DATA PROCESSING - CHAPTER 4

- CRITICAL → DATA IS EXAMINED + PREPROCESSED BEFORE LEARNING.

MISSING DATA

- VERY COMMON → IRL → DATA IS MISSING
- MISSING VALUES → SEEN AS 'NAN' → NOT A NUMBER
  - ↳ MOST COMPUTATIONAL TOOLS → WON'T PROCESS → CRUCIAL WE TAKE CARE OF MISSING VALUES BEFORE FURTHER ANALYSIS.

MISSING VALUES IN TABULAR DATA

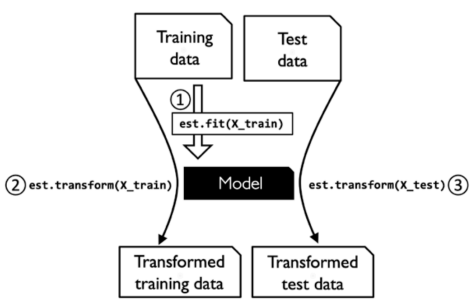
- USE NUMPY ARRAYS WHEN POSSIBLE → MOST SCI-KIT LEARN FUNCTIONS SUPPORT DATAFRAMES.
- REMOVING DATA → MAY SEEM CONVENIENT → COMES WITH DISADVANTAGES
  - ↳ TOO MANY SAMPLES REMOVED ⇒ UNRELIABLE ANALYSIS
  - → TOO MANY FEATURE COLUMNS REMOVED → RISK OF LOSING VALUABLE INFORMATION → CLASSIFIER NEEDS DISCRIMINATION BETWEEN CLASSES.

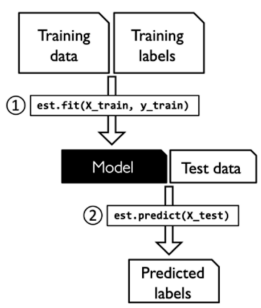
IMPUTING MISSING VALUES

- INTERPOLATION TECHNIQUES
- MOST COMMON INTERPOLATION TECHNIQUE → MEAN IMPUTATION → REPLACE MISSING VALUE WITH MEAN VALUE OF FEATURE COLUMN

SCIKIT LEARN ESTIMATOR API

- SIMPLE IMPUTER → SCIKIT-LEARN TRANSFORMER API
- TWO ESSENTIALS METHODS OF TRANSFORMER → FIT + TRANSFORM
- FIT ⇒ LEARN PARAMETERS FROM TRAINING DATA.
- TRANSFORM ⇒ USES PARAMETERS → TRANSFORM DATA
- DATA ARRAY TO BE TRANSFORMED → SAME NUMBER OF FEATURES AS DATA USED TO FIT.





SUPERVISED LEARNING
↳ PROVIDE CLASS LABEL TO FIT MODEL

- ESTIMATORS (CHAPTER 3) → CAN HAVE BOTH FIT AND TRANSFORM METHOD TOO.
  - ↳ FIT ALSO USED TO LEARN PARAMETERS OF A MODEL → WHEN ESTIMATORS TRAINED FOR CLASSIFICATION.

CATEGORICAL DATA

- REAL-WORLD DATA → WILL INCLUDE MORE THAN JUST NUMERICAL DATA.
- CATEGORICAL DATA → ORDINAL FEATURES → CATEGORICAL VALUES THAT CAN BE SORTED. E.G. T-SHIRT (S, M, L, XL)
  - → NOMINAL FEATURES → DON'T IMPLY ANY ORDER

ORDINAL FEATURES

- CONVERT CATEGORICAL STRING TO INTEGER FOR LEARNING ALGORITHM. → SIMPLE DICTIONARY MAPPING
- DEFINE MAPPING MANUALLY.
- CAN TRANSFORM BACK TO STRING → REVERSE MAPPING.

ENCODING CLASS LABELS

- MANY ML LIBRARIES → REQUIRE CLASS LABELS ENCODED AS INTEGER VALUES.
- SCIKIT-LEARN CLASSIFICATION CONVERTS ⇒ BUT GOOD PRACTICE TO PROVIDE CLASS LABEL AS INTEGERS
- CLASS LABELS → NOT ORDINAL
- SCIKIT-LEARN → LABELENCODER

ONE-HOT ENCODING ON NOMINAL FEATURES

- SCIKIT-LEARN ESTIMATORS FOR CLASSIFICATION → CLASS LABEL TREATED AS CATEGORICAL DATA → NOMINAL ⇒ USED `LABELENCODER`

```
X = df[['color','size','price']].values

color_le = LabelEncoder()

X[:, 0] = color_le.fit_transform(X[:, 0])

X
```

✓ 0.0s

```
array([[1, nan, 10.1],
       [2, nan, 13.5],
       [0, nan, 15.3]], dtype=object)
```

- BLUE = 0
- GREEN = 1
- RED = 2

- DON'T FEED THIS INTO TO CLASSIFIER!!
  - ↳ MOST COMMON MISTAKE IN DEALING WITH CATEGORICAL DATA.
  - ↳ COMMON CLASSIFICATION MODELS → ASSUME GREEN IS LARGER THAN BLUE, AND RED LARGER THAN BLUE.

↓

WORKAROUND → ONE-HOT ENCODING

- ↳ CREATE NEW DUMMY FEATURE FOR EACH UNIQUE VALUE IN NOMINAL FEATURE COLUMN
- CONVERT TO NEW FEATURES → BINARY VALUES FOR INDICATION → E.G. BLUE=1, GREEN=0, RED=0

- `COLUMNTRANSFORMER` → MULTI-FEATURE ARRAY.
- ONE-HOT ENCODING → INTRODUCES MULTI-COLLINEARITY → ISSUE FOR SOME MODELS ⇒ METHODS THAT REQUIRE MATRIX INVERSION
- FEATURES → HIGHLY CORRELATED → MATRICES COMPUTATIONALLY DIFFICULT TO INVERT.
  - ↳ REMOVE ONE FEATURE COLUMN

OTHER OPTIONS THAN ONE-HOT ENCODING:

- USEFUL WORKING WITH CATEGORICAL FEATURES THAT HAVE HIGH CARDINALITY (A LARGE NUMBER OF UNIQUE LABELS)
- BINARY ENCODING: PRODUCES MULTIPLE BINARY FEATURES SIMILAR TO ONE-HOT BUT REQUIRES FEWER FEATURE COLUMNS.
  NUMBERS FIRST CONVERTED TO BINARY REPRESENTATIONS → EACH NEW BINARY NUMBER POSITIONS → FORMS NEW FEATURE COLUMN
- COUNT OR FREQUENCY ENCODING: REPLACES LABEL OF EACH CATEGORY BY NUMBER OF TIMES OR FREQUENCY IT OCCURS IN TRAINING SET.

PARTITIONING TRAINING AND TEST DATASETS

- `train_test_split` → SCIKIT-LEARN → CONVENIENT FOR SPLITTING DATA.
- APPROPRIATE RATIO FOR PARTITIONING:
  - ↳ SMALLER TEST SET → MORE INACCURATE ESTIMATION OF GENERALISATION ERROR.
  - ↳ MOST COMMON SPLIT = 60:40, 70:30, 80:20 → DEPENDS ON SIZE OF DATASET.
    - ↳ LARGE DATASETS → 90:10, 99:1 COMMON + APPROPRIATE.

FEATURES ON THE SAME SCALE

- FEATURE SCALING → CRITICAL STEP IN PREPROCESSING PIPELINE
- DECISION TREES + RANDOM FOREST → DON'T NEED TO WORRY ABOUT FEATURE SCALING → SCALE-INVARIANT
- TWO WAYS TO BRING ONTO SAME SCALE: NORMALIZATION + STANDARDIZATION
- MOST OFTEN → NORMALIZATION MEANS → RESCALING FEATURE TO RANGE (0,1) → MIN-MAX SCALING

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{min}}{x_{max} - x_{min}}$$

- MIN-MAX SCALING → COMMONLY USED TECHNIQUE → USEFUL WHEN WE NEED VALUES IN BOUNDED INTERVAL
- STANDARDISATION → MORE PRACTICAL FOR MANY MACHINE LEARNING ALGORITHMS → E.G. GRADIENT DESCENT.
  - ↳ MANY LINEAR MODELS → LOGISTIC REGRESSION + SVM → INITIALISE WEIGHTS TO 0 OR SMALL VALUE CLOSE TO 0.
  - ↳ CENTRE FEATURE COLUMN AT MEAN 0 WITH STD 1 → STANDARD NORMAL DISTRIBUTION (ZERO MEAN + UNIT VARIANCE)
  - ↳ STANDARDISATION → DOES NOT CHANGE SHAPE OF DISTRIBUTION.
  - ↳ MAINTAINS USEFUL INFORMATION = OUTLIERS → ALGORITHM LESS SENSITIVE.

$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x}$$

μ_x = SAMPLE MEAN OF PARTICULAR FEATURE COLUMN
σ_x = STANDARD DEVIATION.

- STANDARDISATION VS. MIN-MAX SCALING

Input	Standardized	Min-max normalized
0.0	-1.46385	0.0
1.0	-0.87831	0.2
2.0	-0.29277	0.4
3.0	0.29277	0.6
4.0	0.87831	0.8
5.0	1.46385	1.0

`StandardScaler` → FIT ONLY ONCE ON TRAINING DATA
`RobustScaler` → SCI-KIT LEARN → HELPFUL + RECOMMENDED FOR SMALL DATASETS THAT CONTAIN MANY OUTLIERS
→ IF MACHINE LEARNING ALGORITHM PRONE TO OVERFITTING
↳ EACH FEATURE COLUMN → REMOVES MEDIAN VALUE AND SCALES DATASET ACCORDING TO 1st + 3rd QUARTILE OF DATASET.

SELECTING MEANINGFUL FEATURES

- OVERFITTING → MODEL FITS TO PARAMETERS TOO CLOSELY, BUT DOES NOT GENERALISE WELL WITH NEW DATA → MODEL HAS HIGH VARIANCE
- COMMON SOLUTIONS FOR GENERALIZATION ERROR:
  - COLLECT MORE TRAINING DATA
  - INTRODUCE PENALTY FOR COMPLEXITY VIA REGULARIZATION
  - CHOOSE A SIMPLER MODEL WITH FEWER PARAMETERS
  - REDUCE DIMENSIONALITY OF THE DATA

L1 AND L2 REGULARIZATION AS PENALTIES AGAINST MODEL COMPLEXITY

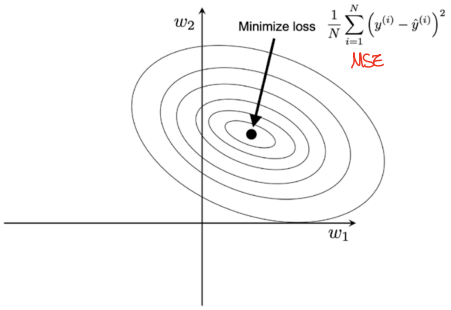
- L2 REGULARISATION → REDUCE COMPLEXITY OF MODEL → PENALISE LARGE INDIVIDUAL WEIGHTS.

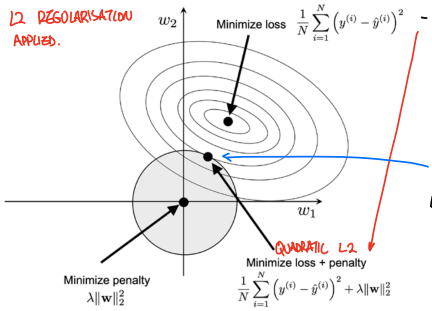
$$L2: \|w\|_2^2 = \sum_{j=1}^{m} w_j^2 \quad , \quad L1: \|w\|_1 = \sum_{j=1}^{m} |w_j| \qquad w = \text{WEIGHT VECTOR}$$

- REPLACED SQUARE OF WEIGHTS WITH SUM OF ABSOLUTE VALUES OF WEIGHT.
- L1 → SPARSE FEATURE VECTORS → MOST FEATURE WEIGHTS WILL BE ZERO.
  ↳ SPARSITY → USEFUL IN PRACTICE (IF WE HAVE HIGH-DIMENSIONAL DATASET WITH MANY IRRELEVANT FEATURES)
  ↳ TECHNIQUE FOR FEATURE SELECTION

L2 - GEOMETRICALLY

- L2 → ADDS PENALTY TERM TO LOSS FUNCTION → LESS EXTREME WEIGHT VALUE
- REGULARISATION → PENALTY TERM TO LOSS FUNCTION.
- INCREASING REGULARISATION WITH λ → SHRINK TOWARDS ZERO + DECREASE DEPENDANCE OF MODEL ON TRAINING DATA

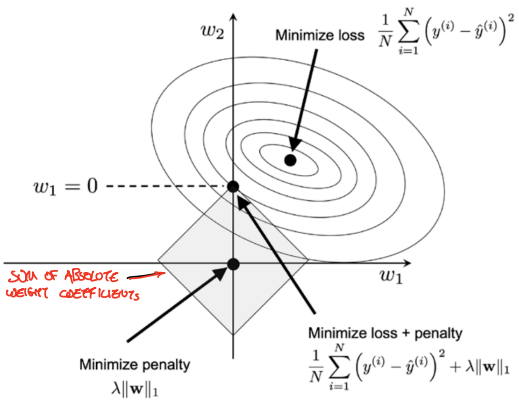




- WEIGHT COEFFICIENT CANNOT EXCEED REGULARISATION BUDGET.
  - ↳ WEIGHT COEFFICIENT CANNOT FALL OUTSIDE THE SHADED AREA.
  - ↳ WE WANT TO MINIMISE LOSS FUNCTION BEST PLACE UNDER PENALTY CONSTRAINTS
  - ↳ LARGER λ → PENALISATION GROWS FASTER
    - ↳ NARROWER L2 BALL

- IN SUMMARY ⇒ OUR GOAL → MINIMISE SUM OF UNPENALISED LOSS + PENALTY TERM
  - ↳ ADDING BIAS + PREFFERING SIMPLER MODEL TO REDUCE VARIANCE IN ABSCENCE OF SUFFICIENT TRAINING DATA

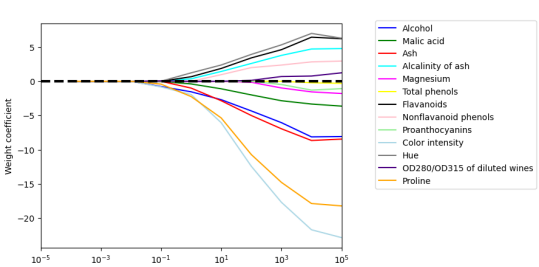
SPARSE SOLUTIONS WITH L1



- LOOK INTO WHY L1 REGULARISATION CAN LEAD TO SPARSE SOLUTIONS.

- WINE DATASET → 13 WEIGHTS FOR EACH ROW → EACH WEIGHT MULTIPLIED BY THE RESPECTIVE FEATURE:

$$z = w_1x_1 + \cdots + w_mx_m + b = \sum_{j=1}^{m} x_j w_j + b = w^T x + b$$



IMPACT OF VALUE OF THE REGULARIZATION STRENGTH HYPERPARAMETER C

SEQUENTIAL FEATURE SELECTION ALGORITHMS

- ALTERNATIVE WAY TO REDUCE COMPLEXITY + OVERFITTING → DIMENSIONALITY REDUCTION VIA FEATURE SELECTION. → USEFUL FOR UNREGULARISED MODELS.
- DIMENSIONALITY REDUCTION:
  1. FEATURE SELECTION – SELECT SUBSET OF ORIGINAL FEATURES
  2. FEATURE EXTRACTION - DERIVE INFORMATION FROM FEATURE SET TO CONSTRUCT NEW FEATURE SUBSPACE

FEATURE SELECTION

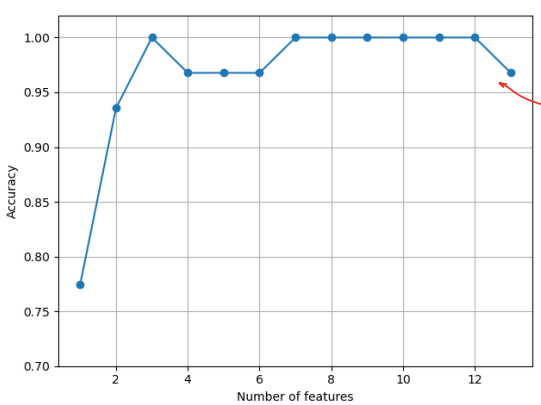
- SEQUENTIAL FEATURE SELECTION ALGORITHM → FAMILY OF GREEDY SEARCH ALGORITHMS
  - ↳ REDUCE INITIAL d- DIMENSIONAL FEATURE SPACE TO k- DIMENSIONAL FEATURE SUBSPACE $k < d$
  - ↳ GOAL:
    1. AUTOMATICALLY SELECT SUBSET OF FEATURES, THAT ARE MOST RELEVANT TO PROBLEM
    2. IMPROVE COMPUTATIONAL EFFICIENCY
    3. REDUCE GENERALIZATION ERROR OF MODEL → REMOVE IRRELEVANT FEATURE OR NOISE → USEFUL FOR ALGO WITHOUT REGULARISATION
- CLASSIC EXAMPLE - SEQUENTIAL BACKWARD SELECTION (SBS)
  - ↳ REDUCE DIMENSIONALITY TO INITIAL FEATURE SUBSPACE WITH MINIMUM DECAY IN PERFORMANCE TO CLASSIFIER TO IMPROVE UPON COMPUTATIONAL EFFICIENCY.
- SBS CAN IMPROVE PREDICTIVE POWER IN SOME CASES

GREEDY SEARCH ALGORITHM

- ATTEMPTS TO FIND THE MOST PROMISING PATH FROM A GIVEN STARTING POINT TO A GOAL.
- EVALUATES COSTS OF ALL PATHS → GOES WITH LOW COST PATH → PROCESS REPEATED UNTIL GOAL REACHED.

SBS ALGORITHM

- SEQUENTIALLY REMOVES FEATURES FROM FULL SUBSET → UNTIL NEW FEATURE SUBSPACE HAS TO DESIRED NUMBER OF FEATURES.
- FEATURE TO BE REMOVED AT EACH STAGE → CRITERION FUNCTION 'J' → WANT TO MINIMISE
- EACH STAGE → ELIMINATE FEATURE THAT CAUSES THE LEAST PERFORMANCE LOSS → AFTER REMOVAL.
- 4 STEPS:
  1. DEFINE $k = d$, d = DIMENSIONALITY OF FULL FEATURE SPACE.
  2. DEFINE x^-, MAXIMISE: $x^- = \text{argmax}\, J(X_k - x) \quad x \in X_k$
  3. REMOVE FEATURE x^-, FEATURE SET: $X_{k-1} = X_k - x^-$; $k = k-1$
  4. TERMINATE $\Rightarrow k$ = NUMBER OF DESIRED FEATURES → OTHERWISE BACK TO STEP 2.



- KNN CLASSIFIER → IMPROVES VALIDATION DATASET AS WE REDUCE NUMBER OF FEATURES. → DUE TO DECREASE IN THE CURSE OF DIMENSIONALITY
  - ↳ CLOSEST NEIGHBOURS SEEM FAR AWAY.

- L1 REGULARISATION → ZERO OUT IRRELEVANT FEATURES VIA LOGISTIC REGRESSION + HOW TO USE SBS FOR FEATURE SELECTION → APPLY TO KNN
  - ↳ SUMMARY OF ABOVE

FINISH CHAPTER + LOOK INTO L1 + L2 IN-DEPTH

FEATURE IMPORTANCE WITH RANDOM FOREST

- ANOTHER USEFUL APPROACH → FEATURE SELECTION WITH RANDOM FOREST.
  - ↳ MEASURE FEATURE IMPORTANCE WITHOUT MAKING ANY ASSUMPTIONS ABOUT WHETHER DATA IS LINEARLY SEPARABLE.
  - ↳ FEATURE IMPORTANCE VALUES ALREADY COLLECTED → SCI-KIT LEARN RANDOM FOREST → 'feature_importance_'
- CAREFUL → IF TWO OR MORE FEATURE HIGHLY CORRELATED → ONE FEATURE MAY BE RANKED VERY HIGHLY, WHILST INFORMATION ON OTHER FEATURES NOT FULLY CAPTURED.
  - ↳ NOT IMPORTANT IF WE ARE ONLY INTERESTED IN PREDICTIVE PERFORMANCE OF MODEL
- SCIKIT-LEARN → 'SelectFromModel' OBJECT → SELECTS BASED ON USER-SPECIFIED THRESHOLD AFTER MODEL FITTING.
  - ↳ USEFUL WITH RANDOM FOREST → INTERMEDIATE STEP IN PIPELINE → CONNECT DIFFERENT PREPROCESSING STEPS WITH ESTIMATOR.