

Questions to the topic Classification

Describe the task of classification. (1 point)

Classification: assigning each data point a class from a finite set of possible classes.

Describe linear classification models. (1 point)

Linear combination of features -> scalar score -> finding a division point

Analogically: finding hyperplane(s) that separate individual classes

Explain how perceptron classifiers work. (2 points)

Essentially similarity to a learned vector that is perpendicular to the separating hyperplane.

Learning via gradient descent. Sigm(x) of similarity -> result is interpretable as probability.

result = (np.dot(b[i], w) > 0)

if result != targets[i]:

 w = w - lr * (w - b[i]) * (targets[i] - result) # w -= lr * (w-b[i]) * errorPolarity

Explain how Naive Bayes works. (2 points)

Naive: assumes features are conditionally independent of each other given target class

$P(e_1, e_2, \dots | ck) = \prod_i P(e_i | ck)$

argmax_ck: $P(ck | e_1, e_2, e_3) \sim P(ck) * P(e_1, e_2, \dots | ck) / P(e_1, e_2, \dots) \sim P(ck) * \prod_i P(e_i | ck)$

Explain how KNN works. How would you choose the value of K? (3 points)

Predicted class is (weighted) average of k elements from training that are closest (in some pre-chosen metric) to current element. Depends on how dense / noised the data are (affected by dimensionality). More dense & more noise -> larger k -> more stable average.

Explain how decision trees work. Give examples of stopping criteria. (3 points)

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).

E.g. max depth, number of training elements in a node, information gain ($IG(T,a) = H(T) - H(T|a)$), purity (once the labels in given node are pure enough, we stop)

Explain how SVM works. (3 points)

One separating boundary located in the middle of two support vectors. Given point is assigned to a class based on the position towards separating boundary. The points that fall into the middle ground are penalized and ideally no points are in the middle.

Compare pros and cons of any 2 classification methods which you choose out of the following 6 methods: perceptron, Naive Bayes, KNN, decision trees, SVM, Logistic Regression (3 points)

Perceptron:

- + converges
- + pretty fast

- + easy to implement
- suitable only for linearly separable data
- binary classification
- is sensitive to not-uniformly-scaled data

Naive Bayes:

- + fast
- + multiple classes classification
- doesn't really reflect real world
- is problematic with non-categorical data (required binning / distribution assumption)

KNN:

- + online (don't need to retrain with new data)
- + easy to implement
- problem with categorical data - how do you measure a distance between *red* and *green*?
- memory consumption
- heavily sensitive to non-scaled data
- sensitive to higher dimensionality (curse of dimensionality)

Decision tree:

- + interpretable (theoretically)
- + almost no hyperparameters that would significantly affect training stability ...
- + absolutely stable learning, small data consumption, fast to execute, not so slow to learn
- + doesn't actually require any data preprocessing - the nodes can handle pretty much anything
- + doesn't care about features' scale
- can overfit easily (forest is usually required for robustness)
- dealing with non-categorical data can be complex
- can't capture more complex relationships in single tree variant

SVM:

- + natively regularized
- + robust
- + low memory consumption (three points + a bit)
- binary classification

Log. regression:

- + Essentially a perceptron with sigmoid at the end
- + Iterative, converges, ...
- Small derivation further from 0

Explain the notion of separation boundary. (1 point)

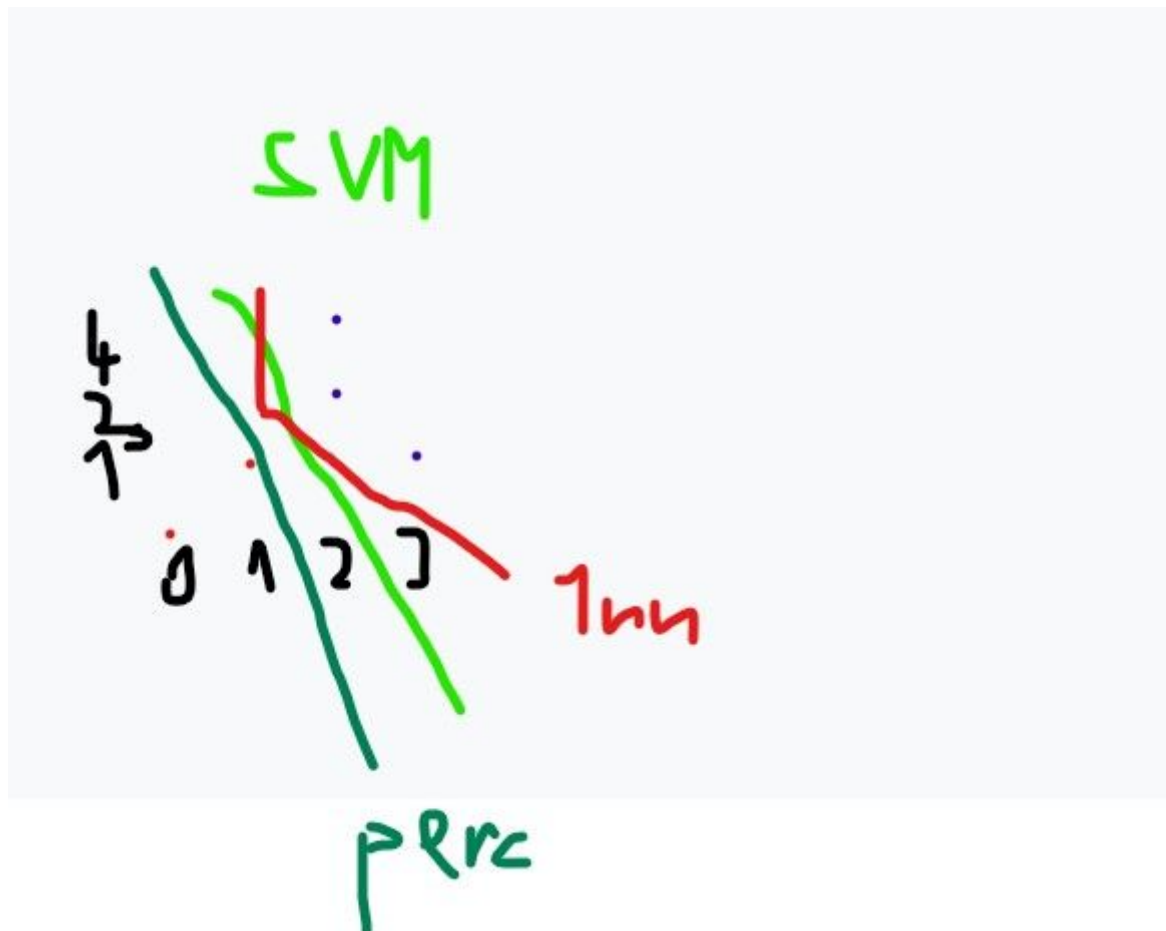
it's a boundary where movement in a specific direction of any magnitude results in change of predicted class.

~~In a feature space the separation boundary is a hyperplane separating the feature space into two affine spaces.~~

What does it mean that a classification dataset is linearly separable. (1 point)

That there exist a separating hyperplane, that would separate the dataset. So that only one target value is at one affine space, and the rest is at the second affine space.

Assume there are two classes of points in 2D in the training data: $A = \{(0,0), (1,1)\}$ and $B = \{(3,1), (2,3), (2,4)\}$. Could you sketch separation boundaries that would be found by (a) SVM, (b) perceptron, (c) 1-NN? (3 points)



How would you use a binary classifier for a multiclass classification task? (2 points)

N classifiers, the one with - essentially one-hot encoding. The one with highest "score" wins. n^2 , each against each, some sort of voting system.

One vs Rest(=All) - highest score wins, One vs One - voting

Give two examples of "native multiclass" classifiers and two examples of "native binary" classifiers. (3 points)

Native binary: SVM, perceptron, logistic regression

Native multi-class: Naive Bayes, KNN, Decision tree

What would you do if you are supposed to solve a classification task whose separation boundary is clearly non-linear (e.g. you know that it's ball-shaped)? (2 points)

Kernel methods, data augmentation (create combination of data, transform them, ...)

Use methods that capture non-linearity out of box: weighted knn, DNNs,

Questions to the topic Regression

Describe the task of regression. (1 point)

Predicting some a real/natural numbered variable based on variety of features.

Describe the model of linear regression. (1 point)

Dot product with well scaled similarity vector / orthogonal projection of target class vector to feature vector space / linear combination of features producing target variable.

Describe any optimization criterion that could be used for estimating parameters of a linear regression model. (2 points)

~~Gradient descent ($w = w + lr * b[i] * (targets[i] - np.dot(b[i], w))$)~~

~~Aforementioned orthogonal projection~~

Mean square error, mean error, minimizing maximal error...

Describe any technique (in the sense of a search algorithm) that could be used for estimating parameters of a linear regression model if the least squares optimization criterion is used. (2 points)

Gradient descent ($w = w + lr * b[i] * (targets[i] - np.dot(b[i], w))$)

Aforementioned orthogonal projection

What would you do if the dependence of the predicted variable is clearly non-linear w.r.t. the input features? (2 points)

Kernels / feature engineering. Create new features as polynomials of original features (known as polynomial regression).

Describe situations in which using the least squares criterion might be misleading (2 points)

Data with far outliers. Since the criterion is quadratic it can be dominated by (even few) very far outliers.

Whenever the noise is obviously not normal distributed (E.g. skewed to one side).

Generally when data are skewed in one dimension. Then the distance criterion must be skewed as well. Least squares method natively is not skewed and perceives all dimension with the same significance.

What are the main similarities and differences between the tasks of regression and classification? Is logistic regression a regression technique or a classification technique? (3 points)

Both are supervised - we need golden data from which we can learn.

Every regression method can be used for classification with addition of boundary value & e.g. sigmoid to enable interpreting distance from boundary as probability. We can also use classification methods for regression into a finite number of values / to finite (training set-defined) regions (in case of average knn e.g.). Classification methods give us less information (one of finite vs one in continuous), however, and so there are obvious limits to going this direction.

Logistic regression is used mainly as a classification technique.

Questions to the topic Clustering

Describe how K-means works. (2 points)

Iterative algorithm to find k-clusters within data. Assigns random cluster centers (random, first k-elements, ...) and then iteratively in each epoch assigns each point to its closest cluster (closest centroid), recomputes clusters' centers and repeats. Iteratively groups of points that are (within the metric) closest to each other (the clusters' centers gravitate towards centers of gravity).

How would you initialize the centroid vectors at the beginning of K-Means. (1 point)

Random, first k-elements, centers of some random sub-selections, ..., extremes (e.g. in case of n-dim data and $k < n$ extremes in each dimension), ...

Describe how Mixture of Gaussians works. (3 points)

Generalization of k-means -- k-means can be viewed as fitting "multidimensional spherical" clusters to the data and represented by a cluster mean (centroid, each cluster has the same diagonal covariance matrix), whereas Gaussian Mixture Model (GMM) models clusters as multivariate Gaussian distributions characterized by mean and variance (general covariance matrix). Then, each data point (even previously not seen) \mathbf{x} can be modelled as a weighted combination of fitted \mathbf{k} clusters -- thus providing soft assignment to several clusters (hard one as in k-means can be set as argmax).

Linear super-position of Gaussians

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)$$

Number of Gaussians
Mixing coefficient: weightage for each Gaussian dist.

We want to maximize log likelihood with respect to weights π_i , means and variances (no closed solution -> EM algorithm):

$$\ln p(\mathbf{X} | \mu, \Sigma, \pi) = \sum_{n=1}^N \ln p(\mathbf{x}_n) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right\}$$

EM in general (introducing latent variable Z):

$$\ln p(\mathbf{X} | \Theta) = \ln \left\{ \sum_z p(\mathbf{X}, Z | \Theta) \right\}$$

Equation 1: Marginal Likelihood with Latent variables

In GMM we use EM by introducing latent variable gamma that can view as a probability that a data point belongs to cluster k:

$$\gamma_k(\mathbf{x}) = p(\mathbf{k} | \mathbf{x}) = \frac{p(\mathbf{k})p(\mathbf{x} | \mathbf{k})}{p(\mathbf{x})}$$

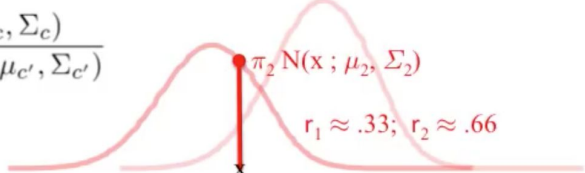
$$= \frac{\pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x} | \mu_j, \Sigma_j)}$$

Latent Variable

From other source (gamma is r_{ic} , k is c):

E-step ("Expectation")

- For each datum (example) x_i ,
- Compute " r_{ic} ", the probability that it belongs to cluster c
 - Compute its probability under model c
 - Normalize to sum to one (over clusters c)

$$r_{ic} = \frac{\pi_c \mathcal{N}(x_i; \mu_c, \Sigma_c)}{\sum_{c'} \pi_{c'} \mathcal{N}(x_i; \mu_{c'}, \Sigma_{c'})}$$


$r_1 \approx .33; r_2 \approx .66$

M-step ("Maximization")

- For each cluster (Gaussian) $z = c$,
- Update its parameters using the (weighted) data points

$$m_c = \sum_i r_{ic} \quad \text{Total responsibility allocated to cluster c}$$

$$\pi_c = \frac{m_c}{n} \quad \text{Fraction of total assigned to cluster c}$$

$$\mu_c = \frac{1}{m_c} \sum_i r_{ic} x^{(i)} \quad \Sigma_c = \frac{1}{m_c} \sum_i r_{ic} (x^{(i)} - \mu_c)^T (x^{(i)} - \mu_c)$$

Weighted mean of assigned data

Weighted covariance of assigned data
(use new weighted means here)

Describe pros and cons of K-Means in comparison to Mixture of Gaussians. (3 points)

K-Means:

- + Convergence to a local optima ensured
- + easy to implement
- + easy to explain (Euclidean distance + cluster variance)
- K-means often doesn't work when clusters are not round shaped w.r.t. given metrics
- Data point is deterministically assigned to one and only one cluster
- Changing or rescaling the dataset either through normalization or standardization will completely change the final results
- Falling into local maximum
- Deciding K

GMM

- + More general than K-means
- + Scaling / rotation independent
- Long computation time
- Falling into local maximum
- Deciding K
- when one has insufficiently many points per mixture, estimating the covariance matrices becomes difficult, and the algorithm is known to diverge and find solutions with infinite likelihood unless one regularizes the covariances artificially.

Give an example of a hierarchical clustering method, and sketch a set of datapoints for which it would lead to a better clustering result compared to that of KNN or MoG. (2 points)

Essentially the same as algs for minimum spanning tree (weight is distance of a tree's center to another one / to closest element), just stop sooner before getting just one tree (one cluster). E.g. one where there are two spirals whirled together.

What is a dendrogram (in the context of clustering)? How can it be created? (2 points)

Graph that shows how individual points sub-clusters were connected together to form bigger ones. Used for hierarchical clustering methods. It can be created by recording the order in which clusters were created.

Questions to the topic Evaluation and diagnostics

Explain what is learning curve and how it can be used. (3 points)

Intuition and some internet searches:

Learning curve shows the model performance in time. Where performance could be pretty much anything we want to optimize - objective function, accuracy, error rate...

- Can be used to estimate best stopping criterion ("the curve has not nudged for 1000 epochs, better stop wasting time", "the model diverges!", etc...)
- Compare two models in terms of convergence and or performance

What they probably want to hear:

Compare performance of one model on dataset with increasing number of training patterns. By comparing the LC of model trained on small number of datapoints to the same model trained on big number of datapoints we can estimate, whether our data have big variance and or bias. More [here](#)

Why does it make sense to evaluate performance both on the training data and on held-out data? (2 points)

Because the model could overfit and train particular training data instead of general ideas -> could perform well on the set it was trained on but not be able to generalize very well.

What is ablative analysis? (1 point)

Removing parts of the model / input data / preprocessing & comparing the result to baseline to see how much the individual removed part contributed to the result. Used to measure influence of individual components & can be useful for regularization via Occam's razor.

Illustrate underfit/overfit problems in regression by fitting a polynomial function to a sequence of points in 2D. (2 points)

Under: Just line for e.g. sinusoid

Over: Number of points + few outliers, function that goes through all of them perfectly

Illustrate underfit/overfit problems in classification by modeling two (partially overlapping) classes of points in 2D. (2 points)

Under: Just line

Over: 1KNN decision boundary

What problem is typically signalled by test error being much higher than training error? (2 points)

Overfitting: the model learns the training data directly (due to being too strong / having too many parameters / learning for too long / not enough regularization). Can't generalize about the data. Learns answers for individual training data points directly, instead.

How would you compare the quality of two classification models used for a given task? (1 point)

Accuracy, recall, RoI curves (false positives vs true positives).

Learning curve

How would you compare the quality of two regression models used for a given task? (3 points)

Least square methods, (cross validation)

comparing optimization criterion

Using Occam's razor - learning time, no. hyperparameters...

Learning curve

How would you compare the quality of two clustering models used for a given task, if some gold classification data exist? (3 points)

~~Since there are golden data - accuracy of classification/clustering of the golden data - same as classification.~~

Purity of clusters - no. of different (wrt clusters) golden data points within a predicted cluster.

How would you compare the quality of two clustering models used for a given task, if no gold classification data exist? (3 points)

Inner distances of clusters - cluster whose two most distant points are very far apart is probably not that good.

Avg distance of cluster points to its centroid (once again - high is bad)

The ratio between the minimal inter-cluster distance (distance between clusters) to maximal intra-cluster distance ("the size of cluster" - distance of two furthest apart clusters).

Possibly some combination - using distance of centroids.

What would you use as a baseline in a classification task? (1 point)

Constant model. Either completely: chooses the most frequent class or random with probabilities proportional to how each class is proportional in the data (both the same expected accuracy, ...)

What would you do when your Naive Bayes classifier runs into troubles because of sharp zero probability estimates? (2 points)

Smoothing: add some epsilon non-zero but insignificant number to all / zero probabilities.

What would you do if it turns out that your decision tree classifier clearly overfits the given training data? (2 points)

More aggressive stopping criterion, shallower max depth, larger min information gain, larger min set for a node. Potentially a forest of multiple decision trees instead of just one to induce some regularization.

Questions to the topic Feature engineering and regularization

Explain the main idea of the kernel trick, give an example. (2 points)

[wiki](#) and an [example](#)

Create new artificial features that represents non linear dependencies between original features in linear way. So the original data were not linearly separable, but transformation to a different feature space made them so.

Other interpretation: Transform initial features via some vector function to a higher dimensional space hoping that in such space the initial, linearly non-separable, data become linearly separable. Nice property of this approach is that in case of optimizing dot product of feature vectors transformed via some function ϕ to a higher dimensional space to $\phi(x_1) \cdot \phi(x_2)$, this can be written as a kernel function $K(x_1, x_2) = \phi(x_1) \cdot \phi(x_2)$. We can imagine that a kernel function can exist on its own, without knowing ϕ explicitly. So, we can convert process of feature transformation and the dot product into a simpler form of such process encoded in kernel (imagine mathematically simplified $\phi(x_1) \cdot \phi(x_2)$). Kernel represents a similarity measure that is used as a cheap measure of unseen example towards train data.

Explain what is regularization in ML used for, give an example of a regularization term. (3 points)

Technique that tries to combat overfitting, usually through assuming Occam's razor (simpler model is more probable). Can be done in various ways: penalizing large weights (L2), penalizing non-zero weights (L1) (essentially add weights sum to loss function that gets SGDed), through smaller k in KNN, early stopping / stricter early stopping criteria (perc, dec trees), enhancing data through noise, dropout (NN, perc), ...

What are pros and cons of selecting features by a simple greedy search such as gradual growing the feature set from an empty set (forward selection) or gradual removing features from the full set (backward selection)? (2 points)

- + fast (compared to other approaches)
- + greedy part is easy to implement

- + customizable
- + natively leads to creating ensemble of models (e.g. good for random forests, ADABOOST, ..)
- very naive - doesn't take into account relations between features
- has to re-evaluate all features after every removal
- doesn't create new features (which could be useful sometimes)
- can't spot correlations - two features could be important together, but not alone

What is the main difference between regularization and feature selection? (3 points)

Feature selection is data specific whereas regularization is model specific.

feature selection is used for reduction dimensionality of features given to data, mainly for avoiding the curse of dimensionality(speed up training, using only relevant features...).

Remove NA's, remove redundant features, combine features (not sure about this one)

regularization prevent model overfitting using all the features. Clipping outliers, normalization etc...

What is the typical way how a regularization factor is employed during training? (2 points)

Regularization factor modifies the objective function (the one being optimised).

Can be done in various ways: penalizing large weights (L2), penalizing non-zero weights (L1) (essentially add weights sum to loss function that gets SGDed), through smaller k in KNN, early stopping / stricter early stopping criteria (perc, dec trees), enhancing data through noise, dropout (NN, perc), ...

If you were supposed to reduce the number of features by selecting the most promising ones, how would you do the selection? (2 points)

- Removing features with variance lower than threshold
- Select features with the highest score:
 - F-value
 - between target and feature
 -

PCA, Correlation with the target value, F-value, removing low variance, Using L1-regularization, mutual information (doesn't select the most promising, but can remove the redundant ones), Generalized Low Rank Model

What would you do if your ML algorithm accepts only discrete features but some of your features come from a continuous domain (are real-valued)? (2 points)

Bin the values to intervals. Intervals could be interpreted as categorical features. Different methods of binning: each interval the same size / same number of included elements, ...

Using Mixture of gaussians and then individual categories represent distinct gaussians, ...

What would you do if your ML algorithm accepts only binary features but some of your features can have multiple values? (1 point)

One hot encoding (n classes: n-arry binary vector).

Give an example of a situation in which feature rescaling is needed, and an example of a rescaling method. (2 points)

- SVM, K-NN, PCA require features to be normalized.
 - PCA - looking for component that maximize the variance. e.g. weight in kg, height in m, weight varies less than height because of their scales. PCA might determine that direction of maximal variance corresponds to weight feature.
 - K-NN - consider features with smaller scale as more important without scaling
 - SVM - All kernel methods are based on distance
- Standardization(z-score standardization) $x' = \frac{x - \text{mean}(x)}{\sigma}$, Mean Normalization $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$

What is feature standardization? (2 points)

Part of preprocessing, rescaling the features that they have properties a standard normal distribution $N(0, 1)$.

$x' = \frac{x - \text{mean}(x)}{\sigma}$ mean(x) of that feature vector, σ is standard deviation.