下表是一个由15 个样本组成的贷款申请训练数据。

希望通过所给的训练数据学习一个贷款申请的决策树，用以对未来的贷款申请进行分类， 即当新的客户提出贷款申请时，根据申请人的特征利用决策树决定是否批准贷款申请。

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第1步计算决策属性的熵

决策属性“是否同意贷款？”。

该属性分两类：是/否

$|C_1|$(是)=9

$|C_2|$(否)=6

|D|= $|C_1|$+ $|C_2|$=15

$$P_1 = \frac{9}{15}, \qquad P_2 = \frac{6}{15}$$

$$H(D) = -P_1 log_2\ P_1 - P_2 log_2\ P_2$$

$$= -(P_1 log_2\ P_1 + P_2 log_2\ P_2)$$

=0.971

$$H\left(D\right) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} log_2 \frac{|C_k|}{|D|}$$

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第2步计算条件属性的熵

条件属性共有4个：

年龄、有工作、有自己的房子、信贷情况。

分别计算不同属性的信息增益。

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否

$$H\left(D|A\right)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H\left(D_i\right)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}log_2\frac{|D_{ik}|}{|D_i|}$$

第2-1步 计算年龄的熵

年龄共分3个组：青年、中年、老年

青年是否同意贷款比例为2/3

$|D_{11}|$(是)=2

$|D_{12}|$(否)=3

$|D_1|$= 5

$$P_1=\frac{2}{5},\qquad P_2=\frac{3}{5}$$

$$H(D_1)=-P_1log_2\ P_1-P_2log_2\ P_2$$

$$=-(P_1log_2\ P_1+P_2log_2\ P_2)$$

=0.971

ID	年龄	有工作	有自己的房子	信贷情况	类别
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是

第2-2步 计算年龄的熵

年龄共分3个组：青年、中年、老年

中年是否同意贷款比例为3/2

$|D_{21}|$(是)=3

$|D_{22}|$(否)=2

$|D_2|= 5$

$$P_1 = \frac{3}{5}, \qquad P_2 = \frac{2}{5}$$

$$H(D_2) = -P_1 log_2\ P_1 - P_2 log_2\ P_2$$

$$= -(P_1 log_2\ P_1 + P_2 log_2\ P_2)$$

=0.971

ID	年龄	有工作	有自己的房子	信贷情况	类别
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第2-3步 计算年龄的熵

年龄共分3个组：青年、中年、老年

老年是否同意贷款比例为4/1

$|D_{31}|$(是)=4

$|D_{32}|$(否)=1

$|D_3|$= 5

$$P_1 = \frac{4}{5}, \qquad P_2 = \frac{1}{5}$$

$$H(D_3) = -P_1 log_2\ P_1 - P_2 log_2\ P_2$$

$$= -(P_1 log_2\ P_1 + P_2 log_2\ P_2)$$

=0.722

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第2-4步 计算年龄的熵

年龄共分3个组：青年、中年、老年所占比例

青年组5/15
中年组5/15
老年组5/15

计算年龄的平均信息期望
E(年龄)=0.971*(5/15)+ 0.971*(5/15)+
0.722*(5/15)=0.888

G（年龄信息增益）
=0.971-0. 888
=0.083 (1)

ID	年龄	有工作	有自己的房子	信贷情况	类别
3	青年	是	否	好	是
4	青年	是	是	一般	是
8	中年	是	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
15	老年	否	否	一般	否

第3步 计算有工作的熵

有工作共分2个组：是、否

计算有工作的平均信息期望

E(有工作)=0.647

G（有工作信息增益）

=0.324 (2)

ID	年龄	有工作	有自己的房子	信贷情况	类别
4	青年	是	是	一般	是
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第4步 计算有自己的房子的熵

有自己的房子共分2个组：是、否

计算有工作的平均信息期望

E(有自己的房子)=0.551

G（有自己的房子信息增益）

=0.420 (3)

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
15	老年	否	否	一般	否

ID	年龄	有工作	有自己的房子	信贷情况	类别
2	青年	否	否	好	否
3	青年	是	否	好	是
7	中年	否	否	好	否
8	中年	是	是	好	是
12	老年	否	是	好	是
13	老年	是	否	好	是

ID	年龄	有工作	有自己的房子	信贷情况	类别
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
14	老年	是	否	非常好	是

第5步 计算信贷情况的熵

信贷情况共分3个组：一般、好、非常好

计算信贷情况的平均信息期望

E(信贷情况)=0.608

G（信贷情况信息增益）

=0.363 (4)

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

第6步 计算选择节点

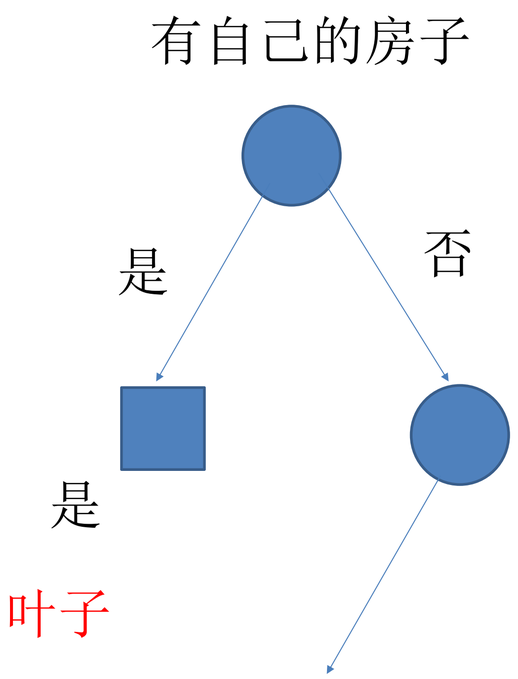
年龄信息增益 =0.971-0. 888
=0.083 (1)

有工作信息增益 =0.971-0. 647
=0.324 (2)

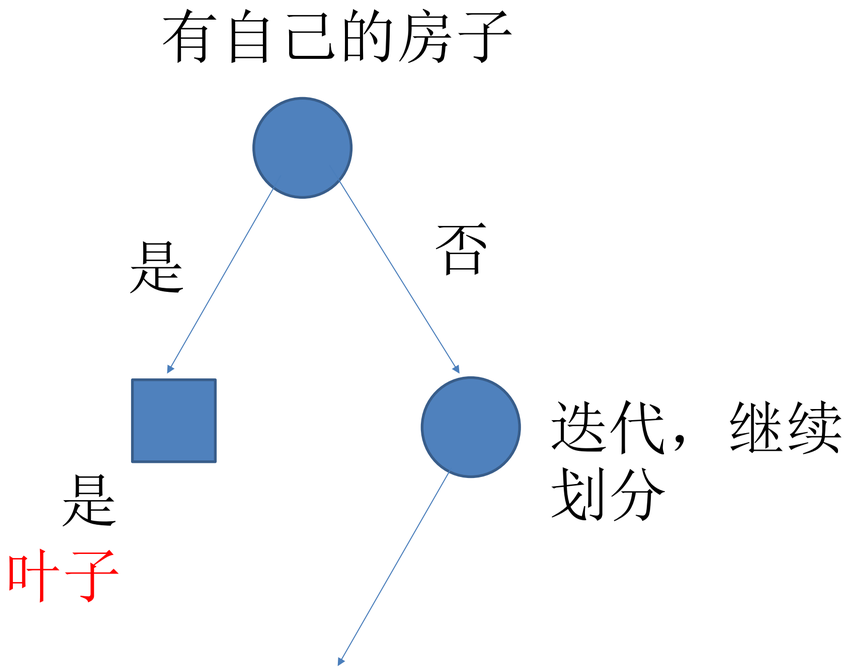
有自己的房子信息增益=0.971- 0.551
=0.420 (3)

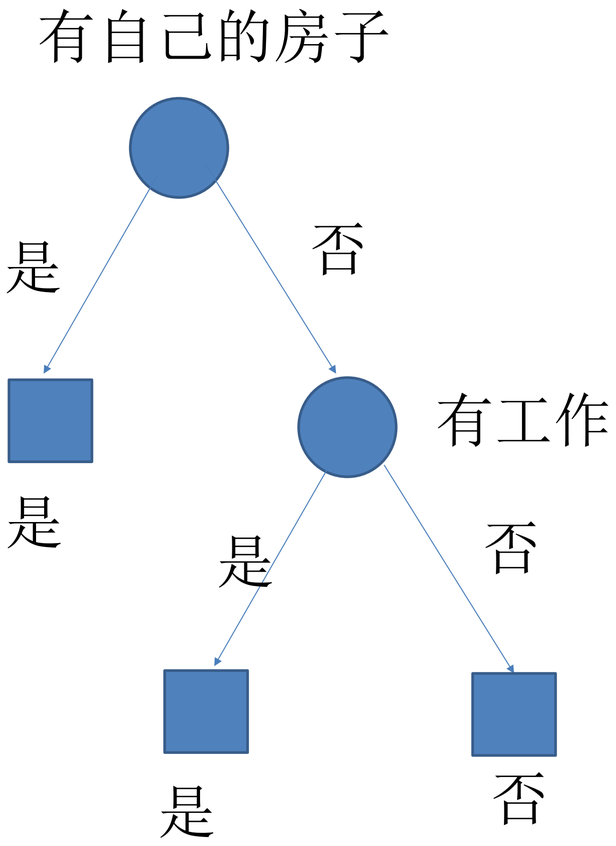
信贷情况信息增益 =0.971- 0.608
=0.363 (4)

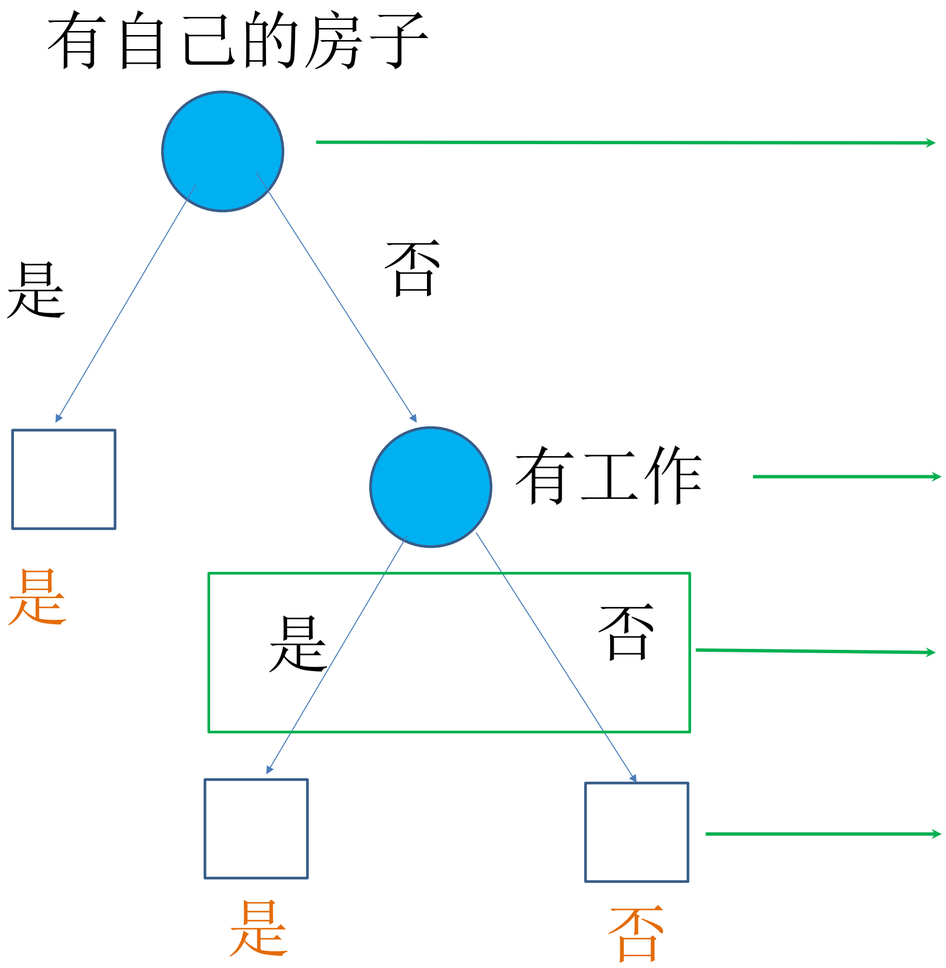
比较各特征的信息增益值。由于特征:有自己的房子的信息增益值最大，所以选择特征有自己的房子作为最优特征。



ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否


有自己的房子
是
否
是
叶子
迭代，继续
划分


有自己的房子
是
否
是
有工作
是
否
是
否



根节点 **root node**

非叶子节点 **non-leaf node**
(代表测试条件，用特征作标签，对数据属性的测试)

分支**branches**(代表测试结果，是否符合特征)

叶节点 **leaf node**
(代表分类后所获得的分类标记)

防止过拟合

基本情况：如果所有数据属于同一个类，使用该标签创建叶节点或者 所有数据有相同的特征值或者

- 我们已经达到树的一定深度?
- 我们剩下一定数量/比例的数据
- 我们达到了一个足够小的训练误差
- 使用验证数据
- ...

决策树(ID3)的实现

- **创建树形结构**
  - 判断终止条件。(纯节点；用完特征)
  - 选择最优特征。(将其加入树字典，删除已用特征)
  - 遍历每个特征值，拿到对应子集来递归创建树。
- **准备工作**
  - 准备数据集
  - 计算熵
  - 划分子集
  - 选择最优特征

编写代码计算决策属性的熵

在编写代码之前，先对数据集进行属性标注。

- 年龄：0代表青年，1代表中年，2代表老年；
- 有工作：0代表否，1代表是；
- 有自己的房子：0代表否，1代表是；
- 信贷情况：0代表一般，1代表好，2代表非常好；
- 类别(是否给贷款)：no代表否，yes代表是。

```
dataset = [[0, 0, 0, 0, 'no'],
           [0, 0, 0, 1, 'no'],
           [0, 1, 0, 1, 'yes'],
           [0, 1, 1, 0, 'yes'],
           [0, 0, 0, 0, 'no'],
           [1, 0, 0, 0, 'no'],
           [1, 0, 0, 1, 'no'],
           [1, 1, 1, 1, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [2, 0, 1, 2, 'yes'],
           [2, 0, 1, 1, 'yes'],
           [2, 1, 0, 1, 'yes'],
           [2, 1, 0, 2, 'yes'],
           [2, 0, 0, 0, 'no']]
```

```
dataset = [[0, 0, 0, 0, 'no'],
           [0, 0, 0, 1, 'no'],
           [0, 1, 0, 1, 'yes'],
           [0, 1, 1, 0, 'yes'],
           [0, 0, 0, 0, 'no'],
           [1, 0, 0, 0, 'no'],
           [1, 0, 0, 1, 'no'],
           [1, 1, 1, 1, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [2, 0, 1, 2, 'yes'],
           [2, 0, 1, 1, 'yes'],
           [2, 1, 0, 1, 'yes'],
           [2, 1, 0, 2, 'yes'],
           [2, 0, 0, 0, 'no']]
```

计算决策属性的熵

```
def calc_ent(dataset):
    n = len(dataset)
    # 统计各类别的样本数
    label_count = {}
    for x in dataset:
        label = _____?_____
        label_count[label] =
label_count.get(label, 0) + 1
    ent = 0
    #计算决策属性熵
                ?
    return ent
```

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

```
dataset = [[0, 0, 0, 0, 'no'],
           [0, 0, 0, 1, 'no'],
           [0, 1, 0, 1, 'yes'],
           [0, 1, 1, 0, 'yes'],
           [0, 0, 0, 0, 'no'],
           [1, 0, 0, 0, 'no'],
           [1, 0, 0, 1, 'no'],
           [1, 1, 1, 1, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [1, 0, 1, 2, 'yes'],
           [2, 0, 1, 2, 'yes'],
           [2, 0, 1, 1, 'yes'],
           [2, 1, 0, 1, 'yes'],
           [2, 1, 0, 2, 'yes'],
           [2, 0, 0, 0, 'no']]
```

划分子集

```
def split_dataset(dataset, feat_i,
value):
    sub_dataset = []
    for x in dataset:
        if x[feat_i] == value:
            x_a = x[:feat_i]
            x_a.extend(x[feat_i + 1 :])
            sub_dataset.append(x_a)
    return sub_dataset
```

选最优特征

```
def choose_best_feat(dataset):
    # 选择使得信息增益最大的特征索引

    ？ ？ ？ ？ ？

    return best_feat
```

创建树

```
def create_tree(dataset, feat_labels):
    # 判断终止条件
    ？ ？ ？ ？

    # 选择最优特征
    ？ ？ ？ ？

    # 遍历最优特征的每一个特征值，拿到子集，并递归创建树

    ？ ？ ？ ？

    return my_tree
```

分类 根据已经构建好的决策树进行分类