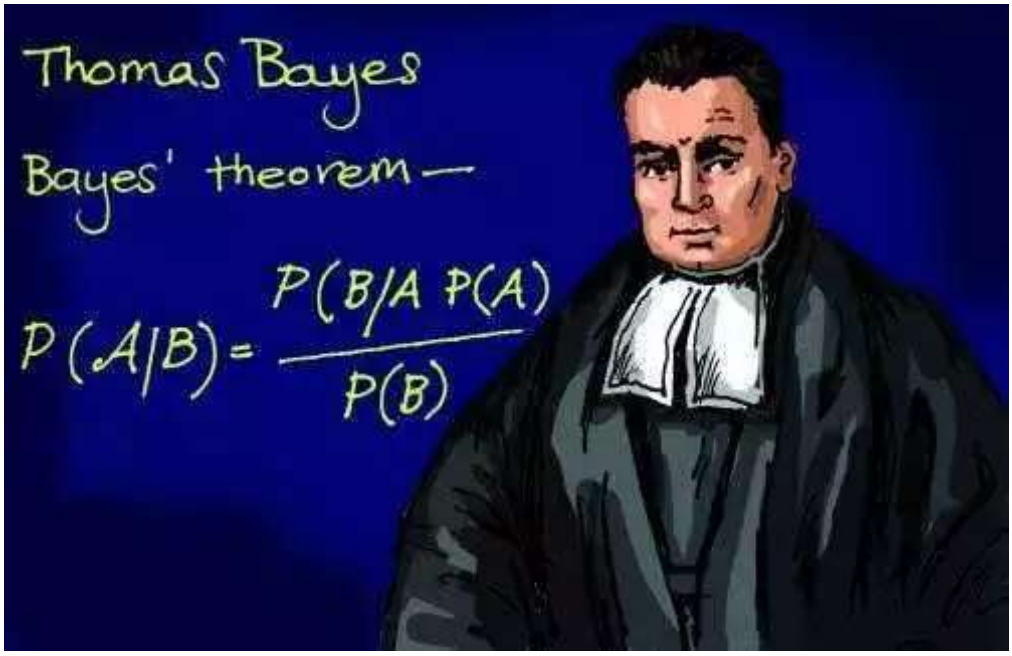
开课吧数据竞赛第四课-钟老师-20191110

笔记本：	开课吧-小钟讲课		
创建时间：	2019/11/4 星期一 15:24	**更新时间：**	2019/11/10 星期日 2:17
作者：	你看起来好像很好吃n_n		
URL：	https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing		

开课吧-数据竞赛及相关问题从小工到专家

时间：2019-11-10



1. 贝叶斯分类器

1.1 贝叶斯分类引入

贝叶斯公式：当分析样本大到接近总体数时，样本中事件发生的概率将接近于总体中事件发生的概率。
假设X,Y是一对随机变量，它们的联合概率P（X=x,Y=y)是指X取值x且Y取值y的概率，条件概率是指一个随机变量在另一个随机变量取值已知的情况下取某一特定值的概率。例如，条件概率P(Y=y|X=x）是指变量X在取值x的情况下，变量Y取值y的概

率。X和Y的联合概率和条件概率如下关系：
P（X,Y）=P(Y|X)×P（X）=P(X|Y)×P（Y）
贝叶斯定理：

$$P(Y \mid X)=\frac{P(X \mid Y) P(Y)}{P(X)}$$

example:考虑两队之间的足球比赛：队0和队1，假设65%的比赛队0获胜，剩余的比赛队1获胜。队0获胜的比赛只有30%是队1的主场，而队1取胜的比赛中75%获胜。如果下一场比赛在队1的主场进行，那一支球队最有可能获胜:随机变量X代表东道主，随机变量Y代表比赛的胜利者。X,Y都在（0,1）中取值。
队伍0取胜的概率是P(Y=0)=0.65
队伍1取胜的概率是P(Y=1)=1-P(Y=0)=0.35
队伍1取胜作为东道主的概率是P(X=1|Y=1)=0.75
队伍0取胜企鹅队1作为东道主的概率是P(X=1|Y=0)=0.3
我们的目的是计算P(Y=1|X=1），即队1在主场获胜的概率，并与P（Y=0|X=1）比较：

$$\begin{aligned}
P(Y=1 \mid X=1) &= \frac{P(X=1 \mid Y=1) \times P(Y=1)}{P(X=1)} \\
&= \frac{P(X=1 \mid Y=1) \times P(Y=1)}{P(X=1, Y=1)+P(X=1, Y=0)} \\
&= \frac{P(X=1 \mid Y=1) \times P(Y=1)}{P(X=1 \mid Y=1) P(Y=1)+P(X=1, Y=0)\ P(Y=0)} \\
&= \frac{0.75 \times 0.35}{0.75 \times 0.35+0.3 \times 0.65} \\
&= 0.5738
\end{aligned}$$

贝叶斯变量在分类中的应用：
设X表示属性集，Y表示类变量。如果类变量和属性之间的关系不确定，那么我们可以把X和Y看做随机变量，用P(Y|X)以概率的方式捕捉二者之间的关系。这个条件概率又称为Y的后验概率，与之相对地，P(Y)称为Y的先验概率。
通过使用P(Y)、类的条件概率P(X|Y)和证据P（X）来表示后验概率：

$$P(Y \mid X)=\frac{P(X \mid Y) P(Y)}{P(X)}$$

1.1 朴素贝叶斯分类器

给定类标号y，朴素贝叶斯分类器在估计类条件概率时假设属性之间条件独立。条件独立假设可以形式化表述为：

$$P(X \mid Y=\mathrm{y})=\prod_{i=1}^{d} P(X_i \mid Y=y)$$

条件独立性：

$$P(X,Y|Z)=P(X|Z)\times P(Y|Z)$$

朴素贝叶斯分类器：

有了条件独立的假设，不必计算X的每一个组合的类的条件概率，只需对给定的Y，计算每一个Xi的条件概率。朴素贝叶斯分类器对每个类Y计算后验概率:

$$P(Y|X)=\frac{P(Y)\prod_{i=1}^{d}P(X_i|Y)}{P(X)}$$

对于所有的Y，P(X)是固定的，因此只要找到使

$$P(Y)\prod_{i=1}^{d}P(X_i|Y)$$

最大的类就足够了。

1. 分类属性的条件概率我们上面已经计算过；
2. 估计联系属性的条件概率：

   1）连续特征特征离散化，后面和分类属性类似

   2）假设连续变量服从某种概率分布，然后使用训练数据估计分布的参数。高斯分布通常被用来表示连续属性的类条件概率分布。该分布有两个参数，均值u和方差σ2，对于每个类yj和属性Xi的类条件概率等于：

$$P(X_i=x_i|Y=y_j)=\frac{1}{\sqrt{2\pi}\sigma_{ij}}e^{\frac{(x_i-u_{ij})^2}{\sigma_{ij}^2}}$$

   给定以下测试记录有以下属性集：X=（有房=否，婚姻状况=已婚，年收入=$120K）需要根据训练集中的可用信息计算后验概率P(Yes|X)和P(No|X）如果P(Yes|X）>P(No|X)，那么记录为Yes,反之，分类为No.

Tid	有房	婚姻状况	年收入	拖欠贷款
1	是	单身	125K	否
2	否	已婚	100K	否
3	否	单身	70K	否
4	是	已婚	120K	否
5	否	离婚	95K	是
6	否	已婚	60K	否
7	是	离婚	220K	否
8	否	单身	85K	是
9	否	已婚	75K	否
10	否	单身	90K	是

   其中年收入属性关于类No否的属性计算如下：

$$\bar{x} = \frac{125+100+70+\cdots+75}{7} = 110$$

$$s^2 = \frac{(125-110)^2+(100-110)^2+\cdots+(75-110)^2}{7(6)} = 2975$$

$$s = \sqrt{2975} = 54.54$$

相应其他类条件概率汇总：

P(有房=是|No) =3/7
P(有房=否|No) =4/7
P(有房=是|Yes) =0
P(有房=否|Yes) =1
P(婚姻状况=单身|No) =2/7
P(婚姻状况=离婚|No) =1/7
P(婚姻状况=已婚|No) =4/7
P(婚姻状况=单身|Yes) =2/3
P(婚姻状况=离婚|Yes) =1/3
P(婚姻状况=已婚|Yes) =0

年收入：
如果类=No： 样本均值=110
样本方差=2975
如果类=Yes： 样本均值=90
样本方差=25

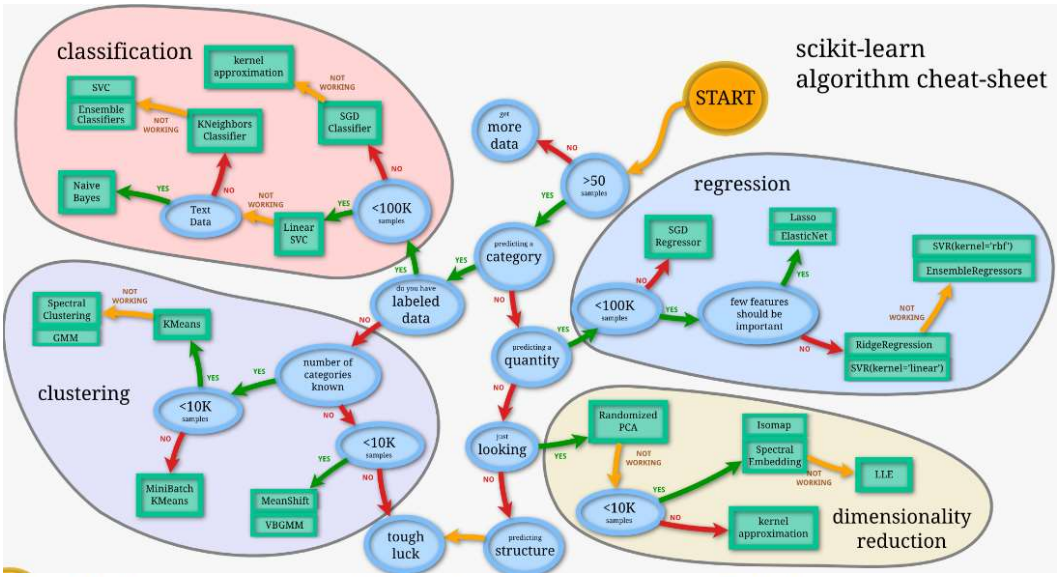
则预测X=（有房=否，婚姻状况=已婚，年收入=$120K）计算后验概率

$$P(Y)\prod_{i=1}^{d} P(X_i \mid Y)$$

- P(No|X)=P(有房=否|No)×P(婚姻状况=已婚|No)×P（年收入=$120K|No）=4/7×4/7×0.0072=0.0024
- P(Yes|X)=P(有房=否|Yes)×P(婚姻状况=已婚|Yes)×P（年收入=$120K|Yes）=1×0×1.2×10-9=0

放到一起的可以得到类No的后验概率P（No|X）=α×7/10×0.0024=0.0016α其中α=1/P(x)是个常量。类0概率为0这样P(No|X)>P(Yes|X）,所以记录分类为No.

2. sklearn部分包介绍



Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

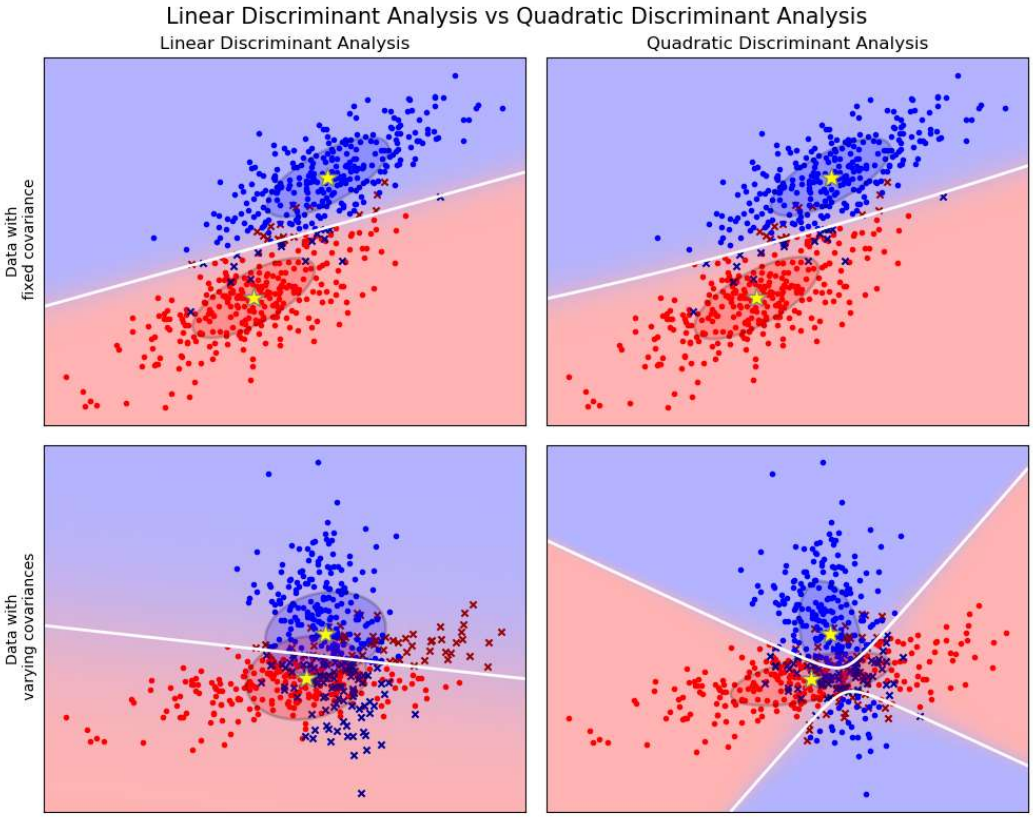
Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

官网地址：[scikit-learn]

2.1 监督学习-Supervised learning

2.1.1 Generalized Linear Models广义线性模型

2.1.1.1 线性判别分析法(discriminant_analysis.LinearDiscriminantAnalysis）和二次判别分析法(discriminant_analysis.QuadraticDiscriminantAnalysis)



两种方法很好计算，无超参数：

LDA（线性判别分析法）和QDA（二次判别分析法）能够使用一个简单的概率模型来分别转化得出。这个模型是关于每一类k 中关于数据概率 P(X|y = k) 的条件分布。然后可以通过使用贝叶斯来获得预测结果：

$$P(y=k|X) = \frac{P(X|y=k)P(y=k)}{P(X)} = \frac{P(X|y=k)P(y=k)}{\sum_l P(X|y=l)\cdot P(y=l)}$$

然后我们选择类别k，该类别使该条件概率最大化。
更具体地说，对于线性和二次判别分析，P（X | y）被建模为具有密度的多元高斯分布：

$$P(X|y=k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}}\exp\left(-\frac{1}{2}(X-\mu_k)^t\Sigma_k^{-1}(X-\mu_k)\right)$$

```
import numpy as np
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
 X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
y = np.array([1, 1, 1, 2, 2, 2])
clf = LinearDiscriminantAnalysis()
clf.fit(X, y)
print(clf.predict([[-0.8, -1]]))
```

2.1.2 SVM

2.1.3 Naive Bayes

代码:

```
from sklearn import datasets
iris = datasets.load_iris()
from sklearn.naive_bayes import GaussianNB
 gnb = GaussianNB()
y_pred = gnb.fit(iris.data, iris.target).predict(iris.data)
print("Number of mislabeled points out of a total %d points : %d"...
% (iris.data.shape[0],(iris.target != y_pred).sum()))
```

2.1.3 Feature selection

2.1.3.1 Removing features with low variance 删除低方差特征:

```
 from sklearn.feature_selection import VarianceThreshold
X = [[0, 0, 1], [0, 1, 0], [1, 0, 0], [0, 1, 1], [0, 1, 0], [0, 1, 1]]
sel = VarianceThreshold(threshold=(.8 * (1 - .8)))
sel.fit_transform(X)
```

2.1.3.2 单变量特征选择:Univariate feature selection

SelectKBest 保留评分最高的 K 个特征
SelectPercentile 保留最高得分百分比之几的特征
对每个特征应用常见的单变量统计测试: 假阳性率（false positive rate） SelectFpr, 伪发现率（false discovery rate） SelectFdr , 或者族系误差（family wise error） SelectFwe 。
GenericUnivariateSelect 允许使用可配置方法来进行单变量特征选择。它允许超参数搜索评估器来选择最好的单变量特征。
这些对象将得分函数作为输入，返回单变量的得分和 p 值 （或者仅仅是 SelectKBest 和 SelectPercentile 的分数）:
对于回归: f_regression , mutual_info_regression
对于分类: chi2 , f_classif , mutual_info_classif

2.1.3.3 Tree-based feature selection 基于树的特征选择

```
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectFromModel
iris = load_iris()
X, y = iris.data, iris.target
X.shape(150, 4)
```

```
clf = ExtraTreesClassifier(n_estimators=50)
clf = clf.fit(X, y)
clf.feature_importances_
model = SelectFromModel(clf, prefit=True)
X_new = model.transform(X)
X_new.shape
```

2.2 模型选择和评估Model selection and evaluation

2.2.1 Cross-validation: 交叉验证

2.2.2 Tuning the hyper-parameters of an estimator 调参

2.2.3 Model evaluation: quantifying the quality of predictions 模型效果评估

2.3 Preprocessing data 预处理数据

2.3.1 Standardization, or mean removal and variance scaling 标准化，或均值去除和方差缩放

2.3.1.1 标准化

```
from sklearn import preprocessing
import numpy as np
 X_train = np.array([[ 1., -1.,  2.],[ 2.,  0.,  0.],[ 0.,  1., -1.]])
X_scaled = preprocessing.scale(X_train)
X_scaled
```

通过删除平均值和缩放到单位方差来标准化特征

```
scaler = preprocessing.StandardScaler().fit(X_train)
```

min_max_scaler 主要是为了train 和test分布保持一致

```
min_max_scaler = preprocessing.MinMaxScaler()
X_train_minmax = min_max_scaler.fit_transform(X_train)
```

max_abs_scaler 通过除以每个特征中的最大值来将训练数据缩放到[-1，1]范围内。它适用于已经以零或稀疏数据为中心的数据。

```
max_abs_scaler = preprocessing.MaxAbsScaler()
X_train_maxabs = max_abs_scaler.fit_transform(X_train)
```

将值映射到高斯分布：

```
quantile_transformer = preprocessing.QuantileTransformer(...
output_distribution='normal', random_state=0)
 X_trans = quantile_transformer.fit_transform(X)
```

2.3.2 Encoding categorical features

preprocessing.OrdinalEncoder()
preprocessing.OneHotEncoder()

2.3.3 Generating polynomial features poly特征

X的特征从（X1，X2）转换为（1，X1，X2，X12，X1X2，X22）：

```
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
X = np.arange(6).reshape(3, 2)
poly = PolynomialFeatures(2)
poly.fit_transform(X)
```

X的特征从（X1，X2，X3）转换为（1，X1，X2，X3，X1X2，X1X3，X2X3，X1X2X3）

```
X = np.arange(9).reshape(3, 3)
 poly = PolynomialFeatures(degree=3, interaction_only=True)
poly.fit_transform(X)
```

3. Kaggle-Instant Gratification： 及时行乐

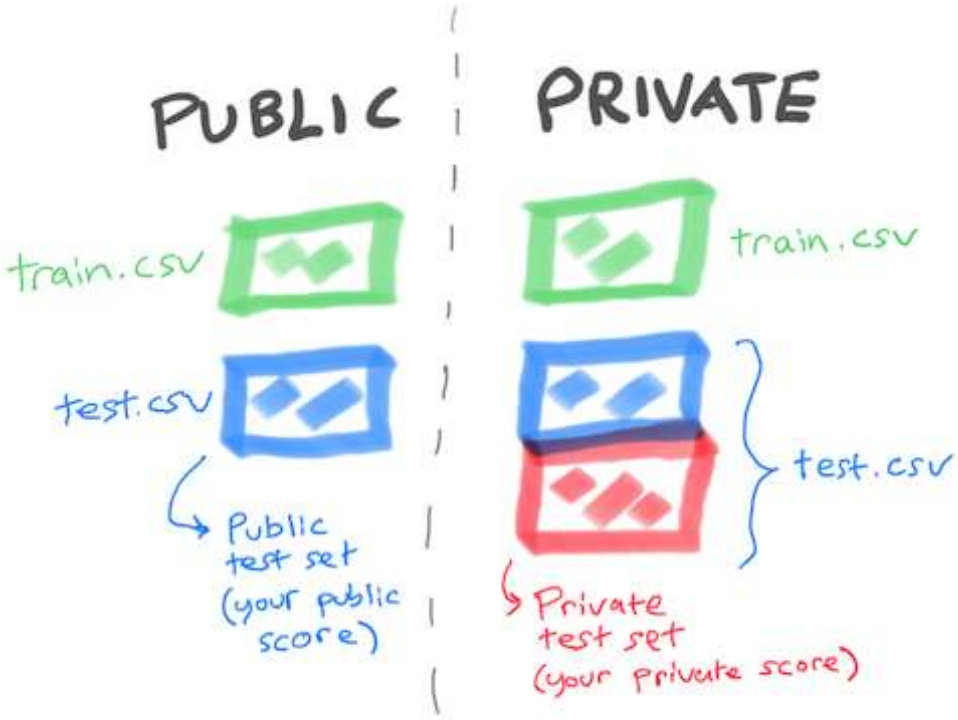
Silly column names abound,愚蠢的列名比比皆是，

but the test set is a mystery.但是测试集是一个谜。

Careful how you pick and slice,小心选择和切片，

or be left behind by history.不然你将被历史遗忘。

kaggle 数据切分：



3.1 数据EDA

发现magic:
train['wheezy-copper-turtle-magic'].value_counts()

3.2 baseline

3.3 QDA

3.4 GMM增加聚类

4. kaggle-Porto Seguro’s Safe Driver Prediction：塞古罗港的安全驾驶员预测-预测驾驶员明年是否会提出保险索赔

4.1 数据EDA

4.2 baseline

4.3 贝叶斯调参

4.3 进一步特征工程

5.开始标准化我们的模块吧