Exam

Niklas Muennighoff
18000092850

I

1. True

2. False, we combine all of them with weights

3. False, ~~...~~ SVM with no kernel can be applied to not linearly separable data, e.g. Soft-Margin SVM (we might make some errors)

4. False, we do not know about the environment beforehand in Q-Learning

5. False, it is more likely to happen when the size of the feature space is much larger.

6. False, if our learning rate = 0, gradient descent does not converge at all. Also it does get stuck at local optima & is not guaranteed to converge to a global optimum

7. False, due to the curse of dimensionality it gets increasingly costlier for kNN to find neighbors in many dims.

8. False, it is done by selecting the split with the lowest entropy afterwards (= the highest information gain)

II

1. C

2. A

3. A

4. A

5. C

IV

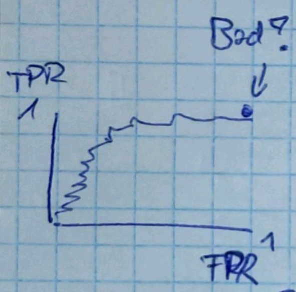
1.

Problem: The problem is that Age and income may cover completely different numeric ranges. Age may range from ~0 to ~100 and Income to ~10000..., hence we cannot compare their distances. Hence income values ~~It could be fixed~~ will probably dominate all of our predictions.

To fix this, we should Normalize the data. We scale each feature to be in roughly the same range. (Their variance will be similar after scaling)

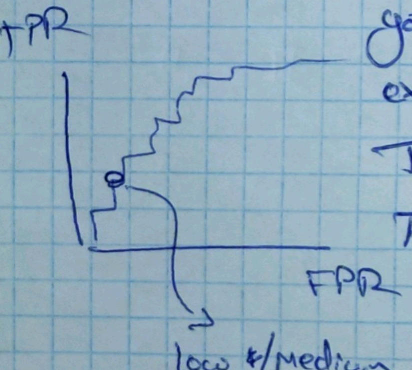
2.

1) The data may be skewed. For a dataset with 90/100 positive examples, accuracy will be high, but a classifier with 90% accuracy is bad! By modelling TPR vs FPR, ROC curves show us exactly the true positives & false positives of the classifier. Hence a classifier predicting class 1 for all will have TPR of 1, but FPR of 1, hence ROC reveals that it's bad



2) ROC curves also let us model the impact of choosing different thresholds. If we're trying to find goldmines and want a ~~very~~ low FPR (because mining is expensive), we can tradeoff ~~(decrease TPR) & with FPR~~ TPR with FPR (e.g. choose a threshold with low FPR)

Accuracy only gives us one score. ~~and~~



low ~~/~~/medium TPR with low FPR → minimize false positives when searching goldmines

3.

No, because

a) He did not use a test set, but tuned based on the validation set

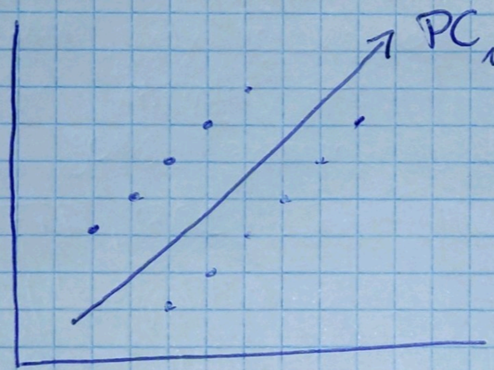
b) He did not use K-fold cross-validation

c) He evaluated 100 algo., a clear sign of overfitting

d) ~~He~~ What is the data distribution? Maybe it is 90 - 10

B.

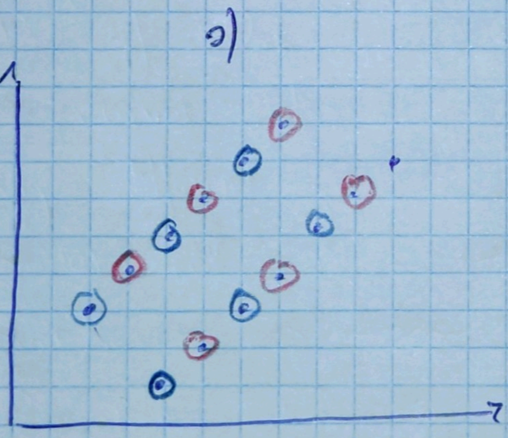
(1)

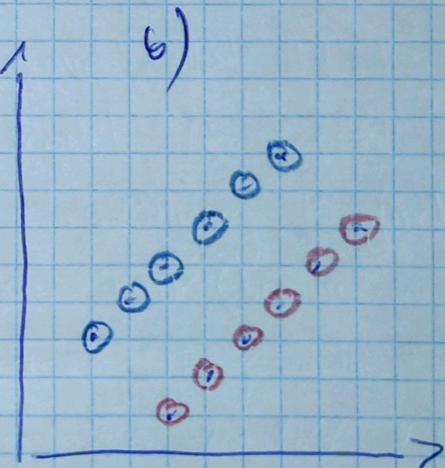


(2)

neg

pos





E

(If $\lambda = 0$, no reg ; If $\lambda \to \infty$, lots of reg ; no complexity)

1.

As λ increases, complexity decreases

2.

(less variance)
(less overfitting)

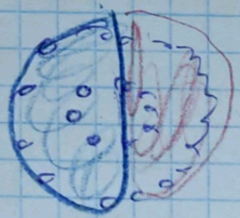
A.1 : Column C

A.2 : Column B (L2) (large weights penalized most)

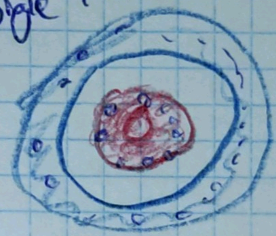
A.3 : Column A (L1)

D

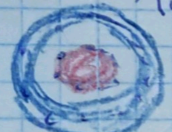
(a) k-means



b) Hierarchical Single (MIN)

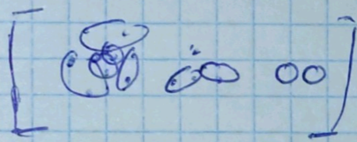


(d) DBSCAN :) (with correct HPs)



c) Avg

Bonus



1)

$$V(\text{cool}) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \cdot V(s')$$

$$4 + \underbrace{0.99 \cdot (4 + 0.99 \cdot 4 + 0.99^2 \cdot 4 \ldots)}_{4 \cdot \frac{1}{1-0.99} - 1}$$

$$\underline{400} = 4 + 4 \cdot 99$$

$$\left[1 + x + x^2 \ldots = \frac{1}{1-x} \right]$$

2) If in state cool, move Fast (→ More reward)

If in state warm, move Slow (→ Risk of turning off)

There is more reward for moving fast, & if we move slow after fast there is 0 probability of being turned off. Hence it is optimal to move fast when cool & slow when warm.

[We do not want to move fast when warm, because it reduces future reward by a lot

Bonus 7/ Due to the discount factor close to 1, future rewards are very important.

Due to the high probability ($\frac{1}{2}$) of being off & never receiving rewards anymore when doing warm $\Rightarrow$ fast, we ~~[illegible]~~ always want to move slow when warm?