>%html_text())
  ############################################
  salary <- c(salary,web %>% html_nodes('div.position') %>%
              html_nodes('div.p_bot') %>% html_nodes('div') %>%
              html_nodes('span') %>%html_text())
  ############################################有一些数据格式混乱，
做简单处理
  require<- c(require, web %>% html_nodes(xpath="//li[@class]/div[1]/div[1]/
div[2]/div/text()")
            %>%html_text())
  require<-require[require!="\n"]
  require<-gsub(" ","",require)
  require<-gsub("\n","",require)
  ############################################
```

```
  time<- c(time,web %>% html_nodes('div.position') %>%
             html_nodes('div.p_top') %>% html_nodes('span') %>%html_text())
  time<-time[grepl("[0-9]",time)]
  ############################################
  company <- c(company,web %>% html_nodes('div.company') %>%
                 html_nodes('div.company_name') %>% html_nodes('a') %>%
                 html_text())
  ############################################
  companysituation <- c(companysituation,web %>% html_nodes('div.company')
%>%
                          html_nodes('div.industry') %>%
                          html_text())
  companysituation<-gsub("\n","",companysituation)
  companysituation<-gsub(" ","",companysituation)
  ############################################
  companyintro <- c(companyintro,web %>% html_nodes('div.li_b_r') %>%
                      html_text())
  ############################################
  tags<- c(tags, web %>% html_nodes("div.list_item_bot")
           %>% html_nodes("div.li_b_l")
           %>%html_text())
  tags<-gsub("\n","/",tags)
  tags<-gsub(" ","",tags)
  ############################################每一页的数据存储在
pracdata里
  pracdata<-
data.frame(name,location,salary,require,time,company,companysituation,compan
yintro,tags)
  print(pracdata)
  ############################################汇总进总数据框里
  data<-rbind(data,pracdata)
  ############################################清空向量

name=salary=require=location=time=company=companysituation=companyintro
=tags=NULL
  ############################################有时会遇到需要重新登
录的情况
  #if(###)login()
  #Sys.sleep(5)
  ############################################
  #等时间间隔的爬取会被服务器识别并要求登陆，此时需要调用登陆操作函数
```

```
  #由于我在实际操作中并没有用到，登陆的操作函数没有附在这
  #而随机时间的爬取并不会被识别，就无需调用
  ##############################################
  x1<-runif(1,3,10)
  Sys.sleep(x1)
}
##############################################关闭浏览器
remDr$closeWindow()
##############################################导出数据
write.table(data,file='data.txt')
write.csv(data,file=‘data.csv')
```

2. 数据清洗

```
library("lubridate")
library("VIM")
library("mice")
############简单而笨拙的导入数据
shenduxuexidata<-read.csv('/Users/mayuheng/Desktop/data/
shenduxuexidata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
jiqixuexidata<-read.csv('/Users/mayuheng/Desktop/data/
jiqixuexidata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
tuxiangchulidata<-read.csv('/Users/mayuheng/Desktop/data/
tuxiangchulidata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
tuxiangshibiedata<-read.csv('/Users/mayuheng/Desktop/data/
tuxiangshibiedata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
yuyinshibiedata<-read.csv('/Users/mayuheng/Desktop/data/
yuyinshibiedata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
jiqishijuedata<-read.csv('/Users/mayuheng/Desktop/data/
jiqishijuedata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
suanfagongchengshidata<-read.csv('/Users/mayuheng/Desktop/data/
suanfagongchengshidata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
ziranyuyanchulidata<-read.csv('/Users/mayuheng/Desktop/data/
ziranyuyanchulidata.csv',fileEncoding = "UTF-8",stringsAsFactors=FALSE)
data<-
list(shenduxuexidata=shenduxuexidata,jiqixuexidata=jiqixuexidata,tuxiangchulida
ta=tuxiangchulidata,tuxiangshibiedata=tuxiangshibiedata,yuyinshibiedata=yuyins
hibiedata,
jiqishijuedata=jiqishijuedata,suanfagongchengshidata=suanfagongchengshidata,
ziranyuyanchulidata=ziranyuyanchulidata)
data1<-cbind(belong=rep(names(data[1]),nrow(data[[1]])),data[[1]])
```

```
for(i in 2:8){data1<-
rbind(data1,cbind(belong=rep(names(data[i]),nrow(data[[i]])),data[[i]]))}
data=data1

sorteddata<-list(NULL)
  city<-NULL
  district<-NULL
  lowsalary<-NULL
  highsalary<-NULL
  experience<-NULL
  degree<-NULL
  isday<-NULL
  companyfield<-NULL
  companymembers<-NULL
  companyfinancial<-NULL
  today<-Sys.Date()
  temp<NULL
  index<-NULL
  ###############处理地名

    temp<-strsplit(data[[4]],split='·')

    for(i in 1:nrow(data)){
    city<-c(city,temp[[i]][[1]])
    if(length(temp[[i]])==2){
    district<-c(district,temp[[i]][[2]])
    }
    else{
      district<-c(district,"市")
      index<-c(index,i)
    }
    }
    temp<-NULL
  ############处理薪资
  temp<-strsplit(data[[5]],split='-')
  for(j in 1:nrow(data)){

    lowsalary<-c(lowsalary,as.numeric(chartr("K"," ",chartr("k"," ",temp[[j]][[1]]))))
    if(length(temp[[j]])==2){
    highsalary<-c(highsalary,as.numeric(chartr("K"," ",chartr("k"," ",temp[[j]]
[[2]]))))
```

```
    }
    else{
      highsalary<-c(highsalary,as.numeric(chartr("K"," ",chartr("k"," ",temp[[j]]
[[1]]))))
      index<-c(index,j)
    }
  }
  #########处理要求
  temp<-strsplit(data[[6]],split='/')
  for(j in 1:nrow(data)){
    experience<-c(experience,temp[[j]][[1]])
    degree<-c(degree,temp[[j]][[2]])
  }
  #########处理公司情况
  temp<-strsplit(data[['companysituation']],split='/')
  for(j in 1:nrow(data)){
    if(length(temp[[j]])==3){
    companyfield<-c(companyfield,temp[[j]][[1]])
    companyfinancial<-c(companyfinancial,temp[[j]][[2]])
    companymembers<-c(companymembers,temp[[j]][[3]])
    }
    else{
      companyfield<-c(companyfield,"blank")
      companyfinancial<-c(companyfinancial,"blank")
      companymembers<-c(companymembers,"blank")
      index<-c(index,j)
    }
  }
  ###########把刚刚得到的向量组成表格
sorteddata<-data.frame(belong=data[['belong']],name=data[['name']],

city=city,district=district,lowsalary=lowsalary,highsalary=highsalary,experience=e
xperience,degree=degree,company=data[['company']],

companyintroduction=data[['companyintro']],companyfield=companyfield,compa
nyfinancial=companyfinancial,
                        companymembers=companymembers,tags=data[['tags']])
###########查看缺失值
  aggr(sorteddata, prop=FALSE, numbers=TRUE,plot = TRUE)
  ################删除缺失值和前面不好处理的数据行
data<-na.omit(sorteddata)
```

```
data<-data[-index,]

#########把包含许多文字关键词的内容分开
field<-NULL
tag<-NULL
for(i in 1:nrow(data)){
  field=c(field,strsplit(chartr("、"," ",chartr(","," ",data$companyfield[[i]]))," "))
}
for(i in 1:nrow(data)){
  temp=strsplit(chartr("/"," ",data$tags[[i]])," ")[[1]][c(strsplit(chartr("/","
",data$tags[[i]])," ")[[1]]!="")]
  tag=c(tag,list(temp))
}
data$companyfield<-field
data$tags<-tag
##########算出平均薪资，默认均匀分布
data$saleray=(data$highsalary+data$lowsalary)/2
```

3.数据分析

```
Majorcity<-table(data$city)[table(data$city)>mean(as.vector(table(data$city)))]
```

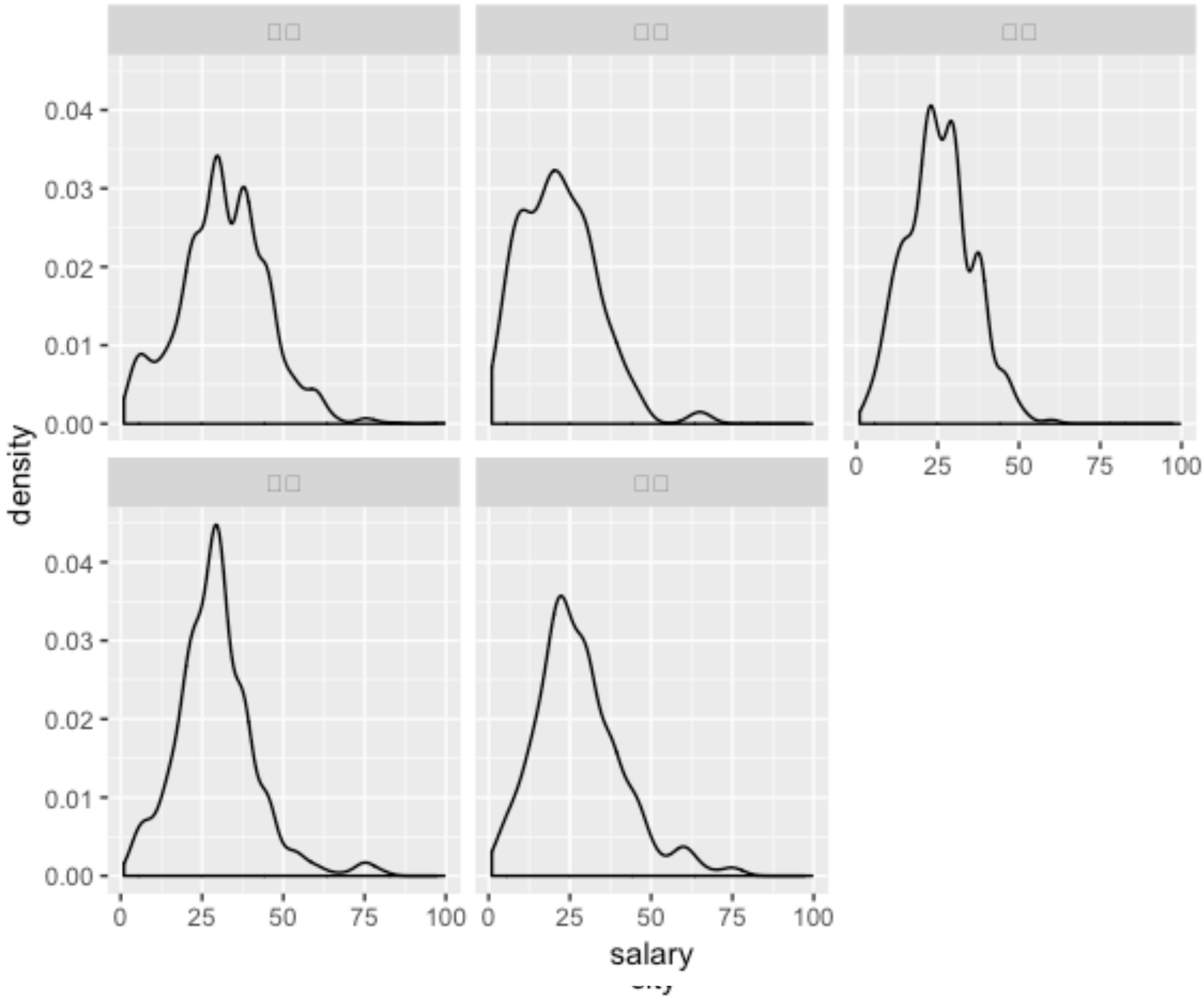
通过罗列出岗位数量大于平均岗位数量的城市，我们发现工作岗位不出意料的集中在北上广深杭，以北京最为突出。原来如此！我只能去大城市才有更多的机会呀！

```
majorcity<-table(data$city)[table(data$city)>mean(as.vector(table(data$city)))]
citydata<-data[data[["city"]]%in%names(majorcity),]
ggplot(citydata,aes(x=city,y=salary),position="jitter")+geom_boxplot(notch =
TRUE)+ scale_size_area() +xlab("city")+ stat_summary(fun.y="mean",
geom="point", shape=23, size=3, fill="white")
```

把五个大城市揪出来做箱线图，发现北京不仅量大，而且钱多，真是程序员的天堂！相比之下，广州和杭州的薪资水平就要稍逊一筹了。上海和深圳水平基本和北京持平。

```
ggplot(citydata,aes(x=salary))+geom_density()
+facet_wrap(vars(citydata$city),nrow=2)
```

同时，通过做密度曲线图，我们发现各个城市的薪资分布都属于高薪岗位少，底层民工堆积严重的情况。即便如此，作为南京大学的一名毕业生，我相信自己可以脱颖



而出，成为一名优秀的程序员而非代码民工，所以我瞄准了高薪岗位很多的北京！所以我截取北京的数据，继续探索北京的区域。

```
beijingdata<-data[as.vector(data[["city"]])=="北京",]
beijingdata$district<-as.character(beijingdata$district)
quantile(as.vector(table(beijingdata$district)))
```

去掉了较少的数据点后，我惊讶的发现，留下来的职位基本都位于朝阳区和海淀区，其中朝阳区347个，海淀区756个。

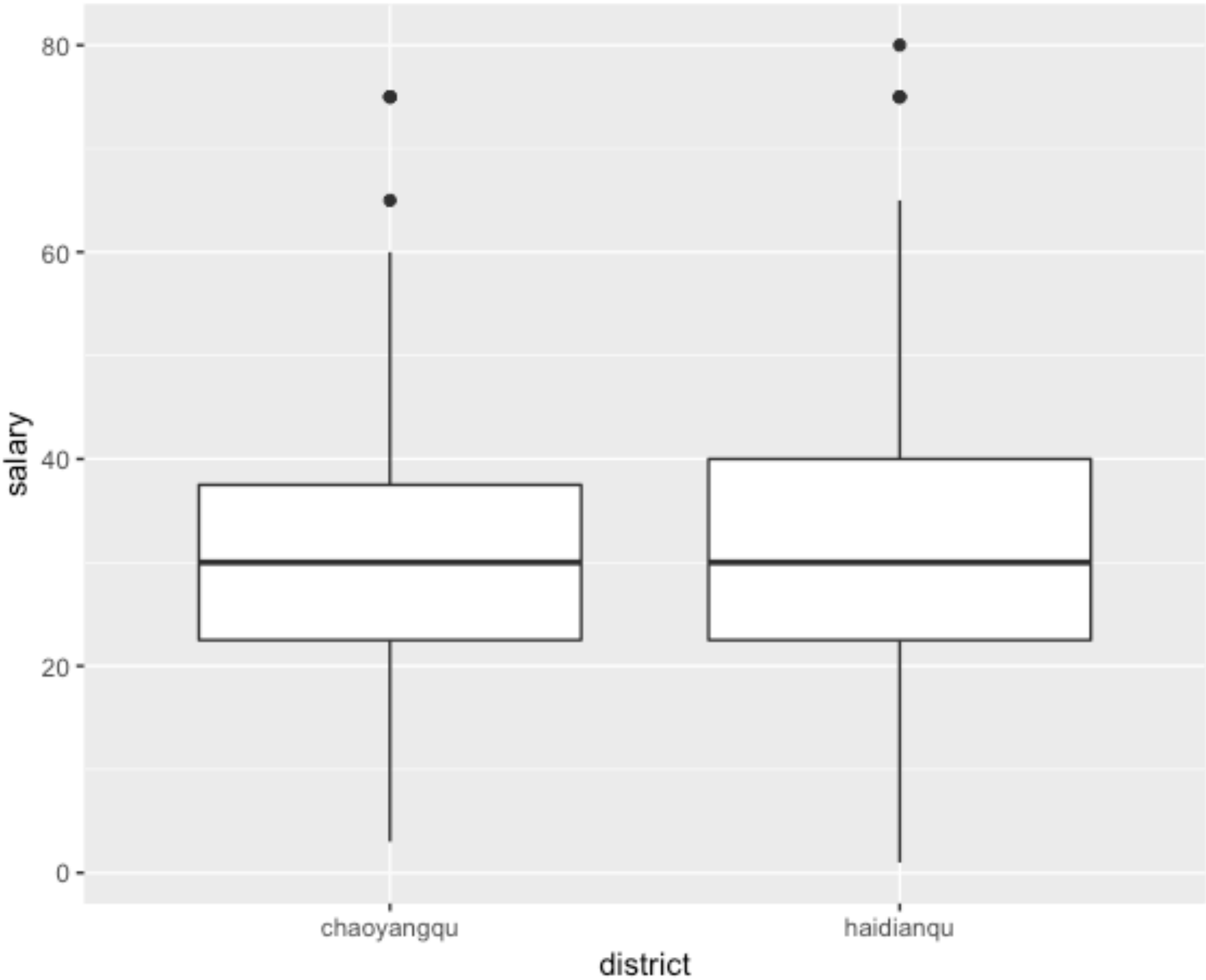
```
beijingdata<-
beijingdata[beijingdata$district%in%names(table(beijingdata$district)
[table(beijingdata$district)>20]),]
```

把他们分配到两个区。

beijingdata$district[beijingdata$district%in%c("北京大学","上地","五道口","西北旺","西二旗","西三旗","学院路","知春路","中关村","海淀区")]="haidianqu"
beijingdata$district[beijingdata$district%in%c(“大望路","酒仙桥","望京","朝阳区")]="chaoyangqu"

画图查看两者的薪资待遇区别。

ggplot(beijingdata,aes(x=district,y=salary))+geom_boxplot()+scale_size_area()



emmm好像目测没什么区别哦，我们需要用参数检验来试一下。因为是自然统计的数据，我们可以认为他们服从正态分布。

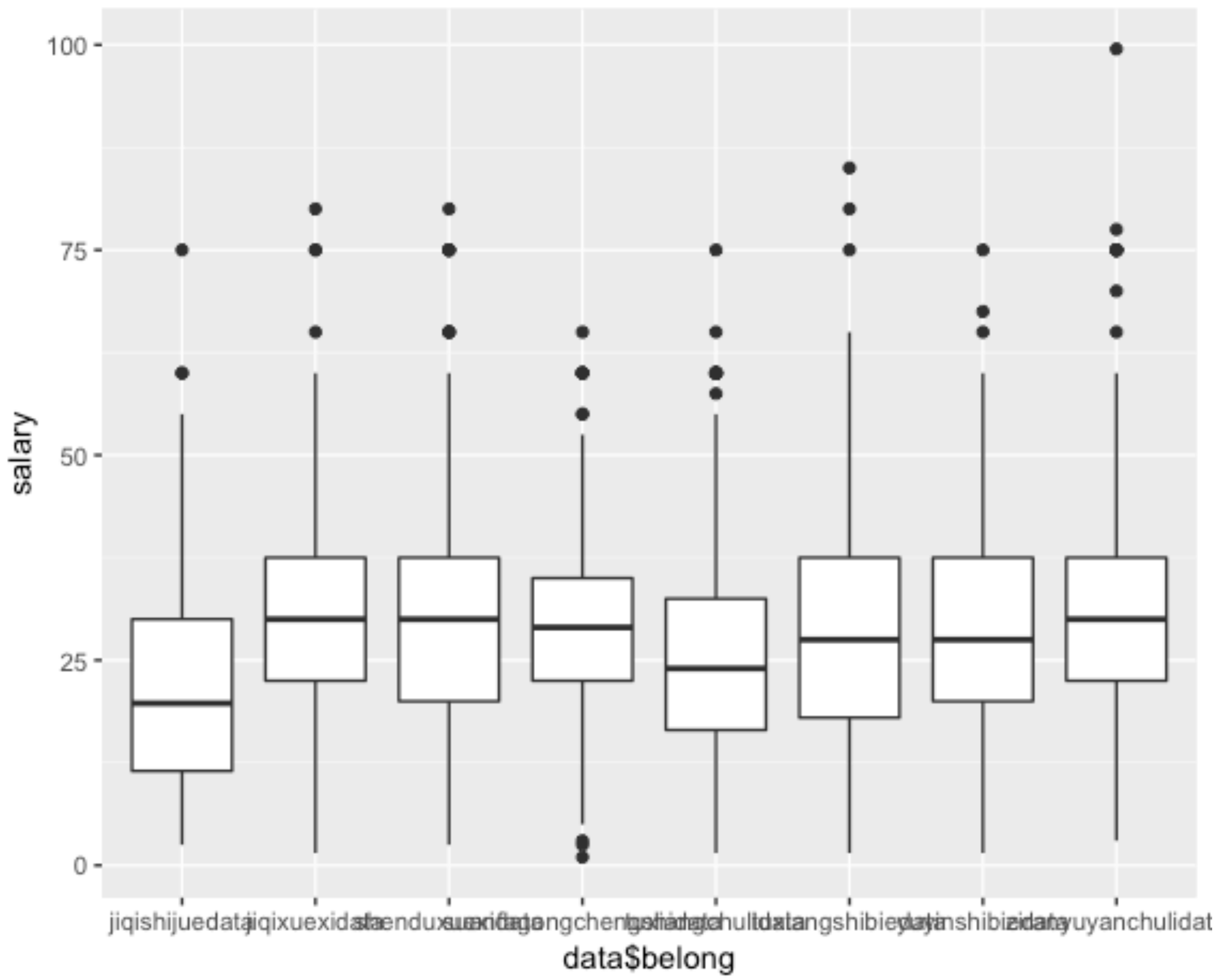
haidian<–beijingdata[beijingdata$district%in%c("北京大学","上地","五道口","西北旺","西二旗","西三旗","学院路","知春路","中关村","海淀区"),]
chaoyang<–beijingdata[beijingdata$district%in%c("大望路","酒仙桥","望京","朝阳区"),]
t.test(haidian$salary,chaoyang$salary)

结果如下：

```
data:  haidian$salary and chaoyang$salary
t = 1.1602, df = 707.28, p-value = 0.2464
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.6881866  2.6764648
sample estimates:
mean of x mean of y 
 31.55754  30.56340
```

　的确没什么区别。所以最终，我有很大的几率选择去海淀区或者朝阳区就业。于是我决定现在在两个地方各购置一套房产，以备日后需要。决定好了这个，我开始着眼于眼前的事情。作为一名统计专业的学生，我具体应当学习什么方向呢？于是，我把包含机器学习、深度学习八种岗位的薪资做箱线图，惊讶的发现他们有着显著的区别。

```
ggplot(data,aes(x=data$belong,y=salary))+geom_boxplot()
```



```
q<-NULL
for(i in 1:8){q<-
c(q,quantile(data$salary[data$belong==names(table(data$belong))[i]])[4])}
```

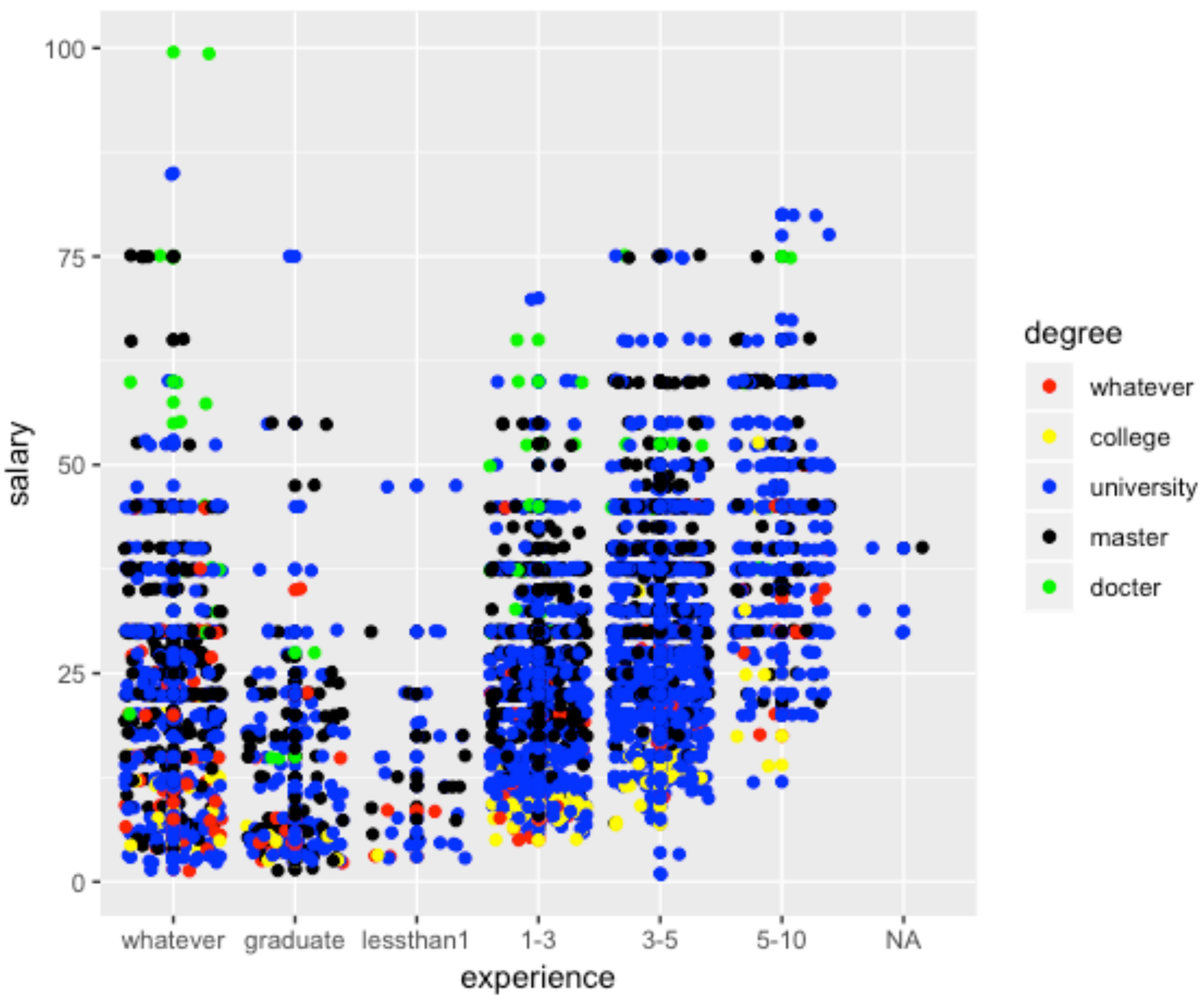
作为南京大学匡亚明学院的学生，我有信心成为前25%的程序员，于是我找出了八种工作类型的四分位数，发现他们有着显著的差异。

```
names(q)<-names(table(data$belong))
```

结果显示，做机器学习和图像处理的人工资明显低于其他工作，于是我下定决心，不学机器学习！

```
jiqishijuedata jiqixuexidata shenduxuexidata suanfagongchengshidata
30.0 37.5 37.5 35.0
tuxiangchulidata tuxiangshibiedata yuyinshibiedata ziranyuyanchulidata
32.5 37.5 37.5 37.5
```

那么我要上学到什么时候，有多少工作经历才可以获得满意的薪资呢？我将工资和学历、工作经历的关系做图。



```
data$degree=factor(data$degree,levels = c("不限","大专","本科","硕士","博士"),labels = c("whatever","college","university","master","docter"))
data$experience=factor(data$experience,c("经验不限","经验应届毕业生","经验1年以下","经验1-3年","经验3-5年","经验5-10年","经验十年以上"),labels = c("whatever","graduate","lessthan1","1-3","3-5","5-10","10+"))
ggplot(data,aes(x=experience,y=salary,color=degree))+geom_jitter()+geom_point()+scale_color_manual(values = c("red", "yellow", “blue”,”black","green"))
```

从图中我们可以发现，总体来说，随着工作时间的加长，薪资水平越来越高。但工作经历一年以下的薪资甚至不如毕业生，想必是综合了各方面的原因。而就学历来说，随着学历的升高，薪资总体不断升高。值得一提的是，两者的“不限”这个选项中都有一些点也有着很高的薪资，应当是想给更多人尝试的机会。

现在我们把工作时间和学历均转换为可以衡量的数值，为之后的使用服务。工作时间取区间平均值，学历取获得学历需要的时间。考虑到不限的选项会造成一定的干扰，所以不予考虑。

```
data$exp=as.numeric(data$exp)
data$exp[data$experience=="whatever"]=NA
data$exp[data$experience=="graduate"]=0
data$exp[data$experience=="lessthan1"]=0.5
data$exp[data$experience=="1-3"]=2
data$exp[data$experience=="3-5"]=4
data$exp[data$experience=="5-10"]=7.5
data$exp[data$experience=="10+"]=15
data$deg=as.numeric(data$deg)
data$deg[data$degree=="whatever"]=NA
data$deg[data$degree=="college"]=2
data$deg[data$degree=="university"]=4
data$deg[data$degree=="master"]=6
data$deg[data$degree==“docter"]=8
```

现在我们把目光转向公司，我最可能去什么样的公司呢？什么样的公司会给出比较满意的薪水呢？首先查看公司的主要领域。

```
for(i in 1:nrow(data)){
  field=c(field,as.vector(strsplit(chartr("、"," ",chartr(","," ",data$companyfield[[i]]))," ")))
}
field<-as.character(field)
```

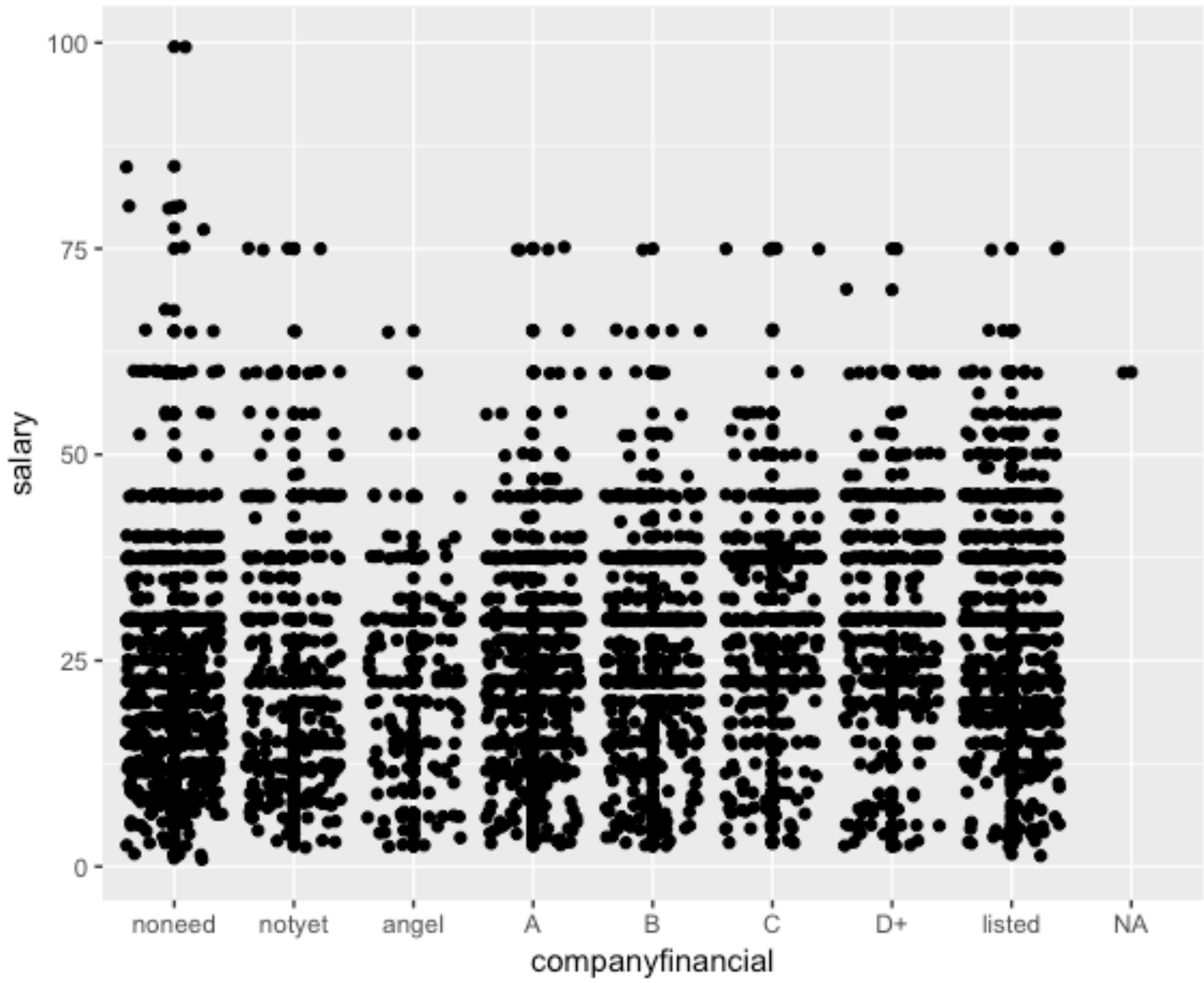
```
wordcloud(words=field)
```

不出意料，业务主要和移动互联网、数据科学相关。那么公司规模呢?


```
data$companyfinancial=factor(data$companyfinancial,levels = c("不需要融资","未融资","天使轮","A轮","B轮","C轮","D轮及以上","上市公司"),labels = c(“noneed","notyet","angel","A","B","C","D+","listed"))
ggplot(data,aes(y=salary,x=companyfinancial))+geom_point()+geom_jitter()
```

做图发现，薪资水平好像确实有一定的趋势随着公司规模上升。所以我们进行数值化的估计。

```
data$companyfin[data$companyfinancial=="不需要融资"]=NA
data$companyfin[data$companyfinancial=="未融资"]=0
data$companyfin[data$companyfinancial=="天使轮"]=1
data$companyfin[data$companyfinancial=="A轮"]=2
data$companyfin[data$companyfinancial=="B轮"]=3
```



data$companyfin[data$companyfinancial=="C轮"]=4
data$companyfin[data$companyfinancial=="D轮及以上"]=5
data$companyfin[data$companyfinancial==“上市公司"]=6
cor.test(data$salary,data$companyfin)

得到如下结果。

Pearson's product-moment correlation

data: data$salary and data$companyfin
t = 10.274, df = 3587, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1371063 0.2006741
sample estimates:
      cor

```
0.169066
```

发现p值出乎意料的小，两者确实有相关关系。对公司人数，我们使用相同的手法。

```
data$companymem<-NULL
data$companymem=as.numeric(data$companymem)
data$companymembers=factor(data$companymembers,levels = c("少于15人","15-50人","50-150人","150-500人","500-2000人","2000人以上"),labels = c("lessthan15","15-50","50-150","150-500","500-2000","2000+"))
data$companymem[data$companymembers=="lessthan15"]=8
data$companymem[data$companymembers=="15-50"]=30
data$companymem[data$companymembers=="50-150"]=100
data$companymem[data$companymembers=="150-500"]=300
data$companymem[data$companymembers=="500-2000"]=1300
data$companymem[data$companymembers=="2000+"]=2500
cor.test(data$companymem,data$salary)
```

```
Pearson's product-moment correlation

data:  data$companymem and data$salary
t = 13.31, df = 3587, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1855395 0.2478992
sample estimates:
      cor
0.2169407
```

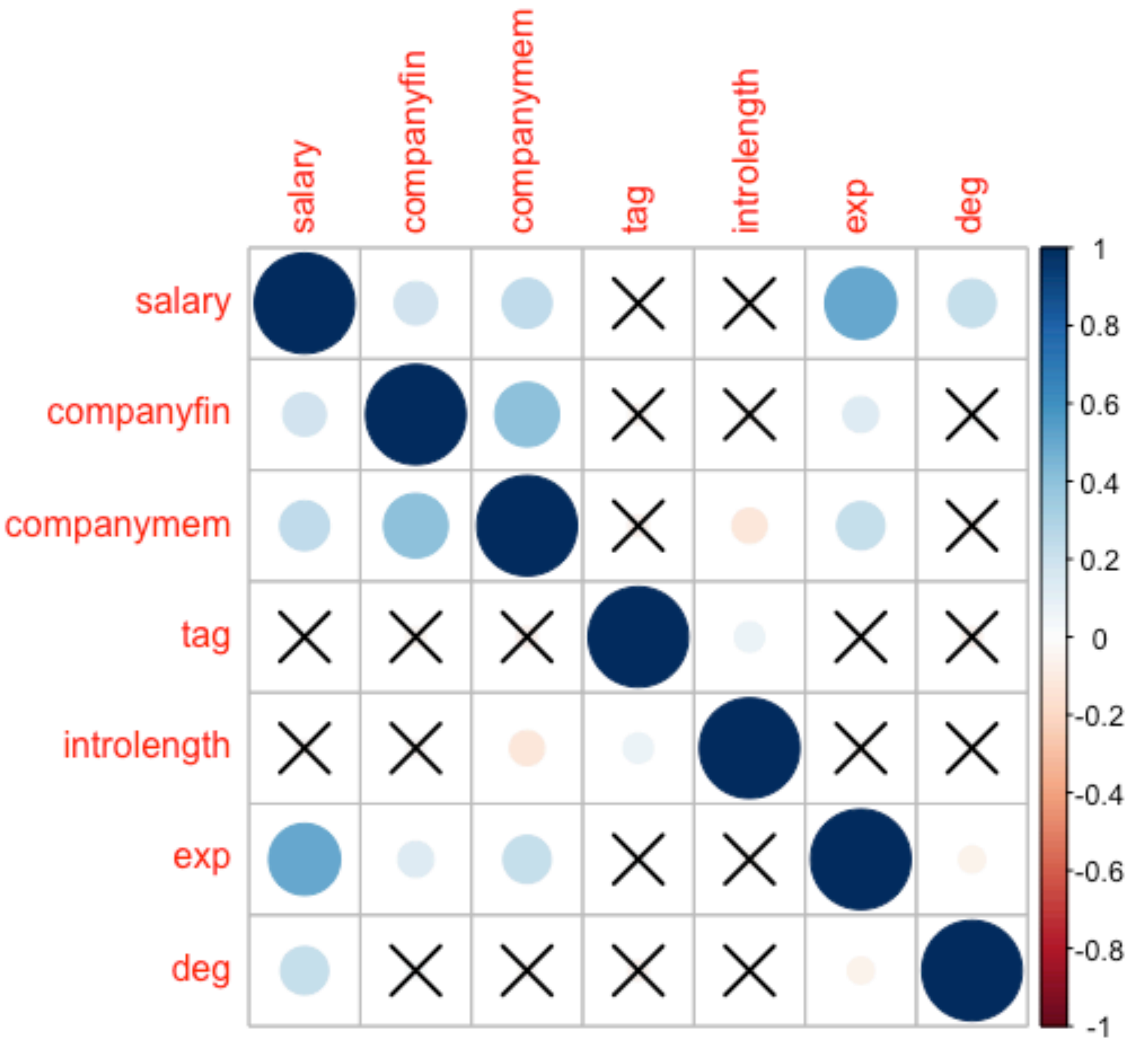
发现相关性也很充分。最后，我们来看看如何宣传自己的公司薪资水平比较高。

```
data$companyintroduction<-as.character(data$companyintroduction)
data$introlength=nchar(data$companyintroduction)
cor.test(data$salary,data$introlength)
data$tag<-as.numeric(data$tag)
for (i in 1:nrow(data)){data$tag[i]=length(data$tags[[i]])}
cor.test(data$salary,data$tag)
```

大胆猜测，会不会有自我介绍的推荐力度和薪资水平的关系呢？结果显示，tags的标签数量并无太大关系，但是自我介绍的长度的相关性检验却显示，有一定的可能性越能嘴炮的公司薪资水平较低。看来求职的时候不能只听忽悠！

```
Pearson's product-moment correlation

data:  data$salary and data$introlength
t = -2.005, df = 3588, p-value = 0.04504
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.066094240 -0.000740322
sample estimates:
        cor
-0.03345304
```

最后，我们考察这些变量彼此之间的相关性。



```
cordata<-
data[c("salary","companyfin","companymem","tag","introlength","exp","deg")]
cordata<-na.omit(cordata)
confdata<-cor.mtest(cordata)
corrplot(cor(cordata),p.mat =confdata$p,sig.level=0.005)
```

从图中我们可以得到有用的信息有：

1. 公司人数和公司融资规模成正相关。（强）
2. 公司越成规模，对工作经历要求越高。（弱）
3. 公司越大，越不倾向于多介绍自己。（弱）
4. 工作经历和学历有一定的负相关趋势，学历弱则需要强的工作经历弥补。（弱）

最后，我们做薪资关于各项数值的多元线性拟合。

```
linear<-
lm(salary~companyfin+companymem+tag+introlength+exp+deg,data=cordata)
summary(linear)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
-30.951  -7.639  -0.635   6.139  59.277 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.723696   1.244160  -0.582    0.561    
companyfin   0.368103   0.086683   4.247 2.24e-05 ***
companymem   0.001369   0.000212   6.458 1.24e-10 ***
tag          0.137221   0.149179   0.920    0.358    
introlength  0.011653   0.037864   0.308    0.758    
exp          3.425337   0.108548  31.556  < 2e-16 ***
deg          3.005675   0.176137  17.064  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.51 on 2872 degrees of freedom
Multiple R-squared:  0.3457,	Adjusted R-squared:  0.3443 
F-statistic: 252.9 on 6 and 2872 DF,  p-value: < 2.2e-16
```

显然应当删去一些变量。

```
step(linear)
```

```
Start:  AIC=13550.17
salary ~ companyfin + companymem + tag + introlength + exp +
    deg

              Df Sum of Sq    RSS   AIC
- introlength  1        10 317084 13548
- tag          1        93 317167 13549
<none>                     317074 13550
- companyfin   1      1991 319064 13566
- companymem   1      4605 321678 13590
- deg          1     32149 349222 13826
- exp          1    109937 427010 14405

Step:  AIC=13548.26
salary ~ companyfin + companymem + tag + exp + deg

             Df Sum of Sq    RSS   AIC
- tag         1        99 317183 13547
<none>                    317084 13548
- companyfin  1      1999 319083 13564
- companymem  1      4613 321697 13588
- deg         1     32141 349225 13824
- exp         1    109927 427011 14403

Step:  AIC=13547.16
salary ~ companyfin + companymem + exp + deg

             Df Sum of Sq    RSS   AIC
<none>                    317183 13547
- companyfin  1      1980 319163 13563
- companymem  1      4566 321749 13586
- deg         1     32050 349233 13822
- exp         1    110283 427466 14404
```

最终和我们的想法一样，公司是否注重宣传自己和薪资没啥关系，还是要看公司和个人的实力。所以最终有

```
linear<-step(linear)
confint(linear,level = 0.95)
```

```
                  2.5 %       97.5 %
(Intercept) -2.0375982884 1.808363859
companyfin   0.1970732754 0.536772052
companymem   0.0009412145 0.001766681
exp          3.2159348262 3.641274413
deg          2.6535753740 3.343620933
```

```
linear<-lm(salary~companyfin+companymem+exp+deg,data=cordata)
summary(linear)
```

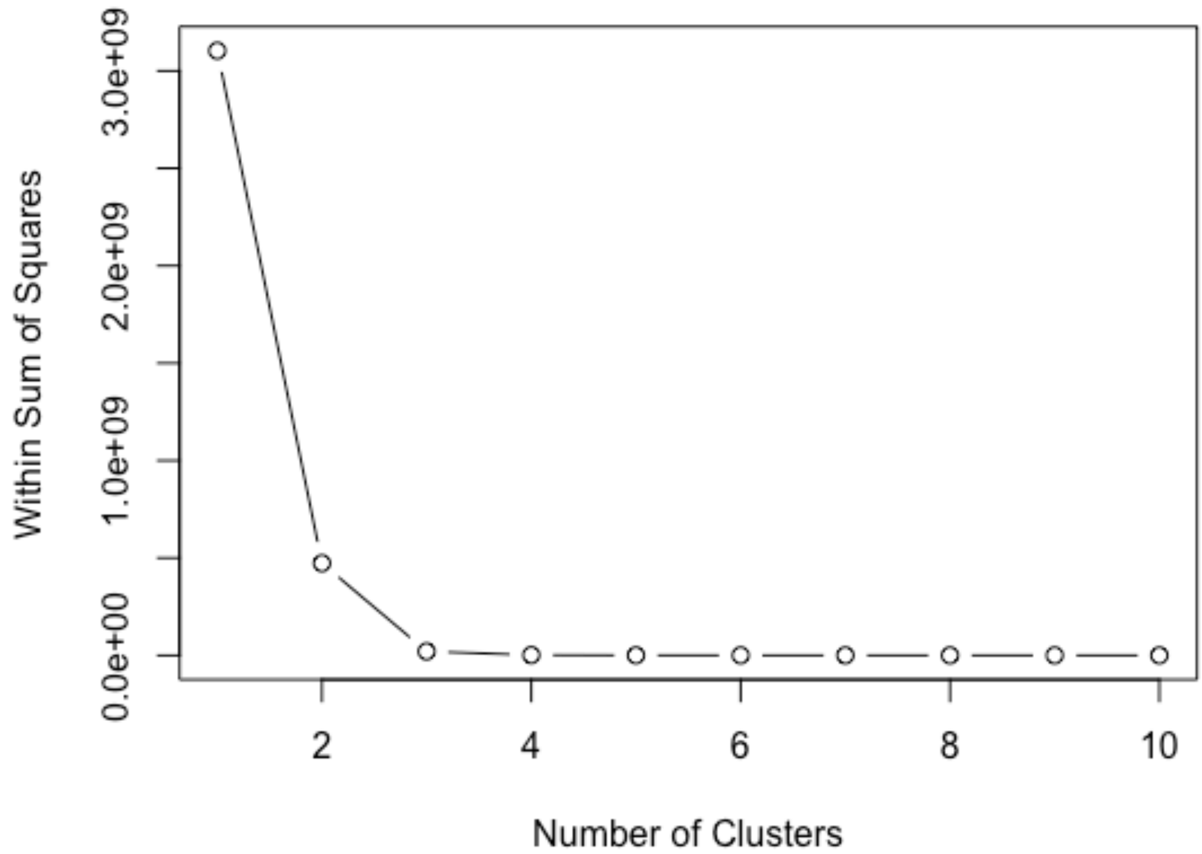
```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.1146172  0.9807176  -0.117    0.907
companyfin   0.3669227  0.0866229   4.236 2.35e-05 ***
companymem   0.0013539  0.0002105   6.432 1.47e-10 ***
exp          3.4286046  0.1084613  31.611  < 2e-16 ***
deg          2.9985982  0.1759611  17.041  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.51 on 2874 degrees of freedom
Multiple R-squared:  0.3454,  Adjusted R-squared:  0.3445
F-statistic: 379.2 on 4 and 2874 DF,  p-value: < 2.2e-16
```

p值很小，结果非常可信。R方较小，拟合程度比较差。

最后，我们使用聚类分析来为招聘公司分个类型。（强行使用课上知识，对不住了老师）。人才的学历和工作经历属于和公司本身不太相关的量，那么直接去除。

```
cordata<-data[c("salary","companyfin","companymem","exp","deg")]
cordata<-na.omit(cordata)
library(plyr)
library(cluster)
library(lattice)
library(graphics)
wss <- numeric(10)
for (k in 1:10) wss[k] <- sum(kmeans(cordata, centers=k, nstart=25)$withinss)
plot(1:10, wss, type="b", xlab="Number of Clusters", ylab="Within Sum of
Squares")
```

做图发现选择三个点为聚类中心比较合适。




```
km = kmeans(cordata,3, nstart=25)
cordata$type=km$cluster
```

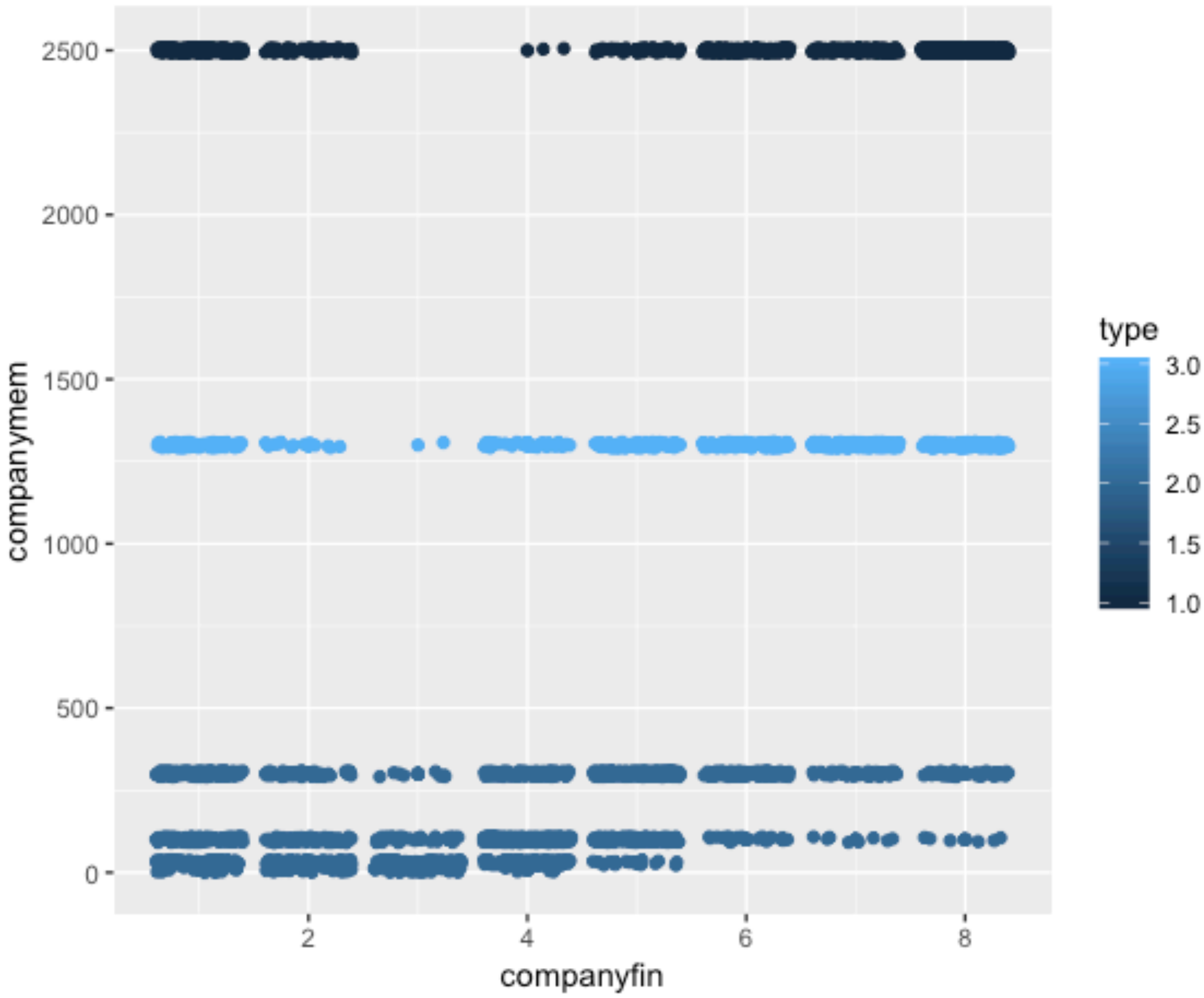
做图观察各个聚类的特点。

```
ggplot(cordata,aes(x=companyfin,y=companymem,color=type))+geom_point()
+geom_jitter()
ggplot(cordata,aes(x=tag,y=introlength,color=type))+geom_point()+geom_jitter()
```

可以看出结果直接按照了公司人数进行聚类。的确，这是因为公司人数的绝对值差异相对于其他三项有着绝对优势，导致度量几乎由公司人数决定。不知道是否有按比例计算的度量呢?

四、最终结论

通过以上记录的分析，我们得到的可信结论如下。



1. 人工智能程序员薪资水平和地区有关，且平均工资最高的城市是北京。
2. 人工智能程序员薪资水平和业务方向有关，其中机器学习和图形处理的薪资显著低于其他方向。
3. 人工智能程序员薪资水平和公司规模有关，大公司的薪资水平显著高于小公司。
4. 人工智能程序员的薪资水平和求职者经历有关，学历越高，工作经历越丰富，薪资水平越高。

五、附录

附录一：爬虫的原始数据（原始数据.zip）

附录二：清理后数据（清洗数据.csv）

附录三：代码打包（代码.zip）