Summary:
This week, we learned about decision trees and how they use parametric methods to make assumptions about the data distribution. We also discussed the XOR function in the context of loan risk factors, which was our case study for this unit. Additionally, we discussed the visual notation used to represent growing trees and decision stumps in combination with the threshold split. More generally, we talked about the pros and cons of using decision trees, why ensemble methods are useful and how to implement them. AdaBoost is a better version of a random forest in many cases. We addressed the fact that decision trees tend to be very overfit, but how it can actually be good. Then, we briefly touched on neural networks and their general idea.

Decision Trees

- Error = #mistakes / data points
  - The split with the lowest error becomes the first split in the decision tree
- Threshold split: this is used for real values, such as income. Midpoints are used to avoid testing every possible value
- Trees are used for classification or regression
- Pros:
  - Easy to interpret
  - No normalization required because you're not comparing features to each other
  - Preprocessing not required
  - Non-linear decision decisioon boundaries are allowed
  - Multiclass classification is possible
- Cons:
  - Prone to overfitting when the tree is deep
  - Only allows axis parallel decision boundaries
- Parametric methods: make assumptions about the data distribution. For example linear regression assumes that the data is linearly distributed.
- Non-parametric methods don't assume anything about the data
- XOR function: true if and only if both re the same. False in all other cases.
- Ensemble methods
  - Combine weaker models together to create a more powerful model
  - Random Forest (Bagging)
    - Many trees, the majority decision is chosen
    - Training is called bootstrapping, where random sampling with replacement is used to create the train/validation splits
    - This is because you need your decision trees to be different for random forest to work
    - Deep, overfitted trees are good because they have low bias, so they have low bias when averaged together
    - When you add more trees, you have less variance
  - AdaBoost (Boosting)
    - Only use decision stumps instead of high depth trees
    - Weighted majority voting system
      - Normalization: normalize so everything adds up to 1
      - Usually, both AdaBoost and decision trees are used, both bagging and boosting
    - W=1/2 ln((1-weightederror(f))/(weightederror(f)))

Uncertainties:

- How does one hot encoding work?
- What is the signum function?
- How do you solve the slido question? I still don't get it
- What does T stand for in normalization?
- Important: Use model ensemble with both boosting and bagging
- How do bagging and boosting differ?
- What are the steps in the AdaBoost algorithm?
- I don't understand anything on slide 32
- Why does using epochs use less resources?