## 机器学习的类型

**监督式**

监督式学习算法基于一组示例进行预测。 例如，历史股票价格可用于做出有关未来价格推测。 用于定型的每个示例都标有相关的值 — 本例中是股票价格。 监督式学习算法查找这些值标签中的模式。它可以使用任何可能相关的信息，包括星期几，季节，公司的财务数据，行业类型，存在的破坏性地缘政治事件，并且每种算法查找不同类型的模式。 算法找到可能的最佳模式之后，使用该模式对未标记的测试数据（明天的价格）进行预测。

监督式学习是一种常用且有用的机器学习类型。 Azure 机器学习工作室中的所有模块都采用监督式学习算法，只有一个例外。 Azure 机器学习工作室中有几种特定类型的监督式学习：分类、回归和异常情况检测。

* **分类**。 在数据用于预测类别时，监督式学习也称为分类。 将图像分配为“猫”或“狗”的图片时，就是这种情况。 如果只有两个选项，这就称为**双类**或**二元分类**。 如果有多个类别，如预测 NCAA 疯狂三月锦标赛的冠军时，则称为**多类分类**。
* **回归**。 在预测一个值时，如股票价格，监督式学习称为回归。
* **异常情况检测**。 有时，目标是识别仅仅不寻常的数据点。 例如，在欺诈行为检测中，任何极不寻常的信用卡消费模式都是可疑的。 欺诈行为可能的变体很多，但定型示例却很少，因此无法了解欺骗性活动的外表形式。 异常情况检测采用的方法是只需了解哪些的正常活动如下所示 （使用非欺诈性交易的历史记录） 并确定任何有很大不同。

**非监督式**

在非监督式学习中，数据点没有与其关联的标签。 相反，非监督式学习算法的目的是以某种方式组织数据或者说明其结构。 这意味着将其分组到群集或查找不同的方法来查看复杂数据以使其显示更简单或更有条理。

**强化学习**

在强化学习中，算法需选择响应每个数据点的操作。 学习算法还会在短时间后收到奖励信号，指示决策的优秀程度。 在此基础上，算法会修改其战略议获得最高奖励。 目前，Azure 机器学习工作室中暂无强化学习算法模块。 强化学习在机器人中非常常见，在此技术中一个时间点的传感器读数集是数据点，并且算法必须选择机器人的下一个动作。 它也是物联网应用程序的理想选择。

**选择算法时的注意事项**

**准确性**

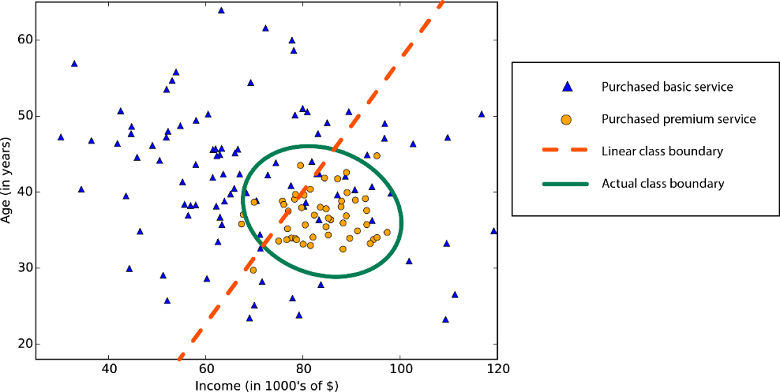
获取最准确的答案可能并不总是必要的。 有时，近似值便已足够，具体取决于想要将其用于何处。如果是这种情况，则可以通过坚持使用更多的近似值的方法大大减少处理时间。 更多近似值方法的另一个优点是，他们自然倾向于避免过度拟合。

**定型时间**

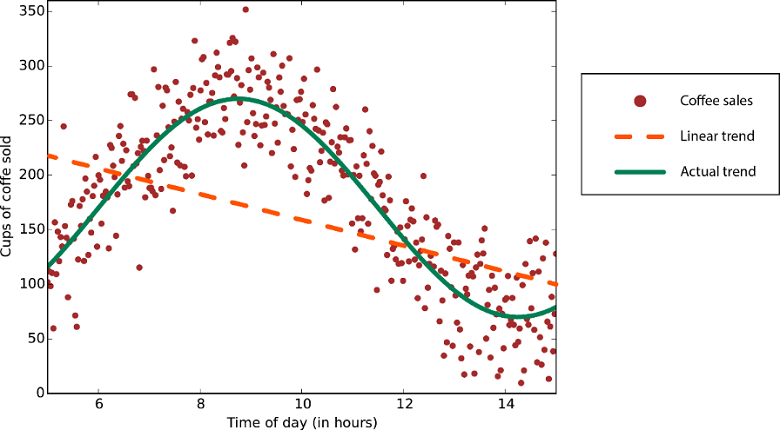
算法之间定型模型所需的分钟数或小时数差异较大。 定型时间通常与准确性密切相关 - 通常一个伴随另一个。 此外，相较于其他算法，某些算法对数据点数目更敏感。 时间有限（特别是数据集较大）时，它可以驱动算法的选择。

**线性**

许多机器学习算法都使用线性。 线性分类算法假定直线（或其更高维的模拟）可以将类分离。 其中包括逻辑回归和支持矢量机（在 Azure 机器学习工作室中实现）。 线性回归算法假定数据趋势遵循一条直线。 这些假设对于某些问题而言还不错，但在其他问题上，会使准确性降低。



***非线性类边界*** - 依赖于线性分类算法会导致较低的准确性



***非线性趋势数据*** - 使用线性回归方法会生成比必要的更大的错误

尽管其危险，线性算法作为攻击的第一线仍然非常受欢迎。 它们往往算法简单且可快速定型。

**参数数目**

参数是数据科学家在设置算法时要旋转的旋钮。 它们是影响算法行为的数字，例如错误容限、迭代次数，或算法行为方式的变体之间的选项。 算法的定型时间和准确性有时可能对获取正确设置相当敏感。 通常情况下，参数数量很大的算法需要充分试验和错误来找到好的组合。

或者，Azure 机器学习工作室中有一个[参数扫描](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio/algorithm-parameters-optimize)模块，它可自动尝试任何所选粒度上的所有参数组合。 虽然这是确保跨越参数空间的好方法，但定型模型所需的时间随参数数量呈指数增长。

优点是通常情况下，参数较多说明算法具有更大的灵活性。 这能获得较好的准确性，提供可以找到的参数设置的正确组合。

**特征数量**

对于某些类型的数据，相较于数据点的数量，特征的数量可能非常大。 这通常出现在遗传学或文本数据的情况下。 大量的特征会导致某些学习算法不可用，从而使得定型时间特别长。 支持向量机是特别适用于这种情况（参见以下内容）。

**特殊情况**

一些学习算法会对数据的结构或期望的结果做出特定假设。 如果找到符合需求的算法，它可以提供更有用的结果、更准确的预测或更快的定型时间。

表格 1机器学习算法

| **算法** | **准确性** | **定型时间** | **线性** | **Parameters** | **说明** |
| --- | --- | --- | --- | --- | --- |
| **二分类** | | | | | |
| [逻辑回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-logistic-regression) |  | ● | ● | 5 |  |
| [决策林](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-decision-forest) | ● | ○ |  | 6 |  |
| [决策森林](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-decision-jungle) | ● | ○ |  | 6 | 内存占用量小 |
| [提升决策树](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-boosted-decision-tree) | ● | ○ |  | 6 | 内存占用量大 |
| [神经网络](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-neural-network) | ● |  |  | 9 | [可以进行其他自定义](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio/azure-ml-netsharp-reference-guide) |
| [平均感知器](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-averaged-perceptron) | ○ | ○ | ● | 4 |  |
| [支持向量机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-support-vector-machine) |  | ○ | ● | 5 | 适用于大型特征集 |
| [本地深度支持向量机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-locally-deep-support-vector-machine) | ○ |  |  | 8 | 适用于大型特征集 |
| [贝叶斯点机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-bayes-point-machine) |  | ○ | ● | 3 |  |
| **多分类** | | | | | |
| [逻辑回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-logistic-regression) |  | ● | ● | 5 |  |
| [决策林](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-decision-forest) | ● | ○ |  | 6 |  |
| [决策森林](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-decision-jungle) | ● | ○ |  | 6 | 内存占用量小 |
| [神经网络](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-neural-network) | ● |  |  | 9 | [可以进行其他自定义](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio/azure-ml-netsharp-reference-guide) |
| [一对多](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/one-vs-all-multiclass) | - | - | - | - | 查看所选双类方法的属性 |
| **回归** | | | | | |
| [线性](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/linear-regression) |  | ● | ● | 4 |  |
| [贝叶斯线性](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/bayesian-linear-regression) |  | ○ | ● | 2 |  |
| [决策林](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/decision-forest-regression) | ● | ○ |  | 6 |  |
| [提升决策树](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/boosted-decision-tree-regression) | ● | ○ |  | 5 | 内存占用量大 |
| [快速林分位](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/fast-forest-quantile-regression) | ● | ○ |  | 9 | 分发而不是点预测 |
| [神经网络](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/neural-network-regression) | ● |  |  | 9 | [可以进行其他自定义](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio/azure-ml-netsharp-reference-guide) |
| [泊松](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/poisson-regression) |  |  | ● | 5 | 从技术上讲是对数线性。 用于预测计数 |
| [序号](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/ordinal-regression) |  |  |  | 0 | 用于预测排序 |
| **异常情况检测** | | | | | |
| [支持向量机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/one-class-support-vector-machine) | ○ | ○ |  | 2 | 尤其适用于大型特征集 |
| [PCA-based anomaly detection](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/pca-based-anomaly-detection)（基于 PCA 的异常情况检测） |  | ○ | ● | 3 |  |
| [K-means](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/k-means-clustering) |  | ○ | ● | 4 | 聚类算法 |

算法属性：

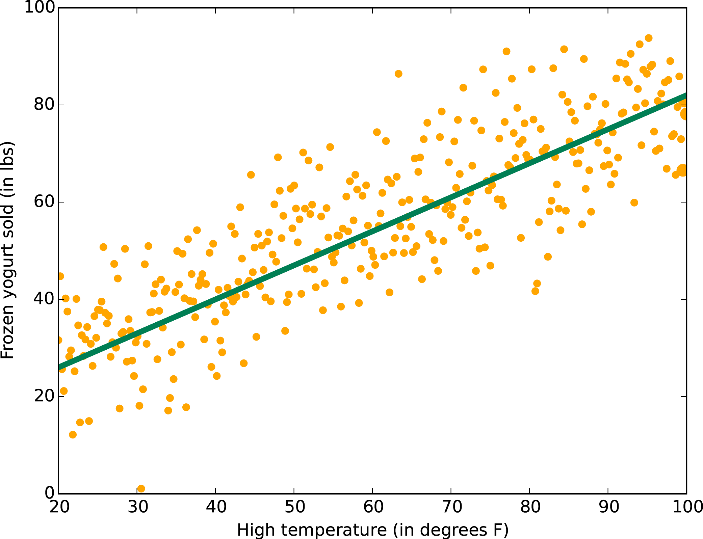
● - 表示出色的准确性、快速的定型时间以及对线性的使用

○ - 表示良好的准确性和适度的定型时间

**算法说明**

**线性回归**

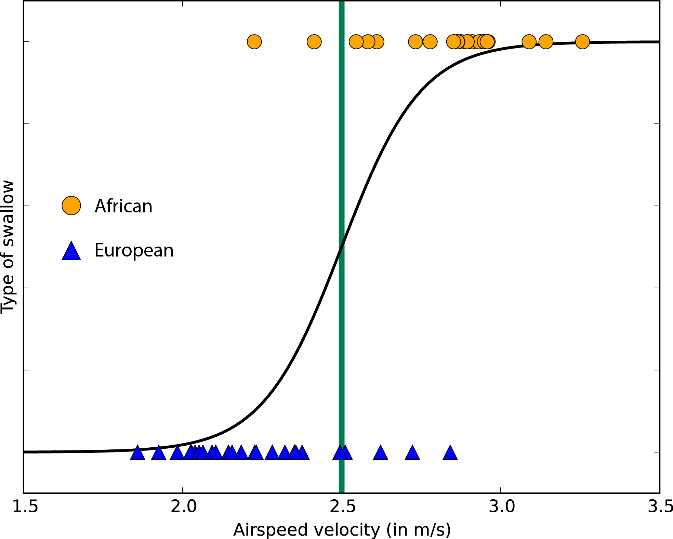
如之前所述，[线性回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/linear-regression)可调整线（或平面、超平面）以适应数据集。 它是主力，简单且快速，但对于某些问题而言可能过于简单。



***线性趋势数据***

**逻辑回归**

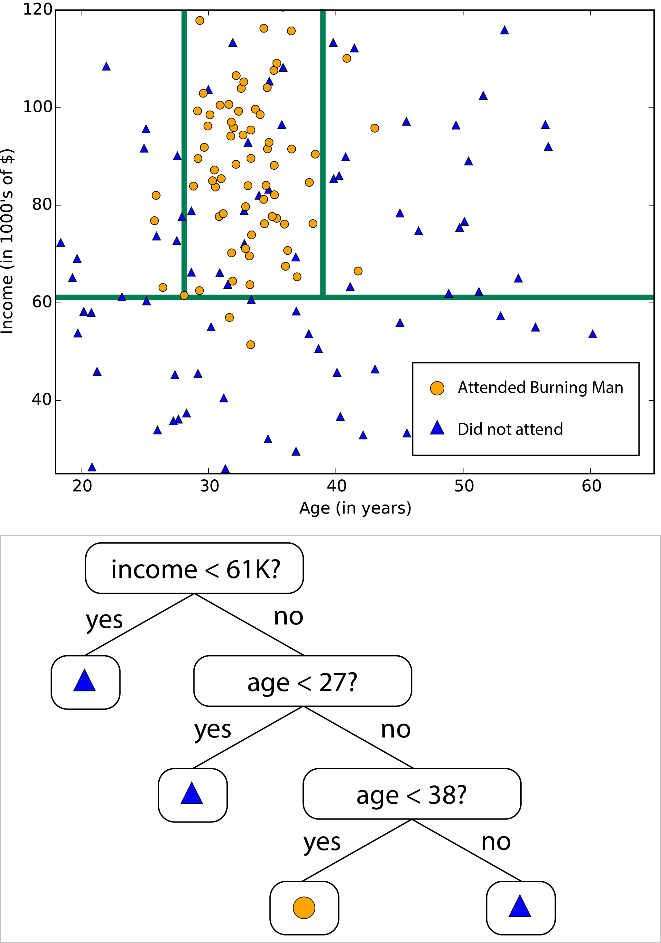
尽管它名称中包含回归，但逻辑回归是用于实际的强大的工具[双类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-logistic-regression)并[多类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-logistic-regression)分类。 它快速而简单。 事实上，它使用“S”形曲线而不是直线，是将数据分成组的理想工具。 逻辑回归提供线性类边界，因此，使用时，请确保线性近似值是可以接受的值。



***逻辑回归到仅具有一个特征的双类数据*** - 类边界是逻辑曲线与两个类都接近的点

**树、林和森林**

决策林（[回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/decision-forest-regression)[双类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-decision-forest)和[多类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-decision-forest)）、决策森林（[双类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-decision-jungle)和[多类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-decision-jungle)），以及提升决策树（[回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/boosted-decision-tree-regression)和[双类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-boosted-decision-tree)）都基于决策树这个基础机器学习概念。 有许多决策树的变体，但它们都执行相同的操作 — 将特征空间细分成具有大致相同标签的区域。 这些可以是类别一致或常数值的区域，具体取决于是进行分类还是回归。



***决策树将特征空间细分为具有大致统一值的区域***

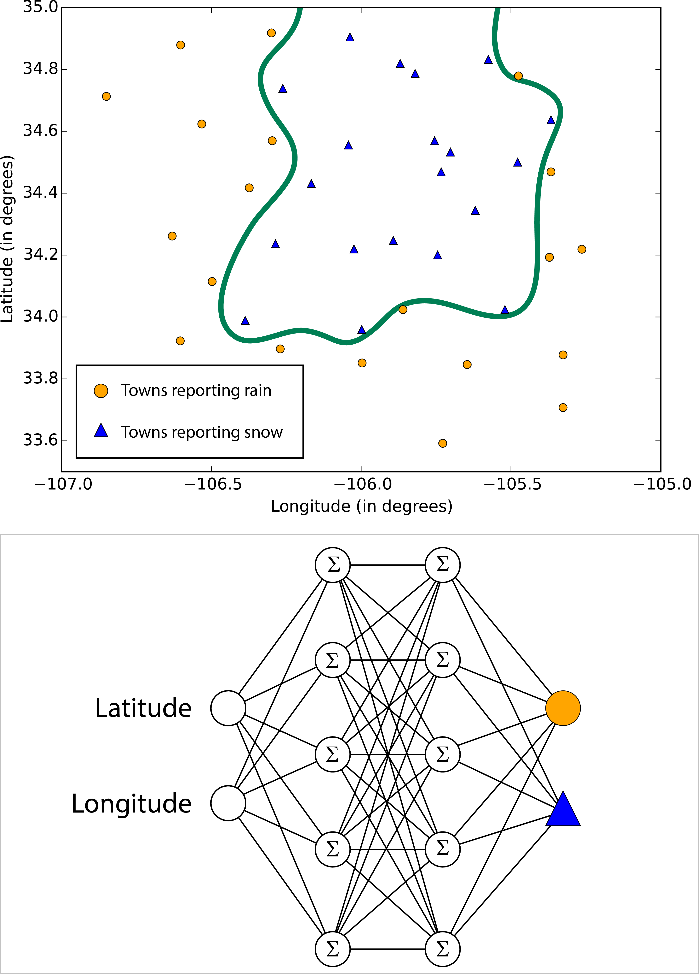
因为特征空间可以细分为任意的较小区域，所以可以想象将它细分为每个区域仅有一个数据点。 这是过度拟合的极端示例。 若要避免此问题，大组树被构造的执行，以便确保树之间互不关联的特殊数学谨慎使用。 此“决策林”的平均值是一个避免过度拟合的树。 决策林可以使用大量的内存。 决策森林是一个变体，其使用的内存较少，但是定型时间稍长。

提升决策树通过限制其细分次数及每个区域中允许的较少数据点数量，可避免过度拟合。 该算法构造树的序列，序列中的每个树都要弥补之前的树留下的错误。 结果是一个倾向于使用大量内存、非常精确的学习者。 了解完整的技术说明，请参阅 [Friedman 的原作](https://www-stat.stanford.edu/~jhf/ftp/trebst.pdf)。

[快速林分位回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/fast-forest-quantile-regression)是特殊情况下决策树的一个变体，在这种情况中不仅要了解区域内数据的典型（中值）值，还要了解其以分位数形式的分布。

**神经网络和感知**

神经网络是涵盖[多类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/multiclass-neural-network)[双类](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-neural-network)和[回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/neural-network-regression)问题的大脑启发式学习算法。 它们有无限种变化，但 Azure 机器学习工作室中的神经网络都是有向无环图的形式。 这意味着输入特征在被转换为输出之前通过一系列层向前（不向后）传递。 在每个层中，输入以各种组合加权、求和，并传递到下一层。 这种简单计算的结合会不可思议的导致产生学习复杂的类边界和数据趋势的能力。 这种多层网络执行的“深度学习”，极大的促进了技术报告和科幻小说。

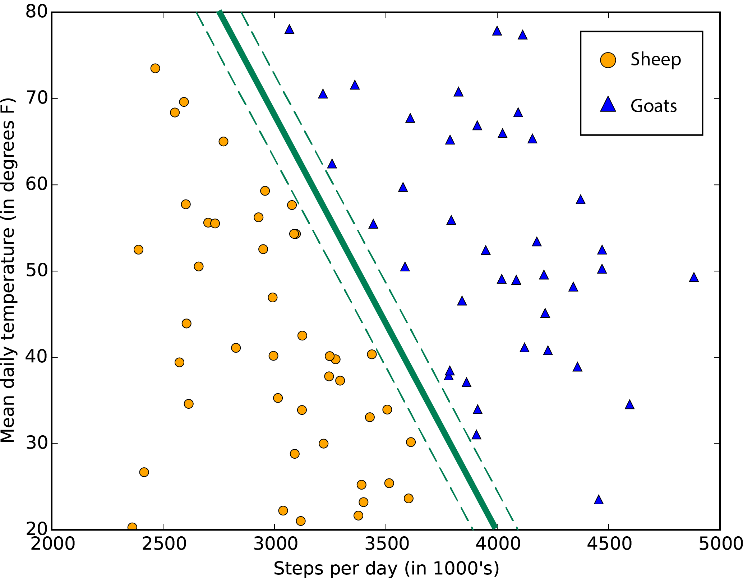
但是，此高性并非免费提供。 神经网络定型可能耗时很长，特别是对于具有大量特征的大型数据集。 它们还具有比大多数算法更多的参数，这意味着参数扫描在很大程度上加长了定型时间。对于那些希望[指定自己的网络结构](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio/azure-ml-netsharp-reference-guide)的成就突出者来说，一切皆有可能。

***神经网络学习的边界可能是复杂而又不规则的***

[双类平均感知器](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-averaged-perceptron)是神经网络对直线上升的定型时间的回答。 它使用提供线性类边界的网络结构。 按照现在的标准，它几乎是原始的，但它已健康的工作了很长时间，并且很小，能够快速学习。

**SVM**

支持向量机 (SVM) 查找以尽可能宽的边距将类分隔的边界。 不能清楚地将两个类分离时，此算法将查找其能找到的最佳边界。 在 Azure 机器学习工作室，编写[双类 SVM](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-support-vector-machine)这么做的一条直线仅与 （SVM 角度而言，它使用线性内核）。 因为它会使此线性相近，以便能够快速运行。 其中它真正的亮点是使用特征型数据数据，如文本或基因组数据。 在这些情况下，除了仅需要使用适度的内存，SVM 与大多数其他算法相比还能更快的分离类，以及能确保更少的过度拟合。



***典型的支持向量机类边界将分隔两个类的边距最大化***

Microsoft Research 的另一个产品[双类局部深层 SVM](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-locally-deep-support-vector-machine) 是 SVM 的非线性变体，其保留有大多数线性版本的速度和内存效率。 它适用于线性方法没有提供足够准确的答案的情况。 开发人员保持快速通过分解为多个小的线性 SVM 问题的问题。 阅读[完整描述](http://proceedings.mlr.press/v28/jose13.html)了解有关他们如何获得这一技巧的详细信息。

使用非线性 SVM 的巧妙扩展，[单类 SVM](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/one-class-support-vector-machine) 绘制了严格概述整个数据集的边界。 它可用于异常情况检测。 任何远在边界之外的新数据点都是不寻常的，值得注意。

**贝叶斯方法**

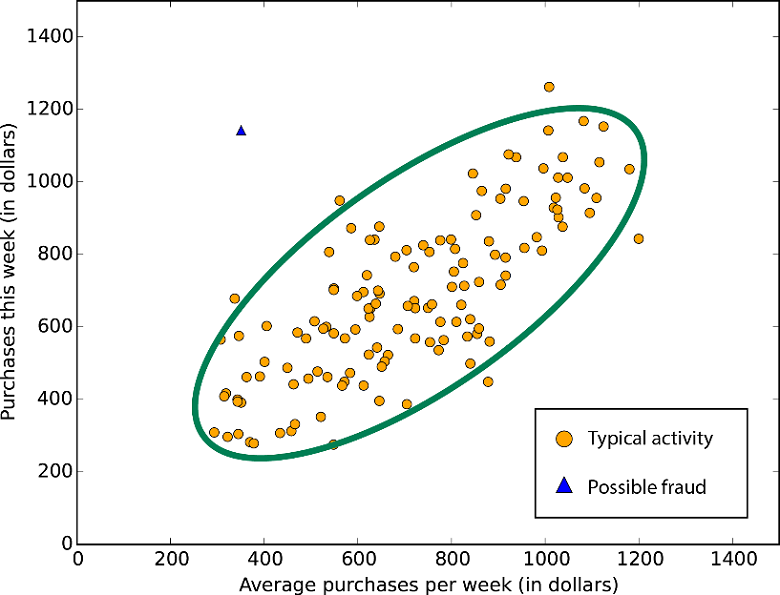
贝叶斯方法具有非常理想的质量：它们避免过度拟合。 其提前对答案的可能分布做出一些假设，从而实现避免过度拟合。 这种方法的另一个副产品是它们的参数非常少。 Azure 机器学习工作室具有两个分类的贝叶斯算法 ([双类贝叶斯点机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/two-class-bayes-point-machine)) 和回归 ([贝叶斯线性回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/bayesian-linear-regression))。 注意，这些方法都假定数据可以分割或用直线拟合。

在历史上，贝叶斯点机是由 Microsoft Research 开发的。 它们背后具有极其出色的理论工作。 感兴趣的学习者可以直接转到阅读 [JMLR 中的原始文章](http://jmlr.org/papers/volume1/herbrich01a/herbrich01a.pdf)和 [Chris Bishop 撰写的充满见地的博客](https://blogs.technet.com/b/machinelearning/archive/2014/10/30/embracing-uncertainty-probabilistic-inference.aspx)。

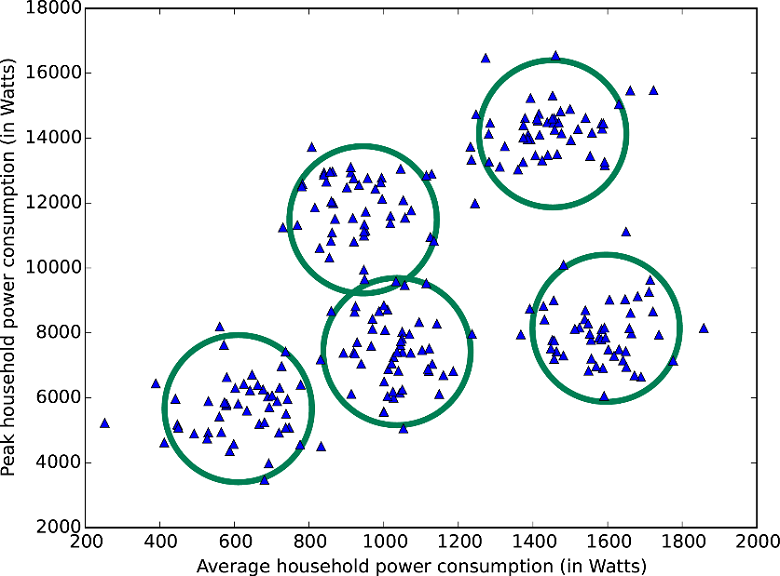
**专用算法**

如果有一个非常具体的目标，可能非常幸运。 在 Azure 机器学习工作室集合中，有一些专门用于以下方面的算法：

* 排名预测（[有序回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/ordinal-regression)）
* 计数预测（[泊松回归](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/poisson-regression)）
* 异常检测（一个基于[主体组件分析](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/pca-based-anomaly-detection)，另一个基于[支持向量机](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/one-class-support-vector-machine)）
* 聚类分析（[K-means](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/k-means-clustering)）

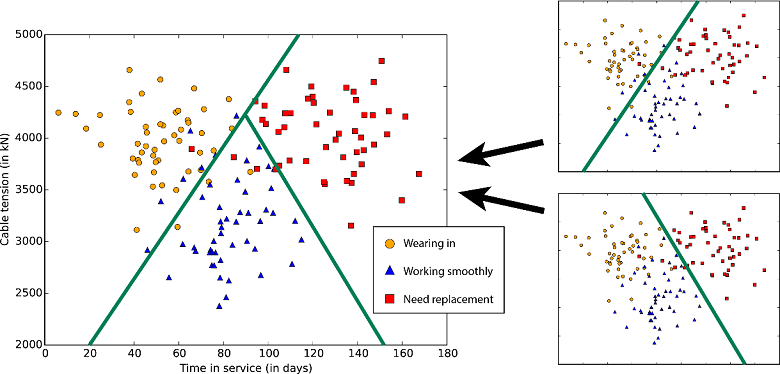


***基于 PCA 的异常情况检测*** - 大多数数据是模式化分布；明显偏离该分布的点都是值得怀疑的



使用 K-means 将数据集分为 5 个群集

还有一个组[一对多多类分类器](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/one-vs-all-multiclass)，它将 N 类分类问题分解成 N-1 个双类类分类问题。 所使用的双类分类器决定了准确度、定型时间和线性属性。



***双类分类器组合形成一个三类分类器***

Azure 机器学习工作室还包括对一个强大的机器学习框架的访问，其标题为 [Vowpal Wabbit](https://docs.microsoft.com/zh-cn/azure/machine-learning/studio-module-reference/train-vowpal-wabbit-version-7-4-model)。 VW 在这里分类，因为可以学习分类和回归问题，甚至可以从部分未标记的数据学习。 可以将其配置为使用多种学习算法、损失函数和优化算法中的任何一种。 它的设计从根本上是高效、并行的，且速度非常快。 其可轻松的处理超级大型的特征集。 VW 由 Microsoft Research 的 John Langford 发起并领导，是改装车算法领域的一级方程式赛车。 不是每个问题都适用于 VW，但如果问题适合，则可能值得继续深入学习。 它也有多种语言的[独立开放源代码](https://github.com/JohnLangford/vowpal_wabbit)。