UNIT–4 : Model Fit in R (PYQ Solution)

PART–A Answers (≈70 Words Each)

Q1. Differentiate between linear regression and logistic regression. (Expected)

Linear regression	Logistic regression
Predicts a continuous value	Predicts probability of a class (classification)
$y = a + bx$	$p = 1/(1 + e^{-(a+bx)})$, the sigmoid
Least squares, measured by RMSE	Maximum likelihood, measured by accuracy and AUC
R: lm()	R: glm(family = binomial)

Q2. What is a random forest? (Expected)

A random forest is an ensemble of many decision trees. Each tree is trained on a bootstrap sample of the data and, at every split, considers only a random subset of the features. The final prediction is the majority vote (classification) or the average (regression) of all trees. This reduces the overfitting of a single tree and gives feature importance. In R: randomForest(y ~ ., data, ntree = 200).

Q3. What are support vectors and the kernel trick in SVM? (Expected)

Support vectors are the training points closest to the separating hyperplane; they lie on the margin and alone decide its position, so the other points can be removed without changing the model.

The kernel trick lets an SVM separate non-linear data by computing similarity in a higher-dimensional space through a kernel function (linear, polynomial, RBF) without explicitly transforming the data.

PART–B Answers (≈250 Words Each)

Q1. Fit a simple linear regression line to the given data and predict a value. (Expected)

Data: x = 1, 2, 3, 4, 5 and y = 2, 4, 5, 4, 5. Find the line and predict y at x = 6.

Formulas: $b = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sum(x-\bar{x})^2}$, $a = \bar{y} - b\bar{x}$

$\bar{x} = 15/5 = 3, \bar{y} = 20/5 = 4$

x	y	x - 3	y - 4	product	(x-3)^2
1	2	-2	-2	4	4
2	4	-1	0	0	1
3	5	0	1	0	0
4	4	1	0	0	1
5	5	2	1	2	4
Sum				6	10

$b = 6/10 = 0.6, a = 4 - 0.6(3) = 2.2$

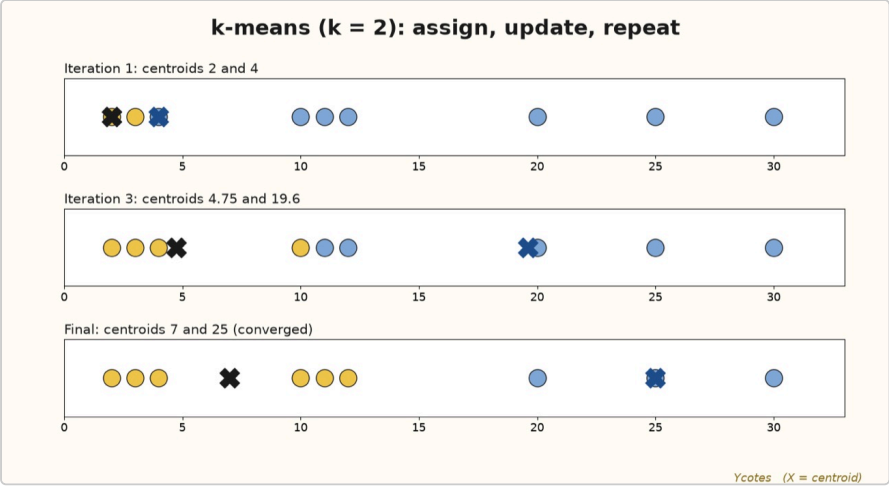
Line: $\hat{y} = 2.2 + 0.6x$. **Prediction:** $\hat{y}(6) = 2.2 + 0.6(6) = 5.8$.

Check: residuals are -0.8, 0.6, 1, -0.6, -0.2 with squares summing to 2.4, so MSE $= 0.48$ and $R^2 = 1 - 2.4/6 = 0.6$.

```
fit <- lm(y ~ x, data = data.frame(x = 1:5, y = c(2,4,5,4,5)))
coef(fit)                              # 2.2  0.6
predict(fit, data.frame(x = 6))        # 5.8
```

Q2. Cluster the given 1-D data into two clusters using k-means.

(Expected)



Data: 2, 4, 10, 12, 3, 20, 30, 11, 25, k = 2, initial centroids $c_1 = 2$, $c_2 = 4$.

Steps: assign each point to the nearest centroid, recompute centroids as means, repeat until they do not change.

Iter	Centroids	Cluster 1	Cluster 2	New centroids
1	2, 4	2, 3	4, 10, 12, 11, 20, 30, 25	2.5, 16
2	2.5, 16	2, 3, 4	10, 12, 11, 20, 30, 25	3, 18
3	3, 18	2, 3, 4, 10	12, 11, 20, 30, 25	4.75, 19.6
4	4.75, 19.6	2, 3, 4, 10, 11, 12	20, 25, 30	7, 25
5	7, 25	2, 3, 4, 10, 11, 12	20, 25, 30	7, 25

Iteration 5 gives the same centroids, so the algorithm stops.

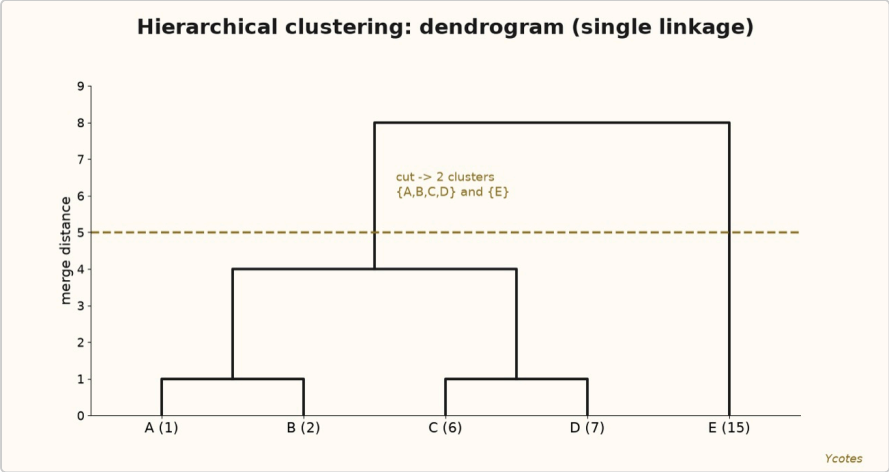
Final clusters: $C_1 = \{2, 3, 4, 10, 11, 12\}$ with centroid 7 and $C_2 = \{20, 25, 30\}$ with centroid 25.

```
x <- c(2, 4, 10, 12, 3, 20, 30, 11, 25)
km <- kmeans(x, centers = c(2, 4))
km[["cluster"]]; km[["centers"]]
```

Choose k with the elbow method and scale the features first.

Q3. Explain hierarchical clustering with a numerical example (single

linkage). (Expected)



Hierarchical (agglomerative) clustering starts with every point as its own cluster and repeatedly merges the two closest clusters until one remains. The merge history is drawn as a **dendrogram**. In **single linkage** the distance between two clusters is the smallest distance between any two of their points.

Data: A = 1, B = 2, C = 6, D = 7, E = 15 (distance = absolute difference).

Step	Clusters merged	Distance
1	A + B	1
2	C + D	1
3	{A,B} + {C,D}	4 (closest pair B = 2 and C = 6)
4	{A,B,C,D} + E	8 (closest pair D = 7 and E = 15)

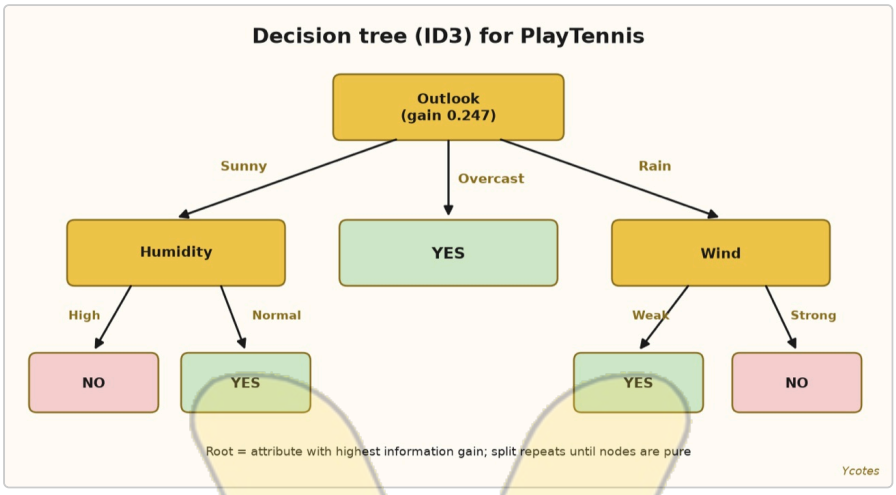
Cutting the dendrogram at height 5 gives two clusters, {A, B, C, D} and {E}, and E is clearly an outlier.

```
hc <- hclust(dist(c(1, 2, 6, 7, 15)), method = "single")
plot(hc)
cutree(hc, k = 2)
```

Comparison with k-means: k is not needed in advance and the dendrogram shows the structure, but it is slower (about $O(n^2)$ memory) and merges cannot be undone.

PART–C Answers (≈500 Words Each)

Q1. Explain the decision tree algorithm. Build the tree for the PlayTennis data using information gain. (Expected)



A **decision tree** classifies by asking a series of questions on attributes. The **ID3** algorithm picks the attribute with the highest **information gain** at each node and repeats on each branch until the node is pure or no attributes remain.

$$H(S) = -\sum p_i \log_2 p_i \qquad Gain(S, A) = H(S) - \sum_v \frac{|S_v|}{|S|} H(S_v)$$

Data: 14 days with attributes Outlook, Temperature, Humidity, Wind and class Play (9 Yes, 5 No).

Outlook	Temp	Humidity	Wind	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Entropy of the whole set: $H(S) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$

Gain of each attribute:

- **Outlook:** Sunny (2 Yes, 3 No) $H = 0.971$; Overcast (4, 0) $H = 0$; Rain (3, 2) $H = 0.971$. $Gain = 0.940 - \left(\frac{5}{14}0.971 + 0 + \frac{5}{14}0.971\right) = 0.940 - 0.694 = \mathbf{0.247}$
- **Humidity:** High (3, 4) $H = 0.985$; Normal (6, 1) $H = 0.592$. $Gain = 0.940 - 0.788 = 0.152$
- **Wind:** Weak (6, 2) $H = 0.811$; Strong (3, 3) $H = 1$. $Gain = 0.940 - 0.892 = 0.048$
- **Temperature:** Hot (2, 2) $H = 1$; Mild (4, 2) $H = 0.918$; Cool (3, 1) $H = 0.811$. $Gain = 0.029$

Outlook has the highest gain, so it is the root.

Sub-trees:

- **Overcast:** all 4 examples are Yes, so the leaf is **Yes**.

- **Sunny (5 examples):** Humidity has gain 0.971 (the highest). High gives No (3 of 3), Normal gives Yes (2 of 2).
- **Rain (5 examples):** Wind has gain 0.971. Weak gives Yes (3 of 3), Strong gives No (2 of 2).

Rules: IF Outlook = Overcast THEN Yes; IF Sunny and Humidity = Normal THEN Yes; IF Sunny and High THEN No; IF Rain and Weak THEN Yes; IF Rain and Strong THEN No.

Gini index (used by CART): $Gini = 1 - \sum p_i^2 = 1 - (9/14)^2 - (5/14)^2 = 0.459$.

```
library(rpart)
fit <- rpart(Play ~ ., data = df, method = "class")
predict(fit, newdata, type = "class")
```

Overfitting is controlled by pruning or by limiting the depth.

Q2. Explain the Naive Bayes classifier. Classify a new tuple using the PlayTennis data. Write R code. (Expected)

Naive Bayes is a probabilistic classifier based on **Bayes theorem**:

$$P(C \mid X) = \frac{P(X \mid C)\, P(C)}{P(X)}$$

The "naive" assumption is that attributes are **conditionally independent** given the class, so $P(X \mid C) = \prod_i P(x_i \mid C)$. The class with the highest posterior is predicted; $P(X)$ is the same for all classes and can be ignored.

Problem: classify X = (Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong) with the 14-row PlayTennis data (9 Yes, 5 No).

Priors: $P(Yes) = 9/14 = 0.643$, $P(No) = 5/14 = 0.357$

Conditional probabilities:

Attribute value	P(. given Yes)	P(. given No)
Sunny	2/9	3/5
Cool	3/9	1/5
High	3/9	4/5
Strong	3/9	3/5

For Yes: $\frac{9}{14} \times \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} = 0.0053$
For No: $\frac{5}{14} \times \frac{3}{5} \times \frac{1}{5} \times \frac{4}{5} \times \frac{3}{5} = 0.0206$

Normalising: $P(No \mid X) = \frac{0.0206}{0.0206+0.0053} = 0.80$ and $P(Yes \mid X) = 0.20$.

Since 0.0206 > 0.0053, the prediction is **Play = No**.

```
library(e1071)
nb   <- naiveBayes(Play ~ ., data = train)
new  <- data.frame(Outlook = "Sunny", Temp = "Cool", Humidity = "High", Wind = "Strong")
predict(nb, new)                        # No
predict(nb, new, type = "raw")          # probabilities
```

Zero-frequency problem: if a value never occurs with a class, the product becomes 0. It is solved by **Laplace smoothing** (add 1 to every count; in e1071 laplace = 1).

Advantages: fast, works with little data, good for text and spam. **Limitation:** the independence assumption is rarely true.

Other models in this unit in short: a **random forest** averages many trees to reduce overfitting; an **SVM** finds the maximum-margin hyperplane and uses kernels for non-linear data.
