Data Wrangling & Data Analysis final Cheat Sheet - Created by Tim Rentenaar

Regression evaluation methods

Mean Squared Error (MSE): $\frac{1}{n}\sum(y-\hat{y})^2$

Root Mean Squared Error (RMSE): $\sqrt{MSE}$

Mean Absolute Error (MAE): $\frac{1}{n}\sum|y-\hat{y}|$

Median Absolute Error (mAE): $median|y-\hat{y}|$

Proportion of variance explained (R^2): $correlation(y,\hat{y})^2$ or $1-\frac{\sum_i(y_i-\hat{y}_i)^2}{\sum_i(y_i-\overline{y}_i)^2}$

Classification confusion matrix evaluation methods

Total error: FP + FN

Specificity: TN / (TN + FP)

Sensitivity / Recall: TP / (TP + FN)

Accuracy: (TP + TN) / (TP + FP + TN + FN)

Error rate: 1 - Accuracy

Negative Predicted Value (NPV): TN / (TN + FN)

Positive predicted value (PPV) / Precision: TP / (TP + FP)

F_1 **score**: (2 * Precision * Recall) / (Precision + Recall)

True Positive Rate (TPR) = Recall

False Positive Rate (FPR) = 1 - Specificity

Receiver Operating Characteristics curve (ROC curve): Plot of the TPR against the FPR, at various threshold settings.

Area Under The (ROC) Curve (AUC): The overall performance of a classifier summarized over all possible thresholds. An ideal ROC curve will hug the top left corner, so the larger the AUC the better the classifier.

Calibration plot: Plots the observed proportion against the predicted probability. Ideally the values for observed and predicted align.

Deep learning

Universal function approximation theorem: Any "well-behaved" function can be represented by neural net of sufficient width and sufficient depth with nonlinear activation function

Activation function: Applies non-linearity to output of each neuron. Enables the model to approximate complex non-linear relationships and introduce a threshold or decision boundary

Sigmoid function: Converts linear functions to values in a range between 0 and 1

Rectified Linear Unit (ReLU): Outputs 0 when x is below 0, else it outputs x (f(x) = max(0,x))

Formula for each neuron: $h_k(X) = g(W_{0k} + \sum W_{pk}x_p)$. Where g is the activation function, W_{0k} is the bias for neuron k, W_{pk} is the weight for the previous output p for neuron k, x_p is the previous output p.

Softmax function: Transforms the raw outputs of a neural network into a vector of probabilities

Convolutional Neural Networks (CNN): Neural Network designed for processing visual data. Applies a kernal (filter) over an image which decides which feature is important in the image.

Convolution Layer: The parameters/weight in a convolution layer are the elements of the filter

Pooling layer: Reduces dimensionality by condensing a larger image into a smaller summary image

Data augmentation: Replicate training images and distort them in natural ways. Creates more training data.

Recurrent Neural Networks (RNN): A neural network where the input is a sequence

Deep learning: loss functions and regularization

Loss function: Quantifies model performance. For continuous outcomes squared error can be used, for binary outcomes binary cross-entropy can be used.

Binary cross-entropy: Quantifies the difference between predicted probabilities and actual binary labels

Local minimum loss: A point in the parameter space where the loss function reaches its lowest value within a specific region

Global minimum loss: The lowest point in the entire parameter space where the loss function reaches its minimum value

Gradient: The direction in which we need to move to decrease loss

Gradient descent: An iterative process to find the minimum loss by taking steps in the direction where loss decreases

Learning rate: Affects how big of a step (magnitude of parameter update) can be taken for each loss optimization iteration

Backpropogation: Calculates the gradient of the loss with respect to the model's parameters by starting at the output layer and working backwards through the network

Stochastic gradient descent: Instead of calculating gradients with respect to the entire loss function, only use a random batch of data

Epoch: The amount of looks at the full data

Regularization: Introduces bias in the parameters to improve generalization. Fights dimensionality and overfitting

Early stopping: Do not train for more epochs but stop when loss does not improve

Dropout regularization: For loss optimization: In each iteration, only update a subset of parameters

Clustering

Agglomerative Hierarchical clustering: Assign all examples to their individual cluster, Combine most similar clusters, keep combining clusters until there is only one cluster left, select number of clusters for the final solution

Divisive Hierarchical clustering: Start with one cluster and keep splitting most different until individual examples are left

Average linkage: Calculate all pairwise distances between observations in cluster A and observations in cluster B, record the average intercluster distance

Complete linkage: Calculate all pairwise distances between observations in cluster A and observations in cluster B, record the largest intercluster distance

Single linkage: Calculate all pairwise distances between observations in cluster A and observations in cluster B, record the smallest intercluster distance

Centroid linkage: Distance between the centroid of cluster A and the centroid of cluster B

Different distance meanings: Continuous (Euclidean, maximum, Manhattan, Minkowski, . . .), Time Series ("Dynamic time warping", Fréchet, cross-corr., wavelets, . . .), Networks (Modularity, Shortest path, . . .), Text/DNA (Edit distance, Hamming distance, TF-IDF distance, . . .), Images ("Structural similarity", GAN loss, ...)

Partitional clustering: Clusters the dataset into non-hierarchical clusters, the amount of clusters that will be found is given by the user beforehand

k-means clustering algorithm: 1. Randomly assign examples to k clusters; 2a. Calculate the centroid for each cluster; 2b. Assign each example to the cluster belonging to its closest centroid; 3. If the assignments changed, go to step 2a, else stop.

Cluster evaluations

External validation of clusters: Are clusters associated with external feature y?

Visual exploration for cluster validation: Make a visual inspection to see if the clusters look correct. Impossible to see with many variables, possible to reduce variables to 2D for visual inspection (e.g. UMAP, t-SNE, MDS, Discriminant Coordinates, PCA)

Stability assessment for cluster validation: How much does clustering change when changing some hyperparameters, observations, features

Internal validation indices: Look at the data and clusters and quantify how "successful" the clustering is by some measure (e.g. Average Silhouette Width (ASW), "Gap statistic", . . .)

Model-based Clustering

Model-Based Clustering: Partitions a dataset into clusters based on the underlying assumption that the data is generated from a finite mixture of component models. Each component model is a probability distribution, typically a parametric multivariate distribution.

Gaussian Mixture Models (GMM): Data within each cluster (multivariate) is normally distributed. Parameters can be either the same or different across groups: Volume, Shape, Orientation.

Advantages of Model-Based Clustering: Assumptions about the clusters are explicit, not implicit; For each observation, get a posterior probability of belonging to each cluster; Reflects that cluster membership is uncertain; Cluster assignment can be done based on the highest probability cluster for each observation.

Latent Profile Analysis (LPA): A statistical method used to identify clusters or subgroups within a larger population based on patterns of responses to multiple observed variables

Latent Class Analysis (LCA): A statistical technique used to identify and characterize latent subgroups within a larger population based on patterns of responses to categorical observed variables

Latent Dirichlet Allocation (LDA): A statistical model used in natural language processing to classify text documents into topics

Gaussian Mixture Model parameters: π_k^x: weight that determines the relative cluster size, the weights of each cluster should add up to 1; μ_k: mean that determines the location of the cluster; σ_k: standard deviation that determines the volume of the cluster;

Estimation Maximization (EM-)algorithm: An iterative process to find maximum likelihood estimates of parameters. Steps: 0: Guess the parameters; 1(E): Computes likelihood of datapoint belonging to a cluster; 2(M): Determine the parameters that maximize the likelihood obtained in step 1; Repeat from step 1 until parameters stop changing.

Multivariate Model-Based Clustering: For each cluster: Mean becomes a vector of p means (μ), Standard deviation turns into a p by p variance-covariance matrix ($\sum$). p is the number of variables.

Number of parameters in a (multivariate) Gaussian Mixture Model: The π_k: number of clusters minus 1 (k-1); The μ_k: that is k*p (where p is the number of variables); The $\sum_k$: k*p variances (or p when variances equal of clusters), k*p(p-1)/2 covariances (or p(p-1)/2 when covariances are equal over clusters, or 0 when variables are uncorrelated, spherical clusters);

Model complexity for Multivariate Model-Based Clustering: Cluster shape parameters (the variance-covariance matrix) can be constrained to be equal across clusters, but can also be different

model loss- complexity tradeoff: The tradeoff of how well the clustering models performs versus how complex it is (e.g. the image compression from the lecture with more clusters performed better but is more complex)

Bayesian Information Criterion (BIC): A model fit criterion that can be used to compare different clustering models. BIC = -2 log(ℓ) + m * log(n); Where ℓ is: likelihood p(data|θ), -2 log(ℓ): "Deviance", m: number of parameters, n: number of observations.

Another/Akaike Information Criterion (AIC): Same as BIC but penalty is m

AIC3: Same as AIC but penalty is 2/3m

Integrated Information Criterion (ICL): Same as BIC but reconstruction loss includes the assigned clusters

Mclust identifier for each possible parametrization: E for equal, V for variable, I for identity; Volume, Shape, Orientation (e.g. VVE model has variable volume and shape but equal orientation)

Merging components to get clusters: GMM has trouble with clusters that aren't ellipses. In this case merging can be used. Steps: Start out with the usual Gaussian mixture solution; Merge "similar" components to create non-Gaussian clusters.

Missing data mechanisms

Not Data Dependent (NDD): It is missing for reasons unrelated to the data. Probability to be missing is consistent for all units

Seen Data Dependent (SDD): It is missing for reasons related to data you have got. Probability to be missing depends on observed data

Unseen Data Dependent (UDD): It is missing because of values you would have obtained. Probability to be missing depends on unobserved data

Missing data solutions

Deductive imputation: Values can be derived based on logical or mathematical relationships between variables (e.g. BMI can be derived by only knowing weight and height)

Listwise deletion: Deleting all rows where data is missing. Advantages: simple, unbiased under NDD; Disadvantages: large loss of information, biased under SDD and UDD

Mean imputation: Replacing missing values with the mean of the observed data. Advantages: simple, unbiased for the mean under NDD; Disadvantages: Doesn't work

Regression imputation: Replace missing values by prediction from a regression on the observed data. Advantages: Unbiased estimate of regression coefficients under SDD, Good approximation of the true data if explained variance is high, The better the prediction the better the approximation; Disadvantages: Artificially increases correlations, underestimates prediction error and the uncertainty, p-values to optimistic and confidence intervals too narrow

Stochastic regression imputation: Like regression imputation but adds noise to the predictions to reflect uncertainty. Advantages: Perserves the distribution and correlation; Disadvantages: fiddly, single imputation does not take uncertainty imputed data into account and treats them as real

Last Observation Carried Forward (LOCF): If a datapoint is missing it is replaced by the last observe value.

"Embedded" methods for missing values (model based): Don't impute but the (prediction) model deals with the missing data inside the model itself. Depends on the model; Almost always assumes SDD

Multiple Imputation: Fixes the incorrect estimate of the standard error. Stochastic imputation but create multiple datasets and pool results. The overall estimate is the average of the estimates

Text mining: normalization

Text normalization: Converting text to a more convenient, standard form

Tokenization: Seperating out words from running text

Lemmatization: The task of determining two words have the same root, despite their surface differences. (e.g. the words sings, sang, sung are forms of the verb sing. The word sing is the common lemma, and a lemmatizer maps these words to sing.)

Stemming: Refers to a simpler form of lemmatization in which we mainly just strip suffixes from the end of words

Sentence segmentation: Breaking up a text into individual sentences, using clues like periods or exclamation points.

Case folding: A type of text normalization where everything is mapped to lower case

Stopword removal: Removing common words / removing stopwords from text (e.g. and, the)

Punctuation removal: Removing punctuations from text (e.g. . , !)

Number removal: Removing numbers from text (e.g. "day 3" becomes

"day")
Spell correction: Corrects spelling mistakes in text

Text mining

Regular expressions (regex): A language for specifying text search strings (used for e.g. extracting emails, validating inputs, reformatting dates, etc.)
Document Term Matrix (DTM): One-document-per-row and one-term-per-column. Cells can contain counts, proportions, or scaled proportions.
Zipf's law: Most words occur rarely and only few words occur frequently.
Term Frequency (tf): The proportion of how many times a term appears in a documents. tf = n times term appears in doc x / total amount of terms doc x
Term Frequency-Inverse Document Frequency (tf-idf): Measure of importance of a word to a document in a corpus. tf-idf = tf*idf, tf = n times term appears in doc x / total amount of terms doc x, idf = log(n documents / n documents containing term)
Sentiment analysis: Try to extract and identify positive/negative valence from a text. Basic idea: Sentiment = Total no. positive words Total no. negative words
AFINN lexicon: Assigns words with a score that runs between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment
Bing lexicon: Categorizes words into positive and negative categories
Nrc lexicon: Categorizes words into categories of positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust
Sentiment analysis hurdles: When only investigating one word at the time, qualifiers before a word are not taken into consideration (e.g. I am not happy); Sentiment analysis is language and dictionary dependent; Long texts often contain positive and negative sentiments, which average out to about zero. Hence, more suited for short texts;
Word embeddings: Numerical representations of words in a continuous vector space
Words as vectors: Words represented as vectors, the representations should: Capture semantics (Similar words should be close to each other in the vector space; Relation between two vectors should reflect the relationship between two words); Be efficient (vectors with fewer dimensions are easier to work with); be interpretable.
Cosine similarity: A measure for similarity between two vectors. Can be used to find similarity between words when they are represented as vectors.
Distributional hypothesis: Words that appear in similar context tend to have similar meanings
Self-supervised learning for word meanings: Use the unlabeled text itself as training data for the model (e.g. train a neural network to predict the next word given previous words)
Bag of Words (BoW): A way to represent text data that involves converting a piece of text into a numerical representation, disregarding the sequence and structure of words and considering only their frequency within the document
Continuous Bag of Words (CBoW): Predict a target word from its context, which includes the surrounding words in a sentence or a specific window of words
Skipgram: Predict the context words given a target word.
Analogy property of word embeddings: We can look at analogies in the vector space, for example: king - man + woman = queen
Biases in word embeddings: Biases in word embeddings can appear when the text the model is trained on contains biases. (e.g. a model trained on twitter data associated nurse with women and surgeon with men)

Time Series

Time Series Trend: The long term growth or decline of the series. Trend estimates are often reliable but can be reduced by: irregularity in the original series or abrupt changes.
Time Series Seasonal Variation: A pattern of change that recurs regularly over time. Usually due to differences between seasons.
Time Series Cyclical Variation: Have recurring patterns but with a longer and more erratic time scale compared to seasonal variation. These cycles can be far from regular, impossible to predict length of expansion/contraction periods. (examples: floods, wars, recessions)
Time Series Irregular Variation: Variation over various (usually short) periods. Follows no pattern and is unpredictable. May be linked to events that also occur randomly.
Additive decomposition: $y_t = T_t + S_t + R_t = \hat{T}_t + \hat{S}_t + \hat{R}_t$, T_t=trend, S_t=seasonal, R_t=the remainder
Multiplicative decomposition: $y_t = T_t \times S_t \times R_t = \hat{T}_t \times \hat{S}_t \times \hat{R}_t$, T_t=trend, S_t=seasonal, R_t=the remainder
Semi-average: Fit a straight line to describe the trend by dividing the data into 2 equal ranges, calculating the average for each range, draw a straight line between the points.
Moving average: Take the average of the values within a sliding window
Linear regression: Fit a straight line to describe the trend by least squares method
Exponential Smoothing: $S_x = \alpha y + (1 - \alpha) S_x - 1$, S_x = the smoothed value of x, y = the actual observed value at time x, $S_x - 1$ = the smoothed value previosly calculated for time x-1, α = the smoothing constant (0¡α¡1). When α is small, more weight for past measures. When α is large more weight for the present trend. This method suffers from propogation error.
Compute the seasonal index: Take the average of each period over at least three years, express values of each period as an index by comparing it to the average. Periods can be weeks, months, seasons, etc.
Remove the seasonal component: Devide the actual value by the seasonal index
Seasonal-Trend decomposition using LOESS (STL): A method that decomposes time series into trend, seasonal and remainder which can handle any type of seasonality, the seasonal component is allowed to change over time, the smoothness of the trend is controlled by the user, can be robust to outliers.

Time Series: Forecasting

Naive forecasting: Next period's forecast = previous period's actual. $\hat{y}_{t+1} = y_t$
Simple mean forecasting: Next period's forecast = average of previously observed data
Moving average forecasting: Next period's forecast = simple average of the last k periods
Weighted moving average forecasting: Next period's forecast = weighted average of the last k periods, where each period is assigned a weight
Exponential smoothing forecasting: Next period's forecast = weighted average of the previous reading and the history. $\hat{y}_{t+1} = \alpha y_t + (1 - \alpha)\hat{y}_t$
Forecast accuracy: Tests of forecast accuracy are based on the difference between the forecast of the variables' values at time t and the actual value at the same time point t
Tracking Signal: $TS = \frac{\sum_{t=T_1}^{T} (y_t - \hat{y}_t)}{MAE}$

Data streams

Characteristics of Data Streams: Entire data is not available; Data arrives at a high speed rate; The system cannot store the entire data stream; Huge volume of continuous data; Distribution is non-stationary;
Online learning: Main idea: perform small changes to update the model. Training: use the first batch of data to update the model; Update: Upon arrival of new streams, update the model; Problem: concept drifts.
Sampling data from data streams: Since we cannot store the entire stream, one obvious approach is to store a sample
Sample a fixed proportion of a stream: Each input has a probability to be sampled equal to the proportion (i.e. if we store 1/10 of a stream each input has 1/10 chance to be sampled). Problem: when the stream is infinite, the sample size is also infinite.
Sample a fixed size of a stream: Suppose we need to maintain a random sample S of size exactly s tuples (examples).
Reservoir Sampling: Store the first s elements of the stream to S; When nth element arrives and $n > s$, with probability s/n keep the nth element else discard it; If we picked the nth element, then it replaces one of the s elements in the sample S, picked uniformly at random.
Windowing models for stream processing: Keep the n most recent elements
Sliding window: Upon the arrival of a new item from the stream, discard the oldest item
Tumbling window (disjoint windows): Upon the arrival of a new batch of items of size n, discard the previous batch
Hopping window: Upon the arrival of a new step m of items, discard the oldest m items. When m=1 the hopping window is a sliding window. When m=n the hopping window is a tumbling window.
Exponentially decaying windows: Every item in the stream is important; Recent values are more important; $\sum_{i=0}^{t-1} a_{t-i}(1-c)^i$
Querying Data Streams examples: Filtering, Counting distinct elements, Estimating moments, Finding frequent elements
Filtering Data Steams: Given a set of keys S, Determine which tuples in the stream are in S. Obvious solution: Hash Table. Problem: We may not have enough memory.
Bloom filters: A Bloom filter is a space-efficient probabilistic data structure that is used to test whether an element is a member of a set. The price we pay for efficiency is that it is probabilistic in nature that means, there might be some False Positive results. Probability of a FP: $p_h = (1 - e^{-\frac{kM}{N}})^k$
Counting Distinct Elements of Data Streams: Obvious approach: maintain a hashtable of all the distinct elements seen so far. What if we don't have enough storage for the hash table? Estimate the count.
Flajolet-Martin Approach: Pick a hash function h that maps each of the N elements to at least $log_2(N)$ bits; For each stream element a, let r(a) be the number of trailing 0s in h(a); Record R = the maximum r(a) seen; Estimated number of distinct elements = 2^R;
Hoeffding Tree (HT): When a new sample arrives, the Hoeffding tree compares attributes and decides which attribute to split the data on based on the statistical significance of the observed differences in these attributes. If the differences are not significant enough to confidently choose the best attribute, more data will be collected. This adaptive nature allows the tree to incrementally improve itself as it receives more information.
Very Fast Decision Tree (VFDT): Similar to Hoeffding trees but: near-ties are broken more aggressively, deactivates certain leaves to save memory, poor attributes dropped. Better time and memory.
Clustering Data Streams: Input: Data stream points from metric space; Goal: Find k clusters in the stream (based on k-median algorithm); Constant factor approximation algorithm;
CluStream: Divide the clustering process into online and offline components. Online component: stores summary statistics about the stream data; Offline component: answers various user questions based on the stored summary statistics;

Algorithmic fairness

Limitation of classification accuracy measure: The problem of imbalanced classes (e.g. 2-classes class 1 has 99 observations and class 2 has 1 observation, if the model predicts everything as class 1 it has a 99% accuracy)
Methods to overcome imbalanced data: Collect more data; Delete data from majority class; Create synthetic data; Adapt your learning algorithm;
Random oversampling: Randomly duplicate datapoints from the minority class. Problem: overfitting and fixed boundaries.
Random under-sampling: Randomly delete datapoints from the majority class. Problem: loss of information
Synthetic Minority Over-Sampling Technique (SMOTE): Create new datapoints from the minority class by selecting a random datapoint along the line between two known datapoints of the minority class. Problem: might introduce the artificial minority class examples too deeply in the majority class space.
Generative Adversarial Networks (GANs): System of two neural networks competing against each other, a generator and a discriminator. Generator improves based on the feedback of the discriminator.
InfoGAN: Approximate the data distribution and learn interpretable, useful vector representations of data.
Conditional GANs (CTGAN): Able to generate samples taking into account external information (class label, text, another image). Force G to generate a particular type of output.
Failure cases of GANs: The discriminator becomes too strong too quickly and the generator ends up not learning anything; The generator only learns very specific weaknesses of the discriminator; The generator learns only a very small subset of the true data distribution;
Cost-sensitive classification: False positive and false negative can have different consequences or costs. Cost-sensitive classification takes this into account and can apply different thresholds for probability of classifying something as positive or negative.
Bias in ML algorithms: The bias is not related to the way the algorithm is built but the bias stems from the training data.
Bias in data: Historical bias in the decision variable; Limited / less informative features; Biased data collection; Imbalanced representation of different demographic groups;
Fairness: "Fairness is the absence of any prejudice or favoritism towards an individual or a group based on their intrinsic or acquired traits in the decision-making context"
Equality: Everyone benefits from the same supports.
Equity: Everyone gets the support they need.
Justice: The cause(s) of the inequity was adressed.
Demographic Parity (DP) Difference: The instances in both protected (unprivileged) and unprotected (privileged) groups should have equal probability to receive positive outcomes. This measure takes values between 0 and 1 where 0 is the optimal.
Disparate Impact (DI) Ratio: The ratio between the probability of protected and unprotected groups getting positive or desired outcomes.A dataset or a classifier is considered fair (by law) if its DI-ratio is between 0.8 and 1.25 (1 is the optimal)
Consistency (fairness measure): An individual fairness measure determines how similar the labels are for the similar instances in a dataset based on the k-neighbors of the instance. This measure: takes values between 0 and 1 with 1 is the optimal.
Equalized Odds (EO) Difference: EO states that instances from privileged and unprivileged groups should have equal True Positive Rate (TPR) and False Positive Rate (FPR).
Predictive Parity (PP): A classifier is fair in term of predictive parity if the prob. that an example is positive in the original dataset given that it is predicted positive from both privileged and unprivileged groups is the same.
Mitigation Algorithms – Pre-Processing Techniques: Fairness through "unawareness": deletes the sensitive attributes in a dataset; Preferential sampling (re-sampling): data objects are sampled with replacement; Massaging (Relabeling): changes the actual class label of some of the instances in the training set; Reweighing: assigns weights to each instance in the training set;
Mitigation Algorithms – In-Processing Techniques: Adjust/tune the classification algorithm; Applied during the model training; Regularization of the model; Dependent on the implemented classifier;
Mitigation Algorithms – Post-Processing Techniques: Eliminate the discrimination from the final predictions; Change the predicted outcomes of classifiers; Changes are based on certain rules or constraints such as equalized odds; Thresholding the critical regions;

Data Wrangling & Data Analysis final Cheat Sheet - Created by Tim Rentenaar