ML and Numerical Software Development

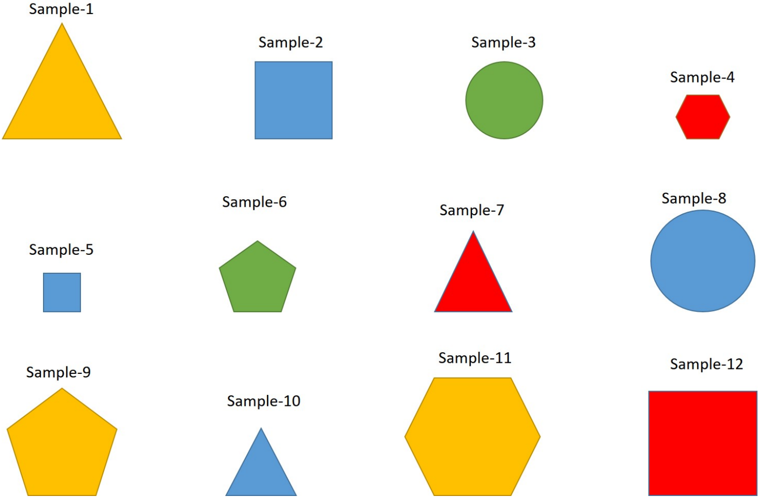
Machine Learning-I

Organon Analytics

December 3, 2019

ML Tasks

Let's create a synthetic database of shapes:



ML Tasks

Recipe for creating a single sample:

1. Pick one shape: {triangle, square, circle, pentagon, hexagon}
2. Color it with {red, yellow, green, blue}
3. Pick a random value between $[0, 180]$ and rotate the circle
4. Record the following data for each sample:
    - *AREA*: Area of each shape in
    - *EDGES*: Number-of-edges of each shape
    - *COLOR*: The color of each shape
    - *PICTURE-GRAY*:Each shape stored as an $32x32$ grayscale picture
    - *PICTURE-RGB*: Each shape stored as an $3X32x32$ RGB picture
    - *LABEL*: Label of each shape: {triangle, square, circle, pentagon, hexagon}

The synthetic database:

Id	*Area*	*Edges*	*Color*	*P-Gray*	*P-RGB*	*Label*
1	2.5	3	"Yellow"	PGray001	PRGB001	"Triangle"
2	6	6	"Red"	PGray002	PRGB002	"Hexagon"
⋮	⋮	⋮	⋮	⋮	⋮	⋮
100	10	5	"Blue"	PGray100	PRGB100	"Pentagon"

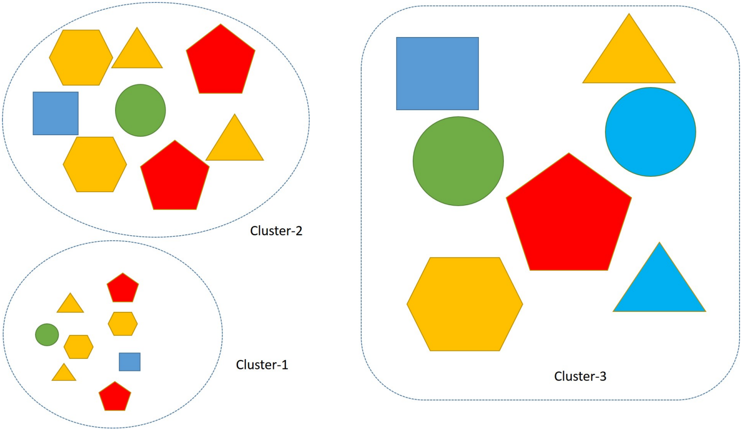
Where *P-Gray* and *P-RGB* fields are of BLOB type.

ML Tasks: Unsupervised Tasks

- Group similar objects together by using the inputs {*AREA*}
- Group similar objects together by using the inputs {*AREA*, *EDGES*}
- Group similar objects together by using the inputs {*AREA*, *EDGES*, *COLOR*}
- Group similar objects together by using the pixel values in each grayscale picture
- Group similar objects together by using the pixel values in each RGB picture

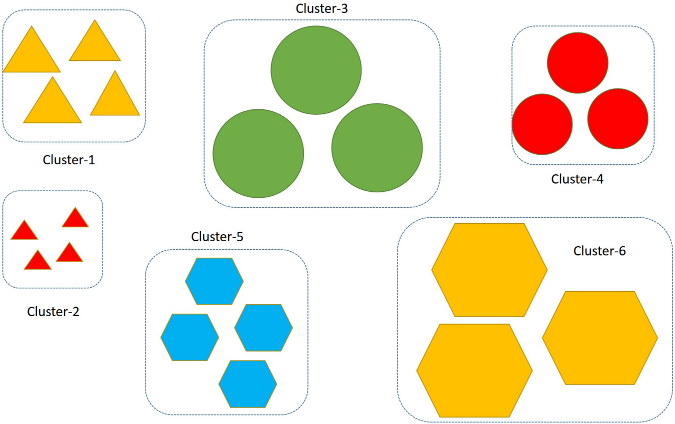
Unsupervised Task-1

Group similar objects together by using the inputs $\{AREA\}$



Unsupervised Task-2

Group similar objects together by using the inputs $\{AREA, EDGES\}$



ML Tasks: Unsupervised Learning Tasks

Def'n: *Tasks where the goal is to find previously unknown patterns (structures, regularities) in the data without the guide of any pre-existing label*

1. **Clustering**: Find clusters of samples that are **close together**. One needs a similarity(distance) function between two vectors:

   $$s_{ij} = d(X_i, X_j)$$

   Algorithms: *k-means*, *hierarchical clustering*, *mixture modelling*, *self organizing maps*

2. **Density Estimation**: To find the density or the mass function that generated the data. The most general problem of unsupervised learning
   Algorithms: *Mixture modelling*, *self organizing maps*, *generative adversarial networks*

3. **Outlier Detection**: Find the outlier samples

Unsupervised Learning Tasks: Important Algorithms

- *K-Means*: The go-to algorithm for low dimensional structural data. Very easy to implement
- *Mixture Modelling*: Used for density estimation
- *Generative Adversarial Networks*: An ANN that learns the PDF from a population of pictures. The model is used to generated new synthetic pictures
- Latent variable Modelling:
    - *i* Factor Analysis (Principal Components, Independent Components)
    - *ii* Matrix Factorization (Ex: Recommendation Systems)
    - *iii* Latent Semantic Indexing (Ex: Document classification)
    - *iv* Expectation Maximization Algorithm (Ex: Record Linkage)

Unsupervised Learning Tasks: Examples

- **Clustering**: Find music pieces that are similar
- **Clustering**: Find user groups whose behavior (listening to music, purchasing, banking behaviour, etc.) are similar
- **Clustering**: Build a taxonomy of objects from their pictures (*classification without labels*)
- **Density Estimation**: Find similar records in a customer database in the absence of a unique identifier
- **Density Estimation**: Learn to generate new synthetic pictures out of a given population
- **Outlier Detection**: Find outlier log records that correspond to suspicious events (malicious code run, suspicious login, data breach, etc.)

Unsupervised Learning Ex.: Record Linkage

Match data records that are similar in a database

NAME_1	NAME_2
ISMAIL	SAHISMAIL
ISMAIL	ISMAYIL
IBRAHIM	IBARHIM
IBRAHIM	IBRAHIN
SELAHATTIN	SELAHADDIN
SELAHATTIN	SELAHITTIN
SELAHATTIN	SELATTIN
OMER	IMER
OMER	TC OMER
OMER	OMERUL

SURNAME_1	SURNAME_2
ALBAYRAK	AGBAYRAK
ARSLAN	ARISLAN
ARSLAN	ARSLN
KARAARSLAN	KAYAARSLAN
RSADIKOGLU	SADIKOGLU
DEGIRMENCI	DIRMENCI
DEGIRMENCI	DEYIRMENCI
YILDIRIM	YILDIRM
YILDIRIM	YIDIRIM
KAHRAMAN	KAHRAMANLI

BIRTH_PLACE_1	BIRTH_PLACE_2
DENIZLI	DENIZLER
ATA	ATCA
KAMBERLI	KAMBER
KOY	KOYU
KAHRAMAN	KAHREMAN
MARDIN	MARTIN
KARADOGAN	KARADIGIN
KARACAVIRAN	KARACAVERAN
SIRNAK	SIRNAN
ARGINCIK	ARINCIK

ADDRESS_1	ADDRESS_2
CUMHURIYET MAH. ISMET INONU BUL. TARIMCILAR SITESI D BLOK NO:249/11 ATAKUM SAMSUN TURKIYE	CUMHURIYET MAH . INONU BULVARI D BLOK 249/11 MERKEZ SAMSUN TURKIYE
MASHAR OSMAN SK CAMLI APT N:5 K.4 D.12 FENERYOLU KADIKOY ISTANBUL TURKIYE	FENERYOLU MAZHAR OSMAN SK. CAMLI APT NO:5 KAT:4 DAIRE:12 KADIKOY ISTANBUL TURKIYE
TASDELEN ANADOLU CAD NO.12 D.5 CEKMEKOY ISTANBUL TURKIYE	ANADOLU CD.NO:12 D.5 TASDELEN CEKMEKOY ISTANBUL TURKIYE
HURRUIYET CAD.NO:38 B BLK D:6 BAKIRKOY ISTANBUL TURKIYE	HURRIYET CD.N38 B BLK D.6 ISTANBUL TURKIYE

Algorithm: Expectation Maximization
Theory: Fellegi-Sunter Theory

Unsupervised Learning Ex.: Recommendation systems

Data: Ratings on each movie provided

$$r_{ui} \equiv \text{The rating the user } u \text{ gives for the movie } i$$

Model r_{ui} with matrix factorization:

$$\begin{aligned} r_{ui} &\equiv b_u + b_i + \mathbf{p}_u \cdot \mathbf{q}_i \\ b_u &\equiv \text{User bias} \\ b_i &\equiv \text{Movie bias} \\ \mathbf{p}_u &\equiv \text{User factor} \\ \mathbf{q}_i &\equiv \text{Item factor} \end{aligned}$$

The objective(loss) function:

$$\mathcal{L}(b_u, b_i, \mathbf{p}_u, \mathbf{q}_i) = \sum_u \sum_i \left(r_{ui} - (b_u + b_i + \mathbf{p}_u \cdot \mathbf{q}_i)\right)^2$$

Unsupervised Learning Ex.: Recommendation systems

The algorithm(s): Stochastic Gradient Descent and Alternating Least Squares

- NetFlix contest (closed at 2009): 17,000 movies, 100M ratings
- Social media: 1.5M customers, 6M videos
- YouTube algorithm: Based on reinforcement learning

Unsupervised Learning Ex.: Generative Adversarial Networks(GANs)

Learn the density function from a set of pictures. Generate synthetic ones

Unsupervised Learning Ex.: Generative Adversarial Networks(GANs)

- DeepFakes
- Generate photorealistic images (fashion, design, games)
- Reconstruct 3-D models of objects from 2-D pictures
- Compose music
- Style transfer

Unsupervised Learning: Epilogue

- Biological learning is mostly unsupervised
- More important than supervised learning
- Labelling data is hard
- Crowd-sourcing was used to label ImageNet pictures
- Most active area of research in ML, DL

Supervised Learning

Model an output variable as a function of input variables

$$
\begin{aligned}
Y &\equiv \text{Output} \\
X &\equiv \{X_1, X_2, \cdots X_d\} \text{ Input variables} \\
\text{Data} &\equiv \{(X_1, Y_1), (X_1, Y_1), \cdots, (X_N, Y_N)\}
\end{aligned}
$$

Countless number of applications:

- *Output*: Default or not, *Inputs*: Anything you can collect on customer
- *Output*: Fraud or not, *Inputs*: Anything you can collect on customer&card
- *Output*: Cancer or not, *Inputs*: Pathology scans
- *Output*: Disease or not, *Inputs*: Medical files
- *Output*: Purchase or not, *Inputs*: Click-stream data, demographics, etc.
- *Output*: Object category, *Inputs*: Image pixel values
- *Output*: Next sentence, *Inputs*: Lots of text

Supervised Learning

What is to be modelled?

- Regression(Continuous output): Model $E[Y|X=x]$ as a function of inputs:

$$E[Y|X=x] \equiv F(x)$$

- Classification(Discrete output): Model $P[Y=C_k|X=x]$ as a function of inputs where C_k represents the k-th class

$$P[Y=C_k|X=x] = F_k(x),\ k=1,2,\cdots$$

The following four determines a Supervised ML algorithm :

0 **The type of output**: Continuous, Binary, Multinomial

1 **Functional specification**: The mathematical form of $F(\cdot)$

2 **Loss function**: A criterion for the goodness of $F(\cdot)$

3 **Search algorithm**: A recipe to find $F_{Best}(\cdot)$

Supervised Learning: Functional specifications

Generalized Linear Model

$$g(E[Y|X = x]) = \beta_0 + \beta_i X_i = \mathbf{X} \cdot \beta$$

1 Linear Regression: Y continuous, normal and $g = identity$:

$$F(x) \equiv E[Y|X = x]) = \mathbf{X} \cdot \beta$$

2 Binary Logistic Regression: Y binary, Bernoulli and $g = logit = \log x/(1-x)$:

$$\begin{aligned} F(x) &\equiv E[Y|X = x]) = P(Y = 1|x) \\ \log\left(\frac{F(x)}{1 - F(x)}\right) &= \mathbf{X} \cdot \beta \\ \log\left(\frac{P(Y = 1|x)}{P(Y = 0|x)}\right) &= \mathbf{X} \cdot \beta \\ P(Y = 1|x) &= \frac{1}{1 + \exp(-\mathbf{X} \cdot \beta)} \end{aligned}$$

Supervised Learning: GLM functional specifications

3. Multinomial Logistic Regression: Y multi-class, Multinomial and $g = logit = \log x/(1-x)$:

$$P(Y = C_k|x) \quad = \quad \frac{1}{1+\exp(-\mathbf{X}\cdot\beta_k)} k = 1, 2, \cdots m$$

4. Poisson Regression: Y non-negative (integer or real), Poisson and $g = log(\cdot)$:

$$Y = e^{\mathbf{X}\cdot\beta}$$

Supervised Learning: GAM functional specification

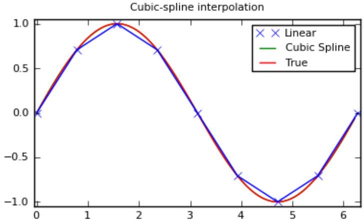
Change $\mathbf{X} \cdot \beta_k$ in GLM formulation to $f_0 + \sum f_i(X_i)$:

$$g(E[Y|X = x]) = f_0 + \sum_i f_i(X_i)$$

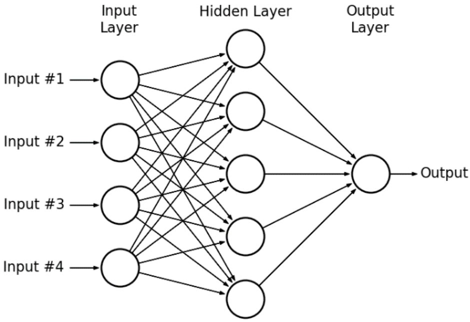
How to specify each $f_i(\cdot)$?
The most popular is cubic-spline specification:

$$f_i(X_i) = \sum_j \beta_{ij} B_j(X_i)$$

where $B_j(X_i)$ is cubic-splines of various orders



Supervised Learning: Multi Layer Perceptrons specification



$$
\begin{aligned}
\mathbf{h}_1 &= \sigma(\mathbf{W}_1 \cdot \mathbf{x}) \\
\mathbf{h}_i &= \sigma(\mathbf{W}_2 \cdot \mathbf{h}_{i-1}), i = 1, 2, \cdots m \\
Y &= \beta \cdot \mathbf{h}_m, \text{ Regression} \\
Y &= 1/(1 + e^{-\beta \cdot \mathbf{h}_m}), \text{ Binary classification}
\end{aligned}
$$

where σ is a nonlinear activation function (e.g. one of RELU, Tanh, logistic functions)

Supervised Learning: Loss functions

All supervised ML learning problems could be formulated as *constrained optimization* problems

$$\text{Minimize Loss}(\mathcal{D}, \beta) \text{ (The objective function)}$$

where $\mathcal{D}$ represents data, and subject to constraints on parameters:

$$\begin{aligned} f_1(\beta) &\leq c_1 \\ f_2(\beta) &\leq c_2 \\ &\vdots \end{aligned}$$

Supervised Learning as an Optimization Problem

Example:

$$\text{Minimize } \sum_i (Y_i - \mathbf{X}_i \cdot \beta)^2$$

Subject to the constraints:

- No constraint: Plain linear regression
- Ridge regression: $\sum_i \beta_i^2 \leq s$
- Lasso regression: $\sum_i |\beta_i| \leq s$

Supervised Learning: Two essential loss functions

Regression: Mean-square error, quadratic loss, L_2-loss

$$\mathcal{L}(F) \;=\; \sum_i (Y_i - F(X_i))^2$$

Classification: Cross entropy loss, minus-log-likelihood loss

$$\mathcal{L}(\theta) \;=\; -\sum_i \sum_k I[Y_i = C_k] \log P(Y_i = C_k | X = x_i; \theta)$$

where $I[Y_i = C_k]$ is the indicator function for classes $\{C_k\}$.

Supervised Learning: How to find $\hat{F}$, or $\hat{\theta}$

Use Gradient Descent:
Problem: Find $\hat{\theta}$ that will minimize $\mathcal{L}(\theta)$
Gradient Descent Algorithm (1st order method for function(al) minimization)

1. Initialize $\hat{\theta}$. Pick a random(or a good one) value $\hat{\theta}_0$
2. Update $\hat{\theta}$

$$\hat{\theta}_{i+1} = \hat{\theta}_i - \lambda \frac{\partial \mathcal{L}}{\partial \theta}\Big|_{\theta=\hat{\theta}_i}$$

3. Stop when $||\hat{\theta}_{m+1} - \hat{\theta}_m||$, OR $||\mathcal{L}(\hat{\theta}_{m+1}) - \mathcal{L}(\hat{\theta}_m)||$ is *small*

Gradient Descent: Notes

(New estimate) = (Old estimate) − (Learning rate)(Latest gradient)

0 $\frac{\partial \mathcal{L}}{\partial \theta}$ should be computable

1 λ adjusts the rate of learning. It is a hyper-parameter. Should be determined with cross-validation

2 Converges to global minimum for globally convex $\mathcal{L}$; converges to local minima (with proper λ)

3 Variable selection is not possible

4 Variable learning rate is used in practice

Stochastic Gradient Descent(SGD)

Remember the loss function

$$\mathcal{L}(\theta) \equiv \sum_i \mathcal{L}_i(\theta)$$

- GD updates $\hat{\theta}$ after seeing all the samples $(X_i, Y_i)_{i=1}^N$
- SGD updates $\hat{\theta}$ after seeing M samples where $M \ll N$
- $M = 1$ is called *online learning*

Algorithm Features vs. Algorithms

Feature	GLM	GAM	CRT	GBM	ANN
Ability to handle mixed data types	🔴	🔴	🟢	🟢	🔴
Ability to handle missing values	🔴	🔴	🟢	🟢	🔴
Robustness to outliers in inputs	🔴	🟢	🟢	🟢	🔴
Ability to handle non-linear relationships	🔴	🟢	🟢	🟢	🟢
Ability to select relevant input(s)	🔴	🔴	🟢	🟢	🔴
Computational complexity with N	🟢	🟢	🟢	🟢	🔴
Interpretability	🟢	🟢	🟡	🔴	🔴
Predictive power	🔴	🟡	🔴	🟢	🟢

🔴: Poor, 🟡: Fair, 🟢: Good