MI 算法

目录

1. 分类& 回归
2. 树、森林
3. 多层分类感知器（类神经网络算法）

- Classification
  - Logistic regression
  - Decision tree classifier
  - Random forest classifier
  - Gradient-boosted tree classifier
  - Multilayer perceptron classifier
  - One-vs-Rest classifier (a.k.a. One-vs-All)
- Regression
  - Linear regression
  - Decision tree regression
  - Random forest regression
  - Gradient-boosted tree regression
  - Survival regression
- Decision trees
  - Inputs and Outputs
    - Input Columns
    - Output Columns
- Tree Ensembles
  - Random Forests
    - Inputs and Outputs
      - Input Columns
      - Output Columns (Predictions)
  - Gradient-Boosted Trees (GBTs)
    - Inputs and Outputs
      - Input Columns
      - Output Columns (Predictions)

Classification and Regression

分类和回归

- **Overview: estimators, transformers and pipelines**
- Extracting, transforming and selecting features
- Classification and Regression
- Clustering
- Advanced topics

- Classification
  - Logistic regression
  - Decision tree classifier
  - Random forest classifier
  - Gradient-boosted tree classifier
  - Multilayer perceptron classifier
  - One-vs-Rest classifier (a.k.a. One-vs-All)
- Regression
  - Linear regression
  - Decision tree regression
  - Random forest regression
  - Gradient-boosted tree regression
  - Survival regression
- Decision trees
  - Inputs and Outputs
    - Input Columns
    - Output Columns
- Tree Ensembles
  - Random Forests
    - Inputs and Outputs
      - Input Columns
      - Output Columns (Predictions)
  - Gradient-Boosted Trees (GBTs)
    - Inputs and Outputs
      - Input Columns
      - Output Columns (Predictions)

分类/回归—tree

- 决策树（decision tree）是一种基本的分类与回归方法，以分类为例。决策树呈树形结构，其中每个内部节点表示一个属性上的测试，每个分支代表一个测试输出，每个叶节点代表一种类别。学习时利用训练数据，根据损失函数最小化的原则建立决策树模型；预测时，对新的数据，利用决策树模型进行分类。
- 决策树学习通常包括3个步骤：特征选择、决策树的生成和决策树的剪枝。
- 特征选择在于选取对训练数据具有分类能力的特征，这样可以提高决策树学习的效率。通常特征选择的准则是信息增益(information gain)(或信息增益比、基尼指数(gini index)等)，每次计算每个特征的信息增益，并比较它们的大小，选择信息增益最大（信息增益比最大、基尼指数最小）的特征。
- 熵（entropy），它是表示随机变量不确定性的度量。熵越大，随机变量的不确定性就越大。而信息增益（informational entropy）表示得知某一特征后使得信息的不确定性减少的程度。简单的说，一个属性的信息增益就是由于使用这个属性分割样例而导致的期望熵降低。

分类/回归—tree 决策树生成

- 从根结点开始，对结点计算所有可能的特征的信息增益，选择信息增益最大的特征作为结点的特征，由该特征的不同取值建立子结点，再对子结点递归地调用以上方法，构建决策树；直到所有特征的信息增均很小或没有特征可以选择为止，最后得到一个决策树。
- 决策树需要有停止条件来终止其生长的过程。一般来说最低的条件是：当该节点下面的所有记录都属于同一类，或者当所有的记录属性都具有相同的值时。这两种条件是停止决策树的必要条件，也是最低的条件。在实际运用中一般希望决策树提前停止生长，限定叶节点包含的最低数据量，以防止由于过度生长造成的过拟合问题。

分类/回归—tree决策树的剪枝

- 决策树生成算法递归地产生决策树，直到不能继续下去为止。这样产生的树往往对训练数据的分类很准确，但对未知的测试数据的分类却没有那么准确，即出现过拟合现象。解决这个问题的办法是考虑决策树的复杂度，对已生成的决策树进行简化，这个过程称为剪枝。
- 决策树的剪枝往往通过极小化决策树整体的损失函数来实现。一般来说，损失函数可以进行如下的定义：使用参数a 来权衡拟合程度和模型的复杂度

$$C_a(T) = C(T) + a|T|$$

其中，T为任意子树，$C(T)$ 为对训练数据的预测误差（如基尼指数），$|T|$ 为子树的叶结点个数，$a \geq 0$ 为参数，$C_a(T)$ 为参数是 a 时的子树T的整体损失，参数 a 权衡训练数据的拟合程度与模型的复杂度。对于固定的 a，一定存在使损失函数 $C_a(T)$ 最小的子树，将其表示为 T_a。当 a 大的时候，最优子树 T_a 偏小；当 a 小的时候，最优子树 T_a 偏大。

分类/回归—tree决策树举例

- 某相亲网站通过调查相亲历史数据发现，女孩在实际相亲时有如下表现：

序号	城市拥有房产	婚姻历史（离过婚、单身）	年收入（单位：万元）	见面（是、否）
1	是	单身	12	是
2	否	单身	15	是
3	是	离过婚	10	是
4	否	单身	18	是
5	是	离过婚	25	是
6	是	单身	50	是
7	否	离过婚	35	是
8	是	离过婚	40	是
9	否	单身	60	是
10	否	离过婚	17	否

分类/回归—tree决策树举例

- 根据上表历史数据可以构建如下决策树：

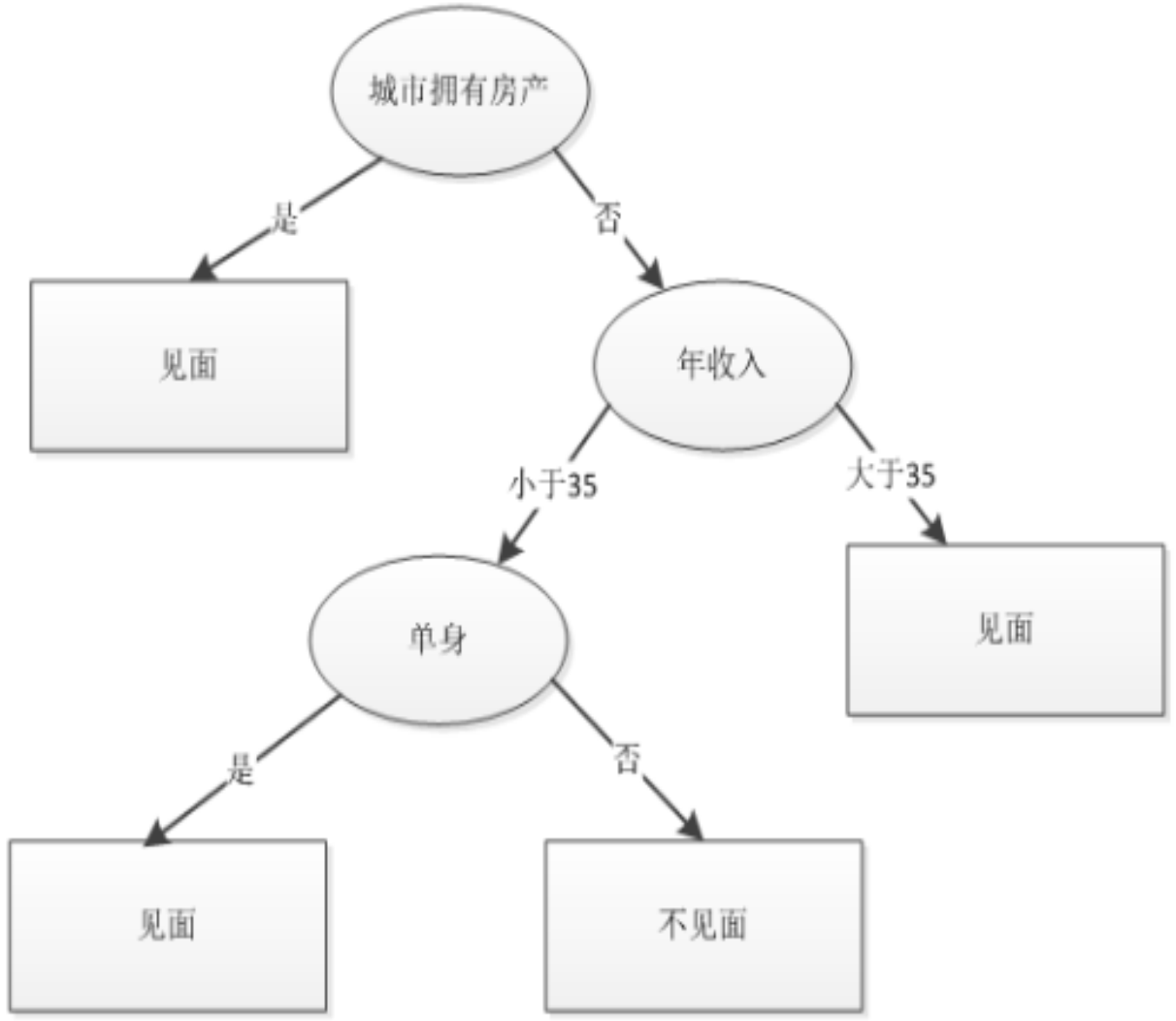
如果网站新注册了一个用户，他在城市无房产、年收入小于 35w 且离过婚，则可以预测女孩不会跟他见面。通过上面这个简单的例子可以看出，决策树对于现实生活具有很强的指导意义。通过该例子，我们也可以总结出决策树的构建步骤：

1.将所有记录看作是一个节点

2.遍历每个变量的每种分割方式，找到最好的分割点

3.利用分割点将记录分割成两个子结点 C1 和 C2

4.对子结点 C1 和 C2 重复执行步骤 2）、3），直到满足特定条件为止



分类/回归—tree决策树举例

- 在构建决策树的过程中，最重要的是如何找到最好的分割点，那怎样的分割点才算是最好的呢？如果一个分割点能够将整个记录准确地分为两类，那该分割点就可以认为是最好的，此时被分成的两类是相对来说是最“纯”的。例如前面的例子中“在城市拥有房产”可以将所有记录分两类，所有是“是”的都可以划为一类，而“否”的则都被划为另外一类。所有“是”划分后的类是最“纯“的，因为所有在城市拥有房产单身男士，不管他是否离过婚、年收入多少都会见面；而所有“否”划分后的类，又被分为两类，其中有见面的，也有不见面的，因此它不是很纯，但对于整体记录来讲，它是最纯的。
- 在上述例子当中，可以看到决策树既可以处理连续型变量也可以处理离散型变量，连续型变量如年收入，它可以用“>=”，“>”,“<”或“<=”作为分割条件，而名称型变量如城市是否拥有房产，值是有限的集合如“是“、”否“两种，它采用”=”作为分割条件。

分类/回归—tree决策树举例

- 在前面提到，寻找最好的分割点是通过量化分割后类的纯度来确定的，目前有三种纯度计算方式，分别是 Gini 不纯度、熵（Entropy）及错误率，它们的公式定义如下：

- Gini 不纯度：$Gini = 1 - \sum_{i=1}^{n} P(i)^2$ 记录中第 i 类记录数占总记录数的比例，例如前面的女孩相亲例子可以根据见面或不见面分为两类，见面的记录占比数为 P(1)=9/10，不见面的记录占比为 P(2)=1/10。
- 熵（Entropy）：$Entropy = -\sum_{i=1}^{n} P(i) * \log_2^{P(i)}$
- 错误率：$Error = 1 - \max\{P(i) \mid i \in [1, n]\}$

- 上面的三个公式均是值越大表示越“不纯”，值越小表示越“纯”。实际中最常用的是 Gini 不纯度公式，后面的例子也将采用该公式进行纯度计算。

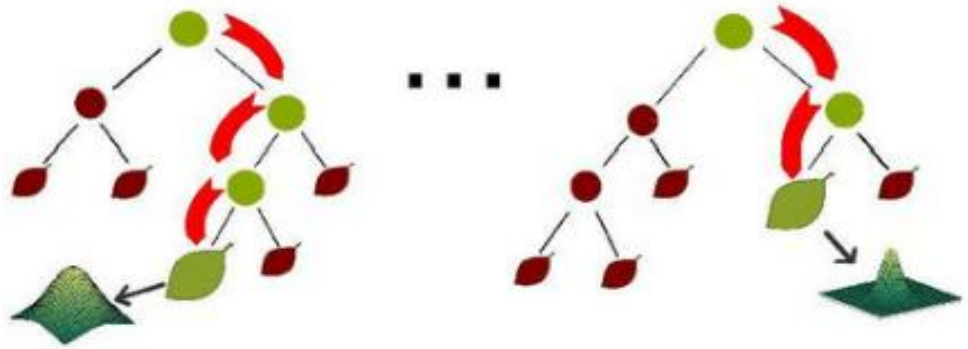
分类/回归—tree决策树举例

- 决策树的构建是一个递归的过程，理想情况下所有的记录都能被精确分类，即生成决策树叶节点都有确定的类型，但现实这种条件往往很难满足，这使得决策树在构建时可能很难停止。即使构建完成，也常常会使得最终的节点数过多，从而导致过度拟合（overfitting），因此在实际应用中需要设定停止条件，当达到停止条件时，直接停止决策树的构建。但这仍然不能完全解决过度拟合问题，过度拟合的典型表现是决策树对训练数据错误率很低，而对测试数据其错误率却非常高。过度拟合常见原因有：（1）训练数据中存在噪声；（2）数据不具有代表性。过度拟合的典型表现是决策树的节点过多，因此实际中常常需要对构建好的决策树进行枝叶裁剪（Prune Tree），但它不能解决根本问题，随机森林算法的出现能够较好地解决过度拟合问题。


Random Forests

分类/回归—随机森林

- 随机森林算法是机器学习、计算机视觉等领域内应用极为广泛的一个算法，它不仅可以用来做分类，也可用来做回归即预测，随机森林机由多个决策树构成，相比于单个决策树算法，它分类、预测效果更好，不容易出现过度拟合的情况。
- 由多个决策树构成的森林，算法分类结果由这些决策树投票得到，决策树在生成的过程当中分别在行方向和列方向上添加随机过程，行方向上构建决策树时采用放回抽样（bootstraping）得到训练数据，列方向上采用无放回随机抽样得到特征子集，并据此得到其最优切分点，这便是随机森林算法的基本原理。如图给出了随机森林算法分类原理，从图中可以看到，随机森林是一个组合模型，内部仍然是基于决策树，同单一的决策树分类不同的是，随机森林通过多个决策树投票结果进行分类，算法不容易出现过度拟合问题。



分类/回归—随机森林

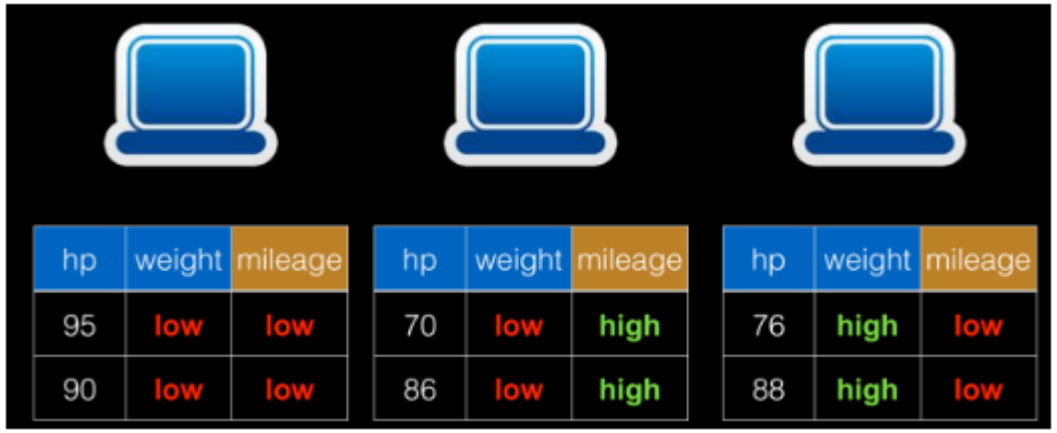
- 随机森林随机森林算法在单机环境下很容易实现，但在分布式环境下特别是在 Spark 平台上，传统单机形式的迭代方式必须要进行相应改进才能适用于分布式环境，这是因为在分布式环境下，数据也是分布式的，算法设计不得当会生成大量的IO 操作，例如频繁的网络数据传输，从而影响算法效率。

horsepower	weight	mileage
95	low	low
90	low	low
70	low	high
86	low	high
76	high	low
88	high	low

分类/回归—随机森林

- 在 Spark 上进行随机森林算法的实现，需要进行一定的优化，Spark 中的随机森林算法主要实现了三个优化策略

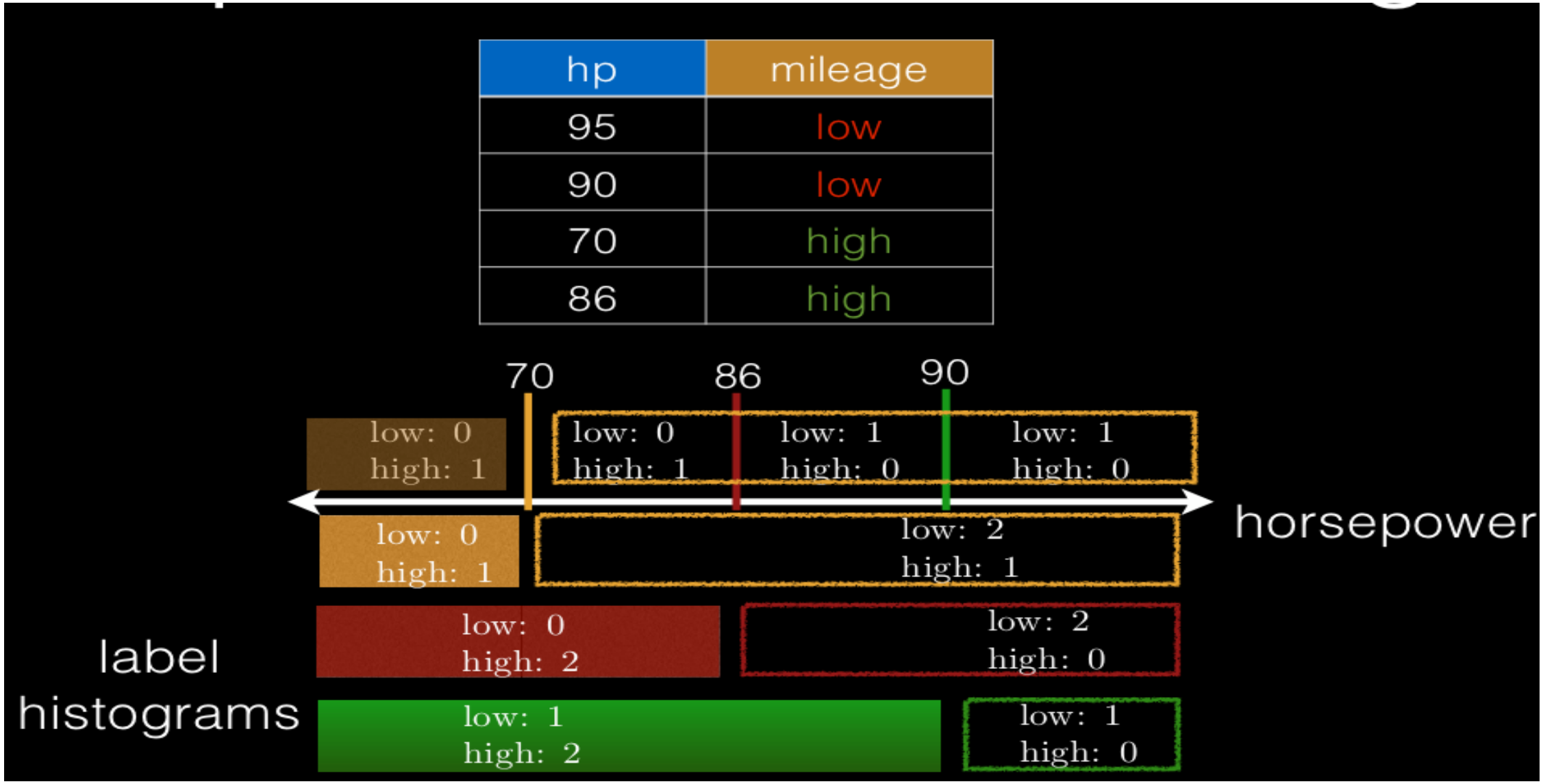
1.切分点抽样统计，在单机环境下的决策树对连续变量进行切分点选择时，一般是通过对特征点进行排序，然后取相邻两个数之间的点作为切分点，这在单机环境下是可行的，但如果在分布式环境下如此操作的话，会带来大量的网络传输操作，特别是当数据量达到 PB 级时，算法效率将极为低下。为避免该问题，Spark 中的随机森林在构建决策树时，会对各分区采用一定的子特征策略进行抽样，然后生成各个分区的统计数据，并最终得到切分点。



分类/回归—随机森林

- 2.切分特征装箱（Binning），决策树的构建过程就是对特征的取值不断进行划分的过程，对于离散的特征，如果有 M 个值，最多 $2^{M-1}-1$ 个划分，如果值是有序的，那么就最多 M-1 个划分。比如年龄特征，有老，中，少 3 个值，如果无序有 3 种划分：老|中，少；老，中|少；老，少|中；如果是有序的，即按老，中，少的序，那么只有 m-1 个，即 2 种划分，老|中，少；老，中|少。对于连续的特征，其实就是进行范围划分，而划分的点就是 split（切分点），划分出的区间就是 bin。对于连续特征，理论上 split 是无数的，在分布环境下不可能取出所有的值，因此它采用的是1中的切点抽样统计方法。

分类/回归—随机森林


hp
mileage
95
low
90
low
70
high
86
high
70
86
90
low: 0
high: 1
low: 0
high: 1
low: 1
high: 0
low: 1
high: 0
low: 0
high: 1
low: 2
high: 1
horsepower
label
histograms
low: 0
high: 2
low: 2
high: 0
low: 1
high: 2
low: 1
high: 0

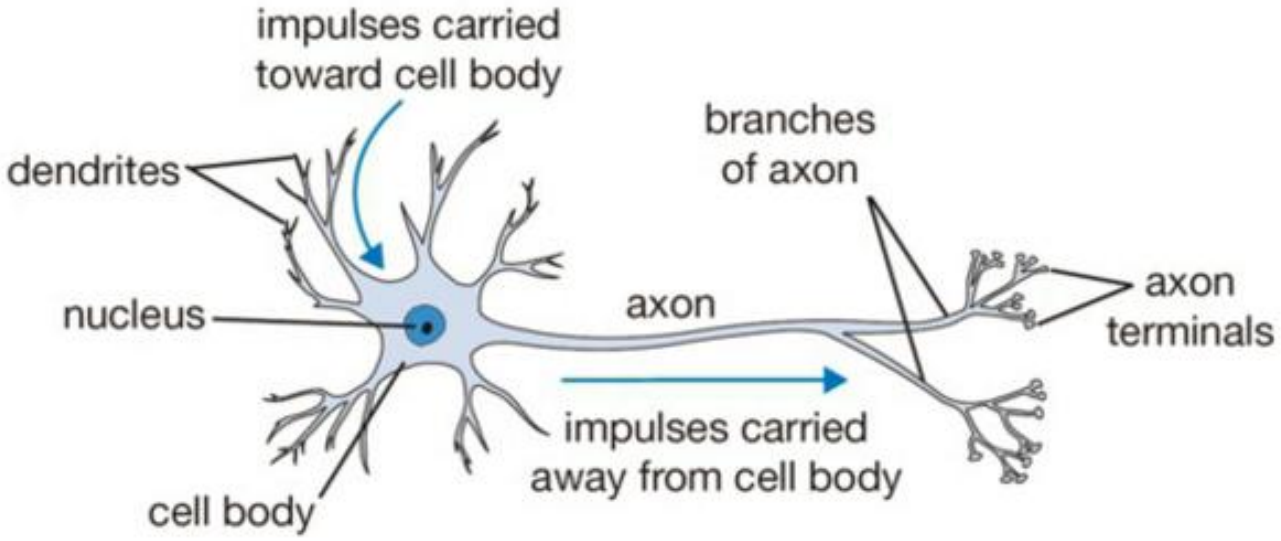
分类/回归—随机森林

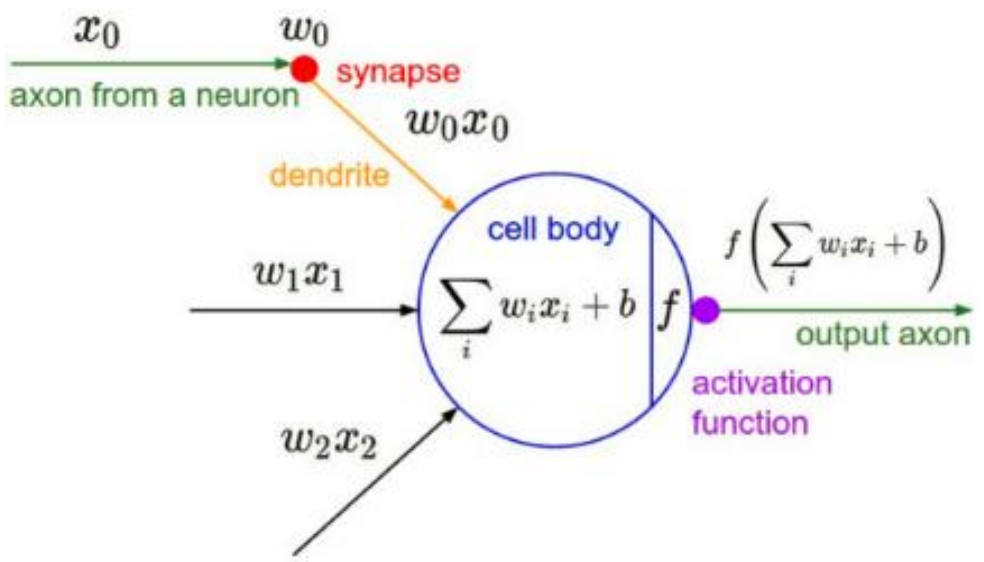
- 3.逐层训练（level-wise training）单机版本的决策数生成过程是通过递归调用（本质上是深度优先）的方式构造树，在构造树的同时，需要移动数据，将同一个子节点的数据移动到一起。此方法在分布式数据结构上无法有效的执行，而且也无法执行，因为数据太大，无法放在一起，所以在分布式环境下采用的策略是逐层构建树节点（本质上是广度优先），这样遍历所有数据的次数等于所有树中的最大层数。每次遍历时，只需要计算每个节点所有切分点统计参数，遍历完后，根据节点的特征划分，决定是否切分，以及如何切分。


Multilayer perceptron classifier

神经网络—模拟人脑

- 人脑中基本的计算单元叫做神经元(neuron).人的神经系统中大约包含860亿个这样的神经元，并且他们之间通过大约10^14~10^15这么多的突触(synapses)连接。下图就显示了一个神经元和它抽象出的数学模型。每个神经元会从它们的树突(dendrites)获得输入信号，然后再将输出信号传给它唯一的轴突(axon)。轴突再通过突触和其他神经元的树突相连。





神经网络—模拟人脑

- 在神经元的数学模型中，轴突所携带的信号(例如：x0)通过突触进行传递，由于突触的强弱不一，假设我们以w0表示，那么我们传到下一个神经元的树突处的信号就变成了w0x0。其中突触强弱(参数w)是可学的，它控制了一个神经元对另一个神经元影响的大小和方向（正负）。然后树突接收到信号后传递到神经元内部(cell body)，与其他树突传递过来的信号一起进行加和，如果这个和的值大于某一个固定的阈值的话，神经元就会被激活，然后传递冲激信号给树突。在数学模型中我们假设传递冲激信号的时间长短并不重要，只有神经元被激活的频率用于传递信息。我们将是否激活神经元的函数称为激活函数(activation function f)，它代表了轴突接收到冲激信号的频率。以前我们比较常用的一个激活信号是sigmoid function σ（ sigma ），因为它接收一个实值的信号（即上面所说的加和的值）然后将它压缩到0-1的范围内。

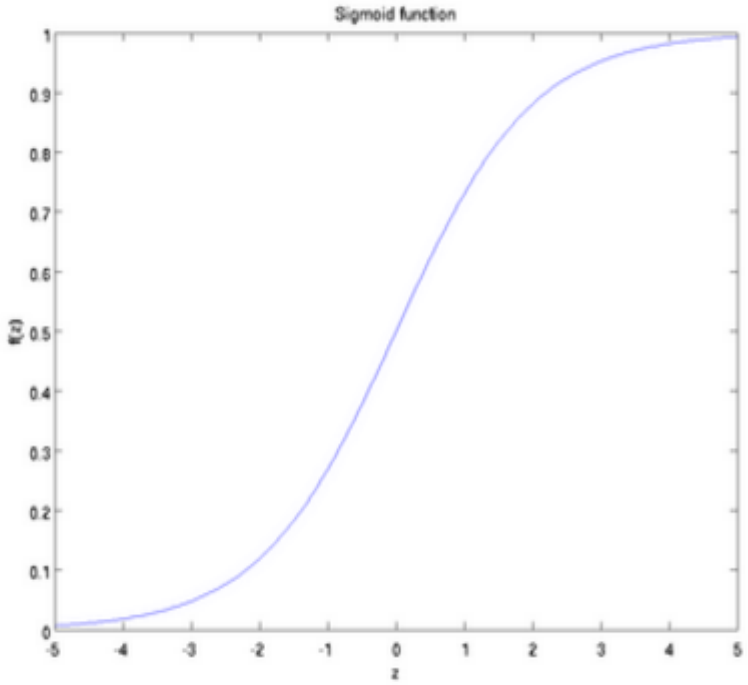
神经网络—模拟人脑

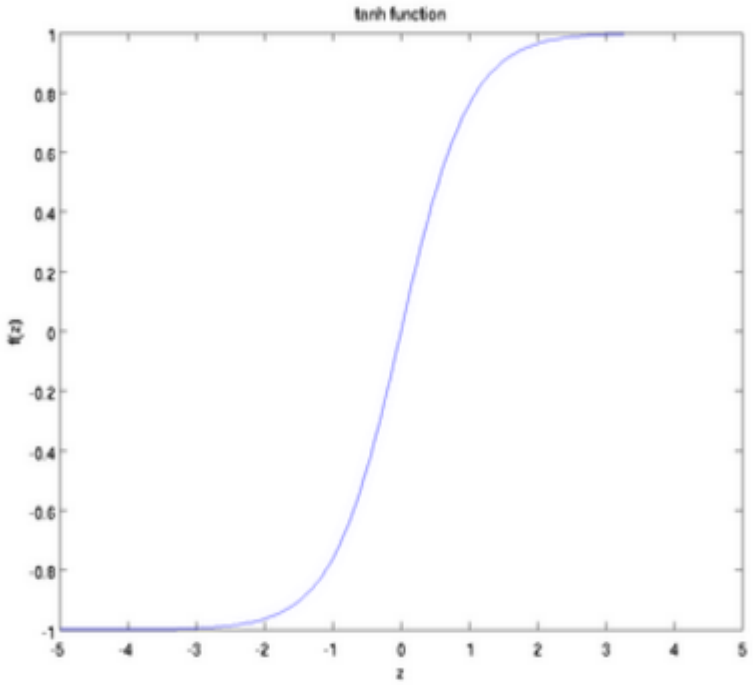
- 一个神经元就是一个线性分类器
- 神经元的上述前向传播过程从形式上看着很熟悉。我们之前在线性分类器中看到，分类器具有判断score好坏的能力，在神经元中也是一样，我们通过激活与否来得到神经元的输出，再通过一个恰当的损失函数就能将一个神经元转化成线性分类器了。
- Binary Softmax classifier. 比如说，我们可以把 $\sigma(\sum_i w_i x_i + b)$ 看成是某类的概率 $P(y_i = 1|x_i; w)$ ，那么另一类的概率则是 $P(y_i = 0|x_i; w) = 1 - P(y_i = 1|x_i; w)$ ，因为对于二值分类器而言两类的概率相加应为1。然后我们再通过loss function 防止过拟合，这就是一个也叫逻辑回归。因为sigmoid function会把只限定于0-1之间，分类器可以通过判断上述概率是否大于0.5来进行分类。

神经网络—激活函数

- 常用的激活函数

以下分别是sigmoid及tanh的函数图像

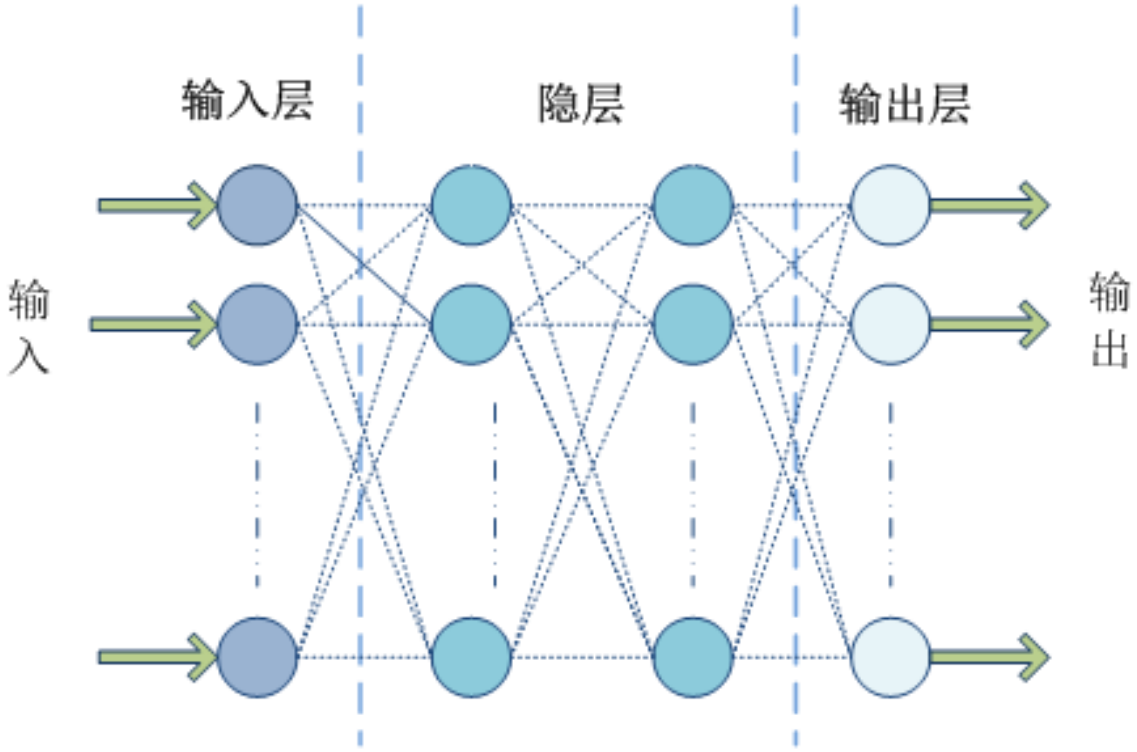




$\tanh(z)$ 函数是sigmoid函数的一种变体，它的取值范围为 $[-1, 1]$ ，而不是sigmoid函数的 $[0, 1]$ 。

MLPC

- 多层感知器模型



- Spark ML 在 1.5 版本后提供一个使用 BP(反向传播，Back Propagation) 算法训练的多层感知器实现，BP 算法的学习目的是对网络的连接权值进行调整，使得调整后的网络对任一输入都能得到所期望的输出。BP 算法名称里的反向传播指的是该算法在训练网络的过程中逐层反向传递误差，逐一修改神经元间的连接权值，以使网络对输入信息经过计算后所得到的输出能达到期望的误差。Spark 的多层感知器隐层神经元使用 sigmoid 函数作为激活函数，输出层使用的是 softmax 函数。

Spark MLPC

- Spark 的多层感知器分类器 (MultilayerPerceptronClassifer) 支持以下可调参数:
- featuresCol:输入数据 DataFrame 中指标特征列的名称。
- labelCol：输入数据 DataFrame 中标签列的名称。
- layers:这个参数是一个整型数组类型，第一个元素需要和特征向量的维度相等，最后一个元素需要训练数据的标签取值个数相等，如 2 分类问题就写 2。中间的元素有多少个就代表神经网络有多少个隐层，元素的取值代表了该层的神经元的个数。例如val layers = Array[Int](100,6,5,2)。
- maxIter：优化算法求解的最大迭代次数。默认值是 100。
- predictionCol:预测结果的列名称。
- tol:优化算法迭代求解过程的收敛阀值。默认值是 1e-4。不能为负数。
- blockSize:该参数被前馈网络训练器用来将训练样本数据的每个分区都按照 blockSize 大小分成不同组，并且每个组内的每个样本都会被叠加成一个向量，以便于在各种优化算法间传递。该参数的推荐值是 10-1000，默认值是 128。
- 算法的返回是一个 MultilayerPerceptronClassificationModel 类实例。

Spark MLPC—例子

- 通过训练一个多层感知器分类模型来预测新的短信是否为垃圾短信。
- 该数据集结构非常简单，只有两列，第一列是短信的标签 ，第二列是短信内容，两列之间用制表符 (tab) 分隔

```
ham  Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
ham  Ok lar... Joking wif u oni...
spam    Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rat
ham  U dun say so early hor... U c already then say...
ham  Nah I don't think he goes to usf, he lives around here though
spam    FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std c
ham  Even my brother is not like to speak with me. They treat me like aids patent.
ham  As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. F
spam    WINNER!! As a valued network customer you have been selected to receivea £900 prize reward! To claim call 090617014
spam    Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The
ham  I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
spam    SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandC
spam    URGENT! You have won a 1 week FREE membership in our £100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&
ham  I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will f
ham  I HAVE A DATE ON SUNDAY WITH WILL!!
spam    XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmob
ham  Oh k...i'm watching here:)
ham  Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
```

Spark MLPC—例子

- 分析实现步骤

在处理文本短信息分类预测问题的过程中，笔者首先是将原始文本数据按照 8:2 的比例分成训练和测试数据集。整个过程分为下面几个步骤

从 HDFS 上读取原始数据集，并创建一个 DataFrame。

使用 StringIndexer 将原始的文本标签 (“Ham”或者 “Spam”) 转化成数值型的表型，以便 Spark ML 处理。

使用 Word2Vec 将短信文本转化成数值型词向量。

使用 MultilayerPerceptronClassifier 训练一个多层感知器模型。

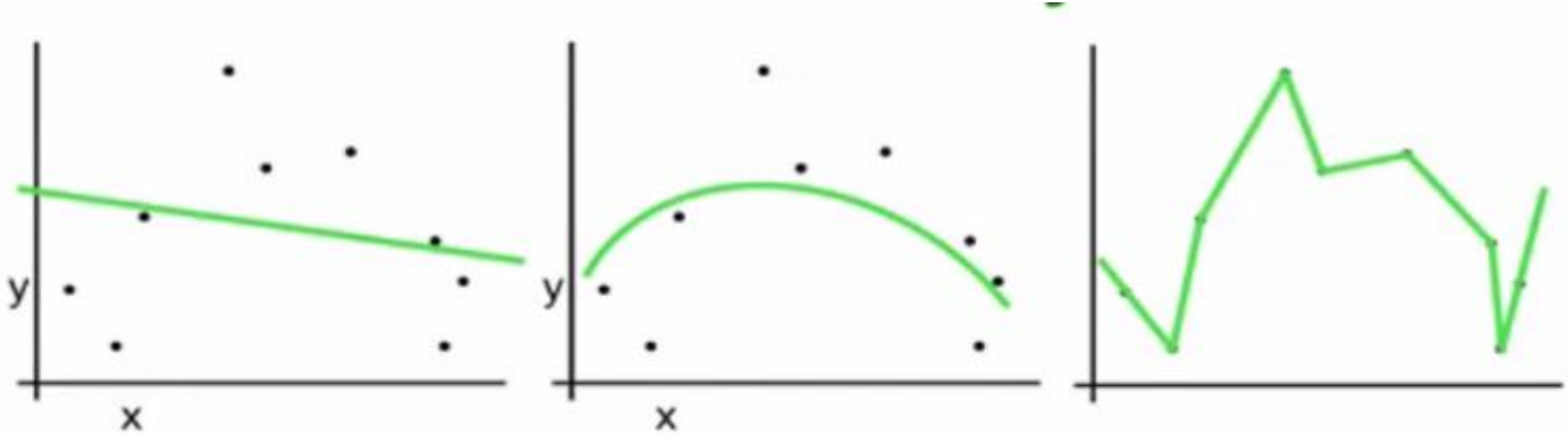
使用 LabelConverter 将预测结果的数值标签转化成原始的文本标签。

最后在测试数据集上测试模型的预测精确度。


cross-validation

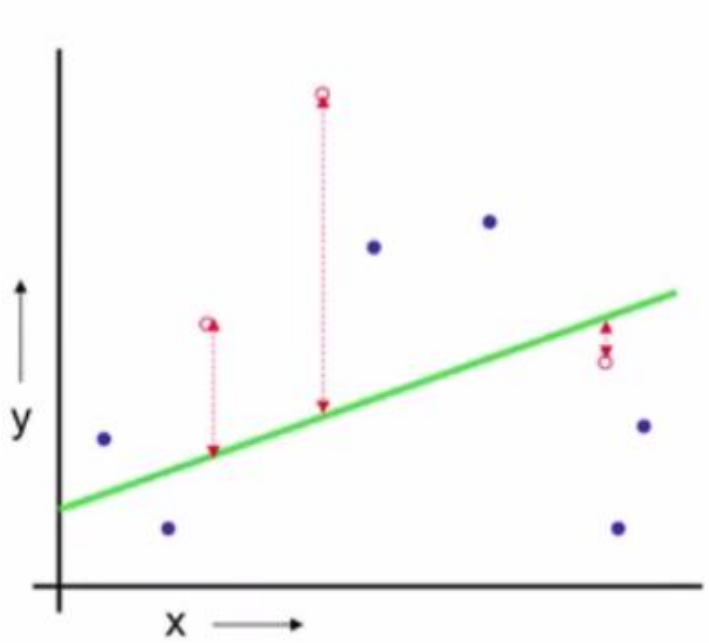
Cross-validation

- Spark中采用是k折交叉验证（k-fold cross validation）。举个例子，例如10折交叉验证(10-fold cross validation)，将数据集分成10份，轮流将其中9份做训练1份做验证，10次的结果的均值作为对算法精度的估计。作用是通过模型评估帮我们选择针对当前数据集的比较好的模型。



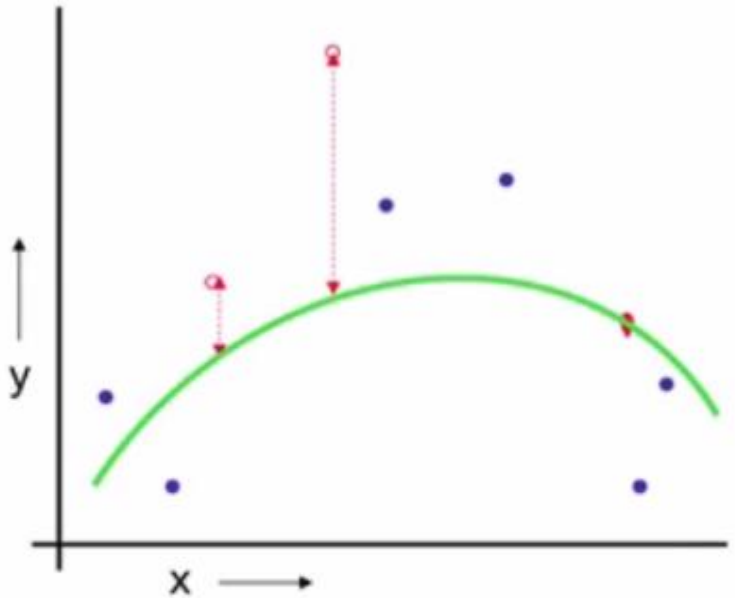
Cross-validation

- 分别计算三种模型的MSE 值



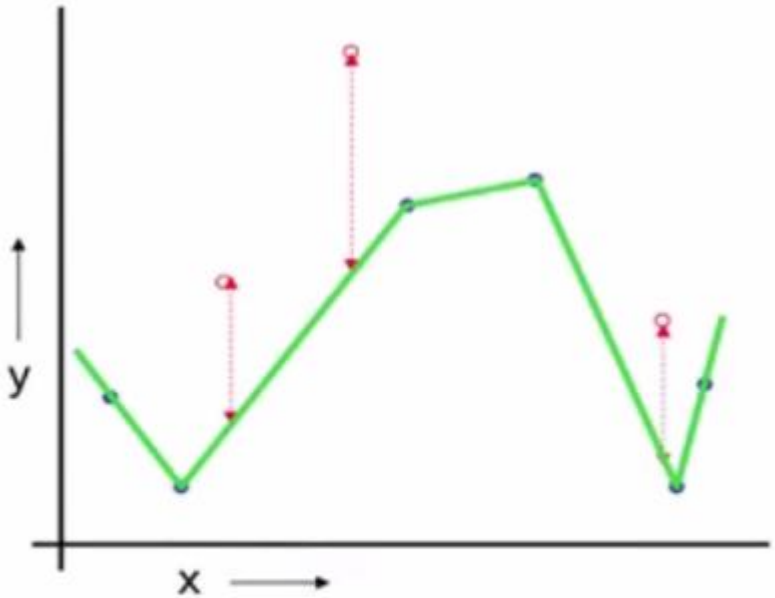
(Linear regression example)

Mean Squared Error = 2.4



(Quadratic regression example

Mean Squared Error = 0.9

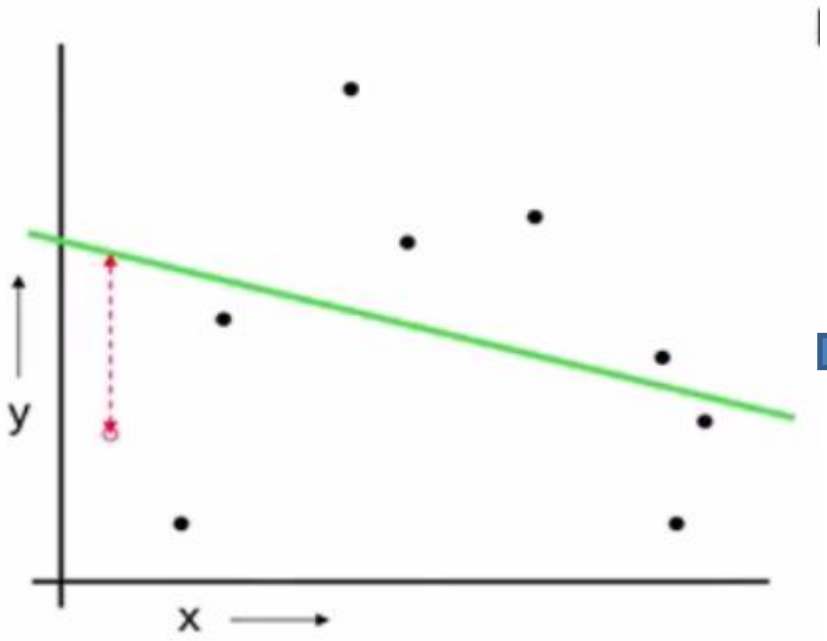


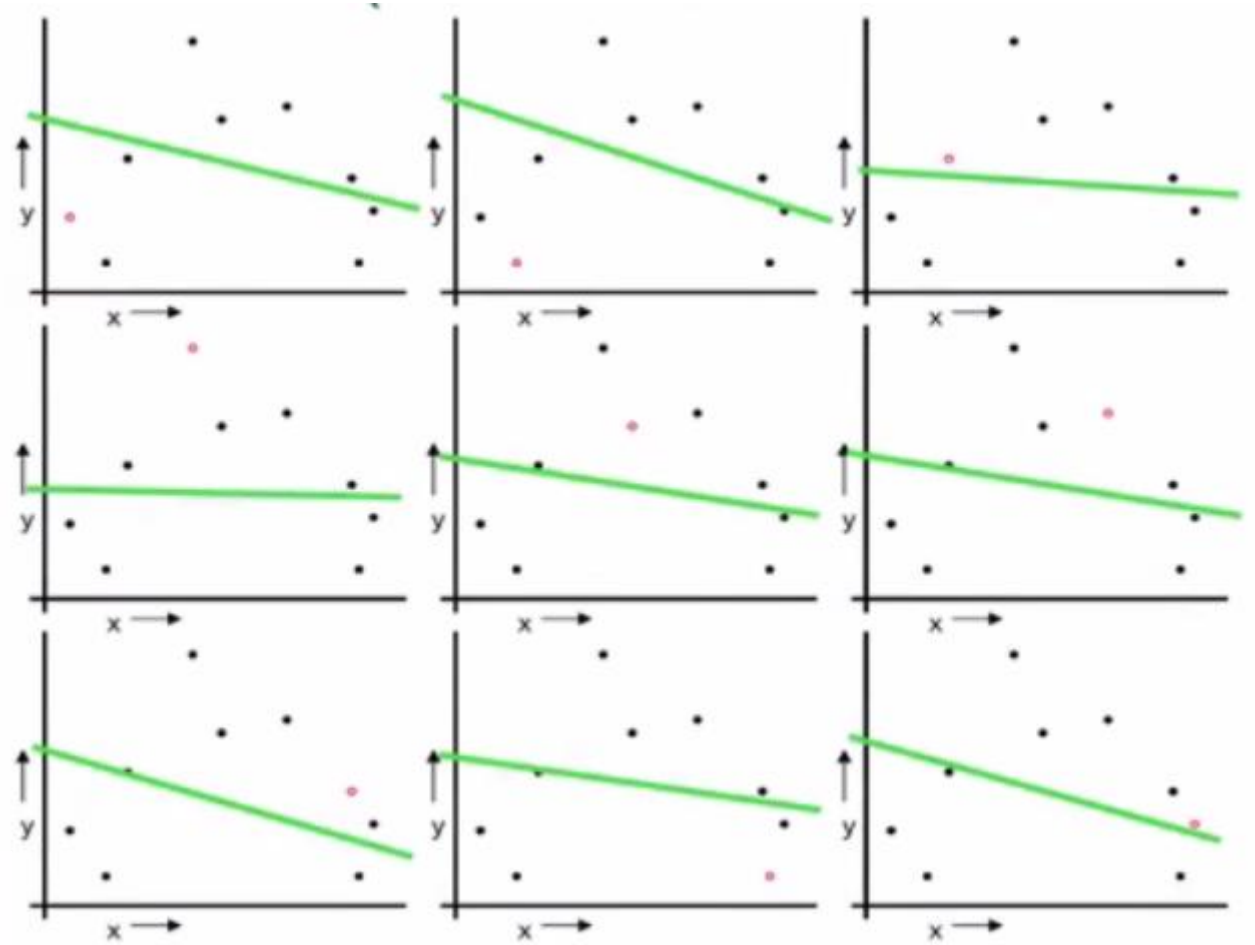
(Join the dots example)

Mean Squared Error = 2.2

Cross-validation-Leave One Out Cross Validation

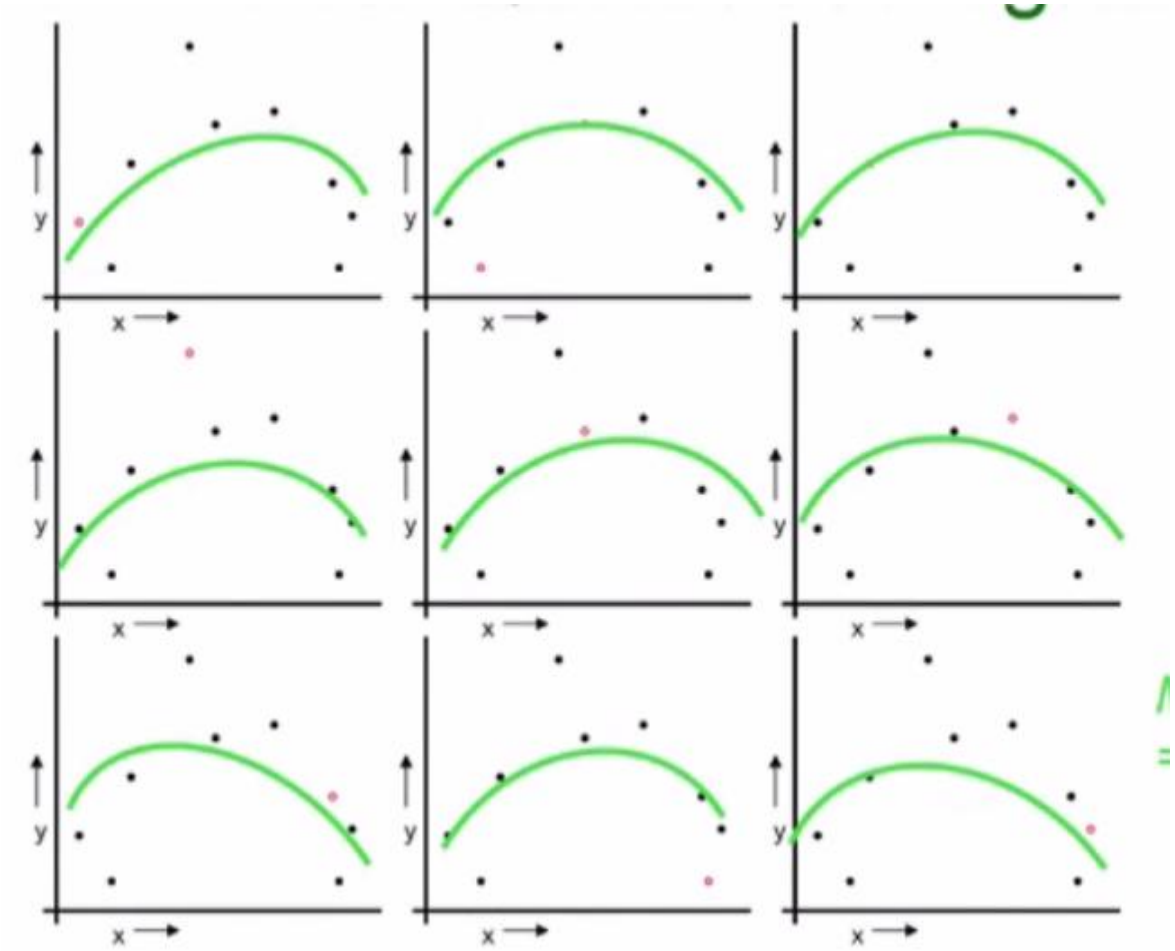
- 分别拿出一个点作为验证点，求平均误差



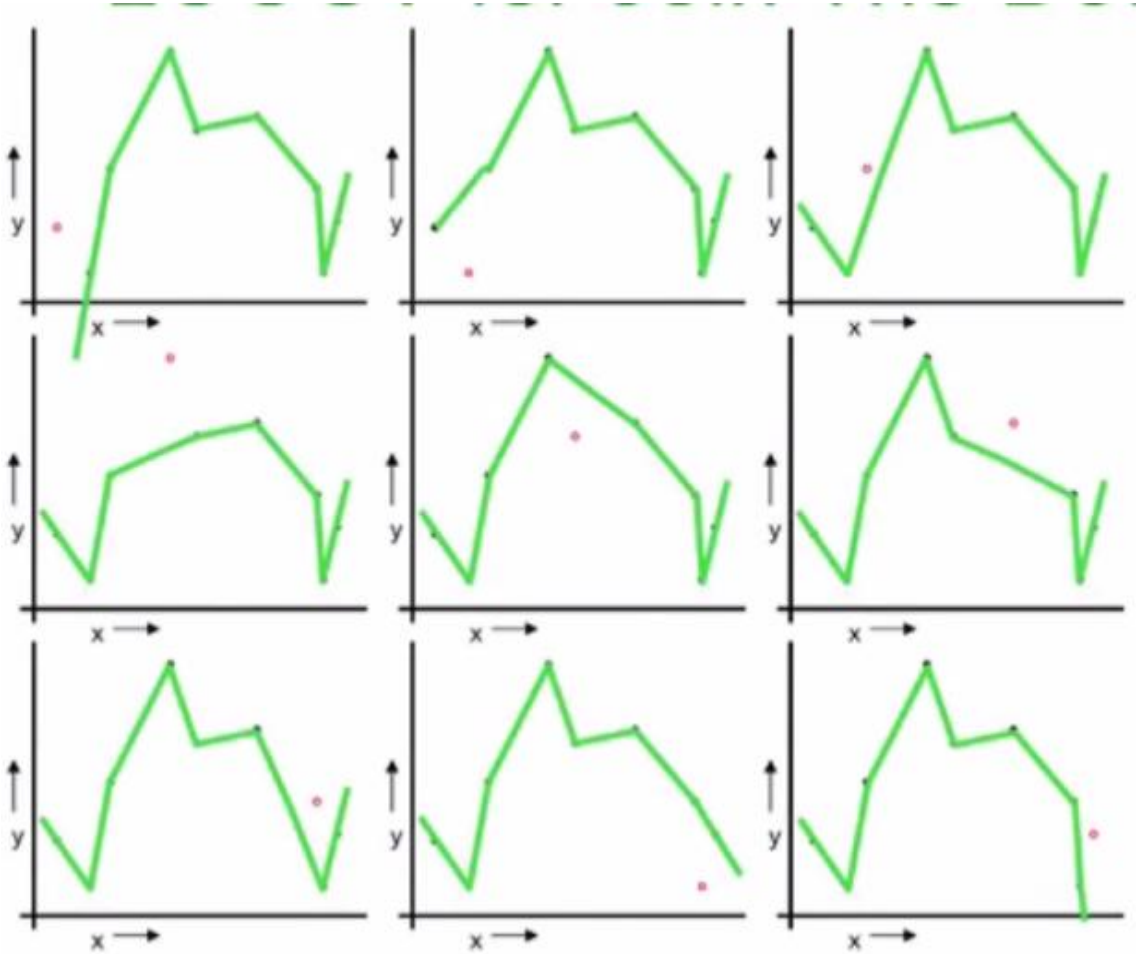


$MSE_{LOOCV} = 2.12$

Cross-validation-Leave One Out Cross Validation

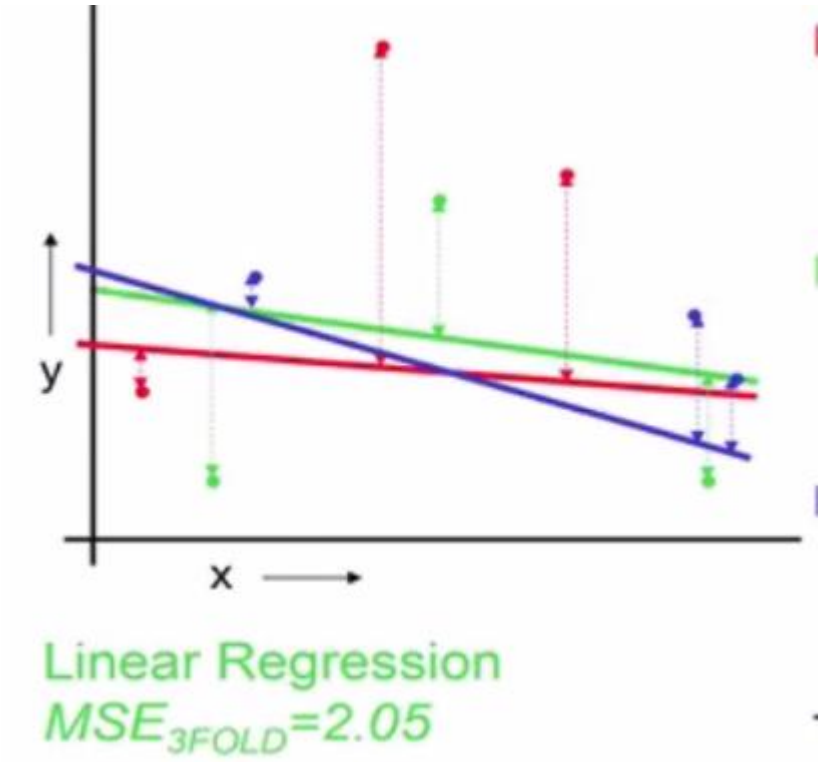


MSE_{LOOCV} =0.962



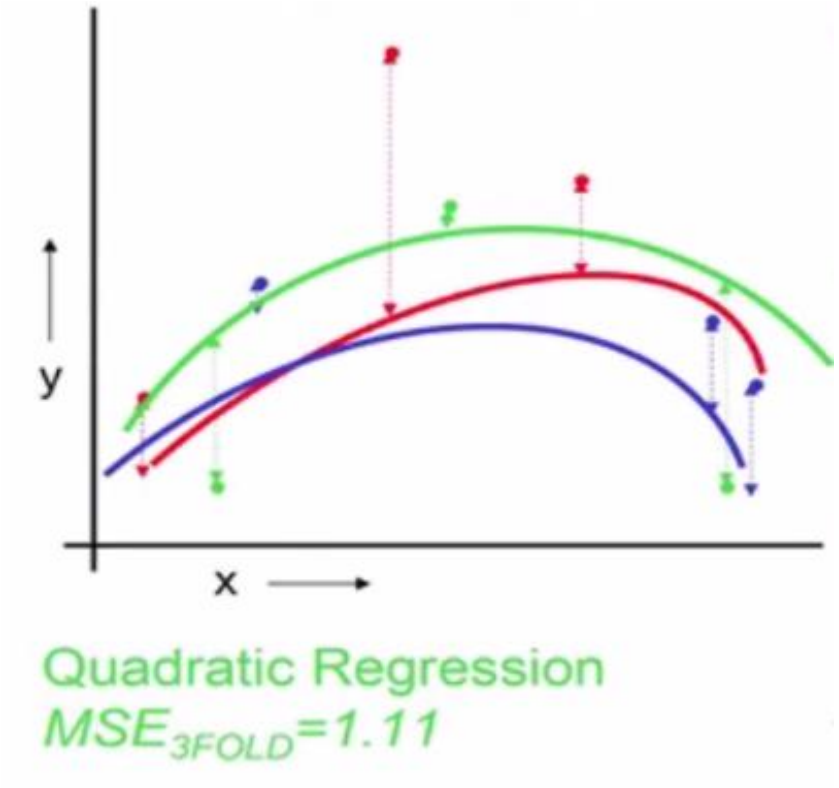
MSE_{LOOCV} =3.33

Cross-validation-k-fold Cross validation

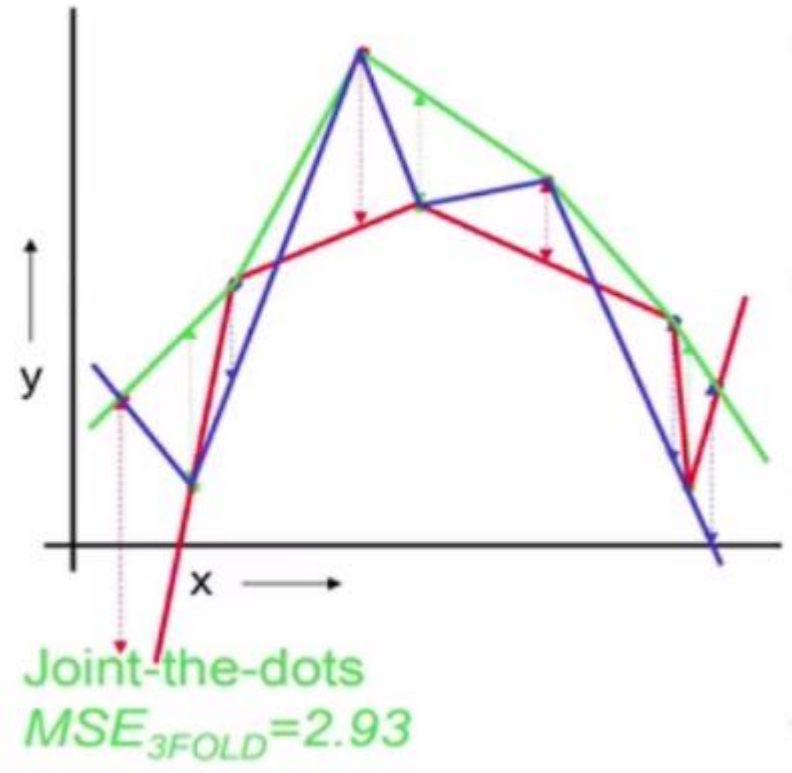


- 把数据集分成三个partition，用三种颜色表示三个partition，用蓝色的点和绿色的点训练模型，然后用红色点计算误差。
- 分别计算蓝色 和绿色误差求平均值

Cross-validation-k-fold Cross validation



- 把数据集分成三个partition，用三种颜色表示三个partition，用蓝色的点和绿色的点训练模型，然后用红色点计算误差。
- 分别计算蓝色 和绿色误差求平均值



Cross-validation for Spark

The `spark.ml` package aims to provide a uniform set of high-level APIs built on top of DataFrames that help tune practical machine learning pipelines. See the algorithm guides section below for guides on sub-packag including feature transformers unique to the Pipelines API, ensembles, and more.

Table of contents


习 题

习题

- 读取pm.csv,将含有缺失值的行扔掉（或用均值填充）将数据集分为两部分，0.8比例作为训练集，0.2比例作为测试集

（1）使用month,day,hour,DEWP,TEMP,PRES,cbwd,Iws,Is,Ir作为特征列（除去No，year，pm），pm作为label列，使用训练集、随机森林算法进行回归建模，使用回归模型对测试集进行预测，并评估。

（2）按照下面标准处理pm列，数字结果放进（levelNum）列，字符串结果放进（levelStr）列

优（0） 50　　**良**（1）50~100　　**轻度污染**（2） 100~150

中度污染（3） 150~200　　**重度污染**（4） 200~300　　**严重污染**（5） 大于300及以上

（3）使用month,day,hour,DEWP,TEMP,PRES,cbwd,Iws,Is,Ir作为特征列（除去No，year，pm）,levelNum作为label列，使用训练集、随机森林算法进行分类建模。使用分类模型对测试集进行预测

对预测结果df进行处理，基于prediction列生成predictionStr（0-5转换优-严重污染）,对结果进行评估