Data Driven - Machine Learning(Assignment 1)

Part 1 — Algorithm Overviews

Logistic Regression (LR)
Logistic Regression is a widely used algorithm for classification tasks, particularly binary classification. It works by modeling the relationship between the features (independent variables) and the probability of a particular outcome (dependent variable) using the logistic, or sigmoid, function. This function transforms a linear combination of inputs into a value between 0 and 1, which can be interpreted as a probability. LR is simple, interpretable, and fast to train, making it an excellent baseline model. One of its major strengths is that the coefficients can be examined to understand how each feature influences the prediction, which is useful for explaining results to non-technical stakeholders. However, LR assumes a linear relationship between the log-odds of the outcome and the features, so it may struggle with complex, non-linear patterns unless polynomial or interaction terms are added. It is also sensitive to outliers and multicollinearity, which can distort coefficients unless addressed through regularization (L1 or L2) or feature engineering. Despite its simplicity, LR remains a powerful tool, especially when interpretability and probabilistic outputs are important in decision-making contexts like healthcare, credit scoring, and risk prediction.

K-Nearest Neighbors (KNN)
K-Nearest Neighbors is a simple, instance-based algorithm used for both classification and regression. Instead of learning an explicit model during training, KNN stores the entire dataset and makes predictions by looking at the ‘k’ most similar training samples to a given query point. Similarity is usually measured using a distance metric like Euclidean, Manhattan, or cosine similarity, depending on the problem. For classification, the algorithm assigns the most common label among the neighbors; for regression, it averages their values. KNN is intuitive and non-parametric, meaning it makes no strong assumptions about the data distribution. However, it has several limitations. First, it can be computationally expensive during prediction since it needs to calculate distances to all training points. Second, its performance heavily depends on the choice of ‘k’—too small a value can lead to noisy predictions, while too large can oversmooth decision boundaries. It is also sensitive to irrelevant features and feature scaling; without normalization, features with larger ranges can dominate distance calculations. Furthermore, in high-dimensional spaces, distances become less meaningful (curse of dimensionality), reducing its effectiveness. Still, KNN can work very well for small datasets with well-behaved feature spaces, especially when interpretability and simplicity are desired.

Decision Tree
A Decision Tree is a supervised learning method used for both classification and regression tasks. It works by recursively splitting the data into subsets based on the feature that results in the highest information gain (using measures like Gini impurity, entropy, or variance reduction). The final structure resembles a flowchart, where internal nodes represent feature tests, branches represent the outcomes of these tests, and leaf nodes represent predictions. Decision Trees are highly interpretable, as they produce human-readable rules, making them popular in domains where explainability is important. They can naturally handle both numerical and categorical data, capture non-linear relationships, and require little data preprocessing—no need for feature scaling or normalization. However, they are prone to overfitting, especially when allowed to grow deep without pruning or regularization. Overfit trees may capture noise rather than underlying patterns, leading to poor generalization. They can also be unstable, as small changes in the data can result in very different tree structures. While single Decision Trees may have moderate accuracy compared to more advanced models, their interpretability and ability to handle complex decision boundaries make them a valuable tool, especially when combined into ensembles like Random Forests or Gradient Boosted Trees.

Support Vector Machine (SVM)
Support Vector Machines are powerful supervised learning models primarily used for classification but also applicable to regression. SVM works by finding the optimal hyperplane that best separates data points of different classes in the feature space. The goal is to maximize the margin—the distance between the hyperplane and the nearest data points from each class, called support vectors. This focus on margin maximization helps SVM generalize well, especially in high-dimensional spaces. When data is not linearly separable, SVM uses the kernel trick to project it into a higher-dimensional space where a linear separation is possible. Popular kernels include the Radial Basis Function (RBF), polynomial, and sigmoid. SVM performs particularly well in text classification, image recognition, and bioinformatics. However, it has some limitations: it can be computationally expensive on large datasets, sensitive to the choice of hyperparameters (C, gamma), and less interpretable compared to simpler models like Logistic Regression or Decision Trees. Moreover, while SVMs can output decision scores, they do not directly provide calibrated probabilities unless paired with additional methods like Platt scaling. Despite these challenges, SVM remains a go-to method for datasets with complex but separable patterns.

———

Part 2 — Application Scenarios

1) High-Dimensional Data (text, gene expression)

In high-dimensional datasets—where the number of features is very large compared to the number of samples—SVM (particularly with a linear kernel) is often the best choice. Text classification problems, such as spam detection, sentiment analysis, and topic categorization, often have thousands of features (words) but relatively fewer samples. SVM handles this by focusing on the most important points (support vectors) to define the decision boundary, ignoring the rest. This reduces overfitting risk and improves generalization. Logistic Regression can also perform well in this space, but SVM tends to produce slightly better margins, which is critical in high dimensions. KNN performs poorly here because distances lose meaning when there are too many features, and Decision Trees can overfit heavily, creating complex partitions that do not generalize. Moreover, SVM's ability to handle sparse representations (common in text data) makes it an efficient and robust choice. In bioinformatics, such as gene expression classification, SVM can process thousands of gene-related features, maintaining high accuracy while avoiding the computational cost of trying to model every feature interaction explicitly.

2) Imbalanced Dataset (fraud, rare disease)
In scenarios where one class is significantly less frequent—such as fraud detection, rare disease diagnosis, or defect identification—Logistic Regression with class weighting is an excellent choice. Class imbalance can cause standard classifiers to be biased toward the majority class, leading to poor detection of rare events. By adjusting class weights, LR can penalize misclassifications of minority class instances more heavily, improving recall without sacrificing too much precision. Additionally, LR's probabilistic outputs allow for threshold tuning, which is vital in imbalanced settings where costs of false negatives are high. For example, in fraud detection, missing a fraudulent transaction can be far more costly than flagging a legitimate one. SVM can also handle imbalances with weighted classes, but LR's interpretability and simplicity make it easier to deploy and explain to stakeholders. Decision Trees can work if paired with oversampling or undersampling techniques, but they can still be unstable. KNN struggles in imbalanced settings because nearest neighbors are likely to be from the majority class. Overall, LR provides a strong, interpretable, and flexible solution for such cases.

3) Small Dataset with Many Features (medical/genetic)
When working with small datasets that have a large number of features—such as medical imaging with hundreds of measurements per patient or genetic data with thousands of gene expression values—SVM with a linear kernel often performs best. These datasets pose a risk of overfitting because there are far more predictors than samples, and many features may be irrelevant or noisy. SVM mitigates this by focusing only on the most important points (support vectors) and maximizing the margin. Regularization via the parameter C helps control model complexity, balancing the trade-off between bias and variance. Logistic Regression with L1 regularization is also a strong competitor in such settings, as it can shrink irrelevant coefficients to zero, effectively performing feature selection. Decision Trees tend to overfit small datasets by creating

overly specific splits, and KNN struggles because the few available points make it hard to find meaningful neighbors. In medical contexts, the robustness and generalization ability of SVM make it a preferred choice, especially when diagnostic accuracy is critical.

4) Non-linear Data Separation (spirals, circles)
Some datasets have inherently non-linear decision boundaries, such as circular patterns in clustering or spiral-shaped distributions in synthetic datasets. In such cases, SVM with a Radial Basis Function (RBF) kernel is ideal. The RBF kernel maps the data into a higher-dimensional space, allowing the model to find a separating hyperplane that is not possible in the original space. For example, in an image classification problem where objects are defined by complex shapes, an RBF SVM can capture subtle, non-linear relationships. Decision Trees can also capture non-linearity but tend to create jagged decision boundaries that may overfit to noise. KNN can work in such scenarios, but its performance depends heavily on choosing an appropriate k and handling feature scaling. Logistic Regression struggles without manual feature engineering to introduce non-linear transformations. SVM with RBF provides a smooth, well-regularized boundary that balances accuracy and generalization, making it a reliable choice for non-linear problems across domains such as image recognition, bioinformatics, and pattern detection.

5) Dataset with Noise / Many Irrelevant Features
When datasets contain many irrelevant features or significant noise, Logistic Regression with L1 regularization (LASSO) is particularly effective. L1 regularization penalizes the absolute size of coefficients, forcing some of them to become exactly zero, effectively removing irrelevant features from the model. This makes the model simpler, more interpretable, and less prone to overfitting to noise. For instance, in marketing analytics with hundreds of customer attributes, many of which may be unimportant, L1-regularized LR can focus only on the variables that truly drive purchase decisions. SVM can also perform well with noise but does not inherently remove irrelevant features, often requiring a separate feature selection step. Decision Trees may overfit to noise, creating unnecessary splits, and KNN can be misled by irrelevant features because distances become distorted. L1-regularized LR not only handles noise gracefully but also outputs probabilities, which is useful for risk assessment and decision-making. This combination of feature selection and interpretability makes it a practical choice in many real-world scenarios.

Prepared by,
Utkarsh Anand