JUNAID . GIRKAR
60004190057
TE COMPS A4

MACHINE LEARNING
ASSIGNMENT - 2

Q1 Explain single value decomposition and its application?

ANS SVD of a matrix is a factorization of that matrix into 3 matrices. It's some interesting algebric properties and conveys importants geomatrical and theoretical insights about linear transformation. It also have some important application in Data science.

Mathematics behind SVD

The SVD of matrix A is given by formula $A = UWV^T$

where U: m×n matrix or orthonomal eight vectors of

V^T: transpose of n*n matrix consisting of orthonormal eight vectors of $A^{T}A$.

W: a n*n matrix diagonal of the singular values which are square roots of eigen vectors.

SINGULAR DECOMPOSITION ANALYSIS

$$C_{m\times n} = U_{m\times r} * \Sigma_{r*r} * V^T_{r\times n}$$

Example: $A = \begin{bmatrix} 3 & 3 & 2 \\ 2 & 3 & -2 \end{bmatrix}$

calculate $AA^T = \begin{bmatrix} 3 & 3 & 2 \\ 2 & 3 & -2 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 3 & 3 \\ 2 & -2 \end{bmatrix} = \begin{bmatrix} 17 & 8 \\ 8 & 17 \end{bmatrix}$

characteristic equation of above matrix :

$$W - \lambda I = 0$$

$$AA^T - \lambda I = 0$$

$$\therefore \lambda^2 - 34\lambda + 225 = 0$$

$$\therefore \lambda^* = (25, 9)$$

So our singular values are $\sigma_1 = 5$, $\sigma_2 = 3$

Now we find the right singular vectors i.e. orthogonal or orthonormal set of eigen vector of A^TA. The eigen vectors of $A^TA = 25, 9$ and 0, and since A^TA is symmetric we know that eigen vector is orthogonal.

$\therefore$ For $\lambda = 25$

$$AA^T - 25.I = \begin{bmatrix} -12 & 12 & 2 \\ 12 & -12 & -12 \\ 2 & -12 & -17 \end{bmatrix}$$

A unit vector in the direction of it is :-

$$V_1 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{bmatrix}$$

Similarly, for \lambda = 9 from eigen vectors

$$V_2 = \begin{bmatrix} 1/\sqrt{18} \\ -1/\sqrt{18} \\ 4/\sqrt{18} \end{bmatrix}$$

For 3rd Eigen vector we could use the property of that it is perpendicular to v1 and v2 such that :

$$V_1^T V_3 = 0$$

$$V_2^T V_3 = 0$$

Solving equation to generate third eigen vector :

$$V_3 = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} a \\ -a \\ a/2 \end{bmatrix} = \begin{bmatrix} 2/3 \\ -2/3 \\ -1/3 \end{bmatrix}$$

Now we calculate u by using formula $u = \frac{1}{\text{sigma}(A * V_i)}$

SVD equation becomes

$$A = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{bmatrix} \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & 1/3 \end{bmatrix}$$

Application :-

1) Calculation of Pseudo Inverse
2) Rank, Range and Null space
3) Curve fitting problem.

Q2 Explain Temporal-Difference learning in Reinforcement Learning?

ANS

- Temporal-Difference learning = TD Learning
- The prediction problem is that of estimating the value function for a policy π.
- The control problem is the problem of finding the optimal policy π_* for non-terminal states occuring in that experience.
- Given current step t, TD methods waits until the next time step to update $V(S_t)$
- Learn from partial returns.

TD(0) :-

H is a simplest form of TD learning. In this form of TD learning, after very step value function is the updated with value of next state and along the way reward behind obtained. This observed reward is the key factor that keeps the learning grounded and algorithm converges after a sufficient number of sampling. Below is backup diagram of TD(0) and an example of TD(0) of our gem collection and examination example.

(1) $V(S_t) \leftarrow V(S_t) + \alpha (R_{t+1} + \gamma V(S_{t+1})) - V(S_t)$

(2) $V(S_t) \leftarrow (1-\alpha) V(S_t) + \alpha (R_{(t)} + \gamma V(S_{t+1}))$

TD target $= R_{t+1} + \gamma V(S_{t+1})$

TD Error $(S_t) = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$.

SARSA :-

One of the TD algorithms for control or improvement in SARSA. It comes from the fact that agent takes one step from one state-action value pair to another state-action. value pair and along the way collect Reward R. SARSA is an on-policy method. SARSA uses action-value function Q. and follow the policy π. GPI is used to take action based on policy π SARSA can be representing with equation. Equation -1 is generally shown in literature but I find the same equation written as per equation 2 is more intuitive.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right)$$

$$Q(S_t, A_t) \leftarrow (1-\alpha) Q(S_t, A_t) + \alpha\left(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})\right)$$

Action value TD target $= R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$

Action value TD error $(\delta_t) = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$

TD learning is an unsupervised technique to predict variable expected value in sequence of states. TD uses a mathematical notations trick to replace complex reasoning about the future with simple learning procedure that can produce the same results. Instead of calculating the total future reward, TD tries to predict the combination of immediate reward and its own reward prediction at the next moment in time.

Mathematically, the key concept of TD learning is the discounted return:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

where reward at time t is combined of discounted rewards in the future. It implies that future rewards are valueless. The TD Error is the difference between the ultimate correct reward (V^*_t) and our current prediction (V_b)

$$E_t = V_b^* - V_t$$

$$= r_{t+1} + \sum_{k=1}^{\infty} \gamma^k r_{t+k+1} - V_t$$

$$= r_{t+1} + \gamma * \left(\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k+1} \right) - V_t$$

$$= r_{t+1} - \gamma \times V_{t+1} - V_t$$

And similarly to other optimization methods, the current value will be updated by its values + learning rate x error

$$V_t \leftarrow V_t + \alpha * E_t = V_t + \alpha * (r_{t+1} + \gamma (V_{t+1} - V_t))$$

Q3 Can Deep Learning be used for unsupervised learning? Justify your answer?

ANS

Deep learning involves the use of the complex models that exceeds the capabilities of machine learning tools such as logistic regression and SVM, but their deep learning models are esentially "function approximations".

Deep learning uses Neural networks. can also be used to cluster images based on similarities. One can extract the features with a neural network then deploy an unsupervised methodology such as K-means clustering. A neural network can be in form of semi-supervised deep neural network.

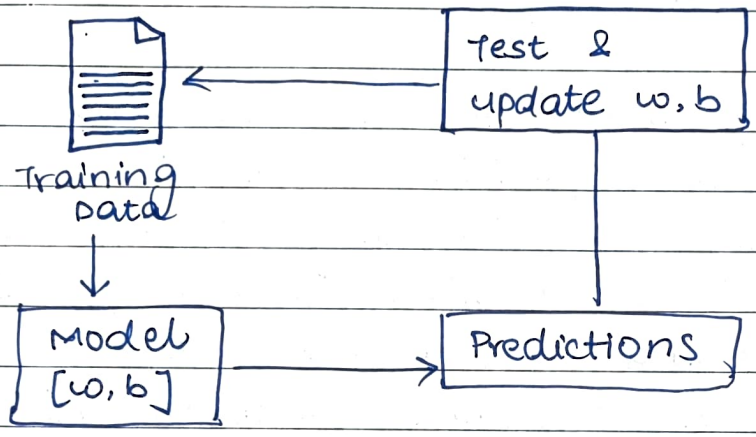
In order, to autoencoders are neural nets that ean be used for image compression and reconstruction via latent space representation of compress data; inshort its output whatever is inputed. These autoencoder are considered sey-supervised learning neural nets.

Therefore reinforcement learning with neural net can be used. and was the methodology behind DeepMind and its victory in gameGo.

Therefore Deep learning can be used for unsupervised learning.

Q4 Steps involved in developing a Machine Learning Application

ANS



i) DATA COLLECTION : → The quality and quantity of your data dictate how accurate your model is.
→ The outcome of this step is a representation of data which we will use for training.
→ Using pre-collected dataset, by way of dataset from Kaggle, UCI, etc.

ii) DATA PREPARATION : → Wrangle data & prepare it for training
→ clean that which may require it.
→ Randomized data, which erase the effects of particular order in which we collector and/or otherwise prepared our data.
→ Visualize data to help detect relevant relationships between variables or class imbalances
→ split training data into training and evaluation sets.

iii) CHOOSE A MODEL : → Different Algorithm are for different task; choose the right one.

iv) TRAIN THE MODEL : → The goal of training is to answer a question or making a particular prediction correctly as often as possible.
→ Linear Regression example : Algorithm would need to learn values for m and b.
→ Each iteration of process is a training step.

v) EVALUATE A MODEL : → Use some metric or combination of metrics to "measure" objective performance of model.
→ Test model against previously unseen data.
→ The unseen data is meant to be somewhat representative of model performance in real world.
→ Good train/test split
e.g: 80/20 , 70/30 , etc.

vi) PARAMETER TUNING : → This step refers to hyperparameter tuning, which is an art form as opposed to science.
→ Tune model parameter for improved performance
→ simple model hyperparameter may include :- Number of training steps, learning rate , initialization values, distribution , etc.

vii) MAKE PREDICTIONS : → Using further (test set) data which have, until this point, being witheld from the model are used to test the model.

→ A better approximation of how the model will perform in real world.