資料科學導論 HW2

撰寫者:
統計系 111 級黃科銓
統計系 111 級葉嘉淀
統計系 111 級高涵毅

1. Introduction

在此次競賽當中，我們需要對七筆不同資料進行預測，並且所有特徵經過去識別化，資料也經過處理產生部分遺失值需要做填補，過程中可以使用任何的套件，但需提出單一方法同時適用於所有資料集，最終競賽成果則依照預測的準確度做排名。

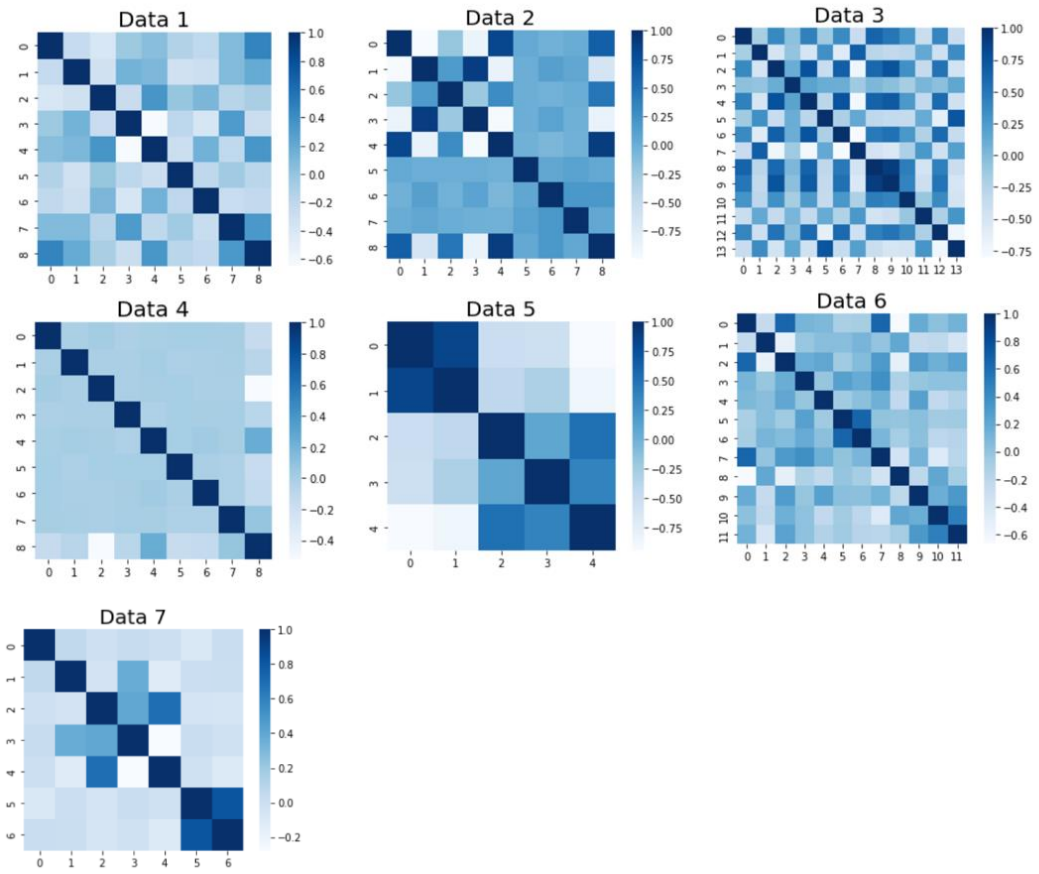
表一：訓練集中各個特徵遺失值數量

# of nan	0	1	2	3	4	5	6	7	8	9	10	11	12
Data 1	234	220	218	225	219	239	249	237					
Data 2	167	161	195	179	151	163	167	173					
Data 3	110	124	103	103	105	118	124	122	118	105	104	110	102
Data 4	1755	1864	1834	1811	1764	1789	1850	1893					
Data 5	2132	2149	2089	2114									
Data 6	367	395	362	357	351	327	342	367	346	366	321		
Data 7	68	62	74	63	78	68							

表二：訓練集中各個特徵不重複值數量

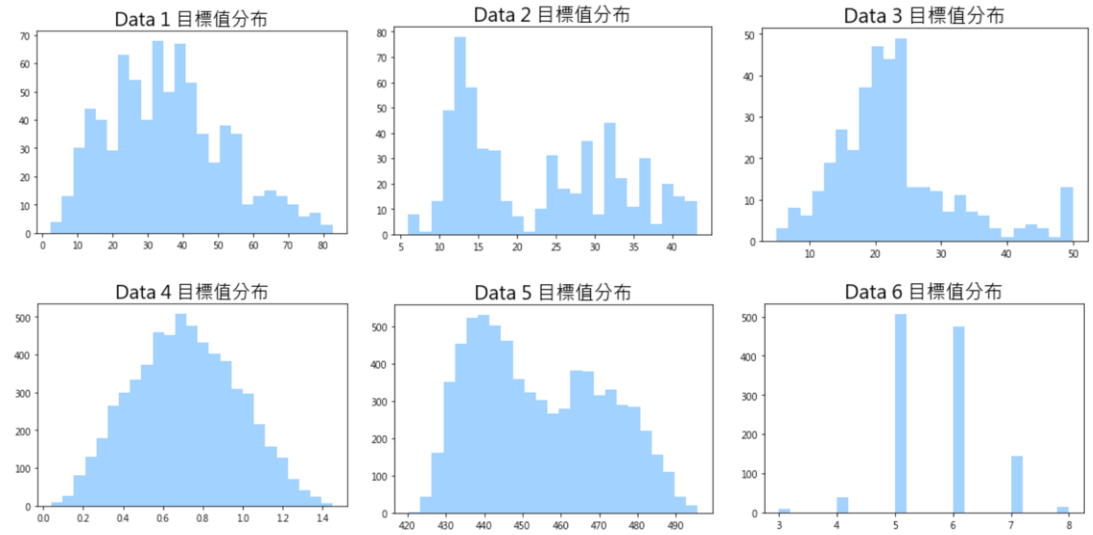
# of n unique	0	1	2	3	4	5	6	7	8	9	10	11	12
Data 1	216	141	104	151	89	204	224	14					
Data 2	12	12	7	4	2	4	4	6					
Data 3	259	23	63	2	74	239	204	226	9	51	40	189	252
Data 4	4013	3960	3962	3982	4010	3999	3958	3900					
Data 5	2302	611	2053	3241									
Data 6	90	127	78	76	121	52	131	98	82	81	55		
Data 7	5	10	8	17	10	14							

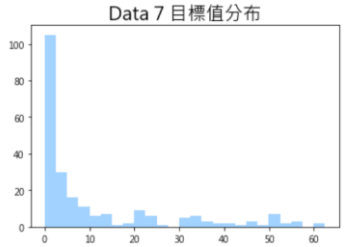
從表一中我們發現每個資料當中，不同特徵出現的遺失值數量並沒有太大的落差，遺失值的出現與否似乎沒有強烈的相關性，因此我們視為完全隨機(MCAR)的機制並直接對遺失值進行填補，而表二中我們發現了比較多有趣的狀況，例如資料二的每個特徵不重複的特徵值較少，但詳細看過資料卻發現某些特徵值出現小數，似乎又不能直接視為類別變數，其他像在資料三或資料七也有部分特徵可能是類別變數，對於這些情況在後續使用程式進行填補時會出現警告，因此我們針對該狀況有相對應的調整措施。



圖一：各資料集變數相關性熱點圖

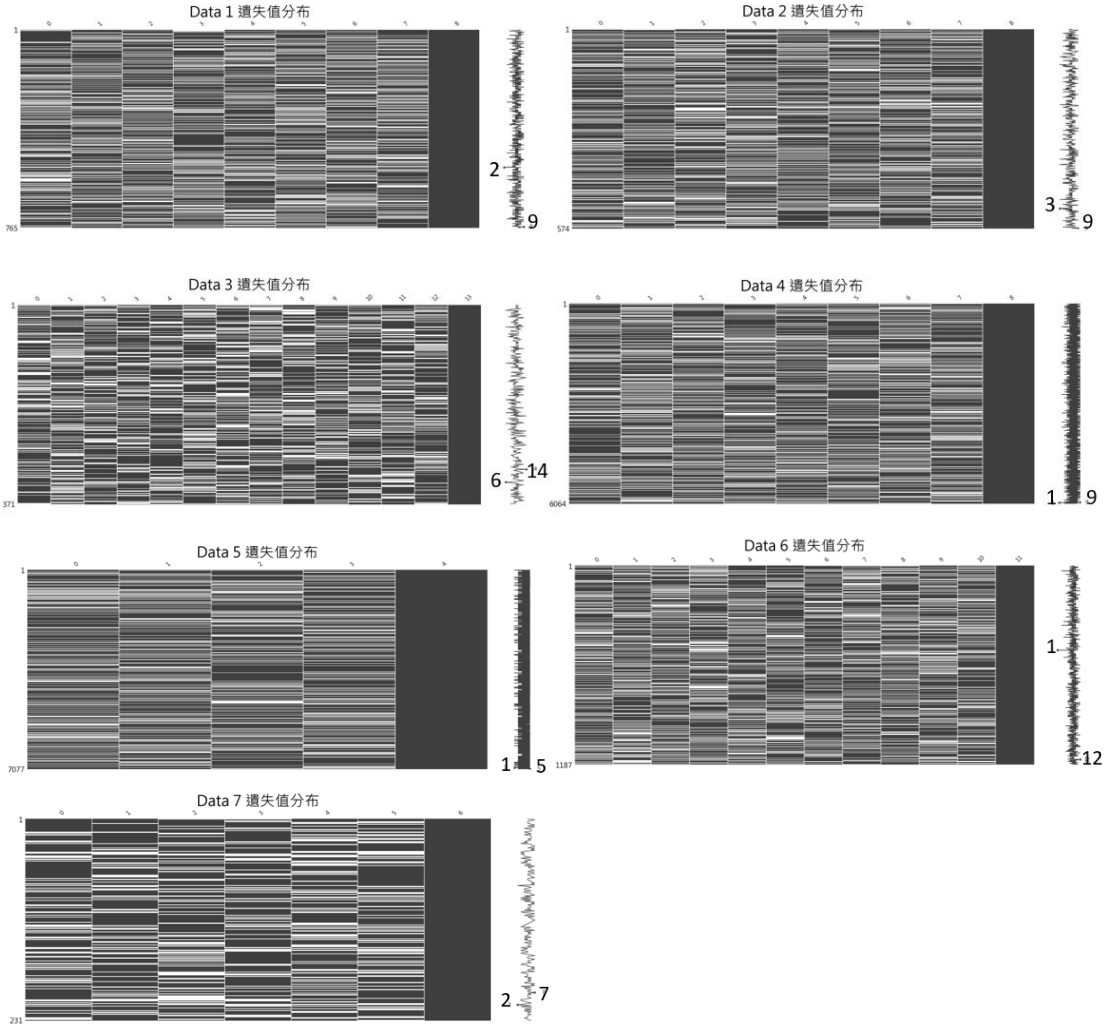
從上圖可知，我們可以掌握各資料集間的變數相關性，除了 Data4 以外，其他 Data 內的彼此之間變數皆有部分相關性比較高，變數間的高相關性也會影響我們未來模型的預估效果，可以考慮利用「主成分分析(PCA)」來降維，將比較有相關的變數轉換成一個更有價值的變數，使我們預測更精確。





圖二：各資料集目標值分佈圖

從上圖可知，可以初步了解每一個資料集被預測的目標值整體分佈樣貌，像是我們發現 Data4 的目標值分佈近似於常態分佈，Data6 的目標值分佈為類別型資料，Data7 的目標值分佈則為嚴重的偏態狀況。透過這些資訊，不但可以供我們調整模型方式的選擇，例如針對 Data6 的預測，我們可以改用分類模型取代迴歸模型，同時也提供我們比較驗證集資料的預測分佈是否與 train 資料集的分佈落差太大，進而思考模型好壞的抉擇。



圖三：缺失值統計圖

從上圖可知，利用 missingno 的 matrix 統計圖，我們可以更清楚看出每個資料集的遺失值分佈以及彼此可能相關性，而圖片右邊有呈現該筆資料集最多缺少

遺失值，可以得知 Data4、Data5、Data6 皆有部分資料只剩下 1 個變數，遺失狀況較差。

2. Methodology

遺失值填補主要可以分為三個部分，第一個是 scikit-learn 中的 impute，當中所使用的方法為 Iterative(MICE)跟 KNN，我們發現使用 sklearn 最大的好處是運算時間較短，而且最初使用 KNN 填值很快就能衝高 TP_mean 的分數。

- Iterative : 使用循環的方式將每個特徵當中出現的遺失值視為要預測的變數，利用剩餘特徵建立多個迴歸模型對遺失值做預測，調整的參數有 max_iter(最大迭代次數)、tol(收斂的容許值)、initial_strategy(遺失值初始化方法)
- KNN : 使用資料集中最接近的幾個鄰居對遺失值進行填補，調整的參數有 n_neighbor(鄰居個數)以及 weights(預測時的權重)

第二個則是使用 fancyimpute，該套件中大概有五種填補方法，我們只有嘗試兩種方法分別是 softimpute 與 matrix factorization，在競賽中段我們發現 matrix factorization 對特定幾筆資料的預測可以有非常顯著的提升，但使用上執行較久而且後續發現效果並不穩定。

- Softimpute : 使用迭代方式進行 soft-thresholded 的 SVD 分解，調整的參數有 convergence_threshold(收斂時閾值大小)、max_iters(最大迭代次數)、max_rank(降維後秩的大小)
- Matrix factorization : 透過 SGD 對遺失值資料矩陣進行分解，產生兩個維度較小的矩陣，調整的參數有 patience(容許次數)、l2_penalty(L2 正規化) min_improvement(分數提升下限)

最後我們改用 R 進行填值，因為 R 當中有較多套件而且安裝上非常簡單，缺點就是文件的描述上比較多數學用語，而且某些套件的執行時間較長，整體來說在遺失值的填補上效果是最好的。

- missForest : 使用 Random Forest 的概念進行填補並且同時適用在連續、類別型的資料中，此外還可以透過平行化方式減少執行時間，調整的參數有 max_iter(最大迭代次數)、ntree(樹的生成數)、parallelize(平行化模式)
- mice : 與 sklearn 的 Iterative 套件原理一樣，但可以運用的預測方法範圍更為多元(如圖四)，可用於類別分類填值、數值迴歸填值等，甚至可以針對不同變數用不同方法填值，經過我們多次嘗試過後，最終選擇以 pmm、cart、midastouch 三種方法，填出遺失值的資料集最佳，調整的參數有 maxit(最大迭代次數)、m(填值次數)、method(預測方法)

pmm	any	Predictive mean matching
midastouch	any	Weighted predictive mean matching
sample	any	Random sample from observed values
cart	any	Classification and regression trees
rf	any	Random forest imputations
mean	numeric	Unconditional mean imputation
norm	numeric	Bayesian linear regression
norm.nob	numeric	Linear regression ignoring model error
norm.boot	numeric	Linear regression using bootstrap
norm.predict	numeric	Linear regression, predicted values
quadratic	numeric	Imputation of quadratic terms
ri	numeric	Random indicator for nonignorable data
logreg	binary	Logistic regression
logreg.boot	binary	Logistic regression with bootstrap
polr	ordered	Proportional odds model
polyreg	unordered	Polytomous logistic regression
lda	unordered	Linear discriminant analysis
2l.norm	numeric	Level-1 normal heteroscedastic
2l.lmer	numeric	Level-1 normal homoscedastic, lmer
2l.pan	numeric	Level-1 normal homoscedastic, pan
2l.bin	binary	Level-1 logistic, glmer
2lonly.mean	numeric	Level-2 class mean
2lonly.norm	numeric	Level-2 class normal
2lonly.pmm	any	Level-2 class predictive mean matching

圖四：mice 填值預測方法

3. Experimental analysis

首先將 train.csv 的資料再切出 15%作為驗證集，另外的 85%則拿來做為訓練，各個資料集在驗證集的 TP_mean 如下表所示，運用預測的模型為 LGBM Regressor、Xgboost、Catboost 及 Deep Forest 四個模型，並測試不同填補的資料集，加上設定 seed 為 69 用來控制切資料與模型預測時的隨機性，最後以 MAE 作為我們評估的參考值。

- **LGBM Regressor**

LGBMRegressor(boosting_type='gbdt', num_leaves=31, random_state=69)

表三：不同填補方式在驗證集的分數比較

	MICE	KNN	Soft	MF	Forest
Data 1	**6.865**	8.320	8.629	8.184	8.378
Data 2	**1.710**	2.188	2.141	3.007	1.975
Data 3	3.123	3.028	3.272	3.951	**2.936**
Data 4	0.166	0.178	0.169	0.172	**0.165**
Data 5	4.325	4.730	4.711	5.111	**4.292**
Data 6	0.510	0.504	0.537	0.546	**0.466**
Data 7	6.835	6.433	6.265	**6.181**	8.651
TP_mean	**23.534**	25.381	25.724	27.152	26.864

- **Xgboost**

XGBRegressor(n_estimators=18, min_child_weight=5, subsample=0.88)

表四：不同填補方式在驗證集的分數比較

	KNN	Pmm	Cart	mid	Forest
Data 1	6.164	5.318	**5.105**	6.105	6.542
Data 2	**0.668**	0.961	0.688	0.929	1.159
Data 3	**2.071**	2.177	2.338	2.369	2.960
Data 4	0.155	0.142	**0.132**	0.149	0.176
Data 5	2.999	**2.882**	2.888	2.982	4.591
Data 6	**0.397**	0.442	0.430	0.423	0.483
Data 7	**0.607**	0.927	0.694	0.911	5.467
TP_mean	**13.069**	14.369	13.106	16.352	24.864

- **Catboost**

CatboostRegressor(iterations, learing_rate, depth,
l2_leaf_regression, loss_funciton,
eval_metric, random_seed=69)

表五：不同填補方式在驗證集的分數比較

	KNN	Pmm	Cart	MF	Forest
Data 1	5.075	**4.630**	4.824	8.005	6.471
Data 2	**0.573**	0.921	0.863	1.675	2.147
Data 3	**1.831**	2.194	1.172	3.277	2.710
Data 4	0.148	0.134	**0.119**	0.167	0.171
Data 5	2.835	2.646	**2.624**	4.65	4.441
Data 6	**0.359**	0.416	0.393	0.487	0.443
Data 7	1.240	1.428	**1.011**	7.079	7.352
TP_mean	12.061	12.369	**11.006**	25.34	23.735

- **Deep Forest**

CascadeForestRegressor(n_estimators=11, n_trees=10, max_layers=3)

表六：不同填補方式在驗證集的分數比較

	KNN	Pmm	Cart	mid	Forest
Data 1	5.337	4.871	**4.792**	6.053	6.631
Data 2	0.562	0.917	**0.554**	0.963	1.274
Data 3	**1.688**	2.071	1.937	2.135	2.821
Data 4	0.147	0.138	**0.120**	0.141	0.174
Data 5	2.742	2.696	**2.582**	2.749	4.457
Data 6	**0.329**	0.415	0.401	0.414	0.457
Data 7	**0.596**	1.645	0.833	1.067	6.594

TP_mean	12.001	12.009	**11.006**	13.132	21.114

- 小結論

1. LightGBM 搭配任一種填值方法，在整體驗證集上的預測結果都很接近。使用 LightGBM 作為預測的模型時，搭配 missforest 填補遺失值，對其中幾筆資料的驗證集具有較好的預測效果，但對另外幾筆資料的預測也是最差的，導致在整體的預測分數僅優於 Matrix Factorization。使用 MICE 則是對整體的預測較為平均，整體的預測結果也是所有填值方法中最優秀的。而使用 KNN、SoftImpute、Matrix Factorization 的效果在驗證集則較為普通。

2. 在使用 XGBoost 時，可以看到相較於 LightGBM 而言，相同的填值方法(KNN、Forest)搭配 XGBoost 預測時，在驗證集的預測結果明顯好上許多。再進一步查看其他填值方法的效果，除了 missforest 的結果較差以外，其他四種填補遺失值的方法在驗證集上的預測結果都非常接近。用 KNN 填補遺失值，除了在第一筆資料的預測結果略差以外，在其他的個別資料及或是整體的預測結果最為優秀；使用 Pmm 或 Cart 填值，在第一筆資料的預測結果稍微勝過 KNN、mid，在不同資料集的分數較為平均。

3. 使用 CatBoost 時，使用不同方法填值，預測效果相較於前面兩種模型，會有更大的落差。使用 KNN、Pmm、Cart 都能得到恨好的預測結果，搭配 Cart 填值，在過半數的資料，預測效果都是最好的，整體分數也明顯低於其他方法。而使用 MatrixFactorization 的預測結果在不同資料之間有最大的落差，整體分數在 CatBoost 之中也是最不好的。在前面兩個模型表現都不太好的 missforest，搭配 CatBoost 的結果也沒有很出色。

4. 使用 DeepForest，在相同的填值方法(KNN、Pmm、Cart、missforest)下，預測的結果都與 CatBoost 相距不遠。搭配 KNN、Cart 的效果都很好，使用 Pmm 的效果也和前面兩種填值差不多，使用 mid 的效果則稍微差於前面的三種方法，而 missforest 在 DeepForest 的效果是最差的。

整理上方的結果，以各個模型在驗證集的預測分數而言，我們認為使用 LightGBM 時，搭配 KNN 填值的效果最好；使用 XGBoost 時，可以搭配 KNN、Cart；使用 CatBoost 時，不論是 KNN、Pmm、Cart 都可以；使用 DeepForest 時，KNN、Pmm、Cart、mid 都是可以考慮的填值方法。再比較不同的模型，我們會認為 DeepForest 可以是最先考慮的選擇，但以上僅是在驗證集實驗的結果，實際在測試集可能會有其他更好的填值方法與模型。

4. Conclusions

在這次競賽過程中，我們花最多心力的便是於遺失值填補方式，一開始發現

Python 語言填補遺失值較為不理想，抑或是 fancyimpute 安裝失敗，進而讓我思考其他填值資源的可能性，最後也意外發現 R 語言對於遺失值的填補資源，更為完善、好用，因此對於後期的預測過程中，皆使用 R 語言填補完後的資料集，並套用預測較佳的 boost 模型，例如 LGBM Regressor、Xgboost 及 Catboost，還有 Deep forest 模型，最終給我們很不錯的預測結果。

在 public set 上，我們最後用遺失值填補的最佳方法為 cart 與 missForest，預測方法為 Catboost，得到第六名的佳績(MP_mean=177.5843、TP_mean=29.0456)，然而在 private set 上的 MP_mean 預測結果落在十名以內，推測可能原因是發現模型過擬和或是原始資料雜訊過多的問題，導致最後 TP_mean 跑到第 15 名，但因為填補與預測都包含 tree-based 的概念，所以我們大膽猜測這種方法的準確率應該都不錯，只不過對於這七筆資料來說，採用比較好的填補方法不見得結果預測會比較精準，如果採用我們前期的方式雖然 public set 的分數較差，或許在 private set 的表現可以比較穩定。

綜合這次競賽中所學，我們未來將可以朝以下方向，持續精進改進：

1. **整合相關性高的變數**
   從相關性熱點圖中，可以知道在資料集內有部分變數相關性極高，未來可以朝是用 PCA 降維方式，得到更有價值性的變數。

2. **優化填補方法**
   基於時間緣故，此次填值方式並沒有嘗試在同一個資料集下，對於不同變數的遺失值做不一樣方法的填值，以及也透過不同填遺失值的資料集加權平均，獲得更為普遍性填補的資料集。

3. **移除極端值**
   在此次預測競賽中，我們發現雖然填值的好壞，會影響我們對於預測結果的好壞，或許從極端值的刪除將可以有效改善資料集以及預測的品質。

4. **Python&R 共同環境**
   我們後來發現可以將 Python 整合到 R 的環境中，因為在 R 裡面的遺失值填補方法較為齊全，使用 reticulate 可以讓我們直接在 RStudio 嵌入 python 程式碼，在程式執行上的效率較高。

5. **加速填補遺失值速度**
   我們在運用 R 語言裡的 MICE 套件時，發現有些方法由於運算資源過繁重，以導致填補速度極慢，未來應尋找其他更優化速度的套件。

6. **加權預測最佳目標值**
   如同我們類別型模型 Voting 的方式，或許我們也可以利用不同模型預測的目標值進行加權平均，或許更可以得到更加的 TP_mean。

5. Citations

這邊僅列出在 R 中所使用的 package 的官方文件與下載網址，對於其他更應用層面的介紹請參考 Github 連結(https://github.com/Ololkao/BDA_CP1)，在 README 中我們附上 reticulate 的安裝過程與使用說明，另外還有針對 R 的遺失值填補教學。

- **missForest: Nonparametric Missing Value Imputation using Random Forest**
  https://cran.r-project.org/web/packages/missForest/index.html
- **mice: Multivariate Imputation by Chained Equations**
  https://cran.r-project.org/web/packages/mice/index.html

- **missForest: Nonparametric Missing Value Imputation using Random Forest**
  https://www.rdocumentation.org/packages/missForest/versions/1.4/topics/missForest

- **mice: mice: Multivariate Imputation by Chained Equations**
  https://www.rdocumentation.org/packages/mice/versions/3.13.0/topics/mice

- **ML02: 初探遺失值(missing value)處理**
  https://morton-kuo.medium.com/ml02-na-f2072615158e

- **R 筆記 – (10)遺漏值處理(Impute Missing Value)**
  https://rpubs.com/skydome20/R-Note10-Missing_Value?fbclid=IwAR0wfH0qDjTO5GB0G--owcs1h01b3W4GmOmREuSEDvk2g8lwf5QVdt4a4ww