机器学习系列

Packt

提供全书程序下载！

Go Machine Learning Projects

Go语言机器学习实战

[澳] 周轩逸（Xuanyi Chew） 著　谭励　连晓峰　等译

机械工业出版社
CHINA MACHINE PRESS

机器学习系列

Go 语言机器学习实战

[澳] 周轩逸（Xuanyi Chew）著
谭 励 连晓蜂 等译

机 械 工 业 出 版 社

译者序

Go 语言能够以其简单的语法结构清楚地描述复杂的算法。在机器学习算法中，使用 Go 语言使得程序更易于部署，其代码不仅容易理解和调试，还能够进行性能测试。

本书主要介绍了使用 Go 语言实现机器学习算法的相关内容，告诉读者如何在实际项目中选择最合适的机器学习算法。用机器学习算法可以解决现实生活中许多复杂的问题，本书用 Go 语言详细介绍了多种机器学习算法及其典型应用，这些应用都来自于实际问题。

本书首先介绍了两类机器学习算法：分类和回归。第 2 ~ 5 章分别给出了四个不同的例子：用线性回归预测房价、对垃圾邮件进行分类、利用时间序列分析分解二氧化碳的趋势、通过聚类整理个人推特账户的时间线。之后，本书又围绕机器学习中的识别算法进行了介绍。第 6 章和第 7 章分别是基于神经网络的手写体识别和基于卷积神经网络的手写体识别。最后第 9 章用人脸检测项目来阐述如何将外部服务集成到机器学习项目中以及如何选择适合的机器学习算法。

本书作者具有丰富的实践经验和深厚的理论背景知识。全书侧重于面向实际工程应用，内容深入浅出，易于理解。

本书主要由谭励、连晓峰进行翻译。唐小江、王敏基、董旭、周丽娜、宋艳艳、王浩宇、马子豪、贾惠、吕芯悦、董笑笑、周昊龙、彭璐、宋宇澄、千佳丽等人也参与了翻译工作。全书由连晓峰校正统稿。

本书适合于从事人工智能、机器学习等研究工作的相关人员、数据科学专业人士以及 Go 语言程序员参考学习使用。

由于译者水平有限，书中不当或错误之处恳请业内专家学者和广大读者不吝赐教。

原书前言

Go 语言是一种针对机器学习的良好编程语言。其语法简单，有助于清晰描述复杂算法，同时又不至于使得开发人员无法理解如何运行高效的优化代码。本书旨在阐述如何在 Go 语言中实现机器学习，使程序易于部署，代码易于理解和调试，同时还可进行性能测试。

本书首先介绍如何利用 Go 语言的库文件和功能来配置机器学习的环境。然后，进行了实际房价数据集的回归分析，并在 Go 语言中构建了分类模型，将电子邮件分类为垃圾邮件或正常邮件。利用 Gonum、Gorgonia 和 STL 进行时间序列分析和分解，以及通过聚类来整理个人推特账户的时间线。除此之外，还介绍了如何利用神经网络和卷积神经网络进行手写体识别，这两种神经网络都属于深度学习技术。在掌握了上述所有技术的基础上，将在人脸检测项目的帮助下介绍了如何在具体项目中选择最佳的机器学习算法。

在本书最后，你将树立一种坚实的机器学习思想，牢固掌握 Go 语言中功能强大的库文件，并对实际项目中机器学习算法的具体实现有深刻理解。

本书读者

如果你是一名机器学习工程人员，数据科学专业人员，或是一名想在实际项目中实现机器学习，想更容易实现更加智能的应用程序的 Go 语言程序员，那么本书就非常适合你。

本书主要内容

第 1 章“如何解决机器学习中的所有问题”，介绍了两类机器学习：回归和分类。在该章结束时，你能够对用于构建机器学习程序的数据结构感到得心应手。大多数机器学习算法都是基于在此介绍的数据结构而构建的。接下来，将介绍 Go 语言机器学习，并启动和运行进一步的项目。

第 2 章“线性回归——房价预测”，对实际房价数据集进行回归分析。在此将首先构建必要的数据结构来进行上述分析，并对数据集的初步探索。

第 3 章“分类——垃圾邮件检测”，包括在 Go 语言中构建一个分类模型。数据集是典型的垃圾邮件和正常邮件，目标是构建一个能将电子邮件分类为垃圾邮件或正常邮件的模型。然后，介绍如何自行编写算法，同时利用外部库（如 Gonum）来支持数据结构。

第 4 章“利用时间序列分析分解二氧化碳趋势”，该章体现了时间序列分析的独特作用。时间序列中的数据通常可根据不同的描述目的进行分解。该章展示了如何执行这种分解，以及如何利用 Gonum 绘图工具和 gnuplot 来进行显示。

第 5 章“通过聚类整理个人推特账户的时间线”，包括推特上的聚类分析。该章使用了两种不同的聚类方法——K 均值和 DBSCAN。在该章中，将利用在第 2 章中所积累的一些技巧。另外，还将采用第 4 章中所用的相同库文件。除此之外，还使用了 Marcin Praski 聚类库。

第6章“神经网络——MNIST手写体识别”，开启了图像识别的新领域。由于有用特征是输入特征的非线性积，因此图像处理相对较难。该项目的目的是介绍各种高维数据的处理方法，尤其是在Gonum库中用PCA算法来清洗数据。

第7章“卷积神经网络——MNIST手写体识别”，阐述了如何利用深度学习的最新进展来进行手写体识别，并通过利用Gorgonia构建卷积神经网络，从而实现了99.87%的准确率。

第8章“基本人脸检测”，介绍了人脸检测的一种基本实现。在该章结束时，将能够利用GoCV和PIGO实现一个可用的人脸检测系统。该章为学习如何在工作中选择一种正确算法打下了重要基础。

第9章“热狗或者不是热狗——使用外部服务”，展示了如何将外部服务集成到机器学习项目中以及需要关注的注意事项。

第10章“今后发展趋势”，列出了在Go语言中进行机器学习所需的下一步发展途径。

目　录

第1章
如何解决机器学习中的所有问题

首先，欢迎学习本书。

这是一本比较特殊的书。这并不是一本介绍机器学习（ML）工作原理的书。事实上，最初是假设读者已熟悉掌握了后面章节中所介绍的机器学习算法。但这样又担心本书内容过于空洞。如果读者已了解机器学习算法，那么接下来的工作就是将机器学习算法简单应用到问题的具体背景中！这样的话，本书的10章内容将会在不到30页内介绍完。任何一个为政府机构编写过报告的人都具有这方面的经验。

那么本书究竟是关于什么内容的呢？

本书是关于在给定问题背景的特定情况下如何应用机器学习算法。这些问题可能是具体的，也可能是一时心血来潮而指定的，但为了探索机器学习算法在具体问题中的应用，读者首先必须熟悉算法和问题！因此，本书必须在理解问题和理解用于解决问题的具体算法之间达到一种平衡。在继续讨论之前，需先了解什么是问题、刚才所提到的算法又是什么意思，以及机器学习的作用是什么。

1.1 什么是一个问题

在口语交流中，问题是指需要克服的一些事情。如人们在讲到资金问题时，那么这个问题就可以通过拥有更多的金钱来解决。当有人提到数学问题时，那这个问题可能会通过数学方法来解决。用来解决问题的事物或过程称为解决方案。

在这一问题上，对我而言，定义这样一个常见的普通单词似乎有点奇怪。但要利用机器学习解决问题，就必须具有精确且清晰的概念。必须明确究竟想要解决什么问题。

一个问题可以分解成多个子问题。但在某种程度上，进一步分解这些问题不再具有任何意义。在此只是想提醒读者，问题有各种不同类型。由于问题的类型繁多，因此没有必要一一列举。尽管如此，还是应考虑问题的紧迫性。

如果正在开发一个照片管理工具（或许想要和Google Photos或Facebook竞争），那么识别照片中的人脸并不比在何处保存照片以及如何检索照片更为重要。如果不知道该如何解决后者，那么解决前者的所有现有技术都是无用的。

我认为，尽管紧迫性具有一定主观性，但在考虑较大规模的问题时，这是一种很好的衡量标准。针对一组更具体的示例，考虑三种都需要某种机器学习解决方案的场景，但所需的解决方案具有不同的紧迫性。这些例子显然都是虚构的，与现实生活没有任何关系。目的只

是阐述说明问题。

首先，考虑房地产咨询企业。整个企业的生存取决于是否能够正确预测待售房屋的价格，尽管也可能在某种形式的二级市场上赚钱。对这种企业来说，所面临的机器学习问题是非常紧迫的。必须完全理解解决方案的来龙去脉，否则就会面临倒闭的风险。鉴于常规的紧迫性/重要性划分，机器学习问题也可认为是重要且紧迫的。

其次，是考虑一家网上超市。超市想要知道哪些商品组合销售得最好，这样就可以将其捆绑在一起，使之更具竞争力。这并不是核心业务，因此相较于上一个例子，所面临的机器学习问题紧迫性相对较弱。但也有必要了解解决方案的工作原理。否则可以想象，算法提示将腹泻药和家用品牌的食品捆绑在一起会是什么情况。因此，这确实需要真正了解解决方案是如何实现的。

最后，考虑上面提到的照片应用程序。人脸识别是一个非常好的加分功能，但并不是主要功能。因此，在这三个场景中，该场景下的机器学习问题是最不紧迫的。

在解决问题时，不同的紧迫性程度也会导致要求不同。

1.2 什么是一个算法

1.1 节中多次提到了算法这一术语。在本书中，也广泛采用该词，但毕竟还是需谨慎使用。那么究竟什么是算法呢?

要回答上述问题，首先必须了解什么是程序？程序是指由计算机执行的一系列操作。算法是解决问题的一组规则。因此，机器学习算法是指用于解决某一问题的规则集。它是作为程序在计算机上实现的。

对我而言，在真正并深刻理解究竟什么是算法的过程中，最令人印象深刻的是我在 15 年前的一次经历。当时寄宿在一个朋友家。朋友有一个 7 岁大的孩子，在教授孩子学编程的过程中，由于孩子的理解问题而一直搞不懂语法规则，这让朋友很恼火。我猜测其根本原因是孩子没有完全理解算法的概念。在第二天早上，我们让孩子自己做早餐，要想吃上早餐，他就要按照母亲所列写的来完成一系列步骤。

早餐其实很简单，就是一碗牛奶燕麦片。尽管如此，孩子还是经过大约 11 次尝试才做好这碗燕麦片。最后孩子还流下了委屈的眼泪，并在操作台上弄撒了许多牛奶和燕麦片，但这对孩子来说是一次很好的教育。

这看起来似乎是在虐待儿童。但其实这对我有很大的启发。尤其是孩子对他妈妈和我所说的。大概意思是，现在你已经学会该如何做燕麦片了，为什么需要按照提示步骤来做呢?他妈妈回答道，这就好比是教我如何玩电脑游戏。在此就涉及算法的基本概念。教孩子如何制作燕麦片其实就是教孩子如何编程；这本身就是一种算法!

机器学习算法可以是指一种用于学习的算法，或告知机器采用正确算法的算法。在本书的大部分内容中，将主要是指后者，但如果将前者看作是一种思维训练也是很有用的。自图灵之后的大部分研究工作，都可用机器来代替算法。

在学习下一节之前，最好花一些时间来领悟上述内容。这将有助于在重温时深刻领会我所表达的含义。

1.3 什么是机器学习

什么是机器学习？正如字面意思，是通过机器学习来完成某些工作。尽管机器不能像人类一样学习，但绝对可以模仿人类的某些学习方式。但机器应该学习什么呢？不同的算法可以学习不同的内容，但在此主要讨论的是机器学习程序。或用不太准确的术语来说，机器学习完成正确的事情。

什么是正确的事情呢？在此并不是要讨论哲学上尚未解决的复杂问题，正确的事情其实是作为计算机编程人员的我们所定义的正确内容。

机器学习系统有多种分类形式，最常见的有两类：监督学习和无监督学习。在本书中，会介绍这两类学习的示例，但我认为这种分类方式是直接存在于大脑中的认知区域，而不是操作上的重要区域。这是因为除了少数一些著名算法之外，无监督学习仍处于不断探索研究中。监督学习也同样，但在行业应用上要早于无监督学习。这并不是说无监督学习的学术价值较低——一些不再是学术界热门的研究方法反而证明是非常有用的。本书将在某一章中深入讨论 K－均值和 K－最近邻（KNN）方法。

假设现在已有一种机器学习算法。不过该算法是一个黑箱——对内部无从得知。只是提供一些数据。那么通过其内部机制，可以产生一个输出。该输出结果可能不正确。所以需检查输出结果是否正确。如果不正确，就会更改其内部机制，并反复尝试，直到输出正确结果为止。这就是机器学习算法的一般工作原理。这一过程称为训练。

当然，这需要有“正确”的概念。在监督学习的情况下，人们向机器提供正确的样本数据。在无监督学习的情况下，正确与否取决于其他指标，如输出值之间的距离大小。每种算法都各有所长，但一般来说，机器学习算法都是如上所述的。

1.4 是否需要机器学习

或许最令人疑惑的一个问题是，是否真正需要机器学习来解决问题。毕竟，需要一个充分的理由来解释为什么在本节来讨论这一问题——必须真正了解确切的问题是什么，并在提出问题之前了解具体算法是什么：确实需要机器学习吗？

首先一个问题是，是否存在一个需要解决的问题？我想答案是肯定的，因为毕竟我们芸芸众生都是在这个世界上生存并融入其中。即使是隐世的苦行僧也有其需要解决的问题。也许这一问题应更加具体些：是否存在一个可通过机器学习来解决的问题？

对此，我曾进行了大量咨询，且在早期咨询时，对大多数咨询需求都是持肯定意见。这正是在年轻且缺乏经验时所采取的行为。在肯定回应后常常会出现各种问题。事实证明，应对商业领域以及计算机科学具有更全面的了解后，才能进行更好的咨询服务。

对我而言，随之而来的一个常见问题是，通常需要信息检索解决方案，而并非机器学习解决方案。几年前所收到的需求总结如下：

您好 Xuanyi：

我是＊＊＊。我们几个月前在 YYYY 会议上认识的。我的公司目前正在构建一个提取实体之间关系的机器学习系统。不知您是否有兴趣聊一聊？

当然，这种能够激发我兴趣的关系提取是机器学习中一项极具挑战性的任务。当时我还很年轻，非常渴望能解决一下棘手的难题。所以我和这个公司具体商谈，并根据一些表层信息来得出具体需要计算什么内容。我推荐了几种模型，这些都受到了极大关注。最后构建了一个基于支持向量机的模型，然后开始具体实现。

任何一个机器学习项目的第一步都是先收集数据。因此开始着手处理。但令我惊讶的是，数据都已经过明确分类，且实体也已识别完成。此外，这些实体间存在一个静态不变的关系。一种实体类型与另一种实体类型之间具有一种固定不变的关系。那么机器学习的问题是什么呢？

这是我在经过一个半月的数据收集之后提出的问题。需要做什么？已经具有干净的数据和明确的关系。所有新数据之间也有明确关系。那么需要机器学习做什么？

后来发现，这些数据都来自于当时法律所要求的手动输入。实体关系的定义相当严格。他们真正需要的唯一数据需求是一个清理过的数据库实体关系图。由于数据库结构非常复杂，因此他们没有了解其所需要做的工作就是定义一个外键关系来强制执行这一关系。当需要数据时，这些数据都是来自于个体 SQL 查询。这根本就不需要机器学习！

对于公司的数据库管理员团队，这就是他们所一直强调的内容。

这件事对我有一个启发：在花时间进行研究之前，一定要先搞清楚是否真的需要机器学习解决方案。

从那以后，我就决定采用一种非常简单的方法来确定是否真正需要机器学习。以下就是我的经验法则：

1）问题是否可以形式化为“给定 X，想要预测 Y”。

2）对于一个“是什么”问题通常是值得怀疑的。一个“是什么”问题大概是“我想要知道 X、Y、Z 产品的转换率是什么”。

1.5 一般问题解决过程

只有满足一般法则，才会进一步参与。对我而言，一般的问题解决过程如下：

1）明确问题。

2）将问题转换为更具体的描述。

3）收集数据。

4）进行探索性数据分析。

5）确定将要采用的正确机器学习解决方案。

6）构建一个模型。

7）训练模型。

8）测试模型。

纵观本书所有章节，都将按照上述过程。探索性数据分析部分只在前几章中进行。这意味着在后面的章节中将要执行上述步骤。

另外，我尽量在章节标题中明确所要解决的问题，但毕竟写作是一项艰巨的任务，因此难免有些疏漏和不足。

1.6 什么是一个模型

所有模型都是不完美的，但有些却是非常实用的。

现在，如果现实世界中存在一个能够由任一简单模型精确表征的系统，那将是非常显著的成果。然而，巧妙选择的简化模型往往能够提供非常有效的近似结果。例如，通过一个常数 R 建立的“理想”气体压力 P、体积 V 和温度 T 之间的定律 $PV = RT$，并不适用于所有实际气体，但该定律可提供有用的近似值，且由于是从气体分子行为的物理现象出发的，从而使得其结构具有丰富信息。

对于这样一种模型，不必再质疑“模型是正确的吗?”如果“正确”的概念是“完全真实”，那答案肯定是“不正确”。其实，唯一感兴趣的问题是“模型是否具有启发性和实用性?”

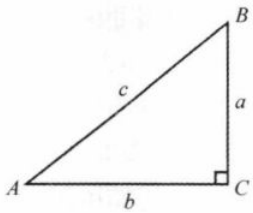
——George Box（1978）

模型训练是一种非常普遍的方法，尽管受到类似《生活大爆炸》的嘲讽。但模型训练并不是真正训练。首先，规模不同。模型训练并不要求按实际训练所需的方式执行。目前模型训练有不同等级，每种等级都要比上一级更接近于实际训练。

从这个意义上来说，模型就是现实世界的一种表征。那么需要用什么来表征呢？总的来说，就是数字。一个模型其实是一组描述现实世界的数字，或更多一些。

在每次试图解释什么是模型时，都会得到这样的回答：“你不能把模型简化为一堆数字!”那么我所说的“数字”是什么意思？

以下图所示的直角三角形为例：



如何描述所有直角三角形？可以具体描述如下：

$$\angle ABC + \angle BCA + \angle CAB = 180°$$

$$\exists \angle = 90°$$

这意味着在一个直角三角形中所有角度之和为 180°，且存在一个 90°的角度。这就足以描述笛卡儿空间中的所有直角三角形。

但这描述的是三角形本身吗？并非如此。从亚里士多德时代起，这一问题就一直困扰着哲学家们。在此并不讨论哲学问题，否则将会导致本章没完没了。

针对本章目的，我们将模型定义为描述现实世界的一些值，以及产生这些值的算法。这些值通常是数字，尽管也可能是其他类型。

1.6.1 什么是一个好的模型

模型是一组描述现实世界的值，以及产生这些值的程序。对此，已从前面得出。现在需要对模型进行一些具体评判——这一模型是好还是坏。

一个好的模型需要能对现实世界进行准确的描述。这也是最一般的表征形式。因此，这一关于好模型的表述包含了很多概念。接下来，必须对这一抽象概念更加具体化。

机器学习算法是通过一组数据进行训练的。对于机器而言，这一组数据就是现实世界。但对于我们而言，提供给机器用于训练的数据并不是现实世界。对人类来说，对现实世界的了解要比机器知道的更多。所以在提到“一个好的模型需要对现实世界准确描述”时，其中的“现实世界”一词具有两种含义——机器所知道的现实世界和人类所了解的现实世界。

机器只能看到我们所了解的现实世界的一部分。对于现实世界中的其他部分，机器还一无所知。因此，若能够针对未知的输入提供正确的输出，那么这就是一个好的机器学习模型。

以一个具体的例子为例，假设现在有一个能够确定一幅图像是否是热狗图像的机器学习算法。在此仅提供了热狗和汉堡的模型图像。对机器来说，现实世界就仅仅是热狗和汉堡的图像。如果将蔬菜图像作为输入传递给机器会产生什么结果呢？一个好的模型可以归纳，并判断这不是一个热狗；而一个差的模型只能崩溃。

由此可知，一个好的模型的定义是指其能够很好地对未知情况进行归纳。

通常，作为构建机器学习系统过程中的一部分，我们希望对这一概念进行测试。因此，需要将数据集拆分为测试数据集和训练数据集。机器将在训练数据集上进行训练，一旦训练完成，就可以将测试数据集输入机器来测试该模型的性能优劣。当然，这要假设机器对测试数据集完全未知。因此，一个好的机器学习模型能够在对测试数据集完全未知的情况下生成正确的输出结果。

1.7 本书主要内容与章节安排

下面是关于本书写作方式的一些说明。正如可能已猜到的，我决定采用一种更具对话性的方式来完成本书。因为发现这种语气会对那些不熟悉机器学习算法的读者更为友好。如果还没注意到的话，其实我在写作方式上是相当固执己见的。在整个写作过程中，我尽量阐述清楚基本概念和注意事项。最终效果由读者自行确定。但作为一份快速指南，在讲到算法及其工作原理时，只是简单描述。而在讨论应完成什么功能，以及如何编写代码时，会详细介绍。

本书中各章的组织通常有两种模式。第 1 种是，随着章节的继续问题越来越难；第 2 种是，可能会选择性地将这些章节分为三部分。第 1 部分——第 2 章，线性回归——房价预测；第 3 章，分类——垃圾邮件检测；第 4 章，利用时间序列分析分解二氧化碳趋势；第 7 章，卷积神经网络——MNIST 手写体识别；第 8 章，基本人脸检测，这些都对应于面临紧迫性机器学习问题的读者。第 2 部分——第 5 章，通过聚类整理个人推特账户的时间线；第 9 章，热狗或非热狗——采用外部服务；第 6 章，神经网络——MNIST 手写体识别；第 7 章，卷积神经网络——MNIST 手写体识别；第 8 章，基本人脸检测，这适用于具有与第二个示例

相似的机器学习问题的读者。第 3 部分是本书最后两章，主要针对机器学习问题并不是很紧迫，但仍需解决的读者。

在第 8 章之前，每一章都会有 1 ~ 2 节来专门解释算法本身。我坚信，如果没有基本了解所采用的算法，那么就无法编写出任何有意义的程序。当然，也可以直接利用他人提供的算法，但如果没有正确理解，也是没有意义的。有时毫无意义的程序可能会产生结果，但这类似于有时大象貌似会进行算术运算，或宠物狗似乎会做出类似说话的行为。

这也意味着可能听起来我对一些想法不屑一顾。例如，我对使用机器学习算法来预测股票价格就很不屑。我认为这不会产生什么富有成效的结果，因为我比较了解股票市场产生价格的基本过程，以及时间对此产生的混杂效应。

然而，时间会证明一切。可能会有一天，某人提出一种机器学习算法可以很好地应用于动态系统，但目前肯定没有。当今正处于一个计算新纪元的开创期，必须要尽量更好地理解这些技术。重温历史，我们可以学到很多。因此，我也尽量介绍一些相关技术产生的历史背景。但这绝不是一个全面的综述。事实上，这也没有太多的规律性。不过，还是希望这些内容能增添本书的趣味性。

1.8 为什么选择 Go 语言

本书是一本关于采用 Go 语言实现机器学习的书。Go 语言是一种相对较为独特的编程语言。只要掌握 Go 语言，就无需学习其他语言了。这似乎有些霸道，但它确实提供了非常愉悦的编程体验。另外，也会极大提高团队的工作效率。

此外，与 Python 相比，Go 语言是一种高效的编程语言。我目前几乎完全采用 Go 语言来完成机器学习和数据科学工作。

Go 语言还具有跨平台开发的优势。在开发过程中，开发人员可以选择在不同的操作系统上进行开发。Go 语言在所有操作系统上都能有效运行。利用 Go 语言编写的程序还可以在其他平台上交叉编译。这会使得部署更加容易。完全没有必要和 Docker 或 Kubernetes 混在一起。

那么利用 Go 语言来实现机器学习有什么不足之处吗？仅对库文件开发人员会有影响。一般来说，可以直接使用 Go 语言中的机器学习库。但为了能够直接使用，必须摒弃之前的编程方式。

1.9 快速启动

首先安装 Go 语言，可在 https：//golang. org 上找到安装文件。另外，这个网站还提供了关于 Go 语言的详细指南。接下来，就可以快速启动了。

1.10 函数

函数是在 Go 语言中进行各种计算的主要方法。

下面就是一个函数：

```
func addInt(a, b int) int { return a + b }
```

可以调用 func addInt(a, b int)，这是一个函数签名。函数签名由函数名、参数和返回类

型组成。

函数名是 addInt。注意要采用正确格式。函数名采用骆驼拼写法，这是 Go 语言中首选的名称拼写法。若函数名的首字母大写，如 AddInt 表明该函数需导出。总体而言，在本书中，不必考虑导出或非导出的函数名，这是因为我们主要使用函数。但如果编写一个包，那就需要注意大小写。导出名称可从包外获得。

接着，需要注意 a 和 b 都是参数，且均为 int 型。稍后将会讨论数据类型，同样的函数也可写为

```
func addInt(a int, b int) int { return a + b }
```

然后，是函数的返回值。该函数 addInt 返回一个 int 型。这意味着函数被正确调用时，如：

```
z := addInt(1, 2)
```

返回的 z 是一个 int 型。

定义返回类型后，{…} 是函数体。若在本书中给出 {…}，意味着函数体的内容对于所讨论的内容而言并不重要。书中的某些部分可能包含了函数体的片段，但没有给出函数签名 func foo (…)。同样，这些片段也是针对所正在讨论的内容。希望读者能根据书中的上下文分析出整个函数。

一个 Go 语言函数可以返回多个结果。函数签名类似于：

```
func divUint(a, b uint) (uint, error) { ... }
func divUint(a, b uint) (retVal uint, err error) { ... }
```

同样，区别主要在于返回值的命名。在第二个示例中，返回值分别命名为 retVal 和 err。retVal 是 unit 型，而 err 是 error 型。

1.11 变量

一个变量的声明如下：

```
var a int
```

这表示 a 是一个 int 型变量，意味着 a 可以包含类型为 int 型的任何值。典型的 int 型应是 0、1、2 等值。上述语句读起来似乎有些奇怪，但实际上通常都是这么使用的。int 型的所有值通常也都是 int 型。

上述是一个变量声明，接下来要将一个值赋予该变量：

```
s := "Hello World"
```

在此，是定义 s 为一个字符串，且值为“Hello World”。语法：= 只能出现在函数体中。主要是使得程序员不必输入`var s string = "Hello World"`。

关于变量的用法需注意：Go 语言中的变量应看作是一个具有名称的容器，其中包含具体值。变量名称很重要，因为这可以告知读者其包含的值的信息。然而，变量名称也不一定非要体现具体不同之处。我就经常用 retVal 来命名返回值，再在别处命名一个不同的名称。具体示例如下：

```
func foo(...) (retVal int) { ... return retVal }
func main() {
    something := foo()
    ...
}
```