熟悉Jupyter notebook

- python 环境与版本
- 几个重要的工具：numpy, scipy, sklearn, pandas, keras, tensorflow
- 安装 tensorflow 与 keras
- jupyter notebook基本使用
- markdown的用法

0. python 环境与版本

In [ ]:

众多的版本（解释器的版本）以及其配套的工具包与安装环境，可能造成了使用起来的混乱。比如说你在命令行里面给python3.6安装了numpy，但在python2.7里面想调用python3.6的numpy就不行，需要对应起来使用。Anaconda就提供了一个比较好的环境，将不同版本的python分开使用，这也是为什么我们选择用anaconda来作为讲解的工具。这里我们花点时间讲一讲ananconda的使用以及python环境与版本的管理。

在选择最新版本的Anaconda时，请选择python3.6版本。现在众多工具包已经宣布停止对python2.7版本的维护和升级，转向python3.6。我们也需要与时俱进（python 2.7可以不用安装了）。我们本节课的重点就是利用anaconda装新的python环境

0.1 安装anaconda

- 请在清华的镜像地址：https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/ (https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/) 中选择合适自己操作系统的安装文件。使用苹果的同学选用结尾为**MacOSX-x86_64.pkg** 的版本，使用windows的同学请选用**Windows-x86_64.exe** 的版本。 ‘64’代表64位的操作系统，现在的电脑一般都是这种，如果还在使用32位的同学，要考虑换一台机器，或者使用32位的对应anaconda.
- 打开安装包，按照默认地址安装即可。

In [ ]:

In [ ]:

1. 打开jupyter notebook

- mac user: 直接在terminal里面，用cd 命令进入到目标文件夹，然后敲击 jupyter notebook
- windows user: 在左下方windows开始键中，找 anaconda prompt 这个应用，点击激活，cd 到相应文件夹，然后敲击 jupyter notebook

进入后，在右方点击 new，选择想用的kernel即可

In [ ]:

2. Jupyter notebook的基本使用技巧

以下所有操作都需要在命令模式下完成（按ESC或者点击编辑框外的空白处）

- 向上增加空白cell, A
- 向下增加空白cell, B
- 运行cell，并将光标移动到下一个cell，shift+enter
- 运行cell，ctrl+enter
- 删除本cell，DD
- 剪切本cell, X
- 粘贴本cell, V
- 合并两个cell,A
- 将cell转为代码状态，Y
- 将cell转为markdown状态，M
- 将选中的几个cells合并，Shift+M
- 打开 / 关闭行号，L
- 恢复删除的最后一个cell，Z 只有最后一个啊
- 选中代码整体左移 / 右移, ctrl+[或者ctrl+]

In [ ]:

3.如何添加安装多个python环境?

这里我们需要分两步走：

1. 用conda新建python环境
2. 用ipykernel 将新建的python环境加到jupyter notebook里面

方法：在命令行中执行以下代码 (相关阅读材料https://ipython.readthedocs.io/en/latest/install/kernel_install.html (https://ipython.readthedocs.io/en/latest/install/kernel_install.html))

3.1 用conda新建环境

下面这，py36是一个自定义的名称，而python=3.6是一个格式，可以变动等号右边的数字来改变python环境的kernel版本,这里我们安装的是3.6版本

```
conda create -n py36 python=3.6
```

In [ ]:

In [ ]:

在创建环境后，需要进入该环境，安装其他工具，进入环境的命令，在windows和mac下面不一样

- Mac用户使用

```
source activate py36
```

- WINDOWS用户使用

```
activate py36
```

- 确认一下python的版本

```
python -V
```

关闭 python 3.6环境（现在先不要关）

```
deactivate
```

3.2 如何将更多环境添加到 jupyter notebook中?

先进入一个python 3.6环境,在这个环境内，通过下面两条命令，安装ipykernel，然后将环境加到jupyter notebook里面去，并命名Python36。 注意，这里的Python36是在jupyter notebook里面的名称

In [ ]:

- 首先检查是否已经处于python 3.6的环境中，如果不是，请用deactivate退出，并用3.1节中介绍的activate命令进入python 3.6的环境.
- 然后，再利用以下命令，在python 3.6环境中安装ipykernel工具，并使用这个工具，将python 3.6的环境添加至jupyter notebook, 并命名为Python36

```
pip install ipykernel

python -m ipykernel install --name Python36
```

查看已有kernel

```
jupyter kernelspec list
```

分清python环境为什么重要?

- 避免**语法版本不一引起的错误**
- 避免**工具包安装与调用的混乱**

In [ ]:

3.3 检测jupyter notebook里面有没有新的kernel

- **重新启动一个anaconda prompt**，在这里面，敲击jupyter notebook，进入之后，点右上**new**那一个键，看里面有没有刚刚安装的kernel，Python36，有就成功了

In [ ]:

In [ ]:

附加题 安装决策树可视化工具 Graphviz

这个工具的安装有很多问题，比较麻烦

下载链接：https://graphviz.gitlab.io/download/ (https://graphviz.gitlab.io/download/)

1. 下载并安装，记录安装的地址，
2. 在windows下调整PATH，把刚才记录的地址添加到Path中，不会调Path的同学请百度更改‘系统环境变量’
3. 在相应环境下的terminal中使用 pip install graphviz
4. 按顺序试一试以下的几块代码，如果能够成功画出决策树，就没有问题了

Mac版本应该更简单：

1. 安装 brew
2. brew install graphviz
3. 在相应环境中pip install graphviz
4. 按顺序试一试以下的几块代码，如果能够成功画出决策树，就没有问题了

In [1]:
```
import graphviz
```

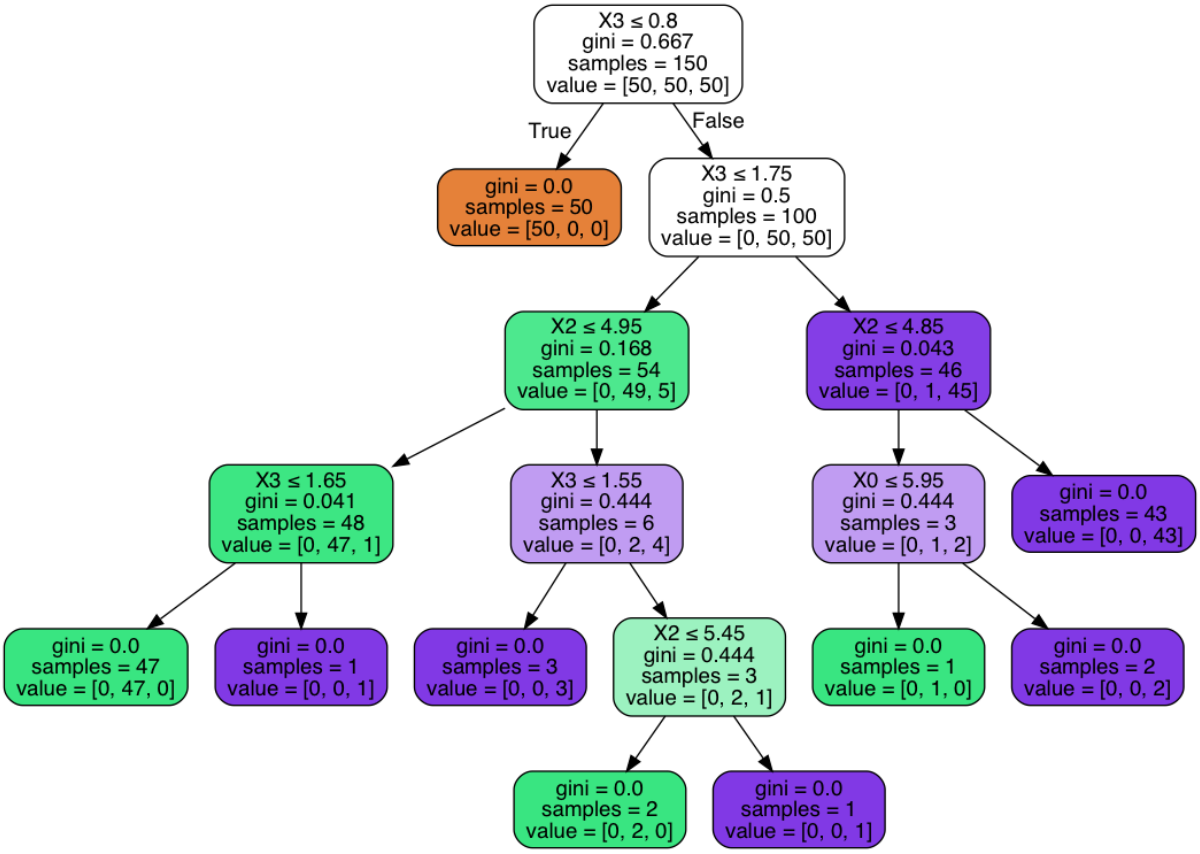
In [11]:
```
import sklearn.datasets as datasets
import pandas as pd
iris=datasets.load_iris()
df=pd.DataFrame(iris.data, columns=iris.feature_names)
y=iris.target
```

In [12]:
```
from sklearn.tree import DecisionTreeClassifier
dtree=DecisionTreeClassifier()
dtree.fit(df,y)
```

Out[12]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=N
one,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False, random_state=N
one,
                       splitter='best')
```

In [13]:
```
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(dtree, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

Out[13]:



In [ ]:
```
```

4. 几个重要的工具

```
import numpy as np

import scipy

import pandas as pd

import sklearn

import keras.backend as K

import tensorflow as tf
```

numpy

numpy定义了python进行矩阵数值计算的基础

```
np.add(A,B)

np.subtract(A,B)

np.dot(C,D)

np.matmul(C,D) ## C@D

np.multiply(C,D) ## C*D
```

In [16]:
```
import numpy as np
A = np.array([1,2,3])
B = np.array([4,3,0])
print('A+B = ',np.add(A,B))
print('A-B = ',np.subtract(A,B))
```

```
A+B =  [5 5 3]
A-B =  [-3 -1  3]
```

In [28]:
```
C = np.array([[1,2,3],
              [1,1,1]])
D = np.array([[4,3,9,9],
              [1,1,2,2,],
              [4,2,1,1]])
print('np.dot(C,D) = ',np.dot(C,D))
print('np.matmul(C,D) = ',np.matmul(C,D))
print('         C@D    = ',np.matmul(C,D))

print('np.multiply(D,D)=',np.multiply(D,D))
print('           D*D   = ', D*D)
```

```
np.dot(C,D) =  [[18 11 16 16]
 [ 9  6 12 12]]
np.matmul(C,D) =  [[18 11 16 16]
 [ 9  6 12 12]]
        C@D    =  [[18 11 16 16]
 [ 9  6 12 12]]
np.multiply(D,D)= [[16  9 81 81]
 [ 1  1  4  4]
 [16  4  1  1]]
           D*D   =  [[16  9 81 81]
 [ 1  1  4  4]
 [16  4  1  1]]
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

重点看看 np.dot 和 np.matmul的区别

- 都是矩阵乘法
- np.matmul中，禁止矩阵与标量的乘法
- np.matmul中，在矢量乘矢量的内积运算中，matmul与dot没有差别
- np.matmul中，多维的矩阵，将前n-2维视为后2维的元素后，进行乘法操作

矢量内积无区别，结果是标量

In [58]:
```
t1 = np.array([1,2,3])
t2 = np.array([1,2,3])
print(np.dot(t1,t2))
print(np.matmul(t1,t2))
print(t1*t2) # 元素1 1 乘积
```

```
14
14
[1 4 9]
```

1维与2维矩阵的乘法

In [74]:
```
a = np.array([1,2,3])
b = np.array([[1,1,1],
              [2,2,2],
              [3,3,3]])
```

In [62]:
```
np.dot(a,b)
```

Out[62]:
```
array([14, 14, 14])
```

In [63]:
```
np.matmul(a,b)
```

Out[63]:
```
array([14, 14, 14])
```

In [75]:
```
#注意
a*b # 扩展维度后，元素乘积
```

Out[75]:
```
array([[1, 2, 3],
       [2, 4, 6],
       [3, 6, 9]])
```

In [ ]:
```

```

多维矩阵乘积的比较

In [78]:
```
m_4_4 = np.array([
                         [1,2,3,4],
                         [3,2,1,4],
                         [5,4,6,7],
                         [11,12,13,14]
                        ])


m_3_4_2 = np.array([
                              [[2,3],
                               [11,9],
                               [32,21],
                               [28,17]],

                              [[2,3],
                               [1,9],
                               [3,21],
                               [28,7]],

                              [[2,3],
                               [1,9],
                               [3,21],
                               [28,7]]
                             ])
```

In [79]:
```
print(m_4_4.shape)
print(m_3_4_2.shape)
```

```
(4, 4)
(3, 4, 2)
```

In [80]:
```
print('4x4*3x4x2 dot:\n {}\n'.format(np.dot(m_4_4,m_3_4_2)))
print('4x4*3x4x2 matmul:\n {}\n'.format(np.matmul(m_4_4,m_3_4_2)))
```

```
4x4*3x4x2 dot:
 [[[232 152]
  [125 112]
  [125 112]]

 [[172 116]
  [123  76]
  [123  76]]

 [[442 296]
  [228 226]
  [228 226]]

 [[962 652]
  [465 512]
  [465 512]]]

4x4*3x4x2 matmul:
 [[[232 152]
  [172 116]
  [442 296]
  [962 652]]

 [[125 112]
  [123  76]
  [228 226]
  [465 512]]

 [[125 112]
  [123  76]
  [228 226]
  [465 512]]]
```

In [81]:
```
print('4 4 矩阵 与')
print('3 4 2 矩阵的乘积结果')
print('dot 结果的维度    =',np.dot(m_4_4,m_3_4_2).shape)
print('matmul 结果的维度 =',np.matmul(m_4_4,m_3_4_2).shape)
```

```
4 4 矩阵 与
3 4 2 矩阵的乘积结果
dot 结果的维度    = (4, 3, 2)
matmul 结果的维度 = (3, 4, 2)
```

检验以下m_4_4与m_3_4_2最后两维的矩阵乘积结果

In [68]:
```
np.dot(m_4_4,m_3_4_2[0,:,:])
```

Out[68]:
```
array([[232, 152],
       [172, 116],
       [442, 296],
       [962, 652]])
```

In [77]:
```
np.matmul(m_4_4,m_3_4_2[:,:,0].T)
```

Out[77]:
```
array([[232, 125, 125],
       [172, 123, 123],
       [442, 228, 228],
       [962, 465, 465]])
```

np.dot 定义：

> dot(a, b)[i,j,k,m] = sum(a[i,j,:] * b[k,:,m])

> import numpy as np
>
> **import scipy** 科学计算工具
>
> **import pandas as pd ** 大杀器
>
> **import sklearn ** 本课程前两周重点
>
> import keras.backend as K
>
> import tensorflow as tf

In [82]:
```
import sklearn
```

In [84]:
```
from sklearn import datasets
from sklearn.metrics import auc
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
```

5. 安装Tensorflow 与 Keras

In [1]:
```
import tensorflow as tf
```

```
/Users/crazychen/anaconda/lib/python3.6/importlib/_bootstrap.py:219: Ru
ntimeWarning: compiletime version 3.5 of module 'tensorflow.python.fram
ework.fast_tensor_util' does not match runtime version 3.6
  return f(*args, **kwds)
```

理论上有很多种安装方式，这里我们用与anaconda相关的方式来安装，以方便以后的使用和管理。

1. 需要新建一个环境，用python 3.5版本

```
conda create -n tensorflow35 python=3.5
```

2. 激活这个环境

```
activate tensorflow35
```

3. 在这个环境中，安装tensorflow

```
pip install tensorflow
```

4. 安装成功后，检验以下 方法：
   - 在命令行中，敲击 python, 进入
   - 敲击import tensorflow as tf 如果没有问题，那就安装成功了
5. 利用第三节的内容，用ipykernel将 tensorflow35这个环境加到jupyter notebook里面

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

6. Markdown的基本技巧

链接在这里哟 (https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)

小抄 (https://guides.github.com/pdfs/markdown-cheatsheet-online.pdf)

In [ ]:

标题

#

# #

#

#

In [ ]:

如何怎样怎么样**划重点**

In [ ]:

如何怎样怎么样*画斜体*

In [ ]:

引用一段话

> 于是我就念了一句诗

In [ ]:

- 列出几个观点
- 不带序号
  - 子观点

In [ ]:

1. 列出我的观点
2. 这次带序号
   - 子观点

In [ ]:

插入数学符号

$e^{\hat{i}} + 1 = 0$

$\sum_{\forall i} x_i^2$

$$\dot{x} = \text{‘} (y - x)$$
$$\dot{y} = \text{‘} x - y - xz$$
$$\dot{z} = \tilde{-} z + xy$$
$$\left(\sum_{k=1}^{n} a_k b_k\right)^2 \leq \left(\sum_{k=1}^{n} a_k^2\right)\left(\sum_{k=1}^{n} b_k^2\right)$$

更多小抄 (https://reu.dimacs.rutgers.edu/Symbols.pdf)

更多小抄2 (http://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Typesetting%20Equations.html)

In [ ]:

引用外部链接 (https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)

In [ ]:

插入图片

In [ ]:

In [ ]:

In [ ]:

In [ ]:

<img src='./spacex.jpg',width=200>

In [3]:
```
from IPython.display import Image
```

In [27]:
```
Image(filename='./spacex.jpg',width=200)
```

Out[27]:

- **插入表格**

第一栏	第二栏	第三栏
col 3 is	right-aligned	1600
col 2 is	centered	12
zebra stripes	are neat	1

In [ ]:

In [ ]:

In [ ]:

In [ ]: