**True or False(Mention reason)**

1. True: Pruning can reduce the size of a decision tree and sometimes improve its ability to correctly classify new, unseen data. While it may perform worse on the training data, its overall classification performance can improve.

2. False: In k-fold cross-validation, the dataset is divided into k roughly equal parts. Each part is used as the test set once, while the model is trained on the remaining k-1 parts. This process is repeated k times, not by leaving out k inputs at each step.

3. True: Gradient Boosting builds models incrementally. Each new model aims to correct the errors made by the previous models by using gradients, similar to gradient descent.

4. False: While random splitting of data is common, stratified splitting is often used for classification to ensure both training and test sets have a similar distribution of classes. Random splitting is not the only method, nor always the best.

5.False: If a model makes no mistakes on the training data, it might be overfitting. However, it’s highly unlikely to make 100% mistakes on the test data because some patterns will usually be similar between the training and test sets.

6.False: Increasing regularization simplifies a model, which can lead to underfitting. This means the model's bias increases, not decreases.

7.False: The ID3 algorithm, without a depth limit, can create a very large, overfitted tree, not necessarily a compact one.

8.True: All classifiers make some assumptions about the data. The "best" classifier depends on how well these assumptions align with the true nature of the data.

9. True: Random Forests use a method called bagging to create many decision trees from random subsets of the data. They also introduce additional randomness when building the trees. This helps reduce the overall error by averaging the results of all the trees.

10. True: The k-nearest neighbor algorithm has an error that, at worst, is no more than twice the Bayes error as the dataset size becomes very large.

11. False: Squared loss regression trees generally have a time complexity of O(n log n) per split, similar to other decision tree algorithms.

12. False: The Bayes optimal error is the lowest possible classification error given the true probability distributions. It includes noise but is not solely due to noise.

**ERM and SVM**

**SVM (Support Vector Machine)**

**Loss Function:** Hinge Loss- Evaluates the distance between the prediction and the correct classification.

**Regularizer:** L2 Regularization-**:** Discourages large weights to avoid overfitting.

**LASSO (Least Absolute Shrinkage and Selection Operator)**

**Loss Function:** Mean Squared Error (MSE)- **:** Calculates the average squared difference between the predicted and actual values.

**Regularizer:** L1 Regularization- Adds the absolute values of weights, promoting sparsity by encouraging some weights to be zero.

**Ridge Regression**

**Loss Function:** Mean Squared Error (MSE)- Computes the average squared difference between predicted and actual values.

**Regularizer:** L2 Regularization-**:** Discourages large weights to reduce the risk of overfitting.

**3) Gradient Descent**

**Least Squares Loss:** Effective for minimizing this loss in both LASSO and Ridge Regression

**Hinge Loss:** Usable but requires careful handling.

**Newton’s Method**

**Least Squares Loss (Ridge Regression):** Efficient for minimizing this loss.**Hinge Loss and L1 Regularization (LASSO):** Challenging to use due to their lack of smoothness and computational complexity.

**Bias and Variance**

1) **High Training and Testing Errors with Limited Depth**

* **High Bias:** The decision trees are too simple (shallow) to capture the complexity of the stock market data, leading to high errors on both training and testing sets.
* **Irrelevant Features:** The features used might not be relevant for predicting stock prices, contributing to poor model performance.

2) **High Errors with Unlimited Depth**

* **Noisy Data:** The stock market data might be very noisy, making it difficult to identify useful patterns, resulting in high errors.
* **Overfitting with Limited Data:** When there isn't enough training data, even deep trees can't generalize well and may overfit the noise in the data.

3) **Bagging (Bootstrap Aggregating)**

* **Simulation:** Bagging simulates the effect of using multiple different datasets by creating random samples (with replacement) from the original dataset.
* **Variance Reduction:** It reduces variance by averaging the predictions from many trees, thereby smoothing out the fluctuations and errors from any single tree.

4) **Boosting Shallow Trees**

* **Bias Reduction:** Boosting reduces bias over time by combining many weak models into a stronger ensemble model.
* **Variance:** Initially, variance might increase slightly due to the addition of new models, but overall, boosting helps in reducing variance by making the model more robust and less sensitive to fluctuations in the data.

**kNN / Curse of Dimensionality**

1. **Speeding Up kNN Classifier**
   * **Dimensionality Reduction:** Use PCA or t-SNE to reduce the number of features.
   * **Approximate Nearest Neighbours:** Use methods like LSH or KD-trees to quickly find nearest neighbours.
   * **Efficient Data Structures:** Utilize Ball Trees or KD-trees for faster searching.
   * **Parallelization:** Use multiple processors to split the workload.
   * **Data Preprocessing:** Normalize or scale data to make distance calculations easier.
2. **Impact of Using Squared Euclidean Distance**
   * **kNN Accuracy:** Using squared Euclidean distance doesn't change the order of the nearest neighbours. The closest points remain the same.
   * **Validity of Advice:** Yes, the suggestions in question 1 still apply. These methods work with squared Euclidean distance too.
3. **Curse of Dimensionality**
   * **Concern:** Yes, Kanye should worry. With very high dimensions (d = 1,000,000) and few data points (n = 5,000), the data points become sparse, making it hard for kNN to find meaningful neighbours, leading to poor performance.
4. **Neighbourhood Size k Effects**
   * **Bias:** Increases with larger k, making the model more general and smoothing over details.
   * **Variance:** Decreases with larger k, making the model less sensitive to specific data points.
   * **Trade-off:** Small k means low bias but high variance. Large k means high bias but low variance.
5. **Choosing Between kNN and Linear SVM**
   * **kNN Classifier:** Best for small datasets with complex, non-linear patterns. Example: Medical diagnosis with few records and complex symptom relationships.
   * **Linear SVM Classifier:** Best for large datasets with linear or nearly linear patterns. Example: Text classification like spam detection, where the data is high-dimensional but roughly linear.

**Decision Matrix:**

1. **Prediction Value at a Leaf of a Regression Tree**
   * **Prediction Value:** The prediction value at a leaf of a regression tree with squared-loss impurity is the average of the target values of the data points in that leaf.
   * **Proof of Optimality:** This is optimal because the average minimizes the squared error.
2. **Gini Index for 3 Classes**
   * **Maximized:** The Gini Index is highest when all classes are equally likely. For 3 classes, this happens when p1=p2=p3=1/3.
   * **Minimized:** The Gini Index is lowest when all items belong to one class, meaning pk=1 for one class and 0 for the others.
3. **Decision Trees are "Myopic"**
   * **Explanation:** Decision trees are considered "myopic" because they make decisions based on the best split at each node without considering the impact on future splits. They focus on optimizing each node locally rather than the entire tree globally.
4. **Preventing Overfitting in Decision Trees**
   * **Method 1: Pruning:** Remove branches that don't significantly improve accuracy on the validation set.
   * **Method 2: Limiting Tree Depth:** Set a maximum depth for the tree to prevent it from becoming too complex and capturing noise in the training data.

**Boosting and Bagging**

1. **Random Forests and Validation/Test Data**
   * **Claim:** Ludwig van Beethoven says he doesn't need validation or test data for Random Forests.
   * **Answer:** False.
   * **Reason:**
     + **Generalization:** Without validation/test data, you can't be sure the model performs well on new data; it might overfit to the training data.
     + **Parameter Tuning:** Parameters like the number of trees and their depth need to be adjusted using validation data.
     + **Cross-Validation:** Techniques like k-fold cross-validation are used to split data for parameter tuning and performance evaluation.
   * **Conclusion:** Exponential loss is an upper bound on training error because it penalizes not only misclassified points but also those close to the decision boundary.