Monitoring

① RMS for binary classifier:

$$RMS = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(r_t - o_t)^2}$$

$$\xi \equiv 2 \cdot \text{Bernoulli}(0.5) - 1$$

$$\Rightarrow RMS = \left(\frac{1}{N} \cdot \frac{1}{2}N\left((0.5+\epsilon\xi-1)^2 + (0.5+\epsilon\xi)^2\right)\right)^{\frac{1}{2}}$$

↳ half of outcomes will be pos half will be neg so $\frac{1}{2}N$ factor in front of each term

$$= \left(\frac{1}{2}\left((0.5+\epsilon\xi)^2 + (-0.5+\epsilon\xi)^2\right)\right)^{\frac{1}{2}}$$

$$= \left(\frac{1}{2}\left(0.25 + \epsilon^2\xi^2 + \epsilon\xi + 0.25 + \epsilon^2\xi^2 - \epsilon\xi\right)\right)^{\frac{1}{2}}$$

$$= (0.25 + \epsilon^2\xi^2)^{\frac{1}{2}}$$

$$= \left(0.25 + \epsilon^2(2\,\text{Bernoulli}(0.5) - 1)^2\right)^{\frac{1}{2}}$$

100 ~~[illegible]~~% $\Rightarrow RMS = \left(\frac{1}{N}\frac{1}{2}N\left((1-1)^2 + (0-1)^2\right)\right)^{\frac{1}{2}}$

$$= \left(\frac{1}{2}(1)\right)^{\frac{1}{2}} = \frac{1}{\sqrt{2}}$$

~~70% [illegible]~~

RMS for binary classifier with $p(\hat{y}|x) = 0.7$

$$RMS = \sqrt{\frac{1}{N}(0.7N(1-1)^2 + 0.3N(0-1)^2)}$$

$$= \sqrt{0.3} = 0.5477...$$

Q2 a.) False

b.) False, it only can increase callibration

c.) True, calibration won't be represented in the final tolly of the species, but wrongly assigned anomolies will.

d.) False, adversarial attacks can result in high softmax probabilities that would not often fall below the threshold for sofmat prob max detection

Q3 a.) False, AUROC does not depend on callibration, a calibrated classifier can still get ~~examples~~ it wrong and so does not have a AUROC of 100%

b.) True, changing temp changes callibration which could change whether or not a probability crosses the Maximum softmax probability anomoly detector threshold c.) True, zero's input will result in zero's logits (the origin) which must be ~~ex~~ contained in the feature subspace spanned by the w_i vectors.

Q4 a.) False, Tuning only affects confidence of prediction

b.) True, reduces variance

c.) True, if it is always certain of its answer but often wrong then it is wrong to be certain

d.) True

Q5.) $H(p,q) = -\sum_x p(x) \log(q(x))$

$$-\sum_i \frac{1}{k} \log\left(\frac{e^{l_i}}{\sum_j e^{l_j}}\right) = -\frac{1}{k} \sum_i \left(l^i - \log\left(\sum_j e^{l_j}\right)\right)$$

$$= -\frac{1}{k}\left(\sum_i l_i - k \log\left(\sum_j e^{l_j}\right)\right)$$

$\log(xy) = \log(x) + \log y$

$\log(e^x) =$ ~~log~~ x

etc

$$= -\frac{1}{k} \sum_i l_i + \log \sum_j e^{l_j}$$

$= H(\mathcal{U}; p)$ as required

Q6.)

least : (c)
highest : (a)

Q7.) a.) high precision
b.) high recall
c.) neither
d.) both

Q8.) a.) True positive
b.) False negative
c.) False positive
d.) True negative

Q9.) Simply invert the graph ie whenever your classifier says positive ~~say negative and~~ use negative and when classifier says negative say positive, now you have Auroc of 95%.

Q10.) An overconfident model in this case would actually be good, as it would be less likely to report positives and so only the very anomolous inputs would be classified as positive. This results in low recall, but high precision as it will not detect many False positives, and it will miss many anomolies and have many False negatives, as desired. in the question.

Q11.) • A biometric scanner with a trojan ~~to~~ that automatically classifies an input as an undesired output (computer vision) to access sensitive material

• A language Model that outputs harmful language upon activation by a keyword to attack twitter bots. A companys ~~the~~ automatic twitter bot could be made say harmful things upon hearing a key Phrase.

• An RL video game bot that kills itself upon finding itself in a certain world state (like a chess AI that aims to lose upon seeing a particular boardstate) could be used to cheat.

Q12.) False, there does not have to be many poisined data points so finding them would be like finding a needle in a haystack, and the "poison" can be subtle to notice even if you are looking at it.

Q13.) Because determining if a network is trojaned by analysing parameters is extremely difficult and a network that takes another networks parameters as inputs would be extremely expensive to train and run due to large input length. Litmus tests simply cost a forward pass of the network in question

Q14.) The kind of changes ~~made in~~ to an input in a trojan vs adversarial attack are different, the former is a consistant change to inputs and also training data, the latter is a change to inputs only in a less consitent pattern. We therefore have no reason to believe that a Model that is robust to adversarial changes to input would learn what parts of the training data are poisoned.

Q15.) (b) is unlikely.
(assuming (d) undergoes well informed code reviews)