**TRUE/FALSE:**

1. True: Pruning a decision tree can help prevent overfitting by reducing the model complexity, which often leads to lower test error.
2. False: In k-fold cross-validation, the data is split into k subsets. Each subset is used once as the test set while the remaining k-1 subsets are used as the training set. This process is repeated k times, not leaving k inputs out each time.
3. True: Gradient Boosting involves building models in a stage-wise fashion and optimizing for a loss function by performing gradient descent in function space.
4. False: While random splitting is common, it's not always necessary.
5. False: It is theoretically possible for a classifier to overfit the training data perfectly (0% training error) and perform very poorly on the test data (potentially 100% test error), especially if the test data is very different from the training data.
6. False:Increasing regularization typically increases the bias and reduces the variance of the classifier.
7. False: Without a depth limit, the ID3 algorithm will create a tree that fits the training data perfectly (likely very large and complex), not necessarily the most compact tree.
8. False: All classifiers make some assumptions about the data.
9. True: Random Forests create multiple CART trees (which are high variance) and reduce the overall variance by averaging their predictions, a process known as Bagging.
10. True: This statement describes a known property of k-nearest neighbor classifiers: as the size of the training set grows infinitely, their error approaches at most twice the Bayes error rate.
11. False: The time complexity for finding the best split in a regression tree is typically O(n log n) per split, not O(n^2).
12. True: The Bayes optimal error is the lowest possible error that can be achieved, considering the inherent noise in the data. It is the error of a classifier that has complete knowledge of the underlying data distribution.

**ERM and SVM:**

1. SVM- loss function is hinge loss: max(0,1−y(w⋅x+b))

Regularisation is L2: ∑∣wj​∣^2

Lasso- loss function is MSE: 1/n\*∑(yi​−w⋅xi​)^2

Regularisation is L1 : ∑∣wj​∣

Ridge- loss function is MSE: 1/n\*∑(yi​−w⋅xi​)^2

Regularisation is L2 : ∑∣wj​∣^2

All loss functions are:

Hinge Loss (SVM)

Mean Squared Error (MSE)

Absolute Error (L1 Loss)

Log Loss (Cross-Entropy Loss)

Huber Loss

1. Hinge Loss (SVM): A

Mean Squared Error (MSE): B

Absolute Error (L1 Loss): C

Log Loss (Cross-Entropy Loss) : D

Huber Loss: E

**BIAS AND VARIANCE:**

1. The two possible reasons maybe:

a.High Bias: Limited depth causes the model to be too simple to capture the underlying patterns in the data.

b.Insufficient Features: The features provided may not be informative enough for the decision tree to make accurate predictions.

1. Overfitting: The model is too complex and fits the training data perfectly, including noise, which leads to poor generalization to the test data.
2. Bagging simulates creating multiple datasets by bootstrapping (sampling with replacement) from the original dataset and training separate models on these datasets. By averaging the predictions of these models, bagging reduces variance because the individual errors of the models average out, leading to a more stable and robust aggregate model.

**kNN and Curse of Dimentionality:**

1. Reduce Dimensionality: Apply dimensionality reduction techniques like PCA to reduce the number of dimensions to escape curse of dimensionality.
2. a) The ranking of nearest neighbors remains the same, so the accuracy of kNN is typically unaffected by using squared Euclidean distance instead of Euclidean distance.

b) The recommendation to use dimensionality reduction remains valid. The squared distance does not affect these techniques.

1. Kanye W. should be worried about the curse of dimensionality because of huge number of dimensions and very less data. It will cause sparse data which will make it difficult for the kNN classifier to get useful information.
2. Bias-Variance trade off:

Small k: Low bias, high variance.

Large k: High bias, low variance.

1. kNN: Suitable when the decision boundary is very complex and nonlinear, and the dataset is relatively small.

Linear SVM: Suitable when the decision boundary is approximately linear, and the dataset is large.

**Decision Trees:**

2. Maximized: When classes are uniformly distributed.

Minimized: When all items belong to a single class.

3. Decision trees are "myopic" because they make locally optimal decisions at each node without considering the global structure of the tree. Each split is chosen based on the immediate reduction in impurity, not on the potential long-term impact.

4. a)Pruning: Reduce the size of the tree by removing splits that have little importance.

b) Setting a Maximum Depth: Limit the depth of the tree to prevent it from becoming too complex.

**Bagging and Boosting:**

1. I think it is FALSE because validation data is necessary to tune hyperparameters and prevent overfitting. Without validation, there is a risk of overfitting to the training data.
2. Loss function of adaboost is Exponential Loss function.