Chapter 1:

1. What are the phases in the ML lifecycle? What type of evaluation is suitable for each phase?

- ML Lifecycle breaks down into two phases, prototyping and evaluation.
- Prototyping is the phase which involves testing out different models for our data, also called model selection.
- Once it has been through prototyping the model will go through testing over live data, which is the evaluation phase.
- Evaluation types can be classified as online evaluation and offline evaluation.
- Online evaluation measures live metrics of the selected model on the live data, whereas
- offline evaluation measures them on historical data or even sometimes live data.

2. Why is the process of evaluation complicated?

Two reasons:
First

- Evaluation, both online and offline, may measure different metrics
- Offline might use metrics like accuracy, precision-recall etc.
- Also, training and validation might use different metrics
- Online evaluation might measure business metrics, which might not be available at training

Second

- Distribution drift of data occurs, i.e., the distribution of data changes over time. But many models assume that the distribution is stationary.

3. What is the difference between model parameter and hyperparameter?
Give examples of each for different models that you have learned. For example, in case of decision tree, what would be a model parameter and what would be a hyperparameter?

Model Parameters:

- knobs that need to be tweaked and are learned from the data.
- These are tweaked by the training algorithm.
- Example: class separating line (in a classification problem) is model parameter.

Hyperparameters:

- Not learned by the training method but also need to be tuned.
- Example: number of features to be used for class separation.

Decision Trees:
Model Parameter: best attribute to use at a particular node
Hyperparameter: Number of decision/leaf nodes
Perceptron:
Model Parameter: weights of features
Hyperparameter: error function
Artificial Neural Networks:
Model Parameter: Weights of the network
Hyperparameter: Number of hidden layers
Support Vector Machines:

Model Parameter: weights of the separating boundary
Hyperparameter: Kernel Function

Naïve Bayes:
Model Parameter: Prior of each class
Hyperparameter: number of features

Logistic Regression:
Model Parameter: Weights for each feature
Hyperparameter: number of features

k-Nearest Neighbour:
Model Parameter: Non-parametric model
Hyperparameter: k (Number of nearest neighbours)

Chapter 2:

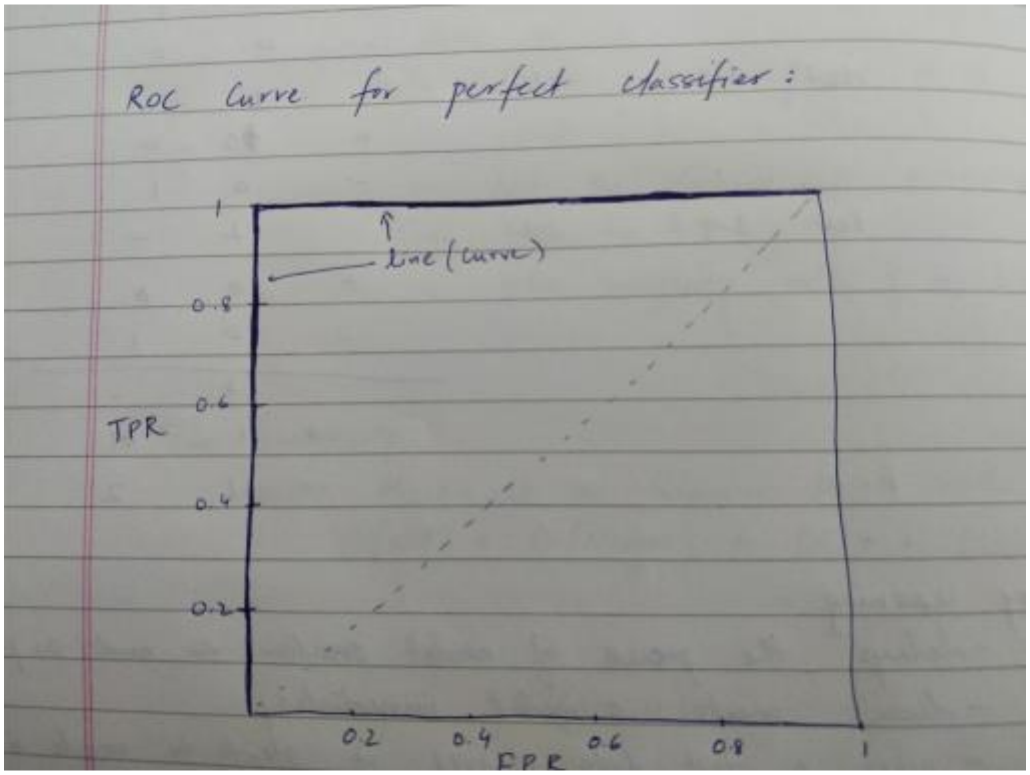
1. What would be the equation for log-loss metric for a binary class dataset when using Logistic Regression as the model? Write the equation in the most simplified format.

For logistic regression, we may replace the class output probability with the class output.

Log loss = (∑ -(yt log(yp) + (1 - yt) log(1 - yp))) / N {Summation over all the training examples}

yt = true label = {0, 1}
yp = predicted label = {0, 1}
N = Number of training examples

2. Read the section on ROC and draw the curve for the perfect classifier i.e. one that makes no mistakes.



3. Understand the concepts of precision, recall, and F1 score. Is it possible for precision and recall both to go up at the same time i.e. do they have a positive or negative correlation? Explain.

Precision = TP/(TP + FP)
Recall = TP/(TP + FN)

Precision

- Precision definition shows that lowering the threshold of a classifier may increase the denominator, by increasing the number of results returned.
- If the threshold was set too high, the new results may all be true positives, which will increase precision. May or may not affect recall.
- If previous threshold was almost correct or too low, further lowering the threshold may introduce more false positives, decreasing precision.

Recall

- TP + FN does not depend on the classifier threshold.
- => that lowering the classifier threshold may increase recall, by increasing the number of true positive results.
- OR it is also possible that lowering the threshold may leave recall unchanged, while the precision fluctuates.

4. What are some of the things you should be cautious about when choosing evaluation functions, and when analyzing the distribution of data.

Evaluation Functions:

- Sometimes training error functions are chosen such that they are different from evaluation functions.
- This confuses model as it is trying to optimize something and is evaluated on something else.
- It is always better to train the model to optimize the metric it will be evaluated on.

Distribution of data:

- Always be on the lookout for data skew. Data skew = where some classes are very rarely found OR where very large/small outliers drastically change output
- Imbalanced classes affect per-class accuracy, so detection through per-class accuracy is possible.
- For imbalanced classes, any metric that gives equal weight to all classes is not useful.
- Large outliers cause problems for regressors.
- Careful data cleaning should be done to handle large outliers.
- Reformulating the task to make it less sensitive to outliers will also help.

Chapter 3:

1. Why is model selection and hyperparameter tuning done using results of validation dataset and not training dataset?

Model selection and hyperparameter tuning is done using the results of validation dataset because:

- Validation dataset has not been seen by the model before.
- The model selection is based on how good model can generalize which is why it needs to be evaluated on data that is not seen during the training process.
- Model Selection is based on generalization error, which is better obtained from validation dataset. The lesser the generalization error, the better the model.
- Over validation dataset, the generalization error needs to be minimized, tuning the parameters and hyperparameters over the results of the validation dataset leads to better model selection.

2. If you are given just one dataset, what are the 3 ways in which you can obtain validation dataset(s) from the given dataset? Explain each and give advantages and disadvantages of each.

Hold-Out Validation:

- Simply hold out part of the data for validation
- Advantages:
    - Simple to program
    - Fast to run
- Disadvantages:
    - Statistically less powerful
- Use this when enough data s present

k-fold Cross Validation:

- First divide the training set k-fold. Assign one of them as a hold-out validation set and the other k-1 as training. Do this for k different validation sets and take the average evaluation metrics of the k. Repeat this for all the hyperparameter combinations and choose the bets hyperparameter combinations.
- Advantages:
    - Useful when working with small datasets

Bootstrap and Jackknife:

- Bootstrap is resampling technique. Generates multiple datasets by sampling from single dataset. Each of the sampled datasets can be used for evaluation. Can also calculate things like variance or confidence interval.
- Jackknife is essentially leave-one-out cross-validation.
- Advantages:
    - Affects similar to cross-validation
    - Resampling is done with replacement, unlike in jackknife
    - Gives an empirical distribution of data
    - Good for small datasets

3. We live in an era of Big Data. Why don't we just get more data rather than obtaining validation data from given data?

- Validation data extraction techniques were invented in the age of small data.
- These techniques help us calculate average estimate of performance and confidence interval
- Prior to this era, the data collection task was difficult and these methods helped in getting statistically convincing conclusions.

4. Why should training, and evaluation data never be mixed?

- It leads to a bad estimate of generalization error, which leads to bad model selected.

Chapter 4:

1. What is the role of a hyperparameter in a model? When are model parameters learned? Do you think a model is more strongly influenced by model parameters or hyperparameters? Explain you reasoning?
Note: There is no right or wrong answer, but you should explain your thinking.
Role of Hyperparameter in a model

- Some hyperparameters (regularization parameters) control the capacity of the model i.e. how flexible the model is, how many degrees of freedom it has in fitting the training data.
- Regularization hyperparameter helps prevent overfitting which means the model is more flexible.
- Optimization techniques use error functions as hyperparameters.
- Some optimization methods use a convergence threshold as hyperparameter.

Model Parameters learned:

- During the training phase

Influence:

- Model parameters are an essential part of a model. Without proper model parameters the model will not be able to perform as good as it is.
- Hyperparameters are over-the-top tuning parameters which help the already developed model. They are also sometimes essential for a model depending on them, such as a k-nearest neighbor classifier or a random forest for which the number of trees to be generated needs to be given.
- I feel that model parameters are a more strong influence to the model. This is because once the model parameters are trained, the hyperparameters may be able to tweak the performance by maybe 5-10 percent. But if the model parameters are trained bad, there is no way the hyperparameters can improve the performance of the model (unless and until they are actually an integral part of the model).

2. What are some techniques for hyperparameter tuning? Explain each briefly and in your own words

Grid Search:

- Picks a grid of hyperparameters, evaluates all and returns the best
- Simple to setup and parallelize
- Most expensive computation time

Random Search:

- Like grid search, but only evaluates a random sample of points from the grid and returns the best among them.
- Cannot beat the best found by grid search but is faster.
- If at least 5%of the points on the grid yield a close to optimal solution, then random search with 60 trials will find the region with high probability.

Smart Hyperparameter Tuning:

- Better techniques pick a few hyperparameter settings, evaluate their quality and decide if sampling next would be useful.
- Iterative and sequential process.
- Not parallelizable.
- Tries to make fewer calculations overall and save on the overall computation time.
- Also contain parameters of their own that need to be tuned. Which is sometimes crucial.
- Derivative Free optimization:
    - Employ heuristics to determine where to sample next.
- Bayesian Optimization & random forest smart-tuning:
    - models the response surface with another function, then samples more points based on what the model outputs.