信息网络专题研究文献阅读报告（应用层-机器学习）

一、文献信息

1.作者：Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Andreas Müller, Joel Nothman, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay

2.论文题目：Scikit-learn: Machine Learning in Python

3.发表途径: Journal of Machine Learning Research

4.发表时间：2012 年 1 月

二、问题、背景与意义

1.研究问题： 集成众多的最先进的机器学习算法，用于中规模的有监督和无监督学习。

2.研究背景： Python 是计算机科学中一种广受欢迎的语言，具有很多优势。机器学习可以用于学术之外的众多工业领域，但非专业人士不易直接使用。

3.研究意义： 文章提出的 scikit-learn 模组用 Python 集成了机器学习算法，使机器学习更易于被非专业人士使用，有更好的性能表现等，其应用程序接口也有一致性，便于操作。

三、Scikit-learn 模块介绍

1.简介

Python 有高度的交互性和成熟的大量的库，十分适用于算法开发和数据分析。作为一种通用的语言，Python 不仅使用在学术环境中，它也越来越多地用在工业当中。本文提出的 scikit-learn 正是利用了 Python 的这些优势，提供了许多著名的机器学习算法的成熟实现方案，并且有着易于使用的接口。这是为了解决在软件和网络工业、以及计算机科学领域之外的诸如生物和物理领域中逐渐增长的非专业人士进行统计数据分析的需求。

Scikit-learn 与 Python 中其他机器学习工具包有许多区别：它发布时使用了 DSB 许可协议，鼓励用户在学术和商业用途中使用；它合并了已编译的代码以提高效率；它只依赖 numpy 和 scipy 仅两个已有的包提供底层支持，易于发布和使用；它着重于指令式编程，不像 pybrain 包使用了数据流框架。

2.项目亮点

代码质量是有所保证的。选文提出的 scikit-learn 的一个目标是提供可靠的执行方案。其代码的质量是经过如 pyflakes 和 pep8 等数据分析工具测试来保证的。并且尽量严格遵守 Python 的编码指南和 numpy 的文档样式，对函数和参数使用统一的命名。

scikit-learn 使用了 BSD 许可协议。大多数 Python 生态系统都是版权受限的，这样的政策适用于商业活动，但也带来了许多限制，不能使用一些已有的代码。本文提出的 scikit-learn 避免使用框架代码，使对象的数量尽量少，减少了使用和开发上的限制，并且使用了 BSD 许可协议，鼓励在学术和商业上使用 scikit-learn。

scikit-learn 是一个社区主导的开发项目。他是以 git 和 github 等工具或社区为基础的，并且欢迎和鼓励研究团队外的人参与进来。

scikit-learn 提供了大量的说明文档，来指导使用者进行安装和操作等，并且提供了 60 多个例程，其中一些还能进行实际应用。这些文档在保持算法使用准确的前提下，尽量减少了术语，增强可读性。

3.代码设计

对象由接口指定，而不是通过继承。为了促进在 scikit-learn 中使用外部对象，不强制使用继承，并且设置了统一的接口。其中心的对象是一个可执行训练操作的估计器，它接收输入数据数组和标签数组作为参数来进行有监督的学习。有监督的分类器（比如支持向量机分类器）可以实现预测。有一些估计器被称为转换器，实现转换功能，能输出经过调整的输入数据。另一个种重要的对象是交叉验证迭代器，它将输入数据分为训练样本和测试数据，用训练样本训练再用测试数据进行验证。

Scikit-learn 可以对估计器的性能表现进行评价或者选择交叉验证的参数，这由含有评价器的 GridSearchCV 对象来完成。特殊的对象可以利用特定的属性使指定的估计器的交叉验证变得更高效，例如 LassoCV 对象。

总的来说，一个管道对象可以由多个转换器和一个估计器组成，它能像一个标准的估计器一样工作，而 GridSearchCV 能够协调所有步骤中的参数。

四、模块分析比较

虽然 scikit-learn 主要的目标是易于使用，并且绝大部分都使用了高级语言实现，但它仍追求最大化计算效率。

文章用 Python 中主流的机器学习工具箱（模块或包）执行了一些算法（SVM 分类器、LARS、弹性网络、KNN、PCA 和 k-means），并比较了不同工具箱所需的执行时间。从给出的文章表格可以看出，scikit-learn 执行这些算法的时间大多相对较短，只有个别较长。以下将分别对执行这类算法时间较短或较长（及效率较高或较低）的原因进行分析：

SVM 分类器：用作对比的包都在后台运行 libsvm（支持向量机库），而 scikit-learn 在绑定时避免内存复制，比起经典的 libsvm 库绑定，这就节省了 40%的开销。scikit-learn 还对 libsvm 打了补丁来提高其在处理高密度数据时的效率。

LARS（最小回归角算法）：在实现 LARS 算法时，scikit-learn 不断迭代精炼残差而不是重新计算残差，这大幅提高了效率。用作对比的 Pymvpa 包则需通过 Rpy R 的绑定来实现，

而这种方法需要进行内存复制,增加了执行时间。相比之下 scikit-learn 的效率就高得多。

弹性网络：文章对 scikit-learn 协调下降的弹性网络模型实现做了测试，在中规模问题上，其性能与高度优化的 Fortrn 版的 glmnet 模块相同。但由于它没有使用 KKT 条件来定义活动集，在大规模问题上其性能受到限制。

KNN：K 最临近分类器构造了样本的球树，在大维度上使用了蛮力搜索。

PCA（主成分分析）：对中到大型的数据集，scikit-learn 实现了基于随机投影的 PCA 截断，从而增加了效率。

k-means：scikit-learn 使用纯的 Python 实现了 k 均值聚类算法，其执行时间比 mlpy 包和 shogun 包都长。这是因为 scikit-learn 使用的 numpy 数组操作需要多次传递数据，因此其性能受到限制，需要较长的执行时间。

五、启发思考

作者团队给出了一个用于解决中规模问题的机器学习算法集成模块 scikit-learn。

文章链接中给出了 scikit-learn 模块的代码和许多例程，通过阅读说明和执行例程可以发现 scikit-learn 确实功能丰富且操作容易，可供非专业人士使用。

scikit-learn 模块的集成理念也有许多值得学习的地方，总结如下：

scikit-learn 集成算法时选用了 Python 这一使用广泛的高级语言，Python 第三方支持库很多，代码的编写、阅读和使用都比较方便。scikit-learn 使用了 BSD 许可协议，使 scikit-learn 不仅能够被广泛用于学术和商业用途，还可以不断自我完善。scikit-learn 对函数和参数统一命名，保证代码可靠性的同时也更加规范，便于他人使用。scikit-learn 中对象由接口指定，不强制使用继承，并且提供了统一的接口，增强了可扩展性。scikit-learn 在实现不同的机器学习算法时，采取了许多优化措施，从而提高了实现各算法的执行效率。由于依赖科学的 Python 生态系统，scikit-learn 还能够被集成到数据统计分析之外的各种领域中去，其应用范围很广。

可以看出 scikit-learn 具有丰富的功能，易于操作，执行效率也较高，是一个性能优异的模块。今后当我碰到需要使用机器学习算法来分析和解决问题时，可以选择使用 scikit-learn 来进行操作。