

机器学习简介

主要内容

- u 什么是机器学习？
- u 机器学习简要发展历程
- u 机器学习三大要素与基础知识

什么是机器学习？

u 有这么一组数列：

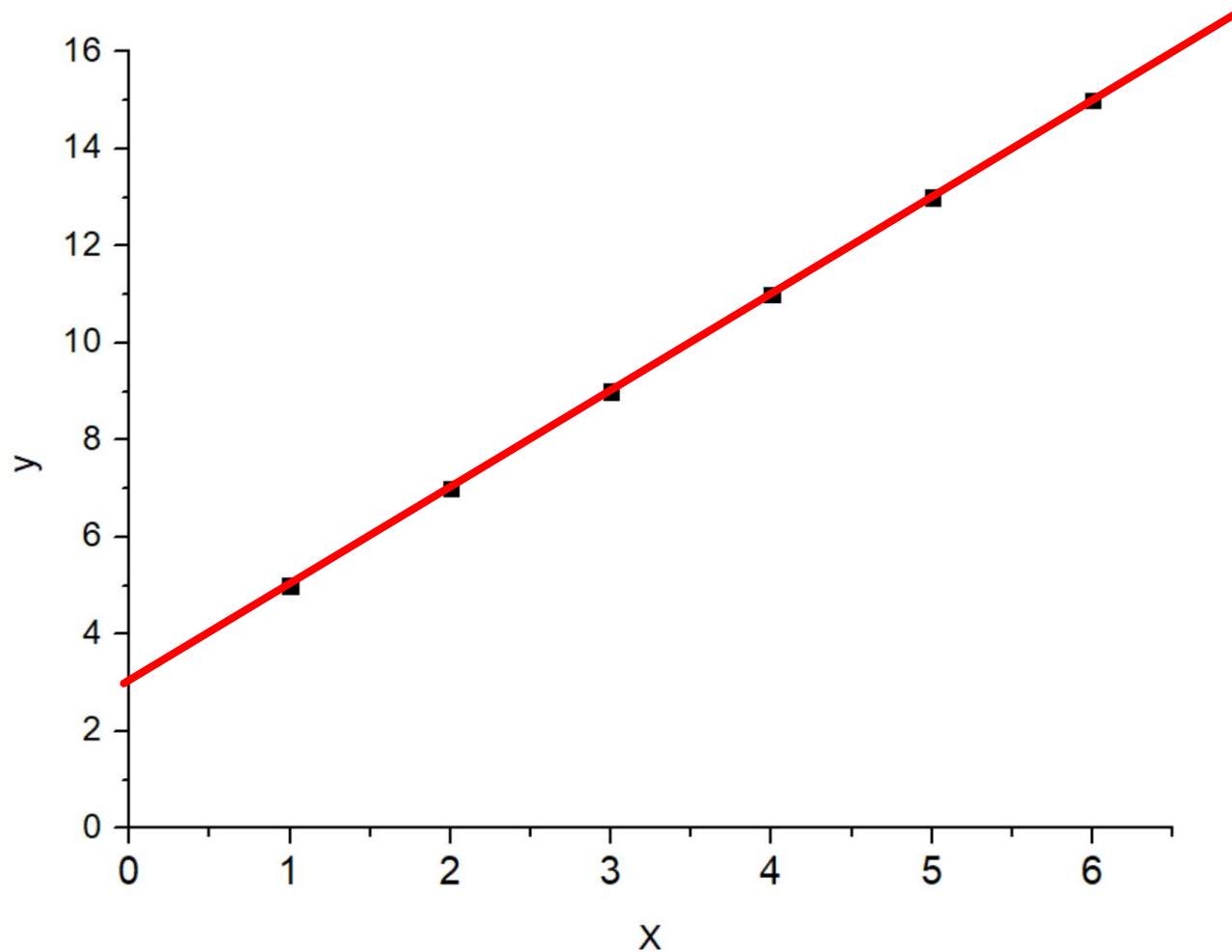
u 5, 7, 9, 11, 13, . . .

接下来一个数字是什么？

第n个数字是什么？

x	y
1	5
2	7
3	9
4	11
5	13
6	15
n	$2n + 3$

最简单的机器学习模型-线性拟合



寻求高性能有机太阳能分子

The Harvard Clean Energy Project

Alan Aspuru-Guzik

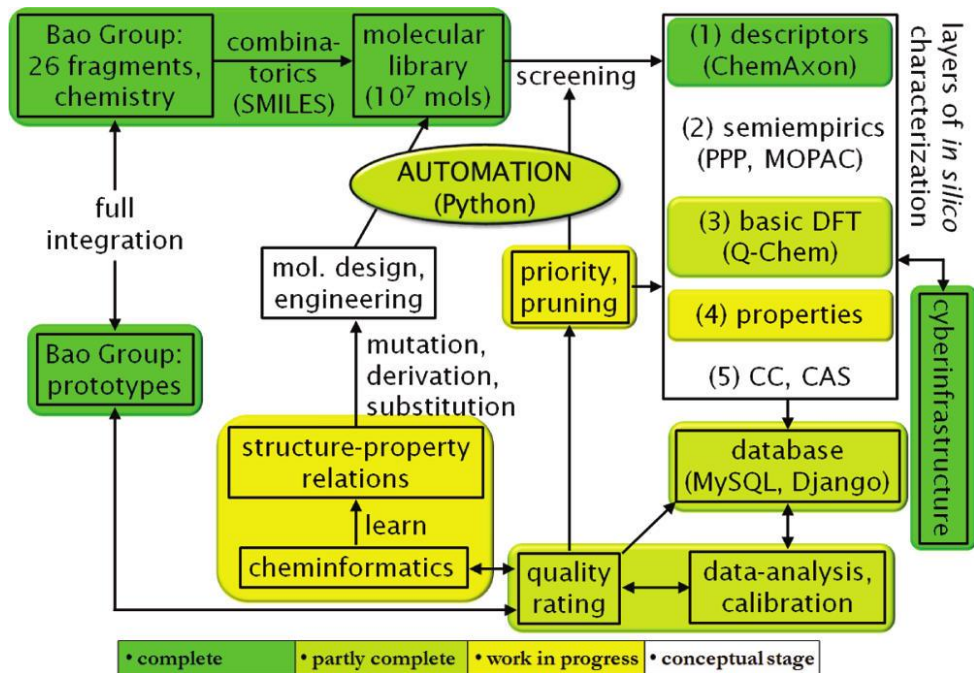


Table 3 Summary of linear fitting results for each of the properties we study. We compare the coefficients of determination (R^2) using all 33 descriptors (all desc.) and the statistically significant ones. The number of significant descriptors ranges from 15–20, but the R^2 is not largely affected in all cases

Property	R^2 (all desc.)	Descriptors	R^2
V_{oc}	0.9580	20	0.9455
J_{sc}	0.9202	18	0.8989
%PCE	0.8937	15	0.8409
FF	0.6567	20	0.6170
$V_{oc}J_{sc}$	0.9025	20	0.8809

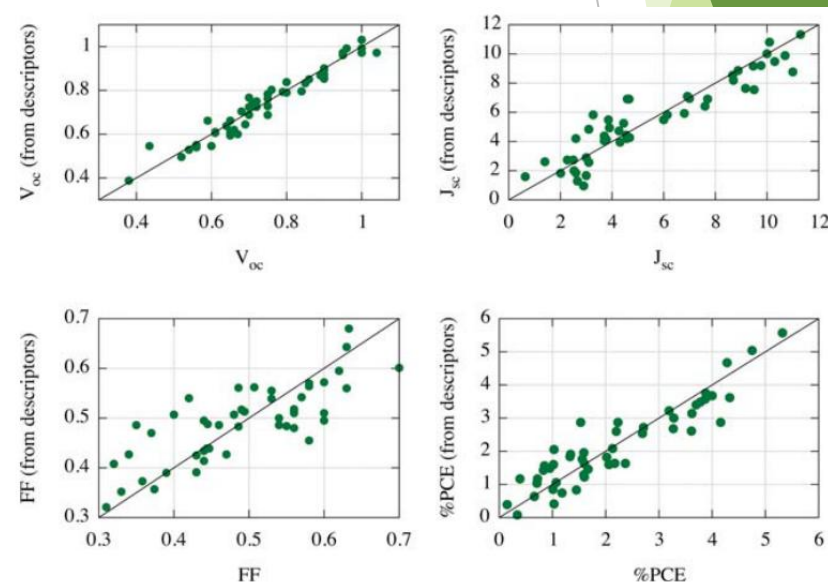
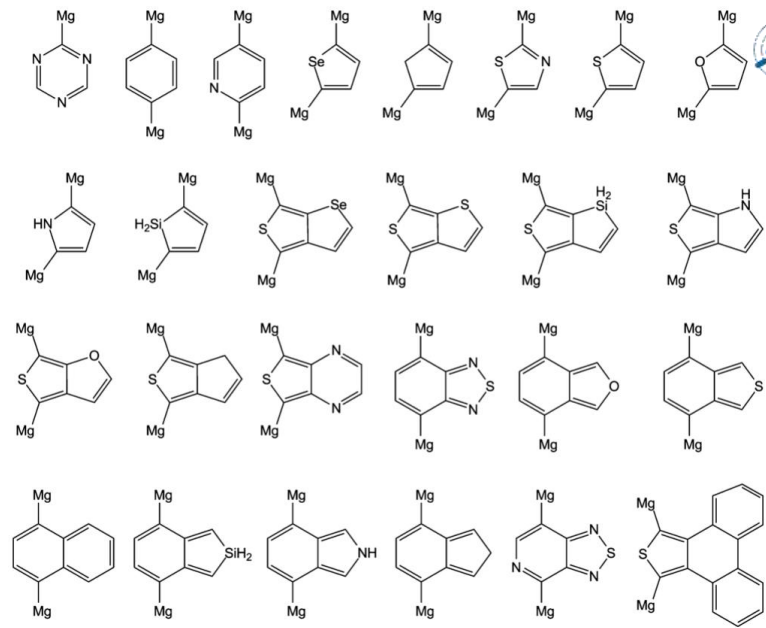


Fig. 3 Results for the multiple linear regression of the four models and the values of the training set. The predicted value from fitted descriptors is compared to the experimental value originally used for fitting. Units are mA cm^{-2} for J_{sc} , and V for V_{oc} .

J. Chem. Theory Comput. **2010**, 6, 135–144

J. Phys. Chem. Lett. **2011**, 2, 2241–2251

Energy Environ. Sci., **2011**, 4, 4849

有这么一组数列：

1, 3, 7, 15, 31, 63, ?

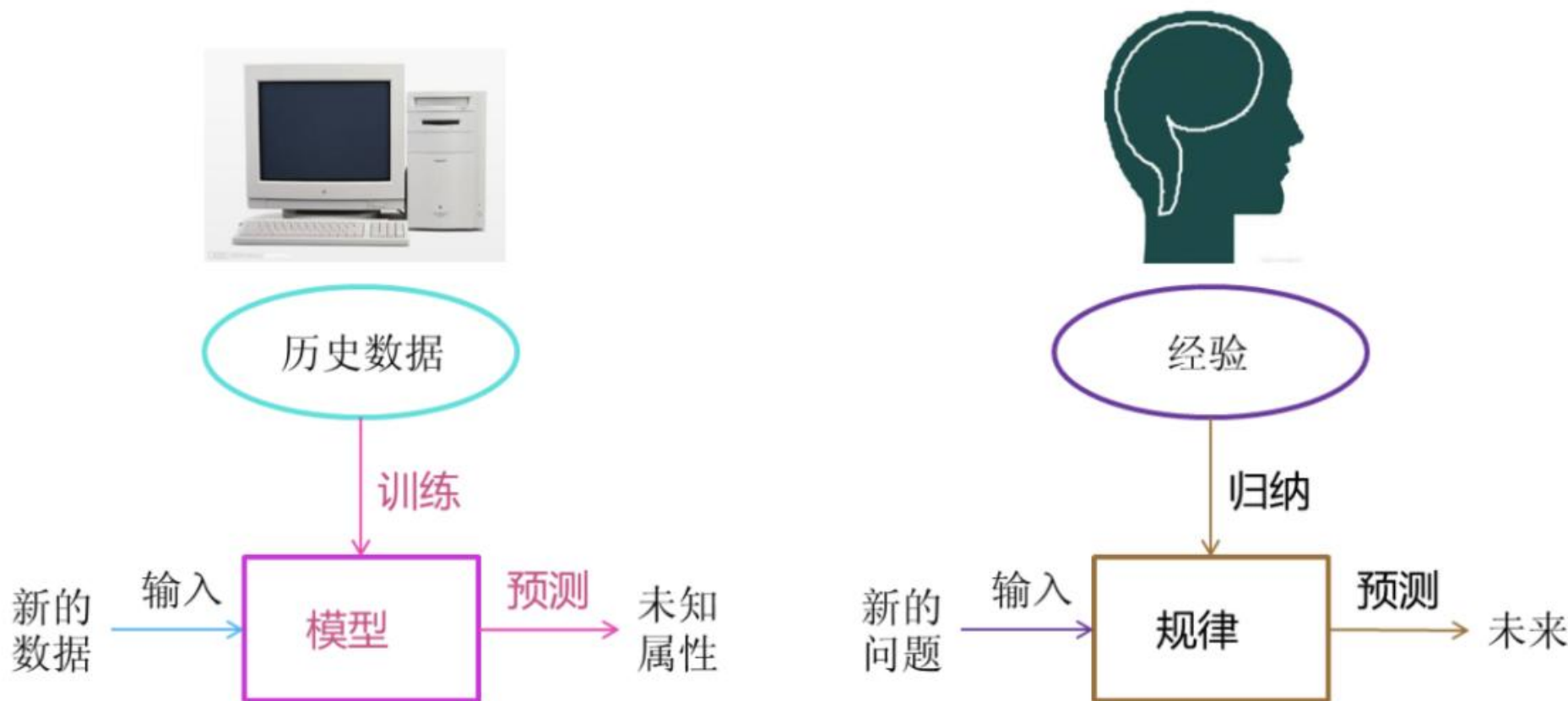
接下来一个数字是什么？

怎么预知其他任意数字？

x	y
1	1
2	3
3	7
4	15
5	31
6	63
n	$2^n - 1$

什么是机器学习？

- 计算机系统能够从数据中自动学习，发现其中的规律，并利用这些规律对新的或未知的数据进行预测和决策。



机器学习的定义

01

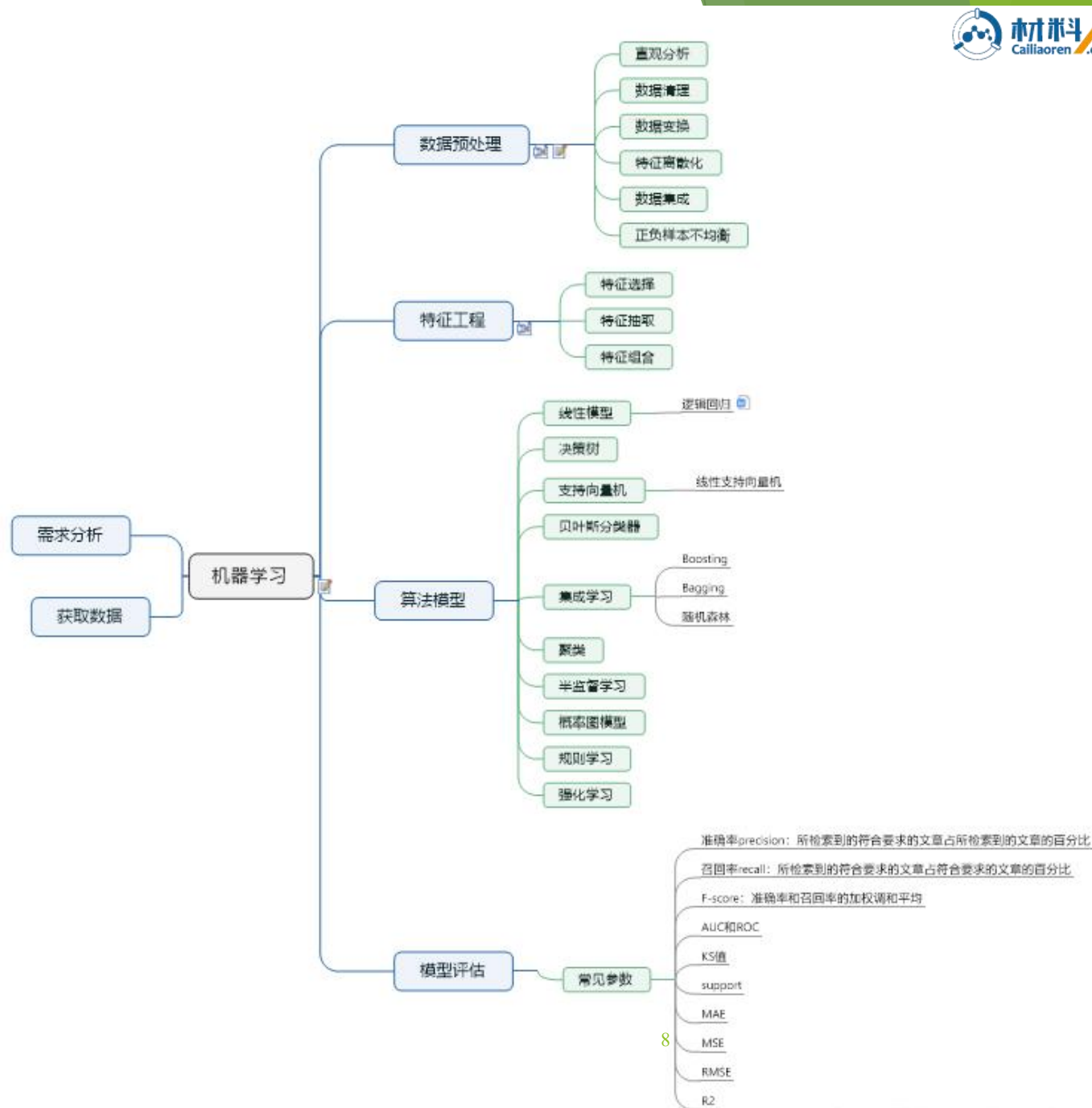
机器学习是一门研究如何通过计算的手段，利用经验数据来改善系统自身的性能的学科。

02

它通过训练和优化算法，使得系统能够自动地从数据中学习并提取有用的信息，从而进行预测或决策。

03

机器学习的目标是让计算机系统能够像人一样具有学习和进步的能力。



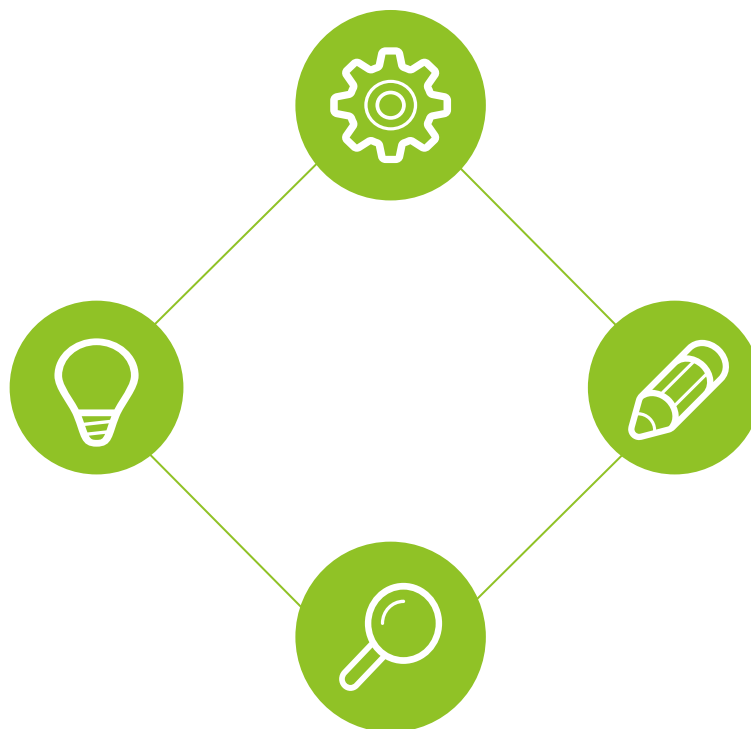
机器学习的基本原理

利用经验数据

机器学习算法需要大量的经验数据进行训练和学习，数据的质量和数量对算法的性能有着至关重要的影响。

特征提取与选择

通过对原始数据进行特征提取和选择，提取出对学习任务有帮助的特征，以提高模型的性能。



优化算法

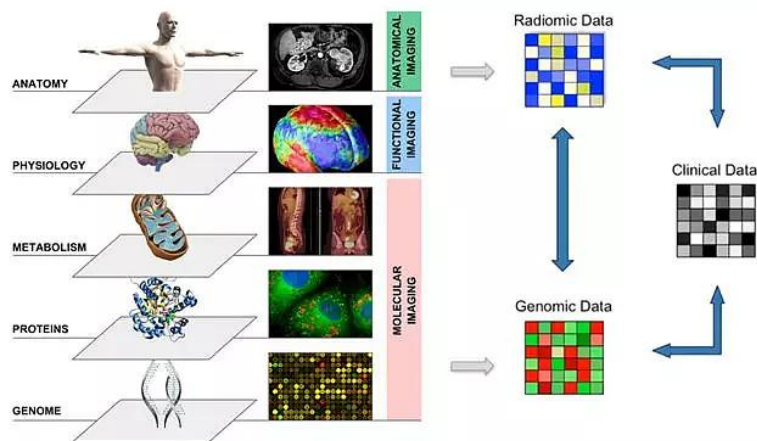
通过选择合适的优化算法，不断调整模型的参数和结构，以提高模型的预测精度和泛化能力。

模型评估与选择

通过合适的评估指标和方法，对训练得到的多个模型进行评估和比较，选择出性能最优的模型进行应用。

机器学习的应用领域

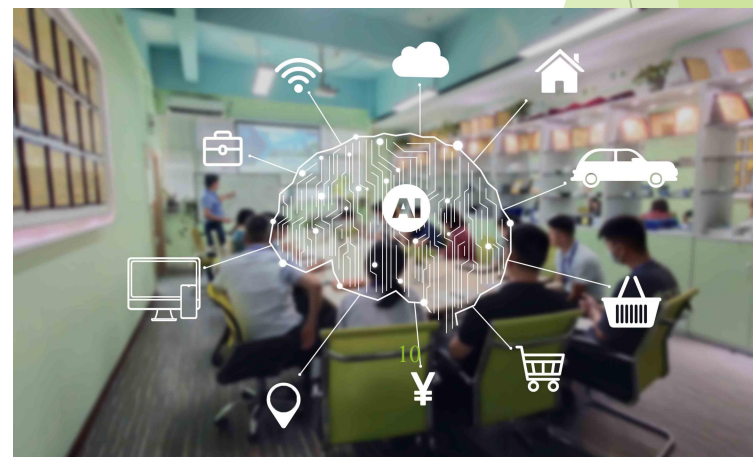
机器学习广泛应用于图像识别、语音识别、自然语言处理等领域。



随着技术的不断发展，机器学习的应用领域还将不断扩大。



在金融、医疗、教育等行业，机器学习也发挥着越来越重要的作用。



机器学习简要发展历程与标志性事件

- 1936年Alan Turing提出Turing机：可以进行逻辑和数学计算的机器，现代计算机的基础；
- 1943年McCulloch-Pitts神经元模型；
- 1945年Von Neumann提出现代计算构架：CPU，内存，IO件；
- 1950年Alan Turing提出Turing测试；
- 1952年Arthur Samuel开发了第一个自学习程序——国际象棋；
- 1950s Frank Rosenblatt开发了感知模型；
- 1957年‘人工智能’在Dartmouth会议上被首次提出；
- 1960s-1970s：AI寒冬；

机器学习简要发展历程与标志性事件

- 1980s: Rumelhart和Hinton等提出反向传播算法;
- 1990s: 决策树、随机森林、Boosting算法、支持向量机器等;
- 1997年: IBM的Blue Gene国际象棋击败人类世界冠军;
- 2006年: Hinton提出深度学习;
- 2012年: Alex Krizhevsky, Ilya Sutskever, 和 Geoffrey Hinton在ImageNet竞赛取得突破;
- 2015年: 微软的ResNet在ImageNet竞赛中首次超越人类
- 2016年: 谷歌DeepMind开发的AlphaGo击败围棋世界冠军李世石;
- 2017年: 谷歌大脑提出基于‘自我注意’ self-attention的Transformer模型;

机器学习的类别

非监督学习

让模型从输入数据中自行发现一些规律和结构，而不需要明确的标签或结果。

强化学习

让模型在与环境的交互中学习，通过尝试不同的行为来最大化某种累积奖励信号。

半监督学习

结合监督学习与非监督学习，利用部分标签数据和大量无标签数据进行学习。

监督学习

通过已有的训练数据集进行训练，得到一个最优模型，再利用这个模型将所有的输入映射为相应的输出。



深度学习

通过构建深层神经网络来模拟人脑的学习过程，实现更加复杂和高级的功能。

深度学习

```
import tensorflow as tf
from tensorflow.keras import layers, models
mnist = tf.keras.datasets.mnist # 加载MNIST数据集
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

train_images, test_images = train_images / 255.0, test_images / 255.0 # 归一化图像数据

model = models.Sequential([ # 构建模型
    layers.Flatten(input_shape=(28, 28)),
    layers.Dense(128, activation= 'relu'),
    layers.Dropout(0.2),
    layers.Dense(10)
])

model.compile(optimizer='adam', # 编译模型
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy' ])

model.fit(train_images, train_labels, epochs=5) # 训练模型

test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2) # 评估模型
print('\nTest accuracy:', test_acc)
```

- `layers.Flatten(input_shape=(28, 28)),`
- # 将28x28的图像展平成784维的向量
- `layers.Dense(128, activation='relu'),`
- #第一个隐藏层，128个神经元，ReLU激活函数

$$output = f_{activation}(weights \times input + bias)$$

128维, 128x784的权重矩阵, 偏置

$$f_{activation} = \begin{cases} x, & \text{if } x > 0; \\ 0, & \text{otherwise} \end{cases}$$

- #有助于引入非线性，使得网络能够学习更复杂的特征。
- `layers.Dropout(0.2),`
- # 设置为0，防止过拟合的Dropout层，以减少模型的复杂度和提高泛化能力
- `layers.Dense(10)` # 输出层，10个神经元（对应10个数字类别）

卷积神经网络 (convolutional neural network, CNN)

```
import tensorflow as tf  
from tensorflow.keras import layers
```

接受一个28x28像素的单通道图像作为输入，使用32个3x3大小的卷积核进行卷积操作，并应用ReLU激活函数。输出将是32个特征图。

```
model = models.Sequential([  
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)), # 卷积层  
    layers.MaxPooling2D((2, 2)), # 池化层  
    layers.Conv2D(64, (3, 3), activation='relu'), # 第二个卷积层  
    layers.MaxPooling2D((2, 2)), # 第二个池化层  
    layers.Conv2D(64, (3, 3), activation='relu'), # 第三个卷积层  
    layers.Flatten(), # 展平层，将特征图转换为一维向量  
    layers.Dense(64, activation='relu'), # 全连接层  
    layers.Dense(10, activation='softmax') # 输出层，使用softmax激活函数进行多类分类  
)  
  
model.compile(optimizer='adam',  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])
```

- layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
- # 使用32个卷积核对3x3的像素进行卷积, 32*26x26
- # k卷积: $output_{ijk} = f_a(b_k + \sum_{m=0}^2 \sum_{n=0}^2 input_{i+m,j+n,k} * kernel_{m,n,k})$
- layers.MaxPooling2D((2, 2)), # 池化层, 选取2x2像素块中最大的值32*13x13
- layers.Conv2D(64, (3, 3), activation='relu'), # 第二个卷积层64*11x11
- # $output_{ijk} = f_a(b_k + \sum_l \sum_{m=0}^2 \sum_{n=0}^2 input_{i+m,j+n,k} * kernel_{m,n,k})$, 其中l为第一次卷积的l个核
- layers.MaxPooling2D((2, 2)), # 第二个池化层
- layers.Conv2D(64, (3, 3), activation='relu'), # 第三个卷积层64*9x9
- layers.Flatten(), # 展平层, 将特征图转换为一维向量
- layers.Dense(64, activation='relu'), # 全连接层

图神经网络（Graph Neural Network, GNN）

```
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.layers import Input, Dense, GraphConv, GlobalMaxPooling1D
```

假设我们有一个图，图的节点特征和边的索引

```
num_nodes = 10
```

```
num_node_features = 4
```

```
num_classes = 3
```

节点特征

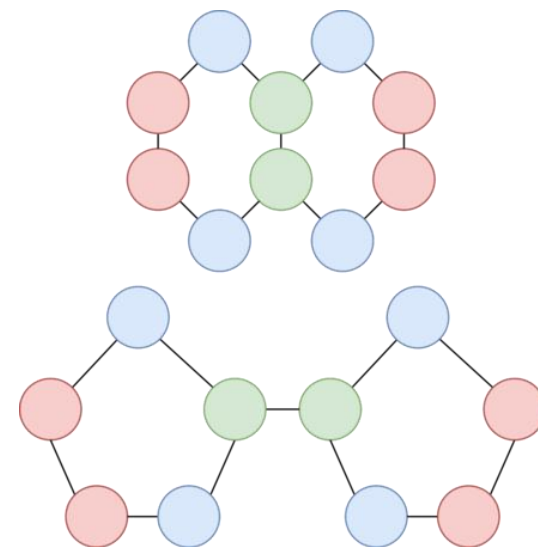
类似原子特征

```
node_features = Input(shape=(num_node_features,), dtype=tf.float32, name='node_features')
```

边的索引

```
edges = Input(shape=(2,), dtype=tf.int32, name='edges')
```

类似化学键特征



图神经网络 (Graph Neural Network, GNN)

图卷积层

```
graph_conv = GraphConv(16, activation='relu')([node_features,  
edges])
```

```
graph_conv = GraphConv(16, activation='relu')(graph_conv)
```

全局最大池化层

```
global_max_pooling = GlobalMaxPooling1D()(graph_conv)
```

分类层

```
classification = Dense(num_classes,  
activation='softmax')(global_max_pooling)
```

构建模型

```
model = keras.Model(inputs=[node_features, edges],  
outputs=classification)
```

```
model.compile(optimizer='adam',  
loss='categorical_crossentropy', metrics=['accuracy'])
```

假设我们有一些训练数据

```
train_node_features = tf.random.normal((num_nodes,  
num_node_features))
```

```
train_edges = tf.constant([[0, 1], [1, 2], [2, 3], [3, 4]])
```

```
train_labels = tf.random.uniform((num_nodes,  
num_classes), maxval=3, dtype=tf.int32)
```

训练模型

```
model.fit([train_node_features, train_edges], train_labels,  
epochs=10)
```

机器学习三大要素

1. 数据库:

足够多的 (scaling-law)、高质量的原始数据...

一般来说, 更多的数据可以提高模型的泛化能力。随着数据量的增加, 模型性能通常会提高, 但这种提升最终会趋于饱和。

2. 特征工程:

数据清洗、有效信息的选择、提取、转换、编码、PCA...

3. 机器学习算法:

RF、KRR, SVM, 深度学习, CNN, GNN

超参数hyperparameters优化

集成学习 (Ensemble Learning)

...

数据介绍

在数据集中一般：

- 一行数据我们称为一个样本，**sample**
- 一列数据我们成为一个特征，**feature, descriptor**
- 有些数据有**目标值（标签值，label, target）**，有些数据没有目标值（如上表中，电影类型就是这个数据集的目标值）

数据类型构成：

- 数据类型一：特征值+目标值（目标值是连续的和离散的）
- 数据类型二：只有特征值，没有目标值

数据收集与预处理

- u 实验数据：人工阅读、爬虫抓取、大语言模型提取
- u 计算数据：程序库生成、DFT计算
- u 原始数据预处理：数据清洗、数据变换、数据归一化等，可提高数据的质量和可用性。

数据预处理：清洗、转换、标准化

01

数据清洗

处理缺失值、异常值、重复值，保证数据质量。

02

数据转换

将数据转换成适合机器学习模型的格式，如独热编码、标签编码等。

03

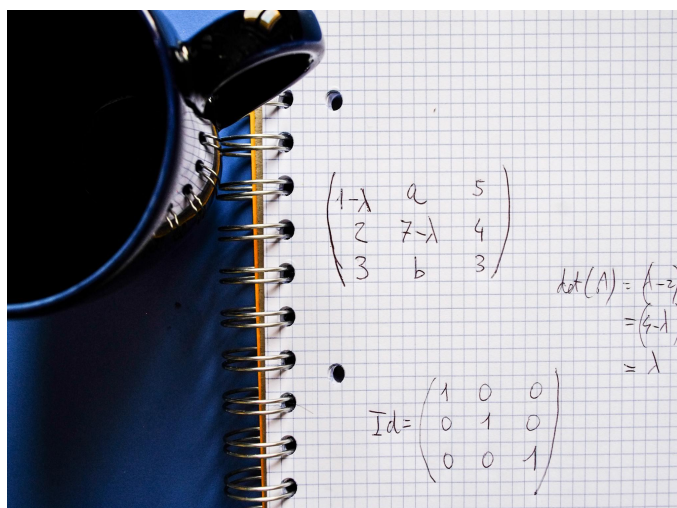
数据标准化

通过缩放数据，使其符合标准正态分布，提高模型的训练效率和准确性。



特征工程：特征选择、特征提取、降维

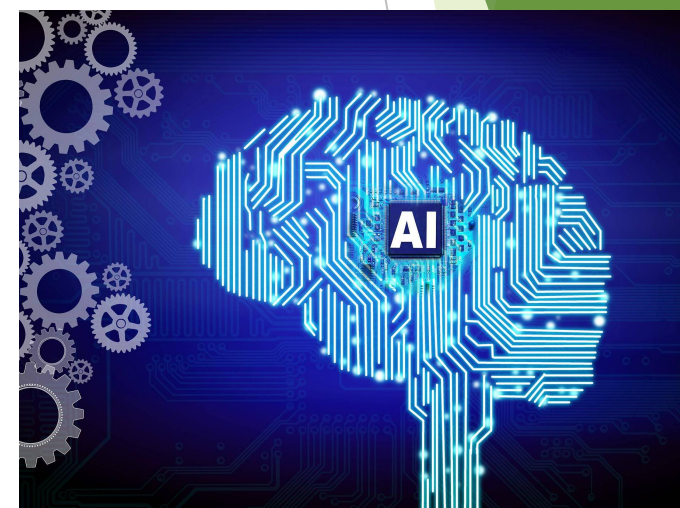
特征工程是使用**专业背景知识**和技巧处理数据，使得特征能在机器学习算法上发挥更好的作用的过程。



特征选择



特征提取



预处理

特征提取与预处理

特征提取：自动提取特征并转化为数字特征

如：将任意数据（如文本或图像）转换为可用于机器学习的

My father was a self-taught mandolin player. He was one of the best string instrument players in our town. He could not read music, but if he heard a tune a few times, he could play it. When he was younger, he was a member of a small country music band. They would play at local dances and on a few occasions would play for the local radio station. He often told us how he had auditioned and earned a position in a band that featured Patsy Cline as their lead singer. He told the family that after he was hired he never went back. Dad was a very religious man. He stated that there was a lot of drinking and cursing the day of his audition and he did not want to be around that type of environment.

```
array([[ 0.04950738,  0.19802951,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.09901475,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.09901475,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.09901475,  0.04950738,  0.04950738,  0.6435999,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.09901475,  0.04950738,  0.04950738,  0.04950738,  0.09901475,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.24753689,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.14852213,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.04950738,  0.19802951,
         0.19802951,  0.04950738,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.09901475,  0.04950738,  0.04950738,  0.04950738,
         0.04950738,  0.04950738,  0.04950738,  0.34655164,  0.04950738,
         0.04950738,  0.09901475,  0.04950738]])
```

特征预处理：将特征数据转换成更加适合算法模型的特征数据过程。

如主成分分析（PCA）等，减少特征的维度，同时保留数据中的主要信息，可提高模型的泛化能力。

特征1 特征2 特征3 特征4

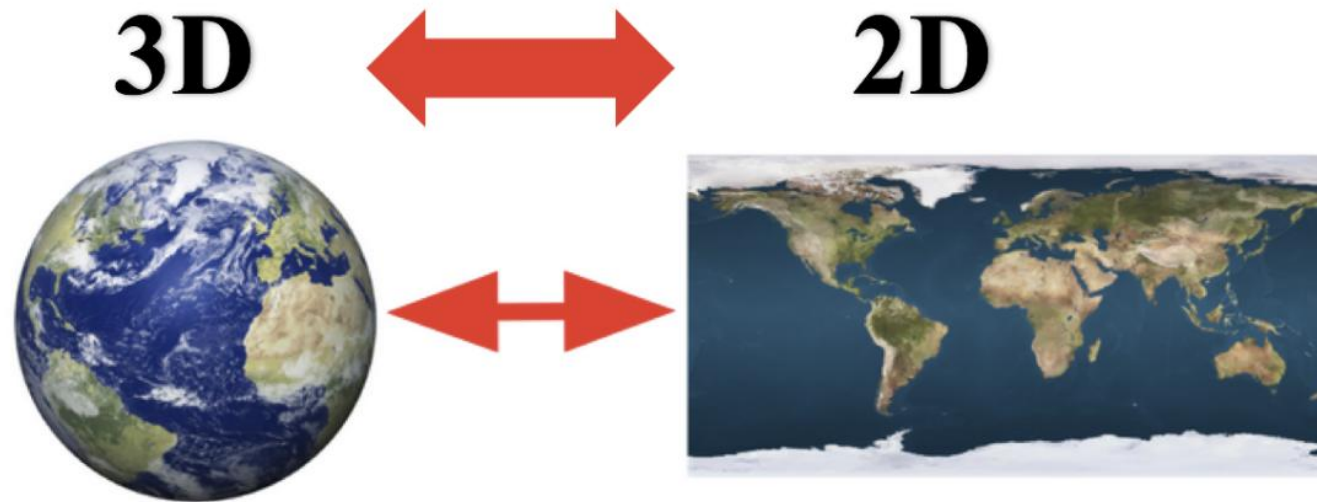
90	2	10	40
60	4	15	45
75	3	13	46



特征1 特征2 特征3 特征4

1.	0.	0.	0.
0.	1.	1.	0.83
0.5	0.5	0.6	1.

- 特征降维：指在某些限定条件下，降低随机变量(特征)个数，得到一组“不相关”主变量的过程:PCA



特征（描述符）选择

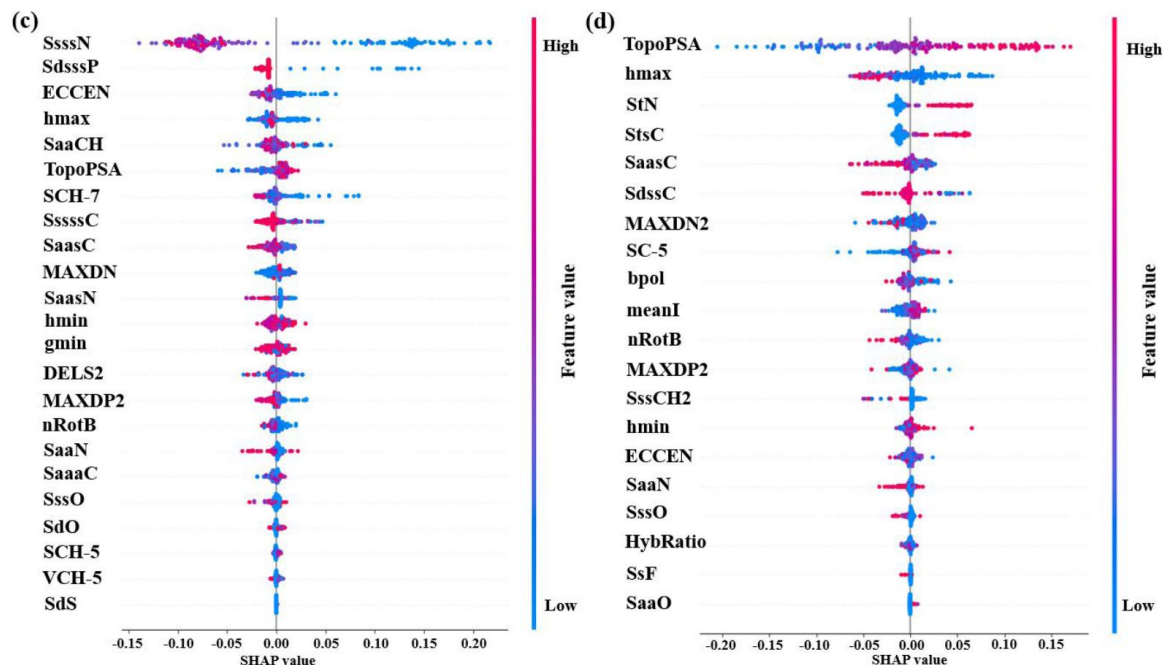
根据专业知识理解和数据分析，选择对目标变量有影响的特征：

1. 直接相关的特征

2. 间接相关的特征

阅读文献！

3. 不进行后处理的情况下，不要加入没有影响的因素。



特征（描述符）选择规则

01

相关性分析

选择与任务高度相关的特征，
去除无关或弱相关特征。

02

冗余性分析

避免选择高度相关的特征，
以减少冗余信息。

03

稳定性

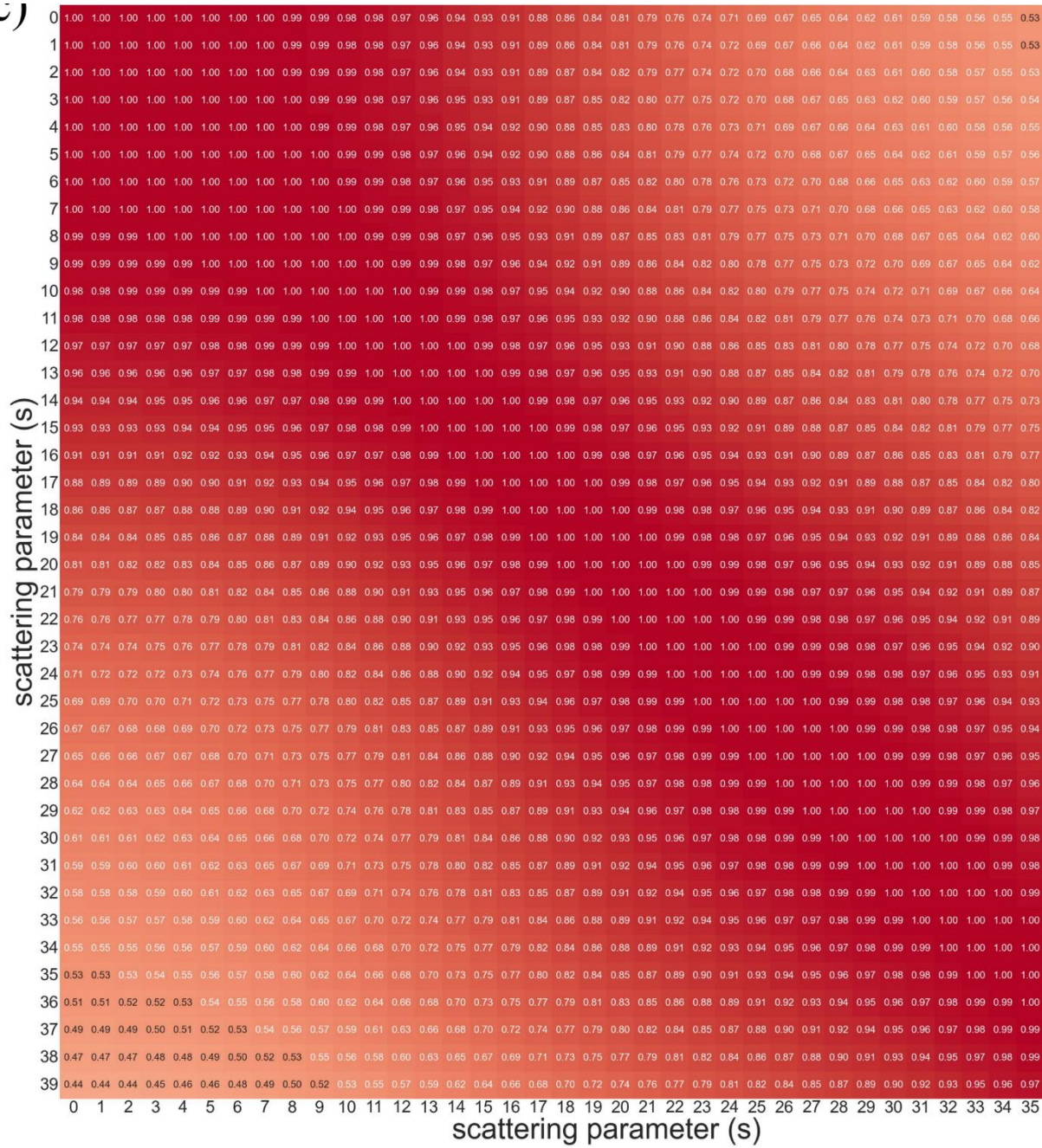
选择在不同场景下表现稳定
的特征。

04

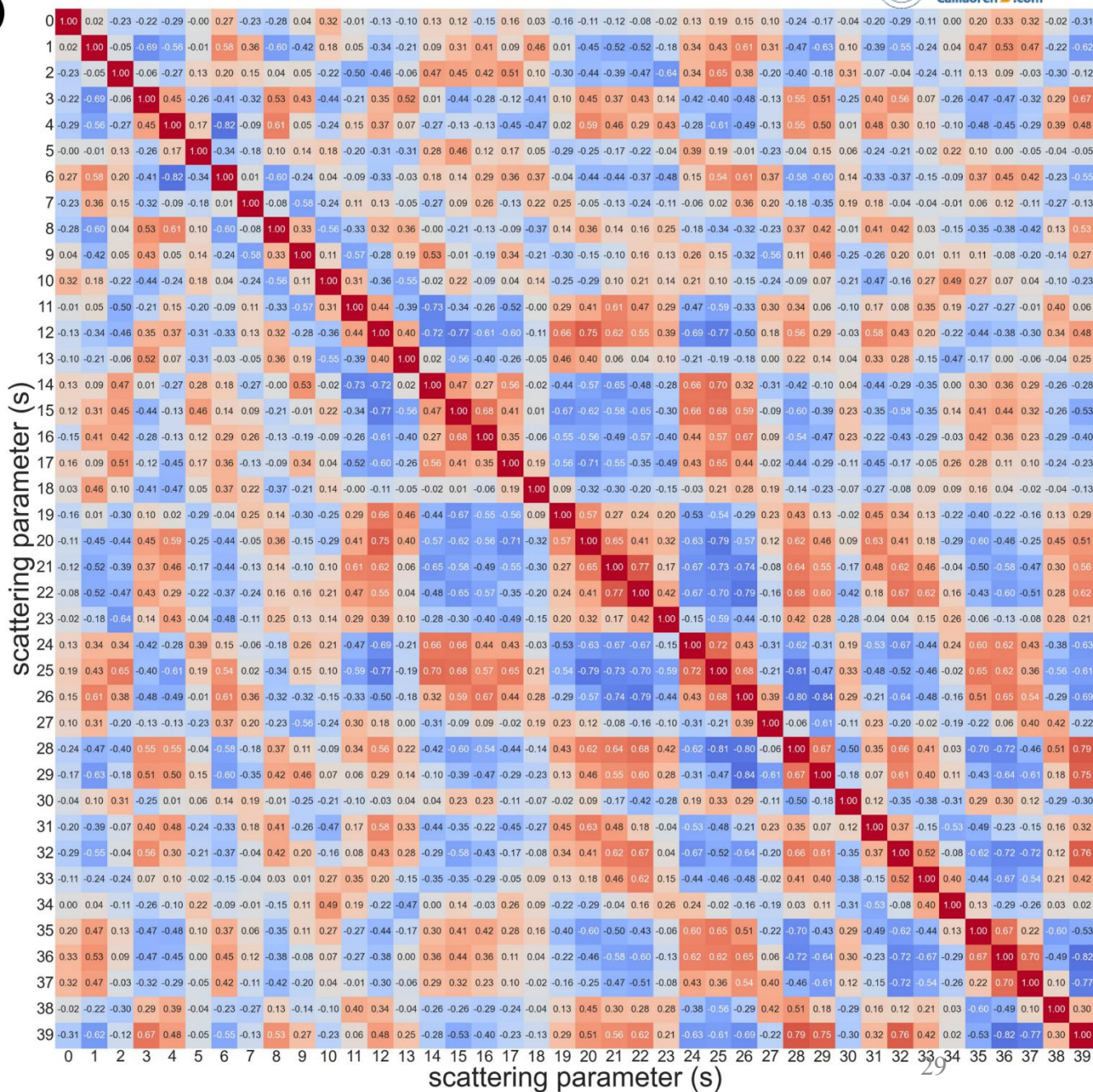
可解释性

优先选择具有明确业务含义、
易于解释的特征。

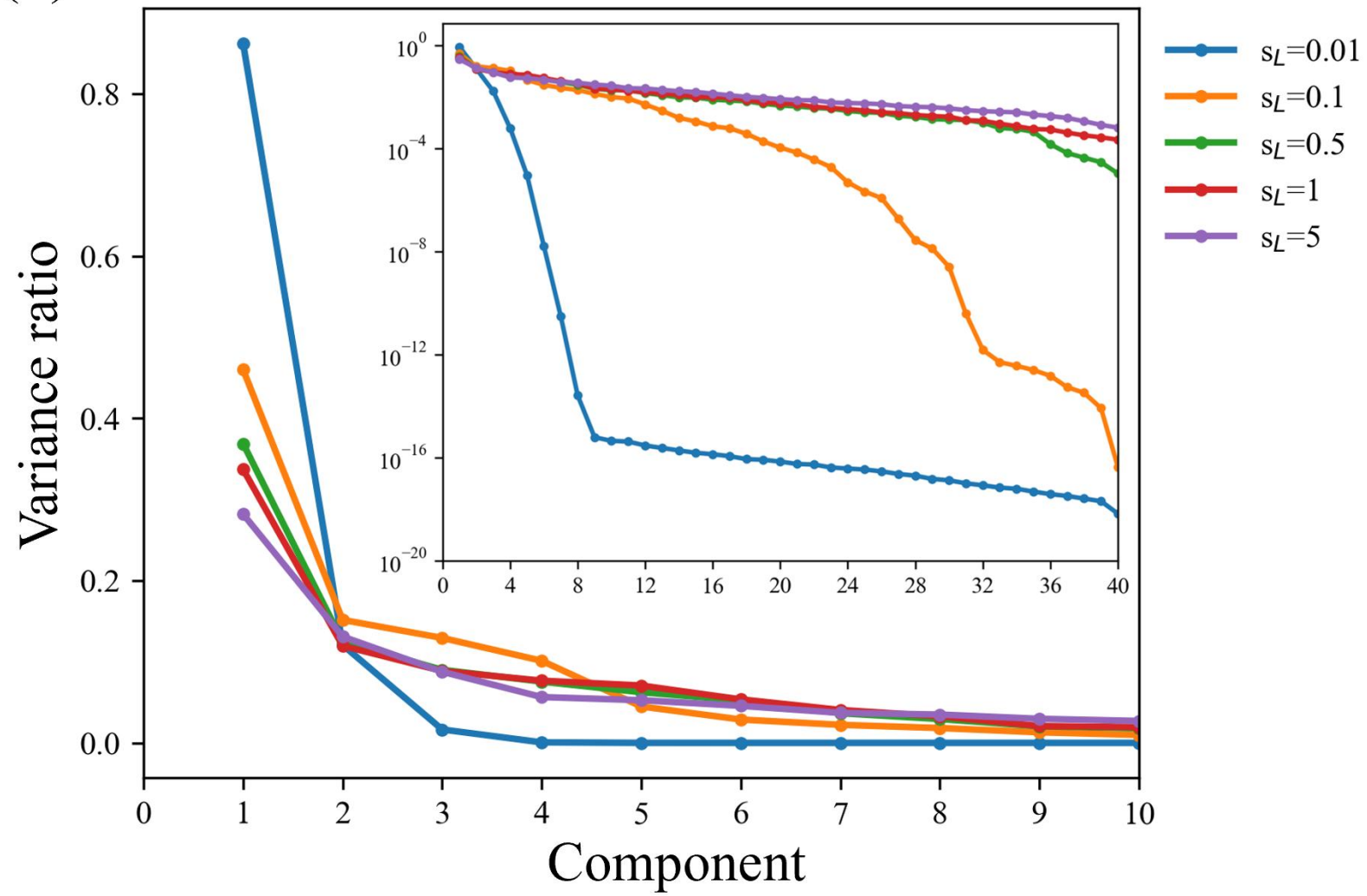
scattering parameter (s)



scattering parameter (s)



(c)



有机半导体材料描述符的产生

RDKIT:

```
from rdkit.Chem.Descriptors import CalcMolDescriptors  
from rdkit.ML.Descriptors import MoleculeDescriptors
```

```
from rdkit.Chem.Descriptors import CalcMolDescriptors  
smi='CCCC=O'  
mol= Chem.MolFromSmiles(smi)  
mh = Chem.AddHs(mol)  
status=AllChem.EmbedMolecule(mh)  
#status=AllChem.EmbedMolecule(mh,randomSeed=0xf00d)  
#status=AllChem.EmbedMolecule(mh,useRandomCoords=True)  
#status=AllChem.EmbedMolecule(mh,maxAttempts=5000)  
status=Chem.rdForceFieldHelpers.MMFFOptimizeMolecule(mh)  
desrdkit=CalcMolDescriptors(mh)  
xtmp=list(desrdkit.values())
```

```
for i,j in desrdkit.items():
    print(i,j)
```

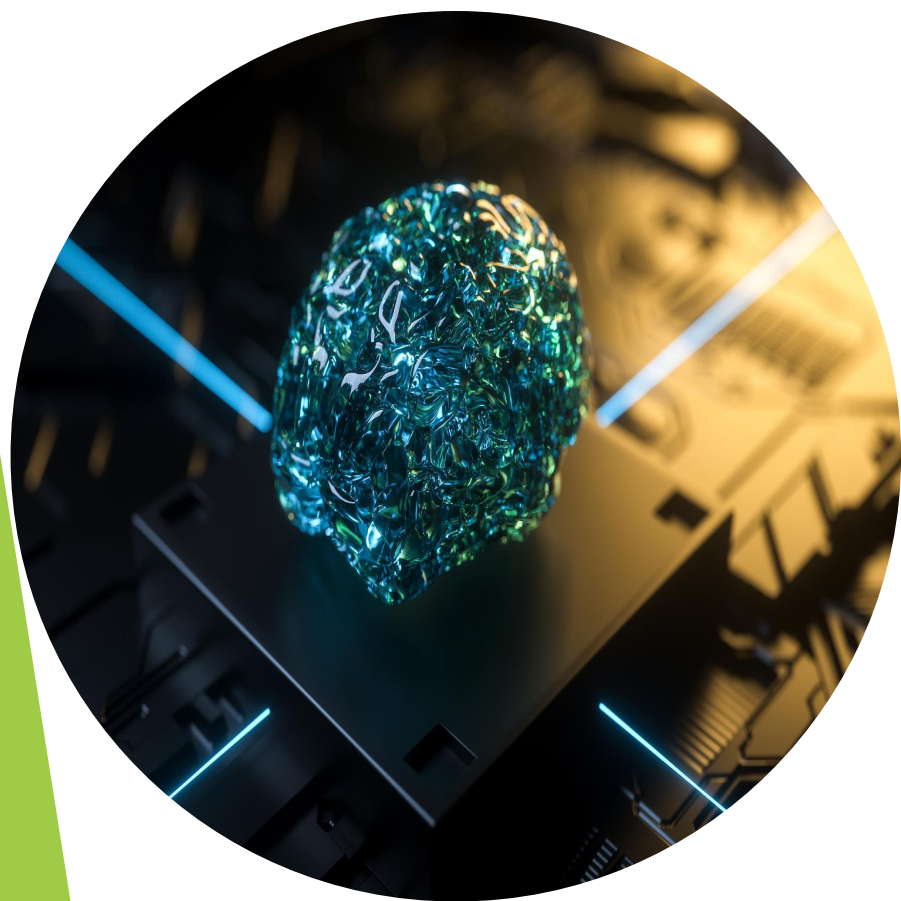
```
HallKierAlpha -0.33
Ipc 260.6586158723568
Kappa1 0.46185675190856235
Kappa2 3.67
Kappa3 3.67
LabuteASA 43.235363472405666
PEOE_VSA1 4.794537184071822
PEOE_VSA10 7.632970718501461
PEOE_VSA11 0.0
PEOE_VSA12 0.0
PEOE_VSA13 0.0
PEOE_VSA14 0.0
PEOE_VSA2 0.0
...
PEOE_VSA9 0.0
SMR_VSA1 15.760605633433581
SMR_VSA10 6.262212162331241
SMR_VSA2 0.0
.....
SMR_VSA9 0.0
SlogP_VSA1 0.0
SlogP_VSA10 0.0
SlogP_VSA11 0.0
SlogP_VSA12 0.0
SlogP_VSA2 6.262212162331241
...
SlogP_VSA9 0.0
TPSA 17.07
```

```
MaxAbsEStateIndex 10.360486111111111
MaxEStateIndex 10.360486111111111
MinAbsEStateIndex 1.8809027777777777
MinEStateIndex -3.3483796296296298
qed 0.4443450475833798
MolWt 72.106999999999999
HeavyAtomMolWt 64.042999999999999
ExactMolWt 72.057514876
NumValenceElectrons 30
NumRadicalElectrons 0
MaxPartialCharge 0.11947978113414225
MinPartialCharge -0.30340309725779463
MaxAbsPartialCharge 0.30340309725779463
MinAbsPartialCharge 0.11947978113414225
FpDensityMorgan1 2.6
FpDensityMorgan2 3.4
FpDensityMorgan3 3.6
BCUT2D_MWHI -4.56376084373792e-16
BCUT2D_MWLOW -0.24754268494773624
BCUT2D_CHGHI -0.19706563501170102
BCUT2D_CHGLO -2.4959170851077577
BCUT2D_LOGPHI -0.10280093566504482
BCUT2D_LOGPLOW -2.404942975642157
BCUT2D_MRHI -0.0006091488779740282
BCUT2D_MRLOW -1.0971836824094108
AvgIpc 1.8356240554391323
BalabanJ 5.158825498434272
BertzCT 214.6741413661228
Chi0 11.077350269189626
Chi0n 10.408248290463863
Chi0v 2.408248290463863
Chi1 5.443375672974064
Chi1n 4.954124145231932
Chi1v 0.9541241452319316
Chi2n 0.3520620726159658
Chi2v 0.3520620726159658
Chi3n 0.11353103630798289
Chi3v 0.11353103630798289
Chi4n 0.025515518153991442
Chi4v 0.025515518153991442
```

```
EState_VSA1 25.85995408228568
EState_VSA10 4.794537184071822
EState_VSA11 0.0
EState_VSA2 0.0
...
EState_VSA9 10.96606844936176
VSA_EState1 53.87726851851852
VSA_EState10 0.0
VSA_EState2 10.360486111111111
...
VSA_EState9 0.0
FractionCSP3 0.75
HeavyAtomCount 5
NHOHCount 0
NOCCount 1
NumAliphaticCarbocycles 0
NumAliphaticHeterocycles 0
NumAliphaticRings 0
NumAromaticCarbocycles 0
NumAromaticHeterocycles 0
NumAromaticRings 0
NumHAcceptors 1
NumHDonors 0
NumHeteroatoms 1
NumRotatableBonds 3
NumSaturatedCarbocycles 0
NumSaturatedHeterocycles 0
NumSaturatedRings 0
RingCount 0
MolLogP 0.9854
MolMR 20.9719999999999994
```

```
fr_Al_COO 0
fr_Al_OH 0
fr_Al_OH_noTert 0
fr_ArN 0
fr_Ar_COO 0
fr_Ar_N 0
fr_Ar_NH 0
fr_Ar_OH 0
fr_COO 0
fr_COO2 0
fr_C_O 1
fr_C_O_noCOO 1
fr_C_S 0
fr_HOCCN 0
.....
fr_sulfide 0
fr_sulfonamd 0
fr_sulfone 0
fr_term_acetylene 0
fr_tetrazole 0
fr_thiazole 0
fr_thiocyan 0
fr_thiophene 0
fr_unbrch_alkane 0
fr_urea 0
```

自动化选取方法



过滤法

基于统计性质进行特征选择，如方差阈值、相关系数等。计算每个特征与目标变量之间的相关性，保留相关性较强的特征。

包裹法

根据模型性能进行特征选择，如递归特征消除。通过不断尝试不同的特征组合，选择出最优的特征子集。

嵌入法

在模型训练过程中进行特征选择，如L1正则化。将特征选择过程与模型训练过程相结合，让模型自动学习到重要的特征。

from sklearn.feature_selection import RFE xxxxx

GenericUnivariateSelect	Univariate feature selector with configurable strategy.
RFE	Feature ranking with recursive feature elimination.
RFECV	Recursive feature elimination with cross-validation to select features.
SelectFdr	Filter: Select the p-values for an estimated false discovery rate.
SelectFpr	Filter: Select the pvalues below alpha based on a FPR test.
SelectFromModel	Meta-transformer for selecting features based on importance weights.
SelectFwe	Filter: Select the p-values corresponding to Family-wise error rate.
SelectKBest	Select features according to the k highest scores.
SelectPercentile	Select features according to a percentile of the highest scores.
SelectorMixin	Transformer mixin that performs feature selection given a support mask
SequentialFeatureSelector	Transformer that performs Sequential Feature Selection.
VarianceThreshold	Feature selector that removes all low-variance features.
chi2	Compute chi-squared stats between each non-negative feature and class.
f_classif	Compute the ANOVA F-value for the provided sample.
f_regression	Univariate linear regression tests returning F-statistic and p-values.
mutual_info_classif	Estimate mutual information for a discrete target variable.
mutual_info_regression	Estimate mutual information for a continuous target variable.
r_regression	Compute Pearson's r for each features and the target.

```
from sklearn.feature_selection import GenericUnivariateSelect, f_regression
transformer = GenericUnivariateSelect(f_regression, mode='fwe')
Tx = transformer.fit_transform(X, y) #X(13952, 900)
Tx.shape #(13952, 767)
```

chi2	Compute chi-squared stats between each non-negative feature and class.
f_classif	Compute the ANOVA F-value for the provided sample.
f_regression	Univariate linear regression tests returning F-statistic and p-values.
mutual_info_classif	Estimate mutual information for a discrete target variable.
mutual_info_regression	Estimate mutual information for a continuous target variable.
r_regression	Compute Pearson's r for each features and the target.

Mode: 'percentile', 'k_best', 'fpr', 'fdr', 'fwe'

```
from sklearn.feature_selection import SelectPercentile, SelectFromModel
from sklearn.linear_model import LinearRegression

model = SelectFromModel(estimator=LinearRegression(), threshold=1e-5)
selector = model.fit(X, y)
Tx = selector.transform(X)

Tx.shape #(13952, 767)
```

```
from sklearn.feature_selection import GenericUnivariateSelect, f_regression
transformer = GenericUnivariateSelect(f_regression, mode='fwe')
Tx = transformer.fit_transform(X, y) #X(13952, 900)

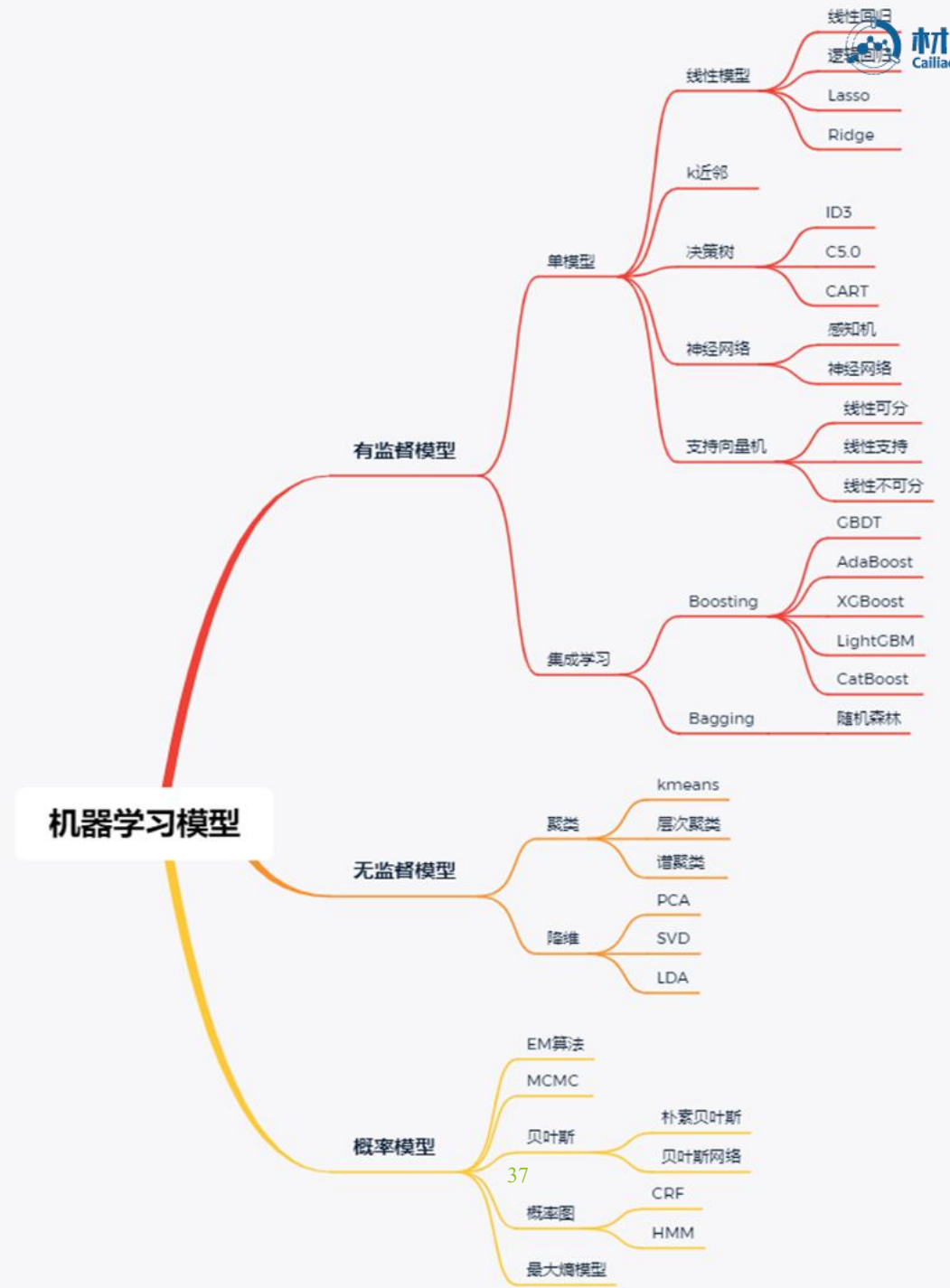
Tx.shape #(13952, 767)
```

机器学习算法

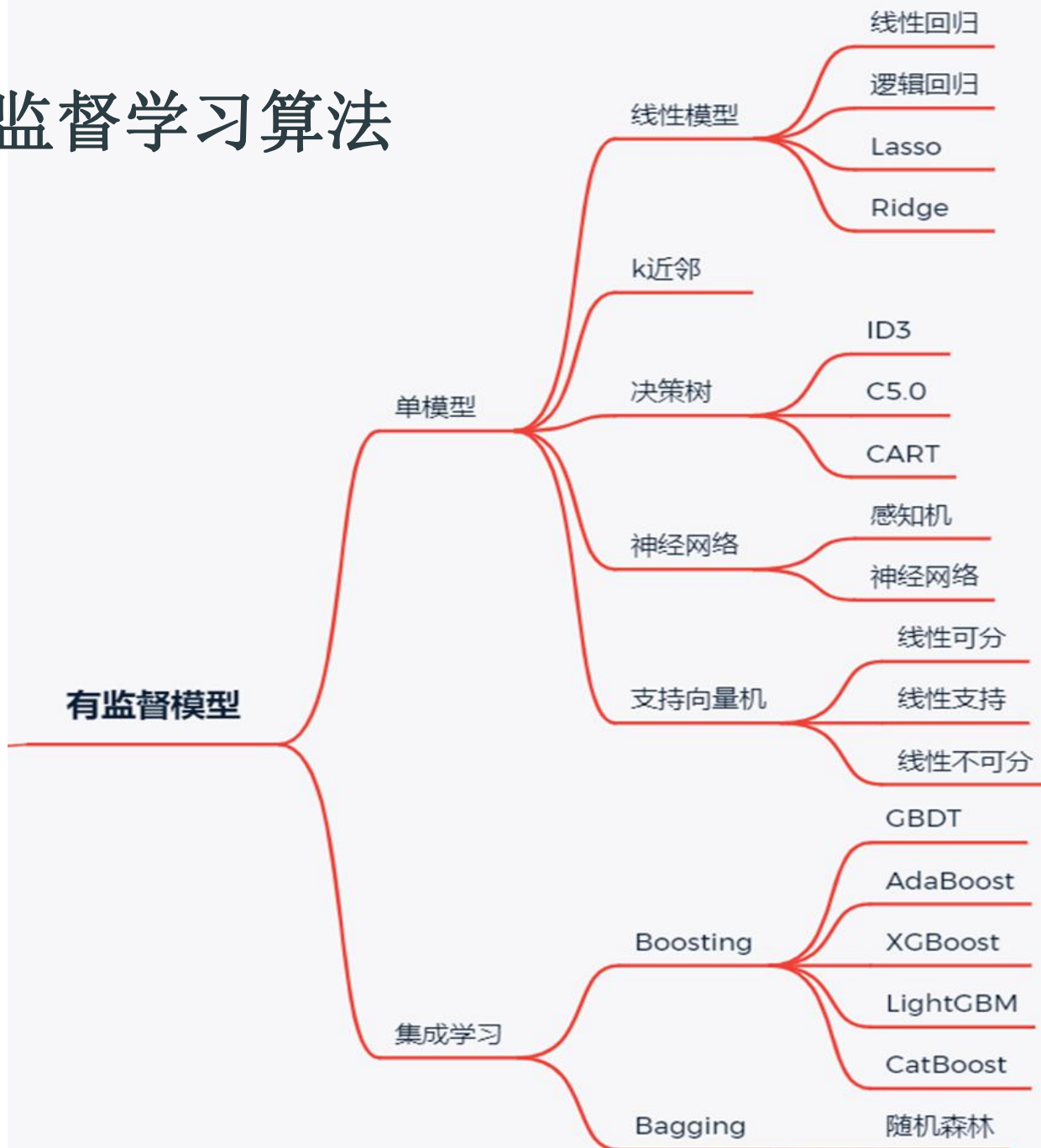
机器学习算法的基本原理是通过对数据进行学习和分析，自动地从数据中发现隐藏的模式和关系。

这个过程可以被概括为以下三个步骤：

- 训练模型
- 验证模型
- 应用模型



监督学习算法



线性回归

通过最小化预测值与真实值之间的误差来拟合数据，常用于预测连续型数值。

逻辑回归

用于处理二分类或多分类问题，通过逻辑函数将线性回归的输出转换为概率值。

决策树

通过树状结构对数据进行分类或回归，易于理解和实现。

支持向量机

在高维空间中寻找一个超平面，使得不同类别的样本被最大化地分隔开。

无监督学习算法



K均值聚类

将数据划分为K个簇，使得同一簇内的数据相似度高，不同簇之间的相似度低。

层次聚类

通过计算不同类别数据点间的相似度来创建一棵有层次的嵌套聚类树。

主成分分析

通过正交变换将原始特征线性组合成新的特征，使得新特征具有最大的方差，以去除数据中的冗余和噪声。

强化学习算法



Q-learning

一种基于值迭代的强化学习方法，通过不断更新Q值表来找到最优策略。

策略梯度

一种基于策略迭代的强化学习方法，通过计算策略梯度来更新策略参数，以实现最大化累积奖励的目标。

深度强化学习

结合深度学习和强化学习的优势，能够处理高维状态空间和动作空间的问题，具有更强的表征能力和泛化能力。

增强采样是一种重要的技术，用于克服传统模拟方法在空间和时间尺度上的限制，**特别是对于复杂体系中的稀有事件和慢动态过程**。以下是一些常见的增强采样方法：

伞形采样 (Umbrella Sampling, US) :

通过在模拟过程中施加一个或多个“伞形”势能，降低能量势垒，从而增强对某些特定区域的采样。

元动力学 (Metadynamics) :

通过在模拟过程中逐渐积累历史信息，形成一个偏置势能，引导系统探索低概率区域。

高斯加速分子动力学 (Gaussian Accelerated Molecular Dynamics, GaMD) :

利用高斯势来增强采样，特别是在蛋白质折叠等生物体系中。

过渡路径采样 (Transition Path Sampling, TPS) :

专注于采样反应路径，通过优化反应坐标来研究复杂体系的动力学过程。

加权系综采样 (Weighted Ensemble Sampling, WES) :

通过多个独立的模拟轨迹，使用不同的概率分布进行采样，然后合并结果以提高效率。

自适应偏置力 (Adaptive Biasing Force, ABF) :

根据系统的动态响应，自适应地调整偏置力，以增强对特定区域的采样。

基于集合变量的增强采样:

选择能够描述慢自由度的集合变量 (Collective Variable, CV) 对复杂体系进行增强采样。

模型评估: 分类模型

真实情况	预测情况	
	正例	反例
正例	TP (真正例)	FN (假反例)
反例	FP (假正例)	TN (真反例)

u 按照数据集的目标值不同, 可以把模型评估分为分类模型评估和回归模型评估。

u 分类模型

u 准确率: 正确预测的数量除以总预测数量。

u 精确度: 模型正确预测为正例的样本中, 实际为正例的比例

u 召回率: 召回率是指实际为正例的样本中, 模型正确预测为正例的比例

u F1分数: 精确度和召回率的调和平均数, 是一个综合指标

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

[🏠](#) > [API Reference](#) > [sklearn.metrics](#) > [accuracy_score](#)

```
from sklearn.metrics import xxxxxx
```

accuracy_score

```
sklearn.metrics.accuracy_score(y_true, y_pred, *, normalize=True, sample_weight=None)
```

Accuracy classification score.

[\[source\]](#)

In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must *exactly* match the corresponding set of labels in y_true.

Read more in the [User Guide](#).

[🏠](#) > [API Reference](#) > [sklearn.metrics](#) > [precision_score](#)

precision_score

```
sklearn.metrics.precision_score(y_true, y_pred, *, labels=None, pos_label=1,  
average='binary', sample_weight=None, zero_division='warn')
```

Compute the precision.

- check_scoring
- get_scorer
- get_scorer_names
- make_scorer
- accuracy_score
- auc
- average_precision_score
- balanced_accuracy_score
- brier_score_loss
- class_likelihood_ratios
- classification_report
- cohen_kappa_score
- confusion_matrix
- d2_log_loss_score
- dcg_score
- det_curve

模型评估：回归模型

回归

- 均方误差MSE：预测值与实际值之差的平方的平均值
- 均方根误差RMSE：MSE的平方根，同观测数据具有相同的单位
- 平均绝对误差MAE：预测值与实际值之差的绝对值的平均值
- 平均绝对百分比误差MAPE：预测误差与实际值的比值的绝对值的平均百分比
- 决定系数 R^2 ：衡量模型对数据集的拟合优度
- 皮尔逊相关系数 r ：度量两个变量之间线性关系强度和方向的统计量。

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

$$\text{RMSE} = \sqrt{\text{MSE}}$$

$$\text{MAPE} = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{Y_i - \hat{Y}_i}{Y_i} \right|$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i|$$

$$R^2 = 1 - \frac{\text{SS}_{\text{res}}}{\text{SS}_{\text{tot}}}$$

$$r_{xy} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$

sklearn.metrics

```
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

```
r2= r2_score(y_test, y_test_predicted)
```

```
mae=mean_absolute_error(y_test, y_test_predicted)
```

```
mse=mean_squared_error(y_test, y_test_predicted,squared=True)
```

```
rmse=mean_squared_error(y_test, y_test_predicted,squared=False)
```

模型验证:

u 随机分组:

机器学习一般的数据一般分为训练集 (train_set), 评估集 (valid_set), 测试集 (test_set)。两者 (三者) 比例相加为100%

训练数据和验证数据用于模型训练与产生优化, 测试数据用于评估模型是否有效

u 交叉验证:

K-fold cross-validation

u 留一法 leave one out

请下载并安装以下python库：

1. 安装anaconda: <https://www.anaconda.com/>
2. `conda create --name myenvname python`
3. `conda activate myenvname`
4. `pip install scikit-learn` （也可`conda install scikit-learn`）

numpy科学计算库、pandas数据分析库、

matplotlib可视化库、shap机器学习模型解释库

openbabel、rdkit化学信息计算

deepchem用于分子数据集的机器学习和深度学习库

torch神经网络框架、torch-geometric图神经网络库

tensorflow神经网络框架、keras神经网络框架、dgl图神经网络库

如果安装太慢，更换源

pip source:

- u `python -m pip install --upgrade pip`
- u `pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple`

Conda source:

- u `conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/free/`
- u `conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkg/main/`
- u `conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge`
- u `conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/msys2/`

不停学习与尝试