Machine Learning Final

Name: Eftakharul Islam Emon

Id: 20-43555-1

Q1) Locally Weighted Regression (LWR) and Linear Regression (LR) differ primarily in their approach to fitting data.

Linear Regression: Formula: $h\theta(x)=\theta 0+\theta 1xh\vartheta(x)=\vartheta 0+\vartheta 1x$ Linear regression aims to find a linear relationship between the independent variable xx and the dependent variable yy by minimizing the sum of squared errors. It assumes a global model that fits all data points equally.

Locally Weighted Regression: Formula: $h\theta(x)=\sum i=1mw(i)(x)\cdot y(i)h\vartheta(x)=\sum i=1mw(i)(x)\cdot y(i)$ Locally weighted regression, on the other hand, constructs a separate model for each prediction by giving more weight to nearby data points and less weight to distant ones. The weight function $w(i)(x)w(i)(x)$ assigns weights to each data point based on its distance from the point being predicted.

Advantage of LWR over LR: LWR adapts better to local patterns in the data, making it more flexible and capable of capturing non-linear relationships between variables compared to LR.

Q2) Binary Logistic Regression Model:

Binary logistic regression is used when the dependent variable is binary (e.g., presence/absence of a tumor). It models the probability that a given input belongs to a certain category using the logistic function.

Model Output: $y=11+e-zy=1+e-z1$ where $z=\theta 0+\theta 1x1+\theta 2x2+\ldots+\theta nxnz=\vartheta 0+\vartheta 1x1+\vartheta 2x2+\ldots+\vartheta n xn$ is the linear combination of input features.

Training:

1. Initialize parameters $\theta\vartheta$.

2. Define a cost function, often the logistic loss function: 𝐽(𝜃)=−1𝑚∑𝑖=1𝑚[𝑦(𝑖)log(ℎ𝜃(𝑥(𝑖)))+(1−𝑦(𝑖))log(1−ℎ𝜃(𝑥(𝑖)))]J(ϑ)=−m1∑i=1m[y(i)log(hϑ(x(i)))+(1−y(i))log(1−hϑ(x(i)))]
3. Minimize the cost function using an optimization algorithm like gradient descent: 𝜃𝑗:=𝜃𝑗−𝛼∂𝐽(𝜃)∂𝜃𝑗ϑj:=ϑj−α∂ϑj∂J(ϑ) where 𝛼α is the learning rate.

Prediction: Given new patient data 𝑥new*x*new, compute ℎ𝜃(𝑥new)*h*ϑ(*x*new). If ℎ𝜃(𝑥new)≥0.5*h*ϑ(*x*new)≥0.5, predict malignant (1); otherwise, predict benign (0).

Q3.

a) Softmax Cost Function and Output Function: Softmax Cost Function: 𝐽(𝑤)=−1𝑁∑𝑛=1𝑁∑𝑘=1𝐾𝑦𝑘(𝑛)log(𝑦^𝑘(𝑛))*J*(**w**)=−*N*1∑*n*=1*N*∑*k*=1*Kyk*(*n*)log(*y*^*k*(*n*)) where 𝑁*N* is the number of training examples, 𝐾*K* is the number of classes, 𝑦𝑘(𝑛)*yk*(*n*) is the 𝑘*k*-th component of the one-hot encoded true label of the 𝑛*n*-th training example, and 𝑦^𝑘(𝑛)*y*^*k*(*n*) is the predicted probability of class 𝑘*k* for the 𝑛*n*-th example.

Softmax Output Function:
𝑓(𝑥;𝑤)=softmax(𝑤𝑇𝑥)=exp(𝑤𝑇𝑥)∑𝑗=1𝐾exp(𝑤𝑗𝑇𝑥)*f*(*x*;**w**)=softmax(**w***T***x**)=∑*j*=1*K*exp(**w***jT***x**)exp(**w***T***x**)

b) Relationship between Softmax and Binary Logistic Regression: Softmax regression is a generalization of binary logistic regression to handle multiple classes. When there are only two classes, softmax regression reduces to binary logistic regression.

Q4) Penalty Term of Ridge/L2 Regularization: The penalty term of ridge/L2 regularization is the sum of squared magnitudes of the coefficients:

Penalty term=𝜆∑𝑗=1𝑝𝜃𝑗2Penalty term=*λ*∑*j*=1*p*ϑ*j*2 where 𝜆λ is the regularization parameter and 𝑝*p* is the number of features.

How it Reduces Overfitting: Ridge regularization penalizes large coefficients, encouraging them to be small. This discourages overly complex models by reducing the variance in the model, thus helping to prevent overfitting.

Q5.

a) Applying Policy Iteration to Solve an MDP:

1. Initialize the value function $V(s)V(s)$ arbitrarily for all states.
2. Repeat until convergence: a. Policy Evaluation:
    - For each state ss, update the value function: $V(s)\leftarrow\sum a\pi(a|s)\sum s',rp(s',r|s,a)[r+\gamma V(s')]V(s)\leftarrow\sum a\pi(a|s)\sum s',rp(s',r|s,a)[r+\gamma V(s')]$ b. Policy Improvement:
    - For each state ss, update the policy: $\pi'(s)\leftarrow\arg\max a\sum s',rp(s',r|s,a)[r+\gamma V(s')]\pi'(s)\leftarrow\mathrm{argmax}a\sum s',r p(s',r|s,a)[r+\gamma V(s')]$
    - If $\pi'\pi'$ is unchanged (or change is negligible), stop and return VV and $\pi\pi$.

b) Advantage of Using Exploration-Based Policy like $\epsilon\epsilon$-Greedy: Exploration-based policies like $\epsilon\epsilon$-greedy ensure that the agent explores different actions and states during learning, rather than exploiting the current knowledge. This helps in discovering better strategies and prevents the agent from getting stuck in suboptimal solutions.

Q6.

a) Q-Learning as an Off-Policy Algorithm: Q-learning is an off-policy algorithm because it learns the value of the optimal policy $\pi*\pi*$ while following a different policy, typically an exploratory policy such as the $\epsilon\epsilon$-greedy policy.

b) Difference between On-Policy and Off-Policy Algorithms:

- On-policy algorithms update their Q-values based on the same policy that is used to select actions (e.g., SARSA).
- Off-policy algorithms update their Q-values based on a different policy than the one used to select actions (e.g., Q-learning).