Chapter2. End-to-End Machine Learning Project

2.1Look at the Big Picture

2.1.1 Frame the problem

机器学习的首要任务是弄清楚模型要解决的问题

Pipelines: A sequence of data processing component 一系列的数据处理管道

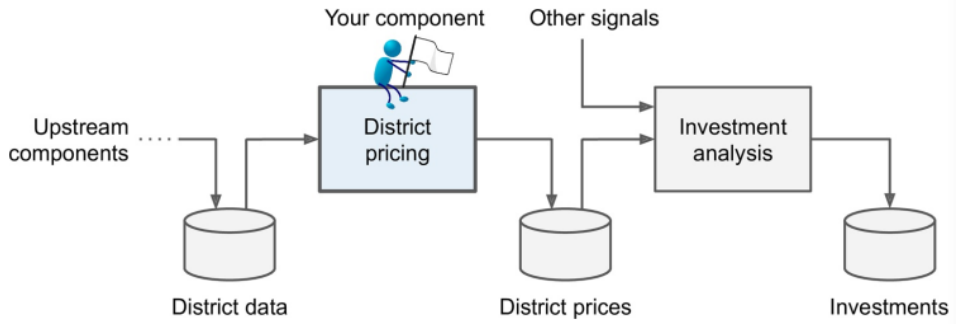


Figure 2-2. A Machine Learning pipeline for real estate investments

每个component之间的由一个data store连接。

优点：

- 每个开发团队可以专注于自己的component。
- 若上游的component损坏，下游的仍可以依靠上游component的最后一次输出继续运行一段时间。提升了架构的robustness。

缺点：

- 如果没有好的component监控程序的话，一些没有注意到的Component故障可能会导致系统性能下降。

第二步是了解当前系统性能如何，以此作为新系统开发的参考。

最后构建问题：supervised/unsupervised? classification/regression? batch learning/online learning? etc.

multiple regression problem: 使用多个features进行训练

univariate regression problem: 输出单一值（房价预测）

multivariate regression problem: 输出多个值

2.1.2 Select a Performance Measure

E.g.

Root Mean Square Error (RMSE) - L2范数:

Equation 2-1. Root Mean Square Error (RMSE)

$$\text{RMSE}(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2}$$

Mean absolute error (MAE) - L1范数:

$$\text{MAE}(\mathbf{X}, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right|$$

Lk范数:

- More generally, the ℓ_k ***norm*** of a vector v containing ***n*** elements is defined as $\|\mathbf{v}\|_k = \left(|v_0|^k + |v_1|^k + \cdots + |v_n|^k\right)^{\frac{1}{k}}$. ℓ_0 gives the number of nonzero elements in the vector, and ℓ_∞ gives the maximum absolute value in the vector.

The higher the norm index, the more it focuses on large values and neglects small ones. This is why the RMSE is more sensitive to outliers than the MAE. But when outliers are exponentially rare (like in a bell-shaped curve), the RMSE performs very well and is generally preferred.

2.1.3 Check the Assumptions

检验假设:

- 如果下游的模型只需要价格的类别（cheap, medium, expensive），则问题变为classification
- 如果需要价格的精确数值，则是regression问题

2.2 Download the Data

- 可以写python函数，直接从链接中下载数据（相比于下载csv数据再处理，可以直接获取最新的数据）

```
housing["ocean_proximity"].value_counts()
```

统计该属性的属性值有哪几种，各种的数量

- 使用pandas.hist()方式直接输出各feature的直方图
- 有些feature值可能会设置了封顶值（比如大于15的一律认为是15），导致模型出现问题。此时有两种方法解决:
    - 1. 抛弃封顶操作，取真实值
    - 2. 舍弃有封顶的sample
    - 选哪种方法取决于模型需要哪种输出，如需要真实值输出，则选1，反之选2
- Creat a Test Set
    - 使用numpy.random.permutation() shuffle index
    - 确保多次运行之后，test set和train set包含的sample是相同的:

        - 1. 设置固定的random seed
        - 2. 第一次运行后保存，再次使用时直接load第一次的数据
        - 这两个方法的缺点是：有新数据进来后，train set和test set包含的数据会改变
    - 为确保train set和test set每次包含的数据都一样：
        - 根据每个insance的unique identifier(e.g. row id, hash value, feature的组合等)确定sample属于train set还是test set。
    - sk-learn的train_test_split()函数，可以同时传入两个data frame，并按照相同的index切割
    - 分层抽样方法：分层抽样法也叫类型抽样法。它是从一个可以分成不同子总体（或称为层）的总体中，按规定的比例从不同层中随机抽取样品（个体）的方法。这种方法的优点是，样本的代表性比较好，抽样误差比较小。缺点是抽样手续较简单随机抽样还要繁杂些。
    - 数据是否需要归一化/标准化
    - 某些属性处于中位数右边的样本数比左边多，这不利于模型学习，可以使用transforming对这些属性进行修改，使得样本分布更类似于beall-shaped。

2.3 Discover and Visualize the Data to Gain Insights

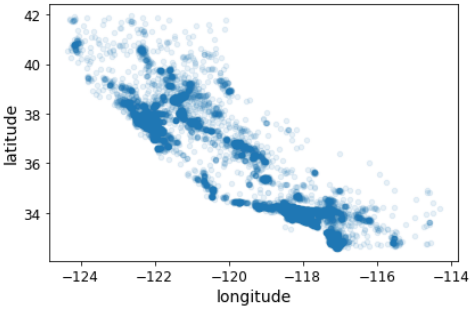
- 一般只对训练集做数据探索分析。当训练集很大时，可以再对训练集取样作探索性分析。当数据集很小时，可以直接对整个数据集进行探索性分析。

2.3.1 Visualizing Geographical Data

- 使用alpha参数使得高密度的散点图区域更明显

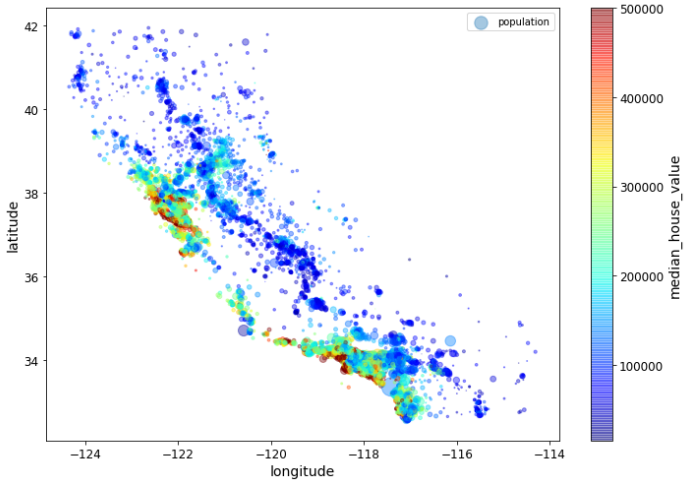
```
housing.plot(kind='scatter', x='longitude', y='latitude', alpha=0.1)
# alpha参数为透明度，当alpha=0.1时，重复的点多的区域会不那么透明

```



- 在散点图上显示多个属性，观察他们之间的correlation。

```
housing.plot(kind="scatter", x="longitude", y="latitude", alpha=0.4,
 s=housing["population"]/100, label="population", figsize=(10,7),
 c="median_house_value", cmap=plt.get_cmap("jet"), colorbar=True,
 sharex=False)
```



- 聚类算法可以找出主要的cluster并且对这些cluster添加新的特征并度量样本和cluster center的接近程度。

2.3.2 Looking for Correlations

- 使用corr()方法，corr_matrix = housing.corr(), corr_matrix['median_house_value']
- 主要属性的单位并不影响属性之间的correlation，即用“米”和“英尺”度量身高时，他们之间的相关系数依然是1。
- pandas中的scatter_matrix()函数可以用来检查两两特征之间的correlation，但只对numerical特征有效。

```
from pandas.plotting import scatter_matrix

attributes = ["median_house_value", "median_income", "total_rooms",
 "housing_median_age"]
scatter_matrix(housing[attributes], figsize=(12, 8))
save_fig("scatter_matrix_plot")
```

2.3.3 Experimenting with Attribute Combinations

- 对特征进行一些组合，再观察新特征与目标变量之间的相关系数。
    - 比如可以构建新特征 每个房屋的房间数=总房间数/总房屋数
- 特征组合是一个迭代的过程，当得到原型模型的时候，可以通过分析模型的输出对数据有更输入的理解，再对数据进行更深入的探索

2.4 Prepare the Data for Machine Learning Algorithms

- pandas中的drop方法（不使用Inplace参数时）会创建data的copy。

2.4.1 Data Cleaning

- 处理缺失值的三种方法：
    - 删除有缺失值的样本
    - 删除该特征
    - 对缺失值进行填充
- 可以使用sklearn中的SimpleInputer进行特征批量填充（使用同一种策略，比如中值、均值等）

2.4.2 Handling Text and Categorical Attributes

- 处理类别型变量一般又ordinal和one-hot两种方法：
    - ordinal encoding针对有大小关系的类别型变量
    - one-hot针对没有大小关系的类别型变量，在sklearn中的OneHotEncoder中，返回的是scipy的sparse matrix类型，而不是numpy的数组。sparse matrix中只保存了非零元素的location，大大节省了空间。但是当类别型变量取值很多时，one-hot encoding表现也很差，有两种方法可以替代：1是将类别型变量转换为数值型。比如在house price predict中，将近海数变为近海距离。2是可以使用embedding方法，即representation learning中的一种。

2.4.3 Custom Transformers

自定义Sklearn中的transformer。

1. 创建一个类，包含fit (returning self)，transform()和fit_transform()三种方法。当类继承TransformMixin时，不用实现fit_transform方法。当继承BaseEstimator类时，自动实现了get_params()和set_params()方法，方便对超参数进行调试。

```
1 from sklearn.base import BaseEstimator, TransformerMixin
```

```
2
3 # column index
4 rooms_ix, bedrooms_ix, population_ix, households_ix = 3, 4, 5, 6
5
6 class CombinedAttributesAdder(BaseEstimator, TransformerMixin):
7  def __init__(self, add_bedrooms_per_room = True): # no *args or **kargs
8  self.add_bedrooms_per_room = add_bedrooms_per_room
9  def fit(self, X, y=None):
10  return self # nothing else to do
11  def transform(self, X):
12  rooms_per_household = X[:, rooms_ix] / X[:, households_ix]
13  population_per_household = X[:, population_ix] / X[:, households_ix]
14  if self.add_bedrooms_per_room:
15  bedrooms_per_room = X[:, bedrooms_ix] / X[:, rooms_ix]
16  return np.c_[X, rooms_per_household, population_per_household,
17  bedrooms_per_room]
18  else:
19  return np.c_[X, rooms_per_household, population_per_household]
20
21 attr_adder = CombinedAttributesAdder(add_bedrooms_per_room=False)
22 housing_extra_attribs = attr_adder.transform(housing.values) #housing.va
lues取出dataframe的值
```

2.4.4 Feature Scaling

- 常用的有min-max scaling (即normalization) 和 standardization。
  - min-max scaling将数据缩放到固定范围之间，一般为0~1.
  - standardization将数据转换为均值为0，标准差为1.范围不固定
  - 神经网络中一般使用min-max scaling，因为它要求输入为0~1.
  - sklearn中提供了StandarScaler实现standardization. Min-Max Scaler实现min-max scaling.
- 只对训练集做scaling.

2.4.5 Transformation Pipelines

- Sklearn中提供了Pipeline实现管道化的操作。Pipeline constructor将对里面的estimator执行顺序操作，除了最后一个estimator不要求是transormer外，其他必须为transformer（即必须有fit_transform()方法）。

- Sklearn中提供了ColumnTransformer处理各种类型的特征，并且对pandas的DataFrame类型数据很友好。ColumnTransformer中的remainder参数可以设置对特征做drop还是passthrough操作。

```
from sklearn.compose import ColumnTransformer

num_attribs = list(housing_num)
cat_attribs = ["ocean_proximity"]

full_pipeline = ColumnTransformer([
 ("num", num_pipeline, num_attribs), # 返回numpy的dense matrix
 ("cat", OneHotEncoder(), cat_attribs), # 返回一个sparse matrix
 # (name, transformer, list of names of columns)
  ])

# ColumnTransformer会根据sparse_threshold (非零元素所占比例)
# 决定返回dense matrix还是
# sparse matrix，小于threshold则返回sparse matrix

housing_prepared = full_pipeline.fit_transform(housing)
```

2.5 Select and Train a Model

2.5.1 Traing and Evaluating on the Training Set

- training error等于0代表可能是过拟合了

2.5.2 Better Evaluation Using Cross-Validation

- 使用sklearn的train_test_split()对training set再进行细分为training set和validation set.
- 使用sklearn的cross_val_score实现cross validation。
    - sklearn中的cross validation的score值越小代表模型效果越好，因此在显示误差的时候要给cross validation的score取反。
- cross validation不仅可以评估模型的表现，还可以度量模型评估的准确性，因为进行了多次的验证。
- 先选效果较好的算法，再对一个算法的超参数进行调优，而不是直接深入地对算法超参数进行调优。
- 每次训练完一个模型后要保存，以便后续对比。

2.6 Fine-Tune Your Model

2.6.1 Grid Search

- 可以实现对所有参数的特定取值进行测试。
- 可以调用sklearn中的GridSearchCV类实现。
- 在当前的参数组合中找到使算法表现最好的超参数，可以测试比该参数组合更大或更小的参数组合进行测试。

2.6.2 Randomized Search

- 当超参数的取值空间较大时，一般使用随机搜索。
- 设置迭代次数为1000时，对每个超参数的1000个不同取值进行搜索和测试。

2.7 Ensemble Methods

- 通常情况下集成学习算法比较好。

2.8 Analyze the Best Models and Their Errors

- 通过对表现最好的模型的输出进行分析，更好地了解问题：
    - 通过查看各个特征的importances
    - 了解误差是如何产生的，才能更好地解决问题
    - 是否需要增加新的特征，或去除一些特征，或消除离群点

2.9 Evaluate Your System on the Test Set

- 调用sklearn中pipeline的transform方法，以测试集数据作为输入（注意不是fit_transform()，不是用测试集去训练模型）
- 使用scipy.stats.t.interval计算置信区间，来衡量目前模型输出结果的可信度。
    - 通过cross validation得到多个误差值，对这些误差值求平均，以该平均数求95%置信区间的上下限值。将此上下限值与原模型的95%置信区间的上下限值作对比，看有多少提升。

2.10 Launch, Monitor, and Maintain Your System

- 可以通过REST API调用模型的predict()方法，使用REST API的话，系统扩展性会跟高。
- 可以在云服务器部署，比如Google Cloud，使用joblib库保存模型并上传到Google Cloud Storage。
- 编写monitoring code，监控模型的实时性能（比如当模型运行失败时发出警告）。
- 有时模型的性能可以通过下游的指标估计。比如推荐系统的性能可以通过商品的销量推断。
- 一些自动化监控模型性能的方法：
    - 定期收集新的数据并重新训练模型。
    - 编写脚本定期对模型的超参数进行更新并重新训练模型。

- 编写脚本使用新的数据训练新的模型，并对新旧模型进行对比。选择性能好的模型重新部署（要明白为何模型质量得到了提高）。

- 确保输入数据的质量：
  - 有时模型效果下降的原因是数据质量下降。有可能是上游数据的问题。一般数据的问题不会马上导致模型效果下降，而是一个长期的过程。因此必须尽早发现数据质量的问题。
  - 比如带缺失值的样本越来越多，类别型变量取值变化，特征值均值或标准差变化太大等。
- 保存模型和数据集以便roll back处理。
  - 有时可以按某些特定的要求再将training set细分，比如按是否近海划分，从而发掘出模型的优点和缺点。