AML HS19

jasonf

September 2019

Contents

1 Introduction

The Learning Problem of Pattern Recognition

- Representation of Objects: Data Representation.
- Definition/Modeling of structure: What is a pattern?
- Optimization: Search for preferred structures.
- Validation: are the structures indeed in the data or are they explained by fluctuations?

Feature Space

- Measurement space X: mathematical space in which data is represented (e.g. numerical R, Boolean, Categorical)
- Features: derived quantities (edges, corners, motion vectors,...)

  Remark: The selection of a specific feature space predetermines the metric to compare data.

Typical learning problems

- Classification: learning an indicator function.
- Regression: learning a real valued function.
- Density estimation: learning a probability density of the data source.
- Dimension reduction: learning a linear or nonlinear projection.
- Data compression: learning a coding efficient representation.

Key Concepts of Machine Learning:

- Trade off training error vs. model complexity

- Regularizers prevent overfitting.
- Hyper parameter/model selection via cross-validation.

- Corresponodence prob. modeling: Loss = Likelihood, regularizer = prior
- Kernel trick: replace inner products by kernel function.
- Neural nets for feature learning.
- Discriminative vs. generative models
  → Discrim.: Learn function $f : X \to Y$
  → Genr.: Learn joint distribution $P(X = x, Y = y)$
- Unsupervised learning as latent variable modeling (clustering = classification, dim. Reduction = regression) → Training by EM algorithm.

2 Representations

Measurements and Data Patterns, Data Types, Transformations, Scale

Loss: $Q(y, f(x))$ (0-1 loss, quadratic loss,...)
Conditional Expected Risk: $R(f, X) = \int_y Q(Y, f(x))P(Y|X)dY$ (r. var X)
Total Expected Risk: $R(f) = \int_X R(f, X)P(X)dX$ (r. var X,Y)
Empirical Test Error: $\hat{R}(\hat{f}, Z^{train}) = 1/n \sum_{i=0}^{n} Q(Y_i, \hat{f}(X_i))$ (tries → $Exp.Risk$)
Expected Risk: $\mathrm{E}_X[R(\hat{f}, X)]$

- **Taxonomy of Data:**
  - **Pattern analysis:** Requires to find structure in sets of object representations.
  - **Object space:** We are given a design/configuration/object space O!
  - **Measurement:** Given an object set, am measurement X maps an object set into a domain K. e.g. objects → feature vectors.
- **Examples of Data:**
  - **Monadic Data:** $X : O \to \mathbb{R}^d, \quad o \to X_0$
    characterize configuration or objects without reference to other configurations. (temperature, depth,...)
  - **Dyadic Data:** $X : O^{(1)} \times O^{(2)} \to \mathbb{R}, \quad (o_1, o_2) \to X_{o1,o2}$
    {users} X {websites}
  - **Pairwise Data:** $O \times O \to \mathbb{R}, \quad (o_1, o_2) \to X_{o1,o2}$ Similar to dyadic but of same kind: {proteins} X {proteins}
  - **Polyadic Data:** $O^{(1)} \times O^{(2)} \times O^{(3)} \to \mathbb{R}, \quad (o_1, o_2, o_3) \to X_{o1,o2,o3}$
    {test persons} X {behaviors} X {traits}

- **Scales:**
  - **Nominal or Cateforical scale:** Qualitative but not quantitative measurements, choice of categories.
  - **Ordinal scale:** measurement values are meaningful only with respect to others (e.g. ranking)
  - **Quantitative scale:**
    * Interval scale: the relation of numerical differences carries the information (Celsius scale).
    * Ratio scale: zero value of the scale carries info but nor the measurment unit (Kelvin scale).
    * Absolute scale: Absolute values are meaningful (school grades).

    ⚠ **Data Whitening:** Normalize the values of a feature vector by the standard deviation. Thereby differences in dynamic range are eliminated.
- **Mathematical Spaces:**
  - Topological spaces:
  - Metric Space:
  - Euclidean Vector space:
- **Probability Spaces:** TODO

3 Density Estimation in Regression: Parametric Models

Maximum Likelihood Method, Efficient Estimators, Bayesian Learning (batch/online)

3.1 Bayesianism

Bayesian view: Both the observations(feature vector X and output variable Y) and the parameters θ of the regression model are random variables!

Parametric Statistics: the functional form of the Likelihood $P(X,Y|\theta)$ is given; we want to estimate the params θ of the likelihood P(data|model).

Non-Parametric Statistics: we sample X, Y to estimate the likelihood.

Statistical Learning Theory: we minimize the empirical risk $min_{f\in C}\hat{R}(f,Z^{train})$ directly without estimating the likelihood.

Bayes Terminology:

prior: P(model) acts as Regularization, initial guess

likelihood: P(data|model)

posterior: P(model|data)

evidence: P(data)

→ **Bayes Rule:** $P(model|data) = \frac{P(data|model)*(P(model)}{P(data)}$

3.2 Frequentism

Maximum likelihood method: $\hat{\theta} \in argmax_\theta P(X|\theta)$

1. Define a parametric model (model depends on parameter).
2. Define the likelihood as a function of the parametric model.
3. Compute an estimator by maximizing the likelihood (i.e. find best params for parametric model). → find extremum of the log-likelihood function

$$\Lambda(\theta) \stackrel{\triangle}{=} \nabla_\theta \log \mathbf{P}(\mathcal{X}|\theta) = \frac{\partial}{\partial\theta} \sum_{i \leq n} \log p(x_i|\theta) = 0$$

Features of ML-Estimators

- **Consistent:** $\theta_{ML} \to \theta_0$ in probability as n $\to \infty$.
- **Asymptotically normal:** $1/\sqrt{n}\ (\theta_{ML} - \theta_0)$ converges in distribution to a random variable.
- **Asymptotically efficient:** θ_{ML} minimizes $\mathbb{E}[(\theta_{ML} - \theta_0)^2$ as $n \to \infty$
- **Equivariance:** If $\hat{\theta}_n$ is MLE of θ the $g(\hat{\theta}_n)$ is MLE of $g(\theta)$

Cramer-Rao bound: No estimator reaches expected mean squared error = 0. $\mathbb{E}[(\theta_{ML} - \theta_0)^2 >= \frac{1}{I_n(\theta_0}$ where I_n is the Fisher information. But ML-Estimator is at least asymptotically efficient (reaches bount) (but not necessarily for finite samples!!)

3.3 Bayesianism vs Frequentism

Bayesianism	Frequentism
-Allows priors -Provides a distribution when estimating parameters -Requires efficient integrastion methods, when computing posteriors. -The prior often induces a regularization term.	-Does not allow priors -Provides a single-point when estimating parameters -Requires only differentiation methods, when estimating parameters. -MLE estimators are consistent, equivariant,...

Bias: $bias(\hat{\theta}_n) = \mathbb{E}[\hat{\theta}_n] - \theta$ The bias measures how much the expected value of the estimator deviates from the true parameter value.

3.4 Bayesian Learning

TODO slides 3 p.37

3.5 Recursive Bayesian Learning

4 Regression

Optimal solution of regression problem: $argmin_f \quad \mathbb{E}(Y - f(X))^2$
$$f^*(x) = \mathbb{E}(Y|X = x)$$

- Maximum likelihood: maximize probability of data as sampled.
- Statistical learning theory: Minimize directly empirical risk.

→ both approaches lead to same solution.

4.1 Linear Regression

$$Y = \beta_0 + \sum_{j=1}^{d} X_j \beta_j \tag{1}$$

β_0 is called bias

Residual Sum of Squares (RSS)

$$RSS(\beta) = \sum_{i=1}^{n} (y_i - x_i^T \beta)^2 \tag{2}$$

For given data minimize the RSS

Optimality of Least Squares Estimate: The least squares estimate of the parameter β has the smallest variance among all linear unbiased estimates.

Gauss Markov Theorem: For any linear estimator $\theta = c^T y$, that is unbiased for $a^T \beta$, it holds that variance is not smaller then the one of least squares estimate:

$$\mathbb{V}(a^T \hat{\beta}) \quad \leq \quad \mathbb{V}(c^T y) \tag{3}$$

- Overfitting can be a problem!
- Bias-variance trade off:
  mean squared error = $bias^2$ + variance + noise variance

→ Hence $\hat{f}(x)$ is best among all unbiased linear models in the sense of minimizing the MSE. But Goal is to minimize generalization error.

- Option: trade bias increase for variance reduction.
- Goal: Minimize bias and variance simultaneously.

Bias vs. Variance

- **bias:** high-bias learns fast but is less flexible, more assumptions on data.
- **variance:** high variance means for different data estimator would change a lot.
- overfitting: low bias, high variance (small data set, large model complexity) → add Regularization, cross validation, ensembles of classifiers
- underfitting: high bias, low variance (large data set, small model complexity)

4.2 Regularized Linear Regression:

Regularization = Bayesian Maximum A Posteriori (MAP) estimates.

- Ridge Regression: add $\lambda\beta^T\beta$ to the squared loss. Suppression of contributions by small eigenvalues → built in model selection.
- Lasso: add $\lambda||\beta^T||_1$ to the squared loss. In addition to ridge some coefficiants are sparse (Least Absolute Shrinkage and Selection Operator).

4.3 Nonlinear Regression

Basis Expansion: Transform the variables X nonlinearly and fit a linear model in the resulting feature space. (e.g. take cubix splines of X).

Regression with Wavelets:

5 Gaussian Processes for Regression

Bayesian Linear Regression: extends multiple linear regression by defining a prior over the regression coefficients, for example ridge regression. Given observed data we can now use Bayes theorem to obtain the posterior distribution over coefficients.

From Bayesian linear regression to Gaussian processes Since the outputs y are a linear combination of normally distributed random variables β, they are jointly Gaussian themselves.

Gaussian processes use Covariance matrix as kernel. The kernel function on the previous slide expresses that the outputs of two points whose inputs are 'similar' to each other have a high covariance. By contrast, the outputs of points with "dissimilar" inputs have a low covariance. The level set of the joint distribution yields an axis aligned ellipse. Kernel functions specify the degree of similarity between any two data points.

Recall: kernel properties

- Symmetry $k(x, x') = k(x', x)$
- Positive semi-definiteness (continuous case). Kernel matrix K must be positive semidefinite. $x^T K x >= 0$ for all x.

Different kernels have different invariance properties e.g. invariant to rotation, translation.

Kernel engineering

Ensembles Average models with different parameters. Unbiased estimators remain ubiased after averaging. But we can reduce variance by a factor of B (nr. of models).

6 Numerical Estimation Techniques

a. Cross-Validation:

b. Bootstrap:

c. Jackknife: Method to compensate for systematic estimation errors (bias reduction)

6.1 K-Fold cross validation

Problem with K-Fold prediction quality is determined for a model which has been trained on approximately n(K - K)=K data; there exists a systematic tendency to underfit since the adapted model is not as complex as it could have been using the full data set. Leave-one out k-fold on the other hand has usually great variance. Engineers solution choose k = $min\sqrt{n}, 10$

Note that it is important to calculate the final prediction error on data which have not been used in model fitting (training) nor in model selection (testing for parameter adaptation).

6.2 Bootstrapping

Resampling with Replacement of the training data yields different bootstrap sample sets; numerical calculation of the estimation error by the empirical distribution.

When does Bootstrap work? if the deviation between empirical and bootstrap estimator converges in probability to the deviation between true parameter value and the empirical estimator. But Bootstrap estimation of classification error is too optimistic - Problem of overlap of training and test set.

Improvement of plain bootstrap:

- Leave-one-out bootstrap: dont use on datapoint, same problems as in CV.
- .632 Bootstrap:

6.3 Sketching distributions:

How can we encode uncertainty of distributions?

- Moment methods: CV and bootstap provide numerical techniques to estimate moments of some statistic.
- Graphical sketch: Box plotes sketch a distribution.
- Density estimation: The most information of data is captured by its probability distribution.

6.4 Jackknife Method

Numerical estimate of the bias of an estimator.

idea:

- Use the leave-one-out estimator S_{n-1} to estimate the bias of S_n
- Subtract the bias from the estimator.
- But Bias corrected estimators can have a considerably larger variance.

7 Model Selection

7.1 Neyman-Pearson Test

???

7.2 Complexity-Based Model Selection

Add a complexity term to the goodness of fit term. Choose the simplest model that explains the data. E.g. Negative log-likelihood usually decreases with increasing model complexity. Correct this tendency to select complex models with a complexity penalty.

7.3 Bayesian Information Criterion (BIC)

not covered

7.4 Minimum Description Length

not covered

7.5 Akaike Information criterion (AIC)

approximates the Kullback-Leibler (KL) divergence between the true model and the estimate. AIX is asymptotically equivalent to leave-one-out cross validation for ordinary linear regression models

not covered

7.6 Takeuchi Information Criterion (TIC)

not covered

8 Design of Linear Discriminant Functions

The problem of statistical decisions: n objects should be grouped in the classes 1,..,k and a doubt class D and a outlier class O.

Classification Error: The indicator function $\mathbb{I}_{\{\hat{c}(x)=y\}}$ counts errors.
Expected errors are also called the **expected risk**

Loss function:

- Weighted mistakes: when classification errors are not equally costly or if we have class imbalance.
- Loss function $L(y,z)$: denotes the loss for decision z if class y is correct.

Different Loss functions

- 0-1 Loss: all classes are treated the same. We assume constant costs for missclassifications (and loss = d if int doubt). Conditional risk function of the classifier is the expected loss of class y (probability of misclassif. + probability of doubt)
- Surrogate Loss: 0-1 loss is non-convex → use exponential loss, logistic loss or hinge loss function.

9 Linear Classifier

simple and computationally efficient.

Linear Discriminant Function:

$$g(x) = w_0 + \sum_{j<=d} w_j x_j = (w_0, w)(1, x)^T =: a^T x' \tag{4}$$

Quadratic Discriminant Function

$$g(x) = w_0 + \sum_{j>=d} w_j x_j + \sum_{j<=d} \sum_{l>=d} w_{jl} x_j x_l \tag{5}$$

Linear discriminant with nonlinear feature transformation Use a linear classifier in a high dimensional feature space, generated by a non-linear transformation of feature vectors.

Homogeneous Coordinates $x' = (1, x)^T$, $a = (w_0, w)^T$
Simplifies the notation and the analysis substantially.

non unique solutions: Introduce a margin b to classify data with a safe distance from decision boundary i.e. $a^T x_i >=$
Regularization of classifier by margin b!

Algorithms for learning the Weight Vector:

- Gradient Descent (GD): Go downwards towards steepest gradient. Takes as parameter the learning rate (stepsize)
- Newtons Algorithm: Choose a(k+1) optimally to minimize second order Taylor expansion of loss function J(a(k+1)).

Perceptron Loss: 1-0 Loss is non-convex use $-a^T x_i$ instead and perform Perceptron Algorithm (slides)

Novikov Theroem: If the training samples are linearly seperable, then the sequence of the perceptron algorithm will terminate at a solution vector.

9.1 Bayes Classifiers

What decision functions do we use in bayes classification?

Bayes Optimal Classifier: The classification rule which minimizes the total risk for 0-1 loss is the maximum posterior of y (MAP) i.e. $max_y p(y|x)$ (if its bigger than doubt d)

Outliers Modeling by an outlier class π_O with $p_O(x)$. **Novelty Detection:** Classify a measurement as an outlier if $\pi_O p_O(x) >= threshold$

9.2 Fisher's Linear Discriminant Analysis:

Project high-dimensional data of two classes to a one-dimensional linear subspace such that the classes are optimally separated.

10 Support Vector Machines (SVM)

Idea: Generalize perceptrons with maximized margin and a kernel.

Agenda:

- Optimization with constraints
- Hard-margin SVM's
- Soft-margin SVM's

10.1 Lagrange multipliers

Assume that f and g_i are continuously differentiable. find: $min\ f(w)$, $w \in \mathbb{R}^d$ s.t. $g_i(w) = 0$, $i <= m$

1. Compute Lagrangian: $L(w,\lambda) = f(w) + \sum_{i<=m} \lambda_i g_i(W)$.
2. An optimal solution must satisfy ∇L = 0.

Lagrange multipliers 2 Assume that f and g_i are continuously differentiable. find: $min\ f(w)$, $w \in \mathbb{R}^d$ s.t. $g_i(w) = 0$, $i <= m$ and $h_j(w) <= 0$, $j <= m$

1. Compute Lagrangian: $L(w,\lambda,\alpha) = f(w) + \sum_{i<=m} \lambda_i g_i(W) + \sum_{j<=n} \alpha_j h_j(w)$, with $\alpha_j >= 0$.
2. An optimal solution must satisfy L differentiated to w = 0 and diff to λ. $\alpha_j h_j(w) = 0$ and $\alpha_j >= 0$

The dual problem TODO

Slaters condition TODO
SVM's fulfill Slaters condition, therefore strong duality holds for that problem.

Complementary slackness TODO

Karush Kuhn Tucker conditions: necessary conditions for an optimal solution TODO
If f is convex, h_j is convex and g_i is affine, then any tuple $w*, \lambda*, \alpha*$ that meets the KKT conditions yields an optimal solution with strong duality.

Maximal Margin Classifier

$$2xmargin = w^T/||w||(y^+ - y^-) \tag{6}$$

Invariance of Maximal Margin Classifier Problem: scaling of the margin is compensated by scaling the $||w||$
Two equivalent solutions for well-posedness:

1. geometric margin problem: maximize m for $||w|| = 1$.
2. functional margin problem: minimize $||w||$ for m = 1.

10.2 Non-seperable case: Soft Margin SVM

How to treat samples that violate the constraint? Introduce Slack variables

Slack variables: relax the margin constraints for the samples that violate the constraints. The invariance under rescaling still holds.

Linear Programming SVM

+ efficient LP solver can be used.
- theory is not as well understood as for standard SVMs

10.3 Non-Linear SVMs

Feature extraction by non linear transformation $y = \phi(x)$. Problem: for very high dim spaces the non linear transformation might be computationally too difficult to compute when we only require the inner prod.

Kernel Function $K(x,z) = \phi^T(x)\phi(z)$

1. Symmetric
2. Kernel matrix has to be positive semi-definite.
3. Merce's Theorem
4. Kernel engineering (slides)

SVM's for Secondary Structure Prediction

10.4 Structural SVMs

- Multiclass SVM's
- Structured SVM's: Each sample y is assigned to a structured output label, e.g. partitions, trees, lists

Multiclass SVMs The notion of the margin must be generalized to multi-class problems: For every class z, we introduce a weight vector w_z and define the margin as the maximum m s.t. $(w_{z_i}^T y_i + w_{z_i,0}) - max_{Z!=z_i}(w_{z_i}^T y_i + w_{z_i,0})$

Similar to before we can define an optimization problem to learn hard functinoal margin multiclass SVMs. For non-seperable SVMs we can again introduce slacks.

Structural SVMs E.g. A parser takes as input a natural language sentence. The task is to predict a parse tree decomposing the sentence into its constituents (nr of possible parse trees is tremendous).

SSVMs Kex Problems:

1. Compact representation of the output space
2. Efficient prediction
3. Prediction error
4. Efficient training

SSVMs could be formulated as multiclass SVMs (one weight vector per class), however this would lead to a blow-up of the overall number of parameters. Key ideas to overcome this:

1. Define a joint feature map that combines properties of inputs and outputs.
2. Define a scoring function.
3. Perform classification via these two functions.
4. Define the margin as seen in slides.

For efficient classification there must be some kind of structural matching between the compositional structure of the outputs z and the designed joint feature map, e.g:

- Decomposable output spaces: into non-overlapping independent parts, then maximization is performed partwise.
- Specific dependency structures:

11 Ensemble Methods for Classifier Design

Idea: An Approach to ML based on the idea of creating a highly accurate prediction rule by combinig many relatively weak and inaccurate rules. Train several sufficiently diverse predictors and use an aggregation scheme to produce a valid solution with low bias and variance.

- **Bagging:** or bootstrap aggregation combines several classifiers which have been trained on random subsamples (with replacement) of the data.
- **Arcing:** or adaptive reweighting and combining is an extension of bagging.
- **Boosting:** in form of AdaBoost is an ensemble method with error sensitive reweighting of the classifiers

Weak Learners Used for Bagging or Boosting:

- Stumps: Single axis parallel partition of space
- Decision Trees: Hierarchical partition of space
- MLP: General non-linear function approx
- Radial basis functions: Non linear expansions based on kernels

Combining Classifiers

- Input: a pool of binary classifiers.
- Objective: Infer a composite classifer with weights α_b
- Need diversity in classifers: use different subsets of data and different features. Decorrelate classifers during training.

Bagging Generate diversity in classifier pool by training on different, e.g. bootstrapped subsets. Classifier selection: first compare, then bag.

Random Forests

- Strategy for bagging trees:

Boosting

- Idea: An adaptive combination of poor learners with sufficient diversity leads to an excellent classifier.
- Base class of simple classifiers (e.g. perceptrons, decision stump, axis parallel splits)
- Train a sequence of simple classifiers on modified data distributions, and form a weighted average.

Data Reweighting Apply weight to each of the training data at the b-th boosting step.

AdaBoost Train each classifier on random subset of data and then reduce the weights on current subset of data and increase the weights on rest of data, then train next classifier.

Loss functions for classification

- exponential loss
- Binominal Deviance

Theorem: The Discrete AdaBoost algorithm builds an additive logistic regression model via Newton-like updates for minimizing exponential loss.

12 Probably Approximately Correct Learning (PAC Learning)