数据科学

机器学习

有监督学习

分类

- 决策树（Decision Tree）
- 朴素贝叶斯
- KNN（K 近邻）
- SVM（支持向量机）
- 神经网络
- 集成学习

决策树(Decision Tree)

```
from sklearn import tree
```

模型创建 `clf = tree.DecisionTreeClassifier()`

模型学习 `clf=clf.fit(x,y)`

模型预测 `y = predict(X)`

生成决策树

```
dot_data = tree.export_graphviz(clf,feature_names = [...])
```

```
from sklearn.tree import export_text
clf=tree = export_text(clf,feature_names = [...])
```

注意事项 必须使用数据 astype(int) 转换为整数类型

支持向量机SVM

```
from sklearn import svm
```

模型创建 `clf = svm.SVC(kernel='...',gamma='...',C='...')`

- kernel 核函数
  - linear 线性
  - poly 多项式
  - 高斯核函数 rbf(gamma 参数决定多项式的最高阶数)
  - rbf 高斯
  - sigmoid 逻辑
- gamma 核函数
  - 一般取 1/n
- C 惩罚系数
  - 一般取 10

模型学习 `clf=clf.fit(x,y)`

模型预测 `y = predict(X)`

注意事项 所有的分类标签都需要是数值类型的整数

随机森林RandomForest

```
from sklearn.ensemble import RandomForestClassifier
```

模型创建 `rf = RandomForestClassifier(n_estimators =7,max_depth =7, random_state = 42,criterion ='')`

固定一个n_estimators(决策树数目)最终结果随机性不断变化增加

max_depth

模型调参步骤 根据数据的最大特征来判断最合适的决策树深度

n_estimators

criterion 决定节点分裂质量的指标 ('gini'=Gini impurity)默认选择

模型学习 `rf.fit(X_train,y_train)`

模型预测 `y_pred = rf.predict(X_test)`

查看特征重要性排序 `feature_importances = rf.feature_importance_`

可视化

回归

- 线性回归
- 逻辑回归

线性回归

```
from sklearn.linear_model import LinearRegression
```

$f(x) = w_0x_0 + w_1x_1 + \cdots + w_nx_n + b$

模型创建 `linreg = LinearRegression()`

模型学习 `clf=(X,y)`

模型预测 `y = predict(X)`

回归系数 `clf.coef_,clf.intercept_`

模型评估 `r_sqlinregscore(X,y),mse(y,clf.predict(X))`

注意事项 **r_sqlinregscore(X,y)**,**mse(y,clf.predict(X))**

逻辑回归

```
from sklearn.linear_model import LogisticRegression
```

$F(x) = P(X \leq x) = \frac{1}{1+e^{-x}}$

模型创建 `logreg = LogisticRegression(max_iter = 1000,random_state = 42)`

模型学习 `clf=(X,y)`

模型预测 `y = predict(X)`

聚类

- 决策聚类原理
- 将为n簇分类问题转化成
- 将n-1簇分类问题的最大簇一分为二
- 重复这个过程直到最终得到n个簇

聚类原理

将为n簇分类问题转化成

将n-1簇分类问题的最大簇一分为二

重复这个过程直到最终得到n个簇

K — means聚类

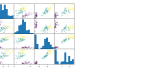
```
from sklearn.cluster import KMeans
```

模型创建 `kmeans = KMeans(n_clusters = 四月 0 到)`

模型学习 `kmeans.fit(X)`

- 获取聚类结果数据 `kmeans.labels_`
- `pd.plotting.scatter_matrix(data_r,c=kmeans.labels_,diagonal='kde')`

- 结果图显示效果



- 为分类效果提供量化指标和标准

模型性能指标 = [2, 3, 4, 5, 6, 7...]

- 轮廓系数 ranking 指标

- 肘部法

降维

PCA，通过少数变量解释原始数据的变异性，**降维是将数据转换成更高层次的抽象特征的一个过程**，并且通过减少数据的维度来保持数据的完整性。降维的目的是为了减少特征的数量，**通过保留最有意义的方差来简化数据**，降低计算成本和复杂性，并且通过将数据投影到较低维度的空间来实现。**降维通常用于处理高维数据集**，例如图像、文本或基因组数据，以提高机器学习模型的效率和性能。降低维度有助于减少过拟合，并且可以帮助揭示数据中的模式和关系。

- 在sklearn.decomposition 中 PCA
- 模型创建 `pca = PCA(n_components =7,whiten = True,...)`
  - n_components 指定主成分的数量
  - whiten 是否对白化处理
- 模型学习 `pca.fit(X)`
- 数据变换 `newX = pca.transform(X)`

- 结果可视化

数据存储

- 保存模型
- 加载数据
- 初步数据
- 特征编码
- 数据标准化

保存模型

```
import joblib
```

模型：`load(pca.pkl)`

数据：`dump(pca.pkl)`

初步数据

```
from sklearn import model_selection
```

```
model_selection.train_test_split(X,y,test_size,random_state)
```

将原始数据集拆分为训练集和测试集

特征编码

```
dum = pd.get_dummies(data['name'],prefix = 'name')
```

数据标准化

```
from sklearn import preprocessing
```

```
X_scale = preprocessing.scale(X)
```

数据评估

- 模型数据评估
- 分类问题
- 回归问题

模型数据评估

- 拟合度（欠/过拟合）

```
from sklearn import metrics
```

```
metrics.mean_squared_error(y,y_pred)
```

- 交叉数据评估 `...score(X,y)`

分类问题

- 混淆矩阵
- 精确率
- 召回率
- F1分数

实际值	预测值	
	Class Yes	Class No
Class Yes	a	b
Class No	c	d

- 混淆矩阵评估模型分类性能的指标 `accuracy = metrics`
- 混淆矩阵计算模型预测为Yes的样本中实际正确的比例 `Precision = a/(a+c)`
- 召回率计算模型实际为Yes的样本中模型正确预测的比例 `Recall = a/(a+b)`
- F1分数是精确率和召回率的调和平均值 `F1 = metrics`

```
from sklearn import metrics
```

- 创建混淆矩阵 `metrics.confusion_matrix(y_pred,y)`
- 分类报告 `metrics.classification_report(y,predicted_y)`
- `Accuracy = clf.score(X,y)`

回归问题

- 可解释性
- 残差分析

- 可解释性
  - `metrics.explained_variance_score(y,y_pred)`
  - 决定系数（R²）`metrics.r2_score(y,y_pred)`
  - 调整决定系数（Adjusted R²）`metrics.adjusted_r2_score(y,y_pred,len(X[0]))`
  - 调整决定系数考虑了模型中自变量的数量，通过对决定系数进行调整，以反映模型的复杂性。调整后的 R² 通常用于比较具有不同自变量数量的模型。
  - `metrics.explained_variance_score(y,y_pred)`
  - 解释方差分数衡量模型能够解释数据中的方差比例。解释方差分数通常用于评估模型的预测能力，特别是在存在多个自变量的情况下。
- 残差分析
  - 残差图 `sns.residplot(y,y_pred)`
  - 残差直方图 `sns.distplot(residuals)`
  - 残差Q-Q图 `sm.qqplot(residuals,line='s')`
  - 残差自相关图 `sm.tsaplots.plot_acf(residuals,lags=40)`
  - 残差图应该呈现随机分布，没有明显的模式或趋势。残差直方图应该呈现正态分布，Q-Q图应该接近对角线，自相关图应该没有显著的自相关性。
  - 残差分析有助于检测模型的假设是否成立，并且可以帮助发现模型的问题和改进方向。