**Tech Summit 2018**

**Hands-on lab**

从零开始用AI

****

**目录**

[从零开始用AI 1](#_Toc524879088)

[概览 3](#_Toc524879089)

[目标 3](#_Toc524879090)

[系统需求 3](#_Toc524879091)

[设置 4](#_Toc524879092)

[练习 5](#_Toc524879093)

[练习 1：理解以及处理数据 6](#_Toc524879094)

[任务 1 – 放置并查看数据集 6](#_Toc524879095)

[任务 2 – 将数据集转化为图片 7](#_Toc524879096)

[练习 2：创建训练学习模型 10](#_Toc524879097)

[任务 1 – 选择机器学习算法 10](#_Toc524879098)

[任务 2 – 链接训练模型 10](#_Toc524879099)

[任务 3 – 链接评分模型 11](#_Toc524879100)

[任务 4 – 链接评估模型并执行R语言脚本 11](#_Toc524879101)

[练习 3：优化模型并且发布 14](#_Toc524879102)

[任务 1 – 更换神经网络 14](#_Toc524879103)

[任务 2 – 发布训练模型 17](#_Toc524879104)

# 概览

AI人工智能不仅是近几年火热的技术话题，更是上升到国家层面的技术战略。 ML机器学习作为这一波AI人工智能迅猛发展的引擎，也越来越多为人所熟知。

为了帮助人们方便快捷的使用ML机器学习来为现在的业务插上AI的翅膀，微软Azure云中除了提供封装好的认知服务，还提供了所见即所得的Azure Machine Learning Studio机器学习工作室。

最为重要的是，Azure ML Studio提供了免费的版本，而入门的教程如此简单，所以本次动手实验我们一起看看，如何借助Azure ML Studio实现零基础、零成本的AI入门。

**目标**

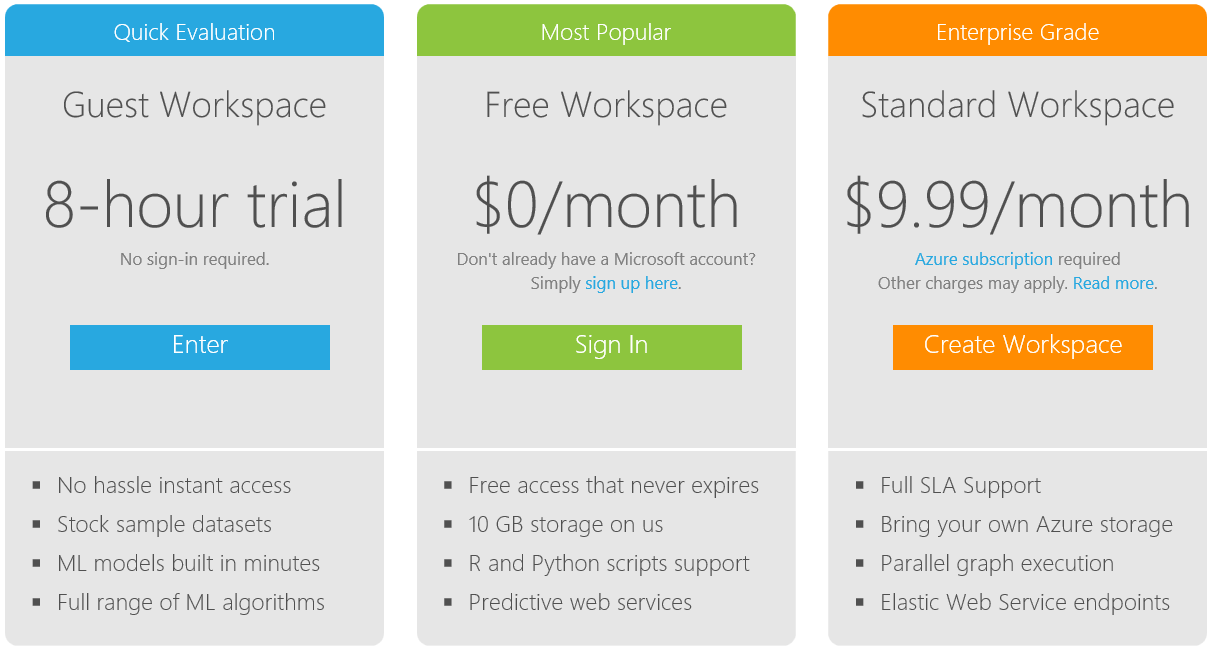
* 1. 在这个动手实验里，您可以体验：
  + 获取并使用免费的Azure Machine Learning Studio环境
  + 使用免费的在线Azure Jupyter Notebooks
  + 尝试创建手写数字识别的机器学习模型
  + 使用不同的神经网络来优化手写数字识别机器学习模型

**系统需求**

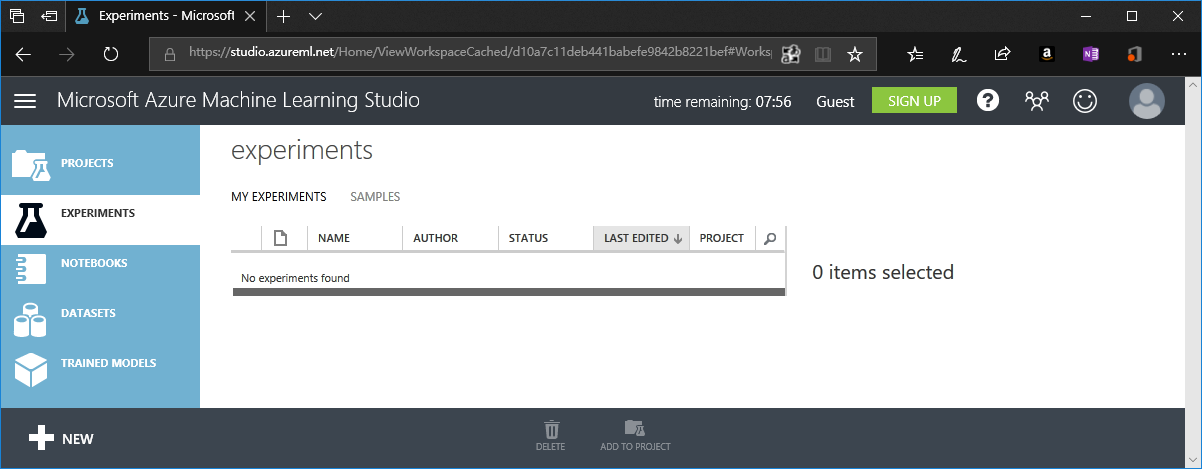
* 1. 为完成本次动手实验，您需要准备:
  + 浏览器，例如Edge、Internet Explorer或Chrome
  + 一个Microsoft Account，例如hotmail账号
  + Excel软件

**设置**

* 1. 您需要使用以下步骤来设置动手实验环境：
  2. 打开浏览器，访问：<https://studio.azureml.net/>
  3. 找到 “Sign up here” 链接，点击



* 1. 选择 “Free Workspace“ 选项。如果还没有准备好您的 Microsoft Account，可以点击 ”sign up here“ 链接来注册。点击 “Sign In” 按钮，使用您的Microsoft Account登录
  2. 您应该能够看到类似如下的机器学习工作室界面：



**练习**

* 1. 本动手实验包含以下练习：
  2. 理解以及处理数据

查看MNIST数据集并理解手写数字图片的数据处理

* 1. 创建训练学习模型

创建用于训练机器学习的模型并使用模型进行训练

* 1. 优化模型并且发布

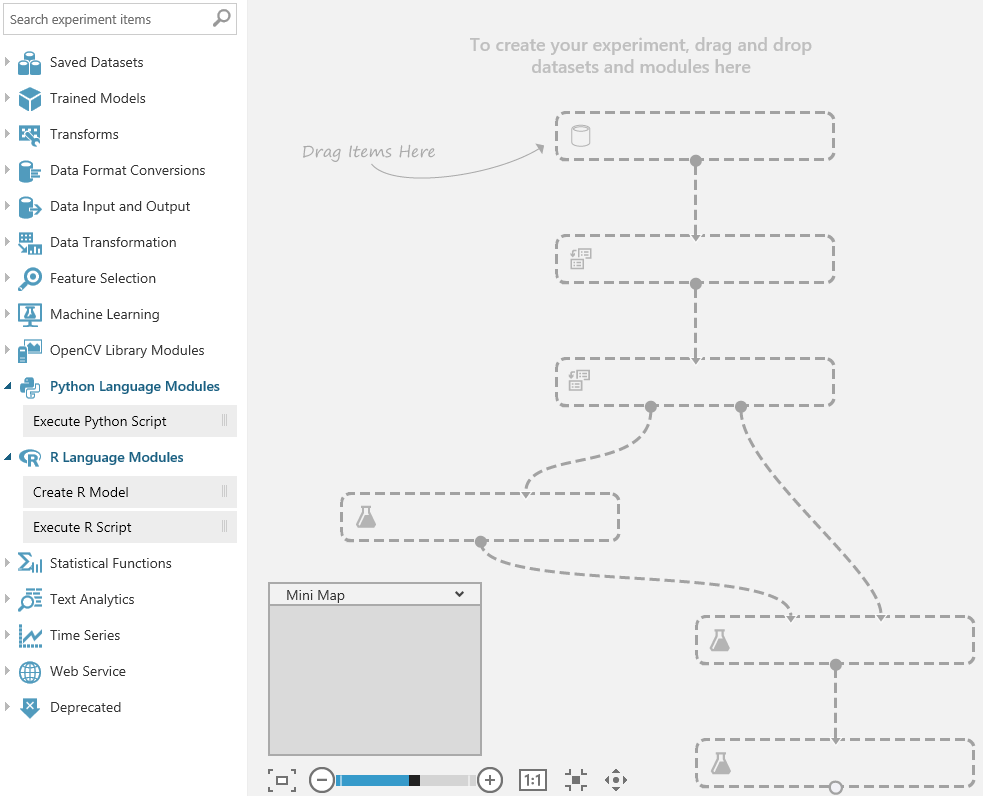
使用不同架构的神经网络来优化手写数字识别模型，并发布训练好的模型

* 1. 预计完成动手实验时间： **30 到 45 分钟**.

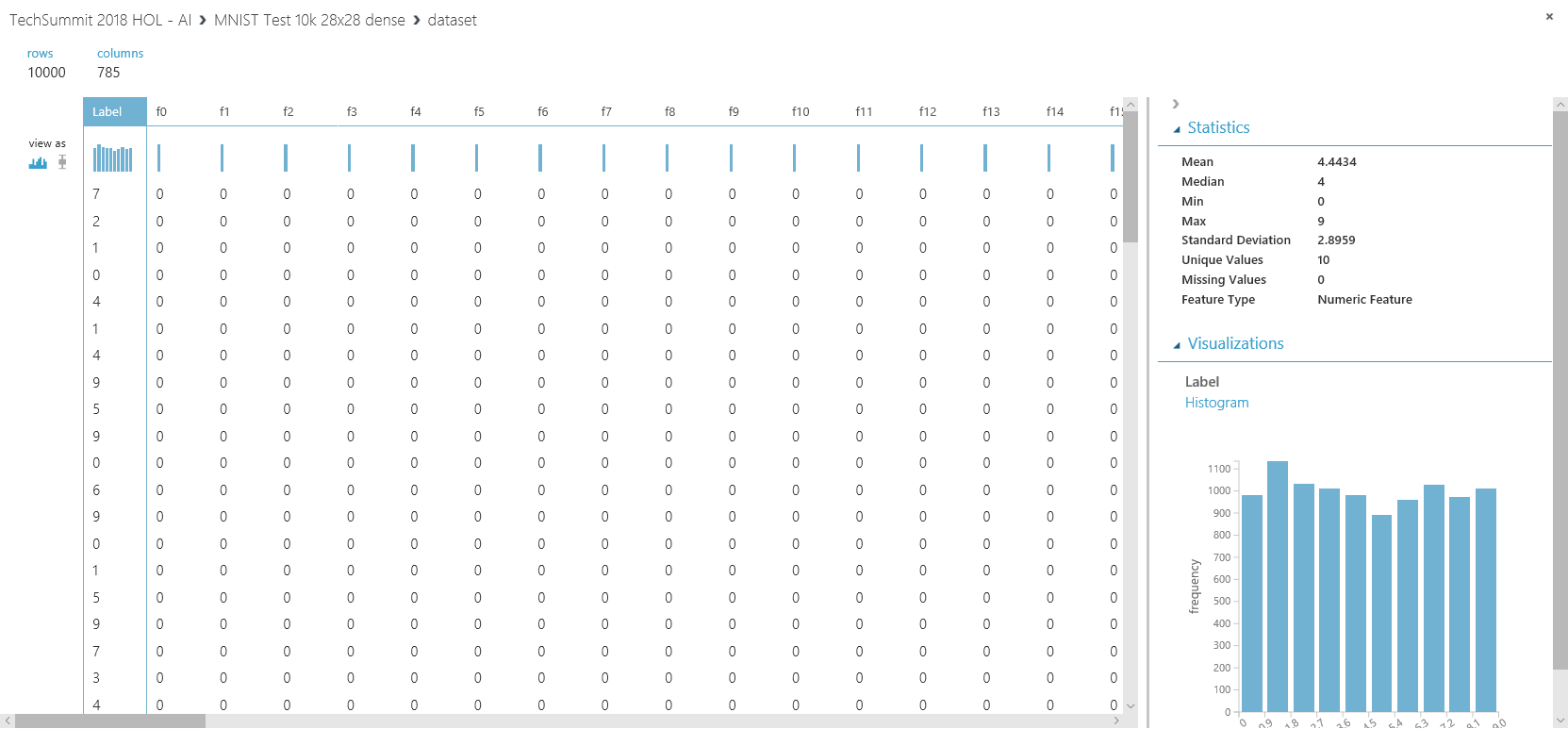
# 练习 1：理解以及处理数据

## 任务 1 – 放置并查看数据集

* 1. 在机器学习工作室下侧工具栏点击“NEW”，新建一个实验，应能看到如下工作区：



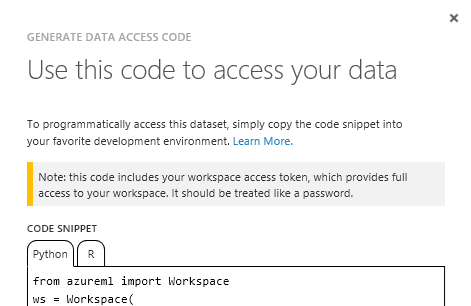
* 1. 在机器学习工作室左侧面板，找到“Saved Dataset“并展开，找到并展开 ”Samples“ 分支，点击然后找到“MNIST Train 60k 28x28 dense“数据集，拖拽并放置到编排区域
  2. 相同位置找到“MNIST Test 10k 28x28 dense”数据集，拖拽并放置到编排区域
  3. 右键点击“MNIST Test 10k 28x28 dense”数据集的输出节点，选择“Visualize”显示数据集内容，应形如下图：



请结合讲师的讲述，了解各字段代表的数据含义。

## 任务 2 – 将数据转化为图片

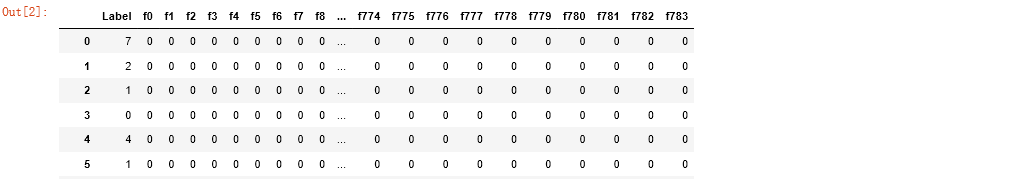
* 1. 右键点击“MNIST Test 10k 28x28 dense”数据集的输出节点，选择“Generate Data Access Code…“，显示形如下图，复制”Python“框内的代码



* 1. 右键点击“MNIST Test 10k 28x28 dense”数据集的输出节点，选择“Open in a new Notebook”，然后选择“Python 3”，打开用于执行Python程序的笔记本
  2. 将第一个代码块中的代码，替换为刚才复制的代码



* 1. 点击上方菜单的“Run”按键，运行代码块，然后运行第二个代码块。如果代码运行成功，应看到如下显示，注意观察最后显示的行列数



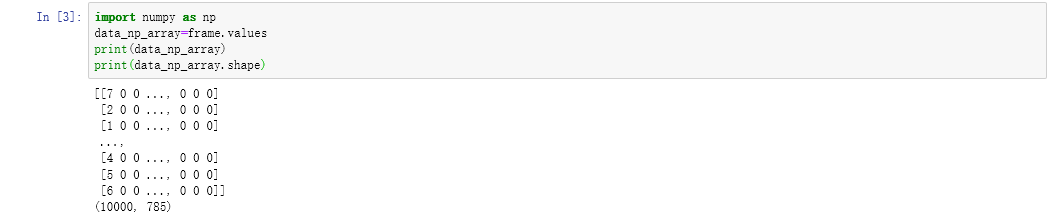
┋



|  |
| --- |
| import numpy as np  data\_np\_array=frame.values  print(data\_np\_array)  print(data\_np\_array.shape) |

* 1. 在最下方的代码块中输入以下内容并运行：

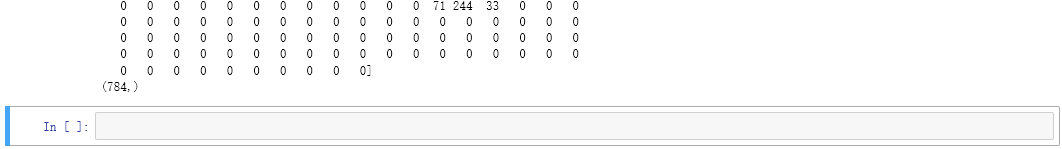
若代码成功运行，将看到如下图所示，注意观察最后显示的数组Array维度



* 1. 在最下方的代码块中输入以下内容并运行：

|  |
| --- |
| select\_img\_array=data\_np\_array[999,1:]  print(select\_img\_array)  print(select\_img\_array.shape) |

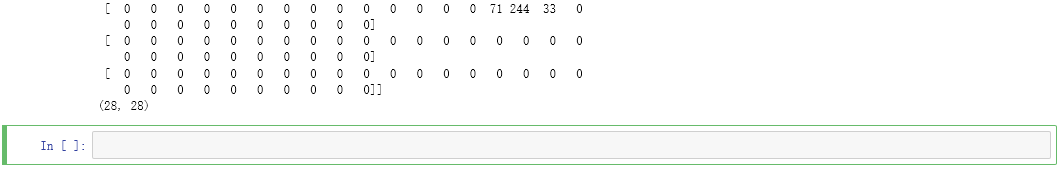
999为选择提取的图片，可自行更改。如代码运行正常，应能看到如下显示，注意观察最后的数组Array维度，思考去掉了哪个数字，为什么去掉？



* 1. 在最下方的代码块中输入以下内容并运行：

|  |
| --- |
| img\_data=select\_img\_array.reshape((28,28))  print(img\_data)  print(img\_data.shape) |

如代码正确运行，应看到如下显示， 注意观察输出数组维度，思考为何要进行转换？



|  |
| --- |
| %matplotlib inline  import matplotlib.pyplot as plt  plt.gray()  plt.imshow(img\_data) |

* 1. 在最下方的代码块中输入以下内容并运行：

如代码正确运行，应能看到如下显示：



# 练习 2：创建训练学习模型

## 任务 1 – 选择机器学习算法

* 1. 在机器学习工作室左侧面板，找到“Machine Learning“并展开，找到并展开 ”Initialize Model“ 分支，点击然后展开 “Classification” 分支，然后找到“Multiclass Neural Network“
  2. 拖拽并放置该组件到编排区域，然后按照如下参数进行配置：

|  |  |
| --- | --- |
| Create trainer mode | Single Parameter |
| Hidden layer specification | Customer definition script |
| The learning rate | 0.01 |
| Number of learning iterations | 30 |
| The initial learning weights diameter | 0.3 |
| The momentum | 0 |
| The type of normalizer | Min-Max normalizer |
| Random number seed | 1 |

* 1. 在 “ Neural network definition“ 编辑框中，输入如下内容：

|  |
| --- |
| input Picture [28,28];  hidden H1 [200] from Picture all;  hidden H2 [200] from H1 all;  output Result [10] softmax from H2 all; |

## 任务 2 – 链接训练模型

* 1. 在机器学习工作室左侧面板，找到 “Machine Learning“ 并展开，找到并展开 ”Train“ 分支，然后找到 “Train Model“
  2. 拖拽并放置该组件到编排区域，链接上一步配置好的算法组件“Multiclass Neural Network“到该组件的左边输入节点，链接 “MNIST Train 60k 28x28 dense” 数据集到该组件的右边输入节点
  3. 点击右侧 “Launch column selector“ ，选择 “Label” 列

## 任务 3 – 链接评分模型

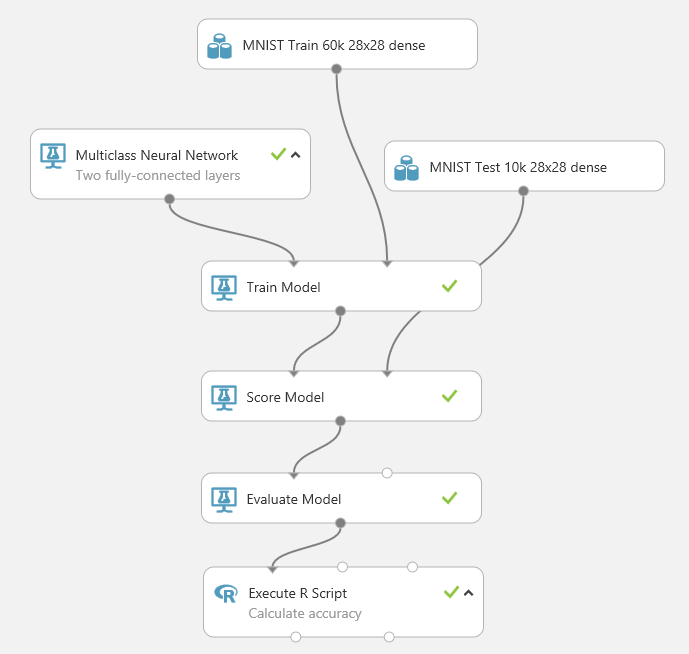
* 1. 在机器学习工作室左侧面板，找到 “Machine Learning“ 并展开，找到并展开 ”Score“ 分支，然后找到 “Score Model“
  2. 拖拽并放置该组件到编排区域，链接上一步配置好的训练模型组件“Train Model“到该组件的左边输入节点，链接 “MNIST Train 10k 28x28 dense” 数据集到该组件的右边输入节点
  3. 确认右侧“Append score column to output”选择框已勾选

## 任务 4 – 链接评估模型并执行R语言脚本

* 1. 在机器学习工作室左侧面板，找到“Machine Learning“并展开，找到并展开”Evaluate“分支，然后找到“Evaluate Model“
  2. 拖拽并放置该组件到编排区域，链接上一步配置好的模型组件“Score Model“到该组件的左边输入节点，空置右边输入节点
  3. 在机器学习工作室左侧面板，找到“R Language Modules“展开，找到”Execute R Script“
  4. 拖拽并放置该组件到编排区域，链接上一步配置好的模型组件“Evaluatue Model“到该组件的左边输入节点，空置中间、右边输入节点
  5. 在右侧“R Script”输入框中输入如下内容：

|  |
| --- |
| dataset1 <- maml.mapInputPort(1)  feat <- data.matrix(dataset1[,2:11])  total = sum(sum(feat))  correct = sum(diag(feat))  accuracy = correct / total  acc = as.data.frame(accuracy)  maml.mapOutputPort("acc"); |

* 1. 查看链接好的所有组件，应如下图所示：



* 1. 点击机器学习工作室下侧的“RUN“，运行训练过程，并等待其结束
  2. 右键点击“Execute R Script”组件的左边输出节点，选择“Visualize”，查看并记录显示的识别准确度accuracy。

# 练习 3：优化模型并且发布

## 任务 1 – 更换神经网络

* 1. 选中“Multiclass Neural Network”组件，然后按照如下参数修改配置：

|  |  |
| --- | --- |
| Create trainer mode | Single Parameter |
| Hidden layer specification | Customer definition script |
| The learning rate | 0.01 |
| Number of learning iterations | 20 |
| The initial learning weights diameter | 0.9 |
| The momentum | 0.7 |
| The type of normalizer | Min-Max normalizer |
| Random number seed | 1 |

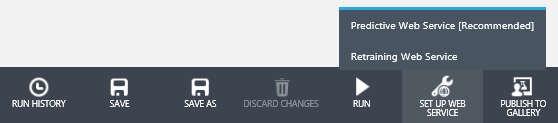
* 1. 在 “ Neural network definition“ 编辑框中，输入如下内容：

|  |
| --- |
| const { T = true; F = false; }  const {  // input image size  ImgW = 28;  ImgH = 28;  // first convolutional layer parameters  C1Maps = 5;  C1KernW = 5;  C1KernH = 5;  C1StrideW = 1;  C1StrideH = 1;  // The following formula computes dimensions with padding enabled.  C1OutW = (ImgW - 1) / C1StrideW + 1;  C1OutH = (ImgH - 1) / C1StrideH + 1;  // first pooling layer parameters  P1KernW = 2;  P1KernH = 2;  P1StrideW = 2;  P1StrideH = 2;  // The following formula computes dimensions with no padding.  P1OutW = (C1OutW - P1KernW) / P1StrideW + 1;  P1OutH = (C1OutH - P1KernH) / P1StrideH + 1;  // second convolutional layer parameters  C2Maps = 10;  C2KernW = 5;  C2KernH = 5;  C2StrideW = 1;  C2StrideH = 1;  // The following formula computes dimensions with padding enabled.  C2OutW = (P1OutW - 1) / C2StrideW + 1;  C2OutH = (P1OutH - 1) / C2StrideH + 1;  // Since Z dimension of the kernel is 1 and sharing is disabled in Z dimension  // total number of maps is a product of input maps and layer maps.  C2OutZ = C2Maps \* C1Maps;  // second pooling layer parameters  P2KernW = 2;  P2KernH = 2;  P2StrideW = 2;  P2StrideH = 2;  // The following formula computes dimensions with no padding.  P2OutW = (C2OutW - P2KernW) / P2StrideW + 1;  P2OutH = (C2OutH - P2KernH) / P2StrideH + 1;  }  input Picture [ImgH, ImgW];  hidden C1 [C1Maps, C1OutH, C1OutW]  from Picture convolve {  InputShape = [ImgH, ImgW];  KernelShape = [C1KernH, C1KernW];  Stride = [C1StrideH, C1StrideW];  Padding = [T, T];  MapCount = C1Maps;  }  hidden P1 [C1Maps, P1OutH, P1OutW]  from C1 max pool {  InputShape = [C1Maps, C1OutH, C1OutW];  KernelShape = [1, P1KernH, P1KernW];  Stride = [1, P1StrideH, P1StrideW];  }  hidden C2 [C2OutZ, C2OutH, C2OutW]  from P1 convolve {  InputShape = [C1Maps, P1OutH, P1OutW];  KernelShape = [1, C2KernH, C2KernW];  Stride = [1, C2StrideH, C2StrideW];  Sharing = [F, T, T];  Padding = [F, T, T];  MapCount = C2Maps;  }  hidden P2 [C2OutZ, P2OutH, P2OutW]  from C2 max pool {  InputShape = [C2OutZ, C2OutH, C2OutW];  KernelShape = [1, P2KernH, P2KernW];  Stride = [1, P2StrideH, P2StrideW];  }  hidden H3 [100]  from P2 all;  output Result [10] softmax  from H3 all; |

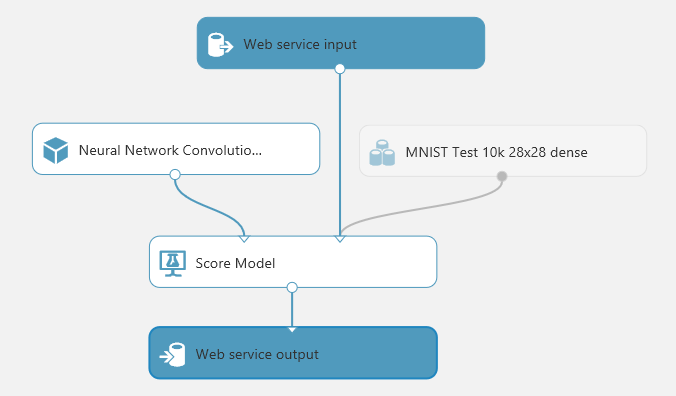
* 1. 点击机器学习工作室下侧的“RUN“，运行训练过程，并等待其结束
  2. 右键点击“Execute R Script”组件的左边输出节点，选择“Visualize”，查看并记录显示的识别准确度accuracy。

## 任务 2 – 发布训练模型

* 1. 点击机器学习工作室下侧的“SET UP WEB SERVICE“，选择“Predictive Web Service”，等待发布过程结束



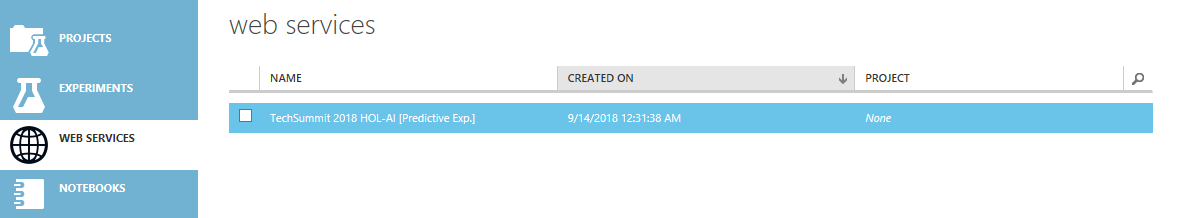
* 1. 机器学习工作会自动生成预测服务并发布



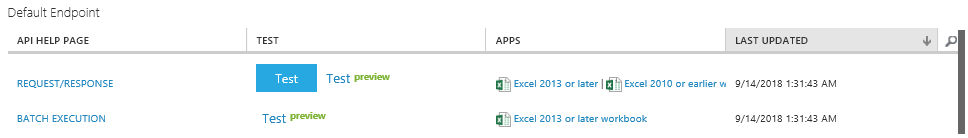
* 1. 点击机器学习工作室下侧的”RUN”，待模型运行结束后点击“DEPLY WEB SERVICE“

## 任务 3 – 使用预测服务

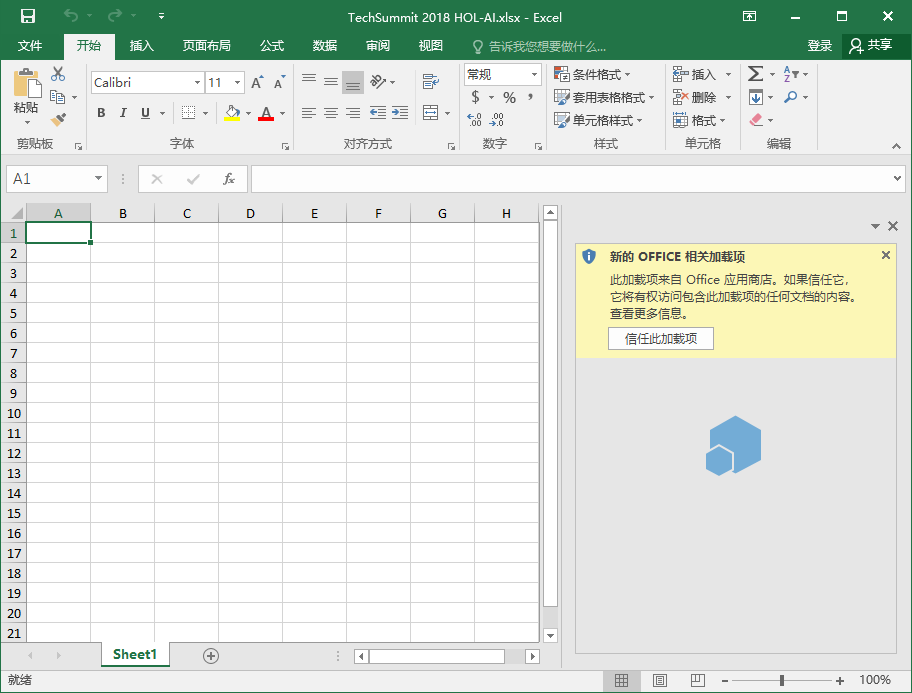
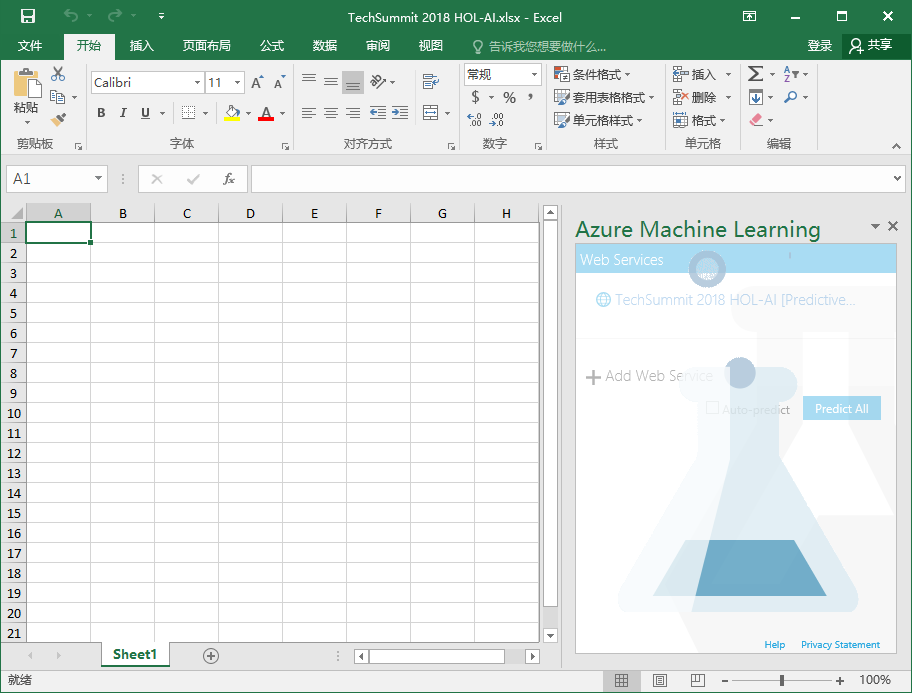
* 1. 点击“WEB SERVICES”，然后点击已经部署的预测Web服务



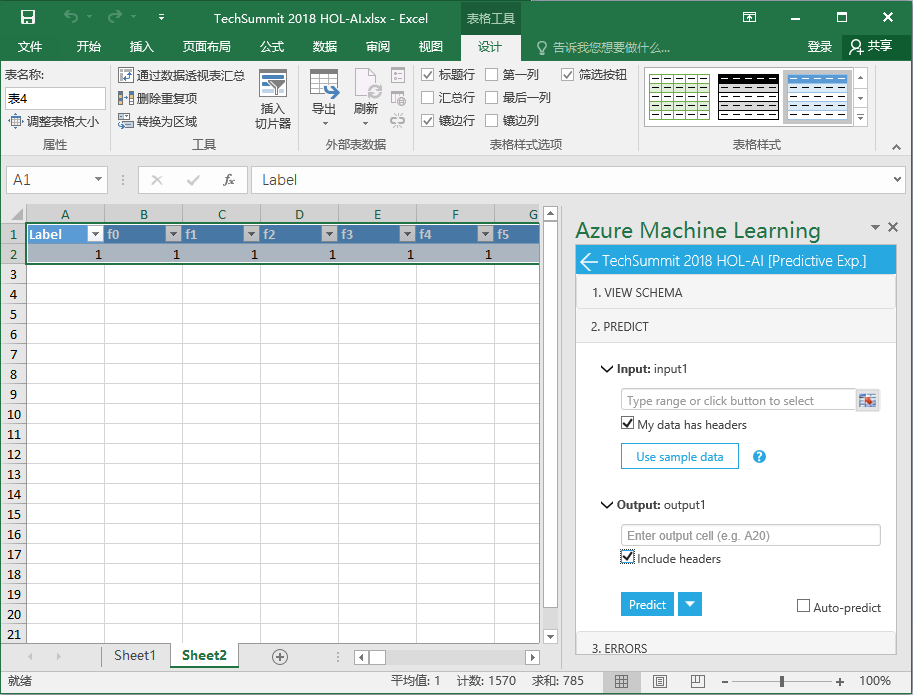
* 1. 点击“BATCH EXEUTION”后面的“Excel 2013 or later workbook”下载并打开Excel文件



* 1. 如提示新的加载项，点击“信任此加载项”，然后等待Excel加载项连接Azure ML服务

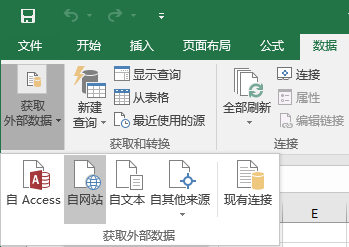
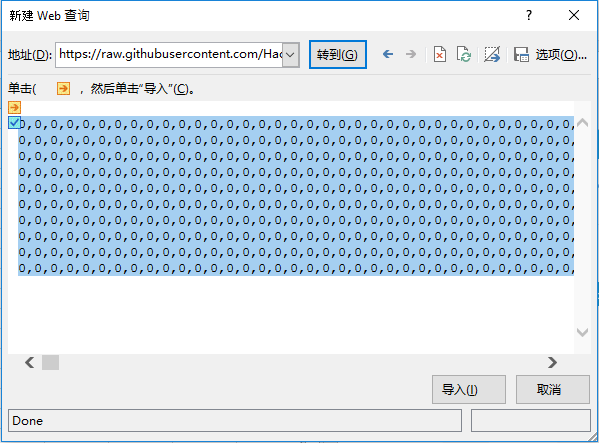
 >>> 

* 1. 为了匹配数据格式，可点击“Use sample data”显示示例数据作为参考

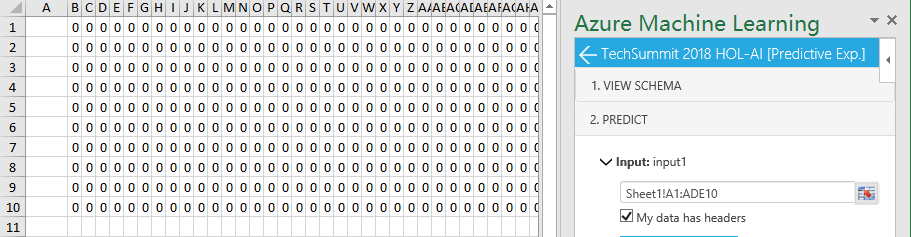


* 1. 删除示例数据，选中B1单元格，然后点击“数据“页，选择”获取外部数据“-”自网站“，在地址栏输入以下地址：

<https://raw.githubusercontent.com/HaoHoo/HOL-AI/master/data/HOL_MNIST_test.csv>

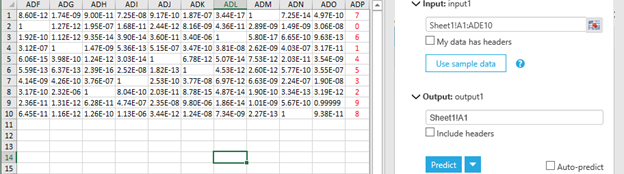
* 1. 点击“数据“页中的”分列“，选择逗号作为分隔符，导入数据完成后，应如下图所示：



请思考为什么选择在B1单元格导入数据，空出A列？

* 1. 在“Input”处输入 ***Sheet1!A1:ADE10*** 或手动选择该单元格区域。在“Output”处输入 ***Sheet1:A1*** ，清除“My data has headers”和“Include headers”选择框，然后点击“Predict”开始识别测试数据集。

如果没有错误，在数据的最后11列应看到类似如下数据：



前10列为对应图片为0~9数字的可能性，第11列为机器学习识别出的数字。

* 1. 回到练习 1中任务 2的notebook，在新代码块中输入如下代码：

|  |
| --- |
| import numpy as np  import matplotlib.pyplot as plt  arrdat=np.loadtxt("https://raw.githubusercontent.com/HaoHoo/HOL-AI/master/data/HOL\_MNIST\_test.csv", 'i2', delimiter=",")  imgdat=arrdat.reshape(10,28,28)  fig, axs = plt.subplots(2, 5)  fig.subplots\_adjust(left=0.06, right=0.75, top=0.43, bottom=0.06, hspace=0, wspace=0.07)  images = []  k=0  for i in range(2):  for j in range(5):  images.append(axs[i,j].imshow(imgdat[k,]))  axs[i,j].set\_xticklabels('')  axs[i,j].set\_yticklabels('')  axs[i,j].set\_xticks([0,13,27])  axs[i,j].set\_yticks([0,13,27])  k=k+1  axs[1,0].set\_xticklabels((0,13,27))  axs[1,0].set\_yticklabels((27,13,0))  plt.show() |

如代码正常执行，应能看到显示测试数据的图片如下：

