BEST PRACTICES FOR MODEL EVALUATION + HYPERPARAMETER TUNING - CHAPTER 6

STREAMLINING WORKFLOWS WITH PIPELINES

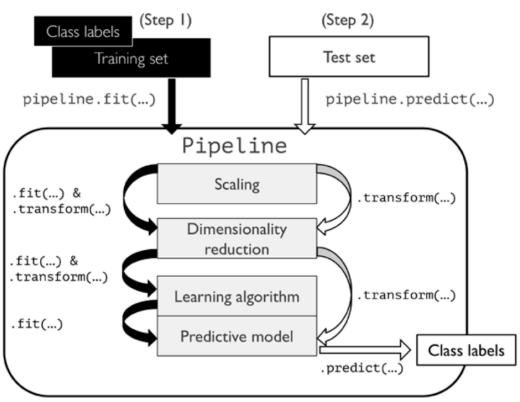
- `Pipeline` CLASS → SCIKIT-LEARN

STEPS:

1. READ DATASET `pd.read_csv()`
2. `LabelEncoder` OBJECT → TRANSFORM CLASS LABELS FROM STRING TO INTEGER
3. CHECK WITH TRANSFORM → CORRECTLY FITTED
4. SPLIT TRAIN - TEST FROM DATASET

COMBINING TRANSFORMER + ESTIMATOR IN A PIPELINE

- BREAST CANCER DATASET.
  - ↳ NEED TO STANDARDISE → MEASURED AT VARIOUS DIFFERENT SCALES
  - ↳ IF WE WANT TO COMPRESS DATA TO LOWER TWO-DIMENSIONAL SUBSPACE (PCA) + DIMENSIONALITY REDUCTION
    - ↳ INSTEAD OF SPLITTING MODEL FITTING + DATA TRANSFORMATION → CHAIN → `StandardScaler` + `PCA` + `LogisticRegression` OBJECT IN PIPELINE.
- `make_pipeline` - TAKES ARBITRARY NUMBER OF TRANSFORMERS (SUPPORT FIT + TRANSFORM METHOD AS INPUT) + ESTIMATOR (FIT + PREDICT METHOD) € SCIKIT-LEARN
  - ↳ META-ESTIMATOR OR WRAPPER AROUND INDIVIDUAL TRANSFORMERS + ESTIMATORS
  - ↳ ESTIMATOR FITTED ON TRANSFORMED TRAINING DATA.
- NO LIMIT TO NUMBER OF STEPS IN PIPELINE, BUT IF PIPELINE USED FOR PREDICTION → LAST ELEMENT MUST BE AN ESTIMATOR.
- PIPELINES → VERY USEFUL WRAPPER TOOLS



K-FOLD CROSS-VALIDATION TO ASSESS MODEL PERFORMANCE

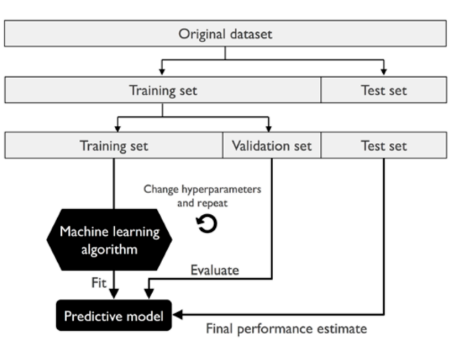
- COMMON CROSS-VALIDATION TECHNIQUE → HOLDOUT CROSS-VALIDATION + K-FOLD CROSS-VALIDATION
  - → HELPS OBTAIN RELIABLE ESTIMATES OF MODEL'S GENERALISATION PERFORMANCE → HOW WELL MODEL PERFORMS ON UNSEEN DATA.

HOLDOUT METHOD

- CLASSIC + POPULAR METHOD
- MODEL SELECTION → CLASSIFICATION PROBLEM → SELECT OPTIMAL VALUES FOR TUNING HYPERPARAMETERS.
- HYPERPARAMETER → TUNING PARAMETER
- HOLDOUT METHOD → SPLIT DATASET → TRAINING + TEST → USE SAME TEST DATASET OVER AGAIN DURING MODEL SELECTION ⇒ MODEL WILL OVERFIT

↳ INSTEAD → SPLIT 3 WAYS - TRAINING, VALIDATION, TEST

- TRAINING → FIT DIFFERENT MODELS
- VALIDATION → MODEL SELECTION
- TESTING → TEST
  - ↳ HASN'T BEEN SEEN DURING TRAINING + MODEL SELECTION
    - ↳ MODEL → LESS BIAS TO GENERALIZE NEW DATA



- DISADVANTAGE:
  - ↳ PERFORMANCE ESTIMATE MAY BE VERY SENSITIVE → DEPENDS ON PARTITION OF TEST + VALIDATION SUBSET.

K-FOLD CROSS-VALIDATION

- TRAINING DATASET → SPLIT INTO k FOLDS WITHOUT REPLACEMENT
- K-1 FOLDS → TRAINING FOLDS → USED FOR MODEL TRAINING
  TEST FOLD → PERFORMANCE EVALUATION
- PROCEDURE REPEATED k TIMES → OBTAIN k MODELS + PERFORMANCE ESTIMATES.
- CALCULATE AVERAGE PERFORMANCE OF MODELS → DIFFERENT + INDEPENDENT TEST FOLDS → OBTAIN PERFORMANCE ESTIMATE → LESS SENSITIVE TO SUB-PARTITIONING OF TRAINING DATA
  ↳ COMPARED TO HOLDOUT
- K-FOLD TYPICALLY USED FOR MODEL TUNING → OPTIMAL HYPERPARAMETER VALUES → YIELD SATISFYING GENERALIZATION PERFORMANCE.
- RETRAIN MODEL WITH COMPLETE TRAINING DATASET → AFTER FINDING SATISFACTORY HYPERPARAMETERS.
  ↳ FIRST → INTERESTED IN SINGLE, FINAL MODEL
  ↳ SECOND → MORE TRAINING EXAMPLES FOR LEARNING ALGORITHM
  } WHY FIT WHOLE TRAINING DATASET TO MODEL.
- K-FOLD CROSS-VALIDATION → RESAMPLING TECHNIQUE WITHOUT REPLACEMENT → ADVANTAGE ⇒ EACH ITERATION, EACH EXAMPLE USED EXACTLY ONCE.
- ALL TEST FOLDS ARE DISJOINT. ⇒ NO OVERLAP BETWEEN TEST FOLDS.

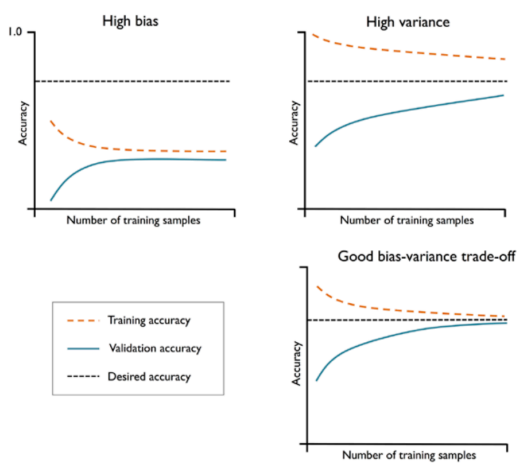
- HOW K-FOLD WORKS WITH, K=10
- OVERALL = K-FOLD USES DATASET BETTER THAN HOLDOUT → ALL DATA POINTS USED FOR EVALUATION.
- GOOD STANDARD FOR K → 10
  ↳ SMALL DATASET → BETTER TO USE MORE FOLDS.
  ↳ MORE K = MORE TRAINING DATA USED IN EACH ITERATION
  ↳ SLOWER COMPUTATION + ESTIMATE NOISIER.
- LARGE DATASET → SMALLER K (5) → ENOUGH FOR GOOD PERFORMANCE + COMPUTATION LOW

DEBUGGING WITH LEARNING + VALIDATION CURVES

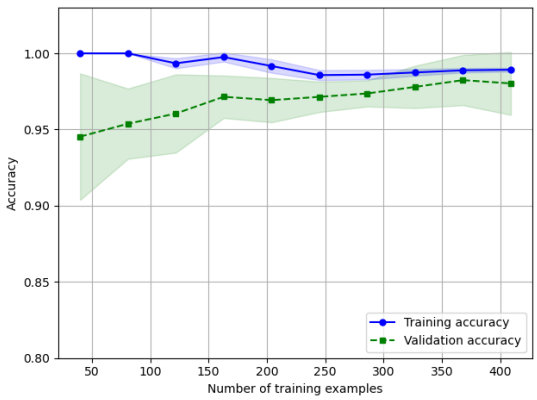
- LEARNING CURVES → DOES LEARNING ALGO HAVE PROBLEMS WITH OVERFITTING (HIGH VARIANCE) OR UNDERFITTING (HIGH BIAS)
- VALIDATION CURVES → HELPS ADDRESS COMMON LEARNING ALGO PROBLEMS

BIAS + VARIANCE WITH LEARNING CURVES

- COMPLEX MODEL FOR TRAINING DATA → MODEL WILL TEND TO OVERFIT. → GET MORE DATA



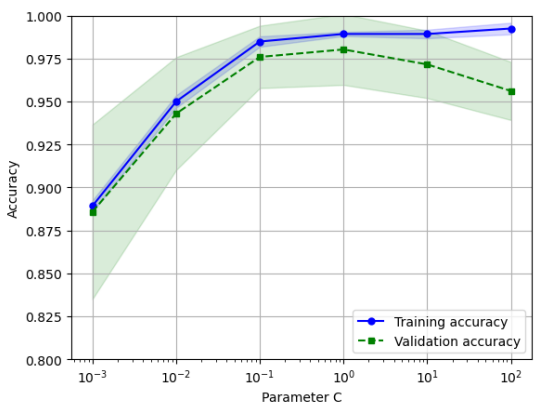
- TOP LEFT → UNDERFIT → SOLUTION → INCREASE NUMBER OF MODEL PARAMETERS
  → CONSTRUCT ADDITIONAL FEATURES
  → DECREASING DEGREE OF REGULARISATION
- TOP RIGHT → OVERFIT → SOLUTION → COLLECT MORE TRAINING DATA
  → REDUCE MODEL COMPLEXITY
  → INCREASE REGULARISATION
- UNREGULARISED MODELS → DECREASE NUMBER OF FEATURES → FEATURE SELECTION OR FEATURE EXTRACTION



- train_sizes = np.linspace(0.1, 1.0, 10) = 10 EVENLY SPACED DATASET SIZES.
- learning_curve = DEFAULT - K-FOLD CROSS VALIDATION => k=10.

ADDRESSING OVER AND UNDERFITTING WITH VALIDATION CURVE

- VALIDATION CURVE = VARY THE VALUES OF THE MODEL PARAMETERS → EXAMPLE = INVERSE REGULARIZATION PARAMETER, C



- VALIDATION CURVE PLOT FOR SVM PARAMETER C.
- validation_curve = STRATIFIED K-FOLD CROSS-VALIDATION → DEFAULT → ESTIMATE PERFORMANCE
- INVERSE REGULARIZATION OF LOGISTIC REGRESSION → 'logisticregression__C'
- PLOTTED AVERAGE TRAINING + CROSS-VALIDATION ACCURACIES.

FINE-TUNING MACHINE LEARNING MODEL VIA GRID SEARCH

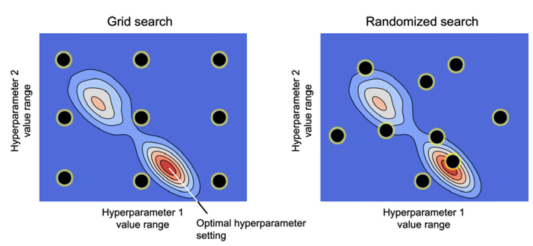
- ML → WE HAVE TWO TYPES:
  1. LEARNED FROM TRAINING DATA + LEARNING ALGORITHM THAT ARE OPTIMISED SEPARATELY
  2. TUNING PARAMETERS
- POPULAR HYPERPARAMETER OPTIMIZING TECHNIQUE → GRID SEARCH → HELP IMPROVE PERFORMANCE OF MODEL.

TUNING HYPERPARAMETERS VIA GRID SEARCH

- SIMPLE → BRUTE-FORCE EXHAUSTIVE SEARCH PARADIGM.
  - ↳ LIST VALUES FOR DIFFERENT HYPERPARAMETERS → COMPUTER EVALUATES THE MODEL PERFORMANCE FOR EACH COMBINATION

HYPERPARAMETER CONFIGURATIONS WITH RANDOMIZED SEARCH

- GRID SEARCH → EXHAUSTIVE SEARCH → GUARANTEED TO FIND OPTIMAL HYPERPARAMETER CONFIGURATION.
  - ↳ LARGE HYPERPARAMETER GRIDS → GRID SEARCH EXPENSIVE IN PRACTICE.
  - ↳ ALTERNATE APPROACH → RANDOMIZED SEARCH
- RANDOMIZED APPROACH → DRAW HYPERPARAMETER CONFIG RANDOMLY FROM DISTRIBUTIONS
  - ↳ ALLOWS EXPLORATION OF WIDER RANGE OF HYPERPARAMETER IN MORE COST + TIME EFFECTIVE WAY.



- GRID SEARCH EXPLORES DISCRETE, USER-SPECIFIED CHOICE → MAY MISS GOOD HYPERPARAMETER CONFIGURATIONS → IF SEARCH SPACE IS TOO SPARSE.
- RANDOMIZED SEARCH FOR SVM.
  ↳ 'RandomizedSearchCV' → DIFFERENCE → SPECIFY DISTRIBUTIONS AS PART OF PARAMETER GRID
  → SPECIFY TOTAL NUMBER OF HYPERPARAMETER CONFIGURATIONS TO BE EVALUATED.

SUCCESSIVE HALVING

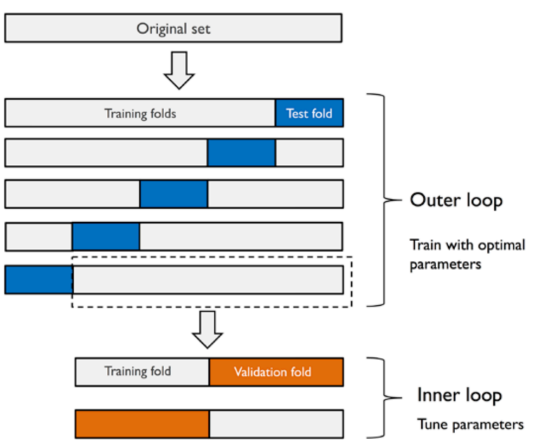
- FINDS SUITABLE HYPERPARAMETERS MORE EFFICIENTLY.
- THROWS OUT UNPROMISING HYPERPARAMETER CONFIGURATION UNTIL ONLY ONE CONFIGURATION LEFT.
- SUMMARY:
  1. DRAW LARGE SET OF CANDIDATE CONFIGS VIA RANDOM SAMPLING
  2. TRAIN MODEL WITH LIMITED RESOURCES → E.G. SMALL SUBSET OF TRAINING DATA.
  3. DISCARD BOTTOM 50% → BASED ON PREDICTIVE PERFORMANCE
  4. BACK TO STEP 2.
- REPEATED UNTIL ONE HYPERPARAMETER CONFIG LEFT.

HYPEROPT

- ANOTHER POPULAR LIBRARY FOR HYPERPARAMETER OPTIMIZATION
- IMPLEMENTS SEVERAL METHODS → RANDOMIZED SEARCH + TREE-STRUCTURED PARZEN ESTIMATORS (TPE)
- TPE → OPTIMIZATION METHOD → PROBABILISTIC MODEL → CONTINUOUSLY UPDATED → BASED ON PAST HYPERPARAMETER EVALUATIONS + PERFORMANCE

ALGORITHM SELECTION WITH NESTED CROSS-VALIDATION

- K-FOLD CROSS-VALIDATION + GRID / RANDOMIZED SEARCH → USEFUL APPROACH FOR FINE-TUNING
- NESTED CROSS-VALIDATION → OUTER + INNER K-FOLD CROSS-VALIDATION
  ↳ OUTER: SPLIT DATA INTO TRAINING + TEST FOLD. → AFTER MODEL SELECTION TEST FOLD USED FOR EVALUATING MODEL PERFORMANCE
  ↳ INNER: SELECTS MODEL USING K-FOLD CROSS-VALIDATION

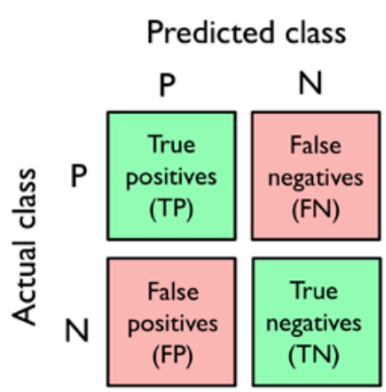


→ 5 x 2 CROSS-VALIDATION

- AVERAGE CROSS-VALIDATION ACCURACY → GOOD ESTIMATE OF WHAT TO EXPECT IF WE TUNE HYPERPARAMETER + USE ON UNSEEN DATA
- WORKS BETTER THAN DECISION TREES

PERFORMANCE EVALUATION METRICS

- SOME OTHER EXAMPLES:
  ↳ PRECISION
  ↳ RECALL
  ↳ F1 SCORE
  ↳ MATTHEWS CORRELATION COEFFICIENT (MCC)



CONFUSION MATRIX

- MATRIX → PERFORMANCE OF LEARNING ALGORITHM
- TRUE POSITIVE (TP)
  TRUE NEGATIVE (TN)
  FALSE POSITIVE (FP)
  FALSE NEGATIVE (FN)

OPTIMISING PRECISION + RECALL OF CLASSIFICATION MODEL

- ERROR (ERR) + ACCURACY (ACC) → INFORMATION ABOUT MISCLASSIFIED EXAMPLES.
- ERROR → SUM OF FALSE PREDICTIONS / NUMBER OF TOTAL PREDICTIONS

$$ERR = \frac{FP + FN}{FP + FN + TP + TN} \qquad ACCURACY = ACC = \frac{TP + TN}{FP + FN + TP + TN} = 1 - ERR$$

- TRUE POSITIVE RATE (TPR) + FALSE POSITIVE RATE (FPR) → USEFUL FOR IMBALANCED CLASS PROBLEMS → FRACTION OF POSITIVE EXAMPLES THAT WERE CORRECTLY IDENTIFIED OUT OF TOTAL POOL.

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \qquad TPR = \frac{TP}{N} = \frac{TP}{FN + TP}$$

- PRECISION (PRE) + RECALL (REC)
  - ↳ RECALL → HOW MANY RELEVANT RECORDS ARE CAPTURED AS TP.
  - ↳ PRECISION → HOW MANY RECORDS PREDICTED AS RELEVANT ARE ACTUALLY TP.

$$REC = TPR = \frac{TP}{P} = \frac{TP}{FN + TP} \qquad PRE = \frac{TP}{TP + FP}$$

- MALIGNANT TUMOR DETECTION:
  - ↳ IF WE OPTIMISE RECALL → MINIMIZE CHANCE OF NOT DETECTING MALIGNANT TUMOR → HOWEVER, INCREASES FP
  - ↳ IF WE OPTIMISE PRECISION → EMPHASISES CORRECTNESS IF WE PREDICT PATIENT HAS MALIGNANT TUMOR → HOWEVER, INCREASES FN

- BALANCE OPTIMISING PRE + REC AND MEAN OF PRE + REC ⇒ F1 SCORE

$$F1 = 2\,\frac{PRE \times REC}{PRE + REC}$$

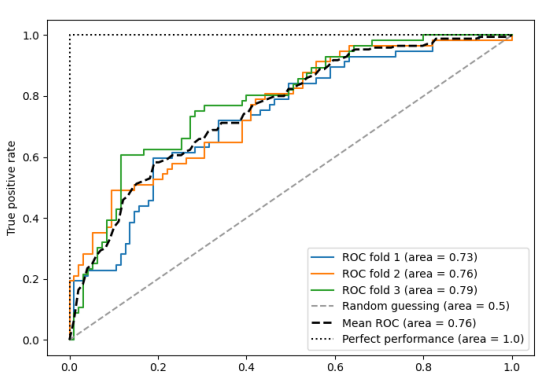
- MCC - POPULAR IN BIOLOGICAL CONTEXT

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

- MCC RANGES BETWEEN $-1, 1$ ⇒ OTHERS DON'T
  - ↳ TAKES ALL ELEMENTS OF CONFUSION MATRIX INTO ACCOUNT.

PLOTTING RECEIVER OPERATING CHARACTERISTIC (ROC)

- RECEIVER OPERATING CHARACTERISTIC (ROC) GRAPHS → USEFUL FOR SELECTING MODELS FOR CLASSIFICATION BASED ON PERFORMANCE
  - ↳ COMPUTED BY SHIFTING THE DECISION THRESHOLD OF CLASSIFIER
- DIAGONAL ROC → RANDOM GUESSING → CLASSIFICATION MODELS UNDER DIAGONAL → WORSE THAN RANDOM GUESSING.
- PERFECT CLASSIFIER → TOP LEFT-CORNER OF GRAPH → TPR = 1, FPR = 0
- ROC AREA UNDER THE CURVE (ROC AUC) → PERFORMANCE OF CLASSIFICATION MODEL.
- PRECISION-RECALL CURVES → FOR DIFFERENT PROBABILITY THRESHOLDS.



SCORING METRIC FOR MULTICLASS CLASSIFICATION

- SCIKIT-LEARN → MACRO + MICRO AVERAGING METHODS → EXTEND SCORING METRICS TO MULTICLASS PROBLEMS VIA OvA

$$PRE_{micro} = \frac{TP_1 + \cdots + TP_K}{TP_1 + \cdots + TP_K + FP_1 + \cdots + FP_K}$$

$$PRE_{macro} = \frac{PRE_1 + \cdots + PRE_K}{K}$$ → AVERAGE SCORE OF DIFFERENT SYSTEMS

- MICRO → USEFUL IF WE WANT TO WEIGHT EACH INSTANCE OR PREDICTION EQUALLY
- MACRO → WEIGHTS ALL CLASSES EQUALLY → EVALUATE OVERALL PERFORMANCE OF CLASSIFIER WITH REGARDS TO MOST FREQUENT CLASS LABELS.

CLASS IMBALANCE

- QUITE COMMON WHEN WORKING WITH REAL LIFE DATA
  - ↳ E.G. SPAM FILTERS, FRAUD DETECTION, DISEAS SCREENING.
- ALGORITHM LEARNS A MODEL THAT OPTIMISES THE PREDICTIONS BASED ON MOST ABUNDANT CLASS IN THE DATASET.
- IMBALANCED CLASS PROPORTIONS → MODEL FITTING → ASSIGN LARGER PENALTY TO WRONG PREDICTIONS ON MINORITY CLASS.
  - → UPSAMPLING MINORITY CLASS → DOWNSAMPLING MAJORITY CLASS.
- TEST AND FIGURE OUT WHAT WORKS FOR YOUR MODEL.
- ANOTHER OPTION:
  - ↳ GENERATION OF SYNTHETIC TRAINING EXAMPLES.
  - ↳ SYNTHETIC OVER-SAMPLING TECHNIQUE (SMOTE)
    - ↳ LOOK INTO THIS