Okay, here's a breakdown of the theory behind common machine learning models you might encounter. Understanding these concepts will be crucial for your practical exam.

Supervised Learning Models

Supervised learning models learn from labeled data, meaning each data point has an input (feature) and a corresponding output (label or target).

1. Linear Regression

- **Core Concept:** Linear Regression is used to predict a **continuous numerical value** (dependent variable) based on one or more input features (independent variables). It assumes a linear relationship between the inputs and the output.

  1

- **How it Works:**
  - It tries to find the best-fitting straight line (in simple linear regression with one input feature) or a hyperplane (in multiple linear regression with multiple input features) that describes the relationship between the independent variable(s) (X) and the dependent variable (Y).
  - The equation for a simple linear regression is: $Y=\beta 0+\beta 1X+\epsilon$
    - Y: Dependent variable (what you're trying to predict)
    - X: Independent variable (the predictor)
    - $\beta 0$: Intercept (the value of Y when X is 0)
    - $\beta 1$: Coefficient or slope (the change in Y for a one-unit change in X)
    - ϵ: Error term (the difference between the actual and predicted value)
  - For multiple linear regression, the equation extends: $Y=\beta 0+\beta 1X1+\beta 2X2+\ldots+\beta nXn+\epsilon$
  - The "best fit" is usually determined by minimizing the **sum of squared residuals (SSR)** or **Residual Sum of Squares (RSS)**. This method is known as **Ordinary Least Squares (OLS)**. Residuals are the differences between the observed values and the values predicted by the model.

- **Key Terms/Concepts:**
  - **Dependent Variable:** The variable you want to predict.
  - **Independent Variable(s):** The variable(s) used to make the prediction.
  - **Coefficients (β):** Weights assigned to independent variables, indicating their importance and the direction of their relationship with the dependent variable.
  - **Intercept (β0):** The value of the dependent variable when all independent variables are zero.
  - **Residuals:** The differences between actual and predicted values.
  - **Cost Function (e.g., Mean Squared Error - MSE):** A function that measures the model's performance. The goal is to minimize this function.
  - **R-squared (R2):** A statistical measure of how close the data are to the fitted regression line. It represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s).

    2

    3
- **Assumptions:**
  1. **Linearity:** The relationship between the independent and dependent variables is linear.
  2. **Independence of Residuals:** The residuals are independent of each other (no autocorrelation).
  3. **Homoscedasticity:** The residuals have constant variance at every level of X.
  4. **Normality of Residuals:** The residuals are normally distributed.
  5. **No or little multicollinearity:** Independent variables are not highly correlated with each other (for multiple linear regression).
- **Use Cases:** Predicting house prices based on features like size and location, forecasting sales, estimating exam scores based on study hours.
- **Advantages:**
  - Simple to understand and interpret.

    - Computationally inexpensive.
    - Provides a clear mathematical equation for the relationship.
- **Disadvantages:**
    - Assumes a linear relationship, which might not hold true for all datasets.
    - Sensitive to outliers.
    - May not be suitable for complex, non-linear relationships.

2. Logistic Regression

- **Core Concept:** Logistic Regression is used for **binary classification problems**, where the output variable is categorical and has two possible outcomes (e.g., Yes/No, True/False, 0/1). It can also be extended for multi-class classification (Multinomial Logistic Regression). It predicts the probability that an instance belongs to a particular class.
- **How it Works:**
    - Instead of fitting a straight line directly to the data (like linear regression), logistic regression uses a **logistic function (or sigmoid function)** to transform the output of a linear equation into a probability value (between 0 and 1).
    - The sigmoid function is an S-shaped curve: $P(Y=1)=1+e-z1$
        - Where z is the linear combination of input features: $z=\beta0+\beta1X1+\beta2X2+...+\beta nXn$
    - The output $P(Y=1)$ is the estimated probability that the instance belongs to class 1.
    - A threshold (commonly 0.5) is then used to classify the instance: if $P(Y=1)>0.5$, predict class 1; otherwise, predict class 0.
    - The coefficients ($\beta0,\beta1,...$) are typically estimated using **Maximum Likelihood Estimation (MLE)**, which finds the parameter values that maximize the likelihood of observing the given data.
- **Key Terms/Concepts:**
    - **Sigmoid Function (Logistic Function):** The S-shaped function that maps any real-valued number into a value between 0 and 1.

    - **Log-odds (Logit):** The natural logarithm of the odds (odds=1−P(Y=1)P(Y=1)). The linear equation z directly predicts the log-odds: log(1−PP)=β0+β1X1+...+βnXn.
    - **Maximum Likelihood Estimation (MLE):** The method used to estimate the parameters of the model.
    - **Decision Boundary:** The threshold that separates one class from another. In logistic regression, this is often linear.
- **Assumptions:**
    1. The dependent variable is binary (for binary logistic regression) or ordinal/nominal (for respective variants).
    2. Observations are independent of each other.
    3. There is little or no multicollinearity among the independent variables.
    4. There is a linear relationship between the independent variables and the log-odds of the dependent variable.
- **Use Cases:** Spam email detection (spam/not spam), medical diagnosis (disease/no disease), credit card fraud detection (fraudulent/not fraudulent), customer churn prediction.
- **Advantages:**
    - Outputs probabilities, which can be useful for ranking or understanding confidence.
    - Relatively simple to implement and interpret (especially the impact of individual features through odds ratios).
    - Computationally efficient.
    - Performs well for linearly separable classes.
- **Disadvantages:**
    - Assumes a linear relationship between the independent variables and the log-odds.
    - May not perform well with complex non-linear relationships or when feature interactions are significant unless explicitly modeled.
    - Can be prone to overfitting with high-dimensional data if not regularized.

3. Support Vector Machines (SVM)

- **Core Concept:** SVM is a powerful and versatile supervised learning algorithm used for **classification, regression (Support Vector Regression - SVR), and outlier detection**. For classification, its primary goal is to find an optimal **hyperplane** in an N-dimensional space (where N is the number of features) that distinctly classifies the data points.
- **How it Works (for Classification):**
    1. **Hyperplane:** A decision boundary that separates the data points of different classes. In 2D, it's a line; in 3D, it's a plane, and so on.
    2. **Optimal Hyperplane:** SVM aims to find the hyperplane that has the **maximum margin**.
    3. **Margin:** The distance between the hyperplane and the closest data points from either class. These closest points are called **support vectors**.
    4. **Support Vectors:** These are the data points that lie closest to the decision boundary (hyperplane). They are critical because they define the margin. If other points are removed (those not support vectors), the hyperplane would not change.
    5. **Hard Margin vs. Soft Margin:**
        - **Hard Margin:** Used when the data is perfectly linearly separable. It doesn't allow for any misclassifications.
        - **Soft Margin:** Used when the data is not perfectly linearly separable (which is common in real-world datasets). It allows for some misclassifications by introducing a penalty term (C parameter) for violating the margin. The C parameter controls the trade-off between maximizing the margin and minimizing the classification error. A smaller C creates a wider margin but allows more violations.

        4
    6. **Kernel Trick:** For non-linearly separable data, SVM can use the kernel trick. This involves mapping the data points into a higher-dimensional feature space where they become linearly separable. Common kernels include:

        - **Linear Kernel:** For linearly separable data (K(xi,xj)=xiTxj).
        - **Polynomial Kernel:** K(xi,xj)=(γxiTxj+r)d
        - **Radial Basis Function (RBF) Kernel:** K(xi,xj)=exp(−γ∣∣xi−xj∣∣2) (Very popular for non-linear data).
        - **Sigmoid Kernel:** K(xi,xj)=tanh(γxiTxj+r)
- **Key Terms/Concepts:**
    - **Hyperplane:** The decision boundary.
    - **Support Vectors:** Data points closest to the hyperplane.
    - **Margin:** The gap between the hyperplane and the support vectors.
    - **Kernel Trick:** A method to handle non-linear data by transforming it into a higher dimension.
    - **C (Regularization Parameter):** Controls the trade-off between a smooth decision boundary (larger margin) and classifying training points correctly.
    - **Gamma (γ):** A parameter for non-linear kernels (like RBF and Polynomial) that defines how much influence a single training example has.
- **Assumptions:** Feature scaling (standardization or normalization) is often highly recommended for SVM to perform well, as it's sensitive to the scale of input features.
- **Use Cases:** Image classification, text categorization, bioinformatics (e.g., protein classification, cancer classification), face detection, handwriting recognition.
- **Advantages:**
    - Effective in high-dimensional spaces (even when the number of dimensions is greater than the number of samples).
    - Memory efficient because it uses only a subset of training points (the support vectors) in the decision function.

      5
    - Versatile due to different kernel functions that can be specified.

    - Generally robust to overfitting, especially in high-dimensional space, if the C parameter is chosen carefully.
- **Disadvantages:**
    - Can be computationally intensive and slow to train, especially with very large datasets.
    - Choosing the right kernel and its hyperparameters (like C and gamma) can be tricky and often requires careful tuning (e.g., using cross-validation).
    - The model can be difficult to interpret directly (often considered a "black box"), especially with non-linear kernels.

4. K-Nearest Neighbors (KNN)

- **Core Concept:** KNN is a non-parametric, **instance-based learning** (or lazy learning) algorithm used for both **classification and regression**. It makes predictions based on the 'K' closest training examples in the feature space.
- **How it Works:**
    1. **Storing Data:** KNN stores the entire training dataset.
    2. **Calculating Distance:** When a new, unseen data point needs to be classified (or its value predicted for regression):
        - It calculates the distance between the new data point and all the data points in the training set. Common distance metrics include:
            - **Euclidean Distance:** ∑i=1n(qi−pi)2

              6

            - **Manhattan Distance:** ∑i=1n ∣ qi−pi ∣
            - **Minkowski Distance:** (∑i=1n(∣ qi−pi ∣)p)p1 (Euclidean is a special case where p=2, Manhattan where p=1)
    3. **Finding Neighbors:** It identifies the 'K' training data points that are closest (i.e., have the smallest distances) to the new data point. 'K' is a user-defined integer.
    4. **Making Prediction:**

    - **For Classification:** The new data point is assigned to the class that is most common among its 'K' nearest neighbors (majority voting).

      7

    - **For Regression:** The predicted value for the new data point is the average (or median) of the values of its 'K' nearest neighbors.

- **Key Terms/Concepts:**
    - **'K':** The number of nearest neighbors to consider. This is a crucial hyperparameter.
    - **Distance Metric:** The function used to measure similarity or dissimilarity between data points.
    - **Lazy Learning / Instance-Based Learning:** The algorithm doesn't learn an explicit discriminative function from the training data but defers computation until classification time.
    - **Non-parametric:** It doesn't make assumptions about the underlying data distribution.
- **Assumptions:**
    - Assumes that similar data points exist in close proximity in the feature space ("feature similarity").
    - Feature scaling (normalization or standardization) is very important because KNN relies on distances; features with larger scales can dominate the distance calculation.
- **Choosing 'K':**
    - Small 'K': Can be sensitive to noise and outliers; the decision boundary can be irregular.
    - Large 'K': Can oversmooth the decision boundary and might misclassify points if neighbors from other classes are included.
    - A common approach is to try different values of 'K' (e.g., using cross-validation) and choose the one that gives the best performance. Often, an odd 'K' is chosen for binary classification to avoid ties.
- **Use Cases:** Recommendation systems, image recognition, anomaly detection, credit scoring, pattern recognition in finance.
- **Advantages:**

    - Simple to understand and implement.
    - No explicit training phase is required (lazy learner).
    - Easy to add new data to the model.
    - Works well with multi-class problems.
    - Effective if the decision boundary is highly irregular.
- **Disadvantages:**
    - Computationally expensive during the testing/prediction phase because it needs to calculate distances to all training samples for each new point (especially with large datasets).
    - Performance can degrade significantly with high-dimensional data (known as the "curse of dimensionality").
    - Sensitive to irrelevant features and the scale of the data.
    - Needs careful selection of 'K' and the appropriate distance metric for the given data.
    - Requires a lot of memory to store the entire dataset.

5. Decision Tree

- **Core Concept:** A Decision Tree is a supervised learning algorithm that can be used for both **classification and regression** tasks. It creates a tree-like model of decisions. Each internal node represents a "test" on an attribute (feature), each branch represents an outcome of the test, and each leaf node represents a class label (in classification) or a continuous value (in regression).

  8

  9

- **How it Works:**
    1. **Recursive Partitioning:** The algorithm works by recursively splitting the training data into subsets based on the values of the input features.
    2. **Choosing the Best Split:** At each node, it selects the feature and the split point (for numerical features) or category (for categorical features) that best separates the data into distinct classes or reduces variance (for regression). Common criteria for "best split" include:

        - **Gini Impurity (Classification):** Measures the probability of misclassifying a randomly chosen element if it were randomly labeled according to the distribution of labels in the subset. A Gini impurity of 0 means all elements in the node belong to the same class.
          $Gini=1-\sum_{i=1}^{C}(p_i)^2$ (where p_i is the probability of class i)
        - **Information Gain (based on Entropy) (Classification):** Measures the reduction in entropy achieved by splitting the data on a particular feature. Entropy measures the impurity or randomness in a set of examples.
          $Entropy=-\sum_{i=1}^{C}p_i\log_2(p_i)$ $InformationGain=Entropy(parent)-\sum_j\frac{N_j}{N}Entropy(child_j)$
        - **Variance Reduction (Regression):** Used for regression tasks. It chooses the split that minimizes the variance in the target variable within the resulting subsets.
    3. **Stopping Criteria:** The splitting process continues until a stopping criterion is met, such as:
        - All instances in a node belong to the same class.
        - Maximum tree depth is reached.
        - The number of instances in a node is below a certain threshold.
        - No split further improves the impurity/variance.
    4. **Pruning:** After the tree is built, it might be pruned to reduce its complexity and prevent overfitting. Pruning involves removing branches that provide little predictive power.
- **Key Terms/Concepts:**
    - **Root Node:** The topmost node in the tree, representing the entire dataset.
    - **Internal Node (Decision Node):** A node that represents a test on a feature and has branches leading to other nodes.
    - **Leaf Node (Terminal Node):** A node that represents a class label (classification) or a value (regression) and has no further branches.
    - **Splitting:** The process of dividing a node into two or more sub-nodes.

- **Pruning:** The process of removing sub-nodes of a decision node to reduce overfitting.

  10

- **Impurity Measures (Gini, Entropy):** Metrics used to evaluate the homogeneity of labels at a node.

- **Algorithms:** ID3 (Iterative Dichotomiser 3), C4.5 (successor of ID3), CART (Classification and Regression Trees).
- **Assumptions:** Decision trees are non-parametric and make relatively few assumptions about the underlying data distribution.
- **Use Cases:** Medical diagnosis, credit risk assessment, customer churn prediction, fraud detection, bioinformatics.
- **Advantages:**
  - Simple to understand and interpret. The tree structure can be visualized.
  - Can handle both numerical and categorical data without much preprocessing (e.g., normalization is typically not needed).
  - Non-parametric: makes no assumptions about the distribution of the data.
  - Can capture non-linear relationships between features and the target.
  - Implicitly performs feature selection (important features tend to appear closer to the root).
- **Disadvantages:**
  - Prone to **overfitting**, especially with deep trees that learn the training data too well (can be mitigated by pruning or using ensemble methods like Random Forests or Gradient Boosting).
  - Can be unstable: small variations in the data can result in a completely different tree being generated.
  - Can create biased trees if some classes dominate the dataset.
  - For tasks involving many classes and a relatively small number of training examples, the tree might not perform well.
  - Decision boundaries are typically axis-parallel.

Unsupervised Learning Models

Unsupervised learning models work with unlabeled data, trying to find patterns, structures, or relationships within the data itself.

6. K-Means Clustering

- **Core Concept:** K-Means is an **unsupervised clustering algorithm** used to partition a dataset into 'K' distinct, non-overlapping clusters. The goal is to group similar data points together, where similarity is typically measured by distance (e.g., Euclidean distance). Each data point belongs to the cluster with the nearest mean (which serves as the cluster's prototype or **centroid**).
- **How it Works (Lloyd's Algorithm):**
  1. **Initialization:**
     - Choose the number of clusters, 'K'.
     - Randomly initialize 'K' centroids (cluster centers). These can be randomly selected data points from the dataset or initialized using other methods like K-Means++.
  2. **Assignment Step (Expectation):**
     - Assign each data point in the dataset to the nearest centroid. This is done by calculating the distance (e.g., Euclidean) from the data point to each of the 'K' centroids and assigning the point to the cluster whose centroid is closest.
  3. **Update Step (Maximization):**
     - Recalculate the centroids of the 'K' clusters. The new centroid for each cluster is the mean (average) of all data points assigned to that cluster in the previous step.
  4. **Iteration:**
     - Repeat the Assignment Step and Update Step until one of the stopping criteria is met:
       - The centroids no longer change significantly (convergence).
       - The cluster assignments of the data points no longer change.
       - A maximum number of iterations is reached.

- **Objective Function:** K-Means aims to minimize the **within-cluster sum of squares (WCSS)**, also known as **inertia**. This is the sum of the squared distances between each data point and the centroid of its assigned cluster. $WCSS=\sum_{i=1}^{K}\sum_{x\in C_i} \| x-\mu_i \|^2$
  - Where C_i is the i-th cluster, and μ_i is the centroid of cluster C_i.
- **Key Terms/Concepts:**
  - **'K':** The pre-defined number of clusters.
  - **Centroid:** The center of a cluster, typically the mean of the data points within that cluster.
  - **Inertia (WCSS):** The sum of squared distances of samples to their closest cluster center. Lower inertia generally means better clustering.
  - **Unsupervised Learning:** Learns from unlabeled data.
  - **Iterative Algorithm:** Achieves the solution through repeated steps.
- **Choosing 'K':**
  - **Elbow Method:** Plot WCSS against different values of 'K'. The "elbow" point on the plot (where the rate of decrease in WCSS sharply slows down) is often considered a good choice for 'K'.
  - **Silhouette Analysis:** Measures how similar a data point is to its own cluster compared to other clusters. The silhouette score ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

    11
- **Assumptions:**

  12
  1. Clusters are spherical (isotropic) in shape.
  2. Clusters are of similar size.
  3. Clusters have similar density.
  4. The number of clusters 'K' must be specified beforehand by the user.
- **Use Cases:** Customer segmentation (grouping customers with similar purchasing behavior), document clustering, image compression (color quantization), anomaly detection (points far from any centroid).

- **Advantages:**
  - Relatively simple to understand and implement.
  - Computationally efficient for large datasets (Lloyd's algorithm has a polynomial smoothed running time).
  - Easy to interpret the results (the centroids represent the "average" member of each cluster).
  - Scales well to large datasets.
- **Disadvantages:**
  - The number of clusters 'K' needs to be chosen beforehand, which can be challenging.
  - Sensitive to the initial placement of centroids; different initializations can lead to different final clusters (local optima). K-Means++ initialization helps mitigate this.
  - Struggles with clusters of varying shapes (non-spherical), sizes, and densities.
  - Sensitive to outliers, as they can pull centroids towards them.
  - Assumes clusters are convex.

Remember to not just memorize these but understand the core intuition behind each model, when to use them, and their inherent strengths and weaknesses. Good luck with your exam!