資料前處理：

1. 缺失值填補
2. 離群值處理
3. 標準化

定義目標：

1. 回歸？分類？
2. 預測目標是什麼？
3. 用什麼資料預測？
4. Training set/ Validation set/ Test set

評估標準：

1. 回歸問題：
   1. RMSE, Root Mean Square Error
   2. Mean Absolute Error
   3. R-Square
2. 分類問題：
   1. Accuracy
   2. F1-score
   3. AUC, Area Under Curve

根據設定目標建立機器學習模型：

數據分析流程：

1. 收集數據

---

1. 數據清理
2. 特徵萃取
3. 資料視覺化>>>了解資料, 發現outliers 或異常數值, 分析個變數間的關練性
4. 建立模型
5. 驗證模型

---

1. 決策應用

Supervised Learning

1. 前處理：
   1. 資料讀取
   2. 格式調整
      1. 資料類型
         1. Discrete variable: 整數單位
         2. Continuous variable
         3. 字串/類別轉換：
            1. Label encoding: 有序 (sklearn.preprocessing.LabelEncoder) Text

               Description automatically generated
            2. One hot encoding: 無序 (pd.get\_dummies)
      2. 特徵類型
         1. 數值型特徵
         2. 類別型特徵：一種類別對應一種分數
         3. 時間型特徵：週期性
   3. 填補缺值
      1. Median: np.median(value\_array)
      2. Quantiles: np.quantile(value\_arrar,q = ..)
      3. Mode: scipy.stats.mode(value\_array) or dictionary method
      4. Mean: np.mean(value\_array)
      5. 連續行數值標準化：
         1. Z 轉換
         2. 空間壓縮 (Ｙ＝0-1, -1~1）
         3. Regression model: 有差
         4. Tree-based model: 沒有太大關係
   4. 去離群值
      1. 透過scatter, histogram 檢查
      2. 透過describe() 檢查
      3. Emprical Cumulative Density Plot

A picture containing text

Description automatically generatedChart, line chart

Description automatically generated(009)

* + 1. 可捨棄或是調整 (需分析離群值之原因)
       1. 對數轉換 ln / log
       2. 縮尾 (將離群值換為 較中間的數)
       3. 截尾 （刪除）
       4. 插值 （賦予較合理之數）（隨機,均值,回歸等）＊＊＊
    2. 減少標準化/最小最大化（特徵縮放）之問題
  1. 特徵縮放
     1. Good:
        1. 對權重敏感或損失函數平滑程度有幫助者
        2. 特徵間的量級差異甚大
     2. Bad:
        1. 有些指標不適合在標準化空間進行
        2. 量的單位在某些特徵上有意義

1. Exploratory Data Analysis：  
   plt.style.use(‘default/ggplot/seaborn’)  
   統計值視覺化
   1. 相關係數
      1. Pd\_df.corr()[“Target”]
   2. 核密度函數KDE (Kernel Density Estimation)
      1. 線下面積和為1
      2. 對稱：K(-u) = K(u)
      3. 常用的Kernel function
         1. Gaussian (normal dist)
         2. Cosine
      4. 不同組間的分布差異
   3. 離散化
      1. 減少Outlier 對分析與模型的影響
      2. 等寬pd.cut(df,num\_group)
      3. 等頻pd.qcut(df,num\_group)
      4. 劇類
   4. 繪圖排版:subplot
      1. The index for subplot starts with 1
      2. 圖形必須直覺
      3. EDA讓人看懂資料
   5. 常用圖形
      1. Heatmap
         1. Sns.heatmap()
      2. Gridplot
         1. Sns.pairplot()
   6. 模型體驗
      1. 使用最簡單的模型當作baseline
2. 特徵工程:
   1. 從事實到對應分數的轉換
      1. 用於資料彙整之後,擬合模型之前
      2. 類別編碼
      3. 特徵縮放
   2. 數值型特徵：若離群值比例太高，或平均值不具代表性，執行去偏態
      1. 目標在於讓數值更接近常態分佈，讓平均值更具代表性
      2. 對數去偏：np.log1p()
      3. 方根去偏：減去min 後開根號 （sqrt）
      4. 分佈去偏(Boxcox):   
         from scipy import stats  
         stats.boxcox(df,lambda= λ)
         1. λ介於0~0.5之間
         2. df內之值必須>0
   3. 類別型特徵：
      1. 標籤編碼(Label Encoding)：單一col輸出
         1. 適合非深度學習模型
         2. 有序資料時可使用
         3. Pd.get\_dummies()
         4. Sklearn.preprocessing.OneHotEncoder()
      2. 獨熱編碼(One Hot Encoding)：多col輸出
         1. 適合深度學習模型
         2. 特徵重要性高，且可能值較少時使用
      3. 若特徵值與目標值有顯著相關：
         1. 均值編碼：容易overfitting，可參考出現頻率
         2. 平滑化：
         3. 計數編碼 (Counting)：類別筆數與目標值有高度相關，筆數可作為特徵
         4. 特徵雜湊 (Feature Hash)：鄉議類別的數量非常龐大時
   4. 時間型特徵：
      1. 時間特徵分解
         1. 年月日時分秒 週 星期
      2. 週期循環特徵：頭尾相接，需注意最高最低點的設置
         1. 週期：季, 月,日,年

Text, letter

Description automatically generated

* + 1. 時段特徵：短暫時段內的事件計數
       1. E.g.點擊網站前一天的累計點擊量等
  1. 特徵組合：機器學習關鍵在特徵工程，特徵工程關鍵在領域知識 （提升預測力）
     1. 合成特徵：由多個col產生
     2. 組合獨熱矢量：
        1. E.g. [behavior type X time of day] >>>狗吼叫 ＋時段
     3. 數值型：加減乘除
     4. 群聚編碼：類別與數值 （不易overfitting）
        1. 類別＋mean/min/max/count/etc e.g. 第三責任險 前一年保費平均為2500 共三筆
        2. Vs. 均值編碼Graphical user interface

           Description automatically generated
  2. 特徵篩選：
     1. 用領域知識挑選，並以特徵重要性選擇
     2. 寧濫勿缺
  3. 特徵評估：
     1. 特徵重要性Featrure Importance (樹狀模型)
        1. 決定分支的次數weight： (sklearn 僅有此方法)
        2. 特徵覆蓋度cover：覆蓋結果數量
        3. 損失函數降低量gain：loss function決定
     2. 排列重要性Permutation Importance
        1. 打散原始資料中單一特徵的排序，並套用於已訓練的模型，檢查預測準確率
  4. 葉編碼 ＋ 邏輯斯回歸
     1. Receiver Operating Characteristic Curve
        1. 越左上越好
        2. 可翻轉結果

1. 模型選擇
   1. 概論：
      1. 驗證基礎
         1. 資料切分Sklearn.model\_selection.train\_test\_split()
         2. K-fold Cross-validation: sklearn.model\_selection.KFold()
         3. sklearn.model\_selection.cross\_val\_score: cross\_val\_score(estimator, train\_X, train\_Y, cv=5).mean()
      2. 預測類型
         1. 回歸
            1. 可轉化為分類問題
         2. 分類
            1. 二分類Binary-class
            2. 多元分類Multi-class：一個樣本只能有一個分類
            3. 多標籤Multi-label：一個樣本可有多個分類
      3. 評估指標
         1. Accuracy:正確分類樣本數/總樣本數
         2. 回歸：
            1. Mean Absolute Error (MAE): [0, ∞]
            2. Mean Square Error (MSE): [0, ∞]
            3. R-square: [0,1]
         3. 分類：
            1. Area Under Curve (AUC): [0,1] 機率超過閥值判定為1
            2. F1-Score (Precision, Recall): [0,1]   
               2\*(Precision \* Recall) / (Precision + Recall)

Precision （精度）模型判定瑕疵,樣本確實為瑕疵的比例

Recall （召回率）模型判定瑕疵, 佔樣本所有瑕疵比例

* + - 1. 混淆矩陣 Confusion Matrix  
         Table

         Description automatically generated
  1. 基礎模型
     1. 線性回歸 Linear Regression：sklearn.linear\_model.LinearRegression
        1. 通常作為baseline參考
        2. 注意貢獻性與標準化限制
     2. 邏輯斯回歸Logistic Regression:sklearn.linear\_model.LogisticRegression
        1. 預測機率值
        2. Penalty:
           1. L1：模型較為稀疏，較快
           2. L2
        3. C: 正則化強度，數字越小，模型越簡單
        4. Solver:
           1. Newton-cg: L2 only / multiclass problems
           2. Lbfgs: L2 only / multiclass problems
           3. Liblinear:二分類問題或少量數據
           4. Sag: multiclass problems / fast for large one
           5. Saga: multiclass problems / fast for large one
        5. Multi-class:
           1. One-vs-rest: OvR, 都可
           2. Many-vs-many: MvM, 只能搭配newton-cg, sag
     3. 套索算法LASSO：Linear Regression + L1，可作feature selecting之用
        1. Sklearn.linear\_mode.Lasso
        2. Alpha:正則化強度
        3. .coef\_ : 印出訓練後的模型參數
     4. 嶺回歸 Ridge Regression：Linear Regression + L2，可解決多重共線性問題
        1. Sklearn.linear\_mode.Ridge
        2. Alpha:正則化強度
        3. .coef\_ : 印出訓練後的模型參數
  2. 樹狀模型
     1. 決策樹 Decision Tree: sklearn.tree.DecisionTreeClassifier/.DecisionTreeRegressor
        1. 使Information Gain 最大化
        2. 衡量資料相似度：
           1. 吉尼係數Gini-index
           2. 熵entropy
        3. .Feature\_importances\_:得知相對有用之feature為何
        4. 回歸決策數：返回range值e.g. 7, x<3 10, x>=3
        5. 缺點：容易overfitting
     2. 隨機森林 [sklearn.ensemble](https://scikit-learn.org/stable/modules/classes.html#module-sklearn.ensemble).RandomForestClassifier/.RandomForestRegressor
        1. 每棵樹隨機使用一部份訓練資料與特徵，每棵樹皆為獨立互不影響
        2. 多棵樹透過投票或加權得到最終結果
        3. 採用Bagging, 應用Bootstrapping思維 抽出又放回
        4. 缺點：訓練樣本小或特徵少時，效果不佳
        5. 特徵重要性會保留相同的特徵
     3. 梯度提升機 Gradient Boosting Machine

ensemble.GradientBoostingRegressor / .GradientBoostingClassifier

* + - 1. 利用後面生成的樹修正前面的樹，所以每棵樹皆有關聯
      2. 採用序列additive的方式產生
      3. 計算Gradient
      4. Y=score1\*learning\_rate+score2\*learning\_rae….
      5. 為回歸樹，調整後可做分類
      6. 特徵重要性可能會忽視相同的特徵

1. 參數調整
   1. 網格搜尋 Grid Search：
      1. 窮舉組合範圍
      2. 利用validation set 做最佳選擇參考
   2. 隨機搜尋 Random Search
      1. 用均勻分佈進行參數抽樣
      2. 通常結果較佳
   3. 先使用預設，再慢慢調整
   4. 超參數影響結果，但效果有限，data cleaning and feature engineering 更有效
   5. 分割train set 來做cross validation 選擇最佳參數組合確保泛化性，最後再用test 評估結果
2. 集成 Ensemble : 結合多個/多種分類器, 作為綜合預測。合議制/多數決
   1. 資料面集成：（不同資料，相同模型，合成結果）
      1. 裝袋法bagging：選出後放回, 如隨機森林
      2. 提升法 boosting ：由之前模型的預測結果，改變資料被抽到的權重或目標值。
         1. 自適應提升（AdaBoost, Adaptive Boosting）: 錯判資料被抽中機率放大，正確的縮小
         2. 梯度提升機 (Gradient Boosting Machine) ：依照殘差項調整新目標值
   2. 模型與特徵的集成：（相同資料，不同模型，合成結果）
      1. 混合泛化 Blending：前提是 單模效果好（有調參數）且模型差異大
      2. 投票泛化 Voting ：給予不同模型的預測值加權合成
      3. 堆疊泛化 Stacking ：用預測結果當新特徵。把模型當作下一階段的特徵編碼器使用

EDA – 統計量化的方式

1. 集中趨勢：
   1. 平均值 (mean)
   2. 中位數 (median)
   3. 眾數 (mode)
2. 資料分散程度：
   1. Min
   2. Max
   3. Range
   4. Quartiles
   5. Variance
   6. Standard Deviation
3. pd.hist(bin=n)  
   Chart

   Description automatically generated
4. pd. value\_counts().plot(kind="bar")

Chart, waterfall chart

Description automatically generated

模型選擇流程：

1. 定義模型
2. 評估模型好壞
   1. 由目標函數objective function/損失函數loss function 來衡量
   2. 線性回歸：MSE (Mean Square error 均方差)越小越好
3. 找出最佳模型參數
   1. 梯度下Gradient Descent/增量訓練Additive Training
   2. Overfitting : 學習到噪音，反而降低實際預測效果
4. Overfitting:
   1. 增加資料量
   2. 降低模型複雜度
   3. 使用正則化(Regularization)：函數=損失函數＋正則化
5. Underfitting:
   1. 增加複雜度
   2. 減輕或不使用正規化