Microsoft

AI Development Fundamentals

使用机器学习
预测火箭发射是否延迟

Predict Rocket Launch Delays with Machine Learning

Dec 2020

Microsoft Reactor | Ryan Chung

Microsoft

Ryan Chung

Instructor / DevelopIntelligence
Founder / MobileDev.TW

@ryanchung403 on WeChat
Ryan@MobileDev.TW

Microsoft

Reactor

developer.microsoft.com/reactor/

@MSFTReactor on Twitter

3100 XP

通过机器学习预测火箭发射延迟

1 小时 12 分钟 剩余 • 学习路径 • 已完成 0 个模块，共 3 个

初级 学生 Visual Studio Code

此学习路径介绍了机器学习的世界。 你将应对 NASA 面临的实际问题并应用机器学习来解决它。 学习目标是调动学生的积极性和好奇心，使其去发现机器学习可如何帮助解决太空探索及生活其他方面的其他问题。

先决条件

- Python 在太空探索中扮演的角色学习路径
- 已安装 Visual Studio Code 和 Python
- 能够以 Python 编写简单的程序

继续 > 书签 添加到集合

https://docs.microsoft.com/zh-cn/learn/paths/machine-learning-predict-launch-delay-nasa/

Microsoft Reactor

学习目标

- 天气对火箭发射时会造成的影响
- 数据科学生命周期
- 机器学习运作流程
- 机器学习 VS. 道德

Microsoft Reactor

学习路径

火箭发射简介

4 分钟 剩余 • 模块 • 已完成 9 个单元，共 10 个

★★★★★ 4.8 (61)

了解 NASA 如何选择火箭发射日期，并了解一些机器学习基础知识。

数据收集和操作

33 分钟 剩余 • 模块 • 已完成 0 个单元，共 8 个

★★★★★ 4.9 (35)

学习相关步骤，了解如何将数据导入 Python 并清理数据用于创建机器学习模型。

生成机器学习模型

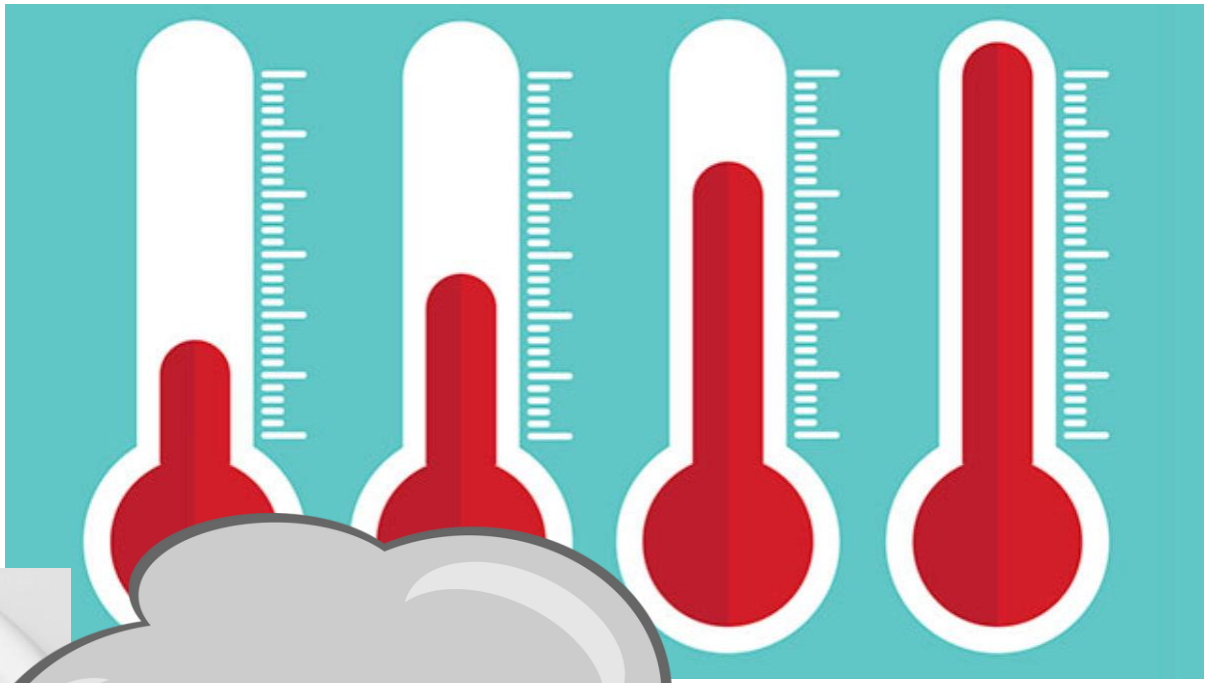
35 分钟 剩余 • 模块 • 已完成 0 个单元，共 10 个

★★★★½ 4.6 (37)

在本模块中，你将重点学习如何使用 scikit-learn 对数据进行本地分析，以及如何使用决策树分类器来掌握有关阴冷天气和火箭发射数据的知识。

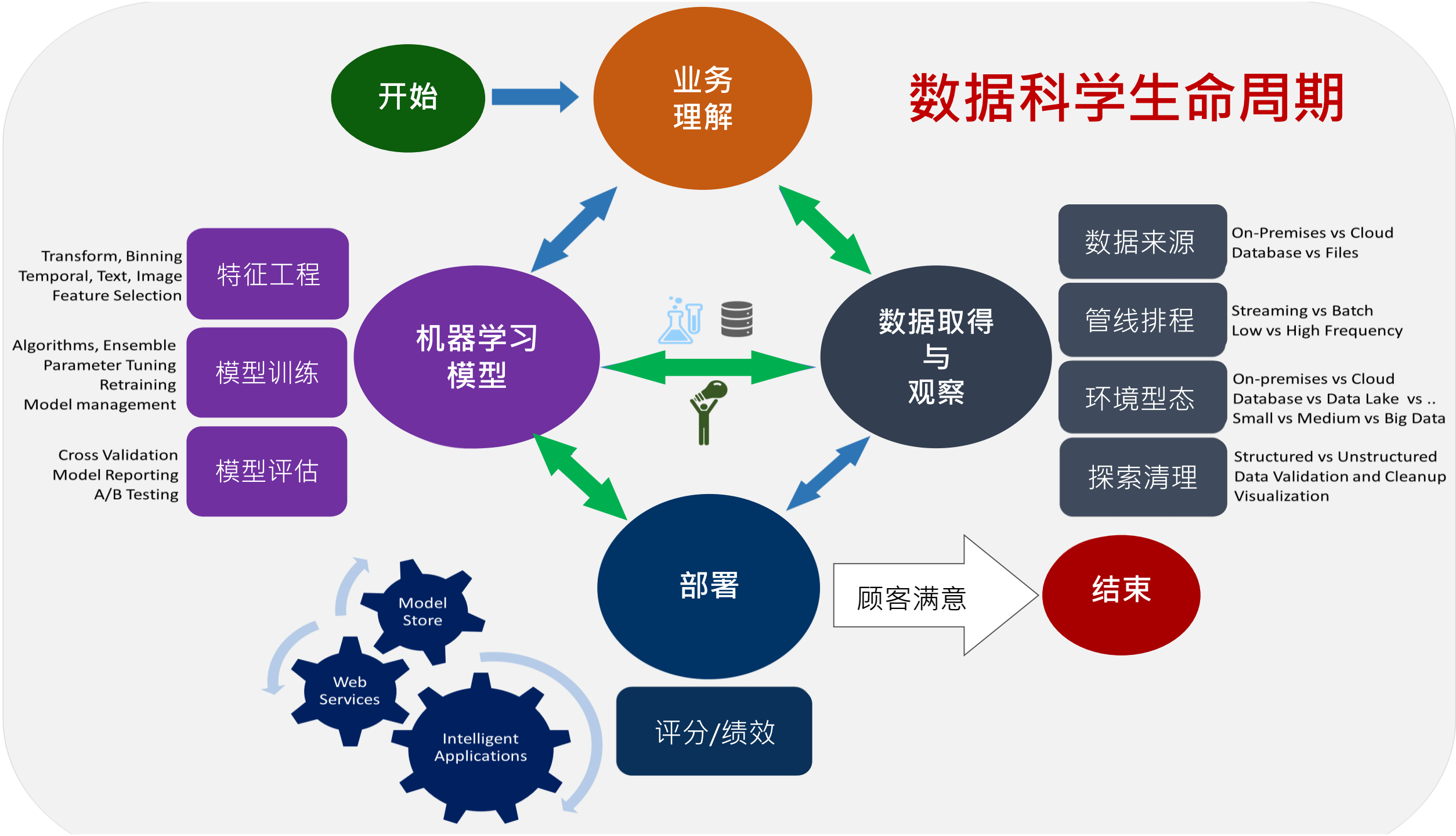
Microsoft Reactor

火箭发射需要考量的因素

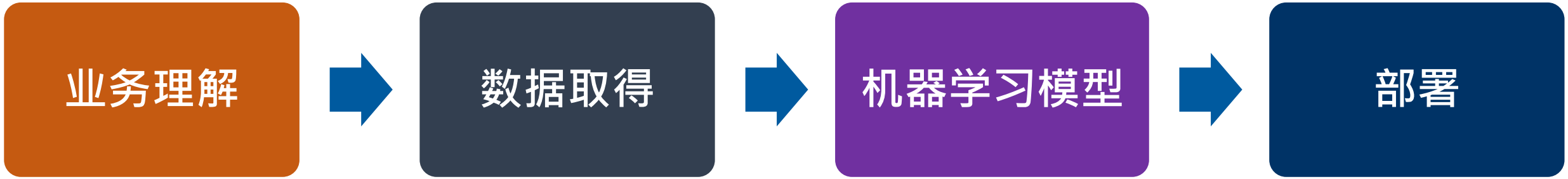


Microsoft Reactor

数据科学生命周期

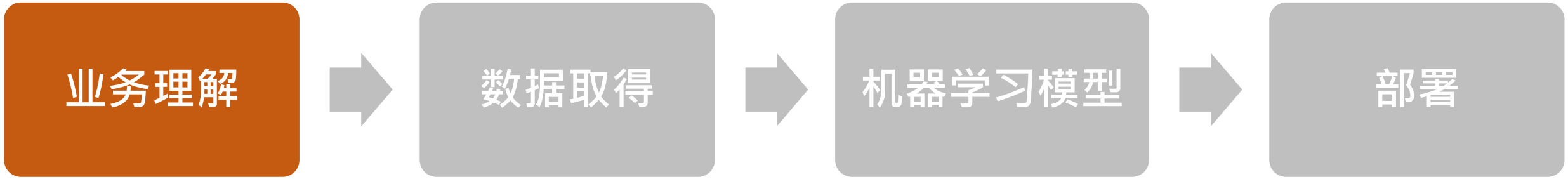

开始
业务理解
机器学习模型
数据取得与观察
部署
结束
顾客满意
特征工程
Transform, Binning
Temporal, Text, Image
Feature Selection
模型训练
Algorithms, Ensemble
Parameter Tuning
Retraining
Model management
模型评估
Cross Validation
Model Reporting
A/B Testing
数据来源
On-Premises vs Cloud
Database vs Files
管线排程
Streaming vs Batch
Low vs High Frequency
环境型态
On-premises vs Cloud
Database vs Data Lake vs ..
Small vs Medium vs Big Data
探索清理
Structured vs Unstructured
Data Validation and Cleanup
Visualization
Model Store
Web Services
Intelligent Applications
评分/绩效

数据科学



Microsoft Reactor

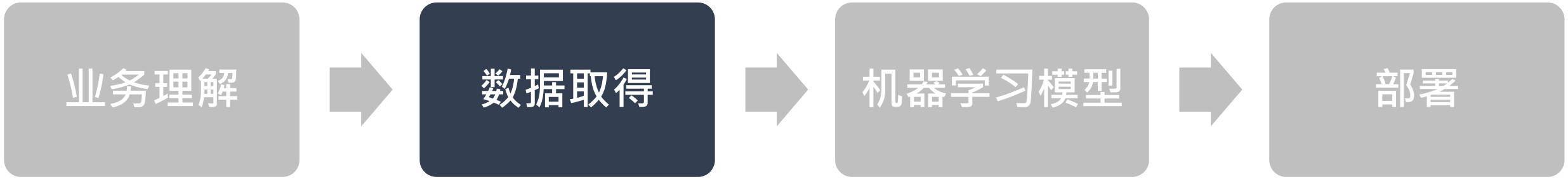
数据科学



- 提高预期火箭发射日是天候良好的可能性
- 了解哪些状况必须一定要停止发射

Microsoft Reactor

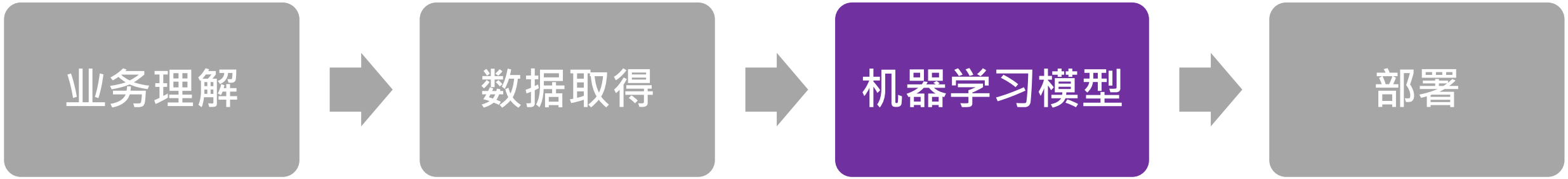
数据科学



- 困难点：实验成本高，很难不断尝试
- 若只有成功发射的案例，识别不良状况效果会较差

Microsoft Reactor

数据科学



- 数据中的那些特征是最关键的影响因素?
- 火箭发射
  - 温度
  - 降水量
  - 湿度
- 目标
  - 使用过去的发射/天气资料，判断未来发射是否可能成功

Microsoft Reactor

知识检查

1. 在数据分析方面，计算机的准确性是否始终比人类更高？ 为什么是或为什么不是？

- ○ 是的，因为计算机是纯逻辑的。 它们不会产生错误。
- ○ 是的，因为计算机能够处理的数据量比人类能够处理的要大得多，而且速度更快。
- ○ 不是，因为计算机无法理解上下文，因此不可能总是准确的。
- ○ 不是. 因为电脑的准确度会取决于软件的准确度，而软件的准确度则取决于加以建置的人类。

2. 数据科学生命周期中的四个步骤是什么？

- ○ 业务理解、数据收集和准备、模型训练和测试、模型部署
- ○ 数据收集、数据验证、机器学习、结果可视化
- ○ 数据标识、模型训练到 100% 的准确性、模型部署、业务理解
- ○ 机器学习算法编码、数据浏览、算法修改、报告生成

Microsoft Reactor

数据观察

- 状况 (多云、局部多云、晴朗、降雨、雷电、暴风雨)
- 温度
- 湿度
- 风速
- 风向
- 降雨
- 可见度
- 海平面
- 气压

Microsoft Reactor

模块汇入

```
# Pandas library is used for handling tabular data
import pandas as pd
# NumPy is used for handling numerical series operations
import numpy as np
# Sklearn library contains all the machine learning packages we need
from sklearn import linear_model, model_selection, metrics
from sklearn.model_selection import train_test_split

# Machine learning libraries used to build a decision tree
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

# Sklearn's preprocessing library is used for processing and cleaning the data
from sklearn import preprocessing

# for visualizing the tree
import pydotplus
from IPython.display import Image
```


Microsoft Reactor

数据汇入与观察

```
launch_data = pd.read_excel('RocketLaunchDataCompleted.xlsx')
launch_data.head()
launch_data.columns
```

	Name	Date	Time (East Coast)	Location	Crewed or Uncrewed	Launched?	High Temp	Low Temp	Ave Temp	Temp at Launch Time	...	Max Wind Speed	Visibility	Wind Speed at Launch Time	Hist Ave Max Wind Speed	Hist Ave Visibility	Sea Level Pressure	Hist Ave Sea Level Pressure	Day Length	Condition	Notes
0	NaN	1958-12-04	NaN	Cape Canaveral	NaN	NaN	75.0	68.0	71.00	NaN	...	16.0	15.0	NaN	NaN	NaN	30.22	NaN	10:26:00	Cloudy	NaN
1	NaN	1958-12-05	NaN	Cape Canaveral	NaN	NaN	78.0	70.0	73.39	NaN	...	14.0	10.0	NaN	NaN	NaN	30.2	NaN	10:26:00	Cloudy	NaN
2	Pioneer 3	1958-12-06	01:45:00	Cape Canaveral	Uncrewed	Y	73.0	0.0	60.21	62.0	...	15.0	10.0	11.0	NaN	NaN	30.25	NaN	10:25:00	Cloudy	NaN
3	NaN	1958-12-07	NaN	Cape Canaveral	NaN	NaN	76.0	57.0	66.04	NaN	...	10.0	10.0	NaN	NaN	NaN	30.28	NaN	10:25:00	Partly Cloudy	NaN
4	NaN	1958-12-08	NaN	Cape Canaveral	NaN	NaN	79.0	60.0	70.52	NaN	...	12.0	10.0	NaN	NaN	NaN	30.23	NaN	12:24:00	Partly Cloudy	NaN

5 rows × 26 columns

```
Index(['Name', 'Date', 'Time (East Coast)', 'Location', 'Crewed or Uncrewed',
       'Launched?', 'High Temp', 'Low Temp', 'Ave Temp', 'Temp at Launch Time',
       'Hist High Temp', 'Hist Low Temp', 'Hist Ave Temp',
       'Percipitation at Launch Time', 'Hist Ave Percipitation',
       'Wind Direction', 'Max Wind Speed', 'Visibility',
       'Wind Speed at Launch Time', 'Hist Ave Max Wind Speed',
       'Hist Ave Visibility', 'Sea Level Pressure',
       'Hist Ave Sea Level Pressure', 'Day Length', 'Condition', 'Notes'],
      dtype='object')
```


Microsoft Reactor

数据汇入与观察

launch_data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 26 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Name                            60 non-null     object
 1   Date                            300 non-null    datetime64[ns]
 2   Time (East Coast)               59 non-null     object
 3   Location                        300 non-null    object
 4   Crewed or Uncrewed              60 non-null     object
 5   Launched?                       60 non-null     object
 6   High Temp                       299 non-null    float64
 7   Low Temp                        299 non-null    float64
 8   Ave Temp                        299 non-null    float64
 9   Temp at Launch Time             59 non-null     float64
 10  Hist High Temp                  299 non-null    float64
 11  Hist Low Temp                   299 non-null    float64
 12  Hist Ave Temp                   299 non-null    float64
 13  Percipitation at Launch Time    299 non-null    float64
 14  Hist Ave Percipitation          299 non-null    float64
 15  Wind Direction                  299 non-null    object
 16  Max Wind Speed                  299 non-null    float64
 17  Visibility                      299 non-null    float64
 18  Wind Speed at Launch Time       59 non-null     float64
 19  Hist Ave Max Wind Speed         0 non-null      float64
 20  Hist Ave Visibility             0 non-null      float64
 21  Sea Level Pressure              299 non-null    object
 22  Hist Ave Sea Level Pressure     0 non-null      float64
 23  Day Length                      298 non-null    object
 24  Condition                       298 non-null    object
 25  Notes                           3 non-null      object
dtypes: datetime64[ns](1), float64(15), object(10)
memory usage: 61.1+ KB
```


Microsoft Reactor

遗漏值处理

```
launch_data.isnull().sum()
```

```
Name                                240
Date                                  0
Time (East Coast)                   241
Location                              0
Crewed or Uncrewed                  240
Launched?                           240
High Temp                             1
Low Temp                              1
Ave Temp                              1
Temp at Launch Time                 241
Hist High Temp                        1
Hist Low Temp                         1
Hist Ave Temp                         1
Percipitation at Launch Time          1
Hist Ave Percipitation                1
Wind Direction                        1
Max Wind Speed                        1
Visibility                            1
Wind Speed at Launch Time           241
Hist Ave Max Wind Speed             300
Hist Ave Visibility                 300
Sea Level Pressure                    1
Hist Ave Sea Level Pressure         300
Day Length                            2
Condition                             2
Notes                               297
dtype: int64
```


Microsoft Reactor

遗漏值处理

- 没有发射资料的也是当天没发射，补上N

```
launch_data['Launched?'].value_counts()
launch_data['Launched?'].fillna('N',inplace=True)
launch_data['Launched?'].value_counts()
```

```
launch_data['Launched?'].value_counts()
```

```
Y    59
N     1
Name: Launched?, dtype: int64
```

填补前

```
launch_data['Launched?'].value_counts()
```

```
N    241
Y     59
Name: Launched?, dtype: int64
```

填补后

Microsoft Reactor

遗漏值处理

- 有太空人的任务比较少，所以都补上没有太空人的

```
launch_data['Crewed or Uncrewed'].value_counts()
launch_data['Crewed or Uncrewed'].fillna('Uncrewed', inplace=True)
launch_data['Crewed or Uncrewed'].value_counts()
```

```
launch_data['Crewed or Uncrewed'].value_counts()
```

填补前

```
Uncrewed    44
Crewed      16
Name: Crewed or Uncrewed, dtype: int64
```

```
launch_data['Crewed or Uncrewed'].value_counts()
```

填补后

```
Uncrewed    284
Crewed       16
Name: Crewed or Uncrewed, dtype: int64
```


Microsoft Reactor

遗漏值处理

- 天气概况仅两笔无数据，使用一般晴天 "Fair"

```
launch_data['Condition'].value_counts()
launch_data['Condition'].isnull().sum()
launch_data['Condition'].fillna('Fair',inplace=True)
```

```
Cloudy            113
Fair               68
Partly Cloudy      68
Rain               24
T-Storm            12
Thunder             7
Mostly Cloudy       2
Heavy T-Storm       1
Partly Cloudly      1
Windy               1
Light Rain          1
Name: Condition, dtype: int64
```

```
launch_data['Condition'].isnull().sum()
```

```
2
```

填补前

```
Cloudy            113
Fair               70
Partly Cloudy      68
Rain               24
T-Storm            12
Thunder             7
Mostly Cloudy       2
Heavy T-Storm       1
Partly Cloudly      1
Windy               1
Light Rain          1
Name: Condition, dtype: int64
```

```
launch_data['Condition'].isnull().sum()
```

```
0
```

填补后

Microsoft Reactor

遗漏值处理

- 其余数值类的，全部补上 0

```
launch_data.isnull().sum()
launch_data.fillna(0, inplace=True)
```

Name	240
Date	0
Time (East Coast)	241
Location	0
Crewed or Uncrewed	0
Launched?	0
High Temp	1
Low Temp	1
Ave Temp	1
Temp at Launch Time	241
Hist High Temp	1
Hist Low Temp	1
Hist Ave Temp	1
Percipitation at Launch Time	1
Hist Ave Percipitation	1
Wind Direction	0
Max Wind Speed	1
Visibility	1
Wind Speed at Launch Time	241
Hist Ave Max Wind Speed	300
Hist Ave Visibility	300
Sea Level Pressure	1
Hist Ave Sea Level Pressure	300
Day Length	2
Condition	0
Notes	297
dtype: int64	

填补前

Name	0
Date	0
Time (East Coast)	0
Location	0
Crewed or Uncrewed	0
Launched?	0
High Temp	0
Low Temp	0
Ave Temp	0
Temp at Launch Time	0
Hist High Temp	0
Hist Low Temp	0
Hist Ave Temp	0
Percipitation at Launch Time	0
Hist Ave Percipitation	0
Wind Direction	0
Max Wind Speed	0
Visibility	0
Wind Speed at Launch Time	0
Hist Ave Max Wind Speed	0
Hist Ave Visibility	0
Sea Level Pressure	0
Hist Ave Sea Level Pressure	0
Day Length	0
Condition	0
Notes	0
dtype: int64	

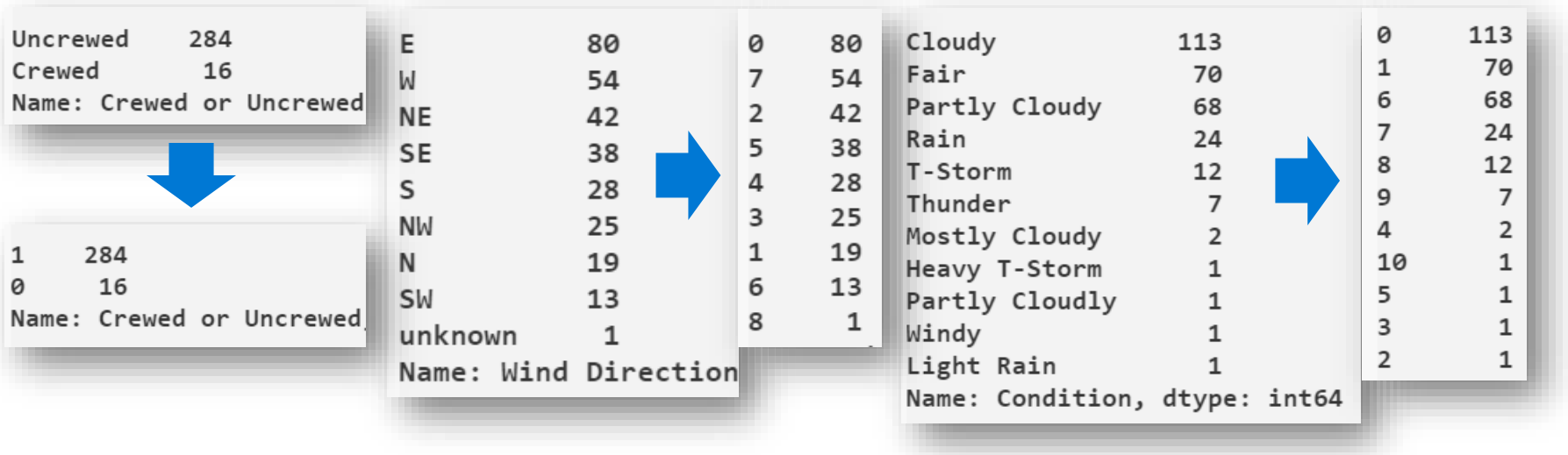
填补后

Microsoft Reactor

数值转换

- 将三个关键但非数值的特征，进行转换

```
launch_data.info()
label_encoder = preprocessing.LabelEncoder()
# Three columns have categorical text info, and we convert them to numbers
launch_data['Crewed or Uncrewed'].value_counts()
launch_data['Wind Direction'].value_counts()
launch_data['Condition'].value_counts()

launch_data['Crewed or Uncrewed'] = label_encoder.fit_transform(launch_data['Crewed or Uncrewed'])
launch_data['Wind Direction'] = label_encoder.fit_transform(launch_data['Wind Direction'])
launch_data['Condition'] = label_encoder.fit_transform(launch_data['Condition'])
```



Microsoft Reactor

数据切割与特征筛选

- Launched? 为预测目标
- 非数值或较不相关的特征可先剔除

```
y = launch_data['Launched?']
# Removing the columns we are not interested in
X = launch_data.drop(['Name','Date','Time (East Coast)','Location','Launched?',
'Hist Ave Sea Level Pressure','Sea Level Pressure','Day Length','Notes','Hist A
ve Visibility', 'Hist Ave Max Wind Speed'],axis=1)
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 15 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Crewed or Uncrewed            300 non-null    int64
 1   High Temp                     300 non-null    float64
 2   Low Temp                      300 non-null    float64
 3   Ave Temp                      300 non-null    float64
 4   Temp at Launch Time           300 non-null    float64
 5   Hist High Temp                300 non-null    float64
 6   Hist Low Temp                 300 non-null    float64
 7   Hist Ave Temp                 300 non-null    float64
 8   Percipitation at Launch Time  300 non-null    float64
 9   Hist Ave Percipitation        300 non-null    float64
 10  Wind Direction                300 non-null    int64
 11  Max Wind Speed                300 non-null    float64
 12  Visibility                    300 non-null    float64
 13  Wind Speed at Launch Time     300 non-null    float64
 14  Condition                     300 non-null    int64
dtypes: float64(12), int64(3)
memory usage: 35.3 KB
```


Microsoft Reactor

模型选择与数据切割

- 使用DecisionTreeClassifier
- 并依80/20比例进行训练/测试数据切割
- 进行训练

```
tree_model = DecisionTreeClassifier(random_state=0, max_depth=5)

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=99)

tree_model.fit(X_train, y_train)
```


Microsoft Reactor

知识检查

1. 为何需要清理数据?

- ○ 便于让其他人更轻松地读取数据
- ○ 数据不一致会扰乱计算机的操作
- ○ 便于使用数据创建更好的可视化效果

2. 以下哪一项不是我们要使用的库?

- ○ PyTorch
- ○ Pandas
- ○ Sklearn
- ○ NumPy

Microsoft Reactor

模型使用与成效评估

- 使用模型进行预测
- 评估成效

```
y_pred = tree_model.predict(X_test)
y_pred

tree_model.score(X_test, y_test)
# is equal to
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

```
tree_model.score(X_test, y_test)

0.9833333333333333
```

```
[55] accuracy_score(y_test, y_pred)

0.9833333333333333
```

```
y_pred

array(['N', 'N', 'N', 'Y', 'N', 'N', 'N', 'N', 'Y', 'Y', 'N', 'Y', 'N',
       'N', 'N', 'N', 'N', 'N', 'N', 'N', 'N', 'Y', 'Y', 'N', 'N',
       'N', 'N', 'N', 'N', 'N', 'Y', 'N', 'N', 'N', 'N', 'N', 'N', 'N',
       'N', 'N', 'N', 'Y', 'N', 'N', 'N', 'Y', 'N', 'N', 'N', 'N', 'N',
       'N', 'N', 'N', 'N', 'N', 'N', 'N', 'Y'], dtype=object)
```


Microsoft Reactor

数据视觉化

- Graphiz 安装
  - pip install graphviz
  - 下载 https://graphviz.org/download/
  - 设定系统路径，让graphviz在任意路径均可执行
- 参数准备

Microsoft Reactor

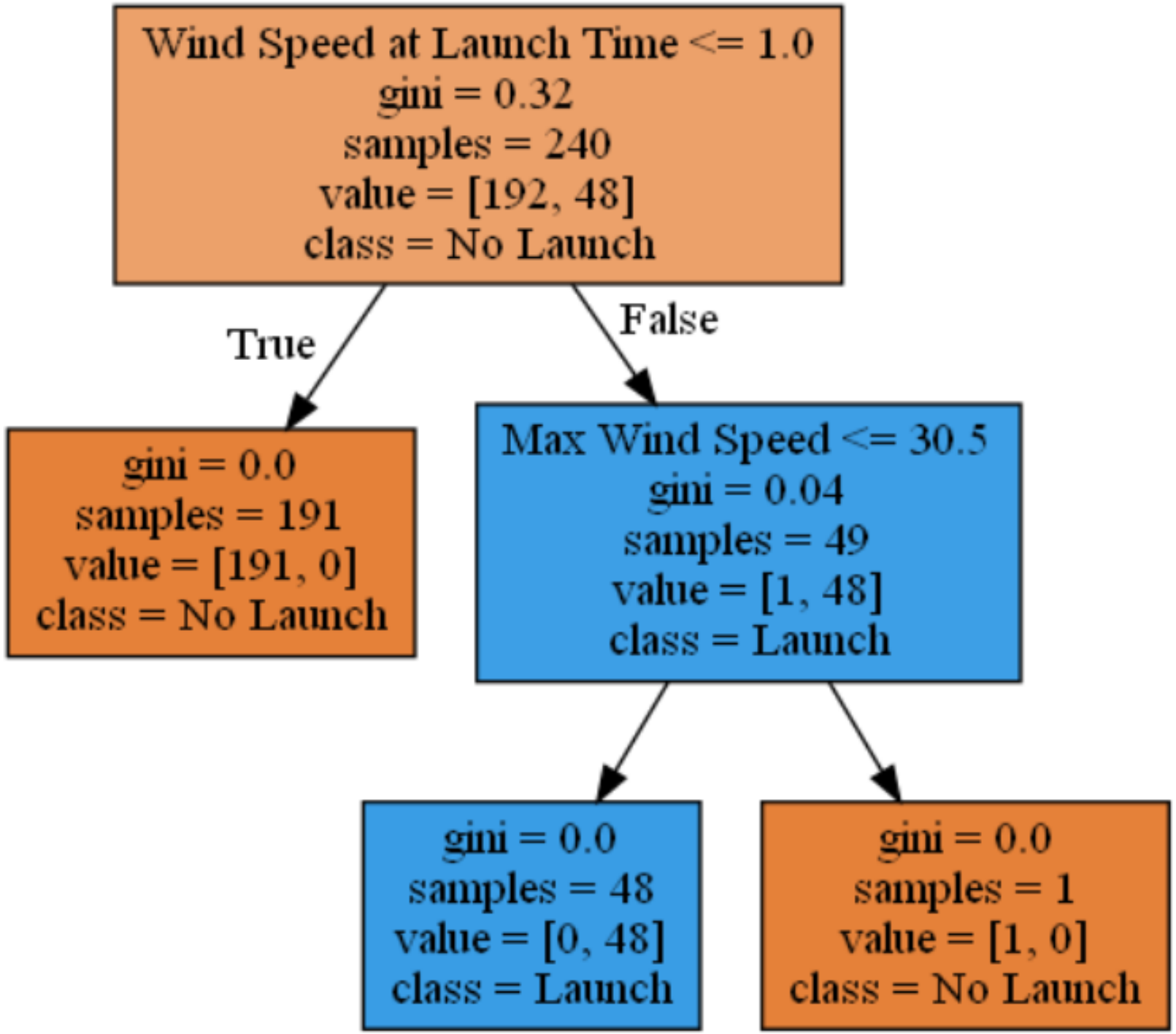
数据视觉化

```
# Let's import a library for visualizing our decision tree.
from sklearn.tree import export_graphviz

def tree_graph_to_png(tree, feature_names, class_names):
    tree_str = export_graphviz(tree, feature_names=feature_names, class_names=class_names,
filled=True, out_file=None)
    graph = pydotplus.graph_from_dot_data(tree_str)
    return Image(graph.create_png())

tree_graph_to_png(tree=tree_model, feature_names=X.columns.values, class_names=['No Launch'
,'Launch'])
```

filled ：是否将主要分类节点着色

Microsoft Reactor

数据视觉化



Microsoft Reactor

测试

```
# ['Crewed or Uncrewed', 'High Temp', 'Low Temp', 'Ave Temp',
#        'Temp at Launch Time', 'Hist High Temp', 'Hist Low Temp',
#        'Hist Ave Temp', 'Precipitation at Launch Time',
#        'Hist Ave Precipitation', 'Wind Direction', 'Max Wind Speed',
#        'Visibility', 'Wind Speed at Launch Time', 'Condition']

data_input = [ 1.  , 75.  , 68.  , 71.  ,  0.  , 75.  , 55.  , 65.  ,  0.  , 0.08,
  0.  , 16.  , 15.  ,  0.  ,  0. ]

tree_model.predict([data_input])
```

```
tree_model.predict([data_input])

array(['N'], dtype=object)
```


Microsoft Reactor

其他探索方向

- 发射前的天气考量通常包含哪些
- 发射日期是否有在特定的季节?日期?
- 其他国家的发射火箭资料?
- 遗漏值的填补是否有更好的选择?
- 是否延期的决策重点是什么?
- 飞机延迟是否考量方式也类似?

Microsoft Reactor

知识检查

1. 我们为机器学习算法选择决策树的原因是什么？

- ○ 决策树算法最复杂，却又最准确。
- ○ 决策树很容易直观呈现。 它非常合适，因为该模型只能做出两种选择：是或否。
- ○ 决策树具有许多分支，并且该模型可做出多种选择。

2. 拆分数据集的目的是什么？

- ○ 通过删除错误数据使模型更准确。
- ○ 使用不同数据尝试不同算法。
- ○ 使用不同的数据来定型和测试模型。

Microsoft Reactor

获得奖杯

https://docs.microsoft.com/zh-cn/users/XXXXX/achievements

Microsoft | Docs

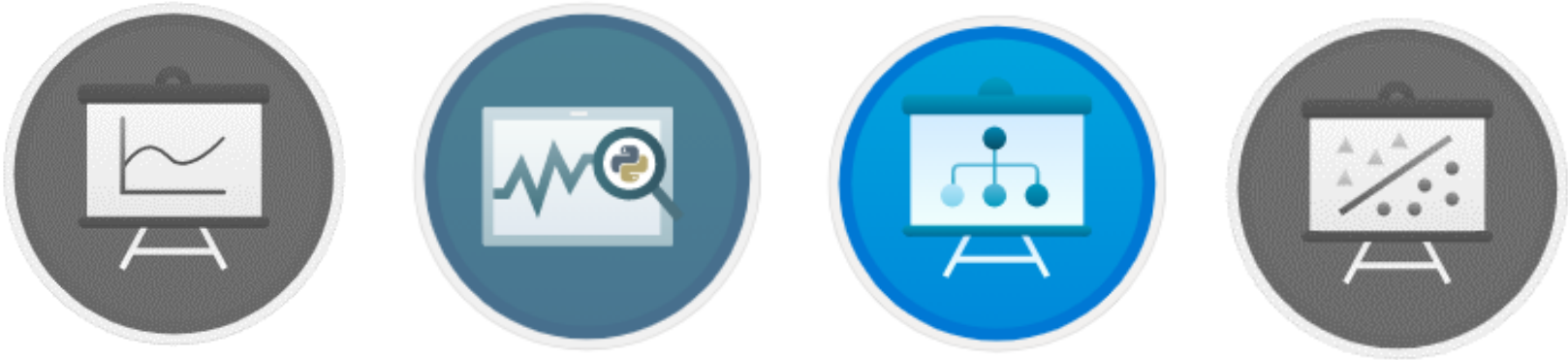
- 概述
- 活动
- 挑战
- 书签
- 集合
- 关注
- **成就**
- 设置

奖 杯

通过机器学习预测火箭发射延迟

小结

- 厘清问题种类，决定使用的机器学习算法
- 剔除没有要用上的特征，也将要用的特征数值化
- 使用决策树来分类，持续兵分两路，产生最后结果



Microsoft Reactor

Microsoft

Reactor

developer.microsoft.com/reactor/

@MSFTReactor on Twitter

Microsoft

议程结束
感谢聆听

请记得填写课程回馈问卷 (Event ID : XXXXX)
https://aka.ms/Reactor/Survey

© 2019 Microsoft Corporation. All rights reserved. The text in this document is available under the Creative Commons Attribution 3.0 License, additional terms may apply. All other content contained in this document (including, without limitation, trademarks, logos, images, etc.) are not included within the Creative Commons license grant. This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. Some examples are for illustration only and are fictitious. No real association is intended or inferred. Microsoft makes no warranties, express or implied, with respect to the information provided here.